Kalle Hoikkala

Osakekurssien ennustaminen koneoppimisen menetelmillä

Tietotekniikan pro gradu -tutkielma

23. lokakuuta 2021

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Kalle Hoikkala

Yhteystiedot: kalle.a.hoikkala@student.jyu.fi

Ohjaaja: Ilkka Pölönen

Työn nimi: Osakekurssien ennustaminen koneoppimisen menetelmillä

Title in English: Forecasting stock prices with machine learning algorithms

Työ: Pro gradu -tutkielma

Opintosuunta: Tietotekniikka

Sivumäärä: 60+2

Tiivistelmä: Osakemarkkinoiden ennustaminen ja ennustettavuus on ollut polttava kysymys sijoittajien ja tutkijoiden keskuudessa jo vuosikymmeniä. Tekoälyn suosion kasvun myötä koneoppimisen menetelmistä on pyritty löytämään keinoja ennustamiseen. Tässä tutkielmassa tutustutaan osakemarkkinoiden ennustettavuuteen liittyvään teoriaan ja toteutetaan vertaileva empiirinen tutkimus ennusteiden välillä, jotka ovat toteutettu tunnetuilla koneoppimisen menetelmillä. Saatuja tuloksia verrataan naiiviin ennustusmenetelmään ja tulosten pohjalta pohditaan osakemarkkinoiden ennustettavuutta.

Avainsanat: koneoppiminen, tekoäly, osakemarkkinat, lstm, arima

Abstract: Stock market predictability has been a relevant topic for decades for both investors and academic researchers. The risen popularity of artificial intelligence has lead to attempts to forecast stock market using machine learning algorithms. In this thesis, we first familiarize ourselves with the relevant theory of market predictability and then conduct an empirical test comparing the performance of forecasts that are made by using known machine learning algorithms. The results are also compared to forecasts using naive forecasting mehtod. Finally we reflect stock market predictablility based on the results.

Keywords: machine learning, artificial intelligence, stock market, lstm, arima

Esipuhe

Tämä työ on ollut pitkä, mutta samalla mielenkiintoinen ja palkitseva kokemus. Kiitos loppumattomasta tuesta erityisesti vaimolleni ja ystävilleni.

Helsinki, 23. lokakuuta 2021

Kalle Hoikkala

Termiluettelo

ARIMA	Autoregressive integrated moving average, tilastollinen autoregressiivinen malli
LSTM	Long-short term memory, takaisinkytketty neuroverkkomalli
ANN	Artificial neural network, keinotekoinen neuroverkko
API	Application programming interface, ohjelmiston rajapinta
MAPE	Mean absolute percentage error, absoluuttisten virheiden prosentuaalinen keskiarvo
MAE	Mean absolute error, absoluuttisten virheiden keskiarvo
RMSE	Root mean square error, normalisoitu keskineliövirhe
NRMSE	Normalized root mean square error, normalisoitu keskineliövirhe
AIC	Akaike information criterion, Akaike infromaatiokriteeri
BIC	Bayesian information criterion, Bayesialainen infromaatiokriteeri
ACF	Auto correlation function, autokorrelaatio funktio
PACF	Partial auto correlation function, osittainen autokorrelaatio funktio
ADF	Augmented Dickey-Fuller test, laajennettu Dickey-Fuller testi
RTRL	Real time recurrent learning, laskevan gradientin optimointialgoritmi
BPTT	Backpropagation through tim, laskevan gradientin optimointialgoritmi

Kuviot

Sisältö

1 Johdanto

Sijoittajat ympäri maailmaa hakevat tuottoa omistuksilleen osakemarkkinoiden välityksellä. Itsestään selvistä syistä johtuen, sijoittajat ovat pyrkineet löytämään keinoja, joilla he voisivat kasvattaa saamia tuottojaan. Yksi merkittävää akateemistakin kiinnostusta herättänyt keino tuottojen kasvattamiseen on pyrkimys ennustaa osakemarkkinoita. Mikäli löydettäisiin keino ennustaa osakemarkkinoita ja niiden arvostuksen kehittymistä, voitaisiin saavuttaa merkittävää ylituottoa yleiseen markkinaan verrattuna.

Vallitsevan käsityksen mukaan osakemarkkinoita ei kuitenkaan pystytä ennustamaan. Nobel palkitun Eugene Faman esittämän tehokkaiden markkinoiden hypoteesin (Malkiel ja Fama 1970) mukaan osakemarkkinoiden ennustaminen on mahdotonta, sillä kaikki olemassa oleva tieto on aina sisällytetty osakkeen hintaan. Lisäksi hypoteesi yhdistetään usein satunnaiskulun teoriaan, jonka mukaan osakkeiden päivittäiset muutokset eivät ole riippuvaisia osakkeen historiallisesta kehityksestä, vaan kyseisen päivän uutisista. Tämä ei ole kuitenkaan estänyt tutkijoita ja sijoittajia yrittämästä löytää ennustamisen keinoja.

Tekoäly ja koneoppiminen ovat saavuttaneet suurta menestystä monilla aloilla, kuten puheen tunnistuksessa, konenäössä ja aikasarjan ennustamisessa. Etenkin LSTM (Long short-term memory) neuroverkkojen kyvyt löytää epälineaarisia riippuvuuksia pitkienkin aikavälien välillä ovat herättäneet toiveita myös taloudellisten aikasarjojen ja osakekurssien ennustamisen mahdollisuudesta (Eğrioğlu 2012). Myös perinteiset tilastolliset aikasarjan ennustamisen menetelmät, kuten ARIMA, ovat saavuttaneet hyviä tuloksia osakkeiden ennustamisessa etenkin lyhyellä aikavälillä (Ariyo, Adewumi ja Ayo 2014).

Tämän tutkielman avulla pyritään vastaamaan seuraaviin tutkimuskysymyksiin:

1. Voidaanko osakekursseja ennustaa koneoppimisen menetelmillä?
2. Millä algoritmilla päästään parhaaseen ennustustarkkuuteen?

Tutkielmaa varten toteutetaan vertaileva tutkimus, jossa pyritään selvittämään tunnetuilla koneoppimisen menetelmillä luotujen ennusteiden tarkkuuksia kuukauden päähän ja verrataan niitä toisiinsa, sekä perustasona pidettävään naiiviin ennusteeseen. Saatujen tulosten perus-

teella pohditaan osakkeiden ennustettavuutta ja tulosten merkitystä tehokkaiden markkinoiden hypoteesin näkökulmasta.

Tutkielma jakautuu johdannon lisäksi seitsemään lukuun. Luvussa 2 perehdytään osakemarkkinoihin, sekä vallitseviin teorioihin osakkeiden arvotukseen ja sen ennustamiseen liittyen. Luvussa 3 tarkastellaan tekoälyä ja koneoppimista, sekä tarkastellaan niitä aikasarjan ennustamisen näkökulmasta. Luvussa 4 perehdytään tarkemmin ARIMA-malliin, joka on yksi tässä tutkielmassa käytetyistä ennustusmenetelmistä. Luvussa 5 tutustutaan ensin yleisesti keinotekoisiin neuroverkkoihin, jonka jälkeen tarkastellaan tarkemmin LSTM-neuroverkkoja, jotka ovat toinen tämän tutkielman ennustusmenetelmistä. Luvussa 6 käydään läpi empiirisen tutkimuksen kuvaus ja käytettävä aineisto. Luvussa 7 esitellään empiirisen tutkimuksen tulokset ja lopuksi luvussa 8 esitellään tutkielman yhteenveto ja pohdinta.

2 Osakemarkkinat

Tässä luvussa käsitellään tutkielman kannalta tärkeää kirjallisuutta osakkeisiin ja osakemarkkinoihin liittyen. Aluksi käydään läpi yleisellä tasolla osakemarkkinoita, jonka jälkeen siirrytään tarkastelemaan osakkeiden arvostuksen määräytymistä, ja lopuksi paneudutaan osakkeiden arvostuksen ennustamiseen.

2.1 Osakemarkkinat lyhyesti

Osakemarkkinoiden historian voidaan katsoa saaneen alkunsa vuonna 1602. Tuolloin Alankomaissa järjestettiin maailman ensimmäinen listautumisanti, kun Hollannin Itä-Intian kauppakomppania perustettiin. (Petram ym. 2011)

Tuohon aikaan Euroopasta tehtiin pitkiä kauppamatkoja laivoilla Itä-Intiaan. Kauppaa varten oli perustettu Alankomaissakin yli kymmenen yksityistä yritystä, joiden välinen kilpailu johti väkivaltaisiin kohtauksiin. Vuonna 1602 Alankomaiden parlamentti päätti yhdistää nämä yksityiset yritykset Hollannin Itä-Intian kauppakomppaniaksi. Kauppakomppanian perustaminen rahoitettiin yksityisten sijoittajien kautta myymällä perustettavan yhtiön osakkeita, jotka oikeuttivat osinkohin yhtiön menestyessä. Halukkaita osakkeen ostajia oli niin paljon, ettei kaikille riittänyt osakkeita osakeannin sulkeutuessa. Tästä johtuen ihmiset, jotka jäivät ilman osakkeita, menivät Amsterdamin pörssiin ostamaan osakkeita, toisilta sijoittajilta, jotka olivat onnistuneet saamaan osakkeita osakeannista. Halukkaat ostajat olivat valmiita maksamaan jopa 14-16% enemmän osakkeelta seuraavien päivien aikana, kuin mitä alkuperäinen osakeannin hinta oli. Tämä merkittävä arvon nousu herätti mielenkiinnon spekulatiiviseen kauppaan ja kauppakomppanian osakkeilla alettiin käymään kauppaa termiinisopimuksilla eli lupauksilla ostaa tai myydä osake tiettynä aikana tulevaisuudessa tiettyyn hintaan. Tämän spekulatiivisen kaupan katsotaan olevan syynä siihen, että Amsterdamin pörssiä pidetään maailman ensimmäisenä arvopaperipörssinä. (Poitras 2016)

Osakemarkkinat ovat osa rahoitusmarkkinaa. Hämäläinen ja Oksaharju (2016) kertovat kirjassaan kolme tehtävää rahoitusmarkkinoille: Nämä ovat pääomien välittäminen säästäjiltä yrittäjille, näiden pääomien jakaminen yrittäjien kesken ja siinä samalla ne hinnoittelevat ar-

vopapereita ja muita varallisuuseriä. Näistä tehtävistä tämän tutkielman kannalta merkittävin on arvopaperien ja erityisesti osakkeiden hinnoittelu.

Osakemarkkinoilla ostajat ja myyjät käyvät kauppaa osakkeilla, jotka ovat omistusosuuksia yrityksistä. Osakkeen omistajuus antaa omistajalleen tiettyjä oikeuksia, kuten oikeuden osallistua ja äänestää yhtiökokouksessa, mutta tärkeimpänä, se oikeuttaa osuuteen yrityksen voitoista. Käytännössä tämä toteutuu siten, että voittoa tekevät yritykset voivat maksaa omistajilleen osinkoa nettovoitostaan. Sijoittaja voi siten saada tuottoa sijoitukselleen osinkojen ja/tai osakkeen arvostuksen nousun johdosta. (Frederic S. 2016)

Osakekauppaa käydään pörssissä. Pörssi on säännelty julkinen kaupankäynnin alusta, jossa voi käydä kauppaa pörssinoteerattujen yhtiöiden osakkeilla. Kaupankäynti on huutokaupan omaista, eli osakkeiden hinta määräytyy vain ja ainoastaan ostajien ja myyjien välillä. Mikäli myyjän asettama myyntihinta ja ostajan asettama ostohinta kohtaavat, tapahtuu kauppa, jossa ostaja maksaa sovitun hinnan myyjälle ja saa vastineeksi osakkeen omistuksen. Ostaja on aina se, joka on valmis maksamaan eniten osakkeesta sillä hetkellä. (Frederic S. 2016)

2.2 Osakkeiden arvostus

Osakkeiden arvonmäärityksessä voidaan käyttää monia erilaisia tapoja, eikä ole olemassa yhtä oikeaa tapaa arvottaa osakkeita. Jokainen sijoittaja voi arvottaa sijoituksiaan omien tapojen ja mieltymystensä mukaisesti, mutta aiheesta on tehty myös akateemista tutkimusta ja teorioita arvonmääritykseen. Tässä osiossa keskitytään muutamiin kirjallisuudessa tunnetuimpiin arvonmääritys menetelmiin.

2.2.1 Yleistetty osinkomalli

Williams (1938) teoksen ”The Theory of Investment Value” katsotaan olevan osakkeiden arvonmäärityksen teorian lähtöpiste. Williams (1938) esitti, että osakkeen arvo vastaa kaikkia osakkeesta tulevaisuudessa saatavien netto-osinkojen nykyarvoa. Nykyarvon määrittämisessä käytetään hyödyksi diskonttausta, jossa tulevaisuuden rahavirran eli tässä tapauksessa osinkojen nykyarvon määräämisessä huomioidaan inflaatio, sekä sijoittajan vaatima tuottotaso. Tätä arvonmääritystapaa kutsutaan yleiseksi osinkomalliksi.

Kaavana nykyarvon laskeminen voidaan ilmaista seuraavalla tavalla (Frederic S. 2016):

$$p0 = \sum_{t=1}^{\infty} \frac{D_t}{(1+k_e)^t} \text{ , jossa}$$

$$\text{p0} = \text{netto-osinkojen nykyarvo}$$

$$\text{D} = \text{maksettava osinko}$$

$$\text{t} = \text{aikajakso}$$

$$k_e = \text{tuotto-odotus osakkeelle}$$

Yleistetyssä osinkomallissa on mukana myös tuotto-odotus, jonka merkitys on yleispätevä muihinkin arvonmääritysmalleihin. Sijoittajat säätävät tuottovaatimusta arvioidun riskin avulla. Tuotto-odotus ilmaistaan prosentteina ja se kertoo minkälaista tuottoa sijoittaja odottaa sijoitukselleen saavan. Mitä vähemmän riskiä osakkeella koetaan olevan, sitä pienemmän tuotto-odotuksen sijoittaja voi sijoitukselleen antaa. (Frederic S. 2016)

2.2.2 Gordonin kasvumalli

Gordonin kasvumalli pyrki hyödyntämään yleistettyä osinkomallia, mutta samalla ottamaan paremmin huomioon yrityksen kasvat osingot (Poitras 2016). Kaavana Gordonin kasvumalli voidaan kirjoittaa muotoon.

$$P_0 = \frac{D_0(1+g)}{(k_e-g)} = \frac{D_1}{(k_e-g)} \tag{2.1}$$

, jossa D_0 on viimeisin maksettu osinko, g on odotettu kasvuprosentti osingoissa ja k_e on tuotto odotus osakkeelle. Tämä malli sisältää oletukset siitä, että osinkojen kasvu on aina samansuuruista ja osinkojen oletetaan kasvavan ikuistesti tai ainakin hyvin pitkän ajan. Lisäksi osingon kasvun oletetaan olevan pienempää kuin osakkeelle annetun tuotto-odotuksen.

Nämä edellä mainitut mallit ovat diskontatun kassavirran malleja, jotka ovat yleisesti hyväksyttäjä osakkeiden arvon määritysmalleja. Osakkeilla on siten sijoittajien silmissä ns. "oikea

hinta", joka määritetään laskennallisesti jollain arvonmääritys mallilla. Seuraavassa luvussa käydään läpi osakemarkkinoiden tehokkuuden teoriaa ja sitä, voidaanko osakkeen arvostuksen kehitystä ennustaa.

2.3 Arvostuksen ennustaminen

Osakkeiden arvostuksen ennustamisen mahdollisuus on ollut tieteellisen väittelyn kohteena jo vuosikymmeniä. Osa tutkijoista on sitä mieltä, että osakkeiden tulevaa hintaa ei pystytä ennustamaan ja toisaalta osa on sitä mieltä, että ennustamista pystytään tekemään ainakin jollain tasolla. Tässä alaluvussa käydään läpi vallitsevia näkemyksiä ja teorioita osakkeiden arvostuksen ennustamiseen liittyen.

2.3.1 Tehokkaat markkinat ja ennustamisen mahdottomuus

Finanssimarkkinoita kuvaillaan kirjallisuudessa usein tehokkaiksi. Tehokkaiden markkinoiden hypoteesi on se, että uuden tiedon tullessa esiin, se leviää markkinoilla niin nopeasti ja tehokkaasti, että se heijastuu osakkeiden hintoihin välittömästi. Tämä tarkoittaa sitä, että kaikkina hetkinä, osakkeen hinta vastaa kyseisellä hetkellä olemassa olevaa informaatiota markkinoista ja osakkeesta. (Malkiel 2003)

Tehokkaiden markkinoiden hypoteesi herätti kiinnostusta jo 1960-luvun taitteessa satunnaiskulun teorian ja järkevien odotusten teorian muodossa. Sen suosio kasvoi nopeasti hypoteesista, jonka vain muutama tutkija otti tosissaan hallitsevaksi paradigmaksi talouskirjallisuudessa (Jensen 1978). Jensen (1978) kuvaakin tehokkaiden markkinoiden hypoteesin olevan hyväksytty fakta rahoituksen, kirjanpidon ja epävarmuus talouden kirjallisuudessa.

Malkiel ja Fama (1970) esittämän tehokkaiden markkinoiden teorian mukaan tiedon välitön kulkeutuminen markkinoilla johtaa siihen, että aliarvostettujen osakkeiden löytäminen on käytännössä mahdotonta, koska markkinoiden asettama arvo osakkeelle vastaa sen todellista arvoa.

Tehokkaiden markkinoiden teoria sisältää tiettyjä olettamuksia, jotta markkinoilla oleva tieto heijastuisi välittömästi hintoihin:

1. Kaupankäynnissä ei saa aiheutua kustannuksia transaktioista.
2. Kaikki saatavilla oleva tieto on ilmaista ja se on kaikille saatavissa.
3. Kaikki markkinoilla toimijat hyväksyvät ajatuksen siitä, että nykyinen hinta vastaa täysin kaikkea saatavilla olevaa tietoa.

Malkiel ja Fama (1970) myöntävät kuitenkin, että edellä mainitut ehdot kuvaavat täydellisiä markkinoita ja eivät päde todellisilla markkinoilla. Tämä ei kuitenkaan ole heidän mukaansa vaatimus markkinoiden tehokkuudelle, vaan edellä mainittujen ehtojen odotetaan toteutuvan vain tietyiltä osin. Esimerkiksi, mikäli markkinoilla toimijat ottavat huomioon sijoituspäätöksissään transaktioista aiheutuvat suuretkin kustannukset, voivat markkinat toimia tehokkaasti kuluista huolimatta.

Tehokkaiden markkinoiden teoriassa Malkiel ja Fama (1970) jakavat markkinoiden tehokkuuden kolmeen eri kategoriaan:

1. Heikot ehdot täyttävä markkinoiden tehokkuus: Sijoittajat eivät voi saavuttaa osakkeen aiemman hintakehityksen tai tuottojen perustella normaalia suurempia tuottoja. Toisin sanoen, historiallinen data ei ole tulevien tuottojen kannalta merkityksellistä.
2. Puolivahvat ehdot täyttävä markkinoiden tehokkuus: Sijoittajat eivät voi saavuttaa normaalia suurempia tuottoja minkään julkisesti saatavilla olevan informaation perusteella. Mikään julkisesti saatavilla oleva tieto ei siten voi vaikuttaa tuleviin tuottoihin.
3. Vahvat ehdot täyttävä tehokkuus: Millään tiedolla, ei edes sisäpiirin tiedolla, pysty ansaita normaalia suurempia voittoja.

Vahvat ehdot täyttävä tehokkuuden hypoteesi katsotaan kirjallisuudessa olevan äärimmäinen muoto teoriasta ja siitä syystä sitä ei ole käsitelty muuna kuin loogisena päätöksenä mahdollisten hypoteesien joukossa. Puolivahvat ehdot täyttävä tehokkuuden hypoteesi sen sijaan katsotaan olevan vallitseva ja yleisesti hyväksytty paradigma, jota tarkoitetaan silloin, kun puhutaan tehokkaiden markkinoiden hypoteesista. (Jensen 1978)

Hommes (2001) mukaan markkinoiden tehokkuuskäsitykset voidaan jakaa ainakin kahteen eri kategoriaan: Informatiiviseen tehokkuuteen, joka tarkoittaa sitä, että markkinaa tulee olla todella vaikeaa ennustaa tai muuten se johtaa arbitraasiin, eli markkinoilla voidaan saada voittoa ilman riskiä. Toinen kategorioista on allokatiivinen tehokkuus, millä tarkoitetaan sitä,

että osakkeiden tai muiden omaisuuserien hinta vastaa niiden fundamentaalista arvoa, kuten diskontattua kassavirtaa.

Tehokkaiden markkinoiden hypoteesiin usein yhdistetyn satunnaiskulun teorian mukaan osakkeen päivittäinen hintavaihtelu johtuu vain ja ainoastaan kyseisen päivän uutisista ja hinnan muutos ei ole millään tasolla riippuvainen historian tapahtumista (Malkiel 2003). Tulevia uutisia ei pystytä ennustamaan, joten satunnaiskulun teorian mukaan myöskään osakkeen hinnan muutosta ei pystytä ennustamaan. Tästä voidaan vetää johtopäätös, että markkinoiden asiantuntijatkaan eivät saa etua markkinoista tietämättömiin sijoittajiin verrattuna, jotka hajauttavat sijoituksensa riittävällä tavalla, sillä tulevaisuuden vaihtelut ovat satunnaisia ja osakkeiden nykyinen arvostus vastaa aina niiden todellista arvoa. Malkiel (2003) vie ajatuksensa jopa niin pitkälle, että hänen mukaansa simpanssi joka, valitsee osakesalkkunsa heittämällä tikkaa Wall Street Journal -lehteen voisi saada samanlaisen tuoton osakkeilleen, kuin asiantuntijat.

Tehokkaiden markkinoiden teoria on merkityksellinen tämän tutkielman näkökulmasta, sillä sen mukaan markkinoilla osakkeiden tulevaa kehitystä ei pysty ennustamaan millään keinoilla, ei edes historiallisella hintatiedolla. Tässä tutkielmassa pyritään ennustamaan pelkän osakkeen historiallisen hintatiedon perustella osakkeen hintakehitystä ja mikäli tässä onnistutaan, on se myös merkkinä markkinoiden tehottomuudesta.

2.3.2 Markkinoita pystyy(kin) ennustamaan

Granger (1992) kertoo artikkelissaan, että vielä 1970 luvulla, jolloin tehokkaiden markkinoiden teoria sai alkunsa, tieteellinen yhteisö oli voimakkaasti yksimielinen siitä, että osakemarkkinat seuraavat satunnaiskulkua tai ainakin oli hyvin vaikeaa todistaa satunnaiskulun teoriaa epätodeksi. Granger vitsaileekin, että hän uskoi, että ainoa varma tapa ansaita rahaa osakemarkkinoilla, on kirjoittaa kirja siitä, miten ansaita rahaa osakemarkkinoilla.

Kuitenkin 1980-luvulla osakemarkkinoiden ennustettavuus nousi pinnalle ja sitä alettiin tutkimaan uusilla menetelmillä, pidemmillä ajanjaksoilla ja uusien selittävien muuttujien perusteella. Tutkimuksissa huomattiin, että markkinoita pystyy usein ennustamaan ainakin jollain tasolla. (Granger 1992)

Markkinoiden ennustettavuuden ehtona on pidetty sitä, että aiemmin julkisesti saatavilla olevalla informaatiolla on ennustavia suhteita tuleviin osaketuottoihin tai -indekseihin. Näitä tietoja voivat olla esimerkiksi taloudelliset muuttujat, kuten korot ja valuuttakurssit, toimialakohtaiset tiedot, kuten kuluttajahintojen kasvuvauhti, sekä yrityskohtaiset tiedot, kuten tuloslaskelmat ja osingon jako. Ennustettavuuden katsotaan olevan vastoin tehokkaiden markkinoiden teoriaa, sillä teorian mukaan kyseisten muuttujien sisältämä tieto on jo heijastunut täysimääräisenä osakkeisiin tai indekseihin ja millään edellä mainitulla tiedolla ei voi olla vaikutusta tuleviin hintoihin. (Enke ja Thawornwong 2005)

Balvers, Cosimano ja McDonald (1990) esittävät artikkelissaan mallin, jolla he pystyivät osoittamaan kulutuksen mahdollisuuksien ja tuotannon vaihteluiden välisen yhteyden. Kun tuotannossa tapahtuu muutoksia, kuten esimerkiksi nyt markkinoilla olevan sirupulan takia, heijastuu se kulutusmahdollisuuksiin ja sitä kautta se heiluttaa tuotteiden myyntiä. Tämä johtaa sijoittajien tuottovaatimuksen muutokseen, sillä tuotteen kulutus ei olekaan enää samalla tavalla ennakoitavissa. Tämän yhteyden vuoksi osakkeen tuottojen tulisi olla jollain tasolla ennustettavissa, jos tuotannon vakautta pystytään ennustamaan. Balvers, Cosimano ja McDonald (1990) muistuttaa kuitenkin siitä, että markkinoiden ennustettavuus ei tarkoita sitä, että ylituottojen saaminen olisi mahdollista systemaattisesti.

Lo ja MacKinlay (1988) osoittivat tutkimuksessaan, että osakemarkkinat eivät viikoittaisella aineistolla seuraakaan satunnaiskulkua, käyttämällä yksinkertaista volatiliteettiin perustuvaa määritystestiä. Tulosten perustella satunnaiskulun hypoteesi pystyttiin hylkäämään ja hylkäämismallit osoittavat, että aiempien tutkimusten stationaarisen keskiarvon palauttavat mallit eivät voi olla syynä tuottojen poikkeamiseen satunnaiskulusta.

Enke ja Thawornwong (2005) osoittaa tutkimuksessaan, että käyttämällä neuroverkkoja kaupankäynnin ohjaamisen apuna, voidaan saavuttaa suurempia tuottoja samalla riskiprofiililla, kuin muilla tunnetuilla kaupankäynnin strategioilla, kuten osta ja pidä strategialla. Enke ja Thawornwong (2005) kuitenkin muistuttavat, että tämä havainto ei suoranaisesti kumoa tehokkaiden markkinoiden hypoteesia.

3 Koneoppiminen ja ennustaminen

Tässä luvussa esitellään aluksi tekoälyn ja koneoppimisen historiaa, jonka jälkeen esitellään tarkemmin, mitä koneoppiminen on. Lopuksi perehdytään aikasarjan ennustamiseen ja käydään läpi, miten koneoppiminen ja aikasarjan ennustaminen yhdistyvät.

3.1 Tekoälyn historia

Tekoälylle ei ole yksiselitteistä yleisesti hyväksyttyä määritelmää, mutta Haenlein ja Kaplan (2019) määrittelevät, että se on järjestelmän kyky tulkita järjestelmän ulkopuolelta tulevaa dataa oikealla tavalla, oppia datan perusteella ja käyttää oppimaansa saavuttaakseen tavoitteensa.

Tekoälyn historian voidaan katsoa alkaneen vuonna 1942, jolloin tieteisfiktiokirjailija Isaac Asimov julkaisi tarinan "Runaround". Tarinassa insinöörien rakentama robotti kehittyy robotiikan kolmen lainalaisuuden mukaisesti: (1) Robotti ei saa toiminnallaan tai toimimattomuudellaan satuttaa ihmistä. (2) Robotin pitää totella ihmisen antamia ohjeita aina, ellei ne ole ristiriidassa ensimmäisen lain kanssa. (3) Robotin tulee suojella itseään ja olemassaoloaan, kunhan suojelu ei aiheuta ristiriitaa ensimmäisen tai toisen lain kanssa. Vaikka Asimovin teos olikin pelkkää tieteiskirjallisuutta, se toimi inspiraationa robotiikan tutkimukselle ja etenkin Marvin Minskylle, joka myöhemmin perusti MIT:n tekoäly laboratorion. (Haenlein ja Kaplan 2019; Nilsson 2009)

Vuonna 1950 tietokoneiden oppi-isänäkin pidetty Alan Turing julkaisi teoksen "Computing Machinery and Intelligence". Teoksessaan hän kuvaili, kuinka luoda älykkäitä koneita ja ehkä tärkeimpänä: Miten testata, että kone täyttää älykkyyden määritelmän. Turingin testiksi nimetty koe mittaa koneen ihmismäisyyttä kommunikoinnissa. Testin mukaan kone on älykäs, mikäli testin tarkkailija eli koneen kanssa keskusteleva ihminen ei pysty erottamaan koneen vastauksista, onko keskustelukumppani kone vai ihminen. Turingin testi on edelleenkin tänä päivänä käytössä koneiden älykkyyden suorituskykytestinä. (Turing 1950; Haenlein ja Kaplan 2019)

Vaikka tekoälyn historia on saanut alkunsa samoihin aikoihin ensimmäisten tietokoneiden kanssa, sen kiinnostavuus tieteellisesti ja kaupallisesti on vaihdellut merkittävästi eri ajanjaksoina. Tekoälyyn liittyviä ajanjaksoja kuvataan vuodenaikoina. Tekoälykeväänä pidetään 1940 - 1950 lukuja, jolloin ensimmäiset tekoälyyn liittyvät julkaisut ilmestyivät. Tekoälykesän aloitti vuonna 1956 järjestetty Dartmouthin konferenssi, jonka järjestäjinä toimivat Marvin Minsky ja John McCarthy. Konferenssin tavoitteena oli tutkia olettamusta, että kaikki oppimisen ja älykkyyden muodot voitaisiin määritellä niin tarkasti, että koneet pystyvät simuloimaan sitä. Tuohon konferenssiin osallistui kymmenen alan johtavaa tutkijaa ja heitä pidetään tekoälytutkimuksen oppi-isinä. Konferenssi osoittautui merkittäväksi ponnahduslaudaksi tekoälytutkimukselle ja konferenssin jälkeen seuraavan kahden vuosikymmenen aikana tekoäly tutkimus eteni harppauksittain ja se sai menestyksen myötä valtavasti rahoitusta. (McCorduck ym. 1977; Haenlein ja Kaplan 2019; Nilsson 2009)

Vuonna 1970 Minsky sanoi Life Magazine-lehden haastattelussa, että kolmen - kahdeksan vuoden päästä pystyttäisiin luomaan kone, jonka älykkyys vastaisi tavallista ihmistä. Väite osoittautui vääräksi ja tekoäly tutkimukseen käytetty rahoitus alkoi saada kritiikkiä osakseen Yhdysvalloissa ja Isossa Britanniassa. Brittiläinen matemaatikko James Lighthill kyseenalaisti tekoälytutkijoiden optimistisen näkemyksen tekoälyn kyvykkyydestä raportissaan vuonna 1973. Hänen mielestään koneet eivät kykenisi koskaan saavuttamaan kokenutta amatööriä korkeampaa tasoa peleissä, kuten shakki, eivätkä koneet pystyisi koskaan yleiseen järkevään päättelykykyyn. Lighthillin raportin perusteella Iso-Britannian hallitus perui tekoälytutkimuksen rahoituksen suurimmaksi osaksi ja Yhdysvallat seurasi Iso-Britannian esimerkkiä pian perästä. Tätä hetkeä pidetään ensimmäisen tekoälytalven aloituksena. (Haenlein ja Kaplan 2019)

Yksi merkittävä syy Minskyn väitteiden epäonnistumiselle ja tekoälytutkimuksen paikalleen jämähtämiselle johtui tavasta, jolla ihmisen älykkyyttä pyrittiin jäljentämään. Tuohon aikaan menestyksekkäimmät tekoälyjärjestelmät olivat niin kutsuttuja "Asiantuntija järjestelmiä". Niiden toiminta perustui oletukseen, että ihmisen älykkyys pystytään formalisoida ja rakentaa sääntöjen ja päättely ohjeiden mukaisesti käyttämällä esimerkiksi peräkkäisiä if-else lauseita. Näin ollen ne pystyvät ratkaisemaan ainoastaan sellaisia ongelmia, jotka asettuvat niille etukäteen koodattujen ohjeiden malliin. Esimerkkinä tällaisesta järjestelmästä on

vuonna 1996 julkaistu IBM:n kehittämä Deep Blue shakkiohjelma, joka tuli kuuluisaksi siitä, että se pystyi voittamaan vuonna 1997 silloisen shakin maailmanmestarin Garry Kasparovin. Deep Bluen menestys perustui siihen, että se pystyi laskemaan jopa 200 miljoonaa siirtoa sekunnissa (Campbell, Hoane Jr ja Hsu 2002). Laskentatehoa hyödyntäen se tutki eri siirtojen variaatioita 20 siirtoa eteenpäin ja valitsi optimaalisimman siirron sen perusteella. (Haenlein ja Kaplan 2019)

Nilsson (2009) kuvaa ensimmäisen tekoälytalven jälkeistä aikaa kukoistuksen ajaksi, jolloin perustettiin useita tekoälyyn liittyviä yrityksiä, sekä etenkin asiantuntijajärjestelmien suosio nousi huippuunsa. Tuona ajanjaksona perustettiin myös tekoälyyn keskittynyt AAAI-järjestö (American Association for Artificial Intelligence), jonka jäsenmäärä kasvoi yli 16 000:een vuoteen 1987 mennessä. Tämän jälkeen kuitenkin suosio tekoälyä kohtaan alkoi hiipua, kun tekoälytutkimus ei pystynytkään vastaamaan lupauksiinsa ja tavoitteisiinsa, aloittaen toisen tekoälytalven. Vuoteen 1996 mennessä tekoälytutkimuksen rahoitus pieneni merkittävästi ja AAAI:n jäsenmäärä tipahti alle 5000 jäseneen.

Toisen tekoälytalven aikana useat tutkijat madalsivat tekoälytutkimuksen tavoitteita saavutettavammalle tasolle. Tutkijat pyrkivät siirtämään keskustelun pois siitä, mitä tekoälyn kautta voidaan tulevaisuudessa saavuttaa siihen, mitä tekoälyllä pystyttäisiin saavuttamaan tällä hetkellä. Tämä johti muutokseen ajatusmallissa, että tekoälyn katsottiin olevan ihmisiä avustava työkalu, eikä ihmisen kokonaan korvaava ratkaisu. Myös rahoitus siirtyi nykyisellään käytössä olevien asioiden, kuten tietokantojen, käyttöliittymien, tietoverkkojen, konenäön ja tiedonlouhinnan kehittämiseen. (Nilsson 2009)

Nykyistä hetkeä Haenlein ja Kaplan (2019) kuvaa tekoälyn syksyksi ja sadonkorjuun ajaksi, jolloin pääsemme nauttimaan aiempien kausien hedelmistä. Etenkin viime vuosina kiinnostus tekoälyä kohtaan on noussut hyvinkin merkittäväksi keinotekoisten neuroverkkojen ja syväoppimisen edistysten myötä. Syväoppiminen ja neuroverkot muodostavatkin hyvin pitkälti nykyisen käsityksen tekoälystä ja ne toimivat puheen- ja kuvantunnistusohjelmien, älykaiuttimien ja itsestään ajavien autojen takana.

3.2 Koneoppiminen

Tieteen näkökulmasta koneoppiminen on yksi merkittävimmistä tekoälytutkimuksen haaroista, ja se sijoittuu jonnekin tietotekniikan ja tilastotieteiden välimaastoon. Koneoppiminen on noussut nopeasti tietotekniikan laboratorioista kaupallisesti käytetyiksi ratkaisuiksi muun muassa puheentunnistuksen ja konenäön sovelluksissa. (Jordan ja Mitchell 2015)

Koneoppimisen perusperiaate on se, että koneoppimisalgoritmi pystyy syötetyn datan perusteella tunnistamaan ja luokittelemaan datassa olevia piirteitä ja oppimaan niistä ilman, että sille erikseen eksplisiittisesti kerrotaan, kuinka sen tulisi toimia (Jordan ja Mitchell 2015; Nilsson 2009). Esimerkkinä koneoppimisalgoritmin toiminnasta Nilsson (2009) kertoo, että mikäli suuri datajoukko sisältää useita tapauksia, joissa joutsen on valkoinen, eikä ollenkaan tapauksia, joissa joutsen olisi muun värinen, kuin valkoinen, voisi koneoppimisalgoritmi tehdä päätelmän, että kaikki joutsenet ovat valkoisia. Päätelmä on luonteeltaan induktiivinen, eli se voi osoittautua virheelliseksi uuden datan perusteella, mutta se edustaa kuitenkin parasta päätelmää, joka voidaan saatavilla olevan datan perusteella tehdä.

Jordan ja Mitchell (2015) mukaan koneoppiminen alana pyrkii vastaamaan kahteen kysymykseen: Kuinka voidaan luoda tietokone järjestelmä, joka pystyy parantamaan suoriutumistaan kokemuksen perusteella. Ja mitkä ovat oppimiseen liittyvät lainalaisuudet, jotka vaikuttavat oppimiseen niin ihmisillä, kuin koneilla. Näihin kysymyksiin vastaamalla on voitu viedä teoriaa käytäntöön ja nykyaikaisissa ohjelmistoissa on käytössä koneoppimista jo laajalla skaalalla.

Automaattinen datan kerääminen ja tiedon tallennuksien edullisuus on johtanut siihen, että saatavilla olevaa dataa on hyvin paljon. Suurten datamäärien prosessointi ja päätösten tekeminen sen perusteella on ihmisille hidasta ja haastavaa, mutta koneoppimisalgoritmit puolestaan hyötyvät suurestakin datan määrästä. Koneoppimisen avulla pystytäänkin automatisoimaan näitä ihmisille hitaita prosesseja. (Jordan ja Mitchell 2015)

Yksi merkittävimmistä aiheista koneoppimisen ja tämän tutkielman kannalta, on keinotekoiset neuroverkot. Neuroverkot ovatkin taustalla suuressa osassa koneoppimisen algoritmeja. Neuroverkot toimivat siten, että ne vastaanottavat syötteenä dataa muuttujista ja tuottavat datan perusteella tulosteen kyseisen datan perusteella (Kwon 2011). Neuroverkkoja ja niiden

sovelluksia on monia erilaisia ja niistä kerrotaan tarkemmin kappaleessa 5.1.2

Vaikka tilastollisia menetelmiä, kuten tässä tutkielmassa käytetty ARIMA ei usein lasketakaan koneoppimisen piiriin, niiden on havaittu toimivan erityisen hyvin aikasarjojen ennustamisessa (George E. P. ym. 2016). Tästä johtuen, tässä tutkielmassa koneoppimisen algoritmeista puhuttaessa, lasketaan mukaan myös tilastolliset ennustamisen menetelmät, kuten edellä mainittu ARIMA.

3.3 Aikasarjan ennustaminen

Aikasarjalla tarkoitetaan havaintojen sarjaa, jotka on tehty ajallisesti peräkkäisessä järjestyksessä. Suuri osa tietoaineistoista (engl. dataset) on aikasarjoja, kuten päivittäiset sateen määrät, kuukausittaiset myynnit tai viikkokohtaiset osakekurssit. Luontainen ominaisuus aikasarjoilla on se, että tyypillisesti vierekkäisten havaintojen arvot ovat riippuvaisia toisistaan. Tämä vierekkäisten havaintojen riippuvaisuus suhde on aikasarja-analyysin näkökulmasta huomattavan kiinnostuksen kohteena ja aikasarja analyysi keskittyykin tuon riippuvuus suhteen tarkasteluun. (George E. P. ym. 2016)

George E. P. ym. (2016) mukaan aikasarjan ennustaminen on yksi osa aikasarjan analyysin kokonaisuutta, jossa pyritään aiempien havaintojen perusteella ennustamaan tulevaisuuden arvoja. Ennustamisessa oletetaan, että tehdyt havainnot ovat ajallisesti erillisiä ja tasaisesti jakautuneita. Ennusteita voidaan hyödyntää muun muassa talouden ja yritystoiminnan, tuotannon ja varastonhallinnan suunnitteluun.

Ennustusfunktio voidaan kirjoittaa matemaattiseen muotoon $\hat{z}_t(l)$, jossa z on ennustettava muuttuja, t on aika alkupisteessä t ja l on tehdyn ennusteen ajankohta tulevaisuudessa. Ennustusfunktio $\hat{z}_t(l)$ tuottaa ennusteet alkupisteestä t kaikille tulevaisuuden ajankodille l käyttämällä hyödyksi nykyistä ja aiempia arvoja $z_t, z_{t-1}, z_{t-2}, \ldots$. Tavoitteena on löytää ennustusfunktio, jossa keskineliöpoikkeamat (engl. mean square deviation) $z_{t+1} - \hat{z}_t(l)$ todellisten ja ennustettujen arvojen välillä ovat mahdollisimman pienet jokaisella ennusteen ajankohdalla l.(George E. P. ym. 2016)

Eğrioğlu (2012) esittää kirjassaan useita eri menetelmiä aikasarjan ennustamiseen. Ennen

1920-lukua aikasarjojen ennusteet laskettiin yksinkertaisesti ekstrapoloimalla aikasarjaa. Vuonna 1927 Yule esitti autoregressiiviset ennustamisen tekniikat, joihin muun muassa luvussa 4 tarkemmin tarkasteltava ARIMA perustuu. Yulen työn katsotaankin olevan modernin ennustamisen perustana. 1980-luvulla tietokoneiden laskentatehon kehittymisen ja koneoppimisen saavutusten myötä neuroverkkoihin perustuvat ennustusmenetelmät yleistyivät.

Neuroverkkojen on havaittu olevan erityisen hyviä tunnistamaan epälineaarisia elementtejä aikasarjoista, niiden ei lineaarisen luonteen vuoksi (Eğrioğlu 2012). Tämä erottaa ne tilastollisista menetelmistä, kuten ARIMA:sta, jotka pystyvät tunnistamaan pelkästään lineaarisia elementtejä aikasarjasta. Näin ollen neuroverkoilla ja ARIMA:lla tehdyt ennusteet voivat poiketa merkittävästi toisistaan.

4 ARIMA

ARIMA-mallit ovat aikasarjojen ennustamiseen yksiä käytetyimmistä menetelmistä. ARIMA:n suosio perustuu sen tilastollisiin ominaisuuksiin sekä hyvin laajasti tunnettuun Box-Jenkins menetelmään ARIMA-mallia rakentaessa (Zhang 2003). Tässä luvussa tutustutaan ensin tarkemmin ARIMA-malleihin, niiden luomiseen Box-jenkin menetelmän avulla sekä aiempiin tutkimuksiin, joissa ARIMA:a on hyödynnetty osakekurssien ennustamisessa.

4.1 ARIMA yleisesti

Autoregressiiviset mallit (AR) esiteltiin ensimmäisen kerran jo vuonna 1926 Yulen toimesta. Vuonna 1937 ne saivat täydennystä Slutskyltä, kun hän kehitti Liukuvan keskiarvon (MA) mallit. Wold puolestaan oli ensimmäinen, joka päätti yhdistää AR ja MA mallit vuonna 1938 ja todisti, että ARMA prosesseilla pystytään mallintamaan suuria stationaarisia aikasarjoja, kunhan mallille pystytään asettamaan oikeat parametrit. Käytännössä tällä tarkoitetaan sitä, että aikasarja y_t voidaan mallintaa olevan lineaarinen funktio useista edeltävistä arvoista ja satunnaisisita virheistä. (Makridakis ja Hibon 1997; Zhang 2003) Aikasarjan muodostuminen voidaan siten kirjoittaa muotoon:

$$y_t = \theta_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} \tag{4.1}$$

$$+\varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - ... - -\theta_q \varepsilon_{t-q} \tag{4.2}$$

jossa y_t ja ε_t ovat todellinen arvo ja satunnainen virhe ajankohdassa t. Lisäksi ϕ_i (i = 1,2,..., p) ja θ_j (j = 0,1,2,...,q) ovat mallin parametreja. Satunnaisvirheiden ε_t oletetaan olevan identtisesti ja itsenäisesti jakaantuneet, niiden keskiarvo on nolla ja niillä on jatkuva varianssi σ^2. (Zhang 2003)

Woldin teoreettisia löydöksiä ei kuitenkaan pystytty hyödyntämään ennen kuin vasta 1960-luvulla, jolloin tietokoneiden laskentateho oli riittävä yhtälön 4.1 ja 4.2 parametrien optimoimiseksi. Box ja Jenkins esittelivät vuonna 1970 käytännöllisen menetelmän ARMA mallien käyttöön ja Box-Jenkins menetelmästä ja ARIMA malleista tulikin erittäin suosittu akatee-

misen tutkimuksen kohde, koska se pystyi suoriutumaan paremmin, kuin monimutkaisemmat ja suuremmat ekonometriset mallit. (Makridakis ja Hibon 1997)

ARIMA (Auto regressive integrated moving average) koostuu kolmesta osasta, jotka ovat AR(p), I(d) ja MA(q). ARIMA malleja kuvataankin tästä syystä seuraavalla tavalla: ARIMA(p,d,q), jossa p,d ja q ovat kokonaislukuja. Mikäli, joku näistä arvoista (p, d, q) on 0, sitä ei tarvitse käyttää mallissa. Näin ollen ARIMA malli yksinkertaistuu esimerkiksi ARMA malliksi silloin, kun d arvo on 0.

Autoregressiivisess mallissa (p) ennustetaan tulevaisuuden arvo edeltävien arvojen lineaarikombinaationa (Hyndman ja Athanasopoulos 2018). Tämä vastaa yllä tehdyn matemaattisen mallinnuksen 4.1 yhtälön osuutta.

Liukuvan keskiarvon (q) osuudessa keskitytään edeltävien arvojen sijaan edeltävien ennusteiden virheisiin. Jokainen arvo y_t voidaan ajatella olevan painotettu liukuvakeskiarvo muutamasta edellisestä ennustusvirheestä. Tätä mallia ei kuitenkaan pidetä regressiivisenä, vaikka sen toiminta onkin sen kaltaista. (Hyndman ja Athanasopoulos 2018) Matemaattisesti tämä vastaa 4.2 yhtälön osuutta.

Integrated (d) tarkoittaa aikasarjan muuntamista stationaariseen muotoon. Aikasarjan stationaarisuus on edellytys ARIMA:n käyttöön, sillä ei stationaarisen aikasarjan perusteella tehdyt ennusteet eivät ole käyttökelpoisia. Stationaarisuus tarkoittaa sitä, että aikasarjassa ei ole havaittavissa trendiä tai kausittaista vaihtelua ja sen tilastolliset arvot, kuten keskiarvo pysyvät samoina ajan saatossa. (Hyndman ja Athanasopoulos 2018; Zhang 2003; Maggi 2018)

Tämän lisäksi, mikäli aikasarjalla havaitaan olevan kausittaista vaihtelua tulee sekin ottaa huomioon, jolloin mallin muoto on SARIMA(p,d,q) x (P,D,Q), jossa P,D ja Q vastaavat mallin kausiluonteisia vastineita arvoille p,d,q. (Hyndman ja Athanasopoulos 2018)

4.2 Box-Jenkins menetelmä

Box ja Jenkins (1970) kehittivät käytännöllisen menetelmän ARIMA-mallien parametrien selvittämiseen. Menetelmä koostuu kolmesta vaiheesta, joita toistetaan iteratiivisesti lopul-

lisen, hyvän mallin löytämiseksi. Menetelmän vaiheet ovat mallin tunnistaminen, mallin arviointi ja mallin toimivuuden tarkastus. (Zhang 2003; Makridakis ja Hibon 1997; George E. P. ym. 2016)

4.2.1 Mallin tunnistaminen

Mallin tunnistamisen tavoitteena on löytää jotkut arvot p,d ja q ARIMA:lle. Mallin tunnistaminen voidaan jakaa kahteen osaan, jotka ovat: Aikasarjan muuttaminen stationaariseksi sekä kausittaisuuden tunnistaminen (1) ja p ja q arvojen tunnistaminen (2). (George E. P. ym. 2016)

Suurin osa luonnollisesti esiintyvistä aikasarjoista on ei stationaarisia (Maggi 2018). Mikäli aikasarja ei ole stationaarinen, se tulee differentioida ennen ARIMA mallissa käyttöä. Differentiointi tarkoittaa uuden aikasarjan luontia aiemman perusteella siten, että lasketaan peräkkäisten arvojen erotus:

$$y_t^{'} = y_t - y_{t-1} \tag{4.3}$$

Differentioidussa aikasarjassa on T-1 arvoa, sillä ensimmäiselle arvolle ei pystytä suorittamaan yllä olevaa laskutoimitusta. Joskus yksi differentiointi kerta ei ole riittävä stationaarisuuden saavuttamiseksi ja silloin differentiointi voidaan suorittaa uudelleen. (Hyndman ja Athanasopoulos 2018)

Pelkästään aikasarjaa tarkastelemalla ei yleensä pystytä päättelemään, onko aikasarja stationaarinen tai, mitkä ovat aikasarjan kohdalla otolliset ARIMA:n p ja q arvot. Nämä pystytään kuitenkin joskus päättelemään käyttämällä hyväksi autokorrelaatiofuntiota (ACF, autocorrelation function) ja osittaisautokorrelaatiofunktiota (PACF, partial autocorrelation function). (Hyndman ja Athanasopoulos 2018; George E. P. ym. 2016).

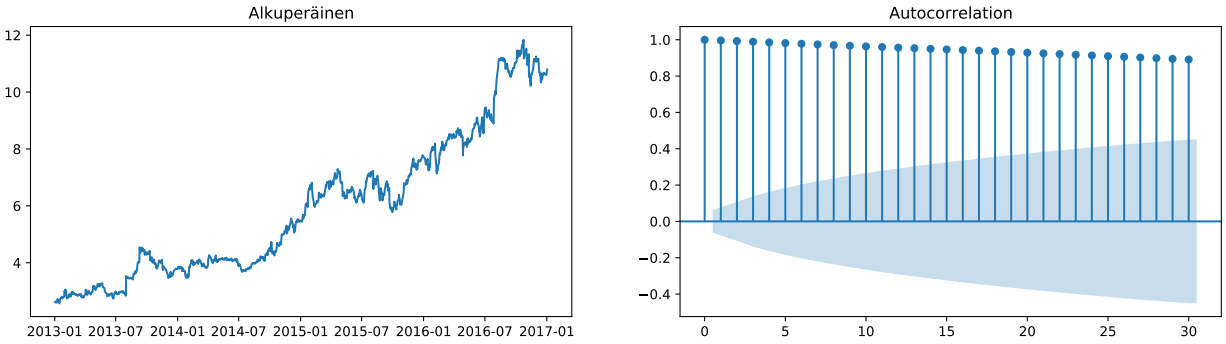
ACF- graafit osoittavat aikasarjan arvojen autokorrelaation, mikä tarkoittaa arvojen y_t ja y_{t-k} välistä riippuvuus suhdetta. Mikäli y_t ja y_{t-1} välillä on havaittavissa korrelaatiota, niin silloin myös y_{t-1} ja y_{t-2} täytyy korreloida. Tämä ei kuitenkaan tarkoita sitä, että y_{t-2} ja y_t välillä olisi automaattisesti korrelaatiota, vaikka sitä saattaakin olla. Jotta y_{t-2} ja y_t väli-

tä korrelaatiota ei tarvitsisi arvuutella, voidaan käyttää PACF- funktiota, jossa samoin, kuin ACF -funktiossa, mitataan autokorrelaatiota y_t ja y_{t-k} välillä, mutta siitä poistetaan viiveiden 1,2,..,k-1 vaikutus. PACF ja ACF saavat ensimmäisen arvon kohdalla siten aina saman tuloksen, sillä ensimmäisen arvon kohdalla ei ole viivettä, jota voisi poistaa. (Hyndman ja Athanasopoulos 2018)

Aikasarjan stationaarisuus voidaan päätellä ACF-kuvaajan muodosta. Ei stationaarisen aikasarjan kohdalla ACF laskee hitaasti kohti nollaa, kun stationaarisen aikasarjan kohdalla ACF laskee nollan lähelle nopeasti. (Hyndman ja Athanasopoulos 2018; George E. P. ym. 2016)

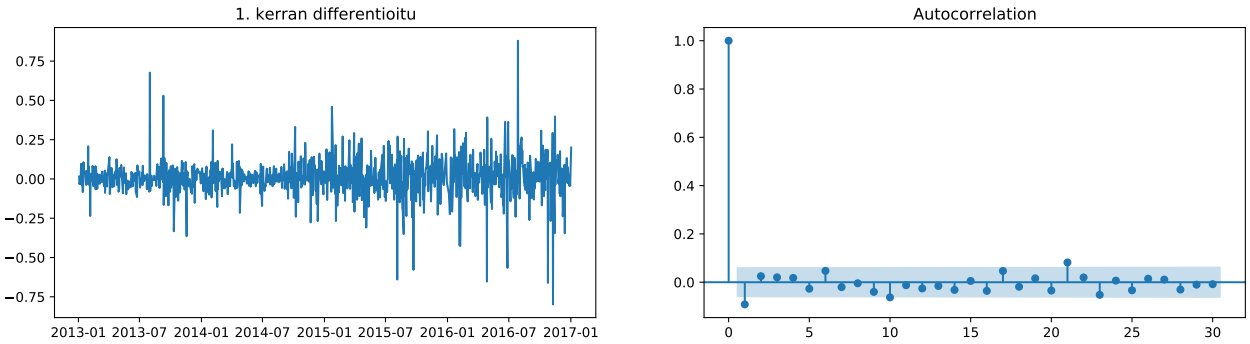
Kuvio 1 osoittaa alkuperäisen aikasarjan, joka ei ole stationaarinen, ACF:n hidasta laskeutumista kohti nollaa, kun taas kuvio 2 osoittaa yhden kerran differentioidun aikasarjan ACF:n nopeaa laskeutumista. Kuvioissa vasemman puolen kuvat osoittavat osakekursseja ja oikeanpuoleiset kuvat ovat ACF-kuvaajia. ACF-kuvaajissa pystyakseli osoittaa riippuvuuden suuruutta ja vaaka-akseli osoittaa, kuinka monen askeleen päässä olevaa riippuvuutta tarkastellaan.

Kuvio 1. ACF-kuvaaja ei stationaariselle aikasarjalle



ACF- ja PACF-graafeista pystytään joissain tapauksissa päätellä stationaarisuuden lisäksi ARIMA-mallin p ja q arvotkin. Mikäli ACF:n kohdalla arvo laskee piikin (p) jälkeen vaiheittaisesti lähelle nollaa ja PACF-graafissa nähdään kyseisen piikin (p) kohdan jälkeen suora pudotus, on malli autoregressiivinen: ARIMA(p,0,0). Sama periaate toimii toisin päin, eli mikäli PACF-graafissa on vaiheittainen putoaminen ja ACF-graafissa pudotus on piikin (q) jälkeen välitön, on malli liukuvan keskiarvon mukainen: ARIMA(0,0,q). Mikäli molemmissa ACF- ja PACF-graafeissa on havaittavissa vaiheittainen putoaminen piikkien jälkeen, on

Kuvio 2. ACF-kuvaaja stationaariselle aikasarjalle



kyseessä ARMA malli: ARIMA(p,0,q). (Hyndman ja Athanasopoulos 2018; George E. P. ym. 2016)

Toinen suosittu aikasarjan stationaarisuuden selvittämisen keino on laajennettu Dickey-Fullerin testi (ADF, Augmented Dickey-Fuller test). ADF-testi tutkii, löytyykö tutkittavalle aikasarjalla yksikköjuurta. Mikäli testistä löytyy yksikin yksikköjuuri, voidaan aikasarjaa pitää ei-stationaarisena. (Maggi 2018) ADF-testi voidaan kirjoittaa matemaattisesti muotoon:

$$\Delta x_t = \mu + \gamma_t + \alpha x_{t-1} + \sum_{j=1}^{k-1} \beta_j \Delta x_{t-j} + \varepsilon_t, \tag{4.4}$$

jossa x on aikasarja, Δ kuvaa eroavaisuutta, μ on vakio, γ kuvaa ajallisten suuntausten kerrointa, α on prosessin kerroin, jonka negatiivisuutta testissä tutkitaan. ε_t kuvaa satunnaista virhettä regressiokertoimella t. (Cheung ja Lai 1995)

Vaihtoehtoinen tapa ACF- ja PACF-graafeille ARIMA:n p ja q arvojen selvittämiseksi on käyttää informaatiokriteereitä, kuten Akaike (AIC) - tai Bayesilaista (BIC) informaatiokriteeriä. Informaatiokriteereitä käytettäessä luodaan useita mahdollisia ARMA(p,q) malleja ja niiden toimivuutta arvioidaan suurimman uskottavuuden menetelmien avulla laskemalla AIC tai BIC arvo. Informaatiokriteerifunktiot käyttävät arvioinnissa suurimman uskottavuuden estimaattia, josta kerrotaan tarkemmin kappaleessa 4.2.2. Paras malli löytyy minimoimalla informaatiokriteerin arvo. (George E. P. ym. 2016; Hyndman ja Athanasopoulos 2018)

$$AIC_{p,q} = \frac{-2ln(\text{maximized likelihood}) + 2r}{n} \approx ln(\hat{\sigma}_a^2) + r\frac{2}{n} + \text{constant} \tag{4.5}$$

$$BIC_{p,q} = ln(\hat{\sigma}_a^2) + r\frac{ln(n)}{n} \tag{4.6}$$

, jossa $\hat{\sigma}_a^2$ on suurimman uskottavuuden estimaatti arvosta σ_a^2 ja $r = p + q + 1$ on arvioitujen parametrien määrä, sisältäen vakion, jossa p vastaa auroregressiivisyyden arvoa, q vastaa liukuvan keskiarvon arvoa. Lisäksi n kuvaa otoskokoa. Funktioissa 4.5 ja 4.6 ensimmäinen termi $ln(\hat{\sigma}_a^2)$ siis vastaa suurimman uskottavuuden laskemista, josta tarkemmin kerrotaan kappaleessa 4.2.2 ja toinen termi $r\frac{2}{n}$ ja $r\frac{ln(n)}{n}$ toimii rangaistuskriteerinä, joka lisätään ylimääräisten parametrien lisäämisestä malliin. (George E. P. ym. 2016)

4.2.2 Mallin arviointi

Toinen vaihe Boxin-Jenkins menetelmässä on mallin arviointi, joka tarkoittaa valitun mallin parametrien arviointia yleisesti pienimmän neliösumman menetelmällä (Least squares method) tai suurimman uskottavuuden estimoinnilla (Maximum likelihood estimation). Parametreina tarkoitetaan tässä tapauksessa funktion 4.1 ja 4.2 arvoja $\phi_1, ..., \phi_p, \theta_0, ..., \theta_q$. Parametrien arvioinnin tavoitteena on löytää parametrit, joilla saadaan virheiden määrä mahdollisimman pieneksi.(Hyndman ja Athanasopoulos 2018; Zhang 2003)

Legendren ja Gaussin työhön perustuva pienimmän neliösumman menetelmä on yksi vanhimmista nykyaikanakin käytetyistä tilastotieteen menetelmistä. Pienimmän neliösumman menetelmä laskee mallille optimaalisimmat parametrit p ja q arvojen tiedoilla. Parametrit lasketaan pyrkimällä minimoimaan ennusteiden ja todellisten arvojen erotuksien neliöiden summa. (Maggi 2018; Hyndman ja Athanasopoulos 2018) Tämä voidaan esittää matemaattisessa muodossa:

$$\sum_{t=1}^{T} \varepsilon_t^2 \tag{4.7}$$

Suurimman uskottavuuden estimoinnissa (MLE) etsitään parametreja, joilla data olisi muodostunut uskottavimmin. MLE:n käyttö esiteltiin ensimmäisen kerran vuonna 1922 Fisherin

toimesta ja se perustuu hänen vuonna 1912 esittelemään numeeriseen prosessiin. (Maggi 2018)

Käytännössä MLE toimii siten, että alkuperäisestä aikasarjasta $\mathbf{y} = y_1, ..., y_n$ luodaan ARIMA(p,d,q) mallin avulla uusi aikasarja $\mathbf{x} = x_1, ..., x_n$. Parametrien joukkoa kuvaa $N = \{\phi_1, ..., \phi_p, \theta_1, ..., \theta_q\}$ ja yhdistetty tiheysfunktio (joint propability density function) on:

$$f(x_n, x_{n-1}, ...x_1; N) \tag{4.8}$$

Uskottavuusfunktio saadaan siitä, kun yhdistetty tiheysfunktio ajatellaan olevan parametrien N funktio datalle x:

$$L(N|x) = f(x_n, x_{n-1}, ...x_1; N) \tag{4.9}$$

Ja suurimman uskottavuuden estimointi kirjataan muotoon:

$$\widehat{N} = \arg\max \; L(N|x(n)), \quad N \in \Theta, \tag{4.10}$$

jossa Θ on mahdollisten parametrien avaruus. Termi arg max tarkoittaa sitä parametrien joukkoa, jolla funktion lopputulema on mahdollisimman suuri. Hyvin usein käytetään helpomman laskutavan vuoksi L:n luonnollista logaritmia uskottavuusfunktiona L:n sijaan ja sillä päästään kuitenkin samaan lopputulokseen, sillä molemmat L ja $\ln(L)$ saavuttavat maksimi arvon samaan aikaan. (Maggi 2018)

Tämä mallin arvioinnin vaihe toteutetaan yleisesti ohjelmallisesti, eli se ei vaadi käyttäjän omaa tarkastelua ja tämä vaihe toteutetaan siitä syystä täysin automatisoidusti. (Zhang 2003)

4.2.3 Mallin toimivuuden tarkastus

Sen jälkeen, kun malli ARIMA(p,d,q):n arvot p,d ja q on päätetty ja mallille parhaimmat parametrit $\phi_1, ..., \phi_p, \theta_0, ..., \theta_q$ on löydetty, voidaan tehdä mallin toimivuuden tarkastus. Mallin toimivuuden tarkastuksen tavoitteena on selvittää, onko löydetty malli tarkoituksen mukai-

nen kyseiselle aikasarjalle. Mikäli malli ei ole tarkoituksen mukainen, tulee selvittää millä tavoin malli ei ole sopiva, jotta voidaan suorittaa muutokset malliin seuraavaa iteraatiota varten. (George E. P. ym. 2016)

Yksi käytetyimmistä keinoista on jäännösarvojen tarkastelu. Jäännösarvoilla tarkoitetaan mallinnuksen ulkopuolelle jääneitä arvoja, jotka jäävät jäljelle sovittamisprosessin jälkeen. Useimmille aikasarjoille se tarkoittaa alkuperäisen aikasarjan arvojen ja mallinnetun aikasarjan arvojen erotusta: (Hyndman ja Athanasopoulos 2018; Zhang 2003)

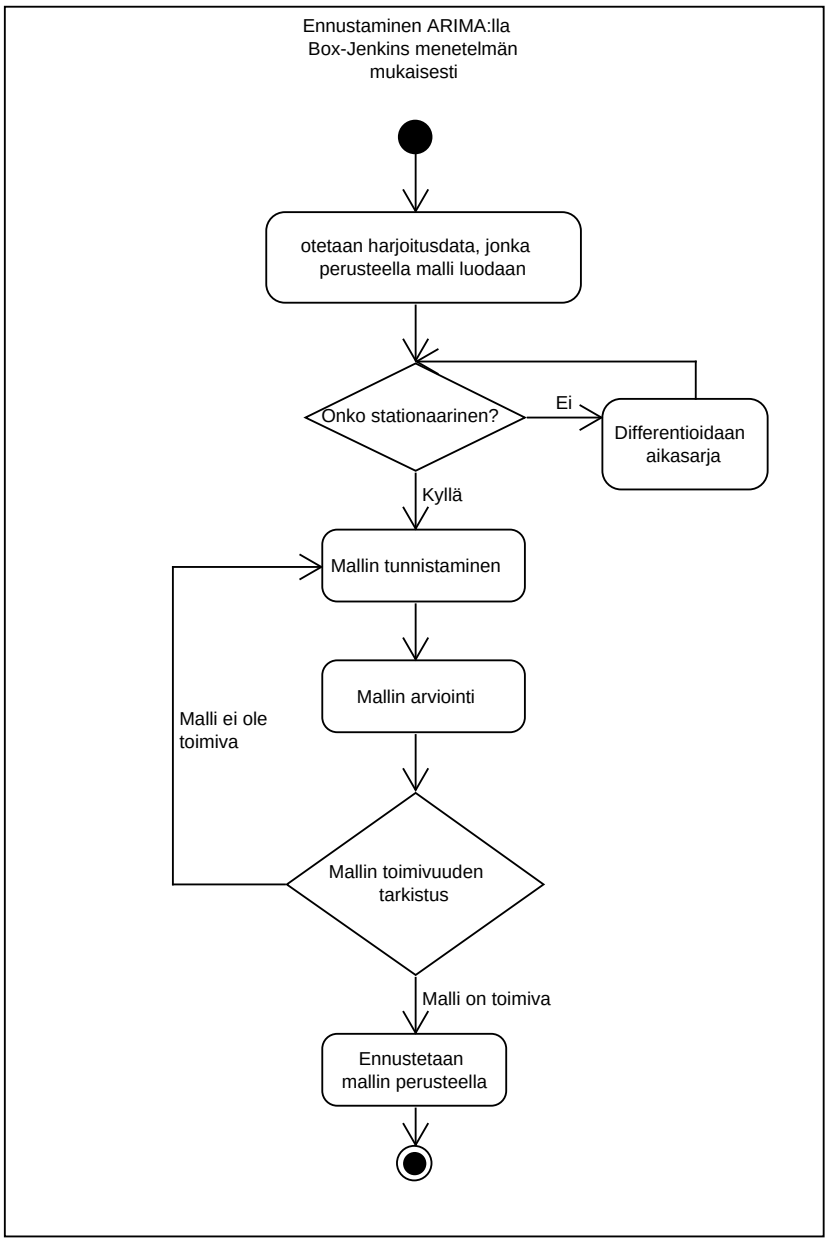
$$e_t = y_t - \hat{y}_t \tag{4.11}$$

Jäännösarvojen tulisi olla satunnaisia eli toisistaan riippumattomia, jotta malli voidaan hyväksyä. Jäännösarvojen keskiarvon tulee myös olla 0, sillä muuten mallin avulla luodut ennusteet tulevat olemaan vinoutuneita. Mikäli jäännösarvot eivät ole satunnaisia, se tarkoittaa sitä, että mallissa ei ole otettu huomioon kaikkia korrelaatioita ja sitä voidaan parantaa uudella iteratiivisella Box-Jenkins menetelmän kierroksella. (Hyndman ja Athanasopoulos 2018; Makridakis ja Hibon 1997)

Tärkeää on kuitenkin muistaa, että mitkään mallit eivät ole tarkkoja kuvauksia todellisuudesta, vaan ne ovat arvioita siitä. Tämän vuoksi jotkut mallit voidaan virheellisesti hylätä toimivuuden tarkastelussa, koska ne eivät läpäise tämän vaiheen testejä, vaikka todellisuudessa ne olisivat riittävän hyviä käytettäväksi ennustamiseen. Samalla tavalla mallit, joissa on selviä puutteita, saattavat läpäistä testit esimerkiksi liian pienen otoskoon vuoksi. Tästä johtuen mallin toimivuuden tarkastelun vaiheessa on järkevintä käyttää mahdollisimman tarkoituksen mukaisia testejä mallin soveltuvuuden tarkistamiseksi, mutta olla samalla valmis käyttämään malleja, jotka eivät testejä täydellisesti läpäisisikään. (George E. P. ym. 2016)

Kuviossa 3 esitetään ennustaminen ARIMA:n avulla Box-Jenkins menetelmän mukaisesti aikasarjalle. Kuviosta on yksinkertaistuksen takia erotettu stationaarisuuden tarkastelu omaksi vaiheekseen, ennen mallin tunnistusta. Todellisuudessa tämä vaihe kuuluu mallin tunnistamiseen.

Kuvio 3. ARIMA ennusteen luomisprosessi Box-Jenkins menetelmän mukaisesti



4.3 ARIMA-mallin tutkimuksia osakekurssien ennustamisesta

ARIMA-mallia on hyödynnetty osakekurssien ennustamiseen laajalti tieteellisissä tutkimuksissa. Osakekurssien ennustamista on pidetty erityisen haasteellisena sen monimutkaisen luonteen vuoksi ja tämän vuoksi erilaisia ennustamisen malleja on pyritty kokeilemaan pa-

rempien ennusteiden toivossa. ARIMA mallien on havaittu olevan tehokkaita ja vakaita etenkin osakkeiden lyhyen ajan ennustamisessa. (Ariyo, Adewumi ja Ayo 2014)

Ariyo, Adewumi ja Ayo (2014) tekivät tutkimuksen osakekurssien lyhytaikaisesta ennustamisesta ARIMA-mallia hyödyntäen. Tutkimuksessa tutkittiin ARIMA:n ennusteiden tarkkuutta Nokian ja Zenith:in osakkeiden kohdalla. ARIMA mallia valittaessa käytettiin Box-Jenkins menetelmää ja useita eri arviointikriteereitä parhaimman mallin löytämiseksi. Loppupäätelmänä oli se, että ARIMA pystyi ennustamaan osakkeita vähintäänkin tyydyttävällä tarkkuudella.

Mondal, Shit ja Goswami (2014) tutkivat ARIMA:n ennustamisen tarkkuutta 60 intialaisella osakkeella. He loivat ennustista ARIMAN avulla 30 päivän päähän eri mittaisilla harjoitusdata jakosilla. He saavuttivat myöskin tyydyttäviä tuloksia ennusteissaan.

Devi, Sundar ja Alli (2013) tutkivat ARIMA:n käyttöä osana sijoitus suositusta vertailemalla eri indeksejä ja ARIMA:n ennusteiden virheitä indekseissä. Indeksit, joissa virheet olivat keskimääräistä pienempiä, on voimakkaampi korrelaatio aiempien ja tulevien hintojen välillä ja siitä syystä kyseistä indeksiä voidaan suuremmalla luottamuksella suositella asiakkaille.

5 LSTM

Tässä kappaleessa esitellään ensin keinotekoiset neuroverkot (ANN) ja niiden teoreettinen tausta, josta siirrytään tarkastelemaan Long short-term memory (LSTM) neuroverkkoa ja sen erityispiirteitä. Lopuksi käsitellään aiempaa kirjallisuutta, jossa LSTM- neuroverkkoja on käytetty ennustamaan osakekursseja.

5.1 Keinotekoiset neuroverkot yleisesti

Keinotekoiset neuroverkot saivat alkunsa 1950-luvulla, kun tieteellinen yhteisö pyrki ymmärtämään ihmisen aivojen toimintaa. Biologinen neuroverkko koostuu hermosoluista eli neuroneista, jotka ovat linkittyneet toisiinsa synapsien avulla. Yksi neuroni voi vastaanottaa viestejä useilta muilta neuroneilta ja yksi neuroni pystyy myös välittämään vastaanottamansa signaalit useille muille neuroneille. Kaikki yhteyksiä ei kuitenkaan painoteta samalla tavalla eli kaikki vastaanotetut viestit (input) eivät välity sellaisenaan kaikille yhdistetyille neuroneille. Tämä viestien kulku ja eri painotukset neuronien välisissä yhteyksissä toimii pohjana myös keinotekoisille neuroverkoille. (Daniel 2013)

Keinotekoisten neuroverkkojen perusoletukset luotiin vuonna 1943 McCullochin ja Pitts:in toimesta (Daniel 2013). Oletuksia on viisi ja usein lasketaan mukaan Hebbin sääntö kuudenneksi ja siitä kerrotaan lisää kappaleessa 5.1.2

1. Neuronin aktivaatio on binäärinen, kaikki tai ei mitään.
2. Neuroni vaatii enemmän kuin yhden aktivoidun synapsin annetun aikaikkunan sisällä, jotta neuroni aktivoituu.
3. Ainut viive neuroverkossa syntyy synapseissa.
4. Mikä tahansa estävästä synapsista tullut aktivaatio estää täydellisesti neuronin aktivaation kyseisellä hetkellä.
5. Neuroverkon struktuuri ei muutu ajan saatossa.
6. Neuronit oppivat muuttamaan painotuksiaan Hebbin säännön mukaisesti.

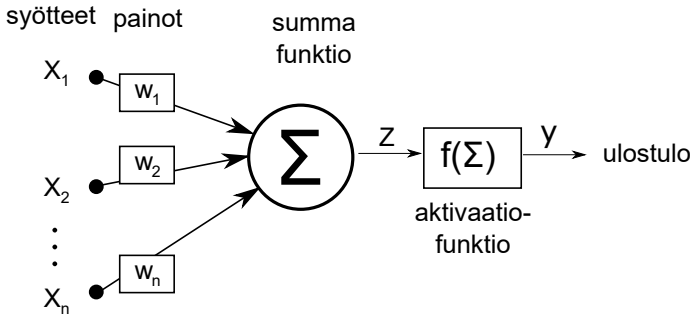
Modernit keinotekoiset neuroverkot eivät noudata kaikkia yllä mainittuja oletuksia, kuten

vaatimusta neuronin aktivaation binäärisyydestä. Nämä oletukset ovat kuitenkin ensimmäisiä systemaattisia periaatteita keinotekoisille neuroverkoille. (Kwon 2011)

5.1.1 Neuronin toiminta

Keinotekoinen neuroverkko koostuu neuroneista, jotka ovat linkittyneet toisiinsa muodostaen verkkomaisen rakenteen. Neuronien toiminta on suoraviivaista siinä mielessä, että ne vastaanottavat syötteen tai syötteitä, syötteiden perustella neuronissa lasketaan aktivoitumistaso, joka välitetään verkossa eteenpäin seuraaville neuroneille, jotka ovat linkittyneet kyseiseen neuroniin. Aktivaatiotason laskeminen eli neuronin sisäinen toiminta sisältää kaksi vaihetta: Summafunktion ja aktivaatiofunktion. (Daniel 2013; Kwon 2011) Kuviossa 4 on esitettynä neuronin rakenne ja toiminta.

Kuvio 4. Biologiseen neuroniin perustuva keinotekoinen neuroni



$$z = \sum_n w_n x_n \tag{5.1}$$

$$y = f_N(z) \tag{5.2}$$

Neuronien väliset suhteet saavat erilaisia painokertoimia, eli kaikkien syötteiden arvo summafunktiossa 5.1 ei ole sama. Syötteiden tyyppejä on kahta erilaista: Inhibitorinen eli ehkäisevä syöte saa negatiivisen painokertoimen ja voimistava (excitatory) syöte saa positiivisen painokertoimen. Syötteet ja niiden painokertoimet summataan summafunktiossa. Summafunktion tulos syötetään aktivaatiofunktiolle 5.2, jonka perusteella lasketaan neuronin aktivaatio. Lopuksi aktivointifunktion tulos aktivaatiosta välitetään tulosteena eteenpäin.

Aktivaatiofunktio, toiselta nimeltään siirtofunktio, on usein epälineaarinen funktio, jossa määritetään neuronin tulosteen arvo. Aktivaatiofunktiossa suositaan yleensä epälineaarisia funktioita, jotta neuronin tulosteen arvot voidaan pitää tiettyjen rajojen sisällä. Aktivaatiofunktioita on useita erilaisia ja niistä yleisimmin käytettyjä ovat *sigmoidi*-funktio 5.3 ja tiukka rajainen (hard limit) -funktio 5.4 (Kwon 2011; Daniel 2013)

$$y = \frac{1}{1 + exp^{-z}} \tag{5.3}$$

$$y = \begin{cases} 1 & \text{jos } z \geq 0 \\ 0 & \text{jos } z < 0 \end{cases} \tag{5.4}$$

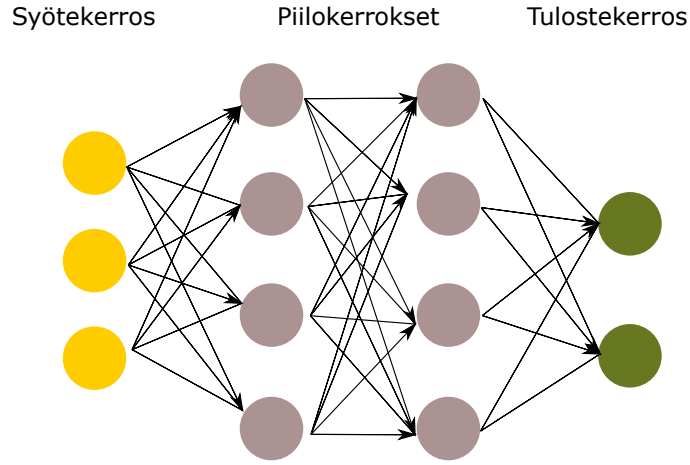
5.1.2 Neuroverkot

Kaikki keinotekoiset neuroverkot koostuvat vähintään kolmesta peräkkäisestä kerroksesta. Ensimmäinen kerros on syötekerros, joka vastaanottaa itsenäiset muuttujat, jotka toimivat neuroverkon syötteenä. Toisena kerroksena on piilokerros, joka sisältää verkon neuronit, jotka eivät ole millään tavoin kontaktissa verkon ulkopuolen kanssa. Piilokerroksia voi olla yksi tai useampia. Viimeisenä on tulostekerros, jossa neuronit laskevat neuroverkon lopulliset tulokset ja lähettävät ne eteenpäin verkon ulkopuolelle. Syötekerroksella olevien neuronien määrä vastaa yksittäisten neuroverkkoon syötettävien muuttujien määrää ja tulostekerroksella olevien neuronien määrä vastaa ennustettavien muuttujien määrää. (Kwon 2011; Abraham 2005)

Kuvio 5 esittää feed-forward neuroverkkoa, jossa kolmen muuttujan perusteella pyritään ennustamaan kahta muuttujaa. Feedforward eli eteenpäinkytketty neuroverkko on neuroverkkomalli, jossa kaikki neuronit, jotka ovat samassa neuroverkon kerroksessa, on kytketty jokaiseen neuroverkon seuraavassa kerroksessa olevaan neuroniin. (Kwon 2011)

Jotta neuroverkko voi tuottaa järkeviä tuloksia, se tulee konfiguroida siten, että annetut syötteet tuottavat halutun joukon tuloksia. Konfiguroinnilla tarkoitetaan syötteiden painokertoimien w_i muuttamista joko suoraan ennakkotietojen perusteella tai syöttämällä neuroverkolle opetusdataa, jonka perusteella se voi itse muuttaa painokertoimia datan perusteella. (Abra-

Kuvio 5. Yksinkertainen feed-forward neuroverkko



ham 2005)

Neuroverkkojen opettamiseen yleisimmin käytettyjä keinoja ovat valvottu oppiminen (supervised learning), valvomaton oppiminen (unsupervised learning) ja vahvistusoppiminen (reinforcement learning. Valvotussa oppimisessa neuroverkolle kerrotaan etukäteen, mitkä tulosteet verkon tulee saada annetuilla syötteillä. Toisin sanoen neuroverkon harjoitusdata koostetaan siten, että syötteet ja halutut tulosteet ovat neuroverkolla tiedossa samanaikaisesti ja se pystyy muokkaamaan painokertoimia vertaamalla saavutettuja tuloksia ja odotettuja tuloksia. Valvottua oppimista käyttävien järjestelmien tavoitteena on ekstrapoloida tai yleistää vastauksia, jotta se pystyy toimimaan tilanteissa, jotka eivät vastaa harjoitusdataa. Valvottua oppimista käytetään hyödyksi etenkin luokittelevissa neuroverkoissa. (Abraham 2005; Fausett 1994; Sutton ja Barto 2018)

Valvomattomassa oppimisessa neuroverkolle ei anneta tiedoksi, mikä tuloste annetuilla syötteillä pitäisi saada, vaan sen datana toimii pelkästään luokittelematon syöte joukko. Neuroverkolle ei myöskään erikseen syötetä harjoitusdataa, vaan se kykenee tekemään ennusteensa suoraan raa'alle syötedatalle. Valvomattomassa oppimisessa neuroverkon tehtävänä on luokitella samankaltaiset syötteet ryhmiin eli klustereihin, jonkin ominaisuuden perusteella, ilman ulkopuolista apua. Valvomatonta oppimista hyödynnetään neuroverkoissa, joilla pyritään ryhmittelemään dataa. (Abraham 2005; Fausett 1994; Sutton ja Barto 2018)

Vahvistusoppiminen on luonteeltaan hyvin erilaista, kuin valvottu - ja valvomaton oppimi-

nen. Sutton ja Barto (2018) mukaan vahvistusoppimista käyttäessä kuuluu selvittää "Mitä pitäisi tehdä? - Miten tilanteet linkitetään toimintoihin? - Jotta voidaan maksimoida numeerinen palkintosignaali."Vahvistusoppimista käyttävä järjestelmä ei lähtötilanteessa tiedä, mitä tehdä ja kuinka sen tulisi toimia, vaan sen tulee itse selvittää kokeilemalla, mitkä toiminnot ovat sille hyödyllisiä palkintosignaalin perusteella. Tämänkaltaisia ongelmia kutsutaan suljetun-silmukan ongelmiksi, sillä jokainen tehty toiminto vaikuttaa tuleviin syötteisiin ja palkintoihin vielä monen askeleen jälkeenkin. Nämä ominaisuudet ovat vahvistusoppimien tunnistettavimmat ominaisuudet.

Oppiminen neuroverkon sisällä tapahtuu aina jonkin oppimissäännön mukaisesti (Abraham 2005). Yksi tunnetuimmista ja yleisimmin käytetyistä säännöistä on Hebbin sääntö. Hebbin sääntöä voidaan kuvata Pavlovin koiran esimerkin mukaisesti: Oletetaan, että neuroni S aiheuttaa syljen eritystä ja se aktivoituu neuronin R toimesta silloin, kun koira näkee ruokaa. Lisäksi neuroni K aktivoituu kellon kilinästä, jota soitetaan saman aikaisesti, kun ruokaa on tarjolla, mutta kellon soittaminen itsessään ei riitä aiheuttamaan neuronin S aktivoitumista. Kun riittävän usein toistetaan kellon soittamista ja ruoan tarjoamista samaan aikaan, eli neuronit K, S ja R aktivoituvat samaan aikaan, alkaa painokertoimet muuttua S ja K välillä ja lopulta K riittääkin yksinään aiheuttamaan S:n aktivaation. Tällaista neuronien välistä ehdollistumista kutsutaan Hebbin säännöksi. (Daniel 2013) Matemaattisesti sääntö voidaan kirjoittaa muotoon:

$$w_i(new) = w_i(old) + x_i o, \tag{5.5}$$

, jossa w on painokerroin, o on toivottu tuloste syötteille $i = 1...n$ ja x on neuronin saama syöte. Melkein kaikki oppimissäännöt perustuvat Hebbin sääntöön tai ovat sen variaatioita. (Abraham 2005)

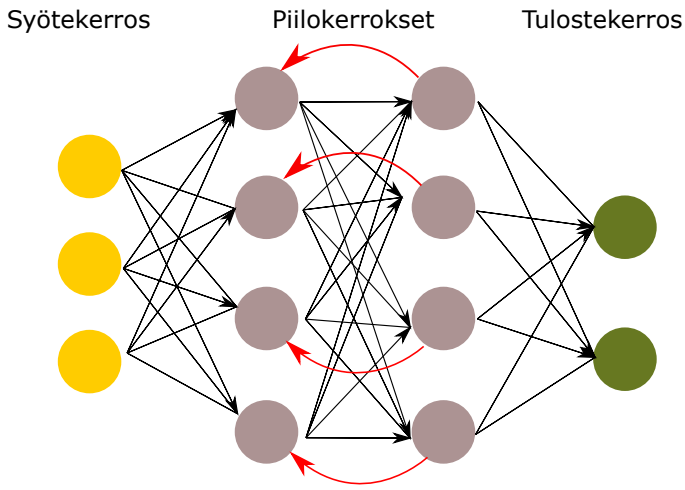
Mikäli neuroverkon neuronien aktivaatiofunktioiksi valitaan epälineaarinen funktio, kuten *sigmoidi-* tai *tanh*-funktio, voidaan neuroverkon opettamisessa käyttää laskevan gradientin menetelmää. Laskevan gradientin menetelmässä pyritään pienentämään jokin differentoituva kustannusfunktion (engl. loss function) arvo. Gradientin lasku tapahtuu siten, että etsitään kustannusfunktion derivaatta jokaisen verkon painon suhteen ja säädetään verkon painoja ne-

gatiivisen kaltevuuden suuntaan. Virhettä siis pyritään pienentämään askeleittain jokaisella kierroksella. Toistamalla tätä prosessia, päästään lopulta johonkin kustannusfunktion minimiin. (Graves 2012; Nielsen 2015)

5.2 Takaisin kytketyt neuroverkot ja LSTM

Takaisin kytketty neuroverkko eroaa kuvassa 5 esitellystä eteenpäin kytketystä neuroverkosta sillä tavalla, että se sisältää neuronien kytkentöjä myös taaksepäin edellisille ja/tai samalle kerrokselle verkossa. Tämä taaksepäin kytkentä mahdollistaa verkolle tietynlaisen muistiominaisuuden, kun aiemmat syötteet ja niiden perusteella tuotetut aktivaatiofunktioiden tulokset voidaan hyödyntää uudelleen syötteinä iteratiivisten silmukoiden avulla. (Graves 2012) Kuvassa 6 on esitelty takaisin kytketty neuroverkko, josta osa takaisinkytkennöistä on poistettu selkeyden vuoksi.

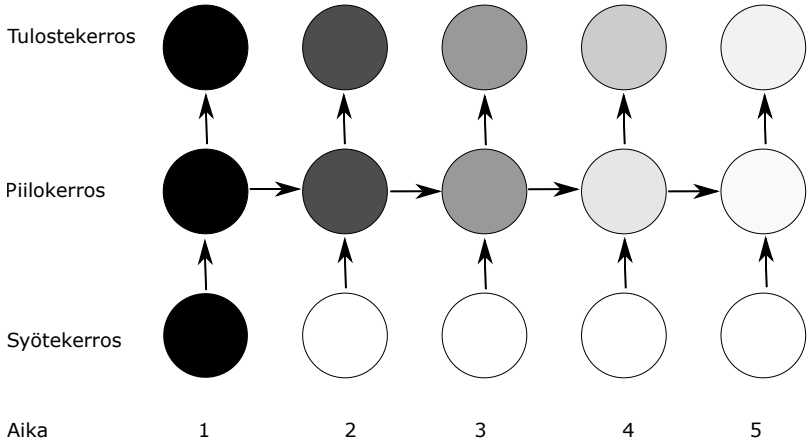
Kuvio 6. Takaisin kytketty neuroverkko



Kun perinteinen eteenpäin kytketty verkko pystyy kartoittamaan yksittäiset syötteet tulosteiksi, takaisin kytketty neuroverkko pystyy ainakin teoriassa kartoittamaan koko syötteiden historian jokaiseen tulosteeseen. Pitkien aikavälien kanssa, tämä takaisinkytkentä ei kuitenkaan toimi takaisin kytkettyjen neuroverkkojen kohdalla kovinkaan hyvin, sillä aiempien syötteiden vaikutus joko häviää tai kasvaa eksponentiaalisesti, kun syöte kulkee useita kertoja neuroverkon läpi. Tätä kyseistä ongelmaa kutsutaan häviävän gradientin ongelmaksi. (Graves 2012; Hochreiter ja Schmidhuber 1997) Kuvassa 7 näkyy häviävän gradientti on-

gelma havainnollistettuna.

Kuvio 7. Häviävän gradientin ongelma (Graves 2012)

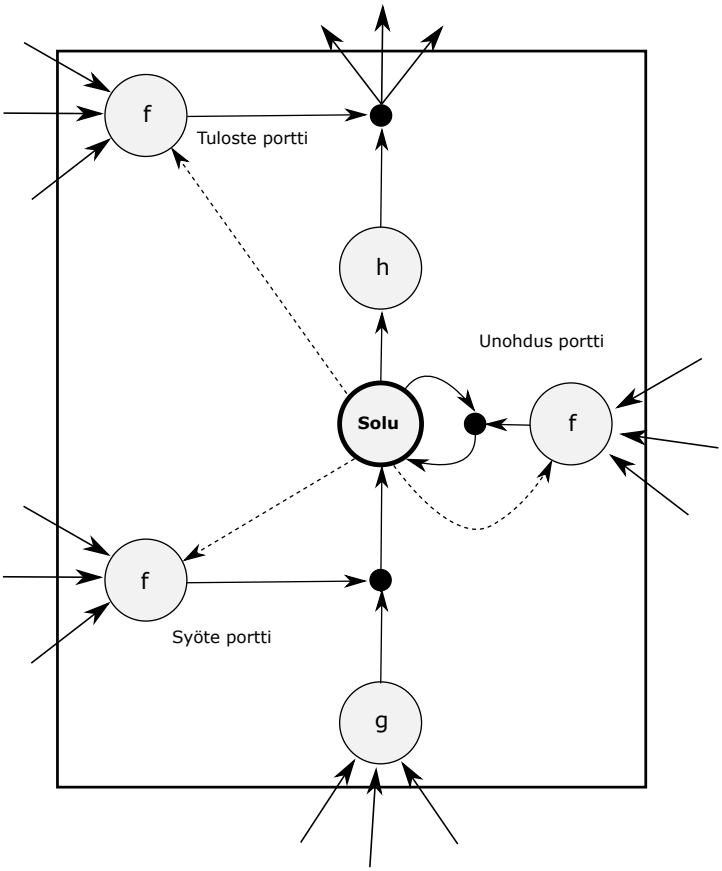


Hochreiter ja Schmidhuber (1997) kehittivät uudenlaisen neuroverkkomallin, joka ratkaisi häviävän gradientin ongelman käyttämällä hyväkseen uudenlaisia muistiyksiköitä. Tämän neuroverkon mallin nimeksi tuli LSTM eli Long short-term memory. LSTM arkkitehtuuri koostuu joukosta taaksepäin kytkettyjä aliverkkoja, joita kutsutaan muistiyksiköiksi. LSTM verkon arkkitehtuuri on muuten täysin samanlainen, kuin perinteisellä takaisin kytketyllä neuroverkolla sillä erotuksella, että piilokerroksen summayksiköt on korvattu muistiyksiköillä. (Graves 2012)

Jokainen muistiyksikkö sisältää yhden tai useamman muistisolun ja kolme kertaavaa yksikköä: syöte, tuloste ja unohdus portit, joiden avulla solut pystyvät lukemaan, kirjoittamaan ja resetoimaan itsensä. Kuvassa 8 esitetään LSTM muistiyksikön toiminta (Graves 2012)

Kuvassa 8 näkyvät portit ovat epälineaarisia summayksiköitä, jotka keräävät aktivaatioita muistiyksikön sisältä ja ulkoa. Summayksiköiden tulokset säätävät solun aktivaatiotasoa kertojayksiköiden kautta, jotka ovat kuvassa näkyvät mustat pisteet. Syöte- ja tuloste portit kertaavat muistiyksikön syötteitä ja tulosteita ja unohdusportti kertaa solun edellistä tilaa. Porttien ’f’ aktivaatiofunktioina toimii yleensä logistinen *sigmoidi*-funktio 5.3 ja tulosteiden ja syötteiden ’g’ ja ’h’ aktivaatiofunktioina toimii yleensä joko logistinen *sigmoidi*-funktio 5.3 tai *tanh*-funktio. Kuvassa näkyvät katkoviivat esittävät summayksiköiden painotettuja syötteitä. Muissa syötteissä ei joko ole painotuksia ollenkaan tai sitten painotukset ovat vakioita

Kuvio 8. LSTM muistiyksikkö, joka sisältää yhden solun. (Graves 2012)



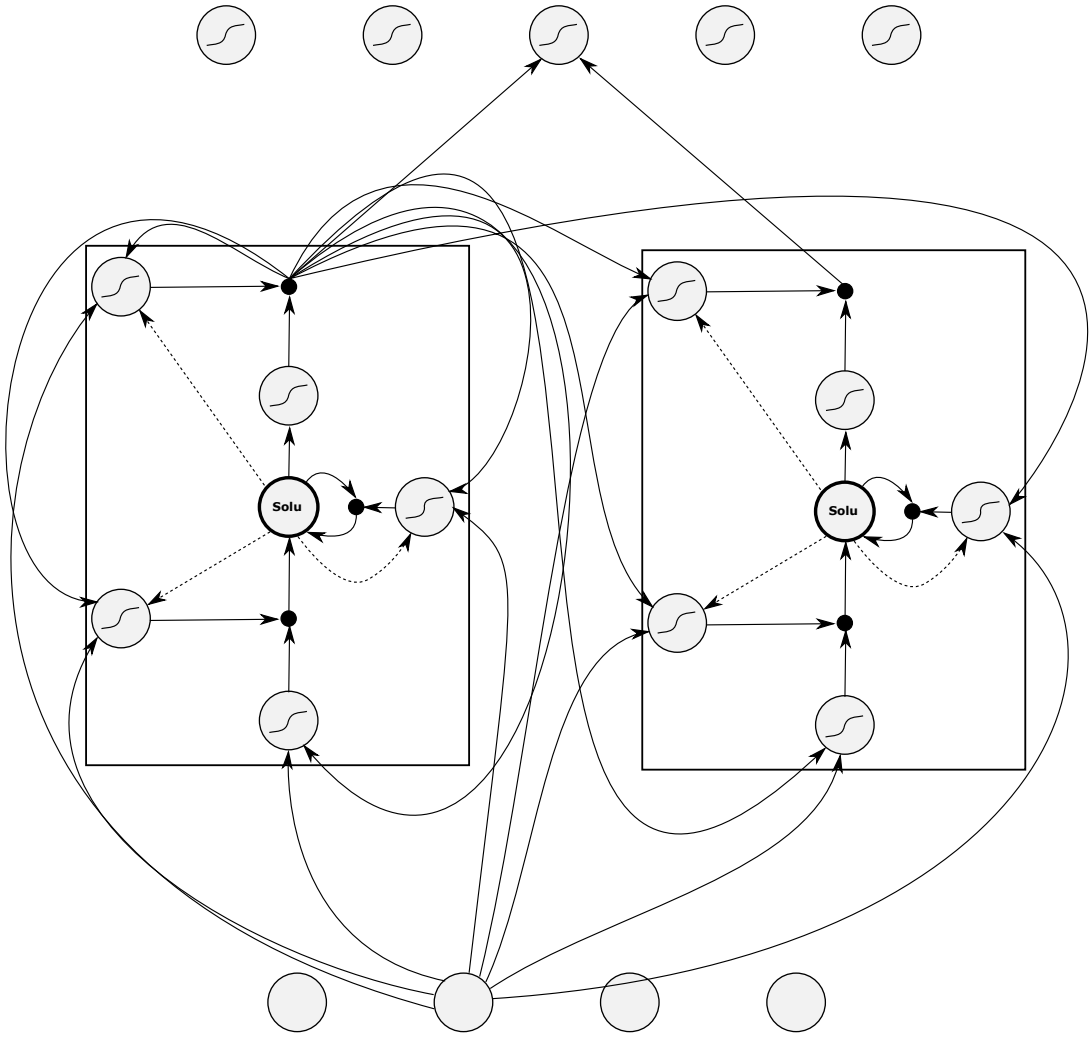
ja samat kaikissa. (Graves 2012)

LSTM:n gradientin säilyttämiskyky perustuu muistiyksiköiden kykyyn pitää syöte ja tulosteportit auki tai kiinni. Esimerkiksi, niin kauan kuin syöteportti pysyy kiinni, solun aktivaatiota ei pysty muuttamaan verkkoon tulleet uudet syötteet. Tästä syystä solun aktivaatio pystytään hyödyntämään myöhäisemmässä vaiheessa silloin, kun tulosteportti aukaistaan.

Kuviossa 9 esitetään LSTM neuroverkon malli, jolla on neljä syötettä, viisi tulostetta ja kaksi LSTM muistiyksikköä. Huomaa, että jokainen LSTM muistiyksikkö koostuu neljästä syöteportista ja vain yhdestä tulosteportista. Osa neuroverkon kytkennöistä on poistettu selkeyden vuoksi. (Graves 2012; Hochreiter ja Schmidhuber 1997)

LSTM neuroverkkoja opetetaan perinteisesti laskevan gradientin mukaisesti käyttämällä jotain optimointialgoritmia (Graves 2012). Alun perin Hochreiter ja Schmidhuber (1997) käyt-

Kuvio 9. LSTM neuroverkko, joka koostuu kahdesta LSTM muistiyksiköstä. (Graves 2012)



tivät gradientin laskemiseen RTRL:n (Real Time Recurrent Learning), sekä BPTT:n (Back-propagation Through Time) yhdistelmää, jossa jokaisen askeleen jälkeen BPTT:n vaikutus katkaistaan, jotta pitkän ajan riippuvuuksista vastaisi muistiyksiköt. Tässä tutkimuksessa gradientin laskennassa käytettiin Kingma ja Ba (2014) esittelemää Adam-optimointi algoritmia, jonka on todettu olevan laskennaltaan hyvin tehokas ja se soveltuu hyvin suurille datajaksoille. Lisäksi Adam -algoritmi löytyi valmiina toteutuksena Tensorflow- kirjastosta, joten sitä ei tarvinnut itse toteuttaa.

5.3 LSTM:n käyttö aiemmissa tutkimuksissa osakekurssien ennustamiseen

LSTM on noussut sen keksimisen jälkeen erittäin suosituksi menetelmäksi aikasarjojen ennustamiseen, luonnollisen kielen tunnistamiseen ja sitä pidetään johtavana ratkaisuna käsin kirjoitetun tekstin tunnistamisessa. (Nelson, Pereira ja Oliveira 2017) Se on myös useiden tutkimusten perustella havaittu olevan toimiva keino osakekurssien ennustamiseen.

Nelson, Pereira ja Oliveira (2017) tutkivat osakekurssien kehityksen suunnan ennustamista LSTM neuroverkoilla onnistuneesti saavuttaen 55.9% onnistumisprosentin osakkeen lähitulevaisuuden suunnasta. He hyödynsivät työssään kurssidataa, sekä teknisiä indikaattoreita ennustaessaan tulevan 15min suuntaa kurssille. Tutkimuksessa LSTM:n havaittiin olevan muita tutkimuksessa vertailuna käytettyjä ennustusmenetelmiä tehokkaampi ennustamaan kurssin suuntaa kyseisellä ajanjaksolla.

Roondiwala, Patel ja Varma (2017) pyrkivät ennustamaan LSTM:n avulla osakekurssien kehitystä. Tutkimuksessa käytettiin datana kuuden vuoden osakekurssi dataa, josta 5-10% otettiin testausdataksi ja loput toimivat harjoitusdatana. Tutkimuksessa ei kerrottu, kuinka pitkän ajan päähän pyrittiin ennustamaan, mutta tulosten perusteella ennusteita pidettiin hyvinä ja riittävän tarkkoina.

Mehtab, Sen ja Dutta (2020) vertasivat tutkimuksessaan useita eri koneoppimisalgoritmeja osakekurssien avaushintojen ennustamiseen viikkotasolla. Tutkimuksessa luotiin useita eri LSTM malleja, jotka saivat harjoitusdatana syötteenä neljän vuoden edeltävän päiväkohtaisen datan. Luodulla mallilla pyrittiin ennustamaan seuraavan viikon osakekurssien avaushinnat. Tulosten perusteella LSTM pystyi ennustamaan avaushintoja merkittävästi paremmin, kuin muut tutkimuksessa käytetyt koneoppimisalgoritmit.

6 Tutkimuksen kuvaus ja käytettävä data

Tässä osiossa käydään läpi tutkielmaa varten toteutettu empiirinen vertaileva tutkimus, jossa verrattiin ARIMA:n ja LSTM:n avulla tehtyjen ennusteiden tarkkuuksia, sekä pohdittiin osakkeiden ennustettavuutta tulosten perusteella. Osiossa 6.1 käydään läpi tutkimuksen toteutuksen kuvaus, osiossa 6.2 käydään läpi aineistoon liittyvät asiat ja lopuksi osiossa 6.5 tarkastellaan ennusteiden tarkkuuden mittauksessa käytettyjä mittareita.

6.1 Tutkimuksen kuvaus

Tutkimuksessa tavoitteena oli verrata kahden yleisesti käytetyn koneoppimispohjaisten menetelmien avulla tehtyjen ennusteiden tarkkuuksia, sekä pohtia tulosten pohjalta osakekurssien ennustettavuutta. Ennusteet luotiin jokaisessa ennustusmenetelmässä yksi askel kerrallaan 20 pörssipäivän eli noin kuukauden päähän ennustushetkestä.

Tutkimukseen valittiin algoritmit aiempien aiheeseen liittyvien tutkimusten perusteella. Valitut algoritmit olivat ARIMA ja LSTM, joilla on saatu aiemman kirjallisuuden perusteella hyviä tuloksia osakekurssien ennustamisessa. Tämän lisäksi ennusteiden graafeja tarkastelemalla päätettiin lisätä vielä kolmas ennuste, jossa ennusteen arvoksi laskettiin LSTM:n ja ARIMA:n ennusteiden keskiarvot. Saatuja ennusteiden arvoja verrattiin naiiviin ennusteeseen, jossa oletettiin, että kurssi tulee pysymään samana, kuin se on ennustettavana päivänä.

Saadut tulokset koottiin yhteen csv-tiedostoon, joka sisälsi ennusteiden tulokset virhemetriikoiden muodossa omina riveinään. Tämän lisäksi jokaisesta ennusteesta tallennettiin pdf-muodossa graafi, joka sisälsi ennusteet ja todelliset arvot ennustettavalta ajanjaksolta.

Huomion arvoista tehdyssä ohjelmassa, jolla ennusteet toteutettiin, on se, että ennustusprosessista tehtiin täysin automaattinen, eikä sisällä manuaalista prosessointia käyttäjältä. Kaikki ennusteet luotiin siten samoilla periaatteilla ja valinnoilla kaikille osakkeille.

6.2 Aineisto ja sen hankkiminen

Tutkielmassa käytettäväksi aineistoksi valitiin OMX Helsinki 25 indeksissä olevat yhtiöt, joilla on osakedataa aikavälillä 1.1.2013 - 31.12.2017. Valittuja yhtiöitä oli yhteensä 23, kun Neles ja Kojamo jätettiin aineiston ulkopuolelle puuttuvan osakedatan vuoksi.

Data haettiin Yahoo Financen tietokannasta yfinance python kirjaston avulla. Yfinance kirjasto sisälsi tämän tutkielman kannalta riittävän laajat ja helppo käyttöiset rajapinnat osakedatan hakemiseen. Muitakin vastaavia yfinancen kaltaisia kirjastoja osakedatan hakemiseen on olemassa, mutta yfinance valittiin tähän tutkimukseen, sillä se oli ilmainen ja sen API:n kautta saatava data pystyttiin hyödyntämään pienellä esiprosessoinnilla sellaisenaan, ilman konversioita tai datan tallentamista tiedostoon välissä. Toinen tärkeä syy Yahoo Financen tietokannan käyttöön oli suomalaisen osakedatan saatavuus, sillä monet muut palvelut tarjoavat ilmaiseksi vain Yhdysvaltojen pörssien dataa.

Haetusta datasta käytettiin ennustamiseen kuuden sarakkeen tietoa, jotka olivat: Päivämäärä ("Date"), päivän avaushinta ("Open"), päivän korkein hinta ("High"), päivän alin hinta ("Low"), päivän sulkuhinta("Close") ja päivän aikana vaihdettujen osakkeiden määrä ("Volume"). Näistä kuudesta sarakkeesta käytetään jatkossa nimeä "LSTM:n lähtödata". Jokaista osaketta kohti haettiin dataa yhteensä 1255 pörssipäivän verran. Data jaettiin harjoitus ja testaus osiin 80/20 jaolla, eli ensimmäiset 80% datasta olivat harjoitusdataa ja loput 20% olivat testausdataa.

Tutkimuksessa käytettävät algoritmit ARIMA ja LSTM vaativat datan olevan hieman eri muodoissa, joten data kopioitiin kahteen samanlaiseen osaan ja näille osille tehtiin esiprosessointi erikseen. ARIMA pystyy käsittelemään pelkästään dataa, joka sisältää yhden muuttujan. Näin ollen ARIMA:n kohdalla datan esiprosessointi ei vaatinut muuta, kuin että alkuperäisestä aineistosta valittiin sulkuhinta ("Close") ja sen arvot jaettiin harjoitus ja testaus osiin.

LSTM:n kohdalla datan esiprosessointi oli monimutkaisempi prosessi. Ensin jokaisen sarakkeen data skaalattiin 0 ja 1 välille, sillä LSTM toimii tehokkaammin ja tarkemmin, kun data on skaalattu tälle välille. Skaalaukseen käytettiin sklearn -kirjaston MinMaxScaler-funtiota. Tämän jälkeen data jaettiin kaksiosaisiin sekvensseihin, joista ensimmäinen sisälsi 80 data-

pistettä LSTM:n lähtödataa, jonka perusteella neuroverkko pyrkii luomaan yhden päivän ennusteen 20 pörssipäivän päähän tulevaisuuteen ja toinen osa sisälsi osakkeen todellisen arvon 20 päivän kuluttua. Tämä sekvenssoitu data toimii syötteenä LSTM neuroverkolle. Tämän jälkeen LSTM:n data jaettiin ARIMA:n tavoin harjoitus- ja testidataan samalla jaolla.

6.3 ARIMA:n käyttö tässä tutkielmassa

Tässä tutkielmassa käytettiin Smith ym. (2017–) luomaa 'pmdarima'- Python kirjastoa, joka perustuu R-kielellä toteutettuun auto.arima() funktioon. Pmdariman auto_arima-funktiota käytetään löytämään paras mahdollinen ARIMA(p,d,q) malli automatisoidun menetelmän avulla, joka jäljittelee Box-Jenkins menetelmää. Auto_ariman käyttö mahdollisti ennusteiden automatisoinnin ARIMA:n osuudelta.

Auto_arima etsii parhaimmat mahdolliset p, ja q parametrit ARIMA-mallille, sekä niitä vastaavat P ja Q parametrit SARIMA-mallille käyttämällä informaatio kriteerejä (AIC, AICc, BIC tai HQIC). Funktio etsii parhaat mallin parametrit kokeilemalla eri malleja annettujen funktion parametrien rajoituksien mukaisesti ja valitsemalla sen mallin, joka saa pienimmän mahdollisen luvun valitusta informaatio kriteeristä. (Smith ym. 2017–)

Aikasarjan stationaarisuuden auto_arima tarkastelee itse useiden mahdollisten testien avulla, kuten laajennetun Dickey-Fuller (ADF) - testin tai KPSS testin avulla. Aikasarjan stationaarisuuden tarkastelusta ja mahdollisesti tarvittavasta differentioinnista huolehtii auto_arima itsenäisesti, eli käyttäjän ei tarvitse itse differentioida aikasarjaa ennen auto_ariman käyttöä.

Auto_arimalle annetaan parametreina aikasarja, jonka perusteella halutaan luoda ennuste, sekä muita mahdollisia tarkentavia parametreja, kuten rajaukset testattavista p, d ja q arvoista, mikäli ne ovat tiedossa. Auto_arima tekee ensin stationaarisuuden testit, jonka jälkeen se tutkii informaatiokriteerien avulla, parhaimman mahdollisimman mallin kuvaamaan aikasarjaa. Kun kyseinen ARIMA-malli on löytynyt, sitä voidaan käyttää ennusteen luomiseen halutun määrän päiviä eteenpäin.

Tässä tutkielmassa käytettiin auto_arimaa ennustamaan liukuvan ikkunan menetelmän tavoin 80 datapisteen perusteella seuraavat 20 datapistettä. Ennustaminen tehtiin kahden pe-

räkkäisen silmukan sisällä, jossa ensimmäisessä luotiin 20 päivän ennuste ja tämän jälkeen nämä 20 päivää lisättiin yksi kerrallaan ennusteiden joukkoon. Auto_ariman parametreiksi ei annettu ikkunassa olevan aikasarjan lisäksi muuta, kuin d=1 , sillä jokainen testattu aikasarja tutkittiin etukäteen ja huomattiin, että yksi differentiointi kerta oli riittävä aikasarjan stationaarisuuden saavuttamiseksi. Liitteestä A näkee lähdekoodin, kuinka auto_arima ennustetta on käytetty.

6.4 LSTM:n käyttö tässä tutkielmassa

Tässä tutkielmassa käytettiin Googlen toteuttamaa avoimen lähdekoodin TensorFlow-kirjastoa, joka on kehitetty erityisesti koneoppimista ja tekoälyä varten. Se sisältää monipuoliset työkalut koneoppimismenetelmien käyttöön Python ohjelmointikielellä.

Tensorflow-kirjaston avulla luotiin LSTM-neuroverkko, joka koostui kahdesta LSTM- kerroksesta, joissa molemmissa oli 256-LSTM-yksikköä. LSTM-yksiköiden aktivointifunktiona toimi *tanh*-funktio. Jokaisen LSTM -kerroksen jälkeen seurasi pudotuskerros, jonka tarkoituksena on vähentää ylisovitusta ja parantaa virheiden yleistettävyyttä. Pudotuskerroksen pudotusprosentti asetettiin 20%:iin. Viimeiseksi kerrokseksi asetettiin koontikerros, joka kokoaa kaikki edeltävän kerroksen tulosteet. Liitteessä B näkyy lähdekoodi, jolla LSTM neuroverkko luotiin.

Syötteenä neuroverkolle toimi kohdassa 6.2 kuvattu sekvenssoitu harjoitusdata. Syötteen perusteella neuroverkon opettaminen toteutettiin laskevan gradientin mukaisesti käyttämällä Adam-optimointitalgoritmia ja ajamalla 170 gradientin päivitys kierrosta (engl. epochs), joiden aikana neuroverkolle syötettiin 64 näytettä ennen jokaista gradientin päivitystä. Opetuksen jälkeen saadulla mallilla pyrittiin ennustamaan testausdatan aikasarja. Lopuksi vielä peruttiin datan valmistelu vaiheessa toteutettu skaalaus, jotta saatiin tulokset oikealle suuruusluokalle.

6.5 Ennusteen tarkkuuden mittaaminen

Ennusteiden tarkkuutta mitataan ennustetun arvon ja todellisen arvon välisen virheen avulla. Perinteisesti ennustamisessa käytettyjä virheiden mittareita ovat MAE (mean absolute error) ja RMSE (root mean squared error). (Hyndman ja Athanasopoulos 2018)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|todellinen_i - ennuste_i|$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(todellinen_i - ennuste_i)^2}$$

MAE kertoo keskimääräisen ennusteiden virheen ottamatta kantaa virheen suuntaan ja kaikilla ennusteilla on sama painoarvo keskimääräistä virhettä laskiessa. Minimoimalla MAE, saadaan ennusteet lähelle aineiston mediaania. RMSE puolestaan antaa suurille virheille suuremman painoarvon, sillä yksittäiset virheet neliöidään ja neliöjuuri lasketaan vasta summattujen yksittäisten arvojen keskiarvosta. Tämä tarkoittaa sitä, että RMSE rankaisee voimakkaammin suurista virheistä, kuin MAE. RMSE:n minimoinnilla saadaan ennusteet lähelle aineiston keskiarvoa. (Hyndman ja Athanasopoulos 2018)

Nämä mittarit ovat kuitenkin skaalasta riippuvaisia eli MAE ja RMSE eivät itsessään kerro ennusteen tarkkuudesta mitään, ennen kuin on tiedossa millä suuruusluokalla ennusteiden oikeat arvot liikkuvat. Tämä johtaa siihen, että eri osakkeiden MAE ja RMSE arvot eivät ole keskenään vertailukelpoisia. Tästä syystä näiden perinteisempien mittarien sijaan käytettiin näistä kahdesta mittarista johdettuja skaalasta riippumattomia mittareita MAPE (mean absolute percentage error) ja NRMSE (normalized root mean squared error)

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{todellinen_i - ennuste_i}{todellinen_i}\right| * 100$$

$$NRMSE = \frac{RMSE}{\bar{y}} \text{ , jossa}$$

$\bar{y} =$ todellisten arvojen keskiarvo

MAPE kertoo keskimääräisen prosentuaalisen poikkeaman ennusteen ja todellisen arvon välillä. Prosentit lasketaan jokaisen ennusteen kohdalla, joten näin ollen saadaan säilytettyä täsmälleen sama informaatio, minkä MAE antaa virheestä, mutta se on muutettu skaalasta riippumattomaan muotoon. NRMSE:n arvon laskennassa ei tehdä laskentaa jokaisen ennustetun arvon kohdalla erikseen, vaan siinä RMSE:n normalisointi suoritetaan jakamalla RMSE todellisten arvojen joukon keskiarvolla. Tästä johtuen NRMSE ei sisällä täsmälleen samaa informaatiota, kuin RMSE, mutta sen avulla saadaan muutettua RMSE:n tulokset skaalariippumattomaan muotoon.

Edellä mainittujen ennusteiden tarkkuutta mittaavien mittareiden lisäksi ennusteiden hyvyyttä verrataan naiiviin ennustusmenetelmään. Tässä tutkielmassa naiivilla ennusteella tarkoitetaan Hyndman ja Athanasopoulos (2018) määritelmän mukaista naiivia ennustetta, jossa jokainen ennustettu arvo vastaa viimeisimmän todellisen havainnon arvoa. Matemaattisesti naiivi ennuste voidaan kirjoittaa muotoon:

$$\hat{y}_{T+h|T} = y_T \tag{6.1}$$

Koska naiivi ennuste on optimaalinen ennuste silloin, kun data seuraa satunnaiskävelyä, kutsutaan naiiveja ennusteita myös satunnaiskulun ennusteiksi (Hyndman ja Athanasopoulos 2018). Tämä ominaisuus onkin tämän tutkimuksen kannalta hyvin mielenkiintoinen, sillä tehokkaan markkinan teorian mukaisesti osakekurssit seuraavat satunnaiskävelyä. Tästä voidaan vetää johtopäätös, että naiivin ennusteen tulisi menestyä ennusteita vertailtaessa hyvin muihin ennustusmenetelmiin verrattuna, jotta tehokkaiden markkinoiden teoria saisi tukea.

7 Tulokset

Tässä luvussa tarkastellaan tutkielman empiirisessä osuudessa saatuja tuloksia. Tulokset saatiin luomalla ennusteet LSTM:n, ARIMA:n, sekä hybridin avulla, hyödyntämällä tutkimuksessa luotua python ohjelmaa. Osakekurssien ennusteet luotiin luvussa 6.2 esitellyllä tavalla OMX Helsinki 25 indeksissä oleville yhtiöille, joilta löytyi osakekurssidataa aikavälillä 1.1.2013 - 31.12.2017. Ennusteiden mittarit MAPE ja NRMSE kertovat molemmat virheen suuruutta, eli pienempi luku tarkoittaa parempaa ennustetta. Saadut tulokset näkyvät NRMSE:n osalta taulukossa 10, sekä MAPE:n osalta taulukossa 11.

7.1 Ennusteiden tarkkuus

Testidata koostui 23 yrityksen osakekurssidatasta, joka sisälsi ARIMA:n tapauksessa sulkuhinnan ja LSTM:n tapauksessa sulkuhinnan lisäksi avaushinnan, korkeimman hinnan, alimman hinnan, sekä tiedot vaihdon määrästä. Tuloksiin laskettiin ARIMA:n ja LSTM:n lisäksi mukaan hybridi ennuste, jossa jokaiselle LSTM:n ja ARIMA:n ennusteelle laskettiin ennusteen keskiarvo. Lisäksi mukaan otettiin naiivi ennuste, jossa ennustettiin hinnan pysyvän samana, kuin se on ennustuspäivänä. Saadut tulokset sisälsivät virhemetriikat MAPE:n ja NRMSE:n muodossa kaikille ennustusmenetelmille.

Tuloksista selviää, että keskiarvojen perusteella kaikki ennustusmenetelmät pääsivät lähestulkoon samoihin ennustustarkkuuksiin. LSTM pystyi ennustamaan osakekursseja hieman paremmin, kuin ARIMA, hybridi tai naiivi ennuste. LSTM päihitti NRMSE:n osalta ARIMA:n kaikissa osakkeissa, paitsi yhdessä, sekä hybridin kaikissa, paitsi kolmessa osakkeessa. Huomion arvoista on kuitenkin se, että hybridin NRMSE:n keskiarvo oli alempi, kuin LSTM:llä. Naiiviin ennusteeseen verrattaessa LSTM oli parempi kaikissa paitsi yhdessä ennusteessa. ARIMA oli ennusteissa NRMSE:n osalta huonoin ja se jäi ennustustarkkuudessa viimeiseksi kaikissa ennusteissa paitsi viidessä, joissa neljässä huonoin ennuste oli naiivilla ja yksi LSTM:llä. Ennusteiden tarkkuuksien väliset erot olivat kuitenkin hyvin pieniä.

Kuvio 10. Ennusteiden tulokset NRMSE

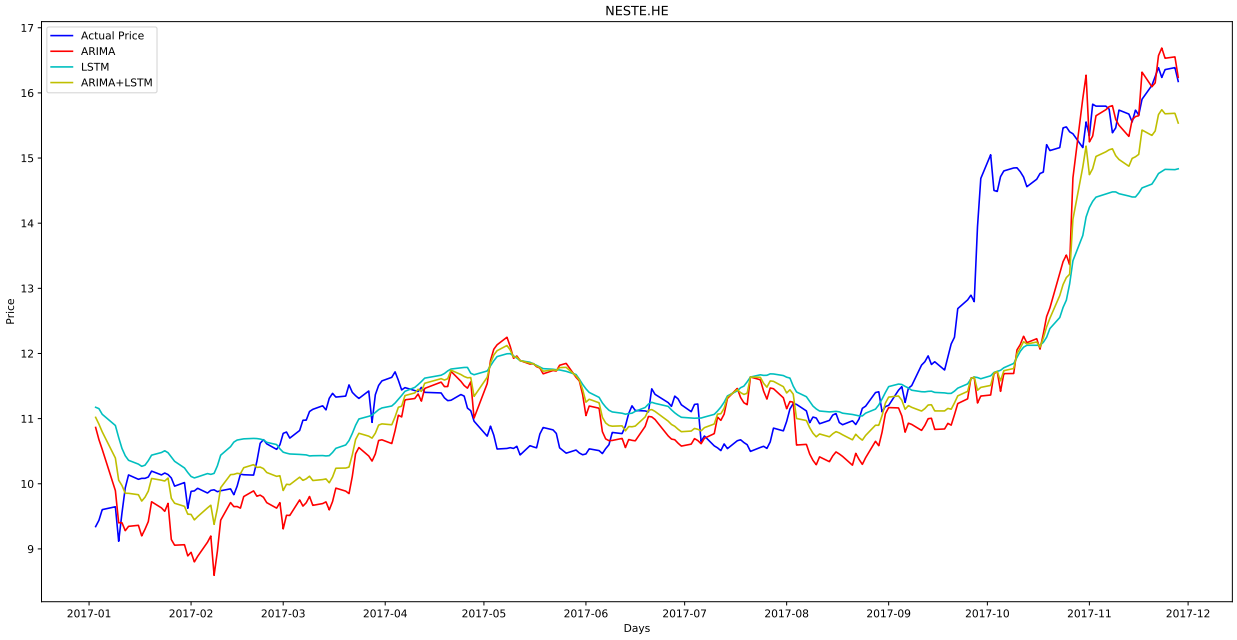
ticker	arima_rmse	lstm_rmse	hybrid_rmse	naive_rmse
NOKIA.HE	8.64	7.83	7.84	8.13
KNEBV.HE	4.43	3.54	3.81	4.15
NESTE.HE	10.12	9.90	9.36	10.38
SAMPO.HE	3.28	2.49	2.74	3.15
STERV.HE	6.34	5.09	5.31	5.47
FORTUM.HE	6.79	5.67	5.83	5.94
UPM.HE	7.31	6.23	6.69	6.43
KESKOB.HE	5.71	4.71	5.15	4.84
NDA-FI.HE	5.86	5.44	5.46	5.52
ELISA.HE	4.12	3.35	3.55	3.88
TYRES.HE	3.93	3.39	3.52	3.95
WRT1V.HE	6.25	7.94	6.12	5.92
MOCORP.HE	8.49	6.80	7.26	8.01
VALMT.HE	7.81	5.93	6.68	6.61
OUT1V.HE	11.71	9.88	10.35	11.06
ORNBV.HE	11.46	11.38	10.63	11.50
TELIA1.HE	11.67	11.07	11.28	11.43
TIETO.HE	5.16	4.58	4.74	5.09
HUH1V.HE	4.84	4.22	4.41	4.51
KCR.HE	5.90	5.57	5.59	5.90
METSB.HE	8.26	5.68	6.55	7.00
CGCBV.HE	7.54	6.25	6.37	7.18
KEMIRA.HE	4.90	4.30	4.17	4.68
Keskiarvot	6.77	6.07	6.00	6.40

MAPE:n osalta tulokset olivat vielä tasaisemmat, kuin NRMSE:n kohdalla. LSTM oli tämänkin mittarin mukaan paras ennustamaan osakekursseja ja se olikin paras kaikissa paitsi kuudessa ennusteessa. Hybridi ja naiivi olivat molemmat parhaita kolmessa ennusteessa ja ARIMA oli tässäkin osuudessa heikoin olemalla huonoin kaikissa paitsi viidessä ennusteessa. Kuviossa 12 näkyy esimerkki ennusteiden käyristä.

Kuvio 11. Ennusteiden tulokset MAPE

ticker	arima_mape	lstm_mape	hybrid_mape	naive_mape
NOKIA.HE	7.13	6.12	6.40	6.55
KNEBV.HE	3.54	2.75	2.95	3.33
NESTE.HE	6.62	6.50	6.19	6.87
SAMPO.HE	2.65	1.99	2.16	2.51
STERV.HE	5.42	4.46	4.71	4.32
FORTUM.HE	5.09	4.38	4.60	4.58
UPM.HE	6.01	4.90	5.40	5.13
KESKOB.HE	4.69	3.83	4.23	4.17
NDA-FI.HE	4.86	4.14	4.40	4.25
ELISA.HE	3.35	2.88	2.93	3.06
TYRES.HE	3.30	2.79	2.94	3.38
WRT1V.HE	5.23	6.64	5.15	4.90
MOCORP.HE	6.63	5.09	5.52	6.43
VALMT.HE	5.87	4.54	4.89	5.08
OUT1V.HE	9.44	7.98	8.46	8.69
ORNBV.HE	7.74	9.01	7.38	8.85
TELIA1.HE	6.81	6.81	6.60	6.44
TIETO.HE	3.39	2.59	2.86	3.34
HUH1V.HE	4.16	3.50	3.76	3.80
KCR.HE	4.60	4.45	4.39	4.59
METSB.HE	7.10	4.51	5.62	6.17
CGCBV.HE	6.05	4.73	5.04	5.72
KEMIRA.HE	3.81	3.27	3.33	3.70
Keskiarvot	5.47	4.69	4.87	5.12

Kuvio 12. Esimerkki graafi ennusteesta



Tulokset olivat samassa linjassa aiempien tutkimusten kanssa, joissa (Siami-Namini, Tavakoli ja Namin 2018; Siami-Namini ja Namin 2018) pystyivät saavuttamaan LSTM:n avulla selkeästi parempia tuloksia ennusteissa, kuin ARIMA:lla. Tässä tutkimuksessa ennusteiden väliset erot olivat kuitenkin huomattavasti pienemmät. Osasyynä tähän on varmasti erilaiset tavat ja parametrit, joilla LSTM ja ARIMA mallit luotiin. Siami-Namini, Tavakoli ja Namin (2018) tutkimuksessa ARIMA:n heikon menestyksen osasyynä uskon olevan etukäteen päätetty ARIMA(5,1,0) malli, jota ei arvioitu uudelleen ennusteita tehdessä.

Tämän tutkimuksen perusteella ARIMA:n ja LSTM:n ennusteiden yhdistämisellä ei saatu hybridin muodossa parannettua osakkeiden ennustettavuutta pelkkään LSTM:ään nähden. Luonnollisestikaan hybridi ratkaisulla ei oltu kummallakaan virheiden mittarilla tarkasteltuna huonoimpia minkään osakkeen kohdalla, mutta ARIMA:n heikkouden vuoksi hybridi oli säännöllisesti huonompi, kuin LSTM.

7.2 Tulokset osakkeiden ennustevuuden näkökulmasta

Osakkeiden ennustettavuuden puolesta tutkimuksen tulokset eivät olleet kovin mairittelevat. Vaikka saatujen ennusteiden tarkkuuksien perusteella voitaisiin olla tyytyväisiä ARIMA:n ja LSTM:n ennusteiden tarkkuuksiin, niin naiivin ennusteen saavuttama vastaava tarkkuus vie pohjan tuolta tyytyväisyydeltä. Kaikki tutkimuksessa luodut ennustajat LSTM, ARIMA, hybridi ja naiivi pystyivät ennustamaan kuukauden päähän osakkeita keskimäärin noin 95% tarkkuudella. Naiivin ennusteen hyvä menestymisen perusteella voidaan tehdä johtopäätös, että osakkeet ainakin jollain tasolla seuraavat satunnaiskulkua ja tehokkaiden markkinoiden hypoteesi saa tukea.

Graafeja tutkiessa on selkeästi havaittavissa, että millään käytetyistä ennustusmenetelmistä ei ole kykyä tehdä ennusteita proaktiivisesti. Kaikkien ennusteiden käyrät selkeästi vain jäljittelevät noin 20 askelta perässä todellisen kurssin käyrää. Tästä voidaan päätellä, että tehdyt ennustajat vain reagoivat kurssin muutoksiin, eivätkä pysty sitä aidosti ennustamaan.

Tämän tutkimuksen tulosten perustella koneoppimisen menetelmillä, jotka käyttävät opetusdatana pelkästään kurssien historiallisia hinta ja volyymitietoja ei pystytty luomaan ennusteita, jotka olisivat olleet merkittävästi parempia kuin naiivi ennuste. LSTM oli käytetyistä ennustusmenetelmistä paras, mutta sekään ei pystynyt merkittävästi parempaan ennustuskykyyn, kuin naiivi ennuste.

8 Yhteenveto ja pohdinta

Tässä työssä tutustuttiin osakemarkkinoihin, osakekursseihin ja niiden aikasarjojen ennustamiseen koneoppimisen menetelmillä. Tutkielman teoreettinen osa koostui katsauksista osakemarkkinoiden ja koneoppimisen teoreettiseen taustaan ja vallitseviin teorioihin, jonka jälkeen perehdyttiin tarkemmin ARIMA:an ja LSTM:ään. Tutkielman empiirisessä osassa luotiin python ohjelmointikielellä ohjelma, joka pystyi luomaan automaattisesti ennusteet halutulle osakkeelle LSTM:n, ARIMA:n, niiden hybridin ja naiivin menetelmän avulla. Tutkimuksen empiirisen osuuden tarkoituksena oli saada vastaus esitettyihin tutkimuskysymyksiin, jotka olivat seuraavat:

1. Voidaanko osakekursseja ennustaa koneoppimisen menetelmillä?
2. Millä algoritmilla päästään parhaaseen ennustustarkkuuteen?

Saatujen tulosten perusteella vastaukseksi toiseen tutkimuskysymykseen saatiin, että LSTM oli ennustusmenetelmistä tarkin, hybridi oli toinen, naiivi menetelmä oli kolmas ja ARIMA oli epätarkin. Ennusteiden erot olivat kuitenkin hyvin pieniä ja kaikki menetelmät saavuttivat lähestulkoon saman ennustustarkkuuden. Etenkin naiivin menetelmän saavuttaman tarkkuuden perusteella voidaan vastata ensimmäiseen tutkimuskysymykseen, että tässä tutkimuksessa käytetyillä koneoppimisen menetelmillä, joiden syötteinä toimi pelkästään kurssien historiallinen hintadata, ei pystytä ennustamaan osakekurssien kehitystä. Näin ollen satunnaiskulun hypoteesi sekä tehokkaiden markkinoiden hypoteesi saavat tukea tämän tutkimuksen tuloksista.

Tulokset olivat odotettuja, sillä mikäli osakekursseja todella pystyttäisiin ennustamaan pelkän aiemman hintatiedon perusteella, johtaisi se tilanteeseen, jossa sijoittajat pystyisivät saavuttamaan riskitöntä tuottoa yksinkertaisten ja kohtalaisen helposti toteutettavissa olevien menetelmien avulla. Oletin kuitenkin ennen tutkimuksen tekoa, että LSTM ja ARIMA olisivat molemmat olleet hieman naiivia ennustetta parempia.

Graafeja tarkastellessa huomasin, että ARIMA noudattaa naiivia ennustetta silloin, kun se ei löydä opetusdatan perusteella sopivia p ja q arvoja. Koska naiivi ennuste oli systemaattisesti ARIMAA parempi, voidaan tehdä oletus siitä, että ARIMA löysi virheellisiä korrelaatioita

arvojen välillä annetuilla opetusdata jaksoilla. ARIMA olisi siten voinut toimia paremmin eri mittaisella opetusdatan ikkunan koolla. Halusin kuitenkin tämän tutkielman näkökulmasta, että sekä LSTM, että ARIMA koulutetaan vertailtavuuden vuoksi saman mittaisella datalla.

Uskon myös, että kattavammalla LSTM:n parametrien optimoimisella ja muiden syötteiden, kuin pelkkien osakekurssin hintatietojen lisäyksellä LSTM:n syötteeseen, voitaisiin päästä tilanteeseen, jossa LSTM:n ennuste toimisi merkittävästi paremmin, kuin naiivi ennuste. Esimerkkejä potentiaalisista syötteistä voisivat olla makrotalouden uutiset, osakkeen fundamenttien aikasarjat tai tekniset indikaattorit.

Koska tämän työn puitteissa ei ehditty lisätä muita LSTM:n syötteiden muuttujia, kuin historialliset hintatiedot, olisi jatkotutkimuksena erityisen mielenkiintoista yhdistää osakkeiden historiallisia fundamenttitietoja, kuten taseen tai p/e luvun tietoja, sekä makrotalouden uutisten tietoa lisäsyötteinä LSTM-neuroverkolle hintatietojen lisäksi. Näin voitaisiin ennusteissa ottaa huomioon sijoittajien yleisimmin käytettyjä osakkeen arvostus menetelmiä sekä päivittäistä uutistietoa, jonka katsotaan olevan satunnaiskulun teorian mukaan ainut syy osakkeiden päivittäiselle vaihtelulle.

Lähteet

Abraham, Ajith. 2005. “Artificial neural networks”. *Handbook of measuring system design.*

Ariyo, Adebiyi A, Adewumi O Adewumi ja Charles K Ayo. 2014. “Stock price prediction using the ARIMA model”. Teoksessa *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation,* 106–112. IEEE.

Balvers, Ronald J, Thomas F Cosimano ja Bill McDonald. 1990. “Predicting stock returns in an efficient market”. *The Journal of Finance* 45 (4): 1109–1128.

Box, George EP, ja Gwilym M Jenkins. 1970. *Time Series Analysis Forecasting and Control.* Tekninen raportti. WISCONSIN UNIV MADISON DEPT OF STATISTICS.

Campbell, Murray, A Joseph Hoane Jr ja Feng-hsiung Hsu. 2002. “Deep blue”. *Artificial intelligence* 134 (1-2): 57–83.

Cheung, Yin-Wong, ja Kon S Lai. 1995. “Lag order and critical values of the augmented Dickey–Fuller test”. *Journal of Business & Economic Statistics* 13 (3): 277–280.

Daniel, Graupe. 2013. *Principles Of Artificial Neural Networks (3rd Edition).* Nide 3rd edition. Advanced Series in Circuits and Systems, vol. 7. World Scientific. ISBN: 9789814522731. `https://search-ebscohost-com.ezproxy.jyu.fi/login.aspx?direct=true&db=nlebk&AN=622050&site=ehost-live`.

Devi, B Uma, D Sundar ja P Alli. 2013. “An effective time series analysis for stock trend prediction using ARIMA model for nifty midcap-50”. *International Journal of Data Mining & Knowledge Management Process* 3 (1): 65.

Eğrioğlu, Erol, toimittanut. 2012. *Advances in time series forecasting.* 135. Oak Park, Ill.: Bentham eBooks. `http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&AN=500676`.

Enke, David, ja Suraphan Thawornwong. 2005. “The use of data mining and neural networks for forecasting stock market returns”. *Expert Systems with applications* 29 (4): 927–940.

Fausett, Laurene. 1994. *Fundamentals of neural networks : architectures, algorithms, and applications.* Toimittanut Laurene Fausett. Englewood Cliffs (NJ): Prentice Hall International.

Frederic S., Mishkin. 2016. *The Economics of Money, Banking and Financial Markets, Global Edition.* Nide Eleventh edition. The Pearson Series in Economics. Pearson. ISBN: 9781292094182. http://search.ebscohost.com.ezproxy.jyu.fi/login.aspx?direct=true&db=nlebk&AN=1419607&site=ehost-live.

George E. P., Box, Jenkins Gwilym M., Reinsel Gregory C. ja Ljung Greta M. 2016. *Time Series Analysis : Forecasting and Control.* Nide Fifth edition George E.P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, Greta M. Ljung. Wiley Series in Probability and Statistics. Wiley. ISBN: 9781118675021. https://search-ebscohost-com.ezproxy.jyu.fi/login.aspx?direct=true&db=nlebk&AN=1061322&site=ehost-live.

Granger, Clive WJ. 1992. "Forecasting stock market prices: Lessons for forecasters". *International Journal of Forecasting* 8 (1): 3–13.

Graves, Alex. kirjoittaja. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks.* 146. Studies in Computational Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-642-24797-2.

Haenlein, Michael, ja Andreas Kaplan. 2019. "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence". *California management review* 61 (4): 5–14.

Hochreiter, Sepp, ja Jürgen Schmidhuber. 1997. "Long short-term memory". *Neural computation* 9 (8): 1735–1780.

Hommes, Cars H. 2001. "Financial markets as nonlinear adaptive evolutionary systems". *Quantitative Finance* 1 (1): 149.

Hyndman, Rob J, ja George Athanasopoulos. 2018. *Forecasting: principles and practice.* OTexts.

Hämäläinen, Karo, ja Jukka Oksaharju. 2016. *Sijoita kuin guru.* Hansaprint.

Jensen, Michael C. 1978. “Some anomalous evidence regarding market efficiency”. *Journal of financial economics* 6 (2/3): 95–101.

Jordan, Michael I, ja Tom M Mitchell. 2015. “Machine learning: Trends, perspectives, and prospects”. *Science* 349 (6245): 255–260.

Kingma, Diederik P, ja Jimmy Ba. 2014. “Adam: A method for stochastic optimization”. *arXiv preprint arXiv:1412.6980.*

Kwon, Seoyun J. 2011. *Artificial Neural Networks.* Engineering Tools, Techniques and Tables. Nova Science Publishers, Inc. ISBN: 9781617615535. https://search-ebscohost-com.ezproxy.jyu.fi/login.aspx?direct=true&db=nlebk&AN=439593&site=ehost-live.

Lo, Andrew W, ja A Craig MacKinlay. 1988. “Stock market prices do not follow random walks: Evidence from a simple specification test”. *The review of financial studies* 1 (1): 41–66.

Maggi, Lida Mercedes Barba. 2018. *Multiscale Forecasting Models.* Springer.

Makridakis, Spyros, ja Michele Hibon. 1997. “ARMA models and the Box–Jenkins methodology”. *Journal of forecasting* 16 (3): 147–163.

Malkiel, Burton G. 2003. “The efficient market hypothesis and its critics”. *Journal of economic perspectives* 17 (1): 59–82.

Malkiel, Burton G, ja Eugene F Fama. 1970. “Efficient capital markets: A review of theory and empirical work”. *The journal of Finance* 25 (2): 383–417.

McCorduck, Pamela, Marvin Minsky, Oliver G Selfridge ja Herbert A Simon. 1977. “History of artificial intelligence.” Teoksessa *IJCAI,* 951–954.

Mehtab, Sidra, Jaydip Sen ja Abhishek Dutta. 2020. “Stock price prediction using machine learning and LSTM-based deep learning models”. Teoksessa *Symposium on Machine Learning and Metaheuristics Algorithms, and Applications,* 88–106. Springer.

Mondal, Prapanna, Labani Shit ja Saptarsi Goswami. 2014. “Study of effectiveness of time series modeling (ARIMA) in forecasting stock prices”. *International Journal of Computer Science, Engineering and Applications* 4 (2): 13.

Nelson, David MQ, Adriano CM Pereira ja Renato A de Oliveira. 2017. “Stock market’s price movement prediction with LSTM neural networks”. Teoksessa *2017 International joint conference on neural networks (IJCNN),* 1419–1426. IEEE.

Nielsen, Michael A. 2015. *Neural networks and deep learning.* Nide 25. Determination press San Francisco, CA.

Nilsson, Nils J. 2009. *The quest for artificial intelligence.* Cambridge University Press.

Petram, Lodewijk Otto, ym. 2011. “The world’s first stock exchange: How the Amsterdam market for Dutch East India Company shares became a modern securities market, 1602-1700”. Tohtorinväitöskirja, Universiteit van Amsterdam [Host].

Poitras, Geoffrey. 2016. *Equity capital: From ancient partnerships to modern exchange traded funds.* Routledge.

Roondiwala, Murtaza, Harshal Patel ja Shraddha Varma. 2017. “Predicting stock prices using LSTM”. *International Journal of Science and Research (IJSR)* 6 (4): 1754–1756.

Siami-Namini, Sima, ja Akbar Siami Namin. 2018. “Forecasting economics and financial time series: ARIMA vs. LSTM”. *arXiv preprint arXiv:1803.06386.*

Siami-Namini, Sima, Neda Tavakoli ja Akbar Siami Namin. 2018. “A comparison of ARIMA and LSTM in forecasting time series”. Teoksessa *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA),* 1394–1401. IEEE.

Smith, Taylor G., ym. 2017–. *pmdarima: ARIMA estimators for Python.* [Online; accessed <today>]. `http://www.alkaline-ml.com/pmdarima`.

Sutton, Richard S, ja Andrew G Barto. 2018. *Reinforcement learning: An introduction.* MIT press.

Turing, Alan M. 1950. “Computing Machinery and Intelligence”. *Mind* 59 (October): 433–460. doi:`10.1093/mind/LIX.236.433`.

Williams, John Burr. 1938. *The theory of investment value.* Tekninen raportti.

Zhang, G Peter. 2003. “Time series forecasting using a hybrid ARIMA and neural network model”. *Neurocomputing* 50:159–175.

Liitteet

A ARIMA ennusteen lähdekoodi

In [17]:
```
def run_rolling_forecasting_auto_arima():
    # arima_data = full_data
    arima_data = testing_data
    history = [x for x in testing_data]
    predictions = [x for x in arima_data[:N_STEPS]]
    for t in range(N_STEPS,len(arima_data), LOOKUP_STEP):
        model = auto_arima(arima_data[t-N_STEPS:t], d=1, D=1)
        model.fit(arima_data[t-N_STEPS:t])
        output = model.predict(n_periods=LOOKUP_STEP)
        for x in range(LOOKUP_STEP):
            if t + x >= len(arima_data):
                break
            yhat = None
            yhat = output[x]
            #confidence.append(output.conf_int)
            predictions = np.append(predictions, yhat)
            obs = arima_data[t+x]
            history.append(obs)
            print("%d/%d predicted=%f, expected=%f" % (t+x, len(arima_data), yhat, obs))

    mae = mean_absolute_error(arima_data[N_STEPS:], predictions[N_STEPS:])
    rmse = mean_squared_error(arima_data[N_STEPS:], predictions[N_STEPS:], squared=False)
    mape = mean_absolute_percentage_error(arima_data[N_STEPS:], predictions[N_STEPS:])*100

    return predictions, rmse, mae, mape
```

B LSTM neuroverkon luonnin koodi

In [14]:
```
def create_lstm_model(length, features_number, units, layers_number, dropout,loss, optimizer):
    model = Sequential()
    for i in range(layers_number):
        if i == 0:
            model.add(LSTM(units, return_sequences=True, batch_input_shape=(None, length, features_number)))
        elif i == layers_number - 1:
            model.add(LSTM(units, return_sequences=False))
        else:
            model.add(LSTM(units, return_sequences=True))
        model.add(Dropout(dropout))
    model.add(Dense(1, activation="linear"))
    model.compile(loss=loss, metrics=['mae', 'mse', 'mape'], optimizer=optimizer)
    return model
```