Tartu Ülikool
Loodus- ja täppisteaduste valdkond
Ökoloogia ja maateaduste instituut
Geograafia osakond

Magistritöö geoinformaatikas ja kartograafias (maht 30 EAP)

Skriptimisvahendid teksti lokaliseerimiseks ja asukohapõhiseks analüüsiks Eesti asulate meediakuvandi tuvastamise näitel

Ott Koik

Juhendajad: PhD Alexander Kmoch

PhD Raivo Aunap

Kaitsmisele lubatud:

Juhendajad:

Osakonna juhataja:

Tartu 2018

Skriptimisvahendid teksti lokaliseerimiseks ja asukohapõhiseks analüüsiks Eesti asulate meediakuvandi tuvastamise näitel

Lühikokkuvõte:

Magistritöö eesmärgiks on, kasutades programmeerimiskeelt *Python*, siduda näidisandmetena kasutatavad, aastatel 2012-2016 portaalis Delfi.ee avaldatud uudisartiklid neis kajastatud asulatega ja lokaliseerimisele järgneva analüüsi ning visualiseerimine käigus tuvastada artiklite põhjal meedias loodud Eesti asulate kuvand. Näidisandmestikust asukohtade tuvastamise järgselt viidi läbi kolm erineva eesmärgiga analüüsi: esiteks leiti TF*IDF meetodiga iga asula kohta meediakajastusest olulisemad sõnad. Teiseks kasutati k-keskmise klasteranalüüsi andmete klassifitseerimiseks, leidmaks kohad, kus domineerivad sarnase temaatikaga artiklid. Kolmandaks teostati meelsusanalüüs meediakajastuse meelestatuse leidmiseks, kasutades selleks Eesti Keele Instituudi poolt välja töötatud töövahendil põhinevat modifitseeritud programmi.

Märksõnad: Loomuliku keele töötlus, veebimeedia, tekstikaeve, asukohatuvastus, Python, meelsusanalüüs

CERCS:

P510 Füüsiline geograafia, geomorfoloogia, mullateadus, kartograafia, klimatoloogia

P175 Informaatika, süsteemiteooria

Scripting Tools for Text Localization and Location Base Analysis of Text Data by Example of Identifying Estonian Settlements Image of Media.

Abstract:

The aim of this thesis was to link automatically large data set of texts to the settlements mentioned in it and to carry out location based analysis to identify Estonian settlements image of media. In order to complete these tasks, a variety of programs, which were based on several mathematical algorithms, were created using the Python programming language. After location recognition from the example data, including news articles published online in news portal Delfi.ee in the time period of 2012-2016, three distinct automated analysis methods were used. Firstly, the TF*IDF method was used to ascertain the most important words from media coverage for each settlement. Secondly, k-means clustering algorithm was used for automated classification process to find places, where similar content was dominating. Thirdly, sentiment analysis, based on the modified program created by The Institute of the Estonian Language, was carried out to find a sentiment of articles for each settlement.

Keywords: Natural language processing, online media, text mining, location recognition, Python**,** sentiment analysis

CERCS:

P510 Physical geography, geomorphology, pedology, cartography, climatology

P175 Informatics, systems theory

Sisukord

Sissejuhatus

Üha suureneva andmetulva taustal on skriptimine ehk selliste tehnoloogiate loomine, mis võimaldavad automaatselt või poolautomaatselt informatsiooni matemaatiliste algoritmide rakendamise teel kontekstist lähtuvalt töödelda, andmeteadustes ja seda toetavates distsipliinides omandamas üha olulisemat rolli (Kadiyala, Kumar, 2017). Sellest tulenevalt on masinõpe arvutiteadustes üha kiireneva arenguga uurimisvaldkond, mis loob ühenduse sotsiaal- ja reaalteaduste vahel, võimaldades matemaatiliste mudelite alusel uurida automaatselt ja suures koguses ka selliseid andmestikke, mille varasem analüüs olnuks liigselt aja- ja ressursimahukas. Üheks selliseks valdkonnaks on tekstiandmete automaatne töötlemine, mis on arvutiteadustes olnud väljakutseks juba aastakümneid, kuid on nüüdseks tehnoloogilise arengu kaasabil kättesaadav ka tavakasutajatele (Hutchins, 2004).

Tekstiandmed esinevad mitmetes erinevates vormides. Üheks selliseks liigiks on uudised, mis pärast veebimeedia võidukäiku 21. sajandi algul, on oma mõõtmetelt ning tootmise pea katkematu vooga omandamas suurandmete mastaape. Kõige selle taustal kasvab kiiresti ka inimeste meediatarbimine ja seega omab uudistes kajastatud informatsioon olulist rolli asukohakuvandi kujunemisel (Kramp, Loosen, 2018). Geograafidele, sealhulgas eriti geoinformaatikutele, on tekstiandmete töötlus võrdlemisi võõras. Isegi ajastul, mille märksõnaks on „*suurandmed*“ ja geograafia suurimaks väljakutseks multidistsiplinaarsed rakendused ning andmete sidumine nende sisuga, on tekstianalüüs levinud enamasti ainult sotsiaalmeediaga seotud temaatikas (Tsou, 2015). Asukohakuvand ja sellest tulenev mõju rändele ning turismile on geograafilistes uuringutes olulisel kohal ning automaatne tekstianalüüs omab suurt potentsiaali selle uurimistemaatika arendamisel.

Sarnaselt globaalsetele tendentsidele, pole ka Tartu Ülikooli geograafia osakonnas tekstidega tegelevaid geoinformaatilisi töid kuigi palju. Tekstide automaatse lokaliseerimisega on koostöös Uus-Meremaa Geoloogia ja Tuumateaduste Instituudi ning Salzburgi Ülikooli teadlastega tegelenud Alexander Kmoch ja Evelyn Uuemaa. 2018. aastal avaldasid nad artikli *„Enhancing Location-Related Hydrogeological Knowledge“*, mis kirjeldab teadusartiklite geokodeerimist Uus-Meremaa näitel. Bakalaureusetöös on teksti lokaliseerimist ja analüüsi kasutanud Laura Kabonen (2016), kelle koostatud töös *„Subjektiivse heaolu kaardistamine Eestis“* kasutati sotsiaalmeedias avaldatud tekste semantiliseks analüüsiks. Suurt mõju eestikeelse teksti automaatse töötluse arendamisel omavad Tartu Ülikooli eesti keele filoloogia taustaga teadlaste Hille Pajupuu ja Rene

Altrovi tööd sõnade automaatsel meelsuse määramisel ning suuresti Tartu Ülikooli informaatika ja arvutilingvistika taustaga teadlaste meeskonna tööd EstNLTK nimelise, eestikeelse teksti automaatse töötluse töövahendite mooduli loomisel.

Ajendatuna väljatoodud potentsiaalist tekstandmestikul põhineva masinõppe rakendamisel geoinformaatikas, on magistritöö eesmärgiks, kasutades programmeerimiskeelt *Python*, siduda näidisandmetena kasutatavad, aastatel 2012-2016 portaalis Delfi.ee avaldatud uudisartiklid neis mainitud asulatega ja lokaliseerimisele järgneva analüüsi ning visualiseerimise käigus tuvastada artiklite põhjal meedias loodud Eesti asulate kuvand. Kuna tegu ühega on esimestest geograafilistest skriptmisalastest töödest eestikeelse tekstitöötluse vallas, keskenduti eelkõige metodoloogilisele problemaatikale ja sellega seoses püstitati järgnevad uurimusküsimused:

- Kuidas on võimalik automaatseid tekstitöötluse töövahendeid rakendada artiklite lokaliseerimiseks?
- Kui suure täpsusega väljatöötatud algoritmid asukohtasid tuvastavad ja milles seisnevad suurimad probleemid?
- Millised automaatsed meetodid on rakendatavad asukohakuvandi väljaselgitamisel ja millisel määral on need meetodid omavahel ühildatavad?

Magistritöö on eesmärgi saavutamiseks jaotatud kolmeks peatükiks. Esimeses peatükis tutvustatakse automaatse tekstitöötluse ja selle erinevate meetodite teoreetilist tausta ning veebimeedia rolli asukohakuvandi kujunemisel. Teises peatükis kirjeldatakse näidisandmestikku ja metoodikat ning kolmandas esitatakse saavutatud tulemused koos meetodite analüüsi ning võrdlusega.

1. Teoreetilised lähtekohad

1.1 Loomuliku keele töötlus

Kõne ja selle edastamise süsteem ehk keel, kui inimese kõige loomulikumad informatsiooni vahetamise viisid, on endas sisalduva suure ja mitmekesise teabe hulga tõttu olnud andme- ja arvutiteadlaste huviorbiidis alates vastava tehnoloogilise võimekuse saavutamisest eelmise sajandi keskpaigas (Hutchins, 2004). Loomuliku keele automaatseks töötlemiseks kasutatakse, sõltuvalt andmete iseloomust ja soovitud tulemusest, mitmeid erinevaid meetodeid. Enamasti on nendeks traditsioonilisest andmetöötlusest tuntud tehnoloogiate ning teoreemide kohandused tekstandmete analüüsimiseks. Käesolevas peatükis tutvustatakse kõige levinumaid loomuliku keele töötlemise eesmärke ja nende saavutamiseks kasutatavaid meetodeid.

Loomulikuks keeleks (ingl *Natural language*) loetakse inimeste poolt omavaheliseks suhtluseks välja kujunenud ja erinevaid semantilisi struktuure sisaldavat märkide süsteemi. Loomuliku keele töötlemise (edaspidi LKT) põhiliste ülesannetena on andmeanalüüsiga tegelev teadlane Dipanjan Sarkar 2016. aastal avaldatud teoses „*Text Analytics with Python. A Practical Real-World Approach to Gaining Actionable Insights from your Data“* loetlenud kuus valdkonda:

- masintõlke arendamine,
- kõnetuvastussüsteemid,
- küsimustele automaatselt vastavad süsteemid,
- konteksti tuvastamine,
- automaatne tekstide kokkuvõtete loomine,
- tekstide kategoriseerimine.

Eelpool mainitud ülesannete lahendamiseks on loodud mitmeid tehnikad, mis masinõppest, lingvistikast ja statistikast pärit meetoditega teksti analüüsida aitavad. D. Sarkar toob oma teoses välja järgnevad tehnikad:

- tekstandmestiku klassifitseerimine,
- teksti klasteranalüüs,
- meelsusanalüüs,
- tekstiüksuste tuvastamine ja eraldamine,
- sarnasusanalüüs ja suhete modelleerimine.

Inimkeele uurimine tehnoloogiliste vahenditega on teadlastele huvi pakkunud juba mitmeid aastakümneid. Suurimaks läbimurdeks selles valdkonnas saab pidada 1954. aastal läbiviidud nn Georgetown'i eksperimenti. Selle käigus loodi mõned aastad varem Alan Turingu poolt avaldatud aluspõhimõtete eeskujul esimene automaattõlke rakendus, mis võimaldas tõlkida 60 venekeelset lauset inglise keelde (Hutchins, 2004). Kaks aastat hiljem kirjeldas N. Chomsky keeletöötlusega kaasnevaid lingvistilisi probleeme, samas kui inimeste vahelise suhtlemise meetodite statistilise analüüsi problemaatikat analüüsisid C. E. Shannon ja W. Weaver juba 1949. aastal. N. Chomsky töös loodi ka aluspõhi automaatse keeleuurimise ühe olulise nurgakivi - regulaaravaldise (ingl *Regular expression*) kasutamiseks grammatiliste reeglite järgimisel. Keeleliste eripärade masinõppele arusaadavaks tegemisele järgnes ka tänapäeval laialdaselt levinud tõenäosuslike meetodite kiire areng (Nadkarni, Ohno-Machado ja Chapman, 2011). Tõenäosuslike meetodite võidukäiguga kaasnes automaatse keeleanalüüsi fookuse nihkumine masintõlkest ja andmepäringutest semantilise suunitlusega sisu kokkuvõtete, automaatse klassifitseerimise ning meelsusanalüüsi läbiviimisele (Cambria, White, 2014).

LKT on tehnoloogiale siiani suur väljakutse. Sarnaselt mitmete teiste suurandmetega töötavate valdkondadega, on üheks olulisemaks LKT-e väljakutseks väga suurest andmehulgast enne analüüsi algust leida esmalt see, mis on uurimustöö kontekstis oluline (Kadiyala, Kumar, 2017). Selle probleemi olulisust suurendab asjaolu, et LKT on ka iseõppivaid programme kasutades väga aega ja ressurssi nõudev andmetöötluse suund. Tekstidokumentide valikut ja ettevalmistamist analüüsiks raskendab asjaolu, et tegu on enamasti struktuuriliselt väga varieeruva andmete liigiga ning seega on ühtse uurimisraamistiku rakendamine töötlemiseks väga keeruline (Sarkar, 2016). Ometi on just suur andmehulk oluliseks aspektiks tänapäevaste LKT-e töövahendite väljakujunemisel, moodustades nn sõnade koti (ingl *Bag of words*), mis võimaldab rakendada korpusepõhist lähenemist, kus suure andmehulga najal on võimalik üles ehitada masinõppe protsessi käigus sõnamustrite tuvastamiseks sobilik aluspõhi (Sabah, 2010).

Lisaks struktuurilisele diferentseerumisele, muudavad LKT-e keeruliseks erinevad lingvistilised eripärad, alustades erinevates keeltes varieeruva grammatikaga ja lõpetades peidetud semantiliste struktuuridega, mis on tihtipeale masinõppes äärmiselt raskesti tuvastatavad. Keelte eripäradest tulenevate suurimate väljakutsetena saab esile tuua sünonüümide ja homonüümide tuvastamise, sõnade tähenduse muutumise seoses käänete vaheldumisega ning sõnade järjekorrast lähtuva konteksti muutuse. Samuti raskendab tekstianalüüsi kirjavigade esinemine, mis muudab masinõppe

jaoks fraasi kasutuskõlbmatuks või halvemal juhul muudab selle sisu (Nadkarni, Ohno-Machado ja Chapman, 2011). Eelpool mainitud problemaatika tõttu on tekstandmete automaatsele analüüsimisele esitatud ka rohkelt kriitikat, heites nendele meetoditele ette valede seoste tekitamist ja vähest korrelatsiooni andmete ning nende tegeliku konteksti vahel, mis toob kaasa eemaldumise reaalse maailma kirjeldamisest ehk teaduse ühest esmasest eesmärgist (Shahin, 2016).

1.1.1 Eesti keele automaatne analüüs

Sarnaselt üldisele automaatsele lingvistilisele analüüsile, initsieeris ka eestikeelse teksti töötlemise meetodeid masintõlke arendamine, millega alustati juba 1991. aastal. Esialgne tekstikorpus kattis ligi 20% sõnavormidest, võimaldades sellega ainult lihtsamate ülesannete täitmist (Kaalep, 1997). Eestikeelse teksti masintöötlemise muudab eriti keeruliseks asjaolu, et sarnaselt teistele soome-ugri keeltele, on eesti keel morfoloogiliselt mitmekesine – sõnadel on väga mitmeid erinevaid vorme ja lisaks esineb palju ebareeglipärasusi. Sõnamustrite analüüsi raskendab võrdlemisi vaba lausesisene sõnade järjekord (Tkachenko, Petmanson, Laur, 2010). Eelnevalt defineeritud tekstikorpuse kasvades loodi eesti keele töötlemiseks mitmeid erinevaid töövahendeid, mis ühendati ingliskeelsete moodulite *Textblob* ja NLTK eeskujul Tartu Ülikooli arvutiteaduse instituudi teadlaste poolt mitmetes erinevates programmeerimiskeeltes, sealhulgas *C* ja *C++*, kirjutatud programmidest töövahendite kogumikuks EstNLTK (ingl *Estonian Natural Language Toolkit*). Kogumik on kasutatav avatud lähtekoodiga *Pythoni* moodulina (Orasmaa, et al., 2016).

Eestikeelse teksti analüüsimise moodul EstNLTK, mis oli ühtlasi olulisel kohal magistritöö metoodika väljatöötamisel, võimaldab teostada järgnevaid loomuliku keele töötlemise põhiülesandeid (Orasmaa, et al., 2016):

- **Teksti segmenteerimine** (ingl *Tokenization*) – terviktekst jaotatakse mooduli autorite poolt meedias avaldatud artiklite põhjal moodustatud tekstikorpuse eeskujul väiksemateks üksusteks, milleks on lõigud, laused ja sõnad.
- **Morfoloogiline analüüs** – tuvastab sõnade algvormid, liited, käändelõpud ja grammatilist aega väljendavad sõnalõpud.
- **Morfoloogiline süntees ja grammatika kontroll** – leiab sõnade kohta kõik võimalikud sobivad morfeemid ja parandab kirjavead.
- **Nimeliste üksuste tuvastamine** (ingl *Named entity recognition*) – tuvastab tekstist isikute, organisatsioonide ja kohtade nimed. Sellele funktsionaalsusele toetub ka

asukohtade leidmine, mida kirjeldatakse lähemalt järgnevates peatükkides.

- **Ajaliste määratluste tuvastamine** (ingl *Temporal expression tagging*) – tuvastab tekstist fraasid, mis viitavad ajalisele täpsustusele, nagu näiteks kuu- ja nädalapäevale või ajamäärsõnale.
- **Osalausete tuvastamine** – segmenteerib algteksti osalauseteks, mille abil on võimalik pikkasid komplekssseid lauseid analüüsi läbiviimiseks lühendada.

EstNLTK töövahendid, eelkõige teksti morfoloogilisele töötlemisele keskenduvad alamprogrammid, tuginevad suuresti juba 1990ndate lõpus programmeerimiskeeles *C++* kirjutatud eesti keele komplekssele analüüsi programmile *Vabamorf*, mis sisaldab endas ligikaudu 98% eestikeelsetest sõnadest ja nende vormidest. Olenemata eesti keele äärmiselt keerukast struktuurist, saavutati väga kõrgest sõnavaralisest esinduslikkusest tulenevalt *Vabamorf'i* loomisel läbiviidud testidel morfoloogilisel analüüsil vähemalt 97% täpsus. Viga vähendati veelgi esmalt tuvastamata sõnadele läbiviidud sisust lähtuva oletusliku analüüsiga. Selle käigus vähendati viga alla 1% kõigist uuritud sõnadest, luues sellega statistiliselt usaldusväärse põhja edasise metoodika ülesehitamiseks (Kaalep, Vaino, 2001). Lisaks eelnevalt väljatoodud funktsionaalsustele, võimaldab EstNLTK mitmete erinevate alamülesannete, nagu näiteks ahelverbide ja sünonüümide tuvastamine, lahendamist (Orasmaa, et al., 2016).

1.1.2 Tekstandmete automaatne lokaliseerimine

Geograafilisest aspektist on tekstiandmete töötluse üheks oluliseks ülesandeks andmete sidumine kindla asukohaga ehk teksti lokaliseerimine. Olenemata asjaolust, et erinevate tekstituvastusalgoritmide areng on kaasa toonud ka mitmete lokaliseerimiseks sobilike programmide esilekerkimise, kasutatakse tekstandmestikku geoinformaatilise analüüsi läbiviimiseks sisendandmetena võrdlemisi harva (Yuxia, 2011). Üheks levinud meetodiks töötlemata tekstidest asukohtade tuvastamisel on geograafilist sisu edastavate nimeliste üksuste leidmine. Selle meetodi rakendamiseks kasutatakse mitmeid erinevaid lähenemisi, nagu näiteks peidetud Markovi ja maksimaalse entroopia mudelid, närvivõrgustiku klassifikaatorid või tugivektor-masinad (Witmer, Kalita, 2009). Eesti keele töötlemiseks mõeldud töövahendite kogumikus EstNLTK kasutatakse nimeliste üksuste leidmiseks lineaarset tinglike juhuslikkude väljade meetodit (ingl *linear chain conditional random fields*). Lineaarne tinglike juhuslikkude väljade meetod on oma olemuslikult tõenäosuslik mudel, kus tinglik tõenäosus $p(\vec{y}|\vec{x})$, mis

väljendab nähtuse $\vec{x}$ kuulumist klassi $\vec{y}$, leitakse valemiga $p(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})}\prod_{j=1}^{n}\psi_j(\vec{y},\vec{x})$, kus $Z(\vec{x})$ tähistab normaliseerimisfunktsiooni ja $\psi_j(\vec{y},\vec{x})$ funktsioonide hulka, mis sisaldab klasside olekufunktsiooni, klassidevahelist üleminekufunktsiooni ja eelnevalt treenitud andmestiku põhjal leitud parameetreid (Lafferty, McCallum, Pereira, 2001; Tkachenko, 2010).

Lisaks nimeliste üksuste tuvastamisele, on levinud tekstist asukohtade tuvastamise meetodiks ka peidetud semantiliste struktuuride tuvastamine (ingl *latent semantic analysis* ehk LSA), kus eelnevalt defineeritud andmestiku põhjal leitakse uuritavast sõnade loendist huvipakkuvatele nähtustele omased sõnade esinemise mustrid (Yuxia, 2011). LSA üheks põhiliseks arengusuunaks tekstiandmete lokaliseerimisel on ontoloogiatel põhinevat mudelid. Ontoloogiaks nimetatakse andmekogumit, milles on vaadeldava valdkonna raames defineeritud lisaks nähtustele ja nende omadustele ka nähtuste omavahelised seosed (Gruber, 1993). Olgugi, et ontoloogiate loomise idee pärineb juba antiikajast ja nendel põhinev lähenemine on oma olemuselt semantiliseks tekstianalüüsiks sobivaim meetod (Agarwal, 2005), puudub töö koostamise hetkel vastav eestikeelne geograafilist informatsiooni sisaldav andmekogu ning seetõttu käesoleva magistritöö raames ontoloogiatel põhinevat lähenemist ei kasutata.

Sarnaselt üldisele loomuliku keele töötlusele, tuleb ka automaatse asukohatuvastuse puhul arvestada mitmete keeleliste eripäradega, mis muudavad geograafilise informatsiooni tuvastamise keerulisemaks ja vigaderohkemaks. Esmase ülesande, milleks on geograafilise konteksti leidmine, lahendusi tutvustati eelmistes lõikudes, kuid see ei ole enamasti täpseks teksti lokaliseerimiseks piisav. Suurimateks probleemideks kirjalike andmete sidumisel asukohaga on tähenduslike, isikunimedega identsete ja korduvate toponüümide esinemine (Jones, Purves, 2008; Kmoch, et al., 2018). Arvestades väljatoodud probleemidega, on oluline esmased tuvastatud asukohad kontekstipõhiselt valideerida, lähtudes meetoditest, mis kasutavad otsuseid imiteerides sarnast loogikat, mida kasutavad inimesed vastavat teksti lugedes (Jones, Purves, 2008). Olgugi, et arvutit inimese kombel mõtlema panna on väga keeruline ja ühtlasi vajavad sellised programmid ka äärmiselt suurt arvutusvõimekust, on esmane valideerimine võimalik läbi viia järgides lihtsaid põhimõtteid:

- Korduvate ja tähenduslike nimede puhul vaadeldakse esmalt geograafilist konteksti – kas samas tekstis leidub viiteid teistele, uuritava kohaga seotud asulatele, omavalitsustele või maakondadele.

- Konteksti puudumisel võetakse otsus vastu lähtuvalt hierarhilisest süsteemist, mille kohaselt eelistatakse näiteks rahvaarvule, asustusüksuse tüübile või muule tunnusele tuginedes konkreetset kohta.

Kahjuks ei eemalda väljatoodud lahendused isikunimedega seotud probleemi.

1.1.3 Tesktandmestikust tunnuste leidmine

Kuna tekstiandmete töötlemine on äärmiselt andmemahukas protsess, eelneb sellele alati dokumenti kõige paremini iseloomustavate terminite ehk tunnuste selekteerimine koguandmestikust. Tunnuste leidmise protsessi meetodid saab jagada oma põhimõttelt kaheks (Aggarwal, Zhai, 2012):

- **Sagedusel** põhinevad ehk nn sõnade koti meetodid, kus igat analüüsitavat teksti vaadeldakse kui teatud sõnade kogumikku koos nende sõnade esinemissagedusega koguandmestikus.
- **Sõnade järjestusel** põhinevad meetodid, kus võrreldakse sõnade järjestuse erinevust tekstides.

Eelnevalt väljatoodud kaks põhilist meetodi klassi jaotuvad veel omakorda suureks arvuks tööpõhimõttelt ja arvutuslikult keerukuselt diferentseeruvateks algoritmideks, millest enim on tekstiandmete töötlemisel kasutust leidnud järgnevad metoodikad (Liu, et al., 2003):

- **Teabe kasvu indeks** (ingl *Information Gain*) – mõõdab informatsiooni hulga muutust termini esinemise või puudumise korral.
- χ^2 **statistik** – mõõdab termini ja dokumendi või kategooria omavahelise seose tugevust.
- **Dokumendi sagedus** (ingl *Document Frequency*) – mõõdab termini esinemise sagedust andmestikus.
- **Termini tugevuse indeks** (ingl *Term Strength*) – mõõdab tõenäosust, et termin, mis esineb ühes dokumendis, esineb ka teises.
- **Termini panus** (ingl *Term Contribution*) – hindab termini mõju dokumentide sarnasusele;
- **Entroopial põhinev klassifitseerimine** (ingl *Entropy-based Ranking*) – mõõdab entroopia muutu, mis kaasnev termini eemaldamisega tekstist.

Kuna teksti andmete töötlemine on aja- ja ressursikulukas protsess, on arvutusliku ja põhimõttelise lihtsuse tõttu suurte andmehulkade puhul enim levinud variandiks just sõnade esinemissagedust

arvestavad meetodid (Aggarwal, Zhai, 2012). Kuid millise sagedusega sõnad sisaldavad suurimat hulka teksti iseloomustavat informatsiooni? Esimese eeldusena võib arvata, et olulised on need sõnad, mis esinevad dokumendis kõige sagedamini või vastupidi, andmestikus harvaesinevad terminid omavad suuremat tähtsust. Tõde peitub eelpool toodud eelduste vahel, olles lähemal pigem eelduse teisele poolele (Leskovec, Rajaraman ja Ullman, 2014). Kõige enam esinevad tekstides side- ja asesõnad, mis endas olulist semantilist tähendust ei sisalda. Selliseid sõnu kutsutakse peatussõnadeks (ingl *Stop words*), mis enne analüüsi alustamist tekstist eemaldatakse. Kuigi olulised sõnad on pigem tekstis haruldasemad, tasub silmas pidada, et mitte kõik haruldased sõnad ei ole olulised – tihtipeale on tegu mõne väga spetsiifilise termini või väljendiga, mis analüüsi osas suurt väärtust ei oma või moonutab sootuks selle tulemust. Olulisuse annab sõnale omadus paikneda vähestes dokumentides, aga nendes, kus see esineb, teha seda sagedasti (Leskovec, Rajaraman ja Ullman, 2014).

Eelpool kirjeldatud olulisuse määra arvutamise matemaatiliseks algoritmiks on TF*IDF mudel. Lühend TF*IDF tuleneb ingliskeelsest terminist *term frequency * inverse document frequency* ehk termini tihedus * pöördvõrdeline dokumendi tihedus. Nagu viitab mudeli nimi, milles avaldub ka eelmise lõigu lõpus esitatud sõnade olulisuse definitsioon, koosneb TF*IDF kahest komponendist:

- **termini tihedus *tf***, mis näitab ühe sõna esinemise hulka ühe dokumendi raames ja avaldub matemaatiliselt järgnevalt: $tf(w, D) = f_{w_D}$, kus f_{w_D} sümboliseerib sõna *w* esinemise arvu dokumendis*;*
- **pöördvõrdeline dokumendi tihedus *idf***, mis seisneb kogu korpuses esinevate dokumentide arvu ja dokumentide arvu, kus esineb vaadeldav sõna, logaritmiliselt teisendatud väärtuse suhtes, millele on liidetud arv 1, vältimaks olukorda, kus väga madala *idf* väärtusega termineid eiratakse analüüsis täielikult. Matemaatiliselt avaldub *idf* valemis: $idf(t) = 1 + log \frac{C}{1+df(t)}$, kus *idf(t)* väljendab *idf* väärtust termini *t* kohta, väärtus *C* näitab kõikide dokumentide arvu tekstikorpuses ja *df(t)* dokumentide hulka, kus leidub termin *t*.

TF*IDF väärtus omistatakse seejärel sõnale valemiga *tfidf = tf * idf* (Jing, 2012). Lisaks tunnuste tuvastamisele, kasutatakse TF*IDF meetodit ka näidisandmestikust asulaid kõige paremini iseloomustavate sõnade leidmiseks.

1.1.4 Teksti klassifitseerimine masinõppe meetoditega

Andmetöötlusega tegelevate teadusharude üheks esmaseks eesmärgiks andmete analüüsil on nende klassifitseerimine mingite sarnasuste alustel ettemääratud või protsessi käigus loodavatesse gruppidesse. Klassifitseerimismeetodeid tuntakse andmeanalüüsis mitmeid ja need on suuresti kohaldatavad ka tekstide töötluseks. Ameerika Ühendriikide automaatse tekstitöötlusega tegelenud arvutiteadlased C. C. Aggarwal ja C. X. Zhai (2012) on välja toonud järgnevad viis tekstitöötluses kasutatavat meetodit:

- **Otsuste puu meetod** – andmete klassifitseerimiseks kasutatakse hierarhilist struktuuri, mis luuakse järk-järgulise analüüsi käigus andmete vaadeldavate tunnuste, milleks tekstianalüüsi käigus on enamasti etteantud sõnade olemasolu või puudumine dokumendis, alusel.
- **Mustri- ehk reeglipõhine meetod** – iga klassi puhul luuakse eelnevalt tunnuste esinemise muster, mis suure tõenäosusega näitab dokumendi kuuluvust sellesse klassi. Loodud mustreid kasutatakse reeglitena kogu andmehulga klassidesse jaotamiseks.
- **Tugivektor-masina klassifikaatorid** – andmete klassifitseerimiseks leitakse andmestikus esinevad lineaarsed piirid. Vaadeldava nähtuse kuuluvust mingisse klassi defineerib tema paiknemine piiride suhtes.
- **Närvivõrgustiku klassifikaatorid** – andmete klassifitseerimisel kasutatakse inimese närvisüsteemi analoogile põhinevat mudelit, kus iga vaadeldav nähtus moodustab sõlmpunkti, mille vahelistele seostele omistatakse kaalud seoses esinemissagedusega andmestikus. Masinõppeprotsessi käigus parandatakse analüüsi alguses määratud kaalusid pidevalt.
- **Bayens'i ehk generatiivsed klassifikaatorid** – luuakse tõenäosuslikud klassifikaatorid, mille abil hinnatakse andmete kuuluvust mingisse klassi selle omaduste põhjal.

Üheks enim levinumaks andmete klassifitseerimise masinõppe meetodiks on multinomiaalne Bayens'i klassifikaator (ingl *Multinomial Naïve Bayes*), mida kasutatakse enamasti prognoosimise ja klassifitseerimise töövahendites suurema hulga klasside puhul (Sarkar, 2016). See klassifitseerimismeetod põhineb Bayens'i teoreemil, mille põhjal arvutatakse, millise tõenäosusega asub dokument *d* mingis klassis *c* valemi $P(c|d) = \frac{P(c)P(d|c)}{P(d)} \propto P(c)P(d|c) \propto P(c)\prod_{i=1}^{n_d} P(w_i|c)^{tf_{id}}$ abil, kus *P(c|d)* on tingimuslik tõenäosus, et dokumendi *d* esinemise puhul

kuulub see klassi *c* ja *P(d|c)* tingimuslik tõenäosus, et klassi *c* esinemise puhul on seal dokument *d*. *P(c)* ja *P(d)* näitavad omakorda dokumentide ja klasside esinemise tõenäosust eraldi. Sümbol *w* tähistab üksikut sõna dokumendis, mis on osa klassifitseerimiseks loodud sõnastikust. Tähisega tf_{id} on märgitud sõna w_i esinemiste arv ja tähisega n_d unikaalsete sõnade arv dokumendis (Ko, 2017). Meetodi naiivsus seisneb eelduses, et iga vaadeldav nähtus, milleks tekstianalüüsi puhul on enamasti artikkel või muu teksti sisaldav dokument, on teistest sõltumatu. Multinomiaalse klassifikaatori puhul on vaadeldavad nähtused esitatud vektorina $d = \{x_1, x_2, \ldots, x_n\}$ (Sarkar, 2016).

Andmete klassifitseerimise alamliigina on levinud andmete klasterdamine, mille eesmärgiks on nähtuste, mida analüüsi käigus vaadeldakse eraldi punktidena, jaotamine klassidesse, lähtudes nende omavahelisest kaugusest tunnusruumis (Leskovec, Rajaraman ja Ullman, 2014). Sellise metoodika kasutamine eeldab, et tekstidokumentidele on võimalik, distributiivsest ehk tähenduslikku sarnasust mõõtvast semantikast lähtuvate tunnuste alusel, omistada ruumis kindlad koordinaadid. Selleks, et sõnaline sisu teisendada ruumiliselt vaadeldavaks punktiks, luuakse esmalt kõikide tunnuste koosesinemiste sagedusi sisaldav sagedusmaatriks ja selle alusel vastavad tunnusvektorid. Seejärel võetakse aluseks distributiivse semantika alushüpotees, mis väidab, et sõnad, mis esinevad sagemini sarnases kontekstis, omavad ka sarnast semantilist tähendust (Bruni, Tran, Baroni, 2014). Sõnade vektoriteks teisendamise järgselt leitakse nende omavaheline sarnasus, põhinedes eeldusele, et sarnasemate tähendustega tunnuste vektorid lähenevad. Asukoht ruumis sõltubki leitud tunnuste sarnasusest, millele numbrilise väärtuse andmiseks kasutatakse erinevaid vektorite kaugusi arvestavaid näitajaid, millest levinumateks on koosinus- ja eukleidiline kaugus (Aedmaa, 2016).

Andmete klasteranalüüsi meetodeid on mitmeid, millest Sarkar (2016) toob välja järgnevad:

- **Hierarhiline klasterdamine** – eeldab andmete klasterdamise põhieelduse, mille kohaselt paiknevad sarnasemad objektid teineteisele lähemal, paikapidavust. Andmed ühendatakse omavahel sõltuvalt kaugusest ja visualiseeritakse enamasti dendrogrammina.
- **Tsentroidipõhine mudel** – klastrid luuakse ümber keskse objekti, mis esindab iga klastri põhilisi tunnuseid ja seega on selle keskmine kaugus klassi kuuluvatest nähtustest võimalikult väike. Selle mudeli üheks levinumaks algoritmiks on k-keskmise klasteranalüüs (ingl *k-means clustering*, tuntud ka kui Lloyd'i algoritm), mis eeldab, et ette on antud klastrite arv *k*, mida arvesse võttes koondatakse andmed kauguse alusel

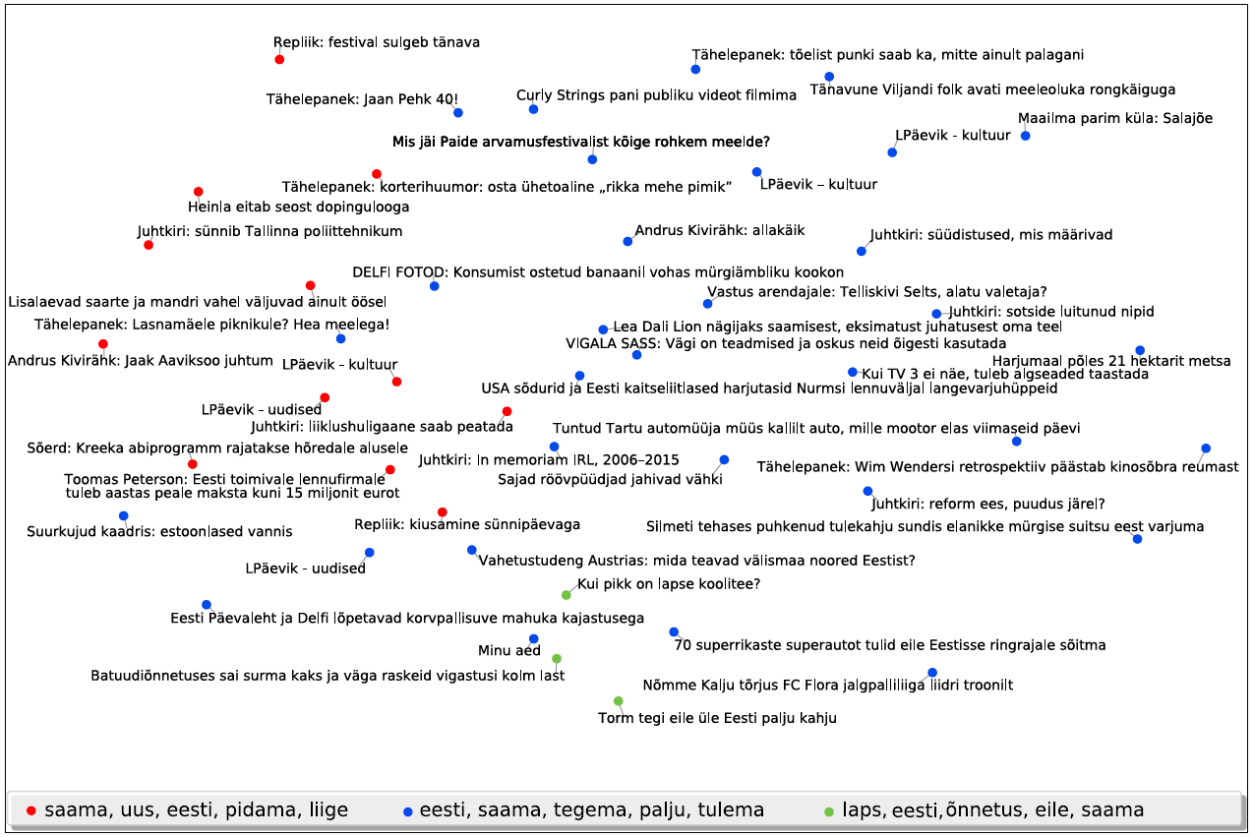
tsentroidide ümber.

- **Jaotusepõhine mudel** – tõenäosuslik algoritm, mille alusel leitakse nähtuste paiknemise jaotused ja moodustatakse klastrid, lähtudes eeldusest, et samasse klastrisse kuuluvad tunnused sobituvad sama jaotusega.
- **Tihedusepõhine mudel** – klastrid moodustatakse piirkondadesse, kus vaadeldavad nähtused on võrreldes ümbritseva nähtusruumiga tihedamini grupeerunud. Klastrite vahelist hõredamat piirkonda vaadeldakse mürana.

Üheks levinumaks tekstiandmete klasterdamise meetodiks saab pidada k-keskmise klasteranalüüsi, mis sobib oma arvutusliku lihtsuse tõttu just suuremate andmehulkade analüüsimiseks (Sarkar, 2016). K-keskmise klasteranalüüs koosneb kolmest etapist (MacKay, 2003):

1. **Initsialiseerimine** – tunnusruumi genereeritakse *k* juhuslikku tsentroidi.
2. **Jaotamine** – vaadeldavad nähtused paigutatakse tunnusruumi ja omistatakse lähima tsentroidi alusel klassiline kuuluvus.
3. **Uuendamine** – tunnusruumi paigutatud nähtuste paiknemise alusel leitakse uued tsentroidid, mille keskmine kaugus on klassi kuuluvatest nähtustest võimalikult väike.

Kolme etapi läbimise järgselt korratakse teist etappi ja tunnuste klassilist kuuluvust muudetakse seoses tsentroidide asukohtade muutumisega. Klasside muutumise korral arvutatakse järjekordselt uued tsentroidid ja kahte viimast etappi korratakse nii kaua, kuni ümberjaotamise käigus ühegi nähtuse klass ei muutu (MacKay, 2003). K-keskmise klasteranalüüsil on tekstiandmete ruumi paigutamise levinumaks meetodiks dokumentide tunnusvektorite koosinuskauguste, mida saab käsitleda ka kui sarnasuse indeksit, leidmine ja selle alusel uuritava dokumendi asukoha leidmine ruumis, mille mõõtmete arvuks on vaadeldavate dokumentide arv (Joonis 2). Tunnusvektorite leidmisel võetakse k-keskmise klasteranalüüsi puhul tunnuste koosesinemiste arvu asemel arvesse nende TF*IDF väärtusi (Sarkar, 2016). Andmete klasterdamist käsitletakse enamasti täisautomaatse analüüsimeetodina. Nagu ka teiste selliste meetodite puhul, tuleb siinkohal arvestada mõningaid ülekatteid loodud klasside vahel, mis muudavad klasside konkreetse defineerimise keeruliseks (Leskovec, Rajaraman ja Ullman, 2014).



Joonis 1. 50 portaalis Delfi.ee avaldatud artikli kolmeklassilise k-keskmise klasteranalüüsi tunnusruum artiklite pealkirjade ja klasside suurima TF*IDF väärtustega tunnustega

1.1.5 Meelsusanalüüs

Meelsusanalüüs (ingl *sentiment anaysis*) on LKT üks populaarseimaid rakendusi. Selle eesmärgiks on tekstandmestiku töötlemise abil teada saada, milliseid seisukohti ja hoiakuid andmestik endas sisaldab. Meelsusanalüüsi, kui võrdlemisi uue meetodi suurimaks tugevuseks loetakse selle võimet võtta arvesse emotsioone edasikandvat informatsiooni, nagu näiteks arvamused ja tunded (Kennedy, 2012). Seda meetodit kasutatakse tihti sotsiaalmeediaga seotud temaatika käsitlusel ning ärilistel eesmärkidel, näiteks tootearenduses, saamaks kiiresti teada klientide arvamuse tagasiside põhjal (Yi, et al., 2003). Käesoleva töö kontekstis on meelsusanalüüsi sisuks välja selgitada, milliseid kohti kajastatakse meedias pigem positiivselt ja milliseid negatiivselt. Sarnaselt teistele automaatsetele keeletöötluse vahenditele tuleb ka meelsusanalüüsi puhul arvestada meetodi kohatise ebatäpsusega – suure koguse erinevaid teemasid kajastavate tekstide puhul peetakse heaks tulemuseks 70-90% täpsust (H. Pajupuu, Altrov ja J. Pajupuu, 2016). Nimelt, arvesse tuleb järjekordselt võtta inimkeele ehituslikku ja semantilist keerukust, millega praeguseks hetkeks

eksimatult ükski arvutiprogramm veel tegeleda ei suuda. Sellisteks keerulisteks struktuurideks saab pidada näiteks huumorit, irooniat ja sarkasmi, kus reaalne teksti mõte võib arvuti poolt tuvastatule olla hoopis vastupidine (Kennedy, 2012).

Meelsusanalüüsi läbiviimiseks kasutatakse masinõppe või leksikonipõhist lähenemist (H. Pajupuu, Altrov ja J. Pajupuu, 2016), kus eristatakse teksti struktuurist lähtuvalt erinevaid klasse, mille tasandil analüüsi läbi viia. Bing Liu toob 2012. aastal avaldatud raamatus *Sentiment Analysis and Opinion Mining* välja kolm ülesehituslikku tasandit:

- **Dokumendi tase** – tuuakse välja kogu tekstidokumendi üldine meelestus.
- **Lause tase** – arvamus esitatakse iga lause kohta.
- **Üksuse tase** – tekstist leitakse arvamus iga tunnuse kohta.

Leksikonipõhine meelsusanalüüs eeldab eelnevalt koostatud sõnade emotsionaalset väärtust sisaldava andmekogu olemasolu. Leksikonis on defineeritud numbrilise väärtusega, millised sõnad kannavad endas positiivseid, millised negatiivseid väärtusi (Bing, 2012). Leksikoni defineerimisel tuleb arvestada, et manuaalse töö puhul on tegemist äärmiselt aeganõudva ülesandega, samas kui automaatseid meetodeid kasutades tekivad paratamatult vead. Sellest lähtuvalt pakub Bing oma teoses „*Sentiment Analysis and Opinion Mining“* parimate lahendustena poolautomaatseid lähenemisi:

- **Sõnaraamatu põhine lähenemine**: kasutaja defineerib hulga sõnade polaarsuse, millele järgneb automaatne analüüs. Selle käigus leitakse esmalt määratletud sõnade sünonüümid ja määratakse neile sama väärtus, mis algsele sõnale. Seejärel leitakse sõnade antonüümid, millele määratakse vastupidine väärtus. Selline meetod eeldab eelnevalt loodud sõnaraamatu olemasolu, kus on defineeritud ka sõnade sünonüümid ja antonüümid.
- **Korpusepõhine lähenemine**: kasutaja defineerib eelnevalt mingi hulga tekstide polaarsuse, mille järel leitakse tekstikorpusest sarnased struktuurid ning omistatakse neile sarnased väärtused.

Sõnade numbrilisi väärtusi saab seejärel kaaluda näiteks nende olulisusega ja seeläbi on võimalik saada uuritava struktuuriüksuse meelsust iseloomustav numbriline väärtus, kus negatiivsed numbrid väljendavad negatiivset avamust ja positiivsed positiivset, samas kui nullilähedased väärtused osutavad pigem neutraalsele tekstile (Sarkar, 2016). Kahjuks nii lihtsa lahendusega on võimalik küll hinnata üksikute sõnade emotsionaalset väärtust, kuid teksti semantilise sisu kohta

järelduste tegemiseks jääb see liiga primitiivseks: selline lahendus ei arvesta sõna meelsuse kontekstipõhist muutumist (Bing, 2012). Näiteks küsimus „*Kas sa oskad soovitada mõnda head Python'i käsiraamatut?*“ sisaldab positiivse tähendusega sõna „*head*“, kuid küsimus ise arvamust ei sisalda ja omab hoopis neutraalset sisu. Tulemuse reaalsusele vastavamaks muutmiseks kombineeritakse meelsust väljendavate väärtustega andmete klassifitseerimise meetodeid, mille abil jaotatakse uuritavad nähtused sarnasuse alusel klassidesse.

1.2 Veebimeedia ja selle roll asula kuvandi kujunemisel

Interneti laialdase levikuga tsiviilkasutuses, algas 20. sajandi viimasel kümnendil meediaväljaannete üha massilisem publitseerimine veebilehekülgedel, seisnedes algselt enamasti paberväljaannete kopeerimisel. Mõne aastaga kujunes veebimeediast juba omaette žanr, muutes uudiste väljaandmise kiirust, mahtu ja kasutatavate kujutlusviiside iseloomu täielikult. Esimesed meediaväljaanded avaldasid Eestis uudiseid internetis juba 1995. aastal, kuid *online*-meedia domineerimise algusajaks saab lugeda 21. sajandi esimesi aastaid (Saks, 2011) ja nüüdseks on veebimeediast saanud uudiste peamine kanal ning inimeste meediatarbimine on üha kasvavas trendis (Kramp, Loosen, 2018). Tehnoloogilise arengu kaasabil ja kiire ning laiaulatusliku informatsiooni leviku ning sellega kaasneva vahetu tagasiside tõttu internetis, on traditsiooniline meedia muutumas üha rohkem tarbija poolt kujundatavaks meediumiks, kus lugejate otsene arvamus omandab üha kasvavat rolli (Fürst, Schönhagen, Bosshart, 2015). Tarbija esilekerkimine on lisaks mitteprofessionaalse ajakirjanduse, sealhulgas blogide ja arvamusportaalide, olulisuse tõusule mõjutanud ka suurte meediakorporatsioonide uudiste avaldamise iseloomu. Eelkõige seisneb see pea katkematus uudiste avaldamise voos ja temaatilise ampluaa laienemises (Kasenõmm, 2014). Temaatiliselt katab veebimeedia kõik olulised eluaspektid, mille saab jaotada järgnevasse kaheksasse klassi: ettevõtlus, poliitika, kuritegevus, tervishoid, sport, meelelahutus, tehnoloogia ja teadus ning loodus (Bracewell, et al., 2009).

Tunnustatud Ameerika Ühendriikirde rekreatsiooni ja turismi teadlane John L. Crompton defineeris oma töös „*An Assessment of the Image of Mexico as a Vacation Destination and the Influence of Geographical Location Upon That Image*“ (1979) asukoha kuvandi kui mingi paiga osas tekkinud tõekspidamiste, ideede ja muljete kogumi. Kuvandi moodustavad arusaamad, hoiakud, teadmised, kogemused, ootused, soovid ja tunded, mida teatud asukoht indiviidis või inimeste grupis tekitab (Šantić, Bevanda, Bijakšić, 2016). Mitmed kuvandi aspektid kujunevad

inimestes läbi kaudsete allikate, mis ei eelda asukoha külastamist. Selliste tunnuste arengus mängib olulist rolli meediakajastus, sealhulgas ka uudisartiklites avaldatu (Muhoho-Minni, Lubbe, 2017). Asukohakuvand on äärmiselt oluline turismi arendamisel – 2007. aastal läbiviidud rahvusvahelises uurimuses „*Promoting Tourism Destination Image"* leidsid autorid, et meedias kajastatu, mis koosneb televisioonis, ajalehtedes ning veebimeedias avaldatust, omab turismi seisukohalt asukohakuvandi kujunemisel primaarset rolli (Govers, Go, Kumar, 2007). Turism on omakorda äärmiselt oluline majandusharu ja Eestis on nii välis- kui siseturismi osakaal majanduses ühtlaselt kasvavas trendis, moodustades sisemajanduse koguproduktist kaudseid mõjusid arvestades umbes 6% (Statistikaamet, 2017). Majandusliku olulisuse kõrval on turism üsnagi ebastabiilne ja meediakajastuse poolt tugevalt mõjutatav tegevusvaldkond – eriti tugevalt mõjutab turismi julgeolekut puudutava negatiivne kajastamine, mis vähendab turistide arvu ja nende puudumisest saamata jäävat tulu kiiresti ning suures mahus (Kaasik, 2014).

Automaatse tekstitöötluse meetoditeks asukohakuvandi leidmisel, on enim levinud meelsusanalüüsi kasutamine. Suurem osa selliseid uurimusi kasutavad lähteandmetena sotsiaalmeedias avaldatud postitusi, turismiagentuuride poolt pakutavaid reisikirjeldusi või reisimisega seotud foorumites avaldatud tagasisidet ja on suunatud eelkõige asukoha turundamise edendamisele. Näiteks analüüsis Souli Ülikooli teadlaste meeskond meelsusanalüüsi abil, milline oli Pariisi külastanud turistide meelestatus erinevate tarbitud teenuste ja külastatud paikade osas, kasutades lähteandmeteks portaalis *www.virtualtours.com* avaldatud tagasisidet. Uurimusest ilmnes, et külastajad olid kõige positiivsemalt meelestatud meelelahutuse ja kõige negatiivsemalt julgeoleku osas (Kim, et al., 2017). Reisisihtkohtade kuvandit on uurinud ka Ameerika Ühendriikides tegutsevad turismile orienteeritud teadlased S. Stepchenkova ja A. M. Morrison, kes analüüsisid aastal 2005 avaldatud töös Venemaa ning selle erinevate piirkondade kuvandit erinevate Ameerika Ühendriikide ning Venemaa reisiportaalide põhjal, kasutades selleks märksõnade esinemise sagedust. Paraku on automaatne meelsuse tuvastamine rahvusvahelises teaduskirjanduses keskendunud eelkõige üksikute sündmuste kajastamisele ja teenuste kvaliteedi uurimisele ning parandamisele ja geograafide poolt vähe kasutatud.

2. Andmed ja metoodika

Magistritöö andmete kogumiseks ja metoodika rakendamiseks kasutati programmeerimiskeele *Python* versiooni 3.4 ja interpretaatorit *Anaconda3.* Läbiviidud lokaliseerimise ja tekstianalüüsi järgselt leiti huvipakkuvad karakteristikud ning visualiseeriti tulemused, kasutades selleks programme ArcGIS 10.3 ja Adobe Illustrator CC 2017. Töövahendites rakendatud *Pythoni* moodulite versioonid ja töö läbiviimiseks kasutatud arvuti tehniline spetsifikatsioon on välja toodud lisas 1.

2.1 Lähteandmed

Tekstiandmete automaatse analüüsi läbiviimseks kasutati lähteandmetena AS Ekspress Meedia uudisteportaalis www.delfi.ee ajavahemikus 1. jaanuar 2012 kuni 31. detsember 2016 avaldatud artikleid. Kasutades uudisteportaali arhiivi (AS Ekspress Meedia, 2018), koguti informatsioon vaadeldava viieaastase perioodi jooksul avaldatud 385 819 artikli kohta. Artiklites sisalduv informatsioon oli talletatud veebilehel HTML koodi, millest loeti andmekogumisprogrammide abil artikli sisu, pealkiri, avaldamise kuupäev ja veebilink edasise analüüsi tarbeks andmeraamistikku.

Andmekogumisprogrammide loomiseks kasutati *Pythoni* moodulit *Scrapy*, mille abil loodi andmete kogumiseks töövahend, inglisekeelse üldnimetusega *Spider*, mis defineeritud reeglite alusel HTML koodist vajaliku informatsiooni arvuti vahemälusse loeb. Töö läbiviimiseks koguti esmalt vaadeldavas ajavahemikus avaldatud artiklite veebilingid programmiga, kus muutujaga *start_urls* oli defineeritud andmeid sisaldava veebilehe URL, muutujaga *next_page* viide järgmisele leheküljele ja muutujaga *LINK_SELECTOR* kogutavat informatsiooni sisaldav HTML'i lõik:

```
1.  import scrapy
2.  from scrapy.contrib.spiders import Rule
3.  from scrapy.linkextractors import LinkExtractor
4.
5.  class DelfiSpider(scrapy.Spider):
6.      name = "delfi_spider"
7.      start_urls =
    ['http://www.delfi.ee/archive/?tod=31.12.2016&fromd=01.01.2012&channel=0&category=0&fl2016=
    1&query=']
8.      rules = (Rule(LinkExtractor(allow=(), restrict_xpaths=('//a[@class="item item-
    next"]',)), callback="parse", follow= True),)
9.
10.     def parse(self, response):
11.         HL_SELECTOR = '.headline'
12.         for news in response.css(HL_SELECTOR):
```

```
            LINK_SELECTOR = 'h1 a ::attr(href)'
            yield {
                'link': news.css(LINK_SELECTOR).extract_first(),
            }
        next_page = response.xpath('.//a[@class="item item-next"]/@href').extract()
        if next_page:
            next_href = next_page[0]
            next_page_url = 'http://www.delfi.ee' + next_href.strip()
            request = scrapy.Request(url=next_page_url)
            yield request
```

Seejärel koguti sarnasel põhimõttel töötava programmi abil edasiseks analüüsiks vajalik andmestik:

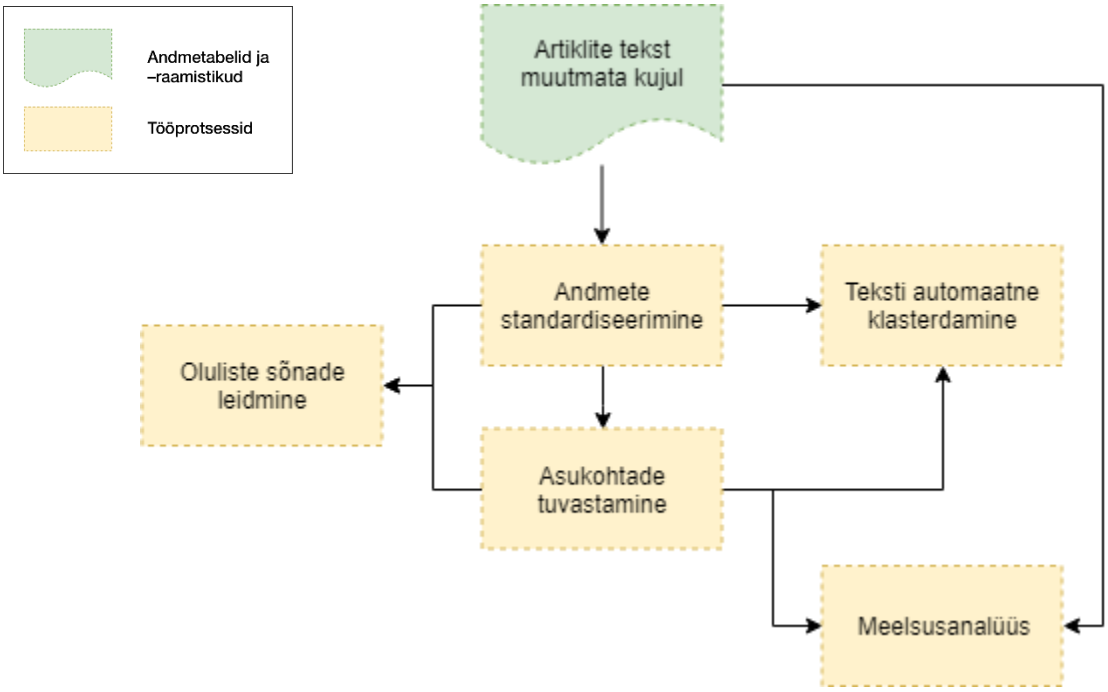
```
from scrapy.item import Item, Field
from scrapy.spider import Spider
from scrapy.selector import Selector
import csv
import time

class MyItem(Item):
    title = Field()
    date = Field()
    content = Field()
    link = Field()

class StockSpider(Spider):
        name = "delfi_sisu"
        allowed_domains = ["delfi.ee"]
        start_urls = []
        with open('link.csv', 'r') as f:
            reader = csv.reader(f)
            for line in reader:
                if line != []:
                    start_urls.append(line[0])

        def parse(self, response):
            time.sleep(1)
            def extract_with_css(query):
                return response.css(query).extract_first().strip()
            sel = Selector(response)
            item = MyItem()
            item ['content'] = ' '.join(sel.xpath("//p[@class='article__chunk
article__chunk--lead']/text()").extract()) + ' ' + '
'.join(sel.xpath("//p[@class='article__chunk article__chunk--lead']/a/text()").extract()) +
' ' + ' '.join(sel.xpath("//div[@class='article__body']/p/text()").extract())+ ' ' + '
'.join(sel.xpath("//div[@class='article__body']/p/a/text()").extract())
            item['title'] =
''.join(sel.xpath("//h1[@itemprop='headline']/text()").extract())
            item['date'] = extract_with_css('.article__date::text')
            item['link'] = str(response).split(' ')[-1].replace('>','')
            yield item
```

Lisaks analüüsitavatele artiklitele, kasutati töö koostamisel asukohtade tuvastamiseks Maa-ameti loodud Eesti haldusreformieelset 4715 asustusüksust sisaldavat andmestikku (Maa-amet, 2017) ja tähenduslike kohanimede defineerimiseks Eesti Keele Instituudi koostatud, üle 200 000

eestikeelset sõnavormi hõlmavat sõnaloendit (Eesti Keele Instituut, 2017).

2.2 Metoodika

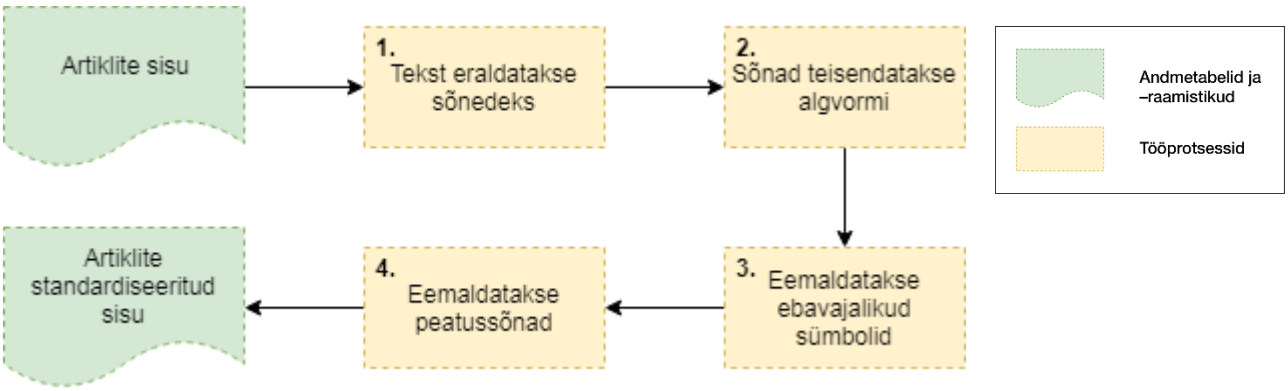
Artiklite automaatne töötlemine viidi läbi viies etapis, millest esimese kahe tulemused olid ühtlasi lähteandmeteks edasise analüüsi läbiviimisel (Joonis 2). Järgnevates alapeatükkides kirjeldatakse iga etapi teostamisel kasutatud programme ja nende rakendamisel saadud tulemusi.



Joonis 2. Tekstiandmete automaatse töötlemise etapid

2.2.1 Andmete standardiseerimine

Mitmed magistritöös kasutatavad meetodid, sealhulgas asukohtasid tuvastavad programmid, töötavad eeldusel, et sisendandmed on viidud standardsele kujule. Selleks tuleb eraldada tekst sõnadeks, eemaldada üleliigsed sümbolid ja sõnad ning viia allesjäänud sõnad algvormi. Andmete standardiseerimiseks töötati välja töövahendite kogumik, millest oli võimalik vastavalt analüüsi tüübile kasutada erinevaid funktsioone (joonis3).



Joonis 3. Andmete standardiseerimise tööetapid

1) Esimese etapina eraldati tekst, mis toodi programmi sisse artikleid sisaldava andmejärjendina (ingl *list*), sõnade kaupa sõnedeks. Selleks kasutati eesti keele töötlemiseks ja analüüsiks loodud mooduli EstNLTK töövahendit *tonekize_words*:

```
1.  from estnltk import Text
2.  def tokenizer(sisu): #sisesta sõnede järjend
3.      token_list = []
4.      for artikkel in sisu:
5.          text = Text(str(artikkel))
6.          text.tokenize_words()
7.          Token_list.append([text['text'][word['start']:word['end']] for word in
    text['words']])
8.      return token_list
```

Näiteks, kui rakendada eelnevalt kirjeldatud funktsiooni näidisjärjendi peal, mis sisaldab ainsa elemendina lauset „*Tartu Ülikooli geograafia osakond asutati 1919. aastal ja see kuulub Tartu Ülikooli loodus- ja täppisteaduste valdkonda, ökoloogia ja maateaduste instituuti.*“, väljastatakse järgnev tulemus:

```
[['Tartu', 'Ülikooli', 'geograafia', 'osakond', 'asutati', '1919.', 'aastal', 'ja', 'see',
'kuulub', 'Tartu', 'Ülikooli', 'loodus-', 'ja', 'täppisteaduste', 'valdkonda', ',',
'ökoloogia', 'ja', 'maateaduste', 'instituuti', '.']]
```

2) Järgmise sammuna teisendati sõnad algvormi ehk lemmatiseeriti. Selleks kasutati mooduli EstNLTK töövahendit *lemmas*:

```
9.  def lemmatize_text(sisend): #sisesta eraldatud sõnede järjend
10.     lemmatiseeritud = []
11.     for sone in sisend:
12.         lemma = [(Text(lem).lemmas[0]) for lem in sone]
13.         lemmatiseeritud.append(lemma)
14.     return lemmatiseeritud
```

Rakendades seda funktsiooni eelmise etapi näidisel, saame tulemuseks sõnade järjendi:

```
[['Tartu', 'Ülikool', 'geograafia', 'osakond', 'asutama', '1919.', 'aasta', 'ja', 'see',
'kuuluma', 'Tartu', 'Ülikool', 'loodus', 'ja', 'täppisteadus', 'valdkond', ',', 'ökoloogia',
'ja', 'maateadus', 'instituut', '.']]
```

3) Edasise töötlemise käigus eemaldati tekstist analüüsiks ebavajalikud sümbolid, nagu näiteks kirjavahemärgid ja numbrid. Kõikide ebavajalike sümbolite üheaegseks eemaldamiseks kasutati esmalt regulaaravaldiste rakendamiseks mõeldud moodulit *re* ja selle töövahendit *compile* sümbolite loendi defineerimiseks. Seejärel asendati igas sõnes need sümbolid tühja väärtusega:

```
15. import re
16. def remove_special_characters(text): #sisesta lemmatiseeritud sõnede list
17.     removed = []
18.     for lst in text:
19.         pattern =
    re.compile('[{}]'.format(re.escape("\/()[]{}#!$%&*+?.^_`~:+“”„,0123456789'")))
20.         filtered_tokens = filter(None, [pattern.sub('', token) for token in lst])
21.         filtered_text = ' '.join(filtered_tokens)
22.         filtered = filtered_text.replace('|',' ')
23.         filtered = filtered.replace('"','')
24.         removed.append(filtered)
25.     return removed
```

Jätkates sama näitega, saame tulemuseks järgneva järjendi:

```
['Tartu Ülikool geograafia osakond asutama aasta ja see kuuluma Tartu Ülikool loodus ja
täppisteadus valdkond ökoloogia ja maateadus instituut']
```

4) Viimase etapina eemaldati andmestikust ebavajalikud väljendid ehk peatussõnad. Selle etapi põhiliseks eesmärgiks oli vähendada töödeldava andmekoguse hulka, eemaldades sõnad, mis kannavad endas vähe sisulist väärtust, kuid esinevad tekstis sagedasti. Sellisteks sõnadeks on side- ja asesõnad, aga näiteks ka sõna „olema“ erinevad vormid. Sõnade eemaldamiseks kasutati lihtsat filtrit, mis lisas sisendist väljundjärjendisse ainult need sõnad, mis ei paikne peatussõnade loendis:

```
26. def remove_stopwords(sisend): #sisesta ilma kirjavahemarkideta lemmatiseeritud sõned
27.     removed = []
28.     stopword_list = 'see ma sa ta me te nad mina sina tema meie teie nemad et sest kuid
    vaid siis ja ning ega ehk või kui ka aga ning sest on ole olema kes mis'
29.     for artikkel in sisend:
30.         sone_lst = artikkel.split(' ')
31.         filtered_tokens = [sone for sone in sone_lst if sone not in stopword_list]
32.         removed.append(filtered_tokens)
33.         return removed
```

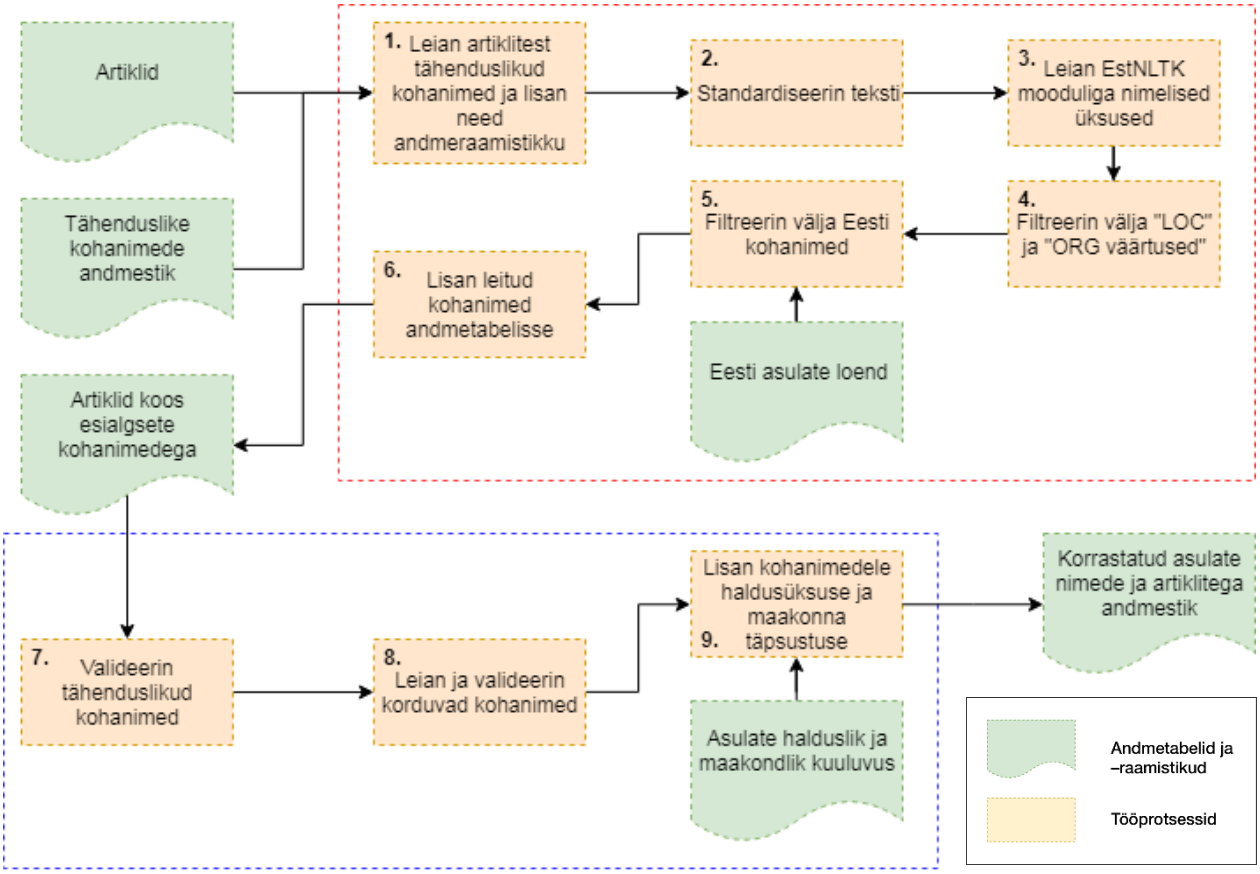
Eelnevalt kasutatud näite töötlemisel, on teksti standardiseerimise tulemus järgnev:

```
[['Tartu', 'Ülikool', 'geograafia', 'osakond', 'asutama', 'aasta', 'kuuluma', 'Tartu',
'Ülikool', 'loodus', 'täppisteadus', 'valdkond', 'ökoloogia', 'maateadus', 'instituut']]
```

Teksti standardiseerimise juures tuleb silmas pidada, et meetodite puhul, mis analüüsivad teksti semantilist sisu, nagu näiteks meelsusanalüüs, on sisuliste muutuste vältimiseks otstarbekam kasutada standardiseerimata teksti. See protsess on samas äärmiselt oluline sõnade arvust sõltuvate meetodite, nagu näiteks TF*IDF mudel, loomise puhul, kuna standardiseerimata teksti puhul ei loeta erinevaid sama sõna vorme samatähenduslikeks tunnusteks.

2.2.2 Asukohtade tuvastamine

Tekstidest kohanimede leidmine koosnes kahest suuremast etapist: kohanimede leidmine ja leitud nimede paikapidavuse kontrollimine. Nende ülesannete täitmiseks loodi kaks programmi, mis omakorda jagunesid üheksaks alamülesandeks. Programmide töö järjestus on välja toodud joonisel 4, kus punktiirjoontega on eraldatud kahes erinevas programmis asetleidvad protsessid.



Joonis 4. Asukohtade tuvastamine tekstandmestikust

1) Esimese, kohanimesid tuvastava programmi abil leiti esmalt tähenduslikud kohanimed.

Siinkohal tasub tähele panna, et see protsess sooritatakse enne teksti standardiseerimist, kuna sõnade teisendamise käigus muudetakse automaatselt mitmed sellised kohanimed nende tähendusele vastavasse algvormi ja sellisel juhul hiljem neid asukohana ei tuvastata. Tähenduslike kohanimede leidmiseks viidi eelnevalt läbi Eesti asulate ja suurt hulka eestikeelseid sõnu sisaldava andmestiku võrdlus, mille tulemusena tuvastati 522 sisulist tähendust omavat nime. Sellisteks nimedeks on sisulise tähendusega sõnadega sarnanevad kohanimed, nagu näiteks Maantee ja Sigala külad, levinud eesnimedega kirjapildis analoogsed toponüümid, nagu näiteks Jaanika ning Sandra külad ja mõne välisriigi või –linnaga identsed nimed, sealhulgas Norra ja Läti külad. Leitud väärtuste hulka ei kuulunud olemuslikult tähenduslikud, kuid eraldiseisvana tähendust mitte omavad nimed, näiteks Suuremõisa või Ülenurme. Andmete töötlemiseks loeti sisendandmed, kasutades moodulit *pandas*, andmeraamistikkudesse, lisati regulaaravaldisena otsitavale kohanimele käändelõpud ja võrreldi mooduli *re* töövahendiga *match* artikli sõnu tähenduslike kohanimede loendiga:

```
1.  import pandas as pd
2.  import re
3.  def t2henduslik_kohanimi(fail, fail1): #fail = artiklid, fail1 = tähenduslikud kohanimed
4.      df1 = fail1
5.      t2henduslik = list(set(df1['\ufeffkohad'].tolist()))#tähenduslikud kohanimed listi
6.      article_data = fail
7.      article_data = article_data.fillna('n/a')
8.      article_content = article_data['content'].tolist() #artiklite sisu listi
9.      kohad = []
10.     for article in article_content:
11.         koht_artikkel = '' #tuhi sone, kuhu olemasolul hakatakse kohanimesid liitma
12.         soned = article.split(' ') #artikkel sonede kaupa eraldi
13.         for sone in soned:
14.             sone = sone.replace('.','') #eemaldan kirjavahemärgid
15.             sone = sone.replace(',','')
16.             sone = sone.replace('"','')
17.             sone = sone.replace('!','')
18.             sone = sone.replace('?','')
19.             sone = sone.replace('”','')
20.             for puuduv in t2henduslik: #hakkan koha kaupa üle vaatama, kas tähenduslikud
    kohad esinevad artiklis
21.                 regex = '^' + puuduv + '(le|s|l|sse|le|st|lt|ks|ni|na|ta|ga|$)' + '($|\s)'
    #lisan käändelõpud, sõne alguse ja lõpu
22.                 if re.match(regex, sone): #otsin vasteid, mille olemasolul kirjutan
    kohanime sõneks
23.                     if koht_artikkel == '':
24.                         koht_artikkel = puuduv
25.                     else:
26.                         koht_artikkel = koht_artikkel + ', ' + puuduv
27.         kohad.append(koht_artikkel) #lisan sõne listi
28.     n = len(article_data.columns)
29.     article_data.insert(loc=n, column='t2henduslik', value=kohad) #kohanimede listist
    väärtused tabeli viimasesse veergu
30.     return article_data
```

2) Järgmiste protsesside läbiviimiseks oli oluline tekstid standardiseerida, kasutades selleks eelnevas peatükis esitatud metoodikat.

3) Seejärel leiti, kasutades EstNLTK töövahendit *named_entities*, kõik nimelised üksused. Nimeliste üksuste tuvastamise käigus leiti suurim osa kajastatud asulatest ja seega oli järgnev programmilõik väljatöötatud metoodikas asukohtade tuvastamisel kesksel kohal:

```
from estnltk import Text
def tagger(input, df): #sisendina standardiseeritud tekst ja andmeraamistik, kuhu on eelnevalt lisatud tähenduslikud kohanimed
    joint = []
    for article in input: #artiklid ühe kaupa
        for sisu in article:
            text=Text(sisu) #defineerin artikli sisu EstNLTK jaoks tekstina
            tagged = (list(zip(text.named_entities, text.named_entity_labels))) #lisan nimeliste üksuste sildid
```

Kasutades eelnevas alapeatükis väljatoodud näite standardiseerimise tulemust sisendandmetena, on nimeliste üksuste leidmise tulemus järgnev:

```
[('Tartu ülikool', 'ORG'), ('Tartu ülikool', 'ORG')]
```

4) Nimeliste üksuste tuvastamise järgselt leiti kõik asukohale või organisatsioonile viitavad sildid. Organisatsioonide leidmise vajalikkuse tingis asjaolu, et mitmed selle sildiga üksused kandsid endas ka informatsiooni asukoha kohta, nagu ilmnes ka kasutatud näites. Üksuste filtreerimiseks kasutati lihtsat tingimuslauset:

```
            for t in tagged: #nimelise üksuse sildiga sõned
                if t[1] == "LOC" or t[1] == "ORG": #kui tegu on asukoha või organisatsiooniga
                    akoht = t[0].split(' ')
                    for place in akoht:
                        bkoht = place.split('|') #mitme variandi puhul lisan mõlemad
                        for place2 in bkoht:
                            if place2 not in asukoht:
                                asukoht.append(place2)
            joint.append(cont + asukoht) #lisan tulemjärjendisse artikli standardiseeritud teksti koos leitud asukohtadega
```

Näidislausete puhul on tulemuseks järgnev andmejärjend:

```
[['Tartu Ülikool geograafia osakond asutama aasta kuuluma Tartu Ülikool loodus täppisteadus valdkond ökoloogia maateadus instituut', 'Tartu', 'ülikool']]
```

5) Tekstist asukohtade tuvastamise viimase etapina filtreeriti leitud nimeliste üksuste hulgast välja need väärtused, mis kajastavad Eestis asuvaid kohanimesid. Protsessi eesmärgiks oli eemaldada

tulemustest välismaised toponüümid ja asukohainfot mitte sisaldavad sõnad, mis lisanduvad andmestikku organisatsioonide nimede kaasamise ja EstNLTK kohatise ebatäpsuse tõttu. Selle läbiviimiseks võrreldi leitud kohanimesid Eesti asulanimede andmestikuga:

```
47.     location_data = pd.read_csv('asustusyksus1.csv', sep=',', header=0, dtype =
    {'ANIMI':str, 'AKOOD': str, 'AKOOD': str, 'ONIMI': str, 'OKOOD': str, 'MNIMI': str,
    'MKOOD': str}, encoding='utf-8') #loen tabelist sisse Eesti kohanimede andmed
48.     taagitud = []
49.     for val in joint: #eraldan andmestikust artiklid
50.         tagd = []
51.         koht = list(val[1:]) #eraldan artiklist leitud kohanimed
52.         for loc in koht:
53.             regex = '^' + loc.title() + '($|\s)'
54.             est_koht_set = set([x for x in kohtlist if re.match(regex, x)]) #otsin
    regulaaravaldisega Eesti kohanimed, eemaldan korduvad nimetused
55.             est_koht = list(est_koht_set)
56.             if len(est_koht) != 0:
57.                 for k in est_koht: # k = koha nimi
58.                     tagd.append(k)
59.             if len(tagd) == 0: #Kui kohanime pole, lisan tühja väärtuse (oluline hiljem
    list andmeraamiga ühendamisel)
60.                 tagd.append('')
61.         taagitud.append(tagd)
62.     kohad = []
63.     for value in taagitud: #kõikide artiklist tuvastatud asukohtade listist ühe kaupa
    artiklid välja
64.         sone = ''
65.         for v in value: #artikli kohanimedest ühe kaupa koha nimed sõneks, mis hiljem
    liidetakse andmetabeliga
66.             if len(sone) < 1:
67.                 sone = v
68.             else:
69.                 sone = sone + ', ' + v
70.         kohad.append(sone)
71.     n = len(df.columns)
72.     df.insert(loc=n, column='placename', value=kohad) #kohanimed andmeraami viimasesse
    veergu
73.     return df
```

Asulanimede tuvastamise programmi lõpptulemusena liideti saadud järjend andmeraamistikuga. Eelnevalt kasutatud näite puhul on tulemuseks üheliikmeline järjend, mille ainsa elemendi väärtuseks on sõne *„Tartu“*.

Selleks, et tuvastatud kohanimed siduda konkreetse Eesti asulaga, tuli esmalt kontrollida, kas andmestikust selekteeritud tähenduslikud nimed inditseerivad konkreetset kohta või oli tegu vastava nime tähendust väljendava sõnaga. Valideerimise teostamiseks loodi programm, mis koosnes suurest hulgast tingimuslausetest, mille abil leiti andmetabeli vastavast veerust need sõnad, mis suure tõenäosusega asukohainformatsiooni sisaldavad. Enne kohanimede filtreerimist loodi asukoha nimede tabelist sõnastik (ingl *dictionary*), kus igale kohanimele vastas üks või

korduva toponüümi puhul mitu selle nimelise koha asustusüksuse tüübi, omavalitsuse ja maakonna täpsustust (programmis defineeritud nimega *dct*). Lisaks loodi korduvaid nimesid sisaldav andmejärjend (defineeritud nimega *korduvad*).

7) Sisendandmete sisselugemise ja vajalike muutujate defineerimise järgselt, alustati kohanimede paikapidavuse kontrolli leitud potentsiaalsele toponüümile järgneva sõna kontrollimisega originaaltekstis, luues selleks esmalt regulaaravaldise, milles oli nimele lisatud käändelõpud. Regulaaravaldise võrdlemisel leiti tekstist kohad, kus uuritavat sõna oli mainitud. Selle alusel kontrolliti, kas järgnev sõna täpsustab asustusüksuse tüüpi. Kui järgneva sõna alusel nime valideerida ei õnnestunud, kontrolliti, kas artiklis oli viiteid haldusüksusele või maakonnale, kus uuritav asula paikneb. Kui ka siis nime õigsust kinnitada ei suudetud, eeldati, et tegu ei ole kohanimega:

```
1.  import pandas as pd
2.  import re
3.  import os
4.  from collections import Counter
5.
6.  def func(koht,dct,artikkel): #sisendiks uuritava koha nimi, kõiki Eesti asulaid ja nende
    omavalituse ja maakonna täpsustust sisaldav sõnastik ning vaadeldav artikkel
7.      koht_list = koht.split(',')[0]
8.     kohanimi = koht_list.split(' ')[0]
9.      ifkohanimi in t2henduslik: #kuikohanimi on tähenduslikkude nimede järjendis
10.         regex = '^' + kohanimi + '(le|s|l|sse|le|st|lt|ni|ga|ta|$)' + '($|/s)' #lisan
    käändelõpud
11.         laused = re.split('[.?!]', artikkel) #tekstist lauseteks
12.         for lause in laused:
13.             index = 0
14.             lause_list = lause.split(' ')
15.             for sona in lause_list: #laused sõnadeks
16.                 if re.match(regex,sona):
17.                     t2psustus = dct[kohanimi] #kohanime liik, haldusüksus ja maakond
18.                     max = len(lause_list) - 1
19.                     kontroll_nr = index + 1 #järgneva sõna indeks
20.                     for t2pne in t2psustus:
21.                         mk1 = ['maakond', 'maakonna', 'maakonda', 'maakonnast',
    'maakonnale', 'maakonnal', 'maakonnalt',
    'maakonnaks','maakonnani''maakonnana','maakonnata', 'maakonnaga']
22.                         mk2= ['maa', 'maad', 'maale', 'maal', 'maalt', 'maaks', 'maani',
    'maana', 'maata', 'maaga']
23.                         ov_list =
    ['vald','valla','valda','vallas','vallast','vallale','vallalt','vallal','vallaks','vallani'
    ,'vallana','vaalata','vallaga']
24.                         omv = t2pne.split(',')
25.                         county = omv[2].split(' ')[0] #maakonna nimi
26.                         omavalitsus = omv[1].split(' ')[0] #haldusüksuse nimi
27.                         if omv[0] == 'linn' and county != kohanimi: #kui tegu on linnaga,
    lisan kontrollimata kohanimedele, kuna linnasid enamasti tekstis ei täpsustata
28.                             return koht
29.                         if kontroll_nr <= max: #kui ei ole viimane sõna
30.                             kontroll_sona1 = lause_list[kontroll_nr] #leian kohanimele
```

```
    järgnev sõna
                            reg = '^' + omv[0] #sõne alguses, järgneb kontrollsõna(sellisel
    moel loetakse ka nt 'linnavalitsus' linna alla)
                            if re.match(reg,kontroll_sona1): #kontrollin, kas järgnev sõna
    täpsustab asustusüksust
                                return koht
                            if omv[0] == 'linn' and county == kohanimi: #kui linna ja
    maakonna nimi on sama, kontrollin, et ei mainitaks maakonda
                                for maakond1 in mk1:
                                    if kontroll_sona1 == maakond1:
                                        return 'n/a'
                            if kohanimi == omavalitsus and omv[0] != 'linn' and omv[0] !=
    'linnaosa': #kui koha ja omavalitsuse nimi on sama, kontrollin, et ei mainitaks valda
                                for vald1 in ov_list:
                                    if kontroll_sona1 == vald1:
                                        return('n/a')
                        for ov_l in ov_list:
                            s = omavalitsus + ' ' + ov_l
                            if s in artikkel: #kontrollin, kas valda on artiklis mainitud
                                return koht
                        for mknd1 in mk1: #kontrollin kahel erineval moel (maakond ja -
    maa), kas maakonda on täpsustatud
                            mk_sone1 = county + ' ' + mknd1
                            if mk_sone1 in artikkel:
                                return koht
                        for mknd2 in mk2:
                            mk_sone2 = county + mknd2
                            if mk_sone2 in artikkel:
                                return koht
                index += 1
        return 'n/a'
    else:
        return koht
```

8) Tähenduslike nimede kontrollimise järgselt täpsustati toponüümide, mida Eestis esineb rohkem kui üks kord, halduslik ja maakondlik kuuluvus. Nagu eelnevalt väljatoodud valideerimisprotsessi käigus, kasutati selleks kõigepealt sarnaseid tingimusi ja analoogset koodi, uurides esmalt nimele järgnevat sõna ning seejärel haldusüksuse või maakonna esinemist samas artiklis. Algoritm kasutas sisendandmetena kõiki leitud kohanimesid, kuhu lisati ka tähenduslike nimede valideerimise tulemused, ja eelnevalt leitud korduvate toponüümide järjendit. Erinevalt eelmisele koodile, lisati positiivse tulemuse korral nimele haldusüksuse ja maakonna nimi. Kuna kirjeldatud protsess oli äärmiselt sarnane eelnevalt väljatooduga, siis siinkohal selle koodi ei esitata.

9) Lisaks eelnevatele tingimustele, kasutati korduvate kohtade filtreerimisel teostatud sammude negatiivse tulemuse puhul eeldust, mille järgi oli juhul, kui artiklis ei olnud lisaks asula nimele ühtegi asukohale viitavat täpsustust, tegu kõrgeima võimaliku asustusüksuse tüübiga. Näiteks olukorras, kus sama nimega on alev ja küla, eelistati alevit:

```
151.    if len(kontroll_lst) == 1: #kui vahelistis on ainult ükskohanimi, kirjutan selle
    lõpplisti, mis lisatakse hiljem tabelisse
152.        sone_lst.append(kontroll_lst[0])
153.    elif len(kontroll_lst) > 1: #kui täpsustust vajavaid kohti on rohkem
154.        tyybid = dct[kohanimi] #leian koha asustusüksuse täpsustuse
155.        vrt = []
156.        for tyyp in tyybid:
157.            vrt.append(tyyp.split(',')[0])
158.            if 'linn' in vrt: #kui üks variant on linn, panen selle lõpplisti (Eestis
    korduvaid linnanimesid pole)
159.                num = vrt.index('linn')
160.                sone =kohanimi+' linn, '+tyybid[num].split(',')[1] + ', ' +
    tyybid[num].split(',')[2]
161.                sone_lst.append(sone)
162.            else:
163.                if 'linnaosa' in vrt: #kui yks variant on linnaosa, panen selle l6pplisti
    (Eestis korduvaid linnaosanimesid pole)
164.                    num = vrt.index('linnaosa')
165.                    sone =kohanimi+'linnaosa, ' + tyybid[num].split(',')[1] + ',' +
    tyybid[num].split(',')[2]
166.                    sone_lst.append(sone)
167.                else:
168.                    if 'alev' in vrt: #kui üks variant on alev, panen selle lõpplisti
    (Eestis korduvaid aleveid pole)
169.                        num = vrt.index('alev')
170.                        sone =kohanimi + ' alev,' + tyybid[num].split(',')[1] + ',' +
    tyybid[num].split(',')[2]
171.                        sone_lst.append(sone)
172.                    else:
173.                        if 'alevik' in vrt:
174.                            num = vrt.index('alevik')
175.                            numb = Counter(vrt)['alevik'] #alevikke võib ka mitu olla sama
    nimega
176.                            if numb == 1: #aga kui ei ole, lisan kohe lõpplisti
177.                                sone =kohanimi + ' alevik,' + tyybid[num].split(',')[1] +
    ',' + tyybid[num].split(',')[2]
178.                                sone_lst.append(sone)
179.                            else: #kui on mitu
180.                                nn = 0
181.                                nlist = []
182.                                for vr in vrt:
183.                                    if vr == 'alevik':
184.                                        nlist.append(nn)
185.                                        nn+=1
186.                                        for nl in nlist:
187.                                            sone =kohanimi + ' alevik,' +
    tyybid[nl].split(',')[1] + ',' + tyybid[nl].split(',')[2]
188.                                            control.append(sone) #lisan kõik variandid
    kontroll-listi
189.                        else: #kui siiani pole õnnestunud nime valideerida, on tegu
    nimega, mida kannab Eestis mitu küla
190.                            nn = 0
191.                            for tp in tyybid: #lisan kõik külad kontroll-listi
192.                                sone = kohanimi + ' küla,' + tyybid[nn].split(',')[1] +
    ',' + tyybid[nn].split(',')[2]
193.                                control.append(sone)
194.                                nn+=1
```

Korduvaid kohanimede, mida ei õnnestunud ka asulatüüpide järgi täpsustada, puhul valideeriti tulemus manuaalselt. Saadud tulemused, mis sisaldasid artiklitest tuvastatud asulate nimesid,

haldusüksusi ja maakondi, lisati valideerimisprotsessi järgselt andmetabelisse.

Lokaliseerimise algoritmidele veahinnangu andmiseks kontrolliti iga asustusüksuse tüübi puhul 1% kõikidest artiklitest, kus esines üks või mitu vastavasse klassi kuuluvat asulat. Kontrolli läbiviimiseks jaotati artiklid asustusüksuse tüübi kaupa alamvalimiteks ja vastav arv juhuslikult valitud artikleid loeti töö autori poolt läbi ning tuvastati lokaliseerimise edukus ja vea esinemisel selle liik.

2.2.3 Oluliste sõnade leidmine – TF*IDF mudel

Asulate meediakajastuse oluliste sõnade leidmisel kasutati TF*IDF mudelit, mille teoreetilisi lähtekohti kirjeldati peatükis 1.1.2. Mudeli loomiseks kasutati *Pythoni* masinõppe paketi *scikit-learn* tunnuste eraldamise mooduli *feature_extraction* tekstandmestikku analüüsiks mõeldud töövahendit *TfidfVectorizer*, millega leiti iga tuvastatud asula kohta suurima TF*IDF väärtusega sõnad. Oluliste sõnade tuvastamise programm kasutas sisendandmetena *Pythoni* sõnastikku, kus võtmeteks olid asulate nimed ja nendele vastavateks väärtusteks asulat sisaldavate artiklite standardiseeritud tekstidest koosnev järjend. Programm väljastas järjendi kohanimede ja neid kajastanud tekstide kõrgeima TF*IDF väärtusega sõnadega:

```
1.  from sklearn.feature_extraction.text import *
2.  def tf_idf(dict):
3.      list_votmed = list(dict.keys()) #sõnastikust kõik asulad
4.      tf = TfidfVectorizer(analyzer='word', ngram_range=(1,1), min_df = 0.1,, max_df = 1.0,
    norm = 'l2') #defineerin tf*idf leidmise meetodi: 'analyzer' = andmetüüp, 'ngram_range' =
    mitmest sõnast koosnevaid kombinatsioone uuritakse, hetkel vaadeldakse sõnu ühe kaupa,
    'min_df' = minimaalne suhteline artiklite arv, kus sõna võib esineda, et see arvesse
    võetakse (hetkel minimaalselt 10%), 'max_df' = maksimaalne suhteline artiklite arv, kus
    sõna võib esineda, et see arvesse võetakse (hetkel maksimaalne piirang puudub), 'norm' =
    normaliseerimse tüüp (l2 viitab l² ehk eukleidilise kaugusega normaliseerimisele)
5.      yld_list = []
6.      for voti in list_votmed: #kohanimed ühe kapua
7.          koha_list = []
8.          tekstid = dict[voti] #tekstid, kus on asulat mainitud
9.          tfidf_matrix = tf.fit_transform(tekstid) #leian tf*idf väärtused
10.         feature_names = tf.get_feature_names() #kõik tf*idf väärtused maatriksisse
11.         episode = dense[0].tolist()[0] #maatriksi väärtused järjendisse
12.         phrase_scores = [pair for pair in zip(range(0, len(episode)), episode) if pair[1] >
    0] #sõnade tf*idf väärtused ja nende indeksid maatriksist teisendatud listis
13.         sorted_phrase_scores = sorted(phrase_scores, key=lambda t: t[1] * -1) #järjestan
    indekseid ja tf*idf väärtusi sisaldavad elemendid kahanevalt
14.         phrase = [(feature_names[word_id]) for (word_id, score) in sorted_phrase_scores][0]
    #lisan väärtusele indeksi alusel sõnad ja valim suurima väärtusega sõna
15.         koha_list.append(voti) #lisan koha nime järjendisse
16.         koha_list.append(phrase) #lisan suurima väärtusega sõna järjendisse
17.         yld_list.append(koha_list) #lisan nime ja olulist sõna sisaldava järjendi
    tulemjärjendisse
18.     return yld_list
```

2.2.4 Tekstide automaatne klassifitseerimine

Automaatseks tekstide klassifitseerimiseks kasutati k-keskmise klasteranalüüsi. Selle läbiviimiseks leiti esmalt TF*IDF mudeliga analüüsiks vajaminevad tunnused analoogselt eelmises peatükis kirjeldatud meetodiga. Seejärel kasutati mooduli *scikit-learn* töövahendit *KMeans* klastrite moodustamiseks. Kuna k-keskmise klasteranalüüs eeldab klastrite arvu eelnevat defineerimist, määrati selleks arvuks, lähtuvalt peatükis 1.2 väljatoodud veebimeedia põhilistest temaatilistest klassidest, kaheksa. Klastrite moodustamise järgselt leiti nende artiklite pealkirjad, mis vastavatesse klassi kuulusid ja iga klass seoti eelnevalt tuvastatud asukohtadega. Seejärel leiti, millisesse klassi kuuluvaid artikleid iga asukohaga seoses kõige rohkem esines:

```
1.  from tools import build_feature_matrix
2.  from sklearn.feature_extraction.text import *
3.  import pandas as pd
4.  import os
5.  from sklearn.cluster import KMeans
6.  n_art = dfr['normalized'].tolist() #eelnevalt normaliseeritud artiklite tekst
    andmeraamistikust järjendisse
7.  vectorizer, feature_matrix = build_feature_matrix(n_art, feature_type='tfidf', min_df=0.1,
    max_df=1.0, ngram_range=(1, 1)) #leian TF*IDF meetodil tunnused
8.  feature_names = vectorizer.get_feature_names()
9.
10. def k_means(feature_matrix, num_clusters): #funktsioon k-means klastrite leidmiseks
11.     km = KMeans(n_clusters=num_clusters, max_iter=10000) #max_iter – maksimaalne arv
    kordusi, mid klasside leidmiseks tehakse (kirjeldatud peatükis 1.1.4)
12.     km.fit(feature_matrix)
13.     clusters = km.labels_
14.     return km, clusters
15.
16. num_clusters = 8 #loodavate klastrite arv
17. km_obj, clusters = k_means(feature_matrix=feature_matrix, num_clusters=num_clusters)
18. dfr['Cluster'] = clusters
19.
20. def get_cluster_data(clustering_obj, article_data, feature_names, num_clusters,
    topn_features=10):
21.     cluster_details = {}
22.     ordered_centroids = clustering_obj.cluster_centers_.argsort()[:, ::-1] #leian klastrite
    tsentroidid
23.     for cluster_num in range(num_clusters):
24.         cluster_details[cluster_num] = {}
25.         cluster_details[cluster_num]['cluster_num'] = cluster_num
26.         key_features = [feature_names[index] for index in ordered_centroids [cluster_num,
    :topn_features]]
27.         cluster_details[cluster_num]['key_features'] = key_features #leian klastrite
    tunnused
28.         articles = dfr[dfr['Cluster'] == cluster_num]['titles'].values.tolist()
29.         cluster_details[cluster_num]['articles'] = articles #leian artilid, mis vastavasse
    klastrisse kuuluvad
30.     return cluster_details
31.
32. cluster_data = get_cluster_data(clustering_obj=km_obj,article_data=dfr,
    feature_names=feature_names, num_clusters=num_clusters,topn_features=5)
```

2.2.5 Meelsusanalüüs

Artiklite sisu meelsuse automaatseks tuvastamiseks kasutati Eesti Keele Instituudi spetsialistide poolt väljaarendatud töövahendit *Valence*. Töövahend on loodud veebibrauseris töötava pistikprogrammina (ingl *plugin*), mille eesmärgiks on etteantud teksti klassifitseerimine positiivseks, neutraalseks või negatiivseks ja vastavalt klassifitseerimise tulemusele väljastada HTML kood, kus meelsust väljendavad terminid on värvitud eelnevalt defineeritud värvi ning kogu tekstile on antud kokkuvõttev meelsuse hinnang (H. Pajupuu, Altrov ja J. Pajupuu, 2016). Kuna eelnevalt nimetatud programm on kirjutatud programmeerimiskeeles *Python* ja avatud lähtekoodiga, saab sealt eksportida vajalikke funktsioone ka teistesse programmidesse.

Programm leiab meelsuse dokumendi tasemel ja kasutab selleks multinomiaalse Bayens'i klassifikaatori meetodit, mida kirjeldati peatükis 1.1.4. Klassifitseerimisel võetakse aluseks Eesti Keele Instituudi spetsialistide poolt eelnevalt defineeritud tekstikorpus, mis koosneb 2122 portaalis Postimees avaldatud artikli sisust ja artiklite meelsust väljendavast väärtushinnangust. Lisaks on programmi autorite poolt loodud 40 000 sõna ja nende emotsionaalsust väljendavat väärtust sisaldav sõnastik (H. Pajupuu, Altrov ja J. Pajupuu, 2016). Meelsust väljendava väärtuse väljastamiseks eraldati analüüsi tulemusena saadud HTML koodist uuritavat näitajat sisaldav lõik, mis esines kuue võimaliku väärtusena: ainult negatiivset sisu iseloomustas väärtus „*only negative*“, suuremas osas negatiivsele emotsioonile osutas näitaja väärtusega „*mostly negative*“, neutraalsele sisule vastas tulem „*neutral*“, segatud meelsusega sisule, kust kindlat emotsiooni eraldada ei saanud, väärtus „*mixed*“, suuremas osas positiivsele „*mostly positive*“ ja ainult positiivsele sisule inditseeris resultaat „*only positive*“.

Programm kasutas sisendandmetena kohanimesid ja artiklite sisu sisaldavad andmetabelit ning väljastas tabeli, kuhu oli lisatud iga artikli kohta selle meelsust väljendav väärtus:

```
1.  import pandas as pd
2.  from valence import valencecolor
3.  def valents(article_data): #funtsiooni parameetriks andmetabel asukohtade ja artiklite
    sisuga
4.      valence = []
5.      df = article_data.fillna('n/a') #täidan tühjad lahtrid väärtusega 'n/a'
6.      article_content = df['content'].tolist() #tabelist sisu järjendisse
7.      koht = df['placename'].tolist() #kohanimed järjendisse
8.      nr=0
9.      for artikkel in article_content:
10.         if koht[nr] != 'n/a': #kui on tuvastatud mõnikohanimi
11.             tulem = valencecolor.marktext(artikkel, dataonly = True, lexiconbased =
    True).split('<')[0] #rakendan funktsiooni valencecolor.marktext ja võtan HTML'st soovitud
```

```
    väärtuse
            valence.append(tulem)
        else: #kui mitte, siis lisan väärtuse 'n/a'
            valence.append('n/a')
        nr += 1
    n = len(df.columns) #leian, kui palju veergusid tabelis on
    df.insert(loc=n, column='valence', value=valence) #lisan andmed viimasesse veergu
    return df
```

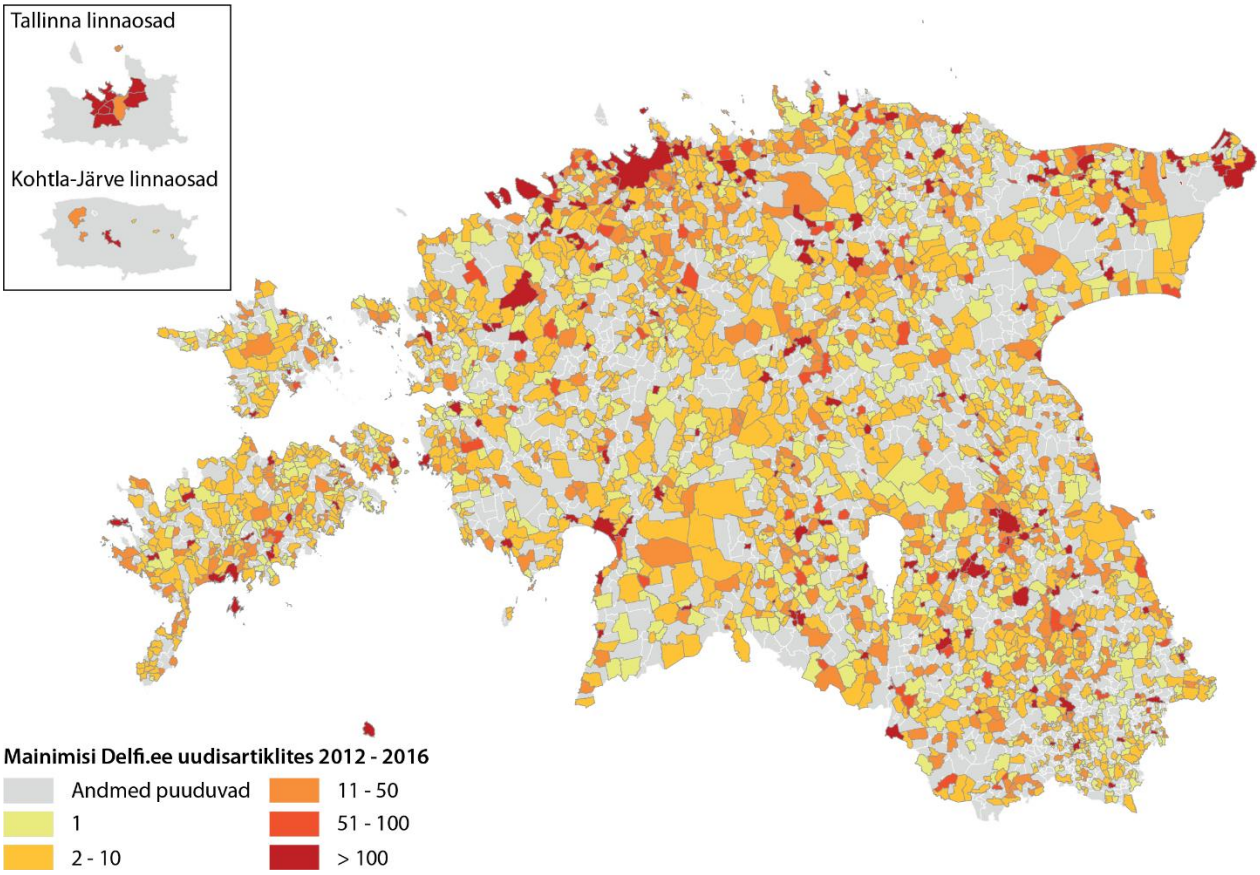
Sisendandmete puhul oli oluline kasutada standardiseerimata teksti, kuna standardiseerimise protsessi käigus võis osa meelsust väljendavast informatsioonist käänete eemaldamisega kaduma minna. Asulate meediakajastuse meelsuse üldistamiseks summeeriti kohanimede kaupa eelnevalt kirjeldatud programmi tulemused, asendades sõnalised väärtused numbrilistega järgnevalt: ainult positiivsele sisule omistati väärtus 1, enamuses positiivsele 0,5, segatud ja neutraalsele tulemile 0, enamuses negatiivsele -0,5 ning ainult negatiivsele -1.

3. Tulemused ja arutelu

Tehniliselt oli tekstandmestiku töötlemine äärmiselt aja- ja ressursimahukas protsess. Sellest tulenevalt oli eelnevalt kirjeldatud metoodika läbiviimisel kohati vajalik andmemahu vähendamine või vähem ressurssi nõudva algoritmi valimine. Näiteks vähendati andmemahtu oluliste sõnade tuvastamisel ja edaspidisel klassifitseerimisel meetodite puhul, mis kasutasid tunnuste leidmisel TF*IDF väärtusi, seades piirangu, mille kohaselt pidi uuritavat sõna sisaldama vähemalt 10% asukohta kajastavatest artiklitest. Lisaks vähendas andmemahtu peatussõnade väljafiltreerimine. Piiratud arvutusliku ressursi tõttu, osutus vajalikuks loobumine mõnest täisautomaatsest klassifitseerimisalgoritmist, nagu Dirichlet'i peitlahutus (ingl *Latent Dirichlet allocation*), mille läbiviimisel oli vajalik maatriksite, mille mõõtmeteks olid artiklite arv ja sõnade hulk artiklites kokku, transponeerimine ning korrutamine. Ajaliselt osutus kõige pikemaks protsessiks tekstide normaliseerimine, milleks kulus näidisandmestiku puhul 33 tundi, 14 minutit ja 30 sekundit. Asukoha tuvastamine võttis aega 23 tundi, 32 minutit ja 18 sekundit ning erinevad läbiviidud analüüsid kestsid alla viie minuti. Kokku kulus programmidel tulemuste väljastamiseks mõned minutid alla 57 tunni. Kuna magistritöö metoodika väljatöötamine eeldas programmidega mitmeid katsetusi, töötati algoritmidega eraldi, mis tingis mõningate ülesannete korduva täitmise. Seda oleks saanud vältida kõiki ülesandeid järjest täitva programmi loomisega. Ajaliselt ökonoomsemate meetodite loomiseks tuleks kaaluda ka regulaaravaldiste kasutamise vähendamist.

3.1 Asukohtade tuvastamine

Lähtuvalt magistritöö esmasest eesmärgist, milleks oli siduda näidisandmetena kasutatavad uudisartiklid neis kajastatud asulatega, tuvastati protsessi käigus ainult asulate nimed ja seega ei sisalda tulemused maakondade, valdade ega muude geograafiliste objektide mainimiste arvu vaadeldavas andmestikus. Kokku uuriti 385 819 artiklit, milles vähemalt üks Eestis paiknev asula tuvastati 107 386 juhul, mis moodustas koguvalimist 28%. Kokku tuvastati ja valideeriti eelnevalt kirjeldatud metoodikat rakendades vaadeldava perioodi jooksul uudislugudest 2621 unikaalset kohanime, sealhulgas 47 linna, 13 linnaosa, 12 alevit, 177 alevikku ja 2372 küla, kattes sellega 56% kõikidest Eesti asulatest (Joonis 5).



Joonis 5. Aastatel 2012-2016 portaalis Delfi.ee avaldatud artiklitest tuvastatud asukohtade mainimiste arv asustusüksuste kaupa (aluskaart: Maa-amet)

Enim mainiti artiklites Tallinna linna, mis tuvastati kokku 47% asukoha infot sisaldavates artiklites. Tallinnale järgnesid Tartu ja Pärnu linnad (tabel 1). Linnaosadest mainiti enim Tallinna koosseisu kuuluvaid asustusüksuseid, mille tuvastamiste hulk ületas kordades teise Eesti linnaosasid hõlmava linna, Kohtla-Järve vastavaid üksuseid (tabel 1).

Tabel 1. Kümme enim mainitud linna ja linnaosa

Linna nimi	**Artiklite arv**	**Linnaosa nimi**	**Artiklite arv**
Tallinna linn	50 394	Nõmme linnaosa	2968
Tartu linn	23 684	Lasnamäe linnaosa	2462
Pärnu linn	10 188	Pirita linnaosa	1957
Narva linn	5965	Põhja-Tallinna linnaosa	1308
Rakvere linn	5057	Mustamäe linnaosa	915
Viljandi linn	4914	Haabersti linnaosa	537

Rapla linn	2808	Ahtme linnaosa	120
Kuressaare linn	2780	Sompa linnaosa	25
Võru linn	2638	Järve linnaosa	24
Haapsalu linn	2632	Kesklinna linnaosa	11

Alevitest mainiti enim Kohilat, millele järgnesid Vändra ja Märjamaa, alevikest domineerisid Saku ja Viimsi, mida tuvastati järgnevates alevikest pea kolm korda enam (tabel 2).

Tabel 2. Enim mainitud alevid ja alevikud

Alevi nimi	**Artiklite arv**	**Aleviku nimi**	**Artiklite arv**
Kohila alev	665	Saku alevik	2180
Vändra alev	459	Viimsi alevik	1908
Märjamaa alev	429	Harku alevik	689
Paikuse alev	209	Ämari alevik	557
Järva-Jaani alev	194	Värska alevik	537
Aegviidu alev	191	Kuusalu alevik	494
Järvakandi alev	143	Väike-Maarja alevik	479
Pärnu-Jaagupi alev	90	Aruküla alevik	465
Kohtla-Nõmme alev	57	Tori alevik	354
Lavassaare alev	33	Orissaare alevik	348

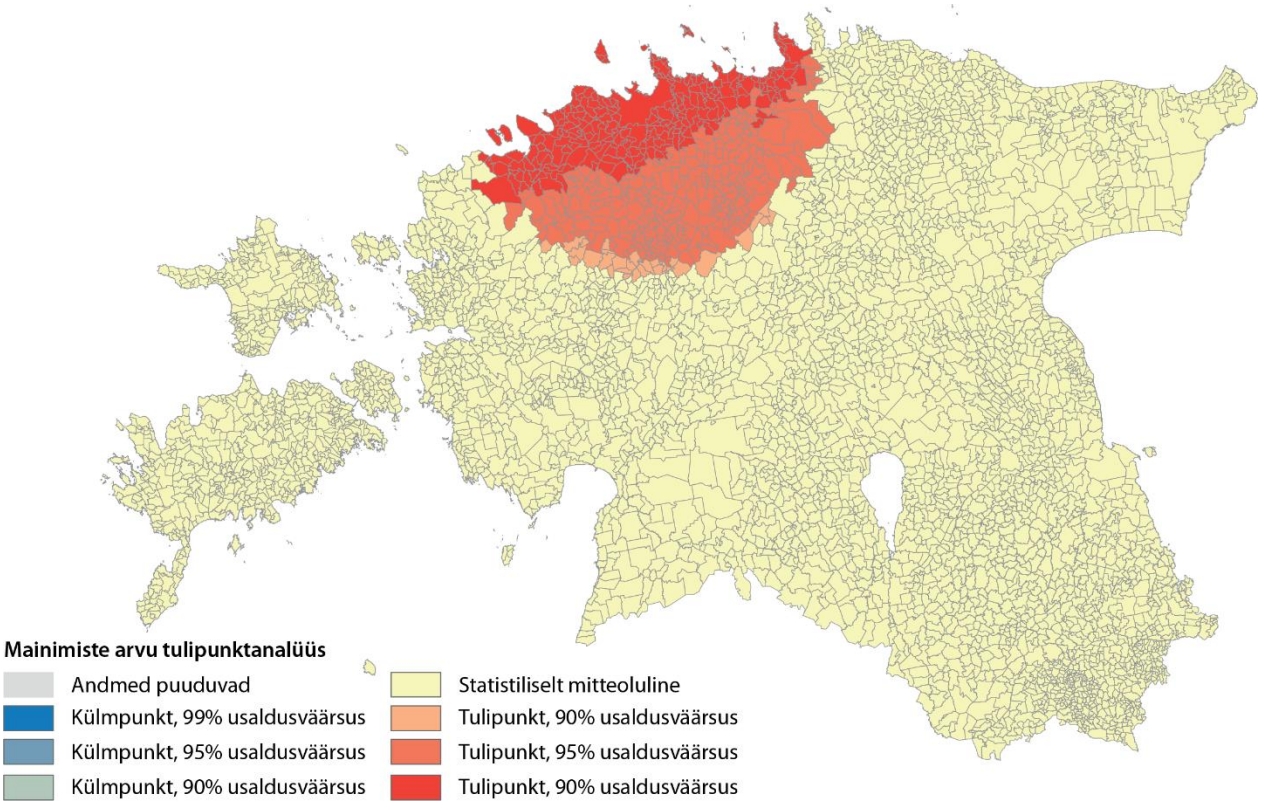
Kümne enim mainitud küla hulgas leidus mitmeid kohanimesid, mille korrektne tuvastamine on küsitav. Enim kordi tuvastatud külaks osutus Padari küla 509 artikliga, millele järgnesid Ruhnu ja Jõelähtme külad. Sarnased küsitavusi tekib, lisaks Padari küla seostamisele isikunimega ja Sõrve küla esinemise pigem poolsaare kui küla nimena, ka tabeli järgnevate asukohtade osas (tabel 3).

Tabel 3. Enim mainitud külad

Küla nimi	**Artiklite arv**
Padari küla	509
Ruhnu küla	395
Jõelähtme küla	394
Sõrve küla	284
Haanja küla	279

Luke küla	271
Laagna küla	265
Kaarma küla	260
Matsalu küla	250
Purga küla	247

Meediakajastuse üldise jaotumise iseloomustamiseks leiti mainimiste arvu ruumilist autokorrelatsiooni iseloomustav *Moran'i I* indeks, mille puhul on nullhüpoteesiks uuritava nähtuse ruumilise paiknemise sarnanemine juhuslikule jaotusele ja sisulisteks hüpoteesideks positiivse *z*-skoori puhul nähtuste koondumine ja negatiivse *z*-skoori puhul nende ühtlane jaotumine (Li, Calder, Cressie, 2007). Testi tulemusena selgus, et suure statistilise usaldusväärsusega ($p < 0,01$) saab nullhüpoteesi ümber lükata ja kuna testi käigus leitud standardhälvet iseloomustava *z*-skoori väärtus oli tugevalt positiivne ($z = 3,25$), oli vaadeldavas ajavahemikus tuvastatud meediakajastus ruumiliselt selgelt koondunud. Jaotumise täpsemaks iseloomustamiseks viidi läbi tulipunktanalüüs (ingl *hot spot analysis*), mille abil leiti suure mainimiste arvuga asulate koondumiskohad ehk tulipunktid ja väikese mainimiste arvuga asustusüksuste koondumiskohad ehk külmpunktid ning nende statistilised usaldusväärsused. Analüüsi tulemusena eraldus selgelt Tallinn ja selle lähiümbrus suure statistilise usaldusväärsusega tulipunktina – 99% ja 95% usaldusväärsusega tulipunktid katsid terve Harjumaa ja Raplamaa põhjaosa. 90% statistilise usaldusväärsusega tulipunkt kattis väikese osa Rapla- ja Järvamaast. Ühtegi külmpunkti ei tuvastatud, millest saab järeldada, et erinevalt kõrgete mainimiste arvuga asulatest, olid selle näitaja madalama väärtusega paigad ühtlasemalt jaotunud (Joonis 6).



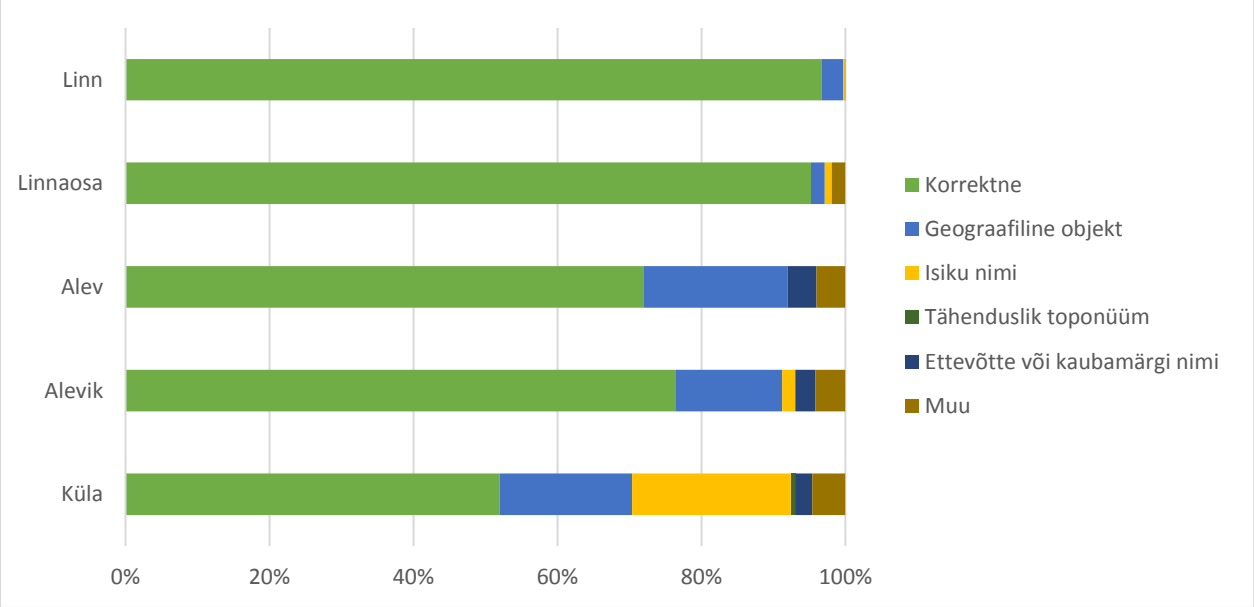
Joonis 6. Aastatel 2012-2016 portaalis Delfi.ee avaldatud artiklitest tuvastatud asukohtade mainimiste arvu tulipunktanalüüs (aluskaart: Maa-amet)

Asukohtade tuvastamise täpsuse hindamiseks kontrolliti iga asustuse klassi puhul manuaalselt 1% kõigist artiklitest, milles vastavaid asulaid leidus. Kuna artiklites tuvastati tihtipeale mitu erinevasse klassi kuuluvat asulat, võis kontrolli läbiviimisel sama artikkel esineda valimis korduvalt. Kokku kontrolliti tuvastamise täpsust 2006 juhul, kus iga asustusüksuse klass esines arvuliselt vastavalt nende esinemissagedusele koguvalimis. Leitud vead jaotati viide klassi:

- muu geograafiline objekt – näiteks jõgi, järv või saar, mis kannab asulaga sama nime;
- kohanimega identne isiku nimi;
- kohanimega identne sõna;
- ettevõtte või kaubamärgi nimi;
- muu.

Veaks ei loetud selliseid objekte ja asutusi, mis on otseselt seotud kindla asulaga, nagu näiteks koolid või spordirajatised. Lisaks ei loetud veaks külaga identsete piiridega geograafiliste objektide esinemist, mis näiteks ilmneb Ruhnu saare puhul.

Kontrollimise käigus tuvastati üldiseks asulate leidmise täpsuseks 87%, olles ülekaalukalt suurim linnade (97%) ja linnaosade (95%) puhul, samas kui külade puhul oli asula korrektselt tuvastatud vaid 52% kontrollitud juhtudest (Joonis 7). Kogu kontrollitud andmestikus esines veana enim mõne muu geograafilise objekti tuvastamine asulana (7%), millele järgnes isikunimede märkimine asukohana (4%). Kohanimega identseid sõnasid, ettevõtete ja kaubamärkide nimesid ning muid vigu esines kogu kontrollandmestikus 1% või vähem. Asustusüksuste kaupa leiti suurim viga külade puhul – 22% kontrollitud asulate tuvastuste puhul oli tegu isikunimega ja 18% muude geograafiliste objektidega. Nimede esinemine tuvastatud asukohtade hulgas oli väga iseloomulik just külade puhul, esinedes vähesemal määral ka alevikkude ja linnaosade seas, samas kui geograafiliste objektide ekslikku tuvastamist leiti iga klassi puhul, olles suurim alevite (20%) ja väikseim linnaosade (2%) hulgas (Joonis 7).



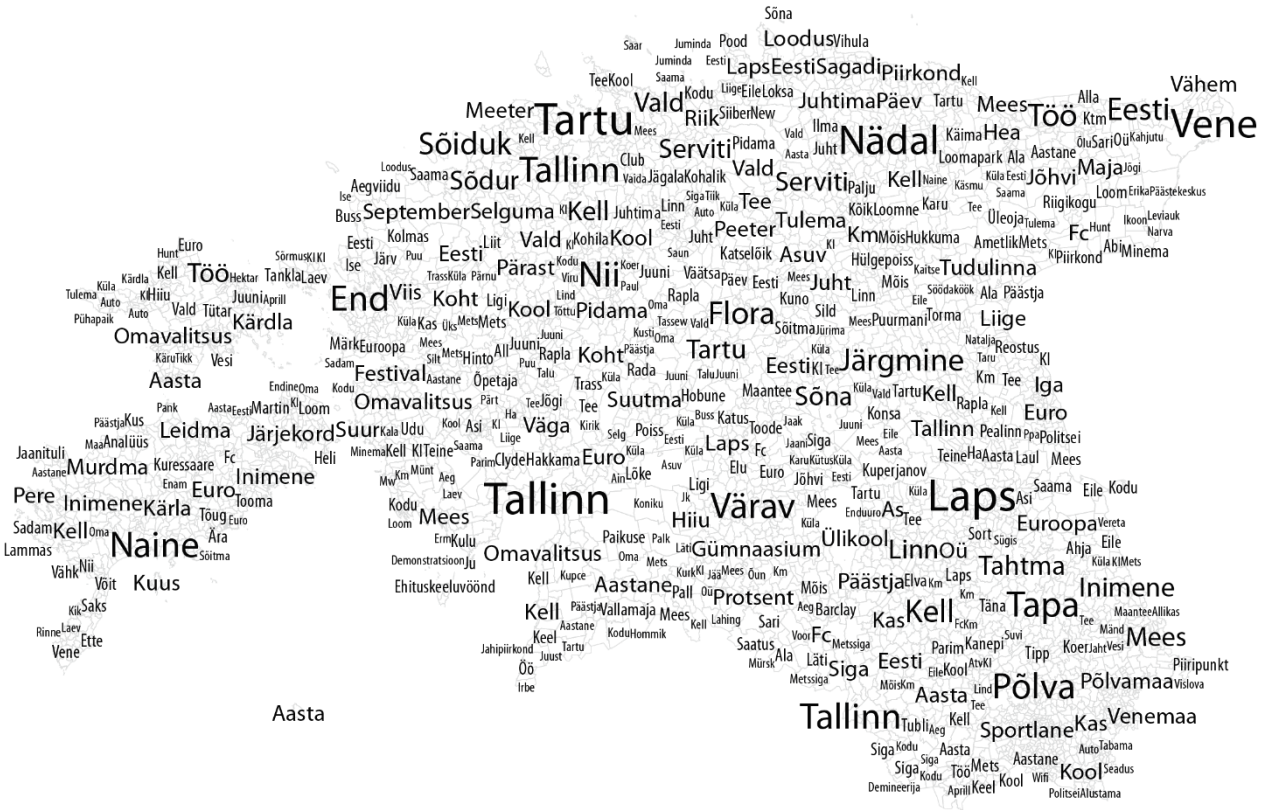
Joonis 7. Asukohtade tuvastamise täpsus manuaalselt kontrollitud andmestikus

Olgugi, et EstNLTK nimeliste üksuste tuvastamise meetodid võimaldavad leida ka isikunimesid, ei tagaks see metoodikaga ühildamiseks, lähtuvalt asjaolust, et EstNLTK kasutab nimede tuvastamiseks eelnevalt defineeritud loendit, piisavat täpsust ja seega otsustati nimetatud funktsionaalsust läbiviidud analüüsis mitte rakendada. Nimede tuvastamiseks ei oleks piisav olemuselt tähenduslike või korduvate kohanimede valideerimisele sarnaste, üksuste esinemise loogikast lähtuvate filtrite rakendamine, sest isikunimede puhul on grammatiline ja sisuline raamistik liiga mitmekesine ning eranditerohke. Üheks lihtsaks eelduseks mainitud filtri loomisel

oleks situatsioon, kus tekstis esineks alati järjestikku ees- ja perekonnanimi – sel juhul oleks isikud tekstist kergemini tuvastatavad ja algoritmi usaldusväärsus suurem. Kahjuks on nimede kirjutamine tekstis tunduvalt mitmekesisem, esinedes ka näiteks kujul, kus perekonnanimi figureerib eraldiseisvana või sellele eelneb eesnime initsiaal ja seega ei ole väljatoodud eeldus vea vähendamiseks ammendav. Muude geograafiliste objektide, nagu näiteks jõgede, järvede või tänavanimede eristamiseks asulanimedest oleks vajalik tuvastatud asulanimede võrdlemine geograafiliste objektide nimesid sisaldava andmestikuga. Seda oleks võimalik teostada kasutades analoogseid programme, nagu metoodika läbiviimisel tähenduslike nimede ja valdade ning maakondade väljafiltreerimisel. Olenemata kohatisest ebatäpsusest, automaatse andmetöötluse eripäradest ja algoritmide vähesest seadistamist näidisandmetele vastavaks, on võimalik asukohtade tuvastamise programmide tulemused lugeda testandmestiku kontrollimisel saavutatud 87% täpsuse põhjal õnnestunuks. Lokaliseerimise edukusele hinnangu andmist raskendab sarnaste uurimuste puudumisest tingitud võrdluse puudumine.

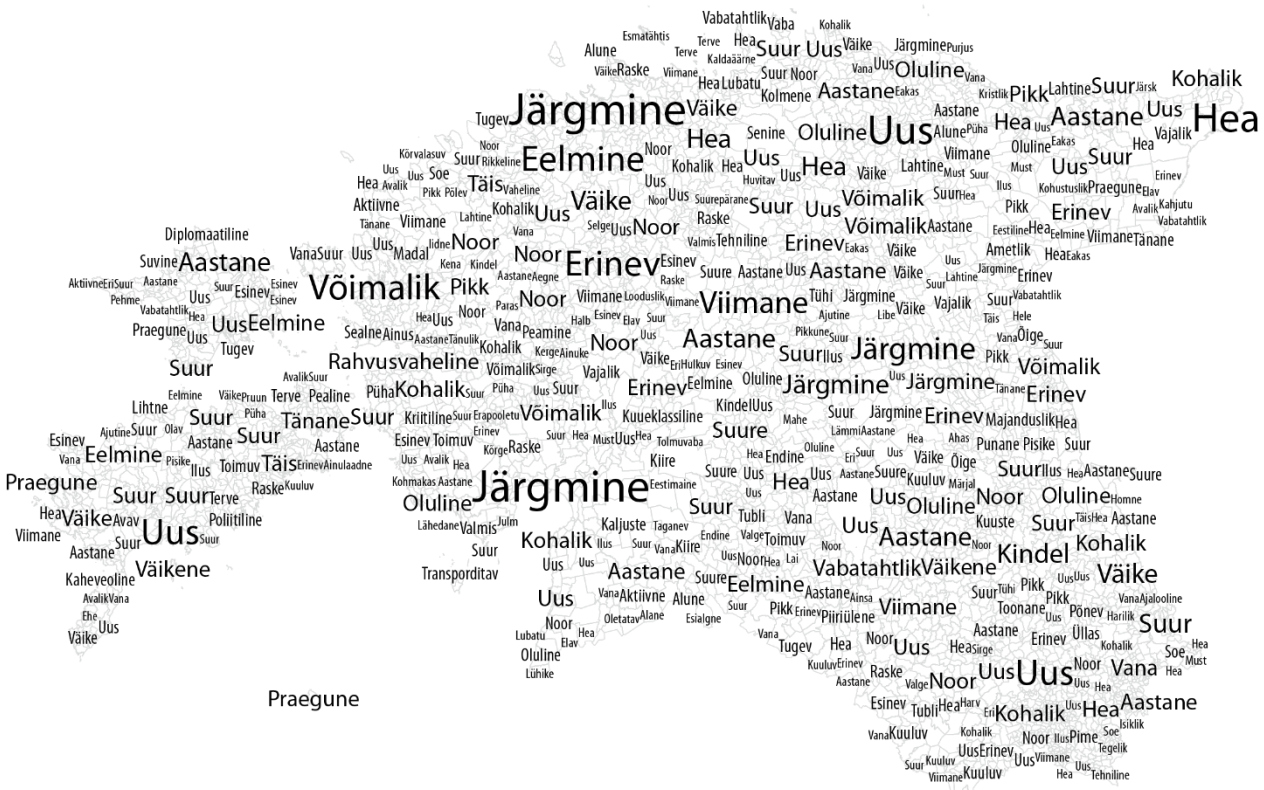
3.2 Oluliste sõnade leidmine

Oluliste sõnade leidmiseks tuvastati iga asustusüksuse suurima TF*IDF väärtusega sõna tingimusel, et kõige olulisem sõna ei ole koha nimi ise. Enim tuvastati kõige olulisema sõnana *„küla“*, mis oli suurima TF*IDF väärtusega 160 asustusüksuse puhul. Esinemissageduselt järgnesid sõnad *„aastane“* 94, *„kodu“* 60, *„mees“* 57 ja *„kell“* 43 mainimisega. Täheldatav on ka lühendite võrdlemisi sagedane esinemine oluliste sõnadena: lühend *„kl“* tähenduses *„kell“* või *„klass“* tuvastati 43, *„km“* ehk *„kilomeeter“* 35 ja *„oü“ „osaühingu“* lühendina 10 korral. Lisaks leidus vähemal määral spordialadega seotud lühendeid, nagu näiteks *„fc“* (ingl *football club*). Tuvastatud oluliste sõnadena kerkisid mitmel juhul esile ka teiste asustusüksuste nimed, eriti Eesti suurimad linnad – Tallinn oli olulisemaks sõnaks 20 ning Tartu ja Pärnu mõlemad 10 korral (Joonis 8).



Joonis 8. Eesti asulate kõrgeima TF*IDF väärtusega sõnad aastatel 2012-2016 portaalis Delfi.ee avaldatud artiklites (aluskaart: Maa-amet)

Oluliste sõnade tuvastamisel tuleb eelkõige silmas pidada olulisuse definitsiooni, mis oli defineeritud lähtuvalt TF*IDF väärtuse arvutamise metoodikale, omistades kõrgeima väärtuse sõnadele, mida võis leida vähestest artiklitest, aga nendes, kus see esines, leidus seda sagedasti. Sellest tulenevalt ei pruugi arvutuslik olulisus alati vastata intuitiivsele. Olenemata asjaolust, et oluliste sõnade leidmine võimaldab meediakajastust võrdlemisi atraktiivselt visualiseerida, ei sisalda üksikute sõnade väljatoomine asukohakuvandi kohta järelduste tegemiseks piisavat hulka informatsiooni. Oluliste sõnade põhjal järelduste tegemist raskendab asjaolu, et leitud sõnade varieeruvus oli, lähtuvalt kasutatud täisautomaatsest metoodikast, kus ainsaks tingimuseks oli sõna erinemine kohanimest, väga suur. Sõnade varieeruvuse vähendamiseks leiti sarnasel meetodil igat mainitud asulat kõige paremini iseloomustavad omadussõnad. Kokku tuvastati 515 unikaalset omadussõna, millest enim levinud olid „*aastane*“ (247 asulat), „*uus*“ (116) ja „*suur*“ (112) (Joonis 9).

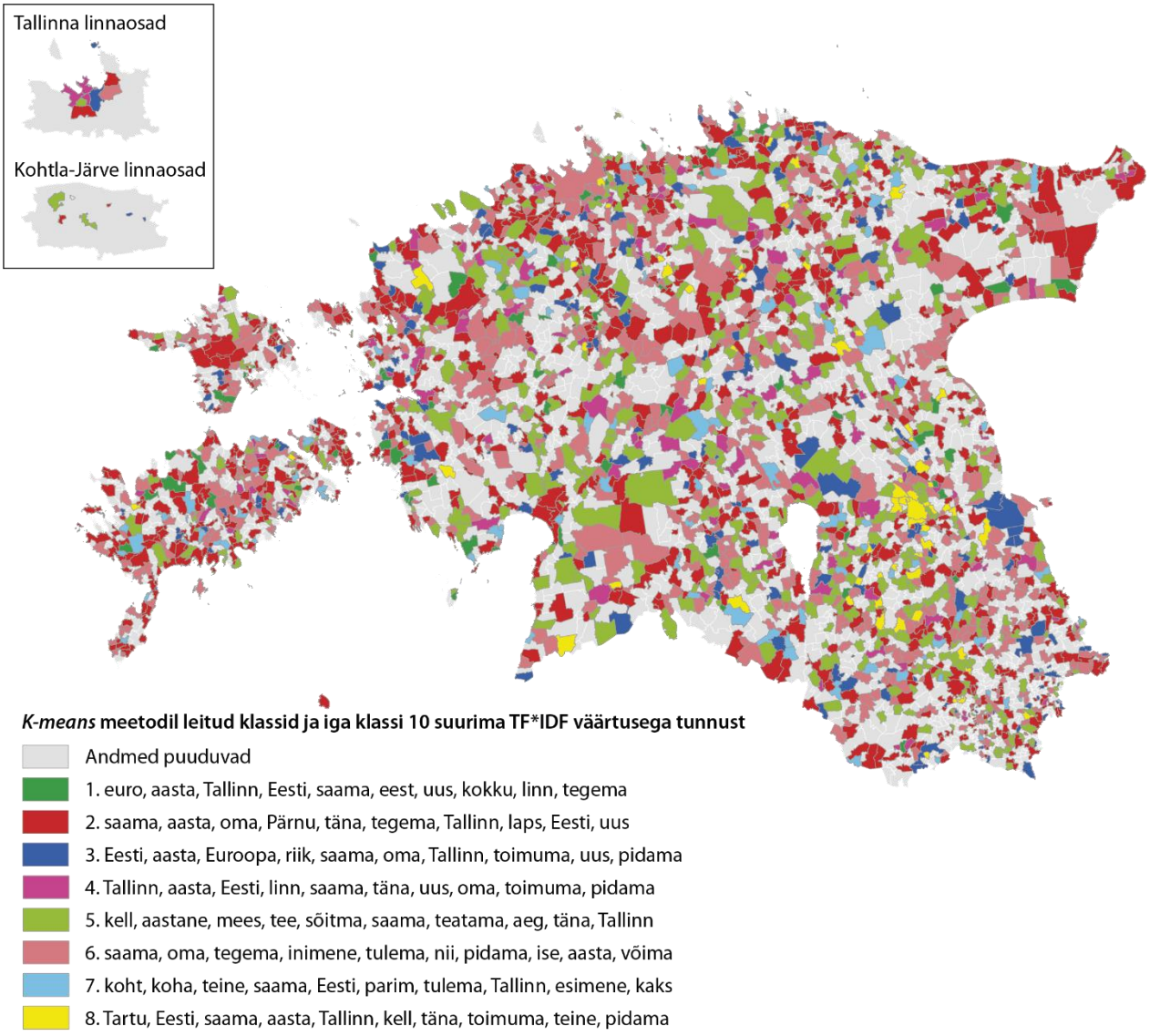


Joonis 9. Aastatel 2012-2016 portaalis Delfi.ee avaldatud Eesti asulaid kajastanud artiklite kõrgeima TF*IDF väärtusega omadussõnad (aluskaart: Maa-amet)

Olgugi, et enim esinevate omadussõnade seas on mitmeid neutraalse sisuga termineid, nagu näiteks „*järgmine*“ ja „*aastane*“, võimaldab omadussõnade väljafiltreerimine seoseid meediakuvandi meelestusega märksa paremini luua, kui filtreerimata sõnade vaatlemine. Näiteks esines sõna „*hea*“ kõige olulisema omadussõnana 72 asula puhul, samas kui sõna „*halb*“ tuvastati ainult neljal juhul.

3.3 Tekstide automaatne klassifitseerimine

Selleks, et välja selgitada, millised kohad läbi meediakajastuse omavahel enim sarnanesid, viidi läbi automaatne klassifitseerimine, kasutades selleks k-keskmise klasteranalüüsi. Artiklid jaotati eelnevalt mainitud meetodi abil esmalt kaheksasse klassi ja seejärel leiti, millisesse klassi kuuluvates artiklites asulat enim mainiti (Joonis 10).

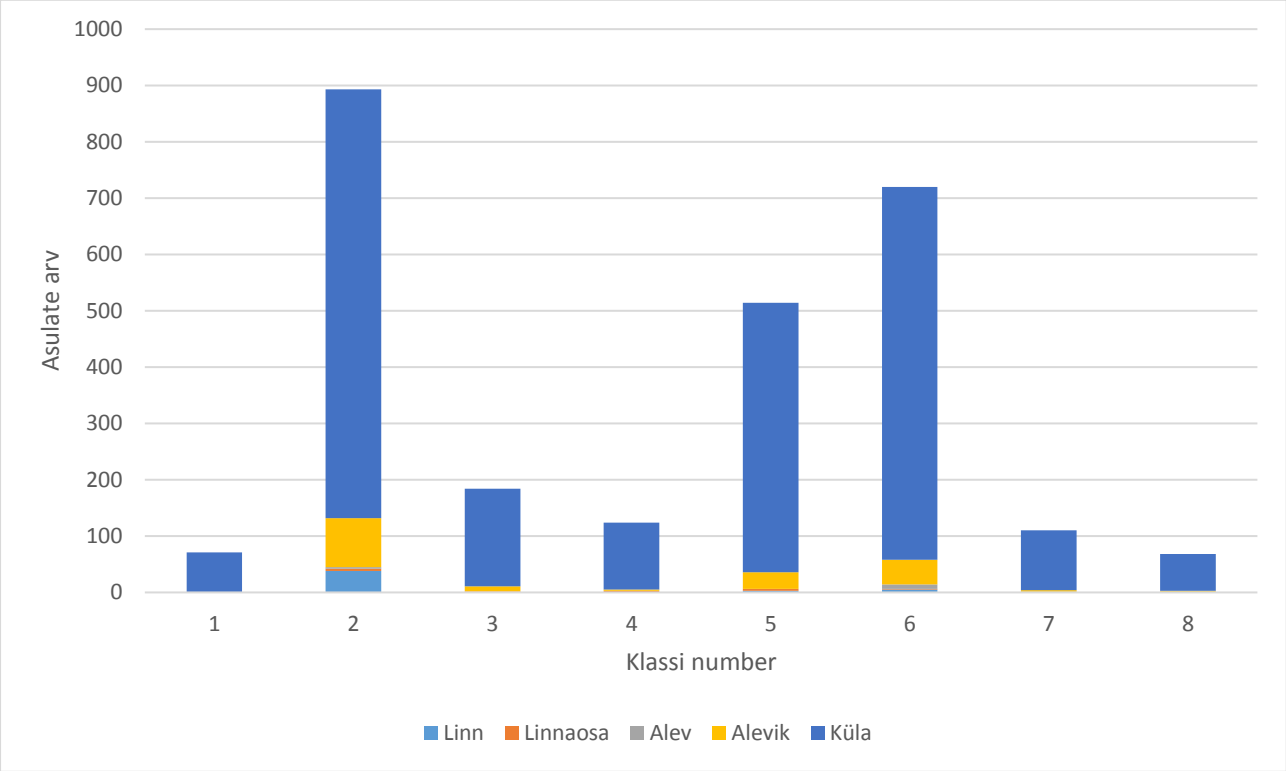


Joonis 10. Suurimat hulka asulat kajastanud artikleid sisaldavad k-keskmise klasteranalüüsil tuvastatud klassid ja 10 suurima TF*IDF väärtusega tunnust (aluskaart: Maa-amet)

K-keskmise klasteranalüüsiga asulate automaatne klassifitseerimine võimaldab tuvastada sarnase sisuga artiklite esinemise erinevates asulates, kuid meetodi suurimaks puudujäägiks, mis on ühtlasi kogu automaatse teksti klassifitseerimise problemaatilisemaks aspektiks, on leitud klassidele sisulise tähenduse andmine – võimalik on küll tuvastada enim mõju avaldavad tunnused, kuid üksikute sõnade põhjal on tuvastatud klasside ühildamine kindla temaatikaga komplitseeritud. Olenemata asjaolust, et klasside suurima TF*IDF väärtusega tunnuste hulgas leidub mitmeid korduvaid sõnu, nagu näiteks *„Tallinn“*, *„saama“* ja *„aasta“*, mis raskendavad klassidele sisulise tähenduse andmist, ilmneb mõnelgi juhul kindlale sisule viitavaid märksõnu – näiteks klass number

7 viitab kaudselt spordiga seotud temaatikale ja klass number 5 liiklusele. Klasside temaatiline kuuluvus ei selgine ka kaks korda suurema hulga oluliste tunnuste vaatlemisel.

K-keskmise klasteranalüüsi puhul on kesksel kohal klasside arv, mis tuleb enne analüüsi algust kindlaks määrata. Kuna nii suure andmestiku puhul oli äärmiselt keeruline ette ennustada reaalset klasside arvu, vajaks arvu määramine reaalsust paremini kajastavate tulemuste saavutamiseks põhjalikumat analüüsi ja ka sel juhul ei saaks olla kindel, et loodud klassid vastaksid loodetud raamistikule. Nagu ilmneb jooniselt 11, erines leitud klassidesse määratud asustusüksuste arv märgatavalt – klass number 2 domineeris kokku 893 asulas, samas kui klassides 1 ja 8 oli alla 100 asula.



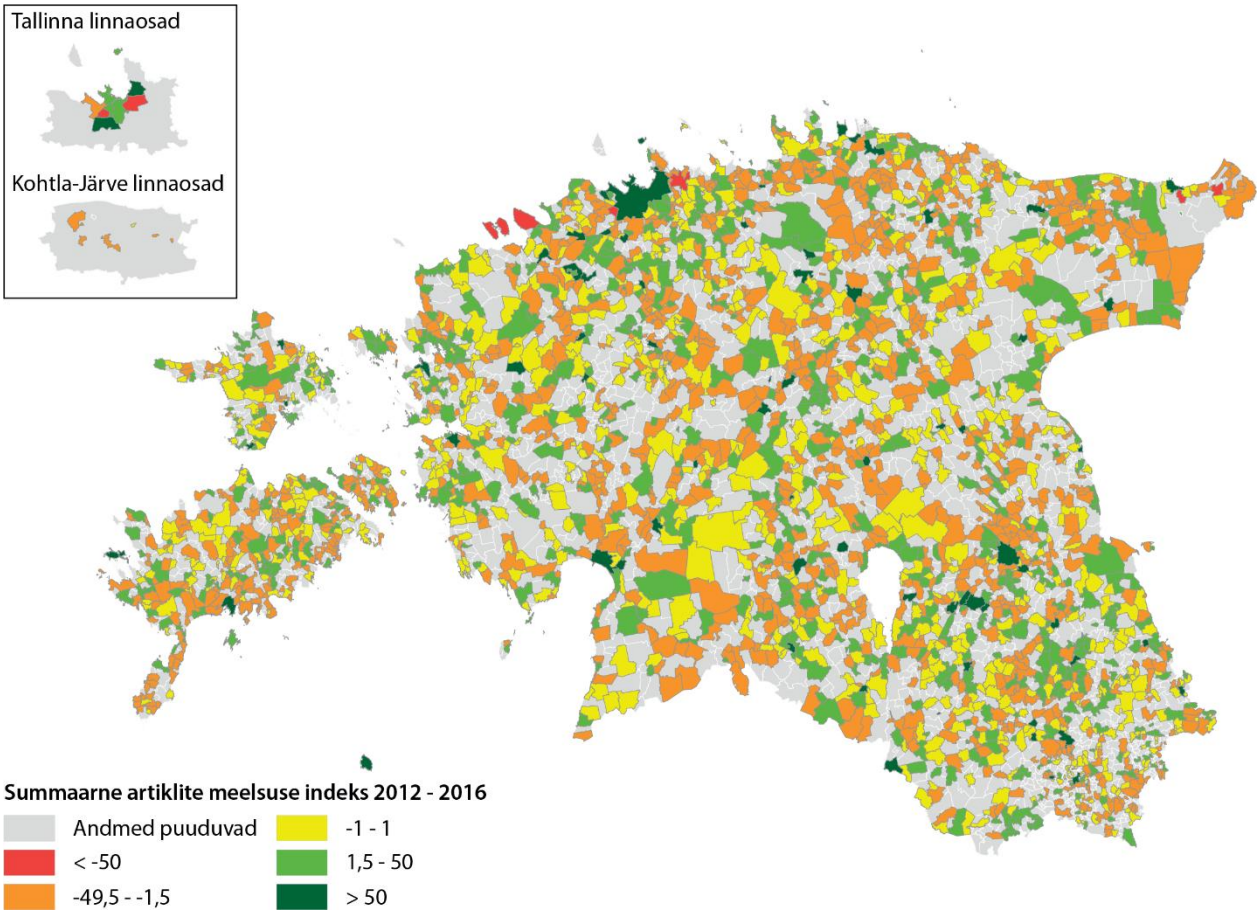
Joonis 11. Asustusüksuste arv, kus domineeris k-keskmise klasteranalüüsi meetodil tuvastatud klassidest vastava numbriga üksus.

Üheks automaatse klassifitseerimise tulemust paremini tõlgendatavaks muutmise meetodiks oleks eelnevalt klassidesse jaotatud ehk treenitud andmekogu uue mudeli ülesehitamiseks kasutamine. Sellisel moel oleks klassid ja nende olulised tunnused kasutaja poolt eelnevalt määratud ning seega tulemus paremas vastavuses loogilise klassilise kuuluvusega. Selliste meetodite suurimaks probleemiks on piisavalt esindusliku näidisandmebaasi defineerimise semantiline keerukus ja

ajamahukus. Treenitud andmestiku põhist klassifitseerimist kasutas näiteks läbiviidud meelsusanalüüs. Lisaks näidisandmete kasutamisele, suurendaks klassifitseerimise sisukust keerukamate valdkondade modelleerimise (ingl *Topic modeling*) algoritmide kasutamine, mis ei vaja analüüsiks etteantud klasside arvu ega defineeritud andmestikku. Sellisteks meetoditeks on näiteks afiinse levimise põhine klasterdamine (ingl *Affinity propagation*) ja Dirichlet'i peitlahutus (Sarkar, 2016). Keerukate modelleerimisalgoritmide puhul on probleemiks eelkõige asjaolu, et nende läbiviimiseks vajaliku äärmiselt suure arvutusliku võimekuse tõttu on need tihtipeale tavakasutajale mahuka andmestiku puhul kättesaamatud. Lisaks võib problemaatiliseks osutuda väga suur loodavate klasside arv, seda eriti varieeruva temaatikaga andmestiku, nagu on uudisartiklid, kasutamise puhul.

3.4 Meelsusanalüüs

Uudisartiklite põhjal loodud asukohakuvandi meelestatuse väljaselgitamiseks viidi läbi meelsusanalüüs, mille tulemusena leitud vahemikus -1 kuni 1 varieeruv indeks esmalt summeeriti, iseloomustades seeläbi kogu vaadeldava ajavahemiku jooksul kumuleerunud meelsust väljendavaid väärtuseid. Kokku tuvastati negatiivne meediakajastus 1136 asustusüksusel, mis moodustas kõikidest vaadeldavas ajaperioodis mainitud asulatest 42%. Vastavad näitajad olid neutraalse kajastuse osas 308 ja 12% ning positiivne meelestatus ilmnes 1240 korral, mis moodustas 46% kõikidest tuvastatud asulatest (Joonis 12).



Joonis 12. Aastatel 2012-2016 portaalis Delfi.ee avaldatud artiklitest tuvastatud asulate summaarne meelsuse indeks (aluskaart: Maa-amet)

Indeksi summeerimine soodustas tugevalt negatiivsete või positiivsete väärtuste väljakujunemist suure mainimiste arvuga asustusüksustes. Seda ilmestas ka asjaolu, et nullist enim erinevad väärtused olid suuresti koondunud linnadesse. Selline seaduspära esines tunduvalt selgemalt

positiivse kajastuse puhul, samas kui negatiivne meelsus oli ühtlasemalt jaotunud. Suurima summaarse meelsusindeksi väärtusega asulaks osutus Tallinna linn, millele järgnesid Tartu ja Pärnu (tabel 4). Väiksematest asustusüksustest omasid suurimat meediakajastuse summaarse indeksi väärtust Nõmme linnaosa, mis oli meelsuse pingereas 8., Saku alevik 6., Kohila alev 18. ja Padari küla 25. kohal.

Tabel 4. Suurima summaarse meediakajastuse indeksiga asustusüksused

	Asula nimi	**Summaarne indeks**
1.	Tallinna linn	6770.5
2.	Tartu linn	4928,0
3.	Pärnu linn	2335,5
4.	Rakvere linn	1466,5
5.	Viljandi linn	1103,5
6.	Saku alevik	905,5
7.	Rapla linn	765,0
8.	Nõmme linnaosa	726,5
9.	Haapsalu linn	599,0
10.	Otepää linn	493,0

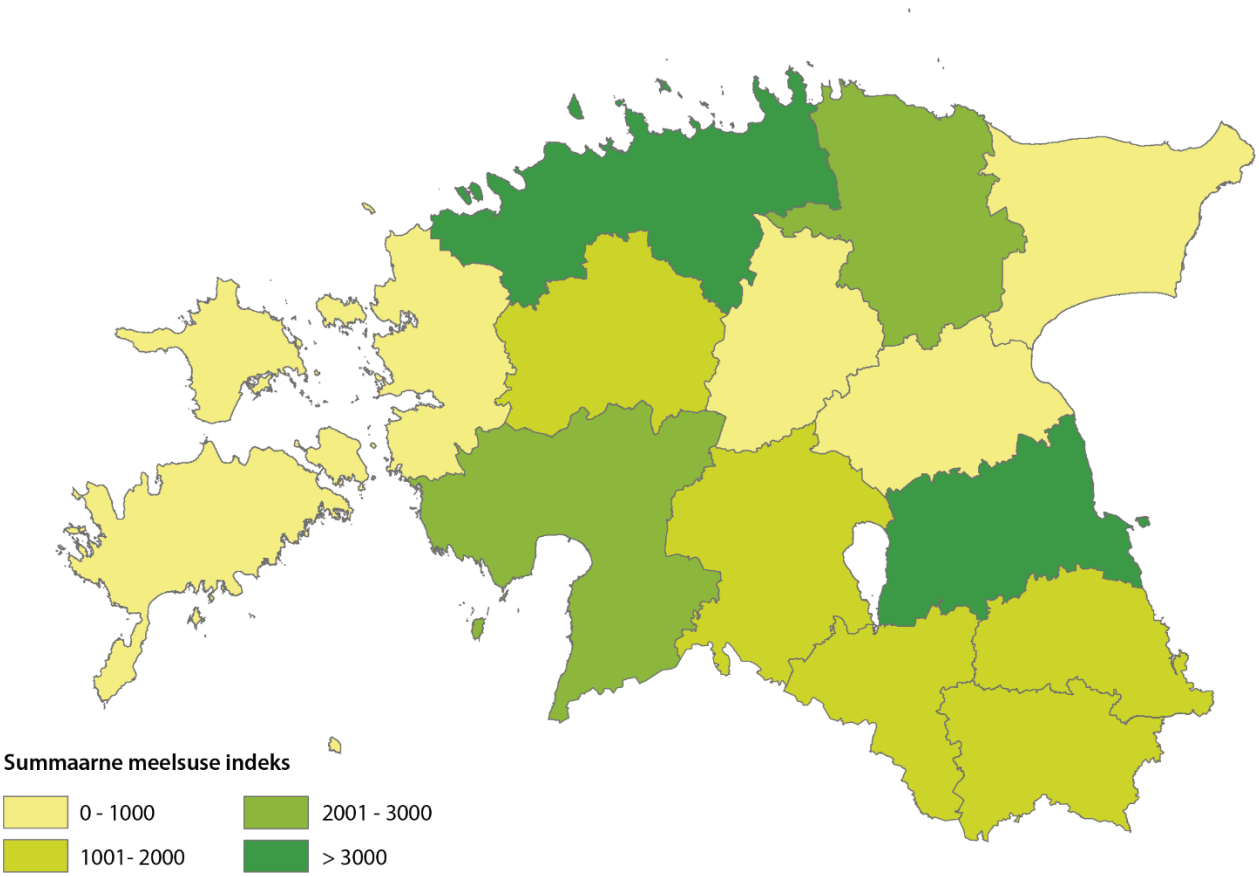
Kõige väiksema summaarse indeksi väärtusega ehk kõige negatiivsema meediakajastusega asustusüksuseks kujunes vaadeldavas perioodis Lasnamäe linnaosa, millele järgnesid Mustamäe linnaosa ja Paldiski linn (tabel 5).

Tabel 5. Kõige väiksema summaarse meediakajastuse indeksiga asustusüksused

	Asula nimi	**Summaarne indeks**
1.	Lasnamäe linnaosa	-260,0
2.	Mustamäe linnaosa	-163,0
3.	Paldiski linn	-147,5
4.	Harku alevik	-118,5
5.	Laagna küla	-92,0
6.	Vaivara küla	-78,0
7.	Maardu linn	-64,5

8.	Allikmaa küla	-48,5
9.	Narva linn	-45,0
10.	Halinga küla	-36,0

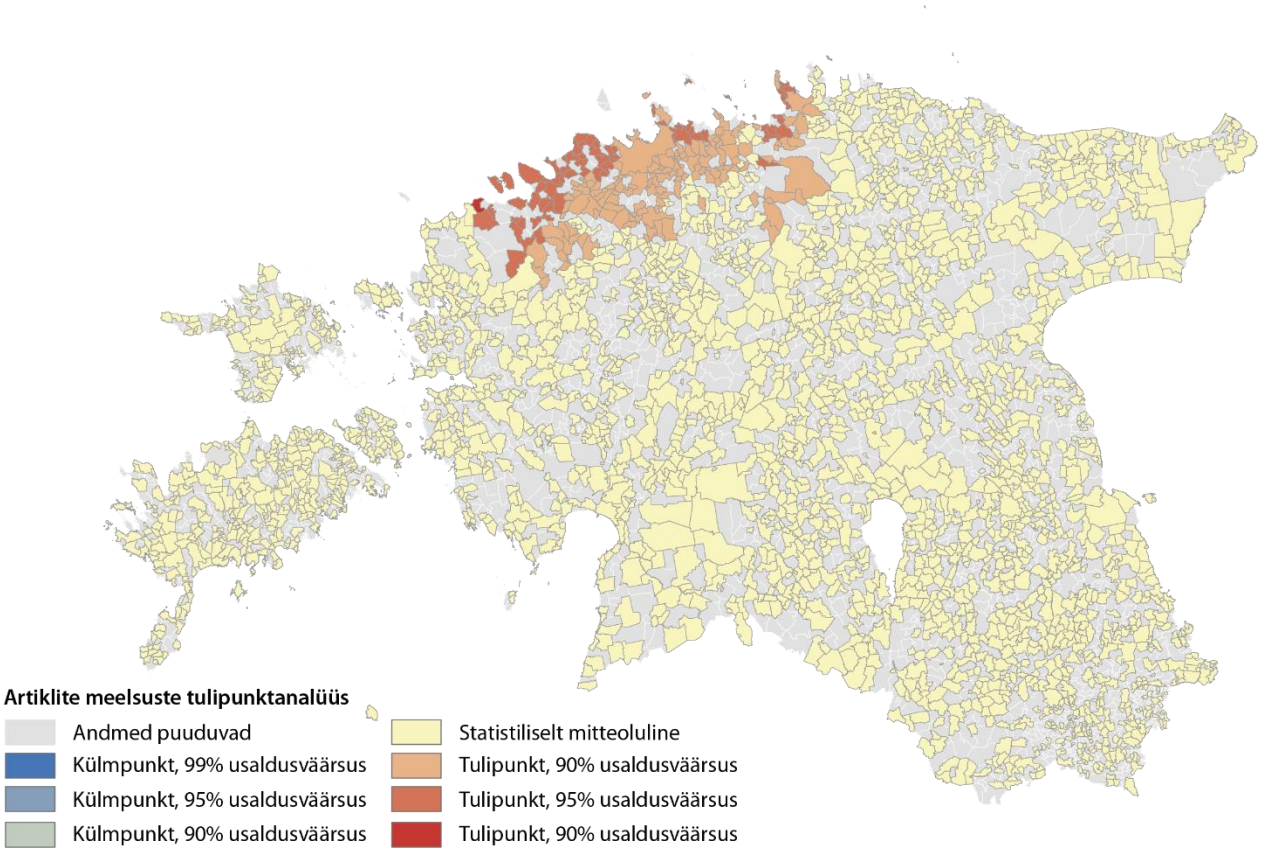
Summeerides asulate meelsusindeksid maakonniti, ilmnes suurim väärtus Harju- ja Tartumaal, samas kui kõige madalam oli nimetatud väärtus Saare ja Ida-Viru maakondades (Joonis 13). Siinkohal tuleb tähele panna, et tegu oli asulate meelsuse summeeritud tulemusega ja arvesse ei võetud haldusüksuste ega maakondade mainimisi.



Joonis 13. Aastatel 2012-2016 portaalis Delfi.ee avaldatud artiklitest tuvastatud asulate summaarne meelsuse indeks maakonniti (aluskaart: Maa-amet)

Ruumilise autokorrelatsiooni analüüsi tulemused summeeritud meelsuste indeksite osas kinnitasid nullhüpoteesi (p = 0,63), mille kohaselt väärtuste ruumilist paiknemist ei ole võimalik juhuslikust jaotumisest statistiliselt usaldusväärselt eristada. Olenemata selgete ruumiliste mustrite puudumisest, moodustus, sarnaselt tuvastatud mainimiste arvuga, summeeritud meediakajastuse

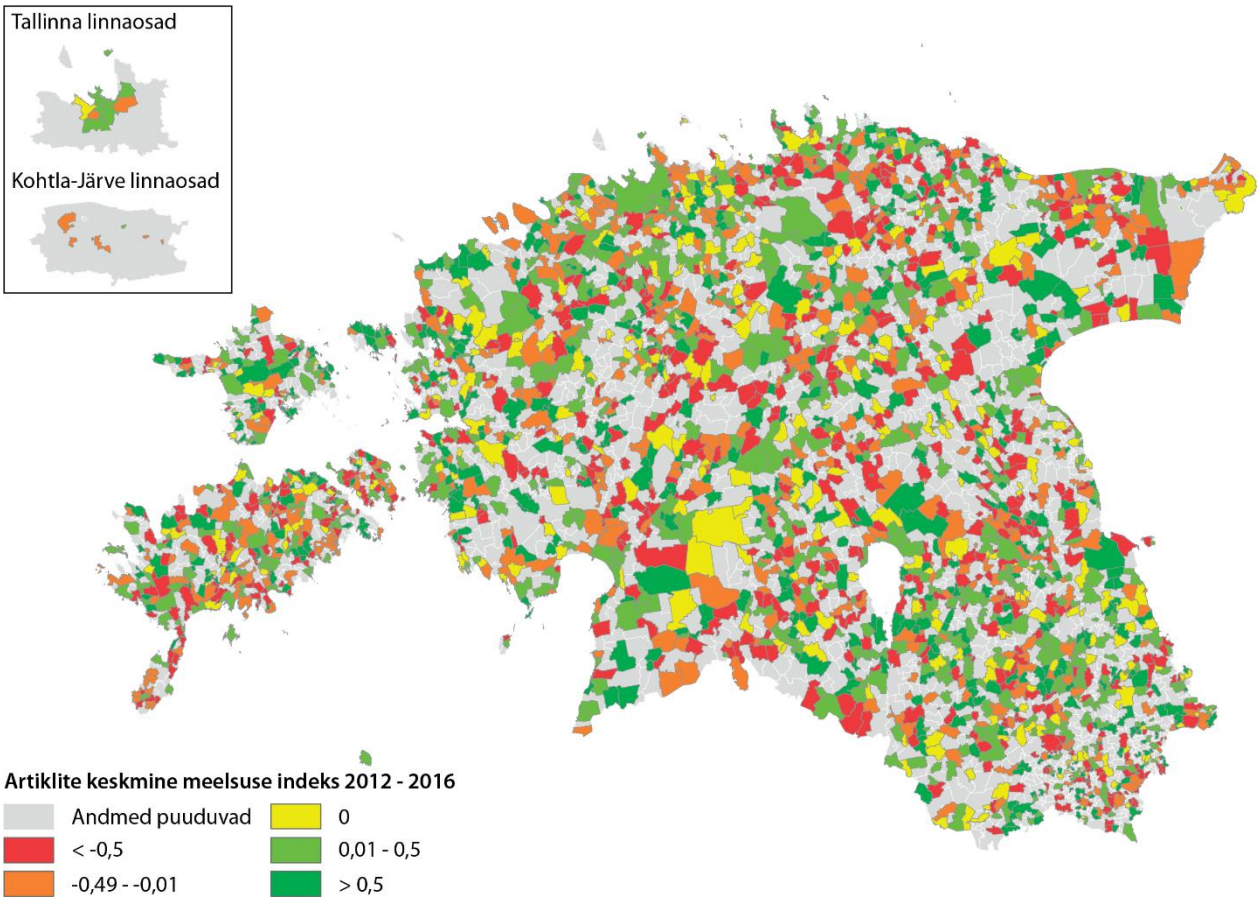
meelsuse tulipunkt Tallinna lähiümbrusesse ning külmpunkte ei tuvastatud. See tõestab kajastuste arvu tugevat mõju analüüsi tulemustele – Tallinn ja selle linnaosade väga suur mainimisete arv domineerib ka summaarse meelsuse puhul, samas kui ülejäänud Eestis on tulemused ühtlaselt jaotunud ning sarnaste väärtuste koondumist ei tuvastatud (Joonis 14).



Joonis 14. Aastatel 2012-2016 portaalis Delfi.ee avaldatud artiklitest tuvastatud asulate summaarse meelsuse indeksi tulipunktanalüüs (aluskaart: Maa-amet)

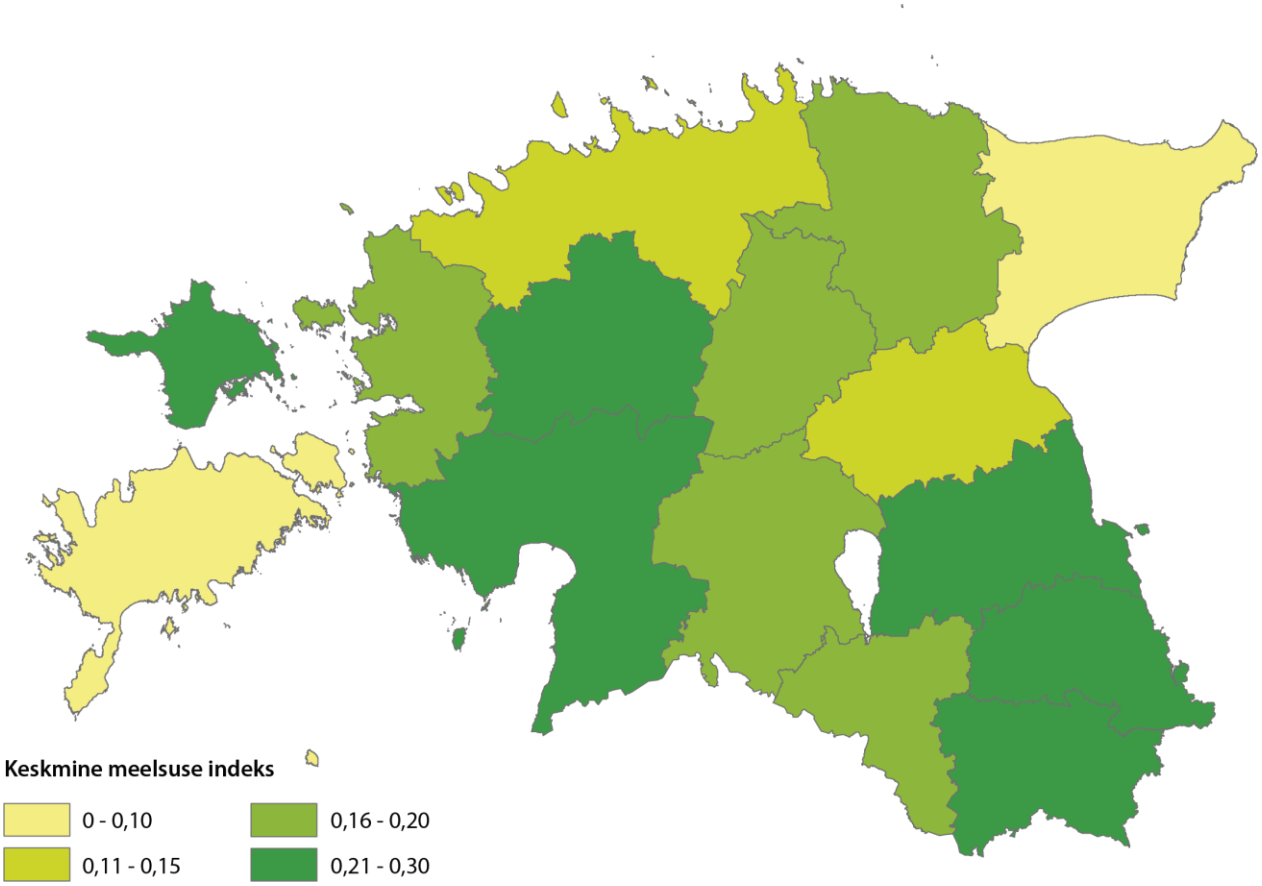
Lisaks meelsusanalüüsi indeksi summeerimisele, leiti ka asustusüksust kajastavate artiklite keskmine meelestatuse väärtus, kus 1 ja -1 lähenevad väärtused näitasid vastavalt tugevalt positiivset või negatiivset meelestatust ning indeksid, mis jäid vahemikku -0,5 kuni 0,5 esindasid neutraalset või mõõdukalt kumbagi meelsusesse kalduvat sisu (Joonis 15). Sellisel moel vähenes suure mainimiste koguarvuga asustusüksuste domineerimine vähem kajastatud asulate üle. Keskmise väärtuse arvestamine meediakajastuse meelsuse hindamisel tingib, võrreldes eelnevalt kasutatud meetodiga, vastupidise probleemi ilmnemise, kus väga kõrged või väga madalad väärtused omistati asulatele, mida oli mainitud ainult ühe korra. Kokku määrati väärtus 1, mis oli

ka kõrgeim võimalik keskmist artiklite meelsust väljendav väärtus, 366 asustusüksusele ja väärtus -1, mis väljendab minimaalset keskmise indeksi väärtust, 494 asulale. Kuna keskmise väärtuse nullist maksimaalselt erinevate väärtuste ilmnemist soodustab madal mainimiste arv, moodustasid vastava väärtusega asulate grupi väiksemad asustusüksused – 856 korral oli tegu külaga ja ülejäänud 4 juhul alevikuga, mida keskmiselt mainiti 1,7 artiklis.



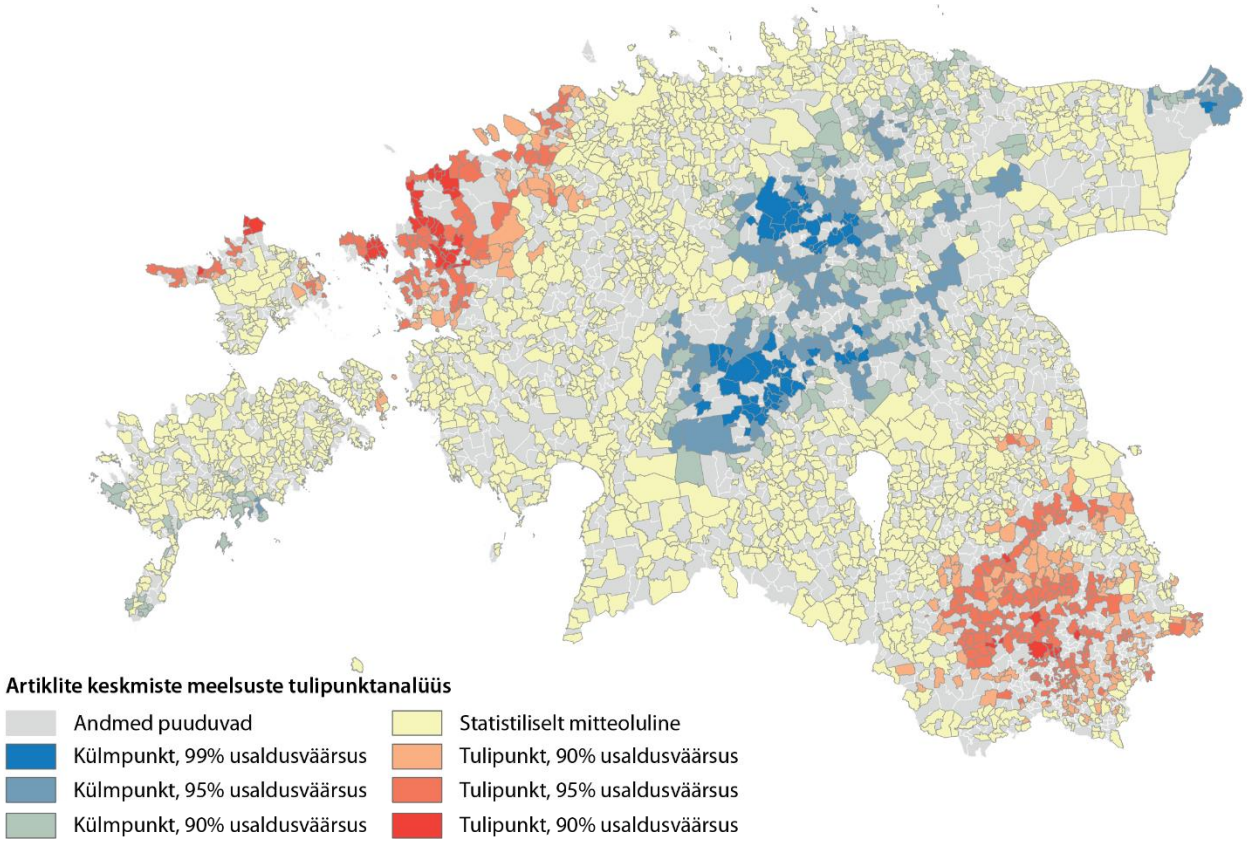
Joonis 15. Aastatel 2012-2016 portaalis Delfi.ee avaldatud uudistest tuvastatud asulate keskmine meelsuse indeks (aluskaart: Maa-amet)

Sarnaselt summaarse meelsuse indeksi visualiseerimisele, leiti ka maakondade kaupa ka keskmised artiklite meelsused. Suurima keskmise meelsuse indeksiga artiklid kajastasid asulaid, mis asuvad Hiiu ja Rapla maakondades (0,28 ja 0,23) ning väikseimad, kuid siiski nõrgalt positiivsed keskmised meelsused kujunesid Saare ja Ida-Viru maakondades (vastavalt 0,05 ja 0,03) (Joonis 16).



Joonis 16. Aastatel 2012-2016 portaalis Delfi.ee avaldatud uudistest tuvastatud asulate keskmine meelsuse indeks maakonniti (aluskaart: Maa-amet)

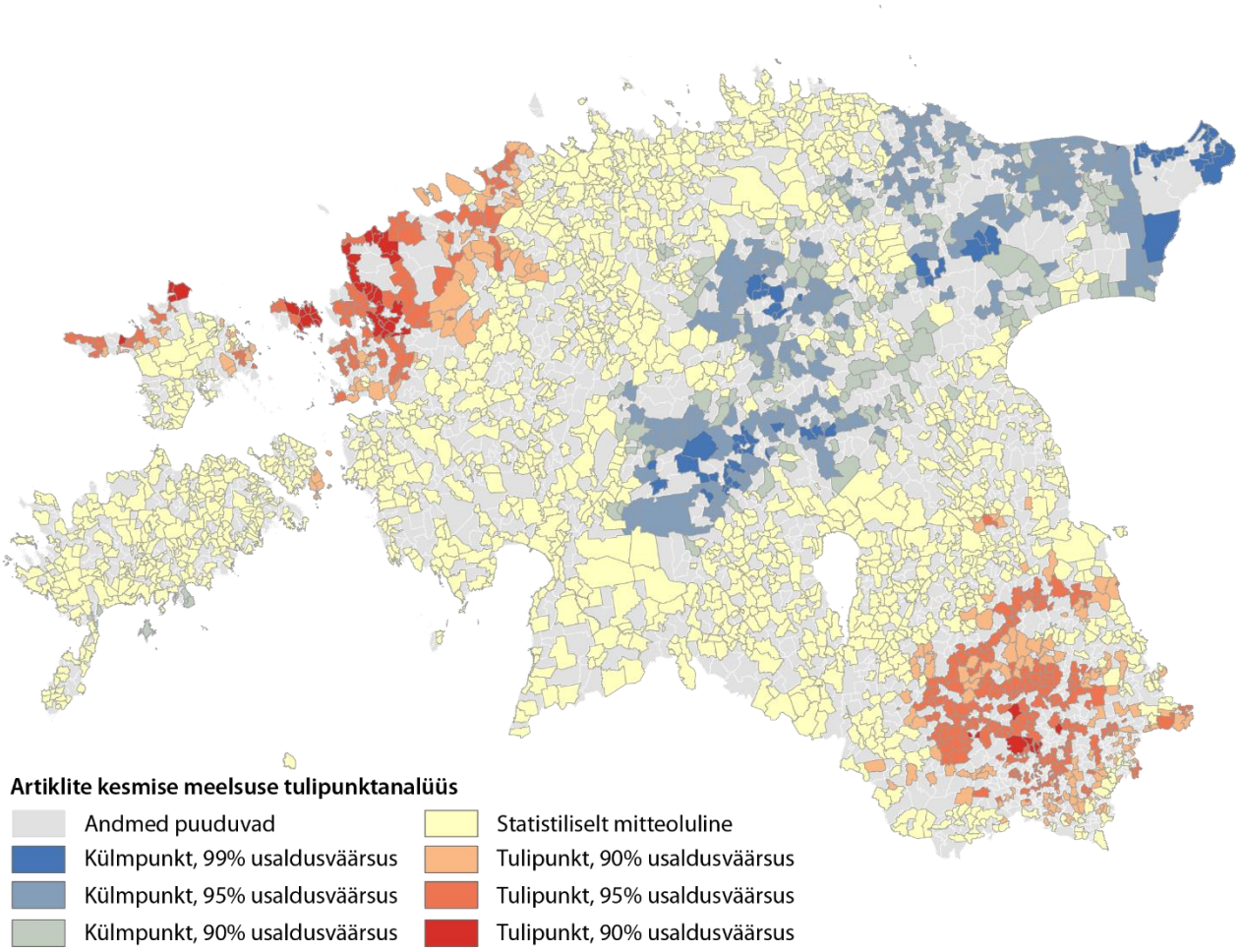
Erinevalt summaarse indeksi ruumilise autokorrelatsiooni analüüsi tulemustest, leiti keskmise meelsuse puhul andmete selge koondumine ($p < 0,01$ ja $z = 21,47$). Kuna keskmiste väärtuste jaotumine erineb selgelt summaarsete väärtuste omast, erinesid ka tuli- ja külmpunktide asukohad. Keskmiste väärtuste tulipunktid ehk tugevalt positiivsete meelsusindeksite koondumiskohad, moodustusid Eesti loode ja kagu osasse, samas kui madalamad väärtused koondusid külmpunktideks Kesk- ja Kirde-Eestis (Joonis 17).



Joonis 17. Aastatel 2012-2016 portaalis Delfi.ee avaldatud uudistest tuvastatud asulate keskmise meelsuse indeksi tulipunktanalüüs (aluskaart: Maa-amet)

Olenemata meediakajastuse meelsuse võrdlemisi varieeruvast jaotumisest, eristus selgelt linnade positiivne kajastamine – 47 linnast olid vastavad näitajad negatiivsed ainult 12 juhul. Nendeks linnadeks olid lisaks eelnevalt väljatoodud Paldiski, Maardu ja Narva linnadele Kunda, Loksa, Võhma, Mustvee, Abja-Paluoja, Antsla, Suure-Jaani, Narva-Jõesuu ja Tapa linnad. Tugevalt positiivne oli meelsusindeks enamiku maakonnakeskuste puhul, olles madalaim, kuid siiski nõrgalt positiivne, Jõhvis.

Lähtuvalt peatükis 3.1 väljatoodud problemaatikast asukohtade tuvastamisel, viidi läbi ka teine keskmise meelsusindeksi tulipunktanalüüs, millest olid välja jäetud asulad, mille tuvastamisel oli selgelt väga suur osakaal mõne isikunime või muu vea sagedasel esinemisel. Kokku eemaldati 20 suurima mainimiste arvuga potentsiaalselt valesti tuvastatud asulat (lisad 2 ja 3). Suuremate muutustena, võrreldes esialgse keskmise indeksi paiknemise tulipunktanalüüsiga, saab välja tuua tulipunktide statistilise usaldusväärsuse vähese tõusu, madala statistilise usaldusväärsusega külmpunkti kadumise Saaremaal ja külmpunkti laienemise Ida-Virumaal (Joonis 18).

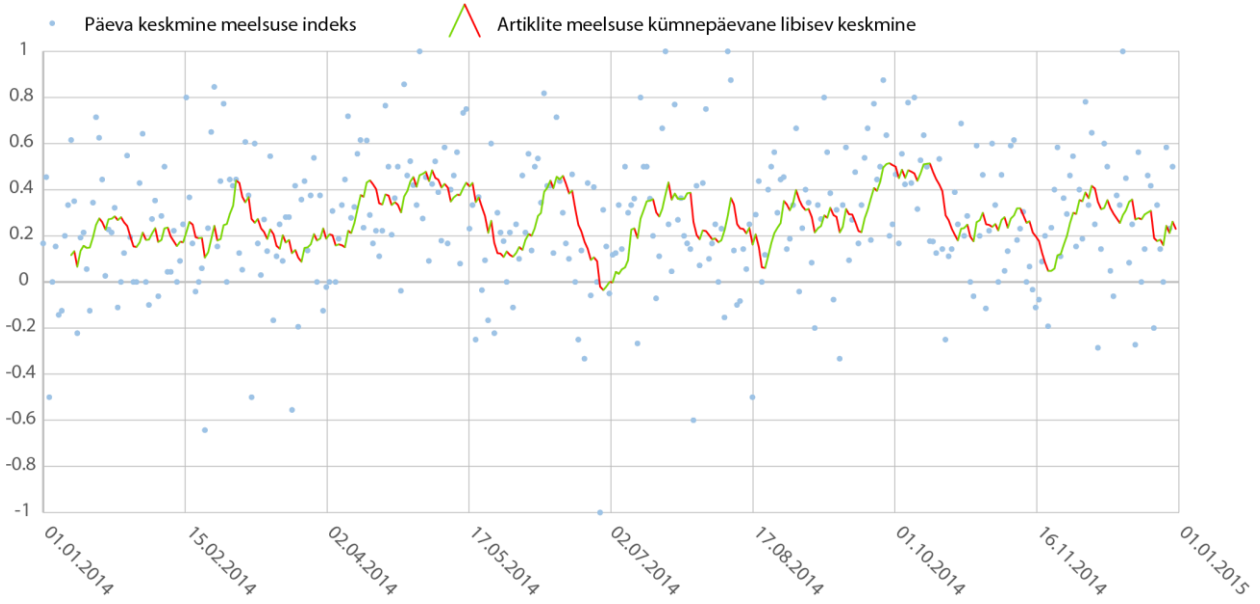


Joonis 18. Aastatel 2012-2016 portaalis Delfi.ee avaldatud uudistest tuvastatud asulate parandatud andmestiku keskmise meelsuse indeksi tulipunktanalüüs (aluskaart: Maa-amet)

Meelsusanalüüsi metoodikat rajades lähtuti EKI poolt eelnevalt defineeritud andmestikust ja väljatöötatud algoritmidest ning seega oli töö autoril meetodi tulemuse täpsusele omapoolse hinnangu andmine raskendatud. Vea hindamise keerukust suurendas ka teksti meelsuse tuvastamiseks vajaliku lingvistilise pädevuse puudumine, mis ilmnes selgelt olukordades, kus eristada tuli negatiivsuse ja positiivsuse erinevaid astmeid. Sellest lähtuvalt tuleb meetodit hinnates piirduda EKI spetsialistide poolt antud veahinnanguga, mis pakkus meetodile maksimaalselt 70-90% täpsust (H. Pajupuu, Altrov ja J. Pajupuu, 2016). Erinevalt eelnevalt väljatöötatud tekstide meelsust tuvastavale programmile, teisendati tulemused numbrilisteks väärtusteks magistritöö metoodika loomisel. Numbriliste väärtuste põhjal järelduste tegemisel oli äärmiselt oluline valida sobivaim üldistamise meetod, mille valikut raskendas asustusüksuste kajastamiste hulga suur erinevus. Näidisandmete võrdlemiseks kasutati kahte erinevat, tugevalt kajastamiste arvust sõltuvat

meetodit – summaarse meelsuse indeksi puhul oli äärmiselt keeruline võrrelda linnades saavutatud tulemust väiksemate asustusüksuste omaga ja sellisel moel domineeris Tallinn väga selgelt, samas kui keskmise meelestatuse arvutamisel saavutati tugevamalt nullist erinevad tulemused just väga väikese kajastatuse arvuga asulates. Eelpool nimetatud probleeme võimaldab vältida näiteks uuritavatele asukohtadele minimaalse mainimiste arvu määramine. Lisaks on paremini kõrvutavate tulemuste saavutamiseks otstarbekas analüüsida tulemusi asustusüksuste tüüpide või üksikute asulate kaupa.

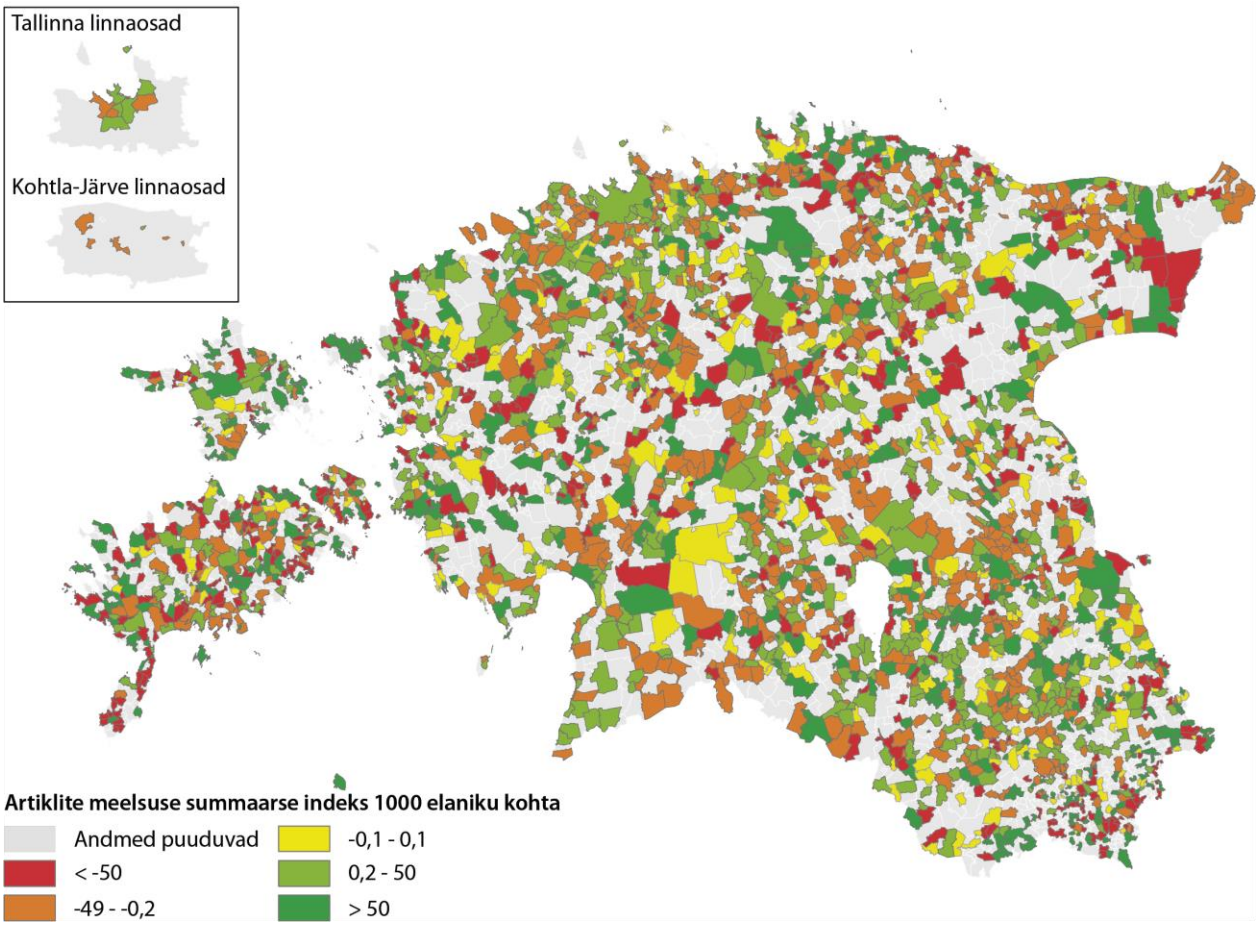
Üksikute asulate meediakajastuse meelsuse vaatlemise üheks võimaluseks on keskmiste meelsuste põhjal aegridade loomine. Aegread võimaldavad tuvastada kindlate sündmuste mõju ja selle kestvuse meediakajastusele. Selle näitlikustamiseks loodi Tartu linna 2014. aastal kajastanud artiklite põhjal keskmiste meelsuste aegrida, kus on välja toodud iga päeva keskmine indeks ja joonena, kus roheline tähistab tõusvat ja punane langevat väärtust, kümne päeva libisev keskmine (Joonis 19). Libiseva keskmise joone dünaamikat vaadeldes, avaldub suurim langus ajavahemikus 17. juuni kuni 1. juuli. Tutvudes sellel ajaperioodil avaldatud uudiste valikuga, ilmneb, seoses juuni keskel asetleidvate pidustustega, liiklusõnnetusi ja alkoholi liigtarbimist kajastavate artiklite esilekerkimine (AS Ekspress Meedia, 2018).



Joonis 19. Aastal 2014 avaldatud Tartu linna kajastavate artiklite päevane keskmine meelsuse indeks ja selle kümnepäevane libisev keskmine

Eelpool mainitud potentsiaalsete lahenduste kõrval, võimaldab suurt sõltuvust avaldatud artiklite

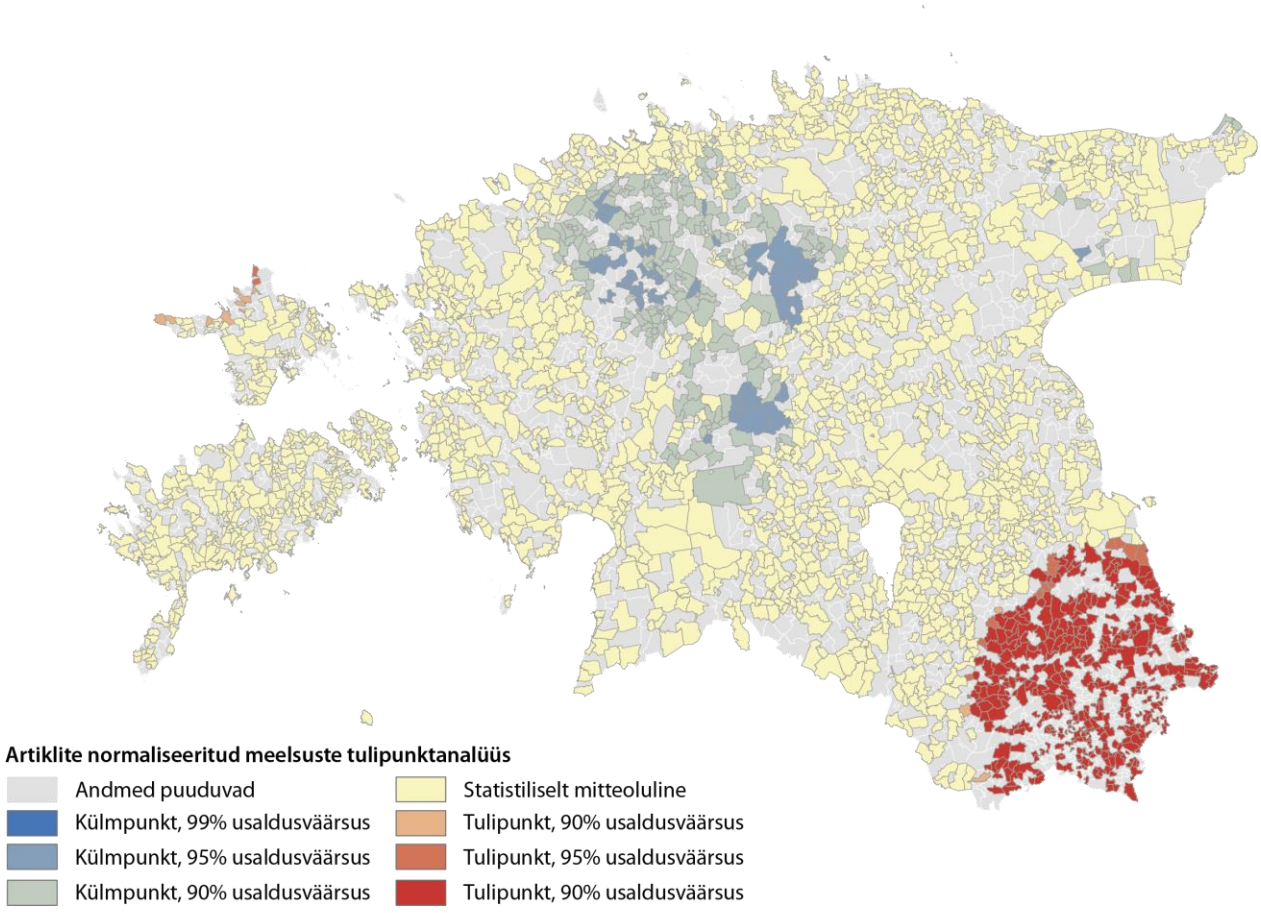
arvust vähendada tulemuste normeerimine. Käesoleval juhul kasutati summaarse meelsuse indeksi normeerimiseks asulate rahvaarvu 2011. aastal läbiviidud rahva ja eluruumide loenduse seisuga (statistikaamet, 2013). Võrreldes summaarse väärtusega, tõstab meelsuse normeerimine rahvaarvuga eelkõige väiksema rahvaarvu ja seega vähemate mainimiste arvuga asulate indeksi väärtust, samas kui linnade domineerimine mõnevõrra väheneb (Joonis 20).



Joonis 20. Aastatel 2012-2016 portaalis Delfi.ee avaldatud uudistest tuvastatud asulate rahvaarvuga normeeritud summaarne meelsusindeks (aluskaart: Maa-amet)

Sarnaselt keskmise meelsuse indeksiga, näitab ka normeeritud tulemuste ruumiline autokorrelatsioon selget koondumist ($p < 0.01$ ja $z = 7.48$). Meelsusindeksi ruumilise jaotumise tulipunktanalüüsi tulemuste visualiseerimisel ilmneb mõningane sarnasus keskmiste väärtuste paiknemisega. Rahvaarvuga normeeritud indeksite puhul moodustub tugev tulipunkt Kagu-Eestisse ja väiksema pindala ning statistilise usaldusväärsusega tulipunkt Hiiumaa looderannikule.

Külmpunkt, mis jääb 90-95% usaldusväärsuse vahemikku, katab suure osa Kesk-Eestist ja osa Ida-Virumaast (Joonis 21).



Joonis 21. Aastatel 2012-2016 portaalis Delfi.ee avaldatud uudistest tuvastatud asulate rahvaarvuga normeeritud summaarse meelsuse indeksi tulipunktanalüüs (aluskaart: Maa-amet)

Leitud oluliste sõnade ja meelsusanalüüsi tulemuste vahel selge seos puudub – olgugi, et väga madala meelsuse indeksiga paikade oluliste sõnade hulgas leidub ka selgelt negatiivse tähendusega sõnu, domineerivad pigem neutraalse iseloomuga või endas emotsiooni mitte kandvad väljendid. Ainsana eristuvad tuvastatud negatiivse meelusega paikades selgelt õnnetusjuhtumitega seostatavad terminid, nagu näiteks *„põleng“*, *„põlema“*, *„kannatanu“*, *„häirekeskus“*, *„politsei“* ja *„päästja“*. Lisaks ilmnevad madalamat meelsust väljendava näitajaga asulates kaudselt liiklusintsidentidega seostatavad sõnad: *„maantee“*, *„tänav“*, *„tee“* ja mitmed erinevad automargid. Veelgi keerulisem on tuvastada seost positiivse meelsuse ja oluliste sõnade vahel – mitmel juhul on tegemist näiteks teiste kohanimede või asustusüksuste tüüpidega. Ainsa tuvastava

temaatilise klassina oli märgata spordiga seotus terminite ja nimede, näiteks *„fc“*, *„flora“*, *„serviti“* ning *„sportlane“*, sagedasemat esinemist just kõrgema meelsust väljendava indeksiga asulate puhul. Vähene tuvastatav seos asukohtade meediakajastuse meelsuse ja leitud oluliste sõnade vahel ei osuta otseselt meetodi sobimatusele etteantud ülesande lahendamiseks. Eelnevalt kirjeldatud analüüside tulemuste paremaks ühildamiseks oleks vajalik lisada rohkem läbiviidava uurimuse iseloomust tulenevaid täpsustavaid parameetreid, näiteks sõna liigi osas.

Selgemini esineb seos meelsusanalüüsi tulemuste ja leitud oluliste omadussõnade vahel. Kõrgeima võimaliku keskmise meelsusindeksiga paikade hulgas esineb sagedasti selgelt positiivse suunitlusega sõnu, nagu näiteks „*hea*“, „*ilus*“, „*vabatahtlik*“, „*noor*“, „*iseseisev*“, „*tugev*“, „*elav*“ ja „*imekaunis*“. Lisaks leidub sellistes kohtades mitmeid loodusega seotud termineid: „*kadakane*“, „*looduslik*“, „*keskkonnasõbralik*“ ja „*metsamajanduslik*“. Negatiivse keskmise meelsusindeksiga asulate puhul domineerisid eelkõige neutraalse sisuga sõnad, kuid leidus ka mõningaid selgelt negatiivseid termineid, nagu näiteks „*haige*“, „*tüse*“, „*julm*“ ja „*kurb*“. Väga sarnased on seosed ka normeeritud andmestiku puhul. Mõnevõrra keerulisem on konkreetseid seoseid tuvastada oluliste omadussõnade ja summeeritud meelsuse vahel. Olgugi, et selle meelsuse iseloomustamise võtte juures on märgata eelnevalt esitatud ilminguid, on tulemused märksa mitmekesisemad. Kuna keskmise meelsuse puhul omistati maksimaalsed ja minimaalsed väärtused just väga väikese mainimiste arvuga kohtadele, võib järeldada, et analüüside tulemused on märksa paremini võrreldavad väikese andmestiku puhul, samas kui väga suure tekstikoguse juures muutuvad seosed nõrgemaks.

Selleks, et võrrelda klassifitseerimise tulemusi meelsusanalüüsi omadega, leiti iga tuvastatud klassi puhul sinna kuuluvate asulate meelsuse indeksite standardhälve ja võrreldi seda koguvalimi omaga, eeldades, et sarnase sisuga artiklitel on tuvastatud ka sarnane meelsus. Selle tulemusel tuvastati, et kuuel juhul kaheksast vähenes summaarse meelsusindeksi standardhälve märgatavalt, samas kui kahes klassis see kasvas. Võttes sarnasel analüüsil aluseks keskmise meelsuse, oli standardhälbe erinevus tunduvalt väiksem, olles koguvalimi näitajast suurem kolmel ja märgatavalt väiksem ainult ühel juhul. Ülejäänud neljas klassis oli leitud standardhälve väga sarnane koguvalimi omaga. Kahe analüüsimeetodi võrdlemisel tuleb lisaks metodoloogilisele erinevusele arvesse võtta ka asjaolu, et sarnane temaatika ei tähenda ilmtingimata sarnast meelsust – samas valdkonnas on võimalik nii negatiivsete kui ka positiivsete artiklite avaldamine.

Lähtuvalt väljatoodud meetodite võrdlusele, tuleb tõdeda, et erinevate analüüside tulemuste võrdlemisel ilmnes selgeid seoseid vähestel juhtudel. Eelkõige tingis selle algoritmide metoodiliste tagapõhjade olemuslik erinevus. Samuti raskendas võrdlust automaatse tekstianalüüsi suurima probleemi, milleks on tulemuste sidumine kontekstiga, ilmnemine – seda eriti täisautomaatse klasteranalüüsi ja oluliste sõnade leidmise puhul. Nagu selgus oluliste omadussõnade võrdlemisel meelsusanalüüsiga, tekkisid selgemad seosed väiksema hulga tekstide töötlemisel. Seega võib järeldada, et selgemalt seotud tulemuste saamiseks oleks vajalik vähendada uuritavat andmehulka, jaotades näiteks vaadeldava ajaperioodi etappideks. Olenemata raskustest meetodite kõrvutamisel, saab sooritatud analüüsi lugeda suuresti õnnestunuks – ainsa meetodina ei väljastanud selgelt defineeritavat tulemust k-keskmise klasteranalüüs. Sellest tulenevalt, on soovitatav tulevaste tööde puhul valida klassifitseerimiseks mõni sobivam meetod. Olgugi, et oluliste sõnade tuvastamise ja meelsusanalüüsi tulemuste väljatoomisel esines mõningaid raskusi, võib eraldiseisvana need õnnestunuks ja asulate meediakuvandit iseloomustavaks lugeda.

Kokkuvõte

Magistritöö eesmärgiks oli, kasutades programmeerimiskeelt *Python*, siduda näidisandmetena kasutatavad uudisartiklid nendes kajastatud asulatega ja lokaliseerimisele järgneva analüüsi ning visualiseerimine käigus tuvastada artiklite põhjal meedias loodud Eesti asulate kuvand. Näidisandmestiku moodustasid aastatel 2012 – 2016 uudisteportaalis Delfi.ee avaldatud artiklid, mida oli vaadeldavas perioodis kokku 385 819. Tekstiandmete automaatne töötlemine on masinõppe arengus olnud aastaid olulisel kohal, kuid geograafiliste uurimuste osas on seda kasutatud vähe. Sellise metoodika loomine võimaldab edaspidi geograafiliste uurimuste läbiviimisel kasutada tunduvalt suuremamahulisemat andmestikku, kui manuaalse töötlemise puhul oleks võimalik.

Artiklite automaatsele asukohapõhisele analüüsile eelnes nende seostamine kindlate asukohtadega. Selleks loodi programm, mis kasutas eestikeelse teksti töötlemise mooduli EstNLTK nimeliste üksuste tuvastamise töövahendeid. Eraldi käsitleti tähenduslike nimedega kohti, mis defineeriti eelnevalt Eesti Keele Instituudi poolt koostatud, eestikeelseid sõnavorme sisaldava andmestiku põhjal. Tuvastamise järgselt valideeriti asulad lähtuvalt geograafilisest kontekstist või selle puudumisest. Asukohtade tuvastamise käigus leiti artiklitest 2621 unikaalset kohanime, sealhulgas 47 linna, 13 linnaosa, 12 alevit, 177 alevikku ja 2372 küla, kattes sellega 56% kõikidest Eesti asulatest. Enim mainiti Tallinna linna (50 394 artiklit), millele järgnesid Tartu (23 684 artiklit) ja Pärnu (10 188 artiklit). Leitud asukohti sisaldavate artiklite manuaalse kontrolli käigus tuvastati lokaliseerimise täpsuseks 87%, olles ülekaalukalt suurim linnade (97%) ja linnaosade (95%) puhul. Madalaim oli täpsus külade puhul, kus korrektselt oli tuvastatud vaid 52% kontrollitud juhtudest. Enim esines veana muude geograafiliste objektide ja isikunimede tuvastamist asulanimena.

Tekstide lokaliseerimise järgselt viidi läbi nende asukohapõhine analüüs. Esmalt kasutati tunnuste leidmiseks kasutatavat TF*IDF mudelit oluliste sõnade tuvastamiseks. Seejärel viidi läbi k-keskmise klasteranalüüs, leidmaks kohad, kus vaadeldavas ajavahemikus domineerisid sarnase sisuga artiklid. Viimase analüüsina teostati meelsusanalüüs, kasutades selleks Eesti Keele Instituudi poolt väljaarendatud, multinomiaalse Bayens'i klassifikaatori meetodil põhinevat meelsuse tuvastamise programmi modifitseeritud töövahendit, mille tulemused teisendati hilisemalt numbrilisteks väärtusteks. Oluliste sõnade tuvastamise puhul tuleb eelkõige arvestada olulisuse definitsiooniga, millest tulenevalt ei pruugi matemaatiliselt oluline termin endas suurt

hulka informatsiooni kanda. K-keskmise klasteranalüüsi meetodil automaatse klassifitseerimise suurimaks probleemiks osutus klassidele sisulise definitsiooni leidmine – võimalik oli küll tuvastada klassifitseerimisele enim mõju avaldavad tunnused, kuid neid oli keeruline kindla kontekstiga siduda. Meelsusanalüüsi problemaatilisemaks aspektiks kujunes leitud meelsusindeksite põhjal järelduste tegemine, mis oli tingitud suurtest erinevustest asulate mainimiste arvus – summaarse indeksi puhul domineerisid ülekaalukalt linnad, samas kui keskmise väärtuse puhul oli ekstreemumitele lähenevad väärtused väheste mainimiste arvuga kohtades. Tulemuste sõltuvuse vähendamiseks mainimiste arvust, kasutatu indeksi rahvaarvuga normeerimist.

Meetodite omavahelisel võrdlusel selgus, et erinevate metoodiliste tagapõhjadega algoritmide tulemusi on omavahel äärmiselt keeruline võrrelda. Võrdluses meelsusanalüüsiga leiti, et k-keskmise meetodil läbiviidud klassifitseerimise tulemusel võis märgata kohatist sarnase meelsusega kohtade koondumist samadesse klassidesse, kuid ilmnes ka vastupidiseid näiteid. Varieeruva tulemuse andis ka oluliste sõnade võrdlemine meelsusanalüüsi tulemustega – olgugi, et kohati oli võimalik tuvastada seos meelsuse väärtuse ja olulistes sõnades sisalduva emotsiooni vahel, oli enamuses nende koosesinemine pigem olemuselt juhuslik. Selgemalt ilmnesid seosed asukohakuvandi meelestatusega omadussõnade puhul, seda eriti kohtades, kus mainimiste arv oli väiksem. Sellest tulenevalt saab väita, et paremini võrreldavate tulemuste saavutamiseks, oleks otstarbekas analüüsitavat andmehulka vähendada või näiteks ajaperioodide kaupa osadeks jaotada. Olenemata meetodite vähesest ühildumisest, võib metoodika rakendamise lugeda õnnestunuks – kui välja arvata k-keskmise klasteranalüüs, on eraldiseisvana saavutatud tulemused asukohakuvandi väljaselgitamisel asjakohased. Seda eriti, kui arvestada asjaolu, et töö läbiviimisel keskenduti meetodite loomisele ja meediaanalüüsi spetsiifikasse ei süvenetud.

Scripting Tools for Text Localization and Location Base Analysis of Text Data by Example of Identifying Estonian Settlements Image of Media.

Ott Koik

Summary

The aim of this thesis was to use the programming language Python to link automatically the news articles used as sample data to the settlements mentioned in them and after localization to carry out location based analysis to identify the image of media of Estonian settlements. The sample data set was formed from articles published by Estonian online news provider Delfi.ee during the time period of 2012-2016, which totalled 385 819 articles. Despite the fact that automatic processing of textual data has been an essential part of the development of machine learning, there has been insufficient use for that in geographical research. Creating such a methodology will allow geographic research, in particular in the field of human geography, to work with large data while manual processing would be impossible due to vast time consumption.

Prior location based text analysis, the articles were associated with certain Estonian settlements. For location recognition, a program based on named entity recognition was created. After identification, the veracity of meaningful names was checked based on the geographical context or its absence in the same sentence the location was recognized. Repetitive names were validated likewise. With location recognition and validation algorithms, 2621 unique place names were found, including 47 cities, 13 districts, 12 townships, 177 boroughs and 2372 villages, covering 56% of all Estonian settlements. The most mentioned settlement was Tallinn (50 394 articles), followed by Tartu (23 684 articles) and Pärnu (10 188 articles). Identifying place names could be considered successful for larger settlements: manual validation for 1% of settlements identified, overall accuracy of almost 90% was detected, as it was the highest for towns (97%), while for the villages, the accuracy was only 52%. The biggest problem was toponyms that are identical to persons' surnames. This problem is also indicated by the fact that villages named Purga and Padari were recognized in large volumes. In addition to person names, the presence of settlements with names identical to other geographical features, such as rivers or islands, increased the error as well.

After standardization, which main purpose was to separate and lemmatize words, the most characteristic features were extracted using TF*IDF method. This method determined the

significance of the words based on their frequency, attributing the highest value to the words that are present in a large quantity in a certain article, but are not found in many articles. The TF*IDF method was also used to extract meaningful words. In identifying key words, the definition of significance must be taken into account – mathematically important words do not always have a significant intuitive importance. Consequently, identifying key words can be considered an appropriate method for creating a variety of visualization of data, while making conclusions based on it is complicated.

Subsequent to identification of features, k-means clustering was used to identify settlements where similar content of articles was dominant over a period of time. With automated classification, the biggest problem was to define the meaning of classes. The results of the k-means clustering analysis was also greatly influenced by the fact that the number of classes must be predefined. As a last method, a sentiment analysis was performed using a modified version of the program developed by the Institute of the Estonian Language. Using sentiment analysis, the most problematic aspect was finding the proper method for making conclusions based on sentiment index, as the number of mentioning was extremely varying and therefore summing up indexes caused overwhelmingly big differences between cities and villages, while the mean values are near-extremity in places with fewer identified mentions.

A comparison between the methods revealed that the results of algorithms with different methodological backgrounds were extremely difficult to compare. By comparing k-means clustering results with the sentiment analysis, it was found that classification could result in locating places with similar sentiment in the same classes at some occasions, but there were also opposing examples. The result of the key word identification also provided a large variation of the words and the connection with sentiment analysis was difficult to find – although, occasionally it was possible to establish a link between the value of the identified sentiment and the emotion contained in the key words, in most cases their coexistence was inherently random. The links appeared more clearly when adjectives were used as important words, especially in places where the number of references was smaller. Regardless of the low compatibility of the methods, the implementation of the methodology can be considered successful – apart from the k-means cluster analysis, the results, especially when observed as a stand-alone characteristics, were relevant in determining the image of location.

Tänuavaldused

Suur tänu minu juhendajatele, Raivo Aunapile ja Alexander Kmochile, huvipakkuva, uuendusliku ja isiklikus plaanis väga arendava teemapüstituse ning igakülgse toe eest magistritöö koostamisel. Lisaks soovin tänada kolleege ja tööandjat Regio OÜ'st, kes võimaldasid mulle nendepoolse riist- ja tarkvara kasutamist olukorras, kus mul seda hädasti vaja oli. Lõpetuseks soovin tänada ka lähedasi, kes töö valmimist moraalse toe ja vajalike näpunäidetega toetasid.

Kasutatud allikad

Aedmaa, Eleri (2016). Eesti keele ühendverbide kompositsionaalsuse määramine. Eesti Rakenduslingvistika Ühingu aastaraamat, 12, 5−23.

Agarwal, P. (2005). Ontological considerations in GIScience. International Journal of Geographical Information Science, 19(5), 501–536.

Aggarwal, C. C., Zhai, C. X (2012). A Survey of Text Classification Algorithms. Mining Text Data, 163-223.

Bing, L. (2012). Sentiment Analysis and Opinion Mining.

Bracewell, D. B., Jiajun, Y., Fuji, R., Shingo, K. (2009). Category Classification and Topic Discovery of Japanese and English News Articles. Electronic Notes in Theoretical Computer Science 225, 51–65.

Bruni, E., Tran, N. K., Baroni, M. (2014). Multimodal Distributional Semantics. Journal of Artificial Intelligence Research, 49, 1-47.

Cambria, E., White, B. (2014). Jumping NLP Curves: A Review of Natural Language Processing Research. IEEE Computational Intelligence Magazine, 9(2), 48-57.

Chomsky, N. (1956). Three Models for the Description of Language.

Crompton, J. L. (1979). An Assessment of the Image of Mexico as a Vacation Destination and the Influence of Geographical Location Upon That Image. Annals of Travel Research vol. 17, no. 4, 18-23.

Fürst, S., Schönhagen, P., Bosshart, S. (2015). Mass communication is more than a one-way street: on the persistent function and relevance of journalism. Javnost - The Public, 22:4, 328-344.

Govers, R., Go, G., Kumar, K. (2007). Promoting Tourism Destination Image. Journal of Travel Research 46 (1), 15–23.

Gruber, T. R. (1993). A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5(2), 199-220.

Hutchins, W.J. (2004). The Georgetown-IBM experiment demonstrated in January 1954. Machine Translation: From Real Users to Research, Proceedings, 3265, 102-114.

Jing, J. (2012). Information Extraction From Text. Mining Text Data, 11-41.

Jones, C.B., Purves, R.S. (2008). Geographical information retrieval. International Journal of Geographical Information Science 22, 219–228.

Kaalep, H.-J. (1997). An Estonian Morphological Analyser and the Impact of a Corpus on Its Development. Computers and the Humanities 31(2), 115-133.

Kaalep, H.-J., Vaino, T. (2001). Complete morphological analysis in the linguist's toolbox. Congressus Nonus Internationalis Fenno-Ugristarum Pars V, 9–16.

Kaasik, K. (2014). Turismisihtkoha turvalisuse roll külastuselamuse kujunemisel. Magistritöö, Tartu Ülikool, loodus- ja tehnoloogiateaduskond, ökoloogia ja maateaduse instituut, geograafia osakond.

Kadiyala, A., Kumar, A. (2017). Applications of Python to Evaluate Environmental Data Science Problems. Environmental Progress & Sustainable Energy, 36(6), 1580-1586 .

Kennedy, H. (2014). Perspectives on Sentiment Analysis. Journal of Broadcasting & Electronic Media, 56(4), 435-450.

Kasenõmm, K. (2014). Ajakirjanduse muutuvad funktsioonid uudistest üleküllastunud ühiskonnas. Magistritöö, Tartu Ülikool, sotsiaal- ja haridusteaduskond, ühiskonnateaduste instituut.

Kim, K., Park, O.-J., Yun, S.,Yun, H. (2017). What makes tourists feel negatively about tourism destinations? Application of hybrid text mining methodology to smart destination management. Technological Forecasting and Social Change, volume 123, 362-369.

Kmoch, A., Uuemaa, E., Klug, H., Cameron, S. G. (2018). Enhancing Location-Related Hydrogeological Knowledge. ISPRS Int. J. Geo-Inf. 7, 132.

Ko, Y. (2017). How to use negative class information for Naive Bayes classification. Information Processing & Management, 53(6), 1225-1268.

Kramp, L., Loosen, W. (2018). The Transformation of Journalism: From Changing Newsroom Cultures to a New Communicative Orientation? Transforming Communications – Studies in Cross-Media Research, 205-240.

Leskovec, J., Rajaraman, A., Ullman, J. D. (2014). Mining of Massive Datasets, 2nd Edition.

Li, H., Calder, C. A., Cressie, N. (2007). Beyond Moran's I: Testing for Spatial Dependence Based on the Spatial Autoregressive Model. Geographical Analysis, Volume 39, Issue 4, 357-375.

Liu, T., Lin, S., Chen, Z., Ma, W.-Y (2003). An Evaluation on Feature Selection for Text Clustering, ICML Conference.

MacKay, D. (2003). Chapter 20. An Example Inference Task: Clustering. Information Theory, Inference and Learning Algorithms, 284–292. Cambridge University Press.

Muhoho-Minni, P., Lubbe, B. A. (2017). The role of the media in constructing a destination image: the Kenya experience. Communicatio, 43:1, 58-79.

Nadkarni, P. M., Ohno-Machado, L., Chapman, W. W. (2011). Natural language processing: an introduction. Journal of the American Medical Informatics Association, 18(5), 544-551.

Orasmaa, S., Petmanson, T., Tkachenko, A., Laur, S., Kaalep, H.-J. (2016). EstNLTK - NLP Toolkit for Estonian. Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016).

Pajupuu, H., Altrov, R., Pajupuu, J. (2016). Identifying polarity in different text types. Folklore. Electronic Journal of Folklore, 64, 25−42.

Sabah, G. (2010). Natural Language Understanding, Where Are We Going? Where Could We Go? Computer Science, 54(9), 1505-1513.

Saks, K. (2011). The formation of editing culture and practice in Estonian newspapers 1988– 2005. Doktoritöö, Tartu Ülikool, , ajakirjanduse ja kommunikatsiooni instituut.

Sarkar, D. (2016). Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from Your Data.

Shahin, S. (2016). When Scale Meets Depth: Integrating Natural Language Processing and Textual Analysis for Studying Digital Corpora, Communication Methods and Measures, 10:1, 28-50.

Shannon, C. E., Weaver, W. (1949). The Mathematical Theory of Communication, University of Illinois Press, Urbana.

Shiliang, D., Chen, L., Junyu, C. (2016). A review of natural language processing techniques for opinion mining systems. Information Fusion, 36, 10-25.

Stepchenkova, S., Morrison, A. M. (2006). The destination image of Russia: From the online induced perspective, Tourism Management, volume 27, issue 5, 943-956

Šantić, M., Bevanda, A., Bijakšić, S. (2016). Influence of media on creation of a tourist destination image. Informatol. 49, 3-4, 180-189

Tkachenko, A. (2010). Named Entity Recognition for the Estonian Language. Magistritöö, Tartu Ülikool, matemaatika ja informaatika teaduskond, arvutiteaduste instituut.

Tkachenko, A., Petmanson, T., Laur, S. (2010). Named Entity Recognition in Estonian. Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing, 78-83.

Tsou, M.-H. (2015). Research challenges and opportunities in mapping social media and Big Data. Cartography and Geographic Information Science.

Witmer, J., Kalita, J. (2009). Extracting Geospatial Entities from Wikipedia. IEEE Third International Conference on Semantic Computing (ICSC 2009), 450-457.

Yi, J.H., Nasukawa, T., Bunescu, R., Niblack, W. (2003) Sentiment Analyzer: Extracting sentiments about a given topic using natural language processing techniques. Third IEEE International Conference On Data Mining, Proceedings, 427-434.

Yuxia, H. (2011). A Latent Semantic Analysis-Based Approach to Geographic Feature Categorization from Text. 2011 IEEE Fifth International Conference on Semantic Computing, 87-94.

Internetiallikad ja andmebaasid:

AS Eesti Meedia, uudisteportaali Delfi arhiiv (viimati vaadatud 09.05.2018), http://www.delfi.ee/archive

Eesti Keele Instituut, eesti keele vormide loend (viimati vaadatud 15.12.2017), http://www.eki.ee/tarkvara/wordlist/

Maa-amet, Maa-ameti Geoportaal, Andmed ja kaardid, Haldus- ja asustusjaotus (viimati uuendatud 15.10.2017), http://geoportaal.maaamet.ee/est/Andmed-ja-kaardid/Haldus-ja-

asustusjaotus/Haldusreformieelsed-haldus-ja-asustusjaotuse-piirid-p577.html

Statistikaamet, Eesmärgid ja Tegevused, Turism (viimati vaadatud 01.04.2017), https://www.mkm.ee/et/tegevused-eesmargid/turism

Statistikaamet, Statistika andmebaas, Rahva ja eluruumide loendus 2011, eestlaste arv ja osatähtsus elukoha (asula) järgi, 31. detsember 2011 (viimati vaadatud 09.05.2018), http://pub.stat.ee/px-web.2001/Dialog/varval.asp?ma=RL004&ti=EESTLASTE+ARV+JA+OSAT%C4HTSUS+ELUKOHA+%28ASULA%29+J%C4RGI%2C+31%2E+DETSEMBER+2011&path=../Database/Rahvaloendus/REL2011/09Rahvastiku_paiknemine/04Elukoht_ja_soo-vanusjaotus/&lang=2

Lisad

Lisa 1. Magistritöö koostamisel kasutatud moodulite ja riistvara tehnilised andmed

Magistritöös kasutatud *Pythoni* moodulid:

- estnltk 1.4
- gensim 3.2.0
- nltk 3.2.5
- numpy 1.11.3
- pandas 0.16.2
- regex 2015.7.19
- scikit-learn 0.18.1
- scrapy 1.5.0

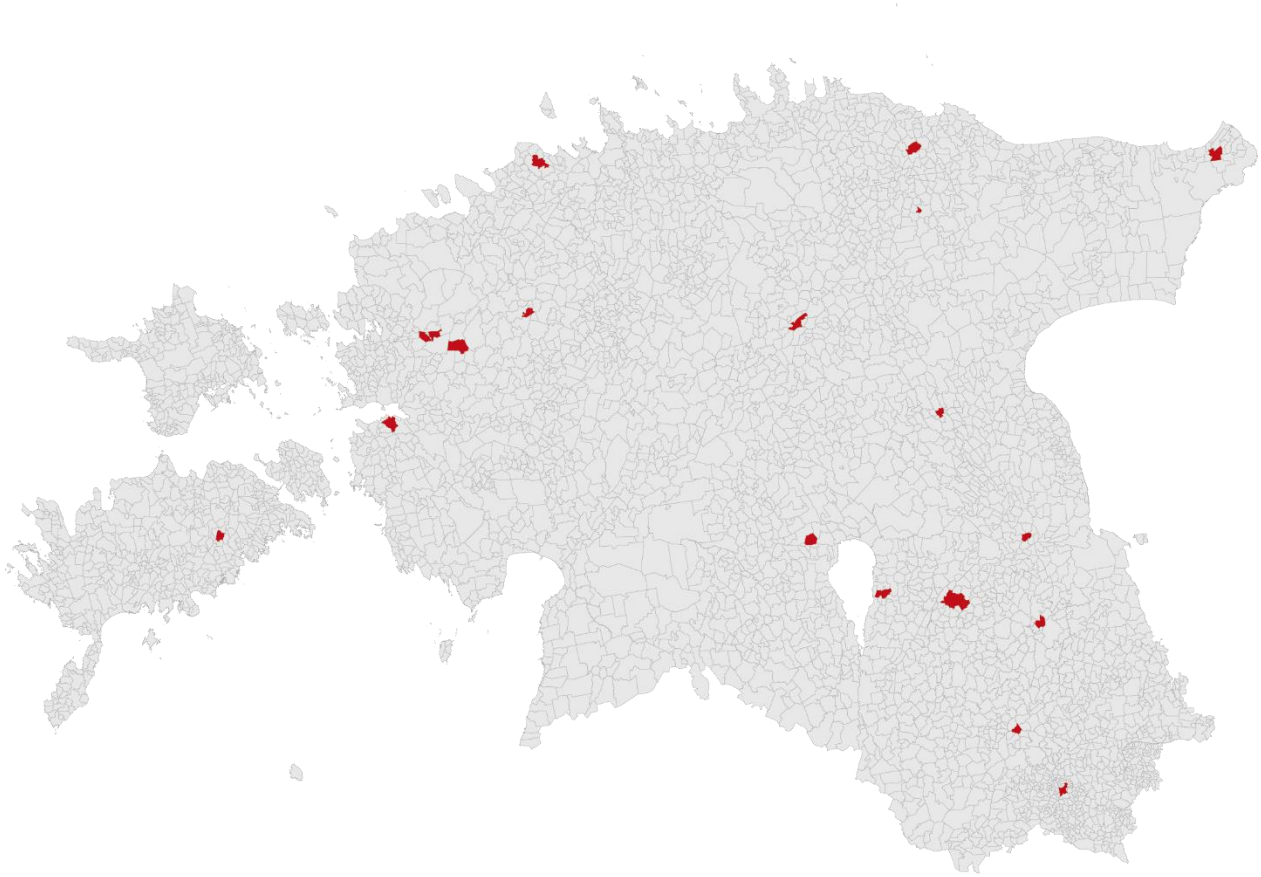
Töö läbiviimseks kasutatud arvuti tehnilised andmed:

- Protsessor – Intel(R) Core(TM) i5-4210M CPU 2.60 GHz
- Operatiivmälu – 8 GB
- Operatsioonisüsteem – Windows 10 Home, 64-bit

Lisa 2. Peatükis 3.4 läbiviidud tulipunktanalüüsi tarbeks eemaldatud potentsiaalselt valesti tuvastatud asulad ja nende karakteristikud.

Asula	**Omavalitus**	**Mainimiste arv**	**Keskmine meelsuse indeks**	**Summeeritud meelsuse indeks**	**Oluline sõna**	**Potentsiaalse vea liik**
Padari küla	Vastse-Kuuste vald	509	0,31	157,5	Tahtma	Isikunimi
Sõrve küla	Harku vald	284	-0,01	-3,0	Meeter	Muu geograafiline objekt
Haanja küla	Haanja vald	279	0,38	106,0	Sportlane	Muu geograafiline objekt
Luke küla	Nõo vald	271	0,20	53,0	Manchester	Isikunimi
Laagna küla	Vaivara vald	265	-0,35	-92,0	Tänav	Muu geograafiline objekt
Matsalu küla	Lihula vald	250	0,32	80,5	Festival	Muu geograafiline objekt
Purga küla	Märjamaa vald	247	0,42	103,0	Pärast	Isikunimi
Leila küla	Kullamaa vald	225	0,47	106,0	Koht	Isikunimi
Allikmaa küla	Lääne-Nigula vald	210	-0,23	-48,5	Aasta	Isikunimi
Visnapuu küla	Kambja vald	194	0,33	64,5	Oü	Isikunimi
Mooritsa küla	Jõgeva vald	179	0,41	73,0	Rohkem	Firma või kaubamärgi nimi
Toomla küla	Sõmeru vald	166	0,11	19,0	Päev	Isikunimi
Kakumäe küla	Vinni vald	154	0,09	13,5	Kell	Muu geograafiline objekt
Osula küla	Sõmerpalu vald	134	0,41	55,5	Palju	Isikunimi
Noorma küla	Rannu vald	110	0,71	78,5	Ülikool	Isikunimi
Kõivu küla	Luunja vald	101	0,28	28,5	Aasta	Isikunimi
Surva küla	Viljandi vald	86	0,62	53,5	Jõhvi	Isikunimi
Esna küla	Kareda vald	41	0,32	13,0	Mihhailova	Isikunimi
Esna küla	Roosna-Alliku vald	21	0,31	6,5	Mõis	Isikunimi
Veeriku küla	Valjala vald	19	0,47	9,0	Eesti	Muu geograafiline objekt

Lisa 3. Peatükis 3.4 läbiviidud tulipunktanalüüsi tarbeks eemaldatud potentsiaalselt valesti tuvastatud asulate paiknemine.



Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Ott Koik

(sünnikuupäev: 14. märts 1992)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

„Skriptimisvahendid teksti lokaliseerimiseks ja asukohapõhiseks analüüsiks Eesti asulate meediakuvandi tuvastamise näitel“, mille juhendajad on Ravio Aunap ja Alexander Kmoch,

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, 28. mai 2018