TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

LASSI HÄMÄLÄINEN

KÄYTTÄJÄTYYTYVÄISYYDEN ENNUSTAMINEN TEKSTIMUOTOISISTA ARVOSTELUISTA

Kandidaatintyö

Tarkastaja: Prof. Heikki Huttunen
Jätetty tarkastettavaksi 30.04.2018

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Tietotekniikan koulutusohjelma, Signaalinkäsittelyn laboratorio
LASSI HÄMÄLÄINEN: Käyttäjätyytyväisyyden ennustaminen tekstimuotoisista arvosteluista
Kandidaatintyö, 19 sivua
Huhtikuu 2018
Pääaine: Machine Learning
Tarkastaja: Prof. Heikki Huttunen
Avainsanat: Koneoppiminen, käyttäjätyytyväisyyden arviointi, TF-IDF, satunnainen metsä, tukivektorikone

Tässä työssä hyödynnetään piirteenirrotus- ja luokittelumenetelmiä ja testataan niiden toimivuutta käyttäjätyytyväisyyden arviointiin tekstimuotoisesta aineistosta. Piirteiden irrottamiseen tekstistä käytetään TF-IDF-algoritmia, jonka antamat piirteet syötetään vertailtaville koneoppimismenetelmille. Käytettävät koneoppimismenetelmät ovat satunnainen metsä ja tukivektorikone, josta käytetään lineaarista ja radiaalista kerneliä käyttäviä toteutuksia. Koneoppimismenetelmistä vertaillaan sekä luokitteluun että regressioon perustuvia versioita menetelmistä. Valitut menetelmät ovat alan julkaisujen perusteella yleisesti käytössä tekstin merkityksen ja sävyn analysointiin liittyvissä ongelmissa.

Algoritmien esittelyn lisäksi käydään läpi aineiston käsittelystä alkaen koko aiheeseen liittyvä koneoppimisprosessi. Työssä esitellään algoritmien testauksen tulokset ja arvioidaan niiden pohjalta käytettyjen menetelmien soveltuvuutta käyttäjätyytyväisyyden ennustamiseen tekstimuotoisten arvostelujen pohjalta.

SISÄLLYS

LYHENTEET JA MERKINNÄT

JSON	*JavaScript Object Notation*, yksinkertainen ja yleinen tekstimuotoinen dataformaatti
MAE	*Mean Absolute Error*, keskimääräinen virheen itseisarvo
MSE	*Mean Squared Error*, keskimääräinen neliövirhe
RBF	*Radial Basis Function*, tukivektorikoneen radiaalinen kernelifunktio
TF-IDF	*Term Frequency-Inverse Document Frequency*, piirteidenirrotusmenetelmä, joka perustuu sanojen painokerrointen laskemiseen niiden esiintymistiheyden perusteella

1. JOHDANTO

Viime aikoina on havaittu kerätyn datan arvokkuus sekä suuren datamäärän ja koneoppimisen yhdistelmän tuomat mahdollisuudet. Suuren ja monimutkaisen datan käsittelyyn on usein haastavaa kirjoittaa algoritmia, joka ratkaisisi ongelman suoraan ja parhaalla mahdollisella tavalla. [6] Datan riippuvuussuhteet ratkaistavan ongelman suhteen ovat usein monikäsitteisiä ja haastavia tulkita. Näiden ongelmien ratkaisuun on kehitetty koneoppimisen menetelmiä, joilla suurta datamäärää voidaan hyödyntää tehokkaasti erilaisten ratkaisujen löytämiseen pienemmällä määrällä tarvittavia työtunteja ratkaisun kehittämiseen. [5]

Koneoppiminen on laajasti käytössä tekstin tunnistamisessa ja muokkaamisessa. Monia verkkokauppoja ja palveluita kiinnostaa erityisesti tekstin sävyn tunnistaminen. Tekstin sävyn tunnistamisella tarkoitetaan tekstin merkityksen analysoimista siten, että pystytään tunnistamaan, millaisen kuvan kirjoittaja haluaa luoda tekstillään. Tekstistä voidaan pohtia, pyrkiikö kirjoittaja olemaan vakava vai humoristinen ja onko kyseessä esimerkiksi satiirisesti kirjoitettu teksti. Tuotearvosteluissa kiinnostaa yrityksiä asiakkaan suhtautuminen tuotteeseen ja tekstin sävyn tunnistamisesta saadaan arvokasta lisätietoa tyypillisen numeroarvosanan rinnalle, jonka asteikon suuri subjektiivisuus hankaloittaa arvostelujen vertailua. [19]

Tässä tutkielmassa pyritään arvioimaan arvostelujen tekstistä tuotteelle annettu arvosana ja vertailemaan erilaisia koneoppimisalgoritmeja. Tarkasteltavista tekniikoista keskitytään yksinkertaisiin ja nopeisiin menetelmiin esimerkiksi syvien neuroverkkojen sijaan. Käytettävänä materiaalina on Amazon-verkkokaupan arvosteluja, jotka sisältävät otsikon, tekstin ja kokonaisluku arvosanan 1 ja 5 välillä. Näiden arvostelujen otsikoiden ja tekstin perusteella arvioidaan Pythonin scikit-learn-kirjastoa hyödyntävän ohjelman avulla vastaava arvosana. Tutkimuskysymys voidaan muotoilla seuraavasti: *Kuinka hyvin tarkasteltavat koneoppimismenetelmät soveltuvat käyttäjätyytyväisyyden arviointiin tekstiarvostelun pohjalta?*

Seuraavassa toisessa luvussa käsitellään koneoppimismenetelmien teoreettista pohjaa. Prosessia kuvataan vaiheittain aloittaen aineiston käsittelystä, jatkaen piirteiden irrotuksella ja lopuksi käsittelemällä luokittelumenetelmiä. Näiden vaiheiden ja

niihin liittyvien algoritmien esittelyn jälkeen käsitellään vielä virheenarviointia ristiinvalidoinnin avulla ja tulosten luotettavuutta. Luku kolme käsittelee menetelmien toteutusta ja niistä saatuja tuloksia. Luvun alussa esitellään käytetyt työkalut ja aineisto, minkä jälkeen kuvataan työn toteutusta ja lopuksi vertaillaan siitä saatuja tuloksia. Viimeisessä, eli neljännessä luvussa kerrotaan johtopäätöksistä, joita työn toteutuksesta ja sen tuloksista muodostettiin.

2. TEORIA

Tässä luvussa käsitellään regressio- ja luokitteluongelmia ratkaisevien koneoppimismenetelmien teoriaa ja esitellään tarkemmin työssä käytettyjä algoritmeja. Tämän kaltaisissa ongelmissa voidaan usein jakaa menetelmät kolmeen ryhmään: aineiston käsittely, piirteenirrotus ja luokittelu tai regressio.

Aineiston käsittelyssä pyritään muokkaamaan aineisto paremmin käsiteltävään muotoon. Käytettävää aineistoa voidaan rajata, tarpeeton data poistaa ja dataformaattia vaihtaa, jos siten voidaan parantaa aineiston toimivuutta seuraavissa vaiheissa. Tekstin käsittelyssä on tyypillisesti käytössä useita menetelmiä, esimerkiksi kirjainkoon vakioiminen, erikoismerkkien poistaminen tai vaihtaminen ja yleisimpien sanojen poistaminen, joilla pyritään yksinkertaistamaan tekstiä. Tekstin sanat voidaan myöskin muuttaa perusmuotoihinsa, mistä erityisesti on hyötyä sanojen määrien laskentaan perustuvien piirteenirrotusmenetelmien kanssa, jotka eivät osaa normaalisti tunnistaa saman sanan eri taivutusmuotoja samaksi sanaksi. Kaiken tämän tavoitteena on helpottaa ja nopeuttaa piirteenirrotusta ja parantaa tuloksia. [21]

Piirteenirrotus (engl. *feature extraction*) tarkoittaa datan muokkaamista ja muuntamista lukuarvoiksi, joita voidaan käsitellä koneoppimisenmenetelmin. Tekstin tai kuvan muodossa oleva data sisältää paljon tietoa, mutta vain pieni määrä siitä saattaa olla merkityksellistä ratkaistavan ongelman kannalta. Piirteenirrotus pyrkii erottamaan datasta sopivat yksityiskohdat ja muokkaamaan ne piirrevektoriksi, joka on lista datan piirteitä kuvaavista numeroarvoista. Tämä myöskin vähentää koneoppimismenetelmille syötettävän datan määrää, mikä nopeuttaa opettamista. [12] Piirrevektoreista käytetään merkintää $\boldsymbol{x}_i \in \mathbb{R}^P$, missä i on näytteen indeksi aineistossa ja P on piirteiden määrä eli piirrevektorin koko.

Luokittelua ja regressioanalyysiä varten aineisto on jaettava opetus- ja testausaineistoksi. Opetusaineistoa käytetään opetusvaiheessa kouluttamaan koneoppimismenetelmä ongelman ratkaisuun ja testausaineistoa käytetään testaamaan opetusvaiheesta saatua mallia sille tuntemattomalla aineistolla, minkä takia opetus- ja testausaineisto eivät saa sisältää samoja näytteitä. Testaaminen aineistolla, jota ei ole käytetty opetukseen, on erittäin tärkeää ylioppimisen tunnistamiseksi. Yliop-

piminen tarkoittaa tilannetta, jossa koneoppimismetodi oppii erikoistumaan opetusaineistoon eikä enää kuvaa tehokkaasti aineistoa yleisesti. Tämä voidaan havaita koulutuksen aikana siitä, että opetusaineistolla saatava tarkkuus kasvaa korkeaksi, mutta testausaineistolla tarkkuus laskee koulutuksen edetessä. [18, s. 827]

Luokittelulla tarkoitetaan näytteiden jakamista ennalta määriteltyihin ryhmiin, regressioanalyysissä sen sijaan ei ole erillisiä ryhmiä vaan näytteen pohjalta algoritmi tuottaa yksittäisen luvun, joka pyrkii kuvaamaan selitettävää muuttujaa. Luokittelussa piirrevektorin pohjalta annetaan näytteelle jokin luokista *1, 2,...,C*. Regressiossa regressiofunktio ottaa piirrevektorin ja palauttaa sen perusteella lukuarvon. Regressiossa Luokittelu ja regressio eivät ole toisiaan pois sulkevia vaan on olemassa laaja joukko ongelmia, joihin molemmat menetelmät soveltuvat. Esimerkiksi iän arviointia tai kouluarvosanan ennustamista voidaan lähestyä sekä luokittelun että regressioanalyysin kannalta. Yhteistä tällaisille ongelmille on tavoiteltu arvot ovat lukuja, joilla on ongelman kannalta selkeä suuruusjärjestys ja suhde toisiinsa. [5]

2.1 Sanojen karsinta

Sanojen karsinnalla eli stemmauksella (engl. *stemming*) tarkoitetaan sanojen päätteiden poistamista sanan vartalon muodostamiseksi. Tämä on tyypillinen tekstinkäsittelyoperaatio, jolla voidaan yksinkertaistaan tekstiä ja muuttaa saman sanan eri taivutusmuodot samanlaisiksi. Sanojen laskentaan perustuvat algoritmien toteutukset eivät tyypillisesti pysty tunnistamaan sanoja samoiksi, ellei niiden kirjoitusasu ole täsmälleen sama. Esimerkiksi erilaisissa hakemistoindekseissä usein käytetään karsittuja muotoja sanoista ja hakusanat karsitaan ennen kuin niitä aletaan etsiä hakemistosta. [9]

Snowball on pieni tekstin stemmaukseen tarkoitettu tekstinkäsittelykieli, jolle on tehty useille eri kielille algoritmit sanojen karsimiseen. Sille tehtyjen toteutusten pohjalta luotuja toteutuksia eri kielille käytetään laajasti tekstin käsittelyssä. Algoritmien tuottamat sanojen rungot eivät ole aina täysin ideaalisia ja tyypillisistä säännöistä poikkeavilla sanoilla esiintyy alistemmausta, ylistemmausta ja väärinstemmausta. Alisteammauksessa sanasta poistetaan liian vähän merkkejä ja osa taivutuksesta jää mukaan. Tällöin ei saavuteta tavoitetta, että kaikki sanan taivutusmuodot saisivat saman muodon. Ylistemmauksessa sanasta poistetaan liikaa, jolloin useat eri sanat voivat saada saman muodon. Väärinstemmauksessa taas sanan osa tunnistetaan virheellisesti päätteeksi ja poistetaan tarpeettomasti. [8]

2.2 TF-IDF-piirteenirrotus

Piirteenirrotukseen on monia menetelmiä, mutta yksi yleisimmistä menetelmistä tekstin piirteiden irrottamiseen on TF-IDF (engl. *term frequency-inverse document frequency*). Nimensä mukaisesti TF-IDF laskee arvon sanan esiintymistiheydestä näytteessä kerrottuna sanan käänteisellä esiintymistiheydellä koko aineistossa. Sanat, joilla on suuri TF-IDF arvo, esiintyvät kyseisessä näytteessä enemmän kuin tyypillisesti koko aineiston kannalta, mikä viittaa sanan tärkeyteen kyseisessä näytteessä. [15]

TF-arvona voidaan käyttää pelkästään sanan esiintymiskertojen lukumäärää näytteessä, mutta useimmissa sovelluksissa käytetään logaritmisesti skaalattua versioita. Olkoon t sana, jonka TF-IDF arvoa lasketaan, ja d näyte, jossa sana t esiintyy. Tällöin TF-arvon kaavaksi saadaan

$$tf(t,d) = log(1 + f_{t,d}), \tag{2.1}$$

jossa $f_{t,d}$ on sanan t esiintymiskertojen lukumäärä näytteessä d. [11, s. 73]

IDF-arvo kuvaa kuinka harvinainen tai yleinen sana on koko aineiston kannalta. Se lasketaan jakamalla koko aineiston näytteiden määrä N, sanan sisältävien näytteiden määrällä n_t, ja ottamalla tuloksesta logaritmin eli

$$idf(t,D) = log\frac{N}{n_t} = log\frac{|D|}{|\{d \in D : t \in d\}|}, \tag{2.2}$$

jossa D on koko aineiston näytteiden joukko. Harvinaisempi sana saa siis suuremman IDF-arvon. TF-IDF-arvo saadaan näiden kahden funktion tulona eli:

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D), \tag{2.3}$$

joten näytteessä erittäin harvinaiset sanat ja aineistossa erittäin yleiset sanat saavat pienen TF-IDF-arvon. [11, s. 74] Yleisimmät sanat, joiden tiedetään jo etukäteen olevan erittäin yleisiä, jätetään usein kokonaan käsittelemättä, näistä sanoista käytetään nimitystä hukkasanat (engl. stop words) [10, s. 71–72].

2.3 Tukivektorikone

Tukivektorikone (engl. *Support Vector Machine, SVM*) on laajasti käytetty koneoppimismenetelmä kuvan ja tekstin luokittelussa. Tukivektorikone pyrkii mak-

simoimaan luokkien välisen etäisyyden muunnetussa piirreavaruudessa sovittamaan kahden luokittelujoukon väliin tason, jonka lineaarinen etäisyys sen kanssa yhden suuntaisiin marginaalitasoihin on mahdollisimman suuri. Näytteitä, jotka rajoittavat marginaalitasoja, kutsutaan tukivektoreiksi ja ne määrittävät luokittelun lopputuloksen. Luokittelutason määrittää kaava

$$f(\boldsymbol{x}) = \boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{\beta} + \beta_0 = 0, \tag{2.4}$$

missä $\boldsymbol{\beta} \in \mathbb{R}^P$ on yksikkövektori ja $\beta_0 \in \mathbb{R}$. Piirrevektori $\boldsymbol{x}_i$ luokitellaan kuuluvaksi luokkaan 1, kun $f(\boldsymbol{x}_i) > 0$, muutoin luokkaan 2. [7, s. 417]

Yleensä luokiteltavat opetusjoukot eivät ole lineaarisesti separoituvia eli niitä ei voida luokitella täysin oikein yhdellä tasolla kahteen ryhmään. Tällöin väärin luokitellut näytteet pitää ottaa huomioon marginaalin optimointiongelmassa ja heikentää tulosta suhteessa marginaalitasojen ulkopuolella olevien pisteiden etäisyyteen marginaalista. Näistä etäisyyksistä käytetään nimitystä *slack*-vakiot. Käyttämällä erilaisia kertoimia, eli regularisointitermejä C, pystytään säätämään virheellisten luokittelujen vaikutusta. Pienemmällä arvolla virheelliset luokittelut vaikuttavat vähemmän kuin suurella C:*n* arvolla, joka taas toimii usein paremmin, kun datassa on paljon hajontaa. [13]

Yksinkertainen lineaarinen kuvaus piirreavaruudesta ei välttämättä kuvaa tehokkaasti ongelmaa. Näytejoukko voidaan kuvata epälineaarisesti korkeampiulotteisessa avaruudessa, jolloin tukivektorikone voi löytää paremman ratkaisun näytteiden jakamiseen hypertasolla. Kaikkien näytteiden muokkaaminen korkeampaan ulottuvuuteen ei kuitenkaan ole tarpeen tukivektorikoneen tapauksessa, koska käyttämällä ns. kernel trickiä, tukivektorikone pystyy kuvaamaan päätöspintaa epälineaarisesti. Tällöin päätöspinta voi olla piirreavaruuden kannalta esimerkiksi elliptinen normaalin lineaarisen sijaan. Näitä tukivektorikoneen luokittelua muokkaavia funktioita kutsutaan kernelifunktioiksi. Lineaarinen kernelifunktio on muotoa

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle, \tag{2.5}$$

missä $\boldsymbol{x}_i$ on tukivektori ja $\boldsymbol{x}$ mielivaltainen luokiteltava piirrevektori. Muut yleisimmin käytetyt kernelifunktiot ovat polynominen

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = (\gamma \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle + r)^d, \tag{2.6}$$

missä d on positiivinen kokonaisluku, $\gamma > 0$ ja $r \in \mathbb{R}$, ja radiaalinen kernelifunktio

eli RBF-kerneli (*radial basis function*)

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = \exp(-\gamma||\boldsymbol{x} - \boldsymbol{x}_i||^2), \tag{2.7}$$

missä $\gamma > 0$. Radiaalinen kernelifunktio muokkaa tukivektorikoneen päätöspinnan elliptiseksi. [17, s. 111–123]

Tavallinen tukivektorikone soveltuu luokittelemaan näytteet vain kahteen luokkaan, mutta myös useampia luokkia tukevia muunnelmia löytyy [7, s. 438]. Yleistä pienellä määrällä luokkia on kuitenkin käyttää "yksi-vastaan-muut"tai "yksi-vastaan-yksi-lähestymistapoja ongelmaan ja kouluttaa useampi tukivektorikone arvioimaan luokkia erikseen ja valita niiden kaikkien pohjalta todennäköisin vaihtoehto [1].

Tukivektorikone soveltuu myös regressio-ongelmiin. Tällöin taso pyritään sovittamaan opetusdataan siten, että kaikki opetusdatan piirrevektorit ovat tietyn valitun marginaalin etäisyydellä tasosta. Samaan tapaan kuin luokittelussakin marginaalien väärälle puolelle jääviä arvoja rangaistaan slack-vakioiden ja regularisointitermin avulla. Marginaalia pienentämällä saadaan tukivektorikone reagoimaan enemmän pieniin eroihin ennustetuissa arvoissa ja opetusaineiston kohdearvoissa, suuremmat marginaalin arvot taas tasoittavat tukivektorikoneen regression tulosta koska pienet virheet, jotka jäävät marginaalien sisään, jätetään huomioimatta.

2.4 Satunnainen metsä

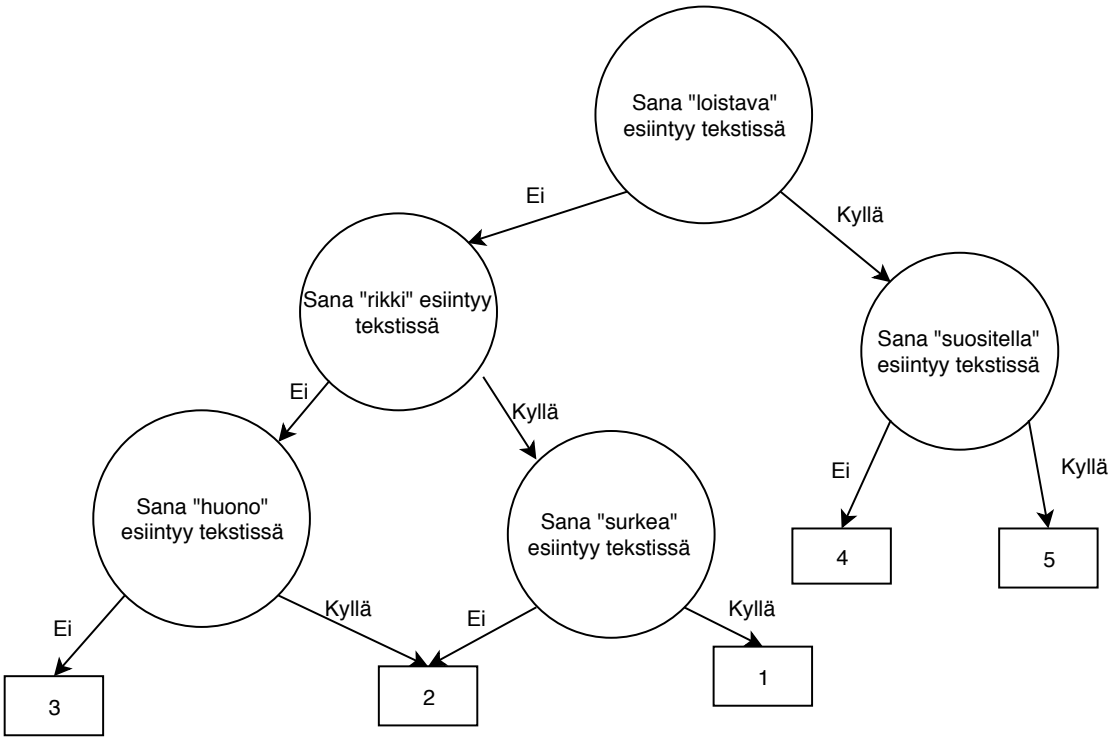
Satunnainen metsä (engl. *Random forest*) on luokittelija, joka perustuu usean osittain satunnaisesti luotujen päätöspuun (engl. *decision trees*) käyttöön, mistä algoritmi on saanut nimensä. Luokiteltaessa näyte luokitellaan kaikilla päätöspuilla ja suosituin luokka valitaan lopputulokseksi. [7, s. 587]

Puurakenteiset luokittelijat eli päätöspuut perustuvat piirteiden tarkasteluun yksi kerrallaan jokaisessa binääripuun solmussa päätyen lopulta lehteen joka määrittää valitun luokan. Esimerkki päätöspuun rakenteesta on kuvassa 2.1. Päätöspuut ovat erittäin taipuvaisia ylioppimiselle, koska puun luominen perustuu suoraan siihen käytetyn aineiston yksityiskohtiin, mitä satunnaisessa metsässä puiden ylioppimisongelmaa korjataan käyttämällä useampia puita [20, s. 81].

Satunnaisen metsän puut luodaan käyttämällä satunnaisesti valittua osajoukkoa opetusaineiston näytteistä ja piirteistä näytetään kullekin puulle luontivaiheessa ainoastaan osa. Näiden rajausten ansiosta puut ovat hieman erilaisia ja huomioivat eri piirteitä valintaprosessissa. Yksittäisten puiden syvyyttä rajataan, etteivät ne yliopi

aineistoa ja käyttävät vain tärkeimmiksi tulkittuja piirteitä. Kun useampi päätöspuu keskittyy hieman eri piirteisiin, tulosten tarkkuus kasvaa. Satunnaista metsää voidaan käyttää myös regressioanalyysissä. Tällöin puiden valitsemista arvoista otetaan keskiarvo yleisimmän arvon sijaan. [7, s. 588]

Satunnaisen metsän oppimista voidaan seurata tehokkaasti opetusvaiheen aikana ns. Out-of-Bag-ennustusten avulla. Tällöin valitaan opetusaineistosta näyte ja testataan sitä kaikilla puilla, joiden luomisessa kyseistä näytettä ei käytetty. Kaikkien puista saatujen vastausten perusteella tavalliseen tapaan joko äänestetään tai lasketaan keskiarvo riippuen, onko kyseessä luokittelu vai regressio, ja näin saadaan ennuste. Toistamalla tämä usealle näytteelle, ja laskemalla keskimääräisen virheen itseisarvo tuloksista, voidaan arvioida satunnaisen metsän tehokkuutta pelkän opetusaineiston avulla opetusvaiheen aikana. [22, s. 165–166]



Kuva 2.1 *Esimerkki päätöspuusta, joka arvioi käyttäjän antamaa arvosanaa tekstissä esiintyvien sanojen perusteella*

2.5 Virheen arviointi

Tieteellisessä tutkimuksessa virheen arvioiminen on aina tärkeää tulosten luotettavuuden määrittämisen kannalta. Koneoppimisen tapauksessa tähän on vakioitunut useita tähän kehitettyjä menetelmiä. Menetelmät vertailevat luokittelun tai regressioanalyysin tuloksia testausaineistossa lueteltuihin tavoitearvoihin ja laskevat oikei-

ta luokitteluja tai virhettä tulosten ja tavoitearvojen välillä. Näiden menetelmien antamien tulosten pohjalta voidaan tehdä kaavioita ja taulukoita havainnollistamaan eroavaisuuksia vertailtavien koneoppimismetodien antamien tulosten välillä.

Virhettä arvioitaessa on otettava huomioon, että koneoppimismetodin tarkkuuteen vaikuttaa aineiston tarkkuus. Virheet opetusaineistossa tai testausaineistossa vaikuttavat suoraan saatuihin tuloksiin. Aineiston sisältämä tieto on myös yleensä rajattua, eikä kaikkea mahdollista tietoa, joka tarvittaisiin tarkan luokittelun tai regression tekemiseen, ole saatavilla aineistosta. Toisaalta aineistossa esiintyvät virheet ovat samat kaikille vertailtaville algoritmeille, vaikka koneoppimismetodien käyttäytymisessä virhearviointien suhteen onkin eroavaisuuksia, mikä pienentää aineiston virheiden vaikutusta nimenomaan vertaillessa koneoppimismetodeja toisiinsa käyttäen täsmälleen samaa aineistoa.

Opetus- ja testausaineiston jako voi vaikuttaa selvästi tuloksiin, tästä aiheutuvaa virhettä pyritään arvioimaan ja rajoittamaan ristiinvalidoinnin (engl. *cross-validation*) avulla. Ristiinvalidoinnissa aineisto jaetaan useaan kertaan opetus- ja testausaineistoksi ja koneoppimisalgoritmit opetetaan ja testataan useampaan kertaan. Jokaisesta testauskerrasta mitataan erikseen tarkkuutta, joista laskemalla keskiarvo saadaan tulos, jossa aineiston jaon vaikutus on pienempi. Ristiinvalidointikertojen tulosten välisestä keskihajonnasta saadaan myös tietoa siitä kuinka paljon eri koneoppimismenetelmiin vaikuttaa aineistonjako verrattuna toisiin vertailtaviin menetelmiin. Tyypillinen ristiinvalidointimenetelmä on K-ositettu ristiinvalidointi (engl. *K-fold cross-validation*). Siinä aineisto jaetaan K:hon yhtä suureen osaan, joista yhtä vuorollaan käytetään testausaineistona ja loppuja opetusaineistona.

Tulosten luotettavuuden kannalta pitää myös pohtia, miten hyvin käytetyt metodit ja niille annetut parametrit soveltuvat ongelman ratkaisuun. Väärät valinnat käytetyissä piirteidenirrotus- ja koneoppimismenetelmien yhdistelmissä voivat voimakkaasti vääristää tuloksia. Järkevät oletusparametrit testauksessa ja useiden eri parametrien arvojen kokeileminen laajentavat testattujen kombinaatioiden määrää ja parantavat tulosten luotettavuutta yleiseltä kannalta.

3. TESTAUS JA TULOKSET

Tämä luku käsittelee koneoppimisalgoritmien testausta ja tuloksia kommentteihin liittyvän arvosanan arvioinnissa. Testauksessa ja datan käsittelyssä käytettiin sitä varten kirjoitettua ohjelmaa, joka yksikerrallaan opettaa kunkin koneoppimismenetelmän mallin ja testaa sitä. Vertailemalla saatuja tuloksia kunkin koneoppimismenetelmän tapauksessa voidaan arvioida menetelmien soveltuvuutta käyttäjätyytyväisyyden ennustamiseen tuotearvostelutekstistä.

Tässä työssä ohjelman kirjoitukseen käytettiin Python-ohjelmointikieltä[1], joka on noussut erityisen suosituksi kieleksi koneoppimiseen liittyvien ongelmien ratkaisussa sen helppouden ja laajan kirjastovalikoiman ansiosta [14]. Koneoppimisessa työssä hyödynnettiin scikit-learn-kirjastoa[2], jossa on laaja valikoima toteutuksia perinteisistä koneoppimisalgoritmeista sekä hyödyllisiä funktioita koneoppimiseen ja datan käsittelyyn. Lisäksi NLTK-kirjastoa[3] hyödynnettiin aineiston tekstin käsittelyssä.

3.1 Aineisto

Koneoppimismenetelmiä hyödynnettäessä tarvitaan aina aineisto menetelmien opettamiseen ja testaamiseen. Aineiston valinta on tärkeä ongelma, koska se vaikuttaa suuresti saatuun lopputulokseen. Jos aineisto ei kuvaa tehokkaasti ratkaistavaa ongelmaa, ei sen pohjalta opetettu koneoppimismallikaan kuvaa ongelmaa hyvin, mikä johtaa heikkoihin tuloksiin käytännössä. Opetusvaiheessa algoritmille syötetään opetusaineistoa ja testausvaiheessa siihen varatun aineiston perusteella arvioidaan opetuksen tuloksen toimivuutta. Opetus- ja testausaineistot eivät saa sisältää yhtään samoja näytteitä, koska silloin testaustuloksista ei voitaisi arvioida algoritmin toimivuutta yleisesti aineistolla, jolla sitä ei ole opetettu [18, s. 827].

Työssä käytettäväksi aineistoksi valittiin Amazon-verkkokaupan[4] arvostelut. Jo 1995 vuodesta asti toiminnassa ollut Amazon on liikevaihdoltaan ja markkina-arvoltaan

[1] `https://www.python.org/`
[2] `http://scikit-learn.org/`
[3] `https://www.nltk.org/`
[4] `https://www.amazon.com/`

suurin verkkokauppa [3]. Amazonin tuotearvostelujen 1 - 5 tähteen ja tekstikenttään perustuva arvostelusysteemi soveltuu hyvin aineistoksi tämän työn tavoitteiden pohjalta. Käytetty aineisto[5] sisältää kokonaisnumeroarvosanan väliltä 1 - 5, otsikon, arvostelun tekstin, arvostelijan tunnuksen, aikaleiman sekä metatietoa kyseisestä tuotteesta, mutta tässä työssä hyödynnettiin ainoastaan arvostelun tekstiä ja otsikkoa, joiden pohjalta kokonaisnumeroarvosanaa pyritään arvioimaan. Esimerkki käytetystä aineistosta taulukoksi muokattuna on kuvassa 3.1.

Koko helposti saatavilla oleva aineisto, 142,8 miljoonaa arvostelua vuosilta 1996 - 2014, oli liian laaja käsiteltäväksi työssä käytetyllä pöytätietokoneella, joten käytettävä aineisto rajattiin 100000 englanninkieliseen kirja-arvosteluun. Aineisto on lisäksi rajattu niin, että mukana on ainoastaan arvostelut kirjoille, joille on vähintään 5 arvostelua, ja joiden arvostelija on antanut yhteensä vähintään 5 muuta arvostelua (*5-core subset*). Tällä rajauksella kaikkien testattavien algoritmien opetus ja testaus oli suoritettavissa alle vuorokaudessa. [2]

Aineisto on koneoppimismielessä varsin haastava. Tuotearvostelujen arvosanat ovat erittäin subjektiivisia ja aineistossa arvostelijoita on lähes yhtä monta kuin arvostelujakin. Täysin sama arvosteluteksti voi yhden arvostelijan mielestä olla arvosanaltaan 3 ja toisen mielestä 4. Tästä johtuen täydellinen luokittelu pelkän tekstin perusteella on mahdotonta ja on pyrittävä parhaaseen mahdolliseen arvioon.

reviewerID	asin	reviewerName	helpful	reviewText
A1MW856C6F1KXO	B00M0RE7CS	tosha	[1, 1]	I am so in love with Shaw. This was an awesome book and cannot give it enough
A3PNKLCGVG1P3P	B00M0RE7CS	Trey	[1, 1]	It's a simple twist of a book . That ONE that makes a young lady blush... While he
A2G3DE7EYTVZG9	B00M0RE7CS	Whitney McGregor	[3, 3]	Whoa! Excuse me while I finish fanning myself, it's still a bit hot in here. 5 stars!!
A76D0PDO97IIJ	B00M13FNSS	Amazon Customer	[2, 2]	Dre is quite possibly the worst character I have ever read about. He is selfish an
A23VHARP9XPOXI	B00M13FNSS	CamilleJoy712	[2, 2]	This book was amazing. It was definitely worth the wait. Had me on the edge of
A2Y66HD4J5S7QZ	B00M13FNSS	Candi	[2, 2]	Yasss hunny! This is a great read. That Dre is a mess, and Cherika she just refuses
A17YHECC8H9NEY	B00M13FNSS	Margie	[0, 0]	I ENJOYED THIS BOOK FROM BEGINNING TO END NOW AS FAR AS LEX SHE A HOE
A20KO0BPMNREJL	B00M13FNSS	Nicki	[1, 1]	Great book! Cherika was a fool. She let that man get away with too much! Alexu
A1BQO66R6OLCCW	B00M13FNSS	Nikey	[0, 0]	When I say this was an excellent book please believe it was definitely a page tu
A2NRGE3CSFY2TQ	B00M13FNSS	Yo	[2, 2]	This book was everything. I just hope Alexus wise up and move on. And lawd I th

Kuva 3.1 *Esimerkki käytetystä aineistosta muokattuna taulukkoformaattiin*

3.2 Toteutus

Koneoppimisohjelman alku sisältää aineiston lukemista ja muuntamista helpommin käsiteltävään muotoon. Käytetty aineisto on yhtenä isona tekstitiedostona, jossa jokainen arvostelu on omalla rivillään JSON-formaatissa[6]. Arvosteluista erotetaan *reviewText*, *summary* ja *overall* -kentät, joiden dataa käytetään opetus- ja testausdatana. Käytettäväksi aineistoksi valitaan 20000 ensimmäistä havaintoa kustakin

[5] `http://jmcauley.ucsd.edu/data/amazon/`
[6] `https://www.json.org/`

numeroarvosanasta, että aineisto on tasainen numeroarvosanojen esiintymisien suhteen. Käsittelemätön aineisto on selvästi epätasapainossa ja 5 tähden arvosteluja on suhteessa huomattavasti enemmän kuin muita, mikä aiheuttaa usein huonon luokittelutuloksen muille luokille. Tähän on kehitetty menetelmiä, joilla voidaan parantaa tuloksia epätasapainoisella aineistolla. Tämän työn tapauksessa aineistoa on riittävästi, että käytettävät näytteet voidaan valita siten, että tasapaino säilyy.[4]

Arvosteluteksti (*reviewText*) ja otsikko (*summary*) yhdistetään yhdeksi merkkijonoksi, joka saneistetaan (*tokenization*) eli jaetaan yksittäisiin sanoihin. Tästä sanalistasta poistetaan erikoismerkit sekä hukkasanat (*stop words*), joilla on vähän merkitystä tekstin sisällön kannalta. Hukkasanalistana käytetään NLTK-kirjaston englanninkieliselle tekstille suunnattua listaa (*nltk.corpus.stopwords.words('english')*). Sanat lisäksi stemmataan eli karsitaan (engl. *stemming*) hyödyntäen NLTK-kirjaston SnowballStemmer-luokkaa. Poistamalla sanoista päätteet saadaan saman sanan eri taivutusmuodot palautettua samaan muotoon, jolloin ne tunnistetaan samoiksi sanoiksi.

Seuraavaksi ohjelmassa suoritetaan piirteidenirrotus. Työssä käytettävästä TF-IDF-piirteidenirrotusmenetelmästä löytyy valmis toteutus scikit-learn-kirjastosta. Edellisestä vaiheesta saadut sanalistat yhdistetään takaisin välilyönneillä erotettuna merkkijonoiksi ja syötetään *TfidfVectorizer*-luokalle, joka muuntaa merkkijonot TF-IDF-menetelmän mukaisesti piirrevektoreiksi. Käytettäväksi piirrevektorin kooksi valittiin kokeiluiden perusteella 1000. Sitä suurempi piirrevektori antaa vain hyvin pieniä parannuksia tulokseen ja hidastaa ja hankaloittaa testausta.

Piirteenirrotuksen jälkeen aineisto on valmiina syötettäväksi vertailtaville koneoppimisalgoritmeille. Tukivektorikoneesta vertaillaan lineaarista ja radiaalista RBF-kerneliä (*radial basis function*) käyttävät versiot, joista molemmista löytyy sekä luokitteleva että regressiomalliin perustuva toteutus. Lineaarista tukivektorikoneesta tässä työssä käytettiin libLINEAR-kirjastoon perustuvia toteutuksia LinearSVC (*sklearn.svm.LinearSVC*) ja LinearSVR (*sklearn.svm.LinearSVR*), jotka skaalautuvat huomattavasti paremmin aineiston ja piirrevektorin koon mukaan kuin libSVM-kirjastoa käyttävät SVC- (*sklearn.svm.SVC*) ja SVR-luokat (*sklearn.svm.SVR*) [1]. Opetusajat käytetyllä aineistolla libLINEAR-kirjastoon perustuvilla toteutuksilla ovat lähes tuhannesosan verrattuna libSVM-pohjaisiin RBF-kerneliä käyttäviin toteutuksiin. Random-Forest-luokittelussa käytettiin RandomForestClassifier-luokkaa (*sklearn.ensemble.RandomForestClassifier*) ja regressiossa RandomForestRegressor-luokkaa (*sklearn.ensemble.RandomForestRegressor*). Molempien tapauksessa käytetään 1000 estimaattoria. Suurempi estimaattorien määrä olisi voinnut hieman parantaa tuloksia, mutta opetusaika oli jo tällä määrällä regression tapauksessa suurin

vertailluista algoritmeista.

Opetuksen jälkeen mallien pohjalta luodaan *predict*-funktion avulla ennuste testidatan numeroarvosanoista, joista lasketaan keskimääräisen virheen itseisarvo ja keskimääräinen neliövirhe verrattuna testiaineistossa oleviin numeroarvosanoihin. Opetus- ja testausvaihe toistetaan kaikille koneoppimismalleille ristiinvalidoinnilla, jolla vähennetään koulutus- ja testidatan jaon vaikutusta tuloksiin. Jolloin pystytään paremmin vertailemaan koneoppimisalgoritmeja yleisesti. Ristiinvalidoinnissa käytetään 5-ositettua ristiinvalidointia (engl. *5-fold cross-validation*). Ristiinvalidointikerrojen tulosten keskiarvosta saadaan lopulliset tulokset.

Lineaarisella tukivektorikoneella saatujen hyvien tulosten pohjalta sen tapauksessa vielä tutkittiin regularisointitermin C vaikutusta tuloksiin. Työssä testattiin *C:n* arvoja 0,01, 0,1, 1, 10, 100 ja 1000.

3.3 Tulokset

Ohjelman tulokset eri algoritmeille ovat listattuna taulukossa 3.1. Parhaisiin tuloksiin päästiin lineaarisella tukivektorikoneella, jolla luokittelussa keskimääräinen virheen itseisarvo oli 0,81. Keskiarvoon perustuvalla luokittelijalla, joka antaa kaikille arvosteluille numeroarvosanan 3, on keskipoikkeama 1,20 ja keskimääräinen neliövirhe 1,97, joten tulos on selvästi sitä parempi, mutta ei kuitenkaan erityisen hyvä. Satunnaiseen metsään perustuvat toteutukset pääsivät hyvin lähelle lineaarisen tukivektorikoneen tuloksia, mutta RBF-kerneliä käyttävät tukivektorikoneet jäivät selvästi muita vaihtoehtoja huonommalle tasolle yltäen vain niukasti parempaan tulokseen kuin keskiarvoon perustuva luokittelija.

	MAE	**MSE**
Lineaarinen tukivektorikone, luokittelu	$0,80 \pm 0,052$	$1,59 \pm 0,124$
Lineaarinen tukivektorikone, regressio	$0,85 \pm 0,025$	$1,11 \pm 0,098$
RBF-tukivektorikone, luokittelu	$1,15 \pm 0,063$	$1,95 \pm 0,148$
RBF-tukivektorikone, regressio	$1,05 \pm 0,030$	$1,55 \pm 0,125$
Satunnainen metsä, luokittelu	$0,83 \pm 0.058$	$1,70 \pm 0,131$
Satunnainen metsä, regressio	$0,86 \pm 0.039$	$1,19 \pm 0,112$

Taulukko 3.1 *Ristiinvalidoinnin tulokset [keskiarvo, keskihajonta]*

Lineaariselle luokittelijalle testattiin useampaa eri regularisointitermin C arvoa, minkä tulokset ovat taulukossa 3.2. Oletuksena käytetty arvo 1,0 osoittautui olevan hyvin lähellä optimaalista arvoa, eivätkä kokeillut vaihtoehdot parantaneet tulosta merkittävästi.

C	MAE	MSE
0,01	$0,8106 \pm 0,0536$	$1,6366 \pm 0,129$
0,1	$0,8023 \pm 0,0517$	$1,5991 \pm 0,127$
1	$0,8024 \pm 0,0522$	$1,5974 \pm 0,124$
10	$0,8048 \pm 0,0534$	$[,5991 \pm 0,130$
100	$0,8161 \pm 0.0557$	$1,6713 \pm 0,133$
1000	$0,9703 \pm 0.0580$	$1,9314 \pm 0,129$

Taulukko 3.2 *Regularisointitermin vaikutus [keskiarvo, keskihajonta]*

RBF-tukivektorikoneen heikko tulos verrattuna lineaariseen versioon ei ollut odotettua ennen testausta. Opetusaineistolla testattaessa saadut keskimääräinen virheen itseisarvo ja neliövirhe ovat vain hieman parempia kuin ristiinvalidoinnilla saadut tulokset ja kaukana lineaarisen tukivektorikoneen tuloksista. Näin ollen heikko tulos ei johdu ainakaan ylioppimisesta. Kuten Shani ja Pant tutkimuksessaan osoittavat: TF-IDF-piirteidenirrotuksella ja tukivektorikoneella RBF-kernelillä on saatu hyviä tuloksia tekstin aiheen tunnistamiseen [16], kuitenkin aiheen tunnistaminen ja käyttäjätyytyväisyyden ennustaminen ovat hieman poikkeavia ongelmia. Syy RBF-kerneliä käyttävän tukivektorikoneen heikkoon tulokseen löytyy todennäköisesti TF-IDF-piirteenirrotuksen ja RBF-kernelin muodostamien elliptisten päätöspintojen yhdistelmästä. Lineaarisen tukivektorikoneen tapauksessa positiivisen sanan saama suuri TF-IDF-arvo aiheuttaa suoraan parempien numeroarvosanojen valitsemisen, mutta RBF-kernelillä samanlaista lineaarista riippuvuutta on hankalampi kuvata. RBF-tukivektorikone pyrkii selittämään riippuvuutta liian monimutkaisesti, mikä ei kuvaa aineistoa tehokkaasti.

Luokittelevat versiot koneoppimismenetelmistä olivat lineaarisen tukivektorikoneen ja satunnaisen metsän tapauksessa hieman parempia mitattaessa keskipoikkeamaa, mutta neliöllistä keskivirhettä mitattaessa regressioon perustuvat versiot olivat aina parempia. Regressioon perustuvat versiot palauttavat desimaalilukuja ja välttävät ääriarvojen (1 ja 5) palauttamista, jolloin pahojen virhearviointien tapauksessa neliöllinen virhe jää usein selvästi pienemmäksi kuin kokonaislukuja palauttavilla luokittelijoilla, mikä pienentää selvästi neliöllisen virheen keskiarvoa. Absoluuttisessa virhettä mitattaessa luokittelevien versioiden palauttamilla kokonaisluvuilla on etu, koska aineistosta saatavat oikeat numeroarvosanat ovat kokonaislukuja ja pienet virheet oikeissakin regression palauttamissa vastauksissa nostavat absoluuttista virhettä.

Saadut tulokset eroavat kokonaisuudessaan varsin paljon tavoitelluista, mutta toisaalta ongelma ja testimateriaali ovat varsin haastavia. Annetut arvosanat ovat erittäin subjektiivisia ja mielipidettä tuotteeseen voi ilmaista monella tapaa. Täydelli-

seen tulokseen pääseminen on lähes mahdotonta, mutta selkeästi täysin vääriä arvioita pitäisi pystyä paremmin välttämään. TF-IDF-piirteidenirrotus hukkaa paljolti sanoihin liittyvän kontekstin laskiessaan pelkästään sanojen esiintymistiheyksiä: negaatiot lauseissa ja muut paljon tekstin merkitystä muuttavat rakenteet häviävät piirrevektorin muodostuksessa. Tämän takia koneoppimisalgoritmien saatavilla ei ole kaikkea tietoa, joka olisi välttämätöntä käyttäjätyytyväisyyden tarkan arvioinnin kannalta. Tämä todennäköisesti aiheuttaa suuren osan pahoista virhearvioinneista.

4. JOHTOPÄÄTÖKSET

Tässä työssä esiteltiin ja tutkittiin perinteisten koneoppimismenetelmien toimivuutta käyttäjätyytyväisyyden ennustamiseen tuotearvostelun tekstin pohjalta. Luokitteluongelman ratkaisussa käytetyn koneoppimisalgoritmien toimintaa esiteltiin siihen liittyvien vaiheiden kautta: aineiston keruu ja käsittely, piirteidenirrotus, opetusvaihe ja testivaihe. Piirteidenirrotuksesta esiteltiin TF-IDF-menetelmä ja luokittelijoista sekä regressiomalleista tukivektorikone ja satunnainen metsä.

Tulosten perusteella voidaan päätellä, että parhaiten tässä työssä käytetyistä menetelmistä tuotearvostelun käyttäjätyytyväisyyden arviointiin soveltuu tukivektorikone lineaarisella kernelillä. Parhaiden tulosten tarkkuuden lisäksi se oli selvästi muita kokeiltuja menetelmiä nopeampi opettaa ja testata käytetyillä toteutuksilla. Valinta regressiomallin ja luokittelun välillä on haastavampaa eikä kumpikaan ole selvästi toista parempi kaikilta osin. Luokittelussa virheen keskimääräinen itseisarvo oli regressiomallia parempi, mutta selvästi pienempi keskimääräinen neliöllinen virhe regressiomallilla nostaa sen yleisessä tapauksessa paremmaksi vaihtoehdoksi. Pieni virhe joka arviossa, mikä nostaa regression virheen itseisarvoa, ei useimmissa käyttötapauksissa ole merkittävä, mutta regression pienempi määrä täysin vääriä arvioita jotka nostavat neliövirhettä tekevät siitä paremman vaihtoehdon.

Tutkimusta voisi jatkaa tarkastelemalla useampia piirteenirrotus- ja luokittelumenetelmiä. Työssä tulivat vastaan selkeästi TF-IDF:n rajoitteet, joten vaihtoehtoiset piirteidenirrotusmenetelmät voisivat tuoda parempia tuloksia. Neuroverkkopohjaisia menetelmiä voitaisiin myös vertailla perinteisiin koneoppimismenetelmiin. Aineistoa on saatavilla huomattavasti enemmän kuin mitä käytettiin tässä työssä. Laajentamalla aineistoa kirja-arvostelujen ulkopuolelle, toisi sanaston laajentumisen myötä lisää haastetta piirteidenirrotukseen ja luokitteluun.

LÄHTEET

[1] "1.4. support vector machines - scikit-learn 0.19.1 documentation," http://scikit-learn.org/stable/modules/svm.html, accessed: 2018-04-03.

[2] "Amazon product data," http://jmcauley.ucsd.edu/data/amazon/, accessed: 2018-04-03.

[3] "Amazon.com, inc. - yahoo finance," https://finance.yahoo.com/quote/AMZN?p=AMZN, accessed: 2018-04-03.

[4] U. Bhowan, M. Johnston, and M. Zhang, "Developing new fitness functions in genetic programming for classification with unbalanced data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 406–421, 2012.

[5] T. G. Dietterich, *Machine Learning.* Department of Computer Science, Oregon State University, Available: http://web.engr.oregonstate.edu/~tgd/publications/nature-ecs-machine-learning.pdf.

[6] L.-K. Gideon, "The great a.i. awakening," *The New York Times Magazine*, Dec. 2016, Available: https://www.nytimes.com/2016/12/14/magazine/the-great-ai-awakening.html.

[7] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*, 2nd ed. Springer, 2009. [Online]. Available: http://www-stat.stanford.edu/~tibs/ElemStatLearn/

[8] M. A. G. Jivani, *A Comparative Study of Stemming Algorithms.* Department of Computer Science & Engineering, The Maharaja Sayajirao University of Baroda, 2011.

[9] K. Kettunen, *Reductive and Generative Approaches to Morphological Variation of Keywords in Monolingual Information Retrieval.* University of Tampere, 2007.

[10] M. Kudlka, J. Pokorný, V. Snáel, and A. Abraham, *Advances in Intelligent Systems and Computing : Proceedings of the Third International Conference on Intelligent Human Computer Interaction (IHCI 2011), Prague, Czech Republic, August 2011*, 1st ed. Berlin, Heidelberg: Springer, 2014.

[11] K. M. Lee, S.-J. Park, and J.-H. Lee, *Soft Computing in Big Data Processing: Advances in intelligent systems and computing volume 271.* DE: Springer Verlag, 2014.

[12] T. Leuhu, *Sentiment analysis using machine learning.* Tampere University of Technology, 2015.

[13] L. Oneto, S. Ridella, and D. Anguita, "Tikhonov, ivanov and morozov regularization for support vector machine learning," *Machine Learning*, vol. 103, no. 1, pp. 103–136, Apr 2016. [Online]. Available: https://doi.org/10.1007/s10994-015-5540-x

[14] J. F. Puget, "The most popular language for machine learning is ..." Dec 2016, Available: https://www.ibm.com/developerworks/community/blogs/jfp/entry/What_Language_Is_Best_For_Machine_Learning_And_Data_Science.

[15] J. Ramos, "Using tf-idf to determine word relevance in document queries," 2003, Available: https://www.cs.rutgers.edu/ mlittman/courses/ml03/iCML03/papers/ramos.pdf.

[16] T. B. Shahi and A. K. Pant, "Nepali news classification using naïve bayes, support vector machines and neural networks." IEEE, 2018, pp. 1–5.

[17] I. Steinwart and A. Christmann, *Support Vector Machines*, 1st ed. New York, NY: Springer-Verlag, 2008;2014;.

[18] I. V. Tetko, D. J. Livingstone, and A. I. Luik, "Neural network studies. 1. comparison of overfitting and overtraining," *Journal of Chemical Information and Modeling*, vol. 35, no. 5, pp. 826–833, sep 1995. [Online]. Available: https://doi.org/10.1021/ci00027a006

[19] P. D. Turney, "Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews," *CoRR*, vol. cs.LG/0212032, 2002. [Online]. Available: http://arxiv.org/abs/cs.LG/0212032

[20] T. Tuar, K. Gantar, V. Koblar, B. enko, and B. Filipi, "A study of overfitting in optimization of a manufacturing quality control procedure," *Applied Soft Computing*, vol. 59, pp. 77–87, 2017.

[21] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Information Processing & Management*, vol. 50, no. 1, pp. 104–112, 2014.

[22] C. Zhang and Y. Ma, *Ensemble Machine Learning: Methods and Applications*, 1st ed. New York, NY: Springer-Verlag, 2012;2015;.