
Svenskt Vatten

UTVECKLING

Rapport
Nr 2025-5

Maskininlärning som metod för minskat läckage på Örbyfältet

Praktisk implementering av nya tekniker inom dricksvattenproduktion

Ingemar Clementson
Scott Charlesworth

Svenskt Vatten

UTVECKLING

Svenskt Vatten Utveckling (SVU) är kommunernas eget FoU-program om kommunal VA-teknik. Programmet finansieras i sin helhet av kommunerna. Programmet lägger tonvikten på tillämpad forskning och utveckling inom det kommunala VA-området.

Författarna är ensamt ansvariga för rapportens innehåll, varför detta ej kan åberopas såsom representerande Svenskt Vattens ståndpunkt.

Svenskt Vatten Utveckling

Svenskt Vatten AB

POSTADRESS BOX 14057, 167 14 Bromma

BESÖKSADRESS Gustavslundsvägen 12, 167 51 Bromma

TELEFON 08-506 002 00

E-MAIL svensktvatten@svensktvatten.se

www.svensktvatten.se

RAPPORTENS TITEL	Maskininlärning som metod för minskat läckage på Örbyfältet. Praktisk implementering av nya tekniker inom dricksvattenproduktion
TITLE OF THE REPORT	Machine learning as a method of reducing leakage at Örbyfältet. Implementations of new technologies within water production
FÖRFATTARE	Ingemar Clementson, NSVA, Scott Charlesworth, Collaborate Water
RAPPORTNUMMER	2025-5
ANTAL SIDOR	111
SAMMANDRAG	Projektet har undersökt möjligheten att använda maskininlärning för att minska läckaget från en anläggning för dricksvattenproduktion i Skåne: Örbyfältet och Örbyverket. Rapporten ger en bred bild av de olika delarna i ett maskininlärningsprojekt, förklarar vad som är viktigt att beakta och vilka metoder och verktyg som finns tillgängliga. Under projektets gång har mer än 300 maskininlärningsmodeller tränats. Resultatet visar att maskininlärning, främst tillämpat på vattenförbrukningsprognoser, kan vara ett användbart verktyg för att minska vattenförluster.
SUMMARY	This project explores the possibility of using machine learning to reduce leakage from a drinking water production facility. The report includes a comprehensive description, definitions, and methodology for applying machine learning. The results indicate that machine learning, primarily applied to water consumption forecasting, can be a useful tool for reducing water losses.
SÖKORD	Maskininlärning, artificiell intelligens, prognos för vattenförbrukning, vattenförbrukning, vattenproduktion
KEYWORDS	Machine Learning, artificial intelligence, forecasting, water consumption, water production
MÅLGRUPPER	Dricksvattenspecialister, AI-entusiaster, strateger, bolagsutvecklare, beslutsfattare, driftpersonal
RAPPORT	Finns att hämta hem som pdf från Vattenbokhandeln. https://vattenbokhandeln.svensktvatten.se/
UTGIVNINGÅR	2025
UTGIVARE	©Svenskt Vatten AB
REFERENS	Clementson I. och Charlesworth S. (2025). <i>Maskininlärning som metod för minskat läckage på Örbyfältet. Praktisk implementering av nya tekniker inom dricksvattenproduktion</i> . SVU-rapport 2025-5. Stockholm: Svenskt Vatten.

Om projektet

PROJEKTNUMMER	22-115
PROJEKTETS NAMN	Maskininlärning som metod för minskat läckage på Örbyfältet
PROJEKTETS FINANSIERING	Svenskt Vatten Utveckling, Sweden Water Research

Förord

De senaste åren har mängden tillämpningar inom artificiell intelligens och maskininlärning formligen exploderat. Trots möjligheterna med dessa tekniker är antalet tillämpade implementeringar inom svensk vattensektor förhållandevis få. Varför är det så och vad är det som är så svårt med att gå från utforskande arbete till tillämpningar som kan ge oss nytta på daglig basis?

I denna rapport vill vi ge en bred bild av de olika delarna inom ett maskininlärningsprojekt, förklara vad som är viktigt att beakta och vilka metoder och verktyg som finns tillgängliga. Vi betonar också vikten av att se på maskininlärning som ett ämnesövergripande område där kunskap om det studerade systemet och önskad nytta är central ihop med kunskap om datavetenskap och teknisk infrastruktur.

Örbyfältet och Örbyverket i Skåne utgör tillsammans ett tekniskt komplicerat system som bidrar till säkrad dricksvattenleverans inom Sydvattens distributionsområde. Systemet är svårt att beskriva med traditionella beräkningsmetoder på grund av tidsförskjutningar i systemet, komplexa magasinsförhållanden, mänsklig faktor och variation i uttag. I detta projekt har vi angripit dessa utmaningar genom att tillämpa maskininlärning. Under projektets gång har mer än 300 maskininlärningsmodeller tränats.

Att tillämpa maskininlärning på insamlade vattendata är fantastiskt roligt. Trots att denna rapport beskriver komplexiteten i stora maskininlärningsprojekt är uppmaningen från oss författare denna: sätt igång att testa! Den potentiella nyttan för vattensektorn är enorm. Och se till att lära, utveckla och framför allt samarbeta så mycket det bara går.

En referensgrupp bestående av Tommy Giertz vid Stockholm Vatten och Avfall, Philip McCleaf vid Uppsala Vatten, Mats Henriksson vid Miljö och Vatten i Örnköldsvik AB och Johan Spännare vid DHI Sverige AB har bidragit med värdefulla synpunkter till projektets genomförande. Även Patrik Lissel vid WSP har bidragit med värdefulla synpunkter om systemdynamik och hydraulisk modellering.

Ett tack ska också riktas till kollegor på NSVA och Sweden Water Research för att ni har stått ut med oändliga frågor om kapacitet, historik och förståelse. Särskilt vill vi nämna Sven Bengtsson och Lars Stefansson som har haft extra stort tålamod och bidragit med mycket kunskap. Ett tack riktas också till Salar Haghighatafshar vid Watrix för synpunkter och kommentarer gällande maskininlärning.

Vi hoppas att denna rapport ska kunna ge inspiration och fungera vägledande för fler implementeringar av maskininlärning.

Helsingborg i november 2024

Ingemar Clementson, NSVA, och Scott Charlesworth, Collaborate Water

Innehåll

Förord	2
Sammanfattning	4
Summary	5
1 Bakgrund	6
1.1 Syfte och mål med projektet	6
1.2 Läsanvisning	7
1.3 Förkortningar och begrepp	8
2 Översiktlig systembeskrivning av Örbyfältet	9
2.1 Arbetsflöden	11
2.2 Om geohydrologi i akviferer	12
2.3 Tillgängliga data	13
2.4 Om vattenförluster	15
2.5 Om beställning av vatten	17
3 Maskininlärning i ett bredare sammanhang	19
3.1 Ramverk och definitioner	19
3.2 Maskininlärning	22
3.3 Vikten av datas lämplighet, förbehandling och rensning	33
3.4 Att navigera i komplexiteten	35
4 Maskininlärning inom vattensektorn – ett sammandrag	37
4.1 Vattenkvalitetsparametrar	38
4.2 Prognos av förbrukning	38
4.3 Vattennivå i akvifer	39
5 Maskininlärning – arbetsflöden och metoder	40
5.1 Inhämtande av domänkunskap/förståelse för problemet	40
5.2 Valet av algoritm	40
5.3 Beräkning och arbetsmetodik	42
6 Använd verktygslåda	47
6.1 Översikt	47
6.2 Utforskande dataanalys - EDA	47
6.3 Förbehandling av data	48
6.4 Beskrivning av utvärderade algoritmer	49
6.5 Prestandamått	55
7 Resultat och diskussion	58
7.1 Resultat av utforskande dataanalys	58
7.2 Dygnsvärden – prognosmodell	67
7.3 Om möjliga tillämpningar av framtaget material	88
7.4 Modellering av grundvattennivåer	89
8 Sammanfattning och råd	92
8.1 Prognosmodell för dygnsvärden och timvärden	92
8.2 Vattenförbrukning och vattenläckage	92
8.3 Nivåer i fältet	93
8.4 Om ML-projekt	93
Referenser	95
Bilaga A Schematiska beskrivningar av neurala nätverk	98
Bilaga B Matematiska sammanfattningar av utvärderade algoritmer	99
Bilaga C Utvärderingstabeller för tränade algoritmer	108

Sammanfattning

Projektet har undersökt möjligheten att använda maskininlärning för att minska läckaget från en anläggning för dricksvattenproduktion i Skåne: Örbyfältet och Örbyverket. Rapporten ger en bred bild av de olika delarna i ett maskininlärningsprojekt, förklarar vad som är viktigt att beakta och vilka metoder och verktyg som finns tillgängliga. Under projektets gång har mer än 300 maskininlärningsmodeller tränats. Resultatet visar att maskininlärning, främst tillämpat på vattenförbrukningsprognoser, kan vara ett användbart verktyg för att minska vattenförluster.

De senaste åren har mängden tillämpningar inom artificiell intelligens och maskininlärning ökat kraftigt. Trots möjligheterna med teknikerna är tillämpningarna inom VA-branschen relativt få. För att få till ett skifte från reaktivt till proaktivt arbete är maskininlärning en viktig komponent. Författarna betonar att det gäller att se på maskininlärning som ett ämnesövergripande område där kunskap om det studerade systemet och den önskade nyttan är central tillsammans med kunskap om datavetenskap och teknisk infrastruktur. Projektet har gjort en ansats att beskriva den uppsjö av begrepp, metoder och tekniker som är viktiga att känna till för att kunna tillämpa och samarbeta inom maskininlärning.

Örbyfältet och Örbyverket utgör tillsammans ett tekniskt komplicerat system som bidrar till säkrad dricksvattenleverans inom Sydsvattens och NSVA:s distributionsområde. En av de största utmaningarna med systemet är att fältets akvifer lider av läckage med stora vattenförluster som följd. Systemet är svårt att beskriva med traditionella metoder på grund av bland annat svårigheter med att prognosticera vattenförbrukningen och med att mäta i en akvifer. Projektet angrep utmaningarna genom att tillämpa maskininlärning.

Resultaten visar att det framgångsrikt gick att förutse vattenförbrukningen för de kommande 28 dagarna på dygnsnivå och för 12 timmar på timnivå, inom 2–3 % felmarginal. Flera variabler utvärderades för att undersöka vilka av dem som kan förbättra en maskininlärningsprognos. Slutsatsen blev att olika algoritmer svarar olika på skilda variabler. Även om det blev en viss förbättring när vissa variabler inkluderades kunde ett tillräckligt gott resultat uppnås genom att använda endast historiska data för vattenförbrukning. På det sättet blir det möjligt med enklare implementering av det framtagna resultatet. Genom analys av förluster och prognosmodeller skapades också förslag på hur framtaget material kan presenteras som ett verktyg för driftoperatörer. Projektet undersökte också nivåmätningar och pumpdata men drog slutsatsen att det inte fanns tillräckligt med data för att gå vidare med de undersökningarna.

Genom att ha tillgång till goda modeller för prognos av vattenförbrukning och analys av förluster finns det möjligheter att använda maskininlärning för att minska läckaget på Örbyfältet. Problemet med att balansera intag och uttag ur ett grundvattenmagasin är inte unikt för NSVA, inte heller behovet av korrekta vattenförbrukningsprognoser. Resultatet av projektet kan ha stor nytta för kommunala VA-organisationer och andra som tillämpar maskininlärning i vattensektorn. Förhoppningen är att rapporten ska kunna fungera som ett referensverk som ska stärka VA-branschens förmåga att tillämpa maskininlärning.

Summary

This project explores the possibility of using machine learning to reduce leakage from a drinking water production facility. The report includes a comprehensive description, definitions, and methodology for applying machine learning. The results indicate that machine learning, primarily applied to water consumption forecasting, can be a useful tool for reducing water losses.

The Örby Field and Örby plant together form a technically complex system that contributes to secure drinking water supply within Sydvatten's and NSVA's distribution area. One of the biggest challenges is that the need for a buffer leads to leakage and significant water losses. The system is difficult to describe using traditional methods due to several factors: consumption forecasting, geohydrology, the need for continuous adaptation, and the difficulty of measuring in an aquifer.

The need to apply new technologies to existing problems is substantial, and the potential benefit for Swedish water utilities is immense. The challenge of balancing intake and extraction from a groundwater reservoir is not unique to NSVA, nor is the need for accurate water consumption forecasts. This project aimed to evaluate whether machine learning is a suitable method for assessing operational strategies and ultimately reducing leakage from the Örby Field.

The results of this project could greatly benefit Swedish Water utilities and others applying machine learning in the water sector. Machine learning is a key component for shifting from reactive to proactive work in the water sector. This report highlights the importance of interdisciplinary work, infrastructure, and competence, making it valuable for decision-makers and specialists within water utility organizations.

In this project, an effort was made to describe the array of concepts, methods, and techniques that are important to understand for both applying and collaborating in machine learning. Our hope is that this will serve as a reference work to strengthen the Swedish water sector's ability to apply machine learning. To achieve significant success and business value, we also emphasize the importance of viewing this type of project as interdisciplinary, where domain knowledge is central alongside knowledge of data science and technical infrastructure.

The results show that we can successfully predict water consumption for the next 28 days on a daily level and for 12 hours on an hourly level. Several variables were evaluated to determine which could improve a machine learning forecast. Different algorithms responded differently to various variables. Although some improvement was seen when certain variables were included, a sufficiently good result could be achieved by using historical water consumption data alone. This enables easier implementation of the developed solution.

Given the overarching goal, only a minor effort was made to forecast consumption based on hourly values. Further evaluation of this would be of interest to the industry, particularly the possibility of extending the forecast, which is considered valuable.

By having access to good forecast models and loss analysis, there are opportunities to use machine learning to reduce leakage in the Örby Field.

1 Bakgrund

De senaste åren har antalet tillämpningar inom maskininlärning och artificiell intelligens ökat markant vilket inte minst märks på närvaron av termen artificiell intelligens. Trots möjligheterna med dessa tekniker är antalet implementeringar inom svensk vattensektor förhållandevis få. En av anledningarna till detta är att dessa tekniker ofta ses som en ”svart låda” vilket försvårar både tilliten till resultat och begränsar möjligheterna att bedriva utveckling inom maskininlärning tillämpat på vattensektorn.

I denna rapport ges en bred bild av de olika delarna inom ett tillämpat maskininlärningsprojekt, förklaring av vad som är viktigt att beakta och vilka metoder och verktyg som finns tillgängliga.

1.1 Syfte och mål med projektet

Det övergripande målet för projektet var att undersöka om maskininlärning kan användas för att minska vattenförlusterna på Örbyfältet i nordvästra Skåne. Att minska vattenförlusterna är viktigt både ur miljö- och hållbarhetsperspektiv samt för att minska kostnaderna som vattenförlusterna medför.

Örbyfältet och Örbyverket i Skåne utgör tillsammans ett tekniskt komplicerat system som bidrar till säkrad dricksvattenleverans inom Sydvattens distributionsområde. Systemet är svårt att beskriva med traditionella beräkningsmetoder då dessa ofta har väldigt långa beräkningstider samt att systemet kontinuerligt förändras utifrån driftsbeslut (flöden och nivåer är helt enkelt inte statiska).

Att optimera driftstrategier för Örbyfältet är en stor utmaning, främst på grund av systemets komplexitet och de stora datamängder som måste samordnas för att balansera intag, uttag, pumpning från ett 30-tal brunnar samt uppmätta nivåer i fältet. Därtill kommer det faktum att det inte finns någon exakt kännedom om var läckage uppstår eller vilken volym det omfattar vid vilket tillfälle. Genom att använda maskininlärning finns möjlighet att tillämpa extremt snabba beräkningsmodeller och möjligen identifiera samband som är svåra att hitta med traditionella och manuella beräkningsmetoder.

Utöver detta syftade projektet till att öka kunskapsnivån kring maskininlärning inom vattensektorn samt att visa på hur man kan arbeta med maskininlärning ur både ett utvecklings- och driftperspektiv.

Att undersöka om maskininlärning kan användas för att minska vattenförluster innefattar två huvuddelar:

- Förbrukningsprognoser för dricksvatten, 30 dagar fram i tiden
- Undersökning av om maskininlärning kan användas för att optimera driftstrategier för pumpning och nivåer i fältet.

Baserat på anläggningens driftförutsättningar, inklusive uppehållstid och behovet av buffertkapacitet, kan en noggrann vattenförbrukningsprognos möjliggöra en mer exakt beställning av vatten (från den regionala vattendistributören, Sydvatten) i förhållande till faktisk förbrukning, vilket kan bidra till minskat läckage. Att skapa och utvärdera en korrekt prognosmodell för dricksvattenförbrukning har dessutom många andra användningsområden och kan vara till stor nytta för fler delar av vattendistributionsnätet. En förbrukningsprognos kan exempelvis användas för att upptäcka avvikelser och identifiera läckor i dricksvattendistributionsnätet (genom att jämföra förväntad förbrukning med den faktiska), samt bidra till en mer säker dricksvattenproduktion och -distribution. I arbetet ingår att undersöka vilka parametrar som bör ingå för bästa möjliga resultat,

vilka typer av algoritmer som är mest lämpliga att använda, samt hur data ska behandlas för att uppnå optimala resultat.

1.2 Läsanvisning

Nedan följer en kort beskrivning av innehållet i de olika kapitlen. Detta för att läsaren ska kunna orientera sig och snabbt kunna ta till sig enskilda delar av innehållet i denna rapport.

1. Bakgrund

Här beskrivs syfte och mål med projektet, samt läsanvisning och förklaring av använda förkortningar och begrepp.

Nyckelord: *bakgrund, orientering, begrepp*

2. Översiktlig systembeskrivning av Örbyfältet

Här beskrivs varför detta projekt är viktigt för Örbyfältet samt hur systemet fungerar, vilka beslut det är som tas i driften av fältet och hur vi har definierat vattenförluster för anläggningen.

Nyckelord: *syfte, systembeskrivning, definition av vattenförluster*

3. Maskininlärning i ett bredare sammanhang

En introduktion till maskininlärning som innefattar definitioner, skillnaden mellan olika typer av algoritmer, dess tillämpningsområden, historik samt en kortare diskussion om hur indata kan förberedas för maskininlärningstillämpningar.

Nyckelord: *orientering, definitioner, val av algoritm*

4. Maskininlärning inom vattensektorn – ett sammandrag

I detta kapitel redovisas exempel på vetenskapliga rapporter som fungerat som en mindre litteraturstudie i detta projekt.

Nyckelord: *litteraturstudie, omvärldsbevakning, existerande tillämpningar*

5. Maskininlärning – arbetsflöden och metoder

Här redovisas en viktig, och ofta förbisedd, del av maskininlärningsprojekt som rör vanligen använda ramverk, projektmetodik, och mer praktiska saker som beräkningskraft och programmeringsspråk.

Nyckelord: *arbetsmetodik, beräkningsförutsättningar*

6. Använd verktygslåda

I detta kapitel redovisas vilka verktyg som tillämpats i projektet gällande förbehandling av data, vilka typer av algoritmer som utvärderats i projektet samt prestandamått.

Nyckelord: *verktyg, beskrivning av algoritmer, resultatutvärdering*

7. Resultat och diskussion

Här redovisas resultat från den utforskande dataanalysen som är grunden för vidare arbete. Därefter resultat från de olika utvärderade algoritmerna samt en beskrivning av möjliga tillämpningsområden för resultatet.

Nyckelord: *analys, vattenförbrukningsprognoser, resultat*

8. Sammanfattning och råd

I detta kapitel redovisas en kortare sammanfattning av resultatet från projektet samt en kortare diskussion om maskininlärningsprojekt i vattenbranschen.

Nyckelord: *sammanfattning, reflektion*

1.3 Förkortningar och begrepp

I rapporten överlappas olika kompetensområden såsom hydraulik, vattenproduktion och datavetenskap. Framför allt det sista ämnesområdet använder en stor mängd engelska termer även i svenskt språkbruk, men det är inte konsekvent och referenslitteratur har många olika översättningar. Att översätta alla begrepp till svenska, och dessutom skapa svenska förkortningar där det vanligtvis används engelska förkortningar, skulle bara förvirra och försvåra om läsaren önskar söka mer kunskap inom ämnet på annat håll. Dessutom saknas definierade svenska begrepp för många termer. I rapporten anges därför ibland svenska termer, ibland engelska termer och ibland en engelsk term bredvid en svensk. Förhoppningen är att detta ska förenkla läsningen och förbättra läsoplevelsen.

Följande förkortningar har varit nödvändiga då de används, och kombineras, väldigt många gånger i rapporten.

Svensk term	Förkortning	Engelsk term
Maskininlärning	ML	Machine Learning
Artificiell Intelligens	AI	Artificial Intelligence
Utforskande dataanalys	EDA	Exploratory Data Analysis
Neurala nätverk	NN	Neural Network
Medelabsolutprocentfel	MAPE	Mean Average Percentual Error
Medelabsolutfel	MAE	Mean Average Error

Värt att ta med sig är att alltför detaljerade beskrivningar, speciellt innehållande geodata och detaljerade systembeskrivningar, är utelämnade från rapporten med hänsyn till informationssäkerhet.

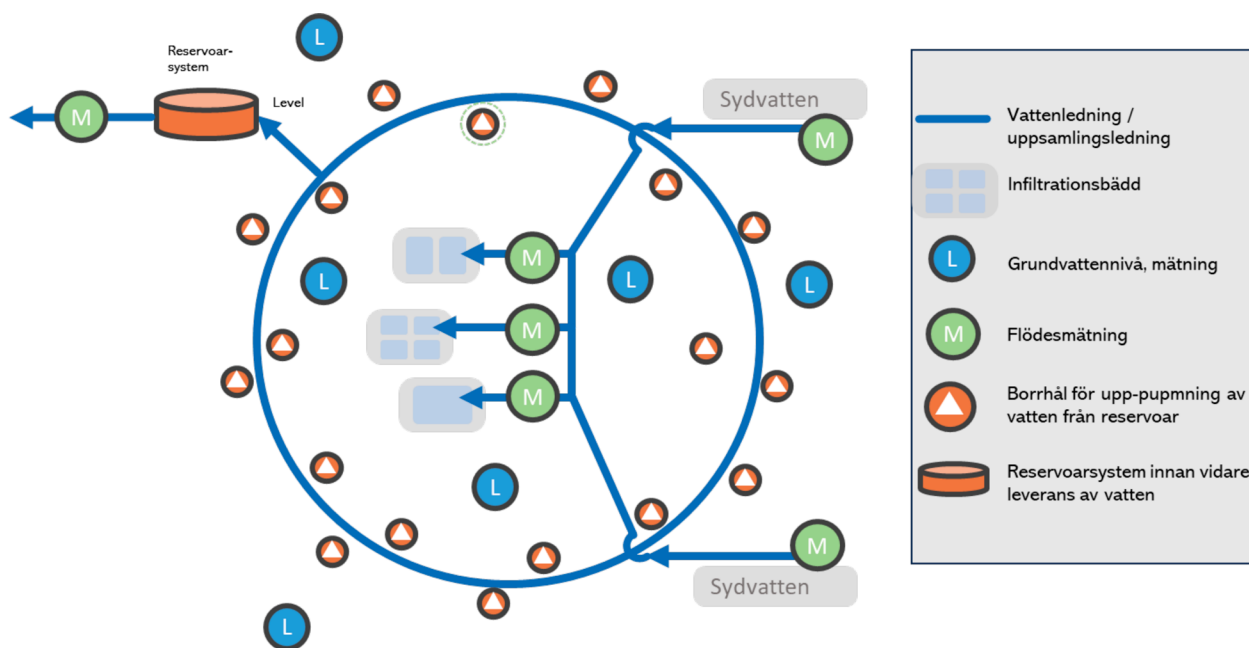
Texter i figurer är översatta till svenska på alla ställen det har gått att göra det. I några av figurerna har det inte varit möjligt eftersom ursprungsbilden inte har varit tillgänglig.

2 Översiktlig systembeskrivning av Örbyfältet

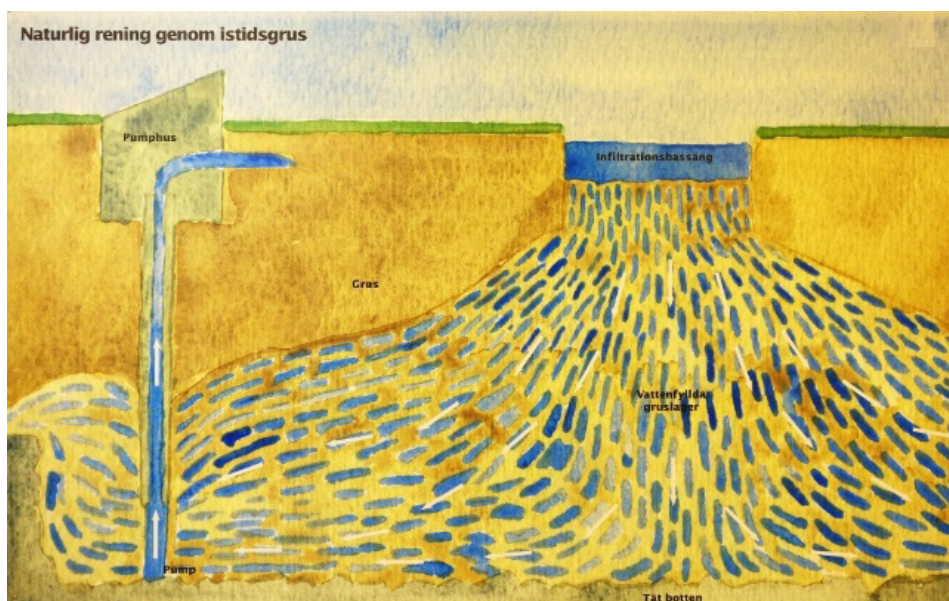
Dricksvattenproduktionen vid Örbyverket består av ett system där inkommande dricksvatten från Ringsjöverket (producerat av Sydvatten) infiltreras ner till en sluten geologisk formation i Örbyfältet. Infiltrationen till akviferen sker via infiltrationsbäddar. Efter cirka 3 veckors uppehållstid uppföras vattnet via ett 30-tal grundvattenbrunnar och pumpas in till Örbyverkets lågreservoar. Högtryckspumpar distribuerar därefter ut vatten på dricksvattnätet via UV-aggregat för avdödande av eventuellt förekommande bakterier. Örbyfältet/Örbyverket förser cirka 200 000 personer och en mängd företag med rent dricksvatten varje dag. Örbyverket ligger söder om Helsingborg i närhet till Öresund. I Figur 2.1 ses en schematisk översikt för systemet.

Figur 2.1

Schematisk översikt över systemkomponenter på Örbyfältet. Inkommande vatten infiltreras ner i akviferen; efter uppehållstiden pumpas vatten upp till ringleden som leder vatten till reservoarsystemet där det därefter pumpas vidare ut på distributionsnätet.



Örbyfältet utgör på så sätt en stor reservoar som innebär att dricksvattendistributionen inte är beroende av en kontinuerlig leverans från Ringsjöverket. Detta utgör en viktig funktion i Sydvattens distributionssystem då fältet fungerar som en buffert. Utöver detta bidrar fältets kapacitet till en säkrad distribution åt Nordvästra Skånes Vatten och Avlopp (NSVA) vid driftproblem/underhåll i Sydvattens distributionsnät. I Figur 2.2 ses en illustration av hur vatten går från infiltrationsbädd ner i akviferen för att slutligen pumpas upp.



Figur 2.2

Schematisk bild på intag och konstgjord infiltration i Örbyfältet.

Tyvärr finns även nackdelar med en naturlig reservoar, och den största är att det finns ett läckage. Det vatten som inte kan fångas upp med pumpar försvinner från fältet som läckage. Detta sker framför allt i fältets västra delar, mot Öresund. Läckaget minskas genom en samlingsbrunn i fältets allra västligaste del men utgör trots detta ca 6–7 % av den mängd vatten som levereras från Sydsvatten. På årsbasis utgör detta 3 000–4 000 m³ per dygn motsvarande den dagliga vattenförbrukningen för cirka 25 000 personer.

Den formation som utgör reservoar för grundvatten är tämligen komplex. Nivåerna i magasinet varierar vilket gör det problematiskt att avgöra exakt när och exakt var läckaget uppstår. Beställning av vatten från Sydsvatten görs manuellt av operatörer som bestämmer omfattning utifrån fyllnadsgraden i reservoaren, erfarenhet om vilka trender det finns gällande förbrukning av vatten samt vilka förändringar vi kan vänta oss framöver baserat på väder och årstid. Tidsförskjutningen mellan intag och uttag (cirka 3 veckor) gör att även det så kallade "beställningsflödet" från Sydsvatten påverkar läckaget, och i dagsläget har den mänskliga faktorn stor betydelse. Hur mycket vatten som beställts kan korrigeras av driftoperatörer beroende på om förbrukningen förväntas öka eller minska.

Trögheten i systemet ökar också komplexiteten gällande intag och uttag av vatten. Om nivåerna i magasinet sjunker för snabbt kan inte intaget ökas i för hög hastighet då detta riskerar att skapa oönskade vattenförflyttningar inom magasinet som riskerar att öka läckaget. Likaså kan ett alltför drastiskt uttag leda till att vattennivåerna i fältet ändras på ett negativt sätt.

Uttaget av vatten från reservoaren görs via grundvattenpumpar som körs i grupper beroende på nivå i lågreservoarerna samt uppmätta nivåer som indikerar fyllnadsgraden i de olika delarna av fältet. Gruppering av pumpar bestäms manuellt och kräver stor erfarenhet för att minska läckage samt för att möta ett varierande konsumtionsbehov. Då nivån skiljer sig i fältet beroende på vilka brunnar som körs så påverkar driften fältets läckage.

Driftoperatörerna arbetar generellt med gruppering av pumpar där olika enskilda pumpar kan lyftas in eller ut beroende på kapacitet och nivåer. Generellt hålls fyllnadsgraden i fältet på runt 50 % under vintertid och 75 % på försommar/sommar, vilket är den högsta nivån som är önskvärd. Den sistnämnda påfyllnaden görs för att buffra upp innan förväntad ökad sommarförbrukning (vilket måste göras i förväg på grund av de långa infiltrationstiderna). Blir fyllnadsgraden för hög ökar läckaget och är den för låg riskeras att för lite vatten finns tillgängligt, speciellt under sommarmånaderna när vattenförbrukningen kan öka tämligen snabbt. Denna balans utgör en stor del av svårigheten i att drifta fältet på ett så effektivt sätt som möjligt.

Under 2021 har nivåmätningen i kontrollpunkterna automatiserats. I stället för att loda manuellt samlas nivåmätningar in automatiserat.

2.1 Arbetsflöden

För att förstå vilka data som ligger till grund för beslut, vilka beslut som tas och hur resultat ska tolkas, genomfördes ett antal intervjuer och workshoppar med driftoperatörer samt experter med lång erfarenhet från Örbyfältet, bland annat inom hydraulisk modellering och geologisk kartering. Exempel på upplägg/intervjumetodik kan ses i Figur 2.3.

Arbetsgången för att beställa vatten inkluderar:

- Kontroll av nuvarande förbrukning (är den trendmässigt på väg upp eller ner?)
- Beräkning av nuvarande fyllnadsgrad i fältet
- Koll av väderprognoser, kort horisont och lång horisont
- Kontroll om det finns planerade driftstopp i leveranskedjan.

När en beställning av vatten är gjord övervakas parametrarna ovan och korrigeringar av beställningen kan göras mellan de vanliga beställningsintervallen.

Driftsbeslut

#	Uppgift	Frekvens	Anteckning*
1	Hur mycket vatten ska beställas	Månadsvis	
	Bestäm mål-nivå för kommande vecka		
2	Utvärdera nuvarande förbrukning mot beställt vatten	Två ggr i veckan	
3	Ändra ordning om vid behov	Mitt i månaden	
4	Utvärdera flöde i utgående huvudledningar	Dagligen	Ses ofta som en summerad beräkning men behöver ibland särskiljas
5	Utvärde % global fyllnadsgrad	Dagligen	
6	Matcha pumpar mot högsta grundvattennivå – rank 1	Dagligen	
7	Kontrollera om enskilda borrhåll har för höga nivåer – rank 2	Dagligen	
8	Kontrollera mot modellerade upptagningsområden – rank 3	Dagligen	
9	Gruppera pumpar i grupp 1-6	Dagligen	Minimiflödeskontroll? Automatisk avstängning?
10	Post –grupperings-övervakning	Sällan	Dubbelkolla
11	Förberedelser inför underhåll		Utveckla
12	Bedömning om användning av brunnar utanför fältet		Utveckla

Figur 2.3

Exempel på resultat från intervjuer/workshoppar med driftoperatörer och experter.

Driftoperatörer Uppgift 2

#	Uppgift	Frekvens	Anteckning*
2	Utvärdera beställningsflöde	Två gånger i veckan	Bedöms också utifrån historic och väderprognos

#	Underuppgift	Anteckning*
2.1	Upprepa steg 1.1 – 1.5	
2.2	Utvärdera väderdata sen senaste beställningstillfälle	Blev vädret som prognosen visade?
2.3	Utvärdera förändringar i långtidsväderprognos	
2.4	Behöver beställning ändras utifrån förändringar i prognosen?	
2.5	Behöver prognosen förändras p.g.a. ökat buffertbehov / förändringar i förbrukning?	

#	Uppgift	Frekvens	Anteckning
11	Förberedelser för underhållsarbete	Ad-hoc	Behöver förtydligas / exemplifieras

#		
11.1	Givet planerat underhåll, vilken påverkan förväntas?	I vilken skala görs bedömningen?
11.1i	Påverkas den totala leveransförmågan?	Behövs det buffras?
11.1ii	Påverkas pump-kapaciteten?	
11.2	Påverkas övervakningsförmåga?	
11.1i	Tidshorisont på problem	
11.1ii	Påverkas infiltrationskapacitet?	Ex. Biofilm eller igensättning i infiltration?
11.1iii	Påverkas magasininskapacitet?	

Uttag av vatten från fältet (via grupperad pumpstyrning) justeras på daglig basis och görs manuellt baserat på behovet av vatten, nivåer i fältet och erfarenhet vilket resulterar i en prioriteringslista för pumpar. Pumpar som ligger nära kanten av fältet, där läckage är en viktig faktor, tenderar att få hög prioritet. När prioriteringslistan är gjord grupperas pumpar med målet att få till ett jämnt stadigt uttag av vatten. Behöver enskilda pumpar i en grupp justeras görs detta efter behov. Ett underlag som ofta nämndes är att resultatet från hydraulisk modellering används. Detta resultat visar upptagningsområde för de olika pumparna, tyvärr redovisas inte denna karta i denna rapport på grund av sekretess.

Värt att nämna är också att pumpar kan stoppas automatiskt om nivån i borrhålet blir för låg samt att det finns dubbelpumpar i vissa borrhål. Vissa av dem kan köras i serie och vissa parallellt. Mer detaljerad information om driften har samlats in och använts i projektet men redovisas inte här på grund av sekretess.

2.2 Om geohydrologi i akviferer

För att skapa förståelse för komplexiteten i en akvifer, och som underlag för tolkning av data, redovisas nedan grundläggande begrepp och processer gällande geohydrologi i en akvifer. Informationen har huvudsakligen inhämtats från *Hydrogeologic Properties of Earth Materials and Principles of Groundwater Flow - The Groundwater Project* (gw-project.org).

Akviferen som undersöks är en halvsluten grusakvifer, med vattenförluster till havet och norra sidan. Akviferer är komplexa system med egenskaper som sträcker sig i tre dimensioner. De viktigaste egenskaperna hos akviferer är:

- Transmissivitet
- Hydraulisk konduktivitet – vattnets förmåga att flöda genom akviferen
- Volym
- Vattenavgivningstal
- Lagringskoefficient/storativitet.

Transmissiviteten hos en akvifer är produkten av den hydrauliska konduktiviteten och tjockleken på den mättade delen av akviferen. Tjockleken hos en öppen akvifer varierar betydligt mer än i en sluten akvifer, där den i stort sett kan vara konstant. På grund av variationen i tjocklek beror transmissiviteten hos en öppen akvifer även på de lokala förutsättningarna.

Hydraulisk konduktivitet (K) är ett mått på hastigheten som visar hur fritt vatten kan passera genom jord eller berg. En hög hydraulisk konduktivitet indikerar hög permeabilitet, vilket innebär att vatten kan flöda lätt. Enheten för hydraulisk konduktivitet är vanligtvis meter per sekund eller meter per dag. Porositeten har stor påverkan på den hydrauliska konduktiviteten hos en akvifer. Hydraulisk konduktivitet är en geoteknisk parameter med ett stort spann på värden. Ett typiskt intervall för denna parameter är från 10^{-2} m/s i genomsläppliga stenar och grus till 10^{-11} m/s i ospruckna leror eller massiv berggrund. Dessutom påverkas hydraulisk konduktivitet av vätskans egenskaper, inklusive temperatur, men temperaturförändringar är ofta försumbara för ett givet system. Hydraulisk konduktivitet kan beskrivas i tre dimensioner: K_x , K_y , K_z . Att analysera hydraulisk konduktivitet i tre dimensioner är viktigt eftersom det hjälper till att förstå hur vattnet rör sig genom akviferen i olika riktningar, vilket är avgörande för att korrekt kunna bedöma vattenflöde och resursutnyttjande. Om den är densamma i alla tre dimensionerna är akviferen isotropisk, vilket betyder att den hydrauliska konduktiviteten är oberoende av riktning. Annars är den anisotropisk, där konduktiviteten varierar beroende på riktning. I en anisotropisk akvifer skiljer sig alltså den hydrauliska konduktiviteten åt beroende på var i akviferen man är.

Vattenavgivningstalet definieras som volymen vatten som frigörs från en öppen akvifer per ytenhet av akviferen och minskning av grundvattennivån. Lagringskoefficienten (storativiteten) hos en öppen akvifer domineras av gravitation och definieras som volymen vatten som frigörs från lagring per ytenhet per minskning i grundvattentryck.

Utöver dessa egenskaper ökar komplexiteten i akviferer ytterligare genom lager av olika material och/eller samma material med olika egenskaper. Dessutom kan lager med tiden röra sig, vilket förändrar egenskaperna över tid.

När vatten pumpas upp från en akvifer bildas en avsänkningsskon. De egenskaper som påverkar vattenuttaget från en akvifer är:

- Brunnens (borrhålets) egenskaper såsom djup och utformning
- Pumpningshastighet
- Avståndet mellan brunnar
- Akviferens egenskaper (beskrivet ovan).

Vid kontinuerlig pumpning av en brunn i en öppen akvifer under steady-state-förhållanden används styrande ekvationer baserade på Darcys lag och Theims ekvation för radiellt flöde mot brunnen. Under stabiliserade förhållanden balanseras sänkningen (d.v.s. förändringen i vattennivå), vilket innebär att inflödet och utflödet av grundvatten är i jämvikt.

Ett stort uttag av vatten, dessutom på många olika ställen, i många olika grupperingar som ständigt förändras, påverkar flera av akviferens parametrar. För att skapa en maskininlärningsmodell som korrekt återger mönster och beteenden som beskrivs ovan behövs tillräcklig mängd data och spatiala indelningar.

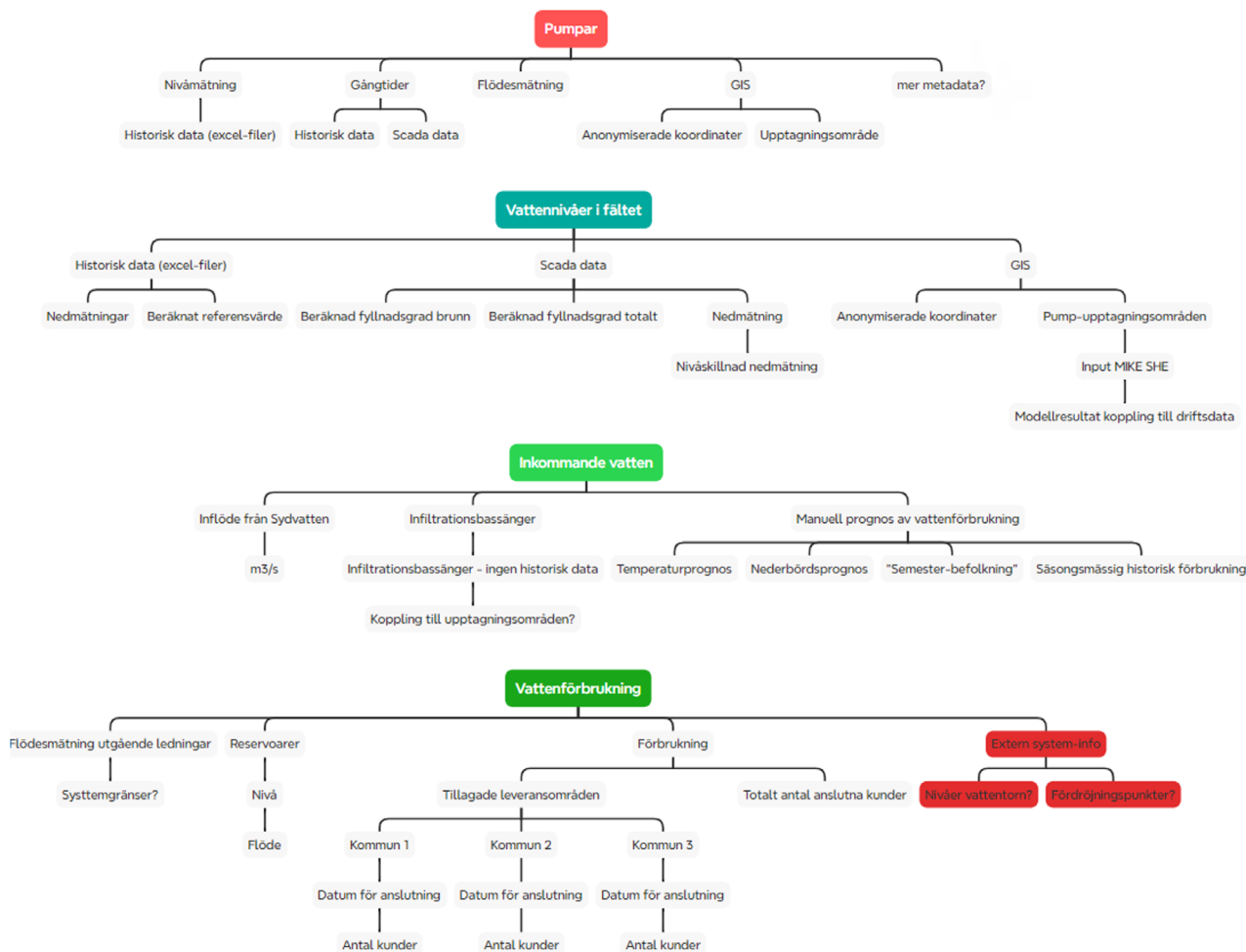
2.3 Tillgängliga data

För att få en översikt gjordes en sammanställning av tillgängliga data för varje systemkomponent vilket även inkluderade en strukturerad uppdelning av data relaterade till varje systemkomponent. Resultatet kan ses i Figur 2.4.

Inledningsvis levererades data med så hög frekvens som ner till minutnivå. Värt att notera är att alla data inte har så hög temporal upplösning. Grundvattennivåmätningar har exempelvis inte nya värden varje minut vilket innebär att föregående värde gäller till dess att ett nytt värde rapporteras in från mätaren. Data levererades som:

- Minutdata
- 15-minutersdata
- Timdata
- Dygnsdata.

Till varje dataset togs statistiska parametrar fram.



Figur 2.4
Översikt över hur tillgängliga data undersöktes.

Även om alla data initialt studerades togs beslutet att fokusera på dygnsbaserade data. Detta då driftoperatörer på fältet var tydliga med att beslut generellt fattas dagligen snarare än flera gånger per dag. Viss modellering har också gjorts på timvärdesdata för att exemplifiera skillnaderna i att arbeta med data med högre upplösning. Ofta är data med högre upplösning mer komplexa och svåra att arbeta med, både matematiskt och beräkningstekniskt, men kan också avslöja mer information och ge bättre insikter.

Under senare år har ny mätutrustning installerats som medger kontinuerlig mätning av grundvattennivåer i fältet. Tidigare har mätning av grundvattennivåer gjorts manuellt via lodning. Installationen av dessa nivåmätare pågick i början av projektet inklusive fortsatt kalibrering och justering av referensvärden. Mätdata skickas från de nya nivåmätarna ett antal gånger per dag vilket bedöms som tillräckligt då nivåerna generellt ändras långsamt.

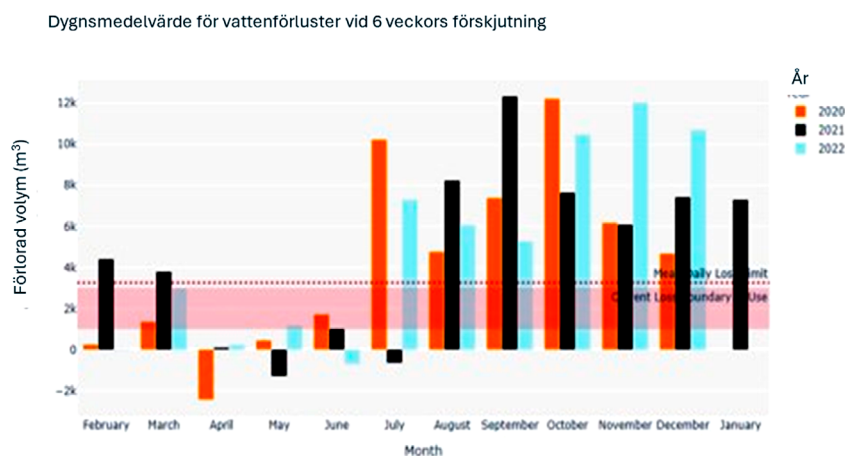
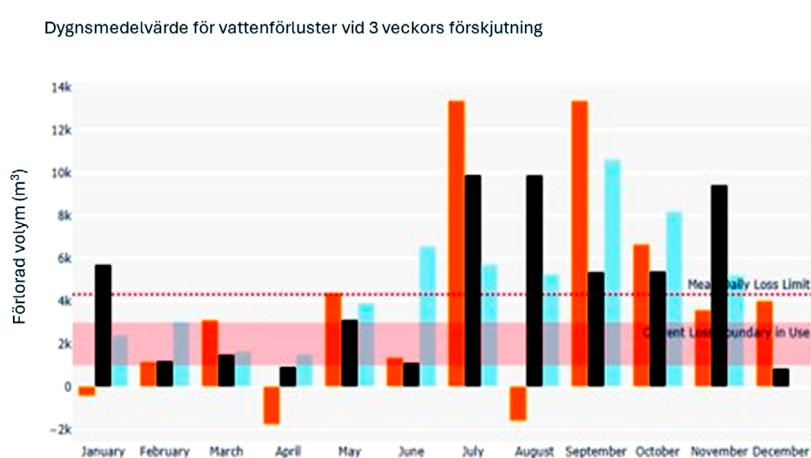
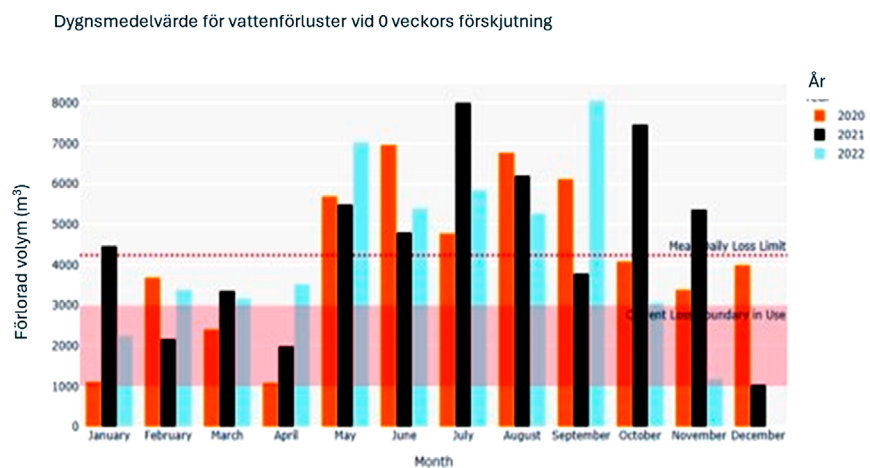
Initialt levererades högupplösta data för en längre tidsperiod där data till stor del aggregerades manuellt. Under projektets gång erhöles möjlighet att få data från ett annat system. Eftersom detta system finns tillgängligt utan behovet av manuell aggregering togs beslutet att fortsätta med data från detta system då det kommer förenkla en möjlig framtida produktionssättning av maskininlärningsmodeller. Den totala tidsperioden av data var dock kortare – två år. Viss diskrepans fanns också mot den ursprungliga datakällan vilket ytterligare stärkte beslutet att använda data från det nyare systemet. Data från de nya uppkopplade nivåsensorerna för grundvattenmätning fanns inte heller tillgängliga för en längre tidsperiod. Två år är en något kort tidsperiod men inom projektet gjordes också ett arbete för att digitalisera historiska data vilket kan användas som utvärdering av framtagna modeller.

Det historiska datasetet var det största sett till tid och täcker en period på 18 år, från 2004 till 2022. Dessa data fanns tillgängliga i en uppsjö av Excel-filer och sammanfogades till ett användbart dataset. Längden på dessa data är idealisk för en maskininlärningsundersökning. Det är inte bara tillräckligt långt för att användas i djupa neurala nätverk, det är också tillräckligt stort för att undersöka och uppskatta hur en modell kommer att presterar över tid. Det bör dock beaktas att systemet har förändrats över tid, förbrukningen av vatten har förändrats, nya leveransområden har tillkommit, driftteknik har förändrats och dessutom har en stor del av data i detta dataset skrivits in manuellt vilket alltid medför en risk.

2.4 Om vattenförluster

För att genomföra en analys av vattenförluster i Örbyfältet behöver det utredas hur förlusten ska definieras. Att beräkna förlusten av vatten från en akvifer är komplicerat. Dels är en akvifer ett komplicerat system på grund av många okända faktorer, dels är det svårt att mäta var förlusterna faktiskt sker. Traditionellt har förlusten definierats som skillnaden mellan beställt vatten och levererat vatten, beräknat på årsbasis för att ta hänsyn till uppehålls- och infiltrationstid. Denna årsvolym har sedan delats på antalet dagar per år för att få fram tidigare redovisade snittvärden per dygn.

Hydraulisk modellering har sedan tidigare visat att uppehållstiden i Örbyfältets akvifer varierar mellan 2–3 veckor i genomsnitt. Detta påverkas av vilket inflöde som finns till fältet och hur pumpning av vatten från fältet hanteras. Inledningsvis undersöktes förlusten av vatten från akviferen genom en tidsförskjutning i enlighet med uppehållstiden för vattnet. Det visade sig att med redan en veckas förskjutning (alltså att förluster uppstår först en vecka efter att vatten tas in till fältet) erhöles instabila modellbeteenden och osannolika resultat, inkluderande positiva förlustvärden för vissa perioder. I Figur 2.5 redovisas förlustberäkningar för en tidsförskjutning på 0–6 veckor. Givet detta resultat, och vad som är känt om komplexiteten i akvifersystem, beslutades att definiera förlusten på enklast möjliga sätt: som skillnaden mellan beställt vatten vid ett givet tillfälle och mängd vatten som levererades ut från fältet vid samma givna tillfälle. Förlustberäkningarna utan tidsförskjutning visar på att överskottet i snitt ligger på cirka 4 100 m³/dygn.

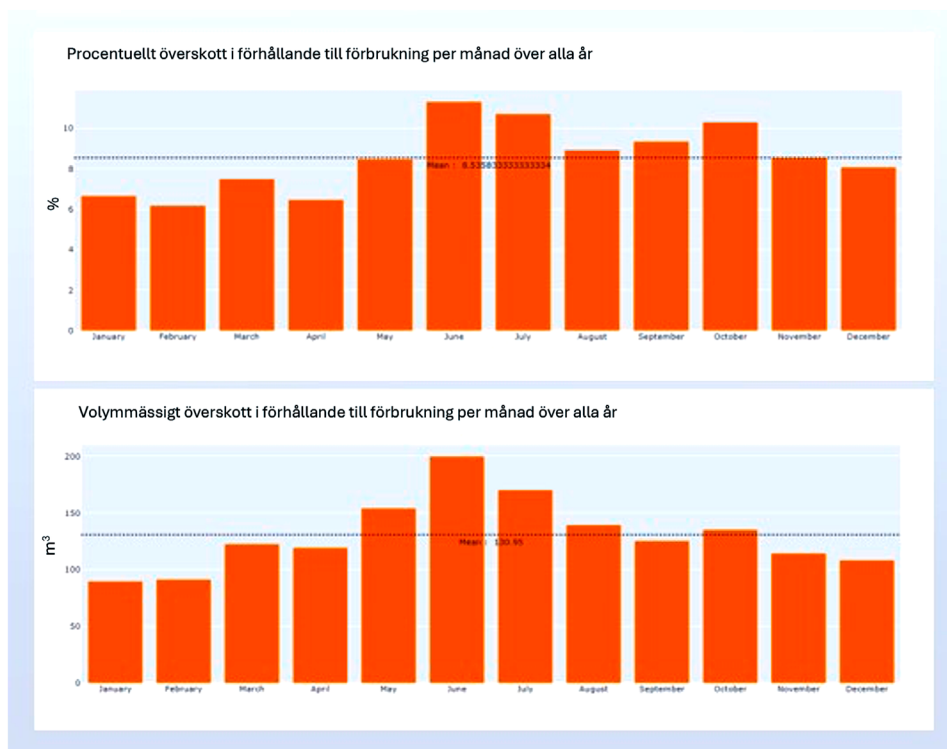


Figur 2.5

Resultat för undersökning om tidsförskjutning för beräkning av förluster. Presenterat som kubikmeter per dag.

I Figur 2.6 ses överst förlusten av vatten, per månad, beräknat i förhållande till förbrukningen, och nedanför den faktiska förlusten beräknad per månad i tusen kubikmeter. De lägsta förlusterna ses i januari- och februarimånaderna med ett ungefärligt värde på 90 000 m³. I junimånaderna uppgår förlusterna till cirka 200 000 m³. I samband med förväntad ökad förbrukning över sommarmånaderna fylls nivåerna i fältet upp från cirka 50 % till 75 % under försommar- och sommarmånader. Att detta leder till ökade förluster är känt sedan tidigare. Förlusterna i förhållande till förbrukat vatten ligger på cirka 6 %

i januari- och februarimånaderna och uppgår till drygt 11 % i junimånaderna. En ökad buffertkapacitet leder alltså inte bara till ökade förluster i faktiska volymer utan också i förhållande till den ökade förbrukningen över sommarmånaderna. Dessutom kvarstår de ökade förlusterna, i förhållande till förbrukningen, med förluster på över 8 % till och med novembermånaderna. Ur ett rent vattenförlustperspektiv, alltså inte inkluderat ett risk- eller leveranssäkerhetsperspektiv, bör alltså uppfyllnad av reservoaren samt överskottet på beställning av vatten utvärderas. Kan beställningar göras utifrån en trovärdig förbrukningsprognos finns möjlighet att minska läckaget.



Figur 2.6

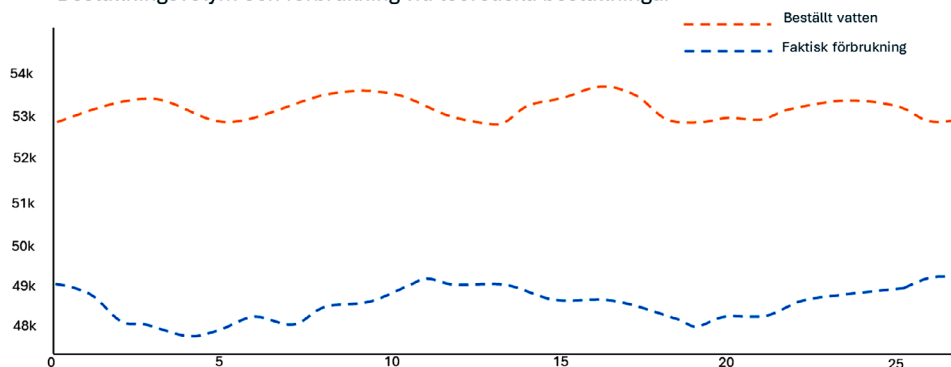
Överst ses förlusten (%) i förhållande till förbrukningen och nedanför den faktiska förlusten i tusen kubikmeter per månad.

2.5 Om beställning av vatten

Beställning av vatten till Örbyfältet genomförs inte på daglig basis, det görs på regelbundna intervall som revideras efter hand som driftoperatörer gör bedömningen att behovet av revidering finns.

Då inga historiska anteckningar finns tillgängliga om när vatten har beställts, och vilken volym beställningarna låg på, har teoretiska datum för beställning av vatten tagits fram. Datum för beställning sattes var 21:e dag på det historiska datasetet för vattenförbrukning, vilket resulterade i 298 teoretiska beställningar. Detta gjordes för att skapa underlag för analys, för att kunna jämföra beställningar och förbrukning samt för att utvärdera möjligheten att analysera beställning av vatten ihop med resultatet från den tidigare framtagna vattenprognosmodellen. De teoretiska beställningarna ska inte missas för faktiska beställningar utan ska ses som en översikt. I Figur 2.7 ses resultatet för ett antal av de teoretiska beställningarna plottat mot faktisk förbrukning. I snitt hade de teoretiska beställningarna ett överskott på 4 684 m³/dag.

Beställningsvolym och förbrukning vid teoretiska beställningar



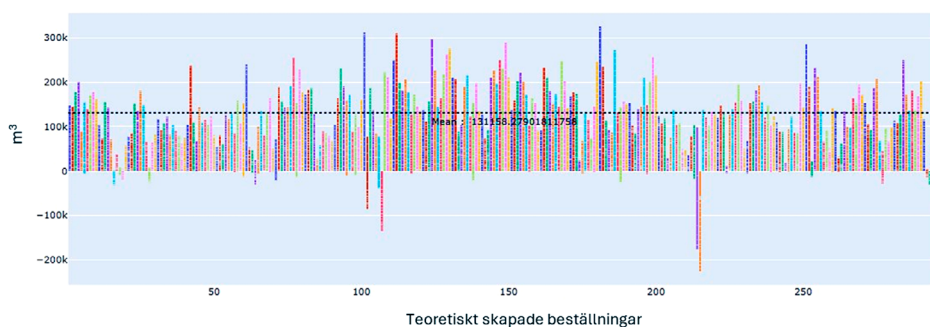
Figur 2.7

Graf som visar snittbeställning per dag (orange), faktisk förbrukning (blått) mot de teoretiska orderdatumen.

I Figur 2.8 ses den beräknade skillnaden mellan beställt och förbrukat vatten för varje beställning. Det är värt att notera att det är stor spridning med både stora överskott och stora underskott. De teoretiska beställningarna har också studerats ihop med faktiska beställningar på vatten där det framgår att operatörerna korrigerar överskott vilket resulterar i att nästa beställning får ett väldigt litet överskott eller till och med underskott. Samma sak kan ses för ökning vid beställning. Vissa av extremvärdena kan också bero på planerad drift eller oväntade händelser som läckage eller leveransstopp.

För de teoretiska beställningsdatumen var det i snitt åtta procent överskott på beställningarna. Ett max på tjugo procents överskott, femton beställningar med över femton procents överskott samt etthundraelva beställningar med ett överskott på mer än tio procent.

Summerat överskott/underskott för teoretiska beställningar och faktisk förbrukning



Figur 2.8

Skillnaden mellan beställt vatten och förbrukat vatten för de teoretiska beställningsdatumen.

3 Maskininlärning i ett bredare sammanhang

3.1 Ramverk och definitioner

Innan en mer komplett definition av maskininlärning (ML) ges, är det viktigt att sätta det i kontext med begreppen datadrivet beslutsfattande, datavetenskap, artificiell intelligens (AI) och statistisk modellering/slutledning. Förkortningarna som anges är vedertagna men baseras på de engelska termerna som finns beskriven i begreppsdefinitionerna.

3.1.1 Datadrivet beslutsfattande

Datadrivet beslutsfattande avser praxis att fatta beslut baserade på dataanalys och empiriska bevis snarare än intuition eller anekdotiska bevis. Detta tillvägagångssätt betonar användningen av data för att vägleda och förbättra beslutsprocesser vilket, i teorin, leder till mer exakta och objektiva insikter (Davenport & Kim 2013).

Några av fördelarna med datadrivet beslutsfattande är:

1. *Förbättrad noggrannhet*; eftersom beslut baserade på dataanalys i allmänhet är mer exakta än de som enbart baseras på intuition (Davenport & Harris 2007).
2. *Förbättrad objektivitet*; data ger en objektiv grund för beslut och minskar därmed risken för att beslut påverkas av fördomar och personliga åsikter (Provost & Fawcett 2013).
3. *Mer informerade strategier*; datadrivna beslut hjälper organisationer att observera trender och utveckla strategier som är baserade på faktiskt observerade data (McAfee et al. 2012).
4. *Ökad effektivitet*; genom att använda data som redan finns tillgängliga inom organisationen kan automatisering och optimering implementeras för att reducera kostnader (Choi et al. 2016).
5. *Prediktiva möjligheter*; möjliggör för organisationer att förutse trender och ta proaktiva beslut (Bishop 2006).

Datadrivet beslutsfattande har fem viktiga aspekter:

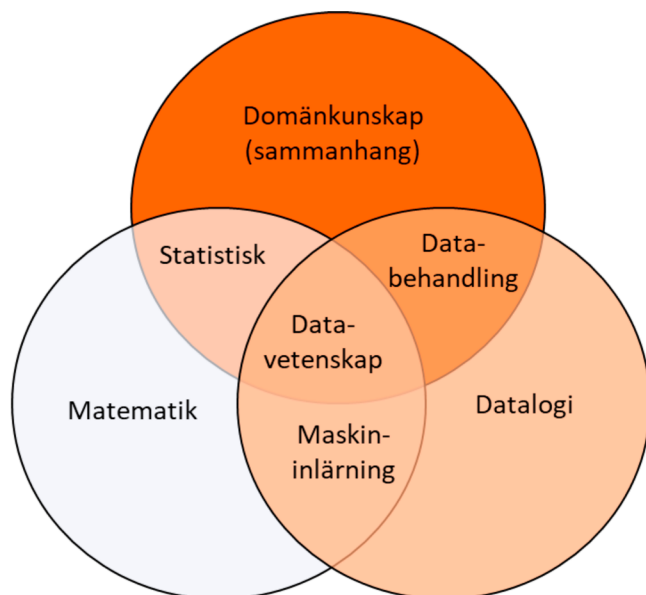
1. Identifiera vilka beslut som ska beaktas och varför.
2. Prioritera beslut, analys och data.
3. Integrera AI (vilket inkluderar maskininlärning) i beslutsfattande.
4. Omvärdera dina data och ditt analyssystem.
5. Bygg kompetens, vanor och organisationsstrukturer inriktade på effektivt beslutsfattande.

3.1.2 Datavetenskap

Datavetenskap är ett tvärvetenskapligt ämne som använder matematik/statistik, mjukvaruutveckling, vetenskapliga beräkningar och domänkunskap i kombination med stora datamängder för att utvinna värdefulla insikter och kunskap. Med domänkunskap avses specialiserad kunskap om hur ett system fungerar, vilka frågeställningar som är relevanta och vilken information som behövs för att fatta välgrundade beslut. Domänkunskap innefattar även förståelse för terminologi, metoder och samband, vilket är avgörande för att identifiera vilken information och insikt som bör inkluderas i projektet för att uppnå ett bra och användbart resultat. Värt att notera är att det engelska begreppet Data Science också används inom svensk terminologi, konceptet illustreras i Figur 3.1.

Datavetenskap involverar flera centrala koncept och aktiviteter som är kritiska för de flesta projekt:

1. *Datainsamling*; inhämtande av data från flertalet olika källor.
2. *Databearbetning och förbehandling*; omvandling av rådata så att data blir lämpliga och möjliga att använda i analys.
3. *Utforskande dataanalys (Exploratory Data Analysis, EDA)*; användandet av statistik och visualisering för att förstå data.
4. *Datavisualisering*; verktyg för att kunna kommunicera insikter genom grafisk representation.
5. *Byggandet av en datamodell*; tillämpning av statistiska tekniker för att utvinna insikter/kunskap från data. Detta kan ske genom användning av beskrivande eller diagnostisk analys, statistisk analys eller maskininläring.
6. *Data Engineering*; hantering av data, arkitektur, lagring och flöden för dataanvändning.
7. *Leverans av erhållna insikter*; utveckling av rapporter och visuell redovisning av data (dashboards) för att stödja beslutsfattande.



Figur 3.1

Venn diagram över datavetenskap, ett tvärvetenskapligt ämne.

3.1.3 Artificial Intelligence (AI)

AI avser den del av datavetenskapen som är dedikerad till att skapa system som kan utföra uppgifter som typiskt kräver mänsklig intelligens. AI innefattar en bred mängd tekniker och tillämpningar. Generellt brukar de delas in i följande kategorier:

- *Narrow AI/Weak AI*; utformade för att utföra specifika uppgifter. Exempelvis virtuella assistenter såsom röststyrda assistenter som finns i flertalet moderna mobiltelefoner, självkörande bilar, rekommendationssystem och bildigenkänning. Dessa typer av system utmärker sig inom specifika områden men saknar generell intelligens eller anpassningsförmåga.
- *General AI/Strong AI*; hypotetisk AI som besitter förmågan att förstå, lära sig och tillämpa intelligens över ett brett spektrum av uppgifter på en jämförbar nivå med mänskliga kognitiva förmågor. General AI/Strong AI är fortfarande ett teoretiskt koncept som ännu inte har uppnåtts.

3.1.4 Statistisk modellering, analys och inferens

Statistisk modellering avser processen att skapa matematiska modeller som kan representera relationer i data och prediktera värden. Detta innefattar vanligtvis ett antal steg:

1. *Val av modell*; valet av en lämplig modell utifrån tillgängliga data och mål som ska uppnås. Vanliga modeller innefattar linjär regression, logistisk regression och tids-seriemodeller såsom ARIMA (Box et al. 2008).
2. *Bedömning/skattning av parametrar*; bedömning av parametrarna för den valda modellen utifrån metoder såsom Maximum Likelihood Estimation eller Least Squares Estimation (Hamilton 1994).
3. *Modellvalidering*; att utvärdera modellens prestanda och robusthet med hjälp av tekniker som korsvalidering, residualanalys och mått på anpassningens kvalitet (goodness-of-fit) för att säkerställa att modellen ger pålitliga förutsägelser och insikter (Breiman et al. 1984). En annan utvärderingsmetod som är vanlig inom hydrauliska tillämpningar är Nash-Sutcliffe Efficiency (NSE).
4. *Tolkning*; analys av modellens resultat för att säkerställa meningsfulla slutsatser och för att kunna fatta beslut baserat på de skattade sambanden och prognoserna (Gelman et al. 2013).

Statistisk analys innefattar processer för att samla in, utforska och tolka data för att identifiera mönster, trender och relationer. För att genomföra en statistisk analys används en rad olika tekniker såsom beskrivande statistik (exempelvis medel, median och variation) samt mer avancerade metoder såsom regressionsanalys och olika former av utvärdering/test av hypoteser. Målet är att sammanfatta och tolka ett datasets underliggande struktur, vilket ger insikter som kan vägleda beslutsfattande (Moore et al. 2016). Ett exempel på regressionsanalys är undersökningar som förklarar en slutsats utifrån bakomliggande faktorer såsom åsikt i vissa frågor utifrån ålder, kön, utbildning och andra sociala faktorer.

Utforskande dataanalys (EDA) är ett viktigt verktyg inom datavetenskap och maskininlärning och är en form av statistisk analys. Inom EDA ligger fokus på att analysera dataset för att summera dess karakteristik genom användandet av statistiska nyckeltal med fokus på variabilitet inom data.

Inom EDA ingår också verktyg för att kunna visualisera och kommunicera analys genom användandet av histogram, sambandsdiagram och olika typer av grafer (Tukey 1977). Målet är att kunna identifiera mönster, trender, anomalier och relationer i data innan tillämpning av mer djuplodande statistik analys och statistisk inferens. Ofta ställer utforskande dataanalys krav på verktyg för att kunna bearbeta stora datamängder.

Statistisk inferens är en statistikgren där fokus ligger på att estimerar olika värden för en population (inom det använda datasetet) med hjälp av stickprov. En typisk annan tillämpning för inferentiell statistik är opinionsundersökningar där man försöker erhålla slutsatser genom att ett representativt urval ur en population förhoppningsvis ska kunna representera trender i hela populationen. Tekniker såsom konfidensintervall, test av hypoteser och användandet sannolikhetsanalys används för att bedöma prediktioner, samband och osäkerheter i förklaringsmodeller (Casella & Berger 2002). Ett av målen är att kunna utvärdera kvalitén på ens förklaringsmodeller och att avgöra hur rimligt det är att tillämpa dem på data som finns utanför det använda datasetet (exempelvis prediktioner).

Sammanfattningsvis används statistik analys för att beskriva och förstå tillgängliga data och statistisk inferens för att möjliggöra prediktioner och generaliseringar baserat på delmängder ur data (Wasserman 2004).

3.2 Maskininlärning

Maskininlärning (ML) är en viktig komponent inom AI som fokuserar på att utveckla algoritmer som kan lära sig av data och göra förutsägelser eller ta beslut baserat på dessa data. Inom det bredare AI-området representerar ML en uppsättning tekniker som är utformade för att förbättra systemens förmåga att autonomt förbättra sin prestanda. Inom datavetenskap används ML för att analysera stora datamängder, identifiera mönster och generera prediktiva modeller som stödjer datadrivet beslutsfattande. Till skillnad från mer traditionell statistisk modellering, som ofta förlitar sig på fördefinierade ekvationer, samband och antaganden, anpassar och förfinar ML-algoritmer sig genom iterativa inlärningsprocesser, vilket innebär test, utvärdering, korrigering och nya tester tills önskat resultat är uppnått. Denna skillnad gör att ML kan hantera högdimensionella data mer effektivt vilket gör det till ett kraftigt verktyg för analys och beslutsstödsystem. Ett exempel på detta är användandet av ML inom hydraulisk modellering där ML-algoritmer kan utföra tunga komplexa beräkningar på sekunder. Beräkningar som vanligtvis kräver stor datorkraft och tar lång tid att utföra med traditionella metoder. Dessutom kan ML-algoritmer uppdateras och lära sig av ny information efterhand som mer data erhålles, så kallad continuous learning.

3.2.1 Från traditionella statistiska modeller till maskininlärning

Generellt utnyttjar ML-modeller statistiska principer och metoder och utvidgar dem för att kunna hantera mer komplexa och storskaliga problem. Genom att utvidga de mer klassiska statistiska principerna kan data analyseras på ett mycket mer omfattande sätt vilket möjliggör både nya insikter och användandet av mer data. Beräkningsalgoritmer används för att iterera modellen och det gäller att definiera vad som är "positiva" förändringar för att på så sätt kunna nyttja den maskinella beräkningsdelen inom ML. Det är denna inlärningsprocess som skiljer ML från traditionella statistiska modeller. Exempelvis är linjär regression en traditionell statistisk teknik men kan även betraktas som en ML-teknik när den formuleras som ett (övervakat) inlärningsproblem.

Andra ML-metoder såsom random forest och neurala nätverk har ingen traditionell motsvarighet men bygger likväl på statistiska principer. Den iterativa inlärnningen från ett dataset gör ML-modeller mer flexibla än traditionella statistiska metoder.

3.2.2 Bakgrundshistorik till maskininlärning

ML har sina rötter i mitten av 1900-talet och uppstod i skärningspunkten mellan datavetenskap, statistik och artificiell intelligens. Ursprunget till ML kan spåras tillbaka till 1950-talet, med tidiga pionjärer som Alan Turing, som föreslog konceptet med en maskin som kunde "lära sig av erfarenhet" vilket han beskrev i sin berömda uppsats från 1950: *Computing Machinery and Intelligence*. Frank Rosenblatt introducerade 1957 perceptronen, en av de tidigaste modellerna av neurala nätverk vilket markerade det första steget mot algoritmer som kunde efterlikna mänskligt lärande. År 1959 publicerade Arthur Samuel en uppsats om ett program för att spela dam (checkers) som kunde förbättra sin prestanda genom erfarenhet. Genom denna uppsats myntades termen *Machine Learning*. Värt att notera är dock att detta program inte använde sig av neurala nätverk utan följde en mer traditionell ansats.

Under 1960-talet fortsatte utvecklingen av grundläggande arbete vilket ledde fram till ett antal populära traditionella maskininlärningsalgoritmer som fortfarande används i dag: K Nearest Neighbors (Cover & Hart 1967), K Means (MacQueen 1967) samt tidiga klustermetoder som Hierarchical Clustering (Johnson 1967) liksom det grundläggande arbetet inom beslutsträd (*decision trees*) med Hunts algoritm (Hunt et al. 1966).

Trots framsteg under 1970-talet avtog den generella entusiasmen när vi gick in i vad som räknas som den första av två "AI-vintrar" (Crevier 1993). 1974–1980 var en period präglad av minskad finansiering och intresse för AI vilket tillfälligt hämmade framsteg

inom maskininlärning. Slutet av 1980-talet och början av 1990-talet benämns ofta som den andra "AI-vintern". Trots generell minskad finansiering fortsatte forskare att utveckla och förfinas ML-algoritmer och den teoretiska grunden med ett skifte mot probabilistiska modeller och neurala nätverk, inspirerade av hjärnans struktur och funktion. Mot slutet av decenniet (och milleniet) återuppväcktes intresset för maskininlärning då algoritmer som tillbakaförökning (*backpropagation*) möjliggjorde träning av flerskiktade neurala nätverk (Rumelhart et al. 1986). Utan detta grundläggande arbete skulle djupa neurala nätverk inte kunna utföra uppgifter inom exempelvis bildigenkänning och tolkning av naturligt språk.

Under andra halvan av 1990-talet och början av 2000-talet inträffade en unik händelse som ökade mängden data och behovet av nya analysverktyg exponentiellt: spridningen av internet. Med detta som motor exploderade utvecklingen inom ML som den dominanta kraften inom AI. Utvecklingen av tekniker såsom stödvektormaskin (*support vector machine*) (Vapnik 1995), ensemble-metoder såsom beslutsträd (*Random Forest*) (Breiman 2001) och djupa neurala nätverk (neurala nätverk med 3 eller fler lager) (Hinton et al. 2006) möjliggjordes tidiga kommersialiseringar av ML vilket ledde till massivt ökade investeringar inom sektorn.

Vikten av utvecklingen inom denna tidsperiod synliggjordes så sent som den 8 oktober 2024 i och med tillkännagivandet av nobelpriset i fysik som tilldelades Geoffrey E. Hinton och John J. Hopfield för deras arbete med att träna artificiella neuronnät med fysik.

2010-talet var ett transformativt årtionde för ML, drivet av teknisk utveckling inom djupa neurala nätverk och med introduktionen av ramverk (baserat på öppen källkod) såsom Tensorflow och PyTorch. Tensorflow släpptes på marknaden av Google 2015 och gav forskare och utvecklare tillgång till en flexibel plattform för att bygga och distribuera ML-modeller i stor skala.

Kort därefter, 2016, släppte Facebook PyTorch som snabbt växte i popularitet då ML-bibliotek kan användas ihop med programmeringsspråket Python vilket möjliggör snabb utveckling och distribution av data processade av ML-algoritmer. Dessa ramverk, med flera, demokratiserade utvecklingen inom ML och möjliggjorde att mängden användare och tillämpningar av ML-algoritmer formligen exploderade.

Sedan 2020-talets början har vi sett framväxten av flera imponerande ML-modeller, särskilt inom området djupinlärning. Tillämpningen av ML inom autonom körning har drivit utvecklingen framåt då behovet av realtidsbeslutsstödssystem är en förutsättning för exempelvis självkörande fordon. Stora språkmodeller, som GPT-4, har revolutionerat naturlig språkbehandling genom att möjliggöra mer nyanserad "dator-människa-interaktion" och innehållsgenerering (Brown et al. 2020). Inom klimatforskningen har tillämpningen av ML också haft en betydande roll för att skapa bättre, fler och snabbare beräkningar av klimatförändringar och meteorologiska fenomen. Användning av klimatmodeller har tidigare varit begränsad till "superdatorer" men kan numera tillämpas och utvecklas av många fler (Rolnick et al. 2022). Utvecklingen inom neurala nätverk pågår ständigt och nya metoder för hur olika tekniker kan förbättra användning och tillämpning utvecklas hela tiden (Song et al. 2024).

Den breda sektoriella användningen av ML märks också av i vattenbranschen där nya verktyg och tillämpningar tillkommer nästan varje vecka. Under den andra halvan av 2024 presenterades det väldigt många modeller som kombinerar olika typer av ML-algoritmer med existerande hydrauliska modeller/beräkningsverktyg, så kallade hybridmodeller.

3.2.3 Uppgifterlämpliga för maskininlärning – med exempel från VA-branschen

ML kan tillämpas på ett flertal olika uppgifter över flera områden. Nedan följer ett antal exempel:

Klassificering: ML kan klassificera olika typer av data i fördefinierade kategorier. Exempel på detta är bildigenkänning och skräppost-igenkänning (Murphy 2012). Inom dricksvattendistribution kan förbrukningsmönster eller läckor klassificeras för att skapa insikter om möjliga problem. Satellitbilder eller kamerabilder kan analyseras för att bedöma ett ledningsnätets kondition, något det finns färdiga kommersiella produkter tillgängliga för.

Regression: ML kan användas för att prognosticera exempelvis huspriser eller trender på börsen (Hastie et al. 2009). Eller för den delen kommande dricksvattenförbrukning eller inkommande flöde till reningsverk.

Klusteranalys: ML kan användas för att gruppera data inom kluster baserat på likhet i både variation och mönster. Exempel på detta är klassificering av tillskottsvatten utifrån typ (snabb regnpåverkan eller trög regnpåverkan).

Dimensionalitetsreduktion: ML kan reducera antalet variabler samtidigt som viktig information bibehålls, vilket underlättar visualisering och utmaningar inom hantering av högdimensionella data. Ett exempel är att man kan reducera antalet dimensioner i en beslutsstödsmatris vilket tillämpas på komplexa vattenresursproblem där det ofta är väldigt många variabler som är involverade i beslutsstödsmatriser.

Anomalidetektion: ML kan identifiera extremvärden eller sällsynta händelser vilket används inom bedrägeri-identifiering och nätverkssäkerhet (Chandola et al. 2009). Exempel på existerande tillämpningar inom vattenbranschen är detektering av läckor inom dricksvattendistribution samt bedömning av vattenkvalitetsparametrar.

Språkteknologi: (Natural Language Processing) ML bearbetar mänskligt språk vilket möjliggör uppgifter som sentimentanalys (ofta benämnt *opinion mining*), översättning och användandet av chattbotar. Exempel från vattenbranschen innefattar att kategorisera och märka upp stora mängder anteckningar från driftspersonal.

Bild- och videobehandling: ML kan tolka och analysera visuella data vilket har lett till framsteg inom objektsigenkänning, ansiktsigenkänning och autonom körning (Krizhevsky et al. 2012). Exempel från vattenbranschen är analys av bild – och videomaterial för realtidsövervakning av igensättning (exempelvis i utlopp, ledningsnät eller på anläggningar).

Tidsserieprognoser: ML kan användas för att prognosticera framtida värden baserat på historiska data vilket används inom allt från finansmarknader till väderprognoser (Hyndman & Athanasopoulos 2018). Exempelen från vattenbranschen är många och varierar från relativt enkla prognoser för exempelvis flöden till mer komplexa prognoser som innefattar större delen av hela anläggningar för vattenrening. Tidsserieprognoser skiljer sig från regression genom att tidens sekvens ges mycket större betydelse i modellen. I regression modelleras främst relationen mellan variabler, med mindre eller inget fokus på tidsaspekten.

Rekommendationssystem: ML rekommenderar produkter eller innehåll baserat på användarens preferenser och tidigare beteenden, allmänt känt från plattformar såsom Netflix eller Amazon (Ricci et al. 2015). Exempel från vattenbranschen kan vara förslag för driftsstrategier av reningsverk för att optimera processen. Ofta är rekommendationssystem uppbyggda av ett flertal olika ML-teknologier.

Det är värt att notera att framstegen inom numeriskt baserade system (såsom tidsserieprognoser och klassificering) är mindre komplicerade än inom vissa andra tillämpningar (såsom språkteknologi och bildigenkänning). Olikheter beror delvis på den ökade komplexiteten i data samt ett mindre fokus på den kollektiva, öppna källkodbaserade, utvecklingen av generiska basmodeller. Trots detta har vi sett framsteg genom att tekniker har flyttats mellan de olika kategorierna. Exempelvis finns det i dag numeriskt baserade system som innefattar språkteknologier för att på så sätt kunna ställa direkta frågor om samband i en datamängd.

3.2.4 Framtiden för maskininlärning

I takt med att ML fortsätter att utvecklas förväntar vi oss betydande genombrott då tillgänglighet och anpassningsbarhet av ML-modeller fortsätter att utvecklas vilket medför att användandet av ML-tekniker kommer att spridas och vara mindre begränsat till specialistkompetens. Många exempel på detta finns redan i dag och att använda ML-tekniker är i dag avsevärt mycket enklare än vad det var för bara 2 år sedan.

Ett problem som ofta nämns inom vattenbranschen är att en stor del av data som finns tillgängliga är känsliga och sekretessbelagda. Detta medför att det är svårt att samla in tillräckligt stora datavolymer för att utveckla generella modeller, alltså modeller som predikterar väl på tidigare osedda data. En tränad modell blir helt enkelt mindre "flyttbar" om den inte tränas på en tillräckligt stor variation av data. Utveckling inom generalisering och anpassningsbarhet pågår dock vilket kommer innebära att modeller blir mer flexibla och kapabla att lära sig med mindre data och färre etiketter (*labels*) vilket kommer möjliggöra mer generaliserade tillämpningar inom AI (Marcus 2020).

Ännu mer spännande är dock utvecklingen inom integritetsbevarande modeller (*privacy preserving models*) vilket är ett begrepp som innefattar allt från att tillämpa krypteringstekniker på träningsdata samt användningen av federerad inlärning. Det sistnämnda innebär att en algoritm kan tränas på flera olika datorer och/eller servrar utan att data flyttas från dataägaren vilket minskar risken för dataläckage. I Sverige finns i dag många uppkopplade vattenmätare installerade. De är dock spridda och ur ett GDPR-perspektiv är det känsligt att dela dessa data. Genom att använda sig av federerad inlärning skulle alla data kunna användas för att träna upp en algoritm utan att data överhuvudtaget behöver lämna dataägarens IT-miljö, en spännande möjlighet.

Utveckling inom kvantdatorer möjliggör också snabbare utveckling inom ML genom att lösa komplexa optimeringsproblem som klassiska datorer har svårt med (Biamonte et al. 2017).

3.2.5 Traditionell maskininlärning och neurala nätverk

ML kan grovt delas in i två kategorier: traditionell och neuralt nätverksbaserad. Det är viktigt att poängtera att termen "traditionell" syftar på de metoder som används för att konstruera modellen snarare än att det innebär användandet av äldre eller förlegad teknologi. Det första neurala nätverket utvecklades långt innan termen maskininlärning ens blev etablerad i vetenskaplig litteratur. Termen traditionell ML speglar alltså modellmetoder och har inget alls att göra med att det skulle representera en äldre teknik.

I båda metoderna används data för att träna en modell. Träningsprocessen fastställer ett generellt beteendemönster som sedan kan användas för olika typer av prediktioner. Hur modellen utformas skiljer sig dock åt mellan traditionell maskininlärning och nätverksbaserad maskininlärning. Den största skillnaden ligger i hur olika arkitekturer möjliggör inlärning för algoritmerna.

3.2.6 Traditionell ML

Traditionella ML-modeller har likheter med statistiska modeller och är i vissa fall statistiska modeller som ramas in som ett övervakat inlärningsproblem, exempelvis linjär

regression. I det fallet anpassas datapunkter till en rät linje och målet med inläringen är att hitta de bästa koefficienterna för den fördefinierade ekvationen. I andra fall är teknikerna helt maskininlärningsbaserade såsom *Random Forest*. Denna teknik, utvecklad av Leo Breiman 2001, är en ensemblemetod som bygger på att skapa fler "beslutsträd" från slumpmässiga delmängder av träningsdata och variabler. Varje träd tränas individuellt på delmängder av data och ger antingen en klassificering eller en regression (beroende på typ av uppgift). För klassificering tar Random Forest den vanligaste klassificeringen från alla träd (majoritetsbeslut) medan det för regression tar medelvärde av alla träds förutsägelser. Random Forest räknas som en väldigt robust algoritm som fungerar bra för både regressions- och klassificeringsuppgifter. Den är särskilt effektiv vid hantering av stora dataset med många variabler men kan bli resurskrävande i träningsfasen.

Tillämpningen av ML-algoritmer får i dag anses vara förhållandevis enkel. Ett par rader kod, eller användning av program/web-baserade lösningar gör det enkelt att tillämpa ML på sin data, i liten skala. Men för att få en god förståelse för resultatet, utvärdera olika metoder och algoritmer samt sätta resultatet i produktion behövs en bred kompetens.

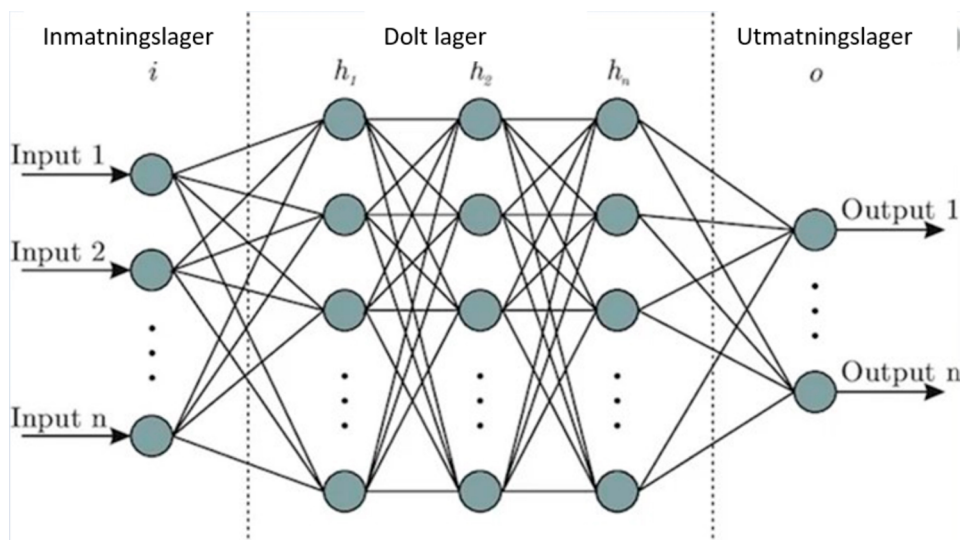
3.2.7 Maskininläring baserad på neurala nätverk

Neurala nätverk (NN) är mer abstrakta än traditionella ML-algoritmer vilket kommer från den mekanism som neurala nätverk bygger på. Som namnet antyder försöker NN-algoritmer efterlikna biologiska neurala nätverk där ett nätverk består av ett antal sammankopplade noder eller "neuroner" som arbetar tillsammans för att lära sig från data.

Arkitekturen består av lager: inmatnings-, dolda och utmatningslager. Först matas data in genom ett inmatningslager. Man kan likna det vid att varje nod i detta lager representerar en enskild bit information. Om det exempelvis är en bild kan varje nod motsvara en pixel i bilden. Denna information skickas därefter vidare till nästa lager.

I de så kallade dolda lagren bearbetas data och kombineras på olika sätt. Varje nod i dessa lager är kopplad till andra noder via vikter, vilket är numeriska värden som bestämmer hur mycket en nods information ska påverka nästa nod. Vikterna initieras slumpmässigt och modellen genomgår en träningsprocess för att bestämma vilka vikter som behöver förändras för att uppnå målbilden. Även om vikterna initieras slumpmässigt, är det den strukturerade träningsprocessen som gör att nätverket kan lära sig från data och gradvis förbättra sina förutsägelser. Aktiveringsfunktionen kan liknas vid hur hjärnceller 'aktiveras' för att skicka signaler vidare.

Neurala nätverk kan klassificeras som grunda eller djupa där de senare har tre eller flera lager (inklusive inmatnings- och utmatningslager) och är bättre lämpade för mer komplexa problem med högre datadimensioner. Djupa neurala nätverk brukar benämnas som *deep learning*. Ofta beskrivs neurala nätverk med den typ av generell bild som visas i Figur 3.2. Dessa typer av schematiska beskrivningar kan vara väldigt svårtolkade för en djupare förståelse av vad som pågår i ett neuralt nätverk.

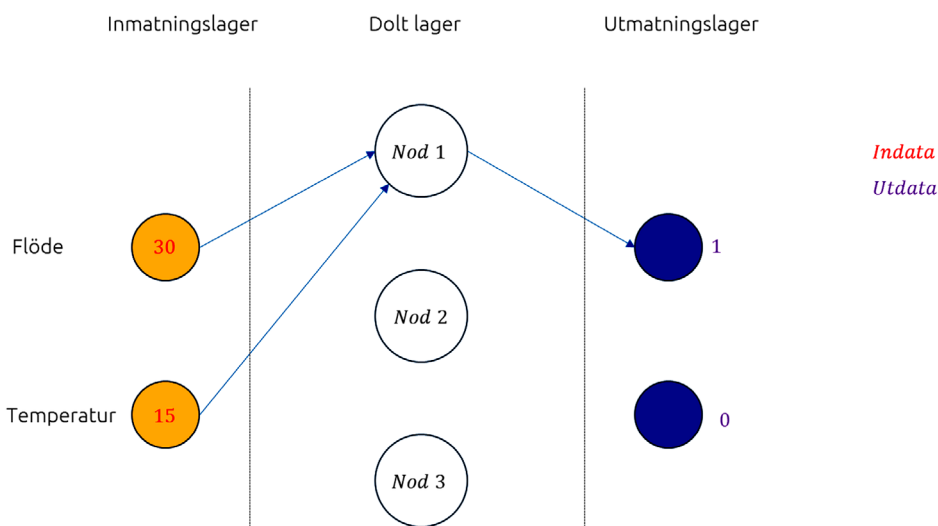


Figur 3.2

Mycket schematisk översikt av ett neuralt nätverk.

För att schematiskt visa på hur ett neuralt nätverk fungerar, och hur det fundamentalt skiljer sig från traditionell ML, redovisas några steg i hur data behandlas i ett neuralt nätverk. Observera att detta inte är en komplett bild av hur ett neuralt nätverk fungerar utan exempel för att öka förståelsen av tekniken.

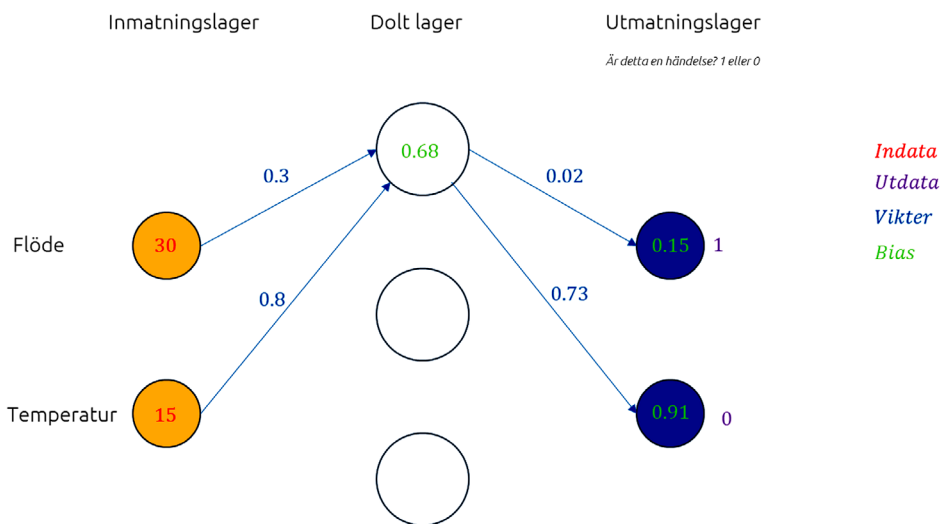
Till vänster i Figur 3.3 redovisas en punkt i ett indataset, här exemplifierat med ett flöde och en temperatur. Utifrån dessa indata vill vi träna ett neuralt nätverk att prediktera ett värde på ett eller noll. I detta fall arbetar vi med träningsdata och känner till utfallet. Vad ett eller noll betyder är irrelevant för exemplet. Från inmatningslagret skickas informationen vidare till nod ett, nod två och nod tre.



Figur 3.3

Schematisk översikt över ett neuralt nätverk med ett dolt lager.

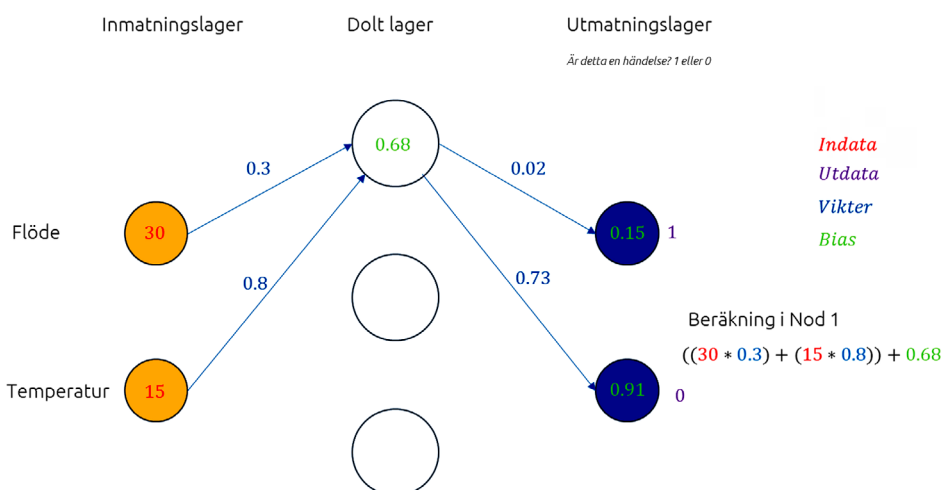
För att hitta ett samband mellan indata och utdata tillsätts slumpmässiga vikter och bias initialt, dessa värden har ingen egentligen relevans utan är bara en utgångspunkt för att kunna få iterativa beräkningar att initieras. Exempelvärden är redovisade i Figur 3.4.



Figur 3.4

Schematisk översikt över ett neuralt nätverk med fiktivt tilldelade vikter och bias-värden.

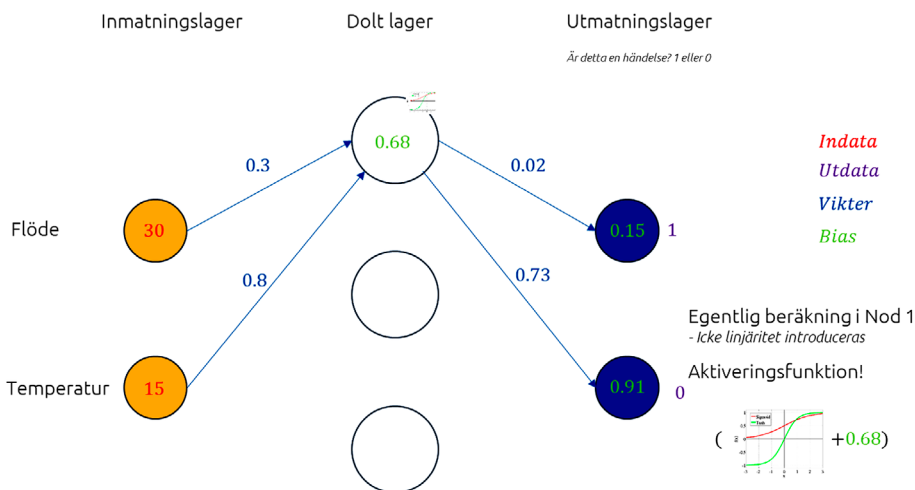
För att illustrera hur beräkningar sker så redovisas i Figur 3.5 hur en enkel linjär regression skulle kunna beräknas i en nod. Detta är en strikt teoretisk beskrivning som är grovt förenklad för att illustrera hur beräkningar sker. För ett rent "linjärt" problem hade detta räckt som beräkningsmetod. Men verkligheten styrs ofta inte av linjära samband. En rent linjär beräkningsmetodik i moderna skulle därmed medföra alltför stora begränsningar för neurala nätverk.



Figur 3.5

Schematisk översikt över hur en linjär beräkning kan ske i en nod utifrån indata, vikter och bias.

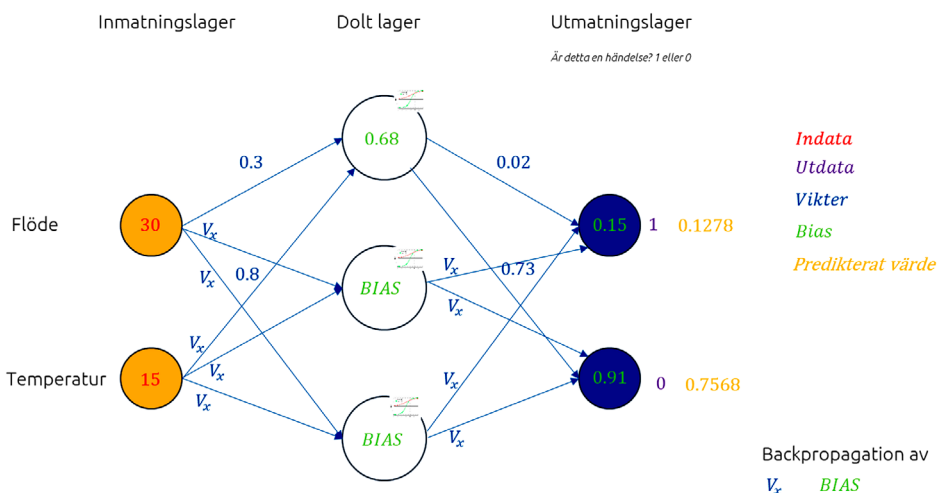
In träder aktiveringsfunktionen, en av de verkligt spektakulära delarna inom neurala nätverk. I stället för att beräkna en linjär funktion mellan de viktade värdena introduceras icke linjära beräkningsmetoder. Ett schematiskt exempel för att illustrera skillnaden redovisas i Figur 3.6. Ofta introduceras hyperboliska funktioner (sin h, cosh, tanh osv) men även andra typer av matematiska funktioner används. För översiktens skull lämnar vi detaljerna i aktiveringsfunktionen och går vidare med att konstatera att viktade indata, ihop med bias för beräkningsmetoderna beskrivs av ett komplexare matematiskt samband och att beräkningarna resulterar i ett värde mellan 0 och 1.



Figur 3.6

Schematisk översikt av aktiveringsfunktionen i noder.

I Figur 3.7 görs ett försök att redovisa hur summan av alla ekvationer, som är justerade med vikter, bias och aktiveringsfunktion, levererar ett prediktionsvärde. Initialt kommer detta med största sannolikhet att vara felaktigt. Nätverket måste få en chans att lära sig och anpassa sina vikter och bias och det är här *backpropagation* kommer in i bilden.



Figur 3.7

Schematisk översikt som syftar till att visa på komplexiteten i beräkningar inom ett neutralt nätverk där backpropagation är ett centralt verktyg för att förbättra resultatet från ett neutralt nätverk.

I grunden är detta en form av algoritm som används för att träna neurala nätverk för att minimera fel. Efter att information har nått utdatalagret beräknas *förlustfunktionen*, vilket i grund och botten är felet mellan rätt svar och det resultat som nätverket har producerat.

”Felet” distribueras tillbaka till nätverket och informerar varje nod om dess bidrag till det totala felet. Därefter justeras vikter och bias för att minimera detta fel i så stor utsträckning som möjligt. Detta görs genom en teknik som kallas för *gradient descent* vilket egentligen kräver ett helt kapitel i sig självt. Men i grund och botten handlar det om att vikter och bias justeras till den punkt där en ökning eller minskning av det numeriska värdet resulterar i ett större fel. Backpropagation är alltså en central teknik som är avgörande för att ett neutralt nätverk ska kunna uppnå goda resultat och vara generellt tillämpbart (på tidigare osedda data).

Backpropagation kan ske efter varje inmatat värde men vanligare är att det sker efter en grupp av värden (mini-batch) för att undvika att nätverket överanpassas till enstaka extremvärden. Det går också att tillämpa backpropagation på hela träningsdatasetet men detta kan vara mycket krävande och används generellt endast för mindre dataset.

Den observanta läsaren har noterat att flödet av information än så länge har rört sig från vänster till höger, i detta exempel beskrivs ett så kallat *feedforward network* (d.v.s. från indatalager mot utdatalager). Indata och beräknade värden går från vänster till höger, däremot används backpropagation för att justera vikter och bias men informationen går fortfarande från vänster till höger. Det finns neurala nätverk som fungerar annorlunda, exempelvis återkopplande neurala nätverk (*Recurrent Neural Network*), faltningsnätverk (*Convolutional Neural Network*) som har tekniker för att föra tillbaka själva informationen i träningssteget. När alla datapunkter i träningsdatasetet har skickats genom det neurala nätverket, och vikter och bias har justerats är det troligen dags att göra det igen. En *epoch* är en komplett genomgång av hela träningsdatasetet och att bestämma antalet epochs är viktigt både för slutresultatet och för hur lång tid det tar att träna nätverket. För få epochs leder till otillräcklig inlärning medan för många kan leda till överanpassning.

När en träningsprocess ska avslutas bestäms ofta av en kombination av kriterier som kan inkludera:

- Förbestämt maximalt antal epochs har uppnåtts
- Ingen ytterligare förändring av vikter eller bias sker
- Förlustfunktionen har nått önskat värde
- Önskad noggrannhet för hela träningsdatasetet är uppnått.

Även om statistiska metoder är viktiga för utformningen och funktionen av neural nätverksarkitektur är slutresultatet mindre beroende av dessa metoder. I stället kommer resultatet från neurala nätverk ur ett mer abstrakt och delvis slumpmässigt samspel mellan aktiveringsfunktionerna, de slumpmässigt tilldelade vikterna och användardefinierade lager. Målet med detta slumpmässiga samband är att minimera förlustfunktionen. Det är denna abstraktion och slumpmässighet som gör att djupa neurala nätverk ofta klassificeras som *black-box*-metoder som för närvarande inte är fullt tolkningsbara. Även om det finns verktyg, som LIME och SHAP som hjälper till att öka tolkningsbarheten hos neurala nätverk, föredras ibland traditionella maskininlärningsmetoder framför neurala nätverk när tolkningsbarhet är avgörande.

En mer detaljerad schematisk beskrivning av ett NN kan ses i Bilaga A.

3.2.8 Likheter mellan traditionell ML och NN

Trots skillnad i tillvägagångssätt och komplexitet delar traditionell och neural nätverksbaserad maskininlärning flera grundläggande principer och metoder. Nedan beskrivs några av de viktigaste gemensamma begreppen och metoderna.

Mål med inlärning – förutsägelser och klassificering: Både traditionell ML och NN syftar till att lära sig från data för att göra förutsägelser eller klassificeringar. Oavsett om metoder såsom linjär regression, beslutsträd (*decision trees*) eller neurala nätverk används är målet att modellera de underliggande mönstren i data och göra exakta prognoser eller kategoriseringar (Hastie et al. 2009).

Träningsprocess – datadriven inlärning: Båda tillvägagångssätten är beroende av träningsdata för att utveckla en modell. Data används för att justera parametrar (vikter i NN, koefficienter i traditionell ML) för att minska prognosfel. Traditionella maskininlärningsmetoder använder optimeringstekniker som gradientnedstigning på ett liknande sätt som neurala nätverk justerar sina vikter genom *backpropagation* (Rumelhart et al. 1986). En generell skillnad mellan de båda metoderna är att neurala nätverk går igenom flera träningsloopar (*epochs*) för att bli mer exakta. Traditionella maskininlärningsalgoritmer är oftast utformade för att nå ett slutresultat under en, eller ett begränsat antal, iterationer.

Feature Engineering – vikten av databehandling: Båda metoderna drar nytta av att förbereda data. Inom traditionell maskininlärning är urval och extraktion av data ofta avgörande för modellens prestanda. Praktiska exempel på detta inkluderar att ta bort ofullständiga rader, göra data enhetliga och eliminera extremvärden eller felaktiga data. I detta sammanhang kan även skapandet av nya funktioner eller variabler (features) från befintliga data spela en viktig roll.

Inom neurala nätverk, särskilt i djupinlärning, är betydelsen av databehandling något mindre eftersom denna process i teorin kan automatiseras inom arkitekturen. Det gäller i högre grad för moderna arkitekturer där en del av databehandlingen sker automatiskt genom inlärningsprocessen. Trots det minskade beroende av ”tvättade” och välstrukturerade data kan neurala nätverk fortfarande dra nytta av noggrant förberedda data. Önskat brus och låg kvalitet på indata kan påverka både slutresultatet och träningsprocessen, eftersom modellen kanske inte lär sig effektivt om indata innehåller önskat brus. Detta kan leda till både dåliga inlärningsresultat och längre träningsprocesser (LeCun et al. 2015).

Prestandamått och modellutvärdering: Både traditionell maskininlärning och neurala nätverk använder liknande prestandamått (eller bedömningsparametrar) för att utvärdera modellens prestanda. Parametrar som noggrannhet, precision, recall, F1-poäng och medelkvadratfel används för att bedöma hur väl en modell presterar på uppgifter som klassificering och regression. Mer om bedömningsparametrar beskrivs i avsnitt 6.5.

Generalisering, överanpassning och regularisering: Båda metoderna står inför utmaningen med överanpassning (*overfitting*), vilket innebär att en modell presterar väldigt bra på träningsdata men dåligt på tidigare osedda data. Överanpassning beror ofta på att vissa enskilda vikter (koefficienter respektive noder) fått för stor betydelse. Det innebär att modellen är tränad ”för bra” på träningsdata och därmed inte kan tillämpas på data som inte ingått i träningsdatasetet. Algoritmen blir helt enkelt inte tillräckligt generellt tillämpbar för att kunna användas på annat än erhållna träningsdata. Genom att sätta vissa vikter till noll (L1-regularisering, eller Lasso) eller att se till att vissa vikter hålls små (L2-regularisering, eller Ridge) minskar risken för att modellen blir för anpassad till träningsdata och gör den därmed mer generellt tillämpbar. En annan vanlig metod är dropout, som används i neurala nätverk, där man slumpmässigt stänger av vissa neuroner som har blivit ”för viktiga”. Detta gör modellen mer robust och bättre på att hantera nya och okända data. Generalisering, eller att göra modellen mer generell, och överanpassning är centrala begrepp som återkommer på många ställen i denna rapport.

Optimeringstekniker: Både traditionella ML-algoritmer och modeller baserade på NN använder optimeringstekniker för att minimera en förlustfunktion. Förlustfunktionen är ett mått på hur bra en modell presterar genom att beräkna skillnaden mellan modellens förutsägelser och de faktiska resultaten. Målet är att justera modellens parametrar för att minimera detta fel. Medan traditionella maskininlärningsmetoder kan använda optimeringar baserade på linjär algebra, tillämpar neurala nätverk gradientbaserade optimeringsmetoder för att uppdatera sina parametrar (Hastie et al. 2009).

En mer detaljerad schematisk jämförelse av traditionell ML och NN kan ses i Bilaga A.

3.2.9 Typer av inlärning för ML-algoritmer

Det finns olika typer av inlärning inom maskininlärning, främst övervakad, oövervakad och förstärkningsinlärning.

Övervakad inlärning är en typ av maskininlärning där modellen tränas med hjälp av ett dataset som innehåller både indata och motsvarande utdata, vilket kallas ”mål” eller ”etiketter” (*labels*). Målet är att lära modellen att göra förutsägelser eller klassificeringar baserat på givna indata.

Ett exempel på övervakad inlärning är identifiering av vattenläckor utifrån historiska data. Träningsdata kan då bestå av tidigare observerade vattenläckor kopplade till information om exempelvis ledningens ålder och material. Modellen tränas för att lära sig känna igen mönster och avvikelser som indikerar en läcka. När modellen är färdigtränad kan den användas för att analysera nya, osedda indata och avgöra sannolikheten för att en läcka ska uppstå. Vanliga algoritmer för övervakad inlärning inkluderar linjär regression, beslutsträd (decision trees) och stödvektormaskiner (support vector machines).

Oövervakad inlärning är en typ av maskininlärning som arbetar med dataset där endast indata är kända, medan mål eller etiketter förblir okända. Denna metod syftar till att låta modellen identifiera mönster och grupper i data utan vägledning från fördefinierade etiketter.

Ett praktiskt exempel på oövervakad inlärning är att identifiera mätfel. Genom att analysera historiska mätdata utan förutbestämda etiketter för vad som är korrekt eller felaktigt kan en oövervakad inlärningsalgoritm upptäcka avvikelser som kan indikera mätfel. Till exempel kan algoritmen klustra mätvärden för att identifiera punkter som avviker kraftigt från normala mönster, vilket kan tyda på felaktiga mätvärden. Vanliga tekniker för oövervakad inlärning inkluderar klustring, såsom k-means, dimensionsreduktion och PCA (Principal Component Analysis).

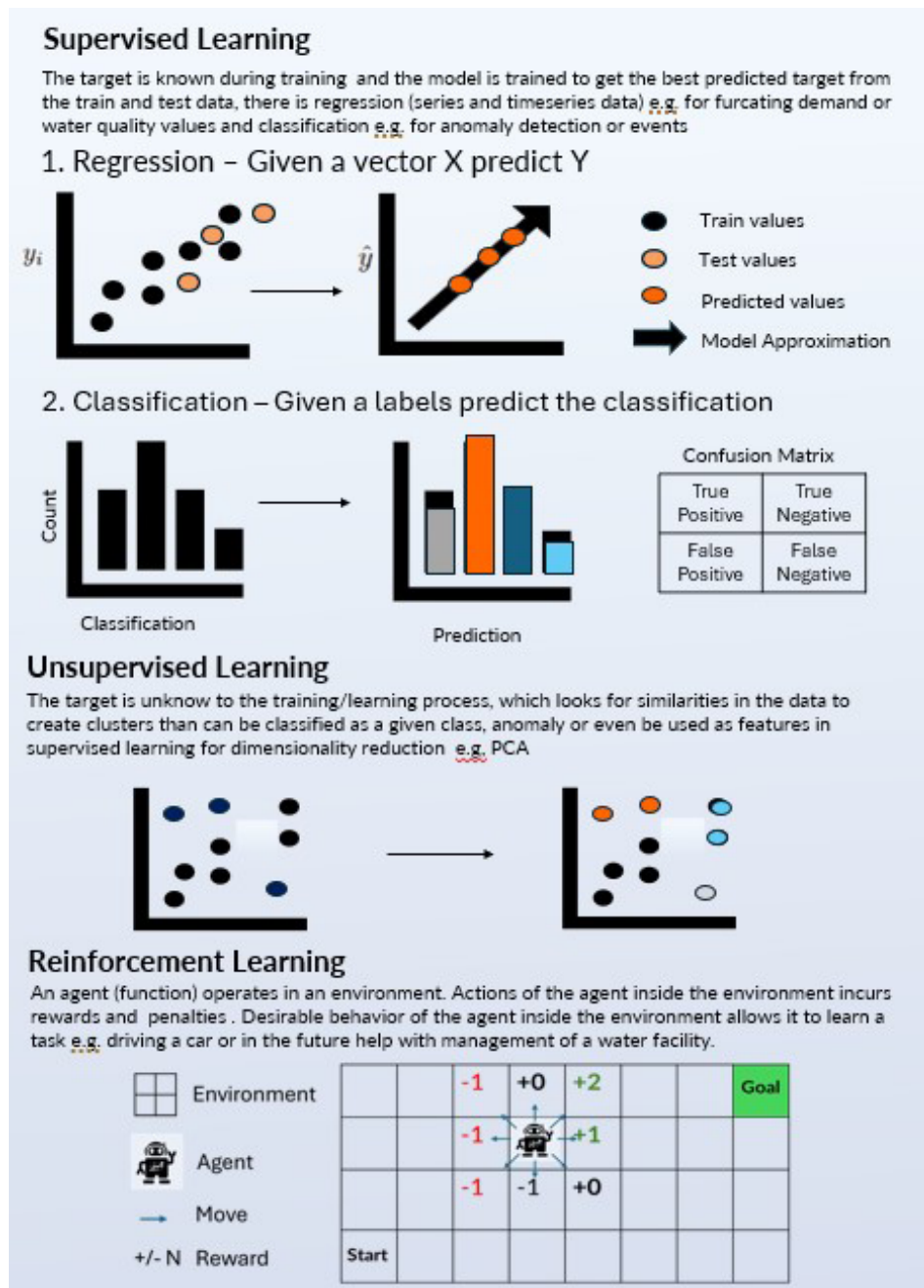
Förstärkningsinlärning (reinforcement learning) är en avancerad form av maskininlärning där en ”agent” lär sig att fatta beslut genom att prova olika handlingar i en viss miljö. Agenten får positiv återkoppling eller negativ återkoppling beroende på utfallet av dess handling. Målet är att maximera den totala belöningen över tid genom att lära sig vilka beslut som leder till bäst resultat.

Ett exempel på tillämpning av förstärkningsinlärning kan vara optimering av processerna på ett reningsverk. Här kan en agent (en automatiserad mjukvara kapabel till att utföra vissa tjänster inom ett system) justera parametrar som flödeshastighet och syrenivåer för att optimera reningsprocessen. Om agenten minskar energiförbrukningen och förbättrar vattenkvaliteten får den positiv återkoppling men om beslut leder till sämre resultat får den negativ återkoppling för att på så vis lära sig mer efter fler iterationer. Förstärkningsalgoritmer används ofta inom robotik, spelutveckling och autonoma system där algoritmer som Q-learning och Deep Q-networks hjälper agenten att förbättra sina val över tid.

Andra typer av inlärning baseras ofta på en eller flera av de tre huvudtyperna som beskrivits ovan och inkluderar:

- *Semiovervakad inlärning*; som använder en kombination av märkta och omärkta data.
- *Självövervakad inlärning*; där etiketter bestäms av modellen.
- *Transferinlärning*; där en förtränad modell (vanligtvis ett djupt neuralt nätverk) anpassas till en liknande men annorlunda applikation.
- *Metainlärning*; som liknar transferinlärning men i grund och botten handlar om att lära sig från lärandeprocessen från andra maskininlärningsprocesser. I praktiken innebär detta att andra maskininlärningsmodeller tränas och att metainlärning använder dessa tidigare erfarenheter för att skapa en generaliserad modell som snabbt kan tillämpas på nya data/nya uppgifter vilket är användbart om mängden tillgängliga data är begränsad.

En översikt över de tre genomgångna typerna av inlärning kan ses i Figur 3.8. Texten i denna figur är på engelska då den tagits fram av Collaborate Water, vilka arbetar på engelska.



Figur 3.8

En översikt över övervakad inlärning, oövervakad inlärning och förstärkningsinlärning.

3.3 Vikten av datas lämplighet, förbehandling och rensning

Processen att förbehandla data styrs av EDA (Exploratory Data Analysis) och kan vara iterativ precis som alla steg inom modellutveckling. Vanligtvis gäller principen ”skräp in-skräp ut” för ML-modeller. De flesta ML-modeller drar nytta av viss förbehandling och rengöring av data eftersom komplexitet och förekomst av brus inom ett dataset kan påverka modellens prestanda (Liu et al. 2004). Med brus avses slumpmässiga fel eller irrelevanta variationer som göra data svårare att tolka och analysera samt att träna upp en ML-algoritm på.

Förutom att data behöver vara i rätt format och struktur är det ofta nödvändigt med mer förbehandling av rådata. Förbehandlingen kan innefatta praktiska aspekter som att ta bort avvikelser och brus, fylla i saknade värden men också mer omfattande arbete som att separera numeriska fördelningar i dess komponenter (exempelvis, normalfördelade och icke-normalfördelade data) och även att skapa nya variabler. Exempelvis genom att märka upp viktiga händelser för att låta algoritmen veta vad som är extra viktig information. Ett exempel på detta kan vara att märka upp ett driftstopp på en sensor för att på så sätt poängtera för algoritmen att det finns en potentiellt avvikande anledning till just det värdet.

När man hanterar dataset som innehåller stora mängder brus är det viktigt att välja en lämplig metod för att förbehandla data, vilket kan variera beroende på datasetet (Xiao et al. 2022). Vanliga metoder för att hantera brus inkluderar att ta bort extremvärden och att jämna ut brus baserat på tidigare och framtida värden (Cheng et al. in 2020). Det är dock viktigt att notera att även om rengöring av data kan eliminera brus och förbättra modellens prestanda riskerar processen att exkludera avvikande träningsdata som kan vara korrekt och innehålla värdefull information. I vissa fall är detta den bakomliggande anledningen till en modell som är överanpassad och inte generellt tillämpbar.

För att en tränad modell i slutändan ska bli användbar är det viktigt att modellen är tränad på både normala och avvikande data. Ofta är det just avvikande händelser som är av extra intresse. Därför är valet av förbehandlingsmetod viktigt.

Olika typer av normalisering tillämpas också ofta på dataset för att skala datamängderna, syftet med skalningen kan variera. När dataset med flera variabler används, hjälper normaliseringen till att anpassa skalorna för varje variabel så att den variabel som har störst numerisk variation inte dominerar träningsprocessen. Skalning kan också minska numerisk instabilitet som kan uppstå med extremvärden. En annan anledning till skalning kan vara beräkningsmässig; ett dataset som är skalat har lägre värden vilket gör beräkningen av förlustfunktionen enklare och hjälper modellen att snabbare konvergera till ett optimalt resultat. En annan metod för att hantera brus i dataset är att använda sig av *wavelet transformation* vilket används för att dela upp signaler och data i olika frekvenskomponenter, vilket gör att man kan fokusera på specifika mönster. Detta bidrar till att minska effekten av mycket avvikande data samtidigt som man bevarar viktig variation. Denna metod är särskilt användbar när det gäller att hantera både små och stora avvikelser, då den kan skilja mellan relevanta signaler och brus. Generellt sätt normaliseras data i ML-modeller och förbättringar ses ofta men det är inte alltid nödvändigt. På senare år har det dessutom tillkommit modeller där normalisering av data är en del av arkitekturen vilket innebär att användaren inte behöver hantera detta. Mer om tillämpad skalning i detta projekt kan läsas i avsnitt 6.3.1.

Mängden data är också viktig. Traditionell maskininlärning kräver ofta mindre datamängder än modeller baserade på neurala nätverk. Detta beror på arkitekturen på de olika modelltyperna det vill säga lager/noder kontra ekvationer.

3.3.1 Feature extraction och feature engineering

Feature extraction är processen att omvandla rådata till en uppsättning meningsfulla och användbara variabler som kan användas i maskininlärningsmodeller. Feature extraction används som begrepp även på svenska men skulle i nödfall kunna översättas till variabel-extraktion. Ett exempel på feature extraction kan vara att beräkna medeltemperaturen per dag från timvisa värden. Målet är att minska datas komplexitet samtidigt som den mest relevanta informationen behålls. Feature extraction kan också involvera mer komplexa metoder som dimensionalitetsreduktion med tekniker som *Principal Component Analysis* för numeriska data och *Linear Discriminant Analysis* för textdata och numeriska data. Båda används ofta för att minska antalet variabler samtidigt som så mycket varians som möjligt bevaras (Jolliffe 2002).

Feature Engineering innebär att skapa nya eller modifiera befintliga variabler för att förbättra prestandan hos maskininlärningsmodeller. Ett exempel är att i ett dataset innehållande energiförbrukning (kWh) och volym (m³) kombinera dessa till energiförbrukning/flöde (kWh/m³). Feature engineering har särskilt stor påverkan på traditionella statistiska maskininlärningsmodeller och grunda neurala nätverk. Däremot har det mindre betydelse för djupa neurala nätverk, där nätverken själva kan lära sig optimala representationer av data under träningsprocessen.

Skillnaden mellan feature engineering och feature extraction är att feature engineering innebär att skapa nya eller justera befintliga variabler för att hjälpa ML-modellen att prestera bättre. Det kräver ofta kreativitet och domänkunskap så väl som kunskap inom data science. Feature extraction handlar däremot om att förenkla data genom att omvandla dem till en mindre, mer hanterbar form som fortfarande behåller den viktigaste informationen. Feature engineering skapar nya variabler, medan feature extraction koncentrerar sig på att komprimera den befintliga informationen.

3.4 Att navigera i komplexiteten

För att förstå möjligheterna med ML är det viktigt att belysa systemkomplexitet. Ett komplext system har många invecklade interaktioner mellan sina komponenter, vilket leder till oförutsägbara beteenden som inte enkelt kan förstås genom att analysera de enskilda delarna. Dessa system karaktäriseras vanligtvis av hög interkonnektivitet, icke-linjäritet och dynamisk interaktion, vilket gör deras beteende svårt att modellera eller kontrollera. En definition ges av Bar-Yam (2003): "Systemkomplexitet är ett mått på antalet komponenter i ett system och komplexiteten i deras interaktioner".

Det finns två kategorier av komplexitet: komplex och verkligt komplex. Komplex innebär att systemet har många komponenter och interaktioner, men det är fortfarande möjligt att modellera. Verkligt komplex innebär att systemet har sådana egenskaper och interaktioner att det trotsar traditionella modelleringsmetoder. Det är viktigt att känna till de subtila skillnaderna när det gäller möjligheten att modellera med tillgängliga data. Denna distinktion är avgörande då den påverkar metoder och förväntningar för att korrekt representera och förutsäga systemets beteende, och vissa system kan förbli bortom fullständig förståelse (Mitchell 2009). Verkligt komplexa system är vanliga inom olika områden, inklusive socio-ekonomiska nätverk, ekosystem, biologiska organismer och teknisk infrastruktur (Bar-Yam 2003). De komplexa egenskaperna uppstår från de dynamiska interaktionerna mellan komponenterna, vilket leder till beteenden som inte kan förutsägas genom att analysera de enskilda delarna (Holland 2014).

Komplexa system

Att modellera komplexa system är förknippat med utmaningar på grund av de många och komplexa interaktionerna mellan komponenter, deras föränderliga natur, samt de stora datakraven för korrekt representation. Några av dessa utmaningar är:

Dimensionalitet: Komplexa system involverar ofta många variabler som interagerar på icke-linjära sätt, vilket leder till datamiljöer med hög dimension som är beräkningsmässigt krävande att modellera (Goodfellow et al. 2016).

Komplexa egenskaper: Komplexa egenskaper, som spontan självorganisation och anpassning, uppstår från interaktionerna mellan komponenter, vilket gör det svårt att förutsäga systembeteendet baserat på individuella element (Page 2011).

Datatillgänglighet och kvalitet: Att få tag på tillräckligt omfattande och högkvalitativa data för att modellera komplexa system kan vara besvärligt. Data kan vara gles, brusig

eller ofullständig, vilket begränsar möjligheten att fånga systemets fulla dynamik (Hastie et al. 2009).

Modellens tolkningsbarhet: När modeller blir mer komplexa för att hantera systemets komplexa natur minskar ofta modellens tolkningsbarhet. Denna brist på transparens kan minska förtroendet för modellens förutsägelser och begränsa deras användbarhet vid beslutsfattande (Lipton 2018).

Verkligt komplexa system

Verkligt komplexa system involverar inte bara flera interagerande komponenter utan uppvisar även egenskaper som trotsar konventionella modelleringsmetoder:

Icke-linjäritet och kaos: Verkligt komplexa system uppvisar ofta icke-linjär dynamik och kaotiskt beteende, där små ändringar i initialvillkoren kan leda till oproportionerligt stora effekter, vilket gör långsiktiga förutsägelser osäkra (Lorenz 1993).

Spontan självorganisation: Spontana fenomen i verkligt komplexa system kan uppstå från okända interaktioner mellan komponenter, vilket leder till självorganisation och spontan bildning av sammanhållna strukturer eller mönster (Kauffman 1995).

Stokasticitet och osäkerhet: Verkligt komplexa system är inneboende stokastiska, påverkade av slumpmässiga fluktuationer och externa störningar som trotsar förutsägelser baserade på deterministiska modeller (Taleb 2010).

Med tanke på de inneboende utmaningarna i att modellera komplexa och verkligt komplexa system är det viktigt att erkänna begränsningarna hos dagens metoder, såsom traditionella statistiska modeller och ML-algoritmer. I vissa fall kan systemets komplexitet överstiga kapaciteten hos tillgängliga data och modelleringsmetoder, vilket gör korrekta förutsägelser eller en fullständig förståelse svåråtkomlig (Gershengfeld 1999). Systemens utvecklande natur, särskilt de som involverar kaotiska element, kräver modeller som kan anpassa sig till kontinuerliga förändringar.

4 Maskininlärning inom vatten- sektorn – ett sammandrag

Både traditionell maskininlärning och neurala nätverksbaserade metoder har framgångsrikt tillämpats på data från vattensektorn. Många tillämpningar har gjorts i avskilda och kontrollerade miljöer, som inte alltid representerar verkliga produktionsmiljöer. Det finns även en stor mängd tillämpningar på verkliga system med varierande komplexitet.

Det är av stor vikt att ta in ny kunskap från framför allt vetenskaplig litteratur. Arbete inom ML är ofta komplext och är i kontinuerlig utveckling. Ofta är också utvärdering av enskilda parametrar eller metoder arbetskrävande varpå mycket arbete kan sparas även genom en mindre litteraturstudie. I avsnitten 4.1-4.3 beskrivs tre publikationer som har fungerat som en mindre litteraturstudie i detta projekt.

Följande fördelar med att använda AI och ML inom dricksvattenbehandling analyseras av Li et al. (2021): AI-teknologier har stora fördelar jämfört med traditionella matematiska modeller och statistiska metoder inom vattenbehandling. AI kan hantera komplexa icke-linjära samband och ge en bättre förståelse av dynamiska processer i vattenbehandling. Detta gör att AI kan övervaka vattenkvalitet, analysera och förut säga vattenkvalitetsparametrar samt förklara hur föroreningar rör sig och omvandlas. Detta innebär att fokus skiftas från att reagera på problem till att proaktivt identifiera risker och optimera behandlingsanläggningar kontinuerligt. AI kan dessutom beräkna vattenkvalitetsparametrar som annars skulle kräva mycket komplicerade och tidskrävande metoder.

AI-teknik kan optimera viktiga delar av vattenbehandlingsprocessen, som processparametrar och materialberedning, till exempel hur membran designas, tillämpas och sköts. Genom denna optimering kan kostnader minskas och effektivitet förbättras inom vattenproduktion (Rall et al. 2020; Peiris et al. 2012).

Tack vare AI:s förmåga till självlärande och självdiagnostik är det möjligt att skapa ett avancerat feedbacksystem och automatiserade styrsystem med hög precision som bygger på AI-teknik. Denna typ av system kan självständigt analysera data, fatta beslut och ge prognoser baserat på inkommande information, vilket minskar arbetsbördan för personalen inom vattenbehandlingen (Bunmahotama et al. 2017). Detta betyder inte att det mänskliga beslutsfattandet förlorar sin betydelse. I stället används informationen från övervakningsutrustning på ett effektivare sätt vilket skapar bättre underlag för beslutsfattande och mer möjlighet till förbättringsarbete. Det är dock viktigt att notera att det fortfarande saknas etablerade säkerhetsstandarder för implementering av AI i känsliga processtyrningssystem. Möjligheterna ser dock mycket ljusare ut för att använda denna typ av beslutsstödsystem ihop med en digital tvilling, i stället för direkt i de känsligaste systemen.

AI-teknik har tidigare setts som en *black box*-metod där man bara ser samband mellan in- och utdata utan att förstå hur AI når sina resultat. Men med hjälp av rätt analysmetoder kan AI-modeller ge oss värdefulla insikter om fysiska och kemiska processer, vilket förbättrar vår förståelse av vattenbehandling (Olden et al. 2002; Park et al. 2019).

I vattensektorn fokuserar en stor del av diskussioner och rapporter (inkluderande denna rapport) på hantering av ”siffror” och kvantitativa data. Men faktum är att minst lika stor vinst finns genom att tillämpa AI och ML inom organisationsutveckling. Inom allt från rapportskrivning, ekonomisk analys, kundservice, resursoptimering och cybersäkerhet finns det enorma möjligheter för AI- och ML-tillämpningar. Här finns mycket att vinna på samverkan mellan organisationer, oavsett om de är direkt involverade i vattensektorn eller ej. Detta ligger dock utanför ramarna för denna rapport.

4.1 Vattenkvalitetsparametrar

Flertalet exempel på hur ML och AI kan tillämpas för att prognosticera vattenkvalitetsparametrar finns tillgängliga i den vetenskapliga litteraturen. Nedan presenteras ett exempel.

Li et al. (2021) har i sin publikation utforskat användningen av AI och maskininlärning för att förbättra dricksvattenbehandlingsprocesser. Studien undersöker hur dessa teknologier kan användas för att analysera och förstå komplexa, icke-linjära samband i vattenbehandling, vilket möjliggör förbättrad processkontroll. Genom att använda tekniker som multipel linjär regression, principalkomponentanalys, artificiella neurala nätverk och *Support Vector Machines*, visar forskningen hur AI och ML kan ge insikter om parametrar som påverkar vattenkvalitet, exempelvis mineralnivåer, cyanobakterier och andra organismer. Studien beskriver också hur AI-modeller kan förutsäga variationer i råvattnets kvalitet och föreslår hur dessa modeller kan bidra till att optimera behandlingsprocesser och minska beroendet av konventionella processkontroller. Övergripande syftar studien till att visa potentialen hos AI för att effektivisera dricksvattenbehandling, minska kostnader och förbättra vattenkvaliteten genom bättre processtyrning och prediktiva analyser.

4.2 Prognos av förbrukning

ML har använts för att prognosticera efterfrågan i många sammanhang, inklusive inom vattenförbrukning. I en översiktsartikel presenterar Niknam et al. (2022) en utvärdering av olika verktyg och tekniker som används för korttidsprognoser av vattenförbrukning, med fokus på varje metods styrkor, svagheter och lämpliga tillämpningar. De konstaterar att traditionella statistiska modeller som (S)ARIMA är fördelaktiga tack vare sin enkelhet, tolkningsbarhet och förmåga att fånga säsongsvariationer, men att de har svårt att hantera icke-linjära och komplexa mönster, vilket gör dem mindre effektiva för dataset med starkt varierad vattenförbrukning.

ML-metoder som artificiella neurala nätverk samt traditionella ML-metoder som Support Vector Machines och Random Forests lyfts fram som populära val av modeller. Artificiella neurala nätverk framhävs för sin förmåga att modellera icke-linjära samband och anpassa sig till komplexa data, särskilt i situationer där vattenförbrukningen är mycket varierande. Support Vector Machines är kända för sin robusthet i hantering av avvikande värden, men kan vara beräkningsmässigt krävande, medan Random Forest erbjuder stark prediktionsförmåga och hanterar analys av flera variabler väl, även om de ofta kräver stora dataset och mycket finjustering av modellparametrar.

Översikten betonar också att många vattenbolag har utmaningar gällande datainsamling och datalagring. I många fall är historiska dataset begränsade eller ofullständiga. Ofta har också historiska data aggregerats om till lägre upplösning vilket innebär att detaljerad information har reducerats till mer övergripande genomsnittsvärden, helt enkelt för att spara lagringsutrymme och därmed pengar. Detta innebär att många organisationer har mindre högupplösta data tillgängliga för ML-projekt än vad som ursprungligen är känt.

Studien framhåller att modellarkitektur bör vara anpassad efter kontexten och varnar för idén om att en enda modell kan passa alla situationer. Studien rekommenderar ett skräddarsytt tillvägagångssätt där modeller väljs baserat på tillgången på data, prognoshorisont och beräkningsresurser samt eventuellt behov av hybrid- och ensemblemetoder. Detta är kombinationer av modeller som fångar både linjära och icke-linjära komponenter vilket kan öka robustheten och noggrannheten. Dessa hybridmodeller kan dock vara mer komplicerade att träna och sätta i produktion på grund av deras komplexitet, behov av noggrann finjustering och krav på beräkningsresurser.

Översikten exemplifierar vad en kompetent data scientist redan känner till men sätter det i kontexten av vattenförbrukning. Dock nämns inte några populära traditionella/hybrida ML-metoder som *XGBoost* och *Prophet*, två modeller som har undersökts i detta projekt.

4.3 Vattennivå i akvifer

Det finns en växande mängd litteratur som tillämpar maskininlärning på akvifersystem. Från att kartlägga grundvattenpotential till att förutsäga vattenkvalitet och prognosticera vattennivåer.

Huang et al. (2024) fokuserar på hur maskininlärningsmodeller kan användas för att bedöma och förutsäga grundvattenkvalitet. Författarna diskuterar vilka framsteg som har gjorts inom området och utmaningar som kvarstår. Studien går igenom olika maskininlärningsmetoder som har använts för att modellera komplexa samband mellan flera variabler som påverkar grundvattenkvalitet exempelvis nederbörd, markegenskaper och mänsklig påverkan. Publikationen betonar vikten av högkvalitativa data och identifierar utmaningar som databegränsningar och svårigheter med att integrera dessa modeller i verkliga applikationer. Dessutom används hybridangreppssätt som kombinerar maskininlärning med traditionella modeller vilket förenar det bästa från geohydrologiskt fysikbaserade modeller och ML.

Khan et al. (2023) analyserar olika modeller som används för att förutsäga grundvattennivåer. Publikationen täcker både konventionella statistiska modeller och moderna ML- och djupinlärningsmetoder såsom Support Vector Machines, neurala nätverk och LSTM (*Long Short-Term Memory*). Studien jämför styrkor och svagheter hos dessa modeller och ger rekommendationer om vilken modell som kan vara bäst lämpad beroende på tillgången på data och de specifika krav som ställs. En stor del av fokus ligger på hybridmodeller som kombinerar traditionella fysikbaserade modeller med maskininlärning för att förbättra prognosernas noggrannhet, särskilt i komplexa och heterogena akviferer. Begränsningar såsom datakvalitet, beräkningskrav och behovet av domänämneskunskap framhålls också.

De nuvarande begränsningarna för dessa angreppssätt är främst brist på data och begränsade beräkningsresurser. Dessutom kräver modellintegrering specialiserad expertis eftersom felaktigt val av modell kan minska hybridmodellernas effektivitet. Det rekommenderas att utveckla standardiserade ramverk och fortsätta utforska djupinlärningsmodeller för grundvattensystem.

5 Maskininlärning – arbetsflöden och metoder

En introduktion till ML och dess tillämpningar inom vattensektorn vore inte komplett utan en diskussion om arbetsflöden och arbetsmetoder. ML-projekt använder ofta en kombination av en iterativ modellstrategi och ett agilt arbetsflöde vilket innebär att man arbetar cykliskt med kontinuerliga utvärderingar. Dessa metoder möjliggör produktion av ML-baserad mjukvara och är avgörande för ett systematiskt arbetssätt som kombinerar både domänkunskap och kunskaper inom datavetenskap.

5.1 Inhämtande av domänkunskap/förståelse för problemet

När ett nytt ML-projekt startas är det viktigt att tydligt förstå varför projektet ska genomföras. Är syftet att hitta nya insikter från data? Att skapa en prognosmodell på kort eller lång sikt? Eller att bygga en algoritm som kan användas i ett existerande system? Målet med projektet ska guida hur arbetet planeras och utförs och bör också utvärderas regelbundet, exempelvis varje månad eller efter viktiga milstolpar. Detta för att säkerställa att målet fortfarande är relevant och uppnås på ett effektivt sätt.

Även om ML kan användas utan att ta hjälp av personer med domänkunskap rekommenderas det inte. Att ta med experter inom ämnet är viktigt för att kunna planera projektet på rätt sätt och för val av teknik. Det är också avgörande för att bedöma resultaten korrekt. I detta projekt var de första mötena med operatörer som arbetar på platsen, geohydrologer och individer med lång historisk domänkunskap avgörande för det fortsatta arbetet med programmering och valet av algoritmstruktur.

5.2 Valet av algoritm

Att identifiera den lämpligaste ML-metoden för ett specifikt problem är oftast en iterativ process baserad på försök och misstag som inleds med några viktiga överväganden.

Typ av problem – val av algoritm: Inledningsvis bör diskussioner föras om vilken typ av problem det är som ska analyseras. Handlar det om att förutsäga kategoriska utfall eller prognosticera kontinuerliga värden? För klassificering används ofta metoder som logistisk regression, beslutsträd och supportvektormaskiner. För regression är tekniker som linjär regression, *ridge regression* och *gradient boosting*-maskiner lämpliga (Hastie et al. 2009) samt klassificeringsalgoritmer som har anpassats till regression eller tidsseriedata, exempelvis XGBoost och Support Vector Regression.

Inledande analys av tillgängliga data: Tillgängliga data bör utvärderas med avseende på mängd, kvalitet och typ (exempelvis numerisk, bild eller text). Stora dataset med komplexa mönster kan gynnas av djupinlärningsmetoder medan mindre dataset ofta fungerar bättre med traditionella ML-algoritmer. Förekomsten av brus eller saknade data kan också påverka valet av förbehandlingsmetoder och modell (Kotsiantis et al. 2006).

Interoperabilitet: Med detta avses möjligheten för olika system, modeller eller komponenter att samverka och utbyta information. Detta är ofta en viktig faktor för att kunna integrera ML-lösningar i existerande arbetsflöden/system och för att undvika

datasilos. Ett exempel på detta är ML-arbete inom detektering av blufffakturor. Där kan en ML-algoritm ha tränats upp i en python-miljö med bibliotek såsom *scikit-learn* och *Tensorflow*. För att kunna integrera detta i ett faktureringsystem byggt på en annan plattform (exempelvis Java) krävs att modellen är interoperabel. Interoperabilitet innebär ofta en kompromiss mellan modellers komplexitet och tolkningsbarhet. Komplexa modeller, som djupa neurala nätverk, kan uppnå hög prestanda men är ofta svåra att tolka. Enklare modeller, såsom beslutsträd eller linjära modeller, är lättare att tolka och erbjuder bättre interoperabilitet men kan ha begränsad förmåga att förutsäga komplexa samband (Molnar 2020).

Dessa tre nyckelöverväganden stärks ytterligare av en god förståelse för modellarkitektur i aktuella eller liknande sammanhang. Ofta finns mycket information om detta tillgängligt i den vetenskapliga litteraturen.

När problemet har ramats in och förståtts, data har analyserats och behovet av interoperabilitet bedömts är det också viktigt att beakta mer praktiska faktorer såsom tillgängliga resurser, kostnader och tidsram. Ofta är det de praktiska faktorerna som begränsar den utforskande delen av ML-projekt. När ovanstående är bedömt är det vanligt att välja ut flera modeller/algoritmer för en initial utvärdering. Ofta är den första utvärderingen med hjälp av tekniker som korsvalidering, parameterbestämning och analys av förbehandlingsmetod inte alltför arbetskrävande. Genom att arbeta systematiskt med exempelvis bra grafiska verktyg och möjlighet att snabbt se bedömningsparametrar är det möjligt att översiktligt bedöma vilka modeller och algoritmer som är värda att gå vidare med.

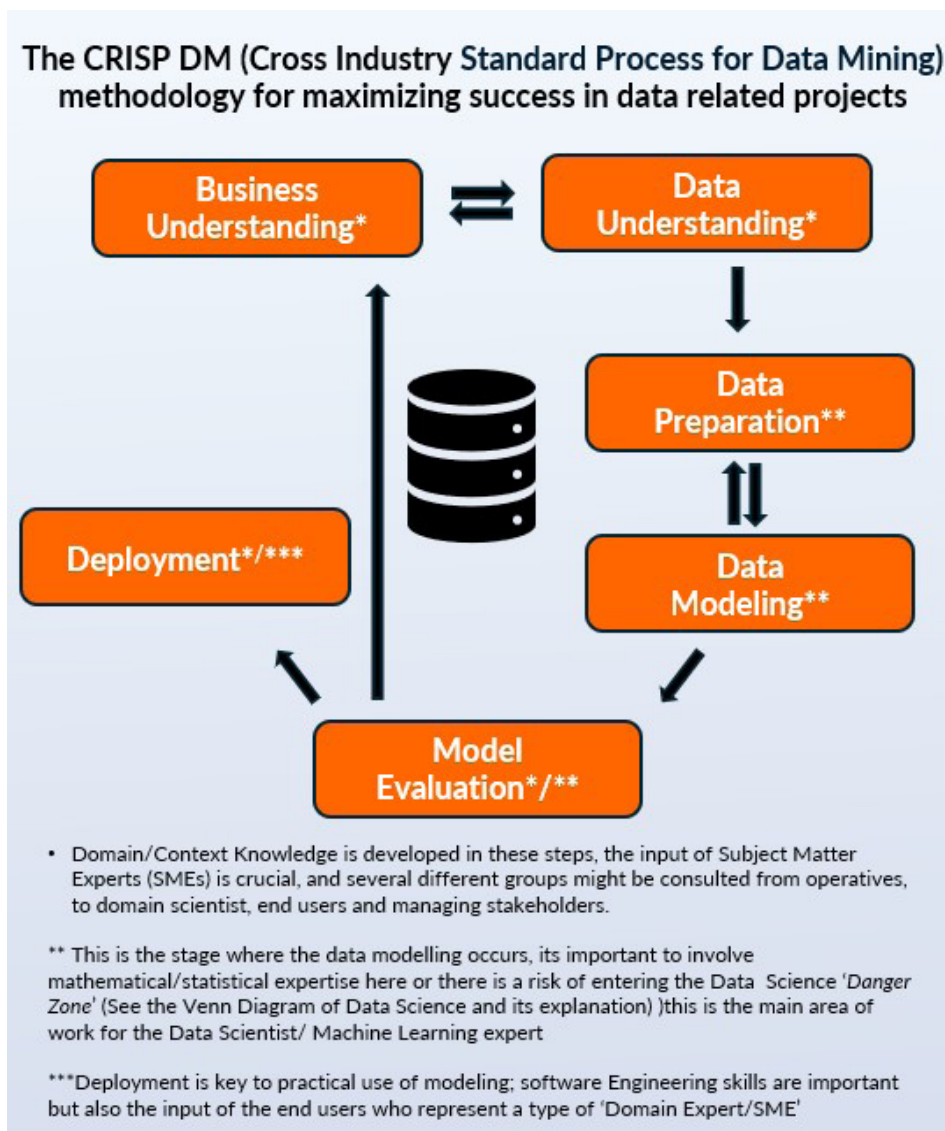
Efter de inledande undersökningarna väljs en modell eller en grupp av modeller för vidareutveckling, exempelvis genom optimering av parametrar eller ytterligare modellträning. Det slutliga modellvalet bör vara det som bäst löser det aktuella problemet. Om en enklare traditionell ML-metod ger bra resultat är det ofta inte ekonomiskt försvarbart att inkludera mindre interoperabla och högkomplexa modeller. Detta kan vara relaterat till både beräkningskostnader och längre utvecklingstid. På samma sätt, om traditionella ML-metoder och grunda neurala nätverk inte fungerar bra men djupa nätverk visar lovande resultat bör beslutet att fortsätta utveckla dessa modeller balanseras mot kostnader, potentiell avkastning på investering och den risk som är involverad.

5.2.1 Iterativa processer

Det finns etablerade ramverk för modellbyggnadsmetoder inom ML/AI ofta baserade på iterativa strategier och med väl definierade steg för praktisk utveckling. Att följa ett ramverk inom ML-projekt är ett bra sätt att säkerställa att rätt steg har tillämpats inom projektet. Det är också ett bra grund för samarbete då det blir enklare att tilldela resurser och få in rätt kompetens i rätt skede av projektet.

En vanlig metodik är CRISP-DM (*Cross Industry Standard Process for Data Mining*) (Shearer 2000) och dess anpassningar. Ett annat exempel är SEMMA (*Sample, Explore, Modify, Model, and Assess*).

CRISP-DM, som utvecklades genom EU-finansiering, är en sexdelad cykel som börjar med affärs- och dataförståelse (genom interaktioner med domänämnesexperter) följt av preliminär dataförberedelse, modellering och utvärdering och avslutas med implementering/leverans av modellen samt erhållna insikter från projektet. Vanligtvis innebär den första cykeln i ett projekt att etablera en preliminär modell som grund för vidare utveckling. En översikt av stegen i CRISP-DM ses i Figur 5.1. Texten i denna figur är på engelska då den tagits fram av Collaborate Water, vilka arbetar på engelska.



Figur 5.1

En översikt över CRISP-DM metoden, en vanlig strategi som används inom större ML-projekt.

5.2.2 Agila metoder

Ramverk såsom CRISP-DM är utvecklade för att kunna integreras med olika etablerade projektledningsmetoder. Agila metoder (Beck 2001) har sitt ursprung i praxis för programvaruutveckling och är ett arbetssätt där projekt delas upp i mindre delar som genomförs i korta iterativa cykler ofta kallade sprintar. Det finns flera olika agila metoder som kan användas såsom *Scrum*, *Kanban* och *Shape-up*. Agila metoder utvecklades som ett svar på bristerna hos traditionella vattenfallsmetoder vilka ofta är baserade på en mer linjär projektledningsstruktur (Turk 2005). Agila metoder lämpar sig väl för hantering av CRISP-DM och andra iterativa datamodeller. Värt att poängtera är dock att agil projektledning är utformade för programvaruutveckling, en process som ibland skiljer sig från arbete inom ML och datorvetenskap.

5.3 Beräkning och arbetsmetodik

Nedan följer en kort redovisning/diskussion om programmeringsspråk, mjukvara och implementering av slutresultat. Även om det i dag finns viss färdig mjukvara tillgänglig för att använda sig av ML är det fortfarande i stor utsträckning nödvändigt att använda programmeringsspråk såsom Python eller R för att skapa, träna och implementera modeller.

På organisationer som tillämpar ML i stor skala finns ofta en roll kallad *Machine Learning Engineer*. Det är en specialist som kombinerar kunskap inom ML och mjukvaruutveckling med målet att utveckla ML-baserade system som kan integreras och fungera effektivt i produktionsmiljöer. En ML-engineer arbetar ofta med att optimera modeller, förbereda och bearbeta data samt att utveckla och underhålla infrastrukturer för modellens driftsättning. Nedan beskrivs några arbetsmoment som en ML-engineer arbetar med.

5.3.1 Beräkningskraft

Beräkningskraft är en viktig aspekt att beakta. Det var inte bara bristen på beräkningskraft som höll tillbaka den tidiga utvecklingen av ML/AI, det är även något som behövs tas hänsyn till i dag. Speciellt avseende tillgänglig tid och budget för ett projekt. Även om en modell är teoretiskt möjlig kan träning av den ta extremt lång tid och medföra betydande kostnader.

ML-modeller innefattar ofta algoritmer med varierande beräkningskrav. Detta beskrivs ofta med så kallad Big O-notation som är ett sätt att beskriva hur effektiv en algoritm är genom att kvantifiera mängden beräkningsresurser som krävs i förhållande till storleken på indata. Det ger en indikation på hur snabbt beräkningskraven ökar när datamängden växer. Exempelvis innebär en algoritm med tidskomplexiteten $O(n^2)$ att beräkningsresurserna ökar kvadratisk med storleken på indata. Detta betyder att om datamängden fördubblas, fyrdubblas beräkningskraven vilket snabbt kan bli opraktiskt när datamängden växer.

Vissa problem inom ML klassificeras som NP-hårda (*Nondeterministic Polynomial time*). Detta innebär att de kräver extremt mycket beräkningskraft och att det inte finns någon känd effektiv lösning för stora problem. Att hitta en exakt lösning för dessa problem kan ta orimligt lång tid vilket gör att man ofta använder förenklade metoder som approximationer (exempelvis heuristiska tillvägagångssätt), Monte Carlo-simuleringar eller gradientbaserade optimeringsmetoder. I dessa sammanhang är beräkningskraft väsentlig för att hantera komplexiteten, möjliggöra snabbare approximationer och ge praktiska lösningar. Avancerad hårdvara som GPU:er (Graphical Processing Unit) kan utföra många operationer parallellt och därmed minska tiden som krävs för att träna och optimera modeller. Större beräkningskraft hjälper till att hantera både den teoretiska komplexiteten hos algoritmer och de praktiska utmaningarna med att bearbeta storskaliga data.

Tidiga ML-arbeten var begränsade av CPU:er (*Central Processing Units*) och RAM (*Random Access Memory*). CPU:er är datorns huvudsakliga processorer som ansvarar för att utföra de flesta beräkningarna men de är inte optimerade för att hantera många beräkningar samtidigt. I dag används fortfarande CPU:er men det har också tillkommit GPU:er (*Graphics Processing Units*) och TPU:er (*Tensor Processing Units*) som gör beräkningar mer effektiva. GPU:er utvecklades ursprungligen för att hantera höga grafikkrav inom spelindustrin men de är mycket snabbare än CPU:er när det gäller att bearbeta stora mängder data parallellt. NVIDIA har varit en ledande aktör inom utvecklingen av GPU-teknik. TPU:er utvecklades av Google specifikt för att hantera tensorberäkningar (dvs. 3D-arrayer) och används främst för att accelerera maskininlärningsuppgifter i TensorFlow. Sammanfattningsvis är beräkningskraft betydande för framgång inom omfattande ML-projekt. Genom att använda avancerad hårdvara som CPU:er, GPU:er och TPU:er kan vi hantera den komplexitet och de stora datamängder som många ML-problem innebär vilket gör det möjligt att hitta praktiska och effektiva lösningar.

5.3.2 Användning av moln inom ML

Molntjänster gör det möjligt att effektivt lagra, hantera och bearbeta stora datamängder vilket är ett kritiskt behov för dem som påbörjar sin resa inom datavetenskap. Genom

att använda molninfrastruktur ges tillgång till skalbara beräkningsresurser vilket gör det enklare att integrera och analysera stora datamängder. Denna flexibilitet accelererar databehandlingen och möjliggör mer sofistikerade prediktiva analyser (Hashem et al. 2015).

Molnplattformar erbjuder också säkra miljöer för datadelning och samarbete, vilket stödjer projekt mellan olika aktörer och ökar smidigheten i datadrivna initiativ. Detta minskar initiala IT-kostnader och möjliggör snabba implementeringar av ML-modeller. Innan molntjänster användes var organisationer ofta beroende av lokal infrastruktur som i sin tur var beroende av betydande kapitalinvesteringar i hårdvara, mjukvara och licenser. Denna uppsättning begränsade förmågan att snabbt och effektivt skala upp verksamheten vilket ledde till begränsningar och ökade säkerhetsrisker.

I detta projekt har arbete utförts i NSVA:s privata tenant i *Microsoft Azure*. Detta innebär att en molnlösning har använts som ligger inom NSVA:s IT-miljö.

5.3.3 Maskininlärningsspråk

När vi pratar om programmeringsspråk för ML avses oftast högnivåspråk (*high level languages*) vilket är programmeringsspråk som är utformade för att vara lätta att läsa och förstå för människor utan att behöva ta hänsyn till detaljer på maskinnivå.

Flera olika programmeringsspråk används inom maskininlärning, exempelvis Python, R, Matlab, Julia, Java, Scala och Swift. Python och R är de mest populära. Akademien föredrar ofta R eftersom det är särskilt utvecklat för statistisk analys. Python används på grund av dess mångsidighet och lämplighet för att bygga applikationer. Båda språken är öppen källkod vilket innebär att vem som helst, var som helst i världen, fritt kan använda och bidra till utvecklingen av dessa språk. Detta är en viktig del av maskininlärning och data science eftersom den öppna källkodsaspekten har kunnat samla upp den kollektiva utvecklingsinsatsen.

Ett Python-bibliotek är en samling färdiga funktioner och moduler som kan användas för att underlätta programmering i Python. Biblioteket kan innehålla kod för specifika uppgifter exempelvis för att hantera data, göra matematiska beräkningar eller bygga ML-modeller. Genom att använda bibliotek behöver programmerare inte skriva all kod från grunden utan kan i stället dra nytta av redan existerande lösningar som utvecklats av andra. Utövare kan använda bibliotek som utvecklats av open source-gemenskapen för att bygga, träna, utvärdera och implementera ML-modeller. Vanliga Python-bibliotek för ML inkluderar: NumPy, som används för snabb och effektiv matrismatematik; Pandas, för att hantera och manipulera data i tabellform (det kan ses som Pythons motsvarighet till Excel); Scikit-Learn, för statistisk analys och traditionella ML-metoder; samt TensorFlow och PyTorch, som är särskilt populära för att utveckla djupa neurala nätverk. Dessa bibliotek och programmeringsspråk utvecklas ständigt. Inte bara genom nya funktioner, utan även genom ändringar i underliggande kod vilket håller biblioteken ajour med ny utveckling.

Bibliotek och de högnivåspråk som används har ofta beroenden till andra bibliotek. Detta skapar en dynamisk miljö som ständigt driver utvecklingen framåt men det innebär också att ändringar i beroenden måste hanteras aktivt i ML-projekt. Utan proaktiv hantering kan modeller snabbt bli inaktuella, bugga eller till och med bli oanvändbara.

5.3.4 MLOps

MLOps (*Machine Learning Operations*) är ett ramverk som kombinerar metoder och verktyg för att hantera och effektivisera arbetsflödet inom maskininlärningsprojekt (ML-projekt). Det bygger på principerna från DevOps (*Development Operations*) och är särskilt inriktat på att operationalisera ML-modeller från utveckling till implementering och drift vilket innebär att göra dem redo för praktisk användning i produktionsmiljöer.

MLOps blir relevant och viktigt när ML-modeller ska tas i bruk. Ett vanligt scenario är att en organisation har utvecklat ett antal ML-algoritmer och nu vill sätta dessa i

produktion i större skala så att nytta kan erhållas på daglig basis. För att göra detta krävs systematiskt arbete och övervakning av modellerna och här kan MLOps användas som grund.

Rent praktiskt används MLOps för att automatisera uppgifter som testning, driftsättning och övervakning av ML-modeller samt uppdatera och hantera kodbasen och beroenden. Att kunna analysera resultat och utvärderingsparametrar i realtid genom grafer och visualiseringar är också viktigt för att kunna fatta snabba beslut och förbättra modeller. Modellövervakning i produktionsmiljöer innebär dessutom utmaningar som konceptförändringar (*concept drift*) och dataförändringar vilka kan påverka modellens prestanda och kräva kontinuerlig övervakning och uppdatering.

MLOps är viktigt för organisationer som vill optimera sina ML-arbetsflöden. Ofta går det snabbt att generera många olika modeller och resultat och genom att använda MLOps minskar risken för förvirring när många tester, körningar och algoritmer ska utvärderas och förfinas.

En grundläggande funktion inom ML-arbete är möjligheten att kopiera och återanvända användbar kod. En av de största fördelarna med MLOps är dess roll i att underlätta skapandet av återanvändbara modulära kodbasen som kan användas i olika projekt. MLOps bygger på konventioner, standardisering och modularitet vilka är avgörande för att skapa kodbasen som kan återanvändas i olika sammanhang. Standardisering säkerställer att samma kodpraxis, versionskontrollsystem och implementeringspipelines används konsekvent vilket gör det enklare att anpassa befintlig kod till nya sammanhang.

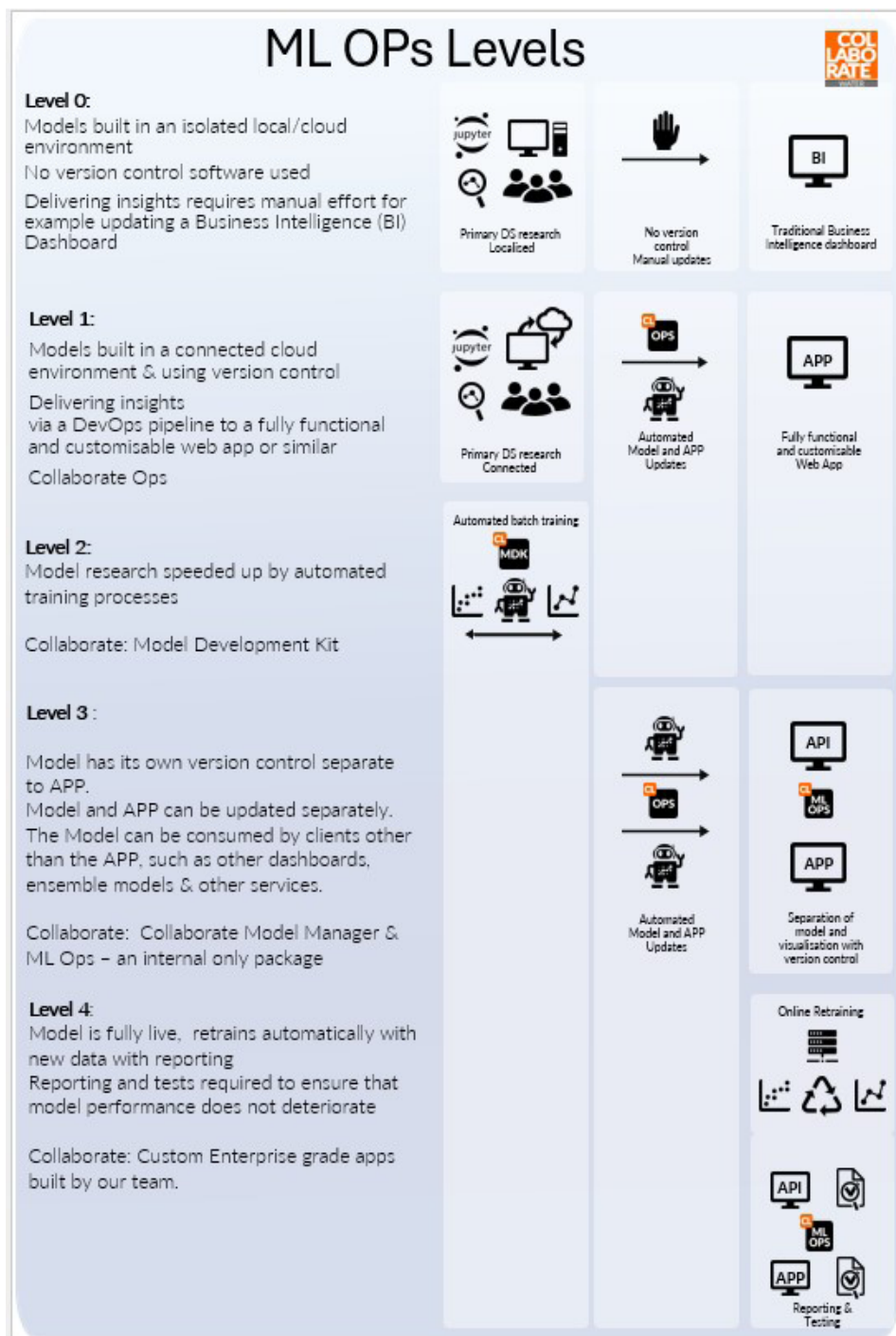
Automatisering och effektivitet är centrala delar av MLOps och syftar till att förenkla repetitiva uppgifter. Genom att automatisera processer minskar MLOps den manuella insatsen som krävs för att utveckla och implementera ML-modeller vilket gör att utvecklare i stället kan fokusera på mer komplexa och innovativa uppgifter (Sculley et al. 2015). Dessutom minskar risken för mänskliga fel genom att återanvända kvalitetsgranskat arbete. Automatisering spelar också en viktig roll för att säkerställa att processer kan återanvändas i olika projekt med minimala ändringar vilket ytterligare förbättrar effektiviteten i ML-arbetsflöden.

Återanvändningen av automatiserade arbetsflöden leder till betydande kostnadsbesparingar eftersom det minskar behovet av omfattande utvecklingsresurser och sänker underhållskostnader. Underhållskostnader uppstår delvis på grund av den öppna källkodsstrukturen i mjukvaruprojekt vilket innebär att källkod och bibliotek är i ständig förändring och kräver aktiv hantering. Ofta läses kodbasen över en viss tid men för ML-system som är i produktion och har lång livstid riskerar alltför långa läsningar att bibliotek och kod missar viktiga uppdateringar både gällande funktionalitet och säkerhet.

Återanvändning av kod är en väletablerad praxis inom mjukvaruutveckling vilket är viktigt för att reducera utvecklingstid och säkerställa mjukvarans kvalitet (Krueger 1992). I MLOps-sammanhang blir kodåteranvändning ännu mer värdefull eftersom det gör det möjligt för organisationer att utnyttja befintliga ML-modeller, datapipelines och implementeringsskript i flera projekt. Detta påskyndar inte bara utvecklingsprocessen utan säkerställer också att modeller och arbetsflöden redan har validerats och optimerats vilket minskar risken för fel vid nya implementationer (Amershi et al. 2019).

Den standardiserade och dokumenterade strukturen i MLOps underlättar dessutom kunskapsöverföring vilket säkerställer att bästa praxis delas och replikeras vilket är avgörande för att skala upp ML-användning i en organisation.

Generellt anses det att det finns fem nivåer av MLOps som börjar med nivå 0. Dessa nivåer visas i Figur 5.2. Texten i denna figur är på engelska då den tagits fram av Collaborate Water, vilka arbetar på engelska.



Figur 5.2

Olika nivåer inom MLOps. Principerna är hämtade från DevOps och kan vara till stor nytta för att bestämma omfattning, tidskrav och budget för ett ML-projekt.

6 Använd verktygslåda

6.1 Översikt

Nedan beskrivs verktyg för behandling och analys av data, tillämpade i avsnitt 7.1. Därefter beskrivs de algoritmer som tillämpats i projektet samt mått för utvärdering av modellresultat, tillämpade i avsnitt 7.2. I resultatdelen förekommer användandet av andra verktyg men dessa förklaras bäst i relation till tolkning av data och lämnas därmed till kapitel 7.

Valet av algoritmer inkluderade inledningsvis ett brett spektrum av algoritmer bestämda utifrån litteraturstudier och tidigare erfarenhet. För att komma fram till bästa möjliga val av algoritm, utifrån det övergripande syftet med projektet, tillämpas CRISP-DM-metodiken beskriven i avsnitt 5.2.1

6.2 Utforskande dataanalys - EDA

Utforskande dataanalys (EDA) är viktig för att samla in information och skapa förståelse för tillgängliga data. Detta bidrar till att en bättre modell kan väljas, att lämplig infrastruktur används och i slutändan få bättre verktyg för utvärdering av resultatet. Ett viktigt syfte med EDA är att förstå de numeriska fördelningarna i data. En numerisk fördelning visar hur värden i ett dataset är fördelade eller arrangerade. Den beskriver intervall av värden och hur ofta varje värde eller intervall förekommer. Numeriska fördelningar bidrar till att förstå formen och spridningen på data vilket kan avslöja mönster eller insikter och bidra till en mer välfungerande modell i slutändan.

Ett dataset kan till exempel visa en klassisk normalfördelning (en jämn spridning av värden kring en central punkt) men vissa datapunkter kan vara avvikande i förhållande till denna fördelning. Orsaken till dessa avvikelser kan vara verklig eller falsk. Oavsett ursprung är det viktigt att identifiera sådana avvikelser, förstå grundorsaken och sedan besluta om en lämplig hantering. Det är viktigt att hantera avvikelser eftersom de kan påverka modellens resultat kraftigt. Särskilt i statistiska modeller, traditionella maskin-inlärningsmodeller och grunda neurala nätverk, som har svårare att hantera avvikelser om de inte är ”robusta” – ett begrepp som beskriver modeller som är mindre benägna att påverkas av avvikande data. Förutom avvikelser kan ett dataset innehålla flera olika fördelningar varpå det kan då vara fördelaktigt att dela upp data i dessa separata fördelningar för att bättre förstå varje del.

EDA utförs med hjälp av statistiska verktyg för att identifiera datasetets allmänna fördelning och eventuella avvikelser. Tolkningen av EDA bör också innefatta domänkunskap som kan hjälpa till att förklara orsaken till avvikelser och sätta dem i kontext. Exempelvis observerades en avvikande ökning i mängden levererat vatten som inte berodde på väderförhållande. Genom att visa detta för driftpersonalen gavs förklaringen att en större läcka hade uppstått varpå denna data togs bort ur träningsdatasetet då den inte var representativ för normala förhållanden. I detta projekt bistod driftpersonalen med viktiga insikter och förklaringar till avvikande data.

När avvikande data identifierats och hanterats förbättras generellt modellens övergripande prestanda. Det är viktigt att komma ihåg att avvikelser ibland är betydelsefulla. Exempelvis är det i detta projekt viktigt att inkludera tidsperioder med ovanligt hög vattenförbrukning. Används endast gängse matematiska modeller för städning av data skulle många av dessa ”toppar” ha exkluderats från träningsdata.

Högupplösta data, ner till minutnivå, var tillgängligt från den 24 januari 2022 fram till och med den 6 januari 2024. Efter diskussion med driftpersonal på Örbyverket togs

beslutet att aggregera data till timvärden och dygnsvärden då beslut inom driften, såsom beställning av mer vatten eller pumpstyrning, ofta baseras på data som visar på mer än ett dygns värde.

För att skapa en prognosmodell för dricksvattenförbrukning används levererat vatten ut från Örbyfältet. Mängden levererat vatten är alltså att likställa med förbrukning av vatten för de områden som Örbyfältet försörjer vilket är grunden för arbete med prognosmodellen.

6.3 Förbehandling av data

6.3.1 Skalning av data

Att skala data innebär att omvandla data så att den numeriska spridningen hos variabler blir mer lätthanterad, vilket är vanlig praxis inom både statistisk modellering och ML. Det finns två huvudsakliga skäl till att skala data. För det första kan det förbättra modellens prestanda för algoritmer som är känsliga för storleken på variabler, särskilt i multivariata (flera variabler) dataset där olika variabler har olika enheter eller skalor. Genom att skala variabler till jämförbara nivåer säkerställer man att ingen enskild variabel dominerar modellens inlärningsprocess, vilket kan leda till en mer balanserad och effektiv modellträning. För det andra kan skalning minska beräkningstiden genom att påskynda konvergensen i maskininläring och underlätta beräkningar med mindre tal. Skalning bör göras separat på träningsdata och testdata för att inte riskera dataläckage. Även om skalning ofta är fördelaktigt för många algoritmer och dataset är det inte alltid nödvändigt och leder inte alltid till bättre resultat. Exempelvis kan vissa algoritmer hantera variabler med olika skalor naturligt, vilket gör skalning överflödig. Två vanliga exempel för skalning av data är min/max- och standardskalning.

Min/max-skalning innebär att transformera variabler så att de faller inom ett specifikt intervall, vanligtvis mellan 0 och 1. Detta görs med formeln:

$$X_{skalad} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Där:

- X är värdet på den ursprungliga variabeln
- X_{min} är det minsta värdet i datasetet
- X_{max} är det största värdet

Standardskalning innebär att transformera variabler så att de får ett medelvärde på 0 och en standardavvikelse på 1. Detta görs med formeln:

$$X_{skalad} = \frac{X - \mu}{\sigma}$$

Där:

- X är värdet på den ursprungliga variabeln
- μ är medelvärdet
- σ är standardavvikelsen

Skillnaden mellan min/max-skalning och standardskalning ligger främst i hur de förändrar variablerna. Min/max-skalning sätter variabler inom ett specifikt intervall, oftast mellan 0 och 1, vilket är användbart när alla värden behöver vara positiva och ha samma intervall. Standardskalning omvandlar variabler så att de får ett medelvärde på 0 och en standardavvikelse på 1, vilket kan hjälpa data att närma sig en normalfördelning om

de redan har vissa normalfördelningsliknande egenskaper. Min/max-skalning ändrar endast skalan och påverkar inte distributionens form, medan standardskalning är användbar för att standardisera data och underlätta modellens inlärning.

6.3.2 Identifiering av extremvärden

Inom den utforskande dataanalysen (EDA) används metoder för att analysera extremvärden i dataset. Z-poäng (eller Z-score i internationella sammanhang) är ett mått som anger hur många standardavvikelser en datapunkt ligger från medelvärdet i en dataset. Formeln för Z-poäng är:

$$Z = \frac{(X - \mu)}{\sigma}$$

Där:

- X är datapunktens värde
- μ är medelvärdet
- σ är standardavvikelsen

Vanligtvis betraktas datapunkter med ett Z-värde större än 3 (eller mindre än -3) som extremvärden, eftersom de ligger långt från majoriteten av data. Val av tröskelvärde görs utifrån tillämpningen och kan variera mellan tillämpningar. Z-poäng är bra när data är normalfördelade men är mindre effektivt om data inte följer en normalfördelning.

IQR (InterQuartile Range) är ett mått på spridningen i mitten av datasetet och definieras som skillnaden mellan den tredje kvartilen (Q_3) och den första kvartilen (Q_1):

$$IQR = Q_3 - Q_1$$

För att identifiera övre och undre gränsvärden för avvikelser används vanligen:

Nedre gräns: $Q_1 - 1.5 \times IQR$

Övre gräns: $Q_3 + 1.5 \times IQR$

Datapunkter som ligger utanför dessa gränser anses vara extremvärden. IQR är effektivt för att identifiera extremvärden i data som inte är normalfördelade och är generellt mindre känsligt för sneda fördelningar jämfört med Z-poäng, men IQR-baserade metoder kan ibland missa avvikelser om data innehåller många extremvärden eller är alltför oregelbundna.

Z-score används primärt för att identifiera extremvärden som baseras på antalet standardavvikelser från medelvärdet, vilket fungerar bäst för normalfördelade data. IQR används för att identifiera extremvärden genom att mäta spridningen i datasetets mitt och fungerar bättre när data inte förutsätts vara normalfördelade.

6.4 Beskrivning av utvärderade algoritmer

Vi testade huvudsakligen ett urval av ML-modeller samt statistiska modeller som baslinje, ofta benämnt som naiva modeller. Naiva modeller används ofta i början av en utredning kring ML som en baslinjesmodell. Syftet med en baslinjesmodell är att skapa en referensnivå som de mer avancerade modellerna ska överträffa.

Naiva modeller gör prognoser på ett enkelt sätt genom att använda statistiska metoder. Exempelvis kan prognoser göras genom att använda det senast observerade värdet, ett medelvärde eller en median. I denna utredning beaktades dessa tre naiva modeller och användes som baslinjesmodeller för jämförelse mot skapade ML-modeller. Nedan följer en kort beskrivning av de ML-algoritmer som har utvärderats under projektets

gång. En mer fullständig matematisk beskrivning av varje algoritm finns redovisat i bilagorna.

För de delar som beskriver neurala nätverk (LSTM, GRU, CNN) behövs en förståelse för begrepp och termer som används för att beskriva neurala nätverk. Vid behov rekommenderas därmed att avsnitt 3.2.7 används som referensmaterial.

6.4.1 (S)ARIMAX

Seasonal, Auto Regressive, Differencing, Moving Average, eXogenous ((S)ARIMAX) är en statistisk modell (och alltså inte rent tekniskt en ML-modell) som består av flera komponenter som tar hänsyn till både säsongsmässiga variationer (S) och exogena variabler (X) med fokus på att förutsäga framtida värden genom att upptäcka mönster som säsongsvariationer och trender. Även om det är en traditionell statistisk modell kan den med auto-ARIMA-funktionalitet anpassa sig och prestera på ett sätt som liknar en maskininlärningsmodell. Skillnaden mot en mer renodlad ML-modell är att (S)ARIMAX söker parametrar genom att testa olika varianter, medan mer renodlade ML-modeller byggs upp iterativt för att förbättra sin förmåga att prognosticera.

Detta görs genom att använda statistiska kriterier (AIC/BIC) för att jämföra olika modeller och välja den som passar datasetet bäst. Det är modellens förmåga att kombinera säsongsmässiga komponenter och exogena variabler, samt att automatiskt anpassa sig till datasetet med hjälp av auto-ARIMA, som gör den till en enkel och kraftfull metod för prognoser inom många tillämpningsområden. I Tabell 6.1 ses en översikt över några fördelar och nackdelar med S(ARIMAX).

Tabell 6.1

Fördelar och nackdelar med S(ARIMAX).

Fördelar	Nackdelar
Bra för tidsserier med säsongsmönster och exogena variabler	Begränsad kapacitet vid komplexa, icke-linjära mönster
Enkel att tolka	Kräver en bedömning av datasetets stationaritet
Minska manuellt arbete med auto-ARIMA	Presterar sämre än moderna ML-metoder vid stora och komplexa dataset

En mer komplett matematisk beskrivning av (S)ARIMAX kan ses i Bilaga B.

6.4.2 Prophet

Prophet är ett verktyg för tidsserieprognoser, utvecklat och släppt av Facebook (META). Det är en hybridmodell som kombinerar statistisk modellering och maskininlärningsprinciper för att hantera komplexa prognosuppgifter. Prophet kan jämföras med moderna ML-metoder och är utformat för att vara användarvänligt för både specialister och icke-specialister, vilket är anledningen till att den inkluderades i detta projekt. Prophet är inte baserat på neurala nätverk vilket innebär att modellen inte lär sig genom iterativ förbättring, utan följer fördefinierade komponenter för att skapa prognoser.

Prophet fungerar genom att dela upp en tidsserie i olika komponenter, som trend, säsongsmönster och helgdagseffekter. Det tydliga fokuset på säsongsmässig variation gör att Prophet är bra på att hantera data som kan uppvisa stabila mönster. Det innebär också att modellen kan vara sämre på att fånga plötsliga eller oförutsägbara förändringar. Modellen kan vara antingen additiv (komponenter adderas) eller multiplikativ (komponenter multipliceras), beroende på datas karaktär. Denna uppdelning gör att Prophet kan identifiera och modellera komplexa mönster i data, vilket gör den till ett kraftfullt verktyg för prognoser. I Tabell 6.2 ses en översikt över några fördelar och nackdelar med Prophet.

Fördelar	Nackdelar
Enkel att använda och kräver inte djupgående kunskap	Kan vara mindre exakt än avancerade ML-metoder
Flexibel för säsongsmässiga och icke-säsongsmässiga mönster	Antaganden om additivitet/multiplikativitet kan begränsa effektiviteten
Kan inkludera exogena variabler som helgdagar	Kräver manuellt justering av parametrar

En mer komplett matematisk beskrivning av Prophet kan ses i Bilaga B.

Tabell 6.2

Fördelar och nackdelar med Prophet.

6.4.3 XGBoost

Extreme Gradient Boosting (XGBoost) är en kraftfull maskininlärningsalgoritm som är känd för sin effektivitet inom prediktiv modellering. Algoritmen bygger på en ensemblemetod där flera beslutsträd byggs sekventiellt, och varje nytt träd försöker rätta till felen från de tidigare träden. Denna boosting-teknik ökar modellens noggrannhet och robusthet genom att kombinera flera svaga modeller (enkla beslutsträd som var för sig har låg prediktiv förmåga) till en stark prediktor. XGBoost är särskilt populär för sin höga prestanda, skalbarhet och förmåga att hantera strukturerade data (organiserad i rader och kolumner, som i databastabeller) i uppgifter som klassificering och regression. XGBoost inkluderar viktiga funktioner som regularisering, tidig stoppning (för att undvika överanpassning) och effektiv beräkning, vilket gör den användbar inom både maskininläringstävlingar, såsom Kaggle-tävlingar, och praktiska tillämpningar. I Tabell 6.3 ses en översikt över några fördelar och nackdelar med XGBoost.

Tabell 6.3

Fördelar och nackdelar med XGBoost.

Fördelar	Nackdelar
Hög prestanda och snabb beräkning	Kan vara resurskrävande, särskilt vid stora datamängder
Kan hantera olika typer av data och uppgifter (klassificering och regression)	Kräver mycket finjustering av hyperparametrar för bästa resultat
Regularisering och tidig stoppning minskar risken för överanpassning	Mindre lämplig för ostrukturerade data som bilder eller text

En mer komplett matematisk beskrivning av XGBoost kan ses i Bilaga B.

6.4.4 SVR

Support Vector Regression (SVR) är en typ av stödvektormaskin som har anpassats för regressionsproblem. Detta innebär att SVR används för att förutsäga kontinuerliga utfall, exempelvis tidsseriedata. SVR bygger vidare på principerna från *Support Vector Machines* (SVM) och används för att förutsäga kontinuerliga utfall istället för klassificeringsetiketter. Kernel-metoder används i SVR för att kunna hantera icke-linjära samband genom att transformera data till högre dimensioner, vilket gör det möjligt att hitta, särskilja och prognosticera olika komplexa samband inom ett dataset, även när mönstren är komplexa. Tack vare användningen av kernel-metoder är SVR en regressionsmetod som är bra på att hantera data med icke-linjära samband.

Målet med SVR är att hitta en funktion som avviker från de faktiska observerade värdena inom en toleransmarginal. Samtidigt ska modellens komplexitet minimeras för att skapa en generell modell som kan hantera nya tidigare osedda data och därmed minimera risken för överanpassning. I Tabell 6.4 ses en översikt över några fördelar och nackdelar med SVR.

Fördelar	Nackdelar
Kan hantera icke-linjära samband genom användning av kernel-metoder	Kan vara beräkningsmässigt kostsam, särskilt för stora dataset
Flexibel modell som fungerar bra för både små och stora dataset	Kräver mycket justering av hyperparametrar för att optimera prestandan
Bra på att balansera komplexitet och generalisering för att undvika överanpassning	Presterar sämre än vissa andra modeller, som neurala nätverk vid mycket stora och komplexa dataset

En mer komplett matematisk beskrivning av SVR kan ses i Bilaga B.

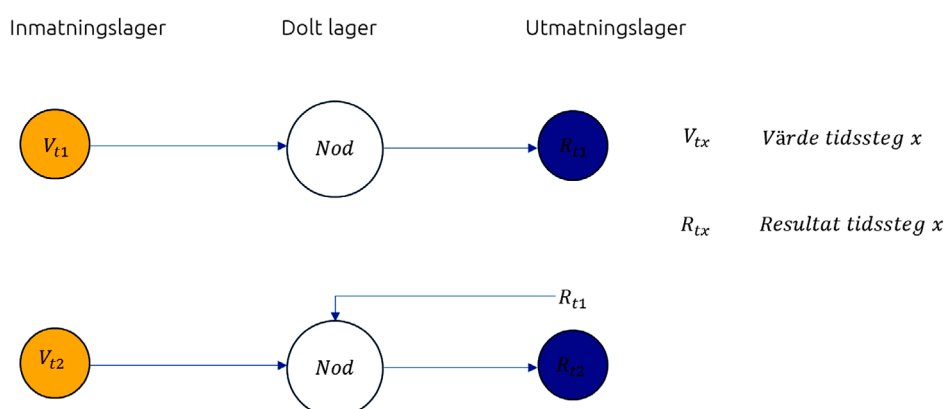
Tabell 6.4

Fördelar och nackdelar med SVR.

6.4.5 LSTM

Long Short-Term Memory (LSTM) är en typ av neuralt nätverk som används för att modellera och förstå sekventiella data, såsom text eller tidsserier. Det är utformat för att hantera långsiktiga beroenden i data. LSTM är en specialiserad typ av återkopplande neuralt nätverk (*Recurrent Neural Network, RNN*).

RNN är utformat för att komma ihåg och känna igen mönster i sekventiella data genom att använda återkoppling från tidigare data. I ett vanligt neuralt nätverk flödar information endast i en riktning, från indata-lager till utdatalager, vilket innebär att dessa nätverk inte har någon mekanism för att bevara information om tidigare steg. Detta gör dem mindre lämpade för att hantera sekventiella data där sammanhang är viktigt. RNN har en återkopplingsmekanism som skickar tillbaka information till föregående noder så att tidigare behandlade data kan bli en viktig del av varje ny punkt i datasetet, vilket är illustrerat i Figur 6.1.

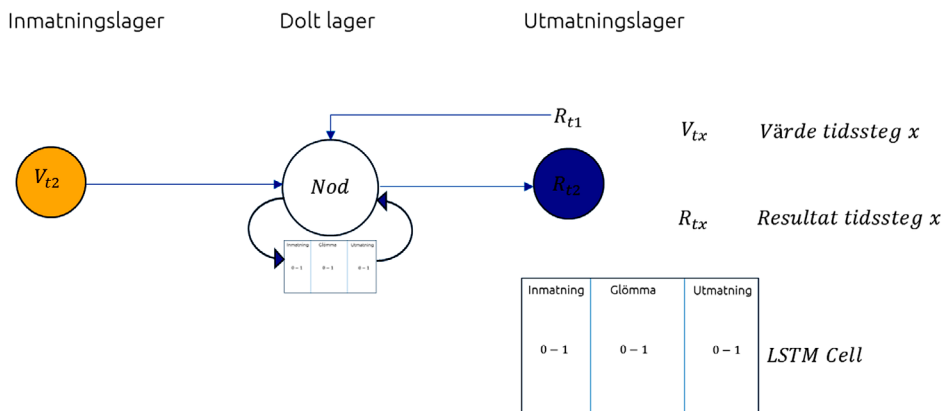


Figur 6.1

Schematiskt exempel på återkoppling i ett RNN.

Återkoppling av tidigare input är väldigt värdefullt inom språkigenkänning och för att fånga säsongsmässiga variationer i tidsseriedata. Det finns en risk med denna metodik: RNN är benägna att tappa relevant information över långa sekvenser, ett problem som kallas för försvinnande gradienter (*the vanishing gradient problem*). Detta uppstår när gradienterna som används för att uppdatera vikterna under träning blir väldigt små, vilket gör att inlärningen blir ineffektiv.

För att adressera detta problem introduceras en LSTM-cell i noden. Cellen använder så kallade grindmekanismer för att styra informationsflödet. Dessa grindar – inmatning, glömma och utmatning – har olika funktioner: inmatningsgrinden bestämmer vilken ny information som ska läggas till i minnescellen, glömgrinden avgör vilken information som ska tas bort, och utmatningsgrinden kontrollerar vilken information som ska användas i nästa steg. De olika grindarna släpper igenom lämplig mängd information genom att ansätta ett värde mellan noll och ett, vilket illustreras i Figur 6.2.



Figur 6.2

Schematiskt exempel hur en LSTM-cell används.

Detta gör att LSTM kan lagra viktig information över längre tidssekvenser och gör LSTM särskilt effektivt för uppgifter som tidsserieprognoser, naturlig språkbearbetning (exempelvis maskinöversättning eller textgenerering) och andra tillämpningar där förståelse för kontext och sekvens är avgörande. LSTM-flexibilitet och prestanda har gjort det till ett kraftfullt verktyg för att modellera komplexa tidsmässiga mönster och beroenden. Till skillnad från traditionella RNN kan LSTM hantera långsiktiga beroenden effektivt tack vare dess mekanismer för att bevara och glömma information vilket gör det mer pålitligt i uppgifter med längre sekvenser. Värt att notera är att detta skiljer sig från backpropagation, som är en metod för att justera vikter och bias i ett neuralt nätverk genom att sprida felen bakåt i nätverket för att minimera fel. LSTM:s grindmekanismer hanterar vilken information som ska bevaras eller glömmas för att bättre kunna modellera långsiktiga beroenden. I Tabell 6.5 ses en översikt över några fördelar och nackdelar med LSTM.

Tabell 6.5

Fördelar och nackdelar med LSTM.

Fördelar	Nackdelar
Är bra på att bevara information över långa sekvenser vilket är viktigt för språk och tidsserier	Beräkningsmässigt kostsam och långsam att träna, särskilt för stora dataset
Motverkar RNN-problematik med att tappa relevant information över långa sekvenser (vanishing gradients)	Har lätt för att överanpassa sig till träningsdata och kan kräva olika regulariseringstekniker för att motverka detta
Bred tillämpning i sekventiella dataområden oavsett om det rör taligenkänning eller tidsseriedata	Komplex arkitektur och kan kräva omfattande justering av hyperparametrar

En mer komplett matematisk beskrivning av LSTM kan ses i Bilaga B.

6.4.6 GRU

Gated Recurrent Units (GRU) är, i likhet med LSTM, en typ av återkopplande neuralt nätverk (RNN) som är utformat för att fånga sekventiella beroenden i data och hantera begränsningarna hos traditionella RNN.

Precis som LSTM förbättrar GRU vanliga RNN genom att använda grindmekanismer för att kontrollera informationsflödet, specifikt genom uppdaterings- och återställningsgrindar. Uppdateringsgrinden bestämmer hur mycket av den nya informationen som ska bevaras, medan återställningsgrinden avgör hur mycket av den tidigare informationen som ska glömmas. Dessa grindar arbetar tillsammans för att säkerställa att relevanta mönster kan läras in över tid och att GRU effektivt kan hantera både kortsiktiga och långsiktiga beroenden. Dessa grindar styr hur information uppdateras och bibehålls mellan tidssteg. Detta gör att GRU kan fånga långsiktiga beroenden och motverka problemet med försvinnande gradienter. Försvinnande gradienter uppstår när gradienterna som används för att justera vikterna i ett nätverk blir mycket små vilket leder till att inläringen blir ineffektiv, särskilt vid längre sekvenser. Som ett resultat är GRU väl lämpade för uppgifter som tidsserieprognoser, naturlig språkbearbetning och andra

sekvensbaserade problem. GRU utgör ett enklare och mer beräkningseffektivt alternativ till LSTM och uppnår ofta jämförbar prestanda med färre parametrar.

Det är dock värt att notera att GRU, på grund av sin enklare struktur, kan ha svårare att hantera mycket komplexa sekvenser och beroenden jämfört med LSTM. Detta eftersom LSTM har fler mekanismer för att kontrollera informationsflödet och lagra information på ett mer detaljerat sätt. I Tabell 6.6 ses en översikt över några fördelar och nackdelar med GRU.

Tabell 6.6

Fördelar och nackdelar med GRU.

Fördelar	Nackdelar
Är enklare och snabbare att träna än LSTM	Kan ha svårare att hantera mycket komplexa beroenden jämfört med LSTM
Har färre parametrar vilket gör GRU mer beräkningseffektiv	Saknar den flexibla kontroll över minneslagring som LSTM erbjuder
Kan ofta uppnå jämförbar prestanda med LSTM för många uppgifter	Är mindre anpassade för uppgifter som kräver mycket detaljerad minneshantering

En mer komplett matematisk beskrivning av GRU kan ses i Bilaga B.

6.4.7 CNN

CNN (*Convolutional Neural Networks*) är en typ av djupinlärningsalgoritm specifikt utformad för att bearbeta rutnätsliknande data (ofta benämnt raster eller grid) som bilder och rumsliga strukturer. CNN använder konvolutionella lager för att automatiskt och adaptivt lära sig hierarkiska rumsliga egenskaper. Detta görs genom att använda konvolutionella filter, som är små matriser som glider över bilden och beräknar konvolutioner för att upptäcka specifika lokala mönster, såsom kanter, texturer och former. Denna hierarkiska funktionsextraktion innebär att enkla mönster först identifieras. Dessa mönster kombineras sedan till mer komplexa representationer på högre nivåer. Detta gör att CNN effektivt kan identifiera och klassificera komplexa mönster med hög precision. Arkitekturen inkluderar vanligtvis pooling-lager, som hjälper till att minska dimensionerna genom att sammanfatta information från närliggande regioner samt fullt anslutna lager som används för slutgiltig klassificering eller regression. Pooling-lager reducerar datamängden och komplexiteten medan fullt anslutna lager fungerar som de sista beslutsfattande delarna i nätverket och tar in alla tidigare extraherade egenskaper för att utföra klassificering eller regression. CNN har revolutionerat områden som datorseende och bildigenkänning tack vare sin förmåga att hantera stora datamängder och sin robusthet i att extrahera relevanta funktioner. Exempel på specifika tillämpningar inkluderar självstyrande bilar, medicinsk bildanalys, och ansiktsgenkänning för säkerhetssystem. I Tabell 6.7 ses en översikt över några fördelar och nackdelar med CNN.

Tabell 6.7

Fördelar och nackdelar med CNN.

Fördelar	Nackdelar
Är mycket effektivt för bild- och videoanalys	Kräver stora datamängder för att uppnå god prestanda
Kan automatiskt extrahera funktioner utan behov av manuell feature engineering	Är beräkningsintensiv och kräver mycket resurser för träning
Har hög noggrannhet i att identifiera komplexa mönster	Är känsliga för rotationer och skaländringar i data

En mer komplett matematisk beskrivning av GRU kan ses i Bilaga B.

6.5 Prestandamått

Prestandamått (eller utvärderingsparametrar) som använts i detta projekt har primärt varit medelabsolut fel (*Mean Absolute Error*, MAE) samt medelabsolut procentfel (*Mean Absolute Percentage Error*, MAPE). Nedan förekommer också en jämförelse mellan MAPE och medelkvadratisk fel, MSE (*Mean Squared Error*), varpå en kortare beskrivning av MSE också inkluderas.

Det är vedertaget att använda de engelska termerna varpå förkortningarna MAE, MAPE och MSE kommer att användas framledes. Det är de mått som beskrivs nedan som använts för att utvärdera resultatet från olika kombinationer av variabler och algoritmer.

6.5.1 MAPE

Medelabsolut procentfel (MAPE) är ett vanligt förekommande mått för att utvärdera prestandan hos regressions- och prognosmodeller. Det uttrycker felprognosen som en procentandel, vilket gör det till ett användbart verktyg för att jämföra prestanda mellan dataset med olika storlekar och skalor. Likt Medelabsolut fel (MAE) har MAPE både fördelar och begränsningar.

MAPE definieras
$$\text{MAPE} = \frac{100\%}{n} \sum_{t=1}^n \left| \frac{y_t - \hat{y}_t}{y_t} \right|$$

Där:

- y_t är värdet vid tiden t
- $\hat{y}_t$ är det predikterade värdet vid tiden t
- n är antalet observationer

Fördelar med MAPE:

Skaloberoende: eftersom MAPE uttrycker felet som en procentandel av de faktiska värdena blir det enkelt att jämföra prestanda mellan dataset med olika skalor. Detta skiljer sig från MAE, som mäts i absoluta tal, det vill säga i samma enheter som de ursprungliga datavärdena.

Tolkning: MAPE ger ett tydligt och lättförståeligt mått. Det visar det genomsnittliga felet som en procentandel av de faktiska värdena. Till exempel betyder en MAPE på 10 % att modellens förutsägelser i genomsnitt avviker med 10 % från de faktiska värdena.

Begränsningar med MAPE:

Division med noll: MAPE är odefinierat när ett faktiskt värde är lika med noll. Detta utgör en utmaning i dataset där nollvärden förekommer eftersom division med noll utgör en matematisk omöjlighet.

Oproportionerlig felbedömning för värden nära noll: MAPE straffar felaktigheter oproportionerligt mycket när de faktiska värdena är små, eftersom nämnaren då blir väldigt liten. Till exempel; om ett faktiskt värde är nära noll och modellen gör en liten felaktig förutsägelse kan felet uttryckt i procent bli väldigt stort vilket snedvrider den totala bedömningen. I ett dataset med stor varians (både många värden som är väldigt små och väldigt stora) kan därmed felet för de väldigt små värdena få oproportionerligt stor påverkan på MAPE.

Jämförelse av MAPE och MAE

Typ av parameter: MAPE uttrycker felet som en procentandel vilket gör det användbart för att förstå felet i relation till de faktiska värdena. MAE ger ett absolut värde på felet i samma enheter som data. Detta gör det enklare att tolka när den faktiska storleken på felet är av intresse.

Användbarhet vid olika skalor: MAPE är skaloberoende och kan därför användas för

att jämföra fel över dataset med olika storlekar eller enheter. MAE är däremot skalberoende vilket gör det mindre lämpligt för jämförelser mellan dataset med olika enheter.

Problem med nollvärden: MAPE kan inte hantera situationer där de faktiska värdena är noll. Detta skulle innebära division med noll och ett odefinierat resultat. MAE har inte detta problem och kan hantera nollvärden utan komplikationer.

Hantering av extremvärden: MAE är mindre känsligt för extremvärden (outliers) eftersom det beräknar ett genomsnitt av de absoluta felen. Detta gör att enstaka extrema värden inte påverkar resultatet oproportionerligt mycket. MAPE kan däremot bli kraftigt påverkad av extremvärden, särskilt när de faktiska värdena är små eftersom det procentuella felet kan bli mycket stort.

Eftersom projektet undersöker data med många olika skalor används MAPE som huvudsaklig redovisningsmetod för tydlighet och jämförbarhet.

6.5.2 MSE

Medelkvadratisk fel (MSE) är ett vanligt använt mått för att utvärdera prestandan hos regressions- och prognosmodeller. Det mäter de kvadrerade felen mellan de faktiska och predikterade värdena, vilket gör det särskilt känsligt för stora fel. MSE är ett bra verktyg för att identifiera hur bra eller dåligt en modell förutsäger ett utfall speciellt när stora avvikelser är extra viktiga att upptäcka.

MSE definieras som:

$$MSE = \frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2$$

Där:

- y_t är det faktiska värdet vid tiden t
- $\hat{y}_t$ är det predikterade värdet vid tiden t
- n är antalet observationer

Fördelar med MSE:

Viktning av stora fel: eftersom MSE kvadrerar felen innan de summeras får större fel en oproportionerligt stor inverkan. Detta innebär att modellen tvingas förändras mer för att reducera stora fel vilket är fördelaktigt i situationer där stora avvikelser är kostsamma eller problematiska.

Matematisk egenskap: MSE är kontinuerligt och deriverbart vilket gör det enkelt att använda i optimeringsalgoritmer som gradient descent. Detta är en anledning till att MSE ofta används som förlustfunktion i neurala nätverk.

Begränsningar med MSE:

Känslighet för extremvärden: eftersom MSE kvadrerar felen kan stora fel ha en mycket stor påverkan på resultatet. Detta gör att MSE kan bli missvisande i dataset där extremvärden förekommer, eftersom dessa värden kan dominera felberäkningen och leda till en skev bild av modellens prestanda.

Enhetsberoende: till skillnad från MAPE är MSE inte skaloberoende. Detta innebär att MSE uttrycks i samma enhet som variabeln, men kvadrerad. För vissa tillämpningar kan detta göra tolkningen svårare. Detta eftersom felen inte uttrycks i samma skala som de ursprungliga datavärdena.

MSE ger en tydlig indikation på hur stora felen i modellens prediktioner är men bör användas med försiktighet i dataset som innehåller extremvärden eller stora variationer i storlek mellan variablerna.

6.5.3 MAE

Medelabsolutfel (MAE) är ett vanligt mått för att utvärdera prestandan hos regressions- och prognosmodeller. Det mäter den genomsnittliga absoluta skillnaden mellan modellens resultat och de faktiska värdena (träningsdata) utan att ta hänsyn till felens riktning. MAE är populärt tack vare sin enkelhet och tydliga tolkning men har också vissa begränsningar.

MAE definieras som:

$$\text{MAE} = \frac{1}{n} \sum_{t=1}^n |y_t - \hat{y}_t|$$

Där:

- y_t är värdet vid tiden t
- $\hat{y}_t$ är det predikterade värdet vid tiden t
- n är antalet observationer

Fördelar med MAE:

Tolkning: MAE är enkelt att tolka eftersom det ger den genomsnittliga storleken på felen i samma enheter som data. Till exempel innebär ett MAE på 5 i en temperaturprognos att modellens resultat i genomsnitt avviker med 5 grader.

Lika viktning av fel: Till skillnad från andra mått som medelkvadratfel (eng. Mean Squared Error, MSE) behandlar MAE alla fel lika oavsett dess storlek. Detta förhindrar att stora fel får en oproportionerligt stor påverkan på det totala resultatet.

Robusthet mot extremvärden: Jämfört med MSE är MAE mindre känsligt för extremvärden eftersom felen inte kvadreras. Detta gör det till ett mer robust mått i dataset som innehåller extremvärden.

Begränsningar med MAE:

Bristande känslighet för stora fel: Även om MAE robusthet mot extremvärden kan vara fördelaktig kan det också vara en nackdel när stora fel bör straffas hårdare. Inom exempelvis finansiell prognos kan stora avvikelser vara mer kritiska och kräva ett mått som betonar sådana fel mer.

Tar inte hänsyn till variabilitet: MAE behandlar alla fel lika men i vissa fall kan variationen i data göra att detta inte är önskvärt. Om data varierar kraftigt mellan olika tidsperioder och segment kan MAE missa att fånga felens betydelse i förhållande till skalan på data.

Icke-normaliserat fel: MAE ger inte ett normaliserat eller procentbaserat fel vilket gör det svårare att jämföra prestanda mellan olika dataset och tidsperioder med skilda skalor.

Värt att notera är också att MAE kan vara problematiskt att använda för att justera modellens parametrar (alltså inte för att utvärdera modellens resultat) i samband med gradientbaserad optimering av vikter och bias.

Jämförelse med MAPE och MSE:

Jämfört med MAPE: MAE har inga divisionsproblem vid nollvärden till skillnad från MAPE. MAE viktar inte heller fel beroende på de faktiska värdenas storlek. Däremot ger MAE, till skillnad från MAPE, inte ett relativt fel i procent.

Jämfört med MSE: MAE behandlar alla fel linjärt medan MSE kvadrerar felen och därmed ger större vikt åt större fel. Detta gör MSE mer känsligt för extremvärden medan MAE är mer robust i dessa situationer.

7 Resultat och diskussion

7.1 Resultat av utforskande dataanalys

I följande kapitel redovisas analys och slutsatser från den utforskande dataanalysen (EDA) som används för att bedöma databehandlingen, val av algoritm och vilka data som ska inkluderas eller exkluderas från erhållna indata.

7.1.1 Dygnsvärden EDA

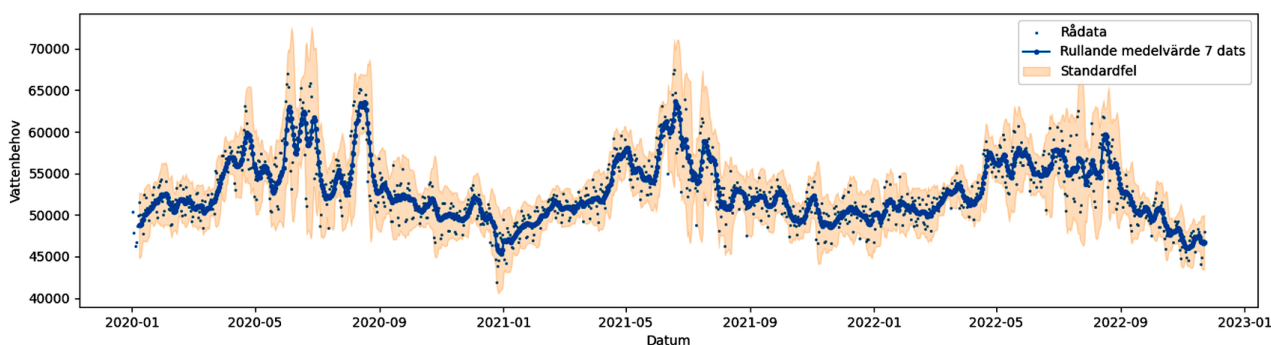
Förbrukning av dricksvatten, per dygn kan ses i Figur 7.1.

Luckor i data: Det dagliga datasetet innehöll en kontinuerlig serie för alla datum. Ett datum hade uppenbart felaktning förbrukning vilket var det första värdet i tidserien. Detta datum togs bort från tidsserien.

Generella mönster: Medelvärdet för förbrukning är 52 000 m³/dygn, med ett minimum på 43 000 m³/dygn och ett maximum på 66 000 m³/dygn, samt en standardavvikelse på 4 000 m³. Efterfrågan når sin topp under vår- och sommarmånaderna. De lägsta värdena ses under höst- och vintermånaderna. Ett veckomönster är också tydligt, där efterfrågan är lägst på helgerna.

Figur 7.1

Använd data för dricksvattenförbrukning, dess sju dagars rullande medelvärde samt standardavvikelse för det rullande medelvärdet.



Identifiering av extremvärden: Förekomst av statistiska avvikelser undersöktes med z-poäng samt IQR. Z-poäng antar att data är normalfördelade eller nästan normalfördelade medan IQR inte antar någon specifik fördelning vilket gör IQR mer lämpat för icke-normalfördelade data.

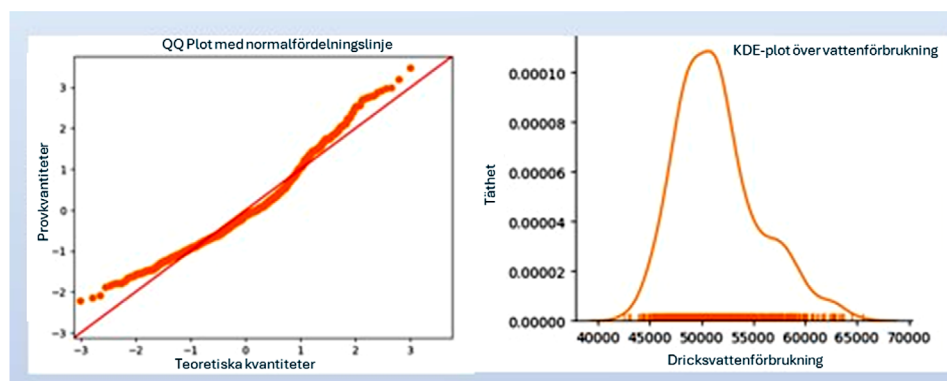
För z-poäng användes standardavvikelser (sigma) i intervallet 2–3,5. Datapunkter betraktas som avvikande om de ligger utanför dessa gränser under antagandet om en normal eller nästan normal fördelning. Undersökt dataset bedöms generellt vara stabilt. Inga extrema avvikelser upptäcktes vilket indikerar att systemet fungerar inom ett förutsägbart intervall. I det lägre spannet av använda standardavvikelser (2) identifierades ett antal extremvärden. Det bedömdes att dessa värden är en del av den normala fluktuationen i data och att dessa värden bör inkluderas i träningsdata. Ett liknande mönster observeras med IQR, där inga signifikanta avvikelser upptäcks och endast mindre avvikelser finns nära gränsen vid 2,5 IQR.

Fördelningen av data: Vilken fördelning data har är viktigt för att många traditionella statistiska metoder skapas utifrån förutsättningar om exempelvis normal- eller icke-normaldistribution. En liknelse kan göras mot beräkningar av flöde där beräkningsvalet styrs utifrån om det är laminärt eller turbulent flöde som ska analyseras. En QQ-plot

(Quantile-Quantile-plot) kan användas för att bedöma hur väl data passar en specifik fördelning genom att jämföra ett datasets kvantiler med kvantiler från en teoretisk fördelning, i detta fall normalfördelning. Följer data en rät linje kan data anses vara normalfördelade. Till vänster i Figur 7.2 ses att merparten av data är ungefär normalfördelade (vilket indikeras av punkterna nära den röda linjen). Dock indikerar avvikelserna från linjäritet att det finns fler extremt höga och låga värden än vad som är vanligt i en ren normalfördelning.

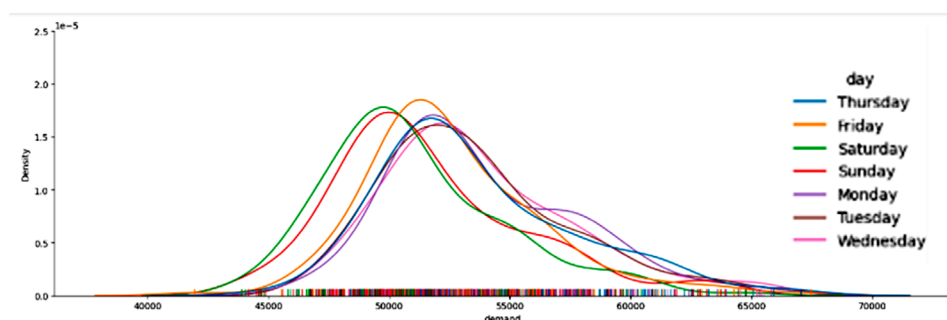
En KDE-plot (*Kernel Density Estimate plot*) är ett verktyg för att visuellt uppskatta sannolikhetsfördelningen av ett dataset. Det fungerar som ett smidigare alternativ till histogram genom att datapunkter jämnas ut och en kontinuerlig kurva skapas. KDE-plottar är användbara för att förstå hur data fördelar sig över ett intervall och kan ge insikter om statistisk fördelning av data. Till höger i Figur 7.2 ses tecken på en bimodal fördelning av data d.v.s. att fördelningen har "två toppar" till skillnad från en ren normalfördelning. Detta kan indikera att datasetet innehåller två undergrupper som driver olika förbrukningsmönster. Detta är viktigt att identifiera eftersom olika undergrupper kan ha egenskaper som påverkar analys och modellering.

För att undersöka om det finns olika typer av bakomliggande distributioner har data delats upp i olika intervall: per månad och per veckodag. För det sistnämnda fann vi att fördelning av förbrukningar på helger var mer jämnt fördelad än på vardagar. Skillnaden i vattenförbrukning på helger mot vardagar är känd i svensk vattensektor sedan tidigare vilket är relevant för analysen eftersom sådan kunskap kan hjälpa till att förbättra modelleringen genom att ta hänsyn till olika regelbundna mönster. Längre ner i Figur 7.3 och Figur 7.4 ses fördelning av data baserat på månad och veckodag.



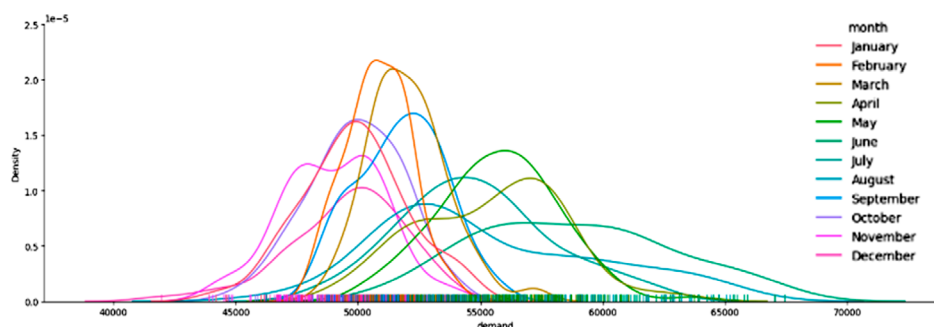
Figur 7.2

QQ- och KDE-diagram för vattenförbrukning. QQ-diagrammet till vänster visar att centrum för data passar en normalfördelning medan de extrema värdena längs med "svansarna" inte gör det. När KDE-diagrammet studeras ser vi indikationer på en bimodal eller till och med multimodal fördelning.



Figur 7.3

KDE-diagram med fördelningsresultat per månad för den totala vattenförbrukningen. I projektet undersöktes om den bimodala fördelningen som identifierats i tidigare diagram kan förklaras med månadsvariation. För vissa månader är den bi- eller multimodala fördelningen tydligare medan det för andra månader är mindre uppenbart. Värt att notera är att distributionen skiljer sig markant mellan olika säsonger.



Figur 7.4

KDE-diagram med fördelningsresultat baserat per veckodag. I projektet undersöktes om den bimodala fördelningen som identifierats i tidigare diagram är konsekvent över veckodagar. För vissa veckodagar är bimodaliteten tydligare än för andra veckodagar. Värt att notera är också de gröna (lördag) och röda (söndag) linjerna som tydligt indikerar ett annat mönster på helger än vardagar.

Stationäritet: Innebär att ett dataset baserat på en tidsserie är stabilt över tid avseende dess medelvärde och variation. Detta innebär att tidsserien inte visar några systematiska förändringar som trender eller säsongvariation. En stationär tidsserie uppvisar alltså ett stabilt mönster som gör den förutsägbart och konsekvent. Stationäritet är viktig i många statistiska modeller, såsom autoregressiva modeller (*AutoRegressive*, AR) och glidande medelvärdesmodeller (*Moving Average*, MA) eftersom dessa modeller bygger på antagandet om att data är stationärt. Om en tidsserie inte är stationär (eller åtminstone svagt stationär) kan den behandlas för att uppnå stationäritet inför modellering. Denna behandling kan sedan återställas efter att prognosen har gjorts.

En praktisk konsekvens av icke-stationäritet är att det ibland kan vara fördelaktigt att träna en ML-modell på perioder av data separat i stället för på hela tidsperioden. Genom att exempelvis träna upp en algoritm på en månads data i taget kan man minska effekten av icke-stationäritet. Stationäriteten för den dygnsbaserade vattenförbrukningen undersöktes med hjälp av fyra tester: ett enkelt test baserat på visuell inspektion av plottade data, ADF (*Augmented Dickey-Fuller*), KPSS (*Kwiatkowski-Phillips-Schmidt-Shin*) för medelvärde och trend.

Alla tester visade att data var icke-stationär vilket innebär att den behöver förbehandlas (t.ex. genom differensbildning) för att uppfylla kraven hos modeller som förutsätter stationäritet.

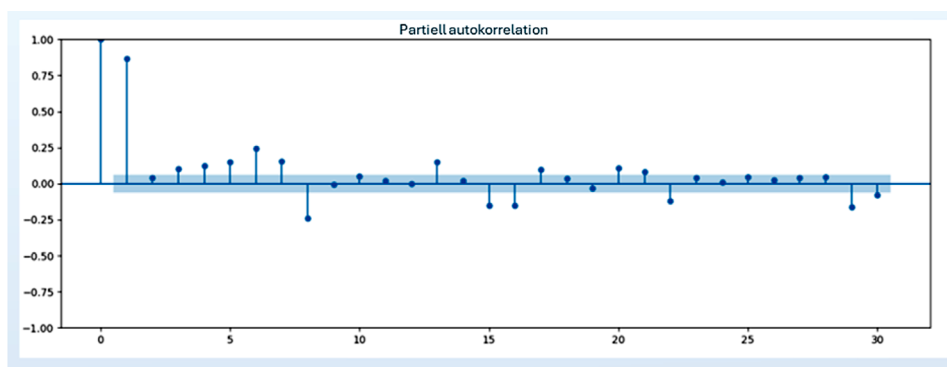
Autokorrelation: Är ett mått på hur värden i en tidsserie är relaterade till tidigare värden vid olika tidsförskjutningar (så kallade lags). Den beskriver i vilken utsträckning ett värde i en serie är beroende av dess föregående värde. Autokorrelation och stationäritet är nära sammanhängande eftersom stationäritet innebär att beroenden eller mönster (autokorrelation) i en tidsserie förblir konstanta över tid. Autokorrelation kan alltså användas som ett underlag för att identifiera den bästa behandlingen för att ett dataset ska närma sig ett stationärt tillstånd. Eftersom datasetet är icke-stationärt kan vi förstå förändringarna över tid genom att använda autokorrelation och partiell autokorrelation som undersöker hur tidigare värden påverkar nuvarande eller framtida värden.

Partiell autokorrelation mäter sambandet mellan en observation och en specifik tidigare observation, samtidigt som den tar bort effekten av alla mellanliggande värden. Detta innebär att om vi vill undersöka sambandet mellan ett värde två dagar bakåt och dagens värde, skulle vi eliminera påverkan från gårdagens värde.

Autokorrelation mäter sambandet mellan en observation och dess tidigare värden vid olika tidsförskjutningar. Detta innebär att autokorrelationsgrafen visar hur starkt ett värde i tidsserien är relaterat till tidigare observationer, vilket kan avslöja mönster som trender eller säsongeffekter i serien. Till skillnad från partiell autokorrelation inkluderar autokorrelation påverkan från alla mellanliggande värden, vilket gör att vi får en total bild av hur tidigare värden påverkar nuvarande värde.

Tolkning av partiell autokorrelation: Diagrammet som visar den partiella autokorrelationen presenteras i Figur 7.5. Diagrammet visar en signifikant partiell autokorrelation vid lag 1 (cirka 0,8) vilket indikerar en stark direkt relation mellan värdet vid tidpunkt

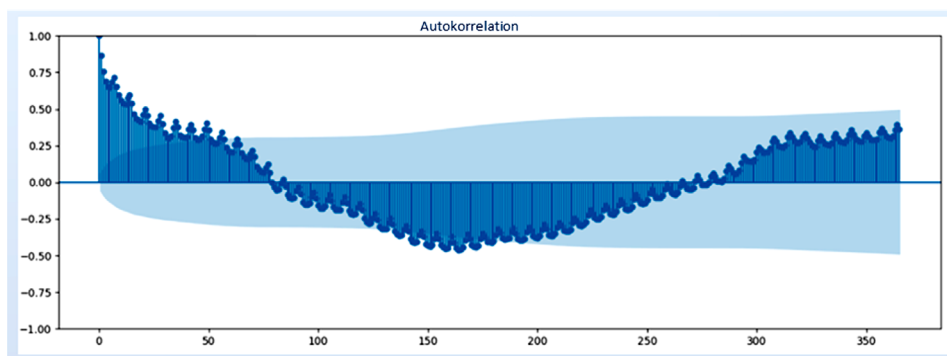
To och T1. Detta betyder att förbrukningen i dag har stark korrelation till förbrukningen i går. Den partiella autokorrelationen vid lag 2 är icke signifikant eftersom den ligger nära noll och inom det skuggade området. Mellan lag 3 och 7 ökar signifikansen successivt från att vara nästan obefintlig vid lag 3 när den upp till 0,25 vid lag 6 för att sedan sjunka något vid lag 7 innan den blir negativ vid lag 8. Förekomsten av partiell autokorrelation vid dessa lags tyder på en korrelation utöver lag 1 vilket kan bero på underliggande cykliska eller säsongsmässiga variationer i vattenförbrukningen, exempelvis kopplat till veckodagar.



Figur 7.5

Partiell autokorrelation för dygnsbaserad förbrukning av dricksvatten med lags på x-axeln.

Tolkning av autokorrelation: Autokorrelationsdiagrammet presenteras i Figur 7.6. Diagrammet visar en hög autokorrelation vid lag 1 som gradvis avtar långsamt och visar ett svängande beteende innan det inte längre är signifikant vid cirka 6 veckor. Vid 9 veckor blir autokorrelationen helt icke-signifikant. Dessutom ser vi att det mellan lag 140 och 180 finns en negativ korrelation som överstiger signifikansgränsen vilket tyder på en säsongsmässig inverkan, till exempel förändrad efterfrågan mellan sommar- och vintermånader.



Figur 7.6

Autokorrelation för dygnsbaserad förbrukning av dricksvatten med lags på x-axeln.

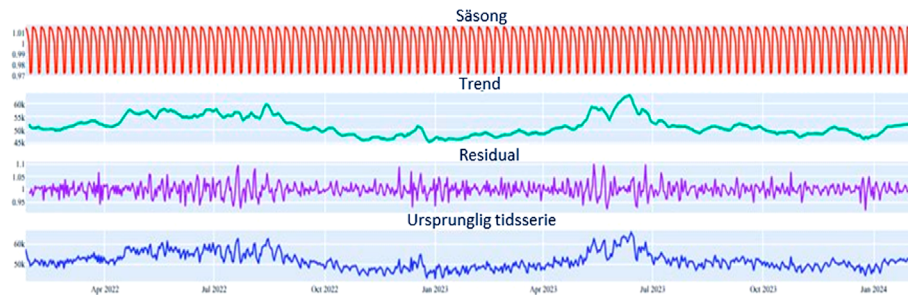
Både partiell autokorrelation och autokorrelation används ofta för att bestämma parametrar för autoregressiva och glidande medelvärdesmodeller såsom ARIMA. I projektet användes auto-ARIMA som iterativt testar olika parametrar för att hitta de mest lämpliga numeriska värdena. Resultatet från autokorrelationsstudien användes för att sätta gränser för sökning av parametrar inom auto-ARIMA vilket hjälper modellen att begränsa intervallet för möjliga parametrar och därmed optimera sökprocessen.

I Figur 7.5 och Figur 7.6 ses bevis på veckovisa säsongsmönster speciellt i den kort-siktiga partiella autokorrelationen. Tydligt är också att föregående värde har signifikant påverkan på nästa värde. Autokorrelationsdiagrammet visar bevis på veckovisa trender där korrelationen sjunker mellan vecka 6 och vecka 8. Dessutom syns bevis på negativ autokorrelation mellan 140–180 dagar vilket antyder säsongsmässiga mönster i vattenförbrukningen. Värt att poängtera är att redovisningen av autokorrelation och

partiell autokorrelation visar den statistiska relationen mellan förbrukningen för olika dygn över hela tidsserien.

Säsongvariation och trend

Flera analyser av säsongvariation och trend genomfördes på data genom att dela upp data i tre komponenter: säsongvariation, trend och residual (det som blir kvar efter att de andra komponenterna har tagits bort) vilket ses i Figur 7.7. Vi ser en veckovis säsongvariation i förbrukningsmönstret. Denna periodisering samverkar sedan med trenden som indikerar den längre säsongsmässiga variationen där skillnad i förbrukningsmönster mellan sommar och vinter är tydlig.

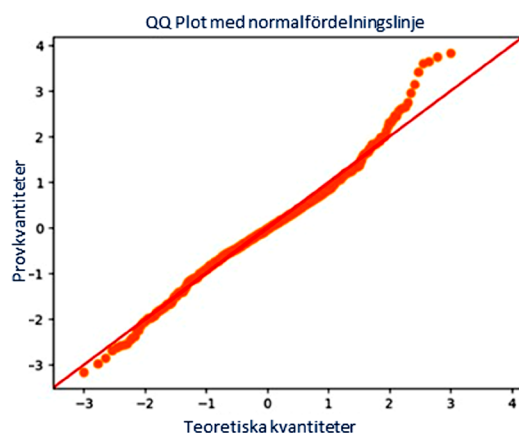


Figur 7.7

I säsongdekompositionsgrafen ser vi en tydlig säsongsmässig komponent över en vecka, vilket styrker tidigare undersökningar om ett förändrat förbrukningsmönster på veckodagar och helger. Trendkomponenten visar ökad förbrukning under sommarmånaderna med en minskning över vintermånaderna. Residualerna visar en grad av slumpmässighet som ökar över sommarmånaderna.

Analysen av residualvärden hjälper oss att bedöma hur väl modellen för säsongvariation och trend passar data. I en ideal värld bör residualerna uppvisa vitt brus vilket innebär konstant medelvärde, konstant varians och oberoende (stationära data utan autokorrelation). Detta indikerar att analysen tillräckligt väl har fångat upp underliggande säongs- och trendmönster. I detta fall ses visuellt att detta inte stämmer för residualerna. Framför allt ses en ökad grad av slumpmässighet för sommarmånaderna vilket indikerar att residualer inte bara består av vitt brus.

För att bättre förstå residualernas beteende undersökte vi deras underliggande fördelning med hjälp av en QQ-graf anpassad till en normalfördelning, se Figur 7.8. Punkterna i QQ-grafen ligger längs referenslinjen (den röda linjen), vilket indikerar att residualerna är ungefär normalfördelade. Avvikelse finns i de lägre och högre kvantilerna. I den övre delen divergerar punkterna ovanför linjen och i den nedre delen divergerar punkterna under linjen. Detta tyder på att residualerna innehåller fler extremvärden (både höga och låga) än vad som förväntas i en normalfördelning. De flesta punkter förhåller sig dock tillräckligt väl längs den röda linjen varpå slutsatsen togs att tillräckligt stor andel av residualerna är normalfördelade för att inte gå vidare med djupare analys.



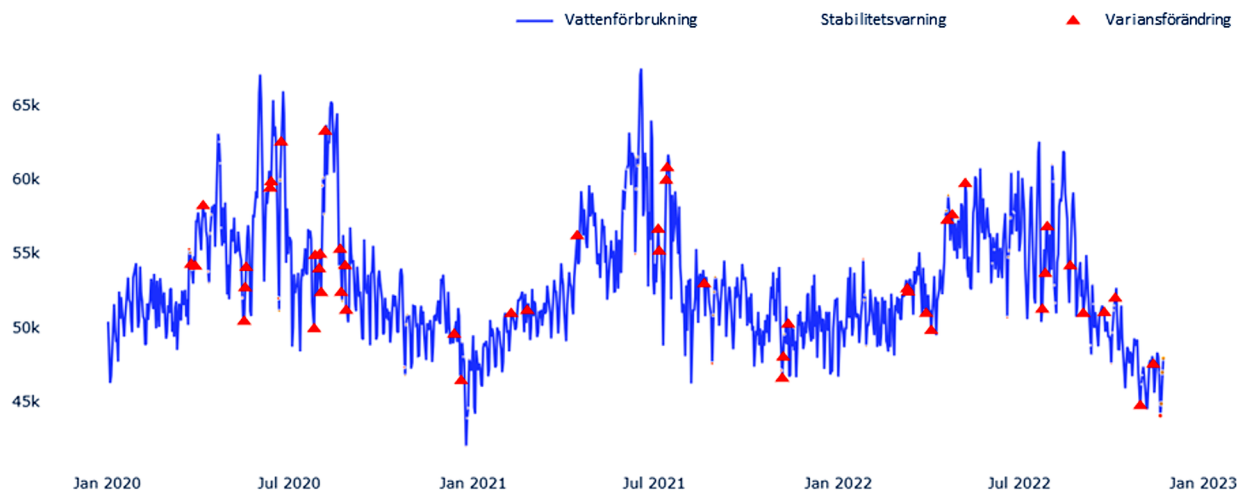
Figur 7.8

QQ-diagram över residualerna (från säsongdekompositionen) anpassad till en normalfördelning.

Tidsseriens stabilitet kontrollerades avseende medelvärde och varians med hjälp av rullande z-värde och Levenes varianstest. Levenes varianstest har tillämpats för att jämföra variansen mellan olika tidsfönster vilket ger en indikation om variansen är konstant över tid. Rullande z-värde används för att bedöma stabiliteten i tidsserien genom att kontinuerligt beräkna hur varje datapunkt avviker från medelvärdet inom ett glidande fönster av data. Detta hjälper till att identifiera om förbrukningen varierar mycket från sitt lokala medelvärde vilket kan indikera instabilitet i mönstret. Till exempel kan stor variation innebära plötsliga efterfrågetoppar eller oväntade dalar som skiljer sig markant från det normala mönstret inom en viss tidsperiod, resultat kan ses i Figur 7.9.

Figur 7.9

Stabilitetsvarningar inom data. Stabilitetsvarningarna är skalade efter storlek och är så små att de knappt syns.



Femtioåtta punkter visade sig ha ett värde som skiljde sig signifikant från medelvärdet av tidigare datapunkter. Av dessa femtioåtta punkter visade sig sexton ha betydande skillnader gentemot de fyra föregående punkterna. Ingen signifikant förändring i variansen observerades. Överlag bedöms tidsserien vara relativt stabil avseende medelvärde, varians och standardavvikelser gentemot omgivande datapunkter. Denna stabilitet i medelvärde och varians kommer att underlätta för vissa modeller.

Sammanfattning:

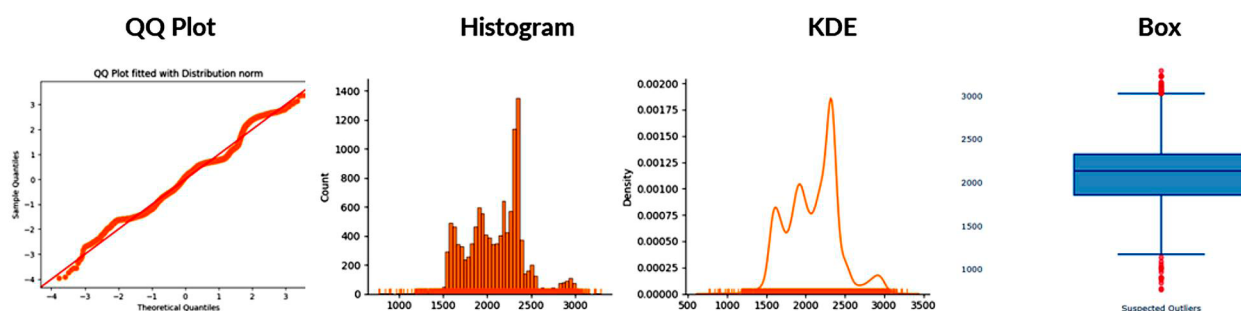
Den dagliga förbrukningen (utgående från Örbyfältet) visar mönster i veckoförbrukningen med stor skillnad på helg- och vardagsförbrukning. Till detta finns en långsiktig säsongsmässig trend mellan vinter och sommar. Under sommaren är graden av slumpmässighet i data som högst och även om det finns tydliga toppar i förbrukningen finns det inga signifikanta avvikelser. Överlag är data nära normalfördelade men det verkar finnas tecken på mer än en underliggande fördelning vilket inte kunde identifieras trots ansträngningar. Med tanke på datasetets stabilitet över tid bedömdes att ingen vidare bearbetning av data behövs.

Daglig förbrukning fanns också tillgänglig för en avsevärt mycket längre tidsperiod, cirka 18 år. Detta dataset har också undersökts och är generellt mindre stabilt. Detta var förväntat då flera nya områden har anslutits till Örbyfältet under de senaste 30 åren. Därtill ökar osäkerheten i data då metoder för datainsamling har förändrats under årens lopp.

7.1.2 Timvärden - EDA

Arbetet med timvärden för förbrukning var mindre omfattande än arbetet med dygnsvärden för förbrukning. Projektets fokus, att utvärdera ML som metod för minskat läckage på Örbyfältet, baseras till stor del på information som är baserat per dygn. En kortare

undersökning om timvärdesdata har genomförts för att visa på komplexiteten med högre frekvens och varians. I Figur 7.10 visas exempel på några av de statistiska analyser som tillämpades på timvärdesdata för vattenförbrukning.



Luckor i data: Timdatadatasetet innehöll tre saknade datum vilket kan tillskrivas övergången från centraleuropeisk tid (CET) till sommartid (CEST). Vid tolkning av datapunkter i närhet till övergången till sommartid är det viktigt att vara medveten om de saknade datumen. I detta dataset har saknade värden hanterats genom framåttifyllnad (forward-filling) vilket är en strategi för att fylla i luckor i data baserat på omkringliggande värden. Vid användning av annan tidzon, exempelvis *Universal Coordinated Time* (UTC) behöver denna typ av problem ej hanteras.

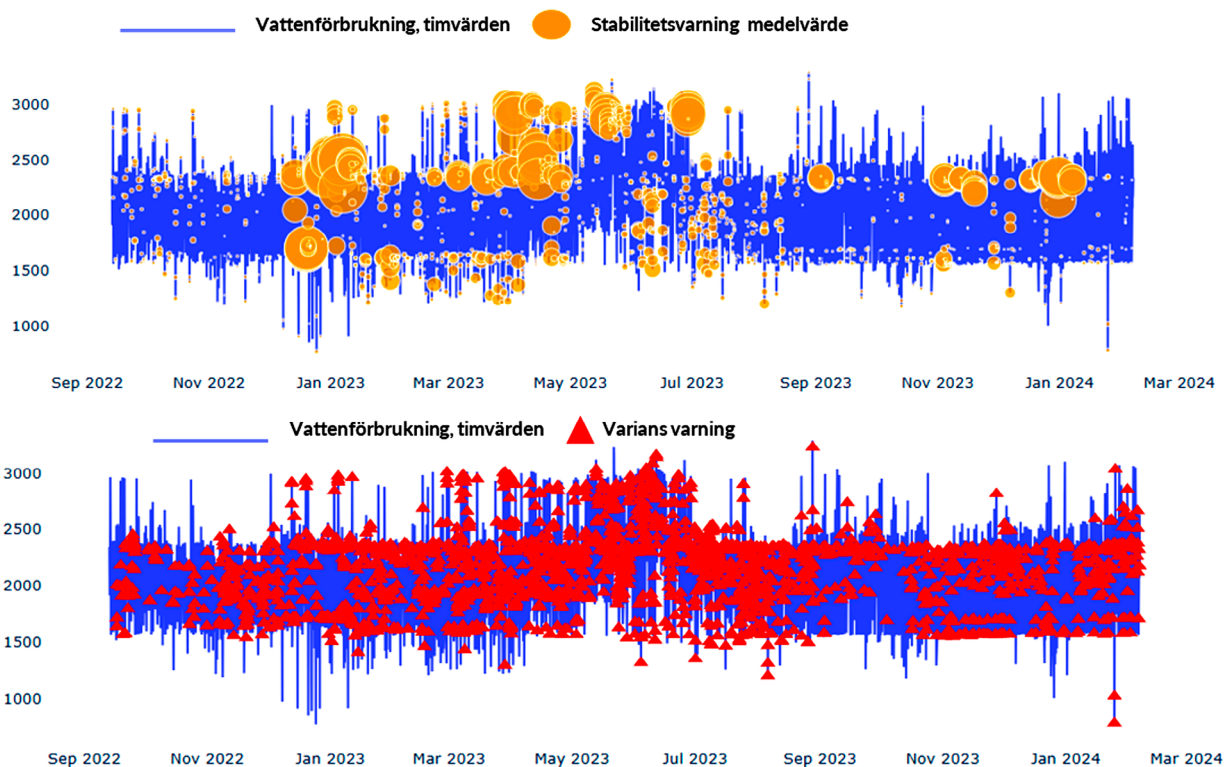
Vissa tidstämplar saknade också data. I början av datasetet saknades data för de 42 första timmarna. Dessa togs bort ur datasetet. Särskild hänsyn bör tas till så kallade NaN-värden (*Not A Number*) då de i vissa system kan representera nollvärden. Detta bedömdes inte vara fallet i detta dataset.

Identifiering av extremvärden: Förekomst av statistiska avvikelser undersöktes även här med z-poäng samt IQR. Förbrukningen per timme visade sig var mindre stabil än förbrukningen per dag, vilket var förväntat. Trots förekomsten av extremvärden bedöms den generella numeriska spridningen vara en del av den naturliga variansen i förbrukningsdata och bör därmed inkluderas i träningsdata. Möjligen kan vissa intervall av värden behöva hanteras separat i senare iterationer av modelleringsarbete. Det finns två nollvärden i datasetet, dessa bedöms vara onormala och ersattes med medelvärdet från de två närliggande värdena. Efter att ha tagit bort nollvärden ser vi en genomsnittlig förbrukning på cirka 2200 m³/h, ett minvärde på 680 m³/h samt ett maxvärde på 4080 m³/h. Maxvärdet skiljer sig markant från det näst högsta värdet på 3700 m³/h och får därmed anses vara en avvikelse.

Tidseriens stabilitet och varians utvärderades med rullande z-värde och Levenes varianstest. Syftet med detta är att identifiera möjliga felaktiga värden som inte har relevans mot föregående punkter. Resultat kan ses i Figur 7.11.

Figur 7.10

Exempel på några av de statistiska analyser som tillämpades på timvärdesdata.



Figur 7.11
Stabilitets- och
variansvarningar inom data.

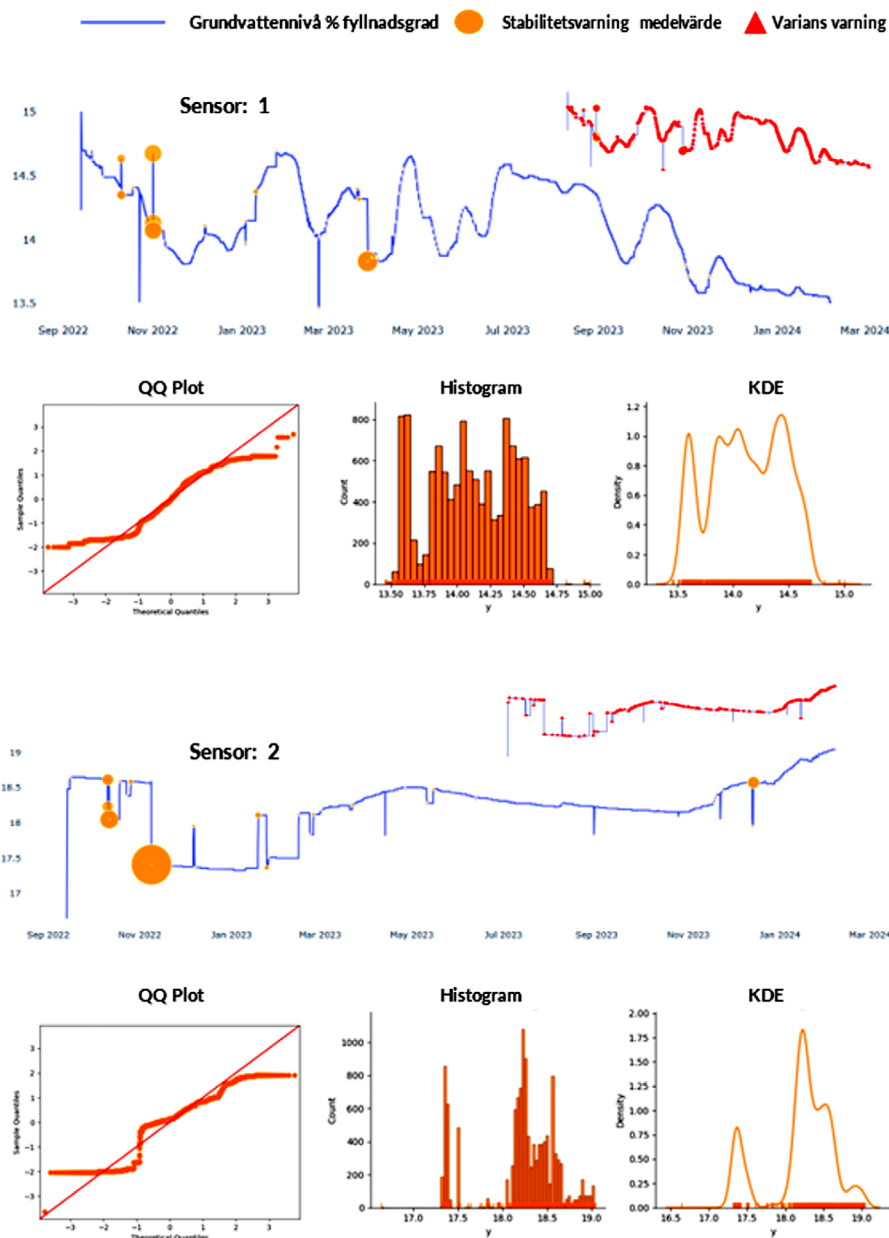
Timvärdesförbrukningen är relativt stabil men med ett tydligt mönster av instabilitet som är mest framträdande under helgerna, men inte begränsat till helgerna. Instabiliteten uppstod främst vid övergången mellan natt- och dagtidförbrukning vilket är förväntat då vi sen tidigare vet att det är stor skillnad på natt- och dagtidförbrukning.

Slutsatser från EDA av timvärdesdata

I förhållande till den dygnsbaserade förbrukningen är timbaserad förbrukning avsevärt mer instabil över tid med fler extremvärden. Fler extremvärden, mer varians och större datamängder ställer större krav på förbehandling och modellering för att uppnå ett gott resultat.

7.1.3 Grundvatten EDA

Driftoperatörer på Örbyfältet använder en beräknad fyllnadsgrad i fältet som underlag för beslut, exempelvis för beställning av vatten. Ingen tillgång fanns till de referensvärden som använts i den ursprungliga beräkningen. Däremot bygger den beräknade fyllnadsgraden på nivåmätningar som finns tillgängliga varpå EDA tillämpats enskilt på alla grundvattennivåmätningar. EDA utfördes på timvärden för varje enskild nivåmätare. I Figur 7.12 visas exempel på statistiska analyser tillämpade på två grundvattenmätare.



Figur 7.12

Exempel på några av de statistiska analyser som tillämpades på nivåmätningar. Här redovisade för 2 enskilda mätare. Resultaten skiljer sig markant för de enskilda mätarna, något som gällde för alla mätare.

Generellt observerades stor skillnad mellan de olika nivåmätarna. Även om det var små variationer i själva mätvärdena visade analyser på en instabil varians och ett instabilt medelvärde. Vid analys av data sågs också flera toppar i fördelningarna (multimodal fördelning). Detta kan tyda på att grundvattennivåerna påverkas av flera olika faktorer eller tillstånd. Varje topp kan representera en separat fas eller händelse. Exempelvis oregelbunden påverkan från styrning (pumpning) men det är också en indikator på ett väldigt komplext system. Det är viktigt att veta detta vid val av förbehandling av data och modelleringsteknik då denna typ av data är komplex att modellera.

Vid två tillfällen observerades ett skarpt negativt och kortvarigt fall av nivåerna. Initialt antogs att detta berodde på förändringar i intag eller uttag av vatten från fältet. Men inget bevis på detta kunde hittas i övriga data och inga liknande förändringar observerades i samband med förändringar av intag/uttag av vatten. Dessa förändringar kan bero på interna förändringar i akviferen där vatten förflyttar sig snabbt på ett "jordskredsliknande" sätt. Detta finns beskrivet i annan vetenskaplig litteratur och visar på att grundvattenakviferer är verkligt komplexa system.

Slutsatser från EDA av grundvatten

Grundvattennivåerna är mycket instabila över tid och visar dessutom på flera toppar i fördelningarna, vilket var förväntat. Dels på grund av komplexiteten i systemet, dels på grund av mänsklig inblandning i form av pumpning av vatten från fältet. Pumpningen sker dessutom inte regelbundet och förändringar i pumpregim (vilka pumpar som körs tillsammans och när de körs) medför att vi studerar ett system som inte har något naturligt normaltillstånd. Både påfyllnad av vatten och uttag av vatten påverkar alltså ett redan komplext system.

Att bearbeta dessa data, och skapa förståelse för underliggande mönster och samband, skulle vara väldigt svårt och inte nödvändigtvis givande. Modellerings bör därmed gynnas av en metod baserad på neurala nätverk där behovet av förbehandling av data är mindre, men behovet av datamängder är större.

7.2 Dygnsvärden – prognosmodell

7.2.1 Förutsättningar och mål

Målet med följande iterationer är att skapa en ML-prognosmodell för vattenförbrukning för 28 dagar fram i tiden. Målet är också att komma fram till en modell som ger ett tillräckligt bra resultat i förhållande till komplexiteten. Både avseende modellstruktur, tränings tid och tillgängliga metadata. Under modellbyggnadsfasen gick 4 CRISP-DM cykler igenom (se avsnitt 5.2.1).

I iteration 1 undersöks hur pass bra en modell kan prognosticera förbrukningen baserat på en variabel, själva vattenförbrukningen. Förekomsten av bara en variabel gör att detta kallas för en univariat analys. Lämplig träningslängd och batchstorlek för olika algoritmer utvärderas.

I iteration 2 undersöks hur framför allt väderdata kan påverka resultatet för vår prognosmodell. Utifrån domänkunskap kommer variabler skapas och utvärderas. Inkluderandet av flera variabler omnämns vanligen som multivariat analys.

I iteration 3 utvärderas fortsatt de bästa resultaten av iteration 1 och 2 med mer tillämpad feature engineering på redan skapade variabler. Där undersöks också påverkan av att inkludera information om exempelvis veckodagar och allmänna helgdagar.

I iteration 4 sammanfattas de bästa resultaten från iteration 1–3 och vidare undersökning görs för att se om tidigare bästa uppnådda resultat kan förbättras.

I nästa steg tillämpas de bästa resultaten från iteration 1–4 på det historiska datasetet som innehåller vattenförbrukning för en betydligt längre period, 18 år. Detta för att se om framtiden modell är generell och kan hantera de förändringar som skett på systemet under denna tidsperiod. Slutligen genomförs en kortare undersökning för att undersöka möjligheten att prognosticera timvärdesförbrukning i stället för dygnsbaserad förbrukning.

Värt att ha i åtanke är att denna process baseras på det övergripande målet för projektet – att undersöka hur ML kan användas för att minska vattenläckaget på Örbyfältet. Även om vidare utforskning kan göras i olika iterationer är detta basen för beslut om vilka modeller och parametrar som ska utvärderas vidare och vilka som väljs bort.

7.2.2 Baslinjesmodell – naiv modell

En baslinjesmodell (eller naiv modell) baserat på enklare statistiska modeller sattes upp för datasetet. I första steget jämförs resultatet från ML-algoritmer mot resultatet från baslinjesmodellerna. Om ML-algoritmer inte kan uppnå ett bättre resultat än baslinjesmodellen finns det ingen anledning att fortgå med en beräkningsmässigt mer komplex modell.

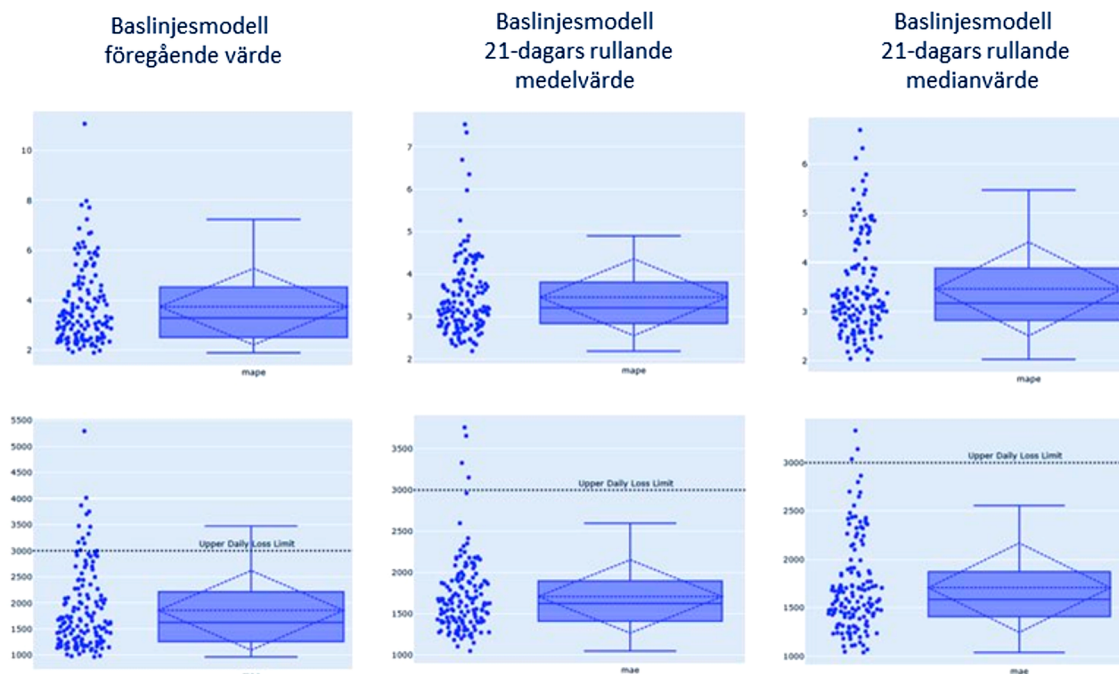
För detta dataset skapades tre olika baslinjesmodeller. Den första baseras enbart på det föregående värdet (last step) vilket betyder att det föregående värdet projiceras

framåt. Den andra baseras på ett 21-dagars rullande medelvärde och den tredje på ett 21-dagars rullande medianvärde.

Den första modellen hade ett MAPE-på $11,5 \pm 6,3 \%$. De rullande medel- och medianvärdesmodellerna hade MAPE på $8,9 \pm 3,6 \%$ respektive $8,3 \pm 3,4 \%$. Det rödmarkerade +-värdet visar på standardavvikelsen för MAPE. Ett låddiagram för MAPE och MAE för varje baslinjesmodell redovisas i Figur 7.13.

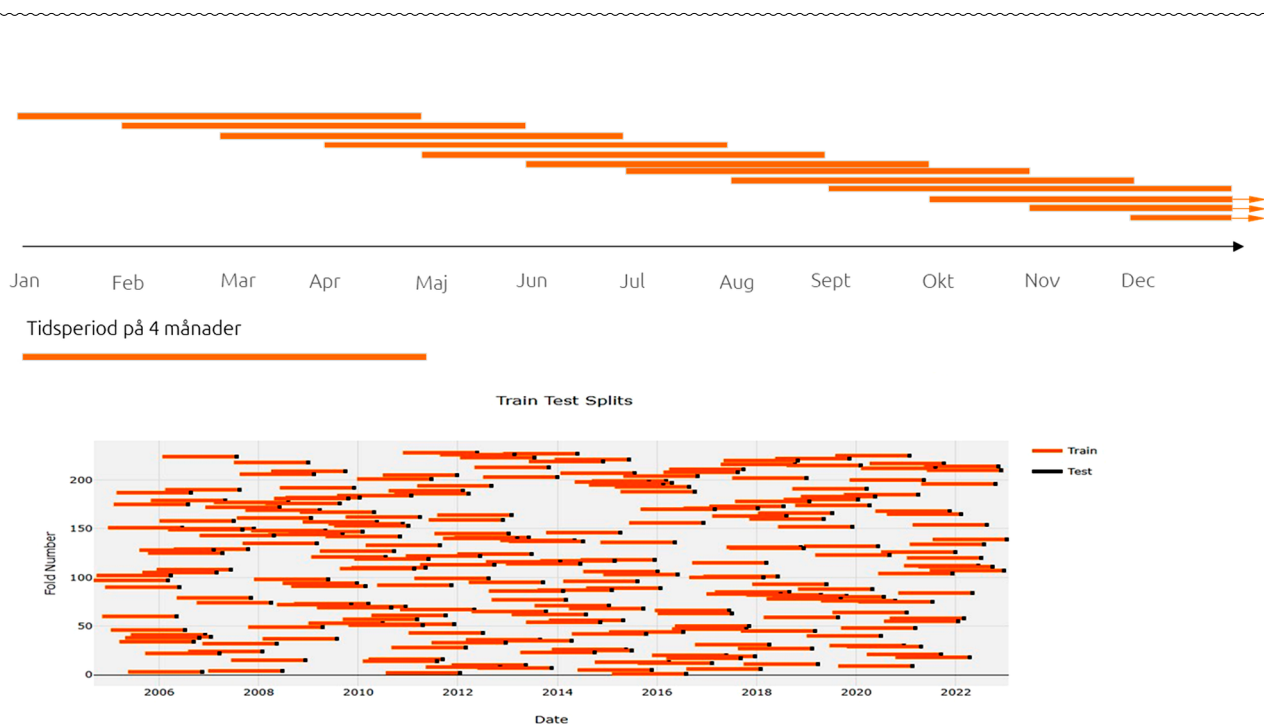
Figur 7.13

Box plot-diagram som visar MAPE-värden överst och MAE-värden för de tre baslinjesmodellerna.



7.2.3 Iteration 1

Under den första iterationen ville vi utvärdera hur väl målvariabeln (vattenförbrukning) kan predikteras utan att inkludera externa variabler såsom väderdata eller annan information. Prediktionstiden som används är 28 dagar framåt. En viktig process är att utvärdera hur träningsdata kan delas upp i lämpliga tidsperioder. Genom att dela upp träningsdata i lämpliga tidsperioder kan man fånga säsongsmässiga variationer mer fördelaktigt, felsöka i delar av data, hantera variationer och motverka överanpassning på träningsdata, vilket leder till en mer generaliserbar modell. Att dela upp träningsdata i olika tidsperioder innebär att man fortfarande använder alla tillgängliga data. Vill man exempelvis träna en ML-modell på ett års data och vill utvärdera en tidsperiod på fyra månader tränar man först på månad 1–4, därefter på 2–5, 3–6 o.s.v. tills hela året är täckt av överlappande perioder. Att utvärdera lämpliga träningsperioder i första iterationen är också viktigt när vi i ett senare skede ska introducera oberoende variabler som väderdata. Ett exempel på hur ett dataset kan delas in i olika träningsperioder kan ses i Figur 7.14.



Figur 7.14

Exempel på hur träningsdata delas in i olika, överlappande tidsperioder. Högst upp ett tematiskt exempel för förståelse, nedan en praktisk tillämpning för ett dataset som sträcker sig över en längre tidsperiod.

Inledningsvis skapades en (S)ARIMAX-modell för att skapa en statistisk referensmodell (d.v.s. inte en ML-modell). Prophet är en hybrid mellan statistisk modell och ML-modell vilken är utformad för att kunna användas av både specialister och icke-specialister. XGBoost och Support Vector Regression användes som traditionella ML-metoder, och slutligen undersöktes även ML-modeller baserade på neurala nätverk. Last Baseline (en referensmodell men mer sofistikerad än baslinjesmodellerna) skapades för att möjliggöra batch-träning av NN-modellerna: Linear, MultiDense, LSTM, GRU, CNN och LSTM-AR.

För traditionella ML-metoder gäller generellt att en längre träningsperiod är mer fördelaktig. Neurala nätverk, särskilt djupa neurala nätverk med många lager, har ett mycket stort antal parametrar och har en större kapacitet att lära sig väldigt specifika mönster inom just träningsdata vilket kan leda till risk för överanpassning. Hypotesen är därmed att vi kommer att se en förändring i resultatet för neurala nätverk i samband med utvärdering av olika längd på tidsperioder. Skalning av data kommer också att ha en effekt. På grund av komplexiteten i data gällande distributioner finns möjlighet att slutresultatet inte kommer förbättras utifrån skalning men att det kommer minska beräkningstiden för vissa algoritmer.

Traditionella ML-metoder

För de traditionella statistiska ML-modellerna fann vi att den bästa tidsperioden för träning var 532 steg vilket motsvarar ungefär 18 månader. Vi hade möjligen kunnat se en förbättring av resultatet om vi ökat tidsperioden men tyvärr fanns inte utrymme inom projektet att utvärdera längre tidsperioder.

För ARIMA såg vi bäst prestanda med oskalade data. Prophet visade förbättrad prestanda med både standardskalning och min/max-skalning. XGBoost-modellerna visade likartade resultat med skalade och oskalade data och SVR-modellerna gav sämre resultat med framför allt standardskalning. Denna minskning i prestanda hos SVR-modellerna kan sannolikt bero på känslighet hos kärnfunktionen (*kernels*) som använder avstånd mellan punkter vilket minskar i storlek genom skalning.

Sammanfattningsvis var resultaten för de traditionella ML-modellerna jämförbara med skalning och utan skalning. Eftersom vi i denna iteration bara arbetar med en variabel (inga tillagda data såsom väderdata) är inte detta helt orimligt. När vi går över till att använda flera variabler blir behovet av skalning generellt större. Användandet

av olika enheter/skalor för fler variabler medför ofta större numerisk spridning vilket innebär att behovet av skalning generellt är större. Generellt syntes inte större fel eller avvikelser på någon specifik del av året vilket gör att vi i kommande iterationer kommer att använda min/max-skalning ihop med en träningsperiod på 18 månader för traditionella ML-metoder.

Den ML-algoritm som konsekvent levererade bäst resultat var XGBoost. En mer komplett redovisning av modellutvärderingen kan ses i Tabell 7.1 till Tabell 7.3 som visar genomsnittligt MAPE för alla träningsperioder. I rött ses standardavvikelsen för MAPE baserat på alla beräkningar. Ett bra resultat är alltså både ett lågt MAPE-värde och en låg standardavvikelse. En mer komplett utvärderingstabell, också innehållande högst och lägst MAPE från varje enskild träningsperiod kan ses i Bilaga C.

Tabell 7.1

Träningstlängd med erhålllet MAPE. Ingen skalning av data där det bästa resultatet ses för XGBoost med en träningsperiod på 18 månader.

Månader:	18	12	6	3	2	1
ARIMA	3,1 +/- 1,0	4,2 +/- 2,4	5,0 +/- 1,5	5,1 +/- 2,9	4,6 +/- 1,8	5,0 +/- 3,0
PROPHET	3,1 +/- 1,0	4,6 +/- 2,1	3,6 +/- 1,4	4,2 +/- 1,3	4,8 +/- 3,2	3,5 +/- 1,0
SVR	2,72 +/- 0,36	4,2 +/- 3,1	4,1 +/- 2,6	4,7 +/- 2,8	7,4 +/- 4,5	4,9 +/- 2,8
XGB	2,1 +/- 0,3	3,5 +/- 0,5	3,4 +/- 1,1	3,7 +/- 1,5	3,7 +/- 1,5	3,9 +/- 1,5

Tabell 7.2

Träningstlängd med erhålllet MAPE. Standardskalning av data där det bästa resultatet ses för XGBoost med en träningsperiod på 18 månader.

Månader:	18	12	6	3	2	1
ARIMA	3,3 +/- 0,8	4,1 +/- 1,9	5,1 +/- 2,4	5,7 +/- 3,0	4,3 +/- 1,7	5,7 +/- 3,0
PROPHET	2,5 +/- 0,5	3,9 +/- 1,8	4,2 +/- 1,8	3,7 +/- 1,3	3,7 +/- 1,1	4,2 +/- 1,8
SVR	4,4 +/- 1,0	6,5 +/- 4,5	6,1 +/- 3,9	7,1 +/- 4,4	7,4 +/- 4,5	7,5 +/- 4,5
XGB	2,1 +/- 0,4	2,6 +/- 0,5	3,3 +/- 1,2	3,6 +/- 0,8	3,8 +/- 1,8	3,9 +/- 1,7

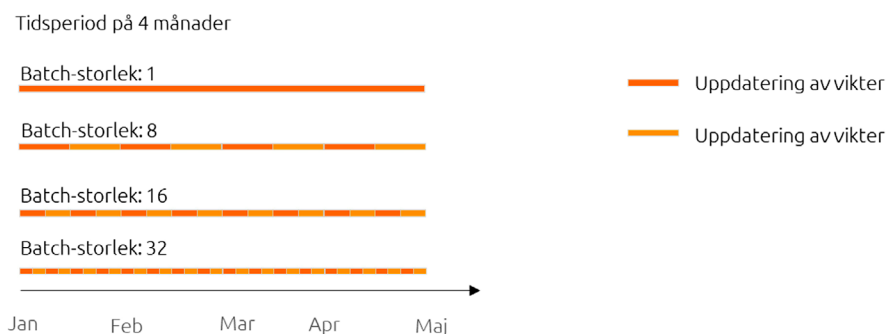
Tabell 7.3

Träningstlängd med erhålllet MAPE. Min/Max skalning av data där det bästa resultatet ses för XGBoost med en träningsperiod på 18 månader.

Månader:	18	12	6	3	2	1
ARIMA	3,3 +/- 0,9	4,6 +/- 2,8	5,0 +/- 2,8	5,1 +/- 2,7	3,6 +/- 1,2	4,1 +/- 2,0
PROPHET	2,6 +/- 0,4	4,3 +/- 2,0	4,4 +/- 2,0	4,3 +/- 2,0	4,3 +/- 2,0	4,3 +/- 2,0
SVR	2,8 +/- 0,4	4,5 +/- 3,6	4,3 +/- 3,0	5,1 +/- 3,3	5,3 +/- 3,4	8 +/- 5
XGB	2,1 +/- 0,4	2,7 +/- 0,5	3,6 +/- 1,2	3,8 +/- 1,6	4,0 +/- 1,9	3,6 +/- 1,8

Neurala Nätverk

Som tidigare diskuterats skiljer sig databehov och datahantering generellt sett mellan neurala nätverk och traditionella ML-algoritmer. För neurala nätverk spelar både indelningen i tidsperioder och antalet iterationer inom varje tidsperiod en viktig roll. Vid varje iteration skickas en delmängd av data från den valda tidsperioden genom nätverket (en så kallad batch av data för en iteration i det neurala nätverket), vilket uppdaterar både vikter och bias. Anledningen till detta är att göra modellen mer robust och minska risken för överanpassning vilket kan vara en stor risk om stora mängder data skickas i sin helhet genom ett neuralt nätverk. Det bör dock noteras att batch-träning inte helt eliminerar risken för överanpassning. En schematisk bild över hur en tidsperiod delas in i olika batcher kan ses i Figur 7.15.



Figur 7.15

Tematisk beskrivning av hur en tidsperiod delas in i olika batcher där varje batch medger justering av vikter och bias i en iteration. Detta är en tematisk bild avsedd att ge förklaring och inte den praktiska tillämpningen i utfört arbete.

Batch-baserad träning ställer generellt krav på större datamängder av god kvalitet. Detta beror på att metoden kräver tillräckligt med data för att skapa flera batcher som är representativa för hela datamängden, vilket säkerställer att modellen kan lära sig på ett tillförlitligt sätt. Utöver testdata behövs det också avsättas en viss mängd data för validering av vikter och bias (valideringsmängden) i en iteration. Denna används för att ställa in hyperparametrar exempelvis inlärningshastighet och antal noder i varje lager. Till skillnad från testdata används valideringsmängden för att upptäcka när modellen börjar överanpassa sig till träningsdata. När modellens prestanda börjar försämrats på valideringsmängden, även om den förbättras på träningsdata, är det ett tecken på överanpassning. För att hantera att data behöver delas in i träningsdata, testdata och valideringsdata används ibland korsvalidering (*cross validation*) där data delas upp i flera varianter (*folds*) så att varje del används omväxlande som valideringsdata och träningsdata mellan olika iterationer. Utifrån datakrav på batcher och total tillgänglig mängd data sattes en maximal träningsperiod på 5 månader.

För att möjliggöra jämförande analys mellan de olika NN-algoritmerna skapades en baslinjesmodell baserad på batch-träningsdata, vilket är en annan metod jämfört med den som användes för de traditionella ML-algoritmerna. Värt att notera är alltså att de resultat som redovisades för de traditionella ML-algoritmerna är en jämförelse med en annan baslinjesmodell än för de neurala nätverken. Generellt sett var de nätverksbaserade modellerna kapabla att överträffa baslinjesmodellen. De algoritmer med något mer komplicerad arkitektur, som CNN, GRU, LSTM och LSTM-AR var de som uppnådde bäst resultat. Vissa NN-modeller överträffade de traditionella ML-modellerna på delmängder av träningsdata. Men när vi summerade resultaten över alla tillgängliga träningsdata sågs att NN-modellerna gav sämre resultat än alla traditionella ML-modeller förutom den naiva baslinjesmodellen samt SVR med standardskalning. Detta indikerar att nästa iteration bör fokusera på att vidareutveckla traditionella ML-modeller. Resultat för de nätverksbaserade modellerna kan ses i Tabell 7.4 till Tabell 7.6. För alla NN-modeller redovisas resultat med standardskalning av indata.

	Batch-storlek 8				
Månader:	5	4	3	2	1
Last Baseline	5,0	5,7	5,8	4,5	4,6
Linear	4,9	4,4	5,1	5,2	5,5
MultiDense	4,2	4,3	4,7	4,8	4,4
LSTM	4,2	4,3	4,0	3,7	5,0
GRU	4,0	3,9	3,8	4,2	4,7
CNN	4,4	4,6	5,2	4,9	4,9
LSTM-AR	3,6	3,9	4,0	4,1	4,3

Tabell 7.4

Erhållet MAPE för olika träningsperioder för batch-storlek 8 där det bästa resultatet ses för LSTM-AR med en träningsperiod på 5 månader.

	Batch-storlek 16				
Månader	5	4	3	2	1
Last Baseline	5,5	4,8	5,3	5,6	5,6
Linear	5,9	4,4	6,3	6,1	5,5
MultiDense	5,1	4,3	4,2	4,4	5,3
LSTM	5,0	4,4	3,5	4,1	4,3
GRU	4,2	4,6	4,3	4,1	4,8
CNN	4,3	4,4	4,0	4,1	4,3
LSTM-AR	4,3	3,9	4,0	4,6	4,0

Tabell 7.5

Erhållet MAPE för olika träningsperioder för batch-storlek 16 där det bästa resultatet ses för LSTM med en träningsperiod på 3 månader.

	Batch-Storlek 32				
Månader	5	4	3	2	1
Last Baseline	5,6	5,2	5,0	5,4	5,1
Linear	4,6	6,3	5,1	5,9	5,9
MultiDense	4,3	4,4	4,6	4,9	5,1
LSTM	5,4	5,9	4,3	4,4	5,0
GRU	4,2	4,6	4,3	4,4	4,2
CNN	4,9	4,8	4,5	4,9	4,3
LSTM-AR	4,3	3,8	4,1	4,7	4,3

Tabell 7.6

Erhållet MAPE för olika träningsperioder för batch-storlek 32 där det bästa resultatet ses för LSTM-AR med en träningsperiod på 4 månader.

Givet att NN-modeller kan ha god förmåga att hantera komplexa, icke-linjära relationer mellan många variabler introducerades fler variabler i indata för de neurala nätverken redan i iteration 1. Detta för att inte avskriva de neurala nätverken utan en mer komplett utvärdering. Hur dessa variabler skapades kommer att beskrivas längre ned i rapporten.

Även efter introduktion av fler variabler (väderdata och märkning av händelser i väderdata) sågs resultat jämförbara med dem som erhöles vid användning av en variabel (vattenförbrukningen). Givet att NN-modeller kräver mer beräkningsmässig prestanda, är svårare att integrera i olika miljöer och utifrån erhållet resultat beslutades att vidare utvecklingsfokus ska ske på traditionella ML-modeller. Resultat för NN-modeller med flera variabler i indata kan ses i Tabell 7.7 till Tabell 7.9.

	Batch-Storlek 32				
Månader	5	4	3	2	1
LSTM	4,6	5,4	4,7	4,5	4,3
GRU	4,3	4,4	3,8	4,4	4,2
CNN	3,5	4,1	3,7	4,1	3,7
LSTM-AR	4,1	4,8	5,2	4,1	4,4

Tabell 7.7

Erhållet MAPE för olika träningsperioder för batch-storlek 32 samt en skapad funktion för soltimmar, nederbörd och temperatur. Det bästa resultatet kan ses för CNN med en träningsperiod på 5 månader.

	Batch-Storlek 32				
Månader	5	4	3	2	1
LSTM	5,1	7,3	6,9	6,2	5,0
GRU	4,6	5,5	5,8	6,0	4,2
CNN	3,8	4,3	4,3	4,6	4,3
LSTM-AR	4,0	6,5	6,2	6,5	4,3

Tabell 7.8

Erhållet MAPE för olika träningsperioder för batch-storlek 32 samt en skapad funktion för veckodag. Det bästa resultatet kan ses för CNN med en träningsperiod på 5 månader.

	Batch-Storlek 32				
Månader	5	4	3	2	1
LSTM	3,7	5,9	4,3	4,4	5,0
GRU	3,7	4,6	4,3	4,4	4,2
CNN	4,5	4,8	4,5	4,9	4,3
LSTM-AR	4,3	3,8	4,1	4,7	4,3

Tabell 7.9

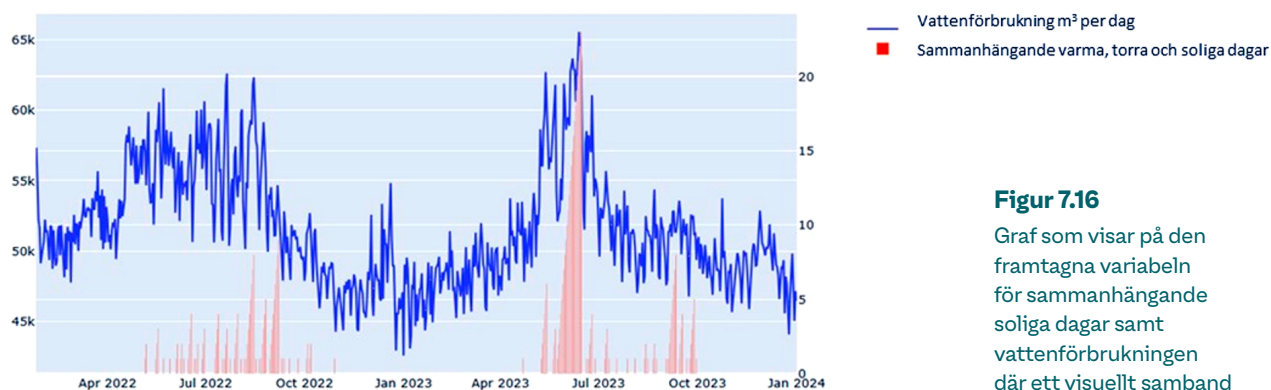
Erhållet MAPE för olika träningsperioder för batch-storlek 32 samt en skapad funktion för soltimmar, nederbörd och temperatur samt en skapad funktion för veckodag. Det bästa resultatet kan ses för LSTM samt GRU med en träningsperiod på 5 månader.

Sammanfattningsvis är NN-modellerna svårare att träna, kräver mer beräkningskapacitet, är mindre interoperabla och gav i detta fall ett likartat eller sämre resultat än mer traditionella ML-algoritmer. Värt att poängtera är att det finns goda resultat från NN-modellerna som troligen kan förbättras med vidare arbete. Men i förhållande till resultaten från de traditionella ML-modellerna är det av större nytta för projektets övergripande syfte att gå vidare med de bäst erhållna resultaten från iteration 1.

7.2.4 Iteration 2

Under möten med driftoperatörer erhöles information om att tre varma, torra och soliga dagar användes som en indikator på ökad dricksvattenförbrukning. Utifrån denna information skapades funktionsvariabler baserade på väderdata. En algoritm skapades som gjorde det möjligt för oss att definiera gränsvärden för varma, torra och soliga dagar samt räkna ihop sammanhängande dygn som uppfyller de valda kriterierna. En del utforskande arbete gjordes för att bestämma gränsvärden. För temperaturen användes dygnsmedeltemperatur där det nedre spannet för en dygnsmedeltemperatur initialt sattes till 12 grader Celsius. För solljusets varaktighet sattes ett undre gränsvärde på 4 soltimmar per dygn. Initialt användes 0 mm nederbörd som gränsvärde. Även om dessa gränsvärden kan te sig låga vid en första anblick stämmer de överens med inhämtade väderdata från SMHI.

I Figur 7.16 ses resultat där vattenförbrukningen redovisas i samma graf som utpekade sammanhängande varma, torra och soliga dagar. Av diagrammet framgår att det finns en tydlig visuell korrelation mellan den skapade variabeln och toppar i vattenförbrukning. Vår skapade variabel innehåller dock huvudsakligen dygn som inte uppfyller kriterierna (vilket ger ett förhållandevis glest dataset för träning).



Figur 7.16
Graf som visar på den framtagna variabeln för sammanhängande soliga dagar samt vattenförbrukningen där ett visuellt samband kan ses för enskilda sommarmånader men inte för övrig tid på året. Den framtagna variabeln är gjord med så kallad "one hot encoding". Vänster y-axel är vattenförbrukning i tusen m³, höger y-axel summan av dagar som gav indata till skapad variabel.

Även om gleshet inte i sig självt behöver vara negativt beslutade vi oss för att minska antalet sammanhängande dagar från tre till en för att se hur detta påverkade modellernas prestanda. Vid reducering av antalet sammanhängande dagar sågs en generell förbättring i modellernas prestanda. För att ytterligare förbättra den skapade variabeln justerades också gränsvärdet för nederbörd. Logiken bakom detta är att några få millimeter regn på en i övrigt torr och varm dag bör ha relativt liten inverkan på exempelvis markfuktighet. Vid körning av modellerna testades både användning av oskalade och min/max-skalade data. Resultaten för både skalade och oskalade data var jämförbara, troligen på grund av datasetets glesa natur. De mest relevanta resultaten av justeringar redovisas i Tabell 7.10.

Modell	Bästa MAPE	Nederbörd	Temperatur	Soltid (tim)	Bättre än en variabel?
ARIMA	3,5 +/- 1,2	0	12	6	Nej
PROPHET	2,7 +/- 0,51	0	20	8	Nej
SVR	2,6 +/- 0,36	5	16	6	Ja
XGB	2,1 +/- 0,51	2	12	8	Jämförbar

Tabell 7.10
Erhållet MAPE för ett antal algoritmer med varierande tröskelvärden för nederbörd, temperatur och soltid.

För ARIMA sågs ingen förbättring med de tillagda variablerna. Troligen beror detta på att variablerna tillför brus till datasetet på grund av dess glesa natur. ARIMA-modeller är beroende av att identifiera mönster i data, vilket gör att sporadiska eller sällsynta förekomster av en variabel kan göra det svårare för modellen att fånga meningsfulla samband mellan indata och målvariabeln. För Prophet var påverkan positiv jämfört med envariabel-analysen utan skalning men något sämre än envariabel-analysen med skalning. Precis som fallet med ARIMA är det sannolikt att den glesa distributionen av de tillförda variablerna tillför ett visst vitt brus till modellen. Det är möjligt att vi skulle kunna se en förbättring om vi levererade funktionen som ett dataset för helgdagar eller händelser, så att dess gleshet kan hanteras med förändringspunkter i stället för en kontinuerlig funktion. Detta fanns det dock ingen möjlighet att genomföra inom ramarna för detta projekt. För SVR såg vi en positiv påverkan för vissa kombinationer, särskilt vid 16 grader Celsius och soltimmar med 0-5 mm nederbörd. För övriga kombinationer var påverkan negativ och bidrog dessutom ibland till mycket stora fel. För XGB såg vi endast små eller relativt försumbara förbättringar utifrån kombinationer av de skapade variablerna.

För att skapa ytterligare insikter och undersöka varje variabels enskilda påverkan analyserades de också separat (och inte tillsammans som en skapad variabel). Vi använde gränsvärden från föregående arbete och definierade torra dagar som dagar utan nederbörd, soliga dagar som dagar med minst sex soltimmar och varma dagar som dygn med medeltemperatur över tolv grader Celsius. Vi experimenterade också med andra gränsvärden och genom att använda andra kombinationer av variablerna som indata. Resultat kan ses i Tabell 7.11.

Modell	Nederbörd	Soltid	Temperatur	Nederbörd Soltid	Nederbörd Temperatur	Temperatur Soltid	Nederbörd Temperatur Soltid
ARIMA	3,5 +/- 0,9	3,5 +/- 0,9	–	–	–	–	–
PROPHET	2,6 +/- 0,4	2,6 +/- 0,3	3,9 +/- 2,2	2,7 +/- 0,3	3,6 +/- 1,7	3,2 +/- 0,9	3,2 +/- 0,9
SVR	3,5 +/- 1,4	3,7 +/- 1,5	2,9 +/- 0,5	3,5 +/- 1,1	3,7 +/- 0,6	3,1 +/- 0,6	3,8 +/- 2,5
XGB	2,3 +/- 0,8	2,2 +/- 0,5	2,3 +/- 0,8	–	–	–	–

För sammanhängande torra dagar gav ARIMA ett resultat som var jämförbart med min/max-skalning av 'torra, varma och soliga dagar'. Resultatet var dock sämre än analysen baserad på en variabel, och träningstiden ökade avsevärt, så till den grad att träningen till slut avbröts. För Prophet såg vi en förbättring jämfört med oskalad analys av en variabel men inte jämfört med skalade data. SVR visade på en mindre förbättring och XGBoost på en mindre försämring.

Sammanhängande soliga dagar undersöktes på samma sätt som sammanhängande torra dagar och resultaten var likartade med undantaget att XGBoost-resultatet inte försämrades. Sammanhängande varma dagar undersöktes också på samma sätt som konsekutiva torra dagar med liknande resultat som följd. Olika kombinationer av variablerna undersöktes också med resultatet att förbättring uteblev.

Eftersom Prophet och SVR visade på bäst resultat för högra gränsvärden för temperatur och en mindre mängd nederbörd skapades också en variant som hade ett gränsvärde för temperatur på 16 grader och ett gränsvärde för nederbörd på 2 mm. Resultat kan ses i Tabell 7.12.

Tabell 7.11

Erhållet MAPE från analys av torra, soliga och varma dagar. Enskilt och i kombination.

Modell	Nederbörd <2 mm	Temperatur > 16 °C
ARIMA	–	–
PROPHET	2,6 +/- 0,5	3,2 +/- 1,1
SVR	3,0 +/- 0,6	4,0 +/- 2,3
XGB	–	–

Tabell 7.12

Erhållet MAPE för varierad analys av gränsvärden för torra dagar (nederbörd under 2 mm) och för en temperatur på 16 grader Celsius.

Sammanfattningsvis undersöktes påverkan av enskilda variabler för varma, torra och soliga dagar samt kombinationer av dessa. Generellt sett introducerade variablerna brus i modellerna vilket försämrade resultatet eller gav jämförbara värden mot vad som tidigare uppnått. Dessutom var dessa modelleringar mer beräkningsmässigt krävande med långa beräkningstider som följd. Detta gällde särskilt för XGBoost och ARIMA där träningstiden blev så lång att vi till slut fick avbryta modellträningen på grund av för höga minneskrav.

Eftersom vi vet att väderdata, ihop med mänsklig erfarenhet, är av vikt beslutades att en fördjupad undersökning av hur denna data kan förbehandlas ska genomföras i nästa iteration.

7.2.5 Iteration 3

Inom projektet beslutades att ytterligare utforska följande variabel separat: nederbörd, soltimmar, temperatur och typ av dagar samt tillämpa viss feature engineering på några av dem.

De skapade variablerna var följande:

Rullande medelvärde för:

- Soltimmar
- Temperatur – Kelvin

- Nederbörd som positivt värde
- Nederbörd som negativt värde
- Allmänna helgdagar

Temperatur:

- Temperatur redovisad på en positiv skala, det vill säga utan negativa värden, men inte Kelvin (där $0\text{ Celsius} = 273\text{ K}$)
- Temperatur uppdelad i negativa och positiva värden (det vill säga under och över fryspunkten)
- Temperatur uppdelad i 4-graders intervall (enligt kännedom att snö kan falla vid och under 4 grader)

Dagar:

- Enskilda veckodagar
- Helgdag eller vardag
- Skollov
- Allmänna helgdagar

Rullande medeltemperatur

Vi undersökte effekten av rullande medeltemperatur som en fristående (eller exogen) variabel. Fyra olika rullande medelvärden undersöktes för en, tre, sju och fjorton dagar. Där en dags rullande medelvärde motsvarar obehandlade data. Det är viktigt att poängtera att användningen av fristående variabler kräver att dessa är kända eller förutsägbara. Används en väderprognos är resultatet alltså beroende av hur bra denna prognos är. I detta fall har vi använt kända historiska värden och vi skulle förvänta oss en minskning i prestanda om vi använde prognosticerade värden.

För ARIMAX såg vi en ökning i modellens prestanda vid användning av rullande medelvärde för temperatur med störst förbättring för 3 och 7 dagars rullande medelvärde.

Prophet är generellt sett mer anpassat för att tillämpa fristående variabler, det kräver helt enkelt inte lika mycket förbehandling av data för att lägga till fler variabler till Prophet. När vi undersökte resultatet för Prophet uppstod ett oväntat beteende. I stort sett alla resultat hade väldigt höga MAPE-värden (över 7 %), ett resultat som är svårt att tolka. Ursprungligen observerades något märkliga värden för den januarimånad som inleder datasetet och beslutet togs att ta bort denna månad från träningsdatasetet. Detta resulterade i en ytterligare försämring i resultatet (med MAPE på 9 %). När vi också tog bort efterföljande februarimånad från träningsdatasetet samt delade in temperaturen i ett intervall på 2 grader erhöles ett MAPE-värde som var på samma nivå som de initiala resultaten utan oberoende variabler. Trots nyfikenheten inför detta märkliga resultat togs beslutet att ägna tid och resurser till annat arbete och detta lämnades därhän.

För XGBoost erhöles bäst resultat för rullande medelvärde på temperaturen för 7 dagar, därefter för 1 dag och slutligen för 3 dagar. Resultat för 1 och 7 dagar var jämförbara med de resultat som erhöles tidigare, utan någon externa metadata. Resultat kan ses i Tabell 7.13. En mer komplett utvärderingstabell, också innehållande högst och lägsta MAPE-värde från varje enskild träningsperiod, kan ses i Bilaga C.

Dagar för rullande medelvärde	1	3	7
ARIMA	3,2 +/- 0,6	3,1 +/- 0,5	3,1 +/- 0,4
PROPHET	3,6 +/- 1	3,4 +/- 1	3,1 +/- 1
XGB	2,3 +/- 0,5	2,6 +/- 0,6	2,3 +/- 0,4
SVR	2,8 +/- 0,4	2,8 +/- 0,4	2,8 +/- 0,4

Tabell 7.13

Erhållet MAPE för analys av rullande medelvärde på temperatur.

Justeringar av temperaturdata

Vi undersökte hur temperatur och dess skala påverkar resultaten. Den första åtgärden var att höja temperaturskalan så att inga negativa värden förekommer. Detta gjordes för att förenkla den numeriska hanteringen av algoritmerna eftersom negativa värden kan skapa problem vid vissa matematiska operationer och försvåra inläringen i vissa typer av modeller. Nästa åtgärd var att dela upp datasetet för temperatur i två delar där den ena serien innehåller värden över 0 grader och den andra värden under 0 grader. Detta gjordes för att hjälpa algoritmen att identifiera olika mönster beroende på om temperaturen är över eller under fryspunkten, vilket kan förbättra modellens inläring och prediktiva förmåga. Snö är relativt vanligt förekommande vid temperaturer upp till 4 grader. Eftersom vi inte känner till den direkta korrelationen mellan kyla och vattenförbrukning, delades temperaturdata vid 4 och vid 2 grader.

Erhållna resultat liknade de som uppnåddes för rullande medeltemperatur. Felaktigheter observerades i data för de två första månaderna i datasetet (januari och februari). När dessa månader togs bort från träningsdata uppnåddes jämförbara resultat som när algoritmerna tränades på endast en variabel. Det finns en möjlighet att indelningen av temperaturer kan ha större effekt i ett kallare klimat, men detta är inte undersökt i denna studie. Resultat kan ses nedan i Tabell 7.14. En mer komplett utvärderingstabell, också innehållande högst och lägsta MAPE-värde från varje enskild träningsperiod, kan ses i Bilaga C.

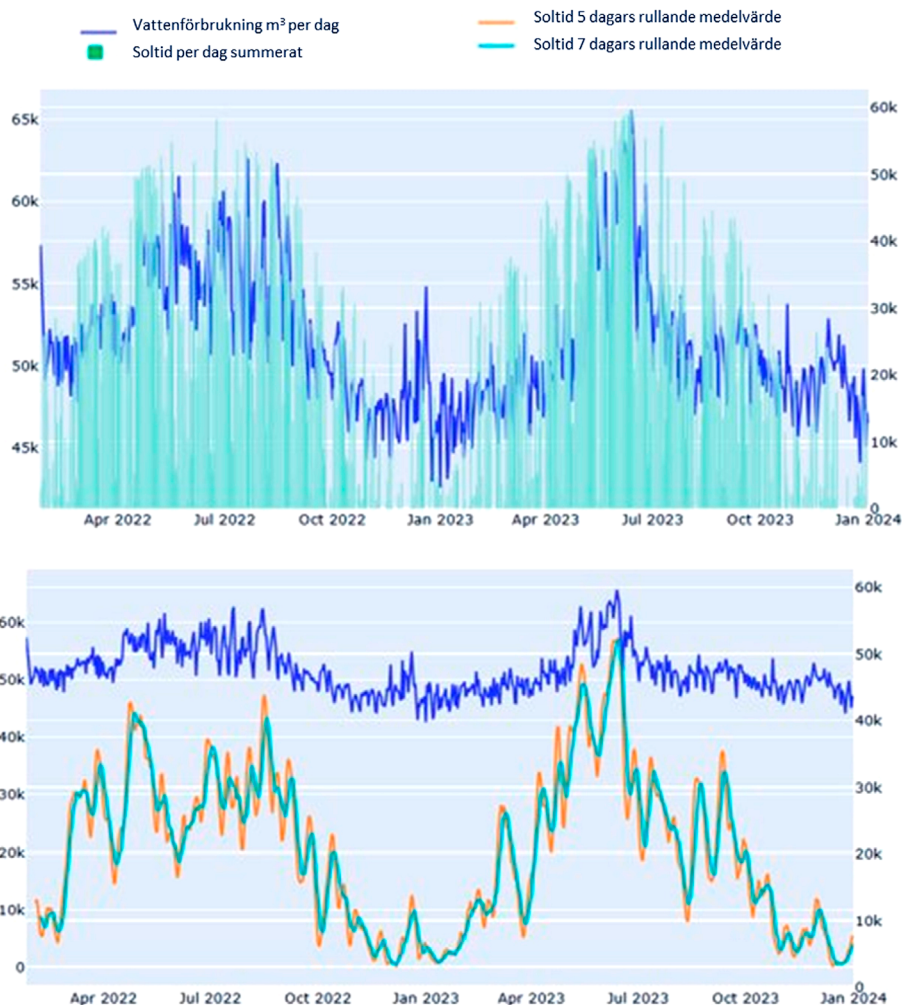
Justering av temperatur	Ingen	Positiv skala	0 splitt	2 splitt	4 splitt
ARIMA	3,2 +/- 0,6	3,2 +/- 0,6	3,0 +/- 0,4	3,1 +/- 0,4	3,0 +/- 0,4
PROPHET	3,6 +/- 1,4	3,6 +/- 1,4	3,5 +/- 1,3	3,5 +/- 1,6	3,4 +/- 1,5
XGB	2,3 +/- 0,5	2,3 +/- 0,5	2,3 +/- 0,5	2,3 +/- 0,5	2,4 +/- 0,5
SVR	2,8 +/- 0,4	2,8 +/- 0,4	2,7 +/- 0,5	2,9 +/- 0,4	2,8 +/- 0,5

Tabell 7.14

Erhållet MAPE för justering av temperaturdata/skala.

Rullande medelvärde för soltimmar

När antalet soltimmar per dygn studeras ihop med förbrukningen av dricksvatten framträder en tydlig visuell korrelation, se Figur 7.17. Denna korrelation syns ännu tydligare med längre rullande medelvärden för soltimmar, varpå dessa variablers påverkan på resultat undersöktes. Antalet soltimmar hämtas från SMHI:s närmsta väderstation som mäter antalet soltimmar, i Lund.



Figur 7.17

Resultat som visar soltid mot vattenförbrukning. Den översta delen visar faktisk soltid och den undre delen visar fem och sjudagars rullande medelvärde. Vänster y-axel är vattenförbrukning i tusen m³, höger y-axel soltid i sekunder.

För ARIMA visade användningen av bara soltimmar en negativ eller jämförbar inverkan jämfört med den oskalade (MAPE 3,1 %) och min-max-skalade (MAPE 3,4 %) modellen innehållande bara vattenförbrukning. För rullande soltimmar sågs en positiv inverkan, med störst förbättring vid fjorton dagars rullande medelvärde (MAPE 2,6 %). Variabeln för sammanhängande soliga dagar visade också en positiv effekt (MAPE 2,9 %).

För Prophet erhöles något sämre resultat än det som tidigare uppnått. XGBoost visade generellt på jämförbara (eller något försämrade) resultat förutom för rullande 14-dagars medelvärde för soltimmar. För SVR erhöles ett sämre resultat än vad som tidigare uppnått för alla variabler. Summerat resultat kan ses i Tabell 7.15. En mer komplett utvärderingstabell, också innehållande högst och lägsta MAPE-värde från varje enskild träningsperiod, kan ses i Bilaga C.

Tabell 7.15

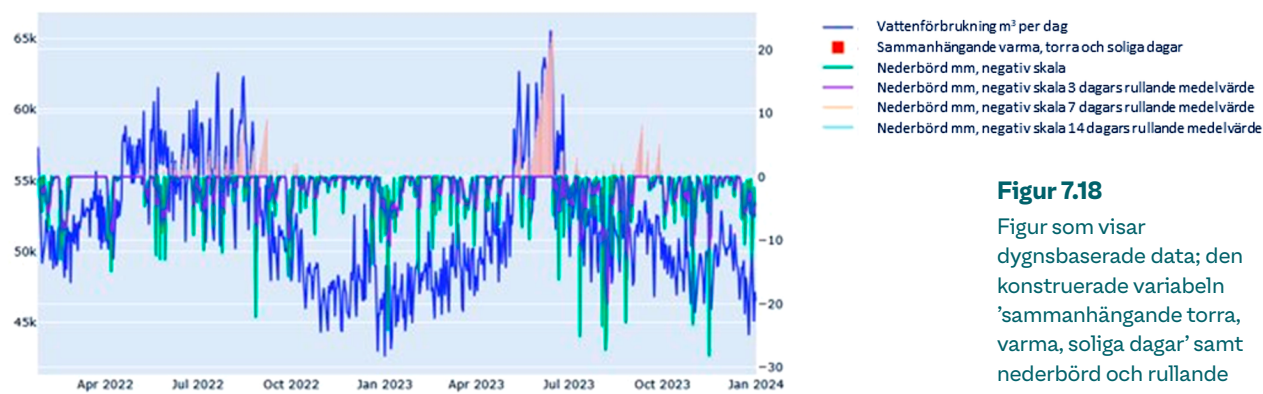
Erhållet MAPE för rullande soltimmar samt sammanhängande dagar med mycket sol.

Dagar för rullande medelvärde	1	3	7	14	Sammanhängande soliga dagar
ARIMA	3,4 +/- 0,9	3,0 +/- 0,5	2,8 +/- 0,4	2,6 +/- 0,3	2,9 +/- 0,6
PROPHET	2,9 +/- 0,30	2,8 +/- 0,40	2,6 +/- 0,42	2,6 +/- 0,3	2,6 +/- 0,23
XGB	2,3 +/- 0,4	2,2 +/- 0,3	2,3 +/- 0,4	1,9 +/- 0,35	2,1 +/- 0,4
SVR	3,0 +/- 0,5	2,9 +/- 0,5	2,9 +/- 0,6	4,0 +/- 1,8	3,0 +/- 0,6

Justering av nederbördsdata

Skapade variabler från nederbördsdata visas i Figur 7.18. För ARIMA försämrade användningen av nederbördsvariabler MAPE generellt gentemot tidigare uppnådda

resultat. För Prophet var resultaten jämförbara gentemot tidigare uppnådda resultat för en variabel med min/max-skalning. För SVR uppvisades en förbättring mot tidigare uppnådda resultat och för XGBoost var resultaten likvärdiga mot tidigare erhållet resultat.



Figur 7.18
Figur som visar dygnsbaserade data; den konstruerade variabeln 'sammanhängande torra, varma, soliga dagar' samt nederbörd och rullande nederbörd som negativa värden. Vänster y-axel är vattenförbrukning i tusen m³, höger y-axel mm nederbörd.

Tabell 7.16. En mer komplett utvärderingstabell, också innehållande högsta och lägsta MAPE-värde från varje enskild träningsperiod, kan ses i Bilaga C.

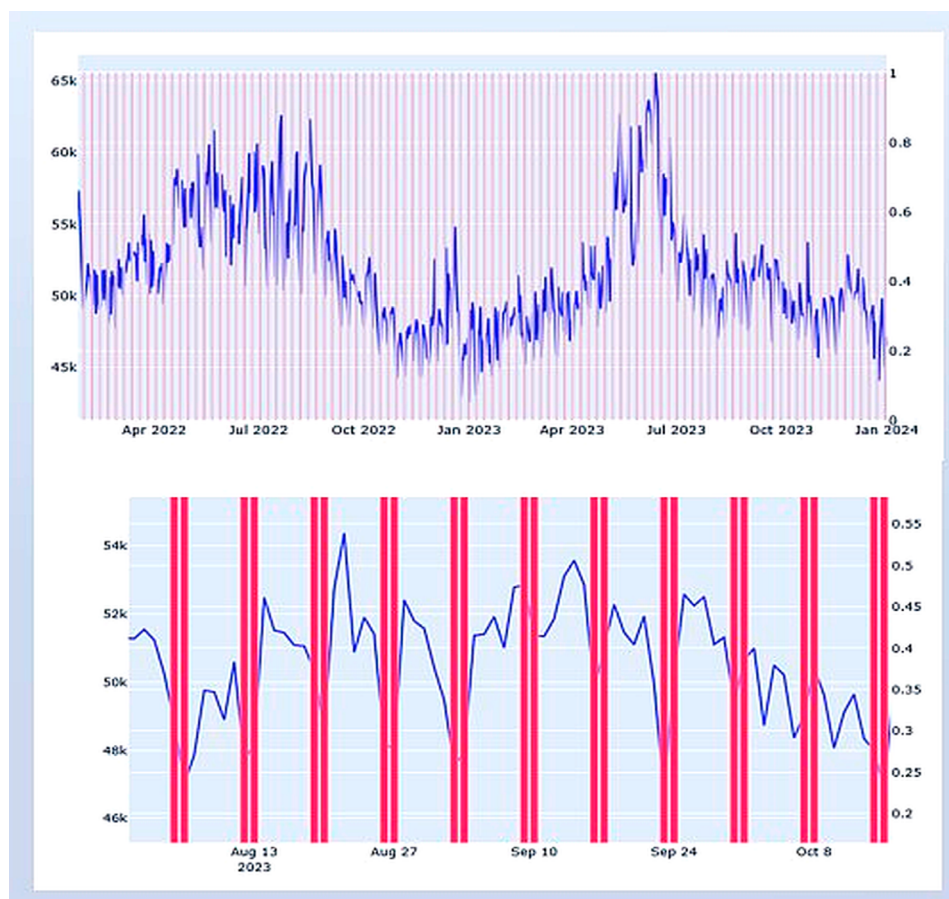
Dagar för rullande medelvärde	1	3	7	14	Neg
ARIMA	3,4 +/- 0,9	3,3 +/- 0,9	3,4 +/- 1,0	3,3 +/- 0,9	3,3 +/- 0,9
PROPHET	2,45 +/- 0,24	2,55 +/- 0,24	2,53 +/- 0,28	2,73 +/- 0,57	2,50 +/- 0,24
SVR	2,6 +/- 0,3	4,1 +/- 3,3	2,7 +/- 0,5	2,8 +/- 0,6	3,3 +/- 1,8
XGB	2,21 +/- 0,28	2,43 +/- 0,26	2,17 +/- 0,27	2,16 +/- 0,31	2,32 +/- 0,36

Justering av indata gällande dygn
Under den utforskande dataanalysen (EDA) upptäcktes vissa dagliga beroenden. Under helger sjunker efterfrågan och har en mer normalfördelad spridning jämfört med vardagar. Fördelningen verkar vara bimodal med den högsta nivån av bimodalitet mitt i veckan. Därför bedömdes det vara relevant att undersöka variabler kopplade till dygn ytterligare.

Tabell 7.16
Erhållet MAPE för behandling av nederbördsdata.

Veckodagar: Information om veckodagar kodades med så kallad one-hot encoding vilket innebär att man representerar kategoriska variabler i numerisk form. Detta är praxis, skulle ett sekventiellt numeriskt värde (från 1–7) ges till dagarna i en kolumn placeras dessa i en numerisk ordning. I stället ges varje veckodag en egen kolumn där värdet 1 eller 0 fylls i.
Både Prophet och SVR visade positiva resultat och gav de jämförbart bästa MAPE-värden än så länge. ARIMA gav jämförbara värden mot tidigare uppnådda resultat medan XGBoost presterade något sämre än tidigare uppnådda resultat.

Helger: Efterfrågan under helger kan ses i Figur 7.19. Under den utforskande dataanalysen sågs ett veckovist säsongsmönster. Detta mönster undersöktes nu som en variabel, en helgindikator.
För Prophet och ARIMA såg vi förbättrade resultat när helgindikatorn lades till som en sekundär funktion. Detta gällde dock inte för XGBoost eller SVR.

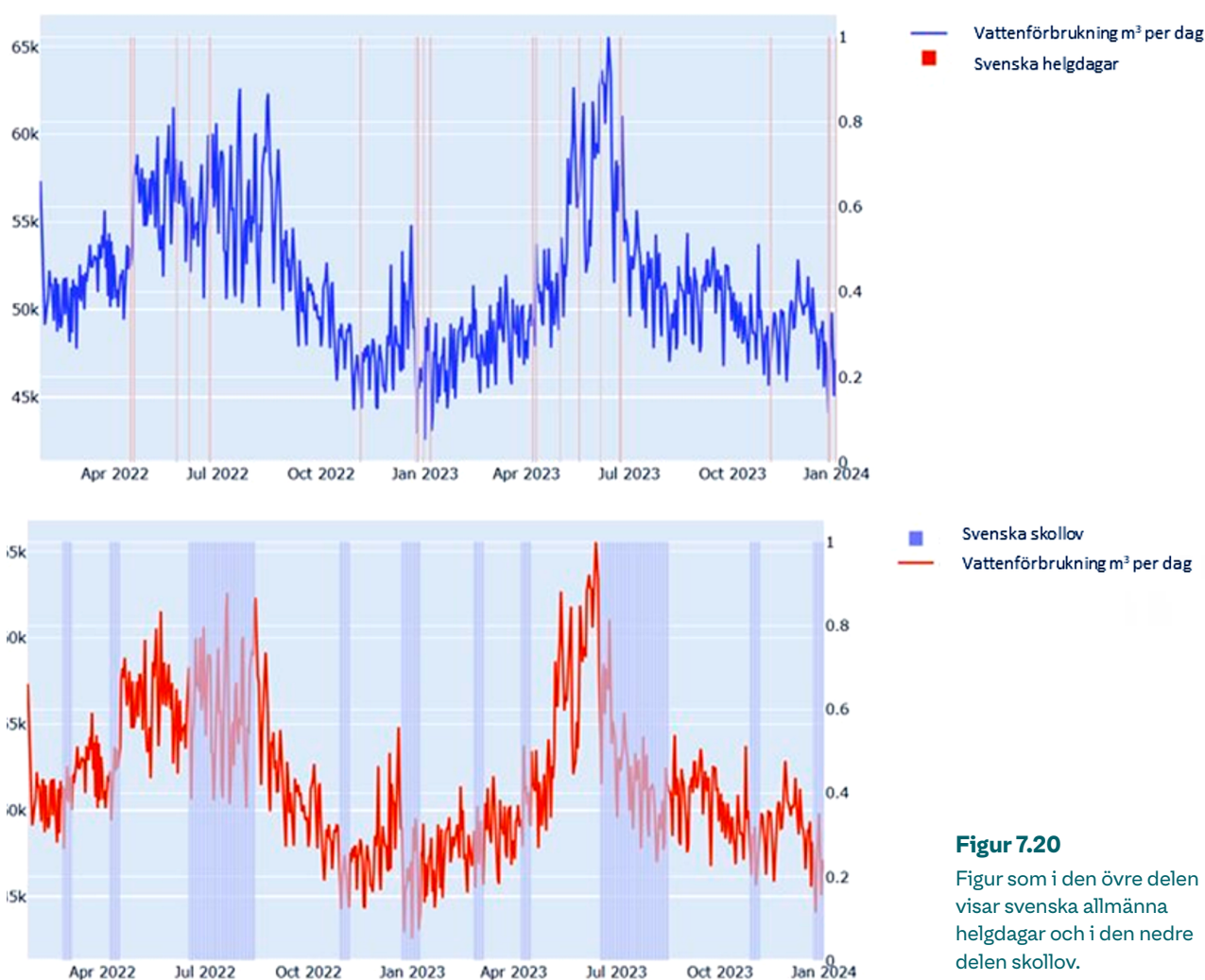


Figur 7.19

Figur som visar helgförbrukning gentemot total förbrukning. Överst ses en längre tidsperiod och längst ner en inzoomning som tydligt visar skillnader i förbrukning på veckodagar gentemot helger. Vänster y-axel är vattenförbrukning i tusen m³, höger y-axel noll till ett (funktion av eller på).

Allmänna helgdagar: I övre delen av Figur 7.20 ses hur allmänna helgdagar har överlagrats mot vattenförbrukningen. För ARIMA sågs en förbättring i prestanda med allmänna helgdagar som en variabel, inte lika stor som för helger men nära. För Prophet uppnåddes jämförbara resultat mot tidigare bästa resultat inkluderande en variabel (skalad för max/min-värden). XGBoost visade en liten förbättring mot resultat uppnådda med en variabel, medan SVR visade en försämring i prestanda.

Skollov: I undre delen av Figur 7.20 ses hur skollov har överlagrats mot vattenförbrukningen. I figuren ses ingen uppenbar korrelation mot vattenförbrukningen. Vi förväntade oss därmed inga förbättringar i prestanda vilket också var resultatet för samtliga modeller.



Figur 7.20
Figur som i den övre delen visar svenska allmänna helgdagar och i den nedre delen skollov.

I Tabell 7.17 till Tabell 7.20 ses resultatet för justering av indata gällande veckodagar, helgindikator samt skollov. Utvärderingstabeller också innehållande högst och lägsta MAPE-värde från varje enskild träningsperiod kan ses i Bilaga C.

Modell	ARIMA	PROPHET	XGB	SVR
MAPE	3,3 +/- 0,9	2,1 +/- 0,3	3,0 +/- 0,6	2,0 +/- 0,4

Tabell 7.17
Erhållet MAPE för "one hot encoded"-information om veckodagar som variabel

Modell	ARIMA	PROPHET	XGB	SVR
MAPE	2,90 +/- 0,41	2,35 +/- 0,36	2,45 +/- 0,44	2,7 +/- 0,5

Tabell 7.18
Erhållet MAPE för helgindikator som variabel

Modell	ARIMA	PROPHET	XGB	SVR
MAPE	3,02 +/- 0,63	2,52 +/- 0,31	2,11 +/- 0,33	4,4 +/- 1,9

Tabell 7.19
Erhållet MAPE för allmänna helgdagar som variabel.

Modell	ARIMA	PROPHET	XGB	SVR
MAPE	3,28 +/- 0,95	2,52 +/- 0,32	2,33 +/- 0,29	9,0 +/- 5,0

Tabell 7.20
Erhållet MAPE för skollov som variabel.

7.2.6 Iteration 4

För varje modell undersökte vi, baserat på resultaten från tidigare iterationer, vilka variabler som hade störst inverkan på prognosen. Från denna slutliga iteration fastställs de bästa variablerna och hyperparametrarna för våra undersökta algoritmer. Observera att det inte bör antas att effekten av flera variabler är additiv, vi kan se en nedgång i prestanda över modellerna när de kombineras. Dessutom kan en kombination av funktioner som visat sig minska modellens prestanda individuellt ha en positiv effekt när de kombineras. En sådan undersökning skulle dock kräva flera hundratals modellkörningar extra och bedöms ligga utanför ramarna för detta projekt, speciellt med avseende på att god prestanda redan har uppnåtts.

Värt att poängtera är också att detta projekt är drivet enligt CRISP-DM-metodiken vilket innebär att alla iterationer och vidareutveckling har bedrivits i enlighet med målet för projektet. Detta är alltså inte en heltäckande undersökning om alla algoritmer och alla variabler som har genomförts under projektets gång.

ARIMA: De bästa funktionerna när de undersöktes individuellt för ARIMA var rullande medelvärde av soltimmar (14 dagar), temperatur delad runt 0/4 grader, helger och allmänna helgdagar. När vi undersökte dessa kollektivt med hjälp av auto ARIMA upptäckte vi att modellens träningsprocess inte kunde slutföras. Beräkningstider för maskininlärningsalgoritmer kan plötsligt öka på grund av flera faktorer, inklusive ökande datamängder, komplexiteten hos de använda algoritmerna, och ineffektiv hantering av resurser som minne och processorkraft. Det kan ibland vara tidskrävande att identifiera den exakta orsaken till dessa ökningsar. Utifrån detta, tidigare resultat samt beräkningstider för en modell i produktion beslutade vi att inte fortsätta undersöka ARIMA-modellen.

Prophet: De bästa funktionerna när de undersöktes individuellt för Prophet var nederbörd, veckodag och helger. Veckodagar visade sig ha den mest signifikanta förbättringen av modellen, medan effekten av helger och nederbörd var marginell. Först undersöktes alla variablerna tillsammans, därefter plockades en variabel i taget bort ur uppsättningen. Genom denna undersökning fann vi att blandningen av variabler förbättrade den ursprungliga modellen baserad på endast vattenförbrukning. Inkluderingen av nederbörd och helger gav marginellt sämre resultat än att bara inkludera veckodagar. Slutsatsen blev därmed att den bästa Prophet-modellen erhöles genom att använda historisk vattenförbrukning i kombination med en märkning av veckodagar. Resultat kan ses i Tabell 7.21 och Tabell 7.22.

Fristående variabel	Nederbörd Veckodag Helg	Utan helg	Utan nederbörd	Utan veckodag
PROPHET	3,2 +/- 0,26	3,2 +/- 0,9	2,3 +/- 0,3	3,8 +/- 1,6

Tabell 7.21

Erhållet MAPE för bästa undersökta kombination av variabler för Prophet.

Variabel	Endast förbrukning	Veckodag	Nederbörd	Helg
PROPHET	2,6 +/- 0,4	2,1 +/- 0,3	2,5 +/- 0,24	2,4 +/- 0,36

Tabell 7.22

Erhållet MAPE för bästa fristående variabler för Prophet

XGBoost: För XGBoost var den enda funktionen som förbättrade modellen, från den initiala univariata analysen, rullande medelvärde av soltimmar (14 dagar). Det fanns dock ett antal funktioner som gav jämförbara resultat. Därför beslutade vi att undersöka dessa tillsammans för att se om det fanns någon förbättring, men fann ingen. Dessa resultat redovisas inte.

SVR: De bästa funktionerna när de undersöktes individuellt för SVR var veckodag, sammanhängande torra (<5 mm), varma (>16 grader), soliga (>6 timmar) dagar och nederbörd. Veckodagar visade sig ha den mest betydande förbättringen för modellen. Enligt samma metodik undersöktes alla variabler tillsammans och därefter togs en variabel i taget bort. Genom denna undersökning fann vi att blandningen av variablerna försämrade modellens prestanda märkbart. Därmed drogs slutsatsen att den bästa SVR-modellen erhålls genom att använda historisk vattenförbrukning i kombination med veckodagar. Resultat kan ses i Tabell 7.23 och Tabell 7.24.

Variabler	Torra dagar Varma dagar Soliga dagar Veckodagar Nederbörd	- - - Veckodagar Nederbörd	Torra dagar varma dagar soliga dagar Veckodagar -	Torra dagar varma dagar soliga dagar - Nederbörd
SVR	3,3 +/- 0,8	3,2 +/- 0,9	3,4 +/- 0,7	3,8 +/- 1,6

Tabell 7.23

Erhållet MAPE för bästa undersökta kombination av variabler för SVR.

Variabel	Endast förbrukning	Veckodag	Nederbörd	Torra dagar Varma dagar Soliga dagar
MAPE	2,8 +/- 0,4	2,0 +/- 0,4	2,6 +/- 0,3	2,6 +/- 0,36

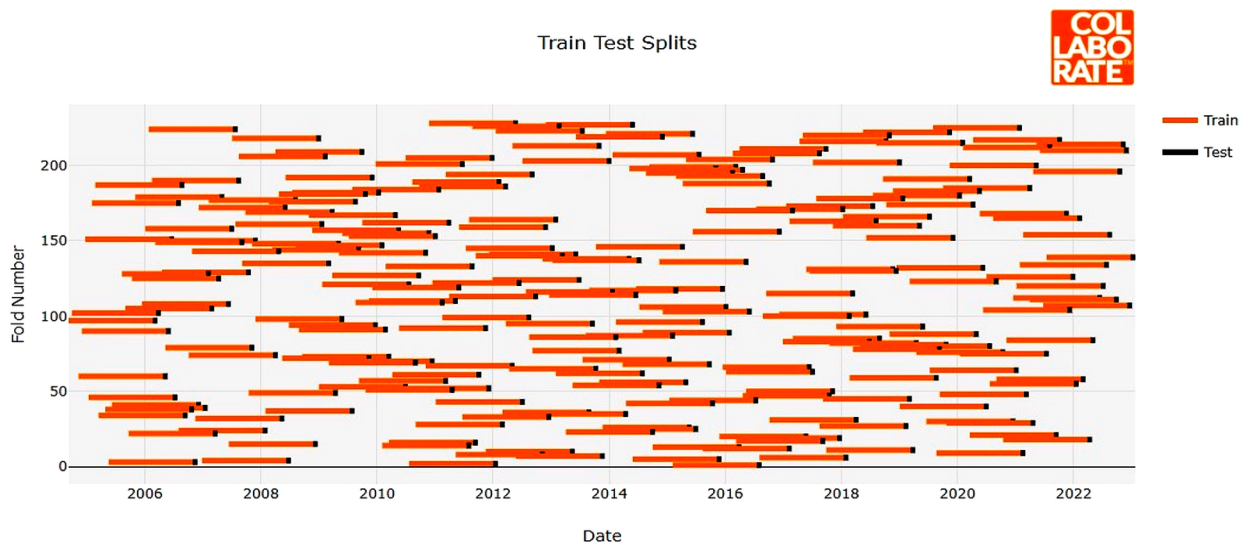
Tabell 7.24

Erhållet MAPE för bästa fristående variabler för SVR.

Sammanfattningsvis har vi undersökt påverkan av funktioner relaterade till efterfrågan på fyra olika modellarkitekturer och fastställt att traditionella ML-modeller, specifikt XGBoost och SVR gav bättre resultat än (S)ARIMAX och Prophet. Varje typ av ML-modell visade olika känslighet för de variabler som introducerades. En viktig slutsats är också denna: om målet kan uppnås med en tillräckligt enkel modell och metod är det troligen det bästa valet för ens tillämpning. Med detta i åtanke valdes den bästa modellen från varje arkitektur och tillämpades på ett större historiskt dataset.

7.2.7 Tillämpning på historiska data

För att utvärdera stabilitet, generaliserbarhet och överförbarhet i arbetet tillämpas modellerna på ett större, ovanligt dataset. Stabilitet i en modells bästa hyperparametrar är en indikation på att modellen har fångat och generaliserat datamönstren på ett tillfredsställande sätt. I detta fall använder vi samma hyperparametervärden och modellstruktur. Modellerna tillämpas på ett större historiskt dataset för vattenförbrukning, med dygnsvärden för en 18-årsperiod. Detta material har inte nyttjats tidigare av flera skäl, där det viktigaste skälet är de många förändringarna i vattendistributionssystemet under denna period. Under denna period har nya distributionsområden lagts till, mätmetodiken förändrats och både befolkningens mängden och individuell hushållsförbrukning ändrats. I Figur 7.21 ses hur data från det historiska datasetet delats in i olika, för datasetet representativa, träningsperioder. I detta försök vill vi undersöka om våra framtagna bästa modeller klarar av att hantera dynamiken i dessa förändringar och ändå leverera ett pålitligt resultat.



Figur 7.21
Använda träningsperioder
för det historiska datasetet
för vattenförbrukning.

Både ARIMAX- och Prophet-modellerna visade stor variation i felmått MAPE och MAE, vilket visar att dess hyperparametrar inte fullt ut lyckades fånga trender över tid för att kunna generalisera resultatet på ett tillräckligt bra sätt. Detta kan bero på att vi, i föregående steg, inte har haft tillräckligt mycket träningsdata som visar på förändringar i distributionssystemet vilket innebär att hyperparametrarna inte lyckas hantera de förändringar som skett i distributionssystemet över lång tid.

XGBoost visade stabila MAPE- och MAE-värden, vilket tyder på att hyperparametrarna och träningsdata var tillräckliga för att fånga de förändringar som finns över tid i det historiska datasetet. Härmed kan vi sammanfatta att XGBoost är den algoritm som passar behoven i detta projekt bäst.

7.2.8 Timvärden – prognosmodell

Under intervjuer/workshoppar med driftpersonal på Örbyfältet framkom att inga beslut gällande beställning av vatten in till fältet, eller beslut gällande driftstrategier togs baserat på timvärdesdata. Självfallet används timvärdesdata vid mer akuta händelser men i det stora hela tas inga övergripande beslut som kan leda till minskat vattenläckage från det geologiska magasinet på timvärden. Med detta som underlag har prognoser på timvärden inte varit mest centrala i detta projekt. Trots detta har en, icke lika omfattande, undersökning gjorts på ämnet. Målet är att undersöka parametrar och visa på komplexiteten i att jobba med data som innehåller större variation än dygnsvärdesdata.

Tre olika baslinjesmodeller (naiva modeller) skapades: en modell baserad på det senaste värdet, en på ett rullande medelvärde och en på ett rullande medianvärde. Baslinjesmodellerna för timvärden gav generellt högre MAPE-värden än baslinjesmodellerna för dygnsvärden. Modellen baserat på föregående steg hade ett MAPE-värde på 11,5 % +/- 6,3%, medelvärdesmodellen 8,9 +/- 3,6 % och medianvärdesmodellen 8,3 +/- 3,4 %.

Vi fann att Prophet-modellen inte slog baslinjesmodellerna. En djupare analys visade att Prophet-modellerna presterade relativt väl för dagtid men var sämre på att förut säga vattenförbrukningen under natten. Vidare arbete hade potentiellt kunnat förbättra resultatet men bedömdes inte vara motiverat eftersom XGBoost-modellerna presterade tillfredsställande. Förslagsvis kan vidare arbete inkludera användning av Prophets "change-point"-funktionalitet för att märka upp tider på dygnet som en fristående variabel. Ett alternativ kan vara att skapa en separat modell för dagtid/nattid. En annan möjlighet är att använda en sinus-/cosinustransformation eller radial-bas-funktion, som ofta används i tidsserier där ett cykliskt mönster upprepas regelbundet. Resultat kan ses i Tabell 7.25.

Månader	18	12	6	3	2	1
PROPHET	15,1 +/- 7,5	14,1 +/- 6,4	12,1 +/- 5,6	14,6 +/- 8,9	11,4 +/- 5,7	13,7 +/- 8,5
XGB	3,1 +/- 0,5	3,5 +/- 1,3	3,2 +/- 0,5	4,3 +/- 1,4	4,8 +/- 3	5,5 +/- 2,4

Vi undersökte också effekten av att inkludera fristående variabler såsom veckodagar och väder vilket gav försämrade resultat som följd. Resultat kan ses i Tabell 7.26.

Tabell 7.25

Erhållet MAPE för olika träningsperioder för 12-timmars prognos av timvärden för vattenförbrukning.

Variabel	Nederbörd	Temperatur	Soltimmar	Alla vädervariabler	Veckodag
PROPHET	18 +/- 11,0	17 +/- 11	17 +/- 11	14 +/- 9	15,0 +/- 9,0
XGB	3,7 +/- 1,4	5,4 +/- 2,0	4,1 +/- 1,6	5,4 +/- 1,7	4,4 +/- 1,8

7.2.9 Vattenförbrukningsprognoser – sammanfattning

Vi undersökte två olika datainsamlingsfrekvenser, timdata och dygnsdata, för att skapa två modeller för att prognosticera vattenförbrukning. Modellen för dygnsbaserad förbrukning är mest värdefull för kontexten och den grundläggande frågeställningen i projektet var på mest fokus lades på denna del.

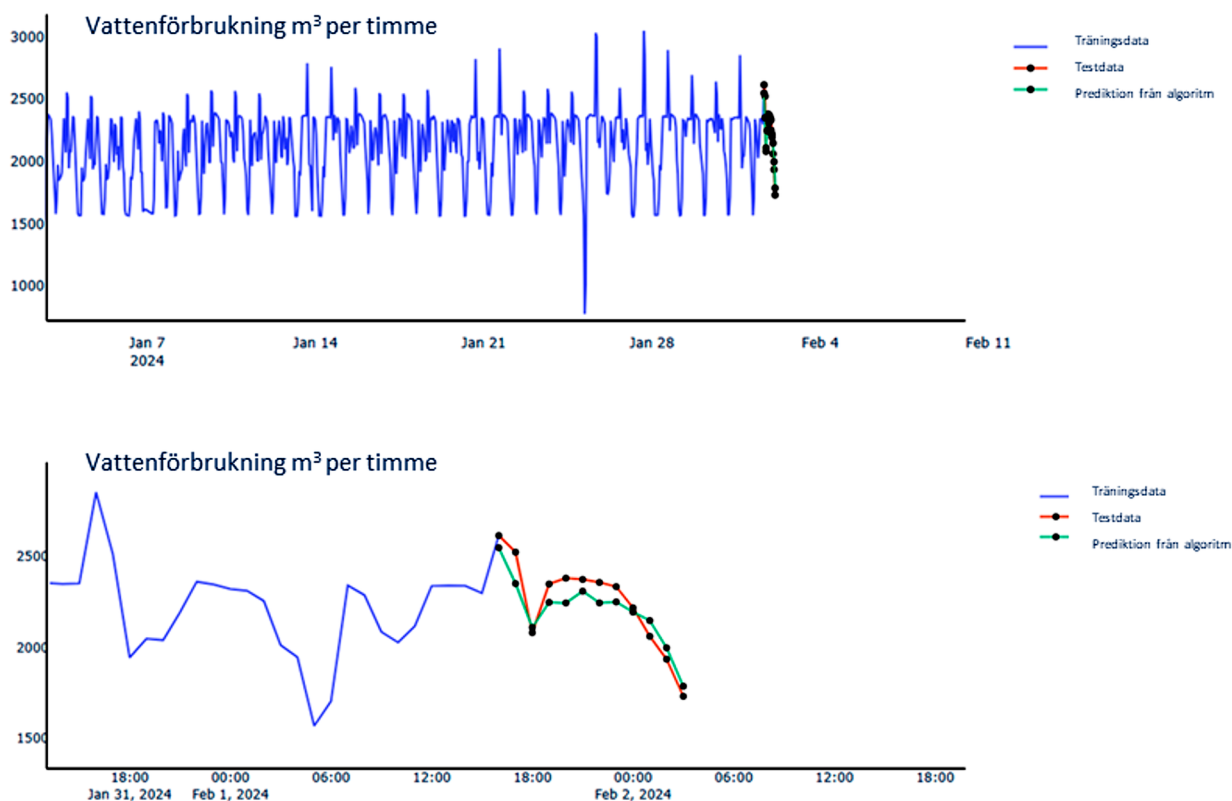
För varje dataset skapades en baslinjesmodell (naiv-modell). Hyperparametrar utforskades programmatiskt med grid search för XGBoost, SVR och Prophet. Grid search är en metod för att systematiskt prova olika kombinationer av hyperparametrar för att hitta den optimala konfigurationen för en modell. Auto ARIMA användes för ARIMAX-modeller samt callbacks för NN-baserade ML-modeller. Callbacks är en metod för att automatisera olika delar av träningsprocessen för NN, exempelvis gällande epoker, stopp av träning eller inlärningshastighet. För att ta del av specifika hyperparametrar använda för en viss algoritm uppmanas läsaren att ta kontakt med rapportförfattarna.

Vi upprepade experimenten med olika variabler, både baserat på en variabel (själva förbrukningen) och med flera fristående variabler exempelvis väderdata och dygnsdefinitioner. Vissa av dessa variabler konstruerades utifrån tillgängliga data.

Modell baserad på timvärden: För timvärdesmodellerna användes Prophet samt XGBoost. Prophet lyckades inte slå baslinjesmodellerna och får därmed anses vara ett dåligt resultat. Resultatet från XGBoost gav något lägre noggrannhet jämfört med modellerna som baserades på dygnsvärden. XGBoost tillämpad utan fristående variabler levererade det bästa resultatet med ett MAPE på cirka 3,1 %. I Figur 7.22 ses resultat för en 12-timmar prognos av vattenförbrukningen i förhållande till träningsdata.

Tabell 7.26

Erhållet MAPE för 12-timmar prognos av timvärden för vattenförbrukning med fristående variabler.



Figur 7.22

Resultat från 12-timmarsprognosmodell med XGBoost. Överst en översikt och längst ner i detalj. Den gröna linjen visar prognosvärde, den blåa träningsdata och den röda utvärderingsdata som undanhållits från modellen under träning. Observera att skalan inte går ner till noll på x-axeln. Detta för att tydliggöra skillnaderna.

Modell baserad på dygnsvärden: För dygnsvärdesmodellerna testades baslinjesmodellerna mot en traditionell statistisk metod ((S)ARIMA(X)), en hybridstatistisk ML-metod (Prophet) och rena ML-metoder, både traditionella (XGBoost och SVR) samt NN-baserade modeller (CNN, GRU, LSTM samt AR-LSTM).

För varje modellarkitektur undersöktes effekten av längden på träningsperioden, fristående variabler, konstruerade variabler samt automatisk justering av vissa hyperparametrar. Träningsperioden undersöktes genom korsvalidering av ett fast rullande fönster. Alla undersökta modeller slog baslinjesmodellen. Vi fann att traditionella ML-metoder gav bäst resultat, särskilt presterade XGBoost bäst.

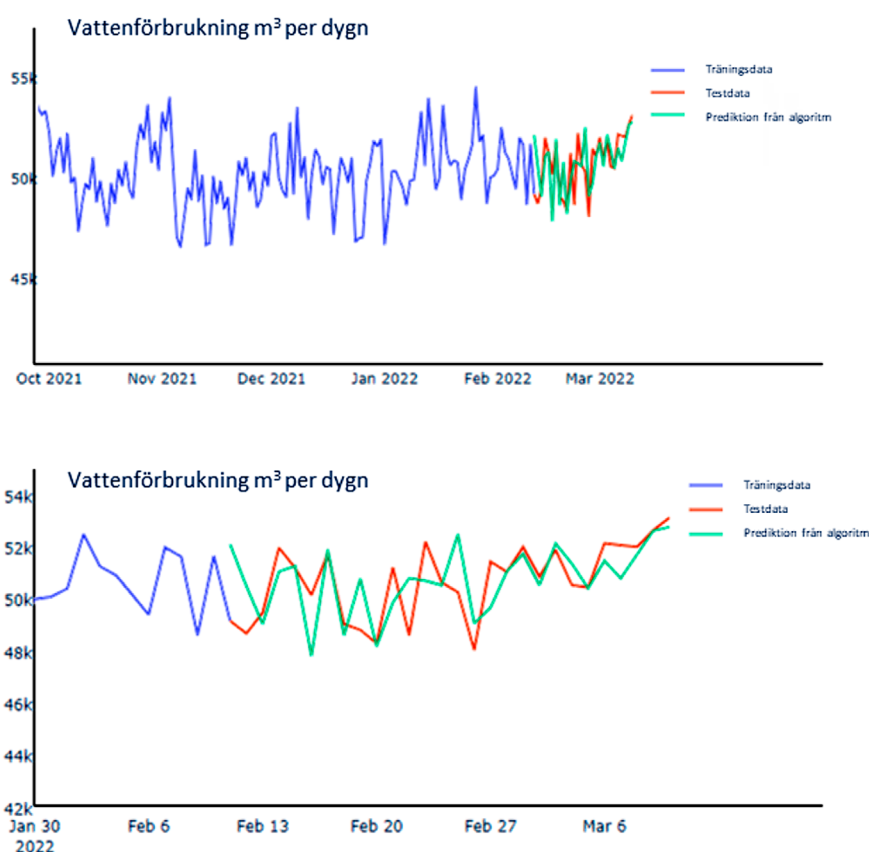
NN-modeller har andra datakrav än statiska metoder och traditionella ML-modeller. Flertalet av de undersökta NN-modellerna gav resultat som var jämförbara med vissa av de traditionella ML-modellerna. Givet de initiala resultaten fokuserades undersökningen på traditionell ML eftersom de är enklare att träna och har en interoperabilitet som neurala nätverk saknar.

Vi fann att de flesta modellarkitekturer gynnades av min/max-skalning och att de viktigaste variablerna varierade mellan olika ML-modeller. Det är inte nödvändigtvis alltid bäst att använda alla tillgängliga data. När mer data läggs till ökar komplexiteten vilket gör förståelse och utvärdering svårare och mer tidskrävande. En generell rekommendation är att använda den modell som är enklast och ger ett tillräckligt gott resultat för det övergripande målet samt att jobba iterativt för att komma fram till vad denna modell ska innehålla.

Något som är viktigt att poängtera är att beakta konsekvenserna av att kombinera modeller/modellresultat. Används exempelvis resultat från en väderprognos (vilket är en modell) riskerar en felaktig väderprognos att bidra till större osäkerhet i en vattenförbrukningsprognos. Används dessutom väderprognosdata som träningsdata till modellen kan samma typ av felaktighet riskera att introducera brus och ge upphov till felaktiga korrelationer. Utvärdering av data och dess konsekvenser på resultat är viktigt för att avgöra vad som ska inkluderas.

Generellt fann vi att en utvärdering av vilka variabler som förbättrar/försämrar resultatet för en specifik ML-modell, och dess specifika hyperparametrar är nödvändig. Olika variabler, samt olika typer av feature engineering, ger helt enkelt varierande resultat beroende på vald algoritm. Vi såg också att de bästa resultaten, för dygnsbaserad dricksvattenförbrukning, erhöles utan inblandning av externa variabler. Detta innebär inte att det inte finns ett samband mellan exempelvis temperatur/soltid och dricksvattenförbrukning. Det innebär bara att modellens resultat inte förbättrades tillräckligt signifikant för att det ska vara värt att inkludera dessa data i slutmodellen. Det kan också vara så att mer tillgängliga data hade kunnat förbättra dessa resultat.

De bästa modellerna, gällande prognos för dygnsbaserad vattenförbrukning för 28 dagar framåt hade ett MAPE på cirka 2,5 %. Ett exempel på resultat från denna prognosmodell kan ses i Figur 7.23.



Figur 7.23

Resultat från 28 dagars prognosmodell för dygnsförbrukning med XGBoost. Överst en översikt och längst ner i detalj. Den gröna linjen visar prognosvärde, den blåa träningsdata och den röda utvärderingsdata som undanhållits från modellen under träning. Observera att skalan inte går ner till noll på x-axeln. Detta för att tydliggöra skillnaderna.

För de traditionella ML-metoderna fann vi att den mest lämpliga längden på träningsperiod var 18 månader. Detta var också den maximala träningsperioden vi hade möjlighet att använda utifrån tillgängliga data. Med 18 månaders träningsdata kunde vi testa modellerna på 6 månaders data. Genom uppdelning av tränings- och testdata över hela den tillgängliga datamängden kunde vi få till testdata som var representativa för hela datasetet. Vi kunde på detta sätt utvärdera att MAPE var konsekvent över de olika tillgängliga säsongerna. Hade fler testdata varit tillgängliga hade denna process underlättats.

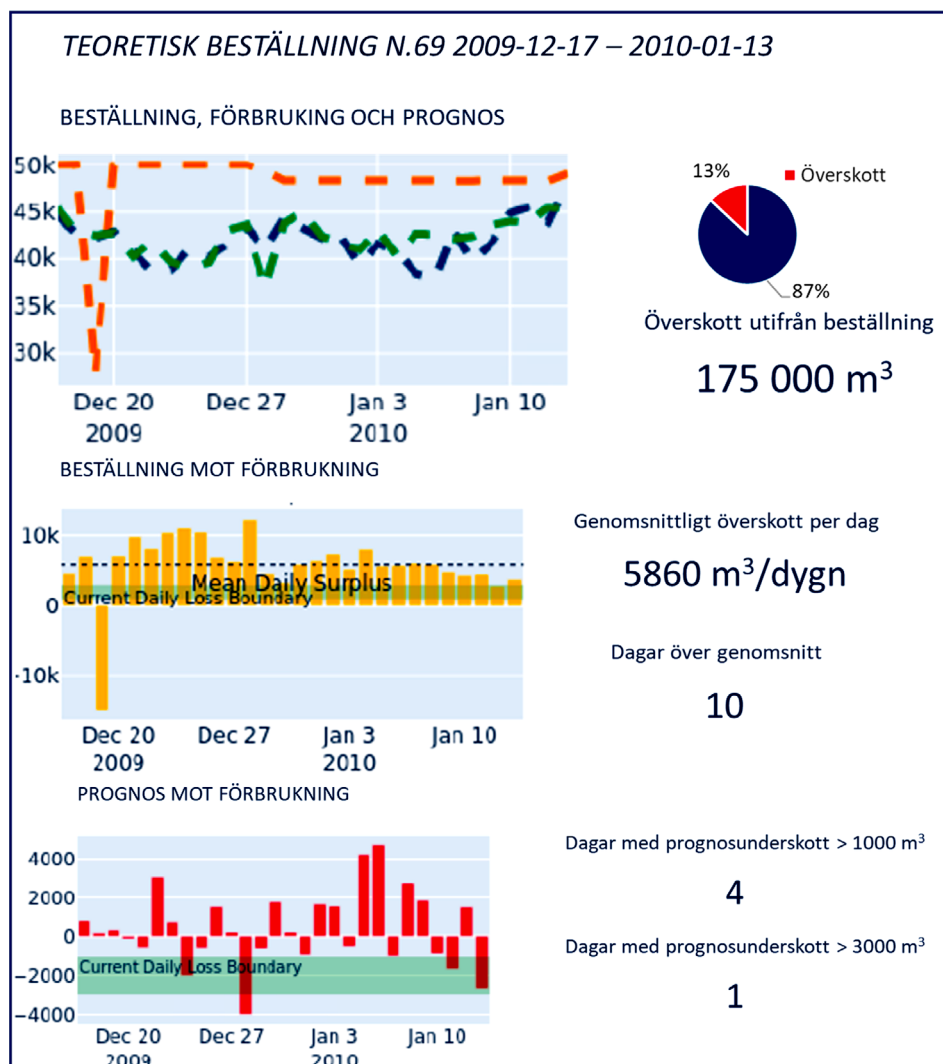
De modeller som gav bäst resultat tillämpades dessutom på det historiska datasetet innehållande nästan 20 års data. Här fann vi att XGBoost-modellen kunde generalisera bäst där andra modeller blev instabila, trots att vattenförbrukningen i det historiska datasetet förändras på grund av externa faktorer.

En längre träningsperiod hade troligen gynnat NN-baserade modeller och kunnat resultera i förbättringar. Men på grund av kommersiella och tidsmässiga begränsningar,

och de goda resultaten som uppnåddes med framför allt XGBoost baserat endast på vattenförbrukningen beslutades att vidare utveckling för andra ML-modeller inom projektet inte är nödvändig. Resultat från XGBoost-modellen kommer att användas och diskuteras i kommande avsnitt.

7.3 Om möjliga tillämpningar av framtaget material

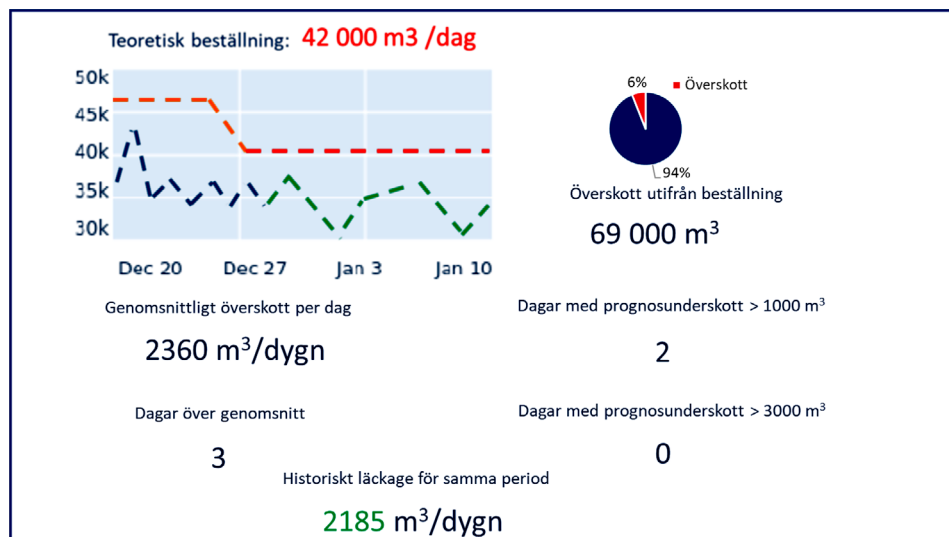
För att undersöka hur framtaget material skulle kunna vara till nytta för driftpersonalen har vi experimenterat med hur en framtagen prognosmodell kan användas för analys i samband med beställningar av vatten. Den tränade XGBoost-modellen kan prognosticera både lågförbrukning och högförbrukning väl, och för att vidareutveckla detta resultat skrevs en algoritm som beräknade hur många dagar det var ett underskott för prognosmodellen gentemot faktisk förbrukning. Resultat från denna algoritm, prognosmodellen, faktisk förbrukning samt beställd mängd vatten för de teoretiska beställningsdatumen kombinerades till en Business Intelligence-rapport (allmänt kallat BI-rapport) som först tillämpades på några av de teoretiska beställningarna. Ett exempel kan ses i Figur 7.24.



Figur 7.24

BI-analys av teoretisk historisk beställning där det i översta delen av figuren redovisas beställd mängd vatten (orange), faktisk förbrukning (blått) samt modellens prognos för samma tid (grönt). I nästa del redovisas överskottet av beställt vatten gentemot faktisk förbrukning, och i nedersta delen differensen mellan modellens prognos och faktisk förbrukning. Gränsvärden bör diskuteras och utvärderas med domänämnesexperter.

I Figur 7.25 redovisas analys av en *teoretisk* beställning. I detta förslag redovisas hur information kan erhållas, utifrån driftdata och prognoser från ML-modell, och bearbetas för att agera som ett beslutsstödsystem vid beställning av vatten. Observera att detta är ett teoretiskt förslag och att utvalda nyckeltal och rapportering behöver tas fram i samråd med domänämnesexperter.



Figur 7.25

Teoretisk skiss av ett analysverktyg som kan tillgängliggöras för driftoperatörer utifrån framtaget material, där historiska data visas i figuren samt den beställningsvolym som driftoperatören skriver in i verktyget. Observera att detta är ett förslag och att historiska siffror inte använts i förslaget.

Med denna typ av underlag finns möjlighet, ihop med domänämnesexperter, att ta fram ett underlag som kan göra att driftoperatörer kan ta mer informerade beslut om beställningar. Att dessutom kunna teoretiskt experimentera med ökad och minskad beställning och få en uppfattning om magnituden för överskott/underskott skulle kunna bidra till förändringar av driftstrategin som medger minskade förluster. Det är naturligt, och riktigt, att inte utgå helt från teoretiska beräkningar. En rekommendation är därmed att om detta system ska sättas i produktion bör en utvärderingsplan göras för hur mycket verktyget används, vilken påverkan det har och vilken förbättringspotential som finns. Det är också viktigt att beakta att framtagna algoritmer och framtagna beslutsstödsystem kontinuerligt bör utvärderas och utvecklas för att ge nytta och bära sina kostnader.

7.4 Modellering av grundvattennivåer

Inom projektet har ML tillämpats på datasetet för uppmätta grundvattennivåer. Både separat och i relation till pumpdata, inkommande vatten till fältet och utgående vatten från fältet. Detta för att få insikter om hur styrning av pumpar kan påverka läckaget från fältet. Att mäta förlusten av vatten från en akvifer är ingen enkel uppgift. Vetenskaplig litteratur i ämnet beskriver ML-applikationer på akviferer som mycket komplext. Utmaningarna är relaterade till att en akvifer är ett verkligt komplext system med många helt okända variabler. I detta fall har vi dessutom en akvifer som är starkt påverkad av mänsklig aktivitet med artificiell påfyllnad av vatten och väldigt många pumpar som jobbar enskilt, i grupper och över olika tidshorisonter. Lägg därtill att olika driftoperatörer under årens lopp har haft olika strategier för pumpning, något som inte finns dokumenterat i relation till tillgängliga data. Slutsatsen blir alltså att vi inte längre har att göra med ett naturligt system som har någon slags "steady-state"-funktion.

Som nämnts tidigare har det under de senaste åren installerats nya automatiserade sensorer för nivåmätning på Örbyfältet. Dessa mäter grundvattennivåer ett antal gånger per dygn och används för att bedöma fyllnadsgraden i fältet. En risk som driftoperatörer upplyste om är att dessa data kan ha en referensvärdesproblematik, vilket innebär

att de inte nödvändigtvis är korrekt kalibrerade i förhållande till ett koordinatsystems nollpunkt. Det är, och har varit, ett pågående arbete under de senaste åren. Projektet har därmed använt skillnaden i uppmätta nivåer. Om det sker en kalibrering av mätarna i efterhand bör vi då kunna relatera skillnaden i nivåer till nya referensvärden. Tidigare mätningar har historiskt gjorts med manuell lodning varpå frekvensen av mätningar var ojämn och gles även om driftoperatörerna var tydliga med att nivån i fältet förändras långsamt, över dagar och veckor och generellt inte över timmar. Projektet valde att initialt gå vidare med de senaste årens mer högupplösta nivåmätningar av två anledningar. Dels fanns där en kontinuitet i data vilket gjorde datasetet enklare att hantera, dels fanns möjligheten att diskutera händelser i data med de driftoperatörer som använt data.

Trots automatiserad insamling av nivådata får det, ur ett datavetenskapsperspektiv, sägas att data var glesa. Initialt undersöktes datasetet på samma sätt som vattenprognosmodellerna hade undersökts, där grundvattennivåer och beställt vatten användes som variabler. Vi testade olika varianter av NN-baserade modeller, statistiska metoder och skapade en baslinjesmodell.

Det visade sig att varken de statistiska metoderna eller NN-modellerna presterade bättre än baslinjesmodellen. Utifrån att gleshet, och de till synes slumpmässiga nivåförändringarna, påverkades av mänskliga beslut var det inte särskilt förvånande att vi inte kunde uppnå ett tillfredsställande resultat.

Ett inledningsvis dåligt resultat är dock ingen garanti för att ett system inte kan modelleras. För att skapa mer insikter om data, och underlag för vidare utveckling av ML-algoritmer, vände vi oss till statistiska metoder för att identifiera samband mellan pumpar och grundvattennivåer. Projektet använde Granger-kausaltet som är en statistisk metod som avgör om en tidsserie kan förutsäga en annan (även om det inte garanterar ett verkligt samband). I Figur 7.26 visas, för två exempel, sambandet mellan pumpvolym och ett antal närliggande nivåsensorer samt Granger-kausaltet mellan nivåsensorerna. Från denna metod fann vi att elva av pumparna inte visade någon korrelation alls med grundvattennivån. Där vi såg korrelation var det mellan en och sju närliggande grundvattennivåsensorer. För pumpar med ett högt antal korrelationer fann vi att vissa var klustrade i samma områden medan andra visade korrelationer med alla andra. Detta ledde till slutsatsen att vi hade otillräckliga data för att avgöra samband.

8 Sammanfattning och råd

Genom både experiment och en granskning av vetenskaplig litteratur har vi visat att maskininlärning (ML) kan tillämpas på vattenproduktionssystem för att generera värdefulla insikter. ML-prognoserna presterade bättre än traditionella statistiska prognoser både vad gäller noggrannhet och beräkningstid. I denna studie undersöktes tillämpning av ML-modeller på tre huvudsakliga dataset: förbrukning på dygnsbaserad nivå, förbrukning baserad på timvärden samt baserat på grundvattennivåer. För varje scenario utvärderades flera olika modellarkitekturer innehållande statistiska modeller, traditionella ML-metoder samt neurala nätverk. Vår studie visade att varje arkitektur och val av algoritm har olika förutsättningar gällande hantering av data och användning av fristående variabler.

8.1 Prognosmodell för dygnsvärden och timvärden

Vattenförbrukningsprognoser kunde skapas med traditionella ML-algoritmer som är relativt lätta att träna, har låga behov gällande metadata och är interoperabla. Detta betyder att de framtagna modellerna är lätta att integrera mot andra system eller komponenter, en faktor som inte ska underskattas när färdiga modeller ska sättas i produktion.

Vi fann att XGBoost presterade bäst av de testade ML-algoritmerna, inte bara genom de bästa felmåten (MAPE och MAE) utan också genom dess möjlighet att fånga toppar och bottnar i förbrukningsprognoserna. Ofta är just toppar i förbrukningen av extra stort intresse gällande vattenförbrukning. Det visades sig också att samma modell bibehöll sin prestanda när den tillämpades på ett historiskt dataset utan större förändringar i modellarkitektur eller av hyperparametrar, vilket indikerar att algoritmen har goda förutsättningar att tillämpas i flera olika vattendistributionssystem.

Ytterligare utforskande arbete hade troligen kunnat förbättra resultatet för även andra modeller. Det är dock osäkert om ett bättre resultat skulle kunna uppnås varför det inte finns någon ekonomisk rimlighet att, i detta projekt, utforska vidare. Inom projektet har förslag på dashboards skapats som kan leverera insikter till driftoperatörer. Eftersom algoritmen också framgångsrikt lyckades fånga de förändringar som finns i vattenförbrukningen över lång tid finns också goda förutsättningar för att tillämpa resultatet på andra vattendistributionssystem vilket skulle kunna vara till stor nytta inom många organisationer.

Eftersom väldigt många olika modeller, med olika variabler, har tillämpats i detta projekt är det alltför opraktiskt att redovisa alla använda hyperparametrar. Om det finns en önskan att ta del av hyperparametrar för specifika algoritmer/modeller går det bra att kontakta rapportförfattarna.

8.2 Vattenförbrukning och vattenläckage

Modellen för vattenförbrukning användes för att skatta skillnaden mellan beställd mängd vatten och faktisk förbrukning samt jämföra detta med förbrukningsprognosen. Det kan visuellt observeras att förbrukningsprognosen fångar toppar i förbrukningen väl. Detta ger möjlighet att analysera beställningar av vatten utifrån läckage och önskad kapacitet. En viktig funktion för Örbyfältet är att skapa en buffert för vattendistributionssystemet för att kunna avlasta det regionala vattendistributionssystemet vid kapacitetsbrist, vid driftproblem eller i händelse av kris. Överskottet på beställningar bör därmed diskuteras med domänämnesexperter för att avgöra hur ett överskott kan minimeras inom acceptabla riskgränser.

8.3 Nivåer i fältet

På grund av komplexiteten med att modellera en kraftigt mänskligt påverkad akvifer togs beslutet att vi troligen inte hade tillräckligt med data för att avgöra om vi skulle kunna modellera grundvattennivåer på ett tillförlitligt sätt. Diskussioner om vidare undersökning fördes där beslutet togs att inte gå vidare på grund av risken att denna typ av modellering skulle kräva mer data än vad som var tillgängligt. Möjligheten att utföra vidare undersökningar på befintliga data, genom att exempelvis gå på djupet med neurala nätverk finns. Rekommendationen i denna rapport är dock att mer data, med högre frekvens, ska samlas in och att modellundersökningen återupptas efter att tillräckligt med data samlats in. Nya framsteg inom sensorteknik har visat att flödet (flux) inom en akvifer kan mätas vilket också är en möjlig väg framåt för ökad förståelse av systemet.

8.4 Om ML-projekt

Hur väl ett ML-projekt lyckas beror på flera faktorer:

Först och främst är tillgången till data av god kvalitet, tillräcklig längd och frekvens, avgörande.

För det andra är det viktigt att identifiera vilka som är domänämnesexperter för de system eller de data som undersöks. När domänämnesexperter har identifierats är det avgörande att dra nytta av deras kunskap om systemet för att involvera denna i modellutvecklingen. Deras kunskap är till hjälp, inte bara i uppbyggnadsfasen, utan även vid tolkning av resultatet, vilket är särskilt viktigt för komplexa system.

För det tredje bör man fundera över systemets komplexitet. Är det ett komplext system, och i så fall hur komplext? Generellt kan man säga att desto mer komplext det system som ska modelleras är, desto mer data och hjälp av domänämnesexperter behövs.

För det fjärde bör modellutveckling bedrivas som en iterativ och utvärderande process som inkluderar förståelse för verksamheten, det faktiska behovet (affärsnyttan), förståelse för data, modellering och utvärdering. Det är viktigt för att inte bara fastställa den bästa modellen (eller uppsättning av modeller) samt att optimera dessa modeller så att de presterar så bra som möjligt, samtidigt som överanpassning undviks.

Beräkningstiden för modellering är en viktig faktor. För att inte bara skapa förståelse utan också praktisk daglig nytta behöver en modell kunna fungera i produktion, vilket ställer krav på modellens körtid. I detta projekt avbröt vi vissa experiment på grund av att beräkningstiden var för lång eller att minneskraven var för höga. Vi fann att ARIMA-modeller i synnerhet led av hög minnesanvändning, vilket resulterade i att beräkningsdatorer stängdes ner. Andra experiment var genomförbara gällande minneskrav men beräkningstiderna var för långa och flera undersökningar avbröts efter att beräkning pågått i 200 timmar. Detta gällde särskilt för modellerna där flera variabler användes.

Dessutom bör hänsyn tas till delning av modellens resultat inom den större organisationen. Detta kräver någon form av modelldistribution eller spridning, vilket vanligtvis sker genom skapandet av webbapplikationer eller mer begränsade BI-dashboards. För att få en god avkastning på investeringar i ML-projekt bör projekt baseras på ett MLOPs-baserat tillvägagångssätt där utvecklingen av specifika ML-modeller hanteras systematiskt snarare än enskilt så att verktyg kan återanvändas och delas vilket gör framtida arbete snabbare, billigare och mer pålitligt.

Ett stort diskussionsämne inom vattensektorn rör bristen på kompetens gällande programmering och mjukvaruutveckling. Dels för arbete inom ML/AI, dels för generell dataanalys och utveckling. Något som också är värt att beakta är att länken mellan domänämnesexpertis och ML/MLOPs-expertis är viktig. Denna kompetens möjliggör att extern detaljkunskap, inom exempelvis datavetenskap, kan nyttjas på ett mer effektivt sätt. Genom att fokusera på denna länk och att arbeta systematiskt finns goda möjligheter

att ta in mer av den kompetens som saknas till vattenbranschen från externa källor och samarbeten.

En slutsats från detta projekt är att det krävs ett ramverk och en infrastruktur för djupdykande arbete inom ML, framför allt om inte bara utforskande arbete utan även affärsnytta ska uppnås. Ett angreppssätt där ML-arbete behandlas generellt och inte bara för enskilda fall är något som går att samarbeta kring. Ett sådant tillvägagångssätt inom den svenska vattensektorn skulle kunna vara, och borde vara, möjligt att samarbeta inom vilket skulle göra svensk vattensektor till en föregångare inom tillämpning av maskininlärning.

Som en potentiell uppföljning på detta projekt pågår arbete med att tillgängliggöra framtagna kod för fler användare och bolag. Hur detta ska göras på effektivaste sätt är ett spännande kommande arbete.

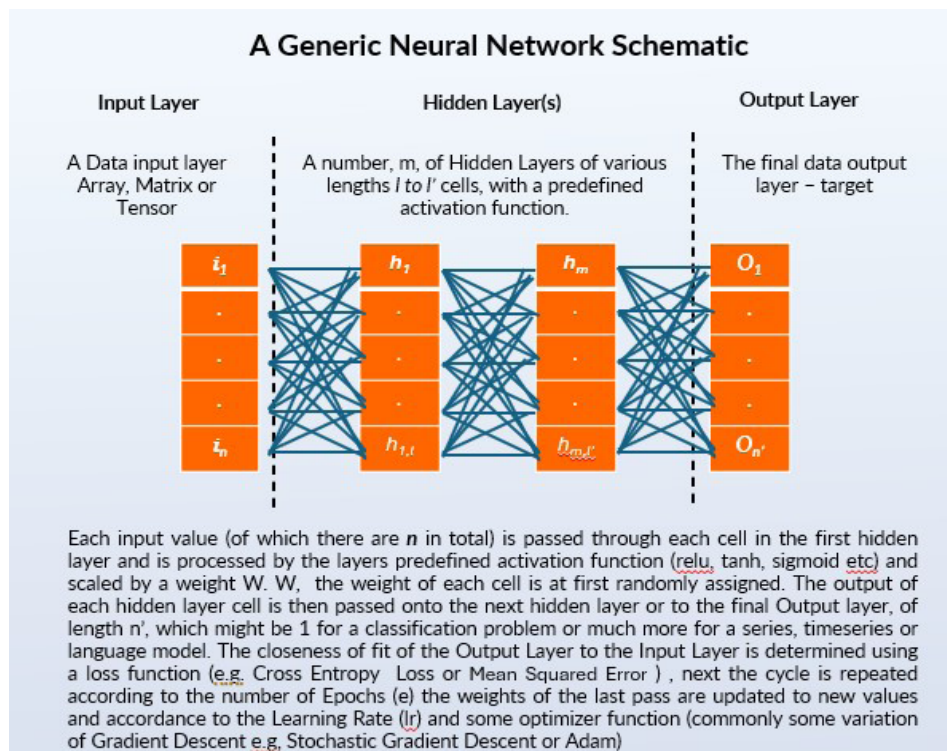
Referenser

- Amershi S. et al. (2019). Software Engineering for Machine Learning: A Case Study. *Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291-300.
- Bar-Yam Y. (2003). *Dynamics of Complex Systems*. Perseus Books.
- Beck K. et al. (2001). *Manifesto for agile software development*. Agile Alliance.
- Biamonte J., Wittek P., Pancotti N., Jong D., Montangero S. och Faccin M. (2017). Quantum machine learning. *Nature*, 549(7671), 195-202.
- Bishop C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
- Box G., Jenkins G. och Reinsel G. (2008). *Time Series Analysis*. John Wiley & Sons.
- Breiman L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.
- Breiman L., Friedman J. H., Olshen R. A. och Stone C. J. (1984). *Classification and Regression Trees*. Wadsworth International Group.
- Brown T. et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877-1901.
- Bunmahotama W., Hung W.-N. och Lin T.-F. (2017). Prediction of the adsorption capacities for four typical organic pollutants on activated carbons in natural waters. *Water Research*, 111, 28-40.
- Casella G. och Berger R. L. (2002). *Statistical Inference*. 2nd Edition. Duxbury Press.
- Chandola V., Banerjee A. och Kumar V. (2009). Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3), Article 15.
- Cheng T. et al. (2020). Forecasting of wastewater treatment plant key features using deep learning-based models: A case study. *IEEE Access*, 8, 184475-184485.
- Choi T. M., Wallace S. W. och Wang Y. (2016). *Big Data Analytics in Supply Chain Management*. Springer.
- Cover T. och Hart P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21-27.
- Crevier D. (1993). *AI: The Tumultuous History of the Search for Artificial Intelligence*. Basic Books.
- Davenport T. H. och Harris J. G. (2007). *Competing on Analytics: The New Science of Winning*. Harvard Business Review Press.
- Davenport T. H. och Kim J. (2013). *Keeping Up with the Quants: Your Guide to Understanding and Using Analytics*. Harvard Business Review Press.
- Gelman A., Carlin J. B., Stern H. S., Dunson D. B., Vehtari A. och Rubin D. B. (2013). *Bayesian Data Analysis* (3rd ed.). CRC Press.
- Gershenfeld N. (1999). *The Nature of Mathematical Modeling*. Cambridge University Press.
- Goodfellow I., Bengio Y. och Courville A. (2016). *Deep Learning*. MIT Press.
- Hamilton J. (1994). *Time Series Analysis*. Princeton University Press.
- Hashem I. A. T., Yaqoob I., Anuar N. B., Mokhtar S., Gani A. och Khan S. U. (2015). The rise of big data on cloud computing: Review and open research issues. *International Journal of Information Management*, 35(3), 137-150.
- Hastie T., Tibshirani R. och Friedman J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

-
- Hinton G. E., Osindero S. och Teh Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), 1527–1554.
- Holland J. H. (2014). *Complexity: A Very Short Introduction*. Oxford University Press.
- Huang Y., Wang C., Wang Y., Lyu G., Lin S., Liu W., Niu H. och Hu Q. (2024). Application of machine learning models in groundwater quality assessment and prediction: progress and challenges. *Front. Environ. Sci. Eng.*, 18(3), 29.
- Hunt E. B., Marin J. och Stone P. J. (1966). *Experiments in Induction*. New York: Academic Press.
- Hyndman R. J. och Athanasopoulos G. (2018). *Forecasting: Principles and Practice* (2nd ed.). OTexts.
- Johnson S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3), 241–254.
- Jolliffe I. T. (2002). *Principal Component Analysis*. Springer Series in Statistics.
- Kauffman S. (1995). *At Home in the Universe: The Search for the Laws of Self-Organization and Complexity*. Oxford University Press.
- Khan J., Lee E., Balobaid A. S. och Kim K. (2023). A Comprehensive Review of Conventional, Machine Learning, and Deep Learning Models for Groundwater Level (GWL) Forecasting Prediction. *Appl. Sci.*, 13(4).
- Kotsiantis S. B., Kanellopoulos D. och Pintelas P. (2006). Data mining: A review of the state-of-the-art. *Artificial Intelligence Review*, 26(3), 159–190.
- Krizhevsky A., Sutskever I. och Hinton G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25, 1097–1105.
- LeCun Y., Bengio Y. och Hinton G. (2015). Deep Learning. *Nature*.
- Li L., Rong S., Wang R. och Yu S. (2021). Recent advances in artificial intelligence and machine learning for nonlinear relationship analysis and process control in drinking water treatment: A review. *Chemical Engineering Journal*, 405, 126673.
- Lipton Z. C. (2018). The Mythos of Model Interpretability. *Communications of the ACM*.
- Lorenz E. N. (1993). *The Essence of Chaos*. University of Washington Press.
- MacQueen J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1, 281–297.
- Marcus G. (2020). The Next Decade in AI: Four Steps Towards Robust Artificial Intelligence.
- McAfee A., Brynjolfsson E., Davenport T. H., Patil D. J. och Barton D. (2012). Big Data: The Management Revolution. *Harvard Business Review*.
- Mitchell M. (2009). *Complexity: A Guided Tour*. Oxford University Press.
- Molnar C. (2020). *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. Christoph Molnar.
- Moore D., McCabe G. och Craig B. (2016). *Introduction to the Practice of Statistics*. W H Freeman & Co.
- Murphy K. P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.
- Nash J. E. och Sutcliffe J. V. (1970). River flow forecasting through conceptual models part I—A discussion of principles. *Journal of Hydrology*, 10(3), 282–290.
- Niknam A. et al. (2022). A Critical Review of Short-Term Water Demand Forecasting Tools—What Method Should I Use? *Sustainability*, 14(9), 5412.
-

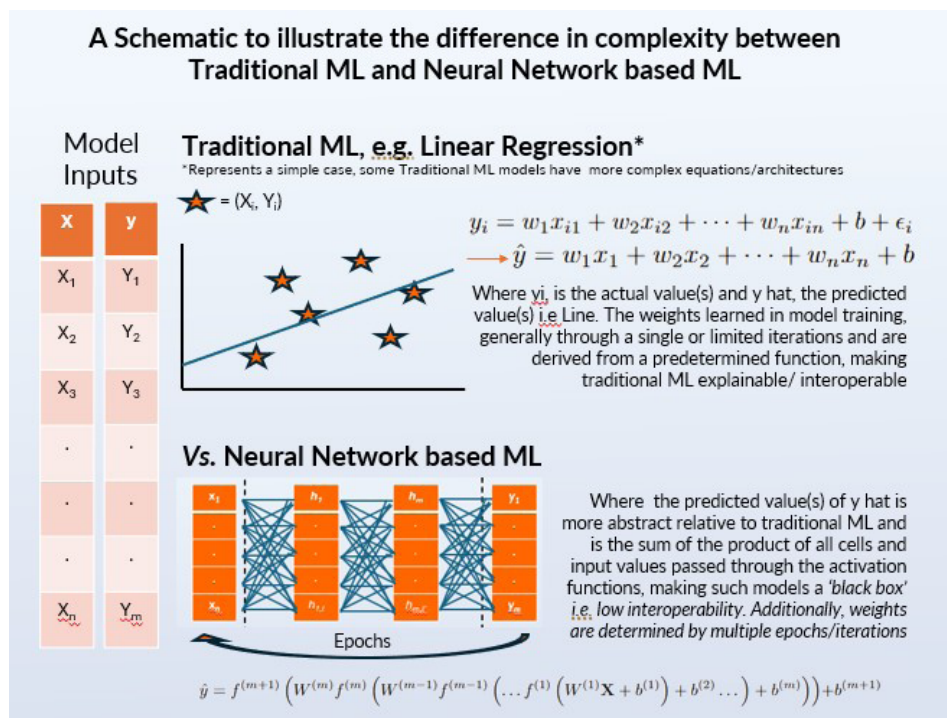
-
- Olden J. D. och Jackson D. A. (2002). Illuminating the “black box”: a randomization approach for understanding variable contributions in artificial neural networks. *Ecological Modelling*, 154(1-2), 135-150.
- Page S. E. (2011). *Diversity and Complexity*. Princeton University Press.
- Park S. et al. (2019). Deep neural networks for modeling fouling growth and flux decline during NF/RO membrane filtration. *Journal of Membrane Science*, 587, 117164.
- Peiris R. H. et al. (2012). Fluorescence-based fouling prediction and optimization of a membrane filtration process for drinking water treatment. *AIChE Journal*, 58(5), 1475-1486.
- Provost F. och Fawcett T. (2013). *Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking*. O'Reilly Media.
- Rall D. et al. (2020). Simultaneous rational design of ion separation membranes and processes. *Journal of Membrane Science*, 600, 117860.
- Ricci F., Rokach L. och Shapira B. (2015). *Recommender Systems Handbook* (2nd ed.). Springer.
- Rolnick D. et al. (2022). Tackling Climate Change with Machine Learning. *ACM Computing Surveys*.
- Rumelhart D. E., Hinton G. E. och Williams R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536.
- Sculley D. et al. (2015). Hidden Technical Debt in Machine Learning Systems. *Advances in Neural Information Processing Systems*, 28, 2503-2511.
- Shearer C. (2000). The CRISP-DM model: The new blueprint for data mining. *Journal of Data Warehousing*, 5(4), 13-22.
- Song Y., Dhariwal P., Chen M., Sutskever I. och Ermon S. (2024). Consistency Models. *arXiv preprint arXiv:2303.01469*.
- Taleb N. N. (2010). *The Black Swan: The Impact of the Highly Improbable*. Random House.
- Turk D., France R. och Rumpe B. (2005). Limitations of Agile Software Processes. *Proceedings of the Third International Conference on Extreme Programming and Agile Processes in Software Engineering (XP 2002)*, Lecture Notes in Computer Science, vol 2675. Springer, Berlin, Heidelberg. doi:10.1007/3 540 45672 4_3.
- Tukey J. (1977). *Exploratory Data Analysis*. Addison-Wesley Publishing Company.
- Vapnik V. N. (1995). *The Nature of Statistical Learning Theory*. Springer.
- Wasserman L. W. (2004). *All of Statistics: A Concise Course in Statistical Inference*. Springer.
- Xiao Z. et al. (2022). Impacts of data preprocessing and selection on energy consumption prediction model of HVAC systems based on deep learning. *Energy and Buildings*, 258, 111832.

Bilaga A Schematiska beskrivningar av neurala nätverk



Figur A.1

Mer detaljerad schematisk beskrivning av ett neuralt nätverk.



Figur A.2

Mer detaljerad schematisk beskrivning av skillnaderna mellan traditionell ML och NN.

Bilaga B Matematiska sammanfattningar av utvärderade algoritmer

(S)ARIMAX

SARIMAX-modellen är en utökning av ARIMA (AutoRegressive Integrated Moving Average) som inkluderar säsongskomponenter och exogena regressorer. Modellen används för att prognostisera tidsseriedata med både icke-säsongsmässiga och säsongsmässiga mönster, och kan även inkludera externa variabler.

Modellkomponenter:

SARIMAX-modellen kan uttryckas som:

$$y_t = \phi(B)\Theta(B^s)[\Phi(B^s)\Theta(B^s)y_t - \mu] + \varepsilon_t$$

Där:

- y_t är den observerade tidsserien.
- $\phi(B)$ är den icke säsongsmässiga autoregressiva (AR) polynomen
- $\Theta(B)$ är det icke säsongsmässiga rullande medelvärdespolynomen (MA).
- $\Phi(B^s)$ är den säsongsmässiga autoregressiva polynomen.
- $\Theta(B^s)$ är den säsongsmässiga glidande medelvärdespolynomen.
- B är backshift-operatorn ($B^k y_t = y_{t-k}$).
- μ är medelvärdet av serien (ofta satt till 0 för att förenkla).
- ε_t är feltermen, vanligtvis modellerad som vitt brus.

Icke-säsongsmässig komponent:

$$\phi(B)(1 - B^p)y_t = \varepsilon_t + \Theta(B)\varepsilon_t$$

Där:

- $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p$ är den icke-säsongsmässiga AR polynomen av ordning p .
- $\Theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \dots + \theta_q B^q$ är den icke-säsongsmässiga MA-polynomen av ordning q .

Säsongsmässig komponent:

$$\Phi(B^s)(1 - B^s)^D y_t = \varepsilon_t + \Theta(B^s)\varepsilon_t$$

Där:

- $\Phi(B^s) = 1 - \phi_1 B^s - \phi_2 B^{2s} - \dots - \phi_P B^{Ps}$ är den säsongsmässiga AR polynomen av ordning P .
- $\Theta(B^s) = 1 + \theta_1 B^s + \theta_2 B^{2s} + \dots + \theta_Q B^{Qs}$ är den säsongsmässiga MA polynomen av ordning Q .
- D är den säsongsmässiga differensieringsordningen.
- s är längden på den säsongsmässiga cykeln.

Exogena regressorer

Om exogena variabler x_t inkluderas, utökas modellen till:

$$y_t = \phi(B)\Theta(B^s)[\Phi(B^s)\Theta(B^s)y_t - \mu] + \beta x_t + \varepsilon_t$$

Där:

- β är vektorn av koefficienter för de exogena regressorerna x_t .

Sammanfattning av viktiga ekvationer:

Helhetsmodellen:

$$y_t = \phi(B)\Theta(B^s)[\Phi(B^s)\Theta(B^s)y_t - \mu] + \varepsilon_t$$

Icke-säsongsmässig komponent:

$$\phi(B)(1 - B^p)y_t = \varepsilon_t + \Theta(B)\varepsilon_t$$

Säsongsmässig komponent:

$$\Phi(B^s)(1 - B^s)^D y_t = \varepsilon_t + \Theta(B^s)\varepsilon_t$$

Med exogena regressorer:

$$y_t = \phi(B)\Theta(B^s)[\Phi(B^s)\Theta(B^s)y_t - \mu] + \beta x_t + \varepsilon_t$$

PROPHET

Facebook's Prophet modellerar tidsseriedatan y_t som en kombination av flera komponenter:

$$y_t = g(t) + s(t) + h(t) + \varepsilon_t$$

Där:

- $g(t)$ är trendkomponenten.
- $s(t)$ är säsongskomponenten.
- $h(t)$ är helgdagskomponenten.
- ε_t är feltermen.

Trendkomponenten $g(t)$:

Prophet stöder två typer av trendmodeller: linjär och logistisk.

- Linjär tillväxt:

$$g(t) = \beta_0 + \beta_1 t$$

Där β_0 är interceptet och β_1 är tillväxttakten.

- Logistisk tillväxt (när data är begränsad):

$$g(t) = \frac{C}{1 + \exp(-(\beta_0 + \beta_1 t))}$$

Där C är bärkapaciteten, β_0 är interceptet, och β_1 är tillväxttakten.

Säsongkomponenten $s(t)$:

Prophet modellerar säsongsmässighet med Fourier-serieutvidgningar. För en given säsongsmässig period p , kan säsongskomponenten skrivas som:

$$s(t) = \sum_{k=1}^K \left[\alpha_k \cos\left(\frac{2\pi kt}{p}\right) + \beta_k \sin\left(\frac{2\pi kt}{p}\right) \right]$$

Där α_k och β_k är Fourier-koefficienterna, och K är antalet Fourier-termer som används.

Helgdagskomponenten $h(t)$:

Helgdagskomponenten tar hänsyn till effekterna av helgdagar eller speciella händelser och modelleras vanligtvis som:

$$h(t) = \sum_{j=1}^J \delta_j I(t \text{ is holiday}_j)$$

Där δ_j representerar effekten av helgdag (*holiday*) $_j$, och $I(t \text{ is helgdag}_j)$ är en indikatorfunktion som är 1 om t är på helgdag j och 0 annars.

Feltermen ε_t :

Feltermen modelleras vanligtvis som Gaussiskt brus:

$$\varepsilon_t \sim N(0, \sigma^2)$$

Där σ^2 är variansen för feltermen.

Sammanfattning av viktiga ekvationer:

Helhetsmodellen:

$$y_t = g(t) + s(t) + h(t) + \varepsilon_t$$

Linjär trend:

$$g(t) = \beta_0 + \beta_1 t$$

Logistisk tillväxt:

$$g(t) = \frac{C}{1 + \exp(-(\beta_0 + \beta_1 t))}$$

Säsongskomponent:

$$s(t) = \sum_{k=1}^K \left[\alpha_k \cos\left(\frac{2\pi kt}{p}\right) + \beta_k \sin\left(\frac{2\pi kt}{p}\right) \right]$$

Helgdagskomponent:

$$h(t) = \sum_{j=1}^J \delta_j I(t \text{ is holiday}_j)$$

Felterm:

$$\varepsilon_t \sim N(0, \sigma^2)$$

XGBOOST för tidsserieregression

XGBoost (Extreme Gradient Boosting) är en ensemblemetod baserad på boosting-tekniker, specifikt utformad för prestanda och effektivitet. När XGBoost tillämpas på tidsserieregression kan den hantera icke-linjera samband och komplexa mönster i datan. Modellen byggs genom sekventiell träning av beslutsträd.

Grundprinciper för XGBoost är att iterativt förbättra modellen genom att minimera förlustfunktionen. Givet en uppsättning av n observationer $\{(x_i, y_i)\}_{i=1}^n$, där x_i representerar funktionerna och y_i är målvariabeln, formuleras prediktionsmodellen som:

$$\hat{y}_i = \sum_{k=1}^K f_k(x_i)$$

Där f_k representerar k -beslutsträdet och K är det totala antalet träd i ensemblen.

Målfunktion:

XGBoost optimerar en målfunktion som består av två delar: en förlustfunktion L och en regulariseringsterm Ω :

$$Obj(\Theta) = \sum_{i=1}^n L(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k)$$

Där:

- $L(y_i, \hat{y}_i)$ är förlustfunktionen som mäter skillnaden mellan de verkliga värdena y_i och de förutspådda värdena $\hat{y}_i$.
- $\Omega(f_k)$ är regulariseringstermen som straffar komplexiteten hos trädet f_k . Den definieras som:

$$\Omega(f_k) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

Där T är antalet blad (slutliga beslut eller utfall) i trädet, w_j vikten för varje blad, och γ och λ är regulariseringsparametrar.

Gradient Boosting-algoritm:

XGBoost använder en gradient boosting-ansats för att minimera målfunktionen. För varje iteration t , beräknar den residualerna och uppdaterar modellen med ett nytt träd f_t :

$$\begin{aligned} \text{Res}_i^{(t)} &= y_i - \hat{y}_i^{(t-1)} \\ \hat{y}_i^{(t)} &= \hat{y}_i^{(t-1)} + \eta f_t(x_i) \end{aligned}$$

Där:

- $\text{Res}_i^{(t)}$ är residualen eller felet vid iteration t .
- η är inlärningshastigheten (skalningsfaktorn) som kontrollerar varje träds bidrag.

Hantering av tidsseriedata:

För tidsserieregression kan XGBoost inkorporera fördröjda värden och andra tidsrelaterade funktioner i funktionsuppsättningen x_i . Funktionerna kan inkludera tidigare värden av tidsserien, säsongindikatorer och externa variabler. Modellen tränas för att lära sig de komplexa sambanden mellan dessa funktioner och de framtida värdena av tidsserien.

Sammanfattning av viktiga ekvationer:

Prediktionsmodell:

$$\hat{y}_i = \sum_{k=1}^K f_k(x_i)$$

Målfunktion:

$$Obj(\Theta) = \sum_{i=1}^n L(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k)$$

Regulariseringsterm:

$$\Omega(f_k) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

Gradientuppdatering:

$$\text{Res}_i^{(t)} = y_i - \hat{y}_i^{(t-1)}$$

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta f_t(x_i)$$

SVR

Inom arbete med SVR är målet att hitta en funktion $f(x)$ som har högst en epsilon (ϵ) avvikelse från de faktiska observerade värdena y_i , samtidigt som modellens komplexitet minimeras. Funktionen formuleras som:

$$f(x) = w^T x + b$$

Där w är viktvektorn och b är bias-termen. Optimeringsproblemet formuleras som:

$$\min_{w, b, \xi_i, \xi_i^*} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*)$$

Under villkor att:

$$y_i - (w^T x_i + b) \leq \epsilon + \xi_i$$

$$(w^T x_i + b) - y_i \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

Där:

- ξ_i och ξ_i^* är slack-variabler som mäter avvikelser större än ϵ .
- C är regulariseringsparametern som balanserar avvägningen mellan att maximera marginalen och minimera träningsfelet.
- ϵ är toleransmarginalen inom vilken prediktioner anses acceptabla.

Kernel-metoder

SVR kan använda kernel-funktioner för att hantera data som inte är linjärt separerbar.

Funktionen $f(x)$ beräknas i ett högre dimensionellt rum med hjälp av en kernel-funktion

$k(x_i, x)$:

$$f(x) = \sum_{i=1}^n (\alpha_i - \alpha_i^*) k(x_i, x) + b$$

Där α_i och α_i^* är Lagrange-multiplikatorerna som är associerade med stödvektorerna.

Sammanfattning av viktiga ekvationer:

Regressionsfunktion:

$$f(x) = w^T x + b$$

Målfunktion:

$$\min_{w, b, \xi_i, \xi_i^*} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*)$$

Begränsningar:

$$y_i - (w^T x_i + b) \leq \epsilon + \xi_i$$

$$(w^T x_i + b) - y_i \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

Kernel-metoder:

$$f(x) = \sum_{i=1}^n (\alpha_i - \alpha_i^*) k(x_i, x) + b$$

LSTM

Long Short-Term Memory (LSTM) är en typ av återkopplande neuralt nätverk (RNN) som är utformat för att modellera sekvenser och långsiktiga beroenden i data. LSTM löser problemet med försvinnande gradienter, som ofta uppstår i traditionella RNN, vilket gör att de kan lära sig och minnas långsiktiga beroenden.

En LSTM-cell består av flera viktiga komponenter: celltillståndet C_t , det dolda tillståndet h_t , och tre grindar: inmatningsgrinden i_t , glömgrinden f_t , och utmatningsgrinden o_t . Dessa komponenter samarbetar för att hantera informationsflödet.

Ekvationer:

Glömgrind:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

Inmatningsgrind:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$C_t = \tanh(W_c[h_{t-1}, x_t] + b_c)$$

Uppdatering av celltillståndet:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

Utmatningsgrind:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$

Där:

- σ betecknar sigmoid-aktiveringsfunktionen.
- $\tanh$ betecknar hyperbolisk tangens-aktiveringsfunktionen.
- $\odot$ representerar elementvis multiplikation.
- W_f , W_i , W_c och W_o är viktmatriser för glöm-, inmatnings-, celltillstånds- och utmatningsgrindarna.
- b_f , b_i , b_c och b_o är bias för grindarna.
- h_{t-1} är det dolda tillståndet från föregående tidssteg, och x_t är indata vid tidssteg t .

Sammanfattning av viktiga ekvationer:

Glömgrind:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

Inmatningsgrind:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

Uppdatering av celltillståndet:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

Utmatningsgrind:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$

GRU

Gated Recurrent Units (GRU) är en variant av återkopplande neurala nätverk (RNN) som adresserar problemet med försvinnande gradienter och förbättrar inläringen av långsiktiga beroenden. GRU förenklar arkitekturen hos Long Short-Term Memory (LSTM)-nätverk genom att kombinera inmatnings- och glömgrindarna till en enda uppdateringsgrind och slå ihop celltillståndet med det dolda tillståndet.

GRU-cellstruktur

En GRU-cell består av två huvudsakliga komponenter: uppdateringsgrinden z_t och återställningsgrinden r_t . Dessa grindar styr informationsflödet och uppdateringen av det dolda tillståndet.

Ekvationer:

Uppdateringsgrind:

$$z_t = \sigma(W_z[h_{t-1}, x_t] + b_z)$$

Återställningsgrind:

$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r)$$

Kandidataktivering:

$$\tilde{h}_t = \tanh(W_h[r_t \odot h_{t-1}, x_t] + b_h)$$

Uppdatering av dolt tillstånd:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

Där:

- σ betecknar sigmoid-aktiveringsfunktionen.
- $\tanh$ betecknar hyperbolisk tangens-aktiveringsfunktionen.
- $\odot$ representerar elementvis multiplikation.
- W_z , W_r och W_h är viktmatriser för uppdateringsgrind, återställningsgrind och kandidataktivering
- b_z , b_r och b_h är bias för grindarna och kandidataktiveringen.
- h_{t-1} är det dolda tillståndet från föregående tidssteg, och x_t är indata vid tidssteg t

Sammanfattning av viktiga ekvationer:

Uppdateringsgrind:

$$z_t = \sigma(W_z[h_{t-1}, x_t] + b_z)$$

Återställningsgrind:

$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r)$$

Kandidataktivering:

$$\tilde{h}_t = \tanh(W_h[r_t \odot h_{t-1}, x_t] + b_h)$$

Uppdatering av dolt tillstånd:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

CNN

Convolutional Neural Networks (CNN) är en klass av djupa neurala nätverk utformade för att bearbeta strukturerad rutnätsdata, såsom bilder. De är särskilt effektiva för uppgifter som involverar rumsliga hierarkier, som bildigenkänning och objekt-detektion. CNN utnyttjar konvolutionella lager för att fånga rumsliga funktioner från indata.

Den centrala komponenten i en CNN är den konvolutionella lagret, som applicerar konvolutionella filter (kärnor) på indata. Den matematiska operationen av konvolution är definierad som:

$$(\text{Conv})(I, K)(i, j) = \sum_m \sum_n I(i + m, j + n) \cdot K(m, n)$$

Där:

- I är indata-bilden (eller funktionskartan)
- K är den konvolutionella kärnan (filtret)
- (i, j) är de rumsliga koordinaterna för utdata-funktionskartan
- Summeringarna sker över dimensionerna på kärnan

Efter att konvolutionen applicerats används vanligtvis en icke-linjär aktiveringsfunktion. Vanliga aktiveringsfunktioner inkluderar:

$$\text{ReLU}(x) = \max(0, x)$$

Där x är indata till ReLU-aktiveringsfunktionen.

Pooling-lager reducerar de rumsliga dimensionerna hos funktionskartor samtidigt som viktig information behålls. Den vanligaste poolingoperationen är max-pooling:

$$(\text{MaxPool})(F)(i, j) = \max_{m, n} F(i + m, j + n)_{m, n}$$

Där F är funktionskartan och det maximala värdet tas över ett poolingfönster (t.ex. ett

2×2 -fönster).

Efter konvolutionella och pooling-lager plattas funktionskartorna ut och skickas vidare genom fullt anslutna lager:

$$(\text{FC})(x) = W \cdot x + b$$

Där:

- x är den utplattade indata-vektorn.
- W är viktmatrisen för det fullt anslutna lagret.
- b är bias-termen.

Sammanfattning av viktiga ekvationer:

Konvolution:

$$(\text{Conv})(I, K)(i, j) = \sum_m \sum_n I(i + m, j + n) \cdot K(m, n)$$

ReLU-aktiveringsfunktion:

$$\text{ReLU}(x) = \max(0, x)$$

Max-poolingoperation:

$$(\text{MaxPool})(F)(i, j) = \max_{m, n} F(i + m, j + n)$$

Fullt anslutet lager:

$$(\text{FC})(x) = W \cdot x + b$$

Bilaga C Utvärderingstabeller för tränade algoritmer

Iteration 1

Tabell C.1

Träningsslängd med erhållet MAPE – ingen skalning av data.

Månader:	18	12	6	3	2	1
ARIMA - MAPE	3,1 +/- 1,0	4,2 +/- 2,4	5,0 +/- 1,5	5,1 +/- 2,9	4,6 +/- 1,8	5,0 +/- 3,0
ARIMA - MAPE MIN/MAX	1,9 // 5,0	2,1 // 13	2,4 // 7,7	2,3 // 13	2,3 // 9,1	2,5 // 19
PROPHET - MAPE	3,1 +/- 1,0	4,6 +/- 2,1	3,6 +/- 1,4	4,2 +/- 1,3	4,8 +/- 3,2	3,5 +/- 1,0
PROPHET - MAPE MIN/MAX	1,9 // 5,2	2,2 // 8,8	1,9 // 7,5	2,2 // 7,1	2,2 // 19	2,2 // 6,5
SVR - MAPE	2,72 +/- 0,36	4,2 +/- 3,1	4,1 +/- 2,6	4,7 +/- 2,8	7,4 +/- 4,5	4,9 +/- 2,8
SVR - MAPE MIN/MAX	2,1 // 3,6	2,1 // 14	2,1 // 14	2,1 // 14	2,4 // 19	1,9 // 14
XGB	2,1 +/- 0,3	3,5 +/- 0,5	3,4 +/- 1,1	3,7 +/- 1,5	3,7 +/- 1,5	3,9 +/- 1,5
XGB - MAPE MIN/MAX	1,3 // 3,2	1,8 // 3,6	1,6 // 8,8	1,6 // 12	1,4 // 13	1,8 // 12

Tabell C.2

Träningsslängd med erhållet MAPE – standardskalning av data.

Månader:	18	12	6	3	2	1
ARIMA - MAPE	3,3 +/- 0,8	4,1 +/- 1,9	5,1 +/- 2,4	5,7 +/- 3,0	4,3 +/- 1,7	5,7 +/- 3,0
ARIMA - MAPE MIN/MAX	2,3 // 5,0	1,9 // 10	2,1 // 11	2,5 // 16	2,6 // 11	2,6 // 11
PROPHET - MAPE	2,5 +/- 0,5	3,9 +/- 1,8	4,2 +/- 1,8	3,7 +/- 1,3	3,7 +/- 1,1	4,2 +/- 1,8
PROPHET - MAPE MIN/MAX	1,9 // 3,6	2,0 // 8,9	2,3 // 8,9	2,3 // 7,0	2,1 // 6,3	2,3 // 9,7
SVR - MAPE	4,4 +/- 1,0	6,5 +/- 4,5	6,1 +/- 3,9	7,1 +/- 4,4	7,4 +/- 4,5	7,5 +/- 4,5
SVR - MAPE MIN/MAX	2,4 // 6,2	2,3 // 19	2,3 // 19	2,3 // 19	2,4 // 19	2,5 // 19
XGB - MAPE	2,1 +/- 0,4	2,6 +/- 0,5	3,3 +/- 1,2	3,6 +/- 0,8	3,8 +/- 1,8	3,9 +/- 1,7
XGB - MAPE MIN/MAX	3,2	1,8 // 3,5	1,6 // 8,1	2,3 // 5,4	2,3 // 11	1,9 // 11

Tabell C.3

Träningsslängd med erhållet MAPE – min/max skalning av data.

Månader	18	12	6	3	2	1
ARIMA - MAPE	3,3 +/- 0,9	4,6 +/- 2,8	5,0 +/- 2,8	5,1 +/- 2,7	3,6 +/- 1,2	4,1 +/- 2,0
ARIMA - MAPE MIN/MAX	2,0 // 5,4	1,9 // 17	2,0 // 20	1,8 // 17	2,1 // 6,9	1
PROPHET - MAPE	2,6 +/- 0,4	4,3 +/- 2,0	4,4 +/- 2,0	4,3 +/- 2,0	4,3 +/- 2,0	4,3 +/- 2,0
PROPHET - MAPE MIN/MAX	1,8 // 3,4	1,9 // 8,1	2,4 // 8,8	2,2 // 7,6	2,1 // 11	2,3 // 9,5
SVR - MAPE	2,8 +/- 0,4	4,5 +/- 3,6	4,3 +/- 3,0	5,1 +/- 3,3	5,3 +/- 3,4	8 +/- 5
SVR - MAPE MIN/MAX	2,0 // 3,6	2,0 // 15	2,0 // 15	2,0 // 15	1,9 // 15	2,0 // 19
XGB - MAPE	2,1 +/- 0,4	2,7 +/- 0,5	3,6 +/- 1,2	3,8 +/- 1,6	4,0 +/- 1,9	3,6 +/- 1,8
XGB - MAPE MIN/MAX	1,3 // 3,2	1,8 // 3,8	2,1 // 6,9	2,4 // 11	2,3 // 11,4	1,5 // 12

Iteration 3

Rullande medelvärde	1	3	7
ARIMA MAPE	3,2 +/- 0,6	3,1 +/- 0,5	3,1 +/- 0,4
ARIMA MAPE min/max	2,5 +/- 4,5	2,6 +/- 3,8	2,6 +/- 3,8
PROPHET MAPE	3,6 +/- 1	3,4 +/- 1	3,1 +/- 1
PROPHET MAPE min/max	2,6 +/- 7,0	2,3 +/- 6,3	2,3 +/- 6,0
XGB MAPE	2,3 +/- 0,5	2,6 +/- 0,6	2,3 +/- 0,4
XGB MAPE min/max	1,4 +/- 2,7	1,8 +/- 3,6	1,7 +/- 2,8
SVR MAPE	2,8 +/- 0,4	2,8 +/- 0,4	2,8 +/- 0,4
SVR MAPE min/max	1,9 +/- 3,8	2,0 +/- 3,9	2,0 +/- 4,1

Tabell C.4

Erhållet MAPE för analys av rullande medelvärde på temperatur.

Justering av temperatur	Ingen	Positiv skala	0 splitt	2 splitt	4 splitt
ARIMA mean	3,2 +/- 0,6	3,2 +/- 0,6	3,0 +/- 0,4	3,1 +/- 0,4	3,0 +/- 0,4
ARIMA min/max	2,5 // 4,5	2,5 // 4,5	2,5 // 3,6	2,5 // 3,6	2,5 // 3,6
PROPHET	3,6 +/- 1,4	3,6 +/- 1,4	3,5 +/- 1,3	3,5 +/- 1,6	3,4 +/- 1,5
PROPHET min/max	2,6 // 7,0	2,6 // 7,0	2,7 // 7,0	2,6 // 8,0	2,6 // 7,0
XGB	2,3 +/- 0,5	2,3 +/- 0,5	2,3 +/- 0,5	2,3 +/- 0,5	2,4 +/- 0,5
XGB min/max	1,4 // 2,7	1,5 // 2,7	1,6 // 2,8	1,6 // 3,0	1,5 // 3,0
SVR	2,8 +/- 0,4	2,8 +/- 0,4	2,7 +/- 0,5	2,9 +/- 0,4	2,8 +/- 0,5
SVR min/max	1,9 // 3,8	2,0 // 3,8	1,7 // 3,6	2,2 // 4,0	2,1 // 5,2

Tabell C.5

Erhållet MAPE för justering av temperaturdata/skala.

Tabell C.6

Erhållet MAPE för rullande soltimmar samt sammanhängande dagar med mycket sol.

Dagar för rullande medelvärde	1	3	7	14	Sammanhängande soliga dagar
ARIMA	3,4 +/- 0,9	3,0 +/- 0,5	2,8 +/- 0,4	2,6 +/- 0,3	2,9 +/- 0,6
ARIMA min-max	2,3 // 4,9	2,3 // 4,2	2,3 // 3,7	2,3 // 3,3	2,3 // 4,2
PROPHET MAPE Mean	2,9 +/- 0,30	2,8 +/- 0,40	2,6 +/- 0,42	2,6 +/- 0,3	2,6 +/- 0,23
PROPHET MAPE min-max	2,5 // 3,4	2,1 // 3,5	1,8 // 3,1	2,2 // 3,1	2,2 // 2,9
XGB MAPE mean	2,3 +/- 0,4	2,2 +/- 0,3	2,3 +/- 0,4	1,9 +/- 0,35	2,1 +/- 0,4
XGB MAPE min - max	1,8 // 2,9	1,9 // 2,9	1,8 // 2,9	1,4 // 2,4	1,4 // 2,6
SVR	3,0 +/- 0,5	2,9 +/- 0,5	2,9 +/- 0,6	4,0 +/- 1,8	3,0 +/- 0,6
SVR MAPE min - max	2,1 // 3,9	2,1 // 3,8	2,1 // 3,8	2,5 // 8,1	2,1 // 4,7

Dagar för rullande medelvärde	1	3	7	14	neg
ARIMA MAPE	3,4 +/- 0,9	3,3 +/- 0,9	3,4 +/- 1,0	3,3 +/- 0,9	3,3 +/- 0,9
ARIMA MAPE MIN/MAX	2,3 // 5,4	2,1 // 5,3	2,1 // 5,8	2,1 // 5,4	2,0 // 5,4
PROPHET MAPE	2,45 +/- 0,24	2,55 +/- 0,24	2,53 +/- 0,28	2,73 +/- 0,57	2,50 +/- 0,24
PROPHET MAPE MIN/MAX	2,10 // 2,85	2,20 // 2,90	1,80 // 2,80	1,70 // 3,70	2,15 // 2,85
SVR MAPE	2,6 +/- 0,3	4,1 +/- 3,3	2,7 +/- 0,5	2,8 +/- 0,6	3,3 +/- 1,8
SVR MAPE MIN/MAX	2,1 // 3,6	2,2 // 22	1,8 // 4,1	1,8 // 4,3	2,0 // 12
XGB MAPE	2,21 +/- 0,28	2,43 +/- 0,26	2,17 +/- 0,27	2,16 +/- 0,31	2,32 +/- 0,36
XGB MAPE MIN/MAX	1,89 // 2,75	2,04 // 2,87	1,82 // 2,56	1,67 // 2,57	1,78 // 3,08

Tabell C.7

Erhållet MAPE för behandling av nederbördsdata.

Modell	ARIMA	PROPHET	XGB	SVR
MAPE Mean	3,3 +/- 0,9	2,1 +/- 0,3	3,0 +/- 0,6	2,0 +/- 0,4
MAPE min-max		1,8 // 2,7	2,2 // 3,5	

Tabell C.8

Erhållet MAPE för "one hot encoded"-information om veckodagar som variabel.

Modell	ARIMA	PROPHET	XGB	SVR
MAPE Mean	2,90 +/- 0,41	2,35 +/- 0,36	2,45 +/- 0,44	2,7 +/- 0,5
MAPE min-max	2,4 // 3,5	1,9 // 3,1	1,9 // 3,3	1,7 // 3,6

Tabell C.9

Erhållet MAPE för helgindikator som variabel.

Modell	ARIMA	PROPHET	XGB	SVR
MAPE Mean	3,02 +/- 0,63	2,52 +/- 0,31	2,11 +/- 0,33	4,4 +/- 1,9
MAPE min-max	2,58 // 4,19	2,10 // 3,07	1,45 // 2,42	2,8 // 10

Tabell C.10

Erhållet MAPE för allmänna helgdagar som variabel.

Modell	ARIMA	PROPHET	XGB	SVR
MAPE Mean	3,28 +/- 0,95	2,52 +/- 0,32	2,33 +/- 0,29	9,0 +/- 5,0
MAPE min-max	2,09 // 4,69	2,00 // 3,09	2,03 // 3,02	2,8 // 16

Tabell C.11

Erhållet MAPE för skollov som variabel.

Iteration 4

Fristående variabel	Nederbörd Veckodag Helg	Utan helg	Utan nederbörd	Utan veckodag
MAPE Mean	3,2 +/- 0,26	3,2 +/- 0,9	2,3 +/- 0,3	3,8 +/- 1,6
MAPE min-max	2,4 // 5,3	2,2 // 6,1	2,3 // 5,0	2,1 // 6,9

Tabell C.12

Erhållet MAPE för bästa undersökta kombination av variabler för Prophet.

Variabler	Torra dagar Varma dagar Soliga dagar Veckodagar Nederbörd	- - - Veckodagar Nederbörd	Torra dagar Varma dagar Soliga dagar Veckodagar -	Torra dagar Varma dagar Soliga dagar - Nederbörd
svr	3,3 +/- 0,8	3,2 +/- 0,9	3,4 +/- 0,7	3,8 +/- 1,6

Tabell C.13

Erhållet MAPE för bästa undersökta kombination av variabler för SVR.

Variabel	Endast förbrukning	Veckodag	Nederbörd	Torra dagar Varma dagar Soliga dagar
MAPE Mean	2,8 +/- 0,4	2,0 +/- 0,4	2,6 +/- 0,3	2,6 +/- 0,36

Tabell C.14

Erhållet MAPE för bästa fristående variabler för SVR.

Svenskt Vatten

UTVECKLING

Svenskt Vatten Utveckling
Svenskt Vatten AB

POSTADRESS BOX 14057, 167 14 Bromma

BESÖKSADRESS Gustavslundsvägen 12, 167 51 Bromma

TELEFON 08-506 002 00

E-MAIL svensktvatten@svensktvatten.se

www.svensktvatten.se