TARTU ÜLIKOOL
Arvutiteaduse instituut
Infotehnoloogia mitteinformaatikutele õppekava

Risto Hinno

E-kirjade klassifitseerimine masinõppe abil Maanteeameti näitel

Magistritöö

Juhendaja: Kairit Sirts

Tartu 2018

E-kirjade klassifitseerimine masinõppe abil Maanteeameti näitel

Lühikokkuvõte:

Lõputöö eesmärgiks on e-kirjadest teemade tuvastamise ja e-kirjade klassifitseerimise raamistiku loomine Maanteeameti näitel. Töö teoreetilises osas antakse ülevaade tekstikaevest, muuhulgas teemade modelleerimisest ja dokumentide klassifitseerimisest. Teemade modelleerimisel keskendutakse mudelile LDA ning optimaalse teemade arvu leidmisele. Dokumentide klassifitseerimise osas antakse ülevaade mudelitest Naïve Bayes, SVM ja fasttext. Lisaks tutvustatakse võimalusi, kuidas suurendada klassifitseerimismudelite täpsust kasutades andmete esinduse muutmist, ansambelmeetodeid ja kalibreerimist. Töö empiirilises osas valmistatakse andmed ette ja analüüsitakse kasutades eelmainitud mudeleid ja meetodeid. Maanteeameti e-kirjade optimaalne teemade arv varieerub kasutatud meetodite lõikes ning on subjektiivne. Siiski võimaldab koherentsus osaliselt automaatselt määrata, millises vahemikus võib optimaalne teemade arv olla. Oluline aspekt arusaadava teemade mudeli loomisel on andmete puhastamine. Teemade modelleerimist saab kasutada andmete hõlpsamaks märgendamiseks klassifitseerimismudelite jaoks. Pärast andmete märgendamist treenitakse klassifitseerimismudelid, võrdlemaks erinevate mudelite ja täpsust suurendavate meetodite mõju täpsusele. Kõige täpsem mudel loodi ansambelmeetodiga kuhjamine. Täpseim mudel, mis ei kasutanud ühtegi täpsust suurendavat meetodit, on lineaarne SVM. Samas on 20 täpseima mudeli täpsuste vahe 0,02 ühikut. Loodud raamistikku on võimalik kasutada mõne teise asutuse e-kirjade analüüsimiseks ning klassifitseerimiseks ja sellest tulenevalt automaatsemaks vastamiseks.

Võtmesõnad: Teemade modelleerimine, loomuliku keele töötlus, dokumentide klassifitseerimine

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

E-mail classification via machine learning in example of Estonian Road Authority

Abstract:

The aim of thesis is to create a framework for e-mail topic detection and e-mail classification using data from Estonian Road Authority. In theoretical part, an overview of text mining including topic modelling and document classification is given. In topic modelling, the focus is on model LDA and finding optimal number of topics. In document classification, models Naïve Bayes, SVM ja fasttext are introduced. Methods for improving classification model accuracy are described: changing data representation, ensemble methods and calibration. In empirical part, data is prepared and aforementioned models and methods are applied. Optimal number of topics varies between different methods and is subjective. Coherence enables semi-automatically detect optimal number of topics. It is important to have sufficiently cleaned data for topic modelling. Topic modelling could be used for annotating data for classification. After annotation several classification models were trained to assess their accuracy. The most accurate model was created using ensemble method stacking. The most accurate model without using any other method was linear SVM. First 20 most accurate models difference in accuracy was up to 0,02 units. The created framework could be used for analyzing and classifying e-mails in other institutions to automate the answering process.

Keywords: Topic modelling, natural language processing, document classification

CERCS: P170 (Computer science, numerical analysis, systems, control)

Sisukord

Mõistete selgitused

AdaBoost – üks võimendamise meetodeid klassifitseeriva mudeli täpsuse suurendamiseks (Mayr, Binder, Gefeller, & Schmid, 2014).

CSS – (*Cascading Style Sheets*) astmelised stiililehed, kaskaadlaadistik. HTML-dokumendi ilmet määravate laadilehtede hierarhiline süsteem.[1]

HTML – (*HyperText Markup Language*) hüperteksti märgistuskeel[2].

Hüperparameeter – masinõppe mudeli väliselt väärtustatud muutuja, mille väärtus määratakse enne mudeli treenimist (Witten, Frank, Hall, & Pal, 2005).

Javaskript – (*javascript*) skriptikeel rakenduste lisamiseks veebilehele[3].

Kernel – Ressursijaotust ja muid põhifunktsioone hõlmav opsüsteemi tuum või süda[4].

LDA – (*Latent Dirichlet Allocation)* juhendamata masinõppe mudel dokumentidest teemade leidmiseks (Blei, Ng, & Jordan, Latent Dirichlet Allocation, 2003).

Lemmatiseerimine – sõna algvormi viimine (Weiss, Indurkhya, & Zhang, 2015).

N-gramm – mitmik, n-elemendiline järjestus. Näiteks bigramm moodustub kahest järjestikku asuvast sõnast (Zhai & Massung, 2016).

Parameeter – masinõppe mudeli sisemine muutuja, mille väärtus hinnatakse mudeli treenimise käigus (Witten, Frank, Hall, & Pal, 2005).

Regulaaravaldis – string, mis kirjeldab või langeb kokku mingi stringide hulgaga vastavalt kindlatele süntaksireeglitele[5].

Ristvalideerimine – (*cross-validation*) mudeli hindamise meetod, mis võimaldab hinnata, kui palju mudeli tulemus sõltub sellest, millist valimi alamhulka kasutati mudeli treenimiseks ja millist mudeli hindamiseks (Gareth, Daniela, Trevor, & Robert, 2014).

SVM – (S*upport Vector Machine*) tugivektorkategoriseerija, mudel dokumentide klassifitseerimiseks (Joachims, 1998).

Sõnakogum – (*Bag of Words*) meetod dokumendi vaatlemiseks sõnade kogumina arvestamata sõnade paigutust lausetest, tekstis (Zhai & Massung, 2016).

Tf-idf – (*term frequency–inverse document frequency*) statistiline mõõdik hindamaks sõna tähtsust dokumentide korpuses (Zhai & Massung, 2016).

[1] http://www.keeleveeb.ee/dict/speciality/aks/
[2] http://www.keeleveeb.ee/dict/speciality/aks/
[3] http://www.keeleveeb.ee/dict/speciality/aks/
[4] http://www.keeleveeb.ee/dict/speciality/computer/dict.cgi?word=kernel&lang=en
[5] http://www.vallaste.ee/

1. Sissejuhatus

Klienditeeninduse parandamiseks on mitmeid võimalusi. Üheks võimaluseks on klientide e-kirjadele vastamise kiirendamine e-kirjade automaatse klassifitseerimise abil. Käesoleva töö eesmärgiks on luua raamistik e-kirjade klassifitseerimiseks teemade järgi. Raamistikku peab olema võimalik rakendada erinevates asutustes tekstidest teemade tuvastamiseks ja selle alusel klassifitseerimiseks. Raamistik kasutab masinõppe meetodeid. Teemade tuvastamisel suudab masinõppel põhinev mudel suhteliselt kiiresti leida tundmatus tekstis esinevaid teemasid. Klassifitseerimisel võimaldab masinõpe luua mudeli, mis suudab klassifitseerida uusi dokumente (tekste), mida ei ole kasutatud mudeli treenimisel (Aggarwal & Zhai, 2012).

Ajalooliselt on kasutatud tekstidest teemade leidmiseks ja tekstide klassifitseerimiseks märksõnu. Tegemist on suhteliselt lihtsa meetodiga, mis eeldab, et eksisteerib võimalikult väikese vaevaga leitav dokumente eristav märksõnade kogum. Masinõppe mudelid ei piirdu ainult konkreetsete sõnade olemasolu arvestamisega dokumendis, vaid võivad lisaks vaadata sõnade sagedusi, koosesinemisi, sõnaosi jne (Aggarwal & Zhai, 2012). Seega on masinõpe potentsiaalselt täpsem ja üldistatav laiemale hulgale dokumentidele ning asutustele. Autorile teadaolevalt pole varem Eesti riigiasutustes e-kirjadest teemade tuvastamiseks ja klassifitseerimiseks masinõpet kasutatud.

Käesolevas töös kasutatakse Maanteeameti e-kirju. Maanteeamet on Majandus- ja Kommunikatsiooniministeeriumi valitsemisalas tegutsev valitsusasutus, mis kujundab turvalist ja toimivat liikluskeskkonda (Maanteeamet, 2018). Maanteeameti tegevusvaldkonnad jagunevad ehitus, hoolde-, liiklus- ja teedevõrgu valdkonnaks (Maanteeamet, 2018). Maanteeameti klientide hulk on Eesti mõistes küllaltki suur. Näiteks on 2016. aasta seisuga kätte antud üle 630 000 juhiloa ning väljastatud üle 25 000 esmase juhiloa (Maanteeamet, 2018). Tulenevalt küllaltki suurest klientide arvust on asutusel tuhandeid kliendikontakte kuus. Teema on aktuaalne, kuna asutusele saadetakse ligikaudu 100 000 e-kirja aastas ning nende maht kasvab umbes 8% aastas (Lisa 1). Käesoleva töö raames analüüsitakse 27 000 Maanteeametile saadetud e-kirja.

Põhjuseid, miks on vaja e-kirjade teemade tuvastamise ja klassifitseerimise raamistikku, on kolm:

1. Esiteks on e-kirjadest käsitsi teemade tuvastamine ja klassifitseerimine rutiinne töö, mistõttu ei ole töötajad motiveeritud seda tegema pikka aega järjest (korraga suudab üks ametnik seda teha kuni pool päeva) (Lisa 1). Samuti on tegemist suhteliselt väikest lisandväärtust loova tööga.
2. Teiseks on käsitsi e-kirjadest teemade tuvastamine ja klassifitseerimine aeglane töö, mistõttu on ka e-kirjadele vastamine aeglasem. See omakorda muudab Maanteeameti teenuse kvaliteedi madalamaks. Käsitsi e-kirjade jaotamisel tekivad pausid (puhkeaeg, lõunapaus) ning töötaja suudab ajaühikus ära jaotada ainult väikese hulga e-kirju võrreldes masinõppe mudeliga. Näiteks suudab dokumentide klassifitseerimise mudel fasttext klassifitseerida 0,5 miljonit lauset ligikaudu 312 000 klassi vahel vähem kui ühe minutiga (Joulin, Grave, Bojanowski, & Mikolov, 2017). Käsitsi kuluks selleks ühel töötajal päevi või nädalaid.
3. Kolmandaks pole võimalik kiiresti suurendada käsitsi suunatavate e-kirjade mahtu. Juhul, kui ajutiselt suureneb sissetulevate e-kirjade maht, on ainus võimalus e-kirjade suunamise kiiruse säilitamiseks suurendada töötajate arvu. See tähendab ressursi ümbersuunamist muude tööde arvelt ning koolitamist. Alternatiiv on jätkata e-

kirjade suunamist sama töötajate arvuga, mille tulemusena muutub e-kirjadele vastamine aeglasemaks ning teeninduse tase halveneb.

Töö eesmärgi saavutamiseks seadis töö autor järgnevad uurimisülesanded koos alam-uurimisülesannetega:

1. Anda ülevaade tekstikaevest, muuhulgas tekstide representeerimise ja tunnuste dimensioonide vähendamise võimalustest.
2. Luua e-kirjade teemade mudel, mis suudab tuvastada e-kirjade hulgas olevad teemad.
   2.1. Anda ülevaade teemade modelleerimisest.
   2.2. Anda ülevaade LDA mudelist.
   2.3. Valmistada ette Maanteeameti e-kirjad teemade modelleerimiseks.
   2.4. Modelleerida Maanteeameti e-kirjades olevaid teemasid.
3. Luua tekstide klassifitseerimise mudel, mis automaatselt tuvastab üldinfo postkasti tulnud e-kirjade teema.
   3.1. Anda ülevaade tekstide klassifitseerimise mudelitest.
   3.2. Anda ülevaade klassifitseerimise mudeli täpsuse suurendamise meetoditest.
   3.3. Anda ülevaade klassifitseerimise mudeli täpsuse hindamisest.
   3.4. Valmistada ette Maanteeameti e-kirjad klassifitseerimiseks.
   3.5. Luua Maanteeameti e-kirjade klassifitseerimise mudelid, leidmaks neist täpseim.

Töö teises peatükis annab autor ülevaate tekstikaevest ja selle eripäradest võrreldes ülejäänud andmekaevega. Töö kolmandas peatükis annab autor ülevaate käesoleva töö kontekstis olulistest tekstikaeve meetoditest: teemade modelleerimisest ja tekstide klassifitseerimisest ning nendega seotud mudelitest. Teemade modelleerimisel keskendutakse mudelile LDA, kuna see on üks enamlevinumaid teemade modelleerimise mudeleid (Aggarwal & Zhai, 2012).

Samuti annab kolmandas peatükis töö autor ülevaate klassifitseerimise mudelitest Naïve Bayes, SVM ja fasttext (Joulin, Grave, Bojanowski, & Mikolov, 2017). Käesolevas töös kasutatakse erinevaid klassifitseerimise mudeleid, et võrrelda nende täpsust ning leida, kas mõni mudel annab süstemaatiliselt täpsemaid tulemusi. Mudelite valikul lähtus töö autor nii nende populaarsusest tekstide klassifitseerimisel kui ka lihtsusest rakendamisel. Esimesed kaks (Naïve Bayes ja SVM) on levinud mudelid dokumentide klassifitseerimisel ning andnud häid tulemusi (Tripathy, Agrawal, & Rath, 2015; Pratama & Sarno, 2015). Kõige uuemaks mudeliks võib pidada fasttexti, mis põhineb närvivõrkudel[6].

Lisaks toob töö autor kolmandas peatükis välja, kuidas saab klassifitseerimise mudeli täpsust hinnata ning milliseid meetodeid saab kasutada, et täpsust suurendada. Täpsuse suurendamiseks kasutatakse käesolevas töös kolme gruppi meetodeid. Esimene grupp tegeleb erinevatesse klassidesse kuuluvate andmete hulkade tasakaalustamisega. Kui mõnes klassis on teistest klassidest oluliselt rohkem või vähem dokumente, võib klassifitseerimismudeli täpsus selle tõttu halveneda. Selle probleemi leevendamiseks võib suurendada vähemuses oleva klassi andmete hulka või vähendada enamuses oleva klassi andmete hulka. Teine grupp meetodeid tegeleb mudeli pakutud klasside tõenäosuste korrigeerimisega. Näiteks võivad mõned mudelid määrata klasside tõenäosusi konkreetsetes vahemikes, mitte ühtlaselt üle kogu võimaliku tõenäosusvahemiku. Tõenäosuste korrigeerimine viisil, et need oleksid ühtlasemalt jaotatud võib aidata mudeli täpsust suurendada. Kolmas grupp on ansambelmeetodid, mis üksikute mudelite asemel tegeleb mitme mudeli alusel lõpliku mudeli loomisega.

[6] https://fasttext.cc/docs/en/faqs.html

Üksiku mudeli tulemus võib olla ebastabiilne ning sõltuda palju juhuslikkusest. Samuti võivad erinevad mudelid olla täpseimad eri klasside dokumentide eristamisel. Mudelite kombineerimine võib aidata neid probleeme leevendada ning luua mudel, mis on täpne ja stabiilne. Klassifitseerimismudelite täpsuse hindamise ülevaade seab aluse töö neljandas peatükis analüüsitavate mudelite võrdlemiseks.

Töö neljandas peatükis annab autor ülevaate Maanteeameti e-kirjadest ning kuidas toimus nende kogumine. Viiendas peatükis modelleerib autor maanteeameti e-kirjade teemasid ning klassifitseerib neid märgendatud klasside alusel. Märgendamise aluseks on e-kirjades tuvastatud teemad. Teemade modelleerimisel võrreldakse erinevaid võimalusi optimaalse teemade arvu tuvastamiseks ning antakse ülevaade e-kirjades leitud teemadest. E-kirjade klassifitseerimisel annab autor ülevaate loodud mudelite täpsusest. Lisaks analüüsitakse, kuidas täpsuse suurendamise meetodid mudelite täpsust mõjutasid ning millised mudelite ja meetodite komplektid andsid parimaid tulemusi.

2. Andmekaeve ja tekstikaeve

Tekstikaeve on andmekaeve alamharu. Andmekaeve tegeleb elektroonselt salvestatud andmete töötlemise ja analüüsiga, leidmaks olemasolevatest andmetest uut informatsiooni (Witten, Frank, Hall, & Pal, 2005). Enamasti kasutatakse andmekaeves andmeid, mida saab kujutada maatriksina, kus veergudes on tunnused ja ridades analüüsitavad vaatlused. Tekstikaeve tegeleb tekstiliste andmete analüüsiga (Witten, Frank, Hall, & Pal, 2005). Tekstikaeve muutub pidevalt populaarsemaks, kuna struktureeritud andmete hulgast üha kiiremini kasvab struktureerimata andmete (sh tekstide) hulk. Tekstikaeve tegeleb tekstikogumist ehk dokumentide korpusest (peidetud) mustrite leidmisega, loomaks informatsiooni, mida on vaja otsuste tegemiseks (Aggarwal & Zhai, 2012). Esmane eesmärk ei ole tekstides olevale infole ligipääsu hõlbustamine (näiteks päringute teel), vaid andmete analüüsimine, tuues välja trendid, mustrid ja erindid (Aggarwal & Zhai, 2012).

Üheks põhjuseks, miks tekstikaevet on kasulik vaadelda muust andmekaevest eraldi, võib pidada tekstikaeve aluseks olevate andmete eripärasid. Aggarwali jt (2012) järgi eristab dokumente teistest andmetest järgnevad aspektid:

- Dokumentide analüüs on seotud hõredate (*sparse*) ja mitmemõõteliste (*high dimensional*) andmete analüüsiga.
- Dokumente saab analüüsida väga erinevatel tasemetel (näiteks tähemärkide, sõnade, lausete, lõikude jne tasemel).
- Tekstikaeve puhul tuleb arvestada keeleliste eripäradega.

Keelelised eripärad omavad olulist rolli. Eesti keel on inglise keelest morfoloogiliselt rikkalikum, mistõttu võib lemmatiseerimine aidata oluliselt vähendada andmete dimensioone (Tamm, 2012; Liin, Muischnek, & Müürisep, 2012) Siiski tuleb arvesse võtta konkreetset ülesannet, mida lahendatakse. Näiteks, kas on oluline arvestada sõnade kõikvõimalikke variatsioone või grupeerida sama sõna erinevad vormid.

Tekstikaeve tegeleb mitmete ülesannete lahendamisega:

- **Informatsiooni otsimine**. Informatsiooni otsimise puhul peab olemas olema dokumentide hulk, kust infot otsitakse. Otsija annab ette vihje (mis võib olla sõna, fraas või mõni muu dokument), mille tulemusena otsitakse korpusest välja sarnased dokumendid (Weiss, Indurkhya, & Zhang, 2015).
- **Informatsiooni eraldamine.** Informatsiooni eraldamine võib olla nii iseseisev ülesanne kui ka suurema tekstikaeve ülesande alamülesanne. Selle tulemusena väheneb dokumendi/korpuse maht, mida analüüsida ning tekstist eraldatakse vajalik informatsioon (Weiss, Indurkhya, & Zhang, 2015).
- **Dokumentide klasterdamine.** Selle tulemusena jaotatakse dokumendid klastritesse (klassidesse) vastavalt nende sisule (Weiss, Indurkhya, & Zhang, 2015). Klasterdamise puhul on tegemist juhendamata masinõppe meetodiga, kus täpsed klastrid (klassid) ei ole ette määratud.
- **Teemade modelleerimine.** Sarnaneb dokumentide klasterdamisega, kuid erineb selle poolest, et konkreetse klastri asemel antakse dokumendile tõenäosuslik väärtus, et see kuulub konkreetsesse klastrisse (teemasse) (Aggarwal & Zhai, 2012).
- **Dokumentide klassifitseerimine.** Dokumentide klassifitseerimise aluseks on eelnevalt on määratud klassid ning püütakse luua klassifitseerija, mis võimalikult täpselt suudaks dokumente klasside vahel jaotada (Weiss, Indurkhya, & Zhang, 2015). Tegemist on juhendatud masinõppe meetodiga.

Üldjoontes võib tekstikaeves ja ülejäänud andmekaeves kasutatavaid põhimõtteid pidada sarnasteks. Oluline aspekt, millele mõlemas tähelepanu pööratakse on mudeli üleõppimise (*overfitting*) vältimine (Gareth, Daniela, Trevor, & Robert, 2014). Tegemist on olukorraga, kus mudel on väga täpne treenimiseks kasutatud andmete korral, kuid oluliselt ebatäpsem uutele andmetele rakendatuna (Gareth, Daniela, Trevor, & Robert, 2014). See tähendab, et mudel ei ole hästi üldistuv ning praktikas kasutamine on ebatäpne. Üleõppimise hindamiseks (ning samuti leidmaks sobivamaid mudelite hüperparameetrite komplekte) kasutatakse käesolevas töös ristvalideerimist, mis aitab võimalikult realistlikult hinnata mudeli täpsust uutele andmetele rakendamisel (Gareth, Daniela, Trevor, & Robert, 2014). Ristvalideerimise eeliseks mudeli ühekordse testandmete põhjal hindamise ees on mudeli täpsuse hindamise väiksem sõltuvus juhuslikkusest. Mudelit treenitakse ja hinnatakse mitu korda ning erinevaid treenimise ja hindamise andmehulki kasutades.

Lisaks üleõppimise vältimisele on oluline saavutada mudeli võimalikult suur täpsus. Kõrge täpsuse saavutamise eelduseks on kvaliteetsed sisendandmed, teiseks eelduseks mudeli optimaalsed hüperparameetrid. Tekstikaeve sisendandmeid käsitletakse alapeatükis 3.1 ning tunnuste valimist alapeatükis 3.2. Optimaalsete hüperparameetrite leidmiseks saab kasutada võrkotsingut (*grid search*) koos ristvalideerimisega (Bergstra & Bengio, 2012). Võrkotsingu puhul moodustatakse kõikidest etteantud võimalikest hüperparameetrite väärtustest komplektid ning iga komplekti kohta treenitakse mudel ning hinnatakse mudeli täpsust (Bergstra & Bengio, 2012). Praktikas on oluline leida tasakaal potentsiaalsete hüperparameetrite väärtuste arvu ja mudeli täpsuse vahel. Suurendades hüperparameetrite ja/või nende väärtuste arvu, võib võrkotsing arvutuslikult väga kulukaks minna ning sellisel juhul tasuks alternatiivina kaaluda juhuotsingut (Bergstra & Bengio, 2012). Juhuotsingu puhul moodustatakse samuti kõikidest hüperparameetrite väärtustest komplektide hulk. Eelmainitud hulgast valitakse juhuslikult välja määratud arv komplekte, mille põhjal treenitakse võrreldavad mudelid (Bergstra & Bengio, 2012).

Käesolevas töös kasutatakse põhjalikumalt teemade modelleerimist ja dokumentide klassifitseerimist. Esimene on vajalik loomaks ülevaadet suures dokumentide hulgas (mille füüsiline läbilugemine oleks liiga ajamahukas) esinevatest teemadest. Teine aitab luua dokumentide eristamiseks vajalikku klassifitseerimise mudelit. Mudelite loomisel kasutatakse nii ristvalideerimist kui ka võrkotsingut. Esimest eesmärgiga vähendada üleõppimise riski ning hinnata hüperparameetrite mõju mudeli täpsusele, teist eesmärgiga efektiivselt leida mudelitele optimaalseid hüperparameetreid.

3. Tehniline taust

3.1 Teksti representeerimise võimalused

Tekstikaeve on seotud hõredate (*sparse*) ning mitmemõõtmeliste maatriksitega (Weiss, Indurkhya, & Zhang, 2015). Ühes korpuses võib kokku olla kümneid ja sadu tuhandeid unikaalseid sõnu, kuid ühes konkreetses dokumendis on üldjuhul ainult mõnikümmend kuni mõnisada sõna. Enne tekstikaevega alustamist tuleb dokumendid viia numbrilisse formaati, et neid oleks võimalik arvutiga analüüsida (Weiss, Indurkhya, & Zhang, 2015). Käesolevas alapeatükis antakse ülevaade mõningatest aspektidest, mis on olulised tekstide representeerimisel andmekaeve jaoks.

3.1.1 Sõnahulgad (bag of words)

Korpuse analüüsimise aluseks võib võtta hõreda maatriksi suurusega n×d (kus n on dokumentide arv ja d unikaalsete sõnade/tunnuste arv) (Weiss, Indurkhya, & Zhang, 2015). Maatriks näitab, mitu korda iga sõna/tunnus esineb konkreetses dokumendis. Kuna ühes dokumendis on väike hulk sõnu/tunnuseid, on maatriks suures osas täidetud 0-ga. Sellest tuleneb hõreda maatriksi mõiste. Eelkirjeldatud dokumentide maatriksina kujutamise viisi aluseks on dokumentide sõnahulgad (*bag-of-words*), kus sõnade järjekord ei ole oluline (Aggarwal & Zhai, 2012).

Hõre mitmemõõtmeline maatriks seab piirangud, kuidas saab dokumente omavahel sisuliselt võrrelda ning dokumentidest ülevaadet luua. Visuaalselt ei ole selline maatriks inimesele kergesti hoomatav. Samuti võib andmetes esineda palju müra (näiteks kirjavigu). Üheks võimaluseks on kasutada meetodeid, mis aitavad maatriksi dimensioone vähendada, võttes samal ajal arvesse fakti, et tegemist on tekstiliste andmetega (Aggarwal & Zhai, 2012). Üheks dimensioonide vähendamise meetodiks on alapeatükis 3.3.1 kirjeldatav LDA mudel. Teisi dimensioonide vähendamise meetodeid tutvustatakse alapeatükis 3.2. Vähendades dimensioone liiga palju on oht, et oluline informatsioon läheb kaduma.

Tekstide representeerimise juures on oluline aspekt, millisel kujul tunnused mudelisse valida. Üldlevinud on unigrammide kasutamine, kus tekst jaotatakse üksikuteks sõnadeks/tunnusteks. Samas võib selle tõttu osa informatsioonist kaduma minna (Zhai & Massung, 2016). Näiteks laused:

- „Ma arvasin, et film on hea“
- „Ma arvasin, et film pole halb“

Ja laused:

- „Ma arvasin, et film pole hea“
- „Ma arvasin, et film on halb“

Kahe eelneva lausekogumi muutmine sõnade kogumiks vähendab dokumendis olevat informatsiooni. Kasutades bigramme (jagades laused järjestikusteks kahesõnalisteks tükkideks), jääb alles rohkem informatsiooni („on halb“, „pole halb“), kuidas kasvab treenimiseks kasutavate tunnuste hulk (Zhai & Massung, 2016). Siiski läheb ka selle käigus osa informatsiooni kaduma, mistõttu tuleb tunnuste valikul leida tasakaal treenimiseks kasutatava informatsiooni ja mudeli kiiruse vahel. Liiga suurte n-grammide lisamine võib muuta mudeli treenimise aeglasemaks ja mahu suuremaks (Zhai & Massung, 2016).

Lisaks võib dokumentide hõlpsamaks analüüsimiseks vähendada andmete variatiivsust (ja seeläbi eelmainitud n×d maatriksi dimensioone) sõnavormide normaliseerimine. Üheks võimaluseks on sõnade lemmatiseerimine (algvormi viimine) või sõnalõppude eemaldamine

(*stemming*) (Weiss, Indurkhya, & Zhang, 2015). See võib aga eemaldada andmekogumist olulise informatsiooni, mis halvendab dokumentide analüüsimist. Mõningatel juhtudel sisaldavad sõnalõpud olulist informatsiooni. Samas võib lemmatiseerimine/sõnalõppude eemaldamine parandada mudeli tulemust, kuna andmete variatiivsus on oluliselt väiksem.

Käesolevas töös kasutatakse teksti representeerimiseks sõnahulki ning andmete variatiivsuse vähendamiseks lemmatiseerimist.

3.1.2 Tähe n-grammid

Lisaks sõnavormide normaliseerimisele võib kasutada sõna tähemärkide n-gramme. Näiteks sõna „where“, saab kujutada tähemärkide trigrammidena järgnevalt (märgid < ja > kujutavad vastavalt sõna algust ja lõppu): <wh, whe, her, ere, re> (Bojanowski, Grave, Joulin, & Mikolov, 2017). Oluline on tähele panna, et näites olev trigramm „her“ on erinev inglisekeelsest sõnast „her“, kuna esimene on sõnaosa, teine aga esitatakse mudelis koos alguse ja lõpu märkidega: <her> (Bojanowski, Grave, Joulin, & Mikolov, 2017). Samas on mõlemal sõnal üks ühine trigramm „her“, kuna sõna „her“ trigrammid on <he, her, er>.

Sõnade tähemärkide n-grammidena esitamine võib aidata toime tulla morfoloogiliselt keeruliste keeltega, kuna sõnalõpud on ainult üks osa, mida vaadeldakse sarnaste sõnade leidmisel (Zhai & Massung, 2016). Lisaks on võimalik leida sarnaseid sõnu sõnadele, mis treeningandmetes puudusid, kuid millel on näiteks sarnane tüvi. Käesolevas töös kasutatakse tähe n-gramme mudelis fasttext.

3.1.3 Tf-idf

Alapeatükis 3.1.1 mainitud dokumendi – sõnade/tunnuste maatriksi rakendamise üheks oluliseks aspektiks on tunnustele kaalude andmine. Näiteks võib väheinformatiivsete sõnadele väiksema kaalu andmiseks kasutada tf-idfi (Zhai & Massung, 2016). Tf-idfi arvutatakse alljärgneva valemiga (Weiss, Indurkhya, & Zhang, 2015):

$$tfidf(w) = tf(w) * log\frac{N}{df(w)} \text{ (1)},$$

kus:

- *tfidf(w)* on sõna w *tf-idf*,
- *tf(w)* on sõna w esinemissagedus dokumendis,
- *N* on dokumentide arv,
- *df(w)* on dokumentide arv, mis sisaldavad sõna *w*.

Tf-idf aitab kaaluda sõnasid sageduse alusel. Esiteks võetakse aluseks sõna sagedus dokumendis ning dokumentide osakaal korpuses, mis sõna sisaldavad. Kui sõna on dokumendis sage, võib see viidata tema olulisele. Samas kui sõna esineb väga paljudes dokumentides võib tema olulisus olla väike (näiteks sidesõnad) (Weiss, Indurkhya, & Zhang, 2015). Juhul kui enamik dokumente sisaldavad konkreetset sõna, ei ole sõna väga oluline klasside eristamisel. Tf-idf aitab sõnu kaaluda vastavalt nende suhtelisele olulisele dokumentides (Zhai & Massung, 2016).

Käesolevas töös kasutatakse tf-idfi dokumentide klassifitseerimise mudelites tekstide representeerimisel.

3.2 Tunnuste valimise meetodid

Lisaks tekstide maatriksina representeerimisele on olulised tunnused, mida klassifitseerimismudelisse sisendina antakse. Mõned tunnused võivad olla teistest informatiivsemad. Tunnuste valimisel on kaks eesmärki (Berry & Kogan, 2010):

- Andmete dimensionaalsuse vähendamine, mis vähendab mudeli treenimisel kasutavate unikaalsete tunnuste hulka. See aitab vähendada mudeli treenimise aega.
- Väheinformatiivsete tunnuste eemaldamine. See aitab suurendada mudeli täpsust ning vähendada mudeli treenimise aega.

Esimese eesmärgi täitmise jaoks kasutatakse alapeatükis 3.1.1 kirjeldatud lemmatiseerimisest/sõnalõppude eemaldamist. Lisaks sellele on üheks lihtsamaks viisiks ebaoluliste tunnuste eraldamiseks tühisõnade eemaldamine. Näiteks sõnakogumipõhise lähenemise üheks miinuseks on, et suur hulk sõnu (tühisõnad) ei anna informatsiooni dokumendi konkreetsesse klassi kuulumise või teema kohta (Aggarwal & Zhai, 2012). Näiteks leidub sidesõnu peaaegu igas dokumendis. Tühisõnade leidmiseks võib kasutada dokumentide visuaalset vaatlust ning moodustada selle põhjal väheinformatiivsetest sõnade kogum, mis eemaldatakse dokumentidest.

Lisaks visuaalsele vaatlusele kasutatakse ka kvantitatiivseid meetodeid. Näiteks koos tf-idf-ga võib kasutada hii-ruut statistikut (Aggarwal & Zhai, 2012). Kui tf-idf vaatleb tunnuste levimust üle kõigi dokumentide, siis hii-ruut test aitab mõõta kategooriliste tunnuste eristusvõimet eri klasside vahel (Sammut & Webb, 2017). Selle tõttu on selle meetodi kasutamine mõttekas dokumentide klassifitseerimisel. Tunnused saab eristusvõime alusel järjestada ning teha neist valiku (Sammut & Webb, 2017). See aitab vähendada tunnuste hulka, mis on üldlevinud üle erinevate klasside ja seetõttu ei aita klasse eristada. Hii-ruut statistik arvutatakse valemiga (2) (Berry & Kogan, 2010).

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i} \quad (2),$$

kus:

- χ^2 on hii-ruut statistik,
- O_i on tunnuse i vaadeldav sagedus,
- E_i on tunnuse i eeldatav sagedus, olukorras, kus tunnus erista klasse.

Käesolevas töös kasutatakse tühisõnade eemaldamist nii teemade modelleerimisel kui ka dokumentide klassifitseerimisel. Lisaks kasutatakse dokumentide klassifitseerimisel hii-ruut statistikut.

3.3 Teemade modelleerimine

Teemade modelleerimise eesmärk on anda ülevaade dokumentidest, et lugejal oleks võimalik ilma kõiki dokumente läbi lugemata saada aru, millest kirjutatakse. Tegemist on juhendamata masinõppe valdkonda kuuluva ülesandega. Teemade modelleerimine on sisuliselt dokumentide klasterdamine teemade põhjal. Teemade modelleerimine põhineb eeldusel, et autor kasutab erinevatel teemadel kirjutamisel teatud määral erinevaid sõnu (Aggarwal & Zhai, 2012). See võimaldab statistilise analüüsi käigus välja selgitada sõnade esinemissagedusi erinevates teemades.

Teema all peetakse silmas sõnade tõenäosuslikku jaotumist teemade vahel. Iga teema puhul on erinevatel sõnadel erinev esinemise tõenäosus. Teemade modelleerimisel esitatakse teemad kõige tõenäolisemate sõnade hulgana (Blei, Topic Modeling and Digital Humanities, 2012). Näidisloetelu kahest teemast ja neid kirjeldavatest sõnadest on toodud tabelis 1.

Tabel 1. Näide teemade modelleerimise käigus leitud teemadest.

Teema 1	**Teema 2**
Muusika	Raamat
Rock	Novell
Jazz	Lood
Laulja	Armastus
Album	Perekond

Tabelis 1 on toodud teemasid kõige tõenäolisemalt iseloomustavad sõnad. Teemadele annab nimetuse modelleerija. Käesoleva näite puhul võib teema 1 nimetuseks olla "muusika" ja teema 2 nimetuseks "armastuslood". Teemadele nimetuse leidmine ja optimaalse arvu teemade leidmine on iteratiivne protsess ning nõuab osalist dokumentide sisu tundmist. Teemade modelleerimise juures on oluline, et leitud teemad oleks inimesele sisuliselt mõistetavad. Eeltoodud näite põhjal saab järeldada, et tegemist on kohati küllaltki subjektiivse protsessiga, kuna dokumentide eelnev tundmine võib mõjutada nii optimaalset arvu kui ka teemade nimetamist. Üks populaarsemaid teemade modelleerimise mudeleid on LDA (Blei, Probabilistic topic models, 2012).

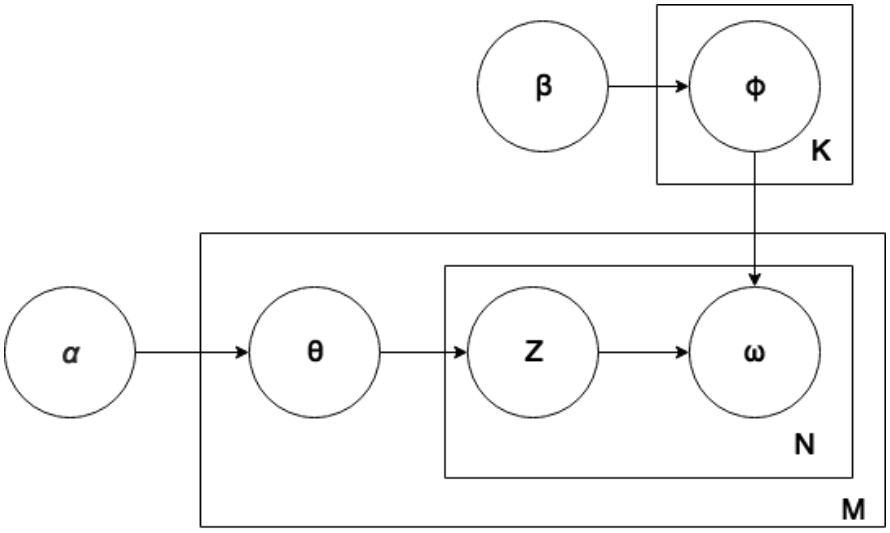
3.3.1 LDA

Teemade modelleerimise kasutatakse käesolevas töös mudelit LDA. LDA tugevuseks on, et dokumentide kohta antakse teemade tõenäosusjaotus (st üks dokument võib koosneda mitmest teemast) (Aggarwal & Zhai, 2012). Samuti saab mudelit LDA kasutada leidmaks teemasid dokumentides, mida treenimiseks ei kasutatud (Aggarwal & Zhai, 2012). See võimaldab eelnevalt loodud mudeli abil määrata teemade tõenäosuseid tundmatule tekstile.

LDA teeb dokumentide ja teemade kohta järgnevad eeldused (loetelu tugineb Blei jt (2003) artiklile):

- Teemade arv K on fikseeritud (see tuleb sisendina mudelile ette anda).
- Iga teemat kirjeldab erinev sõnakasutuse muster. Näiteks tabelis 1 kasutati teemas 1 rohkem teistsuguseid sõnu kui teemas 2. Sõnade järjekord dokumentides ei ole oluline.
- Iga dokument koosneb erinevatest teemadest, mis omakorda koosnevad erinevatest sõnadest. Näiteks võib konkreetse dokumendi puhul teema 1 tõenäosuseks olla 70%, teema 2 tõenäosuseks 30%.

Graafiliselt on LDA mudeli toimimise skeem kujutatud joonisel 2.



Joonis 2. LDA mudeli skeem (Blei, Probabilistic topic models, 2012).

Joonisel 2 tähendavad sümbolid järgmist (Blei, Probabilistic topic models, 2012):

- M – dokumentide arv,
- N – dokumendi M sõnade arv,
- K – teemade arv,
- α – dokumentide teemade Dirichlet' eeljaotus,
- β – teemade sõnade Dirichlet' eeljaotus,
- θ_m – teemade jaotus dokumendis m,
- φ_k – sõnade jaotus teemas k,
- z_{mn} – dokumendi m sõna n teema,
- ω_{mn} – konkreetne sõna.

Joonisel 2 kujutatud muutujatest on sõnad (ω_{mn}) ainsad, mis on otseselt vaadeldavad. Lisaks antakse mudelile ette hüperparameeter, mis määrab teemade arvu korpuses. Teised muutujad on varitunnused, mida mudel treenimise käigus genereerib (Blei, Ng, & Jordan, Latent Dirichlet Allocation, 2003). D. M. Blei põhjal (Blei, Probabilistic topic models, 2012) on tegemist tõenäosusliku generatiivse teemade modelleerimise mudeliga. See tähendab, et andmete kohta eeldatakse, et need on tekkinud generatiivse protsessi käigus, mis hõlmab ka varitunnuseid. Näiteks eeldatakse, et dokumendid on genereeritud erinevatest teemadest, mis omakorda on genereeritud erinevatest sõnadest. Otseselt sõnade jaotust teemade vahel mudel enne treenimist ei tea, need leitakse treenimise käigus. Nii nähtavaid kui ka varitunnuseid kasutatakse, leidmaks tõenäosuslikke jaotusi muutujate vahel. Näiteks konkreetse sõna teema z_{mn} sõltub teemade jaotusest dokumendis (θ_m).

Varitunnustele väärtuste leidmine pole triviaalne ülesanne, kuna need ei ole otseselt vaadeldavad. Vaadeldavad tunnused on dokumendid ja dokumentides esinevad sõnad. Varitunnuste leidmiseks võib kasutada mitmeid meetodeid, näiteks Laplace'i lähendamine (*Laplace approximation*), Monte Carlo meetodid ja variatiivne Bayes (*variational Bayes*) (Blei, Ng, & Jordan, Latent Dirichlet Allocation, 2003). LDA autorid pakuvad algses artiklis välja viimase meetodi kiiruse ja efektiivsuse tõttu (Blei, Ng, & Jordan, Latent Dirichlet Allocation, 2003). Käesolevas töös kasutatakse LDA mudeli parameetrite leidmiseks meetodit variatiivne Bayes. LDA mudelite loomiseks kasutatakse pythoni teeki gensim[7].

[7] https://radimrehurek.com/gensim/

3.3.2 Optimaalne teemade arv

Käesolev lõik tugineb Jonathan Chang jt artiklile (Chang, Boyd-Graber, Gerrish, Wang, & Blei, 2009). Pärast teemade mudeli treenimist on oluline hinnata mudeli täpsust. Teemade mudeli täpsuse hindamiseks saab kasutada kahte liiki lähenemist. Esimene on kvantitatiivne suund, kus hindamiseks kasutatakse mõnda kvantifitseeritavat muutujat (näiteks *perplexity*). Teiseks võimaluseks on lasta hinnata mudeli täpsust inimestel. Näiteks saavad inimesed hinnata, kas nende jaoks teemasid esindavad sõnad on omavahel kooskõlas ja arusaadavad. Lisaks saavad nad hinnata dokumentidele määratud teemade õigsust. Kvantifitseeritud meetodid kipuvad olema ebatäpsemad, kuna teema arusaadavuse ja eristuvuse hindamine ei ole triviaalne ülesanne.

Üheks kvantitatiivseks mõõdikuks, mis võib aidata hinnata teemade mudeli täpsust, on koherentsus (*coherence*). Erinevad uuringud (näiteks (Chang, Boyd-Graber, Gerrish, Wang, & Blei, 2009) ja (Röder, Both, & Hinneburg, 2015)) näitavad, et koherentsus korreleerub teistest näitajatest inimhinnanguga kõige paremini. Üheks põhjuseks võib olla, et koherentsus analüüsib teemat kirjeldavate sõnade omavahelist sobivust nagu teevad seda inimesed (Röder, Both, & Hinneburg, 2015). Omavahelise sobivuse arvutamiseks on olemas mitmeid viise. Üldjoontes võib need jaotada kaheks: meetodid, mis võtavad arvesse sõnade koosesinemise tõenäosust mõnes välises korpuses (näiteks vikipeedia) ning meetodid, mis võtavad arvesse sõnade koosesinemise tõenäosust treeningandmetes (Röder, Both, & Hinneburg, 2015).

Käesolevas töös kasutatakse koherentsuse mõõtmiseks kahte erinevat meetodit: u_{mass} ja c_v. Mõlemad kasutavad koherentsuse hindamiseks treeningandmeid ning hindavad, kui tõenäoliselt mudeli teemadesse leitud sõnad korpuse tekstides koos esinevad (Röder, Both, & Hinneburg, 2015). Iga teema sõna kohta moodustatakse paarid ülejäänud teema sõnadega ning iga paari kohta arvutatakse koherentsus, mille põhjal saadakse summeerimise või keskmise leidmise teel teema ning mudeli koherentsus (Röder, Both, & Hinneburg, 2015). Konkreetselt meetodi u_{mass} arvutamise loogika on toodud valemis (3) (Röder, Both, & Hinneburg, 2015).

$$koherentsus_{U_{mass}} = \frac{2}{N*(N-1)} \sum_{i=2}^{N} \sum_{j=1}^{i-1} log \frac{P(w_i,w_j)+\varepsilon}{P(w_j)} \quad (3),$$

kus:

- N on teemade populaarseimate sõnade arv, mida kasutatakse koherentsuse arvutamisel,
- w_i ja w_j on konkreetse teema sõnad, mille vahel koherentsust arvutatakse,
- $P(w_i,w_j)$ on dokumentide arv, kus sõnad w_i ja w_j esinevad koos,
- $P(w_j)$ on dokumentide arv, kus esineb sõna w_j,
- ε on silumise faktor, et logaritm tagastaks reaalarvu.

Meetodi u_{mass} puhul arvutatakse välja dokumentide osakaalu logaritm, kus mõlemad sõnad paarist esinevad koos jagatuna dokumentide arvuga, kus esineb sõna, mille suhtes koherentsust arvutatakse (Stevens, Kegelmeyer, Andrzejewski, & Buttler, 2012). Tegemist on meetodiga, mis arvestab sõnade otsese koosesinemisega.

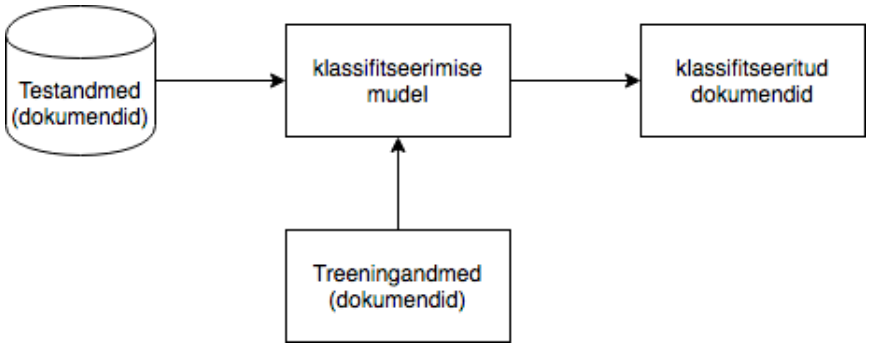
Meetodi c_v erinevus meetodist u_{mass} seisneb selles, arvesse võetakse ka sõnade kaudne koosesinemine (Röder, Both, & Hinneburg, 2015). Näiteks esinevad ühes dokumendis harva koos erinevate automarkide nimed (mistõttu need sõnad tunduvad olevat erinevad). Samas võivad dokumendid, kus räägitakse erinevatest automarkidest, tugevalt korreleeruda sõnaga „automark“. Selle tõttu saab erinevaid automarke kirjeldavaid sõnu vaadelda kui kaudselt

(läbi kolmanda sõna) koos esinevaid sõnu. Kaudse koosesinemise arvesse võtmiseks arvutatakse iga teema kohta kontekstivektorite koosiinussarnasused. Iga teema kontekstivektor koosneb teema iga sõna u_{mass} koherentsustest (otsene sarnasus) ülejäänud sõnade suhtes (Röder, Both, & Hinneburg, 2015). Mudeli koherentsuse saamiseks koherentsused summeeritakse. Katsed on näidanud, et c_v võib tugevamalt korreleeruda inimhinnangutega kui u_{mass} (Röder, Both, & Hinneburg, 2015). Juhul, kui LDA pakutud teemas olevad sõnad esinevad koos korpuses olevates tekstides, on teema koherentsus kõrge. Vastupidisel juhul on koherentsus madal ning viitab sellele, et teemasse on sattunud mõni sõna, mis muudab teema raskesti tõlgendatavaks.

Teemade arvu leidmiseks käesolevas töös kasutatakse eeltoodud koherentsuse meetodeid ning inimhinnangut. See võimaldab võrrelda, kui hästi need kokku langevad ning kas koherentsus korreleerub inimhinnanguga. Pärast dokumentide teemade leidmist ning märgendamist on võimalik hakata tegelema dokumentide klassifitseerimisega.

3.4 Dokumentide klassifitseerimine

S. M. Cheng Xiang Zhai jt (Zhai & Massung, 2016) järgi on dokumentide klassifitseerimise sisu järgmine. On olemas hulk treeningandmeid (dokumente), kus igale dokumendile on määratud sobiv klass. Treeningdokumentide eesmärk on aidata luua klassifitseerimise mudel, mis seob konkreetse dokumendi tunnused (*features*) konkreetse klassiga. Testandmete (andmed, mida pole kasutatud mudeli treenimiseks) puhul peab mudel võimalikult täpselt määrama dokumendi klassi. Üldjuhul eeldatakse, et dokumendi klass on kategooriline muutuja, kuid mõningatel juhtudel võib see olla pidev muutuja (Aggarwal & Zhai, 2012). Dokumentide klassifitseerimise protsess on kujutatud joonisel 3.



Joonis 3. Dokumentide klassifitseerimise protsess (Aggarwal & Zhai, 2012).

C. Z. Charu jt järgi (Aggarwal & Zhai, 2012) saab dokumentide klassifitseerimise mudeleid jaotada mitmeti. Näiteks võib mudeli väljundiks olla ainult konkreetne klass, mis dokumendile määratakse. Lisaks võib väljundiks olla tõenäosus, et dokument kuulub sellesse klassi. Mõned mudelid väljastavad iga dokumendi kohta kõikide klasside järjestuse tõenäosuse põhjal. Teised mudelid lubavad ühele dokumendile omistada mitut klassi. Lisaks võib mudeleid jaotada vastavalt sellele, kas mudeli põhjal on võimalik aru saada, miks konkreetne klass dokumendile omistati (mudeli sisemist otsustusloogikat on võimalik mõista) (Burrell, 2016).

Üldjuhul kasutatakse dokumentide klassifitseerimiseks juhendatud masinõppe mudeleid. Nende mudelite loomine ei ole täiesti automaatne, kuna eeldab, et mudel saab sisendina dokumendid koos infoga, millisesse klassi dokument kuulub (Zhai & Massung, 2016). See nõuab inimtööd dokumentide märgendamise näol. Masinõppe mudelite plussiks on nende võime ise „õppida“ ette antud tunnustest, kuidas võimalikult täpselt määrata dokumendi

klassi. Kui reeglipõhise mudeli puhul tuleb reegel kasutajal selgelt defineerida, siis masinõppe puhul „loob“ mudel ise reeglid treeningandmete põhjal (Aggarwal & Zhai, 2012).

Mudeleid dokumentide klassifitseerimiseks on mitmeid ning käesolevas töös kasutatavate mudelite kirjeldused on toodud järgnevates alapeatükkides.

3.4.1 Naïve Bayes

Üheks lihtsamaks klassifitseerimise mudeliks on Naïve Bayes. Tegemist on generatiivse mudeliga, mis spetsifitseerib, kuidas genereerida nii andmed kui ka klassid (nende ühistõenäosusjaotus) (Aggarwal & Zhai, 2012). Praktikas tähendab see, et osa tunnuseid klasside vahel kattuvad, kuid mõned tunnused peavad olema siiski klassi spetsiifilised, et klassifitseerimine osutuks võimalikuks. Bayes'i reegli põhjal saab tõenäosuse, et dokument kuulub etteantud klassi välja arvutada järgmise valemiga (4) (Weiss, Indurkhya, & Zhang, 2015).

$$P(klass|tunnused) = \frac{P(klass)*P(tunnused|klass)}{P(tunnused)} \quad (4),$$

kus:

- *P(klass|tunnused)* on tõenäosus, et tunnused kuuluvad konkreetsesse klassi.
- *P(klass)* on eeltõenäosus, mis näitab klassi tõenäosust enne, kui nähakse andmeid.
- *P(tunnused|klass)* on tõepära (*likelihood*), et konkreetsesse klassi kuuluvas dokumendis esinevad konkreetsed tunnused. Konkreetse tunnuse *i* puhul arvutatakse tõenäosus valemiga (5).

$$P(tunnus_i|klass) = \frac{klassi\ dokumentide\ arv,\ \ kus\ tunnus\ esineb}{dokumentide\ arv\ klassis} \quad (5)$$

- *P(tunnused)* on konkreetse tunnuse esinemise tõenäosus kogu tunnuste hulgast.

Mudeli (naiivseks) eelduseks on, et tunnused on üksteisest tingimuslikult sõltumatud, mistõttu saab valemi (4) asendada valemiga (6) (Weiss, Indurkhya, & Zhang, 2015). Tegemist on naiivse eeldusega, kuna sõnade (tunnuste) esinemise tõenäosus dokumendis võib sõltuda teistest dokumendis olevatest sõnadest.

$$P(klass|tunnused) = \frac{P(klass)*\prod_{i=1}^{n} P(tunnus_i|klass)}{P(tunnused)} \quad (6),$$

kus $tunnus_i$ on konkreetne tunnus *i*.

Valem (6) näitab, et tõenäosus et dokument on konkreetses klassis, sõltub klassi esinemissagedusest (treeningandmetes) ja iga tunnuse esinemise tõenäosusest selles klassis. Kuna klassifitseerimise eesmärgiks on leida kõige tõenäolisem klass, mitte konkreetse klassi tõenäosus, võib valemist jätta välja teguri *P(tunnused)* (Aggarwal & Zhai, 2012). Siiski võib mõningatel juhtudel täpse tõenäosuse arvutamine osutuda vajalikuks. Kõige tõenäolisema klassi saamiseks arvutatakse *P(klass|tunnused)* iga klassi kohta ning valitakse klass, mille tõenäosus on suurim.

Valemi (6) puuduseks on, et kui konkreetne tunnus (või tunnused) treeningandmetest puuduvad, on kõikide klasside tõenäosus 0 (Berry & Kogan, 2010). See on ebasoovitav, kuna ükskiku tunnuse (sõna) puudumine ei tähenda, et pole võimalik midagi väita dokumendi klassi kohta. Tulemaks toime treeningandmetes puuduvate tunnustega, võib kasutada Laplace'i silumist, mis tähendab, et treeningandmetes puuduva tunnuse $P(tunnus_i|klass)$ arvutamiseks kasutatakse valemit (7) (Zhai & Massung, 2016).

$$P(tunnus_i|klass) = \frac{n_i+\alpha}{N+\alpha d} \quad (7),$$

kus:

- n_i on dokumentide arv, kus tunnus esineb.
- α on silumise parameeter (tavaline väärtus on 1).
- *N* on klassi unikaalsete sõnade arv,
- *d* on unikaalsete sõnade arv.

Valem (7) tagab, et lisades silumise parameetri α, ei ole tundmatu tunnuse (sõna) puhul valemi (5) tulemus 0.

Naïve Bayes'i tüüpi mudeleid saab olenevalt mudeli treenimise aluseks olevatele andmete representeerimisele jagada kaheks (Aggarwal & Zhai, 2012):

- Bernoulli multivariatiivne mudel (*Bernoulli multivariate model*), kus sisendiks on binaarne maatriks, iga tunnuse kohta ei koguta tema esinemissagedust dokumendis vaid ainult fakti, kas ta on konkreetses dokumendis või mitte.
- Multivariatiivne mudel (*multivariate model*), kus sisendiks on maatriks tunnuste esinemissagedustega dokumentides.

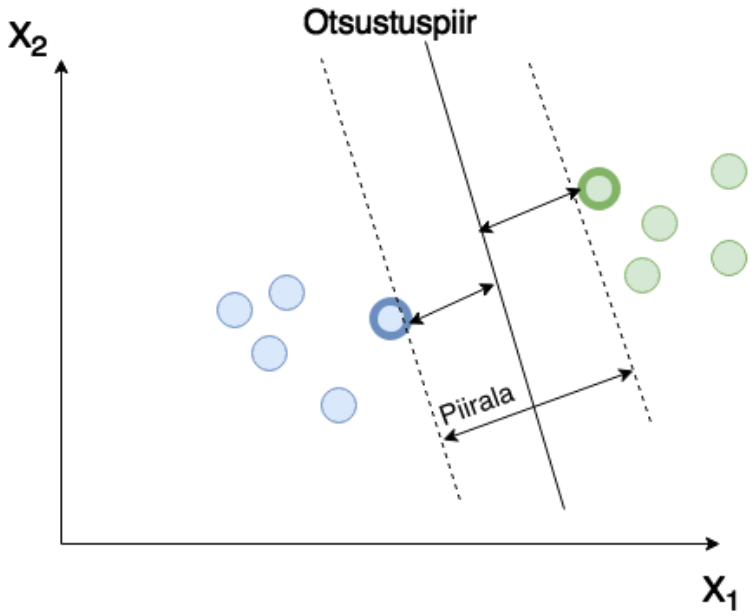
Charu, Aggarwal jt (Aggarwal & Zhai, 2012) põhjal on katsete tulemused näidanud, et Bernoulli multivariatiivne mudel võib olla täpsem olukordades, kus tunnuste arv on väike. Multivariatiivne mudel on täpsem rohkemate arvu tunnuste korral ning peaaegu alati juhtudel, kus sisendiks on valitud optimaalne arv tunnuseid. Siiski tuleb täpsust iga olukorra puhul eraldi testida. Üldiselt on Naïve Bayes täpsuse poolest võrreldav keerukamate mudelitega (Weiss, Indurkhya, & Zhang, 2015). Samas on selle mudeli täpsuse parandamiseks oluline eemaldada ebaolulised tunnused (näiteks tühisõnad), kuna need võivad oluliselt mudeli tulemust mõjutada (Weiss, Indurkhya, & Zhang, 2015).

Käesolevas töös kasutatakse nii Bernoulli multivariatiivset kui ka multivariatiivset mudelit. Mudelite loomiseks kasutati kahte erinevat klassi pythoni teegist skicit-learn (Pedregosa, et al., 2011): multivariatiivse mudeli jaoks klassi MultinomialNB[8], Bernoulli multivariatiivse mudeli jaoks klassi BernoulliNB[9]. Naïve Bayes on lihtsaim käesolevas töös kasutatav mudeli tüüp, mille eesmärgiks on testida, kas selle mudeli täpsus on võrreldav veidi keerukamate mudelitega.

3.4.2 SVM

Üheks populaarseks klassifitseerimise mudeliks on SVM (Zhai & Massung, 2016). SVM on töökindel mudel, mis on andnud häid tulemusi erinevat tüüpi ning eriti kõrge dimensionaalsusega andmete puhul (Joachims, 1998). Erinevalt mudelist Naïve Bayes on SVM eristav mudel (*discriminative*), mis üritab treenimise käigus selgeks õppida klasside vahelised piirid (Aggarwal & Zhai, 2012). Täpsemalt üritab SVM maksimeerida klasside vahel oleva otsustuspiiri kaugust klassi viimasest vaatlusest. See võimaldab jätta ruumi uutele vaatlustele ning neid õigesti klassifitseerida (Tong & Koller, 2001). Joonisel 4 on kujutatud klasside otsustuspiir kahedimensioonilises ruumis (praktikas on tekstilised andmed üldjuhul suurema dimensioonide arvuga).

[8] http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html
[9] http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.BernoulliNB.html



Joonis 4. SVMi klasside otsustuspiir.

Joonisel 4 tähistavad klasside vaatlusi rohelised ja sinised ringid. SVM otsib otsustuspiiri, mis oleks klasside lähimast vaatlusest võimalikult kaugel. Joonisel tähistavad seda nooled klassi lähima vaatluse ja otsustuspiiri vahel. Otsustuspiirile lähimaid vaatluseid nimetatakse tugivektoriteks (Gareth, Daniela, Trevor, & Robert, 2014). Otsustuspiiri ja tugivektorite vahele jäävat ala nimetatakse piiralaks (*margin*, joonisel eraldatud punktiirjoontega). Mida suurem see on, seda tõenäolisem, et mudel suudab klassifitseerida seni nägemata vaatluse õigesti, kuna on piisavalt „ruumi“ otsustuspiirini (Gareth, Daniela, Trevor, & Robert, 2014). Valemina saab otsustuspiiri kujutada järgnevalt (Hastie, Tibshirani, & Friedman, 2008):

$$xw + b = 0 \quad (8)$$

kus *x* on punkt otsustuspiiril ning *w* on otsustuspiiri normaal (risti otsustuspiiriga). Maksimaalse piirala leidmiseks eeldatakse, et andmed rahuldavad järgmiseid piiranguid (Hastie, Tibshirani, & Friedman, 2008):

$$x_i w + b \geq 1, kui\ y_i = 1 \quad (9)$$

$$x_i w + b \leq -1, kui\ y_i = -1 \quad (10)$$

Võrrand (9) näitab, et iga esimesse klassi ($y_i = 1$) kuuluv punkt on kas piirala ääres (tugivektor) või sellest tagapoolt. Võrrand (10) näitab, et teise klassi ($y_i = -1$) kuuluv punkt on kas piirala ääres (tugivektor) või sellest tagapool. Võrrandid kujutavad joonisel 4 kujutatud piirala punktiirjoonega piiravaid sirgeid. Võrrandid (9) ja (10) määravad, et ükski punkt ei tohi jääda piiralasse. Mudeli SVM eesmärgiks on leida maksimaalne piirala.

Reaalsete andmetega esineb harva olukordi, kus õnnestub andmeid täielikult eristada (joonisel 4 tähendaks see olukorda, kus mõni sinine punkt on roheliste hulgas või vastupidi). See tähendab, et ei ole võimalik leida lineaarset otsustuspiiri. Selle probleemiga toimetulekuks kasutab SVM kahte lahendust.

Esiteks on võimalik määrata laiem piirala otsustupiiri ümber (Gareth, Daniela, Trevor, & Robert, 2014). See tähendab, et piiralasse võivad jääda andmepunktid (ning mõned neist võivad olla valel pool otsustuspiiri), mille kohta otsust ei tehta. Tegemist on pehme piiralaga

klassifitseerijaga (*soft margin classifier*), mille vastandiks on klassifitseerija, kus ükski valesti klassifitseeritud andmepunkti pole lubatud (Weiss, Indurkhya, & Zhang, 2015). Piirala suurendamine võib mudeli täpsust testandmete puhul suurendada, kuna võib paremini aidata mudelil leida üldiseid mustreid klasside vahel. Mudel, mis on väga täpne treeningandmete peal, ei pruugi seda olla testandmete peal (Gareth, Daniela, Trevor, & Robert, 2014). Piirala laiuse muutmiseks kasutatakse muutujat C, mis määrab, kui palju valesti klassifitseeritud andmepunkte on lubatud (Berry & Kogan, 2010).

Teine viis, kuidas SVM aitab toime tulla mittelineaarsete otsustuspiiridega, on muutujate projitseerimine. Projitseerides muutujaid kõrgematesse dimensioonidesse (näiteks joonisel 4 olevast kahest dimensioonist kolme dimensiooni), suureneb tõenäosus, et klassid on lineaarselt eristatavad (Gareth, Daniela, Trevor, & Robert, 2014). Selleks kasutatakse kerneli trikki (*kernel trick*), mis suudab originaalsete tunnuste põhjal arvutada skalaarkorrutise (*inner product*) kõrgemas dimensioonis (Hastie, Tibshirani, & Friedman, 2008). Kerneli triki kasutamine on arvutuslikult oluliselt kiirem kui tunnuste otsene kõrgemasse dimensioonidesse projitseerimine (Aggarwal & Zhai, 2012). Seega, kuigi on tegu lineaarse klassifitseerijaga, suudab SVM eristada ka mittelineaarsete otsustuspiiridega klasse.

Käesolevas töös kasutatakse SVM mudelite loomiseks nelja erinevat mudelit pythoni teegist skicit-learn (Pedregosa, et al., 2011):

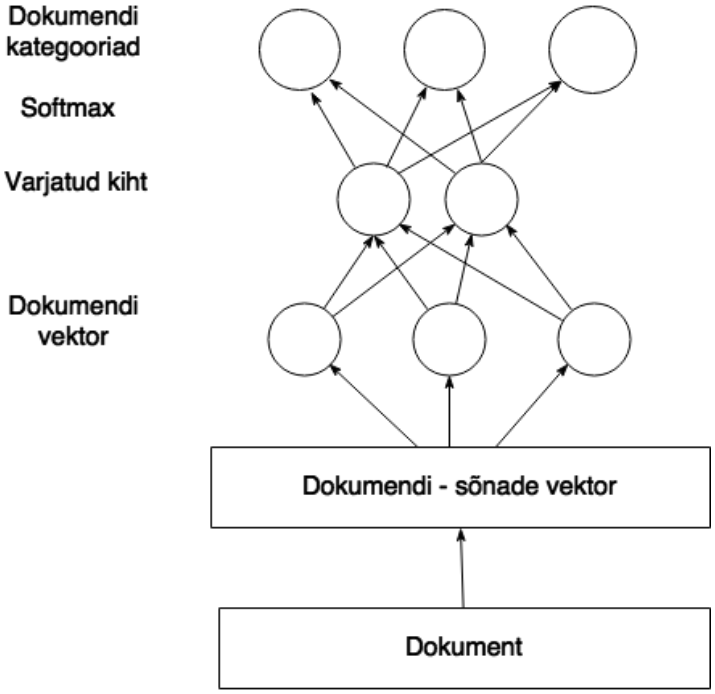
- LinearSVC – lineaarne SVM[10]. Eristab klasse kasutades lineaarseid otsustuspiire.
- SVC – SVM, mis erinevalt eelmisest mudelist võib olla nii lineaarne kui ka projitseerida andmeid kõrgematesse dimensioonidesse (leida mittelineaarseid otsustuspiire)[11]. Kasutab LinearSVC-ga võrreldes teistsugust alusmudelit, mistõttu ei pruugi anda sama tulemust isegi kui kasutatakse lineaarset kernelit.
- nuSVC – SVM, mis on toimeloogika poolest võrreldav mudeliga SVC[12]. Erinevalt mudelist SVC, kus hüperparameeter C võib varieeruda vahemikus 0 kuni lõpmatus, kasutab mudel ühe sisendina hüperparameetrit nu, mis saab omada väärtuseid vahemikus (0, 1]. Selle mudeliga loodud mudelite optimeerimine võib olla keerulisem kui mudeliga SVC loodud mudelite optimeerimine (Chang & Lin, 2002).
- SGD – kasutab lineaarseid mudeleid (sealhulgas SVM) ning treenib neid kasutades stohhastilist gradientlaskumist[13]. Stohhastiline gradientlaskumine võib leida optimaalse lahenduse kiiremini kui jõuaks selleni analüütiliselt (LeCun, Bottou, B., & K.-R., 2012). Stohhastiline gradientlaskumine toimib sarnaselt alapeatükis 3.4.3 kirjeldatavale närvivõrgu treenimisele, kus iga iteratsiooniga uuendatakse mudelit eesmärgiga vähendada mudeli viga[14]. Kasutatava mudeli tüüp on võimalik sisendina ette anda. SGD eelisteks on hea skaleeritavus suurtele andmehulkadele (mis on suured nii valimi kui ka tunnuste poolest).

Käesolevas töös Mudeli SVM kasutamise eesmärgiks on võrrelda selle täpsust lihtsama loogikaga mudeliga Naïve Bayes ning närvivõrgupõhise fasttextiga.

[10] http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html
[11] http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[12] http://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVC.html
[13] http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html
[14] http://scikit-learn.org/stable/modules/sgd.html

3.4.3 Fasttext

Üheks tekstide klassifitseerimise mudeli tüübiks on tehisnärvivõrgud. Käesolevas töös kasutatakse ühe mudelina närvivõrkudel põhinevat fasttexti. Närvivõrgud võivad olla erineva arhitektuuriga ning selle tõttu väga komplekssed, kuid mis ei pruugi alati anda suuremat täpsust dokumentide klassifitseerimisel (Zolotov & Kung, 2017). Fasttext on suhteliselt lihtne närvivõrkudel põhinev mudel, mille täpsus on võrreldav oluliselt keerulisemate mudelitega (Zolotov & Kung, 2017). Skemaatiliselt on kujutatud fasttexti arhitektuur joonisel 5.



Joonis 5. Fasttext klassifitseerija arhitektuur (Zolotov & Kung, 2017).

Käesolev lõik põhineb Vladimir Zolotovi ja David Kungi artiklil (Zolotov & Kung, 2017). Joonisel 5 on kujutatud ühe varjatud kihiga närvivõrk. Närvivõrgu sisendiks on klassifitseeritav dokument, millest moodustatakse dokumendi-tunnuste vektor. Tunnuste vektoritena saab kasutada eeltreenitud sõnavektoreid või treenitakse need klassifitseerija treenimise käigus treeningandmete põhjal. Sõnavektor kujutab iga sõna (tunnust) n-dimensioonilise vektorina, mille abil saab võrrelda sõna semantiliselt sarnasust teiste sõnadega (Mikolov, Chen, Corrado, & Dean, 2013). Üheks kuulsamaks näiteks sõnavektoritega semantiliselt sarnase sõna leidmiseks on: vektor(„King“) - vektor(„Man“) + vektor(„Woman“) = vektor(„Queen“) (Mikolov, Yih, & Zweig, 2013). Iga dokumendi kohta saadakse üks vektor dokumendi tunnuste vektorite keskmise leidmise teel. Fasttexti eripäraks on, et väikseima ühikuna ei kujutata mitte sõnu vaid alapeatükis 3.1.2 mainitud sõna tähemärkide n-gramme. Need lähevad sisendina närvivõrgu varjatud kihti, kus arvutatakse varjatud kihi kaalud. Seejärel liigub info edasi välimisse kihti (softmax-kiht), mis väljastab klasside tõenäosused, et dokument kuulub konkreetsesse klassi. Seejärel võrreldakse mudeli tulemust dokumendi tegeliku klassiga. Kui mudel eksis, uuendatakse varjatud kihi kaale (mudeli viga levitatakse tagasi eesmärgiga leida varjatud kihile kaalud, mis mudeli täpsust suurendavad). Varjatud kihi kaalude eesmärk on võimalikult täpselt „õppida“ treeningandmete pealt dokumente klassifitseerima.

Fasttexti eeliseks võrreldes teiste närvivõrgupõhise klassifitseerijatega on tema kiirus ja lihtsus. Läbiviidud eksperimentide järgi võib tehisnärvivõrkudel põhineva teksti klassifitseerimise mudeli treenimine aega võtta tunde, fasttext võib sarnase ülesande lahendamiseks mudeli treenida sekunditega ilma täpsuses oluliselt kaotamata (Joulin, Grave, Bojanowski, & Mikolov, 2017). N-grammide kiireks ja mälu säästvaks kasutamiseks rakendatakse räsimise trikki (*hashing trick*) (Joulin, Grave, Bojanowski, & Mikolov, 2017). See tähendab, et n-grammidest tehakse räsid, mida kasutatakse indeksina konkreetse n-grammi otsimiseks. Räsimise triki puhul ei ole vaja kasutada seosetabelit, mis seob kokku n-grammi ja tema indeksi maatriksis. Teiseks oluliseks aspektiks kiiruse saavutamise juures on hierarhilise softmax-kihi kasutamine (Joulin, Grave, Bojanowski, & Mikolov, 2017). Hierarhilise softmax-kihi kasutamine aitab vähendada softmax-kihis tehtavate arvutuste hulka, mistõttu muutub mudeli treenimine kiiremaks (Rong, 2018). Siiski tuleb arvestada, et hierarhilise softmax-kihi kasutamine annab märgatavat ajalist kokkuhoidu, kui klasse on palju (näiteks võrreldav unikaalsete sõnade arvuga korpuses) (Rong, 2018). Käesolevas töös on kasutatakse ainult 4 klassi, mistõttu hierarhilise softmax-kihi kasutamine ei anna erilisi eeliseid võrreldes tavalise softmax-kihiga.

Mudeli fasttext puhul on tegemist suhteliselt uue ja lihtsa arhitektuuriga mudeliga. Vaatamata oma lihtsusele on tegemist oluliselt kiirema mudeliga, kui seda on keerulisema arhitektuuriga närvivõrkudel põhinevad mudelid (Joulin, Grave, Bojanowski, & Mikolov, 2017). Samas on fasttexti klassifitseerija täpsus võrreldes keerulisemate mudelitega ainult mõnevõrra väiksem (Joulin, Grave, Bojanowski, & Mikolov, 2017). See muudab fasttexti kasutamise praktikas kiireks ja mugavaks. Käesolevas töös kasutatakse fasttexti mudelite loomiseks pythoni teeki fasttext[15].

3.5 Klassifitseerimismudelite täpsuse suurendamine

Dokumentide klassifitseerimise juures on oluline saavutada võimalikult kõrge täpsus. Tihti ei piisa selleks ainult heast mudelist. Nimelt võib mudeli täpsust mõjutada nii mudeli tüüp kui ka treenimise aluseks olevad andmed. Käesolevas töös kasutatakse kolme tüüpi meetodeid, mudeli täpsuse suurendamiseks:

- **Ansambelmeetodid**. Ühe mudeli asemel treenitakse mitu mudelit. Ansambelmeetoditega saadud tulemused võivad olla täpsemad ja väiksema variatiivsusega kui üksikud mudelid (Hastie, Tibshirani, & Friedman, 2008).
- **Andmete esinduse muutmine**. Andmete esinduse muutmine tähendab, et muudetakse erinevatesse klassidesse määratud andmete hulka. Eesmärgiks on saada võimalikult võrdne arv näidiseid igasse klassi (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). Praktikas esineb tihti olukordi, kus mõnesse klassi kuulub märkimisväärselt vähem vaatlusi, kui enamlevinud klassi. Mingi klassi ala- või üle esindatus võib klassifitseerija täpsuse muuta madalamaks. Probleem on seda suurem, kui mudeli tegelik eesmärk on võimalikult täpset klassifitseerida alaesindatud klassi vaatlusi.
- **Kalibreerimine**. Kalibreerimine ei tegele otseselt mudeli määratud klassidega, vaid klassi aluseks olevate tõenäosustega. Hästi kalibreeritud klassifitseerija puhul vastab tõenäosus andmete tegelikule jaotusele: näiteks vaatlused, millele on määratud klassi tõenäosus 10% või vähem, moodustavad andmetest umbes 10% (Zadrozny & Elkan, 2002). Kalibreerimata klassifitseerija on näiteks mudel, mis annab rohkem tõenäosusi teatud vahemikus (näiteks 10% või 90%). Kui see on nii, võib tõenäosuste kalibreerimine aidata mudeli täpsust suurendada.

[15] https://github.com/pyk/fastText.py

Meetodite valikul lähtus autor nende rakendatavuse lihtsusest ja aspektist, et oleks esindatud eri tüüpi meetodid (mis tegelevad mudelite tulemuste koondamisega, andmetega ja tõenäosustega). Milline meetod aitab täpsust maksimaalselt suurendada (ja kas üldse), sõltub nii andmetest kui ka mudelist. Kui andmetes jagunevad klassid ebaühtlaselt, on oluline proovida jagunemist tasakaalustada. Juhul, kui mudeli pakutud tõenäosused ei vasta andmete tegelikule jaotusele, võib kalibreerimine mudeli täpsust suurendada. Lisaks võivad ansambelmeetodid aidata mudeli variatiivsust ja viga vähendada. Ükski meetod pole universaalne, kuid võib aidata treenida täpsemat mudelit.

3.5.1 Andmete esinduse muutmine

Mudeli täpsuse üheks aluseks on kvaliteetsed algandmed. Üheks kvaliteedi aspektiks on erinevatesse klassidesse kuuluvate dokumentide hulk. Kui mõni klass on teistega võrreldes üle- või alaesindatud, võib mudeli täpsus sellest oluliselt langeda. Soovitatavaks lahenduseks sellises olukorras on andmete täiendav kogumine alaesindatud klassi (Witten, Frank, Hall, & Pal, 2005). Praktikas pole see tihti võimalik (näiteks ebamõistlikke kulude tõttu). Selle kompenseerimiseks kasutatakse käesolevas töös alljärgnevaid meetodeid:

- **Alaesindamine** (*undersampling*) – meetod, kus enamlevinud klassi andmeid vähendatakse, võttes neist juhuvalimi (Last, Douzas, & Baçao, 2017). Meetodi miinuseks on, et andmetest võidakse eemalda oluline osa infost ning klassifitseerija võib muutuda veelgi ebatäpsemaks.
- **Üleesindamine** (*oversampling*) – meetod, kus alaesindatud klassi andmeid suurendatakse, duplikeerides andmeid juhuvalikuga (Last, Douzas, & Baçao, 2017). Selle meetodi miinuseks on andmete duplikeerimine, mis tähendab, et andmete hulk küll suureneb, kuid andmete variatiivsus on sama. Samuti on oht, et treeningandmed satuvad testandmete hulka. Selle vältimiseks on kaks võimalust. Esiteks üleesindamine toimub iga ristvalideerimise iteratsiooni sees ainult treeningandmestiku peal. Teiseks üleesindamine tehakse algsete andmete peal, kuid ristvalideerimise andmehulgad moodustatakse nii, et iga näide saab konkreetses iteratsioonis korraga olla ainult kas treening- või testandmete hulgas. Käesolevas töös kasutatakse esimest meetodit.
- **SMOTE** – (*Synthetic Minority Oversampling Technique*) – meetod, mis genereerib andmeid alaesindatud klassi juurde, kasutades k-lähima naabri (*k-Nearest Neighbors*) leidmise mudelit (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). Alaesindatud klassi vaatluste (aluseks on dokumendi-tunnuste/sõnade maatriks) naabrusesse genereeritakse vaatlusi juurde (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). Meetodi eeliseks üleesindamise ees on üleõppimise ohu vähendamine, kuna samu andmeid ei korrata, vaid nende põhjal genereeritakse uued.

Ala- ja üleesindamist võib lugeda lihtsateks meetoditeks, mille sisuks on juhuvalimi leidmine. SMOTE on selle poolest veidi keerulisem meetod, mis võib aidata paremini vältida mudeli üleõppimist. Siiski tuleb ka SMOTE puhul arvestada, et ta muudab mudeli treenimise aluseks olevaid andmeid nii, et need ei pruugi tegelikkusele vastata. Andmete esinduse muutmiseks kasutatakse käesolevas töös pythoni teeki imbalanced-learn[16].

3.5.2 Ansambelmeetodid

Ansambelmeetodite eripäraks on mitme mudeli kasutamine lõpliku klassi määramisel. Ansambelmeetodite eeliseks on ühest mudelist tuleneva juhuslikkuse vähendamine, kuna see

[16] http://contrib.scikit-learn.org/imbalanced-learn/stable/index.html

„hajub“ erinevate mudelite vahel ära. Käesolevas töös kasutatakse kolme ansambelmeetodit, mille sisu järgnev:

- **Koondamine** (*bagging*) – treeningandmetest moodustatakse alamhulgad (üks dokument võib sattuda mitmesse alamhulka), millest treenitakse paralleelselt erinevad mudelid ning erinevate mudelite tulemustest valitakse „hääletamise“ teel konkreetse hindamise dokumendi klass (Gareth, Daniela, Trevor, & Robert, 2014). Koondamist kasutatakse, kuna see aitab vähendada klassifitseerija lõplikku hinnangu varieeruvust (Gareth, Daniela, Trevor, & Robert, 2014). Näiteks võib üks mudel olla suure või väikese täpsusega tulenevalt treenimisel kasutatud juhuslikkusest. Koondades kokku sama mudeli alusel treenitud mitu erinevat tulemust, on lõplik tulemus vähem mõjutatav ükskiku mudeli juhuslikkusest.
- **Võimendamine** (*boosting*) – treenitakse mitu mudelit treeningandmete põhjal, erinevalt koondamisest sõltub iga järgnev mudel eelmisest mudelist (Gareth, Daniela, Trevor, & Robert, 2014). Dokumendid, mille mudel valesti klassifitseerib, antakse uue mudeli jaoks suurema kaaluga (Gareth, Daniela, Trevor, & Robert, 2014). Võimendamine annab üldjuhul häid tulemusi, kui aluseks olev mudel on väikese täpsusega, täpse klassifitseerija puhul ei pruugi täpsuse kasv olla märkimisväärne (Gareth, Daniela, Trevor, & Robert, 2014). Seega „suunatakse“ klassifitseerijat paremini õppima vaatlusi, kus ta esialgu eksib. Võimendamine võib aidata mudeli variatiivsust ja viga vähendada. Käesolevas töös kasutatakse võimendamiseks *AdaBoost* algoritmi (Mayr, Binder, Gefeller, & Schmid, 2014).
- **Kuhjamine** (*stacking*) – samade treeningandmete põhjal treenitakse eri tüüpi mudelid (näiteks SVM ja fasttext) ning hinnatava dokumendi klass leitakse erinevate klassifitseerijate tulemuste „hääletamise“ või suurima keskmise klassi tõenäosuse leidmise teel (Gareth, Daniela, Trevor, & Robert, 2014). Hääletamiseks on võimalik kasutada erinevaid viise: enamushääletus (*Plurality Voting*, võidab kõige rohkem hääli saanud klass), kvalifitseeritud häälteenamus (*Majority Voting*, võidab klass, mis saab üle 50% häältest), kaalutud hääletamine (*Weighted Voting*, igale mudeli häälele antakse kaal), „pehme“ hääletamine (*Soft Voting*, võtab arvesse iga mudeli pakutud klassi tõenäosust) (Zhou, 2012). Kuhjamine suurendab klassifitseerija täpsust, kui eri mudelid on täpsed eri klassides (Witten, Frank, Hall, & Pal, 2005). Sellisel juhul aitab kuhjamine kokku koondada erinevate mudelite tugevad küljed.

Eeltoodud loetelu põhjal saab väita, et ansambelmeetodid on küllaltki erinevad. Koondamine tegeleb samatüübiliste mudelite kokku koondamisega, võimendamine eelmises etapis treenitud mudeli vigadest õppimisega ning kuhjamine erinevat tüüpi mudelite kokku kuhjamisega. Nende rakendamisel tuleb arvestada, et mudeli treenimise ajakulu võib suureneda. Praktikas kasutatakse ansambelmeetodeid suhteliselt palju (Weiss, Indurkhya, & Zhang, 2015). Käesolevas töös kasutatakse ansambelmeetodite rakendamiseks pythoni teeki skicit-learn (Pedregosa, et al., 2011).

3.5.3 Tõenäosuste kalibreerimine

Alati ei ole võimalik mudeli täpsust suurendada andmete esinduse muutmise või ansambelmeetoditega. Üheks võimaluseks sellisel juhul on mudeli tõenäosuste kalibreerimine. Kalibreerimine otseselt andmeid või mudelit ei muuda, vaid kohendab mudeli väljastatud tõenäosusi (Zadrozny & Elkan, 2002). Hästi kalibreeritud tõenäosus tähendab, et mudeli pakutud tõenäosus ja andmete tegelik jagunemine erineb vähe: näiteks dokumentidest, mille klassi tõenäosuseks pakkus klassifitseerija 50%, peaks ligikaudu 50% ka tegelikult kuuluma pakutud klassi (Cohen & Goldszmidt, 2018). Juhul, kui tegelik osakaal on oluliselt suurem või väiksem, võib kalibreerimine aidata korrigeerida mudeli tõenäosusi ning seeläbi mudeli

täpsust. Kalibreerimise eelduseks on, et lisaks konkreetse dokumendi klassile väljastab mudel ka klassi tõenäosuse. Käesolevas töös kasutatavad klassifitseerijate kalibreerimise meetodid on järgnevad:

- **Sigmoidkalibreerimine** (*sigmoid calibration*) – töötab kõige paremini klassifitseerijaga, mis määrab suurele osale andmetest kõrge tõenäosuse (Zadrozny & Elkan, 2002). Samuti aitab see transformeerida mudeli SVM algselt pakutud klassi skoori (mis ei ole tõenäosus, kuna väärtused võivad olla vahemikus [-∞; +∞]) tõenäosuseks (Niculescu-Mizil & Caruana, 2005). Sigmoidkalibreerimine võib aidata suurendada nende mudelite täpsust, mis määravad suure tõenäosuse andmetele, mis on otsustuspiirist (tõenäosusest 0,5) kaugel (Niculescu-Mizil & Caruana, 2005). Sigmoidkalibreerimise puhul võetakse aluseks klassifitseerija pakutud klassid koos tõenäosustega. Nendele rakendatakse Platt'i skaleerimist (*Platt scaling*, mis on sisuliselt logistiline regressioon), mis halvasti kalibreeritud tõenäosuste puhul aitab neid korrigeerida (Niculescu-Mizil & Caruana, 2005).
- **Isotooniline kalibreerimine** (*isotonic calibration*) – ei tee eeldusi, kuidas mudeli ennustatud tõenäosused jagunevad, kuid vajab kalibreerija treenimiseks rohkem andmeid kuid sigmoidkalibreerimine (Zadrozny & Elkan, 2002). Isotooniline kalibreerimine proovib leida isotoonilist (monotoonselt kasvavat või kahanevat) funktsiooni, mis minimeerib andmete tegeliku jaotuse ja tõenäosuse jaotuste vahet (Niculescu-Mizil & Caruana, 2005).

Mõlema meetodi puhul jaotatakse andmed üleõppimise vältimiseks enne kalibreerimist treening- ja testandmeteks (Niculescu-Mizil & Caruana, 2005). Kalibreerimist saab teha binaarsete klasside puhul. Rakendamaks kalibreerimist multiklassmudeli korral, võib esiteks mudeli väljundi muuta mitmeks binaarseks hulgaks, teha neile kalibreerimine ning kombineerida kalibreeritud tõenäosused (Niculescu-Mizil & Caruana, 2005).

Kokkuvõtvalt võib kalibreerimine aidata transformeerida mudeli pakutud skoore tõenäosusteks ning omakorda tõenäosusi „kohendada" nii, et need vastaksid õigete tulemuste tegelikule jaotusele. Kalibreerimine ei mõjuta otseselt klassifitseerija treenimist, vaid kalibreerimine toimub mudeli pakutud klassi tõenäosuste peal. Käesolevas töös kasutatakse kalibreerimiseks pythoni teeki skicit-learn (Pedregosa, et al., 2011).

3.6 Klassifitseerimismudelite hindamine

Klassifitseerimismudelite võrdlemiseks on vaja nende tulemusi ühtsetel alustel hinnata. Klassifitseerimismudeli tulemused võib jagada nelja hulka: õiged positiivsed (mudeli ennustus ühtib tegeliku klassiga), valepositiivsed (mudeli ennustus on positiivne, kuid tegelik klass on negatiivne), õiged negatiivsed (mudeli ennustus on negatiivne ja tegelik klass on negatiivne) ja valenegatiivsed (mudeli ennustus on negatiivne, kuid tegelik klass on positiivne) (Zhai & Massung, 2016). Vastavaid tulemusi saab kujutada eksimismaatriksina, mis on toodud tabelis 2.

Tabel 2. Eksimismaatriks (Hastie, Tibshirani, & Friedman, 2008).

		Tegelik klass	
		positiivne	**negatiivne**
Mudeli tulemus	**positiivne**	Õiged positiivsed	Valepositiivsed
	negatiivne	Valenegatiivsed	Õiged negatiivsed

Tabelis 2 on punasega kujutatud juhud, kus mudel eksib ning rohelisega juhud, kus mudeli tulemus on korrektne. Hindamise aluseks võetakse konkreetne klass, mille suhtes on tabelis 2 kujutatud hulgad välja arvutatakse (iga klassi kohta on võimalik luua oma eksimismaatriks). Konkreetseid mõõdikuid, mida eksimismaatriksi põhjal saab kasutada erinevate klassifitseerimismudelite võrdlemiseks, on (Hastie, Tibshirani, & Friedman, 2008):

- Täpsus (*precision*), näitab kui suurel määral on mudeli poolt positiivseks klassifitseeritud dokumentides tegelikult positiivseid dokumente, arvutatakse valemiga (11).

$$täpsus = \frac{õiged\ positiivsed}{õiged\ positiivsed + valepositiivsed} \quad (11)$$

- Saagis (*recall*), näitab, kui suure hulga positiivseid dokumente on klassifitseerimismudel tegelikult positiivseks klassifitseerinud, arvutatakse valemiga (12).

$$saagis = \frac{õiged\ positiivsed}{õiged\ positiivsed + valenegatiivsed} \quad (12)$$

- Õigsus (*accuracy*), näitab, kui suure hulga dokumente suudab mudel õigesti klassifitseerida, arvutatakse valemiga (13).

$$õigsus = \frac{õiged\ positiivsed + õiged\ negatiivsed}{õiged\ positiivsed + valepositiivsed + õiged\ negatiivsed + valenegatiivsed} \quad (13)$$

Eeltoodud mõõdikute puhul on ideaalne olukord, kus nad kõik on kõrge väärtusega (1 lähedal), kuid paraku on nad omavahel praktikas seotud negatiivselt (Zhai & Massung, 2016). Kõrge täpsusega mudelid võivad olla madalama saagisega ning vastupidi (Zhai & Massung, 2016). Õigsuse miinuseks on, et tema tulemus sõltub tugevalt klasside proportsioonist. Näiteks, kui tekstides on ühe klassi tekste ainult 5%, on võimalik luua naiivne 95% õigsusega mudel, kui kõik tekstid klassifitseerida enamlevinud klassi.

Üheks võimaluseks, kuidas hinnata mudeli headust, võttes arvesse klasside tasakaalustamatust, on F_1-skoor. F_1-skoor arvutatakse valemiga (14) (Zhai & Massung, 2016).

$$F_1 = \frac{2 \times täpsus \times saagis}{täpsus + saagis} \quad (14)$$

F_1-skoor ei arvutata lihtsalt täpsuse ja saagise aritmeetilise keskmisena, kuna aritmeetiline keskmine soosib suuri väärtusi (Zhai & Massung, 2016). Näiteks, kui mudelil on kõrge täpsus aga madal saagis, on keskmine endiselt suhteliselt kõrge. F_1-skoori mõjutab täpsuse ja saagise suur erinevus tulemust negatiivselt (Zhai & Massung, 2016). Seega lisaks sellele, et F_1-skoor võtab arvesse nii täpsuse kui ka saagise, arvestab ta ka nende väärtuste erinevusi. Juhul, kui klassifitseerimisel omab suuremat olulisust ainult täpsus või saagis, võib neid mudelite hindamisel vaadelda eraldiseisvatena.

Mitme klassi korral on F_1-skoor võimalik arvutada mitmel viisil, näiteks (Parambath, Usunier, & Grandvalet, 2014):

- Lugedes kokku üle kõikide klasside (iga klass hinnatuna ülejäänud klasside vastu) õiged positiivsed, õiged negatiivsed, valepositiivsed ja -negatiivsed ning arvutades selle põhjal F_1-skoori.
- Arvutades iga klassi kohta F_1-skoori (hinnatuna ülejäänud klasside vastu) ning leides kõikide klasside F_1-skooride aritmeetiline keskmise.

- Arvutades iga klassi kohta F_1-skoori (hinnatuna ülejäänud klasside vastu) ning leides kõikide klasside F_1-skooride kaalutud keskmise. Käesolevas töös kasutatakse seda meetodit, kuna koostöös Maanteeametiga leiti, et klasside olulisus sõltub nende suurusest.

Eeltoodud variantide valimisel on oluline hinnata, kas konkreetsel juhul on klassides olevad andmed väga erinevate näidiste hulgaga (on tasakaalustamata) ning kas igal klassil on võrdne kaal. Näiteks kui kõik klassid on võrdselt olulised (olenemata sellest, kui mõnes klassis on palju rohkem/vähem tekste), pole kaalutud keskmise kasutamine oluline. Vastupidisel juhul, kui on vaja arvesse võtta erinevat tekstide arvu, võib kaalutud keskmise kasutamine olla vajalik.

4. Andmete kogumine ja töötlemine

Andmete kogumise ja töötlemise eesmärgiks oli saada andmed ja ning valmistada need ette masinõppe mudelite jaoks. Oluline aspekt sealjuures oli tagada andmete anonüümsus, kuna e-kirjad võivad sisaldada tundlikke isikuandmeid. Teine oluline aspekt andmete töötlemise juures oli andmetest väheinformatiivsete osade eemaldamine ning seeläbi masinõppe mudelite täpsuse tõstmine.

4.1 E-kirjade anonümiseerimine

Analüüsitavad Maanteeameti e-kirjad olid pärit ajavahemikust 01.01.2017 - 31.10.2017. Vahemik hõlmab 2017. aastat kuni hetkeni, mil käesoleva töö autor tegi Maanteeametisse päringu andmete saamiseks. Kokku oli analüüsitava perioodi kohta 27 850 e-kirja. Tegemist oli osaga sissetulevatest e-kirjadest: aasta jooksul saadetakse Maanteeametisse ligikaudu 100 000 e-kirja (Maanteeamet, 2018). Täpsem ülevaade analüüsi kaasatud ja sellest välja jäänud e-kirjadest on alapeatükis 4.2. Andmete kasutamiseks analüüsimisel tuli need anonümiseerida, kuna e-kirjad võivad sisaldada (delikaatseid) isikuandmeid. Anonümiseerimisega alustas Maanteeamet, kuna töö autoril puudus õigus näha Maanteeameti klientide isikuandmeid. Andmed, mis võivad aidata otseselt tuvastada isikute identiteeti olid järgmised:

- Nimi (isiku- ja kohanimed).
- E-posti aadress.
- Link.
- Isikukood.
- Sõiduki registrinumber.

Lisaks sellele võib olla võimalik isikuid tuvastada ka kaudsete tunnuste alusel. Näiteks võib e-kiri sisaldada kirjeldusi unikaalsetest olukordadest või fakte, mis aitavad määrata kindlaks kirjutaja isiku. Hilisemal andmete ülevaatusel selgus, et selliseid juhtumeid esines andmetes harva. Üldjuhul ei kirjuta Maanteeameti kliendid e-kirjades enda (delikaatsetest) isikuandmetest. Seega otsustas Maanteeamet, et andmete anonümiseerimiseks piisab, kui e-kirjadest eemaldada eeltoodud loetelu.

Andmete anonümiseerimiseks arendas töö autor oma tööandja - Feelingstream OÜ - toote põhjal rakenduse, mis andmeid anonümiseerib (Feelingstream OÜ koduleht, 2018). Anonümiseerija kasutab arendusraamistikku Flask, mis põhineb pythoni programmeerimiskeelel[17]. Andmete anonümiseerimisega tegeles Maanteeameti spetsialist, kes pani rakenduse tööle virtuaalkeskkonnas. Spetsialist laadis andmed anonümiseerimiseks rakendusse .xlsx formaadis ning valis, mida andmetest tuleb eemaldada (käesoleva töö jaoks eemaldati nimed ja numbrid). Seejärel tagastas rakendus tulemusena .xlsx laiendiga faili andmetega, kus vajalikud andmed olid asendatud määratud lühenditega.

Rakenduses toimus anonümiseerimise protsess järgmiselt:

- Eemaldati uue rea ja tabulatsiooni märgid („\t“ ja „\n“), vältimaks probleeme töötamisel teegiga estnltk-ga.
- Kirjavahemärkide ette ja järgi lisati tühikud. See on vajalik, et kui nime küljes on mõni kirjavahemärk (näiteks jutumärgid), oleks võimalik sealt nime tuvastada.
- Tuvastati nimed, kasutades pythoni teeki estnltk[18]. Iga sõna e-kirjas analüüsiti ning kui tuvastati, et tegemist on nimega, siis see asendati sõnega „[name]“.

[17] http://flask.pocoo.org/
[18] https://github.com/estnltk/estnltk

- Asendati tekstis numbrid tähemärgiga „#“ kasutades regulaaravaldisi.
- Asendati tekstist e-posti aadressid ja lingid sõnega „[url or email]“, kasutades regulaaravaldisi.

Andmete kogumine ja anonümiseerimine toimus perioodil oktoober 2017 – jaanuar 2018. Anonümiseerimise käigus tuli rakendust täiendada, muutmaks isikuandmete eemaldamist tõhusamaks. Pärast automaatset anonümiseerimist jäi e-kirjadesse endiselt infot, mis võimaldab kasutajaid tuvastada (näiteks Skype kasutajanimed, mille struktuur on varieeruv). Selleks et andmeid oleks võimalik käesolevas töös kasutada, tuli autoril rakendusega anonümiseeritud e-kirjad käsitsi üle vaadata ning isikuandmed eemaldada. Selleks sõlmiti Maanteeametiga leping andmete konfidentsiaalsuse tagamiseks. Käsitsi anonümiseerimine toimus Maanteeameti ruumides ja arvutis. See oli optimaalne lahendus arvestades, et väikese hulga ebastandartsete andmete eemaldamiseks rakenduse arendamine ja testimine oleks võtnud kordades rohkem aega. Kokku tehti e-kirjades järgnev hulk asendusi:

- 266 167 nime (sh Skype’i kasutajanimed) asendust (neist 28 041 tehti käsitsi autori poolt),
- 106 752 lingi ja e-posti aadressi asendust,
- 26 981 numbri asendust.

Käsitsi tehtud asendused olid need, mida rakendus ei asendanud automaatselt ja mida seetõttu ei ole võimalik ka tulevikus automaatselt teha. Sellest tulenevalt avaldab see mõju nii teemade kui ka klassifitseerimise mudelite täpsusele. Arvestades, et käsitsi tehtud asendused moodustasid kõikidest nimeasendustest ligikaudu 10% (ning kõikidest tehtud asendustest 7%), peaks selle mõju mudelite täpsusele olema väike. Pärast anonümiseerimist alustas töö autor e-kirjade esmase analüüsimisega.

4.2 Ülevaade e-kirjadest

Analüüsitavad e-kirjad saatsid Maanteeametisse kliendid. Tegemist oli kirjadega, mida saadavad nii ettevõtted, eraisikud kui ka teised riigiasutused. Kokku oli analüüsiks kasutada 26 461 sisuga e-kirja (1 389 e-kirja sisu oli tühi ning nad jäeti edasisest analüüsist välja). Analüüsis ei kasutatud e-kirjade manuseid. Maanteeameti jaotuse järgi jagunesid analüüsitavad e-kirjad järgnevalt:

- **Kliendiinfopäringud**. Klientide kirjutatud e-kirjad aadressile info@mnt.ee või kodulehe vormi[19] kaudu saadetud pöördumine, mille teemaks liiklusregistrialane infopäring või küsimus e-teeninduse kasutamise kohta.
- **Teenindusbüroodele saadetud e-kirjad.** Klientide kirjutatud e-kirjad, mis on kirjutatud konkreetsele teenindusbüroole (või on Maanteeameti sisemiselt suunanud teenindusbüroole). Hõlmab endas ülevaatuse, eksamite/koolituste, infopäringute ja riigilõivuga seonduvaid kirju.

Lisaks sellele saadetakse Maanteeametile veel e-kirju, kuid mida ei kasutatud käesolevas analüüsis. Need kirjad jäid analüüsist välja Maanteeameti soovil ning lisaks põhjusel, et nende sisu on võrreldes eelmise loeteluga küllaltki täpselt määratud. Samuti oleks nende kirjade lisamine analüüsi oluliselt suurendanud andmete anonümiseerimise aega. Kuna klassifitseerimise mudel treenitakse e-kirja sisu põhjal, võib mõne välja jäänud e-kirja klasside analüüsimine osutuda keeruliseks. Analüüsis kasutati e-kirjade sisu, mis võib piirata mudeli kasutamist, kuna e-kirja sisu võib olla lühike ning oluline info asuda manuses. Selle tõttu

[19] https://www.mnt.ee/et/poordu-maanteeameti-poole

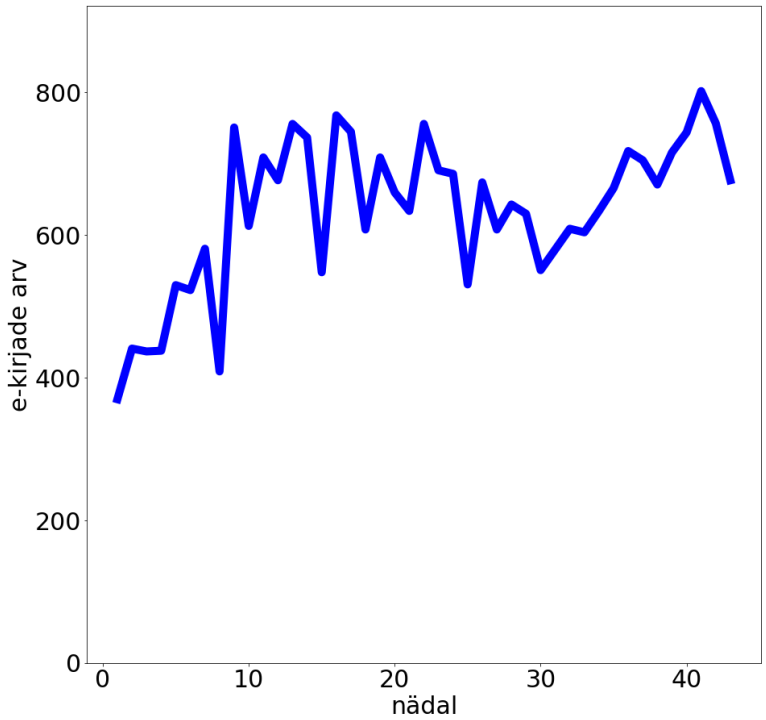
on klassifitseerimine rakendatav osadele Maanteeameti e-kirjadele. Analüüsist välja jäetud e-kirjade sisu võib jaotada järgnevalt (Maanteeamet, 2018):

- Kliendi tagasiside.
- Maanteeametile saadetavad dokumendid (näiteks sõidukite ostu-müügilepingud).
- Vaided.
- Teabenõuded.
- Autokoolide kirjad.

Eeltoodu põhjal oli e-kirjade sisu varieeruv. Osa väljajäänud e-kirja klasside puhul võis neid e-kirju leida ka analüüsitavate e-kirjade hulgast, kuna kliendid võivad kirjutada valele e-posti aadressile või valida vormilt vale teema. Analüüsitavate e-kirjade kohta oli olemas alljärgnev info:

- E-kirja identifikaator (igal e-kirjal unikaalne).
- Loomise aeg.
- Lahendamise aeg.
- Tüüp (tehniline andmeväli, mis muutujana ei aita tekste klassifitseerida).
- E-kirja sisu.

Kuna e-kirjad ei hõlmanud tervet aastat, võis selles tulenevalt ka väljatöötatud mudelite rakendamine reaalsuses anda kehvemaid tulemusi kui seda näitab mudeli täpsus testandmete peal. Näiteks võib detsembris saabunud e-kirjade sisu olla erinev võrreldes aasta alguses laekunud e-kirjade sisust. Alljärgneval joonisel on toodud e-kirjade arv nädalate lõikes (eemaldatud on esimene ja viimane nädal, kuna nende kohta olid andmed poolikud) võttes arvesse e-kirja loomise aega.



Joonis 6. E-kirjade arv nädalas e-kirja loomise aja põhjal.

Jooniselt 6 on näha, et e-kirjade arv oli suhteliselt varieeruv. Minimaalne arv e-kirju nädalas laekus analüüsitava perioodi alguses (alla 400 e-kirja nädalas), maksimaalne arv analüüsitava perioodi keskel ja lõpus (ligi 800 e-kirja nädalas). Väikese languse tegi e-kirjade arv läbi suvekuudel, mis viitab andmete sesoonsusele. Samuti on jooniselt näha, et üksteisele ajaliselt järgnevate nädalate e-kirjade arv võis erineda mitmesaja võrra.

Keskmine e-kiri sisaldas 657 tähemärki (arvesse ei ole võetud anonümiseerimisel asendatud nimesid, numbreid ja linke), sama näitaja mediaan on 346 tähemärki. Maksimaalne tähemärkide ühes kirjas oli 21 616. Keskmine kiri sisaldas 218 sõna (arvesse pole võetud asendatud nimesid, numbreid ja linke), sama näitaja mediaan oli 114 sõna. Maksimaalne sõnade arv e-kirjas oli 9 990. 689 e-kirjas puudus sisu. Nende puhul võis olla tegemist olukorraga, kus kirjaga saadeti ainult manus. Samuti ei saa välistada, et tegemist oli juhtumitega, kus kodulehel olev vorm saatis tühja e-kirja. Lisaks tuleb arvestada, et osa e-kirju sisaldasid eelnevaid kirju (kui on tekkinud pikem e-kirja vestlus), vormilt kaasa tulnud infot ning e-kirja jaluses olevat infot (ettevõtte info, reklaam, konfidentsiaalsushoiatust). Selle tõttu ei näita tähemärkide ja sõnade arv kui palju sisukat infot klassifitseerimiseks võis e-kirjas olla.

5. E-kirjade analüüs

Maanteemeti eestikeelsete e-kirjade analüüsimiseks kasutati peatükis 3 kirjeldatud meetodeid ja mudeleid ning peatükis 4 kirjeldatud andmeid. Analüüsi eesmärgiks oli saada suhteliselt kiiresti ülevaade e-kirjade teemadest ning seejärel luua mudelid, mis neid võimalikult täpselt klassifitseerib.

5.1 E-kirjade ettevalmistamine teemade modelleerimiseks

Andmete edasiseks analüüsimiseks ja modelleerimiseks tuli need puhastada. Esimeseks etapiks e-kirjade puhastamisel oli nende keele määramine. Edasine e-kirjade analüüsimine keskendus eestikeelsetele kirjadele, kuna nende maht moodustas põhilise osa e-kirjadest. Teiste keelte analüüsimine suurendaks mudelite loomiseks kuluvat aega ning ei olnud Maanteeameti jaoks prioriteet. Keele tuvastamiseks kasutati pythoni teeki *langdetect*[20]. Keele tuvastamise tulemused on järgnevad:

- 23 639 e-kirja eesti keeles,
- 1 441 e-kirja inglise keeles,
- 1 381 e-kirja vene keeles.

Keele tuvastamise ebatäpsust suurendas asjaolu, et e-kirjad võivad sisaldada erinevaid keeli (näiteks võib e-kirja jalus olla e-kirja sisust erinevas keeles). Võttes arvesse mõõdukat viga keele tuvastamisel, on eesti keel siiski kõige populaarsem keel. Järgnevad tegevused tehti eestikeelsete kirjadega.

Puhastamine erineb teemade modelleerimisel ja e-kirjade klassifitseerimisel. Teemade modelleerimiseks puhastati andmeid mitmes järgus. Esimeses järgus puhastati e-kirjade sisu järgnevalt:

- Eraldati e-kirjast vestluse osad, tuvastamaks kliendi algne kiri. Hindamaks e-kirjade arvu, milles on lisaks kliendi kirjale ka klienditeenindaja kiri, leiti kirja saatmisele viitavate mustrite arv igas e-kirjas. Selleks kasutati vestlusi eraldavad markerid (näiteks eraldab klienditeenindaja ja kliendi kirju üldjuhul marker „- -„). Mõlema meetodiga leiti, et hinnanguliselt 2 000 eestikeelset kirja (23 639-st) sisaldas lisaks kliendi tekstile ka klienditeenindaja vastust. Tegemist on suhteliselt väikese osaga. Edasisse teemade modelleerimisse jäeti teenindaja vastused sisse. Kui klient kirjutas e-kirja vastuseks Maanteeameti automaatsele teavitusele (näiteks teavitus juhilubade aegumisest), jäeti algne teavitus samuti e-kirja alles. Selle eemaldamine halvendaks teemade tuvastamist, kuna kliendi vastus sellistele teavitustele võib olla väga napisõnaline.
- Eemaldati tekstiosad, mis on pärit sisestusvormilt (sisestusvormile viitab tehniline info), e-kirja jalusest (näiteks konfidentsiaalsushoiatused) ja päisest. Nimekiri eemaldatavatest tekstiosadest saadi e-kirju üle vaadates ning märksõnadega näidiseid otsides.
- Eemaldati tähemärkide kombinatsioonid, mis viitavad CSS-le, HTML-le ja javaskriptile.
- Eemaldati kirjavahemärgid.
- Tähed muudeti väiketähtedeks.

[20] https://github.com/Mimino666/langdetect

- Tekst lemmatiseeriti. Lemmatiseerimiseks kasutati pythoni teeki estnltk[21]. Lemmatiseerimine aitas vähendada andmete variatiivsust ning muuta teemade leidmist kiiremaks ja ning teemasid rohkem arusaadavamaks.
- Eemaldati tühisõnad, mis on loetletud Raigo Kodasmaa bakalaureusetöös (Kodasmaa, 2011).

Seejärel treeniti esimesed LDA mudelid kasutades pythoni teeki gensim[22]. Treenitud mudelit kasutati tühisõnade ja muude väheinformatiivsete osade nimekirja täiendamiseks. Seejärel puhastati e-kirjad uuesti:

- Eemaldati allesjäänud e-kirjade jalused ning muud väheinformatiivsed elemendid ja sõnad (uue nimekirja koostamisel otsiti puhastatud e-kirjadest üles väheinformatiivsed osad, kasutades teemade mudelist välja paistvaid tühisõnu). Kuna tekst oli lemmatiseeritud, oli sõnastuse varieeruvus väiksem, mistõttu oli lihtsam ebavajalikku leida. Tegemist oli korduva protsessiga, kus pärast eemaldatavate osade nimekirja täiendamist treeniti uus LDA mudel ning prooviti teemade kirjeldustest leida uusi tekstiosi eemaldamiseks.
- Tekstist eemaldati tähemärgid, mis olid kirillitsas. Mõnede e-kirjade osa teksti oli vene keeles (näiteks e-kirja päis). Nende eemaldamata jätmisel tekkis LDA mudelisse eraldi teema e-kirjadest, mis sisaldasid kirillitsat, kuid mis ei andnud infot kirja sisu kohta.
- Eemaldati sõnad, mis olid lühemad kui kolm tähemärki (välja arvatud sõna „ei“). Nende sõnade informatiivsus on väike ning viitab allesjäänud tühisõnadele (näiteks eemaldamata HTMLile).

E-kirjade puhastamine oli üks aeganõudvamaid töid, kuna pärast igat iteratsiooni kerkisid esile uued tühisõnad, mida tuli omakorda eemaldada. Andmete puhastamise eesmärgiks ei olnud andmete täielik puhastamine, vaid puhastamine määral, mis aitab leida arusaadavaid ja eristuvaid teemasid. Teema oli selge, kui autor suutis mudeli pakutud sõnadest lugeda välja võimaliku teema ning pakutud sõnad ei sisaldanud olulisel määral tühisõnu.

5.2 E-kirjade teemade modelleerimine

Teemade modelleerimiseks kasutati alapeatükis 5.1 kirjeldatud puhastatud e-kirjade tekste. Esialgsete LDA mudelite eesmärk oli aidata kaasa tekstide puhastamisele tuues välja väheinformatiivsed sõnad, mis tuleks eemaldada. Teemade modelleerimise põhiline eesmärk on saada suhteliselt kiiresti ülevaade e-kirjades käsitletavatest teemadest. Osaliselt kasutatakse teemade modelleerimise tulemusi e-kirjade märgendamisel.

5.2.1 Teemade arvu leidmine

Teemade modelleerimisel oli oluline määrata optimaalne teemade arv korpuses. Optimaalse teemade arvu leidmiseks kasutati alapeatükis 3.3.2 kirjeldatud meetodeid. Kuna teemade arvu leidmine on subjektiivne ning kõige olulisem on inimese poolt tõlgendatavus, seadis töö autor optimaalse teemade arvu leidmiseks kaks kriteeriumi:

- Teemade arv pidi olema selline, et iga teema oli tõlgendatav/selgitatav. Ei tohtinud tekkida teemasid, mida autor ei suuda üldse tõlgendada või mis sisaldasid väheinformatiivseid sõnu. Halvasti tõlgendatavad teemad muudavad teemade modelleerimise sisutuks, kuna eesmärgiks on saada kiire ülevaade tekstis olevatest teemadest.

[21] https://github.com/estnltk/estnltk
[22] https://radimrehurek.com/gensim/

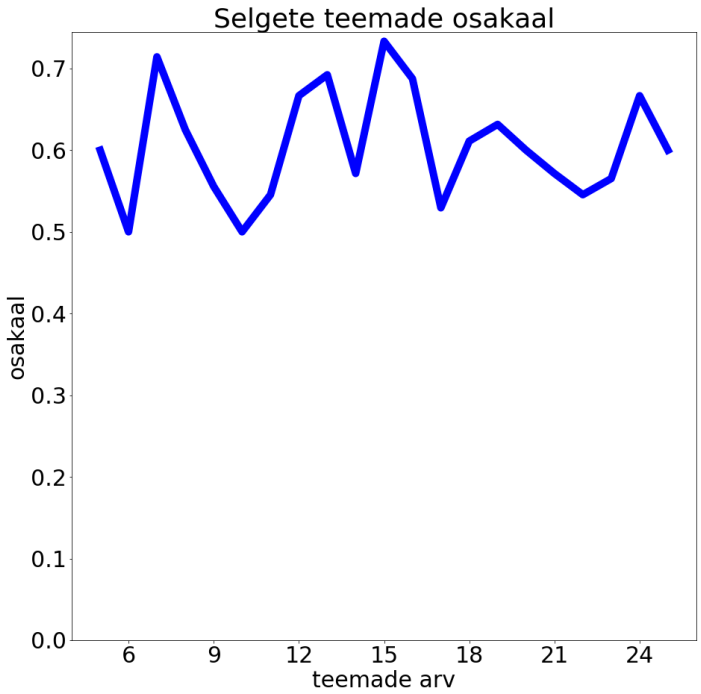
- Teemad ei tohtinud korduda. Ideaalis pidid kõik teemad olema üksteisest eristuvad. Praktikas oli seda raske saavutada, kuna ühes teemas võis olla alamteemasid või samasisulisi tekste, mis oli kirjutatud erinevas sõnastuses.

Leidmaks optimaalse teemade arvu, kasutas autor kahte meetodit:

- Esiteks visuaalne vaatlus, mis aitab hinnata, kui selgelt on iga teema tõlgendatav. Selleks kasutatakse pythoni teeki pyLDAvis[23] ning teemade märksõnade tõlgendamist.
- Teiseks koherentsuse kasutamine teemade arvu hindamiseks.

Visuaalsel vaatlusel teemade arvu leidmiseks alustati 5 teemast, seejärel suurendati teemade arvu ühe võrra kuni 25ni. Vahemik valiti vastavalt sellele, kui palju tekste võiks igasse klassi jääda. Võttes teemade arvuks 25, siis ühtlase jaotuse puhul oleks igas klassis ligi 23 000/25=920 teksti. Arvestades e-kirjade jaotumise ebasümmeetrilisust teemade vahel, tähendab see, et väiksematesse teemadesse kuulub oluliselt vähem e-kirju. Liiga väikese e-kirjade arvu puhul muutub teemast arusaamine keeruliseks (teema on liiga detailne, et olla kasutajale arusaadav) ja selle stabiilsus (näiteks erineva juhuslikkusega mudeli konkreetset teemat välja ei paku) võib olla madal.

Mudelite loomise järel hindas autor iga teemamudeli teemade eristuvust ja tõlgendatavust. Iga teema, mis ei olnud arusaadav või ei eristunud mõnest teisest teemast võeti arvesse kui ebaselge teema. Seejärel jagati iga mudeli selgete teemade arv kõikide teemade arvuga. Tulemused on kujutatud joonisel 7.



Joonis 7. Selgete teemade osakaal autori hinnangul.

Jooniselt on näha, et selgete teemade arv võis teemade arvu muutumisel muutuda suhteliselt olulisel määral (üle 10 protsendipunkti). Maksimaalne osakaal (üle 0,7) saavutatakse 15

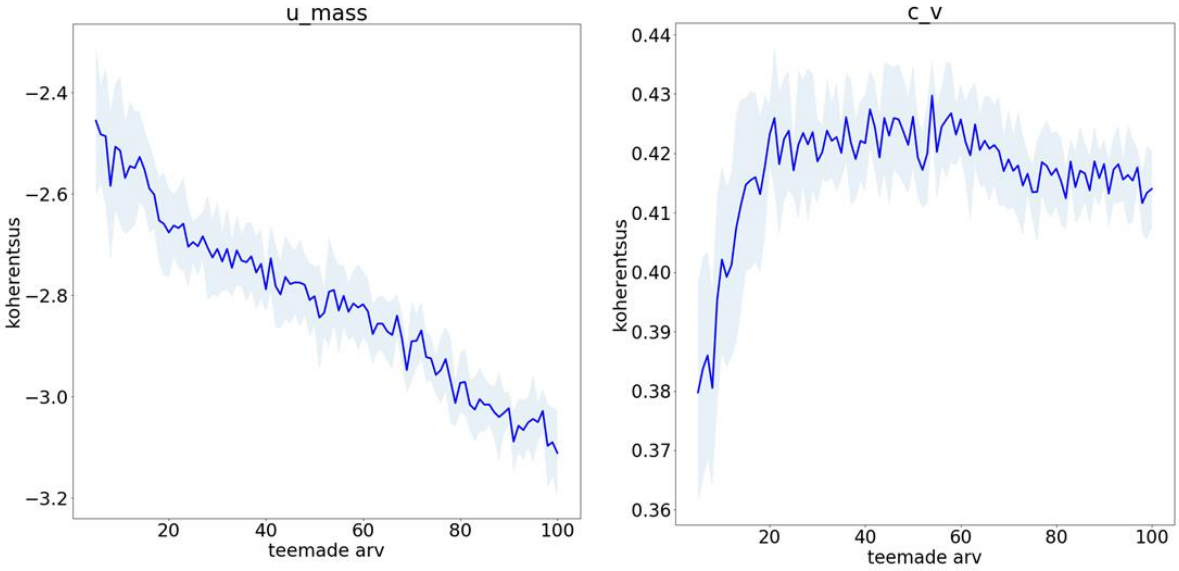
[23] https://github.com/bmabey/pyLDAvis

teema juures. Samas ei ole 7 teema juures ja 13 teema juures selgete teemade osakaal oluliselt madalam. Samuti tuleb arvestada, et joonisel 7 saadud tulemused sõltuvad suurel määral hindaja subjektiivsest arvamusest ning juhuslikkusest (erinevad juhuslikkused võivad anda erinevaid teemasid). Hindaja, kes on autorist rohkem või vähem kursis e-kirjade sisuga ja Maanteeameti teenustega, võib anda teistsuguseid hinnanguid.

Subjektiivsuse vähendamiseks kasutas autor optimaalse teemade arvu leidmiseks ka koherentsust. Selleks viidi läbi järgnev eksperiment:

- Iga teema arvu kohta vahemikus 5-100 genereeriti 10 LDA mudelit. Iga mudel 10-st mudelist treeniti identsete hüperparameetritega, välja arvatud etteantud juhuslikkus (*seed*). See võimaldab hinnata, kui palju muudab juhuslikkus mudeli tulemusi. Teemade arvu vahemik valiti eelduselt, et vahemikku jääb mõistlik teemade arv. 100 teema puhul, kui e-kirjad jaotuksid teemade vahel ühtlaselt, oleks iga teema kohta ligikaudu 23 000/100=230 e-kirja. Arvestades tõenäoliselt ebaühtlast jaotumist, jääb väiksematesse teemadesse väga vähe e-kirju.
- Iga mudeli kohta arvutati koherentsus kasutades alapeatükis 3.3.2 kirjeldatud c_v ja u_{mass} metoodikaid.

Eksperimendi tulemused on kujutatud joonisel 8.



Joonis 8. Eestikeelsete e-kirjade teemade mudelite keskmine koherentsus koos 95% usaldusintervalliga teemade arvu vahemikus 5-100.

Mida kõrgem on koherentsus, seda suurem on tõenäosus, et teemad inimese poolt tõlgendatavad (Röder, Both, & Hinneburg, 2015). Meetodite antavad koherentsuse väärtused olid erinevad, kuna nende arvutamise loogika on erinev. Meetodi c_v puhul pole ükski väärtus negatiivne, kuna arvutati välja koosinussarnasused otseste koherentsuste vahel, meetodi u_{mass} puhul arvutati välja dokumentide osakaalu logaritm (mis on üks võimalus otsese koherentsuse arvutamiseks), mis võis olla negatiivne (Stevens, Kegelmeyer, Andrzejewski, & Buttler, 2012). Jooniselt on selgelt näha erisus kahe metoodika vahel. Metoodi u_{mass} puhul vähenes koherentsus suhteliselt ühtlaselt. Teemade arvu 5-15 vahel oli kahanemine aeglasem. Metoodika c_v puhul kasvas mudelite koherentsus vahemikus 5-30 teemat, seejärel stabiliseerub ning teemade arvust üle 70 näitas langustrendi. Mõlema metoodika sarnasuseks

on, et tulemused sõltuvad juhuslikkusest. Väiksemate teemade arvu juures oli tulemuste varieeruvus suurem. Samuti ei andnud kumbki selget vastust, milline teemade arv võiks olla sobivaim. Samas andsid mõlemad metoodikad vihjeid, mis vahemikus võis sobivaim teemade arv asuda. Meetodi u_{mass} puhul võis sobivaim teemade arv asuda vahemikus 5-15, c_v puhul 20 teema juures.

5.2.2 Teemade jaotumine

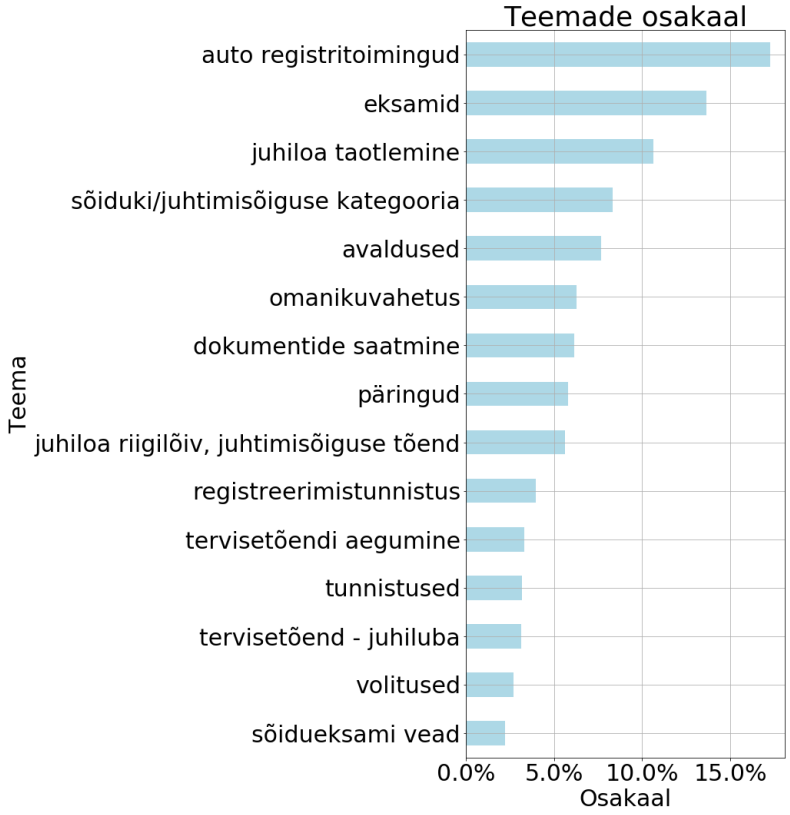
Lõplikuks teemade arvuks valis autor 15. Esiteks jääb see arvestades koherentsuse mõlemat metoodikat optimaalse teemade arvu lähedale. Teiseks oli see autori poolt hinnatud teemade vahemikus kõige kõrgema selgete teemade osakaaluga mudel. Teemad koos neid kirjeldavate sõnadega on toodud tabelis 3.

Tabel 3. Eestikeelsete e-kirjade teemad ja neid kirjeldavad sõnad.

Teemat kirjeldavad sõnad	**Teema nimi**	**Teema number**	**Klass, millesse aitas tekste märgendamisel leida**
Aeg, sõidueksam, eksam, kategooria, tulema	eksamid	1	eksam
Dokument, manus, edastama, kohtumaja, lisatud	dokumentide saatmine	2	muu
Tervisetõend, juhiluba, teenindusbüroo, perearst, tervisedeklaratsioon	tervisetõend - juhiluba	3	load_dokumendid
Manus, päring, osas, elektrooniline, kättesaamine	päringud	4	muu
Juhiluba, tõend, riigilõiv, esmane, teenindusbüroo	juhiloa riigilõiv, juhtimisõiguse tõend	5	load_dokumendid
Tunnistus, reg, liiklemine, jõustunud, teadmine	tunnistused	6	load_dokumendid
Avaldus, allkirjastatud, digitaalselt, edastama, kustutamine	avaldused	7	load_dokumendid /muu
Eksamineerija, viga, määrus, nõue, vaie	sõidueksami vead	8	eksam
Juhiluba, taotlus, luba, dokument, andmed	juhiloa taotlemine	9	load_dokumendid
Auto, omanik, register, kood, kasutaja	auto registritoimingud	10	load_dokumendid
Kategooria, võima, aasta, andma, probleem	konkreetse sõiduki/juhtimisõiguse kategooria	11	load_dokumendid /tehniline
Omanikuvahetus, kood, kaart, omanik, toiming	omanikuvahetus	12	load_dokumendid
Registreerimistunnistus, tulema, teenindusbüroo, büroo, dokument	registreerimistunnistus	13	load_dokumendid /muu
Tervisetõend, mootorsõidukijuht, otsus, arst, juhtimisõigus	tervisetõendi aegumine	14	load_dokumendid
Volitus, ignoreerima, tööpäev, volikiri, ettevõte	volitused	15	load_dokumendid

Tabelist on näha, et teemad on seotud erinevate Maanteeameti teenustega. Esindatud on nii eksamid (sõidu-, teooriaeksamid) kui ka registritoimingud (sõiduki omanikuvahetus, juhiloa vahetamine). Selgelt eristub teema, mis on seotud dokumentide, päringute ja avalduste saatmisega. Osad teemad olid omavahel seotud. Näiteks teema 3 ja 14, mis on seotud tervisetõendiga. Esimene neist on seotud juhiloaga (juhiloa taotlemisel on vaja tervisetõendit), teine rohkem tervisetõendi aegumisega (tervisetõendi aegumisel tuleb vormistada uus, et juhtimisõigus säiliks). Siiski võib neid sisuliselt pidada väga sarnasteks teemadeks.

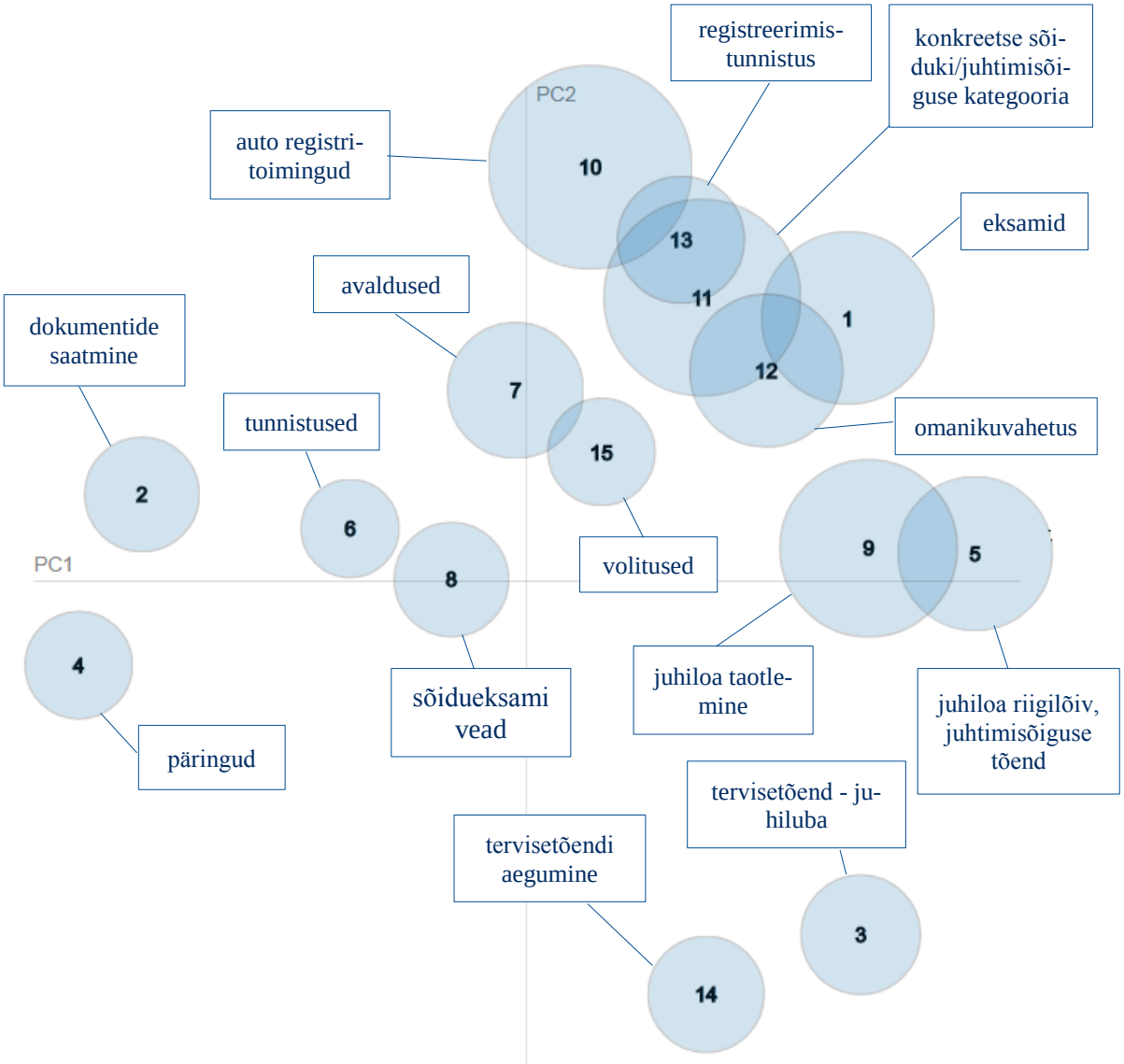
Saamaks teada teemade omavahelised ligikaudsed proportsioonid, lasti mudelil määrata igale eestikeelselsele kirjale teema (samu tekste kasutati ka mudeli treenimiseks). Selle põhjal arvutati teemade osakaal eestikeelsetes e-kirjades. Tulemused on toodud alljärgneval joonisel.



Joonis 9. Teemade osakaal eestikeelsetes e-kirjades.

Joonisel 9 on näha, et kõige populaarsem teema oli registritoimingutega seonduv (üle 15% eestikeelsetest e-kirjadest). See on põhjendatav asjaoluga, et registritoimingud on suure kasutajaskonnaga teenused. Sellele järgnevad eksamitega seonduv (13%), juhiloa taotlemisega (alla 11%) ja sõiduki/juhtimisõiguse kategooriaga (7%) seotud teemad. Kõige väiksema osakaaluga oli sõidueksami probleemidega seotud teema (ligi 2%). Kuna LDA kasutab teemade leidmiseks teksti sõnakogumit (arvestamata sõnade järjekorda) ning tegemist on juhendamata masinõppe meetodiga (täpset sihtmärki pole defineeritud), tuleb arvestada, et mudeli täpsus ei pruugi olla võrreldes juhendatud masinõppe meetoditega väga suur. Joonisel 9 toodud proportsioonid näitavad ainult teemade ligikaudset esinemist, mitte täpset jaotust.

Teemade eristuvus on visuaalselt kujutatud joonisel 10. Joonis on loodud kasutades pythoni teeki pyLDAvis[24]. Joonisel on toodud 15-teemalise LDA teemade paigutus kahedimensioonilises ruumis. Suurema dimensioonide arvuga oleks teemade omavahelisi kaugusi keeruline visuaalselt joonisel kujutada.



Joonis 10. Eestikeelsete e-kirjade teemade paigutus kahedimensioonilises ruumis.

Joonisel 10 on näha, et kahedimensioonilises ruumis eristusid suhteliselt hästi teemad 2 ja 4. Mõlemad olid seotud dokumentide saatmise/päringutega. Lisaks eristusid selgelt teemad 14 ja 3, mis olid seotud tervisetõendiga. Samuti on jooniselt näha, et teemad 10, 11, 12, 13 ja 1 olid koondunud üksteisele suhteliselt lähedale. Nendega seotud teemad olid seotud semantiliselt ja sisuliselt. Need teemad olid seotud registritoimingutega (nii sõidukitega tehtavad kui ka juhiloa taotlemisel tehtavad toimingud).

Teemade modelleerimine on juhendamata masinõppe meetod, mistõttu ei vastanud pakutud teemad täpselt Maanteeametile soovidele teemade klassifitseerimisel. Lisaks sisaldasid pa-

[24] https://github.com/bmabey/pyLDAvis

kutud teemad suhteliselt suurel määral (20%-30%) tekste, mis autori hinnangul konkreetsesse klassi ei kuulu. Selle tõttu kasutati teemade klassifitseerimiseks juhendatud masinõppe meetodeid.

5.3 E-kirjade ettevalmistamine klassifitseerimiseks

E-kirjade ettevalmistamine klassifitseerimiseks toimus kahes osas. Esiteks e-kirjade märgendamine koos ettevalmistamisega märgendamiseks, teiseks märgendatud andmete otsene puhastamine klassifitseerimiseks. Puhastamine toimus pärast märgendamist, kuna eelnev puhastamine oleks eemaldanud märgendamiseks vajalikku informatsiooni.

5.3.1 E-kirjade ettevalmistamine märgendamiseks

Andmete ettevalmistamine märgendamiseks hõlmas e-kirjas oleva vestluste eraldamist. Näiteks võis lisaks algsele e-kirjale olla järgnenud mitu e-kirja (klienditeenindaja ja kliendi vestlus). Nende eraldamine oli vajalik, kuna e-kirju tuleb klassifitseerida vestluse esimese kirja põhjal, sest järgnev vestlus toimub konkreetse ametniku ja kliendi vahel. Lisaks aitab ametniku vastuse alles jätmine kaasa mudeli üleõppimisele, kuna edasine vestlus võib sisaldada vihjeid õigele klassile. Kasutades sellist mudelit klientide esimeste e-kirjade peal, võib mudeli tulemus olla oluliselt halvem. Esimese e-kirja eraldamiseks kasutati regulaaravaldisi. Mustrid, mille põhjal eristada vestlusi, korjati andmete anonümiseerimise ja teemade mudeli jaoks andmete ettevalmistamise käigus. Ligi 10% eestikeelsetest e-kirjadest sisaldas rohkem kui ainult ühte kontakti kliendi ja ametniku vahel.

Vestluste eraldamisel välistati olukorrad, kus klient kirjutas vastuse Maanteeameti automaatsele teavitusele (näiteks lubade, tervisetõendi või ülevaatuse aegumise kohta). Automaatsed teavituste tekst annab informatsiooni, mis põhjusel klient e-kirja kirjutas. Samuti võib kliendi vastus teavitusele olla lühike (näiteks selgitus, et probleem lahendatakse järgmisel nädalal), mis ilma algse teavituseta annab vähe infot kirja teemast. Kogu vestluse eemaldamine ja automaatse teavituse alles jätmise õnnestumine sõltub konkreetsest e-kirja struktuurist. Selle tõttu võis klassifitseerimise mudelisse siiski mõningal määral sisse jääda e-kirju, kus lisaks kliendi tekstile on ka ametniku tekst. Vestluste eraldamisele järgnes andmete märgendamine.

5.3.2 E-kirjade märgendamine

Märgendamine oli oluline samm täpse kategoriseerimise mudeli loomisel. Märgendamisele eelnes Maanteeametiga märgendatavate klasside kokku leppimine. Teemad tuginevad peatükis 5.2.2 teemade modelleerimisel leitud teemadele, kuid ei vasta üks-ühele LDA pakutud teemadele. Selle tõttu tuli märgendamine teha käsitsi. Märgendatavateks klassideks on:

- **Eksamitega seonduv.** Siia klassi kuulusid nii sõidu- kui ka teooriaeksami, lõppastmekoolituse, autokoolis käimisega seonduvad e-kirjad.
- **Liiklusregistri- ja sellega seotud dokumendid.** Siia klassi kuulusid juhilubade vahetamise, omanikuvahetuse, taotlemise, arstitõendi uuendamise, sõidukite registritoimingutega seonduvad e-kirjad.
- **Tehniline.** Siia klassi kuulusid sõidukite ümberehitamise, ülevaatusega seonduvad e-kirjad.
- **Muu.** Siia klassi kuulusid ülejäänud e-kirjad. Näiteks e-kirjad, kus Maanteeametile saadeti mõni dokument, ilma rohkem täpsustamata, mis teemal ja miks seda tehakse.

Andmete märgendamisega tegeles töö autor. Märgendamiseks kasutati eeltoodud klasside alamklasse (kokku ligi 100), et võimalikult täpselt määrata tekstide sisu. See võimaldas vajadusel tulevikus e-kirju teistsugustesse klassidesse jaotada. Kokku märgendati 7 941 e-

kirja. Märgendamiseks aluseks võeti tekstide juhuvalim. Märgendamiseks valiti selline hulk e-kirju kahel põhjusel. Esiteks pidi igas klassis olema märgendatud üle 100 e-kirja. Teiseks aja kokkuhoidmiseks seati märgendamise maksimaalseks kestvuseks 2 nädalat. Märgendamisel sai iga e-kiri ühe (alam)klassi sildi. Märgendamise kiirendamiseks kasutatud treenitud teemade mudeli poolt määratud klasse. See aitas moodustada märgendatavatest andmetest väikesemad andmehulgad, kus suure tõenäosusega oli rohkem mõne konkreetse klassi tekste.

Pärast tekstide märgendamist määrati igale tekstile vastavalt määratud sildile klass, mis vastab eeltoodud loetelule. Märgendatud tekstide jagunemine klasside vahel koos klassi lühikirjeldusega on toodud tabelis 4.

Tabel 4. Märgendatud e-kirjade jagunemine klasside vahel

Klassi nimi	**E-kirjade arv**	**Klassi lühikirjeldus**	**Seotud teemad**
eksam	2498	e-kirjad, mis on seotud teooria-, sõidueksamiga, autokoolis käimisega	eksamid; sõidueksami vead
load_dokumendid	2226	e-kirjad, mis on seotud juhilubade ja muude dokumentidega ning liiklusregistri toimingutega (näiteks omanikuvahetus)	tervisetõend – juhiluba; juhiloa riigilõiv, juhtimisõiguse tõend; tunnistused; avaldused; juhiloa taotlemine; auto registritoimingud; konkreetse sõiduki/juhtimisõiguse kategooria; omanikuvahetus; registreerimistunnistus; tervisetõendi aegumine; volitused
tehniline	427	e-kirjad, mis on seotud sõidukite tehnilise poolega: ülevaatus, ümberehitamine, tüübikinnitus	auto registritoimingud; konkreetse sõiduki/juhtimisõiguse kategooria
muu	2790	E-kirjad, mis ei sobi eelnevatesse klassidesse (näiteks dokumentide saatmine ilma viiteta, miks dokumenti saadetakse)	dokumentide saatmine; päringud; avaldused; registreerimistunnistus

Tabelist 4 on näha, et klass „tehniline“ oli teistest oluliselt väiksemate näidete arvuga, mistõttu võib selle klassi eristumisel tekkida probleeme. Ülejäänud klassidesse kuulus tekste enam-vähem võrdsel määral. Samuti on tabelist näha, miks ei saanud kasutada otse LDA pakutud teemasid tekstide märgendamiseks. Nimelt võis üks LDA pakutud teema sisaldada mitut Maanteeameti jaoks vajalikku klassi (näiteks teema „avaldus“ märgitud tekstid võisid kuulud klassidesse „load_dokumendid“ ja „muu“).

5.3.3 E-kirjade puhastamine klassifitseerimiseks

Märgendamisele järgnes andmete puhastamine klassifitseerimismudelite treenimiseks. Selleks tehti igas e-kirja tekstis järgnevad muudatused:

- Eemaldati kirjavahemärgid ja numbrid.
- Tähed muudeti väiketähtedeks.
- Eemaldati pikemad väheinformatiivsed tekstiosad (näiteks konfidentsiaalsushoiatused). Selleks kasutati juba eelnevalt teemade modelleerimise jaoks ettevalmistamisel leitud väheminformatiivseid tekstiosi.
- Lemmatiseeriti kõik sõnad. See aitas vähendada andmete dimensionaalsust ning muuta klassifitseerijat täpsemaks.
- Eemaldati üksikud tühisõnad, võttes aluseks Raigo Kodasmaa bakalaureusetöös toodud loetelu, mida oli täiendatud teemade modelleerimisel leitud sõnadega (Kodasmaa, 2011).

Andmete puhastamise eesmärgiks ei olnud kõikide väheinformatiivsete tekstiosade eemaldamine, kuna see oleks olnud liiga ajakulukas. Eesmärgiks oli eemaldada piisaval määral väheinformatiivseid tekstiosi ning vältida otseste klassikuuluvust määravate vihjete sattumist treeningandmetesse. Viimase tulemusel võidakse saada üle treenitud mudel. Andmete puhastamisele järgnes klassifitseerimismudelite treenimine.

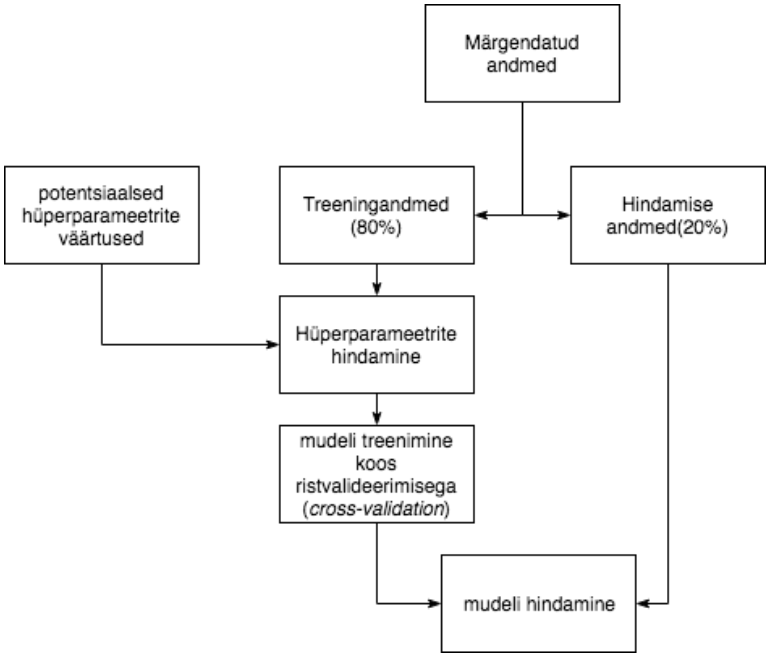
5.4 E-kirjade klassifitseerimine

E-kirjade klassifitseerimiseks kasutati alapeatükis 5.3.3 mainitud e-kirju. E-kirjade klassifitseerimise mudelite loomiseks ja võrdlemiseks rakendati alapeatükis 3.4 kirjeldatud mudeleid ning alapeatükis 3.5 kirjeldatud meetodeid mudelite täpsuse suurendamiseks. Erinevate mudelite ja meetodite rakendamise eesmärgiks oli selgitada välja, kas mõni neist aitaks treenida süstemaatiliselt täpsemaid mudeleid.

5.4.1 Eksperimendi ülesehitus

Mudelite treenimiseks kasutati linux operatsioonisüsteemiga keskkonda: Debian GNU/Linux 8 (jessie). Keskkond töötab protsessoril Intel® Core™ i7-7700K CPU@4.20GHz. Mudelite treenimiseks kasutati lahendust Jupyter notebook, mis kasutas pythoni teeki Jupyter Hub[25]. Käesolevas töös kasutatav klassifitseerimismudelite treenimise protsess on kujutatud joonisel 11.

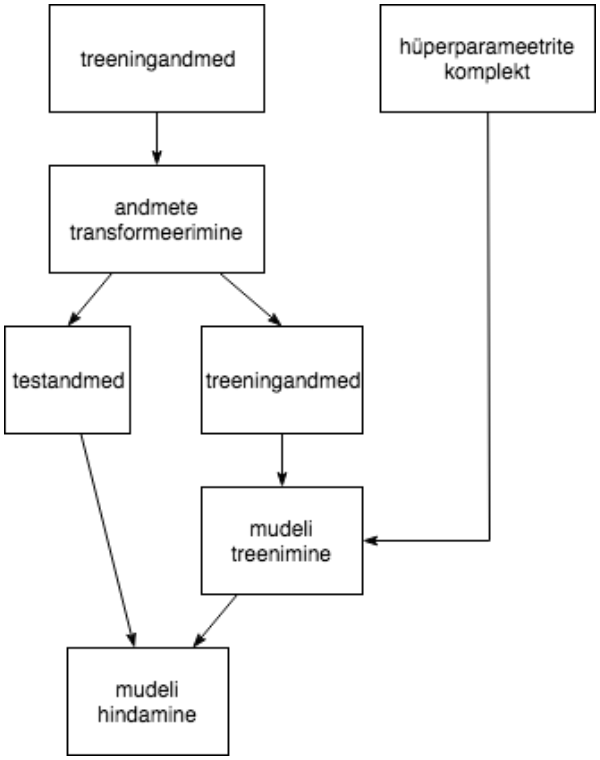
[25] https://jupyterhub.readthedocs.io/en/stable/index.html



Joonis 11. Klassifitseerimismudelite treenimise protsessi skeem.

Enne mudeli treenimist jagati märgendatud e-kirjade hulk kaheks: treeningandmed (80%) ning hindamise andmed (20%). Viimast kasutatakse lõpliku mudeli täpsuse hindamiseks. Kasutades mudeli täpsuse hindamiseks ainult treeningandmeid, võib mudeli täpsust ülehinnata, kuna mudel on suure tõenäosusega üle õppinud. E-kirjade jagamine treening- ja hindamise andmeteks toimus juhuslikkuse alusel.

Pärast andmete jagamist hinnati, millised hüperparameetri väärtuste komplektid võivad aidata treenida ülejäänud komplektidest täpseima mudeli. Selleks valiti välja võimalikud hüperparameetrite väärtused iga mudeli jaoks. Valiku tegi autor tuginedes kirjandusest leitule ja praktilistele eksperimentidele. Tüüpiliselt anti igale hüperparameetrile ette umbes 3-5 väärtust. Seejärel leiti võrkotsingu abil hüperparameetrid, mille puhul mudeli täpsus oli suurim. Hüperparameetrite hindamise protsessi skeem on kujutatud joonisel 12.



Joonis 12. Mudeli hüperparameetrite hindamise protsessi skeem.

Joonisel 12 kujutatud protsess läbitakse iga hüperparameetrite komplekti korral kolm korda, kuna hüperparameetrite mõju mudeli täpsusele hinnatakse 3-kordse ristvalideerimisega (3-*fold cross-validation*). Arvestades hüperparameetrite ja väärtuste suhteliselt suurt hulka (tüüpiline hüperparameetrite hulk mudelis jäi vahemikku 3-8, millest igale anti ette 3-5 väärtust) annab 3-kordne ristvalideerimine märgatava ajalise kokkuhoiu 10-kordse ristvalideerimise ees. Enne mudeli treenimist viiakse eelnevalt läbi järgnev andmete transformeerimine:

- Tekstidest moodustati dokumendi-tunnuse (*document-term*) maatriks, lugedes kokku kui mitu korda iga sõna esineb konkreetsest dokumendis. Lisaks võidi lisada tunnustena ka n-gramme (tegemist on hüperparameetriga, mille väärtust hinnati ristvalideerimisega).
- Andmaks klasse rohkem eristavatele sõnadele suurem kaal maatriksis, kasutati alapeatükis 3.1.3 kirjeldatud tf-idf transformatsiooni. See aitas vähendada nende tunnuste kaalu, mis on enamikus dokumentides laialt levinud ning seetõttu väheinformatiivsed.
- Seejärel valiti välja klasse kõige paremini eristavad unikaalsed tunnused, kasutades alapeatükis 3.2 kirjeldatud hii-ruut statistikut. Tüüpiliselt jäi unikaalsete tunnuste arv 1 000 ja 1 400 vahele. Tunnuste arv on üks hüperparameeter, mille optimaalne suurus leiti hüperparameetrite hindamise etapis kasutades ristvalideerimist.

Transformeerimist ei tehtud fasttexti puhul, kuna fasttext ei kasuta sisendina maatriksit vaid tekste koos määratud klassidega[26]. Selle tõttu ei kasutatud fasttexti puhul ka erinevaid täpsust suurendavaid meetodeid, kuna nende kasutamine eeldaks fasttexti C-keeles kirjutatud koodi muutmist.

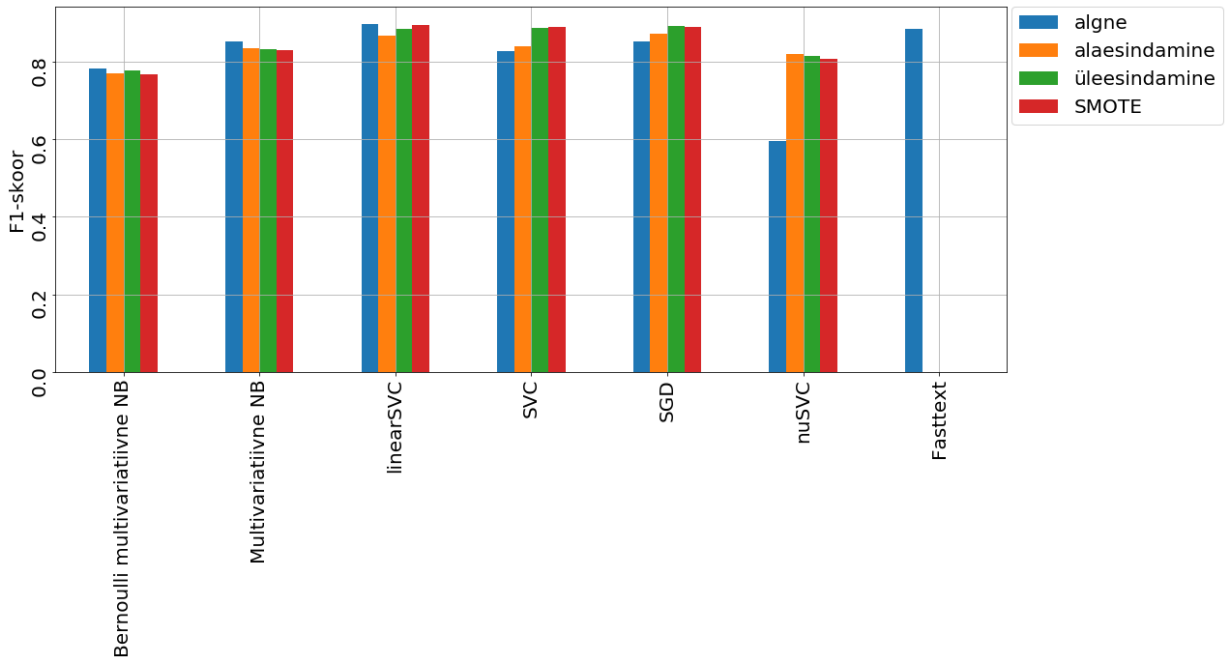
[26] https://fasttext.cc/docs/en/faqs.html

Pärast transformeerimist kasutati treeningandmetel ristvalideerimist, leidmaks võimalikud täpsema mudeli loomise hüperparameetrite komplektid. Ristvalideerimine aitas mõningal määral hinnata hüperparameetrite komplekti mõju püsivust mudeli täpsusele. Kasutades mudeli hindamiseks ainult ühte testandmete kogumit, ei ole võimalik hinnata kui stabiilne on mudeli tulemus. Mudeli täpsuse ja parimate hüperparameetrite komplektide hindamiseks kasutati kaalutud F_1-skoori.

Pärast etteantud hüperparameetrite komplektist optimaalseimate leidmist treeniti mudel. Mudeli treenimise protsess on sarnane joonisel 12 kujutatud protsessile. Erinevus seisneb selles, et hüperparameetrite komplekt oli fikseeritud ning iga mudeli puhul läbitakse protsess 10 korda (10-*fold cross-validation*), kuna treeningandmed jaotatakse kümneks. Kümneks jaotamine aitas täpsemalt hinnata, kui variatiivsed on mudeli tulemused kasutades sisendina erinevaid treeningandmete alamhulki. Põhjuseks, miks hüperparameetrite hindamise etapile lisaks tehakse uus ristvalideerimine, on, et lõpliku ristvalideerimise korral salvestatakse oluliselt rohkem informatsiooni iga ristvalideerimise etapi kohta. Hüperparameetrite hindamisel oleks selle informatsiooni salvestamine iga ristvalideerimise etapi kohta suurendanud oluliselt ebavajaliku info hulka. Lisaks oleks 10-kordne ristvalideerimine 3-kordse ristvalideerimise asemel oluliselt suurendanud mudeli hüperparameetrite hindamiseks kulunud aega. Lõpuks treeniti mudel kogu treeningandmestiku peale. Mudeli täpsuse hindamiseks kasutati kaalutud F_1-skoori. Eelkirjeldatud protsessi põhjal loodud mudelite täpsused on toodud Lisas II.

5.4.2 Andmete esinduse muutmise mõju F_1-skoorile

Esimesena hinnati andmete esindust muutvate meetodite mõju erinevate mudelite täpsusele. Tulemused on toodud joonisel 13.



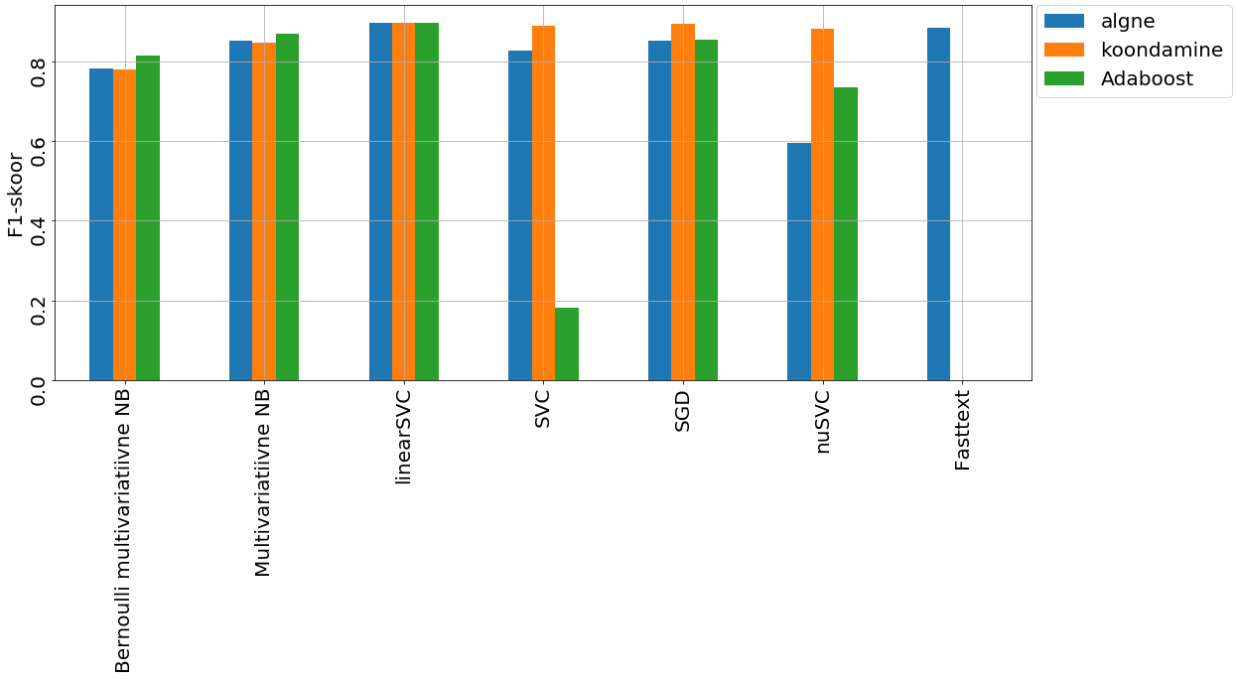
Joonis 13. Algsete ja andmete esinduse muutmise meetoditega mudelite hindamise kaalutud F_1-skoorid.

Jooniselt on näha, et kõige täpsemad algsed mudelid saadi linearSVC (F_1-skoor 0,898) ja fasttextiga (F_1-skoor 0,885). Kaks kõige madalama skooriga mudelit on Bernoulli multivariatiivne NB (F_1-skoor 0,781) ja nuSVC (F_1-skoor 0,586). Andmete esinduse meetodite kasutamisel oli erinev mõju. Kõige täpsemale mudelile - linearSVC - oli mõju negatiivne

(ühegi meetodiga ei tõusnud F_1-skoor üle 0,893). Kõige suurem positiivne mõju täpsusele oli mudelile nuSVC (F_1-skoori kasv 0,461-lt kuni 0,833-ni). Naïve Bayes'i mudelite täpsus oli samuti andmete esinduse meetodite rakendamise tulemusel mõnevõrra vähenenud. SVMi mudelite täpsuste puhul on näha mustrit, et meetodit SMOTE kasutanud mudelid olid täpsemad kui üleesindamist kasutanud mudelid ning viimased olid täpsemad alaesindust kasutanud mudelitest. See on selgitatav faktiga, et SMOTE genereerib uusi andmeid, mis aitab maandada üleõppimise riski ja seega suurendada mudeli täpsust uute andmete peal. Üleesindamise eeliseks alaesindamise ees on andmete variatiivsuse säilitamine. Alaesindamine võib eemaldada andmetest olulise info, mis võib viia madalama täpsusega mudelini.

5.4.3 Ansambelmeetodite mõju F_1-skoorile

Järgmisena hinnati ansambelmeetodite rakendamise mõju mudelitele. Kuna kuhjamine võib aluseks võtta mitu eri tüüpi mudelit, tuuakse selle tulemused alapeatükis 5.4.5. Ansambelmeetodite rakendamisel saadud F_1-skoorid on kujutatud joonisel 14.

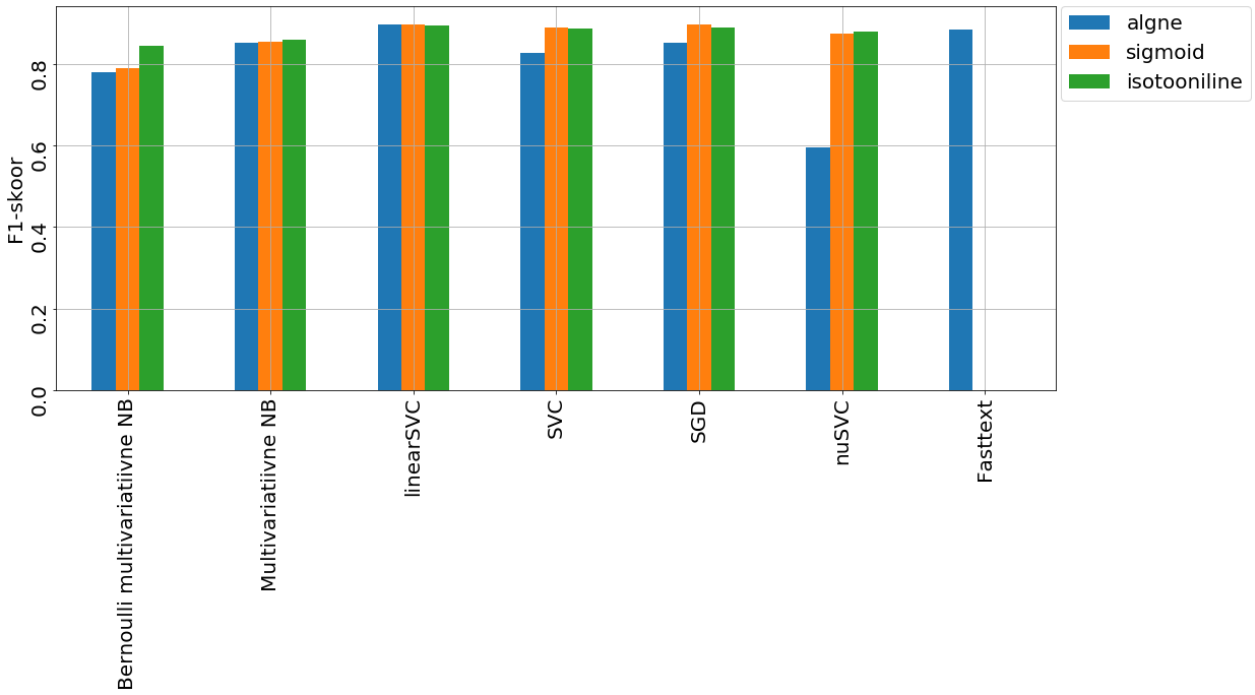


Joonis 14. Algsete ja ansambelmeetoditega mudelite hindamise kaalutud F_1-skoorid.

SVMi erinevate mudelite puhul aitas kõige rohkem täpsust suurendada koondamine (näiteks SVC puhul kasvas algse mudeli F_1-skoor 0,837-lt kuni 0,890-ni). Seda võib põhjustada üksikute mudelite variatiivsus, mis suurema hulga mudelite koondamisel väheneb. Erinevate Naïve Bayes'i mudelite puhul andis parima tulemuse *Adaboost* (näiteks multivariatiivse Naïve Bayes'i puhul kasvas F_1-skoor 0,853-lt kuni 0,861-ni). Koondamine Naïve Bayes'i mudelite täpsust ei suurendanud. Kõige täpsema mudeli puhul (linearSVC) täpsus ansambelmeetodeid kasutades ei suurenenud. Lisaks on jooniselt 14 näha, et mudeli SVC puhul *Adaboost*'i rakendades täpsus ei kasvanud. Üheks põhjuseks võib olla, et algne mudel oli piisavalt hea. Alapeatükis 3.5.2 mainitud võimendamine suurendab täpsust, kui algne mudel on suhteliselt ebatäpne.

5.4.4 Tõenäosuste kalibreerimise mõju F_1-skoorile

Kolmandana hinnati mudelite tõenäosuste kalibreerimise mõju mudelite täpsusele. Kalibreerimise rakendamisel saadud F_1-skoorid on kujutatud joonisel 15.

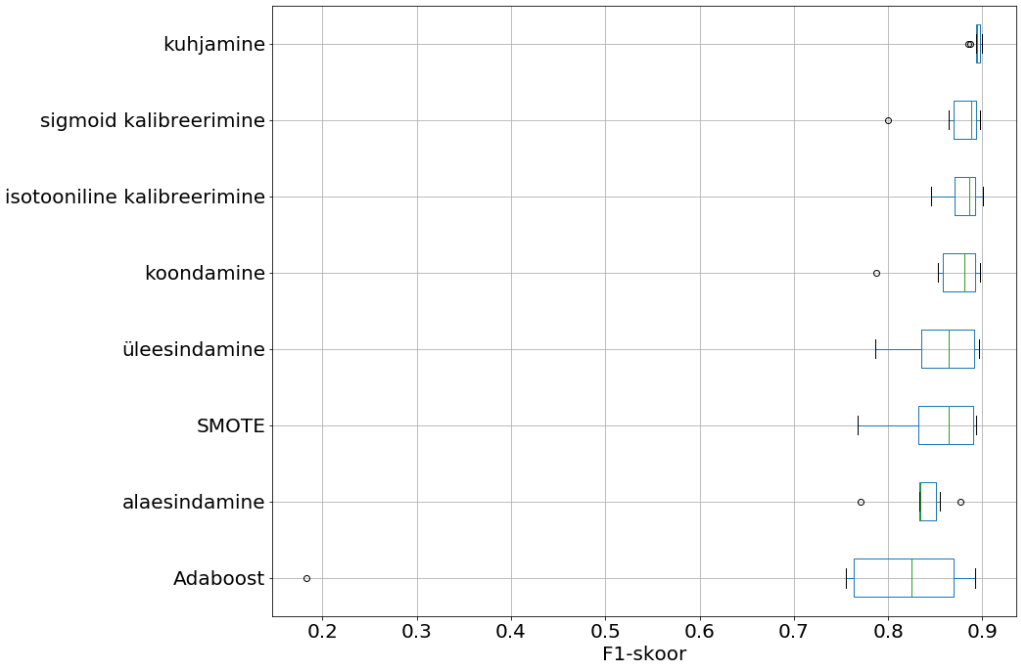


Joonis 15. Algsete ja kalibreeritud mudelite hindamise kaalutud F_1-skoorid.

Jooniselt 15 on näha, et kalibreerimine suurendas peaaegu kõigi mudelite täpsust. Võrreldes teiste meetoditega puuduvad kalibreerimisel mudelid, mille täpsus halvenes. Kõige täpsema mudeli (linearSVC) täpsus muutus kalibreerimise tulemusel minimaalselt (isotoonilise kalibreerimise korral kasvas F_1-skoor 0,897-lt kuni 0,901-ni, sigmoidkalibreerimise korral jäi samaks). Üldjuhul ei ole olulisi vahesid sigmoid- ja isotoonilise kalibreerimisega saadud mudelite täpsustes. Kõige rohkem kasvas kalibreerimise tulemusel kõige madalama algse täpsusega mudeli nuSVC täpsus (F_1-skoor kasvas 0,461-lt kuni 0,883-ni sigmoidkalibreerimisega).

5.4.5 Kuhjamine, meetodite ja mudelite koondvõrdlus

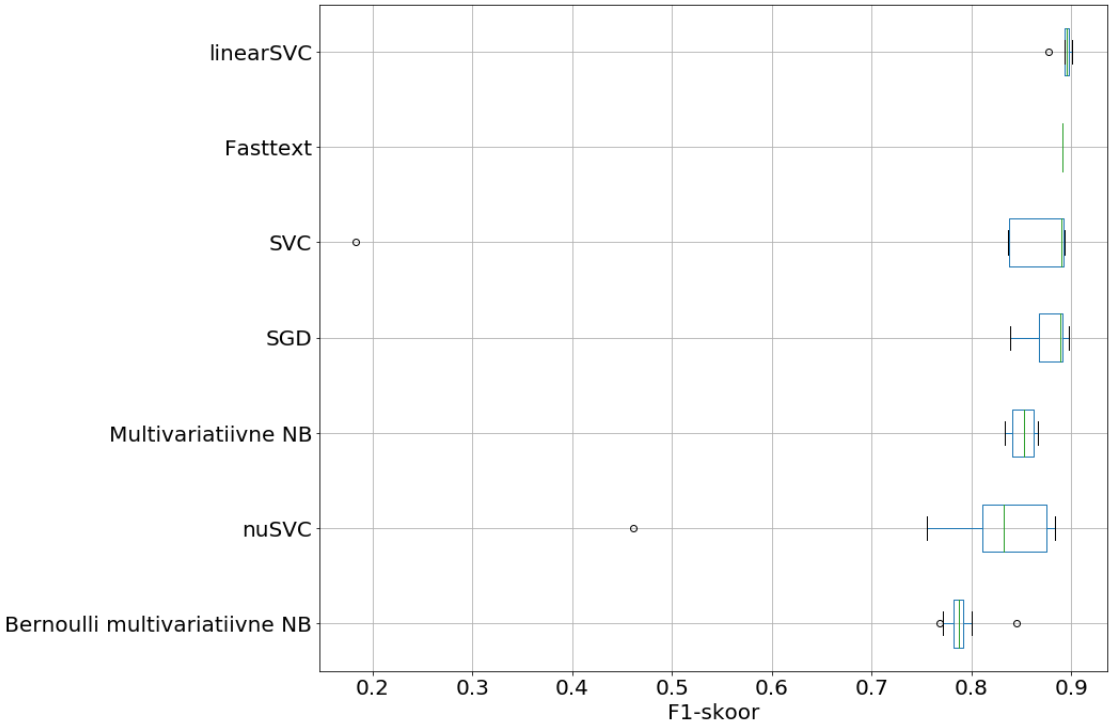
Pärast kalibreerimise kasutamist treeniti kuhjamist kasutavad mudelid. Alusmudeliteks valiti senistest mudelitest täpseimad. Lisaks kombineeriti mõned kuhjatud mudelid ka veidi vähem täpsematest mudelitest. Lisaks erinevatele mudelitele ja mudeli täpsuse suurendamise meetoditele prooviti erinevaid alapeatükis 3.5.2 mainitud hääletamise meetodeid. Kasutati nii enamushääletust (*Plurality Voting*) kui ka „pehmet“ hääletamist (*Soft Voting*). Lisaks kasutati fasttextiga mudelite puhul mudelite häälte kaalumist (fasttexti kaaluks anti 0,5 ning ülejäänud mudelitele 1. Kaaluga korrigeeriti mudeli väljastatud tõenäosusi. Kaalutud tõenäosustele rakendati „pehmet“ hääletamist). Fasttexti puhul rakendati kaalumist, kuna tulenevalt softmax-kihist väljastas mudel suuremal hulgal kõrgemaid tõenäosusi kui teised mudelid. Kaalumine aitas mudelite tulemusi muuta kuhjamise jaoks paremini võrreldavaks. Lisas 1 on toodud kirjeldus, millist hääletamise meetodit kasutati. Kokku loodi 12 kuhjamise teel saadud mudelit. Kuhjamise mudelite täpsuse hajuvus oli väike (minimaalse ja maksimaalse F_1-skoori vahe oli 0,02 ühikut). Seega oli kuhjamine suhteliselt töökindel meetod, mis suudab tagada kõrge täpsuse. Seda tõendab joonis 16, kus on kujutatud mudelite F_1-skooride jagunemine rakendatud meetodite lõikes.



Joonis 16. Hindamise kaalutud F_1-skooride jagunemine meetodite lõikes.

Kuhjamise mudelite hajuvus oli kõige väiksem ning kõige kõrgema mediaan F_1-skooriga (0,893). Teiste meetodite hajuvus oli oluliselt suurem ning sõltus rohkem kasutatavast mudelist. Järgmine meetod, mis aitas mudelite täpsust kõige rohkem suurendada oli kalibreerimine. Kalibreerimise korral ei vähenenud ühegi mudeli täpsus võrreldes algse mudeliga. Kõige suurema hajuvuse ja madalaima keskmise F_1-skooriga oli meetod *Adaboost*. Üheks põhjenduseks on fakt, et võimendamise alusmudeliks peaks olema piisavalt ebatäpne mudel, et mudeli täpsust kasvatada. Vastasel korral võib efekt olla vastupidine.

Lisaks meetoditele hinnati mudelite tulemuste püsivust vaadeldes erinevaid meetodeid kasutades loodud mudelite F_1-skooride hajumist. Tulemused on kujutatud joonisel 17 (välja on jäetud kuhjamisega seotud mudelid).

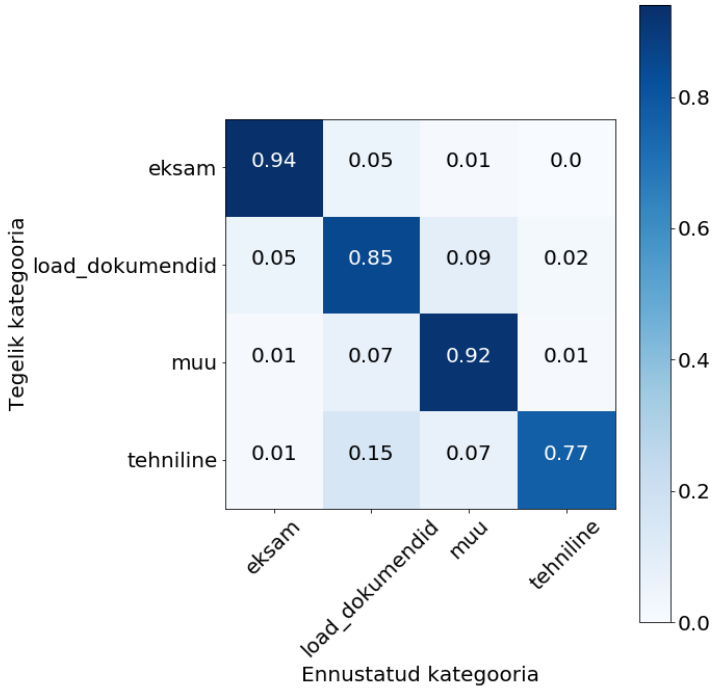


Joonis 17. Hindamise andmete F_1-skooride jagunemine mudelite lõikes.

Jooniselt 17 on näha, et kaks kõige täpsemat mudelit olid linearSVC (mediaan F_1-skoor 0,895) ja fasttext (mediaan F_1-skoor 0,890). Need olid võrreldavad kuhjamisel saadud mediaan F_1-skooriga (0,893). Nii mudeli linearSVC kui ka meetodiga kuhjamine treenitud mudelite F_1-skooride hajuvus oli minimaalne. Küllaltki stabiilne mudel oli Bernoulli multivariatiivne Naïve Bayes, kuid mille keskmine F_1-skoor oli teistest madalaim (alla 0,8). Mudeli nuSVC mediaan F_1-skoor oli kõrgem, kuid samas oli hajuvus teistest mudelitest suurem (minimaalne F_1-skoor oli 0,461, maksimaalne F_1-skoor oli 0,885).

5.4.6 Täpseim mudel

Kõrgeima F_1-skooriga mudel oli kuhjamise teel saadud koondmudel, mis kasutas alusena järgmiseid mudeleid: linearSVC isotoonilise kalibreerimisega, SVC koondamisega ja SGD isotoonilise kalibreerimisega (kasutades „pehmet“ hääletamist). Mudeli eksimismaatriks on toodud alljärgneval joonisel.



Joonis 18. Täpseima mudeli (kuhjamine: linearSVC istooniline kalibreerimine, SVC koondamine, SGD isotooniline kalibreerimine) eksimismaatriks (tegeliku kategooria vaatluste jagunemine ennustatud kategooriate vahel osakaaludena).

Kõige kõrgema täpsusega klass oli „eksam“ (kategooria F_1-skoor oli 0,95). Jooniselt 18 on näha, et 94% sellesse kategooriasse kuuluvatest e-kirjadest klassifitseeriti õigesti, 5% kategooriasse „load_dokumendid“ ja 1% kategoorisasse „muu“. Täpsuselt järgnev klass oli „muu“ (F_1-skoor 0,91). Sellele järgnes klass „tehniline“ (F_1-skoor 0,86) ning klass „load_dokumendid“ (F_1-skoor 0,85). Klassi „load_dokumendid“ madal täpsus oli tingitud sellest, et mudel eksis selle klassi eristamisel kõige rohkem. Tegemist oli klassiga, mis teataval määral oli seotud teiste klassidega (lubade ja dokumentidega seonduv võis olla seotud nii eksamite, sõiduki tehnilise poole kui ka muuga). Valesti klassifitseeritud e-kirjade arvu põhjal eksis mudel kõige rohkem klasside „load_ dokumendid“ ja „muu“ eristamisel (154 klassis „muu“ olevat e-kirja määras mudel klassi „load_dokumendid“ ja 152 klassis „load_dokumendid“ olevat e-kirja määras mudel klassi „muu“). Pärast valesti hinnatud tekstide vaatlust selgus, et suurem osa valesti hinnatud tekste tulenesid mudeli eksimusest mitte märgendamisel tehtud vigadest. Mõne ükskiku e-kirja puhul võis neid määrata nii kategooriasse „load_dokumendid“ kui ka „muu“, kuna täpne klass oli autori hinnangul vaieldav. Mudeli täpsuse suurendamiseks tuleks suurendada „load_dokumendid“ alamklassi täpsust.

5.5 Järeldused

E-kirjade analüüsimisel õnnestus tuvastada optimaalne e-kirjade teemade arv ning modelleerida teemasid. Teemade modelleerimine aitas suhteliselt lühikese ajaga anda ülevaate e-kirjades sisalduvatest teemadest. Automaatselt optimaalse teemade arvu leidmine on keeruline, kuid koherentsus aitab siiski leida ligikaudset piirkonda, kus see asuda võib. Oluline aspekt on eelnev andmete puhastamine.

Samuti õnnestus luua täpsed klassifitseerimise mudelid, mille maksimaalsed F_1-skoorid olid 0,9 lähedal. Siiski on treenitud mudeli täpsus pigem maksimaalne võimalik täpsus ning uute andmete peal võib nende täpsus olla madalam järgnevatel põhjustel:

- Klassifitseerimise mudeli treenimise aluseks ei olnud kogu aasta e-kirjad. Sesoonsusest tulevalt võivad aasta viimaste kuude e-kirjad olla erineva sisuga kui ülejäänud aasta e-kirjad. Samuti võivad aja jooksul muutuda e-kirjades olevad teemad ja klientide sõnakasutus.
- Analüüsist jäid välja e-kirjad, millel puudus sisuline tekst, kuid mis võisid sisaldada manust. Selliste kirjade täpseks klassifitseerimiseks oleks vaja eelfiltrit, mis suudab kirja klassi tuvastada kas manuse sisu või mõne muu info põhjal.
- Andmete käsitsi anonümiseerimine muutis andmeid viisil, mida tulevikus ei ole võimalik automaatselt teha. See võib mõjutada mudeli täpsust, kuna praktikas hinnatavaid andmeid ei ole võimalik enne hindamist käsitsi anonümiseerida. See tähendab, et mudeli loomisel olevad andmed ja praktikas olevad andmed erinevad teataval määral.
- Andmete puhastamisel võis mõnesse e-kirja sisse jääda klienditeenindaja vastus, mis võib põhjustada mudeli üle õppimist (uute andmete puhul kliendi kirjal teenindaja vastust ei ole).

Erinevatest mudelitest andsid kõige parema tulemuse linearSVC ning fasttext. Mudeli täpsuse suurendamise metoodikatest andsid stabiilseid tulemusi kuhjamine ning kalibreerimine. Täpseimate mudelite (linearSVC ja fasttext) tulemused olid maksimaalse F_1-skoori lähedal ka ilma erinevaid täpsuse suurendamise meetodeid kasutamata. Ristvalideerimise ja hindamise tulemused näitavad, et suure osa mudelite täpsused koonduvad suhteliselt väikesesse vahemikku (suure osa mudelite F_1-skoorid jäid vahemikku 0,8 - 0,9).

Samas raamistiku veelgi üldisemaks ja kiiremaks kasutamiseks tuleks seda täiendada. Tulevikus oleks vaja analüüsida järgnevaid aspekte:

- Anonümiseerija täiendamine. Hetkel kasutatav anonümiseerija ei suuda eemaldada näiteks Skype'i nimesid ning on kontekstitundlik (sõltub eelnevatest ja järgnevatest tähemärkidest). Käsitsi anonümiseerimine moodustas olulise osa käesolevale tööle kulunud ajast. Selle aja vähendamine aitaks oluliselt kiirendada raamistiku uuesti rakendamist teistes asutustes.
- Märgendamise kiirendamine. Teine suur käsitsi tehtav töö oli andmete käsitsi märgendamine. Märgendamise juba osaline automatiseerimine võimaldab oluliselt aega kokku hoida.
- Andmete kiirem puhastamine mudeli LDA jaoks. Käesolevas töös oli LDA jaoks andmete puhastamine iteratiivne protsess, mis oli ühtlasi teemade modelleerimise kõige kulukam etapp. Käesolevas töös tekkinud tühisõnade nimekiri on osaliselt rakendatav ka muudes asutustes. Samas võib mõnes muus asutuse e-kirjades olla teistsugune hulk tühisõnu, mis tuleb uuesti käsitsi leida. Tühisõnade kiirem leidmine ja eemaldamine võimaldaks oluliselt kiirendada teemade modelleerimist uues asutuses.

Hinnanguliselt aitab käesolevas töös välja töötatud raamistik Maanteeametil kokku hoida ühe inimese tööaja. Töö autor on seisukohal, et raamistik on rakendatav uutele dokumentidele. Teemade modelleerimisega saab luua ülevaate dokumentides sisalduvatest teemadest. Koherentsuse abil on võimalik määrata tõenäoline teemade arvu piirkond. Olulise osa raamistikus tehtavast käsitsi tööst hõlmab andmete anonümiseerimine, puhastamine ja märgendamine. Töö autori hinnangul on dokumentide klassifitseerimisel saavutatud täpsus piisav rakendamaks mudelit praktikas. Täpsemad tulemused andsid kuhjamise teel saadud mudelid. Neist ei jäänud oluliselt maha nii lineaarse kerneliga SVM kui ka fasttext. Uute dokumentide peal raamistiku kasutamise tulemused sõltuvad nii kasutatavatest mudelitest kui ka mudeli aluseks olevatest dokumentidest.

6. Kokkuvõte

Lõputöö teemaks on e-kirjadest teemade tuvastamise ja e-kirjade klassifitseerimise raamistiku loomine Maanteeameti näitel. Maanteeamet on avaliku sektori organisatsioon, millele kliendid saadavad ligikaudu 100 000 e-kirja aastas ning e-kirjade hulk kasvab ligikaudu 8% aastas (Lisa 1). Töö eesmärgiks oli luua raamistik e-kirjade klassifitseerimiseks teemade järgi. Käesolevas töös loodud tekstide klassifitseerimise võimaldab vähendada Maanteeametis inimeste e-kirja vastamisele kuluvat töötajate ressurssi ning muuta e-kirjadele vastamine kiiremaks. Raamistikku saab rakendada teistes asutustes eesmärgiga muuta e-kirjade vastamine automaatsemaks.

Töö teoreetilises osas anti ülevaade tekstikaevest. Tekstikaeve tegeleb mittestruktureeritud tekstiliste andmete analüüsiga. Käesoleva töö raames on olulised ülesanded, mida tekstikaeve aitab lahendada, teemade modelleerimine ja tekstide klassifitseerimine. Teemade modelleerimise eesmärgiks on saada võimalikult kiiresti ülevaade tundmatustes dokumentides esinevatest teemadest. See aitab kokku hoida aega (kõikide tekstide läbi lugemine on ajamahukas) ning kvantitatiivselt hinnata teemade osakaalu dokumentides. Üheks enamlevinud teemade modelleerimise mudeliks on LDA, mis väljastab teemad sõnade loeteluna.

Dokumentide klassifitseerimiseks kirjeldati mitmeid mudeleid. Üks lihtsamaid neist on Naïve Bayes, mis võtab arvesse sõnade esinemissagedusi erinevates klassides. Mudel teeb naiivse eelduse, et sõnade esinemine konkreetses dokumendis on sõltumatud. Praktikas on mudel näidanud suhteliselt häid tulemusi. Teiseks analüüsitavaks mudeliks oli SVM. SVM püüab erinevate klasside tunnuste vahel leida otsustuspiiri, mis asuks võimalikult kaugel klassi esimestest andmepunktidest. Praktikas võivad erinevate klasside andmepunktid siiski kattuda. Erinevalt mudelist Naïve Bayes, suudab SVM andmeid projitseerida kõrgematesse dimensioonidesse ning selletõttu leida mittelineaarseid (mis ei ole sirgjoonelised) otsustuspiire. Kolmandaks analüüsitavaks mudeliks oli fasttext, mis põhineb närvivõrkudel. Fasttext on hiljuti välja töötatud mudel, mis suhteliselt kiire, kuna kasutab erinevaid optimeerimise võtteid.

Lisaks mudelitele vaadeldi erinevaid meetodeid klassifitseerimismudeli täpsuse suurendamiseks ja täpsuse hindamiseks. Mudeli täpsuse hindamiseks on otstarbekas kasutada F_1-skoori, kui mõni klass on teistest väiksema esindatusega. Ainult õigesti klassifitseeritud andmete osakaalu kasutamine täpsuse hindamiseks olukorras, kus mõni klass on alaesindatud, võib anda mudeli täpsusele kallutatud hinnangu.

Mudeli täpsuse suurendamiseks vaadeldi kolme liiki meetodeid. Esiteks on olemas ansambelmeetodid, kus ühe mudeli asemel treenitakse mitu mudelit. See võib aidata vähendada nii klassifitseerija viga kui ka variatiivsust. Teiseks vaadeldi andmete esinduse muutmise meetodeid, kus eri klassidesse kuuluvate andmehulki proovitakse võrdsustada. Tasakaalustamata klassidega treeningandmed võivad vähese andmepunktidega klassi eristamise muuta keeruliseks. Kolmandaks vaadeldi kalibreerimist. Kalibreerimisel korrigeeritakse mudeli ennustatud klasside tõenäosusi.

Analüüsitavate e-kirjade analüüsimiseks anonümiseeris autor koostöös Maanteeametiga nende sisu. Suurem osa anonümiseerimisest toimus automaatselt spetsiaalse rakenduse abil. Ligikaudu 10% nimede asendustest tehti käsitsi. Pärast anonümiseerimist tuvastati e-kirjade keel, kuna edasises analüüsis kasutati eestikeelseid e-kirju. Teemade modelleerimiseks ja klassifitseerimismudelite loomiseks kasutati 23 639 e-kirja. Keele tuvastamisele järgnes andmete puhastamine, mis hõlmas numbrite, kirjavahemärkide ja väheinformatiivsete tekstiosade eemaldamist ning lemmatiseerimist.

Teemade modelleerimisel hinnati erinevaid meetodeid optimaalse teemade arvu leidmiseks. Selleks kasutas autor nii inimhinnangut kui ka koherentsust. Tulemused varieerusid ning täpselt optimaalse teemade arvu määramine oli keeruline. Samas aitas koherentsus määrata, millises vahemikus võib optimaalne teemade arv olla. Autori hinnangul ei ole täpne teemade arvu leidmine eesmärk omaette. Praktikas piisab, kui teada võimalikku optimaalset teemade arvu vahemikku. Teemade modelleerimine on subjektiivne (sõltub andmete puhastusastmest, modelleerija eelteadmisest tekstidest), mistõttu võib täpne teemade arv olla subjektiivne. Käesoleva töö puhul leiti, et Maanteeameti e-kirjade puhul oli optimaalne teemade arv 15.

Leitud teemade teemad olid aluseks e-kirjade märgendamiseks klassifitseerimismudelite jaoks. Lisaks viis töö autor läbi Maanteeameti klienditeeninduse spetsialistiga läbi intervjuu selgitamaks välja klassid, mis vajalikud e-kirjade klassifitseerimiseks. Pärast andmete märgendamist loodi klassifitseerimismudelid, kasutades erinevaid mudelite Naïve Bayes, SVM implementatsioone ning mudelit fasttext. Lisaks kasutati eelmainitud meetodeid mudeli täpsuse suurendamiseks. Üks täpseimad mudeleid oli lineaarne SVM. Mudelite täpsust aitas kõige rohkem suurendada kuhjamine (eri tüüpi mudelite koondamine üheks tervikmudeliks) ning mudeli tõenäosuste kalibreerimine. Kuhjamisega loodi kõige täpsem mudel. See mudel põhines mudelitel lineaarne SVC isotoonilise kalibreerimisega, SVC koondamisega ja SGD isotoonilise kalibreerimisega (F_1-skoor 0,902). Samas oli 20 täpseima mudeli F_1-skooride vahe 0,02 ühikut. See näitab, et erinevad mudelid on sarnase täpsusega. Süstemaatiliselt mõnevõrra madalamaid tulemusi andis Bernoulli multivariatiivne Naïve Bayes. Viimane on põhjendatav, sellega, et mudel ei võta aluseks sõnasagedusi dokumendis vaid kas konkreetne sõna esineb dokumendis või mitte.

Loodud raamistikku on võimalik kasutada Maanteeameti teiste või mõne uue asutuse e-kirjade analüüsimiseks ning klassifitseerimiseks. Saavutatavat tulemust on keeruline hinnata, kuna lisaks rakendatavatele mudelitele sõltub mudelite headus treenimise aluseks olevatest dokumentidest. Lisaks saab raamistikku kasutada muude organisatsioonide tekstidest teemade tuvastamiseks ja selle põhjal klassifitseerimismudelite loomiseks. Raamistiku arendamiseks võiks analüüsida, kuidas muuta automaatsemaks andmete anonümiseerimine, märgendamine ja puhastamine.

Bibliograafia

Aggarwal, C. C., & Zhai, C. (2012). *Mining Text Data.* New York Dordrecht Heidelberg London: Springer.

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research* (13), 281-305.

Berry, M. W., & Kogan, J. (2010). *Text Mining: Applications and Theory.* Wiley.

Blei, D. M. (2012). Probabilistic topic models. *Communication of the ACM, 4*, 77-84.

Blei, D. M. (2012). Topic Modeling and Digital Humanities. *Journal of Digital Humanities, 2*(1).

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research, 3*, 993-1022.

Bojanowski, P., Grave, G., Joulin, A., & Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics, 5*, 135-146.

Burrell, J. (2016). How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society*, 1-12.

Chang, C.-C., & Lin, C.-J. (2002). Training ν-Support Vector Regression: Theory and Algorithms. *Neural Computation, 14*(8), 1959 - 1977.

Chang, J., Boyd-Graber, J., Gerrish, S., Wang, C., & Blei, D. M. (2009). Reading Tea Leaves: How Humans Interpret Topic Models. *Advances in Neural Information Processing Systems 22 (NIPS 2009), 31*, 1-9.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*(16), 321-357.

Cohen, I., & Goldszmidt, M. (2004). *Properties and benefits of calibrated classifers.* Allikas: Semantic Scholar: https://pdfs.semanticscholar.org/acf9/bcef4d3b436041d353c11b8c16cfe4e3087a.pdf

Feelingstream OÜ koduleht. (2017). Kasutamise kuupäev: 06. 01. 2018. a., allikas http://feelingstream.com/

Gareth, J., Daniela, W., Trevor, H., & Robert, T. (2014). *An Introduction to Statistical Learning.* Springer.

Hastie, T., Tibshirani, R., & Friedman, J. (2008). *The Elements of Statistical Learning.* Springer.

Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. rmt: *European Conference on Machine Learning* (lk 137-142). Berlin: Springer.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (09. 08. 2017. a.). Bag of Tricks for Efficient Text Classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics.* Valencia. Kasutamise kuupäev: 18. 05. 2018. a., allikas https://arxiv.org/abs/1607.01759v3

Kodasmaa, R. (2011). Infootsingus kasutatavad loomuliku keele töötluse tehnikad. Tartu.

Last, F., Douzas, G., & Baçao, F. (2017). Oversampling for Imbalanced Learning Based on K-Means and SMOTE. *CoRR*.

LeCun, Y., Bottou, L., B., O. G., & K.-R., M. (2012). Efficient BackProp. rmt: *Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science* (lk 9-48). Berlin, Heidelberg: Springer.

Liin, K., Muischnek, K., & Müürisep, K. (2012). *Eesti keel digiajastul = The Estonian language in the digital age.* Springer.

Maanteeamet. (2018). E-kirjade järjekorrad. Tallinn.
Maanteeamet. (2018). *Maanteeameti koduleheküIg*. Kasutamise kuupäev: 09. 11. 2017. a., allikas https://www.mnt.ee/et
Maanteeamet. (2018). Ülevaade klienditeenindusest. Tallinn.
Mayr, A., Binder, H., Gefeller, O., & Schmid, M. (2014). The Evolution of Boosting Algorithms - From Machine Learning to Statistical Modelling. *Methods Inf Med, 53*(6), 419-427.
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*, 1-12.
Mikolov, T., Yih, W.-T., & Zweig, G. (2013). Linguistic Regularities in Continuous Space Word Representations. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (lk 746-751). Atlanta: Association for Computational Linguistics.
Niculescu-Mizil, A., & Caruana, R. (2005). Predicting good probabilities with supervised learning. *ICML '05 Proceedings of the 22nd international conference on Machine learning* (lk 625 - 632). Bonn: ACM.
Parambath, S. P., Usunier, N., & Grandvalet, Y. (2014). Optimizing F-Measures by Cost-Sensitive Classification. *Machine Learning, Proceedings of the Twentieth International Conference*.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011). Scikit-learn: Machine Learning in {P}ython. *Journal of Machine Learning Research, 12*, 2825-2830. Kasutamise kuupäev: 4. 4 2018. a., allikas http://scikit-learn.org/stable/
Pratama, B. Y., & Sarno, R. (2015). Personality Classification Based on Twitter Text Using Naive Bayes, KNN and SVM. *2015 International Conference on Data and Software Engineering (ICoDSE)* (lk 170-174). Yogyakarta: IEEE.
Rong, X. (15. 05. 2018. a.). *word2vec Parameter Learning Explained.* Allikas: https://arxiv.org/abs/1411.2738
Röder, M., Both, A., & Hinneburg, A. (2015). Exploring the Space of Topic Coherence Measures. *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining.* Shanghai.
Sammut, C., & Webb, G. I. (2017). *Encyclopedia of Machine Learning and Data Mining.* New York: Springer.
Stevens, K., Kegelmeyer, P., Andrzejewski, D., & Buttler, D. (2012). Exploring Topic Coherence over many models and many topics. *EMNLP-CoNLL '12 Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (lk 952-961). Jeju: Association for Computational Linguistics.
Zadrozny, B., & Elkan, C. (2002). Estimates, Transforming Classifier Scores into Accurate Multiclass Probability. *KDD '02 Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining.* Edmonton.
Zhai, C., & Massung, S. (2016). *Text Data Management and Analysis.* Morgan & Claypool Publishers.
Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms.* Chapman & Hall / CRC Press.
Zolotov, V., & Kung, D. (12. 02. 2017. a.). *Analysis and Optimization of fastText Linear Text Classifier.* Kasutamise kuupäev: 12. 01. 2018. a., allikas https://arxiv.org/abs/1702.05531

Tamm, A. (2012). Optionality: Social Cognitive Factors in Changing Linguistic Complexity in the Dialects of Estonia. *LEA - Lingue e letterature d'Oriente e d'Occidente, 1*(1), 151-162.

Tong, S., & Koller, D. (2001). Support Vector Machine Active Learning with Applications to Text Classification. *Journal of Machine Learning Research, 2*, 45-66.

Tripathy, A., Agrawal, A., & Rath, S. K. (2015). Classification of Sentimental Reviews Using Machine Learning Techniques. *Procedia Computer Science* (lk 821 – 829). Ghaziabad: Elsevier.

Weiss, S. M., Indurkhya, N., & Zhang, T. (2015). *Fundamentals of Predictive Text Mining.* London: Springer.

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2005). *Data Mining, Practical Machine Learning Tools and Techniques.* Cambridge: Elsevier Inc.

Lisad

I. Intervjuu Maanteeameti klienditeenindusjuhi Tatjana Portnovaga 23.02.2018

Kui palju e-kirju aastas Maanteeametisse saadetakse?

Aastas saadavad Maanteeameti kliendid umbes 100 000 e-kirja. Aastate lõikes kasvab saadetavate e-kirjade arv 8%. Telefonikõnede arv väheneb selle võrra. Üheks põhjuseks võib olla uus põlvkond noorukeid, kes pigem suhtlevad kirjalikult kui telefoni teel.

Kuidas jagunevad Maanteeametisse saadetud e-kirjad?

Analüüsiks antud e-kirjad on osa Maanteeametile tulnud e-kirjadest. Analüüsitavaid e-kirju võib jagada kliendiinfokirjadeks ning teenindusbüroodesse saadetud e-kirjadeks. Täpsem jaotus on toodud lisamaterjalis „Ülevaade klienditeenindusest“.

Millisteks klassideks e-kirjade jaotamisest on Maanteeamet huvitatud?

Kõige rohkem pakuvad huvi registritoimingute ja dokumentidega seonduv, eksamitega seonduv, sõidukite tehnilise poolega seonduvad klassid. Eraldada võiks ka klassi, kus asutused saadavad meile dokumente (näiteks kohtud). Lisaks võimalusel ka e-teenindusega seotud probleemide klass. Lisaks klassifitseerimisele oleme huvitatud ka märksõnadest, mida kliendi kasutavad nendest teemadest rääkimisel. Selle abil saaksime kodulehte täiendada muutes sõnakasutust kliendikesksemaks.

Miks on Maanteeametil vaja e-kirju klassifitseerida?

See võimaldaks teenindajatel tegeleda rohkem väärtust loova tööga. E-kirjade käsitsi jaotamine on rutiinne töö, mida suudab üks ametnik järjest teha umbes pool päeva. Selle automatiseerimine aitab oluliselt ressurssi kokku hoida. Hinnanguliselt võib kokku hoida ühe inimese tööaja.

II. Klassifitseerimise eksperimendi F1-skoorid

	mediaan F1-skoor	F1-skoori standardhälve	hindamise tulemuse F1-skoor
SVC Adaboost	0,183	0,000	0,182
nuSVC	0,461	0,017	0,596
nuSVC Adaboost	0,755	0,027	0,734
Bernoulli multivariatiivne NB SMOTE	0,768	0,022	0,769
Bernoulli multivariatiivne NB alaesindamine	0,771	0,023	0,769
Bernoulli multivariatiivne NB üleesindamine	0,786	0,020	0,777
Bernoulli multivariatiivne NB koondamine	0,787	0,018	0,780
Bernoulli multivariatiivne NB	0,788	0,017	0,781
Bernoulli multivariatiivne NB sigmoid kalibreerimine	0,800	0,023	0,791
nuSVC SMOTE	0,830	0,022	0,807
Bernoulli multivariatiivne NB Adaboost	0,789	0,030	0,814
nuSVC üleesindamine	0,833	0,022	0,815
nuSVC alaesindamine	0,833	0,020	0,819
SVC	0,837	0,010	0,828
Multivariatiivne NB SMOTE	0,839	0,015	0,829
Multivariatiivne NB üleesindamine	0,841	0,015	0,832
Multivariatiivne NB alaesindamine	0,833	0,019	0,836
SVC alaesindamine	0,836	0,022	0,839
Bernoulli multivariatiivne NB isotooniline kalibreerimine	0,846	0,018	0,846
Multivariatiivne NB koondamine	0,853	0,016	0,847
SGD	0,839	0,013	0,851
Multivariatiivne NB	0,853	0,017	0,852
SGD Adaboost	0,872	0,013	0,855
Multivariatiivne NB sigmoid kalibreerimine	0,864	0,015	0,856
Multivariatiivne NB isotooniline kalibreerimine	0,866	0,015	0,860
linearSVC alaesindamine	0,877	0,011	0,867
Multivariatiivne NB Adaboost	0,861	0,014	0,871
SGD alaesindamine	0,855	0,013	0,873
nuSVC sigmoid kalibreerimine	0,883	0,010	0,875
nuSVC isotooniline kalibreerimine	0,882	0,011	0,879
kuhjamine FT, SGD koondamine (enamushääletus)	0,885	0,012	0,881
nuSVC koondamine	0,873	0,011	0,882
kuhjamine Multivariatiivne NB, SVC (enamushääletus)	0,893	0,014	0,883
linearSVC üleesindamine	0,897	0,012	0,884
fasttext	0,891	0,010	0,885
SVC isotooniline kalibreerimine	0,893	0,014	0,888
SVC üleesindamine	0,892	0,015	0,888
SVC koondamine	0,890	0,014	0,889
SGD SMOTE	0,890	0,013	0,890
SGD isotooniline kalibreerimine	0,890	0,015	0,890
SVC SMOTE	0,890	0,014	0,891
SVC sigmoid kalibreerimine	0,893	0,013	0,891

kuhjamine FT, Multivariatiivne NB, SGD koondamine (enamushääletus)	0,887	0,011	0,892
SGD üleesindamine	0,888	0,010	0,893
kuhjamine FT, Multivariatiivne NB, SGD koondamine (kaalutud hääletamine)	0,897	0,013	0,894
SGD koondamine	0,897	0,010	0,894
linearSVC SMOTE	0,893	0,016	0,894
kuhjamine FT, SGD koondamine (kaalutud hääletamine)	0,895	0,011	0,894
linearSVC isotooniline kalibreerimine	0,901	0,012	0,895
kuhjamine SVC sigmoid, linearSVC isotooniline, FT (kaalutud hääletamine)	0,894	0,013	0,895
kuhjamine linearSVC, SVC, SGD, SMOTE („pehme“ hääletamine)	0,899	0,013	0,895
kuhjamine FT, SGD koondamine, SVC koondamine (kaalutud hääletamine)	0,898	0,013	0,897
linearSVC koondamine	0,893	0,014	0,897
linearSVC Adaboost	0,893	0,014	0,897
SGD sigmoid kalibreerimine	0,893	0,010	0,897
kuhjamine FT, Multivariatiivne NB kuhjamine, SGD koondamine (kaalutud hääletamine)	0,894	0,014	0,897
linearSVC	0,897	0,011	0,898
linearSVC sigmoid kalibreerimine	0,897	0,011	0,898
kuhjamine SMOTE, linearSVC adaboost isotoonline kuhjamine („pehme“ hääletamine)	0,900	0,012	0,900
kuhjamine FT, SGD koondamine, linearSVC (enamushääletus)	0,895	0,013	0,901
kuhjamine linearSVC isotooniline, SVC koondamine, SGD isotooniline ((„pehme“ hääletamine)	0,894	0,013	0,902

III. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Risto Hinno**,
(*autori nimi*)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose **E-kirjade klassifitseerimine masinõppe abil Maanteeameti näitel**,

   mille juhendaja on Kairit Sirts,

   1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **9.08.2018**