Høgskolen
i Innlandet

Kongsvinger

Kais Bashir Banki

Katrine Bakke Haga

Masteroppgave

En sammenlignende analyse av tradisjonelle og moderne maskinlæringsmodeller for prediksjon av rentenivåer

A comparative analysis of traditional and modern machine learning models for interest rate prediction

Master i økonomi og ledelse

Profil digital ledelse og business analytics

KDBA 950

2024

Forord

Vi ønsker å takke vår veileder Nurilla Avazov for uvurderlig hjelp gjennom hele oppgaveskrivingen. Han har vært tilgjengelig på kort varsel og svart opp våre spørsmål samt gitt inspirasjon og retning underveis i vår oppgave. En stor takk rettes også til vår bi-veileder professor Sjur Westgaard som har veiledet oss og bidratt med inspirasjon gjennom hele perioden. Vi har opplevd denne prosessen som utfordrende, men svært lærerik. Vi har gått langt ut over vår komfortsone når det gjelder maskinlæringsmodeller noe som har tvunget oss til å lære inngående teori om de ulike modellene samt metoder for å kode modellene i praksis. Vi har i vår oppgave benyttet oss av ekspertisen til ChatGPT for støtte til kodeutfordringer og inspirasjon. Sist men ikke minst ønsker vi å takke våre familier for stor tålmodighet gjennom hele perioden.

Sammendrag

De siste tiårene har det blitt utført omfattende forskning på rentepredikSjon og prediksjonsmodeller. Amerikanske statsobligasjoner har også fått mye oppmerksomhet fra forskere. Dette kan skyldes rentens nøkkelrolle som økonomisk indikator og deres innvirkning på både privat og offentlig sektor. I tillegg har amerikanske statsobligasjoner en sentral rolle i det globale finansmarkedet. Dette understreker behovet for mer nøyaktige prediksjonsmodeller.

Hovedformålet med vår forskning er å sammenligne prestasjonene til AutoRegressive Integrated Moving Average (ARIMA), Ridge regresjon, Random Forest og Long-Short Term Memory (LSTM). Vi har undersøkt amerikanske statsobligasjoner for perioden 1986 – 2022. Vår problemstilling er:

«En sammenlignende analyse av tradisjonelle og moderne maskinlæringsmodeller for prediksjon av rentenivåer»

Med kvantitativ metode har vi forsøkt å svare opp forskningsspørsmålene «Forskjellene mellom tradisjonelle og moderne tilnærminger for anvendt rentepredikSjon av amerikanske statsobligasjoner» og «Hvordan nye metoder kan transformere økonomiske analyser».

Resultatene vil kunne gi økt kunnskap om hvilke modeller som best støtter økonomiske beslutninger og bidra til en dypere forståelse av samspillet mellom pengepolitikk og rentemarkeder ved å identifisere hvilke modeller som gir mest mulig nøyaktig resultat. Våre hovedfunn viser at ARIMA presterer dårlig på kort og mellomlang rente. Ridge regresjon tilpasser seg in- og out-of-sample grafene tettest og LSTM presterer godt.

Abstract

In recent decades, extensive research has been conducted on interest rate prediction and prediction models. US Treasury bonds have also received a lot of attention from researchers. This may be due to the key role of the interest rate as an economic indicator and their impact on both the private and public sectors. In addition, US Treasuries play a key role in global financial markets. This underscores the need for more accurate prediction models.

The main objective of our research is to compare the performance of AutoRegressive Integrated Moving Average (ARIMA), Ridge Regression, Random Forest and Long-Short Term Memory (LSTM). We have examined US Treasury bonds for the period 1986 – 2022. Our research question is:

> "A comparative analysis of traditional and modern machine learning models for interest rate prediction"

Using quantitative methodology, we have attempted to answer the research questions "The differences between traditional and modern approaches to applied interest rate prediction of US Treasuries" and "How new methods can transform economic analysis".

The results will provide greater knowledge about which models best support economic decisions and contribute to a deeper understanding of the interplay between monetary policy and fixed income markets by identifying which models provide the most accurate results. Our main findings show that ARIMA performs poorly on short and medium rates. Ridge regression adapts to the in- and out-of-sample graphs most closely, and LSTM performs well.

Innholdsfortegnelse

Figurliste

Tabelliste

1. Introduksjon

1.1. Faglig bakgrunn og motivasjon for studien

Tradisjonelle rentepredikjonsmodeller har lenge vært i bruk for å forutse renteutviklingen. Den mest kjente modellen, Autoregressive Integrated Moving Average (ARIMA) har vist seg å være spesielt nøyaktig i sine prediksjoner (Siami-Namini et al., 2018). Ridge regresjonsmodell trekkes frem som en effektiv modell for prediksjon i markeder med hyppige endringer, da modellen vil bidra til stabilisering selv om økonomien endres raskt. Regresjonsmodellen er ikke kun begrenset til å predikere rentestruktur, men kan også anvendes i større økonometriske modeller (Watson & White, 1976).

Selv om lineære statistiske metoder som ARIMA og Ridge regresjon har vært svært populære, ble det mot slutten av 1970- og 1980-tallet tydelig at lineære modeller ofte ikke er tilpasset mange virkelige anvendelser (Bontempi et al., 2013). Disse modellene analyserer historisk data og predikerer basert på mønstre og trender. Modellene begrenses av antagelsen om linearitet i dataene og kan ha problemer med å tilpasse seg reelle tidsserier (Khandelwal et al., 2015).

Som et resultat av utviklingen av avanserte maskinlæringsalgoritmer har det blitt oppdaget nye algoritmer som er utviklet for å analysere og predikere tidsseriedata (Siami-Namini et al., 2018). De siste to tiårene har maskinlæringsmodeller fått mye oppmerksomhet, og vist seg som reelle konkurrenter for tradisjonelle metoder (Bontempi et al., 2013). Nevrale nettverk, maskinlæring og kunstig intelligens skal gi forbedret nøyaktighet i rentepredikjoner ved å identifisere komplekse mønstre og sammenhenger som de tradisjonelle metodene ofte overser. Disse modellene evner å inkludere og analysere store mengder ulike datakilder og ved å analysere alle faktorer gis et mer nyansert bilde og en omfattende forståelse av markedsforholdene som kan forbedre nøyaktigheten til prediksjonene (Shu & Chou, 2021).

Ved å se på forskjellene mellom disse to tilnærmingene ønsker vi å gi innsikt i modellenes relative effektivitet og nøyaktighet. Analysen skal besvare forskningsspørsmålene «Forskjellene mellom tradisjonelle og moderne tilnærminger for anvendt rentepredikjon av amerikanske statsobligasjoner» og «hvordan nye metoder kan transformere økonomiske analyser».

Amerikanske statsobligasjoner blir ansett som verdens fremste risikofrie aktiva (Tabova & Warnock, 2021:Hager, 2017) og er en av de viktigste finansielle markedene globalt (Andersen & Benzoni, 2010).

På grunn av deres betydelige rolle i globaløkonomien blir det ansett som svært viktig for blant annet investorer og analytikere å predikere disse rentene mest mulig nøyaktig (Shu & Chou, 2021). Disse obligasjonene er benchmark for prising av en rekke lån både i USA og globalt og er benyttet av den amerikanske sentralbanken (en av de viktigste sentralbankene i verden) til kvantitativ lettelse (Tabova & Warnock, 2021).

De siste tiårene har det vært store fremskritt i prediktiv analyse, spesielt innen maskinlæring. Prediksjon av fremtidige utfall er kritisk i flere domener (Putka et al., 2018) og amerikanske statsobligasjoner er ingen unntak. Ved å sammenligne tradisjonelle og moderne modeller ønsker vi å utforske og kvantifisere styrkene og svakhetene til de ulike modellene. Vi har som mål å bidra til økt forståelse av hvordan avanserte moderne maskinlæringsteknikker kan forbedre prediksjonsnøyaktigheten i rentemarkedet. Dette arbeidet er ikke bare akademisk relevant, men henvender seg også blant annet til investorer og finansinstitusjoner som forsøker å navigere et økonomisk landskap med stadig økende uforutsigbarhet.

1.2. Problemstilling og forskningsspørsmål

Vi har forsøkt å svare opp forskningsspørsmålene «Forskjellene mellom tradisjonelle og moderne tilnærminger for anvendt rentepredikskjon av amerikanske statsobligasjoner» og «Hvordan nye metoder kan transformere økonomiske analyser» for å belyse vår problemstilling:

«En sammenlignende analyse av tradisjonelle og moderne maskinlæringsmodeller for prediksjon av rentenivåer»

Ved å analysere historisk rentedata for amerikanske statsobligasjoner vil vi forsøke å svare opp problemstillingen og å gi en anbefaling på hvilken modell som kan prestere godt på historiske rentedata. Vi vil benytte maskinlæring for å utvikle prediksjonsmodellene som gjøres sammenlignbare etter beste evne ved å sørge for lik input og mest mulig lik oppbygning. Vi vil benytte autoseleksjon av hyperparametere der det er mulig og manuell selektering der det er nødvendig. Våre forventninger før analysen er at ARIMA-modellen kommer til å gi et godt resultat da denne modellen har vært svært populær over lang tid . I tillegg antar vi at Long-Short Term Memory (LSTM) gir det beste resultatet da denne modellen er mer kompleks (Zhang, 2003). Når det gjelder Random Forest og Ridge regresjon er vi spent på resultatet da vi ikke kjenner til tidligere sammenligning av disse modellene, sett opp mot ARIMA og LSTM. Det er heller ikke oss bekjent at disse modellene benyttes i stor grad til rentepredikskjon.

1.3. Bidrag – praktiske og teoretiske implikasjoner

Vår forskning adresserer en fundamental utfordring innen økonomisk prediksjon, altså å predikere renten på amerikanske statsobligasjoner. Tradisjonelle maskinlæringsmodeller har lenge vært populær grunnet statistiske egenskaper og relative fleksibilitet. Imidlertid har fremveksten av moderne maskinlæringsteknikker økt i popularitet på grunn av deres fleksibilitet og er drevet fram gjennom behovet for å forutsi komplekse ikke-lineære mønstre som tradisjonelle modeller som ARIMA ikke håndterer godt (Zhang, 2003). Gjennom en systematisk og empirisk evaluering av ARIMA, Ridge regresjon, Random Forest og LSTM bidrar vår oppgave til forskningslitteraturen ved å tilby innsikt i hvilke modeller som best fanger opp dynamikken til rentene på amerikanske statsobligasjoner. Dette er spesielt viktig gitt den sentrale rollen rentene spiller i globaløkonomien da de påvirker alt fra statlig lånekapasitet til private investeringsbeslutninger.

Resultatene av vår forskning søker å gi beslutningstakere bedre verktøy for å vurdere fremtidige økonomiske forhold, noe som kan gi bedre grunnlag til politiske beslutninger og investeringsstrategier. Vår sammenlignende studie søker å bidra til maskinlæringens område ved å belyse relative fordeler og begrensninger av nye modeller sammenlignet med mer etablerte modeller.

1.4. Valg av forskningskontekst

I denne masteroppgaven benytter vi en kvantitativ forskningsmetodikk. Valget av kvantitativ metode er motivert av behovet for å benytte systematisk og standardisert tilnærming til dataanalyse for å identifisere mønstre for å redegjøre for våre forskningsspørsmål. Modellene som er valgt for sammenligning er ARIMA og Ridge regresjon som representerer tradisjonelle tidsseriemodeller. Disse blir sett opp mot de moderne maskinlæringsmodellene Random Forest og LSTM. Målet med forskningen er å ikke bare sammenligne prediktiv nøyaktighet av disse modellene, men også å utforske deres styrker og begrensninger i konteksten av det amerikanske statsobligasjonsmarkedet og bidra til mer informert beslutningstaking innen området.

1.5. Oversikt over hvordan oppgaven er bygget opp

Kapittel 2 tar for seg litteraturgjennomgang av tidligere forskning og teori for de ulike modellene. Kapittel 3 forklarer metodikk for dataanalyse. I kapittel 4 vil vi presenter resultatene og diskutere disse mot hverandre før vi konkluderer oppgaven og kapittel 5 presenterer implikasjoner, begrensninger og behov for fremtidig forskning.

2. Teori – Beskrivelse av teorigrunnlag

Dette kapittelet vil gi en oversikt over relevant teori innen temaene renter, obligasjoner, maskinlæring og tradisjonelle og moderne prediksjonsmodeller. Vi vil også gi et teoretisk grunnlag for tidsserier og dens egenskaper. Videre presenteres teoretisk rammeverk for modellene og kort grunnlag for validering av modeller.

2.1. Søkeprosess over relevant litteratur

I vår litteraturgjennomgang har vi gjort et omfattende søk innenfor renter, rentepredikjon, prediksjonsmodeller og amerikanske statsobligasjoner. Vi har fokusert på modellene vi har benyttet i vår sammenligning, altså ARIMA, Ridge regresjon, Random Forest og LSTM og har undersøkt etter beste evne mangel i eksisterende forskning. For å lete etter relevant litteratur har vi i hovedsak søkt i Google Scholar, Brage database for Høyskolen i Innlandet, Google books og international journal forecasting. Vi startet med å søke etter sammenlignende analyser på andre forskningsområder, før vi gikk videre med å søke opp bakgrunn på amerikanske statsobligasjoner for å få et inntrykk av hvilken forskning som eksisterte fra før. Vi har søkt etter forskning på alle modellene vi har brukt i vår forskning og analysert relevante artikler både for rentemarkedet, men også andre marked som kan være relevant. Vi har etter beste evne forsøkt å identifisere lignende studier, men har ikke funnet noen som sammenligner ARIMA, Ridge regresjon, Random Forest og LSTM for amerikanske statsobligasjoner. Vi ønsker derfor å utforske potensialet til moderne maskinlæringsmodeller for rentepredikjon.

Vår litteraturgjennomgang støtter under våre forskningsspørsmål ved å undersøke sammenligninger av prediksjonsmodeller både for rentemarkedet og andre marked. Litteraturen gir også innsikt i nye metoder ved å presentere eksisterende forskning på de moderne modellene Random Forest og LSTM. På denne måten forsøker vi å undersøke hvilke muligheter disse modellene gir for transformasjon av økonomiske analyser. Etter fullført litteraturgjennomgang sitter vi igjen med et inntrykk av at det finnes mangel på sammenligninger av tradisjonelle og moderne prediksjonsmodeller for rentemarkedet, spesifikt på amerikanske statsobligasjoner. ARIMA og LSTM har typisk blitt sammenlignet i flere marked, men vi ser at det vil være interessant å se disse opp mot Ridge regresjon og Random Forest.

2.2. Drøfting av relevant litteratur

2.2.1. Renter og obligasjoner

Rentenivåer er viktig når det gjelder økonomiske beslutninger og ytelse i enhver økonomi hvor markedet spiller en betydelig rolle. Rentenivåer påvirker viljen til å spare, men den påvirker også etterspørselen etter og allokeringen av lånte midler. Rentenivåene bestemmer også, sammen med utenlandske rentenivåer, forventede endringer i valutakurs og forventet inflasjonsrate, fordelingen av oppsparte midler mellom innen- og utenlandske finansielle eiendeler og fysiske eiendeler. Når det kommer til viktigheten av rentenivåer avhenger dette av hvorvidt landets økonomi hovedsakelig har grunnlag i finansielt system eller pengepolitikk. Jo mer landets økonomi baseres på finansielle system og pengepolitikk, desto viktigere rolle spiller rentenivåer. Renteendringer er et virkemiddel som brukes til å hjelpe med å balansere tilbud og etterspørsel av varer og penger (Lanyi & Rüsdü, 1983).

Obligasjoner defineres som en representasjon av et krav på en forhåndsbestemt sekvens av betalinger. Obligasjoner blir utstedt med en bestemt forfallsdato med forhåndsdefinerte nedbetalingsdatoer og hvor mye som skal betales ved forfall. Dette legger grunnlaget for løpetidsstrukturen, som for renter viser sammenhengen mellom rentenivået og løpetiden for et gjeldsinstrument med fast forfallsdato. Strukturen relaterer rentesats til løpetid og er stort sett stigende hvor renter med lang løpetid er høyere enn renter med kort løpetid og renten stiger med løpetiden. Renter kan også være nedadgående eller pukkelformet hvor de høyeste rentesatsene er på mellomlange løpetider (Shiller & Huston McCulloch, 1990). Videre undersøkes forholdet mellom ulike renter over tid for lange og korte løpetider. Analysens funn viser at obligasjonsmarkedets tidsserier beveger seg tilfeldig og det er ikke funnet tegn til et stabilt og signifikant forhold mellom lange og kortsiktige renter over tid. Dette funnet utfordrer den tradisjonelle oppfattingen av at lange løpetider ofte tilpasser seg de korte løpetidenes rentenivå (Brick & Thompson, 1978).

Når det gjelder statsobligasjoner verdsettes likviditeten og sikkerhetsaspektene av investorer. Dette er dokumentert ved å undersøke endring i tilbudet og inspisere effektene på ulike avkastninger. Statsobligasjoner sammenlignes til et visst nivå med penger, grunnet den ekstreme sikkerheten (Krishnamurthy & Vissing-Jorgensen, 2012). Den fremste sikre eiendelen over hele verden de siste tiårene er amerikanske statsobligasjoner, selv om den store og økende størrelsen på amerikanske underskudd utfordrer dette (Zhiguo & Krishnamurthy, 2020). Viktigheten av renter, spesifikt statsobligasjoner er veldokumentert i litteraturen og det er enighet i at statsobligasjoner spiller en

stor rolle i verdensøkonomien så vel som pengepolitikken. De siste 25 årene har det blitt gjort stor fremgang i modellering av dynamikken i løpetidsstrukturen til rentesatser (Yan, 2001).

Sikre eiendeler er en integrert funksjon til banker, finansmarkeder og det internasjonale finanssystemet. Store økonomiske hendelser gir svingninger i økonomi, og Covid-19-krisen var intet unntak. Denne krisen påvirket statsobligasjonene ved at hendelsene i mars 2020 ikke fulgte det etablerte mønsteret hvor statsobligasjoner vanligvis øker i verdi i urolige tider. Det ble oppdaget en kontrast i prisbevegelser imellom koronakrisen i 2020 og finanskrisen i 2008, men denne kontrasten vises kun for lange løpetider. Den uvanlige økningen i rentene på langsiktige statsobligasjoner indikerer at investorene tvilte på sikkerheten i obligasjonene og det blir stilt spørsmål til hvorvidt hendelsene var en teknisk glipp eller et tegn på at internasjonale investorer beveger seg bort fra dollaren (Zhiguo & Krishnamurthy, 2020).

2.3. Maskinlæring

Hovedmålet med maskinlæring er nøyaktige beslutningssystemer som automatiserer oppgaver som ellers ville blitt gjort av mennesker og det finnes en rekke algoritmer som har hatt viktige suksesser innen vitenskap og industri (Wojciech et al., 2021). Et kritisk område for maskinlæring er tidsserieprediksjon, ettersom flere prediksjonsproblemer innebærer en tidskomponent (Yamak et al., 2020). De siste 25 årene har det vært betydelige fremskritt innen teoretiske modeller for rentestruktur og deres økonomiske estimater, men det er et viktig område da forutsigelse av rentepunkter er avgjørende for forvaltning av obligasjonsporteføljer (Diebold & Li, 2006). I tillegg til å være et av de viktigste områdene, er de også en av de mer utfordrende problemene innen finansområdet, å forutsi fremtidige bevegelser i renter. Dette skyldes blant annet den ikke-lineære og dynamiske naturen til renter samt de komplekse og hyppige svingningene i tidsseriedata da det kompliserer prediksjonene (Enke & Mehdiyev, 2012). Selv om det er et komplekst område, spiller predikering av lange tidsserier en viktig rolle for investeringsbeslutninger (Wang et al., 2023).

Det finnes flere metoder for å predikere rente. Våre forskningsspørsmål baserer seg på tradisjonelle og moderne modeller for rentepredikjon. Det finnes flere tradisjonelle maskinlæringsmodeller og en av de mest kjente, ARIMA, har vært populær for tidsserieprediksjon de siste tre tiårene er kjent for sin nøyaktighet i prediksjon og fleksibilitet i å representere flere ulike typer tidsserier (Zhang, 2003).

ARIMA har i kraft av sin popularitet blitt mye forsket på, mens det er begrenset forskning på bruken av ridge-regresjonsmodeller for rentepredikjon spesielt innenfor univariat modellering. Ridge

regresjon har vist seg å være nyttig, ikke bare for å estimere løpetidsstrukturen til renter, men også når man benytter større økonometriske modeller (Watson & White, 1976). Det finnes mer forskning på ridge regresjon på andre områder. Blant annet er det foreslått en alternativ regresjonsbasert tilnærming metode for prising av renter (Adrian et al., 2013) og Ridge regresjon benyttes til å predikere mislykkede selskaper (Pereira et al., 2016). Regulariseringsmetoder har også blitt forsket på i stor grad innen prediksjon av kvantitativ genetikk (Vlaming & Groenen, 2014)

Random Forest virker heller ikke å være i stor grad brukt til rentepredikSjon, men modellen benyttes i flere folkehelse-studier (Kane et al., 2014). Random Forest er en modell som kan brukes til både klassifiserings- og regresjonsproblemer. Modellen er populær da den kan anvendes på et bredt spekter av prediksjonproblemer. At modellen har få parametere som trenger justeringer gjør den i tillegg enkel å bruke. Modellen har vist seg å være effektiv i mange vitenskapelige felt, som finansielle studier, medisin, og miljø – og geofysiske vitenskaper (Tyralis & Papacharalampous, 2017). En viktig fordel ved å bruke random Forest for prediksjonsmodeller er modellens evner til å håndtere datasett med en stort antall forklaringsvariabler (Speiser et al., 2019).

Nylig har kunstige nevrale nettverk blitt forsket mye på og anvendt i tidsserieprediksjon (Zhang, 2003) og er blant de mest populære dype læringsmodellene som benyttes i dag (Yadav et al., 2020). Dype neurale nettverk har blitt foreslått som en måte å produsere modeller med bedre forutsigbarhet (Wojciech et al., 2021). I slike modeller har hvert lag en lineær transformasjonsfunksjon som samlet sett konstruerer en dyp nevral modell. Ved å inkludere mange funksjoner forbedres modellens prediksjonsevne. Disse modellen har vært utfordrende å modellere på grunn av deres vanskelige bakgrunn, men modelleringen har blitt betydelig bedre med økt tilgjengelighet i innebygde kodebiblioteker (Albeladi et al., 2023).

2.4. Tidsserie-data og -modeller

Tidsseriedata defineres som et sett med kvantitative observasjoner som er sortert i kronologisk rekkefølge. Innen økonomi finner man ofte tidsserier som representerer historiske observasjoner av økonomien (Fuller, 2009). Amerikanske statsobligasjoner målt over tid er tidsseriedata og målene med vår analyse er å prognostisere dataene for å forutsi fremtidige observasjoner. Tidsseriedata innen næringsliv og økonomi har en tendens til å vise mønstre som trender, sesongmessige svingninger, uregelmessige sykluser og sporadiske endringer i nivå eller variasjon, å beregne effekten av kjente ytre påvirkninger samt å identifisere overraskende hendelser (Nerlove et al., 2014).

I de senere årene har modellering av ikke-lineære tidsserier økt i popularitet (Luukkonen et al., 1988), men disse er fortsatt i tidlig fase sammenlignet med lineære tidsserier. På slutten av 1970-tallet og tidlig på 1980-tallet ble det stadig tydeligere at lineære modeller ikke er tilstrekkelige (De Gooijer & Hyndman, 2006).

Tidsseriedata Y_t dekomponeres til komponentene trend T_t, sesong S_t og residual R_t. En multiplikativ dekomponering, som vanligvis benyttes for økonomiske tidsserier, kan skrive som:

$$Y_t = S_t \times T_t \times R_t \quad (1)$$

Alternativt kan man benytte additiv dekomponering:

$$Y_t = S_t + T_t + R_t \quad (2)$$

Etter dataene er transformert til punktet hvor variasjonen i serien virker å være stabil over tid (Hyndman & Athanasopoulos, 2021). At dataen er av type tidsseriedata er en viktig del av forskningsspørsmålene våre og vi vil derfor først presentere komponentene i tidsseriedata før vi går videre med tidsseriens og modellenes egenskaper.

2.4.1. Trend

En trend i data indikerer en langsiktig oppgang eller nedgang og er ikke nødvendigvis lineær. En trend kan også beskrives som å "endre retning", for eksempel fra å øke til å avta. Sesongmønstre oppstår når data påvirkes av periodiske faktorer som dag, uke, måned eller kvartal. Når tidsseriedata viser et mønster uten faste frekvenser, beskrives dette som et syklisk mønster (Hyndman & Athanasopoulos, 2018).

2.4.2. Sesongkomponent

Sesongvariasjoner viser seg som regelmessige svingninger i en tidsserie, enten gjennom samme måned, for flere måneder hvert år eller i samme kvartal hvert år (Hyndman & Athanasopoulos, 2021). Sesongvariasjoner måles innenfor 12 måneder. Slike variasjoner kan oppstå enten som et resultat av naturlige krefter, hvor klima ofte spiler en stor rolle, eller som menneskeskapte mønstre som følge av regelmessige hendelser («Components of Time Series Analysis», 2018).

2.4.3. Residualer

I en tidsseriemodell representerer residualer forskjellen mellom faktiske observasjoner og modellens tilpasninger. Ideelt sett bør residualer ikke vise noen korrelasjon, noe som indikerer at modellen har utnyttet all tilgjengelig informasjon i dataene. Residualer bør også ha et gjennomsnitt på null for å unngå skjevhet i prognosene. Mens justering for skjevhet er rett frem, krever det å løse korrelasjonsproblemer en mer detaljert tilnærming. Konstant varians og normalfordeling av residualer er ønskelig, men ikke nødvendig. Disse egenskapene forenkler beregning av prediksjonsintervaller, selv om de ikke alltid kan oppnås (Hyndman & Athanasopoulos, 2018).

Residualenes oppførsel er avgjørende for en nøyaktig analyse av tidsserie-data. Ved å sikre seg at residualene oppfører seg som hvit støy, kan vi gjøre med nøyaktige prediksjoner og velge riktig modell. Hvit støy er en type tidsserie hvor variablene ikke påvirker hverandre. Variablene har et gjennomsnitt på 0 og en fast varians og variablene vil derfor ligge på eller tett opp mot 0. Dette betyr at variablene er tilfeldige, og uten mønster over tid. Det er derfor ingen sammenheng eller forutsigbarhet fra ett datapunkt til neste (Fuller, 2009). Hvit støy er en indikasjon på at modellen tilpasser seg dataen godt. Dette betyr at den har fanget opp all relevant informasjon i tidsserien og at det ikke er noen gjenværende mønster eller trender som ikke er tatt høyde for (Zanwar, 2023).

Vi vil videre se på egenskapene til tidsserien og modellene.

2.4.4. Stasjonaritet

Når data ikke er stasjonær betyr det at dens statistiske egenskaper, eksempelvis gjennomsnitt og standardavvik, ikke er konstante over tid (Serafeim, 2023). En stasjonær tidsserie endrer ikke egenskaper over tid, så verdiene er uavhengige av når de måles. Tidsserier med trender eller sesongmessige svingninger er ikke stasjonære da disse faktorene påvirker verdien. Hvit støy er derimot stasjonær fordi dens egenskaper er konsistente over tid. En tidsserie med sykliske mønstre uten fast lengde kan også være stasjonær ettersom vi ikke kan forutsi når topp- og bunn-punktene oppstår. Stasjonære serier har vanligvis ingen langvarige forutsigbare mønstre, og vil fremstå horisontale med konstant varians i et tidsplott (Hyndman & Athanasopoulos, 2018).

For å stabilisere en ikke-stasjonær tidsserie kan man bruke differensiering, som innebærer å beregne forskjellen mellom påfølgende observasjoner. Dette hjelper med å stabilisere gjennomsnittet av tidsserien ved å fjerne endringer i nivået og kan redusere både trend og sesongvariasjoner. Ved å

analysere tidsseriens autokorrelasjonsfunksjon (ACF), kan man identifisere om en tidsserie er stasjonær eller ikke. En rask nedgang i ACF indikerer stasjonaritet, mens en langsom nedgang kan tyde på ikke-stasjonaritet. Korrekt bruk av disse teknikkene kan bidra til mer nøyaktige analyser og prognoser (Hyndman & Athanasopoulos, 2018).

Phillips-Perron (PP) testen gjør en ikke-parametrisk korreksjon til t-test statistikken, og fungerer derfor godt for uspesifisert autokorrelasjon. Nullhypotesen for PP-testen er også ikke-stasjonaritet i datasettet (Phillips & Perron, 1986).

Augmented Dicky Fuller (ADF) test antar at data er ikke-stasjonær, og introduserer en bestemt mengde forsinkede verdier (lags) av de avhengige variablene som regresjonsvariabler. Testen velger automatisk antall forsinkelser basert på en forhåndsdefinert standardligning (Cheung & Lai, 1995).

PP-testen har samme nullhypotese som ADF-testen: ikke stasjonær data (Phillips & Perron, 1986). Denne testen skiller seg fra ADF-testen ved å håndtere seriell korrelasjon og heteroskedastisitet i feilene. Heteroskedastisitet refererer til en gitt situasjon i tidsserie og regresjons modeller, hvor variansen i feilene ikke er koststant gjennom hele datasettet. Heteroskedastisitet i tidsserieanalyser er essensielt for å indentifisere og korrigere for å forbedre modellens prediksjon. Å korrigere for heteroskedastisitet vil bidra til en mer nøyaktig standardfeil, bedre konfidensintervaller, og bedre hypotesetester som igjen er viktig når det skal tas konklusjoner basert på gitte statistiske analyser (Shumway & Stoffer, 2017).

En annen objektiv metode å sjekke om dataene er stasjonære kan være enhetsrots test. Vi har valgt å gjennomføre en Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test i vår analyse. KPSS-test velger ofte færre forskjeller enn ADF- og PP-test. Den har en nullhypotese om stasjonaritet i dataen, mens ADF- og PP-test antar at dataene ikke er stasjonære. Som følge av dette vil KPSS-testen bare velge enn eller flere differanser hvis det er tilstrekkelig bevis for å utfordre antagelsen om stasjonaritet. I KPSS-testen vil små p-verdier, eksempelvis mindre enn 0.05, si at differensiering er nødvendig (Hyndman & Athanasopoulos, 2018).

2.5. Tradisjonelle prediksjonsmodeller

2.5.1. Auto Regressive Integrated Moving Average (ARIMA)

For å analysere forskningsspørsmål 1 vil vi først benytte ARIMA modell som en av de tradisjonelle modellene. Auto Regressive (AR) modeller er en metode for regresjon eller predikere nåværende verdier ved hjelp av foregående verdier fra samme tidsserie (Hyndman & Athanasopoulos, 2018). AR-modeller er veldig tilpasningsdyktige, og kan fange opp ulike mønstre i tidsseriedata ved justering av gitte parametere. AR-modellen ble foreslått allerede i 1926 (Yule, 1926). AR er bygget på autokorrelasjon der den avhengige variabelen avhenger av tidligere verdier av seg selv. Eksempelvis vil dagens rentenivå være avhengig av gårdagens rentenivå, altså historisk data (Nathaniel, 2021).

Moving Average (MA) modeller ble på sin side introdusert av Slutsky (1937, som referert i Makridakis & Hibon, 1997). MA modeller genererer nåværende verdier basert på feil i tidligere prognoser i motsetning til AR som bruker tidligere verdier i tidsserien. Begge disse modellene er grunnleggende konsepter innen tidsserieanalyse. Ideene ble syntesistert i 1938 ved å demonstrere at kombinasjonen av disse, ARMA (p, q) prosesser effektivt kan benyttes for å modellere stasjonære tidsserier gitt at man benytter passende valg av AR (p) og MA (q) som beskrevet av Wold (1938, som referert i Makridakis & Hibon, 1997). Det var ikke før 1960-tallet at bruken av ARMA modeller ble mer utbredt etter datamaskinen ble mer tilgjengelig for å kunne utføre komplekse beregninger knyttet til modellen. Tidsserien Y_t kan modelleres som en kombinasjon av tidligere verdier (AR) og tidligere feil (MA). I 1970 ble metoden kjent som Box-Jenkins metodikk for ARIMA modeller introdusert. Modellen som implementerer «I», hvor I står for "Integrert" og definerer behovet for å differensiere tidsseriedata for å gjøre tidsserien stasjonær, ble svært populær på 1970-tallet. Ved å inkludere differensiering ble modellen også i stand til å håndtere trend eller sesongmessige komponenter (Makridakis & Hibon, 1997).

$$Y_t' = c + \phi_1 Y_{t-1}' + \cdots + \phi_p Y_{t-p}' + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t, \quad (3)$$

I formelen er Y_t' den differensierte verdien av tidsserien. c står for konstant mens koeffisienten til AR-termene er definert med ϕ. Koeffisienten til MA-termene defineres med θ og ε_t er hvit støy. I tillegg viser q ordenen til MA, p ordenen til AR og d differensieringsgraden. Det er enkelte trinn som må følges for å modellere tidsserier fra det virkelige liv ved bruk av (3). Y_t må først transformeres til å bli stasjonær rundt gjennomsnitt og varians, før man identifiserer riktig rekkefølge av p og q. I tillegg må verdien av parameterne $\phi_1, \phi_2, \ldots, \phi_q$ og/eller $\theta_1, \theta_2, \ldots, \theta_q$ estimeres med en ikke-lineær

optimaliseringsprosedyre for å minimere summen av kvadrerte feil (Hyndman & Athanasopoulos, 2018).

Populariteten til ARMA (p, q) modeller økte ved å presentere prosedyrer for hvordan man kan gjøre tidsseriene stasjonære samt hvordan man identifiserer optimale verdier for p og q. For å finne de optimale verdiene ble det foreslått å benytte korrellogram autokorrelasjon og partiell autokorrelasjon. Når modellene var estimert, ble det foreslått diagnostikk for å sjekke om residualene var hvit støy. I disse tilfellene ble modellen ansett som riktig definert. Dersom dette ikke var tilfellet, ble det foreslått en ny modell ved bruk av de tidligere stegene (Box et al., 1994). Det er likevel verdt å merke seg at når man modellerer univariat ARIMA-modell finnes det ingen eksakt oppskrift og to forskere kan komme frem til ulike modeller for samme tidsserie (Kang, 1986).

I ARIMA modellering spiller Autocorrelation Function (AFC) og Partial Autocorrelation Function (PACF) en viktig rolle. ACF brukes til å indentifisere MA, mens PACF indentifiserer antall AR. En signifikant verdi i ACF og PACF plott indikerer potensielt et AR eller MR ledd i en ARIMA modell (Hyndman & Athanasopoulos, 2018)

For å velge ARIMA-modell kan Akaike Information Criterion (AIC) gi en god indikasjon. AIC brukes ofte som mål for modellvalg, spesielt innen regresjonsmodeller og tidsserieanalyse. Dette benyttes også til å bestemme ordenen i en ARIMA-modell. Det er viktig å merke seg at AIC generelt ikke er effektive for å bestemme differensieringsgraden (d), men er nyttig for valg av AR (p) og MA (q) ettersom differensieringsprosessen endrer datasettet som brukes til å beregne sannsynligheten (Hyndman & Athanasopoulos, 2018).

ACF viser tidsseriens korrelasjon med egne forsinkede verdier, eksempelvis mellom Y_t og Y_{t-2}. PACF reflekterer korrelasjon ved en spesifikk forsinkelse som ikke allerede er forklart av de lavere forsinkelsene. Typisk for AR er ACF med gradvis nedgang og PACF med tydelig fall etter de første par laggene. Typisk for MA er motsatt tilfelle, ACF med tydelig fall og PACF med gradvis nedgang (Bårsaune, 2017).

AIC er et nyttig verktøy for å definere ARIMA order. AIC er definert som (Hyndman & Athanasopoulos, 2018):

$$AIC = -2\log(L) + 2(p + q + k + 1) \tag{4}$$

L er sannsynligheten for dataen, hvor antall parametere som skal estimeres $K = 1$ hvis $c \neq 0$ og $k = 0$ hvis $c = 0$. For ARIMA modeller kan AIC ligningen presenteres slik (Hyndman & Athanasopoulos, 2018):

$$AICc = AIC + \frac{2(p+q+k+1)(p+q+k+2)}{T-p-q-k-2} \quad (5)$$

Ved å kompensere for antall parametere gis en mer rettferdig sammenligning dersom man ser modeller med ulikt antall parametere mot hverandre. Typisk utfall ved bruk av andre evalueringskriterier er at en modell med flere parametere gir best resultat. AIC baserer tar høyde for kompleksiteten til modellen (antall parametere) i tillegg til hvor godt modellen forklarer dataen, altså maksimert sannsynlighetsfunksjon. På denne måten kan man sammenligne modeller med ulikt antall AR og MA komponenter, hvor modellen med lavest AIC anses som den beste (Bårsaune, 2017).

2.5.2. Ridge regresjon

Den andre tradisjonelle modellen vi har benyttet i vår analyse er ridge regresjonsmodell. Innen maskinlæring finnes ulike former for overvåket læring. Regresjon er en modell som gir et tall som utdata og vil være nyttig for å predikere eksempelvis aksjemarkedspriser og finne mønstre i aksjer (Edwards, 2020). Lineær diskriminantanalyse ble forslått i 1936, før flere forfattere foreslo en alternativ tilnærming kalt logistisk regresjon. I starten av 1970-tallet ble begrepet generalisert lineær modell introdusert, noe som beskrev en hel klasse av statistiske læringsmetoder herunder både lineær og logistisk regresjon. På slutten av 1970-tallet var mange flere teknikker introdusert, men disse var nesten utelukkende lineære metoder fordi tilpasning av ikke-lineære forhold var beregningsmessig vanskelig. På 1980-tallet var ikke dette et like stort problem, og videreutviklingen ga økt popularitet til modeller som nevrale nettverk (James et al., 2023).

En av de grunnleggende antagelsene for lineær regresjon er at feilen i prediksjonene er homoskedastisk, altså med konstant varians. Denne antagelsen er viktig fordi resultatet av lineær regresjon ikke blir pålitelig dersom antagelsen ikke er oppfylt. For å sikre upartisk og robust estimering er det viktig å fjerne heteroskedastisitet, altså ikke-konstant varians i data ved å eksempelvis utføre logaritmebasert datatransformasjon (Dekha, 2021). Dette er en metode hvor hver variabel X erstattes med $\log(X)$. Valg av logaritmebase er opp til analytikeren, vi fokuserer på den naturlige logen som betegnes som ln. Når data ikke følger klokkekurven kan dataene logtransformeres for å gjøre de så

normale som mulig for å sikre høyest grad av gyldighet i resultatene. En log transformasjon vil med andre ord fjerne skjevheten i dataene. Det er viktig å påse at de opprinnelige dataene følger en omtrentlig log-normal distribusjon for at transformasjonen skal fungere (Htoon, 2020).

Enkel lineær regresjon er en simpel metode for å forutsi kvantitativ respons Y basert på enkel prediksjonsvariabel X. Modellen tar utgangspunkt i omtrentlig lineær sammenheng $\approx$ mellom X og Y. Den lineære sammenhengen kan beskrives som (James et al., 2023):

$$Y \approx \beta_0 + \beta_1 X \tag{6}$$

$\beta_0 \, og \, \beta_1$: to ukjente konstanter som representerer skjæringspunktet og stigningstermene i den lineære modellen, også kjent som modellens koeffisienter.

En lineær regresjonskoeffisients tegn forteller om korrelasjonen mellom hver uavhengig og avhengig variabel er positiv eller negativ. Når koeffisienten er positiv indikerer det at verdien av den uavhengige variabelen øker. Videre ser man da også at den avhengige variabelen en tendens til å øke. Motsatt gjelder også for negativ koeffisient, når uavhengig variabel øker, tendenser den avhengige variabelen til å minske (Frost, u.å.).

Regresjonsanalyse går ut på å bestemme hvordan forholdet er mellom uavhengige og avhengige variabler. Koeffisienter forteller om endringene og p-verdiene viser signifikansen til koeffisientene. Nullhypotesen i lineær regresjon er null korrelasjon med den avhengige variabelen. Dersom p-verdien er lavere enn signifikansnivå forkastes nullhypotesen og man kan si at endringer i uavhengig variabel assosieres med endring i avhengig variabel (Frost, u.å.).

Ridge regresjon er kjent som en Tikhonov-regularisering og er en metode som brukes til å analysere data fra multippel regresjon hvor man står ovenfor utfordringer knyttet til multikollinearitet i datasett. Multikollinearitet refererer til tilfeller der uavhengige variabler i en regresjonsanalyse er sterkt korrelerte (Gomede, 2024). Ridge regresjon, som også er kjent som L2–regularisering i regresjonsmodeller, bruker parameteren lambda (λ) for å kunne justere styrken på straffemekanismen i en regresjonsmodell. Valg av λ er kritisk og utføres vanligvis ved hjelp av kryssvalidering for å finne en balanse som minimerer summen av kvadrerte feil i valideringssettet. En λ som viser null vil i Rigde regresjonens resultater vise at den har en vanlig minste kvadraters regresjon. Når verdien av λ økes blir straffetermen mer effektiv og det vil føre til at koeffisientene

trekkes mot null. Dette kan dempe effekten av multikollinearitet og forbedre modellens nøyaktighet ved å redusere variansen (James et al., 2023).

Selv om Rigde regresjonsmodellen krymper koeffisientene vil ikke modellen sette noen av dem helt til null. Dermed forblir alle variablene inkludert i modellen selv med redusert påvirkning. Før modellering vil det være essensielt å standardisere prediktorene for å sikre at straffetermen påvirker alle variablene jevnt i modellen. Ved å optimalisere λ gjennom kryssvalidering vil man kunne opprettholde en passende balanse mellom bias og varians i modellen og dermed forhindre under– og overtilpasning (James et al., 2023).

Formelen for ridge regresjon presenteres på følgende måte:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2 \qquad (7)$$

$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)$ er summen av de kvadrerte residuale feilene for alle n i datasettet, hvor y_i er den observerte verdien, β_0 er intercept- termen, x_{ij} er verdien av prediktor j for observasjon i og β_j er koeffisientene tilknyttet prediktor j. Rigde regresjonen reguleringsterm, hvor λ er regulariseringsparameteren og β_j er de ikke-intercept koeffisientene i modellen (James et al., 2023).

Residual sum of squares (RSS) er den første delen av kostnadsfunksjonen. $\lambda\sum_{j=1}^{p}\beta_j^2$ er regulariseringstermen der λ er en ikke negativ hyperparameter som styrer styrken på regulariseringen. Når λ er stor blir straffen for store koeffisienter større noe som kan føre til at modellen prioriterer enkelthet og kan dermed få økt bias. Når λ er liten er liten vil modellen ligne mer på vanlig multiple square error (MSE) med mindre vekt på regularisering og dermed lavere bias og høyere varians (James et al., 2023).

For å videre svare på forskningsspørsmål 1 har vi brukt moderne modeller i tillegg til de tradisjonelle modellene presentert over. De moderne modellene har tydelige forskjeller fra de moderne, noe vi vil se nærmere på nå. Blant annet håndterer Random Forest og LSTM godt ikke-linearitet i dataene i tillegg til å håndtere støy bedre enn de tradisjonelle modellene.

2.6. Moderne maskinlæringsmodeller

2.6.1. Random Forest

Random forest er en ensemble-læringsmetode som brukes for klassifisering og regresjon ved å konstruere tilfeldige beslutningstrær og predikerer ved å bruke gjennomsnitt av resultatene. Etter Leo Breiman (2001, som referert i Scornet et al., 2015) publiserte sin artikkel på læringsmetoden har metoden blitt svært utbredt (Scornet et al., 2015). Random forest modellen er populær da den er anvendelig på et bredt spekter av prediksjonsproblemer, den er enkel i bruk og har blitt brukt på mange praktiske problemer tidligere (Biau & Scornet, 2016).

Random Forests består av beslutningstrær hvor man tar tilfeldig utvalg av M prediktorer på treningsutvalg av det fulle datasettet som kandidater, hver gang man vurderer en splitt i et tre. Det gis kun mulighet for å benytte en av disse M prediktorene for hvert utvalg, før man på nytt velger et annet utvalg. Dette gjøres ved hver splitt. Beslutningstrær kan brukes til både regresjon og klassifikasjonsproblemer i Random Forest (James et al., 2023).

Når man bruker Random Forest til regresjonsproblemer kan regresjonstrær eksempelvis benyttes til å forutsi lønninger til baseballspillere basert på antall år med erfaring og antall treff de hadde sesongen før. Regresjonstrærne deles inn i forskjellige grupper basert på terskler og benyttes videre til å predikere lønninger for spillerne som er basert på deres lønninger og treff (James et al., 2023).

Det er tre hoved hyperparametre som må defineres før Random Forest-modellen kan trenes: nodestørrelse, antall trær og antall egenskaper som skal prøves. Algoritmen er bygget opp av flere beslutningstrær. Et ytterligere element av tilfeldighet, i tillegg til treningsutvalg med tilfeldig valg av M føres inn ved hjelp av 'feature bagging', som øker mangfoldet i datasettet og reduserer likheten mellom beslutningstrærne (IBM, u.å.). Den generelle algoritmen til random Forest ser ut som dette (Xiao, 2015):

```
1  Select the number of models to build, m
2  for i = 1 to m do
3      Generate a bootstrap sample of the original data
4      Train a tree model on this sample
5      for each split do
6          Randomly select k (< P) of the original predictors
7          Select the best predictor among the k predictors and
           partition the data
8      end
9      Use typical tree model stopping criteria to determine when a
       tree is complete (but do not prune)
10 end
```

Figur 1 Generell algoritme for Random Forests.

Gjennom denne tilnærmingen så kan vi se nytten av regresjonstreets evne til å kunne håndtere komplekse datasett og avdekke mønster som man ellers ikke vil kunne sett. Det vil kunne gi oss en dypere forståelse av vår tidsseriedata.

2.6.2. Long Short-Term Memory (LSTM)

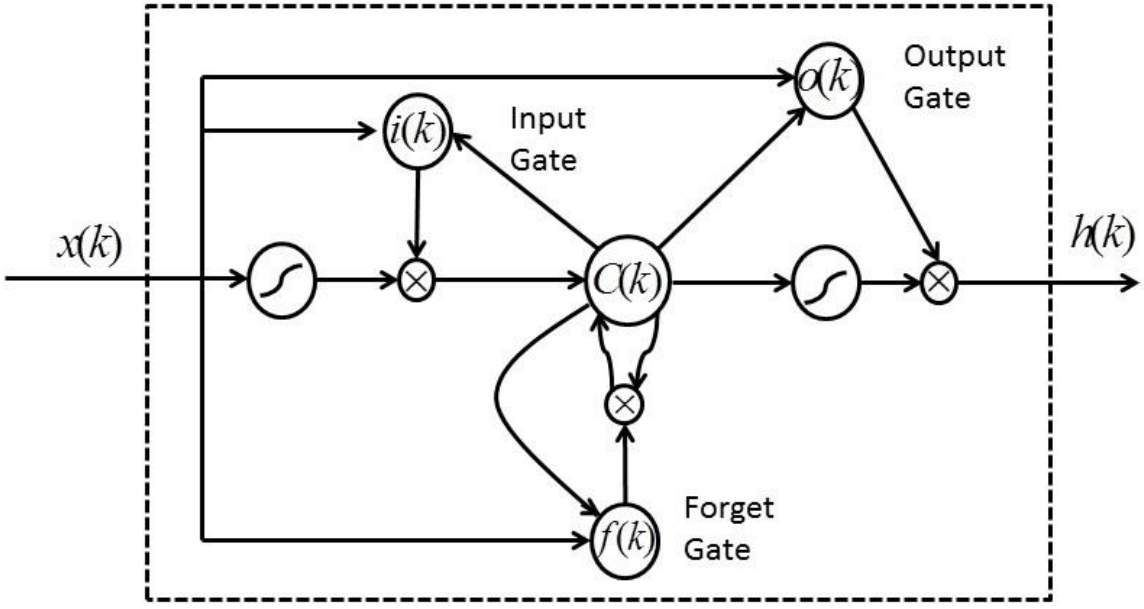
I senere år har LSTM blitt populær innen maskinlæring for flere forskningsområder (Yu Wang, 2017) og er en av de mest avanserte dype læringsteknikkene for å fange opp dynamikk i finansiell og økonomisk tidsseriedata (Park & Yang, 2022).

Recurrant Neural Network (RNN) er en velkjent struktur som benytter seg av blant annet tidsseriedata (Yu Wang, 2017). RNN brukes vanligvis for ordnede eller tidsmessige problemer, eksempelvis språkoversettelse, talegjenkjenning og bildeteksting. RNN bruker i likhet med mange modeller treningsdata for å lære, men skiller seg ut ved å hente ut informasjon fra tidligere inndata for å påvirke nåværende inndata og utdata. Dette kan betegnes som et minne. Tradisjonelle dype nevrale nettverk antar uavhengighet mellom inndata og utdata, mens i RNN avhenger utdataen av de tidligere elementene i sekvensen (IBM, u.å.).

Undersøkelser av effektiviteten av dyp læring for å forutsi avkastningen til amerikanske statsobligasjoner, nærmere bestemt 10 års løpetid, ved hjelp av LSTM modell gir lovende resultater med lav gjennomsnittlig kvadrert feil. Dette tilsier høy nøyaktighet i forutsigelsene (Shu & Chou, 2021). Det har blitt utført forskning hvor Artificial Neural Network (ANN) benyttes for å forutsi retningen på amerikanske statsobligasjoner med 30 års løpetid. ANN modell for ukentlig data fra 1989 og utover på kortsiktig, mellomlang og langsiktig løpetid viser en gjennomsnittlig nøyaktighet på 67% over en

tidsperiode på 5 år (Wei et al., 2001). En LSTM modell har også blitt forsket på for å forutsi valutakursen for USD/CAD, GRB/USD, og AUS/USD med data fra 1997-2019. LSTM modeller har blitt mer populære i ulike analyser da modellens evne til å håndtere ikke-lineære sammenhenger mellom variabler og utfallet som skal forutses. Dataene lar LSTM modellen forså å vurdere effekten av variabler på valutakursene som skal predikeres (Wijesinghe, 2020).

En av de mest populære RNN-strukturene er LSTM som ble introdusert av Hochreiter & Scmidhuber i (1991, som referert i Yu Wang, 2017). De forklarer at LSTM er en RNN modell som er en avansert type tilbakevendende Neuralt nettverk. Modellen er designet for å overkomme problemer med eksplorerende gradienter som vanligvis oppstår når man trener langtidsavhengigheter, selv når de minste tidsforsinkelsene er veldig lange. Generelt kan dette forhindres ved å bruke en Constant error Carousel (CEC) som vil opprettholde feltsignalet innenfor hver enkelt celle. Det er slike celler som er tilbakevendende nettverk i seg selv, men i LSTM er CEC utvidet med funksjoner som inngangsport og utgangsport som danner enhetscelle (Hochreiter & Schmidhuber, 1997).



Figur 2 LSTM stuktur (Yu Wang, 2017)

For å sikre konstant overflod av feil inneholder LSTM en minnecelle. Minnecellen inneholder tre porter. Inngangsport, utgangsport og glemmeport. Minnecellen gjør det mulig å bevare viktig informasjon som mates inn i modellen og glemme informasjon som er irrelevant for videre trinn i

modellen (Yu Wang, 2017). Denne strukturen er fremstilt i Figur 2. Glemmeporten ble forslått av Gers, Shmidhuber & Cummins (1999) og er designet til å lære seg å resette minneblokker når innholdet går ut på dato for å unngå at cellestatusene vokser ubegrenset. Dersom dette skjer kan det føre til at viktig informasjon blir oversett (Gers et al., 2000). Glemmeporten reduserer betydningen av eldre data gradvis, og sørger for at ny, ofte mer relevant data ligger igjen i minnecellen. Inngangs- og utgangsportens verdier beskytter minnecellens verdi ved å regulere inngående og utgående data. Dersom inngangs- og utgangsportene er stengt, men glemmeporten er åpen vil minnecellens verdier forbli uforandret. Dette illustrerer LSTMs evne til å bevare data over tid (Pulver & Lyu, 2017).

Hvordan et nevralt nettverk opptrer avhenger av blant annet aktiveringsfunksjonen som blir brukt i hver node. Verdien av aktiveringsfunksjonen setter grensene og total styrke på inn- og utgangssignalet til noden. Aktiveringsfunksjonen har stor innvirkning på ytelsen til det nevrale nettverket (Farzad et al., 2019). En aktiveringsfunksjon er en matematisk funksjon som transformerer utgangene til et ikke-lineært format før det sendes til neste lag og kartlegger summeringsresultatet til et ønsket område. På denne måten bidrar funksjonen til å identifisere om nevronet må bli aktivert. Aktiveringsfunksjonen kartlegger verdier til området [0,1], dette er nyttig når modellen skal predikere sannsynligheter (Madan & Madhavan, 2020). I treningsdatasettet er skaleringstallene minimums- og maksimumsverdiene, disse brukes til å skalere både trenings- og testdatasettene, i tillegg til de predikerte verdiene. Dette gjøres for å sikre at minimums- og maksimumsverdiene i testdataene ikke påvirker modellen (James, C, 2019). Den vanligste aktiveringsfunksjonen i en LSTM-modell er sigmoid σ, som vi har benyttet i vår oppgave, og hyperbolic tangent (Farzad et al., 2019).

Sigmoid funksjonen:

$$\sigma(x) = \frac{1}{1+e} \tag{1}$$

Verdier nær 0 i en LSTM modells glemmeport betyr at informasjonen som er kommet inn glemmes, mens en verdi nær 1 betyr at informasjonen skal lagres og brukes videre. Beslutningen som tas for å glemme eller lagre vil igjen påvirke hvor mye av den tidligere cellestatusen vil bli videreført og påvirke trinnene videre i modellen. Glemmeporten i modellen bestemmer hvilken informasjon skal glemmes og det brukes et Sigmoid lag for å ta denne avgjørelsen (Yu Wang, 2017).

Formlene til en LSTM modell ser ut som følger:

$$i(k) = g\big(W_i\big(h(k-1), x(k)\big) + b_i\big) \quad (2)$$

$$f(k) = g(W_f\,(h(k-1), x(k)) + b_f\,) \quad (3)$$

$$o(k) = g(W_o(h(k-1), x(k)) + b_o) \quad (4)$$

$$\tilde{C}(k) = \tilde{g}W_c(h(k-1), x(k)) + b_c) \quad (5)$$

$$C(k) = f(k)C(k) + i(k)\tilde{C}(k) \quad (6)$$

Formel (2) viser LSTM inngangsport $i(k)$, (3) viser glemmeport $f(k)$, (4) viser utgangsport $o(k)$. I formel (5) ser vi ny minnecelle $\tilde{C}(k)$, (12) er den oppdaterte nye minnecellen $C(k)$. $g(\cdot)$ er aktiveringsfunksjonen for inngangs- glemme- og utgangsport, som vanligvis er valgt ved hjelp av sigmoid aktiveringsfunksjon. $\tilde{g}(\cdot)$ er aktiveringsfunksjonen for tilstanden til minnecellen $\tilde{C}$ som generelt benytter $tahn$ funksjon (Yu Wang, 2017).

3. Forskningsdesign og metode

Kvantitativ metode har eksistert siden rundt 1250 E.K. og er drevet frem av behovet for å kvantifisere data. Metoden benyttes ofte når målet er å skape mening og ny kunnskap. Kvantitativ undersøkelse benytter data for å objektivt måle reelle forhold og begynner alltid med en problemstilling, litteraturgjennomgang og kvantitativ dataanalyse. Funnene kan være prediktive, forklarende og bekreftende (Williams, 2007).

Denne typen forskning benyttes i tilfeller hvor man skal analysere kvantitative data, altså data som har tallform eller mengdetermer (Grønmo, 2023) og skiller seg fra den kvalitative metoden hvor man operer med tekst i analysen (Johannessen et al., 2016). Metoden anvendes på problemer som kan kvantifiseres og benyttes ofte i tilfeller hvor målet er å kartlegge forekomst av fenomener som forklart av Creswell (2003, som referert av Williams, 2007).

I tillegg til å dreie seg om statistiske verktøy gjelder kvantitativ metode hvordan man tolker dataen i analysen. Dette er nødvendig for å forstå underliggende data (Johannessen et al., 2016), som i vår forsking er amerikanske statsobligasjoner og dens utvikling over tidsperioden vi forsker på. For å analysere rentepredikskjonsmodeller er det vesentlig å inkludere historisk og tallrik data. En kvantitativ tilnærming legger til grunn objektiv analyse og denne metoden var et naturlig valg for vår forskning.

3.1. Forskningstilnærming

I empirisk forskning er det utydelige linjer mellom teoretiske referanserammer og data. Forskning som ikke bygger på empiri kan enkelt bli spekulerende uten noe forankring i teori. Dette kan resultere i isolerende beskrivelser av henvendelser som vil ha begrenset verdi. I kvantitativ forsking er det viktig å integrere teorien og empiri (Johannessen et al., 2016).

Det finnes to ulike tilnærminger til vitenskapelig metode. Deduktiv tilnærming begynner med en generell teori før den testes ved å undersøke spesifikke tilfeller for å se om resultatet støtter teorien eller ikke. Deduktiv tilnærming bruker eksisterende teorier for å utvikle en hypotese som deretter kan bekreftes eller avkreftes gjennom empirisk data. Den andre tilnærmingen er induktiv som begynner med spesifikke observasjoner uten noe teoretisk utgangspunkt og brukers til å utvikle generelle mønstre, teorier og begreper. Den induktive tilnærmingen, motsatt av deduktiv tilnærming, går fra empiri til teori (Johannessen et al., 2016).

I vår forskning benytter vi deduktiv tilnærming for å redegjøre for våre forskingsspørsmål og belyse vår problemstilling.

3.2. Forskningsdesign

I tidlig fase av undersøkelser må man ta stilling til hva som skal undersøkes, og hvordan dette skal gjennomføres. Dette kalles forskningsdesign (Johannessen et al., 2016) og har som mål å gjøre forskningsspørsmål om til forskningsprosjekter (Saunders et al., 2009).

Forskningsdesign kan deles inn i tre klassifiseringer: deskriptiv, eksperimentell og kausal forskningsdesign som definert av Leedy & Ormrod (1990, som referert i Williams, 2007). Det deskriptive forskningsdesignet undersøker nåværende tilstand av en situasjon og identifiserer egenskaper ved et fenomen basert på observasjoner eller ved å undersøke korrelasjon mellom to eller flere fenomen (Williams, 2007). Det kausale forskningsdesignet undersøker kausale forhold om hvordan en variabel påvirker en eller flere andre variabler og kan testes ved å benytte statistisk eller økonometrisk metode (Oppewal, 2010).

Dersom vi hadde tatt utgangspunkt i en studie på enkelte av modellene og utført spesifikke endringer i modellbyggingen eller i datasettet for så å undersøke resultatet kunne vi benyttet en eksplorativ analyse, men ettersom vi har gjort vår egen oppbygning av modellene anså vi ikke dette som hensiktsmessig. Vi kunne også benyttet en kausal tilnærming for å forsøke å gi svar på årsakssammenhenger, eksempelvis mellom økning i rentenivå og pengepolitiske avgjørelser. I vår forskning har vi benyttet deskriptivt forskningsdesign ettersom vi ønsket å undersøke nåværende tilstand hvor vi ser prestasjonen til modellene på et gitt tidspunkt og gi en beskrivelse av prestasjon uten å gå inn på årsakssammenhenger. Det kan tenkes at et kausalt design ville gitt en god forklaring av forskingsspørsmål 2, men for å besvare forskningsspørsmål 1 mener vi en deskriptiv tilnærming passer best. Vi mener også denne tilnærmingen vil kunne gi et godt grunnlag for å besvare forskingsspørsmål 2 ved å presentere forskjellene mellom tradisjonelle metoder og moderne metoder.

3.3. Reliabilitet og validitet

Reliabilitet forklares som konsistens eller stabilitet i undersøkelser (Grønmo et al., 2024) og er et grunnleggende spørsmål i all forskning ved at det knytter seg til nøyaktigheten, innsamlingsprosessen og bearbeidelsen av data. Reliabiliteten kan testes på ulike måter, men en metode som ofte er

benyttet er at flere forskere undersøker samme problem. Desto flere som kommer frem til samme resultat, jo høyere reliabilitet har man i forskningen (Johannessen et al., 2016).

Validitet benyttes som kvalitetskriterium (Grønmo et al., 2024) og handler om forholdet mellom undersøkelsesobjektet og de konkrete dataene (Johannessen et al., 2016). Validiteten kan forbedres gjennom nøye utvalg og ved å sørge for korrekt behandling av dataene. Validitet i kvantitativ forskning vil aldri være 100% ettersom standardfeil må anerkjennes (Cohen et al., 2007).

3.4. Prosessen med å samle inn data

Vi vurderte å se på det Norske markedet, nærmere bestemt NIBOR, men opplevde datainnsamling og tilgjengelighet av data som utfordrende. Derimot ble vi oppmerksom på et eksisterende datasett utarbeidet av (Liu & Wu, 2021) gjennom artikkelen Predicting interest rate distributions using PCA & quantile regression (Westgaard et al., 2022). Dette datasettet inneholder renteinformasjon fra det amerikanske statsobligasjons-markedet, et marked som har stor påvirkningskraft på verdensøkonomien.

Datasettet gjør det mulig å forske på statsobligasjonsrenter med både kort og lang løpetid ettersom de har benyttet ikke-parametrisk metode kalt kernel-smoothing som ikke forkaster statskasseveksler (Bianchi et al., 2021). Sett opp mot Gurkaynak, Refet, Sack og Wright (2007, som referert i Bianchi et al., 2021) sin forskning fra som ikke benytter denne metoden (Bianchi et al., 2021) ble det naturlig for oss å forske videre med datasettet spesielt med tanke på at vi forsker både på kortsiktig og langsiktig løpetid. Vi legger merke til at renten med kortsiktig løpetid inneholder viktig informasjon som påvirker utviklingen til de mer langsiktige løpetidene (Liu & Wu, 2021).

Fullstendig datasett inneholder rentedata for alle løpetider for amerikanske statsobligasjoner, fra 1mnd til 30år. Dataen strekker seg fra 1985 til 2022. Utover dette er ikke datasettet spesielt kompleks, det inneholder kun to variabler: dato og løpetid, altså tidsperioden fra statsobligasjonen utstedes til den skal være tilbakebetalt i sin helhet. Datasettet inneholder 444 rader som viser rentesatsen i starten av hver måned. I tillegg har datasettet 361 kolonner, hvor løpetidene vises i stigende rekkefølge. Datasettet har observasjoner for alle løpetider, bortsett fra 1985 som kun har observasjoner for 20 års løpetider. På grunn av manglende observasjoner i 1985 har vi gått videre med data fra 01.1986.

Datasettet inneholder omfattende historikk av renteverdier for amerikanske statsobligasjoner og kan gi et detaljert bilde over hvordan rentene har utviklet seg over tid. Vi må også anerkjenne at økonomiske marked er komplekse og kan uforutsett bli påvirket av globale hendelser som kan føre til markante avvik fra historiske mønstre.

3.5. Dataprøve

En dataprøve er et utdrag av en populasjon hvor man observerer en liten del av populasjonen for å kunne gi estimater om hele populasjonen. Man skiller ofte på observasjonsstudier og dataprøver. For observasjonsstudier har man liten eller ingen kontroll over hvordan observasjonene ble gjort, mens ved dataprøve kan man målrettet velge ut deler av populasjonen for videre studier (Thompson, 2012). Vårt datasett inneholder rentedata fra 1 måned til 30 år, og dekker perioden 1961-2022 (Wu & Liu, 2021). I vår forskning er dataen begrenset til bestemte løpetider: 3 måneder, 6 måneder, 1 år, 5 år og 10 år. Ved å benytte disse løpetidene mener vi å ha inkludert et representativt utvalg av populasjonen ved å inkludere korte, mellomlange og lange løpetider i vår analyse. Vi antar at dette vil gi et helhetlig bilde av hvordan ulike løpetider påvirkes av økonomiske hendelser og pengepolitiske tiltak over tid.

Beslutningen om å fokusere på et enkelt datasett i vår oppgave har flere fordeler, spesielt i sammenligning av ulike modellers prestasjon. For det første tillater det dypere analyse av modellens ytelse, hvor vi kan utforske detaljert hvordan forskjellige modeller oppfører seg under samme forhold. Dette gir en klarere forståelse av hver modells styrker og svakheter. Effektiv ressursbruk er en annen fordel, ettersom begrensning til ett datasett gjør analysen mer håndterbar. I tillegg mener vi et godt datasett fungerer som en benchmark, noe som er svært nyttig for å etablere en standard for modellenes ytelse. Til slutt, ved å fokusere på ett datasett, kan vi utforske flere modeller med høyere kompleksitet. Selv om dette kan gi et noe ensidig bilde av resultatene, mener vi at fordelene ved å sammenligne flere modeller på denne måten overgår fordelene ved å analysere færre modeller med flere datasett.

3.6. Datahåndtering

Et essensielt steg ved dataprosessering er datahåndtering. Rådata kan være ufullstendig, uoversiktlig eller inkonsistent og når dette er tilfellet kan det være vanskelig å analysere dataen. Datahåndtering inkluderer duplikatfjerning, håndtering av manglende data, justere feil i dataen, endre dataformatering etc. På denne måten sørger man for nøyaktighet i dataen og at den er klar for analyse. Uten å gjøre disse trinnene kan en dataanalyse være upålitelig og villedende som igjen kan føre til

feilaktige konklusjoner og beslutninger. Det finnes tre steg innen datahåndtering: Steg 1 er å rense dataen ved å fjerne manglende verdier, håndtere uteliggere og løse inkonsistens i dataen. Steg 2 er å transformere dataen ved å normalisere, aggregere og/eller konvertere format på dataen. Steg 3 er å forberede dataen for analyse ved å eksempelvis velge riktige variabler (Magnimind, 2023).

For å rense datasettet har vi fjernet manglende verdier ved å utelate 1985 fra vår data i tillegg til å standardisere formatet på dataen ved å først gjøre om komma til punktum for å gjøre dataen lesbar i R. Videre har vi konvertert til numerisk verdi for rentedataen før vi har hentet ut aktuelle løpetider til en tabell. Steg 2 har blitt gjennomført for alle modellene, men med litt ulik tilnærming. Vi ønsket å utføre datatransformasjon best tilpasset de enkelte modellene. Vi har differensiert dataen da denne ikke var stasjonær. Random Forest og LSTM kunne håndtert dataen uten differensiering, men vi ønsket å gi et mest mulig generelt grunnlag for modellene for å kunne sammenligne på best mulig måte. For de moderne maskinlæringsmodellene var det også nødvendig å normalisere dataen.

4. Resultat, diskusjon og konklusjon

I dette kapittelet vil vi presentere resultatene fra de fire modellene vi har valgt å forske på. Vi har utført en analyse for de fire modellene for hver løpetid vi har valgt å fokusere på. Alle modellene er trent og testet på differensiert data for å modellere på stasjonær data. Random Forest og LSTM kunne vært modellert på ikke stasjonær data da disse identifiserer mer komplekse mønstre og sammenhenger enn de tradisjonelle prediksjonsmodellene som forklart i kapittel 1. Vi har tilstrebet å generalisere modellbyggingen mest mulig for å kunne gi et best mulig sammenligningsgrunnlag og har derfor differensiert all data som benyttes i modellene. Ved kryssvalidering har vi gått for den mest hensiktsmessige metoden for hver modell, for å sikre at modellens ytelse ikke er tilfeldig og for å kunne evaluere modellenes evne til å generalisere ny data.

ARIMA-modellene har nøyaktighetsmålinger presentert både med og uten kryssvalidering grunnet modellens krav til data og struktur ved valg av AR, I og MA. Disse baseres basert på egenskapene til tidsserien. Vi har derfor først identifisert den beste sammensetningen av disse verdiene før vi har undersøkt modellens prediktive ytelse ved å utføre en walk forward kryssvalidering.

Random Forest og Ridge modellene er bygget ved hjelp av Caret-pakken i R som forenkler modelltrening for blant annet regresjonsproblemer ved å automatisere flere skritt i modellbyggingen. Pakken gir mulighet for å velge optimale modeller basert på ytelse ved å gjennomføre kryssvalidering og tilpasning av treningsprosess med forskjellige innstillinger og hyperparametere (*A Short Introduction to the caret Package*, U.Å.) ARIMA og LSTM er ikke omfattet av denne pakken (Kuhn, u.å.) så disse har en litt mer omfattende oppbygning. For LSTM er det foretatt et grid search for å ivareta prosessen med å velge den optimale modellen basert på ytelse. For ARIMA-modellene har vi inkludert *auto.arima*-funksjonen som en aktuell modell i hver liste, i tillegg til modellene vi selv har identifisert.

For Random Forest har vi ikke gjort endringer i standard antall trær i Caret-pakken. Standarden på 500 trær handler om å sette et høyt beregningsmessig håndterbart antall trær (Bogdanovist, 2013). Ytelsen til modellene vil i de fleste datasett ha nådd en verdi som ikke kan forbedres mye ved 250 trær, utover 500 trær finner man ingen betydelig ytelsesgevinst (Probst & Boulesteix, 2017).

Målet er å avdekke hvilke modeller som best tilpasser seg faktisk data og som igjen klarer å gi en god prediksjon av fremtidig rentesats. Vi har splittet dataen i 70% in-sample-data og 30% out-of-sample-data og vil vil sammenligne in- og out-of-sample resultater for 3 måneders løpetid. I maskinlæring

deles data typisk tilfeldig mellom trenings- og testsett da det skjelden er avhengigheter mellom observasjonene. Dette er ikke tilfelle med tidsseriedata, hvor det er viktig å ta hensyn til tidens rekkefølge. For tidsserier brukes vanligvis de siste dataene i datasettet til testing og de tidligere dataene til trening (Radečić, 2022). In-sample data blir brukt i evalueringen av modellparametre, mens out-of-sample data brukes til å evaluere modellens treffsikkerhet. Ettersom out-of-sample dataen ikke benyttes i predikeringen vil denne dataen gi en god indikasjon på hvor treffsikker modellen er mot ny data (Hyndman & Athanasopoulos, 2018).

Vi har valgt å presentere nøyaktighetsmålinger for alle løpetider, men har begrenset grafer til 3 måneder løpetid ettersom den vanlige oppfatningen er at lange løpetider ofte tilpasser seg korte løpetider som forklart i kapittel 2.2.1. Vi har også lagt merke til at mønsteret i dataen ikke ser ut til å variere i stor grad og mener derfor det er tilstrekkelig å presentere grafene for kun 3 måneder løpetid.

Vi har benyttet kodespråket R for å modellere alle prediksjonsmodellene. R er et viktig verktøy for dataanalyse og statistisk modellering. Styrkene til verktøyet ligger i dens omfattende samling av pakker og biblioteker som er spesifikt designet for å støtte ulike typer dataanalyse. Pakkene dekker et bredt spekter av maskinlæringsalgoritmer og har i stor grad støtte for de modellene vi ønsker å modellere. R har også pakker som TensorFlow og Keras som gir tilgang til avanserte dype læringsalgoritmer og beregningsmotorer. Samlet sett gir R fleksibilitet, TensorFlow gir økt støtte for avanserte dype læringsmodeller og er et kraftig verktøy for å utvikle og implementere maskinlæringsmodeller. Denne synergien gir muligheten til å utforske dataen og bygge sofistikerte modeller (Galijasevic, 2020). På denne måten var også LSTM representert i verktøyet og det ble mulig for oss å benytte dette i vår analyse. Det var også en stor fordel at vi kjente til R fra tidligere, da dette har vært pensum på studiet.

Alle modellene er modellert som univariate tidsseriemodeller. Slike modeller predikerer fremtidige verdier for en enkelt variabel i stedet for å se på forholdet mellom ulike variabler (Henson, 2023). Disse modellene tilbyr en enkel måte å forutsi fremtidig utvikling av en variabel og det har vist seg at univariate modeller som bare tar hensyn til informasjon gitt av en enkelt tidsserie, ofte presterer bedre enn prognoser som er basert på store økonometriske modeller(Fuller, 2009).

Vi ønsket å predikere løpetidene for seg selv da disse modellene er enklere å forstå og tolke og på denne måten etablere et referansepunkt før man eventuelt utvider til mer komplekse multivariate modeller.

Dette kapittelet vil gi en detaljert oversikt over resultatene fra disse modellene, samt en grundig analyse og tolkning av funnene.

4.1. Modellevaluering

For ikke-lineære modeller benyttes informasjonskriterier som AIC for å evaluere modellenes kvalitet relativt til hverandre. Ved evaluering av prediksjonene er man opptatt av å sjekke at prediksjonene har minst mulig avvik fra faktiske verdier. For å finne prediksjonsmodeller med god ytelse estimerer man modellparametre ved å anvende en subset av tilgjengelige data. De estimerte parameterverdiene brukes videre for å generere prognoser for datapunkter som var ekskludert i opprinnelig estimeringsprosess. Ulike evalueringskriterier kan benyttes når man sammenligner predikert verdi opp mot faktiske verdier (Bårsaune, 2017).

Root Mean Square Error (RMSE), Mean Absolute Error (MAE) og Mean Absolute Percentage Error (MAPE) er ofte brukte nøyaktighetsmålinger for prediksjonsmodeller på tidsseriedata. RMSE kvantifiserer gjennomsnittlig feil mellom predikert og faktisk verdi. På denne måten vises størrelsen på feilene. MAE kvantifiserer gjennomsnittlig absolutt feil mellom predikert og faktisk verdi, mens MAPE måler nøyaktigheten på prediksjonene i prosent av faktisk verdi ved å kvantifisere gjennomsnittlig prosentvise feil mellom predikert og faktisk verdi (Chaudhuri, 2023).

Vi har benyttet følgende evalueringskriterier i oppgaven for å vurdere modellenes prediksjonskraft hvor A_t er faktisk verdi, F_t viser predikert verdi, og n står for antall observasjoner (Mehrotra, 2012):

$$Root\ mean\ square\ error: RMSE = \sqrt{\frac{\sum_{t=1}^{n}(A_t - F_t)^2}{n}} \tag{7}$$

$$Mean\ Absolure\ error: MAE = \frac{\sum_{t=1}^{n}|A_t - F_t|}{n} \tag{8}$$

$$Mean\ absolute\ percentage\ error: MAPE = \frac{\sum_{t=1}^{n}\frac{|A_t - F_t|}{A_t} x100}{n} \tag{9}$$

MAPE er den vanligste evalueringsmetrikken og beregnes ved å benytte gjennomsnitt av absolutte verdier av prosentfeil. Det er viktig å merke seg at en viktig bakdel med prosentfeil er at de blir uendelige eller ikke definerbare dersom faktisk verdi er lik eller tilnærmet lik null, og de kan dermed i større grad vektlegge negative feil enn positive (Hyndman & Athanasopoulos, 2018). Grunnen til at

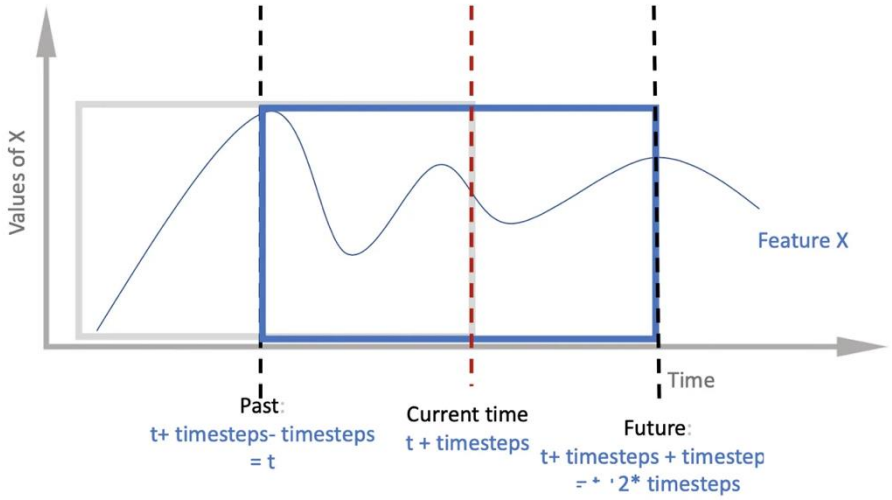
MAPE er den mest populære nøyaktighetsmålinger er at den er intuitiv og enkel å tolke (Amar et al., 2019).

Vi har på bakgrunn av dette presentert MAPE, RMSE og MAE for alle modeller, men har valgt å fokusere på den mest brukte nøyaktighetsmålingen MAPE. Dersom vi får ulike resultater mellom nøyaktighetsmålingene vil vi vektlegge MAPE høyere enn de to andre metrikkene. Basert på evalueringsmetrikkene har vi valgt modellenes ordre og hyperparametere.

4.1.1. Valg av hyperparametere

Tidssteg

Tidssteg er et viktig parameter i LSTM modellbygging. Som vist i Figur 3 er det et rullerende vindu som er delt inn i to like deler. Midtpunktet i vinduet representerer nåtid (t), høyre side representerer $t + tidssteg$, og venstre side representerer t-tidssteg. Det vil si at dersom man bruker 10 tidssteg vil modellen lære fra de 10 siste tidsstegene, og forsøke å predikere 10 tidssteg fram i tid. Vinduet forskyves så et steg fram (et steg = et tidssteg. Dersom tidssteg = 10 forskyves vinduet 10 verdier fram), og prosessen gjentas (Caner, 2020a).



Figur 3 Tidssteg i LSTM model (Caner, 2020a).

Batch

For LSTM modeller så er batch størrelser avgjørende, da de brukes for å håndtere store datasett som kan inneholde millioner av eksempler. For at modellen skal kunne identifisere felles mønstre og funksjoner blant dataen som trenes i modellen er det nødt til å trenes i batch størrelser. En optimal

batch størrelse i de fleste tilfeller er 64. Det vil også være tilfeller der man kan bruker verdier som eksempelvis 32 og 128. Batch størrelsen må kunne deles med 8. Det er viktig å merke seg at batch størrelsen må velges basert på ytelsesobservasjoner av modellen (Caner, 2020b).

Epoker

Epoker refererer til hvor mange ganger modellen ser hele datasettet. I teorien kan antall epoker settes til hvilken som helst heltallig verdi mellom 1 og ∞, men det optimale antallet er i bruddpunktet hvor valideringsnøyaktigheten begynner å synke selv om treningsnøyaktigheten stiger. Dette gjøres for å senke risiko for overtilpasning. Man kan legge til metoden for tidlig stopp. Her spesifiseres først et stort antall trenings-epoker, før man stopper treningen når modellens ytelse slutter å forbedre seg ved å benytte en forhåndsinnstilt terskel på testdatasettet (Chowdhury, 2021).

Optimaliserer

Adaptive Moment Estimation (Adam) skiller seg ut som en svært effektiv optimaliseringsalgoritme og defineres som den viktigste optimalisereren (Brownlee, 2021). Optimalisereren er designet til å justere læringsratene for hvert hyperparameter. Adam skal tilpasse seg dataens egenskaper noe den gjør ved å opprettholde individuelle læringshastigheter for hvert parameter. Det som gjør Adam unik er at hver parmeteroppdatering skaleres individuelt. Ved å tilpasse justeringen av parameter fører det til mer effektiv optimalisering, spesielt i de tilfellene hvor LSTM-modellen er kompleks med mange parametere (Leo, 2024). Nesterov-accelerated Adaptive Moment Estimation (Nadam) bygger på Adam, og inkluderer Nesterov Momentum. Denne kan resultere i bedre prestasjon i optimaliseringsalgoritmen (Brownlee, 2021).

Læringsrate

Læringsrate spiller en viktig rolle for om prosessen kan konvergere og hvor raskt dette kan skje. De fleste optimaliseringsmetoder benytter en gradvis avtagende læringshastighet for å sikre konvergering. Dette kan være en tidkrevende prosess, spesielt i starten av treningen. Derfor er det viktig med en læringsratestrategi som automatisk kan tilpasses. Det vil også kunne redusere tap og være nyttig for komplekse LSTM modeller (Yu et al., 2020).

4.1.2. K-fold og Walk-Forward validering

K-fold kryssvalidering deler observasjonene tilfeldig inn i k grupper på omtrentlig like stor størrelse. Valideringssett er første fold og modellen tilpasses de resterende $k-1$ foldene før MSE blir deretter beregnet på observasjonene hentet ut fra første fold. Denne prosedyren blir gjentatt k ganger med nye tilfeldige folds, hvor valideringssettet består av ulike grupper. Dette resulterer i k estimater av testfeilen. K-folds i kryssvalideringsestimatet beregnes ved å ta et gjennomsnitt av disse verdiene og formelen kan skrive som (James et al., 2023):

$$CV_{(k)} = \frac{1}{K}\sum_{i=1}^{k} MSE_i \tag{10}$$

Det har blitt empirisk bevist at K-fold kryssvalidering bør utføres med $k = 5$ eller $k = 10$ ettersom disser verdiene gir testfeilrate som hverken lider av svært høy bias eller svært høy varians (James et al., 2023).

Walk forward validering brukes for å teste og validere prediktive modeller, spesielt i analyser av tidsseriedata hvor modellens stabilitet over tid er viktig. Valideringsmetoden sikrer at modellen blir trent og validert på en måte hvor den tar hensyn til tidsserierekkefølgen av dataene. Fordeler med å bruke walk forward validering er at den testes på flere uavhengige perioder som bidrar til å vurdere modellens styrke over tid og tar hensyn til at markedet eller dataen som studeres kan endre seg over tid. Dermed må modellen kunne tilpasse seg visse endringer (Maitra, 2021).

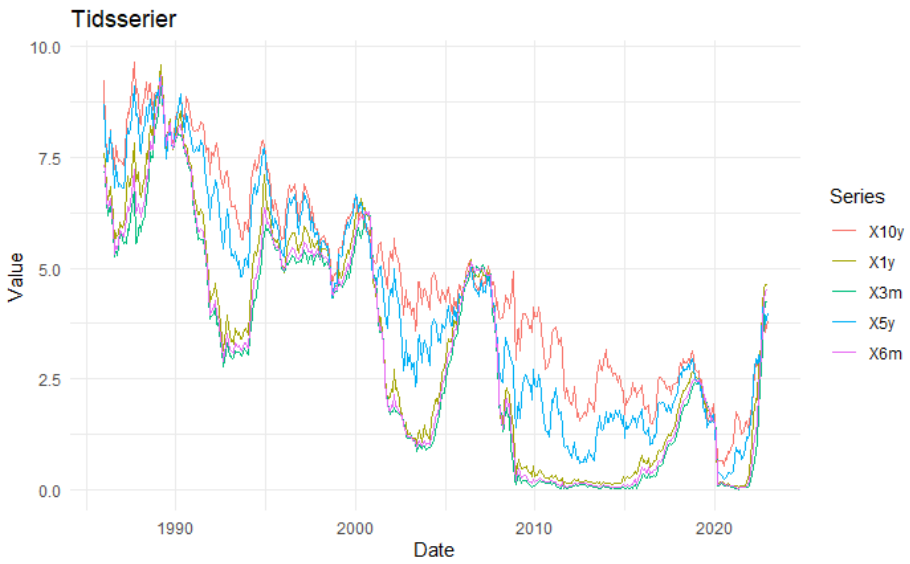
4.1.3. Lagget data

Lags, også kalt forsinkelser, er forskjøvede verdier justert med den faktiske tidsserien. Grunnet høy forekomst av autokorrelasjon i tidsseriedata er lags av denne typen data svært nyttig (Articles, 2017). Når man velger forsinkelser vil ACF-plot gi en god indikasjon på hvor mange lags som er hensiktsmessig å benytte (Holbrook & Cook, u.å.).

4.2. Deskriptiv statistikk

Deskriptiv statistikk brukes ofte til å gi en beskrivelse av variabler. Denne type statistikk innebærer å analysere en variabel om gangen eller det vi kaller for enveisanalyse. Det er vanlig å gjennomføre en deskriptiv statistikk før man eventuelt begynner på noen form for dataanalyse (P. Patel, 2009).

For å kunne definere hvilke endringer som muligvis må utføres i forbehandlingsfasen ser vi på deskriptive statistikker for rentedataene. Vi vil se på trender og metrikker for dataen for å gjøre oss kjent med dataen før modellbyggingen. Vi har følgende oversikt over de ulike løpetidene for de amerikanske statsobligasjonene:



Figur 4 Månedlige amerikanske statsobligasjonsrenter fra 1986-2022.

I Figur 4 ser vi at rentene har en synkende trend, med flere svingninger over tid. Svingningene kan representere sykliske mønster, og vi ser at topp- og bunnpunkter i seriene korresponderer. Dette indikerer sterke korrelasjoner mellom tidsseriene. Vi observerer mindre svingninger i de lengre løpetidene, enn for de korte løpetidene, noe som kan forklares med at lengre løpetider er mindre utsatt for økonomiske hendelser enn de kortere løpetidene. Mot slutten av dataperioden ser vi en markant stigning. En stor del av stigningen mener vi kan forklares med økt inflasjon etter koronaperiodens nedgang.

Amerikanske statsobligasjoner med løpetider på tre år eller mindre betegnes som kortsiktige obligasjoner. Disse har ofte lavere rente enn de langsiktige løpetidene, og gir mindre risiko for rentesvingninger. Selskaper og finansinstitusjoner bruker ofte de korte løpetidene som en del av sin likviditetsstyring da de som regel har lav risiko og disse betraktes som viktige kortsiktige investeringer. Ettersom det har mindre risiko for rentesvingninger så vil de korte løpetidene gjøre dem mindre volatile enn de langsiktige løpetidene, så ved renteøkning vil sannsynligvis de korte løpetidene bli mindre påvirket enn de lange løpetidene (Greenwood et al., 2015).

Mellomlange løpetider har løpetid på 4-10 år. Avkastningen på 5 til 10 års løpetid kan tydelig vise Den føderale reservebankens (Fed) sin strategi for å fokusere på obligasjoner med lang løpetid samtidig med ønsket om å styre retningen for avkastningen på korte løpetider. I tillegg fungerer den som benchmark for avkastningen på alle obligasjoner. 5 og 10 års løpetider er viktige da avkastningene til disse statsobligasjonene kan spille en hovedrolle i bestemmelse av form og endringer i avkastningskurven (Chen, 2023). De kortsiktige rentene blir i stor grad påvirket av beslutningene til Fed som har gitt utrykk for at de vil holde styringsrenten mot null i flere år. De langsiktige obligasjonene har dermed større frihet, noe som gjør at investorer får utsikter om økonomisk vekst og inflasjon. Den 30 årige løpetiden blir veldig usikker å investere i og det er naturlig å sette søkelyset på 10 årige statsobligasjoner istedenfor. Dette har da resultert i at den 10 årige statsobligasjonen er den foretrukne og betegnes som verdens viktigste kostnadsreferanse ved risikofri lånekostnad globalt. Det understrekes også at den 10 årige løpetiden ikke kun predikerer fremtidige økonomisk vekst og inflasjon, men benyttes også som en essensiell faktor prising av diverse finansielle produkter verden over (McCormick & Regan, 2021).

Lange løpetider er de som overstiger 10 år og har ofte høyere avkastning enn de med kortere løpetider. På denne måten får investorer en kompensasjon for den forhøyede risikoen for svingninger i rentene. Det finnes ulike typer obligasjoner, og amerikanske statsobligasjoner, som vi undersøker i vår oppgave, har løpetider opp til 30 år. For investorer kan det være en fordel å gå for lange løpetider ettersom disse vanligvis gir høyere avkastning, mindre portfolio-håndtering og muligheter for kapitalgevinst dersom rentene synker, da man ofte kan oppleve at prisen øker betydelig. På den andre siden kan det være bakdeler med lange løpetider som er mer utsatt for svingninger i pris som et resultat av endringer i rentenivå. Dette kan igjen føre til kaptitaltap. Ved å ha obligasjonene bundet over lengre perioder senkes også investorers likviditet (Greenwood et al., 2015).

Ved å inkludere 3 måneder, 6 måneder, 1 år, 5 år og 10 år mener vi å være innom de viktigste løpetidene for amerikanske statsobligasjoner. De korte løpetidene reflekterer kortsiktige svingninger i markedet, mens de lengre løpetidene tillater analyse av langsiktige økonomiske trender. Ved å inkludere 10 års løpetid har vi inkludert den viktigste benchmarken for andre renter, og ved å teste modellene over ulike løpetider får vi et godt inntrykk over modellenes evne til å tilpasse seg dataene, samt gi gode prediksjoner for fremtidige rentenivåer. Ved å fokusere på disse løpetidene forsøker vi å bidra til forskningen ved å sikre at funnene har praktisk relevans og er anvendbare i finansmarkedet.

Vi har videre sett på deskriptive målinger for de ulike løpetidene.

Tabell 1 Deskriptiv statistikk for månedlige amerikanske statsobligasjonsrenter.

	Gjennomsnitt	**Standardavvik**	**Maks**	**Min**	**Median**	**Kurtose**	**Skjevhet**
3mnd	3.04	2.53	9.09	0.01	2.94	-1.14	0.32
6mnd	3.16	2.56	9.33	0.03	3.10	-1.16	0.30
1år	3.35	2.60	9.59	0.06	3.35	-1.18	0.28
5år	4.19	2.47	9.34	0.23	4.15	-1.17	0.23
10år	4.79	2.30	9.64	0.53	4.59	-1.06	0.17

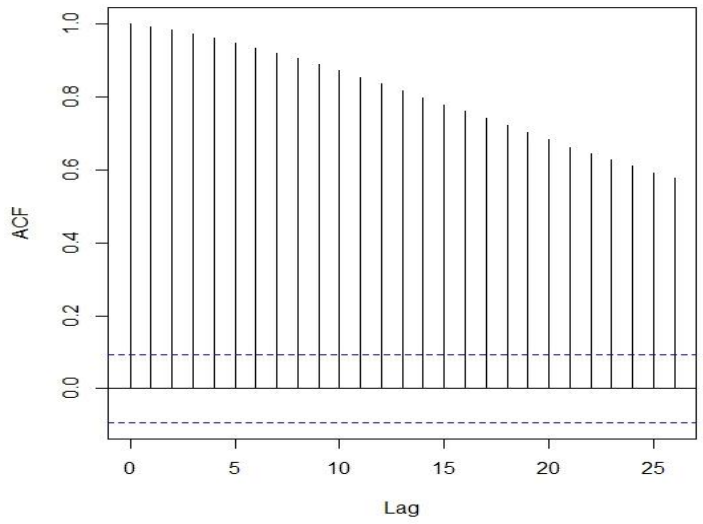
Tabell 1 hjelper oss med å danne oss et bilde over hvordan løpetidene presterer. Vi ser at gjennomsnittlig rentenivå gradvis stiger fra 3.04 for 3 måneders løpetid, til 4.79 for 10 års løpetid, noe som indikerer høyere rentenivå for lenger løpetid. Dette er forventet da lengre løpetider justerer for økning i risiko og inflasjonsforventninger. Median følger gjennomsnittet tett med verdier rett under gjennomsnittlige verdier og indikerer relativt symmetrisk fordeling av dataen. Vi legger også merke til at minimums- og maksimums-rentenivåene ligger tett opp mot hverandre, med noe lavere minimumsverdier for kortere løpetider, og noe høyere maksimumsverdier for lengre løpetider.

Standardavviket måler dataens spredning fra gjennomsnittet. Når standardavviket gir lav verdi betyr det at dataene er konsentrert rundt gjennomsnittet, mens et høyt standardavvik betyr større spredning. Standardavvik nær null indikerer at datapunktene ligger tett inntil gjennomsnittet mens høyt standardavvik indikerer at dataene ligger over gjennomsnittet og motsatt (Macwan, 2021). For våre data ser vi at standardavviket øker opp til 1 års løpetid før verdien avtar mot 10 åringen igjen, og de lange løpetidene har lavere standardavvik enn de korte løpetidene. Dette kan eksempelvis være grunnet markedsforhold, økonomiske eller politiske hendelser i økonomien. Vi forventer også at kortere løpetider blir påvirket av kortsiktige rentebevegelser i større grad enn lengre løpetider hvor rentesvingningene skjer over et lengre tidsrom som jevnes ut.

Kurtose måler graden av skarphet i en fordeling. En kurtoseverdi større enn 3 kalles leptokurtiske, mens kurtoseverdier mindre enn 3 kalles platykurtiske. En kurtose på 3 indikerer perfekt normalfordeling (Yenigün, 2023). Vi ser at alle våre løpetider har platykurtisk kurtose. En slik kurtose viser lav forekomst av ekstremverdier sett opp mot en normalfordeling. I disse tilfellene vil man finne færre datapunkter langs halen og kurtosis vil være under 3. Disse tilfellene har en tynnere hale og en

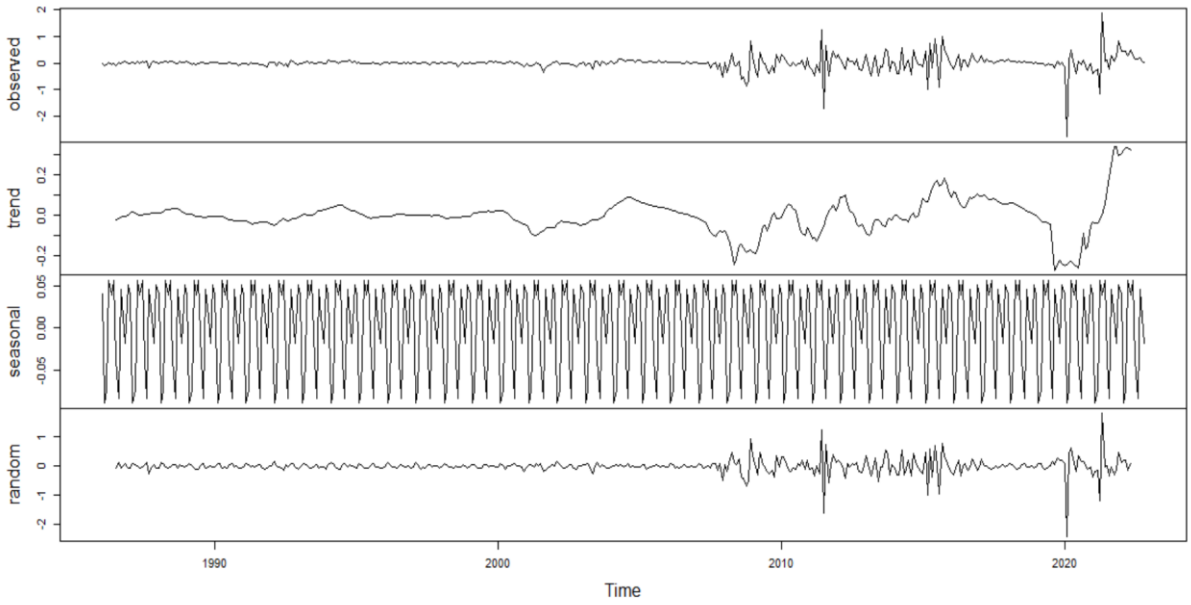
kortere distribusjon sett opp mot normalfordeling. Man ser kurtosis i sammenheng med standardavvik, hvor en lavere verdi for standardavvik vil gi en brattere distribusjon, og omvendt (Singh, 2022). For de amerikanske statsobligasjonene indikerer dette at løpetidene generelt har lite ekstreme rentebevegelser da kurtosisverdien er negativ for alle løpetider.

Skjevheten viser graden av asymmetri i dataen. Dersom dataen er normalfordelt vil dette gi null skjevhet, positiv skjevhet indikerer skift mot høyre, mens negativ skjevhet indikerer skift mot venstre i kurven. I en normalfordeling vil gjennomsnitt og median være like (Yenigün, 2023). Vi ser at statistikken viser en svak positiv skjevhet og kurven vil derfor være moderat skiftet mot høyre. Dette indikerer at flere verdier vil være høyere enn gjennomsnittet sammenlignet med hva de ville vært i en helt symmetrisk distribusjon. Vi antar derfor at det har vært flere tilfeller av uvanlig høy rente enn uvanlig lav rente for alle løpetidene.



Figur 5 Korrellogram for 3mnd løpetid.

Figur 5 er en grafisk representasjon av ACF funksjonen for tidsseriedata. ACF måler korrelasjon mellom tidsserien og seg selv gjennom ulike lags, og et korrellogram kan brukes blant annet til å undersøke tilfeldigheter eller identifisere orden på en AR-modell. Dersom autokorrelasjonen er nær null for alle lags kan man si at tidsserien er tilfeldig mens ved høye verdier indikerer det at tidsserien ikke er tilfeldig og har seriekorrelasjon. Korrellogrammet kan benyttes til å identifisere AR orden ved å se på hvilke lag som har en autokorrelasjon signifikant over null (Jain, 2023). Vi ser at vårt korrellogram viser signifikante verdier for alle 25 lags noe som indikerer høy positiv autokorrelasjon i dataene. Dette antyder at tidligere verdier har sterk påvirkning på fremtidige verdier.



Figur 6 dekomponering av 3 måneder løpetid.

Trend er en generell bevegelse i dataen og gir en oversikt over antatt fremtidig retning basert på historisk data. Når man undersøker trend-grafen ser man etter repetitive oppføringer i trendanalysen. Når man sjekker om trenden er oppad- eller nedadgående ser man etter om trendlinjen treffer 2 eller flere høyere lavpunkter (Johnson, 2020). Vi ser en nedadgående trend for våre data.

Sesongvariasjon viser en gjentagelse av dataen over en bestemt tidsperiode. Eksempelvis kan man ofte i tidsseriedata merke store forskjeller i perioder med fellesferie, midt på sommeren eller ved juletider. Sesongvariasjon er en av de viktigste egenskapene ved tidsserieanalyse, og det måles vanligvis ved autokorrelasjon etter trenden er trukket ifra dataen (Anish, 2020). Vår data viser mønster som gjentas med jevne mellomrom. Dette kan skyldes kvartalsvis økonomiske sykluser, regelmessige hendelser i politikken eller lignende hendelser.

Tabell 2 PP, ADF & KPSS – verdier for alle løpetider.

	PP P-verdier	**ADF P-verdier**	**KPSS P-verdier**
3mnd	0.8649	0.2079	NA
6mnd	0.8964	0.2275	NA
1år	0.8838	0.3131	NA
5år	0.4307	0.5093	NA
10år	0.0352	0.0774	0.01

Vi har benyttet ADF og PP test for å sjekke for stasjonaritet i dataen. I tillegg har vi testet med KPSS-test for 10 års løpetid, da denne ga ulikt resultat mellom PP og ADF test. Vi ser i Tabell 2 at verken PP eller ADF gir signifikant p-verdi (sett bort fra PP-test tom gir signifikant P-verdi for 10 års løpetid) og vi forkaster derfor ikke nullhypotesen om at data ikke er stasjonær. KPSS-test for 10 års løpetid gir signifikant P-verdi og vi forkaster nullhypotesen om at dataen er stasjonær. Vi går derfor videre med de opprinnelige dataene på endringsform for alle løpetider.

Tabell 3 Deskriptiv statistikk for logaritmen av dataen.

	Gjennomsnitt	**Standardavvik**	**Maks**	**Min**	**Median**	**Kurtose**	**Skjevhet**
3mnd	0.26	1.74	2.21	-4.73	1.08	-0.68	-0.87
6mnd	0.42	1.56	2.23	-3.59	1.13	-0.75	-0.83
1år	0.61	1.36	2.26	-2.75	1.21	-0.74	-0.80
5år	1.19	0.78	2.23	-1.46	1.42	0.18	-0.86
10år	1.42	0.58	2.27	-0.63	1.52	0.28	-0.81

Tabell 4 PP, ADF & KPSS – verdier for alle løpetider på logaritmedata.

	PP P-verdier	**ADF P-verdier**	**KPSS P-verdier**
3mnd	0.7118	0.3863	NA
6mnd	0.7720	0.3395	NA
1år	0.7592	0.3761	NA
5år	0.2469	0.0955	NA
10år	0.0536	0.0136	0.01

Logaritmetransformasjon reduserer uteliggeres påvirkning på dataen, i tillegg til å redusere skjevhet til tilnærmet normalfordeling. Transformasjonen sørger også for lineære forhold mellom variablene i tillegg til å stabilisere varians og forenkle komplekse forhold (De la Calle, 2023). I Tabell 3 ser vi at skjevhet på logaritmen av dataen er, som forventet, betydelig redusert sammenlignet med tabell Tabell 1. Median har blitt redusert og spekteret av median mellom variablene har blitt redusert. PP og ADF test på logaritmedataen har, som vi ser i Tabell 4, samme resultat som i Tabell 2 hvor vi forkaster

nullhypotesen og vi ser at dataen må differensieres. Standardavviket på logaritmeform er lavere enn for original data, noe som er forventet.

Tabell 5 Deskriptiv statistikk for differensiert data.

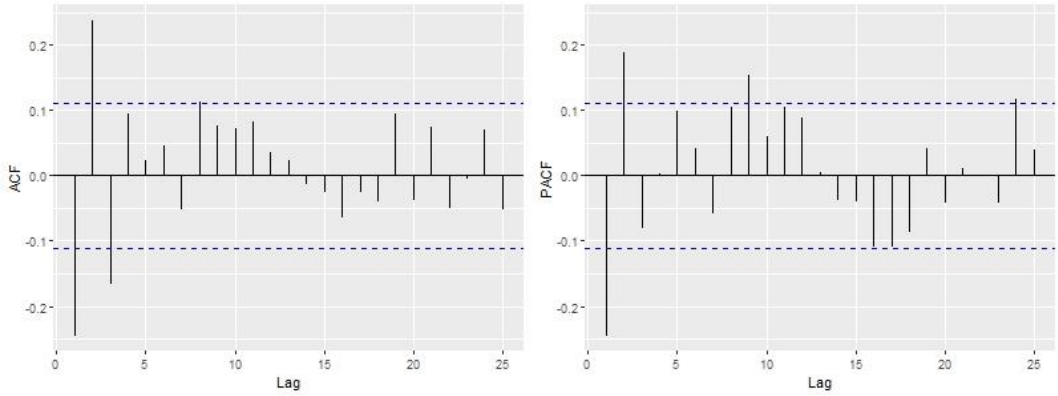
	Gjennomsnitt	Standardavvik	Maks	Min	Median	Kurtose	Skjevhet
3mnd	0.00	0.28	1.86	-2.75	0.00	29.44	-1.87
6mnd	0.00	0.21	0.88	-2.39	0.00	39.79	-3.37
1år	0.00	0.18	0.66	-1.96	0.00	37.83	-3.25
5år	0.00	0.12	0.53	-0.85	-0.01	7.85	-0.53
10år	0.00	0.08	0.29	-0.50	0.00	4.80	-0.39

Tabell 6 PP & ADF verdier for alle løpetider på differensiert data.

	PP P-verdier	ADF P-verdier
3mnd	0.01	0.01
6mnd	0.01	0.01
1år	0.01	0.01
5år	0.01	0.01
10år	0.01	0.01

Etter èn differensiering av dataen ser vi i Tabell 6 at vi oppnår vi stasjonaritet i dataen for alle løpetider og dataen kan nå benyttes i modellering. Ettersom vi nå har signifikant p-verdi for alle løpetider var det ikke behov for å utføre KPSS-test på differensiert data. Vi ser i Tabell 5 at kurtose har ekstremt høye verdier for 3 måneder, 6 måneder og 1 års løpetid, mens 5 års og 10 års løpetid har mindre topper. Dette kan være grunnet økonomiske sjokk i original data, eksempelvis som følge av korona i 2020. I Figur 4 ser vi at 5 års og 10 års løpetid generelt har mindre svingninger enn de øvrige løpetidene. Det er derfor naturlig at kurtose for 5 års og 10 års løpetid er lavere enn de resterende.

4.3. ARIMA



Figur 7 ACF & PACF for 3 mnd løpetid.

ACF og PACF indikerer potensielle signifikante korrelasjoner da noen av stolpene krysser konfidensintervallinjene, men dette gir kun grunnlag for en innledende antakelse. Som forklart i kapittel 2.5.1 kan AIC gi en god indikasjon på modellordren i en ARIMA-modell. Vi har identifisert flere modeller for hver løpetid som kan være aktuelle. Vi har derfor sjekket AIC opp mot hverandre og går i stor grad videre med de modellene som har lavest AIC. Kompleksitet i modellene er tatt med i beregningen og i de tilfellene hvor reduksjon i AIC er marginal, mens kompleksiteten i modellen øker betraktelig, har vi gått videre med modellen som har nest lavest AIC.

4.4. 3 måneder løpetid

Ut fra ACF og PACF plot ser det ut for at ARIMA(1, 1, 3), ARIMA(9, 1, 3), ARIMA(24, 1, 3) kan være passende modeller. *auto.arima*-funksjon i R presenterer også ARIMA(1, 1, 2) som aktuell.

Tabell 7 AIC tabell for 3 måneder løpetid.

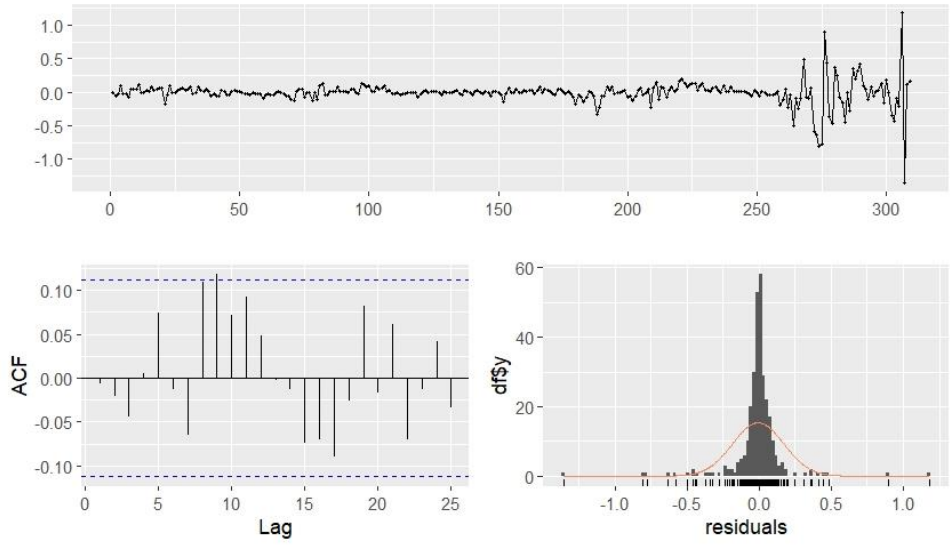
Model	**AIC 3 Mnd**
ARIMA (1, 1, 3)	-176.5108
ARIMA (9, 1, 3)	-184.1427
ARIMA (24, 1, 3)	-185.8021
ARIMA (1, 1, 2)	-178.4894

Vi ser i Tabell 7 at ARIMA(9, 1, 3) har lavest AIC-verdi.

Tabell 8 Nøyaktighetsmålinger for ARIMA-modeller for 3 måneders løpetid.

	MAPE	RMSE	MAE
ARIMA(1, 1, 3)	**289.3406**	**0.1777**	**0.0842**
ARIMA(9, 1, 3)	422.6376	0.1696	0.0850
ARIMA(24, 1, 3)	464.4313	0.1608	0.0825
ARIMA(1, 1, 2)	295.9396	0.1777	0.0843

ARIMA(1, 1, 3) presterer best sett ut fra MAPE noe som indikerer at denne modellen er mer pålitelig dersom man ser på forholdet mellom prediksjonsfeil og faktiske verdier. ARIMA (24, 1, 3) har lavest RMSE noe som tilsier at denne modellen gir best resultat i nøyaktighet per kvadratisk feil. Alle tre modeller har identisk ytelse når man ser på MAE. Dette indikerer at uavhengig av regning på feilen, gir modellene konsistent prediktiv nøyaktighet i absolutte termer.



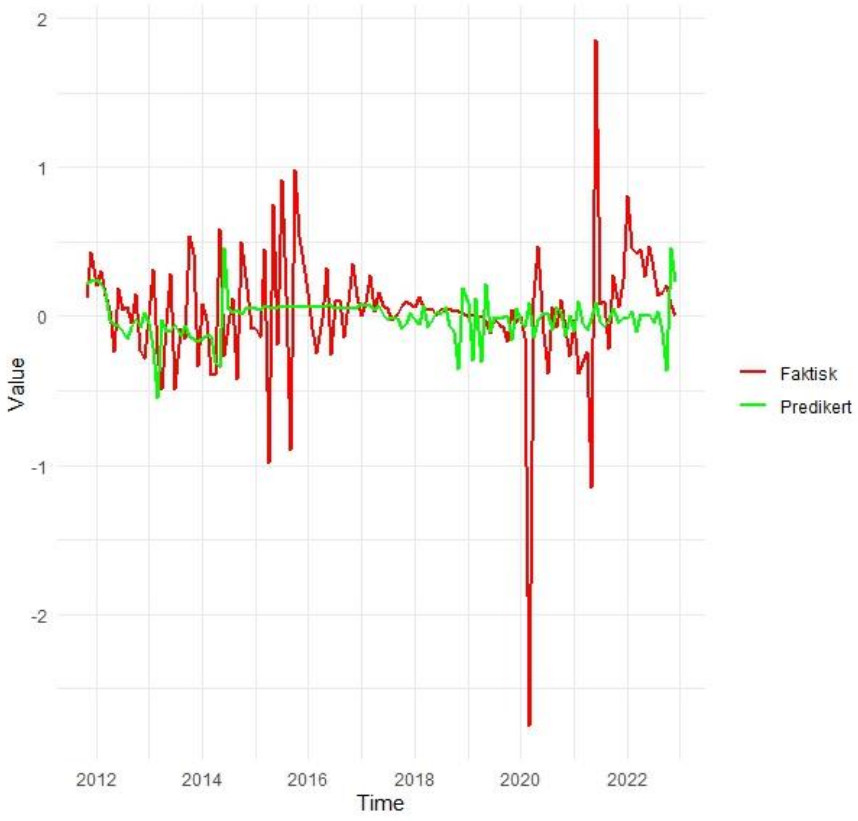
Figur 8 residualer for ARIMA(1, 1, 3).

Tidslinjen for residualene i Figur 8 (øverst) viser en flat linje, i alle fall frem til slutten av tidsserien. Vi ser ingen åpenbare trender eller mønstre og betrakter derfor residualene som hvit støy. Dette underbygges av ACF-plottet (nederst til venstre) da vi ikke ser noen signifikante verdier. I histogrammet nederst til høyre ser vi at kurven er tilnærmet normalfordelt. Samlet sett gjør dette at vi anser residualene for å være hvit støy for ARIMA(1, 1, 3).

Tabell 9 Kryssvaliderte nøyaktighetsmålinger for ARIMA-modeller for 3 måneders løpetid.

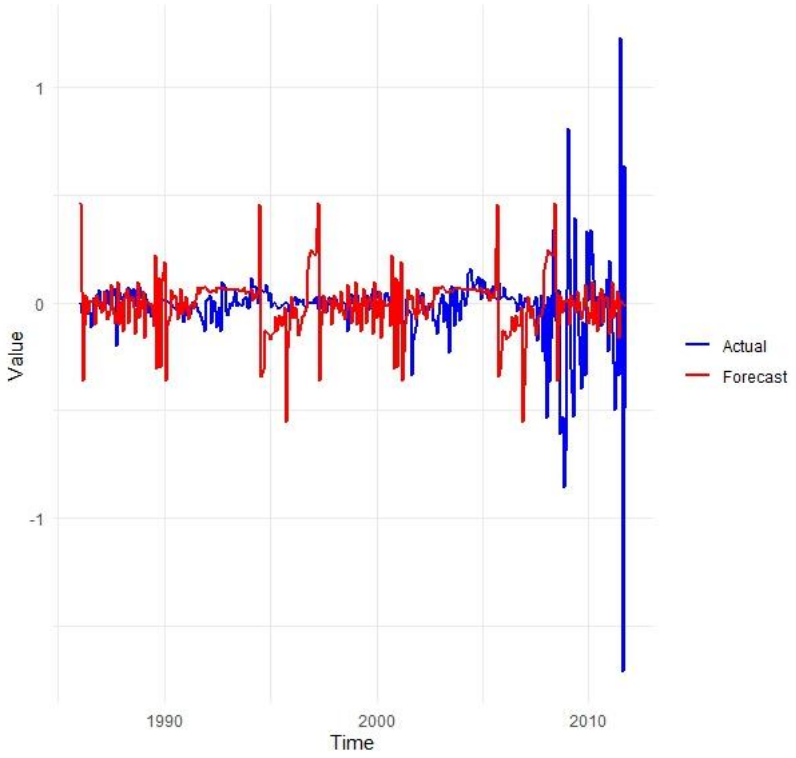
	MAPE	RMSE	MAE
ARIMA(1, 1, 3)	454.9232	0.4424	0.26900

Tabell 9 har RMSE og MAE som antyder at modellen er moderat nøyaktig. Modellen har en MAPE på 454,92% noe som tyder på at modellens forutsigelser i gjennomsnitt avviker med mer enn fire ganger de faktiske verdiene. Dette kan indikere at modellen ikke presterer godt nok, eller at det kan skyldes høyre tall i de faktiske verdiene som modellens prosentvise feilberegning.



Figur 9 Out-of-sample ARIMA(1, 1, 3) for 3 mnd løpetid.

Figur 9 indikerer at modellen til en viss grad evner å følge de faktiske verdiene, men den viser klare avvik på de faktiske verdiene. Avvikene er tydeligst i de største topp- og bunnpunktene i faktisk data, som modellen ikke ser ut til å klare å plukke opp og modellen ser i liten grad ut til å få med seg svingninger i opprinnelig data. Modellen ser ut til å tilpasse seg dataen bedre i begynnelsen av perioden, enn på slutten.



Figur 10 In sample ARIMA(1, 1, 3) for 3 mnd løpetid.

Figur 10 viser at den predikerte linjen ligger tett på de faktiske verdier, spesielt i starten av grafen, mens vi ser større avvik når det det store utslag ved slutten av perioden. Vi legger også merke til at modellen antar større svingninger i dataen enn det som faktisk har vært. Det kan virke som modellen til en viss grad har lært seg mønsteret i historiske data og gjør en akseptabel jobb i å følge faktiske verdier.

4.4.1. 6 måneder løpetid

Når vi ser på ACF og PACF plot for 6 måneders løpetid gir disse indikasjoner på at ARIMA(9, 1, 2) og ARIMA(18, 1, 21) kan være passende modeller. *auto.arima*-funksjonen gir oss en tredje modell, ARIMA(1, 1, 2) og vi går videre med å se på AIC-verdiene for disse tre modellene.

Tabell 10 AIC tabell for 6 måneder løpetid.

Model	**AIC 6 Mnd**
ARIMA (9, ,1, 2)	-331.00
ARIMA (18, 1, 21)	**-375.59**
ARIMA (1, 1, 2)	-322.66

Vi ser i Tabell 10 at i ARIMA(18, 1, 21) kan være den mest passende.

Tabell 11 Nøyaktighetsmålinger for ARIMA-modeller for 6 måneders løpetid.

	MAPE	***RMSE***	***MAE***
ARIMA(9, 1, 2)	*371.1080*	*0.1355*	*0.0737*
ARIMA(18, 1, 21)	*478.0964*	*0.1087*	*0.0696*
ARIMA(1, 1, 2)	***232.8511***	***0.1404***	***0.0711***

Denne modellen har i tillegg lavest RMSE og MAE noe som gir tegn til at modellen er den mest nøyaktige. Den høye verdien av MAPE gir på sin side indikasjon om at ARIMA(18, 1, 21) kan ha betydelige feil, når feil oppstår i forhold til de faktiske verdiene. ARIMA(9, 1, 2) utpeker seg ikke i noen retning. Denne modellen har også relativt høy MAPE noe som indikerer store prosentvise avvik mellom predikerte og faktiske verdier. Når det gjelder ARIMA(1, 1, 2) har denne modellen høyest RMSE og indikerer en lavere prediktiv nøyaktighet. Modellen har derimot en relativt lav MAE og den klart laveste MAPE. Dette antyder at ytelsen når det kommer til prosentvise avvik mellom prediksjoner og faktiske verdier er den beste.

Tabell 12 Kryssvaliderte nøyaktighetsmålinger for ARIMA-modeller for 6 mnd løpetid.

	MAPE	**RMSE**	**MAE**
*ARIMA(1, ,1 ,2**)***	817.4053	0.3258	0.1982

Tabell 12 viser kryssvaliderte nøyaktighetsmålinger for den beste ARIMA modellen for 6 mnd løpetid. Likt som for 3 mnd løpe tid, ser vi at MAPE viser betydelig høye verdier på 817.40% som indikerer at det er betydelig prosent feil i modellens forutsigelser. Den høye MAPE verdien kan også være resultat av noen spesifikke resultater eller datapunkter som er veldig ulik fra modellens forutsigelser. RMSE på 0.3258 antyder at det kan være et moderat nivå på avvik mellom de forutsagte og faktiske verdier. MAE på 0.1982 kan på en måte bekreftet dette som viser at modellen i gjennomsnitt har en forholdvis liten feil pr forutsigelse.

4.4.2. 1 års løpetid

ACF og PACF plottene for 1 års løpetid antyder at ARIMA(9, 1, 9), ARIMA(11, 1, 19) kan være aktuelle. Vi ser også at ARIMA(2, 0, 2) kan være aktuell ved å benytte oss av *auto.arima*-funksjonen.

Tabell 13 AIC tabell for 1 års løpetid.

	AIC 1år
ARIMA (9, 1, 9)	-396.6967
ARIMA (11, 1, 19)	**-401.0432**
ARIMA (2, 0, 2)	-393.8026

Tabell 13 viser at ARIMA(11, 1, 19) har lavest verdi, men vi legger merke til at det ikke er store forskjeller mellom modellene. Det kan tenkes at den økte kompleksiteten til ARIMA(11, 1, 19) ikke gir tilsvarende økt verdi når vi ser på AIC-verdien. Dette vil vi se nærmere på i nøyaktighetsmålingene til modellen.

Tabell 14 Nøyaktighetsmålinger for ARIMA-modeller for 1 års løpetid.

	MAPE	***RMSE***	***MAE***
ARIMA(9, 1, 9)	*583.9977*	*0.1177*	*0.0751*
ARIMA(11, 1, 19)	*408.5482*	*0.1109*	*0.0722*
ARIMA(2, 0, 2)	***364.6156***	***0.1256***	***0.0761***

ARIMA(9, 1, 9) har relativt høy MAPE sett opp mot de andre to modellene i Tabell 14. Dette indikerer at modellen kan ha systematiske feil. Selv om modellen har gode resultater for RMSE og MAE, gjør den høye MAPE-verdien at vi ser bort fra modellen. ARIMA(11, 1, 19) har bedre RMSE og MAE, noe som indikerer en bedre gjennomsnittlig nøyaktighet, i tillegg til at MAPE er redusert sett opp mot ARIMA(9, 1, 9). Dette antyder at modellen håndterer prosentvise feil bedre. ARIMA(2, 0, 2) har høyest RMSE, men MAE ligger på linje med de to øvrige, og MAPE-verdien er den absolutt laveste. Dette antyder at selv om modellen kan ha store enkeltfeil vil gjennomsnittlige prediksjonsfeil over tid være lavest.

Tabell 15 Kryssvaliderte Nøyaktighetsmålinger for ARIMA-modeller for 1år løpetid.

	MAPE	**RMSE**	**MAE**
ARIMA(2, 0, 2)	346.3962	0.2481	0.1441

Vi ser i Tabell 15 en MAPE-verdi på 346,39 % som er høy. Sett opp mot 3mnd og 6 mnd løpetid ser vi at selv om MAPE for 1 års løpetid vurderes som høy, er den vesentlig lavere enn de to øvrige løpetidene. Sett opp mot Tabell 14 er det liten forskjell i kryssvaliderte verdier. MAPE-verdien tyder på at det er enkelte tilfeller hvor modellens forutsigelser er unøyaktige i forhold til de faktiske verdiene. RMSE på 0.2481 og MAE på 0.1441 indikerer at modellens forutsigelser ikke utgjør store forskjeller fra de faktiske verdiene og at den i gjennomsnitt gjør mindre feil.

4.4.3. 5 års løpetid

Etter å ha sett på ACF og PACF plot har vi identifisert 2 mulige ARIMA-modeller for 5 års løpetid. ARIMA(7, 1, 7) og ARIMA(13, 11, 1) kan være passende. Vi har videre sjekket med *auto.arima*-funksjonen og får indikasjon på at ARIMA(0, 0, 2) kan være passende. Vi går derfor videre med de tre modellene.

Tabell 16 AIC tabell for 5 års løpetid.

Model	**AIC 5 år**
ARIMA (7, 1, 7)	**-666.9552**
ARIMA (13, 1, 11)	-648.6311
ARIMA (0, 0, 2)	-648.6311

Ut fra Tabell 16 ser vi at det kan være hensiktsmessig å gå videre med ARIMA(7, 1, 7), men det er små forskjeller som gjør at denne ligger på topp. Vi ser videre på nøyaktighetsmålingene for de ulike modellene:

Tabell 17 Nøyaktighetsmålinger for ARIMA-modeller for 5 års løpetid.

	MAPE	RMSE	MAE
ARIMA(7, 1, 7)	308.6685	0.0770	0.0549
ARIMA(13, 1, 11)	299.6849	0.0756	0.0537
ARIMA(0, 0, 2)	**146.8385**	**0.0828**	**0.0559**

MAPE gir en tydeligere distinksjon mellom modellene og vi ser at ARIMA(0, 0, 2) har lavest MAPE-verdi. Alle tre modeller gir tilsvarende verdier i MAE, med små forskjeller. Vi ser noe ulikheter i RMSE, men heller ikke her er forskjellene av spesielt stor grad.

Tabell 18 Kryssvaliderte Nøyaktighetsmålinger for ARIMA-modeller for 5år løpetid.

	MAPE	RMSE	MAE
ARIMA(0, 0, 2)	120.6198	0.1699	0.1233

Tabell 18 viser MAPE verdi på 120.61% som igjen er lavere enn foregående løpetider. Den viser at det er noe forutsigelser som avviker betydelig fra de faktiske verdiene i tillegg til at dette skjer relativt ofte. MAPE indikerer at modellen generelt predikerer ganske godt. RMSE på 0.1699 er relativt lavt og indikerer at modellens forutsigelser er nesten like faktiske verdier. MAE-verdien på 0.1233 støtter den lave RMSE-verdien som viser at gjennomsnitt feilen per forutsigelse er veldig liten.

4.4.4. 10 års løpetid

Vi har for 10 års løpetid identifisert fire modeller ved å se på ACF og PACF plot. ARIMA(2, 1, 2), ARIMA(3, 1, 2), ARIMA(2, 1, 5) og ARIMA(3, 1, 5) anses som aktuelle modeller som vi ønsker å sjekker AIC på. Funksjonen *auto.arima* gir oss en femte modell: ARIMA(5, 0, 1).

Tabell 19 AIC tabell for 10 års løpetid.

Model	**AIC 10 år**
ARIMA (2, 1, 2)	-872.9920
ARIMA (3, 1, 2)	-872.8311
ARIMA (2, 1, 5)	-870.4711

ARIMA (3, 1, 5)	-868.4498
ARIMA (5, 0, 1)	**-890.9515**

I Tabell 19 ser vi at ARIMA(5, 0, 1) har lavest verdi, og vil være den mest interessante sett ut fra AIC.

Tabell 20 Nøyaktighetsmålinger for ARIMA-modeller for 10 års løpetid.

Model	**MAPE**	**RMSE**	**MAE**
ARIMA(2, 1, 2)	**208.2196**	**0.0573**	**0.0417**
ARIMA(3, 1, 2)	270.4104	0.0572	0.0416
ARIMA(2, 1, 5)	329.3411	0.0570	0.0413
ARIMA(3, 1, 5)	316.3411	0.0570	0.0413
ARIMA(5, 0, 1)	509.3581	0.0560	0.0410

Tabell 21 Kryssvaliderte Nøyaktighetsmålinger for ARIMA-modeller for 10 år løpetid.

	MAPE	**RMSE**	**MAE**
ARIMA(2, 1, 2)	129.6262	0.1228	0.0907

I Tabell 21 ser vi en MAPE-verdi på 129.62%. Dette indikerer at det er visse tilfeller hvor prosentvis avvik er betydelig. Generelt indikerer de kryssvaliderte nøyaktighetsmålingene at modellen presterer relativt godt. RMSE på 0.1228 viser en lav gjennomsnittlig kvadrert avstand mellom de faktiske og predikerte verdiene. En MAE på 0.0907 viser at de gjennomsnittlige absolutte feilene er veldig små.

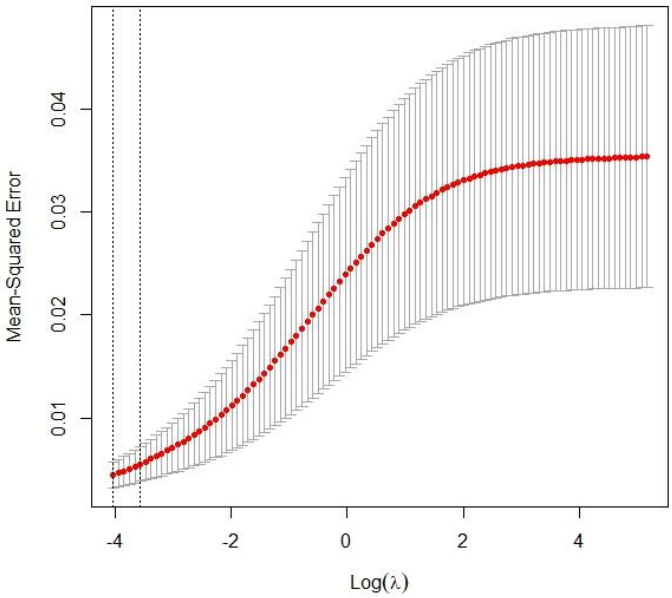
4.4.5. Sammenligning ARIMA

Tabell 22 Sammenligning av de beste kryssvaliderte nøyaktighetsmålinger for ARIMA.

	3mnd	**6mnd**	**1 år**	**5 år**	**10 år**
AIC	-176.5186	-322.66	-393.8026	-648.6311	-872.8311
MAPE	454.9232	817.4053	346.3962	120.6198	129.6262
RMSE	0.4424	0.3258	0.2481	0.1699	0.1228
MAE	0.2690	0.1982	0.1441	0.1233	0.0907

I Tabell 22 ser vi oversikt over de beste kryssvaliderte nøyaktighetsmålingene for ARIMA for alle løpetider. Vi ser at 5 – og 10 års løpetid viser en relativ lavere MAPE enn andre løpetider, som også har lave RMSE og MAE som indikerer lavere gjennomsnittlige feil på de predikerte verdiene.

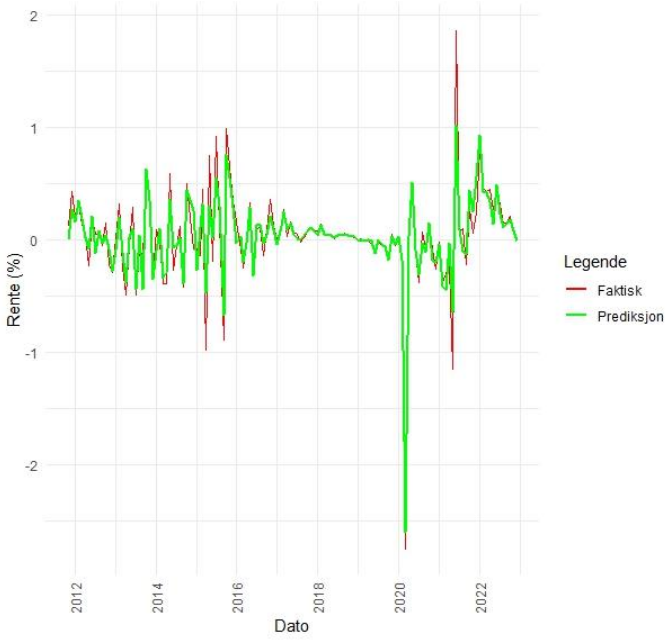
4.5. Ridge regresjon



Figur 11 Valideringskurve for Rigde regresjon 3 mnd.

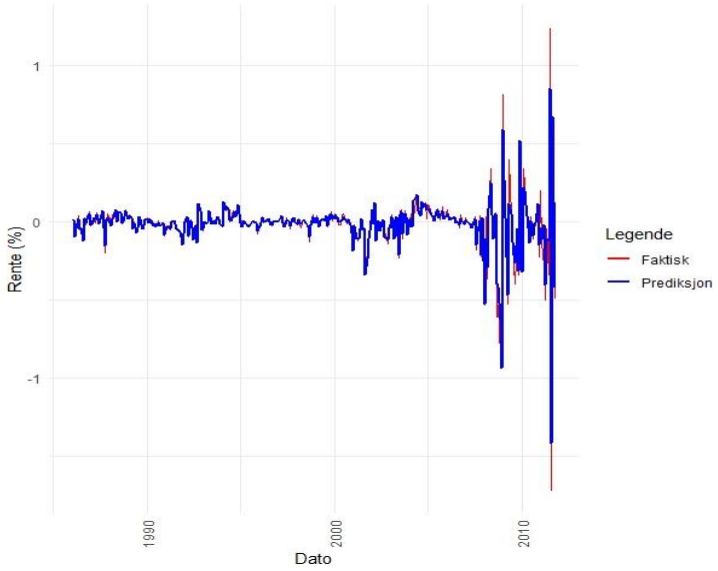
X-aksen i Figur 11 viser logaritmen til regulariseringsparameteren λ, som er Alpha i Rigde regresjon mens Y-aksen viser MSE for modellen. Den røde linjen viser MSE for forskjellige verdier av Alpha, med konfidensintervaller indikert av de vertikale linjene. Kurven vil bidra til å finne en balanse mellom under – og overtilpasning ved å indentifisere en Alpha-verdi som minimerer MSE, noe som igjen vil indikere en effektiv modellkompleksitet for å forutsi rentenivå uten å ende opp med mye bias eller varians (D. Patel, 2021).

4.5.1. 3 måneder løpetid



Figur 12 out-of-sample ridge regresjon for 3 mnd løpetid.

Figur 12 viser en out-of-sample prediksjon for 3 mnd løpetid sett opp mot faktiske data. Den røde linjen viser de faktiske verdiene, mens den grønne linjen viser Ridge verdiene fra modellen. Grafen viser den predikerte verdien følger trenden de faktiske verdiene som indikerer at modellen har visse prediktive egenskaper. Vi ser også noen klare avvik til visse tider, og spesielt ved toppene og dalene i de faktiske verdier, noen som tilsier at modellen ikke alltid klarer å predikere på de mer store svingningene.



Figur 13 In sample tilpasning av Rigde regresjonsmodell 3 mnd.

I Figur 13 ser vi den røde linjen representere de faktiske renteverdiene over tid, mens den blå linjen representerer modellens prediksjoner som er basert på de faktiske historiske dataene. Nærheten mellom den blå og røde linjen indikerer hvor godt ridge regresjonen klarer å tilpasses seg og forutse de faktiske renteverdiene. Modellen tilpasser seg ikke topp- og bunnpunktene. Det kan skyldes markante svingninger i de faktiske verdiene som modellen ikke klarer å forutsi.

Tabell 23 Sammenligning av de beste kryssvaliderte nøyaktighetsmålinger for ridge.

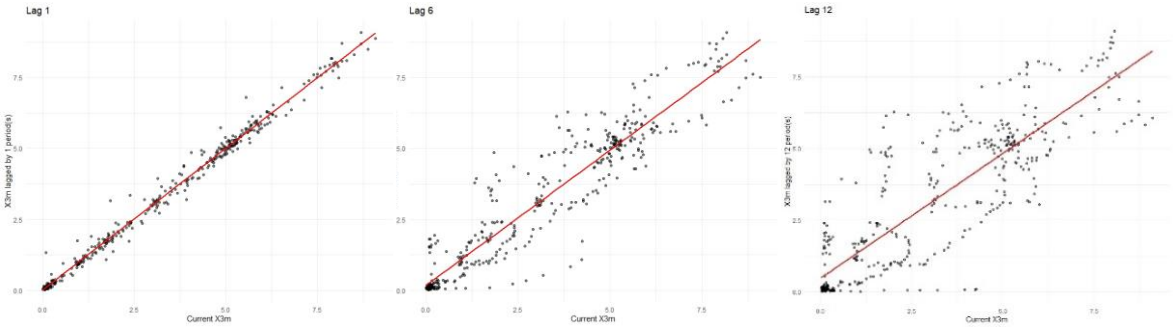
	3mnd	**6mnd**	**1y**	**5y**	**10y**
MAPE	102.9435	86.1183	357.5126	56.7831	96.8737
RMSE	0.1506	0.0589	0.0759	0.0541	0.0456
MAE	0.09241	0.0370	0.0503	0.0360	0.0348

I Tabell 23 ser vi at de lange løpetidene ser ut til å prestere bedre enn kortere løpetider. Vi legger merke til at modellen for 6 mnd løpetid presterer godt, faktisk bedre enn 10 års løpetid gjør. Generelt ser vi lave verdier for MAPE, bortsett fra for 1 års løpetid som har over 300% i verdi for MAPE. Både verdier for RMSE og MAE har svært lave verdier og indikerer lite feil i prediksjonene. Det er som forventet at RMSE og MAE til en viss grad følger hverandre.

4.6. Random Forest

For å definere antall lags har vi først sett på ACF-plot/korrellogram. Fra Figur 5 ser vi at autokorrelasjonen gradvis avtar etter hvert som lagene øker. For å inkludere lags som inneholder relevant informasjon om fremtidige verdier anser vi det som riktig å inkludere de første lagene der autokorrelasjonen er sterkest. For mange lags kan redusere ytelsen og vi ønsker ikke å inkludere alle de autokorrelerte lagene.

Vi har benyttet 12 lags for å forsøke å plukke opp årlig sesongmessig trend i dataen samt gi økt kompleksitet i modellen for å forbedre nøyaktighet i prognosene.

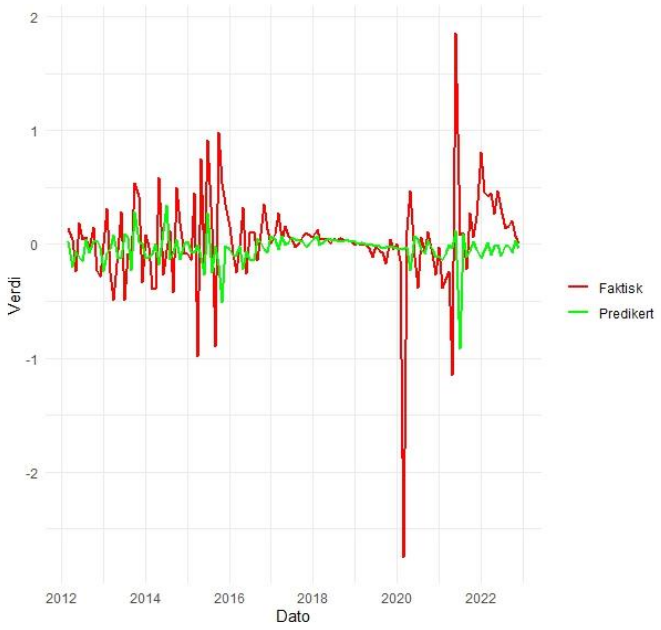


Figur 14 Lag plot for 3 mnd løpetid

Figur 14 viser forskyvelser over 12 lags fra original data. Plottet for lag 1 viser forholdet mellom raten på et gitt tidspunkt og neste tidsperiode. Plottet viser som forventet sterk lineær sammenheng. I plottet for lag 6 ser vi at det fortsatt er positiv sammenheng mellom raten og dens verdier 6 måneder tilbake. I siste plot for lag 12 ser vi denne effekten enda sterkere. Sammenhengen er svakere og minnet i tidsserien blir gradvis svakere over tid. Plottene indikerer at dataen er autokorrelert, og at tidligere verdier påvirker fremtidige verdier.

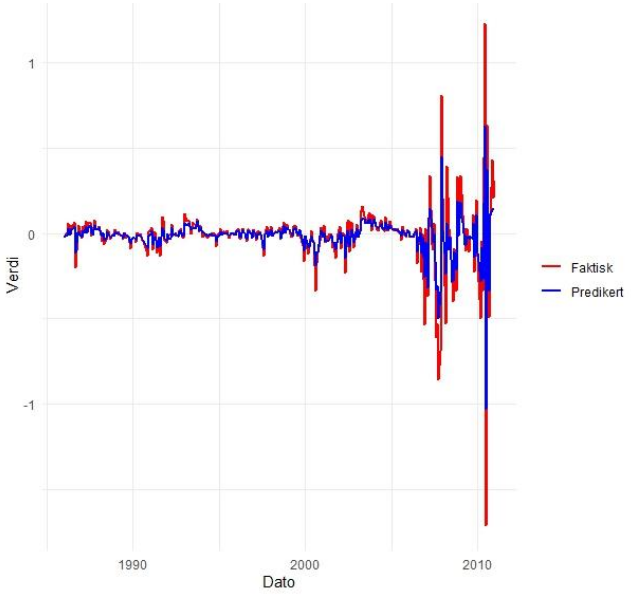
Vi benytter 10 fold kryssvalidering. Som forklart i kapittel 4.1.2 benyttes 5 fold eller 10 fold kryssvalidering da disse er empirisk bevist til å være de som kommer best ut i bias/varians trade off. Ettersom vi har et datasett som strekker seg over 36 år benytter vi 10 fold kryssvalidering i vår modellering. I denne valideringsmetoden deles datasettet inn i 10 like store deler før modellen gjennomgår trening og evaluering 10 ganger. Modellen skal til slutt ha brukt alle delene som testsett en gang, med resten som treningssett.

4.6.1. 3 måneder løpetid



Figur 15 out-of-sample random forest for 3 mnd løpetid.

Figur 15 viser at den predikterte verdien følger den faktiske verdien tett i visse punkter, men vi ser et tydelig avvik som spesielt vises ved større utslag i faktiske verdier. Grafen indikerer at modellen kan ha ferdigheter til å forutsi generelle trender, men kan slite med å nøyaktig forutse mer volatile svingninger i faktiske verder. Det kan se ut til at modellen predikerer tettere opp mot faktisk verdi tidlig i perioden, mot slutten ser vi større avvik fra faktisk data.



Figur 16 in- sample for random forest for 3 mnd løpetid.

I Figur 16 ser vi at den blå linjen ligger veldig tett mot den røde linjen som indikerer modellen har en god passform og klarer å fange opp de underliggende mønstrene i de faktiske verdiene relativ godt. Likevel så ser mot slutten av tidsrekken, hvor det er tydelig avvik, som indikerer at modellen kan ha utfordringer med å fange opp større utslag i den faktiske verdien, eller at det er rom for ytterligere forbedringer i modellen.

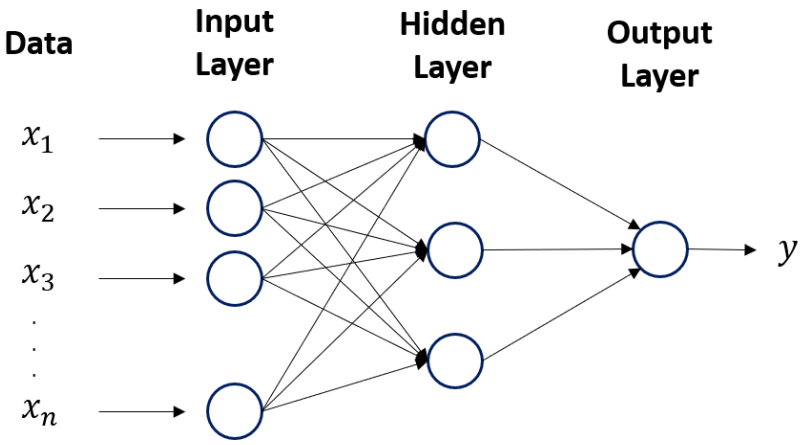
Tabell 24 Sammenligning av de beste kryssvaliderte nøyaktighetsmålinger for Random Forest.

	3mnd	**6mnd**	**1y**	**5y**	**10y**
RMSE	0.1808	0.1452	0.2564	0.1755	0.1268
MAE	0.2705	0.0747	0.1518	0.1258	0.0941
MAPE	174.58	190.90	286.25	123.54	126.90

Tabell 24 viser MAPE verdier som varierer noe. 5 år og 10 års løpetid har de laveste verdiene, noe som indikerer at modellens prosentvise avvik fra de faktiske verdier er minst ved de lengre løpetidene. Når vi ser RMSE verdiene observerer vi en trend hvor modellen presterer bedre jo lengre løpetid, med den laveste RMSE verdien på 0.1268 for 10 års løpetid. MAE-verdiene støtter dette, da vi også her ser den laveste verdien for 10 års løpetid.

4.7. LSTM

For at vi skal kunne anvende LSTM må vi definere modellens ulike faktorer. Som nevnt i kapittel 2.6.2 er LSTM delt i tre lag(porter). Vi må velge hvor mange neuroner det skal være i hvert lag og det er viktig å legge merke til at siden alle lagene er koblet til hverandre som vist i Figur 17 vil kompleksiteten øke eksponentielt. Det er derfor viktig å begrense kompleksiteten i modellen.



Figur 17 Eksempel med ulike lag i en LSTM modell. TowardsAI, 2020.

Siden LSTM forventer at dataen er i overvåket læringsmodus ved å ha en målvariabel Y og prediktor X, må vi transformere dataen ved å opprette et tilsynsdatasett med forsinkede verdier. På samme måte som i andre neurale nettverksmodeller skalerer vi inndataene X til arkiveringsfunksjonens område. Ettersom vi har benyttet univariate datamodeller benytter vi kun èn inngangsnøkkel i vår LSTM-modell og vi har valgt å gå for sigmoid funksjonen som er blant de vanligste metodene som vi beskriver i kapittel 2.6.2. Modellen benytter 12 tidssteg, ettersom vi har månedlige rentedata ønsket vi å ta med 1 år for å kunne plukke opp viktige mønstre i dataen. Vi har valgt et sett hyperparametre og iterert gjennom en løkke som kjører hver av parameterne i kombinasjon med hverandre og måler resultatene fra hver modell som blir generert. Modellene med høyest nøyaktighet blir presentert i en liste til slutt.

Tabell 25 Oversikt over hyperparametre i LSTM modell.

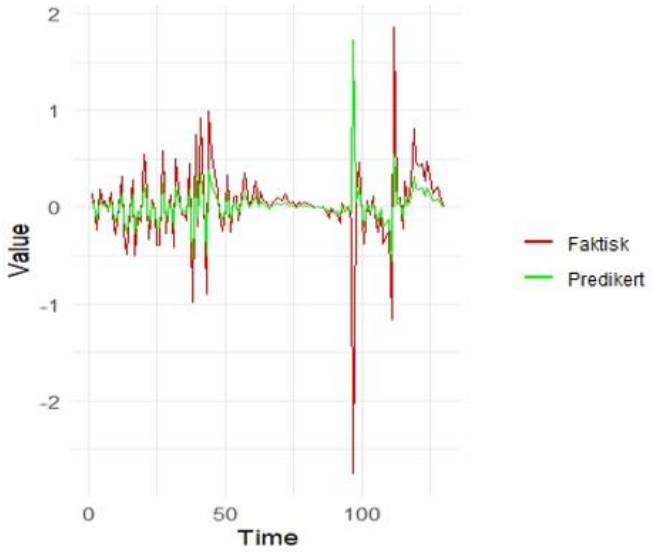
	Verdi 1	**Verdi 2**	**Verdi 3**
Time steps	12	N/A	N/A
Unit options	100	200	N/A
Learning rate options	0.2	0.3	N/A
Batch Size	32	64	128
Epoch options	50	100	N/A
Optimizer options	Adam	Nadam	N/A

4.7.1. 3 måneders løpetid

Tabell 26 Topp 3 LSTM modeller for 3 mnd løpetid.

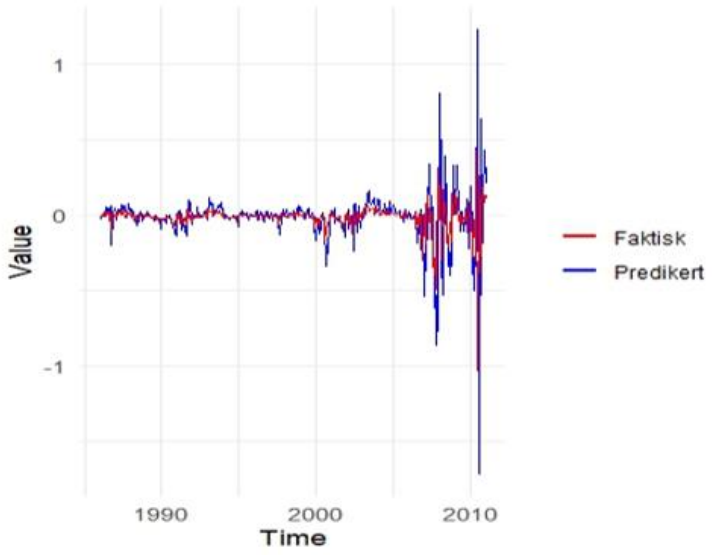
	MAPE	RMSE	MAE	Enheter	Læringsrate	Epoker	Batch Størrelse	Optimaliserer
Model 11	**16.1938**	**0.1444**	**0.0835**	**100**	**0.2**	**100**	**128**	**Adam**
Modell 7	16.2528	0.1441	0.0829	100	0.2	100	32	Adam
Modell 31	16.2410	0.1441	0.0830	200	0.2	100	32	Adam

I Tabell 26 ser vi at modell 11 har en MAPE på 16.19 %, og selv om den ikke har den laveste RMSE og MAE sammenlignet med modell 7 og 31 så det MAPE verdien som blir prioritert for å bedømme den beste modellen. Dette indikerer at modell 11 er bedre tilpasset eller har en bedre generaliseringsevne på tross av marginalt høyere RMSE og MAE.



Figur 18 Out-of-sample LSTM for 3 mnd løpetid.

Figur 18 viser at predikert verdi til en viss grad følger faktisk verdi, men modellen har vanskeligheter med å tilpasse seg ytterpunktene. Vi legger også merke til at på et punkt, når faktiske verdier har en stor dupp i renteverdiene predikerer modellen stikk motsatt – et hopp i renteverdi. Videre ser det ut for at modellen klarer å plukke opp mønstrene igjen og får til dels med seg oppturer og nedturer i renteverdi.



Figur 19 In-sample LSTM for 3 mnd løpetid.

Figur 19 viser at den predikterte verdien følger den faktiske verdien tett, noe som indikerer at modellen plukker opp mønstrene i dataene i treningsdatasettet. Likevel er det perioder hvor den predikterte modellen ikke klarer å følge de faktiske verdiene og det er i de store utslagene modellen ikke har klart å fange topp og bunnpunkter i de historiske dataene.

4.7.2. 6 måneders løpetid

Tabell 27 Topp 3 LSTM modeller for 6 mnd løpetid.

	MAPE	RMSE	MAE	Enheter	Læringsrate	Epoker	Batch Størrelse	Optimaliserer
Model 11	**14.9247**	**0.1655**	**0.0974**	**100**	**0.2**	**100**	**128**	**Adam**
Modell 7	15.0344	0.1658	0.0983	100	0.3	100	32	Adam
Modell 21	15.0388	0.1657	0.0981	100	0.3	100	64	Adam

I Tabell 27 ser vi at modell 11, med en MAPE på 14.92%, er den laveste blant de tre modellene. Selv om modell 7 og 21 har lignende verdier for RMSE og MAE er det MAPE som er den avgjørende faktoren for å velge modell 11 som den mest nøyaktige.

4.7.3. 1 års løpetid

Tabell 28 Topp 3 LSTM modeller for 1 år løpetid.

	MAPE	RMSE	MAE	Enheter	Læringsrate	Epoker	Batch Størrelse	Optimaliserer
Modell 12	**16.6304**	**0.1699**	**0.0982**	**100**	**0.2**	**100**	**128**	**Nadam**
Modell 10	16.6331	0.1689	0.0961	100	0.2	100	64	Nadam
Modell 11	16.7818	0.1702	0.0984	100	0.2	100	128	Adam

Tabell 28 viser marginale forskjeller mellom de beste modellene. Den beste modellen som er mest nøyaktig er likevel modell 12.

4.7.4. 5 års løpetid

Tabell 29 Topp 3 LSTM modeller for 5 års løpetid.

	MAPE	RMSE	MAE	Enheter	Læringsrate	Epoker	Batch Størrelse	Optimaliserer
Modell 21	**37.2818**	**0.2449**	**0.1823**	**100**	**0.3**	**100**	**64**	**Adam**
Modell 14	37.4403	0.2359	0.1718	100	0.3	50	32	Adam
Modell 15	37.5512	0.2350	0.1708	100	0.3	50	64	Adam

Modell 21 har lavest MAPE på 37.28%, og vi går videre med denne kombinasjonen av hyperparametere for denne modellen.

4.7.5. 10 års løpetid

Tabell 30 Topp 3 LSTM modeller for 10 år løpetid.

	MAPE	RMSE	MAE	Enheter	Lærings-rate	Epoker	Batch Størrelse	Optimaliserer
Modell 20	**110.9891**	**0.5023**	**0.4551**	**100**	**0.3**	**100**	**32**	**Adam**
Modell 34	133.7660	0.3931	0.3417	200	0.2	100	64	Nadam
Modell 29	168.5620	0.2884	0.2291	100	0.2	50	128	Adam

I Tabell 30 ser vi marginale forskjeller mellom topp 3 modellene. Vi går videre med modell 20 med en MAPE på 110.98 %.

4.7.6. Sammenligning LSTM

Tabell 31 Sammenligning av de beste kryssvaliderte nøyaktighetsmålinger for LSTM.

	3mnd	**6mnd**	**1 år**	**5 år**	**10 år**
RMSE	0.1444	0.1655	0.1699	0.2449	0.5023
MAE	0.0835	0.0974	0.0982	0.1823	0.4551
MAPE	16.1938	14.9247	16.6304	37.2818	110.9891

Tabell 31 viser lave verdier for RMSE og MAE for samtlige modeller. Vi ser tydelig høyere MAPE for de mellomlange løpetidene, spesielt skiller 10 års løpetid seg fra de resterende. Dette er i seg selv ikke en veldig høy verdi for MAPE, men den skiller seg fra resten av modellene i tabellen. Vi ser også at MAE og RMSE ligger høyere for 10 års løpetid enn de resterende.

4.8. Modellsammenligning

Tabell 32 Sammenligning av alle modeller.

Modell	**Løpetid**	**MAPE**	**RMSE**	**MAE**
	3mnd	454.92	0.4424	0.2690
	6mnd	817.41	0.3258	0.1982
ARIMA	1 år	346.40	0.2481	0.1441
	5 år	120.62	0.1699	0.1233
	10 år	129.63	0.1228	0.0907
	3mnd	102.94	0.1506	0.0924
	6mnd	86.12	0.0589	0.0370
Ridge	1år	357.51	0.0759	0.0503
	5år	56.78	0.0541	0.0360
	10år	96.87	0.0456	0.0348
	3mnd	174.58	0.1808	0.2705
	6mnd	190.90	0.1452	0.0747
Random Forest	1år	286.25	0.2564	0.1518

	5år	123.54	0.1755	0.1258
	10år	126.90	0.1268	0.0941
LSTM	3mnd	16.19	0.1444	0.0835
	6mnd	14.93	0.1655	0.0974
	1år	16.63	0.1699	0.0982
	5år	37.28	0.2449	0.1823
	10år	110.99	0.5023	0.4551

Det er naturlig å sammenligne ARIMA og Ridge da dette er de to tradisjonelle maskinlæringsmodellene vi har benyttet i vår forskning og dataforberedelsene til disse to modellene er gjort på samme måte. For de korte løpetidene ser vi i Tabell 32 at ridge modellen har prestert vesentlig bedre sett ut fra både MAE, MAPE og RMSE enn ARIMA-modellene.

De to moderne modellene krever en annen preprosessering av datasettet enn de tradisjonelle modellene gjør og vi ser derfor resultatene til disse opp mot hverandre. LSTM modellene ser på det jevne ut til å prestere bedre enn Random Forest. For de moderne modellene, spesielt for LSTM ser vi motsatt enn for de tradisjonelle, at det er mellomlange og lange løpetider som presterer dårligere enn de øvrige løpetidene for alle nøyaktighetsmålingene. Random Forest modellen har mindre grad av varians i resultatene, men 1 års løpetid stikker seg ut som noe dårligere enn de andre løpetidene i modellen.

Med fokus på MAPE ser vi at LSTM presterer best av alle modellene med unntak av 10 års løpetid hvor ridge regresjon presterer bedre. Vi legger merke til at Ridge regresjon har lavere RMSE enn LSTM. Dette betyr at selv om LSTM har lavere prosentvise feil i forhold til de faktiske verdiene har ridge regresjon færre feil enn LSTM-modellen.

4.9. Diskusjon

I dette kapittelet vil vi diskutere det vi mener er de viktigste funnene fra vår forskning. Gjennom omfattende analyser har vi presentert flere modellresultat som gir grunnlag for å forutsi rentesatser over flere løpetider fra kortsiktige til langsiktige, i tillegg til å dekke både tradisjonelle og moderne modeller. Dette har resultert i en oversikt over hvilke modeller som presterer best på våre data. Resultatene har i stor grad vært innenfor våre forventninger, men prestasjonen til ARIMA og Ridge har overrasket oss noe. Avslutningsvis vil vi adressere forskningsspørsmålene som la grunnlaget for vår forskning hvor vi så på forskjellene mellom tradisjonelle og moderne tilnærminger for anvendt

rentepredikSjon av amerikanske statsobligasjoner, samt hvordan nye modeller kan transformere økonomiske analyser.

4.9.1. ARIMA som benchmark

Før vi startet analysen av modellene hadde vi en forventning om at ARIMA skulle prestere på linje med de moderne modellene, selv om moderne modeller har større muligheter til å plukke opp mer komplekse mønstre i dataen som forklart i kapittel 1.1. Denne forventningen bygger på at ARIMA ofte har vært brukt som benchmark i flere forskninger (Cholette, 1982), i tillegg til å være en foretrukket modell over lang tid ved tidsserieprediksjon (Zhang, 2003).

For korte løpetider presterer ARIMA vesentlig dårligere enn de øvrige 3 modellene. En av årsakene til at modellen presterer dårligere på de korte løpetidene kan være at korte løpetider dårligere håndterer sjokk i markedet enn lengre løpetider (Mallick & Mishra, 2019). Siden ARIMA, som forklart i kapittel 1.1 forventer linearitet i dataene og har begrenset mulighet til å lese dataens mønstre kan prestasjonen til ARIMA-modellen på korte løpetider forklares med at ARIMA-modellen ikke har klart å tilpasse seg raske endringer og uforutsette hendelser.

En annen mulig forklaring til hvorfor ARIMA presterer dårligere på de korte løpetidene kan være på grunn av deres raske tilpasning til endringer i markedsforhold og pengepolitikk da disse i stor grad blir påvirket av valgene til Amerikas sentralbank (McCormick & Regan, 2021). Sjokk i markedsforhold eller pengepolitikk kan gjøre at dataene for korte løpetider ikke er lineære. Forskningen til Radha og Thenmozhi viser at ARIMA kan ha en tendens til å prestere dårlig på korte løpetider dersom rentene viser volatilitetsclustering, altså effekten hvor store endringer i volatilitet tenderer til å følges av flere store endringer (Radha & Thenmozhi, 2006). Dette kan indikere at vår tidsserie har volatilitetsclustering som ARIMA-modellen spesielt, ikke klarer å følge godt opp.

En tredje forklaring av prestasjonen til ARIMA-modellen på korte løpetider kan være menneskelig feil i analyseprosessen. Det kan tenkes at vi har gjort feil i undersøkelsene av ACF og PACF plott eller valgt feil ordre av ARIMA. Vi ser blant annet at *auto.arima*-funksjonen har blitt brukt med vellykkede resultater i andre forskninger (Mbah et al., 2021). Det er mulig at vi hadde fått et annet resultat dersom vi kun hadde benyttet *auto.arima*-funksjonen for våre modeller, men basert på AIC-verdiene valgte vi å gå videre med egendefinerte ordre da denne fikk lavere verdi. Det er også verdt å merke seg at verdiene for nøyaktighetsmålingene steg betraktelig etter kryssvalidering, hvor man plukker opp større andel av støyen. Det kan tenkes at det finnes ARIMA-modeller med bedre sammensetning

av ordre. Kanskje kunne caret-pakken bidratt til å identifisere en bedre modell dersom denne var tilpasset ARIMA, men dette blir kun hypotetiske tanker fra vår side

Tidligere forskning indikerer at ARIMA presterer bedre på lengre tidsskala, enn korte (NASDAQ). Selv om lange tidsskalaer i denne forskningen fortsatt er relativt korte i forhold til vår data kan det være et bilde på at ARIMA-modeller bedre håndterer lengre tidsskalaer. Dette sammenfaller med våre resultater som gir dårligere resultat for korte løpetider sett opp mot de lengre løpetidene. Tidligere forskning viser at ARIMA presterer bedre enn LSTM (NASDAQ: Mbah et al., 2021). Dette ser vi ikke tilsvarende resultater for i vår forskning. Likevel ser vi at ARIMA presterer på linje med LSTM for 10-års løpetid, så det kan tenkes at ved enda lengre tidsskala i vår forskning kunne vi fått et lignende resultat hvor ARIMA hadde prestert bedre enn LSTM.

Ettersom rentedata har en ikke-lineær og dynamisk struktur som forklart i kapittel 2.3 kan dette være en avgjørende faktor for hvorfor ARIMA, som forventer linearitet i dataen (se kapittel 1.1), presterer relativt dårlig. Det er likevel ikke alltid tilfellet at ARIMA presterer dårlig på rentedata. For rentedata med ulike løpetider i India har ARIMA prestert godt og kurven for predikert verdi følger kurven for faktisk verdi tett (Mallick & Mishra, 2019b). Det finnes også andre lineære modeller som har vist seg å prestere godt på ikke lineære data. Ridge regresjon har vist seg å kunne håndtere kompleksitet og utfordringer selv når forholdene er ikke-lineære og preget av høy multikollinearitet (Ngo et al., 2003).

4.9.2. Ridge regresjon

Av alle modellenes in- og out-of-sample plots er det Ridge regresjons predikerte verdier som følger faktiske verdier tettest. Figur 12 viser at ridge-modellen klarer å plukke opp ytterpunktene i grafen, dette ser vi ikke hos noen av de øvrige modellene. Ettersom Ridge regresjon inkluderer regularisering for å unngå overfitting, som forklart i kapittel 2.5.2, kan denne modellen tettere følge verdiene både i in-sample og out-of-sample ved å glatte over mindre variasjoner og støy som de andre modellene kan plukke opp. Selv om vi får et visuelt inntrykk av at Ridge regresjon presterer best ser vi likevel at LSTM har lavere MAPE for de fleste løpetider. Tabell 32 viser at ridge presterer bedre enn ARIMA og Random Forest for alle løpetider utenom 1 år, men dårligere enn LSTM for alle løpetider utenom 10 år.

Ridge regresjon har hatt gode resultater i studier for ulike renter. For korte og mellomlange renter tilpasser Ridge regresjon seg bedre til dataene enn andre modeller (Watson & White, 1976). Dette sammenfaller med våre resultater hvor Ridge regresjon presterer godt. I motsetning til dette viser en

tidligere forskning hvordan en variant av Ridge regresjon sammenlignes med blant annet Random Forest for amerikanske statsobligasjoner med 10 års løpetid. Dette resultatet viste at Random Forest presterte bedre enn Ridge-modellen (Luque Raya & Luque Raya, 2023) noe som er motsatt resultat enn vi ser i Tabell 32 hvor Ridge regresjon presterer bedre enn Random Forest for alle løpetider bortsett fra 1 års løpetid.

Det kan tenkes at treningsmetodene for modellene spiller en rolle for hvordan resultatet blir. Ridge regresjonsmodell og Random Forest har begge blitt bygget ved hjelp av Caret pakken i R hvor hyperparametere velges automatisk. ARIMA og LSTM har blitt bygget med manuell seleksjon av hyperparametere så her kan det forekomme menneskelige feil i større grad. I tillegg senkes terskelen etter hvert som kompleksiteten senkes og disse modellene

4.9.3. 1 års løpetid

Vi ser at de korte løpetidene presterer dårligere over tid, men det er ulikheter mellom modellene hvilke løpetider som gjør det best og verst. Vi legger merke til at 1 års løpetid for Ridge regresjon og Random Forest presterer klart dårligst av alle løpetidene. For ARIMA-modellen har heller ikke 1 års løpetid et godt resultat. Det kan virke som om modellene ikke klarer å plukke opp mønstrene for 1 års løpetid like godt som for resterende løpetider.

En mulig årsak til dette kan forklares av Swanson og Williams sine funn, at 1-års løpetid er delvis begrenset av den nedre nullgrensen og ikke reagerte like kraftig på økonomiske nyheter som de langsiktige obligasjonene (Swanson & Williams, 2014). At vi opplever dårligere prestasjon i våre modeller for 1 års løpetid kan dermed ha sammenheng med at 1 års løpetid har vært mindre responsiv til økonomiske faktorer og dermed være mindre forutsigbare.

Et annet fenomen som kan støtte opp under våre funn er Swanson og Williams (2014) som finner at effekten av penge- og finanspolitikken sannsynligvis ikke var så redusert som en gang antatt, gitt de mellomlange løpetidenes respons til nyheter (Swanson & Williams, 2014). For våre resultater kan dette være med å forklare de høye prediksjonsfeilene i 1 års løpetid for enkelte modeller. At pengepolitiske signaler og markedsforventninger ikke stemte helt overens kan kompliserer prediksjonen for 1 års løpetid. Det kan tenkes at vi burde gjort justeringer til modellene for 1 års løpetid for å ta høyde for effekten av den nedre nullgrensen på statsobligasjonsavkastninger. Dette kan være interessant å ta med seg i videre forskning.

4.9.4. Tolkning av forskningsspørsmål

Når det gjelder forskjellene mellom de tradisjonelle og de moderne tilnærmingene ser vi ut fra våre resultater at det ikke finnes noen klart svar på hvilken tilnærming som er den beste når det gjelder anvendt rentepredikSjon av amerikanske statsobligasjoner. Forskjellene ligger i hvordan modellene bygges, hvilke antagelser som ligger til grunn og anvendelse av modellene som vi har hatt en gjennomgang av i vår masteroppgave. De tradisjonelle metodene synes vi har vært vesentlig enklere å forstå og analysere enn de moderne modellene som drar med seg mer kompleksitet. Likevel finnes hjelpemidler som Caret-pakken, som tidligere nevnt i diskusjonen over, som har redusert kompleksiteten av modellbyggingen betraktelig. Vi har også lagt merke til at de tradisjonelle modellene benytter mindre datakapasitet og mindre tid på prediksjonene. Ettersom vi bemerket oss dette i modellbyggingene ga det også en forventning om at den mest komplekse modellen å forstå og bygge opp, også skulle være den modellen som presterte best. Dette stemmer i vår analyse. Likevel er det interessant å legge merke til den tradisjonelle modellen Ridge regresjon som presterer godt selv med mindre kompleksitet. Vi gjør oss derfor en tanke om at denne modellen kan være å foretrekke.

Når det gjelder forskningsspørsmål 2 har vi sett på hvordan de moderne tilnærmingene kan transformere økonomisk analyse. LSTM spesielt har som nevnt økt nøyaktighet sammenlignet med de tradisjonelle metodene og dens evne til å modellere komplekse ikke-lineære sammenhenger i økonomiske data kan bidra til å gi investorer og finansinstitusjoner tidlige og nøyaktige estimater over fremtidens økonomiske bilde. Dette mener vi kan gi utslag i at pengepolitikken justeres mer effektivt og mer presist til riktig tid. På denne måten kan man eksempelvis stimulere økonomisk vekst eller styre inflasjon under nedgangstider. Dette understøttes av funnene til Leo et al., (2019) som identifiserer maskinlæring som en av teknologiene med viktige implikasjoner for risikostyring. Vi synes det vil være interessant å følge med på hvordan videreutvikling av teknologier og algoritmer kan fortsette å forbedre nøyaktigheten i rentepredikSjon.

4.10. Konklusjon

I denne masteroppgaven har vi forsket på fire forskjellige prediktive modeller – ARIMA, Ridge regresjon, Random Forest og LSTM – med mål om å sammenligne tradisjonelle prediksjonsmodeller med moderne. Vi har benyttet datagrunnlag fra amerikanske statsobligasjoner og har gjennom evalueringer basert på historiske rentedata funnet at modellene har varierende grad av suksess med å forutsi fremtidige rentenivåer. Vår analyse har fokusert utelukkende på univariate tidsserier.

Våre forskningsspørsmål ga grunnlag for å undersøke forskjeller mellom de ulike tilnærmingene for anvendt rentepredikSjon av amerikanske statsobligasjoner, samt å se på hvordan de moderne tilnærmingene kan transformere økonomiske analyser. Forskningsspørsmål 1 mener vi å være besvart ved å presentere de ulike resultatene, samt identifisere forskjeller i oppbygning og bakgrunn.

Vår forskning viser at benchmarkmodellen ARIMA presterer markant dårligere enn de øvrige modellene, dette var for oss overraskende. Dette gjør forskningsspørsmål 2 interessant, da vi identifiserer et behov for nye og bedre metoder for rentepredikSjon. Ridge regresjon har også gitt oss et overraskende resultat sett ut fra våre forventninger, da denne har prestert bedre enn Random Forest på tilnærmet alle løpetider. Ettersom denne modellen ikke er av de som er mest forsket på innenfor rentepredikSjon kan det gi indikasjoner på at til tross for ikke-lineær data kan Ridge regresjon ved hjelp av sin straffeterm prestere godt. Med grunnlag i forskningsspørsmål 2 antar vi at de moderne tilnærmingene vil kunne gi betydelig forbedring i prediksjonsevne, men resultatene fra Ridge regresjon viser at de tradisjonelle tilnærmingene også kan gi et godt resultat.

Selv om moderne tilnærminger kan bidra til å transformere økonomiske analyser vil de også bringe med seg økt kompleksitet. Vi tenker det kanskje kan være fordelaktig å benytte moderne tilnærminger som støtte til de tradisjonelle tilnærmingene der det er aktuelt, men ut fra våre resultater ser vi at man ikke bør avskrive de tradisjonelle tilnærmingene.

Våre resultater indikerer at selv om ARIMA-modellen er en kraftfull prediksjonsmodell på mange tidsserieanalyser er den kanskje ikke optimal for løpetidene for amerikanske statsobligasjoner vi har fokusert på i vår forskning. Ridge regresjon virker å være effektiv for å predikere amerikanske statsobligasjoner for både korte, mellomlange og lange løpetider og resultatene understreker styrken til lineære modeller i rentepredikSjonssituasjoner. Random Forest virker å være anvendbar på rentepredikSjonsproblemer, selv om denne ikke ser ut til å være best i klassen, mens LSTM som forventet gir et godt resultat.

5. Implikasjoner, begrensninger og videre forskning

5.1. Implikasjoner

Tidligere forskning vi har funnet viser at ARIMA og LSTM har blitt relativt mye forsket på i sammenheng med tidsserieprediksjon, men at det ikke finnes like mange forskningsartikler som benytter Random Forest og Ridge regresjonsmodell for samme formål. Vår forståelse er at Random Forest-modellen relativt nylig ble introdusert til tidsseriemodellering og at det derfor ikke finnes like mange forskningsartikler om Random Forest som eksempelvis ARIMA, som har vært populær til tidsseriemodellering lenge. Nevrale nettverk som LSTM har også i stor grad blitt benyttet til å forske på forskjellige tema innen rentepredikSjon, og spesifikt amerikanske statsobligasjoner. Analyse av rentepredikSjonsmodeller kan gi en verdifull innsikt i effekten av pengepolitikk som styres av sentralbanker. Sammenligningen av prediksjonsmodellene kan bidra til en dypere forståelse av samspillet mellom pengepolitikk og rentemarkeder. På samme måte kan prediksjonsmodeller bidra til å forbedre prognosemodeller som igjen kan hjelpe porteføljeforvaltere med å optimalisere investeringsbeslutninger ved å justere porteføljesammensetningen i takt med forventede renteendringer. Dette vil kunne gi en bedre risikojustert avkastning. Videre kan man anvende avanserte maskinlæringsmodeller, eksempelvis som vist i vår forskning, til å fremme utviklingen av nye eller forbedrede økonomiske teorier. Dette kan gi økt forståelse av de faktorene som bestemmer rentenivåer og hvordan de varierer over tid.

Våre analysefunn, hvor vi har forsket på ulike modeller, kan være interessante for finansiell planlegging og risikostyring for finansinstitusjoner. Dette vil kunne støtte virksomhetene og investorer i å utforme strategier, håndtere gjeldsporteføljer og beskytte seg selv mot renterisiko. Nøyaktige prediksjonsmodeller har viktige roller i virksomheter som har sine primære inntekter i det finansielle markedet. Innsikt i fremtidige rentenivå og risiko er fundamentet for enhver finansinstitusjon som skal investere i obligasjoner, aksjer og eiendom. Modellene kan brukes til både kortsiktige og langsiktige beslutninger. Når det gjelder de kortsiktige kan man bruke modellen med best nøyaktighet, mens for de langsiktige beslutningene kan en sammensetting av de ulike modellene være støtte for viktige beslutninger.

5.2. Begrensninger i studien og videre forskning

For å oppsummere våre begrensninger har vi kun benyttet univariate tidsseriemodeller. Vi har også begrenset oss til å se på 3 mnd, 6 mnd, 1 år, 5 år og 10 års løpetid. For å skape mest mulig sammenlignbare modeller har vi ikke benyttet bootstrap for Random Forest, selv om vi identifiserte

at dette kunne gitt en bedre prediksjonsevne for modellen. Vi har også valgt å benytte differensiert data for Random Forest og LSTM, selv om disse kunne håndtert ikke stasjonær data. Vi har valgt å kryssvalidere alle modellene, selv om vi oppfatter at dette ikke er vanlig praksis for ARIMA og LSTM. Nøyaktighetsmålingene begrenses til MAE, RMSE og MAPE da vi mener disse gir et helhetsbilde av modellene. Vi ser likevel at det kunne vært interessant å inkludere bias, varians, MSE og flere nøyaktighetsmålinger.

Det vil være naturlig at det finnes korrelasjon mellom løpetidene som kunne vært interessant å utforske. Dette vil kunne gi økt forståelse for fremtidige renter og gi bedre grunnlag for støtte til økonomiske beslutninger. Vi kunne også tenkt oss å benytte flere datasett for å se om resultatene i vår forskning er generelle for rentepredikSjon, og om samme resultat ville dukket opp på andre data. Dette fikk vi dessverre ikke anledning til i denne omgang, men det kan være interessant å underbygge våre resultater med en slik utvidelse. Et alternativ til å benytte flere datasett kunne vært å utvide antall løpetider og kanskje inkludere flere lange løpetider som 30 års horisont.

Videre ser vi at det kunne vært interessant å ha flere modeller med i sammenligningen, og et forslag til videre forskning kan være å introdusere flere modeller. Dersom vi hadde benyttet multivariat prediksjon hadde vi benyttet VAR i vår forskning, men ettersom vi gikk for univariat mente vi dette ikke var den mest aktuelle modellen.

Forslag til videreutvikling av våre modeller er å implementere moving block bootstrap for Random Forest modellen. Dette tror vi ville gitt et bedre resultat, og kanskje hadde Random Forest prestert bedre enn ridge regresjonsmodell. For LSTM kan det være interessant å utvide grid search til å inkludere flere hyperparametere for å muligens identifisere enda bedre modeller.

Avslutningsvis kan forslag til videre forskning være å forske på sammenhengen mellom amerikanske statsobligasjoner og andre globale finansmarkeder samt påvirkningsfaktorer på disse markedene. Dette vil kunne gi implikasjoner på pengepolitikken utover våre funn.

6. Kilder

A Short Introduction to the caret Package. (U.Å.). Cran R Project. https://cran.r-project.org/web/packages/caret/vignettes/caret.html

Adrian, T., Crump, R. K. & Moench, E. (2013). Pricing the term structure with linear regressions. *Journal of Financial Economics*, *110*(1), 110–138. https://doi.org/10.1016/j.jfineco.2013.04.009

Albeladi, K., Zafar, B. & Mueen, A. (2023). Time series forecasting using LSTM and ARIMA. *International Journal of Advanced Computer Science and Applications, 14*.

Amar, S., Sudiarso, A. & Herliansyah, M. K. (2019). The Accuracy Measurement of Stock Price Numerical Prediction. *IOP Publishing*. https://doi.org/10.1088/1742-6596/1569/3/032027

Andersen, T. G. & Benzoni, L. (2010). Do Bonds Span Volatility Risk in the U.S. Treasury Market? A Specification Test for Affine Term Structure Models. *The Journal of Finance*, *65*(2), 603–653.

Anish, A. (2020, 25. november). Time Series Analysis. *The Startup*. https://medium.com/swlh/time-series-analysis-7006ea1c3326

Articles, business-science io-. (2017, 30. august). *Tidy Time Series Analysis, Part 4: Lags and Autocorrelation*. https://www.r-bloggers.com/2017/08/tidy-time-series-analysis-part-4-lags-and-autocorrelation/

Bianchi, D., Büchner, M. & Tamoni, A. (2021). Bond Risk Premiums with Machine Learning. *The Review of Financial Studies*, *34*(2), 1046–1089. https://doi.org/10.1093/rfs/hhaa062

Biau, G. & Scornet, E. (2016). A random forest guided tour. *TEST*, *25*(2), 197–227. https://doi.org/10.1007/s11749-016-0481-7

Bogdanovist. (2013, 18. februar). *Answer to «Caret and randomForest number of trees»*. Cross Validated. https://stats.stackexchange.com/a/50211

Bontempi, G., Ben Taieb, S. & Le Borgne, Y.-A. (2013). Machine Learning Strategies for Time Series Forecasting. I *Lecture Notes in Business Information Processing* (Bd. 138). https://doi.org/10.1007/978-3-642-36318-4_3

Box, G. E. P., Jenkins, G. M. & Reinsel, G. C. (1994). *Time Series Analysis: Forecasting and Control* (tredje utgave).

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Brick, J. R. & Thompson, H. E. (1978). Time Series Analysis of Interest Rates: Some Additional Evidence. *The Journal of Finance*, *33*(1), 93–103. https://doi.org/10.2307/2326352

Brownlee, J. (2021, 18. mars). Gradient Descent Optimization With Nadam From Scratch.

MachineLearningMastery.Com. https://machinelearningmastery.com/gradient-descent-optimization-with-nadam-from-scratch/

Bårsaune, O. (2017). *Framskriving av lakseprisen—Sammenligning av prediksjonskraften til State Space og ARIMA modeller* [Master thesis, NTNU]. https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2470896

Caner. (2020a, 25. mars). *Selecting LSTM Timesteps*. https://medium.com/@canerkilinc/selecting-lstm-hyperparameter-timesteps-edf27a243a9

Caner. (2020b, 25. mars). Selecting Optimal LSTM Batch Size. *Medium*. https://medium.com/@canerkilinc/selecting-optimal-lstm-batch-size-63066d88b96b

Chaudhuri, S. (2023, 26. oktober). Assessment of Accuracy Metrics for Time Series Forecasting. *Analytics Vidhya*. https://medium.com/analytics-vidhya/assessment-of-accuracy-metrics-for-time-series-forecasting-bc115b655705

Chen, Y.-L. (2023). The crucial role of the five-year Treasury in the US yield curve. *International Review of Financial Analysis*, *90*, 102828. https://doi.org/10.1016/j.irfa.2023.102828

Cheung, Y.-W. & Lai, K. S. (1995). Lag Order and Critical Values of the Augmented Dickey–Fuller Test. *Journal of Business & Economic Statistics*, *13*(3), 277–280. https://doi.org/10.1080/07350015.1995.10524601

Cholette, P. A. (1982). Prior Information and ARIMA Forecasting. *Journal of Forecasting*, *1*(4), 375–383. https://doi.org/10.1002/for.3980010405

Chowdhury, K. (2021, 25. mai). *10 Hyperparameters to keep an eye on for your LSTM model—And other tips*. Medium. https://medium.com/geekculture/10-hyperparameters-to-keep-an-eye-on-for-your-lstm-model-and-other-tips-f0ff5b63fcd4

Cohen, L., Manion, L. & Morrison, K. (2007). *Research methods in education* (6th ed). Routledge.

Components of Time Series Analysis. (2018, 8. oktober). *Toppr-Guides*. https://www.toppr.com/guides/business-mathematics-and-statistics/time-series-analysis/components-of-time-series/

De Gooijer, J. G. & Hyndman, R. J. (2006). 25 years of time series forecasting. *International journal of forecasting*, *22*(3), 443–473.

De la Calle, J. E. (2023, 9. mai). *Best Tips and Tricks: When and Why to Use Logarithmic Transformations in Statistical Analysis*. https://juandelacalle.medium.com/best-tips-and-tricks-when-and-why-to-use-logarithmic-transformations-in-statistical-analysis-9f1d72e83cfc

Dekha. (2021, 11. oktober). *Heteroscedasticity Analysis in Time Series Data*. Medium. https://python.plainenglish.io/heteroscedasticity-analysis-in-time-series-data-fee51503cc0e

Diebold, F. X. & Li, C. (2006). Forecasting the term structure of government bond yields. *Journal of*

Econometrics, *130*(2), 337–364. https://doi.org/10.1016/j.jeconom.2005.03.005

Edwards, G. (2020, 21. januar). *Machine Learning | An Introduction*. Medium. https://towardsdatascience.com/machine-learning-an-introduction-23b84d51e6d0

Enke, D. & Mehdiyev, N. (2012). *A New Hybrid Approach For Forecasting Interest Rates*. https://www.sciencedirect.com/science/article/pii/S1877050912006576

Farzad, A., Mashayekhi, H. & Hassanpour, H. (2019). A comparative performance analysis of different activation functions in LSTM networks for classification. *Neural Computing and Applications*, *31*(7), 2507–2521. https://doi.org/10.1007/s00521-017-3210-6

Frost, J. (u.å.). *How to Interpret P-values and Coefficients in Regression Analysis*. Statistics By Jim. Hentet 11. mars 2024 fra https://statisticsbyjim.com/regression/interpret-coefficients-p-values-regression/

Fuller, W. A. (2009). *Introduction to statistical time series*. John Wiley & Sons.

Galijasevic, M. (2020, 15. juni). *Deep Neural Networks with R, TensorFlow and Keras*. Authority Partners. https://authoritypartners.com/deep-neural-networks-with-r-tensorflow-and-keras/

Gers, F. A., Schmidhuber, J. & Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, *12*(10), 2451–2471. https://doi.org/10.1162/089976600300015015

Gomede, E. (2024, 2. januar). *Understanding Ridge Regression in Statistical Analysis*. Medium. https://medium.com/the-modern-scientist/understanding-ridge-regression-in-statistical-analysis-de42947ed18b

Greenwood, R., Hanson, S. G., Rudolph, J. S. & Summers, L. H. (2015). THE OPTIMAL MATURITY OF GOVERNMENT DEBT. *Brookings Institution Press*.

Grønmo, S. (2023, 26. januar). Kvantitativ metode. I *Store norske leksikon*. https://snl.no/kvantitativ_metode

Grønmo, S., Dahlum, S. & Svartdal, F. (2024, 14. februar). Validitet – Store norske leksikon. I *Store norske leksikon*. https://snl.no/validitet

Hager, S. B. (2017). *A global bond: Explaining the safe-haven status of US Treasury securities*. https://journals.sagepub.com/doi/abs/10.1177/1354066116657400

Henson, J. (2023, 27. juni). Univariate vs Multivariate Time Series Forecasting. *Medium*. https://medium.com/@jesse.henson/univariate-vs-multivariate-time-series-forecasting-cfcc4150e20a

Hochreiter, S. & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Holbrook, R. & Cook, A. (u.å.). *Time Series as Features*. Hentet 31. mars 2024 fra https://kaggle.com/code/ryanholbrook/time-series-as-features

Htoon, K. S. (2020, 29. februar). *Log Transformation: Purpose and Interpretation*. https://medium.com/@kyawsawhtoon/log-transformation-purpose-and-interpretation-9444b4b049c9

Hyndman, R. J. & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice (2nd ed)* (2nd edition). https://otexts.com/fpp2/

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

IBM. (u.å.). *What are Recurrent Neural Networks?* IBM. Hentet 3. mars 2024 fra https://www.ibm.com/topics/recurrent-neural-networks

Jain, N. (2023, 1. august). Periodogram and correlogram. *Medium*. https://medium.com/@nikitajain.jain1111/periodogram-and-corrlogram-1275364ad27f

James, G., Daniela, W., Trevor, H. & Robert, T. (2023-januar). *An Introduction to Statistical Learning with applications in R* (2nd edition). Springer.

Johannessen, A., Christoffersen, L. & Tufte, P. A. (2020). *Forskningsmetode for økonomisk-administrative fag*. https://www.akademika.no/studieteknikk/metode/forskningsmetode-okonomisk-administrative-fag/9788279354017

Johannessen, A., Tufte, P. A. & Christoffersen, L. (2016). *Introduksjon til samfunnsvitenskapelig metode* (5 utgave).

Johnson, E. (2020, 13. april). Trend Analysis. *Medium*. https://medium.com/@ethan_johnson03/trend-analysis-3a138055211

Kane, M. J., Price, N., Scotch, M. & Rabinowitz, P. (2014). Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks. *BMC Bioinformatics*, *15*(1), 276. https://doi.org/10.1186/1471-2105-15-276

Kang, H. (1986). Univariate ARIMA Forecasts of Defined Variables. *Journal of Business & Economic Statistics*, 81–86.

Khandelwal, I., Adhikari, R. & Verma, G. (2015). Time series forecasting using hybrid ARIMA and ANN models based on DWT decomposition. *Procedia Computer Science*, *48*, 173–179.

Krishnamurthy, A. & Vissing-Jorgensen, A. (2012). The Aggregate Demand for Treasury Debt. *Journal of Political Economy*, *120*(2), 233–267. https://doi.org/10.1086/666526

Kuhn, M. (u.å.). *6 Available Models*. Hentet 20. mars 2024 fra https://topepo.github.io/caret/available-models.html

Lanyi, A. & Rüsdü, S. (1983). The importance of interest rates in developing economies. *Finance and*

development.

Leo, C. (2024, 3. februar). *The Math behind Adam Optimizer*. Medium. https://towardsdatascience.com/the-math-behind-adam-optimizer-c41407efe59b

(Leo et al., 2019)Leo, M., Sharma, S. & Maddulety, K. (2019). Machine Learning in Banking Risk Management: A Literature Review. Risks, 7(1). https://doi.org/10.3390/risks7010029

Liu, Y. & Wu, J. C. (2021). Reconstructing the yield curve. *Journal of Financial Economics*, *142*(3), 1395–1425. https://doi.org/10.1016/j.jfineco.2021.05.059

Luque Raya, I. M. & Luque Raya, P. (2023). Machine learning algorithms applied to the estimation of liquidity: The 10-year United States treasury bond. *European Journal of Management and Business Economics*, *ahead-of-print*(ahead-of-print). https://doi.org/10.1108/EJMBE-06-2022-0176

Luukkonen, R., Saikkonen, P. & Teräsvirta, T. (1988). Testing Linearity in Univariate Time Series Models. *Scandinavian Journal of Statistics*, *15*(3), 161–175. JSTOR.

Macwan, V. (2021, 14. november). Standard Deviation in Statistics. *Geek Culture*. https://medium.com/geekculture/standard-deviation-in-statistics-c4165e67fdf1

Madan, P. & Madhavan, S. (2020. mars). *An introduction to deep learning*. IBM Developer. https://developer.ibm.com/articles/an-introduction-to-deep-learning/?mhsrc=ibmsearch_a&mhq=Learn%20article%20on%20Neural%20Networks

Magnimind. (2023, 8. mars). DATA WRANGLING: PREPARING DATA FOR ANALYSIS. *Medium*. https://magnimind.medium.com/data-wrangling-preparing-data-for-analysis-799954fc0b1a

Maitra, S. (2021, 23. september). Take Time-Series a Level-Up with Walk-Forward Validation. *Medium*. https://sarit-maitra.medium.com/take-time-series-a-level-up-with-walk-forward-validation-217c33114f68

Makridakis, S. & Hibon, M. (1997). ARMA Models and the Box–Jenkins Methodology. *Journal of Forecasting*, *16*(3), 147–163. https://doi.org/10.1002/(SICI)1099-131X(199705)16:3<147::AID-FOR652>3.0.CO;2-X

Mallick, A. K. & Mishra, A. K. (2019a). Interest rates forecasting and stress testing in India: A PCA-ARIMA approach. *Palgrave Communications*, *5*(1), 1–17. https://doi.org/10.1057/s41599-019-0236-7

Mallick, A. K. & Mishra, A. K. (2019b). Interest rates forecasting and stress testing in India: A PCA-ARIMA approach. *Palgrave Communications*, *5*(1), 32. https://doi.org/10.1057/s41599-019-0236-7

Mbah, T. J., Ye, H., Zhang, J. & Long, M. (2021). Using LSTM and ARIMA to Simulate and Predict Limestone Price Variations. *Mining, Metallurgy & Exploration*, *38*(2), 913–926.

https://doi.org/10.1007/s42461-020-00362-y

McCormick, L. C. & Regan, M. P. (2021, 16. mars). Why 10-Year Treasury Yields Get All the Attention. *Bloomberg.Com*. https://www.bloomberg.com/news/articles/2021-03-16/why-10-year-treasury-yields-get-all-the-attention-quicktake

Mehrotra, D. (2012). Performance comparison of time series data using predictive data mining techniques. *Advances in Information Mining*, *4*(1).

Nathaniel, J. (2021, 10. august). *Introduction to ARIMA for Time Series Forecasting*. Medium. https://towardsdatascience.com/introduction-to-arima-for-time-series-forecasting-ee0bc285807a

Nerlove, M., Grether, D. M. & Carvalho, J. L. (2014). *Analysis of economic time series: A synthesis*. Academic Press.

Ngo, S., Kemény, S. & Deák, A. (2003). Performance of the ridge regression method as applied to complex linear and nonlinear models. *Chemometrics and intelligent Laboratory systems*, *67*(1), 69–78.

Oppewal, H. (2010). Causal Research. I *Wiley International Encyclopedia of Marketing*. https://doi.org/10.1002/9781444316568.wiem02001

Park, S. & Yang, J.-S. (2022). Interpretable deep learning LSTM model for intelligent economic decision-making. *Knowledge-Based Systems*, *248*, 108907. https://doi.org/10.1016/j.knosys.2022.108907

Patel, D. (2021, 3. mars). *Ridge & Lasso Regression. Solving overfitting and underfitting*. https://deeppatel23.medium.com/ridge-lasso-regression-4272a1990aea

Patel, P. (2009). *Introduction to Quantitative Methods*.

Pereira, J. M., Basto, M. & Silva, A. F. da. (2016). The Logistic Lasso and Ridge Regression in Predicting Corporate Failure. *Procedia Economics and Finance*, *39*, 634–641. https://doi.org/10.1016/S2212-5671(16)30310-0

Phillips, P. C. B. & Perron, P. (1986). *Testing for a Unit Root in Time Series Regression*.

Probst, P. & Boulesteix, A.-L. (2017). *To tune or not to tune the number of trees in random forest?* (arXiv:1705.05654). arXiv. http://arxiv.org/abs/1705.05654

Pulver, A. & Lyu, S. (2017). LSTM with working memory. *2017 International Joint Conference on Neural Networks (IJCNN)*, 845–851. https://doi.org/10.1109/IJCNN.2017.7965940

Putka, D. J., Beatty, A. S. & Reeder, M. C. (2018). *Modern Prediction Methods: New Perspectives on a Common Problem*. https://journals-sagepub-com.ezproxy.inn.no/doi/full/10.1177/1094428117697041

Radečić, D. (2022, 23. mars). Time Series From Scratch—Train/Test Splits and Evaluation Metrics.

Towards Data Science. https://towardsdatascience.com/time-series-from-scratch-train-test-splits-and-evaluation-metrics-4fd654de1b37

Radha, S. & Thenmozhi, M. (2006). *Forecasting Short Term Interest Rates Using Arma, Arma-Garch and Arma-Egarch Models* (SSRN Scholarly Paper 876556). https://doi.org/10.2139/ssrn.876556

Saunders, M., Lewis, P. & Thornhill, A. (2009). *Research Methods for Business Students*. Pearson Education.

Scornet, E., Biau, G. & Vert, J.-P. (2015). Consistency of random forests. *The Annals of Statistics*, *43*(4), 1716–1741. https://doi.org/10.1214/15-AOS1321

Serafeim, L. (2023, 21. juli). Time-Series Forecasting: Predicting Stock Prices Using An ARIMA Model. *MLearning.Ai*. https://medium.com/mlearning-ai/time-series-forecasting-predicting-stock-prices-using-an-arima-model-627db94590e6

Shiller, R. J. & Huston McCulloch, J. (1990). The term structure of interest rates. I *Handbook of Monetary Economics* (Bd. 1, s. 627–722). Elsevier. https://doi.org/10.1016/S1573-4498(05)80016-5

Shu, L. & Chou, J.-K. (2021). Using Deep Learning Techniques to Predict 10-Year US Treasury Yield. *2021 11th International Conference on Information Science and Technology (ICIST)*, 545–552. https://doi.org/10.1109/ICIST52614.2021.9440560

Shumway, R. H. & Stoffer, D. S. (2017). *Time Series Analysis and Its Applications: With R Examples*. Springer International Publishing. https://doi.org/10.1007/978-3-319-52452-8

Siami-Namini, S., Tavakoli, N. & Siami Namin, A. (2018). A Comparison of ARIMA and LSTM in Forecasting Time Series. *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 1394–1401. https://doi.org/10.1109/ICMLA.2018.00227

Singh, A. K. (2022, 9. januar). Interpretation of Measures of Shape: Skewness & Kurtosis: *Medium*. https://ashington.medium.com/interpretation-of-measures-of-shape-skewness-kurtosis-b8b87c72c65

Speiser, J. L., Miller, M. E., Tooze, J. & Ip, E. (2019). A comparison of random forest variable selection methods for classification prediction modeling. *Expert Systems with Applications*, *134*, 93–101. https://doi.org/10.1016/j.eswa.2019.05.028

Swanson, E. T. & Williams, J. C. (2014). Measuring the Effect of the Zero Lower Bound on Medium- and Longer-Term Interest Rates. *American Economic Review*, *104*(10), 3154–3185. https://doi.org/10.1257/aer.104.10.3154

Tabova, A. M. & Warnock, F. E. (2021). Foreign Investors and US Treasuries. *NBER Working Papers*, Article 29313. https://ideas.repec.org//p/nbr/nberwo/29313.html

Thompson, S. K. (2012-februar). *Sampling*. John Wiley & Sons.

Tyralis, H. & Papacharalampous, G. (2017). Variable Selection in Time Series Forecasting Using Random Forests. *Algorithms*. https://www.mdpi.com/1999-4893/10/4/114

Vlaming, R. de & Groenen, P. J. F. (2014, 28. november). *The Current and Future Use of Ridge Regression for Prediction in Quantitative Genetics*. https://www.hindawi.com/journals/bmri/2015/143712/

Wang, X., Kang, Y., Hyndman, R. J. & Li, F. (2023). Distributed ARIMA models for ultra-long time series. *International Journal of Forecasting*, *39*(3), 1163–1184. https://doi.org/10.1016/j.ijforecast.2022.05.001

Watson, D., E. & White, K., J. (1976). Forecasting the Demand for Money under Changing Term Structure of Interest Rates. *Southern Economic Journal*, *43*. https://www-jstor-org.ezproxy.inn.no/stable/1057334

Wei, C., Wagner, L. & Lin, C. (2001). Forecasting the 30-Year US Treasury Bond with a System of Neural Networks. *Journal of Computational Intelligence in Finance*, *4*.

Westgaard, S., Pimentel, R. & Risstad, M. (2022). Predicting interest rate distributions using PCA & quantile regression. *Digital Finance*, *4*(4), 291–311. https://doi.org/10.1007/s42521-022-00057-7

Wijesinghe, S. (2020). Time Series Forecasting: Analysis of LSTM Neural Networks to Predict Exchange Rates of Currencies. *Instrumentation*, *7*, 25.

Williams, C. (2007). Research Methods. *Journal of Business & Economics Research (JBER)*, *5*(3), Article 3. https://doi.org/10.19030/jber.v5i3.2532

Wojciech, S., Grègorie, M., Lapuschkin, S., Christopher, A., J. & Müller, K.-R. (2021). Explaining Deep Neural Networks and Beyond: A Review of Methods and Applications. *IEEE*, *109*. https://ieeexplore.ieee.org/document/9369420

Wu, J. C. & Liu, Y. (2021). *Reconstructing the Yield Curve* [Data set]. https://sites.google.com/view/jingcynthiawu/yield-data

Xiao, C. (2015). *Using machine learning for exploratory data analysis and predictive models on large datasets* [Master thesis, University of Stavanger, Norway]. https://uis.brage.unit.no/uis-xmlui/handle/11250/299600

Yadav, A., Jha, C. K. & Sharan, A. (2020). Optimizing LSTM for time series prediction in Indian stock market. *Procedia Computer Science*, *167*, 2091–2100. https://doi.org/10.1016/j.procs.2020.03.257

Yamak, P. T., Yujian, L. & Gadosey, P. K. (2020). A Comparison between ARIMA, LSTM, and GRU for Time Series Forecasting. *Proceedings of the 2019 2nd International Conference on*

Algorithms, Computing and Artificial Intelligence, 49–55. https://doi.org/10.1145/3377713.3377722

Yan, H. (2001). Dynamic Models of the Term Structure. *Financial Analysts Journal*, *57*(4), 60–76. https://doi.org/10.2469/faj.v57.n4.2466

Yenigün, O. (2023, 28. februar). Skewness, Kurtosis, and Normality Test. *Python in Plain English*. https://python.plainenglish.io/skewness-kurtosis-and-normality-test-27479be7a55b

Yu, C., Qi, X., Ma, H., He, X., Wang, C. & Zhao, Y. (2020). LLR: Learning learning rates by LSTM for training neural networks. *Neurocomputing*, *394*, 41–50. https://doi.org/10.1016/j.neucom.2020.01.106

Yu Wang. (2017). A new concept using LSTM Neural Networks for dynamic system identification. *2017 American Control Conference (ACC)*, 5324–5329. https://doi.org/10.23919/ACC.2017.7963782

Yule, G., U. (1926). Why do we sometimes get nonsense-correlations between time series? A study in sampling and the nature of time series. *Journal of the Royal Statistical Society*, (89(1)), 1–63.

Zanwar, S. (2023, 2. mai). Residual Analysis in Time Series. *Medium*. https://medium.com/@ShankiiZ_/residual-analysis-in-time-series-612a450b08f5

Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, *50*, 159–175. https://doi.org/10.1016/S0925-2312(01)00702-0

Zhiguo, H. & Krishnamurthy, A. (2020). Are US treasury bonds still a safe haven? *National Bureau of Economic Research (NBER)*. https://hdl.handle.net/10419/234013

7. Appendix

7.1. Appendix A – R Script ARIMA

Rens datasettet:

```
library(readr)
library(urca)
library(xts)
library(ggplot2)
library(tidyr)
library(forecast)
library(tseries)
library(tibble)
library(dplyr)

setwd("C:\\Users\\c240185\\OneDrive - DSS AS\\Skole\\Master thesis")


my_data <- read_delim("Dataset master thesis 2.csv",
    delim = ";", escape_double = FALSE, col_types = cols(...1 = col_charac
ter()),
    trim_ws = TRUE)

## Warning: The following named parsers don't match the column names: ...1

# Sjekker hvor mange NA-verdier vi har i datasettet 'my_data'
num_na <- sum(is.na(my_data))



# Konverterer kolonnen "Dato" til "character" datatypen
my_data$Date <- as.character(my_data$Date)

#Itererer gjennom alle kolonnene og erstatter komma med punktum
for (col_name in names(my_data)[-1]) {  # [-1] ekskluderer den første kolo
nnen
  # Sjekker om kolonnen er av datatypen "character"
  if (is.character(my_data[[col_name]])) {
    # Erstatter komma med punktum i kolonnen
    my_data[[col_name]] <- gsub(",", ".", my_data[[col_name]])
    # Konverterer kolonnen til numerisk datatype
    my_data[[col_name]] <- as.numeric(my_data[[col_name]])
  }
}

#Legg til kolonne for Dato
my_data$Date <- as.Date(paste0(my_data$Date, "01"), format = "%Y%m%d")

#Gjør dataen om til tidsseriedata
ts_numeric_df <- data.frame('3m' = as.numeric(my_data$`3 m`), '6m' = as.nu
meric(my_data$`6 m`), '1y' = as.numeric(my_data$`12 m`), '5y' = as.numeric
(my_data$`60 m`), '10y' = as.numeric(my_data$`120 m`))
```

```r
xtsData <- xts(ts_numeric_df, order.by = as.Date(my_data$Date), frequency
= 12)

#Utfører Phillips Perron test på opprinnelig data for å se om dataen er st
asjonær.
pp_test_result <- PP.test(xtsData$X3m)


#Kjører ADF-test
if (!requireNamespace("tseries", quietly = TRUE)) {
  install.packages("tseries")
}

# Utfør Augmented Dickey-Fuller-testen
adf_test_result <- adf.test(xtsData$X3m)


# Utfører log-transformasjon på original tidsserie
log_transformert_data <- log(xtsData)

#Utfører tester for stasjonaritet
pp_test_log3m <- PP.test(log_transformert_data[ ,1])
adf_test_log3m <-adf.test(log_transformert_data[ ,1])
log_kpss_test_result10y <- kpss.test(log_transformert_data$X10y)

# Utfører differensiering på den log-transformerte tidsserien
differensiert_data <- diff(log_transformert_data, lag = 1)
differensiert_data <- na.omit(differensiert_data)


pp_test_diff <- PP.test(differensiert_data[ ,1])
adf_test_diff <-adf.test(differensiert_data[ ,1])

diff_kpss_test_result10y <- kpss.test(differensiert_data$X10y)

# Oppretter indekser for trening og test
total_observations <- nrow(differensiert_data)
train_index <- 1:309
test_index <- 310:total_observations


# Deler opp dataene i trenings- og testsett
train <- differensiert_data[train_index, ]
test <- differensiert_data[test_index, ]
# Lag ACF-plot
acf_plot3m <- acf(train[ ,1], lag.max = 24, main = "ACF Plot train3m")

# Lag PACF-plot
pacf_plot3m <- pacf(train[, 1], lag.max = 24, main = "PACF Plot train 3m")

# Plot ACF og PACF
par(mfrow = c(2, 1))  # Setter opp en rute med 2 rader og 1 kolonne for be
gge plottene
plot(acf_plot3m)
plot(pacf_plot3m)
```

```r
# Gjenoppretter standard oppsett
par(mfrow = c(1, 1))

auto_arima_model3m <-auto.arima(train[, 1])
auto_arima_model6m <-auto.arima(train[, 2])
auto_arima_model1y <-auto.arima(train[, 3])
auto_arima_model5y <-auto.arima(train[, 4])
auto_arima_model10y <-auto.arima(train[, 5])


# 3 måneders løpetid
arima113_3m <- arima(train[ ,1], order=c(1,1,3))

arima913_3m <- arima(train[ ,1], order=c(9,1,3))

arima2413_3m <- arima(train[ ,1], order=c(24,1,3))

arima112_3m <- arima(train[ ,1], order=c(1,1,2))



summary(arima113_3m) #AIC: -177,39

summary(arima913_3m) #AIC: -184,41

summary(arima2413_3m) #AIC: -185,8

summary(arima112_3m) #AIC: -179,26

feilmetrikker113 <- accuracy(arima113_3m)
feilmetrikker913 <- accuracy(arima913_3m)
feilmetrikker2413 <- accuracy(arima2413_3m)
feilmetrikker112 <- accuracy(arima112_3m)



aic_values3m <- c(AIC(arima113_3m), AIC(arima913_3m), AIC(arima2413_3m), A
IC(arima112_3m))

# Oppretter en tibble
arima_summary_tibble <- tibble(
  Model = c("arima113_3m", "arima913_3m", "arima2413_3m", "arima112_3m"),
  AIC = aic_values3m,
  RMSE = c(feilmetrikker113[,2], feilmetrikker913[,2], feilmetrikker2413[,
2], feilmetrikker112[,2]),
  MAE = c(feilmetrikker113[,3], feilmetrikker913[,3], feilmetrikker2413[,3
], feilmetrikker112[,3]),
  MAPE = c(feilmetrikker113[,5], feilmetrikker913[,5], feilmetrikker2413[,
5], feilmetrikker112[5])
)

# 6 måneder løpetid
arima912_6m <- arima(train[ ,2], order=c(9,1,2))
arima18121_6m <- arima(train[ ,2], order=c(18,1,21))
```

```
## Warning in arima(train[, 2], order = c(18, 1, 21)): possible convergenc
e
## problem: optim gave code = 1

arima112_6m <- arima(train[ ,2], order=c(1,1,2))

summary(arima912_6m) #AIC: -331.00

summary(arima18121_6m) #AIC: -375.59

summary(arima112_6m) #AIC: -322.66

feilmetrikker912 <- accuracy(arima912_6m)
feilmetrikker18121 <- accuracy(arima18121_6m)
feilmetrikker112_6m <- accuracy(arima112_6m)

aic_values_6m <- c(AIC(arima912_6m), AIC(arima18121_6m), AIC(arima112_6m))

# Oppretter en tibble
arima_summary_tibble_6m <- tibble(
  Model = c("arima912_6m", "arima18121_6m", "arima112_6m"),
  AIC = aic_values_6m,
  RMSE = c(feilmetrikker912[,2], feilmetrikker18121[,2], feilmetrikker112_
6m[,2]),
  MAE = c(feilmetrikker912[,3], feilmetrikker18121[,3], feilmetrikker112_6
m[,3]),
  MAPE = c(feilmetrikker912[,5], feilmetrikker18121[,5], feilmetrikker112_
6m[,5])
)



# 1 års løpetid
arima919_1y <- arima(train[ ,3], order=c(9, 1, 9))
arima11119_1y <- arima(train[ ,3], order=c(11, 1, 19))

arima202_1y <- arima(train[ ,3], order=c(2,0,2))

summary(arima919_1y) #AIC: -394,47

summary(arima11119_1y) #AIC: -394,66

summary(arima202_1y) #AIC: -391,51

aic_values_1y <- c(AIC(arima919_1y), AIC(arima11119_1y), AIC(arima202_1y))

feilmetrikker919_1y <- accuracy(arima919_1y)
feilmetrikker11119_1y <- accuracy(arima11119_1y)
feilmetrikker202_1y <- accuracy(arima202_1y)

# Oppretter en tibble
arima_summary_tibble_1y <- tibble(
  Model = c("arima919_1y", "arima11119_1y", "arima202_1y"),
  AIC = aic_values_1y,
```

```
  RMSE = c(feilmetrikker919_1y[,2], feilmetrikker11119_1y[,2], feilmetrikk
er202_1y[,2]),
  MAE = c(feilmetrikker919_1y[,3], feilmetrikker11119_1y[,3], feilmetrikke
r202_1y[,3]),
  MAPE = c(feilmetrikker919_1y[,5], feilmetrikker11119_1y[,5], feilmetrikk
er202_1y[,5])
)



#5 års løpetid
arima717_5y <- arima(train[ ,4], order=c(7,1,7))
arima13111_5y <- arima(train[ ,4], order=c(13, 1, 11))

## Warning in arima(train[, 4], order = c(13, 1, 11)): possible convergenc
e
## problem: optim gave code = 1

arima002_5y <- arima(train[ ,4], order=c(0,0,2))

summary(arima717_5y) #AIC: -663,82

summary(arima13111_5y) #AIC: -646,04

summary(arima002_5y) #AIC: -653,85

aic_values_5y <- c(AIC(arima717_5y), AIC(arima13111_5y), AIC(arima002_5y))

feilmetrikker717_5y <- accuracy(arima717_5y)
feilmetrikker13111_5y <- accuracy(arima13111_5y)
feilmetrikker002_5y <- accuracy(arima002_5y)

# Oppretter en tibble
arima_summary_tibble_5y <- tibble(
  Model = c("arima717_5y", "arima13111_5y", "arima002_5y"),
  AIC = aic_values_5y,
  RMSE = c(feilmetrikker717_5y[,2], feilmetrikker13111_5y[,2], feilmetrikk
er002_5y[,2]),
  MAE = c(feilmetrikker717_5y[,3], feilmetrikker13111_5y[,3], feilmetrikke
r002_5y[,3]),
  MAPE = c(feilmetrikker717_5y[,5], feilmetrikker13111_5y[,5], feilmetrikk
er002_5y[,5])
)



#10 års løpetid
arima212_10y <- arima(train[ ,5], order=c(2,1,2))
arima312_10y <- arima(train[ ,5], order=c(3, 1, 2))
arima215_10y <- arima(train[ ,5], order=c(2,1,5))
arima315_10y <- arima(train[ ,5], order=c(3,1,5))
arima501_10y <- arima(train[ ,5], order=c(5,0,1))

summary(arima212_10y) #AIC: -870,51
```

```r
summary(arima312_10y) #AIC: -869,74

summary(arima215_10y) #AIC: -867,2

summary(arima315_10y) #AIC: -865,3

summary(arima501_10y) #AIC: -890,98

aic_values_10y <- c(AIC(arima212_10y), AIC(arima312_10y), AIC(arima215_10y
), AIC(arima315_10y), AIC(arima501_10y))


feilmetrikker212_10y <- accuracy(arima212_10y)
feilmetrikker312_10y <- accuracy(arima312_10y)
feilmetrikker215_10y <- accuracy(arima215_10y)
feilmetrikker315_10y <- accuracy(arima315_10y)
feilmetrikker501_10y <- accuracy(arima501_10y)

# Oppretter en tibble
arima_summary_tibble_10y <- tibble(
  Model = c("arima212_10y", "arima312_10y", "arima215_10y", "arima315_10y"
, "arima501_10y"),
  AIC = aic_values_10y,
  RMSE = c(feilmetrikker212_10y[,2], feilmetrikker312_10y[,2], feilmetrikk
er215_10y[,2], feilmetrikker315_10y[,2], feilmetrikker501_10y[,2]),
  MAE = c(feilmetrikker212_10y[,3], feilmetrikker312_10y[,3], feilmetrikke
r215_10y[,3], feilmetrikker315_10y[,3], feilmetrikker501_10y[,3]),
  MAPE = c(feilmetrikker212_10y[,5], feilmetrikker312_10y[,5], feilmetrikk
er215_10y[,5], feilmetrikker315_10y[,5], feilmetrikker501_10y[,5])
)


# Kombinerer alle tibblene til én tibble
combined_arima_summary <- bind_rows(
  arima_summary_tibble %>% mutate(Series = "3m"),
  arima_summary_tibble_6m %>% mutate(Series = "6m"),
  arima_summary_tibble_1y %>% mutate(Series = "1y"),
  arima_summary_tibble_5y %>% mutate(Series = "5y"),
  arima_summary_tibble_10y %>% mutate(Series = "10y")
)


#Henter ut residualer
res3m <- checkresiduals(arima113_3m)
```

Kryssvalidering

```r
data <- rbind(train$X3m, test$X3m)
# Initialiserer start- og sluttpunkt for treningssettet
start_train <- 1
end_train_initial <- nrow(train$X3m)
end_data <- nrow(data) # Totalt antall observasjoner
```

```r
# En liste for å lagre resultater
results <- list()

# Walk forward-kryssvalidering
for (end_train in end_train_initial:end_data) {

  # Definerer treningsdata opp til nåværende punkt
  train_set <- data[start_train:end_train, , drop = FALSE]

  # Trener ARIMA-modellen på treningssettet
fit <- Arima(train_set$X3m, order=c(1,1,3))

  # Tester modellen hvis det er mer data å teste på
  if (end_train < end_data) {
    test_set <- data[(end_train + 1), , drop = FALSE]
    forecasted_values <- forecast(fit, h=24) # Forutsi den neste verdien

    # Lagrer de faktiske og forutsagte verdiene
    results[[end_train]] <- list(
      time = end_train + 1,
      actual = test_set[1, 1],
      forecast = forecasted_values$mean[1]
    )
  }
}

# Konverter resultatene til en data.frame for enklere analyse
results_df <- do.call(rbind, lapply(results, as.data.frame))
rownames(results_df) <- NULL


# Evaluerer resultatene
accuracy_results <- forecast::accuracy(results_df$forecast, results_df$X3m
)

full_data <- data.frame(time = index(differensiert_data), actual = differe
nsiert_data$X3m)
train_data <- data.frame(time = index(differensiert_data[1:nrow(train)]),
actual = train)
test_data <- data.frame(time = index(test), actual = test)

# Legger til forutsigelsene fra 'results_df' til 'full_data'
full_data$forecast <- NA  # Initialiser kolonnen med NA
full_data$forecast[start_train:end_data] <- results_df$forecast

## Warning in full_data$forecast[start_train:end_data] <- results_df$forec
ast:
## number of items to replace is not a multiple of replacement length

#In sample:
is_arima <- ggplot(train_data, aes(x = time)) +
```

```
  geom_line(aes(y = train$X3m, colour = "Actual"), size = 1) +
  geom_line(aes(y = full_data$forecast[1:nrow(train)], colour = "Forecast"
), size = 1) +
  labs(x = "Time", y = "Value", title = "Actual vs Forecasted Values") +
  scale_colour_manual("", values = c("Actual" = "blue", "Forecast" = "red"
)) +
  theme_minimal() +
  theme(legend.title = element_blank())



#Out-of-sample:
oos_arima <- ggplot(test_data, aes(x = time)) +
  geom_line(aes(y = test$X3m, colour = "Faktisk"), size = 1) +
  geom_line(aes(y = full_data$forecast[267:nrow(test)], colour = "Prediker
t"), size = 1) +
  labs(x = "Time", y = "Value", title = "Actual vs Forecasted Values") +
  scale_colour_manual("", values = c("Faktisk" = "red", "Predikert" = "gre
en")) +
  theme_minimal() +
  theme(legend.title = element_blank())
```

7.2. Appendix B – R Script Ridge regresjon

Rens datasettet:

```
library(readr)
library(urca)
library(glmnet)
library(readr)


setwd("C:\\Users\\c240185\\OneDrive - DSS AS\\Skole\\Master thesis")
my_data <- read_delim("Dataset master thesis 2.csv",
    delim = ";", escape_double = FALSE, col_types = cols(...1 = col_charac
ter()),
    trim_ws = TRUE)

# Sjekk hvor mange NA-verdier vi har i datasettet 'my_data'
num_na <- sum(is.na(my_data))
num_na

# Konverter kolonnen "Dato" til "character" datatypen
my_data$Date <- as.character(my_data$Date)


#Iterer gjennom alle kolonnene og erstatt komma med punktum
for (col_name in names(my_data)[-1]) {  # [-1] ekskluderer den første kolo
nnen
  # Sjekk om kolonnen er av datatypen "character" (tekst)
  if (is.character(my_data[[col_name]])) {
    # Erstatt komma med punktum i kolonnen
    my_data[[col_name]] <- gsub(",", ".", my_data[[col_name]])
    # Konverter kolonnen til numerisk datatype
    my_data[[col_name]] <- as.numeric(my_data[[col_name]])
  }
}


my_data$Date <- as.Date(paste0(my_data$Date, "01"), format = "%Y%m%d")
```

Gjør om til tidsseriedata og hent ut deskriptiv statistikk

```
library(xts)
library(zoo)
library(ggplot2)

library(tidyr)

ts_numeric_df <- data.frame('3m' = as.numeric(my_data$`3 m`), '6m' = as.nu
meric(my_data$`6 m`), '1y' = as.numeric(my_data$`12 m`), '5y' = as.numeric
(my_data$`60 m`), '10y' = as.numeric(my_data$`120 m`))
xtsData <- xts(ts_numeric_df, order.by = as.Date(my_data$Date), frequency
= 12)


# Utfør log-transformasjon på original tidsserie
log_transformert_data <- log(xtsData)
```

```r
# Utfør differensiering på den log-transformerte tidsserien
differensiert_data <- diff(log_transformert_data, lag = 1)

differensiert_data <- na.omit(differensiert_data)

# Opprett indekser for trening og test
total_observations <- nrow(differensiert_data)
train_index <- 1:309
test_index <- 310:total_observations

# Del opp dataene i trenings- og testsett
train <- differensiert_data[train_index, ]
test <- differensiert_data[test_index, ]

X <- model.matrix(~ . - X3m, data = differensiert_data[train_index, ])
Y <- differensiert_data$X3m[train_index]
```

legg inn modellen

```r
library(Metrics)

set.seed(123)  # For reproduserbarhet
ridge_model <- glmnet(X, Y, alpha = 0, lambda = 10^seq(10, -2, length = 10
0))

X_test <- model.matrix(~ . - X3m, data = differensiert_data[test_index, ])
Y_test <- differensiert_data$X3m[test_index]

optimal_lambda <- ridge_model$lambda.min

cv_ridge_3m <- cv.glmnet(X, Y, alpha = 0)
cv_ridge_3m_plot <- plot(cv_ridge_3m)

optimal_lambda_cv <- cv_ridge_3m$lambda.min
predictions_cv_3m <- predict(cv_ridge_3m, s = optimal_lambda_cv, newx = X_
test)



if (is.matrix(predictions_cv_3m)) {
  predictions_cv_vector <- predictions_cv_3m[, 1]
} else {
  predictions_cv_vector <- predictions_cv_3m
}

# Definerer en funksjon for MAPE
mape <- function(actual, predicted) {
  mean(abs((actual - predicted) / actual)) * 100
}

# Beregne RMSE
rmse_value_ridge_cv <- rmse(Y_test, predictions_cv_vector)
```

```r
# Beregne MAE
mae_value_ridge_cv <- mae(Y_test, predictions_cv_vector)

# Beregne MAPE
mape_value_ridge_cv <- mape(Y_test, predictions_cv_vector)

# Skriv ut nøyaktighetsmålingene
print(paste("RMSE:", rmse_value_ridge_cv))

print(paste("MAE:", mae_value_ridge_cv))

print(paste("MAPE:", mape_value_ridge_cv))

test_df <- data.frame(Date = as.Date(index(test)), X3m = test$X3m)
predictions_df <- data.frame(Date = as.Date(test_df$Date), X3m = as.vector
(predictions_cv_3m))

# Plotter både prediksjoner og faktiske verdier for testsettet med linjer
p_3m_out_of_sample <- ggplot() +
  geom_line(data = test_df, aes(x = Date, y = X3m, color = "Faktisk"), siz
e = 0.5) +
  geom_line(data = predictions_df, aes(x = Date, y = X3m, color = "Prediks
jon"), size = 1.0) +
  scale_color_manual(values = c("Faktisk" = "red", "Prediksjon" = "green")
) +
  labs(title = "Tilpasning av Ridge-regresjonsmodell til testsett",
       x = "Dato",
       y = "Rente (%)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  guides(color = guide_legend(title = "Legende"))

p_3m_out_of_sample

# Predikere in-sample med den kryssvaliderte modellen
predictions_cv_in_sample <- predict(cv_ridge_3m, s = optimal_lambda_cv, ne
wx = X)

# Konvertere in-sample prediksjoner til en vektor hvis de er returnert som
en matrise
if (is.matrix(predictions_cv_in_sample)) {
    predictions_cv_in_sample_vector <- predictions_cv_in_sample[, 1]
} else {
    predictions_cv_in_sample_vector <- predictions_cv_in_sample
}

# Opprett en dataramme for in-sample prediksjoner
in_sample_cv_df <- data.frame(Date = index(train), X3m = predictions_cv_in
_sample_vector)

# Plotting de faktiske in-sample verdiene sammen med de predikerte in-samp
le verdiene
```

```r
p_3m_in_sample_cv <- ggplot() +
  geom_line(data = train, aes(x = index(train), y = X3m, color = "Faktisk"
), size = 0.5) +
  geom_line(data = in_sample_cv_df, aes(x = Date, y = X3m, color = "Predik
sjon"), size = 1.0) +
  scale_color_manual(values = c("Faktisk" = "red", "Prediksjon" = "blue"))
+
  labs(title = "In sample tilpasning av kryssvalidert Ridge-regresjonsmode
ll",
       x = "Dato",
       y = "Rente (%)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  guides(color = guide_legend(title = "Legende"))

# Vis plottet
print(p_3m_in_sample_cv)
```

7.3. Appendix C – R Script Random Forest

Rens datasettet:

```
library(readr)
library(urca)
library(dplyr)
library(tidyr)
library(xts)
library(randomForest)
library(ggplot2)
library(tibble)
library(forecast)
library(caret)

setwd("C:\\Users\\c240185\\OneDrive - DSS AS\\Skole\\Master thesis")
my_data_rf <- read_delim("Dataset master thesis 2.csv", delim = ";",
    escape_double = FALSE, col_types = cols(Date = col_double()),
    trim_ws = TRUE)

# Sjekk hvor mange NA-verdier vi har i datasettet 'my_data'
num_na_rf <- sum(is.na(my_data_rf))

# Konverter kolonnen "Dato" til "character" datatypen
my_data_rf$Date <- as.character(my_data_rf$Date)

#Iterer gjennom alle kolonnene og erstatt komma med punktum
for (col_name in names(my_data_rf)[-1]) {  # [-1] ekskluderer den første
kolonnen

  # Sjekk om kolonnen er av datatypen "character" (tekst)
  if (is.character(my_data_rf[[col_name]])) {
    # Erstatt komma med punktum i kolonnen
    my_data_rf[[col_name]] <- gsub(",", ".", my_data_rf[[col_name]])
    # Konverter kolonnen til numerisk datatype
    my_data_rf[[col_name]] <- as.numeric(my_data_rf[[col_name]])
  }
}

my_data_rf$Date <- as.Date(paste0(my_data_rf$Date, "01"), format =
"%Y%m%d")
ts_df_rf <- data.frame('3m' = as.numeric(my_data_rf$`3 m`),
                       '6m' = as.numeric(my_data_rf$`6 m`),
                       '1y' = as.numeric(my_data_rf$`12 m`),
                       '5y' = as.numeric(my_data_rf$`60 m`),
                       '10y' = as.numeric(my_data_rf$`120 m`))

xtsData_rf <- xts(ts_df_rf, order.by = as.Date(my_data_rf$Date), frequency
= 12)

# Utfør log-transformasjon på original tidsserie
log_transformert_data <- log(xtsData_rf)
```

```r
# Utfør differensiering (lag 1 differens) på den log-transformerte
tidsserien
data <- diff(log_transformert_data, lag = 1)

ggtsdisplay(xtsData_rf$X10y)
#Plot ACF
acf_resultat <- acf(xtsData_rf$X10y, plot = TRUE, type = "correlation")
acfplot <- plot(acf_resultat)
# Funksjon for å lage lags uten å endre datoindeksen
lag_data <- function(data, n_lags, dates) {
  lags <- lapply(1:n_lags, function(lag) {
    lagged_data <- stats::lag(data, k = -lag)
    lagged_data_xts <- xts(coredata(lagged_data), order.by = dates)
    return(lagged_data_xts)
  })
  return(do.call(cbind, lags))
}

# Hent datoene fra det opprinnelige xts-objektet
dates <- index(xtsData_rf)

lags_rf <- 1:12
lagged_data_rf <- stats::lag(data$X10y, k = lags_rf )   # Oppretter lagget
data

# Kombiner lags med den opprinnelige dataen og beholder xts-strukturen
lagged_data_rf <- data.frame(lagged_data_rf)
colnames(lagged_data_rf) <- paste0("lag", lags_rf)  # Endre navn til lag
prefix.

# Flett lagged data med orginal data
final_data_rf <- cbind(data.frame(data$X10y), lagged_data_rf)

# fjerne rader med uten innhold ved lagged data
final_data_rf <- final_data_rf[complete.cases(final_data_rf), ]

names(final_data_rf)[names(final_data_rf) == "10 y"] <- "X10y"

head(final_data_rf)
# Splittet dataen i treningssett og testsett
train_percentage <- 0.7
train_size <- floor(train_percentage * nrow(final_data_rf))

train_rf <- final_data_rf[1:train_size, ]
test_rf <- final_data_rf[(train_size + 1):nrow(final_data_rf), ]
```

Validert modell

```r
set.seed(125)

# Definer kontrollparametere for kryssvalidering
trControl <- trainControl(method = "cv", number = 10)

# Tren modellen uten å spesifisere 'ntree'
validated_model_rf <- train(X10y ~ ., data = train_rf, method = "rf",
                            trControl = trControl)
```

```r
# Opprette prediksjon og evaluere RMSE
predictions_validated_rf <- predict(validated_model_rf, newdata =
test_rf)
in_sample_predictions_validated_rf <- predict(validated_model_rf,
train_rf)

# Beregn MAE for testsettet
mae_test <- mean(abs(test_rf$X10y - predictions_validated_rf))
cat("MAE på testsettet:", mae_test, "\n")
## MAE på testsettet: 0.09411745
# Beregn RMSE for testsettet
rmse_test <- sqrt(mean((test_rf$X10y - predictions_validated_rf)^2))
cat("RMSE på testsettet:", rmse_test, "\n")
# Beregne MAPE
mape_test <- mean(abs((test_rf$X10y - predictions_validated_rf) /
test_rf$X10y)) * 100
cat("MAPE på testsettet:", mape_test, "%\n")
test_df <- data.frame(Date = as.Date(index(test_rf)), X10y =
test_rf$X10y)

# Eksisterende prediksjoner fra den kryssvaliderte modellen
predictions_validated_rf <- predict(validated_model_rf, newdata =
test_rf)

# Out of sample
out_of_sample_validated_rf <- ggplot(test_rf, aes(x =
test_df$Date[1:130])) +
  geom_line(aes(y = X10y, colour = "Faktisk"), size = 1) +
  geom_line(aes(y = predictions_validated_rf, colour = "Predikert"), size
= 1) + # Grønn linje for kryssvalidert modell
  scale_colour_manual(values = c("Faktisk" = "red", "Predikert" =
"green")) +
  labs(title = "Predikert Ut-av-sample vs Faktisk Data",
       y = "Verdi",
       x = "Dato") +
  theme_minimal() +
  theme(legend.title = element_blank())

#In sample
in_sample_validated_rf <- ggplot(train_rf, aes(x =
index(xtsData_rf)[1:nrow(train_rf)])) +
  geom_line(aes(y = X10y, colour = "Faktisk"), size = 1) +
  geom_line(aes(y = in_sample_predictions_validated_rf, colour =
"Predikert"), size = 1) + # Grønn linje for kryssvalidert modell
  scale_colour_manual(values = c("Faktisk" = "red", "Predikert" = "blue"))
+
  labs(title = "Predikert In-sample vs Faktisk Data",
       y = "Verdi",
       x = "Dato") +
  theme_minimal() +
  theme(legend.title = element_blank())
```

7.4. Appendix D – R Script LSTM

Rens datasettet:

```
library(readr)
library(urca)
library(dplyr)
library(tidyr)
library(xts)
library(randomForest)
library(ggplot2)
library(tibble)
library(forecast)
library(caret)

# Sett arbeidsmappen
setwd("C:/Users/magnus.haga/OneDrive - Brommeland/Master")

library(readr)
my_data_lstm <- read_delim("Dataset master thesis 2.csv",
    delim = ";", escape_double = FALSE, trim_ws = TRUE)
# Sjekk hvor mange NA-verdier vi har i datasettet 'my_data'
num_na <- sum(is.na(my_data_lstm))

# Konverter kolonnen "Dato" til "character" datatypen
my_data_lstm$Date <- as.character(my_data_lstm$Date)

#Iterer gjennom alle kolonnene og erstatt komma med punktum
for (col_name in names(my_data_lstm)[-1]) {  # [-1] ekskluderer den første
kolonnen
  # Sjekk om kolonnen er av datatypen "character" (tekst)
  if (is.character(my_data_lstm[[col_name]])) {
    # Erstatt komma med punktum i kolonnen
    my_data_lstm[[col_name]] <- gsub(",", ".", my_data_lstm[[col_name]])
    # Konverter kolonnen til numerisk datatype
    my_data_lstm[[col_name]] <- as.numeric(my_data_lstm[[col_name]])
  }
}


my_data_lstm$Date <- as.Date(paste0(my_data_lstm$Date, "01"), format =
"%Y%m%d")

ts_df_lstm <- data.frame('3m' = as.numeric(my_data_lstm$`3 m`),
                         '6m' = as.numeric(my_data_lstm$`6 m`),
                         '1y' = as.numeric(my_data_lstm$`12 m`),
                         '5y' = as.numeric(my_data_lstm$`60 m`),
                         '10y' = as.numeric(my_data_lstm$`120 m`))

xtsData_lstm <- xts(ts_df_lstm, order.by = as.Date(my_data_lstm$Date),
frequency = 12)

# Utfør log-transformasjon på original tidsserie
```

```r
log_transformert_data <- log(xtsData_lstm)

# Utfør differensiering på den log-transformerte tidsserien
data <- diff(log_transformert_data, lag = 1)

# Funksjon for å lage lags uten å endre datoindeksen
lag_data <- function(data, n_lags, dates) {
  lags <- lapply(1:n_lags, function(lag) {
    lagged_data <- stats::lag(data, k = -lag)
    lagged_data_xts <- xts(coredata(lagged_data), order.by = dates)
    return(lagged_data_xts)
  })
  return(do.call(cbind, lags))
}

# Hent datoene fra det opprinnelige xts-objektet
dates <- index(xtsData_lstm)

lags_lstm <- 1:12
lagged_data <- stats::lag(data$X10y, k = lags_lstm )   # Oppretter lagget
data

# Kombiner lags med den opprinnelige dataen og beholder xts-strukturen
lagged_data <- data.frame(lagged_data)
colnames(lagged_data) <- paste0("lag", lags_lstm)  # Gir kolonnene nytt
navn med lag prefix

# Flett lagget data med original data
final_data <- cbind(data.frame(data$X10y), lagged_data)

#Fjern rader med NA opprettet ved opprettelse av lags
final_data <- final_data[complete.cases(final_data), ]

names(final_data)[names(final_data) == "10 y"] <- "X10y"

# Splitt skalert data inn I trenings- og testsett
train_percentage <- 0.7
train_size <- floor(train_percentage * nrow(final_data))

train_lstm <- final_data[1:train_size, ]
test_lstm <- final_data[(train_size + 1):nrow(final_data), ]

# Definer scale_data-funksjonen
scale_data <- function(train, test, feature_range = c(0,1)) {
  x_min <- min(train, na.rm = TRUE)
  x_max <- max(train, na.rm = TRUE)

  std_train <- (train - x_min) / (x_max - x_min)
  std_test <- (test - x_min) / (x_max - x_min)

  fr_min <- feature_range[1]
  fr_max <- feature_range[2]
  scaled_train <- std_train * (fr_max - fr_min) + fr_min
  scaled_test <- std_test * (fr_max - fr_min) + fr_min
```

```
  return(list(scaled_train = scaled_train, scaled_test = scaled_test,
scaler = c(min = x_min, max = x_max)))
}

# Skaler dataen
Scaled <- scale_data(train_lstm, test_lstm, c(0, 1))

time_steps <- 12

# Forbered data for LSTM-modellen
x_train <- array(Scaled$scaled_train[, 1], dim = c(nrow(train_lstm),
time_steps, ncol(train_lstm)))
y_train <- Scaled$scaled_train[, 2]

x_test <- array(Scaled$scaled_test[, 1], dim = c(nrow(test_lstm),
time_steps, ncol(test_lstm)))
y_test <- Scaled$scaled_test[, 2]

library(reticulate)
use_python("C:/Users/magnus.haga/OneDrive -
Brommeland/Master/env/Scripts/python.exe", required = TRUE)

#Opprett løkke
library(keras)
set.seed(123)

# Definer konstante verdier for hyperparametere
time_steps <- 12
unit_options <- c(100, 200)
learning_rate_option <- c(0.2, 0.3)
epoch_options <- c(50, 100)
batch_size_options <- c(32, 64, 128)
optimizer_options <- c('adam', 'nadam')

# Forbered en tom data.frame for å lagre resultatene
results <- data.frame()

# Antall tidssteg for walk-forward validering
n_forecast <- length(y_test)

# Iterer gjennom hver kombinasjon av hyperparametere
for (units in unit_options) {
  for (learning_rate in learning_rate_option) {
    for (epochs in epoch_options) {
      for (batch_size in batch_size_options) {
        for (optimizer_name in optimizer_options) {

          # Velg optimizer basert på navn
          if (optimizer_name == 'adam') {
            optimizer <- optimizer_adam(learning_rate = learning_rate)
          } else if (optimizer_name == 'nadam') {
            optimizer <- optimizer_nadam(learning_rate = learning_rate)
          }
```

```
          # Bygg modellen
          model <- keras_model_sequential() %>%
            layer_lstm(units = units, input_shape = c(time_steps,
ncol(train_lstm)), return_sequences = FALSE) %>%
            layer_dense(units = 1)

          # Kompiler modellen
          model %>% compile(
            loss = 'mse',
            optimizer = optimizer
          )

          # Tren modellen
          history <- model %>% fit(
            x_train, y_train,
            epochs = epochs,
            batch_size = batch_size,
            validation_split = 0.2
          )

          # Utfør walk-forward validering
          predictions_wf <- numeric(n_forecast)
          for(i in 1:n_forecast) {
          predictions_wf[i] <- model %>% predict(array(x_test[i, , ], dim
= c(1, dim(x_test)[2], dim(x_test)[3])))
        }

          # Beregn feilmetrikker for walk-forward validering
          mae_wf <- mean(abs(predictions_wf - y_test))
          mape_wf <- mean(abs((predictions_wf - y_test) / y_test)) * 100
          rmse_wf <- sqrt(mean((predictions_wf - y_test)^2))

          # Lagre resultatene
          results <- rbind(results, data.frame(
            Units = units,
            LearningRate = learning_rate,
            Epochs = epochs,
            BatchSize = batch_size,
            Optimizer = optimizer_name,
            MAE_WF = mae_wf,
            MAPE_WF = mape_wf,
            RMSE_WF = rmse_wf
          ))
        }
      }
    }
  }
}

# Skriv ut resultater
print(results)
```

```r
#Kode for å kun kjøre den beste modellen fra løkken

# Sett hyperparameters
learning_rate <- 0.2
units <- 100
epochs <- 100
batch_size <- 128
optimizer <- optimizer_nadam(learning_rate = learning_rate)


# Bygger LSTM Modell
model <- keras_model_sequential() %>%
  layer_lstm(units = units, input_shape = c(time_steps, ncol(train_lstm)),
return_sequences = FALSE) %>%
  layer_dense(units = 1)

# Kompillere modellen
model %>% compile(
  loss = 'mse',
  optimizer = optimizer
)

# Trene modellen
history <- model %>% fit(
  x_train, y_train,  datasets
  epochs = epochs,
  batch_size = batch_size,
  validation_split = 0.2
# Bruker en del av treningsettet for validering
)
# Plot trenings progress
history_plot <- plot(history)

# Predikere på test data
predictions <- model %>% predict(x_test, batch_size = batch_size)
## 2/2 - 1s - 610ms/epoch - 305ms/step
# Kalkulaere  MAE, MAPE, and RMSE
mae <- mean(abs(predictions - y_test))
mape <- mean(abs((predictions - y_test) / y_test)) * 100
rmse <- sqrt(mean((predictions - y_test)^2))

# Antall tidssteg for walk-forward validering
n_forecast <- length(y_test)

# Lagre forutsigelser og faktiske verdier
predictions_wf <- numeric(n_forecast)
actuals_wf <- y_test  # Dette er allerede en vektor med faktiske verdier

for(i in 1:n_forecast) {
  # Forutsi neste steg ved hjelp av det nåværende treningssettet
  predictions_wf[i] <- model %>% predict(array(x_test[i, , , drop =
```

```
FALSE], dim = c(1, ncol(x_test), ncol(x_test[1, ,])))))
}
# Beregn feilmetrikker for walk-forward validering
mae_wf <- mean(abs(predictions_wf - actuals_wf))
mape_wf <- mean(abs((predictions_wf - actuals_wf) / actuals_wf)) * 100
rmse_wf <- sqrt(mean((predictions_wf - actuals_wf)^2))

list(MAE = mae_wf, MAPE = mape_wf, RMSE = rmse_wf)

#Funksjon for å reversere skalert data
reverse_scaling <- function(scaled, scaler, feature_range = c(0,1)) {
  min = scaler[1]
  max = scaler[2]
  t = length(scaled)
  mins = feature_range[1]
  maxs = feature_range[2]
  inverted_dfs = numeric(t)

  for(i in 1:t) {
    X = (scaled[i] - mins) / (maxs - mins)
    rawValues = X * (max - min) + min
    inverted_dfs[i] <- rawValues
  }
  return(inverted_dfs)
}

# Inverter skalering for prediksjoner og faktiske verdier
predictions_inverted <- reverse_scaling(predictions_wf, scaler =
Scaled$scaler)
y_test_inverted <- reverse_scaling(y_test, scaler = Scaled$scaler)


# Opprett en data.frame for plotting
comparison_df <- data.frame(Time = index(final_data[302:431,]),
                            Actual = final_data$X10y[302:431],
                            Predicted = predictions_inverted)

# Plot sammenligningen med ggplot2
out_of_sample_lstm <- ggplot(comparison_df, aes(x = Time)) +
  geom_line(aes(y = Actual, colour = "Actual")) +
  geom_line(aes(y = Predicted, colour = "Predicted")) +
  labs(title = "out-of-Sample Predictions vs Actual Values",
       y = "Value",
       x = "Time") +
  theme_minimal() +
  scale_colour_manual(values = c("Actual" = "red", "Predicted" = "green"))
+
  theme(legend.title = element_blank())


# Generer in-sample prediksjoner for treningsdataene
in_sample_preds <- model %>% predict(x_train, batch_size = batch_size)
## 3/3 - 0s - 39ms/epoch - 13ms/step
```

```r
# Inverter skalering for de in-sample prediksjonene og de faktiske
treningsverdiene
in_sample_preds_inverted <- reverse_scaling(in_sample_preds, scaler =
Scaled$scaler)
y_train_inverted <- reverse_scaling(y_train, scaler = Scaled$scaler)

# Opprett en data.frame for plotting av in-sample resultater
in_sample_comparison_df <- data.frame(
  Time = index(xtsData_lstm[1:train_size, ]),
  Actual = y_train_inverted[1:301],
  Predicted = in_sample_preds_inverted
)

# Plot in-sample sammenligningen med ggplot2
in_sample_lstm <- ggplot(in_sample_comparison_df, aes(x = Time)) +
  geom_line(aes(y = Actual, colour = "Actual")) +
  geom_line(aes(y = Predicted, colour = "Predicted")) +
  labs(title = "In-Sample Predictions vs Actual Values",
       y = "Value",
       x = "Time") +
  theme_minimal() +
  scale_colour_manual(values = c("Actual" = "blue", "Predicted" = "red"))
+
  theme(legend.title = element_blank())
```