Euroopa Liit
Euroopa
Regionaalarengu Fond

Eesti
tuleviku heaks

Kuidas koguda ja analüüsida Facebooki postitusi?

Online-kogukondade hoiakute kaardistamise juhend

Sander Salvet, Tartu Ülikool

RITA-RÄNNE
projekt

RITA-RÄNNE projekt aitab välja töötada teaduslikult põhjendatud innovaatilisi lähenemisi rände ja lõimumise protsesside juhtimiseks Eestis, eesmärgiga aidata kaasa majanduse arengule ja ühiskonna sidususe suurenemisele

SISUKORD

SISSEJUHATUS **2**

1. ETTEVALMISTAVAD SAMMUD **3**

1.1. Uurijabrauseri loomine 3

1.2. Otsimootori päringu abil avalike Facebooki lehtede kaardistamine 7

1.3. Andmekogumis- ja andmeanalüüsitarkvara allalaadimine ja paigaldamine 9

2. ANDMEKOGUMINE FACEPAGERIGA **9**

2.1. JSON-formaadis eelsätete loomine 10

2.2. Andmekogumine 12

3. MEELESTATUSANALÜÜS JA TEEMA MUDELDAMINE PYTHONI JA R-IGA **15**

3.1. Eeltöötlus (Python) 16

3.1.1. Teksti puhastamine ja lemmatiseerimine 16

3.1.2. Andmestiku osadeks jagamine ja postituste käsitsi märgendamine 20

3.1.3. Teksti vektoriseerimine 20

3.2. Postituste meelestatuse tuvastamine (Python) 21

3.3. Postituste teemade tuvastamine (Python) 22

3.4. Jooniste ja tabelite tegemine, statistiline analüüs (R) 24

KASUTATUD KIRJANDUS **26**

SISSEJUHATUS

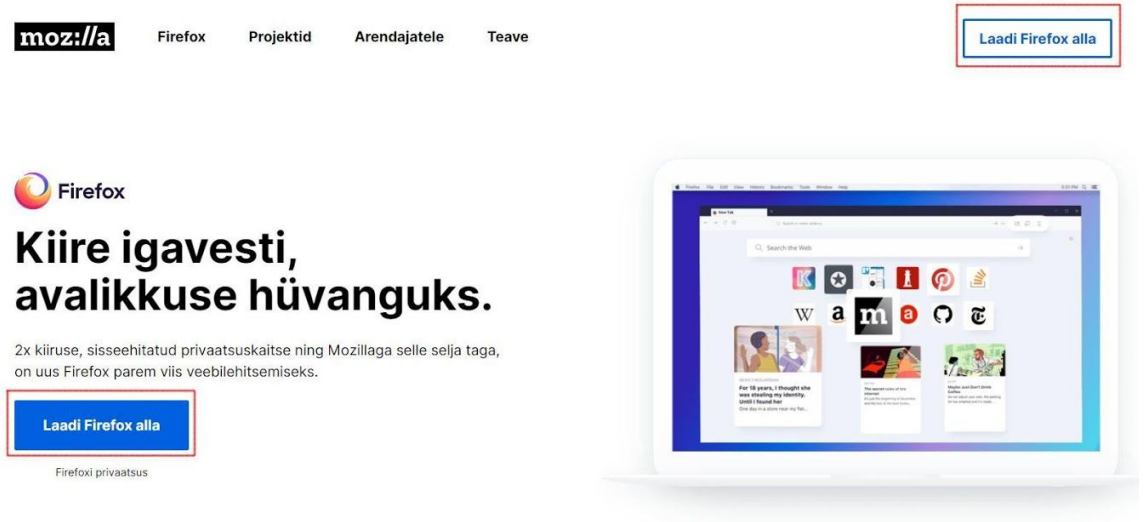
Käesolev *online*-kogukondade hoiakute kaardistamise juhend, annab praktilisi juhtnööre hoiakute uurimiseks migrantide suhtes avalike eestikeelsete Facebooki lehtede postitustes. Juhend koosneb kolmest osast: 1) ettevalmistavatest sammudest andmekogumiseks ja -analüüsiks ning 2) andmekogumise ja 3) andmeanalüüsi õpetusest. Mõned juhendis väljapakutud analüüsimeetodid, nagu meelestatusanalüüs ja teema mudeldamine, on mõeldud vaid väga suurte tekstikorpuste uurimiseks. Pisut väiksema, nt mõnesajast postitusest koosneva valimi korral on otstarbekam kasutada meelestatusanalüüsi ja teema mudeldamise asemel nt standardiseeritud kontentanalüüsi (vt Kalmus, 2015). Lisaks siinse juhendiga tutvumisele, läheb uurijal hoiakute kaardistamiseks vaja ka eelteadmisi sotsiaalteaduslike uuringute läbiviimisest (vt Sotsiaalse..., 2014), päringupõhisest uuringudisainist (vt Rogers, 2019), meelestatusanalüüsist (vt Liu, 2015) ja teema mudeldamisest (vt Alghamdi & Alfalqi, 2015) ning andmetöötlusest Pythoni (vt TÜ..., 2017) ja R-iga (vt Kolnes, 2020). Juhendi kasutamise puhul tuleb arvestada sedagi, et nii digimeetodite kui sotsiaalmeediauuringute valdkond muutub kiiresti ning mõned antud nõuannetest ei pruugi enam mõne kuu või aasta pärast kehtida.

1. ETTEVALMISTAVAD SAMMUD

Enne andmekogumist tuleb kaardistada huvipakkuvad avalikud Facebooki lehed ja tuvastada nende unikaalsed ID-koodid. Facebooki lehtede kaardistamiseks on soovitatav kasutada nt Google'i, Bingi vm otsimootri päringut. Reeglina annab otsimootori päring rohkemaid ja mitmekesisemaid tulemusi kui Facebooki enda otsing. Selleks, et vähendada otsimootori algoritmide personaliseerivat mõju tulemustele, tuleb aga uuringu jaoks ette valmistada uurijabrauser. Samuti on tarvis enne andmekogumist ja -analüüsi paigaldada arvutisse vajalik tarkvara. See peatükk annabki näpunäiteid uuringuks valmistumisel: uurijabrauseri loomiseks, otsimootori päringu abil Facebooki lehtede kaardistamiseks ning tarkvara paigaldamiseks.

1.1. Uurijabrauseri loomine

- Paigaldage Mozilla Firefoxi veebilehitseja. Mozilla Firefoxi saab alla laadida leheküljelt https://www.mozilla.org/et/ (vt joonist 1).

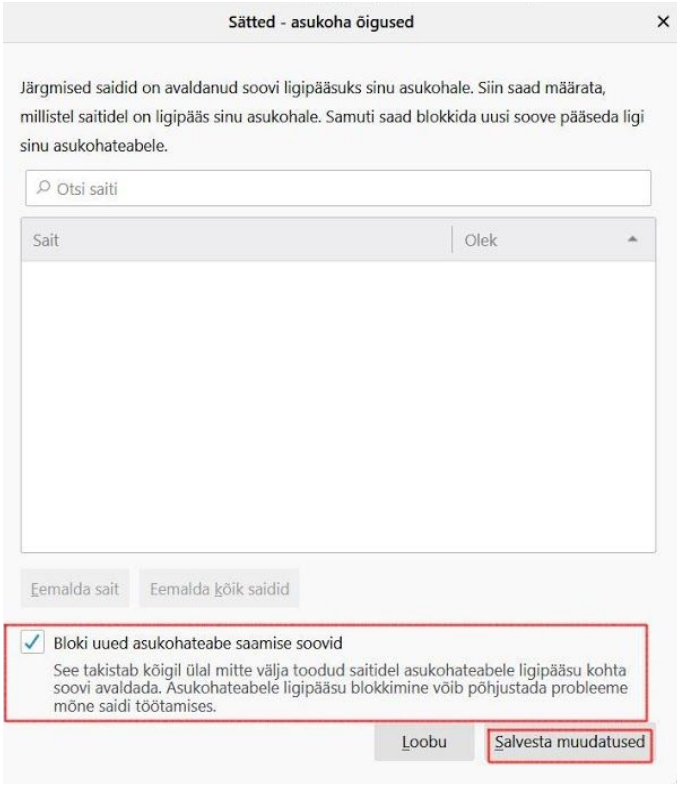


Joonis 1. Mozilla Firefoxi allalaadimine, valides *"Laadi Firefox alla"*

- Pärast Firefoxi paigaldamist muutke veebilehitseja privaatsussätteid:
  - keelake veebilehtedel teie jälgimine ("*Sätted" > "Privaatsus ja turvalisus" > "Saitidele saadetakse signaal, et sa ei soovi olla jälitatud: alati"*), vt joonist 2;

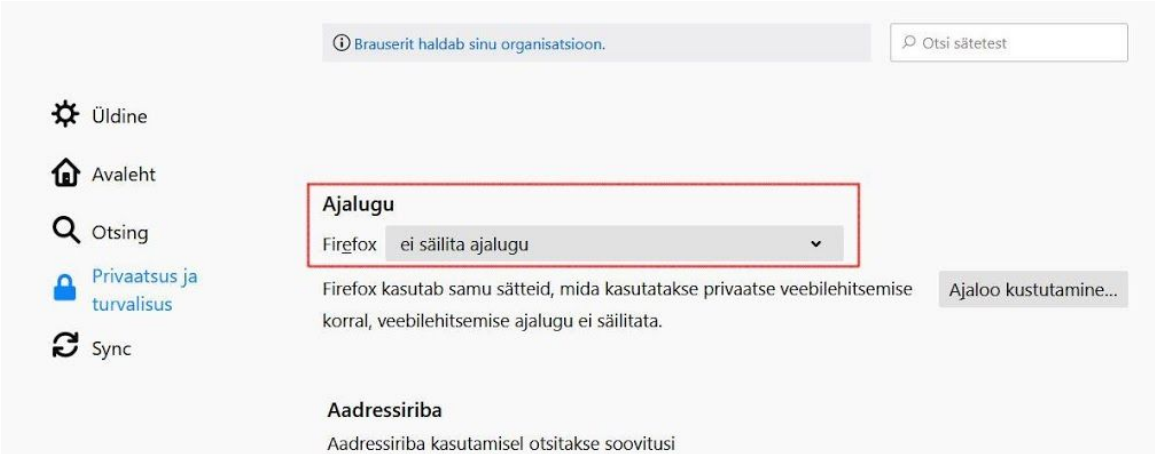
Joonis 2. Veebilehtedepoolse jälgimise keelamine

➢ lülitage välja asukohaandmete edastamine (“*Sätted” > “Privaatsus ja turvalisus” > “Asukoht” > “Sätted” > “Bloki uued asukohateabe saamise soovid”*) ja salvestage muudatused (*“Salvesta muudatused”*), vt joonist 3;



Joonis 3. Asukohaandmete edastamise väljalülitamine

➢ lülitage välja veebilehtede külastamise ajalugu (“*Sätted” > “Privaatsus ja turvalisus” > “Ajalugu” > “Firefox: ei säilita ajalugu”*; seejärel nõustuge brauseri taaskäivitamisega), vt joonist 4.



Joonis 4. Veebilehtede külastamise ajaloo väljalülitamine

- Kustutage küpsised (“*Sätted*” > *“Privaatsus ja turvalisus”* > *“Küpsised ja saidi andmed”* > *“Kustuta andmed...”*), vt joonist 5.

Joonis 5. Küpsiste ja veebilehtede andmete kustutamine

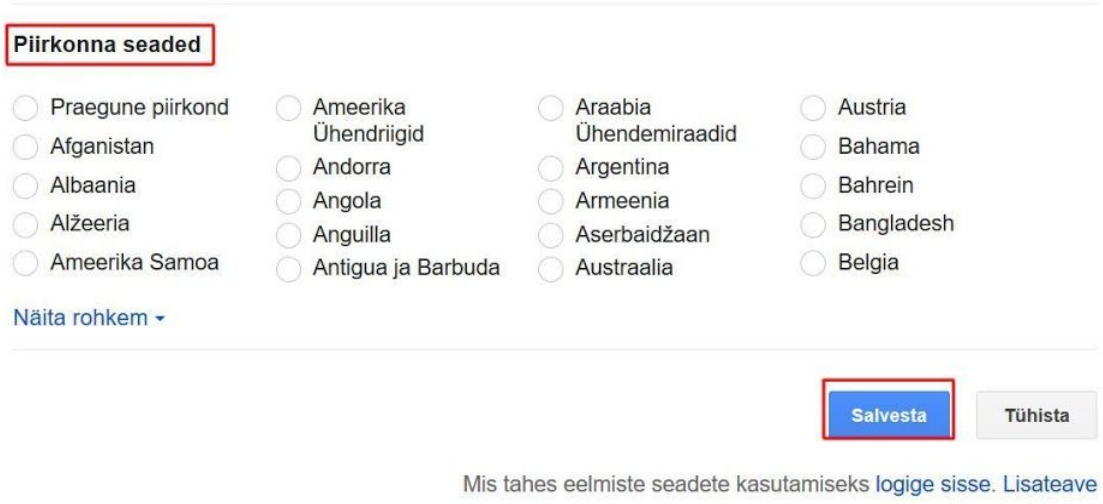
- Logige välja kõigist keskkondadest, mis võivad otsimootoriga seotud olla, nt Google’i otsimootori puhul Google’i kontolt, Bingi otsimootori puhul Microsofti kontolt.

- Google’i otsimootori korral kontrollige otsitulemuste personaliseerimise sätteid. Selleks avage lehekülg www.google.com/history/optout ja veenduge, et väljalogitud kasutaja otsingutegevuse jälgimine oleks välja lülitatud (vt joonist 6).

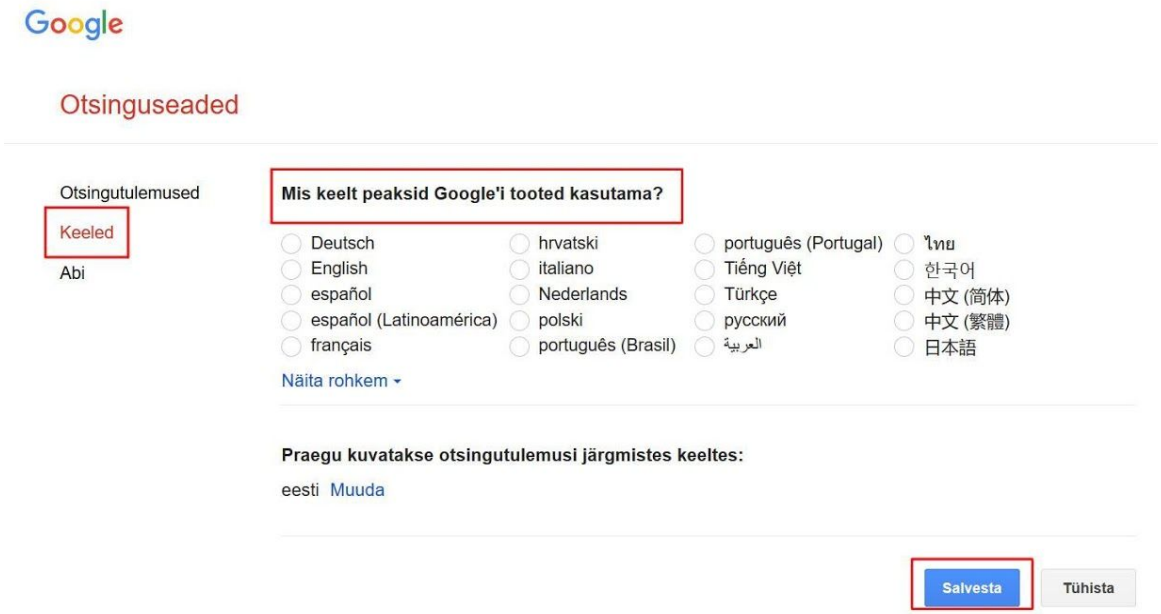
Joonis 6. Google'i kontolt väljalogitud kasutaja otsitegevuse jälgimise väljalülitamine

- Google'i otsimootori korral määrakse sätetes asukoht ("*Eelistused*" > "*Otsinguseaded*" > "*Piirkonna seaded*"; vt jooniseid 7 ja 8) ja tulemuste keel ("*Eelistused*" > "*Otsinguseaded*" > "*Keeled*"; vt joonist 9) (Digital Methods Initiative, 2015).

Joonis 7. Google'i otsimootori otsinguseadete avamine



Joonis 8. Piirkonna seadete määramine (nt klõpsake *"Näita rohkem"*, valige *"Eesti"* ja vajutage *"Salvesta"*)



Joonis 9. Otsingutulemuste keele valimine (nt klõpsake *"Näita rohkem"*, valige *"eesti"* ja vajutage *"Salvesta"*)

1.2. Otsimootori päringu abil avalike Facebooki lehtede kaardistamine

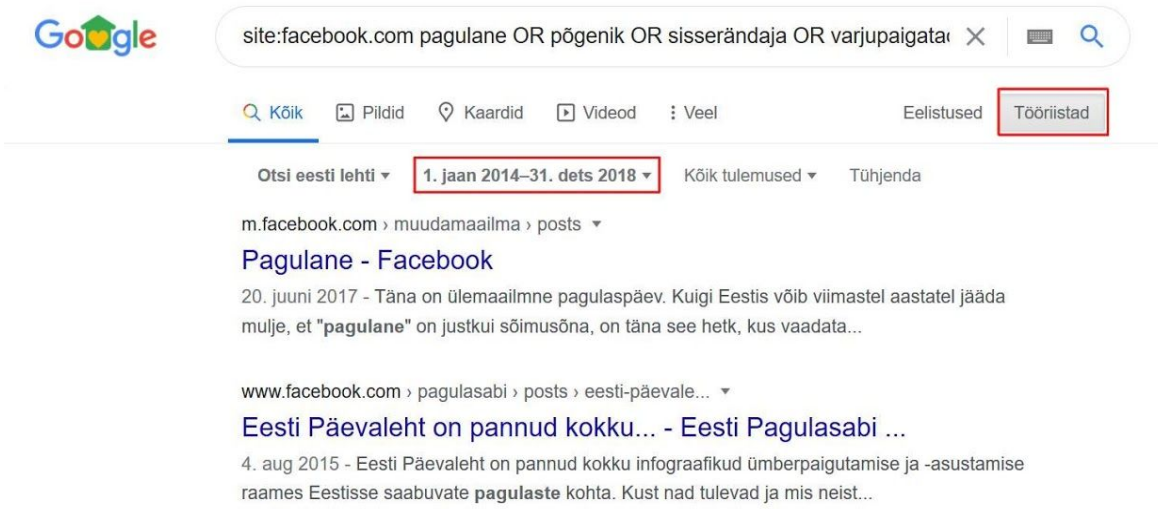
- Päringu otsisõnadena kasutage kas: a) uurimisteema kontekstis universaalseid, üldtuntud sõnu (nt "sisserändaja", "immigrant", "pagulane"), või b) võimalikult erineva tähendusvarjundiga sõnu (nt "abivajaja", "pagulane", "sissetungija") (Rogers, 2019).

- Pärast esimese päringu läbiviimist, tuleks otsitulemustega tutvuda ja päringu sõnastust selle põhjal täiustada. Sama protsessi tuleks korrata, kuni päring on võimalikult täpne.

- Päringu täiustamisel kasutage ka operaatoreid, nt:

- ➢ “” ehk jutumärgid on vajalikud, kui te ei soovi, et otsimootor nt parandaks võtmesõna kirjapilti, kasutaks sünonüüme või muid sarnaseid sõnu, kasutaks võtmesõna tüve või jätaks osa võtmesõnu välja (nt *“pagulane”*);
- ➢ * ehk tärn markeerib teadmata tähti või sõnu (nt *pagul**)
- ➢ *site:* võimaldab määrata kindla veebilehe, millelt otsimootor tulemusi otsib (nt *“pagulane” site:facebook.com*);
- ➢ - ehk miinusmärk võimaldab määrata sõnad või veebilehed, mille soovite päringu tulemustest välja jätta (nt *“migrant” -“emigrant”*).

❖ Määrake otsitulemuste ajaline piirang (nt Google’i otsimootori korral klõpsake valikul *“Tööriistad”*, valige *“Igal ajal”* asemel *“Kohandatud valik…”* ja sisestage täpne kuupäevade vahemik, vt joonist 10). Näiteks RITA-RÄNNE projekti raames läbiviidud sotsiaalmeediauuringus on kasutatud päringut *site:facebook.com pagulane OR põgenik OR sisserändaja OR varjupaigataotleja OR asüülitaotleja*, mille ajaliseks piiranguks märgiti *1/1/2014 – 12/31/2018* (Salvet, 2020).



Joonis 10. Otsingutulemuste ajalise piirangu määramine

❖ Kui olete sõnastanud piisavalt täpse päringu, koostage otsitulemustest (st veebiaadressidest) kas käsitsi või kraapija abil nimekiri.

❖ Toimetage otsitulemuste nimekirja. Nt eemaldage korduvad veebiaadressid, kontrollige, kas tegu on avaliku Facebooki lehe aadressiga (kui mitte, siis eemaldage nimekirjast), viige veebiaadressid standardsele kujule (nt *facebook.com/leheaadress*).

❖ Kasutades veebiaadresside nimekirja, tuvastage iga Facebooki lehe unikaalne ID-kood. Vastavaid päringuid saab läbi viia nt veebilehel https://findmyfbid.com/. Koostage Facebooki lehtede ID-koodidest nimekiri.

1.3. Andmekogumis- ja andmeanalüüsitarkvara allalaadimine ja paigaldamine

- Andmekogumiseks paigaldage arvutisse Facepageri tarkvara. Facepager on mõeldud veebilehtedelt avalike andmete kogumiseks kas rakendusliideste kaudu, kraapimise või failide allalaadimise teel (Jünger & Keyling, 2020).
    - Laadige Facepager alla siit: https://github.com/strohne/Facepager.
    - Facepageri kasutusjuhend: https://github.com/strohne/Facepager/wiki.
- Meelestatusanalüüsi ja teema mudeldamise jaoks paigaldage värskeim Pythoni Anaconda versioon. Python on vabavaraline programmeerimiskeel, millele on loodud arvukalt teeke, mis pakuvad mitmekesiseid võimalusi andmeanalüüsiks (Python Software Foundation, 2020). Võrreldes nt R-iga on Pythonis eestikeelsete tekstide eeltöötluseks avaramad võimalused, seda eeskätt tänu estnltk teegile. Pärast Pythoni installimist looge vähemalt Pythoni 3.5 või uuemal versioonil põhinev keskkond, kuhu paigaldage koodi hõlpsamaks muutmiseks, salvestamiseks ja käivitamiseks Jupyter Notebooki tarkvara ning analüüsiks vajalikud teegid: pandas, numpy, re, langdetect, estnltk, sklearn ja gensim.
    - Laadige Pythoni Anaconda versioon alla siit: https://www.anaconda.com/products/individual.
    - Sissejuhatus Pythonisse: https://progeopik.cs.ut.ee/.
- Jooniste-tabeliste tegemiseks ja statistilise andmeanalüüsi jaoks paigaldage R ja RStudio. R on vabavaraline programmeerimiskeel, mis pakub häid võimalusi nii andmeanalüüsiks kui jooniste tegemiseks, ning RStudio on kasutajaliides, mis hõlbustab R-i kasutamist (Kolnes, 2020).
    - Laadige R alla siit: https://cran.r-project.org/.
    - Laadige RStudio alla siit: https://rstudio.com/products/rstudio/.
    - Sissejuhatus R-i ja RStudiosse: http://samm.ut.ee/sissjuhatus-r-i-ja-rstudiosse.

2. ANDMEKOGUMINE FACEPAGERIGA

Facepageri kogub Facebookist andmeid suheldes Facebooki Graphi rakendusliidesega (vt Facebook, 2020a). See, milliseid andmeid Graphi rakendusliidese kaudu koguda saab, on sageli muutunud ning lisanduvateks piiranguteks tuleks valmis olla ka edaspidi. Näiteks tuleb arvestada, et Graphi rakendusliides ei võimalda koguda iga avaliku Facebooki lehe kõiki postitusi, vaid ligikaudu 600 postitust aasta kohta (Facebook, 2020b). Seda, milliseid andmeid soovite Facepageri abil Facebookist koguda, tuleks aga täpsustada eelsätetes, mida on soovitatav talletada JSON-formaadis. JSON on andmevahetusvorming, mis hõlbustab andmevahetust erinevate programmeerimiskeelte vahel ning koosneb nimi-väärtus-paaridest ja järjestatud jadadest (Ecma International, 2017). Käesolev peatükk annab põgusa ülevaate, kuidas kirjutada JSON-formaadis Facepageri eelsätted, samad eelsätted Facepageris laadida, luua andmebaasifail, lisada Facebooki ligipääsuvõti (*access token*), käivitada andmekogumine ning kogutud andmed CSV-failina eksportida.

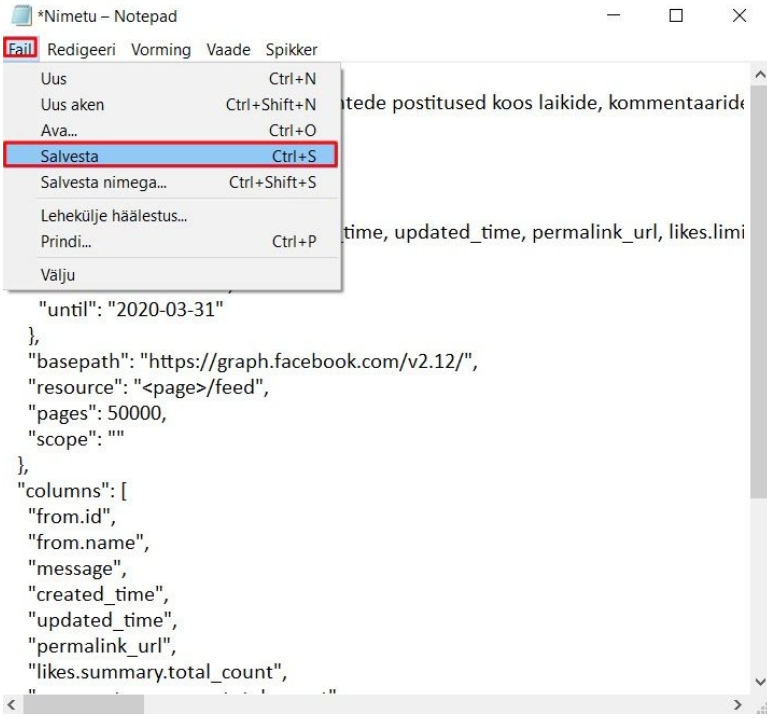
2.1. JSON-formaadis eelsätete loomine

- Allpool on JSON-kood, mis sisaldab eelsätteid Facebooki postituste kogumiseks Facepageriga. Nende eelsätete järgi kogutakse andmeid järgmiste tunnuste lõikes: Facebooki lehe ID-kood ja nimi, postituse sisu, avaldamise ja viimase muutmise aeg, postituse püsilink, laikide, kommentaaride, jagamiste ja erinevate reaktsioonide arv. Kopeerige alljärgnev JSON-kood, avage Notepad või mõni muu analoogne tekstitöötlusprogramm ja kleepige see sinna:

```
{
  "description": "Avalike Facebooki lehtede postitused koos laikide, kommentaaride, jagamiste ja reaktsioonide arvuga. Samuti sisaldavad andmed Facebooki lehe ID-koodi ja nime, postituse avaldamise ja muutmise aega ning püsilinki.",
  "module": "Facebook",
  "speed": 200,
  "options": {
    "params": {
      "fields": "from, message, created_time, updated_time, permalink_url, likes.limit(0).summary(true), comments.limit(0).summary(true), shares.limit(0).summary(true), reactions.type(LOVE).limit(0).summary(1).as(reactions_love), reactions.type(WOW).limit(0).summary(1).as(reactions_wow), reactions.type(HAHA).limit(0).summary(1).as(reactions_haha), reactions.type(SAD).limit(0).summary(1).as(reactions_sad), reactions.type(ANGRY).limit(0).summary(1).as(reactions_angry)",
      "<page>": "<Object ID>",
      "since": "2020-01-01",
      "until": "2020-03-31"
    },
    "basepath": "https://graph.facebook.com/v2.12/",
    "resource": "<page>/feed",
    "pages": 50000,
    "scope": ""
  },
  "columns": [
    "from.id",
    "from.name",
    "message",
    "created_time",
    "updated_time",
    "permalink_url",
    "likes.summary.total_count",
    "comments.summary.total_count",
    "shares.count",
    "reactions_love.summary.total_count",
    "reactions_wow.summary.total_count",
    "reactions_haha.summary.total_count",
    "reactions_sad.summary.total_count",
    "reactions_angry.summary.total_count"
  ],
  "name": "Facebooki postitused"
```

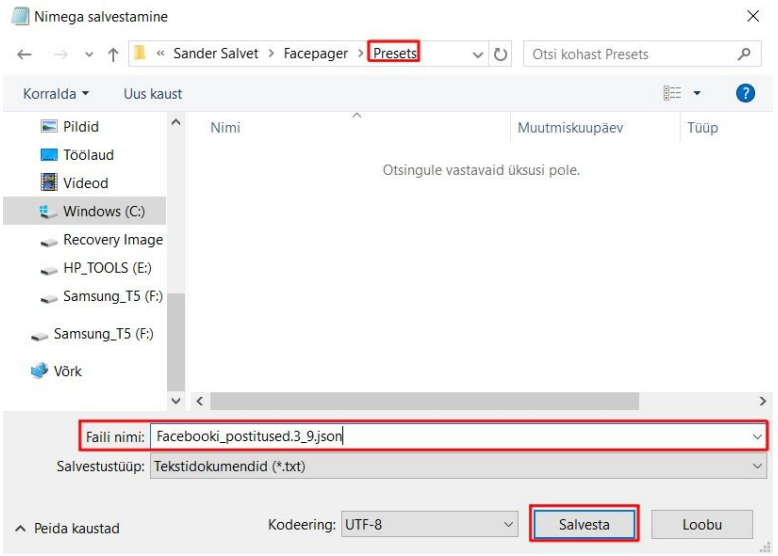
```
}
```

- Kui olete JSON-koodi tekstitöötlusprogrammi kleepinud, tuleks koodis kindlasti muuta kogutavate postituste avaldamisperioodi. Selleks muutke perioodi algus- (*"since"*) ja lõppkuupäeva (*"until"*), järgides kuupäeva vormingut *"AAAA-KK-PP"*. Soovi korral on võimalik täpsustada ka kogutavaid tunnuseid, kuid selleks tuleb koodis muuta nii andmevälju (*"fields"*) kui ka andmestiku veergusid (*"columns"*). Andmeväljade ja veergude muutmisel on näidistena abiks Facepageri vaike-eelsätete kaustast (*"C:\Users\Username\Facepager\DefaultPresets"*) leitavad JSON-failid, samuti Facepageri kasutusjuhend (Jünger & Keyling, 2020) ja Facebooki Graphi rakendusliidese dokumentatsioon (Facebook, 2020a).
- Eelsätete JSON-formaadis salvestamiseks vali Notebookis kõigepealt *"Fail" > "Salvesta"* (vt joonis 11).



Joonis 11. Eelsätete salvestamine: *"Fail" > "Salvesta"*

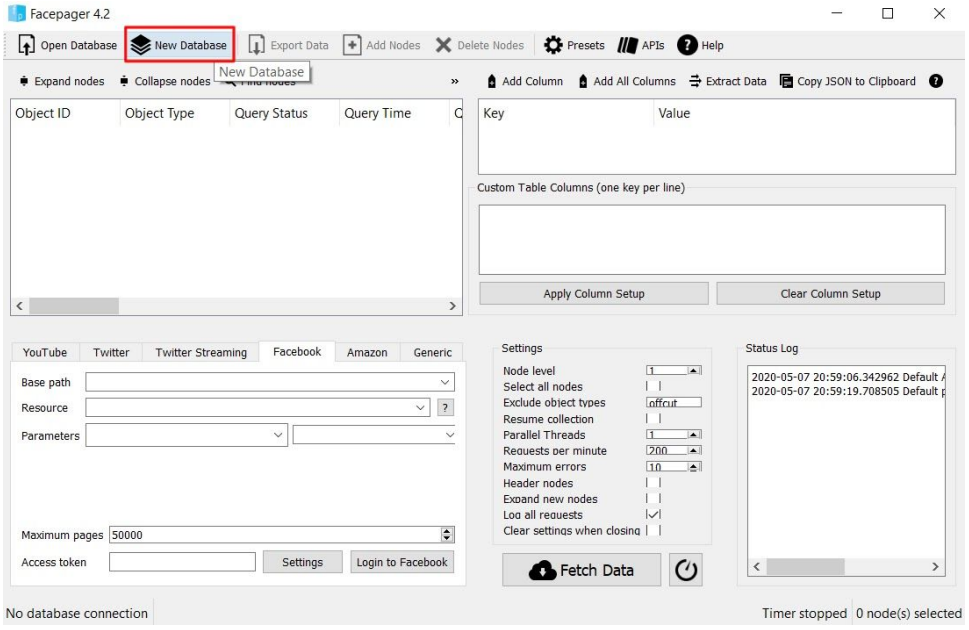
- Seejärel valige faili salvestamise asukohaks Facepageri eelsätete kaust (*"C:\Users\Username\Facepager\Presets"*). Määrake faili nimi, järgides vormingut *failinimi.3_9.json*. Salvestage fail, klõpsates *"Salvesta"* (vt joonist 12). Pärast salvestamist sulgege loodud JSON-fail.



Joonis 12. Eelsätete salvestamine: faili asukoha ja nime valimine

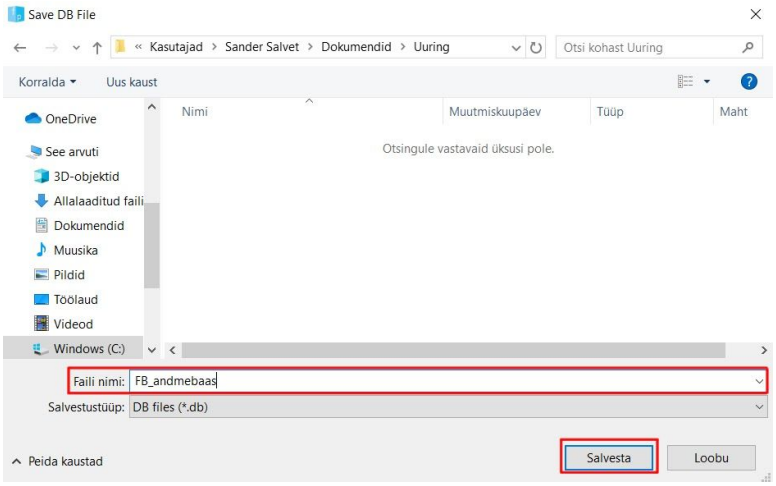
2.2. Andmekogumine

- Avage Facepager ja alustage uue andmebaasi loomist, valides *"New Database"* (vt joonist 13).



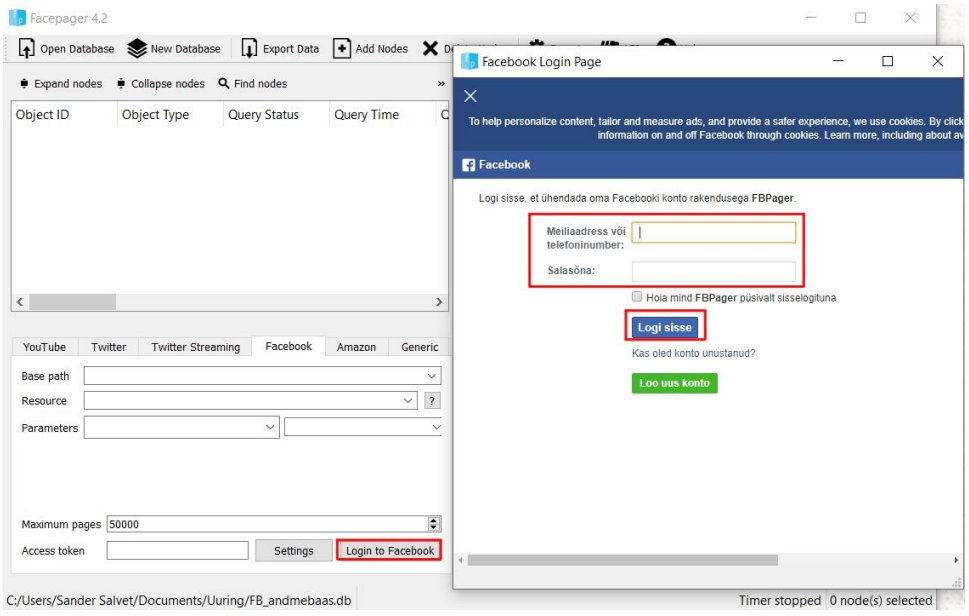
Joonis 13. Uue andmebaasi loomine: valik *"New Database"*

- Valige andmebaasile asukoht kõvakettal ja nimi ning kinnitage need, klõpsates *"Salvesta"* (vt joonist 14).



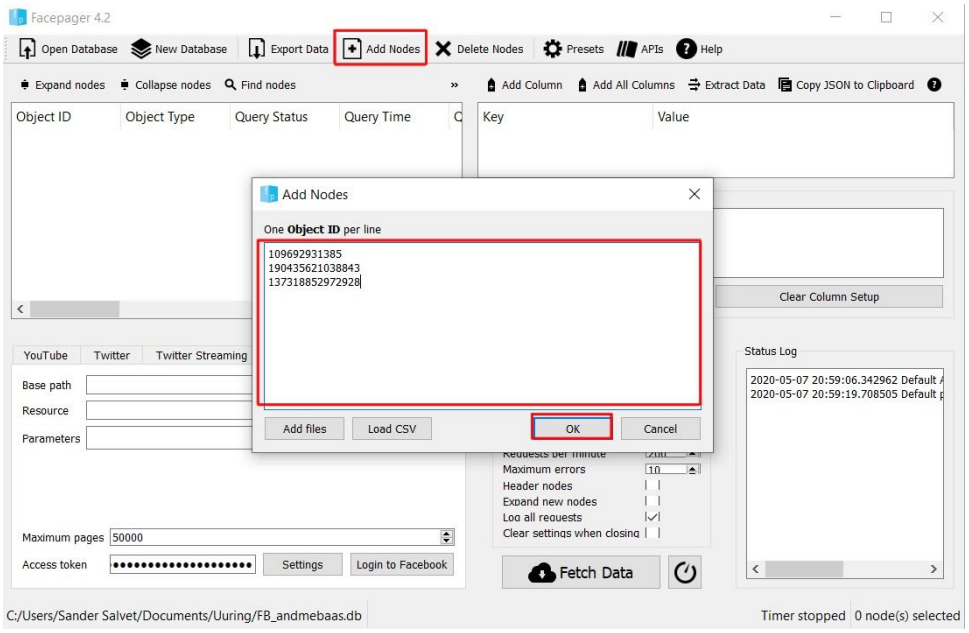
Joonis 14. Uue andmebaasi loomine: faili asukoha ja nime valimine

❖ Logige Facepageri kaudu enda Facebooki kontole sisse. Sisselogimine on vajalik ligipääsuvõtme (*access token*) saamiseks. Selleks klõpsake valikul *"Login to Facebook"* ja sisestage hüpikaknas enda meiliaadress ja salasõna (vt joonist 15).



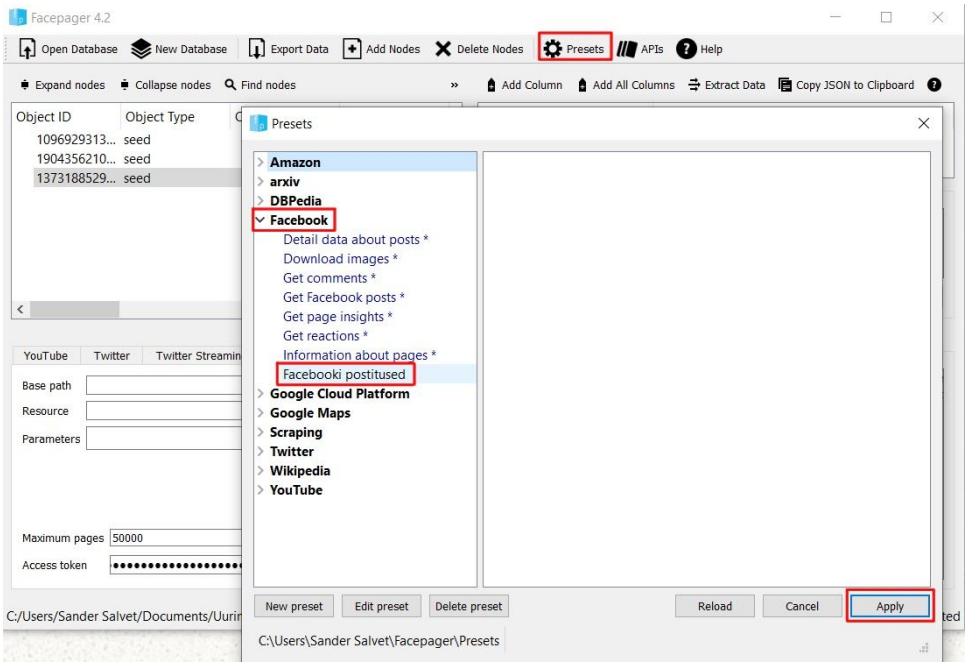
Joonis 15. Facepageri kaudu Facebooki sisselogimine

❖ Sisestage nende avalike Facebooki lehtede ID-koodid, mille andmeid soovite koguda (vt alapeatükki 1.3). Selleks valige *"Add Nodes"*, sisestage ID-koodid uues aknas nii, et iga ID-kood on eraldi real, ja klõpsake *"OK"* (vt joonist 16). Mahukamate andmehulkade korral on soovitatav koguda andmeid väiksemate osadena ja jaotada päringud pikema aja peale, et tulenevalt ühistest andmekogumispiirangutest arvestada ka Facepageri tarkvara teiste kasutajatega.



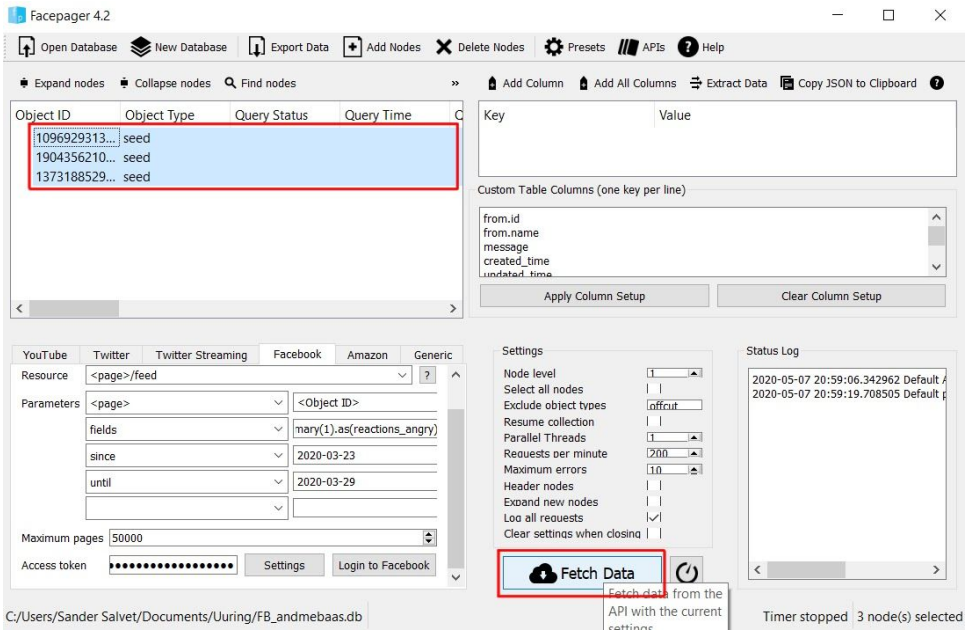
Joonis 16. Avalike Facebooki lehtede ID-koodide lisamine

- Laadige eelnevalt loodud päringu eelsätted. Selleks valige *"Presets"* > *"Facebook"* > *"Facebooki postitused"* ning kinnitage valik, klõpsates *"Apply"* (vt joonist 17).



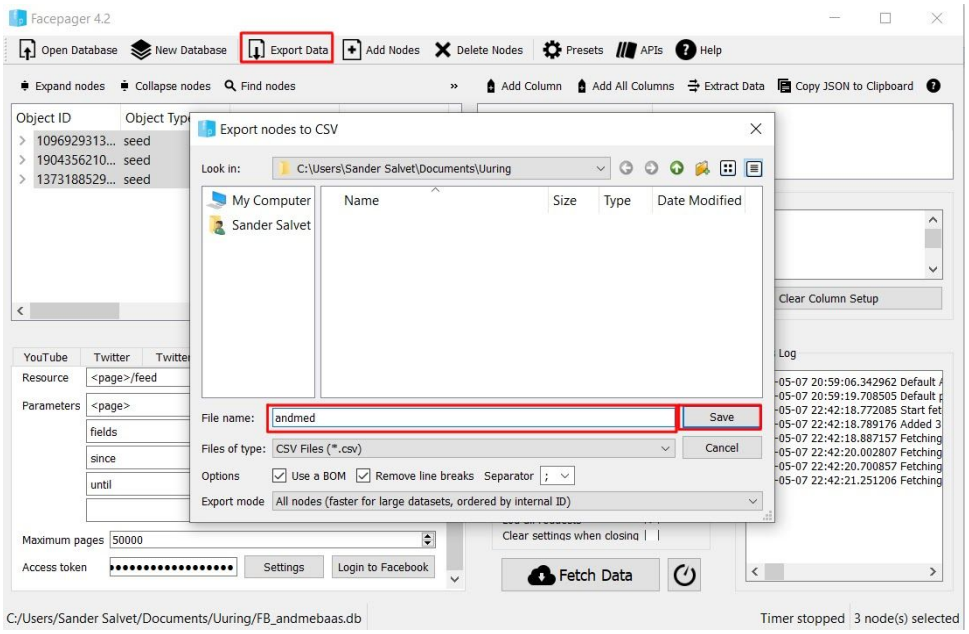
Joonis 17. Eelsätete laadimine

- Hoides all vasakut hiireklahvi märgistage kõik Facepageris kuvatud Facebooki lehtede ID-koodid. Seejärel vajutage *"Fetch Data"*, et alustada päringut Facebooki Graphi rakendusliidese kaudu (vt joonist 18).



Joonis 18. Andmekogumise käivitamine

- Kogutud andmete salvestamiseks CSV-failina kõvakettale vajutage *"Export Data"*, valige failile asukoht ja nimi ning kinnitage valikud klõpsates *"Save"* (vt joonist 19).



Joonis 19. Kogutud andmete salvestamine CSV-failina

3. MEELESTATUSANALÜÜS JA TEEMA MUDELDAMINE PYTHONI JA R-IGA

Meelestatusanalüüsi (*sentiment analysis*) kasutatakse hoiakute automaatseks tuvastamiseks tekstides, kui on vaja uurida mahukaid tekstikorpuseid (vt Liu, 2015). Meelestatusanalüüs koosneb mitmest etapist: eeltöötlusest (nt suurtähtede asendamine väiketähtedega, numbrite ja kirjavahemärkide eemaldamine, kirjavigade parandamine, lemmatiseerimine ehk sõnade algvormis

esitamine, sõnade vektoriseerimine), juhendatud masinõppe korral ka klassifitseerija treenimisest, klassifitseerija täpsuse hindamisest, teksti meelestatuse automaatsest tuvastamisest.

Käesolev juhend näpunäiteid juhendatud masinõppel põhinevaks meelestatusanalüüsiks. Juhendatud masinõpe tähendab, et statistilist mudelit, mille alusel tekstides hoiakuid tuvastatakse, on vaja eelnevalt treenida ja mudeli täpsust testida. Selleks tuleb uurijal valida juhuslikult teatud arv tekste (nt 1000 Facebooki postitust), lisada andmestikule käsitsi meelestatusandmed (nt kas postituses on väljendatud positiivset või negatiivset hoiakut), ning jagada tekstid kaheks osaks: treening- (nt 800 postitust) ja testandmestikuks (nt 200 postitust). Treeningandmestikku kasutatakse klassifitseerija treenimiseks ja testandmestikku klassifitseerija täpsuse hindamiseks. Seejärel tuvastatakse treenitud klassifitseerija abil ülejäänud tekstide meelestatus.

Teema mudeldamine (*topic modeling*) võimaldab seevastu kaardistada suurtes tekstikorpustes esinevaid teemasid (vt Alghamdi & Alfalqi, 2015). Sarnaselt meelestatusanalüüsile koosneb ka teema mudeldamine mitmest etapist, sealhulgas on vajalik tekstide eeltöötlus. Käesolev peatükk keskendub juhendamata masinõppel põhinevale teema mudeldamisele. Juhendamata masinõpe tähendab, et teemad tuvastatakse statistilise mudeli alusel, mis võimaldab tekstides esinevaid sõnu klasterdada, ning erinevalt juhendatud masinõppest pole mudeli eelnev treenimine vajalik. Väljundiks on teemad kui teatud koosesinevate sõnade kogumikud. Mudeli sobivust tuleks hinnata nii koherentsusnäitaja kui teemade sisulise interpreteeritavuse põhjal. Loodud teema mudelit on võimalik seejärel kasutada ka postituste klassifitseerimiseks tekstis esinevate teemade alusel.

Peatükist võib leida Pythoni koodi Facebooki postituste eeltöötluseks ning meelestatuse ja teema alusel klassifitseerimiseks, samuti R-i koodi näited tabelite ja jooniste tegemiseks ning statistilise analüüsi läbiviimiseks (nt Crameri V seosekordaja arvutamiseks ja dispersioonanalüüsiks).

3.1. Eeltöötlus (Python)

Eeltöötluse juhised on jagatud kolmeks osaks: 1) teksti puhastamine ja lemmatiseerimine (vajalik nii meelestatusanalüüsi kui teema mudeldamise jaoks), 2) andmestiku osadeks jagamine ja käsitsi märgendamine (vajalik ainult meelestatusanalüüsi jaoks), 3) teksti vektoriseerimine (vajalik ainult meelestatusanalüüsi jaoks).

3.1.1. Teksti puhastamine ja lemmatiseerimine

Tuvastage postituste keel ja filtreerige postitusi keele alusel (mõningatel Facebooki lehtedel võib esineda erinevates keeltes postitusi), eemaldage postitustest URLid, teemaviited, mainimised, numbrid, kirjavahemärgid, üleliigsed tühikud ja sagedased, kuid väheinformatiivsed sõnad, asendage suurtähed väiketähtedega, parandage kirjavead ja lemmatiseerige tekstid ehk esitada sõnad algvormis.

```
# Teekide ja moodulite importimine
import pandas as pd
import re
from langdetect import detect
from estnltk import Text
from estnltk import Disambiguator
from estnltk.names import TEXT, ANALYSIS, ROOT, POSTAG, FORM
```

```
# CSV-formaadis andmestiku sisselugemine
df = pd.read_csv('/home/sander/Documents/Uuring/andmed.csv', encoding = 'utf-8')

# Postituste keele tuvastamine
posts = df['message'].tolist()

languages = []
for x in posts:
    try:
        language = detect(x)
    except:
        language = "error"
    languages.append(language)

# Keeleandmete salvestamine andmestikku
df['language'] = languages

# Võõrkeelsete postituste eemaldamine
df = df[df.language == 'et']

# URLide eemaldamine
posts = df['message'].tolist()
posts =
[re.sub('(http(s)?:\/\/.)?(www\.)?[-a-zA-Z0-9@:%._\+~#=]{0,}\.[a-z]{2,4}\b([-a-zA-Z0-9@:;%_\+.~#?
&\/=\(\)]*)', '', x) for x in posts]

# Hashtag'ide ehk teemaviidete eemaldamine (nt #ränne)
posts = [re.sub('#\w+', '', x) for x in posts]

# Maninimiste eemaldamine (nt @Mari_Tamm)
posts = [re.sub('@\w+', '', x) for x in posts]

# Kirjavigade automaatne parandamine
corrected_posts = []
for text_str in posts:
    text = Text(text_str)
    corrections = text.fix_spelling()
    corrected_posts.append(corrections)

# Kirjavigade parandamise käigus tekkinud andmestruktuuri muutuse
korrigeerimine,ettevalmistamine lemmatiseerimiseks
corr_p = []
for x in corrected_posts:
    temp = list(x.values())
```

```
    corr_p.append(temp)

corrected_posts = []
for x in corr_p:
    for i in x:
        temp = re.sub('\s+', ' ', i)
        temp = [temp.strip()]
    corrected_posts.append(temp)

# Teksti lemmatiseerimine ehk sõnade esitamine algvormis, tulemuste salvestamine listi.
Mugandatud versioon Kristel Uiboaia (2017) koodist
disamb = Disambiguator()
lemmas = []
for x in corrected_posts:
    text = disamb.disambiguate(x)
    for analysis in text:
        temp_lemmas = []
        for word in analysis.words:
            first_analysis = word[TEXT], word[ANALYSIS][0][ROOT], word[ANALYSIS][0][POSTAG],
word[ANALYSIS][0][FORM]
            if first_analysis[2] == 'V' and first_analysis[1] != "ei" and first_analysis[1] != "ära":
                temp_lemmas.append(re.sub("[=_]", "", first_analysis[1]) + 'ma')
            elif first_analysis[2] == 'Y':
                temp_lemmas.append(re.sub("[=_]", "", first_analysis[1]))
            else:
                temp_lemmas.append(re.sub("[=_]", "", first_analysis[1]))
        lemmas.append(temp_lemmas)

# Lemmade ühendamine sõnedeks
lemmas = [' '.join(x) for x in lemmas]

# Suurtähed väiketähtedeks
lemmas = [x.lower()for x in lemmas]

# Kirjavahemärkide eemaldamine
lemmas = [re.sub('[^\w\s]','', x) for x in lemmas]

# Numbrite eemaldamine
lemmas = [re.sub('\d+', '', x) for x in lemmas]

# Sagedaste, kuid ebainformatiivsete sõnade eemaldamine (nt asesõnad, sidesõnad, verbid
'olema', 'saama', 'hakkama')
lemmas =
[re.sub('(?<![a-z])(mina|sina|tema|teie|see|too|ja|ning|ega|ehk|või|kui|ka|ehkki|kuigi|kuid|a
ga|ent|sest|seepärast|toopärast|sestap|küllap|kuna|et|kui|siis|vaid|kes|mis|millal|mil|kas|k
```

```
uidas|kus|kuhu|kust|miks|millepärast|milleks|nagu|justkui|otsekui|missugune|milline|selline|
niisugune|mitmes|olema|saama|hakkama)(?![a-z])', '', x) for x in lemmas]

# Üleliigsete tühikute eemaldamine
lemmas = [re.sub('\s+', ' ', x.strip()) for x in lemmas]

# Puhastatud postituste salvestamine andmestikku
df['message_clean'] = lemmas

# Andmestiku salvestamine kõvakettale
df.to_csv('/home/sander/Documents/Uuring/puhastatud_andmed.csv', encoding = 'utf-8', index =
False)
```

Valikuline: eeltöötluse eri etappides, nt filtersõnade ja eemaldatavate sagedaste sõnade valikul, võib olla abi sõnade esinemissageduse analüüsist. Sõnade esinemissagedusi saab uurida alamklassi *Counter* abil. Alljärgnevas näites on kasutatud sama listi *lemmas*, mida ülalpoolgi. Eemaldatavate sõnade valikul võib abi olla ka eesti kirjakeele sagedussõnastikust (Kaalep & Muischnek, 2002).

```
# Alamklassi Counter importimine
from collections import Counter

# Sõnede ühendamine
tokenized_words = ', '.join(lemmas)

# Lemmade tokeniseerimine
tokenized_words = tokenized_words.split()

# Lemmade sagedusanalüüs
counter = Counter(tokenized_words)

# Kõige sagedasemate lemmade kuvamine (sulgudes märgitud arv määrab ekraanil kuvatavate
lemmade arvu)
top_lemmas = counter.most_common(100)
print(top_lemmas)

# Kõige vähem kasutatud lemmade kuvamine (esimene kandilistes sulgudes olev arv määrab
kuvatavate lemmade arvu)
rare_lemmas = counter.most_common()[:-50000-1:-1]
print(least_common)

# Kui huvipakkuvaid tulemusi on palju, on otstarbekam salvestada need CSV-faili ja uurida
lähemalt tabelarvutusprogrammis (nt Microsoft Excelis, LibreOffice Calcis)
rare_lemmas = pd.DataFrame(rare_lemmas)
rare_lemmas.to_csv('/home/sander/Documents/Uuring/harvaesinevad_lemmad.csv', encoding =
'utf-8', index = False)
```

3.1.2. Andmestiku osadeks jagamine ja postituste käsitsi märgendamine

Enne teksti käsitsi märgendamist tuleks andmestik jagada juhuslikkuse alusel kaheks osaks: üks osa (nt 1000 postitust) käsitsi märgendamiseks ja teine osa automaatse analüüsi jaoks. Kui mõlemad failid on kõvakettale salvestatud, tuleks avada esimene fail *kodeerimiseks.csv* mõnes tabelarvutusprogrammis, nt Excelis või LibreOffice Calcis, seejärel luua uus tulp *sentiment*, kuhu märkida iga postituse meelestatus, ning muudatused salvestada.

```
# Postituste järjekorra muutmine andmestikkus juhuslikkuse alusel
df = df.sample(frac = 1).reset_index(drop = True)

# Andmestiku jagamine kaheks uueks andmestikuks:
# 1) ühes andmestikus ('df1') 1000 postitust käsitsi märgendamise jaoks,
# 2) teises andmestikus ('df2') kõik ülejäänud postitused automaatse analüüsi jaoks.
df1 = df.iloc[:1000]
df2 = df.iloc[1000:]

# Mõlema andmestiku salvestamine kõvakettale
df1.to_csv('/home/sander//Documents/Uuring/kodeerimiseks.csv', encoding = 'utf-8', index =
False)
df2.to_csv('/home/sander/Documents/Uuring/mittekodeerimiseks.csv', encoding = 'utf-8', index =
False)
```

Käesolevat juhendit luues andis kõige täpsemaid tulemusi märgendamisviis, mis eristab meelestatuse kolme kategooriat:

- -1 = tugevalt negatiivne hoiak;
- 0 = nõrgalt negatiivne, nõrgalt positiivne või neutraalne hoiak;
- 1 = tugevalt positiivne hoiak.

Tugevalt negatiivne või tugevalt positiivne hoiak iseloomustab postitusi, milles on avaldatud arvamust kõigi migrantide (nt “sisserändajad”) või väga suure rühma migrantide kohta (nt “aafriklased”) ning väljendatud seejuures kas tugevat vastuseisu või tugevat poolehoidu. Nõrgalt negatiivsete või nõrgalt positiivsete postituste korral on käsitletud mõnd üksikjuhtumit (nt “Vao pagulaskeskuses elav pagulane”) või väikest hulka migrante (nt “Tallinnas elavad sisserändajad”), samuti on avaldatud arvamust nt migrantide üldise sobivuse ja kohanemisvõime kohta sihtriigi ühiskonnas. Neutraalsete postituste puhul pole aga selget meelestatust migrantide suhtes võimalik tuvastada.

3.1.3. Teksti vektoriseerimine

Järgmiseks vektoriseerige tf–idf tehnika abil postitustes esinevad sõnad. Tf–idf vektroriseerimistehnika võimaldab omistada igale sõnale kaalu, mis näitab sõna olulisust postituses, võttes arvesse ka selle esinemissagedust tekstikorpuses üldiselt (What…, [2020]).

```
# Teekide ja moodulite importimine
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer

# CSV-formaadis andmestike sisselugemine:
# 1) 'df' sisaldab kogu andmestikku,
# 2) 'df1' sisaldab käsitsi märgendatud andmeid (meelestatus lisatud),
# 3) 'df2' sisaldab märgendamata andmeid (ülejäänud andmed, kus meelestatus puudu)
df = pd.read_csv('/home/sander/Documents/Uuring/puhastatud_andmed.csv', encoding = 'utf-8')
df1 = pd.read_csv('/home/sander/Documents/Uuring/kodeerimiseks.csv', encoding = 'utf-8')
df2 = pd.read_csv('/home/sander/Documents/Uuring/mittekodeerimiseks.csv', encoding = 'utf-8')

# Käsitsi märgendatud andmestiku jagamine treening- (800 postitust) ja testandmestikuks (200 postitust)
df1 = df1.sample(frac = 1).reset_index(drop = True)
train = df1.iloc[:800]
test = df1.iloc[800:]

# Uuritavate postituste tekstide ja meelestatusandmete salvestamine listidesse
posts = df['message_clean'].tolist() # Kõik postitused
train_posts = train['message_clean'].tolist() # Treeningandmestiku postitused
train_sentiments = train['sentiment'] # Treeningandmestiku postituste meelestatus
test_posts = test['message_clean'] # Testandmestiku postitused
test_sentiments = test['sentiment'] # Testandmestiku postituste meelestatus
analysis_posts = df2['message_clean'] # Ilma meelestusandmeteta postitused, mida automaatselt analüüsida

# TF-IDF tunnuste maatriksi loomine ja normaliseerimine, kasutades kõiki postitusi
tfidf_vect = TfidfVectorizer(max_features=300)
tfidf_vect.fit_transform(posts)

# Treeningandmestiku postituste vektoriseerimine
train_X_tfidf = tfidf_vect.transform(train_posts)

# Testandmestiku postituste vektoriseerimine
test_X_tfidf = tfidf_vect.transform(test_posts)

# Ilma meelestusandmeteta postituste vektoriseerimine
analysis_X_tfidf = tfidf_vect.transform(analysis_posts)
```

3.2. Postituste meelestatuse tuvastamine (Python)

Treenige klassifitseerija, hinnake selle täpsus ning määrake automaatselt ka märgendamata postituste meelestatus.

Kasutage klassifitseerimiseks multiklassilist tugivektormasinat (SVM): tugivektormasina mudel võimaldab jagada andmeid kahe kategooria vahel (nt “positiivne” ja “negatiivne”), multiklassilist

mudelit kasutatakse kui kategooriaid on rohkem kui kaks (nt “tugevalt negatiivne”, “nõrgalt positiivne / nõrgalt negatiivne / neutraalne”, “tugevalt positiivne”). Viimasel juhul klassifitseerib mudel tekste n-ö üks-ühe-vastu-põhimõttel: nt kõigepealt jaotab tekstid kategooriate “tugevalt negatiivne” ja “tugevalt positiivne” vahel, siis “tugevalt negatiivne” ja “nõrgalt positiivne / nõrgalt negatiivne / neutraalne” jne.

```
# Teekide ja moodulite importimine
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_recall_fscore_support

# Klassifitseerija treenimine
svm_model = svm.SVC()
svm_model = svm_model.fit(X = train_X_tfidf, y = train_sentiments)

# Klassifitseerija testimine
y_pred = svm_model.predict(test_X_tfidf)

# Tulemuste täpsuse hindamine
accuracy_score(test_sentiments, y_pred)

# Ülejäänud postituste meelestatuse automaatne tuvastamine
y_analysis = svm_model.predict(analysis_X_tfidf)

# Meelestatusandmete salvestamine andmestikku
df2['sentiment'] = y_analysis

# Käsitsi ja automaatselt määratud meelestatusega andmestike ühendamine üheks andmestikuks
sentiment_data = df1.append(df2, sort = False)

# Andmestiku salvestamine kõvakettale
sentiment_data.to_csv('/home/sander/Documents/Uuring/meelestatusandmed.csv', encoding =
'utf-8', index = False)
```

3.3. Postituste teemade tuvastamine (Python)

Teema mudeldamise läbiviimiseks looge hierarhilise Dirichlet’ protsessi (HDP) mudel. HDP-mudeli üks eripärasid on, et mudeli loomiseks pole uurijal endal vaja määrata korpuses esinevate teemade arvu (vt ka Wang, Paisley & Blei, 2011). HDP-mudeli loomise järel hinnake mudeli koherentsust ja teemade sisulist interpreteeritavust ning kasutage mudelit postituste klassifitseerimiseks neis esinevate teemade alusel.

```
# Teekide ja moodulite importimine
import pandas as pd
import numpy as np
from gensim.corpora.dictionary import Dictionary
from gensim.models.hdpmodel import HdpModel
```

```
from gensim.models import CoherenceModel

# Andmestiku sisselugemine
sentiment_data = pd.read_csv('/home/sander/Documents/Uuring/meelestatusandmed.csv',
encoding = 'utf-8')

# Postituste salvestamine listi
posts = sentiment_data['message_clean'].tolist()

# Sõnade tokeniseerimine postitustes
prepared_posts = []
for x in posts:
    temp = x.split(' ')
    prepared_posts.append(temp)

# Tekstikorpuse loomine
posts_dictionary = Dictionary(prepared_posts)
posts_corpus = [posts_dictionary.doc2bow(text) for text in prepared_posts]

# HDP mudeli loomine
hdp = HdpModel(corpus = posts_corpus, id2word = posts_dictionary, random_state = 1234)

# Kõige sagedasemate teemade kuvamine ekraanil ('num_topics' näitab kuvatavate teemade arvu
ja 'num_words' iga teema juures kuvatavate sõnade arvu) )
hdp_topics = hdp.print_topics(num_topics = 200, num_words = 5)
for topic in hdp_topics:
    print(topic)

# Teemade koherentsuse arvutamine
hdp_cm = CoherenceModel(model = hdp, corpus = posts_corpus, dictionary = posts_dictionary,
texts = prepared_posts, coherence = 'c_v')
HDP_cm = hdp_cm.get_coherence()
HDP_cm

# Iga postituse peamise teema tuvastamine (teema n-ö nimena on kasutatud nimekirja viiest kõige
iseloomulikumast sõnast). Mugandatud versioon Animesh Pandey (2013) koodist.
topics = []
for text_vec in posts_corpus:
    topic_vec = []
    topic_vec = hdp[text_vec]
    word_count_array = np.empty((len(topic_vec), 2), dtype = np.object)
    for i in range(len(topic_vec)):
        word_count_array[i, 0] = topic_vec[i][0]
        word_count_array[i, 1] = topic_vec[i][1]
    idx = np.argsort(word_count_array[:, 1])
```

```
    idx = idx[::-1]
    word_count_array = word_count_array[idx]
    final = []
    final = hdp.print_topic(word_count_array[0, 0], 5)
    topics.append(final)

# Postituste peamiste teemade salvestamine andmestikku
sentiment_data['topic'] = topics

# Andmestikku salvestamine kõvakettale
sentiment_data.to_csv('/home/sander/Documents/Uuring/meelestatusandmed.csv', encoding =
'utf-8', index = False)
```

3.4. Jooniste ja tabelite tegemine, statistiline analüüs (R)

Alljärgnev R-i kood sisaldab näiteid, kuidas luua uusi tunnuseid, tabeleid ja joonised, arvutada Crameri V seosekordaja väärtusi ja viia läbi dispersioonanalüüsi. Crameri V seosekordaja on hii-ruut-statistikust edasi arendatud seosekordaja, mille väärtus näitab kahe nominaal- või järjestustunnuse vahelise seose tugevust ja võib olla vahemikus 0–1 (vt Rootalu, 2014). Dispersioonanalüüs ehk ANOVA võimaldab võrrelda kahe tunnuse lõikes rühmade keskmisi ja hinnata keskmiste erinevuse statistilist olulisust (vt Tooding, 2014).

```
# Pakettide laadimine
library('readr')
library('questionr') # Vajab eelnevat paigaldamist: install.packages('questionr')

# Andmestiku sisselugemine
df = read_csv('/home/sander/Documents/Uuring/meelestatusandmed.csv')

# Uue tunnuse loomine tulemuste filtreerimise teel: postitamisaasta (nt '2015-07-23 06:47:28'
asemel '2015')
df$year = NA # Uue tunnuse 'year' loomine
df$year[grepl('2014', df$created_time) == TRUE] = '2014' # Kui tunnus 'created_at' väärtus
sisaldab fragmenti '2014', siis on tunnuse 'year' väärtus '2014'
df$year[grepl('2015', df$created_time) == TRUE] = '2015' # Kui tunnus 'created_at' väärtus
sisaldab fragmenti '2015', siis on tunnuse 'year' väärtus '2015'
df$year[grepl('2016', df$created_time) == TRUE] = '2016' # Kui tunnus 'created_at' väärtus
sisaldab fragmenti '2016', siis on tunnuse 'year' väärtus '2016'
df$year[grepl('2017', df$created_time) == TRUE] = '2017' # Kui tunnus 'created_at' väärtus
sisaldab fragmenti '2017', siis on tunnuse 'year' väärtus '2017'
df$year[grepl('2018', df$created_time) == TRUE] = '2018' # Kui tunnus 'created_at' väärtus
sisaldab fragmenti '2018', siis on tunnuse 'year' väärtus '2018'

# Uue tunnuse loomine olemasolevate tunnuste kokkuliitmise teel: interaktsioonide arvu
(jagamiste, kommentaaride, laikide ja reaktsioonide summa) leidmine
df$interactions = NA # Uue tunnuse 'interactions' loomine
```

```
df$interactions = df$likes + df$comments + df$shares + df$reactions_love + df$reactions_wow + df$reactions_haha + df$reactions_sad + df$reactions_angry # Tunnuste kokkuliitmine

# Sagedustabeli loomine: postituste arv aastate lõikes
table(df$year)

# Sagedustabeli põhjal joonise loomine: postituste arv aastate lõikes; plot() funktsioon joonise loomiseks ja text() punktide juurde väärtuste lisamiseks
plot(names(table(df$year)), # Aastad (tabelist eraldamiseks vajalik names() funktsioon)
    as.vector(table(df$year)), # Postituste arv (tabelist eraldamiseks vajalik as.vector() funktsioon)
    main = 'Postituste arv', # Joonise pealkiri
    ylab = '', # X-telje nimi
    xlab = '', # Y-telje nimi
    xlim = c(2014, 2018), # X-telje miinimum- ja maksimumväärtus
    ylim = c(0, 4500), # Y-telje miinimum- ja maksimumväärtus
    type = 'b', # Joonise tüüp
    pch = 17, # Punktide kuju
    cex = 1.2, # Punktide suurus
    col = 'dodgerblue', # Joonise värv
    bty = 'L') # Joonise raami kuju
text(names(table(df$year)), # Aastad (tabelist eraldamiseks vajalik names() funktsioon)
    as.vector(table(df$year)),  # Postituste arv (tabelist eraldamiseks vajalik as.vector() funktsioon)
    labels = as.vector(table(df$year)), # N-ö tekst (postituste arv) punktide juurde lisamiseks
    pos = 3, # Teksti asukoht punktide suhtes
    col = 'dodgerblue', # Teksti värv
    cex = 1, # Teksti suurus
    font = 1, # Šrift (püstkiri)
    family = 'sans') # Kirjatüüp

# Risttabeli loomine: meelestatus ja Facebooki lehe tüüp (absoluutarvud)
table(df$page_type, df$sentiment)

# Risttabeli loomine: meelestatus ja Facebooki lehe tüüp (reaprotsendid, ümardatud sajandikeni)
round(prop.table(table(df$page_type, df$sentiment), 1) * 100, 2)

# Seosekordaja Crameri V ja p-väärtuse arvutamine: seos Facebooki lehe tüübi ja meelestatuse vahel
round(cramer.v(table(df$page_type, df$sentiment)), 2) # Crameri V, ümardatud sajandikeni
chisq.test(table(df$page_type, df$sentiment)) # p-väärtus

# Dispersioonanalüüs ehk ANOVA: interaktsioonide arv ja meelestatus
tulemus1 = aov(interactions ~ factor(sentiment), data=df)
summary(tulemus1)

# Tukey HSD test
```

```
TukeyHSD(tulemus1)
```

KASUTATUD KIRJANDUS

Alghamdi, R., & Alfalqi, K. (2015). A Survey of Topic Modeling in Text Mining. *International Journal of Advanced Computer Science and Applications 6*(1), 147–153. Kasutatud: https://www.researchgate.net/publication/276327703_A_Survey_of_Topic_Modeling_in_Text_Mining

Digital Methods Initiative (2015). *The research browser.* Kasutatud: https://www.youtube.com/watch?v=bj65Xr9GkJM

Ecma International (2017). *Standard ECMA-404. The JSON data interchange syntax.* Kasutatud: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

Facebook (2020a). *Graph API.* Kasutatud: https://developers.facebook.com/docs/graph-api

Facebook (2020b). *Page feed.* Kasutatud: https://developers.facebook.com/docs/graph-api/reference/v7.0/page/feed

Jünger, J., & Keyling, T. (2020). *What is Facepager?* Kasutatud: https://github.com/strohne/Facepager/wiki

Kaalep, H.-J., & Muischnek, K. (2002). *Eesti kirjakeele sagedussõnastik.* Tartu: Tartu Ülikooli Kirjastus. Kasutatud: https://www.cl.ut.ee/ressursid/sagedused/index.php?lang=et

Kalmus, V. (2015). Standardiseeritud kontentanalüüs. *Sotsiaalse analüüsi meetodite ja metodoloogia õpibaas.* Kasutatud: http://samm.ut.ee/kontentanalyys

Kolnes, M. (2020). Sissejuhatus R-i ja RStudiosse. *Sotsiaalse analüüsi meetodite ja metodoloogia õpibaas.* Kasutatud: http://samm.ut.ee/sissjuhatus-r-i-ja-rstudiosse

Liu, B. (2015). *Sentiment analysis: Mining opinions, sentiments, and emotions.* Cambridge: Cambridge University Press.

Pandey, A. (2013). *How to predict the topic of a new query using a trained LDA model using gensim?* Kasutatud: https://stackoverflow.com/questions/16262016/how-to-predict-the-topic-of-a-new-query-using-a-trained-lda-model-using-gensim/29218397

Python Software Foundation (2020). *About.* Kasutatud: https://www.python.org/about/

Rogers, R. (2019). *Doing digital methods.* London: Sage.

Rootalu, K. (2014). Risttabelid ja seosekordajad. *Sotsiaalse analüüsi meetodite ja metodoloogia õpibaas.* Kasutatud: http://samm.ut.ee/risttabelid-ja-seosekordajad

Salvet, S. (2020). Hoiakud rände ja migrantide suhtes eestikeelsetes sotsiaalmeediapostitustes. RITA-RÄNNE projekt.

Sotsiaalse analüüsi meetodite ja metodoloogia õpibaas. (2014). Kasutatud: http://samm.ut.ee/avaleht

Tooding, L.-M. (2014) Dispersioonanalüüs. *Sotsiaalse analüüsi meetodite ja metodoloogia õpibaas.* Kasutatud: http://samm.ut.ee/dispersioonanalyys

TÜ arvutiteaduse instituudi programmeerimise algkursuse õpik. (2017). Kasutatud: https://progeopik.cs.ut.ee/index.html

Uiboaed, K. (2017). *Lemmatize txt-files in the working directory. Output only the first analysis.* Kasutatud: https://github.com/kristel-/Tekstikoolitus-2017/blob/master/15-02-praktikum/lemmatizeFiles.py

Wang, C., Paisley, J., & Blei, D. M. (2011). Online variational inference for the hierarchical Dirichlet process. *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS),* Fort Lauderdale, FL, USA, pp. 752–760. Kasutatud: http://proceedings.mlr.press/v15/wang11a/wang11a.pdf

What does tf-idf mean? [2020]. Kasutatud: http://www.tfidf.com/