Jussi Kauppinen

Voiko vähästä oppia - koneoppimisen haasteet pienellä aineistolla

Tietotekniikan kandidaatintutkielma

29. huhtikuuta 2019

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Jussi Kauppinen

Yhteystiedot: jussi.a.kauppinen@student.jyu.fi

Työn nimi: Voiko vähästä oppia - koneoppimisen haasteet pienellä aineistolla

Title in English: Machine learning with small data

Työ: Kandidaatintutkielma

Sivumäärä: 20+0

Tiivistelmä: Tämä kandidaatintutkielma käsittelee koneoppimista pienellä aineistolla. Koneoppimisessa kone parantaa suorituskykyään jonkin tietyn tehtävän ratkaisemiseksi itsenäisesti sitä mukaa kun lisää kokemusta tai dataa kertyy. Koneoppimisongelmat voidaan jakaa luokittelu- ja regressio-ongelmiin. Yleensä koneoppimistehtävät vaativat ison aineiston tarkan koneoppimismallin opettamiseksi, mutta usein kattavan aineiston hankkiminen muodostuu ongelmaksi. Tämän tutkielman tavoitteena on käydä läpi minkälaisia ongelmia koneoppimismallin opetuksessa ilmenee kun käytettävissä on pieni aineisto ja esitellä ratkaisuja näihin ongelmiin. Tutkielma tehtiin kirjallisuuskatsauksena. Tutkitut julkaisut käsittelivät edellä mainittuja ongelmia, sekä niihin kehiteltyjä ratkaisuja. Tutkielmassa selvisi, että pienellä aineistolla on haastavampaa opettaa hyvin yleistyvää koneoppimismallia, ja ylisovittumisen välttäminen on vaikeaa. Yleistymisen parantamiseksi esitellään keinotekoista lisädataa generoiva SMOTE-tekniikka, ja ylisovittumista yritetään saada kuriin regularisoinnin avulla.

Avainsanat: koneoppiminen, luokittelu, pieni data, pieni aineisto, regularisointi

Abstract: This bachelor's thesis deals with machine learning with little data. In machine learning, the machine improves its performance to solve a specific task independently as more experience or data accumulates. Machine learning problems can be divided into classification and regression problems. Usually, machine learning tasks require large data to train an accurate machine learning model, but often obtaining large enough data is problematic. The aim of this thesis is to review the problems encountered in training a machine learning model when there is only little data available and solutions to these problems. The thesis was made as a literature review. The publications examined deal with the above-mentioned

problems, as well as the solutions developed for them. In the thesis it became clear that it is more challenging to teach a machine learning model that generalizes well with little material, and it is difficult to avoid overfitting. In order to generalize better, we examine SMOTE technology to generate synthetic data and to prevent overfitting we talk about regularization.

Keywords: machine learning, classification, small data, regularization

Kuviot

Sisältö

1 Johdanto

Tämä tutkielma tarkastelee koneoppimiseen liittyviä haasteita, kun koneoppimismallin kouluttamiseen on saatavilla ainoastaan vähäinen tai muilla tavoin puutteellinen aineisto. Lisäksi tutkielmassa käydään läpi näiden haasteiden ratkaisemiseksi kehiteltyjä metodeja ja koneoppimista yleisesti. Koneoppimiselle on olemassa monia käytännön sovelluskohteita (Rudin ja Wagstaff 2014), mutta usein ongelmaksi muodostuu tarpeeksi kattavan aineiston hankkiminen (Roh, Heo ja Whang 2018). Aineiston ongelmia pienuuden lisäksi voi olla esimerkiksi puuttuva tieto. Puutteellinen aineisto voi aiheuttaa erilaisia ongelmia koneoppimisen eri vaiheissa. Tutkielmani tavoitteena on esitellä pienellä aineistolla tehtävän koneoppimisen ongelmia ja koota yhteen jo kehitettyjä ratkaisuja.

Luvussa 2 esitellään koneoppimisen perusteet ja käydään läpi koneoppimisalgoritmin eri elementit. Luvussa 3 käydään läpi pienellä aineistolla tehtävään koneoppimiseen liittyviä ongelmia. Luvussa 4 käydään läpi erilaisia ratkaisuja aikaisemmin esiteltyihin ongelmiin. Luku 5 sisältää yhteenvedon, jossa käydään läpi millaisin keinoin pienestä aineistosta voidaan yrittää saada aikaiseksi mahdollisimman tarkka koneoppimismalli.

2 Koneoppiminen

Koneoppiminen on tekoälyn osa-alue, jossa kone parantaa suorituskykyään jonkin tietyn tehtävän suhteen sitä mukaa kun lisää dataa tai kokemusta kertyy (Domingos 2012). Tässä luvussa käydään läpi koneoppimismallin eri elementit.

2.1 Koneoppimismallin elementit

Koneoppimismalli on matemaattinen esitysmuoto jollekin reaalimaailman ilmiölle. Koneoppimisen avulla ratkottavat ongelmat voidaan jakaa luokittelu- ja regressiotehtäviin. Luokittelussa yritetään ennustaa datan perusteella jokin diskreetti luokka, esimerkiksi kuuluuko sähköpostiviesti roskapostiin vai ei, kun taas regressio ennustaa jotakin jatkuvaa ominaisuutta, esimerkiksi kiinteistön myyntihintaa.

2.1.1 Mallin esitysmuoto

Koneoppimismallia luodessa täytyy valita sopiva luokitin, joka joko luokittelee sille annetun havainnon ennalta määrättyyn luokkaan, tai palauttaa vasteena jonkin jatkuvan muuttujan (Domingos 2012). Pienten aineistojen tapauksessa klassisten luokitteluongelmien ratkaisemiseen soveltuvat hyvin esimerkiksi tukivektorikone ja naiivi Bayes -luokitin (Forman ja Cohen 2004). Jatkuvien muuttujien luokitteluongelmia ratkotaan usein regressiomenetelmien avulla.

Jatkuvaa muuttujaa voidaan ennustaa lineaarisen regressiomallin avulla, jos selittävien muuttujien ja ennustettavan muuttujan välillä on lineaarinen riippuvuus. Tilanteessa, jossa on yksi selitettävä muuttuja y ja yksi selittävä muuttuja x, voidaan lineaarinen regressiomalli esittää muodossa

$$y = \beta_0 + \beta_1 x \tag{2.1}$$

Yhtälössä (2.1) β_0 on vakiotermi, joka kuvaa vastemuuttuja y odotettua arvoa tilanteessa,

jossa selittävän muuttujan x arvo on nolla. Kerroin β_1 kuvaa muutosta vastemuuttujan y odotusarvossa, kun selittävän muuttujan x arvo kasvaa yhdellä. Yhtälön β-kertoimet estimoidaan käyttämällä pienimmän neliösumman menetelmää (Gelman, Hill ym. 2007).

Tukivektorikone (engl. *Support Vector Machine*) toimii yksinkertaisimmillaan kahden eri luokan luokittelijana. Se luokittelee havainnot etsimällä hypertason, joka erottaa eri luokkiin kuuluvat datapisteet toisistaan. Tukivektorikoneita voidaan soveltaa myös useamman luokan luokitteluun kouluttamalla jokaiselle luokalle oma luokittelija, joka erottelee luokan havainnot muiden luokkien havainnoista (Cortes ja Vapnik 1995).

Naiivi Bayes -luokitin on yksinkertainen generatiivinen malli, jossa selittävien muuttujien oletetaan olevan toisistaan riippumattomia jokaisessa luokassa. Epärealistisista oletuksista huolimatta naiivi Bayes -luokitin on osoittautunut erittäin toimivaksi useissa eri sovelluskohteissa (Forman ja Cohen 2004).

2.1.2 Mallin evaluaatio ja optimointi

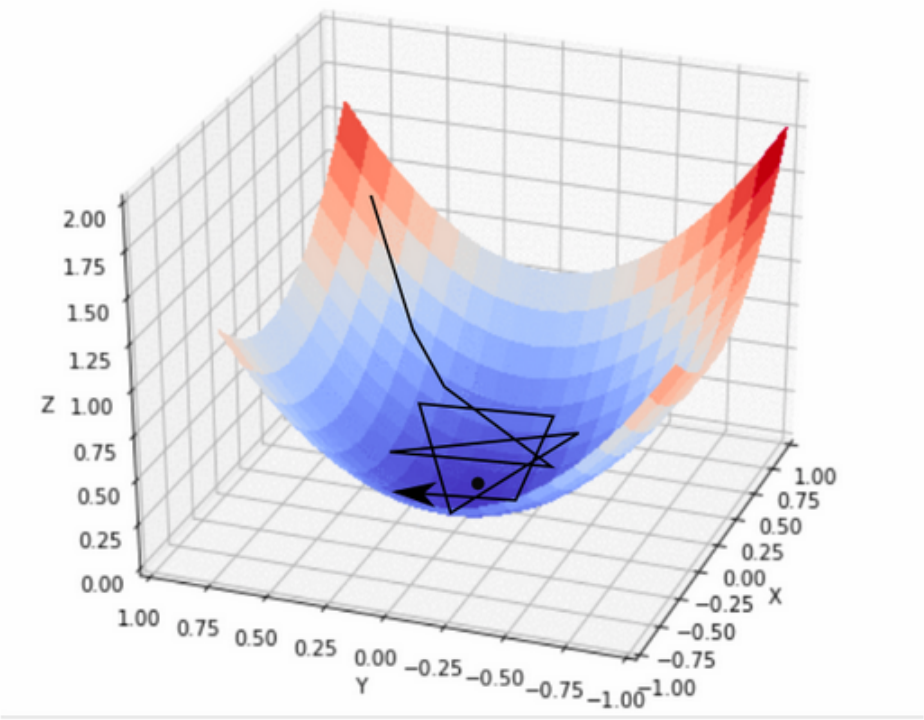
Jotta koneoppimismallille saadaan valittua oikeanlainen luokitin, tarvitaan virhefunktio erottelemaan hyvät luokittelijat huonoista. Virhefunktiolla lasketaan, kuinka kaukana mallin ennuste on oikeasta havainnosta, eli mitä lähempänä ennuste on vastaavaa havaintoa, sitä tarkempi malli on. Koneoppimismallia opetettaessa virhefunktio pyritään minimoimaan optimointimenetelmän avulla, jotta saadaan valittua paras mahdollinen luokitin ja mahdollisimman tarkka koneoppimismalli.

Keskineliövirhe (engl. *Mean squared error*) on yleinen ja yksinkertainen virhefunktio, jossa lasketaan keskiarvo ennusteiden ja todellisten havaintojen erotuksen neliöistä.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.2}$$

Yhtälössä (2.2) n on havaintojen määrä, Y_i todellinen havainto ja $\hat{Y}_i$ koneoppimismallin tekemä ennuste. Pienten aineistojen tapauksessa tavallisen keskineliövirheen käyttö johtaa herkästi mallin ylisovittumiseen ja tätä pyritään välttämään lisäämällä yhtälöön ns. "sakkotermi". Ylisovittumista käsitellään tarkemmin luvussa 3 ja sen välttämistä luvussa 4.

Yksi yleisimmistä virhefunktion optimointimenetelmistä on iteratiivinen laskevan gradientin menetelmä (engl. *Gradient descent*). Tätä menetelmää voidaan soveltaa myös pienten aineistojen tapauksessa.



Kuvio 1. Esimerkki laskevasta gradientista

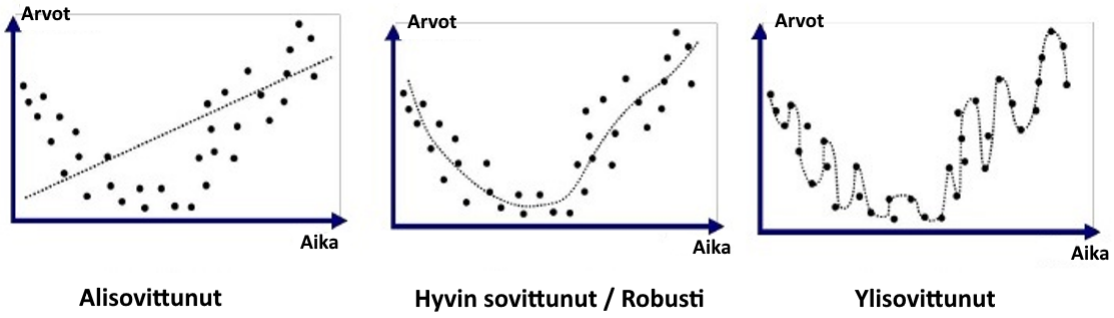
Kuviossa 1 x- ja y-akselit kuvaavat luokittimen painokertoimien arvoja ja z-akseli virhefunktion arvoa tietyillä painokertoimilla. Tavoitteena on löytää painokertoimien arvot, joilla virhefunktio minimoituu.

3 Koneoppimisen ongelmat pienellä aineistolla

Tässä luvussa esitellään yleisimpiä ongelmia, joita esiintyy koneoppimisessa. Luvussa esiteltyjä ongelmia esiintyy kaikenkokoisilla aineistoilla, mutta ne korostuvat pienten aineistojen tapauksissa.

3.1 Ylisovitus

Ylisovitus (engl. *overfitting*) tarkoittaa ilmiötä, jossa koneoppimismalli kuvaa liian tarkasti harjoitusaineiston, mistä seuraa virhe-ennusteita harjoitusaineiston ulkopuolelta tuleville havainnoille. Käytännössä tämän voi huomata koneoppimismallia opettaessa, jos mallin tarkkuus harjoitusaineistolla olisi esimerkiksi 95%, mutta testausaineistolla ainoastaan 55%.

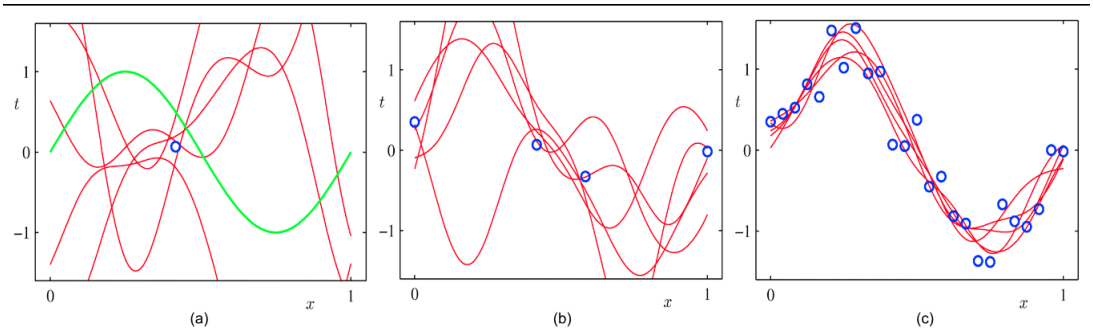


Kuvio 2. Esimerkkejä eri tavoilla sovittuneista malleista aineiston pysyessä samana.

Kuvio 2 havainnollistaa kuinka alisovittunut malli ei kuvasta datapisteiden mukaista ilmiötä, kun taas hyvin sovittunut malli kuvastaa ilmiötä ja on lisäksi robusti, eli muutamista poikkeavista havainnoista huolimatta malli onnistuu kuvailemaan todellista ilmiötä. Ylisovittunut malli kulkee "pisteestä pisteeseen", eli sopii täydellisesti harjoitusaineistoon, mutta mallintaa todellista ilmiötä huonommin kuin hyvin sovittunut malli. Kun dataa on vähän, ylisovittumisen välttämisestä tulee vaikeampaa ja poikkeavat havainnot ovat vaarallisempia.

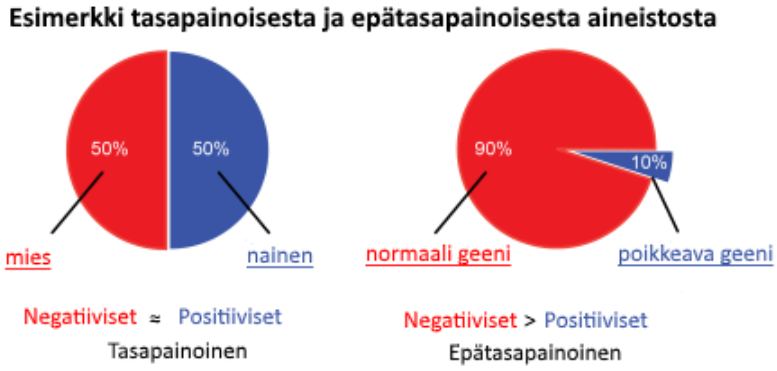
3.2 Yleistettävyyden haastavuus

Koneoppimisen tavoitteena on tuntemattomien havaintojen ennustaminen aiempien havaintojen perusteella. Eli halutaan saada oikea vastemuuttuja harjoitusaineiston ulkopuoliselle havainnolle, jonka oikeaa vastemuuttujaa ei ole annettu harjoitusaineistossa. Mallin kykyä ennustaa oikea vaste harjoitusaineiston perusteella kutsutaan sen yleistettävyydeksi (Alpaydin 2010). Kun käytettävää dataa on vähän, useammat erilaiset mallit sopivat aineistoon. Suurempi datamäärä tuottaa paremmin yleistyvän mallin (Bishop 2006).



Kuvio 3. Esimerkki mallin yleistettävyydestä. Havaintojen määrän kasvaessa sopivien mallien määrä on huomattavasti pienempi kuin yhden havaintopisteen tapauksessa.

3.3 Aineiston tasapainottomuus



Kuvio 4. Tasapainoisen ja epätasapainoisen aineiston välisen eron havainnollistaminen.

Aineiston tasapainottomuus tarkoittaa koneoppimisen kontekstissa tilannetta, jossa kaksi tai useaa eri luokkaa sisältävässä aineistossa eri luokkiin kuuluvien havaintojen määrä on eri-

suuri. Kuvio 4 havainnollistaa epätasapainoista aineistoa, jossa vähemmistöluokan havaintoja on ainoastaan 10% kaikista aineiston havainnoista. Epätasapainoisella aineistolla opetetun koneoppimismallin tarkkuuden arviointi voi olla vaikeaa, esimerkiksi kuvion 4 epätasapainoisessa aineistossa 90% havainnoista on normaaleja geenejä, jolloin malli, joka luokittelisi kaikki sille annetut havainnot normaaleiksi saavuttaisi 90%:n tarkkuuden, ilman että se on oppinut ilmiöstä yhtään mitään. Tarkkuuden arvioimisen haastavuuden lisäksi koneoppimismallilla voi olla vaikeuksia oppia vähemmistöluokan piirteitä epätasapainoisesta aineistosta (Batista, Prati ja Monard 2004).

4 SMOTE, bagging, boosting ja regularisointi - ratkaisuja koneoppimisen ongelmiin

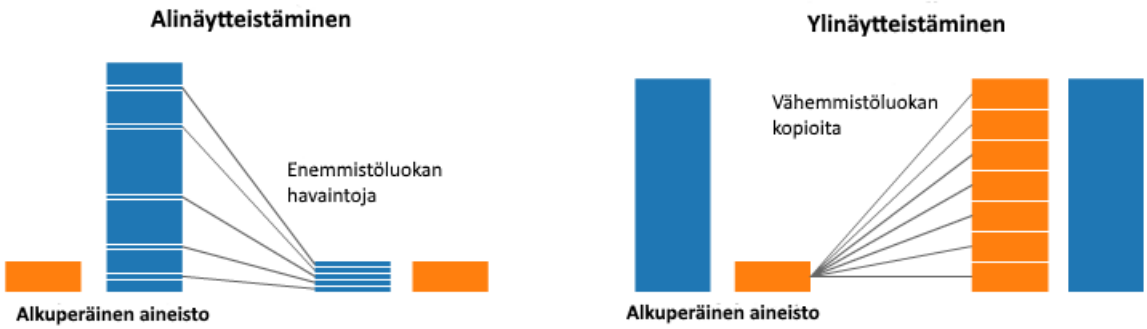
Tässä luvussa käydään läpi ratkaisuja luvussa 3 esiteltyihin ongelmiin. Alaluvussa 4.1 esitellään menetelmiä käytettävissä olevan aineiston muokkaamiseen ja uudelleenkäyttöön. Alaluku 4.2 käsittelee ylisovittumisen välttämistä.

4.1 Aineiston käsittely

Yksi tapa ratkaista pienen aineiston koneoppimiseen liittyviä ongelmia on muokata käytettävää aineistoa ennen sen käyttöönottoa. Epätasapainoista aineistoa voidaan tasapainottaa erilaisilla tasapainotusmenetelmillä. Pienten aineistojen tapauksessa on myös mahdollista hyödyntää aineistoa mahdollisimman paljon uudelleenkäyttämällä sitä. Aineiston tasapainottamiseen ja uudelleenkäyttämiseen käytettäviä menetelmiä esitellään tarkemmin seuraavissa alaluvuissa.

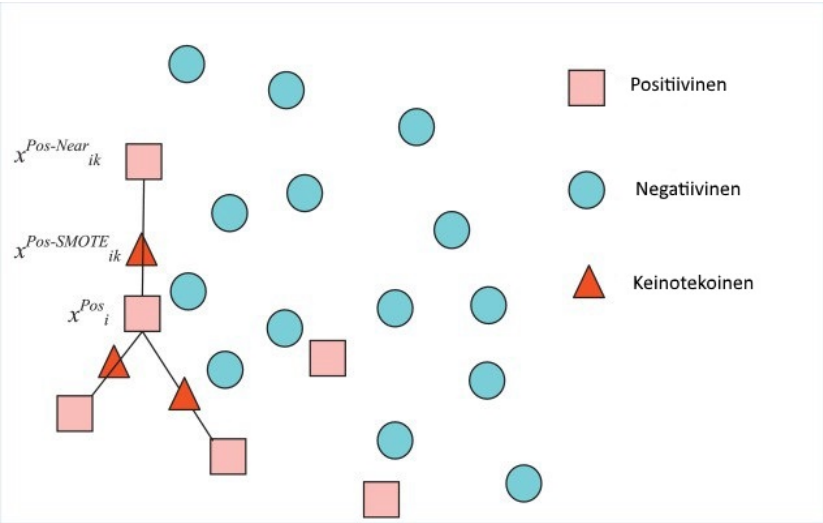
4.1.1 Aineiston tasapainottaminen ylinäytteistämällä: SMOTE

Epätasapainoinen harjoitusaineisto voidaan tasapainottaa käyttämällä yli- tai alinäytteistämistä (engl. *over-/under-sampling*). Ylinäytteistyksessä vähemmistöluokan havaintoja lisätään esimerkiksi keinotekoisesti, kun taas alinäytteistyksessä enemmistöluokan havaintojen määrää vähennetään. Pienten aineistojen kohdalla ei ole mielekästä soveltaa alinäytteistysmenetelmiä, koska se pienentäisi aineistoa entisestään. Tämän vuoksi tarkastelemme tarkemmin ainoastaan ylinäytteistämistä.



Kuvio 5. Esimerkki yli-/alinäytteistämisestä.

SMOTE (*Synthetic Minority Over-sampling Technique*) on ylinäytteistämismenetelmä, jossa aineiston kokoa kasvatetaan luomalla vähemmistöluokasta synteettisiä havaintoja. Uusia havaintoja luodaan käymällä läpi kaikki vähemmistöluokan datapisteet, valitsemalla datapisteen k-lähintä naapuria ja luomalla uuden datapisteen satunnaiseen kohtaan havainnon ja sen naapurin väliselle janalle. Synteettiset havainnot auttavat luokitinta luomaan suurempia ja vähemmän spesifisiä päätösalueita, minkä seurauksena malli yleistyy paremmin (Chawla ym. 2002).



Kuvio 6. Esimerkki SMOTE:n avulla luoduista datapisteistä.

4.1.2 Aineiston uudelleenkäyttömenetelmät: bagging ja boosting

Bagging (Bootstrap aggregating) on menetelmä, jossa valitaan m bootstrap-otosta harjoitusaineistosta, käydään läpi kaikki otokset ja opetetaan jokaisella sama luokittelija. Tällöin saadaan m hiukan erilaista samantyyppistä luokittelijaa. Näille luokittelijoille annetaan uusi tuntematon havainto, jonka jälkeen kerätään jokaisen luokittelijan tekemä ennuste uuden havainnon luokasta tai luokkatodennäköisyydestä. Uuden havainnon lopulliseksi luokaksi valitaan näiden ennusteiden moodi tai luokkatodennäköisyyksien keskiarvo (Forman ja Cohen 2004).

Boosting on iteratiivinen menetelmä, jonka tavoitteena on peräkkäin opetettavien luokittelijoiden avulla minimoida luokitteluvirhe. Ensimmäisellä iteraatiokerralla luokitin opetetaan koko opetusaineistolla, ja sen jälkeen opetusaineistoa muokataan systemaattisesti, jonka jälkeen luokitin muodostetaan uudelleen tällä uudella opetusaineistolla (Bishop 2006). Boosting-algoritmi yksinkertaistettuna:

- Oletetaan harjoitusaineisto $A = \{(x_1, y_1)...(x_n, y_n)\}$, missä y on piirrettä x vastaava luokka.
- Tehdään 1,...,K kertaa:
  - Valitaan harjoitusaineistosta A otos A_t
  - Opetetaan luokitin aineistolla A_t, muodostettu luokitin minimoi luokitteluvirheen joukolle A_t
- Lopuksi luokitellaan testiaineisto yhdistämällä K eri luokittimen ennusteet.

4.2 Ylisovittumisen välttäminen

Pienten aineistojen tapauksessa ylisovittuminen korostuu aineiston vähyyden vuoksi. Koska koneoppimisella pyritään selittämään havaittua ilmiötä, on ylisovittumista järkevää pyrkiä välttämään. Yleinen ylisovittumisen välttämiskeino on käyttää regularisointia, johon kuuluvat Ridge- ja Lasso-regressio. Näitä regressiomenetelmiä esitellään tarkemmin seuraavaksi.

4.2.1 Regularisointi

Regularisointi on tekniikka, jolla pyritään välttämään ylisovittumista (James ym. 2014). Tekniikan ideana on lisätä sakkotermi minimoitavaan virhefunktioon ja näin estää mallin parametreja saavuttamasta suuria arvoja. Kaksi yleisintä regularisointimenetelmää ovat Ridge- ja Lasso-regressio.

Ridge-regressiossa virhefunktioon lisättävänä sakkoterminä toimii kertoimien neliöiden summa. Ridge-regressio on muotoa

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2, \tag{4.1}$$

missä λ on tuunattava parametri, joka määrittelee kuinka paljon mallin joustavuudesta sakotetaan, n on havaintojen määrä ja p on selittävien muuttujien määrä.

Lasso-regressio on puolestaan muotoa

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|, \tag{4.2}$$

missä λ on tuunattava parametri, joka määrittelee, kuinka paljon mallin joustavuudesta sakotetaan, n on havaintojen määrä ja p on selittävien muuttujien määrä. Lasso eroaa Ridgestä siinä, että se sakottaa ainoastaan isoista kertoimista. Se käyttää sakkona kertoimien itseisarvoa neliön sijaan. Tilastotieteessä tämä itseisarvojen summa tunnetaan L1-normina. Molemmista regularisointitekniikoista on olemassa implementaatioita eri ohjelmointikielillä kuten R ja Python. R-kielessä Ridge- ja Lasso-regressiot on toteutettu esimerkiksi MASS-paketissa (Venables ja Ripley 2002) ja Pythonissa Scikit-learn-moduulissa (Pedregosa ym. 2011).

5 Yhteenveto

Koneoppimisella on useita eri sovelluskohteita nykypäivän maailmassa ja sitä halutaan soveltaa moniin eri tutkimusongelmiin. Tutkimusongelmien ratkaisuun tarvittavan tarpeeksi kattavan aineiston hankkiminen on usein haastavaa, ja estää koneoppimisen soveltamisen ongelman ratkaisemiseksi. Tässä tutkielmassa on esitelty koneoppimista yleisellä tasolla, pienellä aineistolla tehtävään koneoppimiseen liittyviä ongelmia ja esitelty muutamia ratkaisuja näihin pienen aineiston aiheuttamiin ongelmiin.

Koneoppimisen saralta esiteltiin yleisimmät oppimistehtävät mitä koneoppimismalleilla pystytään ratkomaan. Näitä tehtäviä ovat luokittelu- ja regressiotehtävät. Lisäksi esiteltiin koneoppimismallin eri elementit kuten luokitin, virhefunktio ja optimointimenetelmä. Luokittimista käytiin läpi tarkemmin luokitteluongelmiin soveltuvat tukivektorikone ja naiivi Bayes-luokitin, ja regressio-ongelmiin sopiva lineaarinen regressiomalli. Käytettävänä virhefunktiona esiteltiin keskineliövirhe ja virhefunktion minimointiin käytettiin iteratiivista laskevan gradientin menetelmää.

Koneoppimiseen liittyvistä ongelmista esiteltiin ne, jotka korostuvat erityisesti pienellä aineistolla. Tällaisia ongelmia ovat ylisovittuminen, aineiston epätasapaino ja yleistyvyyden puute. Ylisovittuminen ja yleistyvyyden puute vaikuttavat negatiivisesti mallin tarkkuuteen. Aineiston epätasapainon todettiin aiheuttavan vaikeuksia mallin tarkkuuden arvioimisessa, sekä vähemmistöluokan piirteiden oppimisessa.

Ratkaisuista esiteltiin erilaisia aineiston käsittelymenetelmiä, joiden avulla aineiston tasapainoa voidaan korjata ja pientä aineistoa hyödyntää tehokkaammin. Aineistoa voidaan tasapainottaa SMOTE-menetelmän avulla, ja pientä aineistoa uudelleen käyttää bagging- ja boosting-menetelmillä. Ylisovituksen estämiseksi tarkasteltiin regularisointimenetelmistä Ridge- ja Lasso-regressio.

Tutkielman tavoitteena oli auttaa lukijaa välttämään pienen aineiston aiheuttamia sudenkuoppia koneoppimismallia kehittäessä ja antaa työkaluja näiden ongelmien ratkaisemiseksi. Tutkimuksessa kävi ilmi, että tutkielmassa läpikäydyt menetelmät eivät takaa tarkkaa koneoppimismallia, vaan joissakin tapauksissa aineiston rajoitteet tulevat vastaan eikä ilmiötä

pystytä selittämään käytössä olevilla resursseilla.

Lähteet

Alpaydin, Ethem. 2010. *Introduction to Machine Learning.* 2nd. The MIT Press. ISBN: 026201243X, 9780262012430.

Batista, Gustavo E. A. P. A., Ronaldo C. Prati ja Maria Carolina Monard. 2004. "A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data". *SIGKDD Explor. Newsl.* (New York, NY, USA) 6, numero 1 (kesäkuu): 20–29. ISSN: 1931-0145. doi:10.1145/1007730.1007735. http://doi.acm.org/10.1145/1007730.1007735.

Bishop, Christopher M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Pg. 157-158. Berlin, Heidelberg: Springer-Verlag. ISBN: 0387310738.

Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall ja W. Philip Kegelmeyer. 2002. "SMOTE: Synthetic Minority Over-sampling Technique". *Journal of Artificial Intelligence Research* 16:321–357.

Cortes, Corinna, ja Vladimir Vapnik. 1995. "Support-vector networks". *Machine Learning* 20, numero 3 (syyskuu): 273–297. ISSN: 1573-0565. doi:10.1007/BF00994018. https://doi.org/10.1007/BF00994018.

Domingos, Pedro. 2012. "A Few Useful Things to Know About Machine Learning". *Commun. ACM* (New York, NY, USA) 55, numero 10 (lokakuu): 78–87. ISSN: 0001-0782. doi:10.1145/2347736.2347755. http://doi.acm.org/10.1145/2347736.2347755.

Forman, George, ja Ira Cohen. 2004. "Learning from Little: Comparison of Classifiers Given Little Training". Teoksessa *Knowledge Discovery in Databases: PKDD 2004,* toimittanut Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti ja Dino Pedreschi, 161–172. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-540-30116-5.

Gelman, A, J Hill ym. 2007. "Data analysis using regression and multilevel/hierarchical models".

James, Gareth, Daniela Witten, Trevor Hastie ja Robert Tibshirani. 2014. *An Introduction to Statistical Learning: With Applications in R.* Springer Publishing Company, Incorporated. ISBN: 1461471370, 9781461471370.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel ym. 2011. "Scikit-learn: Machine Learning in Python". *J. Mach. Learn. Res.* 12 (marraskuu): 2825–2830. ISSN: 1532-4435. http://dl.acm.org/citation.cfm?id=1953048.2078195.

Roh, Yuji, Geon Heo ja Steven Euijong Whang. 2018. "A Survey on Data Collection for Machine Learning: a Big Data - AI Integration Perspective". *CoRR* abs/1811.03402. arXiv: 1811.03402. http://arxiv.org/abs/1811.03402.

Rudin, Cynthia, ja Kiri L. Wagstaff. 2014. "Machine learning for science and society". *Machine Learning* 95, numero 1 (huhtikuu): 1–9. ISSN: 1573-0565. doi:10.1007/s10994-013-5425-9. https://doi.org/10.1007/s10994-013-5425-9.

Venables, W. N., ja B. D. Ripley. 2002. *Modern Applied Statistics with S.* Fourth. ISBN 0-387-95457-0. New York: Springer. http://www.stats.ox.ac.uk/pub/MASS4.