Norges miljø- og biovitenskapelige universitet

Masteroppgave 2022 30 stp
Handelshøyskolen

Utvikling av salgsprognoser: en sammenligning av tradisjonelle og maskinlæringsmodeller

Developing Sales Forecasts: a Comparison of Traditional and Machine Learning Models

Henriette Aasen og Marion Sunde
Business Analytics

Forord

Denne masteroppgaven er en avsluttende del på et toårig masterstudie i økonomi og administrasjon ved Handelshøyskolen på Norges miljø- og biovitenskapelige universitet (NMBU).

Vi ønsker å benytte anledningen til å rette en stor takk til våre veiledere Frode Alfnes og Thore Larsgård, som gjennom hele prosessen har gitt oss konstruktive og nyttige innspill. I tillegg ønsker vi også å rette en stor takk til E. Sæther AS og spesielt Ricardo Hernandez, for innspill om tema for oppgaven og overleveringen av salgsdata som analysene er basert på.

Til slutt vil vi rette en takk til Birk Johansson, familie og venner. Takk for deres tålmodighet, oppmuntrende ord og støtte gjennom denne perioden.

Sammendrag

Formålet med denne oppgaven er å undersøke og sammenligne om tradisjonelle modeller eller maskinlæringsmodeller gir best resultat når de benyttes til salgsprognoser. Per i dag har ikke E. Sæther AS en standardisert prosess for hvordan salgsprognoser utvikles, noe som gir oss en god mulighet til å utføre en rekke analyser for å finne en modell som på sikt kan tas i bruk av selskapet.

Problemstillingen vi har besvart i denne oppgaven er: *"Hvilken prognosemodell bør Sæther ta i bruk til fremtidig prognosearbeid?".* For å besvare denne problemstillingen har vi analysert det ukentlige aggregerte salget i produktkategorien cleanser. Analysene er basert på historisk salgsdata fra januar 2019 til desember 2021.

De tradisjonelle modellene som testes er enkel eksponentiell glatting, dobbel eksponentiell glatting, og trippel eksponentiell glatting, ARIMA, SARIMA og SARIMAX. Maskinlæringsmodellene som testes er Light Gradient Boosting, Random Forest og lineær regresjon. Dataen splittes opp i et treningssett og testsett og modellene sammenlignes ved hjelp av Root Mean Squared Error (RMSE).

Til tross for et enkelt datagrunnlag bestående av 157 observasjoner med historisk salgsdata viser resultatene tydelig at maskinlæringsmodellene i stor grad utkonkurrerer de tradisjonelle når man sammenligner RMSE. Modellen med lavest RMSE er Light Gradient Boosting. Hvorvidt denne modellen kan anbefales for videre arbeid avhenger av tilgjengelige ressurser og kunnskap i

Sæther. Dersom modellen implementeres anbefaler vi at den utvides med flere variabler, eller brukes i kombinasjon med dømmekraft.

Abstract

The purpose of this paper is to compare how classical and machine learning methods perform in sales forecasting. As of now E. Sæther AS does not have a standardized process for developing sales forecast, which gives us the opportunity to perform multiple analysis to find a suitable model that the company may use for future work.

The research question we aim to answer is: "Which forecasting model should Sæther use for future forecasting?". When performing the analysis, we used the aggregated weekly sales for the product category cleanser. The sales data is collected from January 2019 to December 2021.

The classical models tested consist of Simple Exponential Smoothing, Double Exponential Smoothing, Triple Exponential Smoothing, ARIMA, SARIMA and SARIMAX. For the machine learning we have tested Light Gradient Boosting, Random Forest, and Linear Regression. The data is split into a training and test set, and the models are compared using the Root Mean Squared Error (RMSE).

Despite using simple data consisting of 157 sales observations the machine learning models outperform the classical models. The model with the lowest RMSE is Light Gradient Boosting. Whether this is the optimal model for future work depends on the available resources and knowledge at Sæther. If the model is implemented we recommend extending it with further features, or that it is used in combination with judgmental forecasts.

Innholdsfortegnelse

Figuroversikt

Tabelloversikt

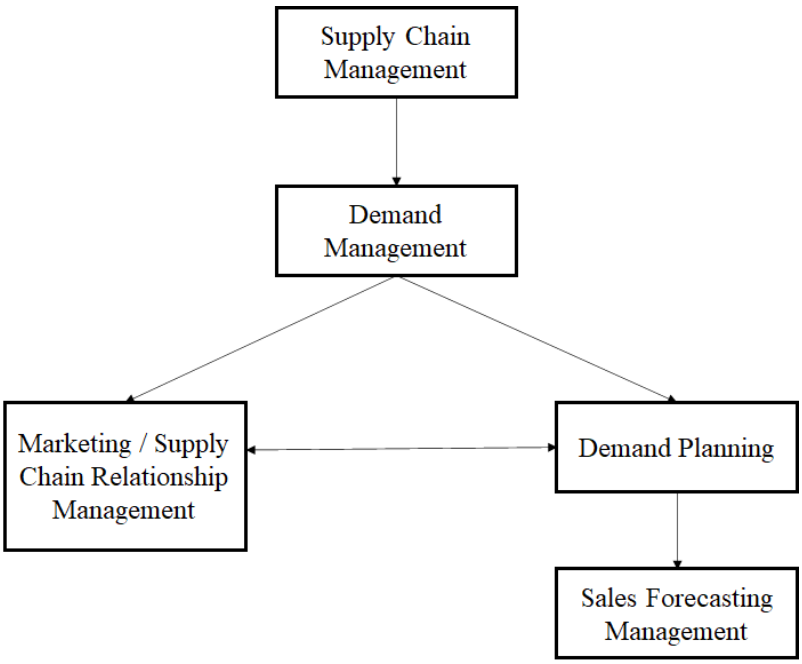
Formeloversikt

1.0 Introduksjon

1.1 Bakgrunn

Muligheten til å vite hvor mye man kommer til å selge en uke, en måned eller lenger frem i tid kan vi med en viss sikkerhet si at de fleste bedrifter gjerne skulle visst. Selv om det er umulig for bedrifter og klare å forutsi fremtiden helt perfekt kan salgsprognoser være til god hjelp. Gjennom gode salgsprognoser kan bedrifter sørge for at de har nok varer på lager til å forhindre utsolgtsituasjoner, men samtidig unngå overflødig varelager som er ineffektivt og kan føre til økte driftskostnader. Salgsprognoser er ikke et nytt fenomen i forretningsprosessen og opp gjennom årene har det skjedd endringer som påvirker hvordan man utarbeider prognoser. Faktorer slik som økt bruk av informasjonsteknologi har banet vei for nye måter å analysere og bruke selskapets interne data.

I figur 1 ser vi en illustrasjon som viser noen av de ulike prosessene som inngår i etterspørselsstyring. Her kan vi se at salgsprognoser er en funksjon som ligger under etterspørselsplanlegging. Det er derfor viktig å huske på at selv om det kalles salgsprognoser er det etterspørselen vi ønsker å forutsi. Det vil si at vi ønsker å vite hva kundene etterspør slik at vi kan planlegge å oppnå salg så nært som mulig det vi predikerer (Mentzer & Moon, 2005). Det er viktig å bemerke at begrepene prediksjon og salgsprognoser vil brukes om hverandre.



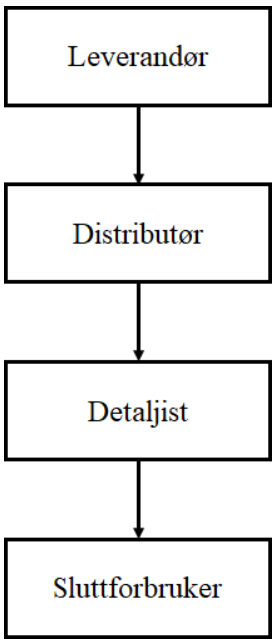
Figur 1: Etterspørselsstyring (Mentzer & Moon, 2005).

Rollen til salgsprognoser er avhengig av hvor i forsyningskjeden en bedrift befinner seg. En forsyningskjede har bare ett punkt med det som kalles uavhengig etterspørsel. Dette er mengden produkt som etterspørres av sluttforbruker i forsyningskjeden. Enten sluttbrukeren handler i fysisk butikk eller på nett, såkalt business-to-consumer (B2C). Eller vi har en bedrift som kjøper inn varer for videresalg, business-to-business (B2B), er det sluttbrukeren som bestemmer den sanne etterspørselen i en forsyningskjede. For selskapene i forsyningskjeden som er B2B kalles etterspørselen for en avledet etterspørsel. Dette vil si at etterspørselen er avledet fra hva andre selskaper i forsyningskjeden gjør for å møte etterspørselen fra deres nærmeste kunde. Det er viktig å merke seg at det er bare ett ledd i en forsyningskjede som er direkte påvirket av den uavhengige etterspørselen. De resterende leddende i forsyningskjeden er påvirket av avledet etterspørsel (Mentzer & Moon, 2005). Generelt vil informasjon om etterspørsel forvrenges jo lenger opp man befinner seg i forsyningskjeden. Dette kalles for bullwhip-effekten og tilsier at jo lenger opp i forsyningskjeden jo større blir feilprognosene. Bullwhip-effekten kan reduseres gjennom god informasjonsdeling i hele forsyningskjeden, og ved at alle ledd baserer sine prognoser på den uavhengige etterspørselen (Chopra & Meindl, 2013).

1.2 Oppgavens formål

Gjennom denne oppgaven skal vi se nærmere på hvordan vi kan utarbeide salgsprognoser for en distributør (B2B). E. Sæther AS, fremover kalt Sæther eller selskapet, er en norsk distributør av ulike merkevarer innen duft, hudpleie og sminke. Sæther er et av datterselskapene til Sæther Nordic som også har tilhørende datterselskap i Sverige, Danmark og Finland. Grunnet selskapets overordnede nordiske modell er flere av de store driftsfunksjonene sentralisert hos Sæther Nordic i Farum Danmark. Funksjoner slik som PR, logistikk og lager, innkjøp, økonomi, HR og IT blir håndtert på et overordnet nivå for alle datterselskapene.

Figur 2 viser en forenklet illustrasjon av forsyningskjeden vi skal se nærmere på. Som vi kan se fra figuren er Sæther leddet mellom leverandør og detaljist, og opplever derfor avledet etterspørsel.



Figur 2: Forsyningskjeden.

Oppgavens hovedformål vil være å undersøke og sammenligne hvordan tradisjonelle modeller og maskinlæringsmodeller presterer i å predikere det fremtidige salget for Sæther. Selskapet har ingen standardisert prosess for prognosesetting og bruker i liten grad prognosemodeller til dette arbeidet. Oppgaven er derfor spennende og aktuell, ikke bare for Sæther, men også for mindre selskaper som befinner seg i en lignende situasjon. Når de ulike modellene er vurdert opp mot

hverandre ønsker vi å kunne sitte igjen med en modell som selskapet kan videreutvikle og benytte i sin fremtidige prognoseprosess.

Det å utarbeide salgsprognoser er en kompleks prosess som kan ha stor påvirkning på en bedrifts beslutningstaking. Noen av faktorene som gjør denne prosessen kompleks er økonomiske og politiske faktorer, ulike salgsprosesser, og leverandører og kunder som har motstridende interesser. Vi vil senere i oppgaven se nærmere på de ulike faktorene som er med på å gjøre prognoseprosessen til Sæther kompleks.

1.3 Problemstilling og formål for oppgaven

Per i dag har selskapet ingen fastsatt prosess for prognosearbeid og benytter seg i liten grad av prognosemodeller. Den overordnede problemstillingen vi har valgt for oppgaven gjenspeiler selve hovedformålet med oppgaven. Problemstillingen er som følger:

"Hvilken prognosemodell bør Sæther ta i bruk til fremtidig prognosearbeid?"

For å kunne finne det som er å regne som den optimale modellen for selskapets fremtidige prognosearbeid ønsker vi å definere tre formål for oppgaven. Før vi begynner med analysene må det først velges ut modeller. Vi ønsker derfor å starte med en teoretisk gjennomgang av ulike prognosemodeller. Det første formålet for oppgaven er som følger: «d*efiner de tradisjonelle modellene og maskinlæringsmodellene.»*.

Videre vil vi utføre analyser knyttet til de utvalgte prognosemodellene. Vi ønsker å danne et overblikk over resultatene fra de ulike modellene slik at vi kan sammenligne deres prestasjoner. Det andre formålet med oppgaven vil være: *«er det store ulikheter i resultatene fra de tradisjonelle modellene og maskinlæringsmodellene?»*.

For å kunne gå videre med å anbefale hvilken modell Sæther burde ta i bruk til fremtidig prognosearbeid ønsker vi å velge den modellen som treffer best i sine prediksjoner. De ulike modellenes nøyaktighet måles ut ifra feilprognosene de gir. Det siste formålet med oppgaven vil derfor være: *«hvilken modell gir de mest presise prognosene?»*.

1.4 Struktur

Oppgaven er delt inn i seks kapitler.

I kapittel én presenteres oppgavens formål og problemstilling.

I kapittel to vil vi ta for oss tidligere forskning og teori om prognoser. Vi vil se nærmere på hva som er viktig i prognoseprosessen og hvordan salgsprognoser kan påvirke bedriften og omvendt. Vi ser også på noen interne prosesser i Sæther som er direkte knyttet opp til salgprognosene. Til slutt vil vi ta for oss teorien bak de ulike prognosemodellene og hvordan feilprognoser kan brukes til å evaluere prognosemodellene.

I kapittel tre gjennomgår vi metoden og datamateriale som er benyttet i oppgaven. Først vil datahåndtering gjennomgås, før valgt metode og prognosemodeller presenteres. Det vil også bli gjort en vurdering av validiteten og rentabiliteten av datamaterialet.

I kapittel fire presenterer vi resultatene fra analysen. Vi viser grafene fra de ulike modellene og sammenligner hvordan de presterer basert på RMSE.

I Kapittel fem diskuteres resultatene fra de ulike prognosemodellene. Praktiske implikasjoner og videre arbeid, samt oppgavens begrensninger og videre forskning vil også presenteres.

I Kapittel seks konkluderer vi og svarer på problemstillingen.

2.0 Teori

2.1 Tidligere forskning

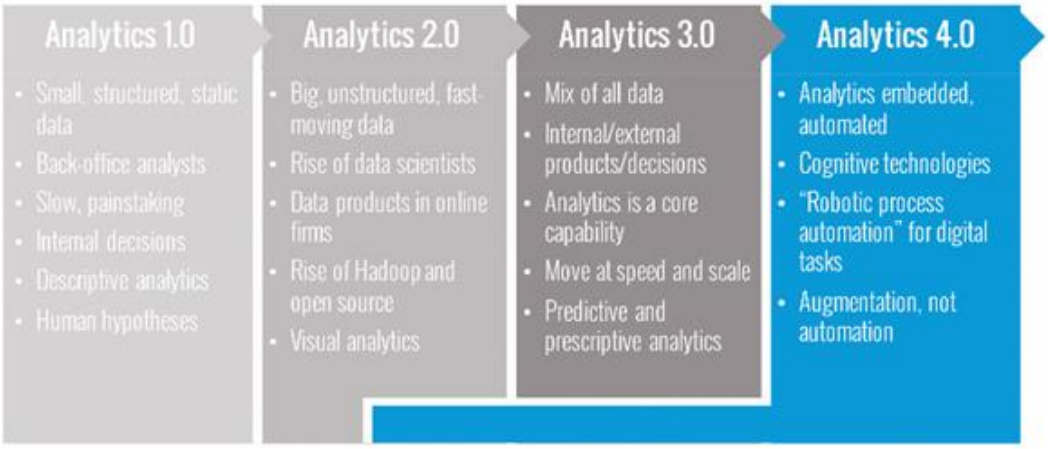
I varehandel er man avhengig av nøyaktige prognoser for å kunne ta beslutninger som gjelder planlegging, markedsføring, innkjøp og distribusjon. Unøyaktige prognoser kan føre til unødvendige kostnader og dårlig kundetilfredshet. Varelageret bør verken være for høyt, for å unngå avfall og ekstra kostnader ved lagring og arbeidskraft, og heller ikke for lavt for å forhindre utsolgtsituasjoner og tapt salg (Ma et al., 2016).

Evolusjonen av prognoser har med tiden gått fra enkle statistiske modeller til mer avanserte maskinlæringsmodeller. Hvorvidt disse modellene gir treffsikre prognoser eller ikke har vært tema for en rekke forskningsartikler.

Lawrence et al. (2000) undersøker i sin feltstudie hvorvidt 13 bedrifter klarer å lage treffsikre prognoser basert på dømmekraft. Resultatene fra studien viser at organisasjonenes prognoser basert på dømmekraft ikke var jevnt over mer nøyaktige enn en enkel, ikke-sesongjustert, naïv prognose.

Det finnes en rekke tradisjonelle prognosemodeller som er mer avanserte enn naïv ved at de blant annet inkluderer faktorer som sesongvariasjon, trend og andre sykliske mønstre. Udom og Phumchusri (2014) tar i sin artikkel for seg hvordan tidsseriemodellene glidende gjennomsnitt og Holts og Winters eksponentielle metode predikerer salg sammenlignet med ARIMA. De ulike prognosemodellene testes på fem datasett og sammenlignes ved hjelp av Mean Absolute Percentage Error (MAPE). Resultatene i studien viser at ARIMA modellen gir lavere feilprognoser enn de andre modellene.

Når det gjelder å estimere fremtidig salg er evnen til å analysere og tolke data som selskapet besitter både viktig og nyttig for selskapets beslutningstaking. I takt med digitalisering og teknologiske fremskritt har fokuset skiftet fra rene statistiske analyser, til å ta i bruk kunstig intelligens. Davenport (2018) mener at selskaper bør se på kunstig intelligens som en utvidelse av selskapets eksisterende analytiske kunnskaper. I figur 3 kan vi se Davenports fire analytiske faser.



Figur 3: De fire analytiske fasene (Davenport, 2018).

Fasene viser hvordan kunstig intelligens er bygget på tradisjonell statistikk og dermed kan ses på som en naturlig analytisk utvikling. Den blå halen fra fase 4.0 illustrerer fra hvilke faser det er mulig å ta i bruk kunstig intelligens. Når det kommer til å bruke kunstig intelligens til prognosearbeid har maskinlæring blitt et populært felt. Det finnes en rekke studier hvor maskinlæringsmodeller benyttes i prognosearbeid.

En studie utført av Bohanec et al. (2017) ser på hvordan maskinlæring kan brukes til å forbedre salgsprognoser som baseres på dømmekraft gjennom å bruke maskinlæring for å revidere og forbedre sine B2B salgsprognoser. I denne studien lager en ekstern konsulent salgsprognoser gjennom å bruke maskinlæring, i tillegg til prognosene de ansatte lager basert på dømmekraft. Gjennom å sammenligne sine prognoser med konsulenten kunne de ansatte revidere sine prognoser, noe som igjen gjorde de mer nøyaktige og treffsikre.

Vhatkar og Dias (2016) ser i sin studie på salgsprognoser for tre varer i kategorien munnpleieprodukter. Salgprognosene blir konstruert med ulike maskinlæringsalgoritmer og modellenes nøyaktighet blir evaluert ved hjelp av målene Mean Absolute Error, Mean Squared Error og Root Mean Squared Error. Resultatene viser at backpropagation neural network gir de mest nøyaktige prognosene for disse produktene. Forfatterne legger dog vekt på at valget av prognosemodell avhenger av datasettet man har.

Det er gjort en rekke studier på sammenligningen av tradisjonelle- og maskinlæringsmodellers prestasjoner for prognosesetting. Sharma og Sharma (2012) presenterer en komparativ analyse

hvor de i tillegg til å teste de ulike modellene hver for seg, også benytter en kombinert metode. Resultatet viste at den kombinerte metoden ga de mest nøyaktige resultatene når sammenligningsgrunnlaget ble målt på modellenes Mean Absolute Error, Mean Absolute Percentage Error, Mean Squared Error og Root Mean Squared Error.

Cecaj et al. (2020) forsøker i sin studie å predikere hvordan folkemengder fordeler seg under daglige aktiviteter i urbane områder, gjennom å sammenligne tradisjonelle og maskinlæringsmodeller. Tettheten og distribusjonen av mennesker predikeres ved å analysere et aggregert mobiltelefondatasett. Resultatene av denne studien viser at maskinlæringsmodeller er å foretrekke når det kommer til enkelhet og umiddelbar bruk, og at disse er hensiktsmessige for å redusere den maksimale prognosefeilen. Forfatterne trekker likevel frem at de tradisjonelle modellene er overlegne når det gjelder å gi mer presise prognoseresultater. Men at disse modellene krever mer datadomenekunnskap og beregningsmessig dyre teknikker for å velge de beste parameterne.

Demir og Akkaş (2018) har i sin casestudie utført en sammenligning av tradisjonelle tidsseriemodeller opp mot maskinlæringsmodellene Artificial Nerural Network og Support Vector Regression. Resultatene av studien vier at Support Vector Regression gir signifikant bedre resultater enn både tidsserie og Artificial Neural Network.

2.2 Salgsprognoser

I mange selskaper er salgprognoser en svært viktig del som inngår i prosessen for å matche tilbud og etterspørsel. Denne prosessen blir av Mentzer og Moon (2005) kalt for «sales and operation planning», forkortet til S&OP.

Figur 4: Sales and Operations Planning. Basert på en figur av Mentzer og Moon (2005).

Figur fire illustrerer hvordan to av de primære funksjonene i et selskap, etterspørselen og tilbudet påvirker S&OP og omvendt. En bedrifts etterspørsel drives av salg- og

markedsføringsavdelingen. Salgsavdelingen er ansvarlig for å generere og opprettholde etterspørselen fra kundene. Markedsføringsavdelingen er ansvarlig for å opprettholde og skape etterspørsel fra sluttforbruker. Selskapets tilbudsside består av funksjoner slik som innkjøp, logistikk, menneskelige ressurser og økonomi. Tilbudssiden av selskapet har blant annet ansvaret for å sikre at det er varer på lager samt lagerflyten ut til kunder. Det er svært viktig at de to sidene «snakker» med hverandre, og det er der S&OP prosessen kommer inn. S&OP virker som en «koblingsboks» der informasjon kan flyte mellom etterspørselssiden og tilbudssiden til en bedrift (Mentzer & Moon, 2005).

Salgsprognoser bør ha sitt utspring i etterspørselssiden til selskapet. Det er salg og markedsføring som er ansvarlige for å skape etterspørsel og som skal ha det beste perspektivet på hva fremtidig etterspørsel bør være. Ut ifra salgsprognosene vil man utarbeide en etterspørselsplan som består detaljert av hvilke produkter og antallet som trengs. På den andre siden bidras det med en kapasitetsplan som er en prediksjon for hvordan forsyningskapasiteten vil være, gitt en rekke interne og eksterne forutsetninger. Tilbake fra S&OP kommer det en driftsplan som blant annet består av innkjøpsplaner og distribusjonsplaner (Mentzer & Moon, 2005).

Selv om salgsprognoser er et viktig planleggingsverktøy, trekker Hyndman & Athanasopoulos (2021) frem at bedrifter ofte ikke gjør en tilstrekkelig jobb med dette arbeidet. En årsak til dette er at man ofte forveksler prognoser med planlegging og mål, til tross for at dette er tre forskjellige ting.

Prognoser handler om å forutsi fremtiden så nøyaktig som mulig, gitt all tilgjengelig informasjon, inkludert historiske data og kunnskap om eventuelle fremtidige hendelser som kan påvirke prognosene.

Mål er det man vil at skal skje. Mål bør knyttes til prognoser og planer, men dette skjer ikke alltid. Altfor ofte settes mål uten noen plan for hvordan de skal nås, uten realistiske prognoser.

Planlegging er en respons på prognoser og mål. Planlegging innebærer å bestemme passende handlinger som kreves for å få prognosene til å samsvare med målene.

2.3 Interne prosesser i Sæther

Informasjon om selskapets interne prosesser og rutiner kommer fra oppgavens ene forfatter, Marion Sunde, som for øyeblikket er ansatt i Sæther og har tilgang på informasjonen til

selskapet. Det har derfor ikke vært nødvendig å utføre intervjuer for å danne seg et bilde av dagens prosesser og rutiner.

Salgsprognoser er som nevnt en viktig del av prosessen som inngår i å matche tilbud og etterspørsel. Ved å utarbeide salgsprognoser vil Sæther kunne planlegge bedre frem i tid for å nå sine mål. Det er en rekke interne og eksterne faktorer som vil kunne påvirke prognosearbeidet og planleggingen til Sæther. Et høyst aktuelt tema nå er krigen i Ukraina og pandemien som forårsaker råvaremangel. Råvaremangel fører til at Sæthers leverandører ikke klarer å produsere og levere varer som normalt. Dette gjør at Sæther må justere sine salgsprognoser for varene som påvirkes og planlegge annerledes enn de normalt ville gjort, for å nå sine mål.

La oss ta for oss noen av de interne faktorene og prosessene som både påvirker, men også påvirkes av salgsprognoser. Funksjonene som blant annet innkjøp, lager og logistikk blir håndtert på et overordnet gruppenivå av Sæther Nordic.

Innkjøp

De norske markeds- og merkevaresjefene har seg imellom ansvaret for i underkant av 50 merkevarer fordelt på 14 leverandører. Et av deres mange ansvarsområder er å sende et estimat på antall varer som skal bestilles til de nordiske innkjøperne. Estimatet på antall varer som skal bestilles baseres i dag på selskapets forventede salg, som i stor grad er basert på dømmekraft, og fremtidige salgskampanjer. Kampanjeplanen er et viktig verktøy i Sæthers planleggingsprosess og forteller blant annet når de ulike salgskampanjene er og hvilke varer de inkluderer. Til de ulike kampanjene vil det derfor være nødvendig å øke antall kvantiteter for å sikre nok varer til butikkene. Det er de nordiske innkjøperne har ansvaret for å samle inn totalt estimert antall varer fra Norge, Sverige, Danmark og Finland, og sender bestillingene til de ulike leverandørene.

Ettersom Sæther har flere leverandører med ulik geografisk plassering, vil ledetiden på de ulike varene kunne variere. Ledetiden avhenger av leverandør, men den vil som regel være et sted mellom 3-6 måneder. Sæther trenger derfor å utarbeide salgsprognoser som strekker seg minst 3 måneder frem i tid, men i visse tilfeller lenger, slik at de kan planlegge det fremtidige innkjøpet. Selskapet vil derfor ikke kunne handle raskt dersom det skulle bli utsolgt for en eller flere varer, noe som forsterker viktigheten ved å ha et sikkerheteslager. Et reelt problem man kan møte på er at det ikke er mulig å kjøpe inn ønsket antall kvantiteter av en vare. I disse tilfellene blir

fordelingen av kvantiteter utført av Sæther Nordic. For å håndtere en slik situasjon må salgsprognosene justeres, og man må planlegge annerledes for å skulle nå det satte målet.

Bestillingene fra kundene påvirker hva Sæther igjen bestiller fra leverandør. Kundene til Sæther består av en rekke butikker, både frittstående, men også større kjeder. Med ulike kunder kommer ulike bestillingsprosesser. Hovedsakelig vil de større kjedene med sentrallager sende inn normalordre ukesvis. Det bestilles da ofte et større antall varer til kjedekundene da de handler inn til sitt eget sentrallager for å så videredistribuere det selv til de ulike butikkene. De frittstående kundene bestiller selv varer til sin butikk. Disse bestillingene er derfor mindre, og kan komme med ujevn frekvens.

Lager og logistikk

Sæther har et nordisk sentrallager i Farum i Danmark som betjener alle de nordiske landene. Hvert år mottar lageret ca. 12000 paller, noe som tilsvarer ca. 400 fulle lastebiler. I en perfekt verden hadde dette tilsvart ca. 2 lastebiler om dagen, men i praksis kan det være dager uten noen leveringer og dager hvor det kommer 10 lastebiler. Dette kan være ressurskrevende da tiden det tar å motta og pakke opp varer kan påvirke hvor mye tid de ansatte har til å pakke og sende bestillinger. På den andre siden vil uåpnede esker kunne forsinke utsendelsen av bestillinger hvis varen som er bestilt ikke er pakket opp enda.

Selv om vi i denne oppgaven ikke inkluderer andre variabler enn historisk salgsdata er det viktig å vite om de interne faktorene som påvirker selskapets vareflyt, innkjøp og det faktiske salget. Det er også nødvendig å forstå hvordan viktige prosesser i de ulike avdelingene vil påvirke hvordan man betrakter og lager prognoser.

2.4 Prognoseprosessen

Prosessen med å utarbeide prognoser kan deles inn i fem steg (Hyndman & Athanasopoulos, 2021):

Problemdefinisjon

Det første steget i en prognoseprosess er å få en forståelse for hva som er det faktiske problemet. Hva ønsker man å finne ut av, hvordan skal prognosene brukes og hvem skal bruke de.

Samle informasjon

For å kunne lage prognoser er det nødvendig å ha tilstrekkelig informasjon. Informasjonen kan være kvantitativ eller kvalitativ og den kan bestå av både intern og ekstern data.

Eksplorativ analyse

Ved å alltid starte med eksplorativ analyse vil vi raskt kunne se om det er mønster, trender, outliers eller sesongvariasjoner i dataen vi har hentet inn.

Valg og tilpasning av modell

Hvilken modell som er best avhenger blant annet av den historiske dataen som er tilgjengelig, grad av sammenheng mellom variablene og hva modellen skal brukes til. Det er vanlig å sammenligne et knippe ulike modeller før man velger den endelige modellen.

Bruk og evaluering av modell

Når den endelige modellen er valgt og parameterne er estimert er modellen klar for å sette prognoser. Hvor bra modellen presterer kan evalueres ved hjelp av feilprognoser eller ved å sammenligne opp mot reell salgsdata for perioden man predikerer.

2.5 Prognosemetoder

Før man kan utarbeide prognoser er man nødt til å ta en rekke valg angående hvilken modell man skal bruke, hvor langt frem i tid man ønsker å se, og hva slags data man har tilgjengelig. Prognoser deles ofte opp etter tre tidshorisonter (Hyndman & Athanasopoulos, 2021):

Kortsiktige prognoser

Brukes i planlegging av personell, produksjon og transport. Som en del av planleggingsprosessen brukes det ofte for å lage etterspørselsprognoser.

Mellomsiktige prognoser

Brukes for å bestemme fremtidige ressursbehov. For å kjøpe råvarer, leie inn personell eller kjøpe maskiner og utstyr.

Langsiktige prognoser

Brukes i strategisk planlegging og i beslutninger som må tas i hensyn til markedsmuligheter, miljøfaktorer og interne ressurser.

Videre kan prognoser gjøres basert på enten et kvalitativt datagrunnlag eller et kvantitativt datagrunnlag. Uavhengig av prognosemetoden er det viktigste at modellen er i stand til å fange opp både mønstre og relasjoner i dataene uten å replikere tilfeldige tidligere hendelser (Hyndman & Athanasopoulos, 2021).

Brown et al. (2013) definerer tre tradisjonelle prognosemetoder: (1) tidsserier, (2) kausale metoder, og (3) kvalitative metoder. Vi starter med å se nærmere på noen ulike modeller innen tradisjonelle prognosemetoder som er nevnt i tabellen under.

TIDSSERIER	KAUSALE METODER	KVALITATIVE METODER
Glidende gjennomsnitt	Regresjonsmodeller	Delphi
Eksponentiell glatting	Kausale modeller	Ekspertjury
ARIMA		Scenarioanalyse
		Sales Force Composite
		Markedsundersøkelse

Tabell 1: Prognosemetoder.

2.5.1 Tidsserier

Tidsserier er en kvantitativ statistisk metode som hyppig brukes når man har et godt historisk datagrunnlag. Teknikken tar for seg hvordan en variabel x utvikler seg over tid, som for eksempel det månedlige salget av en vare. I en tidsseriemodell er det fire komponenter man ofte analyserer nærmere:

Trend

Om dataen har en trend vil det si at vi ser en stabil oppgang eller nedgang i dataen. Trenden i en tidsserie kan være både lineær og ikke lineær. En trend kan være både stigende, synkende eller stabil, til ulike tidspunkt innenfor en gitt periode. Samlet sett er den generelle trenden enten stigende, synkende eller stabil (*Components of Time Series*, n.d.).

Sesongvariasjon

Et vanlig fenomen i tidsseriedata og karakteriseres ved et mønster som gjentas med faste intervaller. En sesongvariasjon kan være til stede i datamaterialet dersom dataen er registrert med time- dag- uke- kvartal- eller månedsfrekvens. Vanlige grunner til at sesongvariasjoner oppstår er klima og de ulike årstidene (*Components of Time Series*, n.d.).

Sykler

Sykliske mønstre er lignende sesongvariasjon, men de gjentas med varierende intervaller.

Tilfeldig støy

De fleste tidsseriedata antas å inneholde både systematiske mønstre og tilfeldig støy. Tilfeldigheten gjør vanligvis mønsteret vanskelig å identifisere. Tilfeldige eller uregelmessige bevegelser, som støy, kan forårsake variasjoner i tidsserien. Denne støyen er gjerne uregelmessig og tilfeldig, som gjør at den er vanskelig å forutse. Støy kan forårsakes av blant annet faktorer som krig, naturkatastrofer, pandemier og lignende (*Components of Time Series*, n.d.).

2.5.1.1 Glidende gjennomsnitt:

Hvert punkt i et glidende gjennomsnitt av en tidsserie er det aritmetiske eller vektede gjennomsnittet av et antall tellingspunkter i serien, der antall datapunkter er valgt slik at effekten av sesongmessige og/eller andre uregelmessigheter elimineres (Hyndman & Athanasopoulos, 2021). Formelen for et glidende gjennomsnitt kan skrives som:

$$F_{t+1} = (Y_t + Y_{t-1} + Y_{t-2} + \cdots + Y_{t-[N-1])}/N \quad (1)$$

F_{t+1} er prognoser for perioden t+1.

Y_t er etterspørselen for perioden t.

N er antall perioder.

Formel 1: Glidende gjennomsnitt (Hyndman & Athanasopoulos, 2021).

2.5.1.2 Eksponentiell glatting

Følger samme metode som det glidende gjennomsnittet, bortsett fra at nyere datapunkter tillegges mer vekt. Rammeverket bak eksponentiell glatting generer pålitelige prognoser for et bredt spekter av tidsserier. Det finnes flere varianter av eksponentiell glatting og modellene varierer blant annet ut ifra hvorvidt man identifiserer trender, nivå og sesong i dataen (Hyndman & Athanasopoulos, 2021).

Enkel eksponentiell glatting

Som navnet antyder, er dette den enkle modellen innen eksponentiell glatting. Metoden er passende når man har data som ikke viser noen klar trend eller sesongvariasjon (Hyndman &

Athanasopoulos, 2021). Modellen består av en prognoseligning og en glattingsligning, disse kan skrives som:

$$\hat{y}_{t+h|t} = \ell_t$$
$$\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}$$

Formel 2: Enkel eksponentiell glatting (Hyndman & Athanasopoulos, 2021).

Dobbel eksponentiell glatting

Dobbel eksponentiell glatting, også kalt Holts metode, er en utvidelse av enkel eksponentiell glatting og tar hensyn til trend i dataen. Modellen består av en prognoseligning og to glattingskonstanter, nivå (α) og trend (β) (Hyndman & Athanasopoulos, 2021).

$$\hat{y}_{t+h|t} = \ell_t + hb_t$$
$$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$$
$$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$$

ℓt angir et estimat av nivået til serien på tidspunkt *t.*

Bt angir et estimat av trenden (hellingen) til serien på tidspunkt *t.*

A er en glattingsparameter for nivået, $0<\alpha<1$.

β^* er en glattingsparameter for trenden, $0<\beta^*<1$

Formel 3: Dobbel eksponentiell glatting (Hyndman & Athanasopoulos, 2021).

Trippel eksponentiell glatting

Trippel eksponentiell glatting, også kalt Holt-Winters metode, er Winters videreutvikling av Holts metode. Denne modellen består i tillegg til prognoseligningen av tre glattingskonstanter, nivå (α), trend (β) og sesong (γ). Det er to ulike varianter av modellen, multiplikativ og additiv. Den multiplikative metoden er hyppigst brukt og tar hensyn til at sesongvariasjonene endrer seg proporsjonelt med nivået i serien (Hyndman & Athanasopoulos, 2021). Den multiplikative modellen kan skrives som:

$$\hat{y}_{t+h|t} = (\ell_t + hb_t)s_{t+h-m(k+1)}$$
$$\ell_t = \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1})$$
$$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$$

$$s_t = \gamma \frac{y_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m}$$

ℓt angir et vektet gjennomsnitt mellom sesongjustert observasjon ($y_t - s_{t-m}$) og ikke-sesongbasert prognose ($\ell_{t-1} + b_{t-1}$) på tidspunkt *t.*

Bt angir et estimat av trenden (hellingen) til serien på tidspunkt *t.*

S_t viser den sesongmessige vektede gjennomsnittet mellom nåværende sesongindeks ($y_t - \ell_{t-1} - b_{t-1}$) og sesongindeks for den samme sesongen forrige år.

A er en glattingsparameter for nivået, $0<\alpha<1$.

β^* er en glattingsparameter for trenden, $0<\beta^*<1$

γ er en glattingsparameter for sesongvariasjonen, $0<\gamma<1-\alpha$.

Formel 4: Trippel eksponentiell glatting (Hyndman & Athanasopoulos, 2021).

2.5.1.3 ARIMA

ARIMA, også kalt Box-Jenkins er betegnelsen på en gruppe modeller brukt innen tidsserieanalyse. ARIMA står for Auto-Regressive Integrated Moving Average. I en ARIMA modell brukes det tre ulike notasjoner. De ulike notasjonene beskriver hvilken type modell man bruker. Det grunnleggende mønsteret er som følger: ARIMA(p,d,q) (Carlberg, 2018).
For å bedre forstå modellens oppbygning kan vi videre dele den opp i tre modeller:

AR – Auto-Regressive(p)

Modellen og prognosene kan baseres delvis eller helt på autoregresjon. P er antall autoregressive parametere i modellen. Ligningen for en AR-modell kan skrives som:

$$\hat{y}_t = c + \Phi_1 y_{t-1} + \Phi_2 y_{t-2} + \cdots + \Phi_p y_{t-p} + \varepsilon_t$$

c er en konstant.

E er hvit støy på tid t.

Φ er parametere.

Formel 5: Auto-Regressive.

I – Intergrated(d)

Datagrunnlaget må muligens differensieres for å bli stasjonært. D er antall ganger serien har blitt differensiert for å oppnå stasjonaritet. En 1. ordensdifferanse kan derfor skrives som:

$$\acute{y}_t = y_t - y_{t-1} = (1 - B)y_t$$

B er en "backshift operator", også kalt lag.

Formel 6: Intergrated.

MA – Moving Average(q)

Det glidende gjennomsnittet referer til et gjennomsnitt av prognosefeilene. Q er antall glidende gjennomsnittsparametere i modellen. Ligningen for en MA-modell kan skrives som:

$$\hat{y}_t = c + \Phi_1\varepsilon_{t-1} + \Phi_2\varepsilon_{t-2} + \cdots + \Phi_q\varepsilon_{t-q}$$

c er en konstant.

E er hvit støy på tid t.

Φ er parametere.

Formel 7: Moving Average.

Når vi slår sammen de tre overnevnte modellene blir resultatet en ARIMA(p,d,q) modell:

$$\hat{\acute{y}}_t = c + \Phi_1\acute{y}_{t-1} + \Phi_2\acute{y}_{t-2} + \cdots + \Phi_p\acute{y}_{t-p} + \Phi_1\varepsilon_{t-1} + \Phi_2\varepsilon_{t-2} + \cdots + \Phi_q\varepsilon_{t-q} + \varepsilon_t$$

Formel 8: ARIMA.

SARIMA

Hvis datagrunnlaget viser sesongmessige mønstre kalles modellen for Seasonal Auto-Regressive Integrated Moving Average. Når vi har sesongmessige mønstre, må vi legge til 3 parametere i grunnmodellen. Modellen blir da som følger: ARIMA(p,d,q)(P, D,Q). I denne modellen representerer de store bokstavene det samme som de små, men er gjeldene for sesongmessige parametere (Carlberg, 2018).

SARIMAX

Seasonal Auto-Regressive Integrated Moving Average with eXogenous factors. Modellen er en videreutvikling av SARIMA og i tillegg til de sesongmessige parameterne tar modellen også høyde for eksogene faktorer.

2.5.2 Kausale metoder

Metoden ser på hvordan etterspørselen varierer basert på interne og eksterne faktorer gjennom en årsakssammenheng. I kausale metoder inkluderer man ofte økonomiske og sosioøkonomiske faktorer i tillegg til historisk data.

2.5.2.1 Regresjonsmodeller

Disse modellene undersøker forholdet mellom hvordan en avhengig variabel kan påvirkes av en eller flere uavhengige variabler. Det finnes en rekke regresjonsmodeller, både lineære og ikke-lineære. Generelt kan en standard regresjonsmodell skrives som:

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

Hvor y er den avhengige variabelen.

B0 er skjæringspunktet.

B1 angir helningen til linjen.

X er den uavhengige variabelen.

Et er modellens feilledd.

Formel 9: Regresjonsmodell.

2.5.2.2 Økonometriske modeller

I noen tilfeller kan de estimerte parameterne ved standard regresjonsanalyse bli upassende grunnet det svært dynamiske forholdet mellom de avhengige og uavhengige variablene. En måte å håndtere dette på er å bruke et sett med simultane regresjonsmodeller for å beskrive dynamikken, kalt økonometriske modeller (Wang & Chaovalitwongse, 2011).

2.5.3 Kvalitative metoder

Kvalitative metoder benyttes ofte når man har lite eller ingen historisk data, for eksempel når et nytt produkt lanseres på markedet. I denne metoden brukes dømmekraft og vurderinger for å gjøre kvalitativ informasjon om til kvantitative estimater. Målet her er å samle på en logisk, objektiv og systematisk måte all informasjon og vurderinger som er relatert til faktorene som estimeres.

2.5.3.1 Delphi

Et panel av eksperter blir bedt om å besvare en serie spørreundersøkelser anonymt over flere runder. I første runde blir panelmedlemmene bedt om å skrive ned sine intuitive prognoser, deretter blir svarene oppsummert og delt med hvert panelmedlem. Hensikten med denne prosessen er å innsnevre meninger. Undersøkelsene er ferdig enten etter et visst antall runder, eller når man når en viss grad av konsensus eller stabilitet i resultatene. Denne teknikken eliminerer den såkalte bandwagon-effekten, som er et psykologisk fenomen der folk gjør noe først og fremst fordi andre mennesker gjør det, uavhengig av deres egen tro (Wang & Chaovalitwongse, 2011).

2.5.3.2 Ekspertjury

En ekspertjury, ofte bestående av selskapets ledere, blir spurt om deres beste estimat av fremtidige trender og forventet salg. Denne metoden er en av de enkleste og mest brukte prognosemetodene i næringslivet. Hovedforskjellen mellom denne metoden, og Delphi er at i en ekspertjury forhindrer man ikke interaksjoner mellom deltagerne (Wang & Chaovalitwongse, 2011).

2.5.3.3 Scenarioanalyse

Denne metoden går ut på å kombinere ulike politiske planer med alle kjente fakta og forventede endringer om fremtiden. Scenarioanalyse kan føre til plausible prognoser dersom årsakssammenhengen mellom faktorer og deres konsekvenser har blitt simulert riktig (Wang & Chaovalitwongse, 2011).

2.5.3.4 Sales Force Composite

Ulikt fra ekspertjury hvor man bruker selskapets ledere bruker denne metoden selskapets selgere. Årsaken til dette er fordi selgerne er i direkte kontakten med kunden, og må kunne antas å ha et

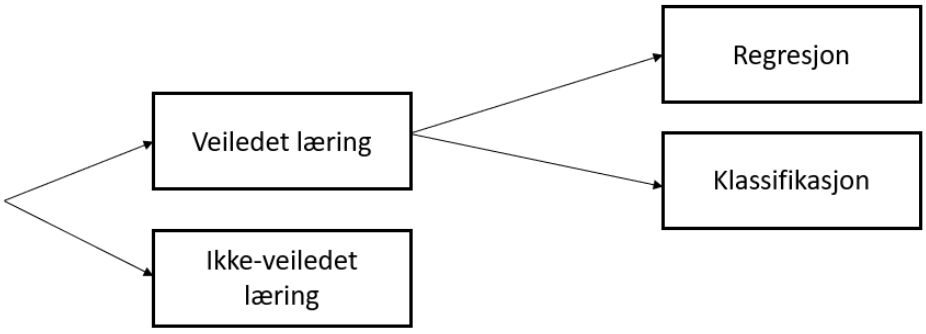
estimat for salg, endringer og trender i markedet. En ulempe som trekkes frem med denne metoden er at selgernes prognoseresultater har en tendens til å være i overkant optimistiske (Wang & Chaovalitwongse, 2011).

2.5.3.5 Markedsundersøkelse

En populær tilnærming for å forutsi fremtidige trender. Metoden går ut på å spørre kunder eller mulige brukere om hvordan de forutser deres fremtidige forbruk av et produkt eller en tjeneste. Denne metoden er ment å gi prognoser uten unødig optimisme, da kundene/brukerne ikke har en offisiell tilknytning til selskapet det spørres om. En ulempe med denne metoden er i midlertidig at brukernes meninger kan være sterkt preget av øyeblikkets tilstand. For eksempel blir det rapportert flest negative meninger i perioder med lavkonjunktur, og flere positive meninger i perioder med høykonjunktur (Wang & Chaovalitwongse, 2011).

2.5.4 Maskinlæringsmetoder

Begrepet maskinlæring referer til automatisert gjenkjenning av mønstre i data. Maskinlæringsmetoder kan som illustrert i figur 5, deles inn i veiledet læring og ikke-veiledet læring. Videre derfra kan veiledet læring deles inn i regresjon og klassifikasjon. En regresjonsalgoritme undersøker hvordan en eller flere uavhengige variabler påvirker den avhengige variabelen. En klassifikasjonsalgoritme analyserer treningsdataen for å forutsi resultatet og plasserer det i en målklasse, for eksempel "ja" eller "nei", "høy" eller "lav" osv. (Shalev-Shwartz & Ben-David, 2014).



Figur 5: Maskinlæringsmetoder.

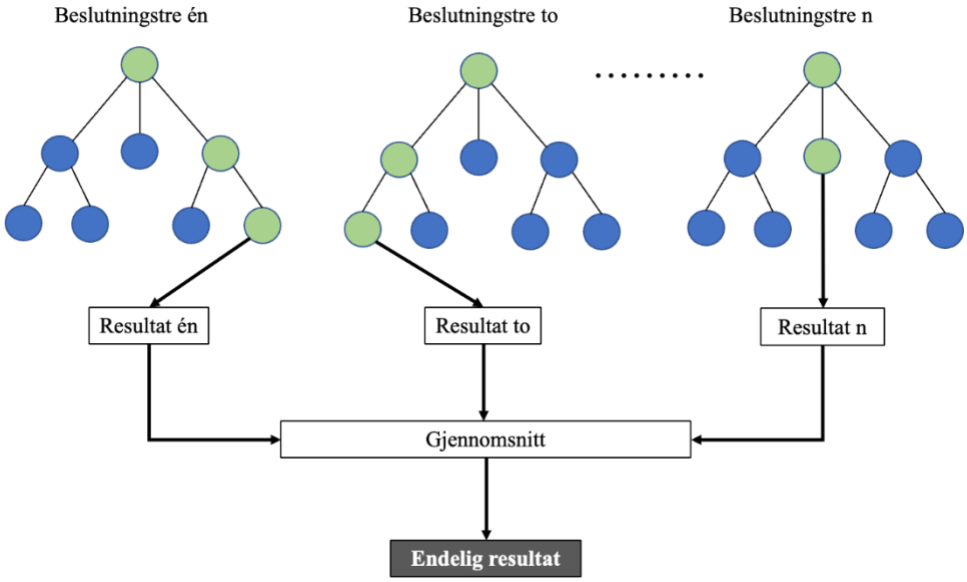
En maskinlæringsmodell vil ofte kunne brukes til både regresjon og klassifikasjon. For denne oppgavens formål vil vi bruke regresjonsmodeller. Nedenfor følger tre eksempler på maskinlæringmodeller som kan benyttes til predikasjon av salgsprognoser.

2.5.4.1 Light Gradient Boosting

Light Gradient Boosting (LGB) er en av flere Gradient Boosting modeller. Bruken av denne typen modeller i prognosearbeid har økt i løpet av de siste årene da modellene det har vist seg at de kan oppnå svært konkurransedyktige resultater. Noen av fordelene med å bruke en LGB modell er at det er enkelt å inkludere eksogene variabler i tillegg til autoregressive og de tillater å inkludere ikke-lineære relasjoner i modellen (Rodrigo, 2022).

2.5.4.2 Random Forest

En random Forest består av et ensemble av beslutningstrær og har derfor lavere varians enn andre maskinlæringsalgoritmer. I denne teknikken lages det en rekke beslutningstrær som alle gir et resultat. Resultatene fra alle beslutningstrærne blir da lagt sammen og et gjennomsnitt blir beregnet. Dette gir oss modellens endelige resultat. Tanken bak denne metoden er at gjennomsnittet til et flertall av modeller predikerer bedre enn en modell alene.



Figur 6: Random Forest.

2.5.4.3 Lineær regresjon

Lineær regresjon er både en statistisk modell og en maskinlæringsmodell. I denne modellen undersøker vi det lineære forholdet mellom den avhengige variabelen y og den uavhengige variabelen x. Formelen for lineær regresjon er definert under kausale metoder.

2.6 Prognosefeil

Flere metoder har blitt utviklet for å oppsummere feilene som genereres av en bestemt prognosemodell. De fleste av disse metodene innebærer en beregning av differansen i den observerte verdien og predikerte verdien i en prognosemodell. Disse forskjellene mellom observert og predikert verdi er det som kalles prognosefeil (Hanke & Wichern, 2014). Prognosefeilen kan skrives som:

$$e_t = Y_t - \hat{Y}_t$$

e_t = prognosefeilen i periode t.

Y_t= observert verdi i periode t.

$\hat{Y}_t$ = predikert verdi i periode t.

Formel 10: Prognosefeil (Hanke & Wichern, 2014).

2.6.1 Mean Absolute Deviation (MAD)

En metode for å måle feilprognosene er ved å beregne Mean Absolute Deviation. Denne metoden måler gjennomsnittet av absoluttfeilene i en prognosemodell (Hanke & Wichern, 2014). Formelen for MAD kan skrives som:

$$MAD = \frac{1}{n}\sum_{t=1}^{n}|Y_t - \hat{Y}_t|$$

Formel 11: Mean Absolute Deviation (Hanke & Wichern, 2014).

2.6.2 Mean Squared Error (MSE)

Mean Squared Error er en metode hvor hver enkelt prognosefeil blir kvadrert, summert og delt på antall observasjoner. Man får da gjennomsnittet til de kvadrerte feilene. Ved å kvadrere feilene fjerner vi eventuelle negative fortegn og tillegger størst vekt på de større verdiene (Hanke & Wichern, 2014). Formelen for MSE er gitt ved:

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

Formel 12: Mean Squared Error (Hanke & Wichern, 2014).

2.6.3 Root Mean Squared Error (RMSE)

I likhet med Mean Squared Error tillegger denne metoden også størst vekt på de større verdiene. Denne metoden er lettere å tolke da den har samme enhet som serien vi predikerer (Hanke & Wichern, 2014). Formelen for RMSE er:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(Y_t - \hat{Y}_t)^2}$$

Formel 13: Root Mean Squared Error (Hanke & Wichern, 2014).

2.6.4 Mean Absolute Percentage Error (MAPE)

Prognosefeil kan også beregnes prosentvis. Denne typen prognosefeil har fordelen av å være enhetsfri og brukes derfor ofte til å sammenligne prognoseytelser mellom ulike datasett. Metoden er også svært gunstig å bruke når Yt verdiene er høye (Hanke & Wichern, 2014). Formelen for MAPE er:

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|Y_t - \hat{Y}_t|}{|Y_t|}$$

Formel 14: Mean Absolute Percentage (Hanke & Wichern, 2014).

Beslutningen om å bruke en bestemt prognosemodell er delvis basert på om modellen vil gi prognosefeil som vurderes til å være tilstrekkelig lave. Det er å forvente at en god prognosemodell vil produsere relativt lave og stabile prognosefeil (Hanke & Wichern, 2014). Hensikten til de fire målene nevnt over er:

- For å sammenligne nøyaktigheten til to eller flere prognosemodeller.
- Å måle en bestemt modells nytte eller pålitelighet.
- Som et hjelpemiddel for å finne en optimal modell.

Vi vil i denne oppgaven vurdere resultatene til de ulike modellene ved å bruke Root Mean Squared Error. Bakgrunnen for dette valget er at resultatene er lette å tolke og sammenligne, og

siden vi bruker samme data på alle modellene er det tilstrekkelig med en modell til å dømme resultatene.

3.0 Metode og data

3.1 Forskningsdesign

Bell et al. (2018) definerer forskningsdesign som et rammeverk for innsamling og analyse av data. Valget av forskningsdesign er knyttet til fenomenet man undersøker, og hva man ønsker svar på. Noen viktige punkter man ønsker å få svar på ved hjelp av riktig forskningsdesign er:

- Årsakssammenhenger mellom variabler.
- Generalisering til større grupper av individer enn de som faktisk inngår i etterforskningen.
- Forstå atferd og betydningen av atferden i dens spesifikke sosiale kontekst.
- Å ha en tidsmessig (dvs. over tid) forståelse av sosiale fenomener og deres sammenhenger.

De ulike strategiene som brukes innen forskning er kvalitativ, kvantitativ eller en blanding av begge to. Ut ifra disse strategiene definerer Bell et al. (2018) fem ulike typer forskningsdesign: eksperimenter, tverrsnittstudie, longitudinelle studier, casestudie og komparative studier.

Begrepet casestudie er studiet av en spesifikk case og kan knyttes til en geografisk plassering, som for eksempel en arbeidsplass eller en organisasjon. Det som skiller casestudier fra andre forskningsdesign, er fokuset på en avgrenset situasjon (Bell et al., 2018). Vår studie har som formål å undersøke og sammenligne en rekke prognosemodeller for å finne den best egnede modellen for et spesifikt selskap, som i dag ikke har implementert prognosemetoder i sine prosesser.

På bakgrunn av dette er casestudie valgt som forskningsdesign. Dette gjør oss i stand til å finne ut hvilke prognosemetoder, enten det er de tradisjonelle eller maskinlæringsmodellene som vil være best egnet å implementere for Sæther i fremtiden.

Forskningsmetoder kan assosieres med ulike typer forskningsdesign og begrepene blir ofte forvekslet. Casestudier blir ifølge Bell et al. (2018) ofte referert til som en metode. Å bestemme seg for en organisasjon og studere den vil ikke alene skaffe datamateriale. Bell et al. (2018) forklarer forskningsmetoder som teknikken for å samle inn data. Etter en case er bestemt er det

derfor nødvendig med valg av en eller flere forskningsmetoder for å kunne samle inn data. Noen vanlige metoder for å samle inn data er blant annet gjennom observasjoner, intervjuer, dokumentundersøkelser eller spørreundersøkelser. Datamaterialet i denne studien er basert på salgsdata og er hentet ut fra Sæther sine systemer, og vil bli gått gjennom i kapittel 3.3.

3.1.1 Metode

Å bruke en metode handler om å følge en bestemt vei mot et mål gjennom innsamling, analyse og tolkning av data (Johannessen et al., 2020). Det handler om hvordan vi kan gå frem for å undersøke en valgt problemstilling opp imot virkeligheten, uten å trekke konklusjoner for raskt. Bell et al. (2018) forklarer en forskningsstrategi som en bred orientering til forretnings- og ledelsesforskning, og skiller mellom kvalitativ og kvantitativ forskning som ulike strategier. I denne oppgaven baserer vi oss på en kvantitativ metode, da vi har et datagrunnlag bestående av salgshistorikk.

3.2 Oppgavens kvalitet

3.2.1 Reliabilitet

Reliabilitet, eller pålitelighet, brukes om hvor stabile målingene er, og hvorvidt vi får samme resultat dersom vi utfører målingen flere ganger. Bell et al. (2018) definerer stabilitet som en nøkkelfaktor når man skal vurdere om en måling er reliabel eller ikke. Det er derfor nødvendig med stabile målinger over tid for å kunne trekke konklusjoner om at resultatet er reliabelt.

3.2.2 Validitet

Validitet betyr i hvilken grad man ut fra resultatene kan trekke gyldige slutninger om det man har satt seg som formål å undersøke (Dahlum, 2021). Det er vanlig å ytterlig dele validitet opp i indre validitet og ytre validitet.

Indre validitet brukes om hvorvidt en årsakssammenheng mellom to eller flere variabler kan forklares gjennom den antatte hypotesen (Bell et al., 2018).

Ytre validitet brukes om hvorvidt resultatene fra en studie av et begrenset omfang kan generaliseres, og dermed gjelde for en større mengde data enn det studien undersøkte (Dahlum, 2021). Oppgavens ytre validitet er særdeles interessant for vår oppgave da vi bruker en mindre

del med data for å lage prognosemodeller. Dersom vi har ytre validitet, kan resultatene våre være generaliserbare og gjøres gjeldende for større deler av selskapets totale salg.

3.3 Datainnsamling

For å kunne undersøke hvilke av de ulike prognosemodellene som er best egnet er vi nødt til å ha et datagrunnlag. I denne oppgaven har vi fått salgsdata hentet ut fra Sæther sin database. Til datainnsamlingen har vi fått god hjelp fra Ricardo Hernandez, Nordic Power BI hos Sæther, til å hente frem salgstall fra 2019 til 2021. Ettersom Sæther selger produkter innen ulike kategorier har vi valgt å fokusere oppgavens datagrunnlag på kun en av kategoriene. Etter en samtale med Ricardo, om hvilken kategori vi burde velge har vi endt opp med kategorien hudpleie, da dette er en kategori hvor vareflyten er relativt jevn og vi derfor kan sikre et godt datagrunnlag.

3.3.1 Klargjøring av data

Han et al. (2011) poengterer at dataen man bruker må ha en viss kvalitet for å tilfredsstille kravene til tiltenkt bruk. Det er mange faktorer som kan påvirke datakvalitet, disse inkluderer: nøyaktighet, fullstendighet, konsistens, aktualitet, troverdighet og tolkbarhet. For å sørge for at dataen man bruker har god nok kvalitet er vi nødt til å forbehandle dataen.

Videre deler Han et al. (2011) prosessen for dataforbehandling inn i fire steg:

Data rengjøring

Data rengjøring, også kalt datavask, har som oppgave å «rense» dataen ved å fylle inn manglende verdier, utjevne støyende data og identifisere eller fjerne outliers.

Dataintegrasjon

Om man ønsker å benytte data fra ulike kilder, som for eksempel databaser, datakuber eller filer må disse slås sammen gjennom dataintegrasjon.

Datareduksjon

Gjennom datareduksjon er målet å redusere datasettets størrelse uten at det påvirker de analytiske resultatene. Dette kan gjøres blant annet gjennom dimensjonalitetsreduksjon og tallreduksjon.

Datatransformasjon

Det finnes en rekke teknikker for å transformere dataen man skal bruke. Noen eksempler på datatransformasjon er aggregering, normalisering og hierarkigenerering.

For å kvalitetssikre vår data har vi startet med data rengjøring. I denne prosessen har vi fjernet overflødige variabler og alle linjer som inneholder produkter under andre kategorier enn hudpleie. Med dataen vi mottok fra Sæther har vi prøvd å dele inn salget etter ulike tidsfrekvenser. Vi startet med å se på det månedlige salget, men fant fort ut at dette ga oss for få observasjoner. For å få et større antall observasjoner prøvde vi oss frem med den daglige salgsdataen. I prosessen med å rengjøre den daglige dataen som ikke hadde registrerte observasjoner for hver dag, ble alle blanke felter erstatte med verdien «0». Den daglige dataen ga oss et høyt antall observasjoner, men ettersom det ikke selges produkter hver dag fikk vi et datasett med mye støy og svært varierende observasjoner. Noen eksempler på grafer fra da vi testet daglig salgsdata finnes i vedlegg 2. For å minimere støy og samtidig beholde et greit antall observasjoner endte vi derfor med å aggregere den ukentlige salgsdataen.

Vi har også tatt i bruk hierarkigenerering. I tidsserier kan data deles inn på flere nivåer basert på geografiske eller logiske grunner til å danne hierarkiske strukturer. Et eksempel er at enkeltsalg av produkter kan grupperes i kategorier og familier av relaterte produkter (Oliveira & Ramos, 2019). I kategorien hudpleie finnes det en rekke underkategorier, som for eksempel, solkrem, øyekrem, hudkrem, cleanser osv. Vi har da valgt å se på underkategorien cleanser. For disse produktene har vi valgt å slå sammen den totale etterspørselen, vi ser derfor ikke på prognoser for hvert enkelt produkt, men for kategorien cleanser sammenlagt. Studier på hierarkiske prognosemetoder viser til at resultatene av disse prognosene bør vurderes for beslutningstaking på de forskjellige nivåene (Yagli et al., 2019).

3.4 Datahåndtering og analyse

3.4.1 Programvare og programmeringsspråk

I denne studien er det tatt i bruk ulike programvarer og et programmeringsspråk for å håndtere og analysere data, samt for å teste og sammenligne ulike prognosemodeller.

For denne oppgaven har vi tatt i bruk Microsoft Excel og programmeringsspråket Python. Excel er brukt for å sette sammen et passende datasett med alle nødvendige variabler. Videre er datasettet konvertert til en csv fil og lastet opp i Python. For de ulike analysene i Python har vi importert en rekke pakker tilhørende de ulike modellene. Videre er en rekke av kodene og oppsettet hentet fra Effrosynidis (2020) sin GitHub «Forecasting Wars».

```
import time
import warnings
import numpy as np
import pandas as pd
import seaborn as sns
import lightgbm as lgb
from itertools import cycle
from sklearn.svm import SVR
import statsmodels.api as sm
from pmdarima import auto_arima
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.tsa.holtwinters import SimpleExpSmoothing, ExponentialSmoothing
from statsmodels.tsa.seasonal import seasonal_decompose
```

Figur 7: Importerte pakker.

3.4.2 Deskriptiv analyse

Før vi tester de ulike prognosemodellene vil det kunne være hensiktsmessig å utforske datasettet gjennom en deskriptiv analyse. Målet med deskriptiv statistikk er å oppsummere og beskrive variablene i datasettet man bruker (Fisher & Marshall, 2009).

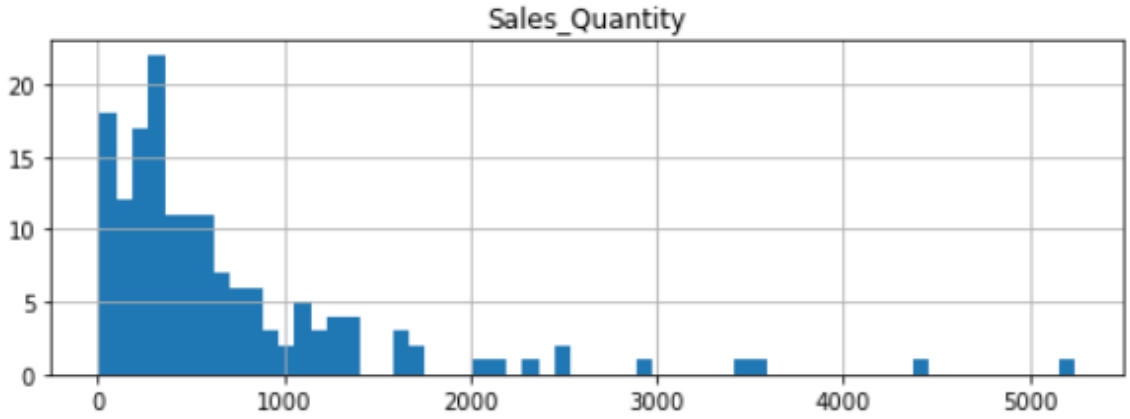
Dataen vi har innhentet består av ukentlig salgsdata over en periode på 3 år som gir totalt 157 observasjoner. Som det fremkommer fra tabell 2 er det store variasjoner i antall solgte kvantiteter, med et minimum på 11 og et maksimum på 5245. Antall kvantiteter solgt per år øker med 60% fra 2019 til 2020, før det igjen synker med ca. 5% fra 2020 til 2021.

Sales Quantity	
Gjennomsnitt	687
Median	436

Min	11
Maks	5245
Sum 2019	26113
Sum 2020	41848
Sum 2021	39845
Sum total	107806
Antall observasjoner	157

Tabell 2: Deskriptiv analyse.

Ved hjelp av et histogram er det mulig å få en oversikt over fordelingen av størrelsen på de ulike observasjonene. Histogrammet nedenfor viser at de fleste observasjonene inneholder et salgsantall på mellom 11 og 1000 produkter, og at større observasjoner forekommer sjeldnere.



Figur 8: Histogram over salgskvantiteter.

Ved å utforske salgstallene nærmere i Python, ser vi at det er 49 observasjoner som er større enn gjennomsnittet, og 108 observasjoner som ligger under gjennomsnittet. Dette viser at de store observasjonene som forekommer sjeldnere drar opp gjennomsnittet som er på 687. Ved å se på ordre med kvantum under 1500, som utgjør 142 av totalt 157 observasjoner, ville gjennomsnittlig kvantum vært 482.

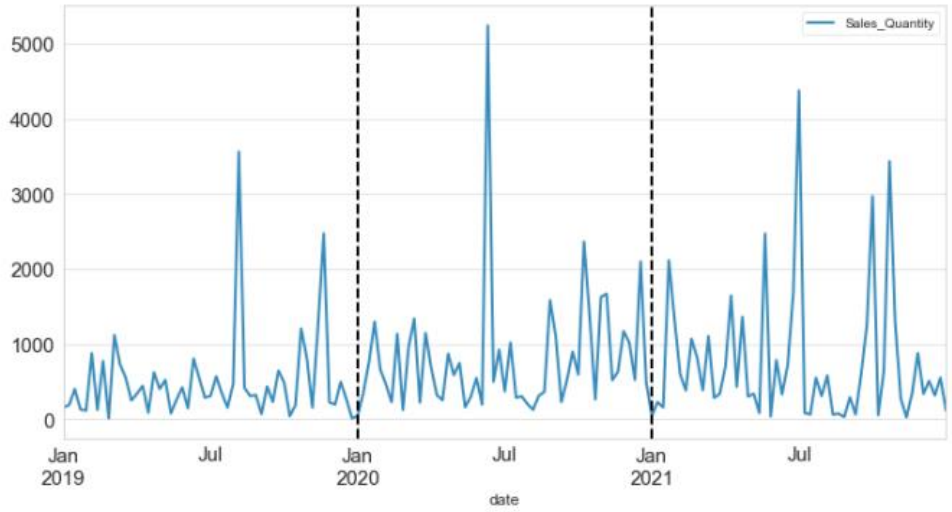
Antall observasjoner:	
Over gjennomsnittet	49
Under gjennomsnittet	108
Under 100	18
Over 1000	33

Tabell 3: Antall observasjoner.

3.4.3 Komponenter i tidsserien

For å forstå oppgavens datamateriale bedre finnes det en rekke ulike teknikker som kan benyttes. Tidsserier kan bruke linjediagrammer for å blant annet kunne visualisere og kartlegge ulike komponenter. En prognose kan ikke forutsi fremtiden, men kan gi oss et beregnet estimat basert på hva som allerede har skjedd for å kunne gi oss en ide om hva som kan skje i fremtiden. Det kan være vanskelig å forutse hvordan noe skal bli i fremtiden, da eksterne faktorer også kan ha en innvirkning. Ulike faktorer som kan påvirke verdiene i en tidsserie er komponentene. Noen vanlige komponenter i tidsserier er; trend, sesongvariasjon og støy. Disse komponentene ble nærmere beskrevet i delkapittel 2.5.1.

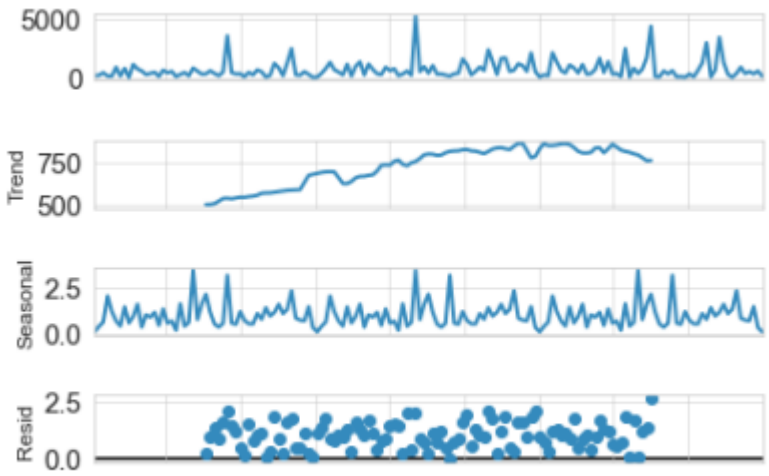
Ved hjelp av Python plottes datamaterialet for å forstå og kartlegge om det er komponenter til stede. Grafen nedenfor viser historisk salg fra 2019 til 2021. Et gjentakende mønster er at salgstallene øker rundt sommertider, og før juletider.



Figur 9: Historisk salg 2019-2021.

Figur 9 viser det solgte antallet fra 2019 til 2021 aggregert på ukenivå. Fra grafen kan vi se at det er fem topper som skiller seg ut.

Videre dekomponeres tidsserien ved hjelp av modellen seasonal_decompose fra statsmodels. Statsmodels er en modul i Python som tilbyr ulike funksjoner som kan brukes til å estimere en rekke ulike statistiske modeller, samt brukes til gjennomføring av statistiske tester og utforskning av data. Resultater testes mot andre eksisterende statistiske pakker for å sikre at resultatene er korrekte (Seabold & Perktold, 2010). Dekomponering gjøres ved at de ulike komponentene, trend, sesongvariasjon og støy, brytes ned og blir visualisert som grafer.



Figur 10: Komponenter i tidsserien.

Det er tydelig at datamaterialet inneholder en stigende trend. En trend kan være både stigende, synkende eller stabil, til ulike tidspunkt innenfor en gitt periode. Men samlet sett er den generelle trenden enten stigende synkende eller stabil (*Components of Time Series*, n.d.) Dette er også tilfellet her, hvor grafen beveger seg opp og ned til ulike tidspunkt, men er samlet sett stigende over de tre årene. En økende trend samsvarer med tallene vi så i den deskriptive analysen, hvor antall kvantiteter solgt per år økte med 60% fra 2019 til 2020, før det synker med ca. 5% fra 2020 til 2021.

3.5 Valgte prognosemodeller

Vi har valgt ut totalt seks tradisjonelle modeller og tre maskinlæringsmodeller som vi ønsker å trene og evaluere. Blant de tradisjonelle modellene har vi valgt ut enkel eksponentiell glatting, dobbel eksponentiell glatting, trippel eksponentiell glatting, ARIMA, SARIMA og SARIMAX. Disse modellene er valgt på bakgrunn av at de er tidsseriemodeller, og at alle modellene i begynnelsen er relativt enkle før de går videre til å ta for seg andre faktorer som blant annet trend og sesong i ulik grad. For maskinlæringsmodellene har vi valgt å teste Light Gradient Boosting, Random Forest og lineær regresjon. Disse modellene varierer også i kompleksitet, med lineær regresjon som er den enkleste, til LGB og Random Forest som er mer komplekse. Modellene testes på det samme datagrunnlaget og sammenlignes med prognosefeilen Root Mean Squared Error.

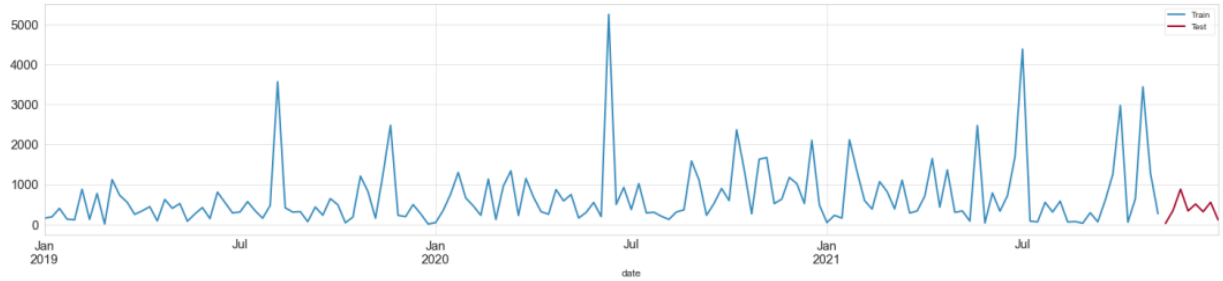
3.5.1 Datastruktur

Formålet er å kunne predikere fremtidig salg for produktgruppen «cleanser» gjennom data som inneholder historisk salg. Tabellen nedenfor viser en oversikt over hvordan datasettet ser ut. Vi bruker da variabelen Sales_Quantity, som består av alle solgte kvantiteter fra 2019 til 2021 for produktgruppen.

	id	year	month	week	date	Sales_Quantity	Sales_Netto_Amount
0	cleanser	2019	1	1	2019-01-04	158	15567
1	cleanser	2019	1	2	2019-01-11	190	19605
2	cleanser	2019	1	3	2019-01-18	402	37560
3	cleanser	2019	1	4	2019-01-25	134	13132
4	cleanser	2019	2	5	2019-02-01	115	13242

Figur 11: Oversikt over datasettet.

For å kunne trene og evaluere de ulike modellene deles datasettet inn i to deler: et trenings- og et testsett. Testsettet kommer til å være de åtte siste ukene av datasettet, altså datoer etter 05-11-2021 og frem til 21-12-2021. Prediksjoner fra de ulike modellene vil testes og sammenlignes mot det faktiske salget i testsettet. Treningssettet som brukes for å trene modellene vil da inneholde alle uker som ikke inngår i testsettet. Grafen nedenfor illustrerer forholdet mellom trenings- og testsettet.



Figur 12: Trening- og testsett.

3.5.2 Tradisjonelle modeller

3.5.2.1 Eksponentiell glatting

Det finnes flere varianter av eksponentiell glatting og modellene varierer i kompleksitet, alt etter om de er i stand til å identifisere trender eller sesongvariasjon. I denne studien er det brukt både enkel- dobbel- og trippel eksponentiell glatting. Teorien om de valgte modellene er gjennomgått i delkapittel 2.5.1.2.

For å predikere fremtidig salg ved hjelp av eksponentiell glatting modeller er pakker fra modulen statsmodels i Python tatt i bruk. Ettersom det brukes tidsseriedata i studien, hentes det ut relevante pakker fra statsmodels.tsa. Tsa er en forkortelse for time series analysis, og modulen inneholder diverse modeller og funksjoner som kan være til hjelp når man arbeider med tidsseriedata. Pakken SimpleExpSmoothing brukes for enkel eksponentiell glatting, og pakken ExponentialSmoothing brukes for dobbel- og trippel eksponentiell glatting. For å kunne bruke pakkene importeres disse til Python.

Enkel eksponentiell glatting adresserer ikke trend eller sesongvariasjon, og inneholder kun glattingsparameteren nivå (α). Glattingsparameteren, nivå (α), er til stede i alle variantene av eksponentiell glatting modellene. Det er denne parameteren som bestemmer hvor mye vekt modellen skal legge på siste salgsobservasjon. I enkel eksponentiell glatting modellen defineres glattingsparameteren ved hjelp av parameteren span. Parameteren span er i denne modellen satt lik antall sesongmessige perioder. Parameteren er knyttet til antall observasjoner per syklus, altså 52. Figur 11 viser et utklipp fra koden som er brukt i Python for å lage modellen. Fullstendig script fra Python for studien som viser koder for alle modeller er lagt ved som vedlegg 1.

```
t0 = time.time()
model_name='Enkel Eksponentiell Glatting'
span = 52
alpha = 2/(span+1)

##Tren
simpleExpSmooth_model = SimpleExpSmoothing(train['Sales_Quantity']).fit(smoothing_level=alpha,optimized=False)
t1 = time.time()-t0

##Prediker åtte uker
prediksjoner[model_name] = simpleExpSmooth_model.forecast(8).values

##Graf som plotter åtte uker av treningsdelen med åtte uker av testdelen
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t1, 'RMSE':score},ignore_index=True)
```

Figur 13: Kode for enkel eksponentiell glatting.

For dobbel eksponentiell glatting inkluderer vi i tillegg til glattingsparameteren nivå (α), en glattingsparameter for trend (β).

```
t0 = time.time()
model_name='Dobbel Eksponentiell Glatting'

##tren
doubleExpSmooth_model = ExponentialSmoothing(train['Sales_Quantity'],trend='add',seasonal_periods=52).fit()
t1 = time.time()-t0
```

Figur 14: Kode for dobbel eksponentiell glatting.

I trippel eksponentiell glatting blir en tredje parameter lagt til. Denne glattingsparameteren tar for seg sesong (γ).

```
t0 = time.time()
model_name='Trippel Eksponentiell Glatting'

##Tren
tripleExpSmooth_model = ExponentialSmoothing(train['Sales_Quantity'],trend='add',seasonal='add',seasonal_periods=52).fit()
t1 = time.time()-t0
```

Figur 15: Kode for trippel eksponentiell glatting.

I både dobbel- og trippel eksponentiell glatting definerer vi parameteren seasonal_periods. I likhet med parameteren span fra enkel eksponentiell glatting er denne også satt til 52, da parameteren er knyttet til antall observasjoner per syklus.

3.5.2.2 ARIMA

Til ARIMA modellene bruker vi pakken auto_arima fra det statistiske biblioteket pmdarima. Pakken auto_arima finner de optimale parameterne for en ARIMA modell. Parameterne (p, d, q) (P, D, Q) for ARIMA modeller er gjennomgått i kapittel 2.5.1.3.

For ARIMA velger pakken ut følgende parametere:

```
Best model:  ARIMA(1,1,2)(0,0,0)[0]
Total fit time: 5.220 seconds
```

Figur 16: Optimale parametere ARIMA.

SARIMA er en forlengelse av ARIMA og inkluderer et ekstra sett av parametere som tar for seg det sesongmessige mønstre i tidsserien. Når seasonal er aktivert søker og identifiserer auto_arima også parameterverdiene (P, D, Q). I SARIMA bruker vi to ekstra parametere i forhold til ARIMA; m og seasonal. Parameteren m referer til antall perioder i hver sesong. Ettersom datamaterialet er i ukesfrekvens er denne parameteren satt lik 52. De optimale parameterne for SARIMA bestemmes til å være:

```
Best model:  ARIMA(1,1,2)(0,0,0)[52]
Total fit time: 118.281 seconds
```

Figur 17: Optimale parametere SARIMA.

I motsetning til ARIMA og SARIMA som lar oss bruke kun historisk data, introduserer SARIMAX ideen om at eksterne faktorer også kan ha en innvirkning på tidsserier. I SARIMAX modellen legger vi til en ekstra variabel, Sales_Netto_Amount, for å assistere modellen ved prediksjon av fremtidig salg. De optimale parameterne for SARIMAX bestemmes til å være:

```
Best model:  ARIMA(0,1,1)(0,0,1)[52]
Total fit time: 295.657 seconds
```

Figur 18: Optimale parametere SARIMAX.

3.5.3 Maskinlæringsmodeller

Til maskinlæringsmodellene laster vi ned pakkene lightgbm, sklearn.ensemble og sklearn.linear_model i Python. Det som skiller maskinlæringsmodellene fra de tradisjonelle er at

vi i tillegg til å bruke den historiske salgsdataen også er nødt til å konstruere noen såkalte «features», også kalt egenskaper. Egenskapene vi lager er få og relativt enkle.

Lag 2: Solgte kvantiteter 2 uker tidligere.

Rullende gjennomsnitt 2_2: på lag 2 lages det et rullende gjennomsnitt for 2 uker tilbake i tid.

```
def lags_windows(df):
    lags = [2]
    lag_cols = ["lag_{}".format(lag) for lag in lags ]
    for lag, lag_col in zip(lags, lag_cols):
        df[lag_col] = df[["id","Sales_Quantity"]].groupby("id")["Sales_Quantity"].shift(lag)

    wins = [2]
    for win in wins :
        for lag,lag_col in zip(lags, lag_cols):
            df["rmean_{}_{}".format(lag,win)] = df[["id", lag_col]].groupby("id")[lag_col].transform(lambda x : x.rolling(win).mean())
    return df

def per_timeframe_stats(df, col):
    ##Beregner gjennomsnitt og annen deskriptiv statistikk for hver måned og uke i datasettet
    months = df['month'].unique().tolist()
    for y in months:
        df.loc[df['month'] == y, col+'_month_mean'] = df.loc[df['month'] == y].groupby(['id'])[col].transform(lambda x: x.mean()).astype("float32")
        df.loc[df['month'] == y, col+'_month_max'] = df.loc[df['month'] == y].groupby(['id'])[col].transform(lambda x: x.max()).astype("float32")
        df.loc[df['month'] == y, col+'_month_min'] = df.loc[df['month'] == y].groupby(['id'])[col].transform(lambda x: x.min()).astype("float32")
        df[col + 'month_max_to_min_diff'] = (df[col + '_month_max'] - df[col + '_month_min']).astype("float32")

    dayofweek = df['week'].unique().tolist()
    for y in dayofweek:
        df.loc[df['week'] == y, col+'_week_mean'] = df.loc[df['week'] == y].groupby(['id'])[col].transform(lambda x: x.mean()).astype("float32")
        df.loc[df['week'] == y, col+'_week_median'] = df.loc[df['week'] == y].groupby(['id'])[col].transform(lambda x: x.median()).astype("float32"
        df.loc[df['week'] == y, col+'_week_max'] = df.loc[df['week'] == y].groupby(['id'])[col].transform(lambda x: x.max()).astype("float32")
    return df

def feat_eng(df):
    df = lags_windows(df)
    df = per_timeframe_stats(df,'Sales_Quantity')
    return df
```

Figur 19: Kode for egenskaper i maskinlæringsmodeller.

Etter egenskapene er laget må de tre modellene tilpasses. Dette gjøres på forskjellige måter for hver modell.

3.5.3.1 Light Gradient Boosting

For Light Gradient Boosting kan man se de ulike parameterne i figur x. For modellen er det anbefalt å bruke en lavere learning_rate sammen med et høyt antall num_iterations. I denne modellen muliggjør vi bagging med parameteren bagging_freq og vi legger til L2-regulering med lambda_l2.

```
##Tilpass Light Gradient Boosting
t0 = time.time()
lgb_params = {
        "objective" : "poisson",
        "metric" :"rmse",
        "force_row_wise" : True,
        "learning_rate" : 0.075,
        "sub_row" : 0.75,
        "bagging_freq" : 1,
        "lambda_l2" : 0.1,
        'verbosity': 1,
        'num_iterations' : 1000,
        'num_leaves': 50,
        "min_data_in_leaf": 5,
}
np.random.seed(777)
fake_valid_inds = np.random.choice(X_train.index.values, 52, replace = False)
train_inds = np.setdiff1d(X_train.index.values, fake_valid_inds)
train_data = lgb.Dataset(X_train.loc[train_inds] , label = y_train.loc[train_inds], free_raw_data=False)
fake_valid_data = lgb.Dataset(X_train.loc[fake_valid_inds], label = y_train.loc[fake_valid_inds],free_raw_data=False)

m_lgb = lgb.train(lgb_params, train_data, valid_sets = [fake_valid_data], verbose_eval=0)
t_lgb = time.time()-t0
```

Figur 20: Parametere i LGB.

3.5.3.2 Random Forest

For Random Forest modellen bestemmer vi antall trær i skogen, den lengste veien mellom rotnoden og bladnoden, tilfeldigheten til estimatoren, og antall jobber som kjøres parallelt.

```
##Tilpass Random Forest
t0 = time.time()
m_rf = RandomForestRegressor(n_estimators=100,max_depth=5, random_state=26, n_jobs=-1).fit(X_train.loc[train_inds], y_train.loc[train_inds])
t_rf = time.time()-t0
```

Figur 21: Parametere i Random Forest

3.5.3.3 Lineær regresjon

Ingen parametere settes i denne modellen.

```
##Tilpass Lineær Regresjon
t0 = time.time()
m_linreg = LinearRegression().fit(X_train[linreg_train_cols].loc[train_inds], y_train.loc[train_inds])
t_linreg = time.time()-t0
```

Figur 22: Tilpasning av lineær regresjon

Det siste vi gjør før modellene analyseres er å lage et skyvevindu som flyttes en dag frem av gangen. Når dette er gjort er modellene klare til å testes.

```
fday = datetime(2021,11, 12)
max_lags = 5
for tdelta in range(0, 8):
    day = fday + timedelta(weeks=tdelta)
    tst = test[(test.date >= day - timedelta(weeks=max_lags)) & (test.date <= day)].copy()
    tst = feat_eng(tst)
    tst_lgb = tst.loc[tst.date == day , lgb_train_cols].copy()
    test.loc[test.date == day, "preds_LightGB"] = m_lgb.predict(tst_lgb)
    tst_rf = tst.loc[tst.date == day , lgb_train_cols].copy()
    tst_rf = tst_rf.fillna(0)
    test.loc[test.date == day, "preds_RandomForest"] = m_rf.predict(tst_rf)

    tst_linreg = tst.loc[tst.date == day , linreg_train_cols].copy()
    tst_linreg = tst_linreg.fillna(0)
    test.loc[test.date == day, "preds_LinearReg"] = m_linreg.predict(tst_linreg)

test_final = test.loc[test.date >= fday]
```

Figur 23: Oppretting av skyvevindu.

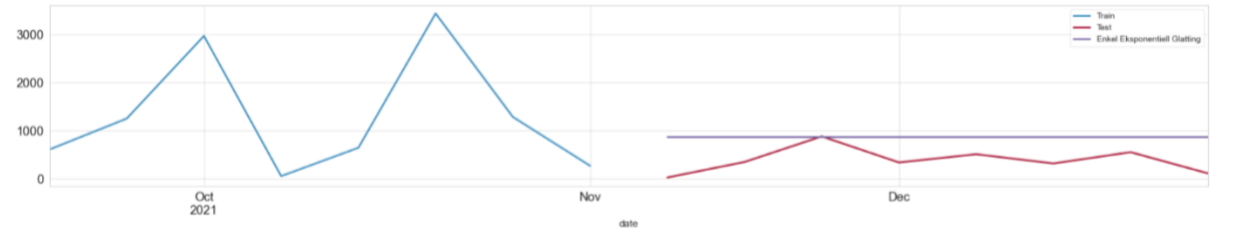
4.0 Resultater

4.1 Tradisjonelle modeller

Felles for de tradisjonelle modellene er at de har høye feilprognoser og gir dårlige predikasjoner. Basert på hva vi kan se når vi sammenligner verdien i testdataen opp mot den predikerte verdien ser vi at ingen av de tradisjonelle modellene klarer å fange opp variasjonen i antall solgte kvantiteter og produserer derfor en flat predikasjonslinje. Predikasjonslinjen er plassert litt ulikt i de forskjellige modellene, noe som gjør at enkelte modeller får en lavere RMSE. Dersom vi betrakter modellenes RMSE kan vi se at enkel eksponentiell glatting og ARIMA får de beste resultatene.

4.1.1 Enkel eksponentiell glatting

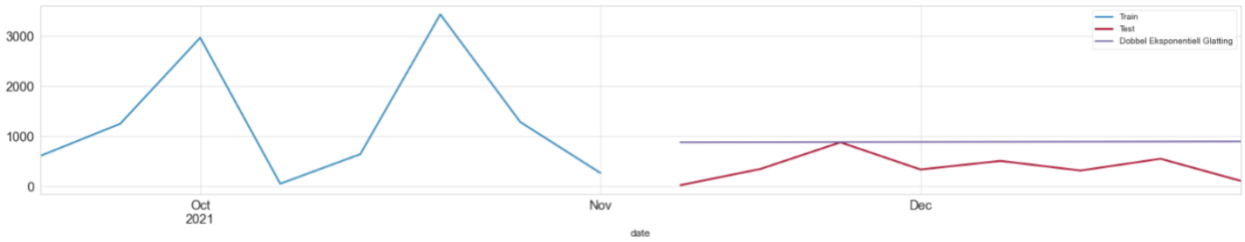
RMSE = 439.58



Figur 24: Resultat av enkel eksponentiell glatting.

4.1.2 Dobbel eksponentiell glatting

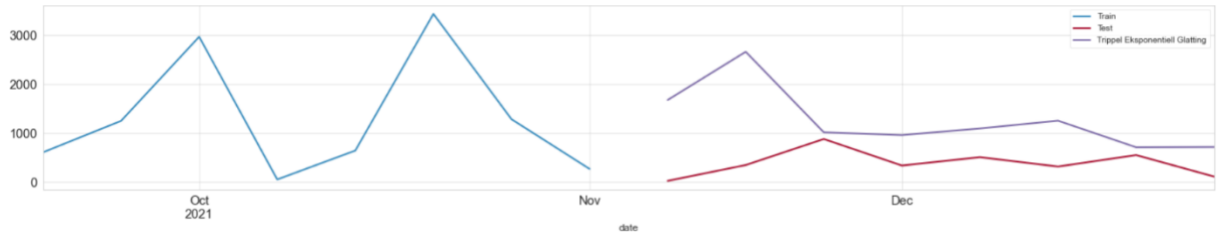
RMSE = 562.57



Figur 25: Resultat av dobbel eksponentiell glatting.

4.1.3 Trippel eksponentiell glatting

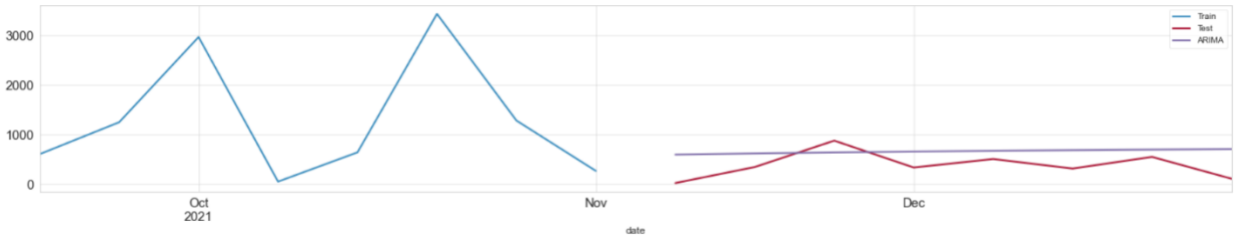
RMSE = 1123,93



Figur 26: : Resultat av trippel eksponentiell glatting.

4.1.4 ARIMA

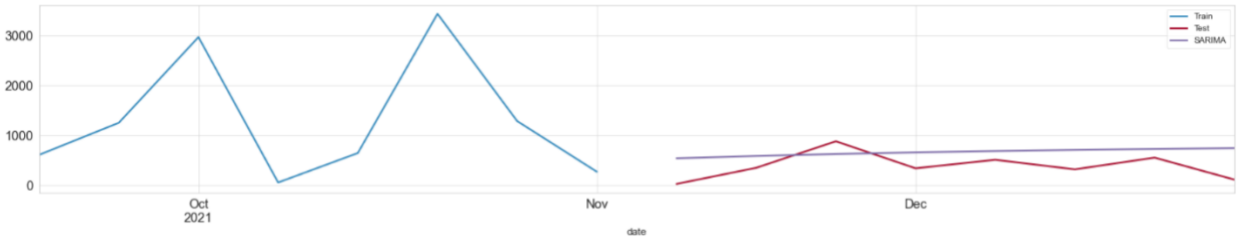
RMSE = 372.30



Figur 27: Resultat av ARIMA.

4.1.5 SARIMA

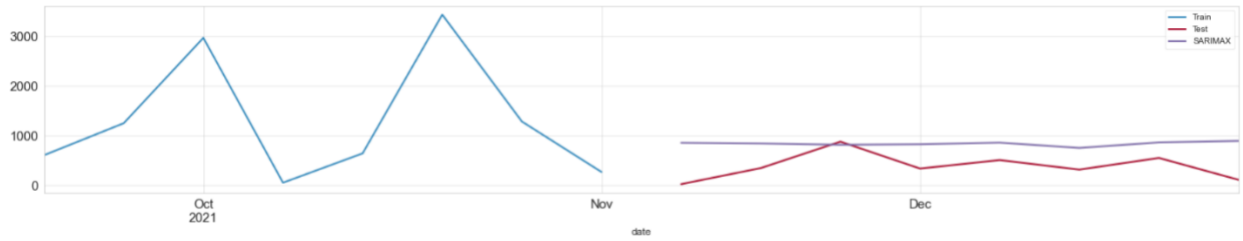
RMSE = 370.36



Figur 28: Resultat av SARIMA.

4.1.6 SARIMAX

RMSE = 525.15



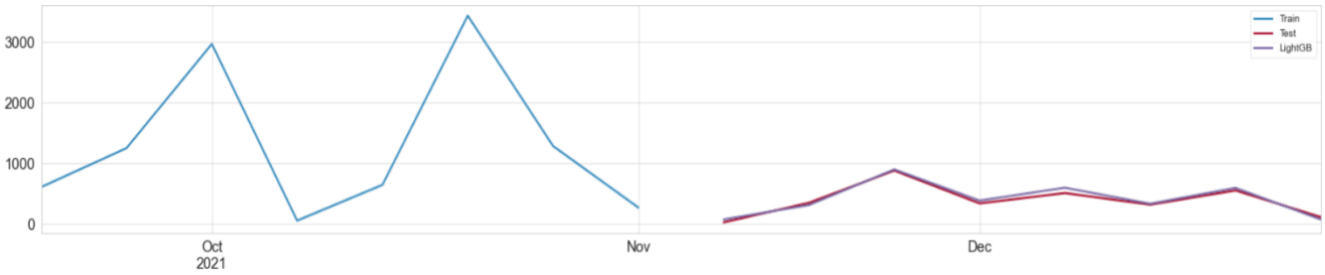
Figur 29: Resultat av SARIMAX.

4.2 Maskinlæringsmodeller

Om vi betrakter maskinlæringsmodellenes predikerte verdi sammenlignet med observert verdi i testdataen kan vi se at de alle ligger svært nærme. Det at den predikerte verdien i større grad klarer å følge den observerte verdien gjør også at alle modellene får relativt lave feilprognoser. Modellene er i visse tilfeller litt for optimistiske og predikerer et høyere salg enn hva som er tilfellet, men er jevnt over svært nøyaktige.

4.2.1 Light Gradient Boosting

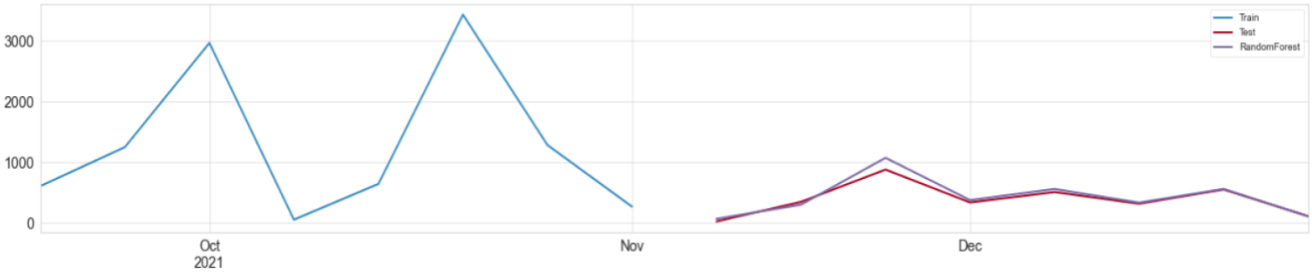
RMSE = 46.40



Figur 30: Resultat av LGB.

4.2.2 Random Forest

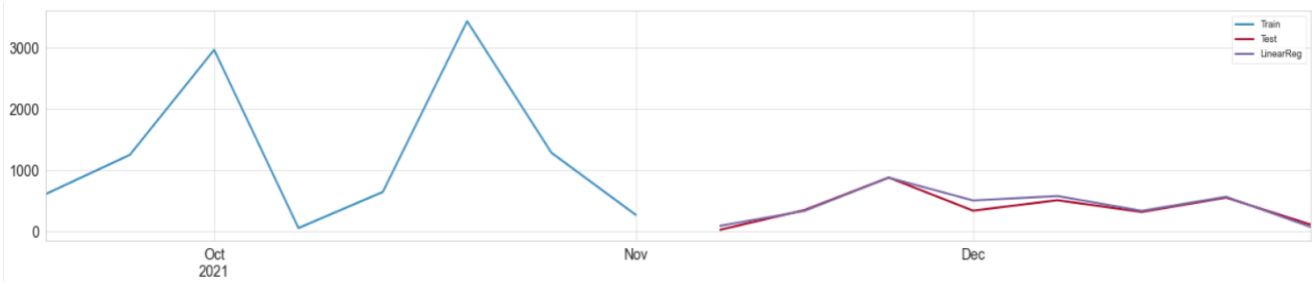
RMSE = 75.88



Figur 31: Resultat av Random Forest.

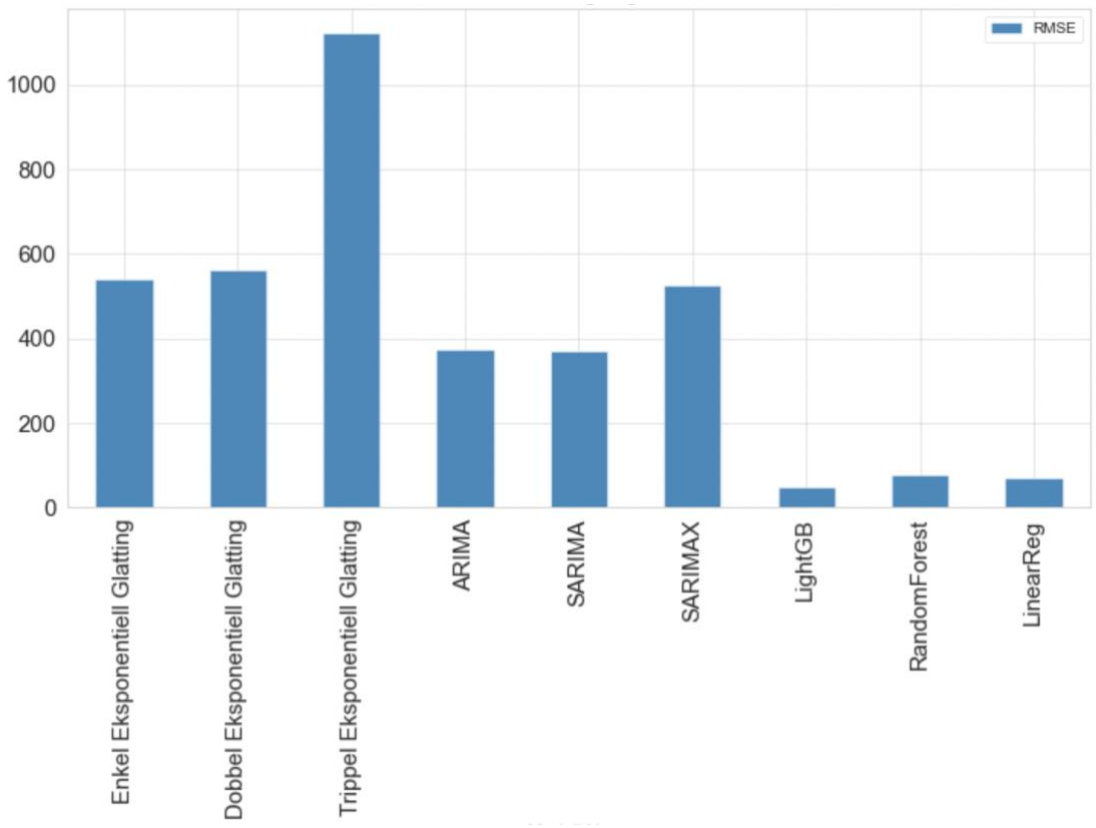
4.2.3 Lineær regresjon

RMSE = 69.81



Figur 32: Resultat av lineær regresjon.

4.3 Sammenligning av RMSE



Figur 33: Sammenligning av RMSE.

Når vi sammenligner modellene basert på RMSE kan vi se at det er svært ulike resultater for de tradisjonelle modellene og maskinlæringsmodellene. De tradisjonelle modellene har alle RMSE over 370 sammenlignet med maskinlæringsmodellene som alle har en RMSE under 76. Modellen som treffer best basert på lavest feilprognoser er Light Gradient Boosting, som har en RMSE på 46.40.

5.0 Diskusjon

Før vi kan konkludere på hvilken modell Sæther bør ta i bruk til videre prognosearbeid er vi nødt til å rette et kritisk blikk mot oppgavens resultater. Selv om modellenes RMSE score er en viktig faktor er det viktig å betrakte andre faktorer, oppgavens svakheter og begrensninger.

Resultatene fra RMSE og grafene viser at de tradisjonelle prognosemodellene treffer dårligere enn maskinlæringsmodellene. Av de tradisjonelle prognosemodellene er det ARIMA modellene som treffer best totalt sett. I ARIMA modellene er det flere parametere som skal defineres og utfallet av prognosene avhenger sterkt av de valgte verdiene. Parameteren m refererer til antall observasjoner i hver sesong, denne settes til 52 når det brukes data med ukefrekvens. Verdien på parameteren kan i stor grad påvirke utfallet av ARIMA modeller og vi kan derfor ikke utelukke at en annen verdi på m hadde gitt andre resultater.

ARIMA og SARIMA scorer relativt likt på RMSE. Det som skiller modellene er at SARIMA også adresserer sesongmessige mønstre gjennom parameterverdiene (P, D, Q) som identifiseres ved hjelp av auto_arima prosessen. Auto_arima identifiserer verdiene for (P, D, Q) som null og fanger derfor ikke opp sesongmessige mønstre i tidsserien. En SARIMA modell uten sesongmessige mønstre er det samme som en ARIMA modell og er årsaken til hvorfor modellene fikk tilsvarende likt resultat. Parameterne for SARIMA og ARIMA var derfor identiske.

Differensieringsparameteren (D) krever et sett av tester for stasjonaritet for å estimeres. Parameteren brukes vanligvis bare om dataen ikke er stasjonær. Det er enklere å predikere om dataen er stasjonær enn om den ikke er stasjonær. Med differensieringsparameteren er det mulig å tvinge ARIMA modellen til å selv justere for ikke-stasjonaritet, og det antas at ARIMA modeller som inkluderer denne parameteren blir stasjonære etter differensieringen. Ettersom auto_arima prosessen identifiserer differensieringsparameteren i ARIMA modellene med verdien én, kan det tyde på at tidsserien i utgangspunktet ikke var stasjonær. Dersom dette er tilfellet, kunne det vært en god ide å teste andre former for transformasjon av datamaterialet for å gjøre den stasjonær.

Av ARIMA modellene er det SARIMAX som scorer dårligst. I SARIMAX er variabelen Sales_Netto_Amount inkludert for å assistere modellen. I datamaterialet fra Sæther har vi kun tilgang på verdier fra tidsserien, og Sales_Netto_Amount er derfor eneste mulige variabel å inkludere. Resultatet av RMSE viser at inkluderingen av Sales_Netto_Amount gjør at modellen presterer dårligere. Det kan ikke utelukkes at modellen hadde truffet bedre dersom mer passende eksogene variabler hadde vært inkludert.

Modellene for eksponentiell glatting viser seg å være de som presterer dårligst av de tradisjonelle modellene. Resultatene fra enkel- og dobbel eksponentiell glatting viser seg å være relativt like, med en liten økning av RMSE i dobbel eksponentiell glatting. Det som skiller modellene er glattingsparameteren trend som i dette tilfellet fører til en økning av RMSE. Blant modellene i eksponentiell glatting er det trippel eksponentiell glatting som skiller seg mest ut og treffer dårligst med nesten dobbel så høy RMSE som enkel- og dobbel eksponentiell glatting. Inkluderingen av glattingsparameteren for sesongmessig mønster viser seg å redusere prestasjonen av modellen kraftig. Etter å ha testet ulike verdier for parameterne viser det seg at ulike verdier gir ulike resultater. Med en prøve og feile metode ser vi at en høyere verdi på glattingsparameteren nivå (α) resulterer i lavere RMSE. Det er allikevel besluttet å benytte verdien av seasonal_periods og span som 52, da vi ønsket å sammenligne modellene på likt grunnlag.

Dersom vi betrakter maskinlæringsmodellenes resultater basert på RMSE er det tydelig at disse modellene har en betydelig lavere score, sammenlignet med de tradisjonelle. Det kan derfor virke som at maskinlæringsmodellene er de som predikerer best i denne oppgaven. Resultatene bør allikevel betraktes med en viss skepsis, da maskinlæringsmodeller kan ha en tendens til å overtilpasse seg dataen når det brukes mindre datasett. Det er flere måter å takle dette problemet på. I Light Gradient Boosting har vi brukt L2 regularisering, som hjelper med å gjøre modellen mer konservativ. I modellen Random Forest har vi også valgt en mindre verdi for max_depth som er med på å begrense modellens evne til å se mønstre og ikke-eksisterende sammenhenger. For maskinlæringsmodellene lager vi også noen egenskaper basert på datagrunnlaget vi allerede har. Valg av egenskaper vil også kunne ha en innvirkning på resultatet.

Den tidligere forskningen vi har tatt for oss i denne oppgaven tilsier at ARIMA modeller gir lavere feilprognoser sammenlignet med andre tradisjonelle modeller slik som glidende

gjennomsnitt og eksponentiell glatting. I de studiene hvor de tradisjonelle og maskinlæringsmodellene sammenlignes er det maskinlæringsmodellene, eller en kombinert metode som gir de mest presise resultatene når man sammenligner modellenes feilprognoser. Det trekkes frem at maskinlæringsmodellene er hensiktsmessige å bruke når det kommer til enkelhet og umiddelbar bruk. Det vises også at en kombinasjon av kvantitative maskinlæringsprognoser justert med dømmekraft kan gjøre prognoser mer nøyaktige og treffsikre.

Oppgavens reliabilitet tar for seg hvorvidt vi får de samme resultatene dersom vi utfører målingen flere ganger. Den ukentlige salgsdataen varierer fra uke til uke, men vi kan allikevel anta at dersom vi utfører samme test om x antall måneder at vi vil få de samme resultatene med tanke på hvilken modell som får lavest RMSE.

Når det gjelder den indre validiteten må denne sies å være svakere i de tradisjonelle metodene. Disse modellene klarer ikke å bruke den uavhengige variabelen til å konstruere treffsikre prognoser når de testes mot testdataen. I maskinlæringsmodellene ser vi en høyere grad av indre validitet da de klarer å bruke den uavhengige variabelen til å lage mer treffsikre prognoser når de testes på testdataen.

Den ytre validiteten vil si om resultatene kan gjøres gjeldende for en større mengde data enn det studiet undersøker. Vi har ikke testet noen andre datasett som inneholder andre produktgrupper eller produkter fra en annen kategori. Her vil vi trekke frem hva tidligere forskning på lignende studier har resultert i. I de studiene vi har undersøkt kommer det frem at når man sammenligner tradisjonelle og maskinlæringsmodeller så er det maskinlæringsmodellene som gjør det best. Vi kan derfor trekke visse antagelser om det er en viss generaliserbarhet. Det er ikke sikkert at LGB alltid vil være modellen med lavest feilprognoser, men at det samlet sett vil være sannsynlig at maskinlæringsmodellene som gjør det best.

5.2 Praktiske implikasjoner og videre arbeid

Basert på RMSE resultatene er det Light Gradient Boosting som peker seg ut som en aktuell modell for videreutvikling. Modellen vi har analysert er relativt enkel, i form av at den kun baserer seg på historisk salgsdata, er aggregert på kategorinivå og inkluderer alle kunder. For

fremtidig arbeid kan det være interessant å utvide modellen med flere variabler, dele opp slik at man får en prognosemodell per kunde og teste med ulike nivåer.

Legge til variabler

Når det gjelder å forutsi fremtidig salg er det flere faktorer som påvirker, man kan derfor forsøke å gjøre prognosemodellen mer robust gjennom å inkludere flere relevante variabler. Variabler som kan inkluderes er for eksempel produktvekst, kategorivekst og salgsvekst per butikk. Man kan også inkludere spesifikke variabler som kan ha direkte påvirkning på en kategori, som for eksempel å inkludere værmeldingen når man skal predikere salget på solkrem.

Oppdeling etter kunde

Vår modell slår sammen salget til alle kundene, det kunne derfor vært hensiktsmessig å lage modeller delt inn etter de forskjellige kundene. Ved å dele inn etter kundene vil man få en bedre oversikt over hva som kan forventes av salg for hver enkelt kunde. Ettersom Sæther har ulike kunder i form av at noen kun er fysiske butikker, noen er rene E-com og andre er en blanding kunne det vært interessant å se på om det er store forskjeller på disse.

Ulike nivåer

Testingen i denne oppgaven er gjort for hele produktkategorien cleanser. Det kunne derfor vært interessant for videreutvikling å gjennomføre analyser både for de andre kategoriene, men også på de ulike nivåene. Med nivåer menes det å for eksempel gjøre prognoser helt ned på produktnivå eller høyere opp i hierarkiet og kanskje dele inn etter merker eller kategorier slik som hudpleie. Det er i dette tilfellet viktig å ta høyde for at prognosene generelt blir mer nøyaktige når de aggregeres på et høyere nivå.

Videre for å få en mer robust modell kan den kvantitative modellen kombineres med salgsavdelingens kunnskap og deres prognoser basert på dømmekraft. Ved å bruke modellen som en baseline og samtidig legge til den interne kunnskapen om når en kampanje kommer vil man være godt rustet til å treffe de større salgstoppene vi har sett tidligere i dataen. Det må også hensyntas at estimatene fra prognosene kommer i tide slik at Sæther kan reagere og bestille inn varer. Vi har i denne oppgaven brukt et mindre testvindu på grunn av færre observasjoner, for fremtidig arbeid kan det være ideelt å utvide modellens testvindu til å ta inkludere flere uker. Ved å gjøre dette kan man teste modellens evne til å predikere på mer enn åtte ukers testdata.

En praktisk implikasjon når det kommer til å implementere en Light Gradient Boosting modell er at det krever ressurser i form av ansatte med kompetanse innen maskinlæring. Til tross for at maskinlæring har eksistert en stund er det fortsatt ikke like utbredt i små og mellomstore bedrifter. For å kunne ta i bruk denne teknologien kreves det at man enten har personale som innehar kunnskapen som trengs, eller at man trener opp eller ansetter en ny person. Å gå videre med en maskinlæringsmodell kan derfor være kostbart, og selskapet må vurdere kostnaden opp mot nytten.

5.3 Oppgavens begrensninger og videre forskning

Denne studien har noen begrensninger som det er verdt å trekke frem. Den første begrensningen vi ønsker å ta for oss er oppgavens datasett. Datasettet vi benytter er svært enkelt og inneholder kun en variabel, antall solgte kvantiteter. Dette gjør modellen svakere da vi ikke har med noen andre forklaringsvariabler. Det er slik at det er mer enn det tidligere salget som påvirker det fremtidige. For videre forskning kunne det vært interessant å ta utgangspunkt i denne analysen, hvor man i tillegg til å teste de samme modellene med kun salgsdata, også utførte samme test på et datagrunnlag med flere uavhengige variabler. En annen begrensning ved denne oppgaven er at den tester flere modeller uten å gå i dybden på én enkelt modell. Det kan også betraktes som en svakhet ved oppgaven. For videre forskning kunne man tatt utgangspunkt i en av de tradisjonelle modellene, som i denne oppgaven ikke treffer så bra, og sett om man kunne fått den til å treffe bedre. Gjennom å fokusere på en av tradisjonelle modellene kunne man gjort et større forarbeid med datagrunnlaget, ved å transformere og forberede dataen. Dette kan blant annet gjøres gjennom følgende metoder:

- Kvadratrot eller N-te-rot transformasjon
- De-trending av tidsserie
- Differensiere tidsserien en eller flere ganger
- Log transformasjoner

6.0 Konklusjon

Problemstillingen vi forsøker å besvare i denne studien er følgende:

Hvilken prognosemodell bør Sæther ta i bruk til fremtidig prognosearbeid?

For å besvare problemstillingen er det tatt utgangspunkt i oppgavens tre formål:

«Definer de tradisjonelle modellene og maskinlæringsmodellene.»

For å kunne velge hvilke prognosemodeller som skulle inkluderes i studien var det nødvendig å starte med en gjennomgang av teori og tidligere forskning. Vi ønsket å teste datamaterialet uten å foreta store endringer slik at modellene ble testet på samme datagrunnlag. Forbehandlingen av datamaterialet bestod av å kontrollere for nullverdier og eventuelle feil. På bakgrunn av dette var det nødvendig å velge ut en rekke ulike modeller som varierer i kompleksitet for å kunne fange opp eventuelle komponenter. Det er inkludert modeller som adresserer ulike komponenter som trend og sesong, samt modeller som tar hensyn til om dataen ikke er stasjonær.

«Er det store ulikheter i resultatene fra de tradisjonelle modellene og maskinlæringsmodellene?»

Det viser seg å være store ulikheter i resultatene mellom de tradisjonelle modellene og maskinlæringsmodellene. Basert kun på oppgavens resultater målt etter RMSE utkonkurrerer de valgte maskinlæringsmodellene de tradisjonelle modellene.

«Hvilken modell gir de mest presise prognosene?»

Light Gradient Boosting gir de mest presise prognosene, basert på RMSE resultatene.

Gjennom arbeidet vi har utført i denne oppgaven har vi ikke kommet frem til en prognosemodell som kan tas i bruk slik den er nå. Basert på resultatene og oppgavens diskusjonsdel er det vanskelig å skulle komme med en veldig klar anbefaling for hvilken prognosemodell Sæther bør ta i bruk. Vi har testet en rekke modeller og basert på RMSE resultatene konkluderer vi med at Light Gradient Boosting vil kunne være en aktuell modell å gå videre med. Det må allikevel poengteres at hvorvidt denne modellen kan anbefales for videre arbeid avhenger av tilgjengelige ressurser og kunnskap i Sæther. Dersom modellen implementeres anbefaler vi at den utvides med flere variabler, eller brukes i kombinasjon med dømmekraft.

7.0 Referanser

Bell, E., Bryman, A., & Harley, B. (2018). *Business Research Methods* (5th ed.). Oxford University Press Academic UK. https://online.vitalsource.com/books/9780192545916

Bohanec, M., Robnik-Šikonja, M., & Kljajić Borštnar, M. (2017). Organizational Learning Supported by Machine Learning Models Coupled with General Explanation Methods: A Case of B2B Sales Forecasting. *Organizacija*, *50*(3), 217–233. https://doi.org/10.1515/orga-2017-0020

Brown, S., Blackmon, K., Cousins, P., & Maylor, H. (2013). *Operations Management: Policy, Practice and Performance Improvement*. Routledge.

Carlberg, C. (2018). *Predictive Analytics: Microsoft® Excel* (2nd ed.). Pearson Education, Inc.

Cecaj, A., Lippi, M., Mamei, M., & Zambonelli, F. (2020). Comparing Deep Learning and Statistical Methods in Forecasting Crowd Distribution from Aggregated Mobile Phone Data. *Applied Sciences*, *10*(18), 6580. https://doi.org/doi:10.3390/app10186580

Chopra, S., & Meindl, P. (2013). *Supply Chain Management Strategy, Planning, and Operation* (5th ed.). Pearson.

Components of Time Series. (n.d.). Toppr. https://www.toppr.com/guides/business-mathematics-and-statistics/time-series-analysis/components-of-time-series/?msclkid=6aca6551d11c11ec9252d0ce0df3edb0

Dahlum, S. (2021). Validitet. In *Store norske leksikon*. https://snl.no/validitet

Davenport, T. H. (2018). From analytics to artificial intelligence. *Journal of Business Analytics*, *1*(2), 73–80. https://doi.org/10.1080/2573234X.2018.1543535

Demir, L., & Akkaş, S. (2018). A comparison of sales forecasting methods for a feed company: A case study . *Pamukkale University Journal of Engineering Sciences*, *24*(4), 705–712.

Effrosynidis, D. (2020). *Forecasting Wars—Classical Forecasting Methods vs Machine Learning*. Data-Science-Portfolio. https://github.com/Deffro/Data-Science-Portfolio/blob/master/Notebooks/Forecasting%20Wars%20-%20Classical%20Forecasting%20Methods%20vs%20Machine%20Learning/Forecasting%20Wars%20-%20Classical%20Forecasting%20Methods%20vs%20Machine%20Learning.ipynb

Fisher, M. J., & Marshall, A. P. (2009). Understanding descriptive statistics. *Australian Critical Care*, *22*(2), 93–97. https://doi.org/10.1016/j.aucc.2008.11.003

Han, J., Kamber, M., & Pei, J. (2011). *Data Mining Concepts and Techniques* (3rd ed.). Morgan Kaufmann.

Hanke, J. E., & Wichern, D. (2014). *Business Forecasting* (9th ed.). Pearson Education Limited.

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and Practice* (3rd ed.). OText. https: //otexts.com/fpp3/

Johannessen, A., Christoffersen, L., & Tufte, P. A. (2020). *Forskningsmetode for økonomisk-administrative fag* (4th ed.). Abstrakt.

Lawrence, M., O`Connor, M., & Edmundson, B. (2000). A field study of sales forecasting accuracy and processes. *European Journal of Operational Research*, *122*(1), 151–160. https://doi.org/10.1016/S0377-2217(99)00085-5

Ma, S., Fildes, R., & Huang, T. (2016). Demand forecasting with high dimensional data: The case of SKU retail sales forecasting with intra-and inter-category promotional

information. *European Journal of Operational Research*, *294*(1), 245–257. https://doi.org/10.1016/j.ejor.2015.08.029

Mentzer, J. T., & Moon, M. A. (2005). *Sales Forecasting Management: A Demand Management Approach*. SAGE Publications, Inc. https://dx.doi.org/10.4135/9781452204444

Oliveira, J. M., & Ramos, P. (2019). Assessing the Performance of Hierarchical Forecasting Methods on the Retail Sector. *Entropy*, *21*(4), 436. https://doi.org/10.3390/e21040436

Rodrigo, J. A. (2022, March). *Forecasting time series with gradient boosting: Skforecast, XGBoost, LightGBM y CatBoost* [Attribution 4.0 International]. https://www.cienciadedatos.net/documentos/py39-forecasting-time-series-with-skforecast-xgboost-lightgbm-catboost.html

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and Statistical Modeling with Python. *Proceedings of the 9th Python*. Science Conference. https://www.statsmodels.org/stable/index.html?fbclid=IwAR0ovrYNE7C_c9OuCcjazqh7t5V6qKIOfzVYO96q6Rr684uh9ssnDUlVJzs

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding Machine Learning from Theory to Algorithms*. Cambridge University Press.

Sharma, S. K., & Sharma, V. (2012). Comparative Analysis of Machine Learning Techniques in Sale Forecasting. . . *International Journal of Computer Applications*, *53*(6), 51–54.

Udom, P., & Phumchusri, N. (2014). A comparison study between time series model and ARIMA model for sales forecasting of distributor in plastic industry. *IOSR Journal of Engineering*, *4*(2), 32–38. http://dx.doi.org/10.9790/3021-04213238

Vhatkar, S., & Dias, J. (2016). Oral-Care Goods Sales Forecasting Using Artificial Neural Network Model. *Procedia Computer Science*, *79*, 238–243. https://doi.org/10.1016/j.procs.2016.03.031

Wang, S., & Chaovalitwongse, W. A. (2011). *Evaluating and Comparing Forecasting Models*. https://www.researchgate.net/publication/313991989_Evaluating_and_Comparing_Forecasting_Models

Yagli, G. M., Yang, D., & Srinivasan, D. (2019). Reconciling solar forecasts: Sequential reconciliation. *Solar Energy*, *179*, 391–397. https://doi.org/10.1016/j.solener.2018.12.075

8.0 Vedlegg

Vedlegg 1: Script med ukefrekvens

```
##Nedlastning av pakker som skal brukes
import time
import warnings
import numpy as np
import pandas as pd
import seaborn as sns
import lightgbm as lgb
from itertools import cycle
from sklearn.svm import SVR
import statsmodels.api as sm
from pmdarima import auto_arima
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.tsa.holtwinters import SimpleExpSmoothing, ExponentialSmoothing
from statsmodels.tsa.seasonal import seasonal_decompose

%matplotlib inline
plt.style.use('bmh')
sns.set_style("whitegrid")
plt.rc('xtick', labelsize=15)
plt.rc('ytick', labelsize=15)
warnings.filterwarnings("ignore")
pd.set_option('max_colwidth', 100)
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
color_pal = plt.rcParams['axes.prop_cycle'].by_key()['color']
color_cycle = cycle(plt.rcParams['axes.prop_cycle'].by_key()['color'])
```

```
##Last inn data
data = pd.read_csv(r"C:/python/data_ukefrekvens.csv")

##Konverter kolonnen "date" til datetime format
data['date'] = pd.to_datetime(data['date'])

##Print fem første rader fra data
data.head()
```

```
##Spesifiser hva som er treningsdata (all data før inkludert 2021-11-05)
train = data[data['date'] <= '2021-11-05']

##Spesifiser hva som er testdata (all data etter 2021-11-05 til 2021-12-31)
test = data[(data['date'] > '2021-11-05') & (data['date'] <= '2021-12-31')]
```

```
##Plot trenings- og testdata
fig, ax = plt.subplots(figsize=(25,5))
train.plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
```

```
##Lager datarammer for både prediksjoner og statistikk, for å kunne lagre resultatene
##fra de ulike modellene
prediksjoner = pd.DataFrame()
prediksjoner['date'] = test['date']
statistikk = pd.DataFrame(columns=['Modell Navn','Utførelsestid','RMSE'])
```

```
##plott for å forstå tidsserien
ax = data.plot(x='date', y='Sales_Quantity', figsize=(12,6))
```

```
##Legg på en "skillevegg" i januar
ax = data.plot(x='date', y='Sales_Quantity', figsize=(12,6))
xcoords = ['2020-01-03', '2021-01-01']

for xc in xcoords:
    plt.axvline(x=xc, color='black', linestyle='--')
```

```
##dekomenser tidsserien i ulike komponenter
from statsmodels.tsa.seasonal import seasonal_decompose

data.set_index('date', inplace=True)

analysis = data[['Sales_Quantity']].copy()

decompose_result_mult = seasonal_decompose(analysis, model='multiplicative')

trend = decompose_result_mult.trend
seasonal = decompose_result_mult.seasonal
residual = decompose_result_mult.resid

decompose_result_mult.plot();
```

```
##Lager plot for autokorrelasjon
fig, ax = plt.subplots(figsize=(15, 3))
plot_acf(data['Sales_Quantity'].tolist(), lags=30, ax=ax);
```

```
t0 = time.time()
model_name='Enkel Eksponentiell Glatting'
span = 52
alpha = 2/(span+1)

##Tren
simpleExpSmooth_model = SimpleExpSmoothing(train['Sales_Quantity']).fit(smoothing_level=alpha,optimized=False)
t1 = time.time()-t0

##Prediker åtte uker
prediksjoner[model_name] = simpleExpSmooth_model.forecast(8).values

##Graf som plotter åtte uker av treningsdelen med åtte uker av testdelen
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t1, 'RMSE':score},ignore_index=True)
```

```
t0 = time.time()
model_name='Dobbel Eksponentiell Glatting'

##tren
doubleExpSmooth_model = ExponentialSmoothing(train['Sales_Quantity'],trend='add',seasonal_periods=52).fit()
t1 = time.time()-t0

##Prediker åtte uker
prediksjoner[model_name] = doubleExpSmooth_model.forecast(8).values

##Graf som plotter åtte uker av treningsdelen med åtte uker av testdelen
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t1, 'RMSE':score},ignore_index=True)
```

```
t0 = time.time()
model_name='Trippel Eksponentiell Glatting'

##Tren
tripleExpSmooth_model = ExponentialSmoothing(train['Sales_Quantity'],trend='add',seasonal='add',seasonal_periods=2).fit()
t1 = time.time()-t0

##Prediker åtte uker
prediksjoner[model_name] = tripleExpSmooth_model.forecast(8).values

##Graf som plotter åtte uker av treningsdelen med åtte uker av testdelen
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i dataframmene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t1, 'RMSE':score},ignore_index=True)
```

```
t0 = time.time()
model_name='ARIMA'
arima_model = auto_arima(train['Sales_Quantity'], start_p=0, start_q=0,
                         max_p=14, max_q=3,
                         seasonal=False,
                         d=None, trace=True,random_state=2020,
                         error_action='ignore',
                         suppress_warnings=True,
                         stepwise=True)
arima_model.summary()
```

```
##Tren
arima_model.fit(train['Sales_Quantity'])
t1 = time.time()-t0

##Prediker åtte uker
prediksjoner[model_name] = arima_model.predict(n_periods=8)

##Graf som plotter åtte uker av treningsdelen med åtte uker av testdelen
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i dataframmene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t1, 'RMSE':score},ignore_index=True)
```

```
t0 = time.time()
model_name='SARIMA'
sarima_model = auto_arima(train['Sales_Quantity'], start_p=0, start_q=0,
                          max_p=14, max_q=3,
                          seasonal=True, m=12,
                          d=None, trace=True,random_state=2020,
                          out_of_sample_size=8,
                          error_action='ignore',
                          suppress_warnings=True,
                          stepwise=True)
sarima_model.summary()
```

```
##Tren
sarima_model.fit(train['Sales_Quantity'])
t1 = time.time()-t0

##Prediker åtte uker
prediksjoner[model_name] = sarima_model.predict(n_periods=8)

##Graf som plotter åtte uker av treningsdelen med åtte uker av testdelen
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t1, 'RMSE':score},ignore_index=True)
```

```
t0 = time.time()
model_name='SARIMAX'
sarimax_model = auto_arima(train['Sales_Quantity'], start_p=0, start_q=0,
                           max_p=14, max_q=3,
                           seasonal=True, m=52,
                           exogenous = train[['Sales_Netto_Amount']].values,
                           d=None, trace=True,random_state=2020,
                           out_of_sample_size=8,
                           error_action='ignore',
                           suppress_warnings=True,
                           stepwise=True)
sarimax_model.summary()
```

```
##Trem
sarimax_model.fit(train['Sales_Quantity'])
t1 = time.time()-t0

##Prediker åtte uker
prediksjoner[model_name] = sarimax_model.predict(n_periods=8)

##Graf som plotter åtte uker av treningsdelen med åtte uker av testdelen
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t1, 'RMSE':score},ignore_index=True)
```

```
def lags_windows(df):
    lags = [2]
    lag_cols = ["lag_{}".format(lag) for lag in lags ]
    for lag, lag_col in zip(lags, lag_cols):
        df[lag_col] = df[["id","Sales_Quantity"]].groupby("id")["Sales_Quantity"].shift(lag)

    wins = [2]
    for win in wins :
        for lag,lag_col in zip(lags, lag_cols):
            df["rmean_{}_{}".format(lag,win)] = df[["id", lag_col]].groupby("id")[lag_col].transform(lambda x : x.rolling(win).mean())
    return df

def per_timeframe_stats(df, col):
    ##Beregner gjennomsnitt og annen deskriptiv statistikk for hver måned og uke i datasettet
    months = df['month'].unique().tolist()
    for y in months:
        df.loc[df['month'] == y, col+'_month_mean'] = df.loc[df['month'] == y].groupby(['id'])[col].transform(lambda x: x.mean()).astype("float32")
        df.loc[df['month'] == y, col+'_month_max'] = df.loc[df['month'] == y].groupby(['id'])[col].transform(lambda x: x.max()).astype("float32")
        df.loc[df['month'] == y, col+'_month_min'] = df.loc[df['month'] == y].groupby(['id'])[col].transform(lambda x: x.min()).astype("float32")
        df[col + 'month_max_to_min_diff'] = (df[col + '_month_max'] - df[col + '_month_min']).astype("float32")

    dayofweek = df['week'].unique().tolist()
    for y in dayofweek:
        df.loc[df['week'] == y, col+'_week_mean'] = df.loc[df['week'] == y].groupby(['id'])[col].transform(lambda x: x.mean()).astype("float32")
        df.loc[df['week'] == y, col+'_week_median'] = df.loc[df['week'] == y].groupby(['id'])[col].transform(lambda x: x.median()).astype("float32")
        df.loc[df['week'] == y, col+'_week_max'] = df.loc[df['week'] == y].groupby(['id'])[col].transform(lambda x: x.max()).astype("float32")
    return df

def feat_eng(df):
    df = lags_windows(df)
    df = per_timeframe_stats(df,'Sales_Quantity')
    return df
```

```
##Laster inn data
data = pd.read_csv(r"C:/python/data_ukefrekvens.csv")

##Setter "date" til datetime format
data['date'] = pd.to_datetime(data['date'])

##Spesifiserer hva som er test- og treningsdata
train = data[data['date'] <= '2021-11-05']
test = data[(data['date'] > '2021-11-05' )& (data['date'] <= '2021-12-31')]

##Lager dataramme med alle nye variabler
data_ml = feat_eng(train)
data_ml = data_ml.dropna()

useless_cols = ['id','Sales_Quantity','date','demand_month_min']
linreg_train_cols = ['Sales_Netto_Amount','year','month','week','lag_2','rmean_2_2']
lgb_train_cols = data_ml.columns[~data_ml.columns.isin(useless_cols)]
X_train = data_ml[lgb_train_cols].copy()
y_train = data_ml["Sales_Quantity"]
```

```
##Tilpass Light Gradient Boosting
t0 = time.time()
lgb_params = {
        "objective" : "poisson",
        "metric" :"rmse",
        "force_row_wise" : True,
        "learning_rate" : 0.075,
        "sub_row" : 0.75,
        "bagging_freq" : 1,
        "lambda_l2" : 0.1,
        'verbosity': 1,
        'num_iterations' : 1000,
        'num_leaves': 50,
        "min_data_in_leaf": 5,
}
np.random.seed(777)
fake_valid_inds = np.random.choice(X_train.index.values, 52, replace = False)
train_inds = np.setdiff1d(X_train.index.values, fake_valid_inds)
train_data = lgb.Dataset(X_train.loc[train_inds] , label = y_train.loc[train_inds], free_raw_data=False)
fake_valid_data = lgb.Dataset(X_train.loc[fake_valid_inds], label = y_train.loc[fake_valid_inds],free_raw_data=False)

m_lgb = lgb.train(lgb_params, train_data, valid_sets = [fake_valid_data], verbose_eval=0)
t_lgb = time.time()-t0

##Tilpass Lineær Regresjon
t0 = time.time()
m_linreg = LinearRegression().fit(X_train[linreg_train_cols].loc[train_inds], y_train.loc[train_inds])
t_linreg = time.time()-t0

##Tilpass Random Forest
t0 = time.time()
m_rf = RandomForestRegressor(n_estimators=100,max_depth=5, random_state=26, n_jobs=-1).fit(X_train.loc[train_inds], y_train.loc[train_inds])
t_rf = time.time()-t0
```

```
fday = datetime(2021,11, 12)
max_lags = 15
for tdelta in range(0, 8):
    day = fday + timedelta(weeks=tdelta)
    tst = test[(test.date >= day - timedelta(weeks=max_lags)) & (test.date <= day)].copy()
    tst = feat_eng(tst)
    tst_lgb = tst.loc[tst.date == day , lgb_train_cols].copy()
    test.loc[test.date == day, "preds_LightGB"] = m_lgb.predict(tst_lgb)
    tst_rf = tst.loc[tst.date == day , lgb_train_cols].copy()
    tst_rf = tst_rf.fillna(0)
    test.loc[test.date == day, "preds_RandomForest"] = m_rf.predict(tst_rf)

    tst_linreg = tst.loc[tst.date == day , linreg_train_cols].copy()
    tst_linreg = tst_linreg.fillna(0)
    test.loc[test.date == day, "preds_LinearReg"] = m_linreg.predict(tst_linreg)

test_final = test.loc[test.date >= fday]
```

```
model_name='LightGB'
prediksjoner[model_name] = test_final["preds_"+model_name]

##Graf
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test_final.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test_final['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t_lgb, 'RMSE':score},ignore_index=True)
```

```
model_name='RandomForest'
prediksjoner[model_name] = test_final["preds_"+model_name]

##Graf
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test_final.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test_final['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t_lgb, 'RMSE':score},ignore_index=True)
```

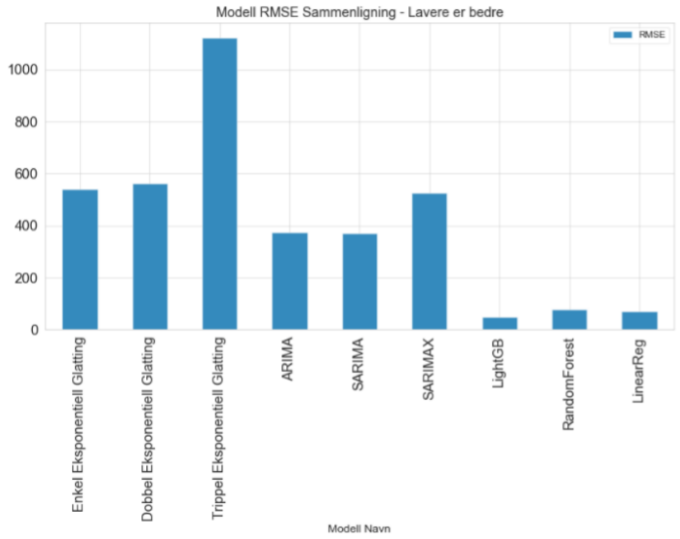
```
model_name='LinearReg'
prediksjoner[model_name] = test_final["preds_"+model_name]

##Graf
fig, ax = plt.subplots(figsize=(25,4))
train[-8:].plot(x='date',y='Sales_Quantity',label='Train',ax=ax)
test_final.plot(x='date',y='Sales_Quantity',label='Test',ax=ax);
prediksjoner.plot(x='date',y=model_name,label=model_name,ax=ax);

##Evaluer og legg inn i datarammene for resultater
score = np.sqrt(mean_squared_error(prediksjoner[model_name].values, test_final['Sales_Quantity']))
print('RMSE for {}: {:.4f}'.format(model_name,score))

statistikk = statistikk.append({'Modell Navn':model_name, 'Utførelsestid':t_linreg, 'RMSE':score},ignore_index=True)
```

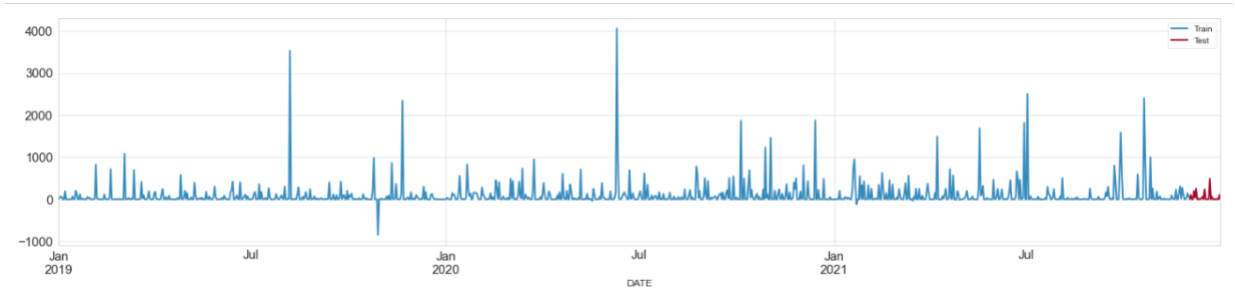
```
statistikk.plot(kind='bar',x='Modell Navn', y='RMSE', figsize=(12,6), title="Modell RMSE Sammenligning - Lavere er bedre");
```

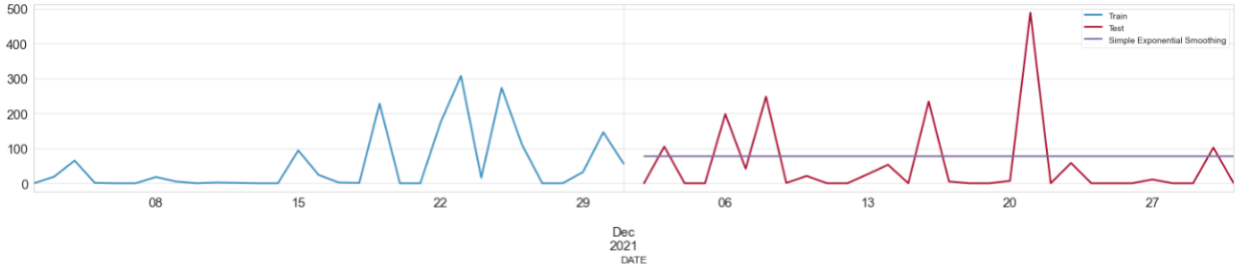


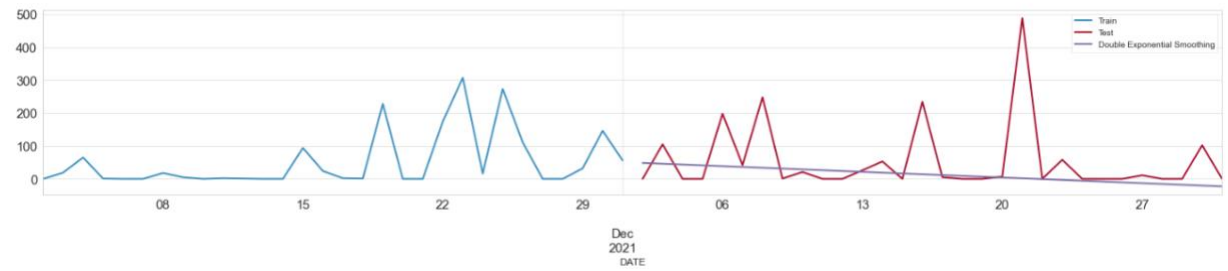
Vedlegg 2: Output dagfrekvens

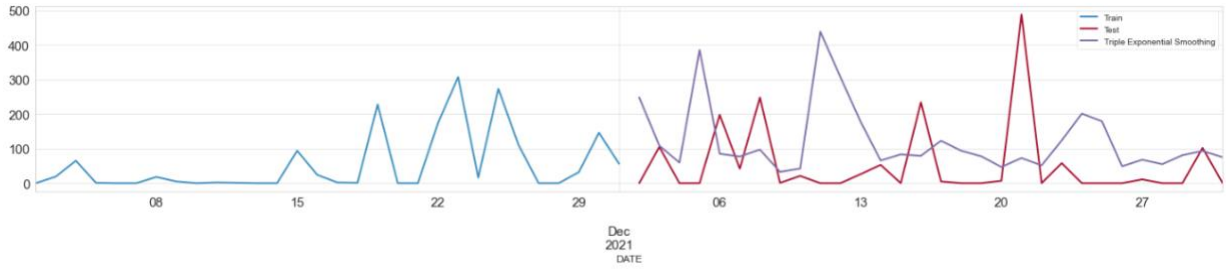
	ID	CATCODETXT	DATE	Sales Quantity	Sales Netto Amount	Price
0	1	Cleanser	2019-01-01	0	0	0
1	2	Cleanser	2019-01-02	46	4322	94
2	3	Cleanser	2019-01-03	62	6621	107
3	4	Cleanser	2019-01-04	50	4623	92
4	5	Cleanser	2019-01-05	0	0	0
...	...	...	...	...	...	...
1091	1092	Cleanser	2021-12-27	11	1182	107
1092	1093	Cleanser	2021-12-28	0	0	0
1093	1094	Cleanser	2021-12-29	0	0	0
1094	1095	Cleanser	2021-12-30	102	11717	115
1095	1096	Cleanser	2021-12-31	0	0	0

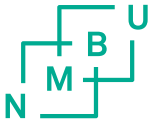
1096 rows × 6 columns








500
400
300
200
100
0
08
15
22
29
06
13
20
27
Dec
2021
DATE
Train
Test
Triple Exponential Smoothing



Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway