KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

Vilius Pranckaitis

LIETUVIŠKŲ NAUJIENŲ GRUPAVIMO ALGORITMŲ TYRIMAS

Baigiamasis magistro projektas

Vadovas

Dr. Mantas Lukoševičius

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

LIETUVIŠKŲ NAUJIENŲ GRUPAVIMO ALGORITMŲ TYRIMAS

Baigiamasis magistro projektas
Informatika (kodas 621I10003)

Vadovas
(parašas) Dr. Mantas Lukoševičius
(data)
Recenzentas
(parašas) Doc. dr. Rita Butkienė
(data)

Projektą atliko
(parašas) Vilius Pranckaitis
(data)

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS

Informatikos fakultetas

(Fakultetas)

Vilius Pranckaitis

(Studento vardas, pavardė)

Informatika, 621I10003

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto „Lietuviškų naujienų grupavimo algoritmų tyrimas“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 _____ m. _____ _____ d.

Kaunas

Patvirtinu, kad mano, **Viliaus Pranckaičio**, baigiamasis projektas tema „Lietuviškų naujienų grupavimo algoritmų tyrimas“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

_____ _____

(vardą ir pavardę įrašyti ranka) *(parašas)*

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

Pranckaitis, Vilius. Lietuviškų naujienų grupavimo algoritmų tyrimas. Magistro baigiamasis projektas / vadovas dr. Mantas Lukoševičius; Kauno technologijos universitetas, Informatikos fakultetas.

Mokslo kryptis ir sritis: fiziniai mokslai, informatika.

Reikšminiai žodžiai: dokumentų klasterizavimas; požymių atranka; lietuviškų naujienų straipsniai; k-vidurkių metodas; hierarchinis klasterizavimas.

Kaunas, 2017. 48 p.

SANTRAUKA

Šiame darbe tiriamas dokumentų klasterizavimo procesas, taikant jį naujienų straipsniams iš trijų didžiųjų lietuviškų naujienų portalų. Darbo metu nagrinėjami įvairūs klasterizavimo aspektai, pradedant požymių atrankos procesu ir baigiant k-vidurkių bei hierarchinio klasterizavimo metodų palyginimu. Tyrimo metu pasiūlyta metrika, skirta įvertinti, kaip gerai skirtingi žodžiai apibūdina klasteryje esančių straipsnių turinį. Taip pat pasiūlytas dviejų lygių klasterizavimo metodas, apjungiantis hierarchinį ir k-vidurkių algoritmus. Tyrimo rezultatai parodė, kad *TF–IDF* ir kamienizavimas ženkliai pagerino klasterizavimo kokybę, lyginant su paprastu *TF* ar neatliktu kamienizavimu. K-vidurkių algoritmas parodė geresnius klasterizavimo rezultatus nei hierarchiniai metodai bei buvo atsparesnis požymių erdvės mažinimui pasitelkiant žodžių filtravimą. Pasiūlytas dviejų lygių klasterizavimas parodė neblogus rezultatus, tačiau kokybe neprilygo k-vidurkių algoritmui.

Pranckaitis, Vilius. *Clustering of Lithuanian News Articles*: Master‘s thesis in Computer Science / supervisor Ph.D. Mantas Lukoševičius. The Faculty of Informatics, Kaunas University of Technology.

Research area and field: Physical Sciences, Computer Science.

Key words: document clustering; feature selection; Lithuanian news articles; k-means; hierarchical clustering.

Kaunas, 2017. 48 p.

SUMMARY

This work studies document clustering application for clustering news articles from three major Lithuanian news sites. Different aspects of clustering are studied, including feature selection and comparison of k-means and hierarchical clustering algorithms. This study proposes a metric for measuring how well particular words describe the contents of the cluster. In addition, a two level clustering method was proposed, combining hierarchical and k-means algorithms. The results show that *TF–IDF* with stemming produce significantly better results than simple *TF* and/or no stemming. Also, k-means produced better quality clustering than hierarchical methods and was less sensitive to feature space reduction. The proposed two level clustering showed promising results, however, clustering quality didn’t match the one produced by k-means algorithm.

TERMINŲ IR SANTRUMPŲ ŽODYNAS

Kamienizavimas	Procesas, kurio metu iš žodžio gaunamas jo dalis be galūnės, priešdėlio ir / arba priesagos
Klasteris	Panašių ar panašaus tipo objektų visuma
Dokumentų klasterizavimas	Tekstinių dokumentų suskirstymas į klasterius
Mašininis mokymasis	(*angl. machine learning*) informatikos sritis, kuri tiria algorimus, skirtus atlikti spėjimus apie duomenis, kuriuos apdorojo (kuriais buvo „apmokyti“)
Prižiūrimas mokymasis	(*angl. supervised learning*) mašininio mokymosi būdas, kai programai pateikiama įvestis ir norimas gauti rezultatas
Neprižiūrimas mokymasis	(*angl. unsupervised learning*) mašininio mokymosi būdas, kai turimuose duomenyse bandoma atrasti paslėptas struktūras
Nuolatinis algoritmas	(*angl. online algorithm*) algoritmas, kurio vykdymo pradžioje prieinama tik dalis įvesties, o algoritmo vykdymo metu dalimis paduodami tolimesni duomenys
Požymių atranka	(*angl. feature selection*) procesas, kurio metu iš visų objekto požymių išrenkami tik aktualūs
Požymių išgavimas	(*angl. feature extraction*) procesas, kurio metu iš tam tikrų objekto požymių sukonstruojami nauji požymiai
Temų modeliavimas	(*angl topic modeling*) mašininio mokymosi metodas, kuris dokumentų aibėje randa abstrakčias „temas“ – dokumentams bendrus bruožus
Klaidų įverčiai	Skaitinės charakteristikos, kurios nusako, kaip lyginamas objektas skiriasi nuo pavyzdinio
Leksema	Reikšminis kalbos vienetas
Morfologija	Mokslas apie kalbos dalis, žodžių sandarą ir jų darybą

1. ĮVADAS

Per pastaruosius keletą dešimtmečių drastiškai pasikeitė tai, kaip žmonės perduoda duomenis. Iki kompiuterių eros informacija įprastai buvo perduodama pasitelkiant spaudą, radiją ir televiziją. Visi šie kanalai turi jiems būdingų apribojimų. Pavyzdžiui, spausdintos informacijos perdavimo sparta priklauso nuo to, kaip sparčiai įmanoma informaciją atspausdinti ir fiziškai nugabenti skaitytojui. Informacija, perduodama radiju ir televizija, keliavo kur kas sparčiau, tačiau šiais kanalais perduodamos informacijos kiekis turėjo tiek techninių apribojimų (tik ribotas skaičius radijo stočių ir televizijos kanalų galėjo būti transliuojamas vienu metu), tiek buvo ribojami adresato sugebėjimo priimti duomenis.

Situacija pasikeitė atsiradus internetui. Nors iš pradžių egzistavo panašios duomenų perdavimo spartos ir pralaidumo problemos, kaip ir kitose informacijos perdavimo kanaluose, tačiau per visą interneto vystymo laiką buvo atlikta didžiulė pažanga. Dabar internetas suteikia prieigą prie didžiulio informacijos kiekio, kuris iki šiol nebuvo pasiekiamas jokiais kitais būdais. Augant interneto plėtrai jis tapo neatsiejama kiekvieno civilizuoto žmogaus gyvenimo dalis. Kartu neatsiejami tapo ir didžiuliai informacijos kiekiai, pasiekiami per internetą.

Žmogus, priešingai nei kompiuteris, nėra pajėgus apdoroti didelius informacijos kiekius. Tačiau jei informacija susisteminta, žmogus gali lengvai atmesti jam neaktualią informaciją, taip sumažindamas dėmesio reikalaujančios informacijos kiekį.

Šio darbo metu tiriami įvairūs metodai ir algoritmai, skirti tekstinių dokumentų klasterizavimui į grupes. Klasterizavimui pasirinkti tekstai iš naujienų portalų, dėl jų generuojamo didelio informacijos srauto (didesni naujienų portalai išleidžia daugiau kaip šimtą straipsnių per dieną). Taip pat naujienų portalai yra viena iš pagrindinių vietų, per kur naujienos pasiekia vartotojus.

1.1. Darbo tikslas ir uždaviniai

Šio darbo tikslas – ištirti dokumentų klasterizavimo metodus, taikomus naujienų straipsniams iš lietuviškų naujienų portalų. Darbo uždaviniai:

1. Ištirti, kokią įtaką galutiniams rezultatams daro pirminis lietuviškų tekstų apdorojimas;
2. Ištirti klasterizavimo metodų pranašumus ir trūkumus;
3. Įvertinti algoritmų tinkamumą lietuviškų naujienų agregatoriaus kūrimui.

Šio darbo mokslinis naujumas kyla iš to, kad tyrimui naudojami tekstai yra parašyti lietuvių kalba. Didžioji dalis tyrimų apima dokumentų klasterizavimo metodų taikymą anglų kalbos tekstams. Taip pat naujienų portalų straipsniai turi savitą kalbos stilių ir žodyną, kas taip pat gali daryti įtaką tyrimo rezultatams.

1.2. Dokumento struktūra

Toliau dokumente pateikiamas šio darbo aprašymas. 2-ame skyriuje apžvelgiami klasterizavimo metodai ir atlikti tyrimai. 3-iame skyriuje pateikiamas šiam tyrimui skirtos programinės įrangos projektas. Su realizacija susijusi informacija pateikiama 4-ame skyriuje. 5-ame skyriuje apžvelgiami tyrimo rezultatai. Darbo išvados pateikiamos 6-ame skyriuje.

2. DOKUMENTŲ KLASTERIZAVIMO METODŲ ANALIZĖ

Klasterizavimas – tai objektų skirstymas į grupes, vadinamas klasteriais. Objektai skirstomi taip, kad viename klasteryje esantys objektai būtų panašesni tarpusavyje nei objektai, esantys skirtinguose klasteriuose. Klasterizavimas naudojamas mašininiame mokyme (*angl. machine learning*), paveikslėlių analizėje, struktūros aptikime (*angl. pattern recognition*), bioinformatikoje, informacijos paieškoje (*angl. information retrival*).

Dokumentų klasterizavimas yra atskira klasterių analizės šaka, kuri nagrinėja tekstinės informacijos grupavimą. Šioje srityje sunkumų sukelia tai, kad apdorojama ne struktūrizuota informacija, bet natūralios kalbos tekstai. Pavyzdžiui, sudėtinga tekstus pateikti formatu, kuris būtų tinkamas klasterizavimo algoritmams. Standartinis būdas būtų sukonstruoti vektorių $D=\{w_1, w_2, \dots, w_m\}$, kur kiekviena dimensija atitinka tam tikrą žodį w_i, o jų vertės nurodo žodžio w_i santykį su tekstu (yra / nėra ar žodžio pasikartojimo skaičių dokumente). Tačiau šaltinyje [1] pateikiama keletas charakteringų savybių, kodėl toks metodas nėra itin veiksmingas:

- Tekstiniams dokumentams pavaizduoti reikia itin daug dimensijų, tačiau patys duomenys yra reti. Kitaip sakant, skirtingų žodžių skaičius, naudojamas tam tikros tematikos literatūroje gali būti 10^5 eilės, tačiau konkretų dokumentą gali sudaryti tik keli šimtai skirtingų žodžių.
- Nors ir tam tikros tematikos dokumentuose pasitaikantis žodynas gali būti platus, tačiau įprastai žodžiai koreliuoja tarpusavyje. Todėl esminių komponentų skaičius yra kur kas mažesnis nei požymių erdvė (*angl. feature space*).
- Žodžių skaičius skirtinguose dokumentuose gali smarkiai skirtis, todėl svarbu atitinkamai normalizuoti dokumentų reprezentacijas.

Dėl šių priežasčių itin svarbus tampa pirminis teksto apdorojimas. Siekiant geresnių rezultatų, požymių išgavimui (*angl. feature extraction*) privalu skirti ne ką mažesnį dėmesį nei patiems klasterizavimo metodams (pavyzdžiui, dokumentų rezultatus galima pagerinti prieš tai randant žodžių klasterius [2]; iš tekstų atmetus didelę dalį informacijos, galima ne tik išlaikyti klasterizavimo kokybę, bet ir ją pagerinti [3]).

Toliau šiame skyriuje aptariamas pirminis teksto apdorojimas (2.1 skyrius), dokumentų klasterizavimo metodai (2.2 skyrius), priemonės klasterizavimo kokybei įvertinti (2.3 skyrius), kitų autorių atlikti darbai klasterizuojant lietuviškus tekstus (2.4 skyrius) ir rinkoje esantys naujienų agregatoriai (2.5 skyrius).

2.1. Pirminis teksto apdorojimas

Pirminį teksto apdorojimą galima skirstyti į dvi grupes: *požymių atranka* (*angl. feature selection*) ir *požymių išgavimas* (*angl. feature extraction*). Požymių atrankos atveju paimama tik dalis objekto informacijos. Požymių išgavimo atveju esama informacija yra transformuojama, taip gaunant naujus objektą charakterizuojančius požymius. Abiejų šių procesų tikslas yra išskirti požymius, kurie geriausiai charakterizuoja objektą.

2.1.1. Požymių atranka tekstiniams dokumentams

Požymių atrankos metu iš tekstinio dokumento gaunama kita, klasterizavimo metodams tinkamesnė, reprezentacija. Įprastai dokumentas pavaizduojamas kaip daugiamatis vektorius, kurio dimensijos atitinka žodžius ar teminus, o jų vertės – sąryšį tarp dokumento ir atitinkamo žodžio (nors priklausomai nuo klasterizavimo metodo, gali būti naudojama grafo [4] ar kita reprezentacija). Sąryšiui tarp dokumento ir termino nusakyti gali būti naudojami įvairūs modeliai.

Vienas iš dokumento reprezentacijos modelių – *termino dažnumo* (*angl. term frequency, TF*) modelis. Be jau minėtų paprastų binarinės (1 atitinka termino buvimą dokumente, 0 – nebuvimą) ir kiekybinės (reikšmė atitinka termino pasikartojimų dokumente skaičių) versijos, kur kas populiaresnė santykinio dažnumo versija. Šiuo atveju termino pasikartojimo skaičius padalinamas iš

dokumente esančių terminų skaičiaus. Tokiu būdu išvengiama neproporcingo įvertinimo, kai lyginami didelės ir mažos apimties dokumentai (pavyzdžiui, žodis „klasterizavimas“ gali būti pavartotas po penkis kartus tiek šimto, tiek tūkstančio žodžių ilgio tekstuose).

TF modelis turi trūkumą, kad neįvertina žodžio pasikartojimo visoje dokumentų aibėje. Akivaizdu, kad žodis „ir“ bus kur kas dažnesnis už žodį „klasterizavimas“ bet kokios srities tekstuose. Šiuo atveju tinkamesnis *atvirkštinio dažnumo dokumentuose* (*angl. inverse document frequency*, *IDF*) modelis. Jame atitinkamai sumažinami įverčiai terminams, kurie ir taip pasitaiko dažnai. Galimas ir jungtinis šių dviejų modelių variantas *TF-IDF*.

Be jau paminėtų galimi ir kiti dokumento pavaizdavimo modeliai. Šaltinyje [3] palyginimui panaudota *TF*, *informacijos prieaugio* (*angl. information gain*), χ^2 (*chi* kvadratu), *termino stiprumo* (*angl. term strength*), *entropija paremto įvertinimo* (*angl. entropy-based ranking*) ir *termino įnašo* (*angl. term contribution*) įverčiai. Daugiau informacijos apie šiuos modelius pateikta šaltiniuose [1, 3].

Turimų modelių informaciją galima panaudoti informacijos kiekio sumažinimui. Žodžiai, kurie labai retai pasikartoja tekstuose, nesuteikia naudingos informacijos lyginant keletą tekstų tarpusavyje ir veikia kaip triukšmas [1]. Tyrimo, aprašyto šaltinyje [3], metu pavyko gauti geresnius klasterizavimo rezultatus pašalinus 90 % ir daugiau požymių iš tekstų (pašalinimas atliktas pritaikant įvairius dokumento pavaizdavimo modelius ir pašalinant požymius, neviršijančius tam tikros vertės). Šaltinyje [1] teigiama, kad labai dažnai pasitaikantys terminai taip pat turėtų būti pašalinami ir kad *TF-IDF* būtent tai ir atlieka.

Dalį požymių iš tekstinio dokumento reprezentacijos galima pašalinti ir remiantis kalbos savybėmis. Kai kurie žodžiai klasterizavimo požiūriu yra nereikšmingi. Tokie žodžiai angliškai vadinami *stop words*. Pavyzdžiui, žodžiai „ir“, „o“, „bet“, „tačiau“, „kadangi“ atlieka jungiamąjį sakinio dalių vaidmenį ir dokumentų palyginimo atveju nesuteikia beveik jokios informacijos. Be šių, taip pat prieš apdorojimą galima pašalinti kitus teksto elementus, tokius kaip skaičius, matavimo vienetų sutrumpinimus ir panašiai.

2.1.2. Požymių išgavimas tekstiniuose dokumentuose

Požymių išgavimo atveju esama objekto informacija transformuojama siekiant sukurti naujus požymius, kurie geriau charakterizuotų objektą. Tam galima panaudoti įvairias objekto savybes. Tekstų atveju galima pritaikyti kalbos gramatiką. Pavyzdžiui, žodžiai „klasterizavimas“, „klasterizavimo“ ir „klasterizavimui“ iš esmės reiškia tą patį, skiriasi tik žodžio linksniai. Tačiau imant dokumento *TF-IDF* šie žodžiai būtų laikomi visiškai nesusijusiais. Tokius informacijos praradimus, kylančius iš to, kad žodis gali turėti daug formų, galima sumažinti kamienizuojant kiekvieną žodį. Tokiu atveju anksčiau minėti žodžio „klasterizavimas“ linksniai būtų pakeisti vienu „klasterizavim“ su atitinkamai didesniu dažniu. Tiesa, negalima vienareikšmiškai nusakyti, koks turėtų būti kamienizavimo laipsnis – galima būtų sutrumpinti žodį iki „klasteriz“ taip įtraukiant ir veiksmažodžius („klasterizuoti“, „klasterizavo“) arba kamienizuoti dar agresyviau iki „klaster“, taip apimant ir žodžio „klasteris“ formas.

Siekiant išgauti geresnius požymius, galima pasinaudoti savybe, kad terminai gali koreliuoti tarpusavyje. Taip gali atsitikti dėl pačios kalbos savybių (pavyzdžiui, žodžiai „apsiauti“ ir „batai“ dažniausiai aptinkami kartu) ar dalykinei sričiai būdingų bruožų (pavyzdžiui, kalbant apie atomo sandarą žodžiai „protonas“, „neutronas“ ir „elektronas“ dažnai vartojami kartu). Reikia pastebėti, kad priklausomai nuo teksto tematikos, žodžių koreliavimas gali keistis (pavyzdžiui, kalbant apie elektrinius reiškinius žodis „elektronas“ gali vis tiek būti vartojamas neužsimenant apie „protonus“ ir „neutronus“).

Vienas iš būdų, kuriuo galima aptikti žodžių koreliavimą, yra žodžių klasterizavimas. Šis metodas buvo pritaikytas atliekant tyrimą, aprašytą šaltinyje [2]. Šiame darbe iš pradžių buvo klasterizuojami patys terminai, taip nustatant terminus, kurie tekstuose dažnai aptinkami kartu.

Vėliau ši informacija buvo panaudota klasterizuojant dokumentus. Pagal šaltinyje [2] pateiktus rezultatus, tokia dvigubo klasterizavimo technika pranoko tris bandytus viengubo klasterizavimo metodus.

Požymių išgavimui gali būti pritaikyti ir matricų skaidymo dauginamaisiais metodai. Šaltinyje [1] minimi du tokie metodai: *latent semantic indexing* (*LSI*) ir *non-negative matrix factorization* (*NMF*). Šie metodai dokumentų–žodžių matricą aproksimuoja kelių matricų sandauga. Nuo metodo priklauso gautų matricų savybės. Abu metodai naudojami sumažinti požymių erdvės dimensijų skaičiui. Dimensijų skaičiaus sumažinimas *LSI* metodu leidžia sumažinti sinonimijos ir polisemijos (tas pats žodis turi kelias reikšmes) sukeliamą triukšmą [1]. Papildomai, *NMF* metodas gali būti panaudotas žodžių klasterių nustatymui [1].

2.2. Dokumentų klasterizavimo algoritmai

Turint dokumentų pavaizdavimą su atrinktais ir išskirtais požymiais, galima šiems dokumentams pritaikyti klasterizavimo algoritmus. Egzistuoja įvairios klasterizavimo metodų rūšys, paremtos skirtingais modeliais ir principais. Kai kurie algoritmai reikalauja specifinio požymių atrinkimo ir išgavimo. Šaltinyje [1] išskiriamos tokios klasterizavimo algoritmų ir metodų klasės:

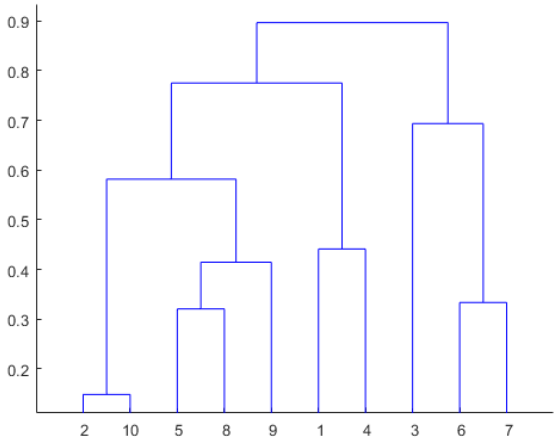
- paremti panašumu;
- žodžių ir frazių klasterizavimas;
- tikimybiniai;
- nuolatiniai (*angl. online*) algoritmai tekstų srautams;
- tekstinių dokumentų interneto tinkluose;
- pusiau prižiūrimas (*angl. semi-supervised*) klasterizavimas.

Toliau bus aptariami šių klasių algoritmai ir metodai.

2.2.1. Panašumu paremti klasterizavimo algoritmai

Panašumu paremtų klasterizavimo algoritmų veikimas remiasi funkcijomis, leidžiančiomis nusakyti dokumentų panašumą. Žinant, kad viena dokumentų pora yra panašesnė tarpusavyje nei kita dokumentų pora, galima atitinkamai tuos dokumentus paskirstyti į klasterius. Toliau šiame skyriuje bus aptariami keletas panašumu paremtų klasterizavimo algoritmų, o skyriuje 2.2.1.1 „Panašumo įverčiai“ įvardinti galimi panašumo įverčiai.

Viena iš šios klasės algoritmų grupių yra hierarchiniai klasterizavimo algoritmai. Šie algoritmai iteratyviai suranda porą panašiausių dokumentų klasterių (galimai sudarytų ir iš vieno dokumento) ir juos apjungia tarpusavyje. Šis procesas kartojamas tol, kol visa aibė dokumentų apjungiama į vieną klasterį (galimas ir atvirkštinis variantas, kai pradžioje turimas vienas klasteris, apimantis visus dokumentus, dalijamas į smulkesnius klasterius kol galiausiai kiekvienas dokumentas lieka atskirame klasteryje). Turint tokį hierarchinį klasterių suskirstymą jį galima pavaizduoti dendrogramoje – medžio tipo diagramoje, parodančioje klasterių apjungimo eilės tvarką. Dendrogramos pavyzdys pateiktas 2.1 paveikslėlyje.



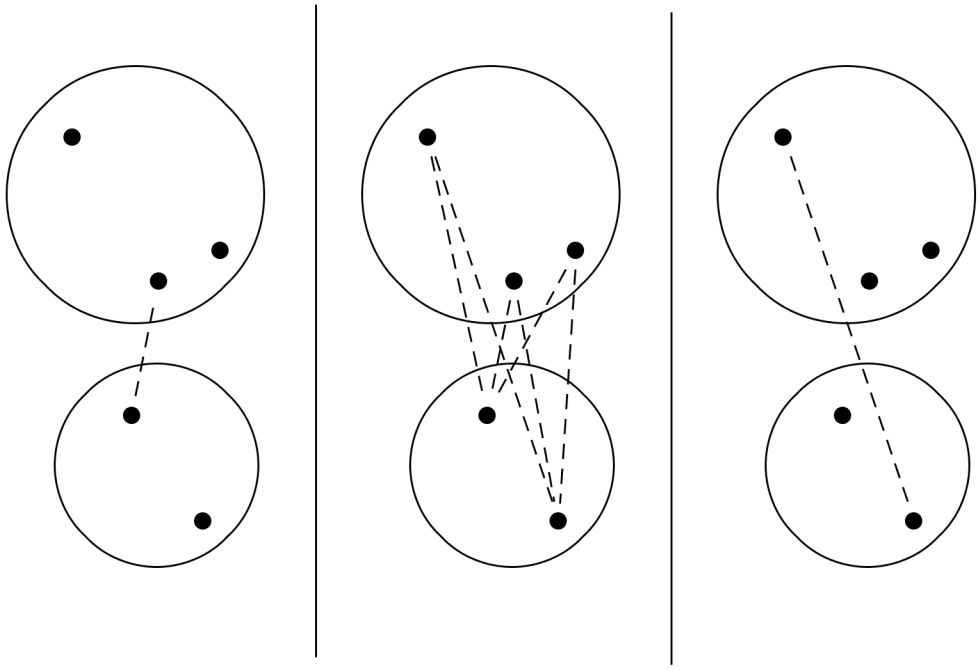
2.1 pav. Dendrogramos pavyzdys (šaltinis: https://mathworks.com)

Hierarchinio klasterizavimo rezultatas stipriai priklauso nuo pasirinkto klasterių tarpusavio panašumo įvertinimo metodo. Šaltinyje [1] įvardijami trys tokie metodai:

- *vienos jungties* – kai paimami visi dokumentai iš vieno klasterio ir sudaromos visos įmanomos poros su dokumentais iš kito klasterio. Šių klasterių panašumas bus lygus dokumentų poros, turinčios mažiausią tarpusavio atstumą, panašumui (žiūrėti 2.2

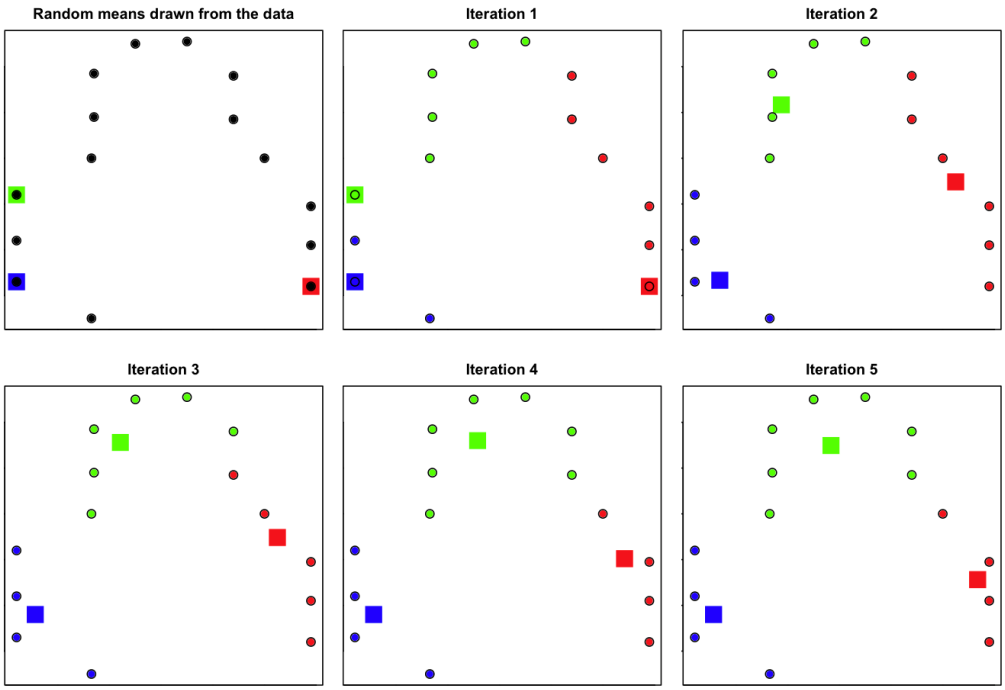
paveikslėlį). Toks metodas turi trūkumą, nes vienam klasteriui gali būti priskirta dokumentų grandinėlė, kur gretimi dokumentai tarpusavyje panašūs, tačiau kraštiniai nariai labai skirtingi [1]. Šaltinyje [5] pateikiamas algoritmas, kuris leidžia vienos jungties hierarchinį klasterizavimą atlikti per $O(n^2)$ laiką, kur n – dokumentų skaičius;

- *grupių vidurkių* – kai panašumas tarp klasterių atitinka dokumentų porų panašumų vidurkį. Šis metodas išsprendžia dokumentų grandinėlės problemą, tačiau išauga reikalingų skaičiavimų kiekis. Taip yra dėl to, kad norint nustatyti klasterių tarpusavio panašumą reikia apskaičiuoti tarp tų klasterių esančių dokumentų porų panašumų vidurkį (priešingai vieno jungties atvejui, kai užtenka vienos poros dokumentų, turinčios didžiausią panašumą). Siekiant paspartinti skaičiavimus, klasterių tarpusavio panašumą galima aproksimuoti surandant kiekvieno klasterio elementų vidurkį ir apskaičiuojant panašumą tarp vidutinio kiekvieno klasterio elemento. Tokiu atveju algoritmo sudėtingumas sumažėja iki $O(n^2)$. Tokia optimizacija tinka ne visiems duomenų tipams, tačiau veikia gana gerai su tekstiniais dokumentais [1];
- *pilnos jungties* – tai metodas panašus į *vienos jungties*, tačiau klasterių panašumui nusakyti pasirenkama ne artimiausia, bet tolimiausia dokumentų pora. Šiuo atveju taip pat išvengiama dokumentų grandinėlės problemos [1]. Šaltinyje [5] pateikiamas algoritmas, skirtas pilnos jungties hierarchiniam klasterizavimui rasti, kurio sudėtingumas $O(n^2 \log n)$.



2.2 pav. Hierarchinio klasterizavimo klasterių tarpusavio panašumo įvertinimo metodai: (iš kairės) vienos jungties, grupės vidurkio ir pilnos jungties

Be hierarchinio klasterizavimo dar žinomi atstumu (atstumas yra atvirkštinis dydis panašumui) paremti klasterizavimo algoritmai. Vienas jų – *k-vidurkių* (*angl. k-means*) algoritmas. Šis algoritmas parenka pradinius *k* dokumentų, kurie atitiks klasterių centrus. Tada kiekvienos iteracijos metu kiekvienas dokumentas priskiriamas prie jam artimiausio (t. y. jam panašiausio) klasterio centro, o vėliau visi centrai pakoreguojami, kad atitiktų prie klasterio priskirtų dokumentų vidurkiui. Šie veiksmai kartojami tol, kol galiausiai sukonverguojama į būseną, kuri daugiau nebekinta. Paveikslėlyje 2.3 pateiktas k-vidurkių algoritmo veikimo pavyzdys.



2.3 pav. K-vidurkių algoritmo veikimo pavyzdys (šaltinis: http://compprag.christopherpotts.net/)

K-vidurkių algoritmai gali turėti įvairių atmainų. Standartinis k-vidurkių algoritmas iš pradžių parenka atsitiktinius k klasterių centrų, o vėliau juos nustato į vidurkį pagal atstumą Euklidinėje erdvėje. Tuo tarpu *sferinis k-vidurkių* (*angl. spherical k-means*) algoritmas perskaičiuodamas centro koordinatę siekia gauti vidurkį pagal vektoriaus kampą (t. y. minimizuoti kosinuso atstumą tarp centro ir klasterio dokumentų). *Dalijantis* (*angl. bisecting*) *k-vidurkių* algoritmas pradeda darbą su vienu klasteriu, apimančiu visus dokumentus, ir vėliau kiekvienos iteracijos metu dalija didžiausią klasterį į du naujus, tam panaudodamas k-vidurkių algoritmą su dviem centrais.

K-vidurkių algoritmas jautrus pradinių centrų pasirinkimui, ypač dokumentų klasterizavimo atveju [1]. Todėl taikomos specialios priemonės jų parinkimui. Kita vertus, k-vidurkių metodas (algoritmo sudėtingumas $O(kn)$, kur k – pasirinktų centrų skaičius, o n – dokumentų skaičius) kur kas spartesnis už hierarchinį (priklausomai nuo klasterių palyginimo metodo ir realizacijos, sudėtingumas gali siekti $O(n^3)$) [1].

Siekiant išvengti hierarchinio ir atstumu paremto skaidymo algoritmų trūkumų, buvo pasiūlytas *išsklaidyti-surinkti* (*angl. scatter-gather)* algoritmas [6]. Šis algoritmas pradinių centrų parinkimui naudoja hierarchinio klasterizavimo metodą. Parinkus centrus toliau vykdomas k-vidurkių algoritmas. Dokumentų skaičius, naudojamas centrų nustatymui, parenkamas taip, kad hierarchinio algoritmo sudėtingumas neviršytų k-vidurkių algoritmo sudėtingumo. Toks hibridinis metodas padeda išvengti netinkamai parinktų pradinių centrų, tačiau ir neviršija k-vidurkių algoritmui būdingo sudėtingumo.

2.2.1.1. Panašumo įverčiai

Norint, kad dokumentai būtų klasterizuojami pagal jų panašumą, reikalingos priemonės panašumui įvertinti. Kadangi įprasta dokumentus pavaizduoti kaip vektorius, tai galima pasitelkti metrikas, kurios nusakytų panašumą (arba atstumą) tarp požymių vektorių. Šaltinis [7] pateikia keletą panašumui nustatyti tinkamų įverčių:

- Euklidinis atstumas,
- kampo kosinuso panašumas,
- Džakardo koeficientas,
- Pearsono koreliacijos koeficientas,
- Kullback–Liberer divergencija.

Iš šių įverčių literatūroje dažnai naudojamas kampo kosinuso panašumas [8, 9, 10], kuris būtent ir atitinka kampo, esančio tarp dviejų požymių vektorių, kosinusą. Šaltinyje [7] aprašyto tyrimo metu buvo nustatyta, kad blogiausius rezultatus duoda Euklido atstumas, o kiti įverčiai duoda gana panašius rezultatus. Šią išvadą atitinka šaltinyje [11] aprašyto tyrimo rezultatai (tiesa, šiame darbe netirta Kullback–Liberer divergencija).

2.2.1.2. Klasterių hierarchijos suplokštinimo algoritmai

Hierarchiniai algoritmai pasižymi tuo, kad sudaroma klasterių hierarchiją. Tam tikrais atvejais ši ypatybė gali būti naudinga, tačiau naujienų agregatoriaus atveju priimtinesni plokšti klasteriai ar nedidelio lygių skaičiaus hierarchija (hierarchinių algoritmų atveju hierarchijos lygių skaičius atitinka elementų skaičių). Be to, tyrimo metu siekiama tarpusavyje palyginti hierarchinio ir k-vidurkių algoritmo sudarytus klasterius, ko nebūtų galima padaryti nesuplokštinus klasterių hierarchijos.

Literatūroje galima rasti įvairių algoritmų, kurie iš klasterių hierarchijos sudaro plokščius klasterius. Vienas iš geriausiai žinomų yra *DBSCAN* (*angl. density-based spatial clustering of applications with noise*) algoritmas [12]. Šis algoritmas parenka klasterius atsižvelgiant į elementų tankumą. Algoritmo veikimui reikalingi keli parametrai, vienas iš kurių yra maksimalus atstumas tarp dviejų elementų, kad šie elementai galėtų būti prijungiami prie bendro klasterio. Tačiau šis parametras yra neparankus klasterizuojant tekstinius dokumentus, nes sunku nusakyti optimalią parametro vertę, o ir atstumai tarp požymių vektorių labai priklauso nuo požymių atrankos procedūros.

Parankesnėmis savybėmis pasižymi *HDBSCAN* algoritmas [13]. Šio algoritmo veikimui reikalingas tik vienas parametras – minimalus elementų skaičius klasteryje. Elementai, kurie nesuformuoja reikiamo dydžio klasterio, priskiriami triukšmui. Kitas *HDBSCAN* privalumas – priešingai nei *DBSCAN*, šis algoritmas gali rasti skirtingo tankumo klasterius.

2.2.2. Žodžių ir frazių klasterizavimas

Skyriuje 2.1.2 trumpai aptarta idėja, kad tekstiniuose dokumentuose žodžiai tarpusavyje koreliuoja ir kad būtų galima sumažinti dimensijų skaičių radus tarpusavio ryšius tarp jų. Būtent tai, kad dokumentai suvokiami ne kaip pavienių, tarpusavyje nesusijusių žodžių rinkiniai, ir išnaudojama žodžių ir frazių klasterizavimo algoritmuose. Šaltinyje [1] šie algoritmai sugrupuoti į keturias kategorijas: klasterizavimas su dažnai pasitaikančių terminų rinkiniais, žodžių klasterizavimas, ko-klasterizavimas ir frazėmis paremtas klasterizavimas.

Nors kiekvienas dokumentas turi skirtingą terminų aibę, tačiau tam tikri terminų rinkiniai gali būti aptinkami keletoje dokumentų. Tokie dažni terminų rinkiniai gali būti laikomi kaip nusakantys klasterius [1], svarbu tinkamai juos parinkti. Šaltinyje [14] aptariamos strategijos, kaip parinkti pasikartojančių terminų rinkinius, bei pateikiami hierarchinio ir plokščio klasterizavimo algoritmai, kurie naudoja dažnus terminų rinkinius klasterių parinkimui.

Kita galima strategija – prieš atliekant dokumentų klasterizavimą suklasterizuoti žodžius. Tai galima atlikti pavaizdavus žodžius kaip vektorių, kurio dimensijos atitinka dokumentus, o reikšmės – žodžio pasikartojimus atitinkamame dokumente. Turint šią reprezentaciją žodžiai suklasterizuojami taip, kad informacijos nuostolis būtų kuo mažesnis. Vėliau, pasinaudojant *dokumento–žodžių klasterių* reprezentacija, atliekamas antrasis klasterizavimas, vėlgi stengiantis išlaikyti kuo daugiau informacijos. Toks dviejų etapų klasterizavimas smarkiai sumažina duomenyse esantį triukšmą [1]. Aprašymas, kaip minimizuoti kiekvieno etapo metu prarandamą informaciją, bei keleto metodų palyginimas viengubo ir dvigubo klasterizavimų atvejais pateiktas šaltinyje [2]. Pateikto tyrimo metu gauta, kad beveik visais atvejais dvigubo klasterizavimo technika pagerino rezultatus lyginant su atitinkamu viengubu klasterizavimu.

Ko-klasterizavimo metodas panašus į dvigubo klasterizavimo metodą. Pagrindinis skirtumas – priešingai nei dvigubo klasterizavimo atveju, žodžių ir dokumentų klasteriai gaunami ne pažingsniui (iš pradžių žodžių klasteriai, vėliau – dokumentų), bet vieno žingsnio metu (algoritmas vienu metu randa ir žodžių, ir dokumentų klasterius). Šaltinyje [1] išskiriami du dažnai literatūroje pasitaikantys ko-klasterizavimo metodai:

- *ko-klasterizavimas pasitelkiant dvidalio grafo padalinimą*. Šis metodas remiasi tuo, kad terminų ir dokumentų tarpusavio santykį galima pavaizduoti kaip dvidalį grafą, kur briaunos atitinka sąryšį *terminas–dokumentas*. Siekiant gauti terminų ir dokumentų klasterius, reikėtų atlikti grafo padalijimą. Padalijimo metu siekiama minimizuoti briaunų, kurios jungia skirtingas grafo padalijimo dalis, svorių sumas (t. y. minimizuojami grafo dalių tarpusavio ryšiai). Tai – standartinis grafų teorijos uždavinys. Šaltinyje [4] aprašoma, kaip ko-klasterizavimas atliekamas pasitelkiant *spektriniu grafo padalijimo* algoritmu;
- *ko-klasterizavimas pasitelkiant informacijos teoriją*. Šio metodo metu siekiama sumažinti informacijos nuostolius, atsiradusius klasterizavimo metu. Klasterizavimo metu terminų aibė X priskiriama terminų klasterių aibei X^*, o dokumentų aibė Y priskiriama dokumentų klasterių aibei Y^*. Kadangi tiek X^*, tiek Y^* yra aukštesnio lygio reprezentacija, tai atsiranda bendros informacijos nuostolis [1]. Ko-klasterizavimo metu siekiama parinkti klasterius X^* ir Y^* taip, kad informacijos nuostolis $I(X,Y)-I(X^*,Y^*)$ būtų kuo mažesnis.

Visi šiame skyriuje aptarti metodai nekreipia dėmesio į tai, kaip žodžiai išdėstyti tekste. Tokius metodus dar vadina „žodžių maišo“ *(angl. bag of words*) metodais. Frazėmis paremto klasterizavimo atveju dokumentas laikomas kaip vienas po kito einančių žodžių seka. Šaltinyje [1] išskiriami trys žingsniai, naudojami nustatant dokumentų klasterius šiuo metodu:

1. Pirmo žingsnio metu atliekamas dokumento žodžių išvalymas – pažymimos sakinių ribos, atliekamas kamienizavimas, pašalinami nereikšmingi elementai (skaičiai, skyrybos ženklai ir pnš.);
2. Sudaromi baziniai klasteriai. Tam panaudojamas sufiksų medis. Kiekviena šio medžio viršūnė atitinka tam tikrą frazę (paeiliui einančių terminų seką) bei dokumentų rinkinį, kuriame randama ši frazė. Tokius dokumentų rinkinius galima laikyti baziniais klasteriais. Labiau pageidaujami tie baziniai klasteriai, kurie atitinka ilgą frazę, apjungiančią daug dokumentų;
3. Kadangi baziniai klasteriai nenusako griežto sudalijimo (t. y. dokumentas gali priklausyti keliems baziniams klasteriams), vykdomas bazinių klasterių apjungimas. Baziniai klasteriai apjungiami pagal jiems bendrų dokumentų skaičių. Po apjungimo persidengimų skaičius sumažinamas, bet ne panaikinamas. Šaltinyje [1] teigiama, kad tai gali praversti atvejais, kai nedideli persidengimai leidžiami siekiant pagerinti klasterizavimo kokybę.

2.2.3. Tikimybėmis paremti metodai

Tikimybėmis paremti klasterizavimo metodai susiję su temų modeliavimu (*angl. topic modeling*). Tam sukuriamas modelis, kurio parametrai nustatomi pasitelkiant turimą dokumentų rinkinį. Temų modeliavimas remiasi prielaida, kad turint temų rinkinį galima nustatyti kiekvieno dokumento priklausymo tikimybę tam tikrai temai. Šias temas galima laikyti kaip atitikimą klasteriams. Temų modeliavimas neduoda griežto dokumentų suskirstymo, nes dokumentai turi tikimybę priklausyti kelioms temoms. Todėl gaunamas *minkštas klasterizavimas* (*angl. soft clustering*). Temų modeliavimo proceso vienas iš produktų yra matrica, nusakanti kiekvieno dokumento tikimybę priklausyti kiekvienai iš temų. Papildomai gaunama matrica, parodanti kiekvieno termino (iš visų apdorotų dokumentų) tikimybę priklausyti tam tikros temos žodynui.

Šaltinyje [1] aptariami du temų modeliavimo metodai – *tikimybinį latentinį semantinį indeksavimą* (*angl. Probabilistic Latent Semantic Indexing, PLSI*) ir *latentinį Dirichlė paskirstymą* (*angl. Latent Dirichlet Allocation*, *LDA*). Šie metodai tarpusavyje panašūs, pagrindinis skirtumas – temos–dokumento ir temos–termino tikimybių modeliavimui *LDA* naudoja Dirichlė skirtinį, kai

PLSI naudoja atsitiktines reikšmes. *LDA* modelis laikomas pranašesniu už *PLSI*, nes yra mažiau linkęs į persimokymą (*angl. overfitting*) bei gali geriau modeliuoti naujus dokumentus, kurių nebuvo apmokymo rinkinyje [1]. Nepaisant to, šaltinyje [15] aprašyto tyrimo rezultatai parodė, kad abu metodai duoda panašius rezultatus.

2.2.4. Kitos klasterizavimo metodų grupės

Prieš tai aptarti dokumentų klasterizavimo metodai tinka visiems tekstiniams dokumentams. Tačiau tam tikrais atvejais pačių dokumentų formatas ir šaltinis gali reikalauti ar įgalinti specializuotų metodų panaudojimą. Keletas pavyzdžių: nuolatinis (*angl. online*) tekstų klasterizavimas, tekstų klasterizavimas interneto tinkluose ar pusiau prižiūrimas (*angl. semi-supervised*) klasterizavimas.

Nuolatinio dokumentų klasterizavimo atveju iššūkių sukelia tai, kad esamu laiku nėra prieinamas visas duomenų rinkinys, tačiau jis nuolatos auga vis pateikiant naujus dokumentus. Tokiu atveju klasterizavimo algoritmas privalo sugebėti adaptuoti sudarytus klasterius bėgant laikui ir atsirandant naujiems duomenims. Šaltinyje [1] pateikiama keletas technikų: kai dokumento įtaka klasterių sudarymui keičiama pagal dokumento amžių, klasterizuojama pagal nedidelį kiekį laikinai padažnėjusių požymių, kurie nuolat atnaujinami, ar naudojamas temų modeliavimas.

Internetiniai dokumentai pasižymi tuo, kad šalia teksto jie taip pat dažnai turi nuorodas į kitus šaltinius. Tokius sąryšius galima panaudoti klasterizavimo kokybės pagerinimui. Vienas iš būdų būtų palyginti originalų dokumentą su jo nuorodose esančiais dokumentais ir pagal tai pakoreguoti požymių atrankos metu terminams skiriamus svorius. Sudėtingesnė strategija būtų iš nuorodų sudaryti grafo struktūrą ir šią informaciją panaudoti klasterių parinkimui. Informacija apie dokumentų tarpusavio ryšius taip pat gali būti panaudota ir temų modeliavimui [1].

Pasitaiko atvejų, kai kartu su tekstiniu dokumentu prieinama ir **tekstą apibendrinanti informacija** (pavyzdžiui, kategorija, kuriam dokumentas priskirtas, ar dokumentui suteikti raktažodžiai). Tokia informacija galėtų praversti klasterizavimo metu (pavyzdžiui, kai siekiama suklasifikuoti dokumentus į stambesnes ar smulkesnes kategorijos nei šiuo metu jie suskirstyti ar kai skiriasi dokumentų kategorizacija dėl skirtingų jų šaltinių). Šaltinyje [1] aptariama keletas pusiau prižiūrimo klasterizavimo metodų.

2.3. Klaidų įverčiai

Siekiant apibendrinti ir palyginti tarpusavyje klasterizavimo rezultatus, būtinos priemonės gautų rezultatų kokybės įvertinimui. Šiam tikslui naudojami įvairūs statistiniai klaidų įverčiai. Tiesa, daugumai tokių įverčių klasterizuojami tekstai privalo būti sužymėti pagal kategorijas, nes įvertinant lyginama kiek tiksliai gauti klasteriai atitinka šias kategorijas. Reikia pastebėti, kad toks įvertinimo būdas nėra idealus, nes daroma prielaida, kad tekstui priklausanti kategorija (tikėtina priskirta žmogaus) yra teisinga. Taip pat kartais tekstas gali būti susijęs su keliomis kategorijomis (pavyzdžiui, prezidento vizitas svečioje šalyje gali būti pažymimas tiek kategorijomis „politika“, tiek „užsienis“), tačiau jam gali būti priskirta tik viena iš jų. Tokiu atveju tiek žmogus, tiek klasterizavimo algoritmas gali įtraukti tekstą į skirtingas grupes abu būdami daugiau mažiau teisūs, tačiau vertinant klasterizavimo kokybę tai bus laikoma klaida.

Literatūroje galima rasti panaudotus įvairius klaidų įverčius. Dažniausiai pasirenkami bent du įverčiai, kurių pagalba palyginami rezultatai. Vienas populiaresnių įverčių yra *entropija* [3, 7, 8, 10, 11]. Klasterio C_i, kurio dydis n_i, entropija yra

$$E(C_i) = \frac{1}{\log c} \sum_{h=1}^{k} \frac{n_i^{(h)}}{n_i} \log\left(\frac{n_i^{(h)}}{n_i}\right), \tag{2.1}$$

kur c yra kategorijų skaičius, o $n_i^{(h)}$ – dokumentų iš kategorijos h, priskirtų į klasterį C_i, skaičius. Entropija įvertina kategorijų pasiskirstymą duotame klasteryje [7].

Kitas dažnai naudojamas įvertis yra *grynumas* (*angl. purity*) [7, 10, 11]:

$$purity(C_i) = \frac{1}{n_i} \max_h \left(n_i^{(h)}\right). \quad (2.2)$$

Grynumas parodo, kokią dalį klasterio sudaro elementai iš dominuojančios (turinčios didžiausią narių skaičių tame klasteryje) kategorijos. Šis įvertis yra mažiau išsamus nei entropija, nes neįvertina nedominuojančių kategorijų, esančių klasteryje, pasiskirstymo [7].

Be jau minėtų metrikų, literatūroje galime aptikti naudojamą *F1 balą* (*angl. F1–score*) [8, 9, 10]. Jo įvertis gaunamas iš *tikslumo* (*angl. precision*)

$$\mathrm{P} = \frac{tp}{tp + fp} \quad (2.3)$$

ir *atkūrimo* (*angl. recall*) įverčių

$$\mathrm{R} = \frac{tp}{tp + fn}, \quad (2.4)$$

čia tp – *tinkamai atrinkti* elementai (*angl. true positives*), fp – *netinkamai atrinkti* elementai (*angl. false positives*), fn – *neatrinkti tinkami* elementai (*angl. false negatives*). Klasterizavimo atveju šie parametrai skaičiuojami kiekvienai porai elementų: jei du elementai priklauso tai pačiai kategorijai ir tam pačiam klasteriui, ši pora priskaičiuojama prie tp, jei kategorijos sutampa, bet klasteriai skiriasi – priskaičiuojama prie fn, ir t. t.

Tikslumas parodo santykį tinkamai atrinktų elementų su viso klasterio dydžiu. Atkūrimas parodo tinkamai atrinktų elementų santykį su visais elementais, esančiais atitinkamoje kategorijoje. *F1* balas sukombinuoja abu šiuos įverčius (2.3) ir (2.4) į vieną formulę

$$\mathrm{F1} = \frac{2\,\mathrm{P}\,\mathrm{R}}{\mathrm{P} + \mathrm{R}}. \quad (2.5)$$

Tikslumo ir atkūrimo įverčiai yra linkę įgauti dideles reikšmes, kai yra labai didelis ar labai mažas klasterių skaičius (pavyzdžiui, atkūrimas įgauna maksimalią vertę, kai visi elementai atrenkami į vieną klasterį). Šiuos įverčius sukombinavus atveriami jų polinkiai pervertinti kraštutinumus, todėl *F1* balas nepasižymi pastariems įverčiams būdingomis problemomis.

2.4. Lietuviškų tekstų klasterizavimas

Didžioji dalis dokumentų klasterizavimo tyrimų atlikta su anglų kalbos tekstais. Kur kas mažesnis literatūros kiekis prieinamas apie lietuviškų tekstų klasterizavimą. Literatūros analizės metu peržvelgta keletas įvairių darbų, kuriuose duomenų rinkinius sudaro lietuviški tekstai:

- Kauno technologijos universiteto baigiamieji magistro darbai apie individualiai pasirenkamas tekstų kategorizavimo sistemas [16] ir interneto naujienų agregatoriaus prototipą [17];
- dokumentų klasterizavimo metodų tyrimai [9, 10], atlikti Vytauto Didžiojo universitete;
- klasterizavimo metodų palyginimas ieškant panašumų tarp tekstinių dokumentų [18], atliktas Vilniaus universitete.

Nors šie darbai turėjo skirtingus tikslus ir buvo tiriami skirtingi klasterizavimo metodai (išskyrus *k-vidurkių* algoritmą, kuris buvo panaudotas visuose šiuose darbuose), daugumoje iš jų pastebėta, kad stipriai sumažinus požymių skaičių (ar tai būtų atliekama kamienizuojant žodžius, pašalinant žodžius pagal sąrašą ar atmetant terminus pagal jų dažnumą) galima ne tik išlaikyti, bet ir pagerinti klasterizavimo kokybę. Ši išvada atitinka 2.1.1 skyriuje aptartus tyrimus anglų kalbos tekstams.

2.5. Rinkoje egzistuojančių naujienų straipsnių agregavimo sprendimų apžvalga

Internete galima rasti jau egzistuojančių sprendimų, kurie į vieną vietą surenka naujienų straipsnius iš įvairių portalų. Šių sprendimų apžvalgą apsunkina faktas, kad informacija apie tai, kaip atliekamas straipsnių rūšiavimas ir grupavimas, dažniausiai nėra viešai prieinama. Dėl šios

priežasties objektyviai galima aptarti patį straipsnių pateikimo principą, o ne sistemos automatizavimo laipsnį (kiek grupavimo ir rūšiavimo procesas reikalauja žmogaus įsikišimo), naudojamus metodus ir jų detales.

Vienas iš puslapių, agreguojančių naujienų straipsnius iš įvairių šaltinių, yra „Microsoft“ priklausantis „Bing News“. Jame naujienos suskirstomos į įvairias kategorijas: „pasaulis“, „verslas“, „politika“, „pramogos“ ir pnš. Kartu su straipsnio pavadinimu taip pat pateikiamas ir trumpas jo aprašymas. Paspaudus ant pasirinkto straipsnio nukreipiama į atitinkamą straipsnį originalioje publikacijoje.

Nors dokumentų klasifikavimas yra atskiras uždavinys nuo klasterizavimo, tačiau abu šie uždaviniai yra itin artimi (terminais „*neprižiūrimas* klasifikavimas“ ir „klasterizavimas“ apibūdinamas tas pats procesas). Klasifikuojant naujienų straipsnius viena iš galimybių yra pasikliauti kategorijomis, į kurias straipsniai suskirstyti originaliuose šaltiniuose. Tačiau toks naivus sprendimas galimai neduotų gerų rezultatų, nes:

- skirtingi šaltiniai turi skirtingas kategorijų aibes;
- skirtingi šaltiniai gali skirtingai traktuoti į kurią kategoriją turėtų patekti tos pačios tematikos straipsnis;
- kai kurios originalaus šaltinio kategorijos gali būti per siauros ar per plačios naujienų agregatoriaus kontekste (pavyzdžiui, puslapyje „15min.lt“ yra atskira kategorija pokeriui).

Šias problemas galima būtų spręsti automatiškai klasifikuojant straipsnius į naują kategorijų aibę, galimai tam pritaikant ir dokumentų klasterizavimą.

Įvairūs naujienų portalai kartu su straipsniu vartotojui pateikia ir sąrašą susijusių ar panašių straipsnių. Atitinkamas naujienų agregatoriaus pavyzdys – puslapis „memeorandum.com“. Jame straipsniai, kuriuose kalbama apie tą patį įvyki, pateikiami kartu, vienas straipsnis – kaip vedantysis, ir kiti – kaip susiję straipsniai.

Reikia atkreipti dėmesį, kad straipsnių panašumą vieno šaltinio ribose galima įvertinti rankiniu būdu ar ieškant panašių raktažodžių (dažnai kartu su straipsniu pateikiami keletas raktinių žodžių, kurie geriausiai atspindi straipsnio turinį). Esant keletai skirtingų šaltinių grupavimas pagal raktažodžius gali būti neparankus dėl atitinkamų priežasčių kaip ir klasifikavimas pagal originalaus šaltinio priskirtas kategorijas. Tokiu atveju praverstų automatizuotas būdas kaip surasti pasirinktam straipsniui pagal turinį panašiausius kitus straipsnius. Klasterizavimo metodai būtent tai ir atlieka – suskirsto objektų aibę į tam tikrą skaičių grupių taip, kad kiekvienos grupės elementai būtų kuo panašesni tarpusavyje.

Abu aukščiau aprašytus straipsnių grupavimo metodus turi portalas „Google News“. Šiame puslapyje straipsniai, aprašantys tą patį įvykį, pateikiami kartu. Taip leidžiama vartotojui pirmiausia išsirinkti norimą naujieną, o tik tada pasirinkti vieną iš susijusių naujienų straipsnių pagal straipsnio aprašymą ar publikacijos šaltinį. Tokie nedideli straipsnių klasteriai papildomai susisteminti į keletą kategorijų. „Google News“ išsiskiria iš anksčiau paminėtų portalų tuo, kad jame apdorojami straipsniai ne vien anglų, bet ir kitomis kalbomis, įskaitant ir lietuvių. Tačiau galima tik spekuliuoti, ar „Google News“ taikomi klasterizavimo metodai priderinti prie konkrečios kalbos savybių ir bruožų, ar naudojamas apibendrintas, visoms kalboms tinkantis sprendimas.

Be jau paminėtų naujienų agregatorių galima rasti ir daug kitų: „Feedly“, „Fark.com“, „SmartNews“, „Flipboard“, „News 360“ ir kt. Vieni iš jų viso labo leidžia pasirinkti norimus naujienų šaltinius, kiti klasifikuoja naujienas pagal tematiką ar sugrupuoja pagal panašumą, treti stebi vartotojo skaitomus straipsnius ir pagal tai parenka naujienų srautą, kuris labiausiai atitiktų vartotojo pomėgius. Nepaisant to, kaip šie naujienų agregatoriai pateikia turinį, dauguma jų yra orientuoti į naujienų šaltinius, pateikiančius turinį anglų kalba.

Sąrašas naujienų agregatorių, išskirtinai orientuotų į lietuviškus naujienų portalus, itin menkas. Atlikus paiešką internete galima rasti užuominas į kelis tokius puslapius, kurie šio tiriamojo darbo metu jau nebeegzistuoja. Iš egzistuojančių galima paminėti „visosnaujienos.lt“. Šis puslapis surenka straipsnius iš įvairių lietuviškų naujienų portalų ir pateikia jų antraštes chronologine tvarka. Šiame

puslapyje taip pat yra „svarbiausių naujienų“ skiltis, kurioje naujienos sugrupuojamos pagal aprašomus įvykius. Tačiau peržvelgus šios skilties turinį galima prieiti išvados, kad grupavimas vykdomas pagal antraščių tarpusavio panašumą.

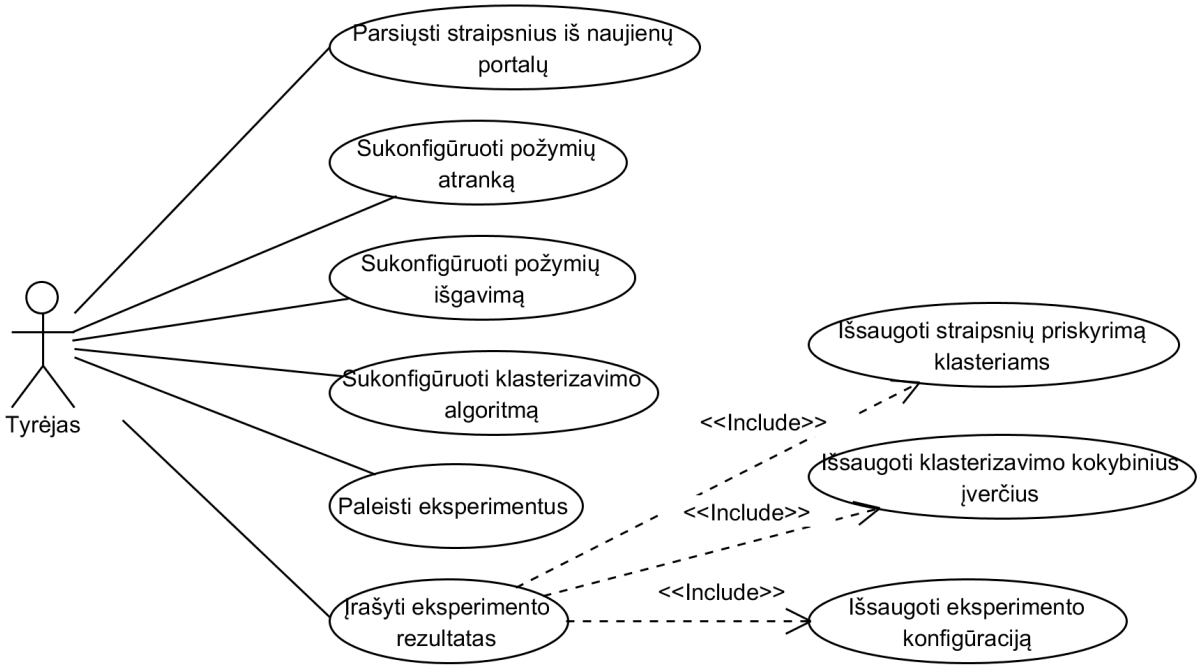
3. TYRIMUI SKIRTOS PROGRAMINĖS ĮRANGOS PROJEKTAVIMAS

3.1. Reikalavimai programinei įrangai

Programinei įrangai keliami šie **funkciniai reikalavimai**:

- Galimybė parsisiųsti ir išsaugoti naujienų straipsnius iš įvairių naujienų portalų;
- Galimybė konfigūruoti požymių atrankos proceso žingsnius;
- Galimybė konfigūruoti požymių išgavimo proceso žingsnius;
- Galimybė pasirinkti vieną iš keleto klasterizavimo algoritmų ir valdyti jų parametrus;
- Galimybė pasirinkti vieną iš keleto duomenų rinkinių eksperimentui;
- Galimybė išsaugoti eksperimento konfigūraciją ir rezultatus;

3.1 paveikslėlyje pateikta funkcinius reikalavimus atitinkanti panaudos atvejų diagrama.



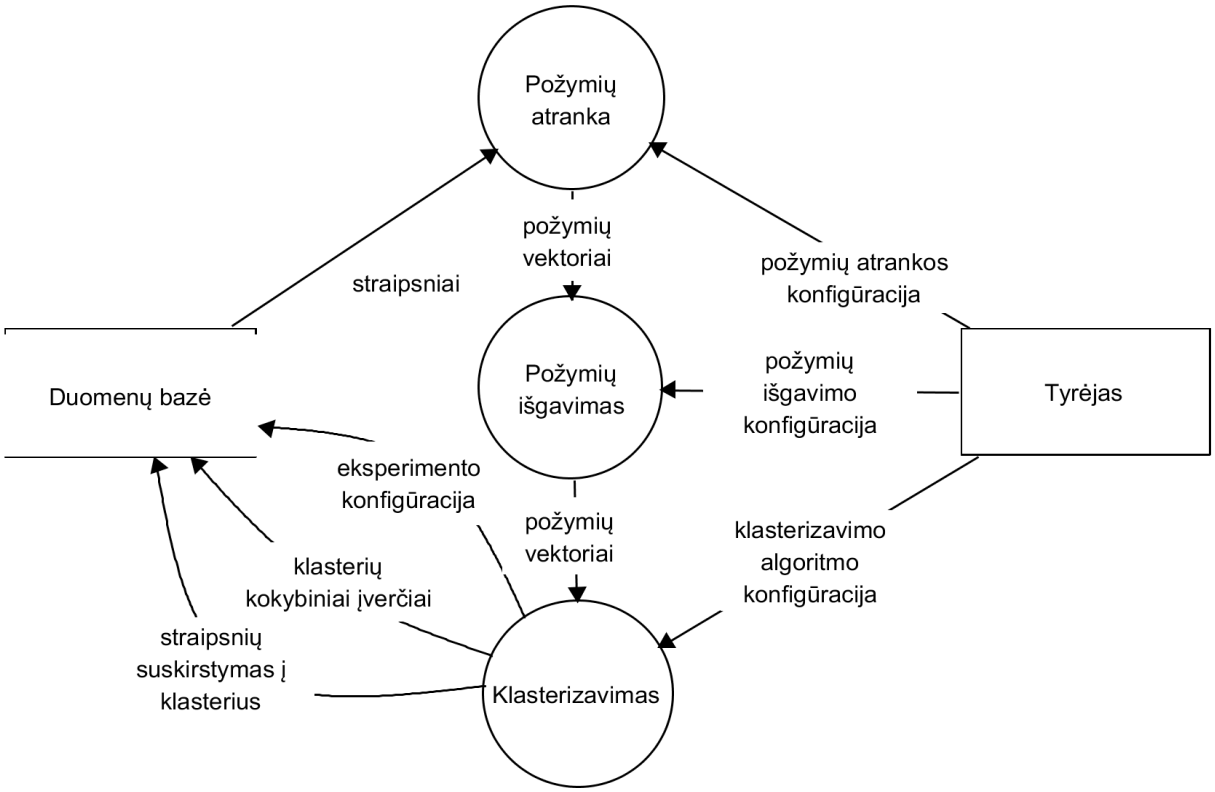
3.1 pav. Panaudos atvejų diagrama

Programinei įrangai keliami šie **nefunkciniai reikalavimai**:

- Eksperimentų konfigūracijų valdymui sukuriama *aplikacijos programavimo sąsaja* (*angl. API*) pasirinktoje programavimo kalboje;
- Programa turi automatiškai registruoti eksperimento konfigūraciją bei pasibaigus eksperimentui išvesti eksperimento parinktis struktūrizuotu tekstiniu formatu;
- Programinė įranga privalo automatiškai apskaičiuoti bei išvesti eksperimento rezultatų įvertinimo metrikas: grynumą (*angl. purity*), tikslumą (*angl. precission*), atkūrimą (*angl. recall*) ir *F1* balą (*angl. F1-score*);
- Pageidautina, kad eksperimento rezultatai būtų saugojami ir kaupiami automatiškai po kiekvieno eksperimento;

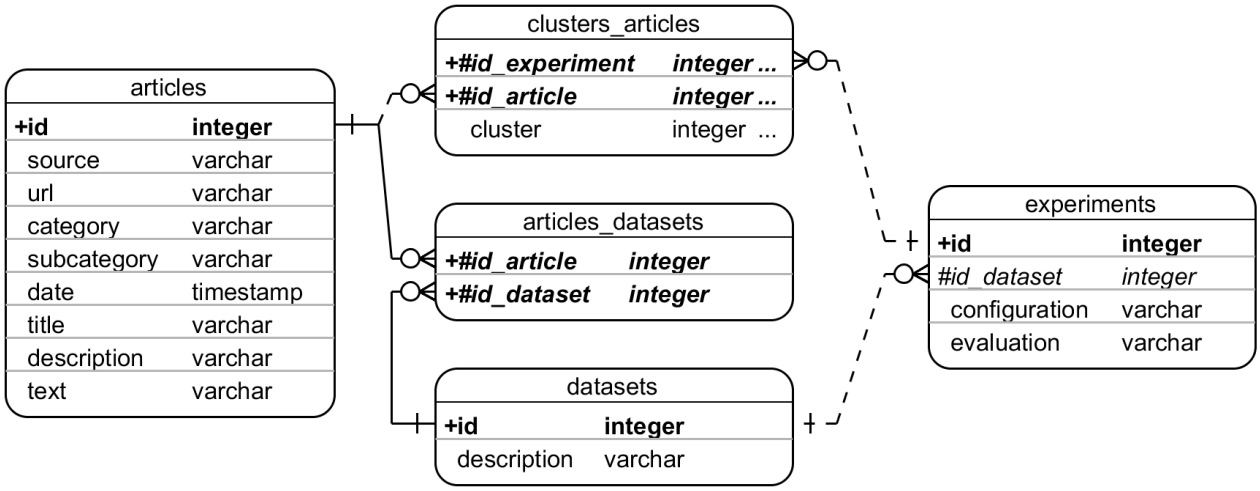
3.2. Duomenų modelis

Kuriama programinė įranga apdoros naujienų straipsnius, paimtus iš pasirinktų naujienų portalų. Visą klasterizavimo procesą galima suskirstyti į tris žingsnius: požymių atranka, požymių išgavimas ir duomenų klasterizavimas. Eksperimentų metu bus išbandomos įvairios šių žingsnių konfigūracijos, taip siekiant išsiaiškinti jų poveikį eksperimento rezultatams. Kiekvieno eksperimento metu bus išsaugomi gauti dokumentų klasteriai bei užregistruojama eksperimento konfigūracija ir rezultatų kokybiniai įverčiai. 3.2 paveikslėlyje pateikta aptarto proceso duomenų srautų diagrama.



3.2 pav. Duomenų srautų diagrama

3.3 paveikslėlyje pateikta duomenų bazės schema. Joje bus saugoma informacija apie straipsnius bei eksperimentų duomenys. Kiekvienas straipsnis gali būti priskirtas į vieną ar kelis duomenų rinkinius. Taip siekiama užtikrinti, kad skirtingus eksperimentus būtų galima pakartoti su tais pačiais duomenų rinkiniais ir kad tą patį eksperimentą būtų galima atlikti keliems duomenų rinkiniams, nepriklausomai nuo to, kaip pakis duomenų bazėje saugomi straipsniai.



3.3 pav. Duomenų bazės schema

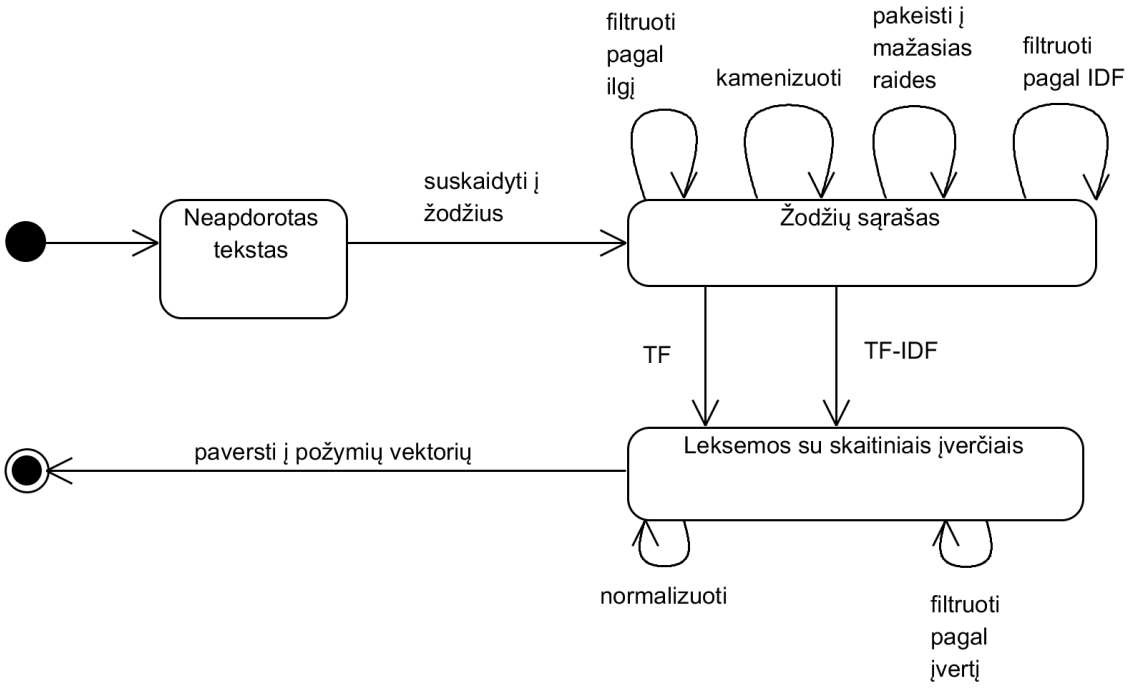
Reikia pastebėti kad *articles* lentelė nėra tinkamai normalizuota. Šis sprendimas buvo pasirinktas sąmoningai. Duomenų bazės normalinės formos apsaugo nuo nepageidaujamų anomalijų atsiradimo, kai į duomenų bazę įtraukiami, keičiami ar pašalinami įrašai. Kadangi straipsnių duomenis eksperimento atlikimo metu bus siekiama išlaikyti statiškus, juos keičiant tik išskirtiniais atvejais, tai priimtas kompromisas tarp duomenų normalizavimo ir supaprastinto duomenų bazės modelio. Atitinkamai nenormalizuota ir lentelė *experiments*, nes ji skirta tik registruoti eksperimentų duomenis – visas rezultatų apdorojimas bus atliekamas su išoriniais įrankiais (pvz. *Microsoft Excel*) į juos išeksportavus duomenis iš šios lentelės. Eksperimento konfigūracija ir įvertinimas (stulpeliai

configuration ir *evaluation*) bus serializuojami į pasirinktą formatą ir saugomi duomenų bazėje tekstiniu pavidalu. Šiuo atveju tiktų *JSON* formatas, nes yra lengvai apdorojamas programiškai ir lengvai konvertuojamas į kitus formatus.

3.3. Statinis sistemos vaizdas

3.3.1. Požymių atrankos ir išgavimo komponento struktūra

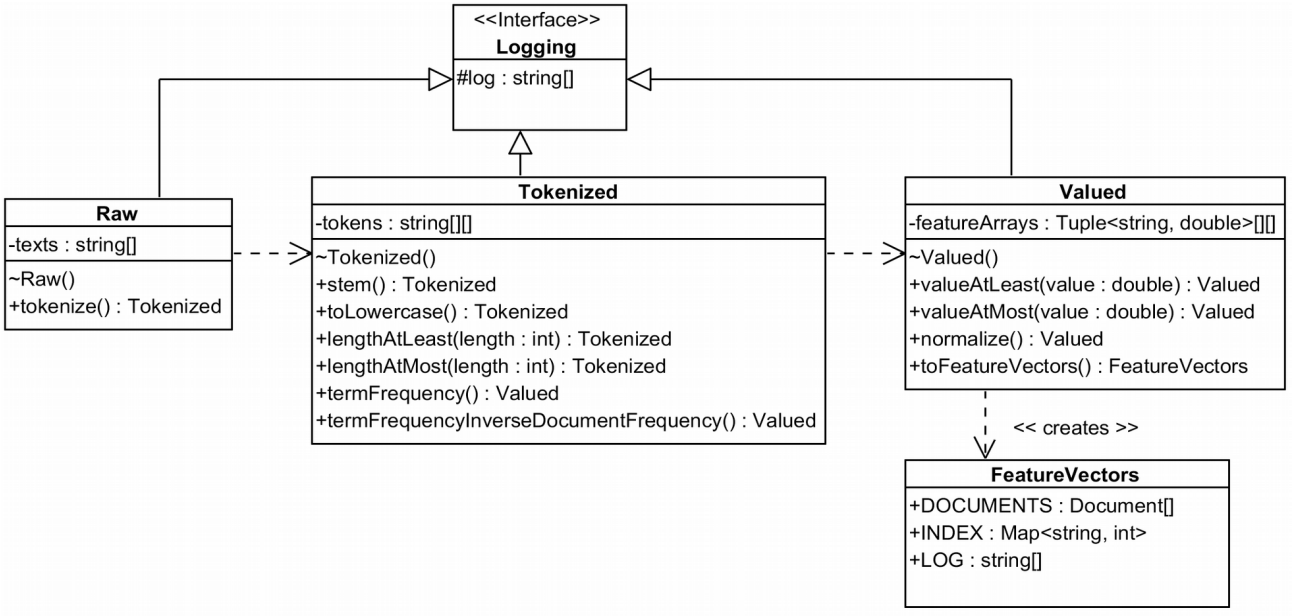
Vienas iš sistemai taikomų reikalavimų yra eksperimento konfigūracijos išvedimas tekstiniu formatu, įskaitant ir požymių atrankos ir išgavimo proceso konfigūraciją. Dėl šios priežasties svarbi griežtai apibrėžta komponento struktūra, kad būtų galima konkrečiai išvardinti visus atliktus žingsnius (pavyzdžiui, teksto sudalinimas į leksemas, kamienizavimas, raidžių pakeitimas į mažąsias). Tačiau taip pat svarbu, kad požymių atrankos procesas būtų lanksčiai konfigūruojamas – priklausomai nuo tyrimo eigos gali prireikti kai kuriuos žingsnius praleisti, sukeisti vietomis ar įtraukti naujų žingsnių. Šiems tikslams pasiekti visas požymių atrankos procesas suskaidytas į atskiras logines stadijas, kurios pavaizduotos būsenų diagramoje 3.4 paveikslėlyje.



3.4 pav. Požymių atrankos ir išgavimo būsenų diagrama

Požymių atrankos procese išskirtos stadijos, kurios tiesiogiai susijusios su apdorojamais duomenimis. Pradinė būsena atitinka neapdorotą naujienų straipsnio tekstą. Į tolimesnę būseną pereinama suskaidant tekstą į leksemas (*angl. tokens*). Turint straipsnio leksemas galima atlikti jų filtravimą ar modifikavimą (pavyzdžiui, pakeisti raides į mažąsias ar pakeisti žodžius į jų kamienus). Kita būsena pasiekiama pritaikius pasirinktas statistines metrikas ir kiekvienai leksemai paskaičiuojant skaitinę reikšmę. Šioje būsenoje atliekami veiksmai su skaitiniais įverčiais (filtravimas ar modifikavimas). Galiausiai iš šio būsenos išeinama suformuojant tolimesniam apdorojimui tinkantį požymių vektorių.

Aukščiau aprašytai struktūrai realizuoti pasirinktas pamodifikuotas *kūrėjo projektavimo šablonas* (*angl. builder design pattern*). 3.5 paveikslėlyje pateikta atitinkama klasių diagrama. Kiekvieną būseną atitinka atskira klasė su atitinkamais metodais. Kiekvieno metodo iškvietimo metu papildomai į masyvą užregistruojama iškviestas metodas ir metodui pateikti parametrai. Šis masyvas perduodamas iš būsenos į būseną, taip užregistruojant visus požymių atrankos žingsnius. Nuosekliai perėjus per visas būsenas grąžinami dokumentų klasterizavimui paruošti požymių vektoriai, jų konstravimui panaudoti žingsniai bei papildoma informacija (pavyzdžiui, indeksas, parodantis, kurią leksemą atitinka kiekviena požymių vektoriaus dimensija).

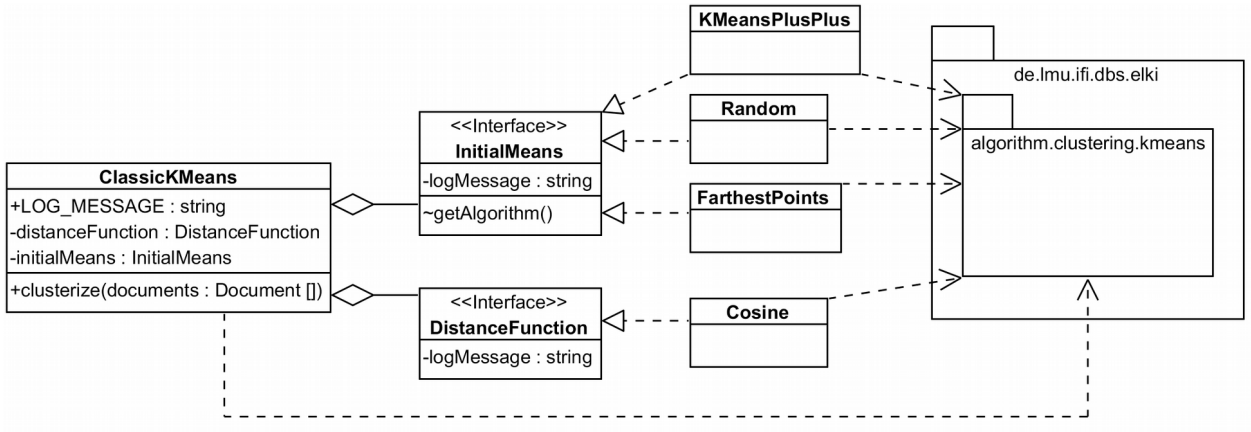


3.5 pav. Požymių atrankos ir išgavimo proceso klasių diagrama

3.3.2. Klasterizavimo komponento struktūra

Straipsnių klasterizavimui bus pasitelkta keli skirtingi klasterizavimo algoritmai. Programoje bus panaudotos jų realizacijos, paimtos iš trečių šalių bibliotekų. Šie algoritmai gali būti iš skirtingų bibliotekų, gali turėti skirtingais formatais perduodamus parametrus ar algoritmo paleidimui gali reikalauti parametrų, kurie nesvarbūs atliekamam tyrimui. Visas šis sąsajų nesutapimas ir konfigūracijos perteklius bus paslėptas pasitelkiant *fasado projektavimo šabloną* (*angl. facade design pattern*). Taip pat fasado sluoksnyje bus realizuota ir konfigūracijos registravimas – pasirinktas algoritmas bei jo paleidimui panaudoti parametrai bus išsaugomi struktūrizuotu tekstiniu formatu. Konfigūracijos aprašymą galima bus išsisaugoti į duomenų bazę.

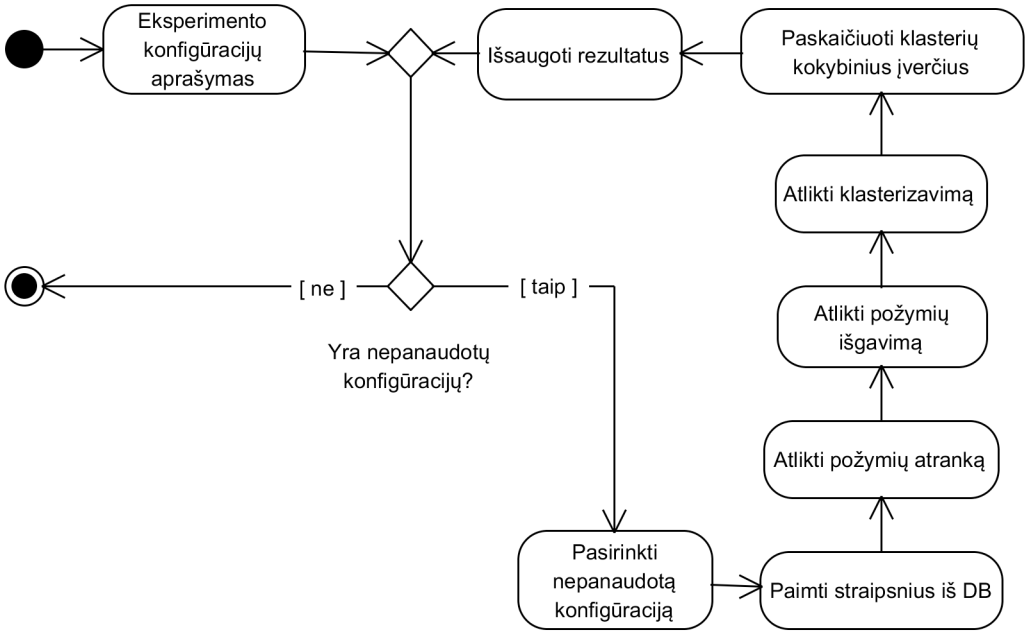
3.6 paveikslėlyje pateikta klasių diagrama, kurioje pavaizduoti elementai, panaudojami k-vidurkių algoritmo sukonfigūravimui. Panaši struktūra bus pritaikyta ir kitiems klasterizavimo algoritmams.



3.6 pav. K-vidurkių algoritmo konfigūravimo klasių diagrama

3.4. Dinaminis sistemos vaizdas

3.7 paveikslėlyje pateikta programos veiklos diagrama. Programos vykdymo pradžioje bus aprašomos eksperimentų konfigūracijos, t. y. naudojami algoritmai bei reikiami parametrai požymių atrankai, požymių išgavimui bei klasterizavimui. Tada nuosekliai bus pereinama per eksperimentų konfigūracijų sąrašą bei kiekvienai iš jų atliekamas testas.

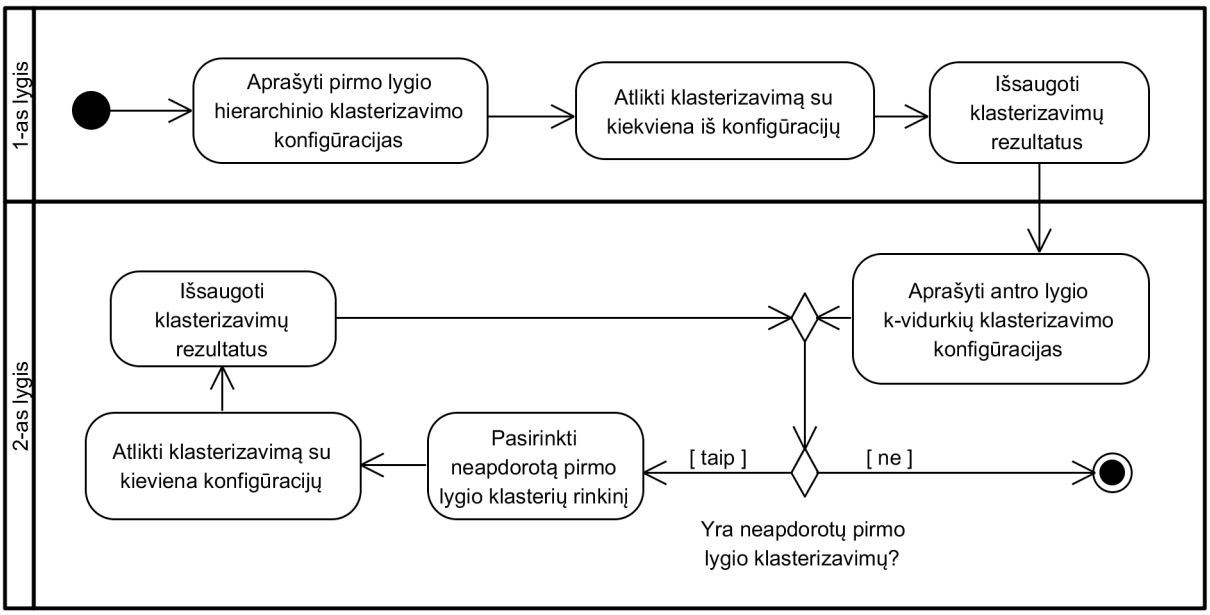


3.7 pav. Klasterizavimo proceso žingsniai

Paveikslėlyje 3.7 pateikta diagrama tik apibendrintai pavaizduoja programos veikimą. Atskiri vykdymo etapai, tokie kaip požymių atranka ar klasterizavimas, detaliau apibūdinami aprašant kiekvieną iš eksperimentų.

3.4.1. Dviejų lygių klasterizavimas

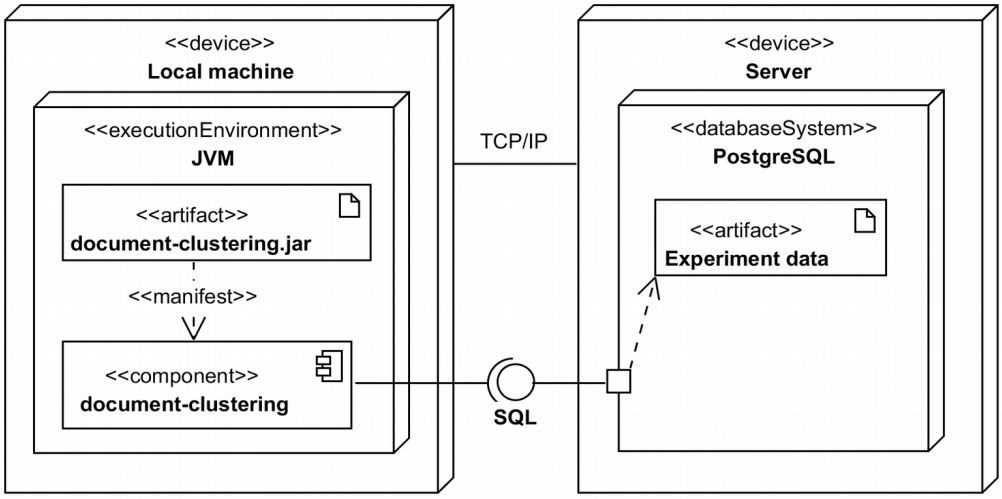
Darbo metu pasiūlytas dviejų lygių klasterizavimo sprendimas. Jo metu klasterizavimas atliekamas dukart: visų pirma dokumentai suklasterizuojami pasitelkiant hierarchinį klasterizavimo algoritmą, o vėliau gauti klasteriai suklasterizuojami pasitelkiant k-vidurkių algoritmą. Kadangi tiek pirmojo, tiek antrojo klasterizavimų metu bus išbandomos skirtingos parametrų konfigūracijos, pasirinkta tokį klasterizavimą vykdyti ne nuosekliai, o per du paleidimus. Pirmo paleidimo metu aprašomos visos pirmam lygiui skirtos konfigūracijos, atliekamas klasterizavimas ir išsaugomi gauti klasteriai. Antro paleidimo metu atskirai aprašomos antro klasterizavimo lygio konfigūracijos bei atliekamas klasterizavimas atitinkamai paduodant vis kitą pirmo lygio klasterizavimo metu gautą klasterių rinkinį. Šio proceso veiklos diagrama pateikta 3.8 paveikslėlyje.



3.8 pav. Dviejų lygių klasterizavimo veiklos diagrama

3.5. Sistemos išdėstymas

Sistemos išdėstymo (*angl. deployment*) diagrama pateikta 3.9 paveikslėlyje. Apibendrinant, sistemą galima suskirstyti į du komponentus – pati programa, kuria bus atliekami tyrimai, ir duomenų bazė, kurioje saugomi naujienų straipsniai ir su eksperimentais susiję duomenys. Duomenų bazė bus patalpinta į serverį, turintį prieigą prie interneto. Tyrimo programa leidžiama iš lokalaus kompiuterio ar, prireikus daugiau skaičiavimo resursų, pasirinkto serverio.



3.9 pav. Sistemos išdėstymo diagrama

4. REALIZACIJA

Šiame skyriuje pateikiama informacija apie sistemos realizavimą ir eksperimentų vykdymą.

4.1. Naudojamos technologijos ir programiniai įrankiai

Sistemai realizuoti pasirinkti tokie įrankiai:

- „Scala“ funkcinio programavimo kalba;
- „ELKI“ duomenų gavybos (*angl. data mining*) biblioteka;
- „PostgreSQL“ duomenų bazė ir „Slick“ biblioteka duomenų prieigai per *SQL*.

„Scala“ programavimo kalba pasirinkta dėl jos priklausymo „Java“ kalbos ekosistemai ir dėl taikomų funkcinio programavimo principų, kurie pravartūs apdorojant duomenys. „ELKI“ biblioteka turi daug įvairių mašininio mokymosi algoritmų, įskaitant ir skirtus klasterizavimui, bei papildomus įrankius duomenų apdorojimui, rezultatų pavaizdavimui ir įvertinimui. Algoritmai šioje bibliotekoje realizuoti pagal jų aprašymus moksliniuose darbuose bei turi nuorodas į atitinkamus mokslinius darbus. Duomenų bazei nebuvo keliami jokie išskirtiniai reikalavimai ir „PostgreSQL“ atitiko poreikius.

4.2. Straipsnių surinkimas

Straipsnių tekstas ir metaduomenys parsiunčiami tiesiai iš naujienų portalų, išanalizuojant *HTML* dokumento turinį. Duomenų išgavimą palengvina specialūs *HTML* žymių atributai „itemtype“ ir „itemprop“, kurie nurodo, koks turinys patalpintas atitinkamoje *HTML* žymėje. Pavyzdžiui, portalo „delfi.lt“ *HTML* kode randamas atributas „itemtype“ su reikšme „http://schema.org/Article“, o giliau randamos žymės su „itemprop“ atributais, kurių reikšmės:

- „datePublished“;
- „headline“;
- „description“;
- „articleBody“;
- ir pnš.

Pasinaudojant šia informacija galima nesudėtingai išgauti straipsnio duomenis iš *HTML* dokumento. Atitinkami atributai aptinkami ir puslapyje „15min.lt“. Puslapyje „alfa.lt“ informacija sužymima pasitelkiant „Open Graph“ protokolą.

4.3. Duomenų rinkiniai

Tekstai klasterizavimui surinkti iš trijų didžiųjų lietuviškų naujienų portalų: „delfi.lt“, „15min.lt“ ir „alfa.lt“. Bandymams atlikti buvo panaudoti straipsniai iš šių trijų naujienų portalų, išleisti pirmąją 2016 metų savaitę (nuo sausio 1 d. iki sausio 7 d. imtinai). Šį duomenų rinkinį sudaro 3572 straipsniai.

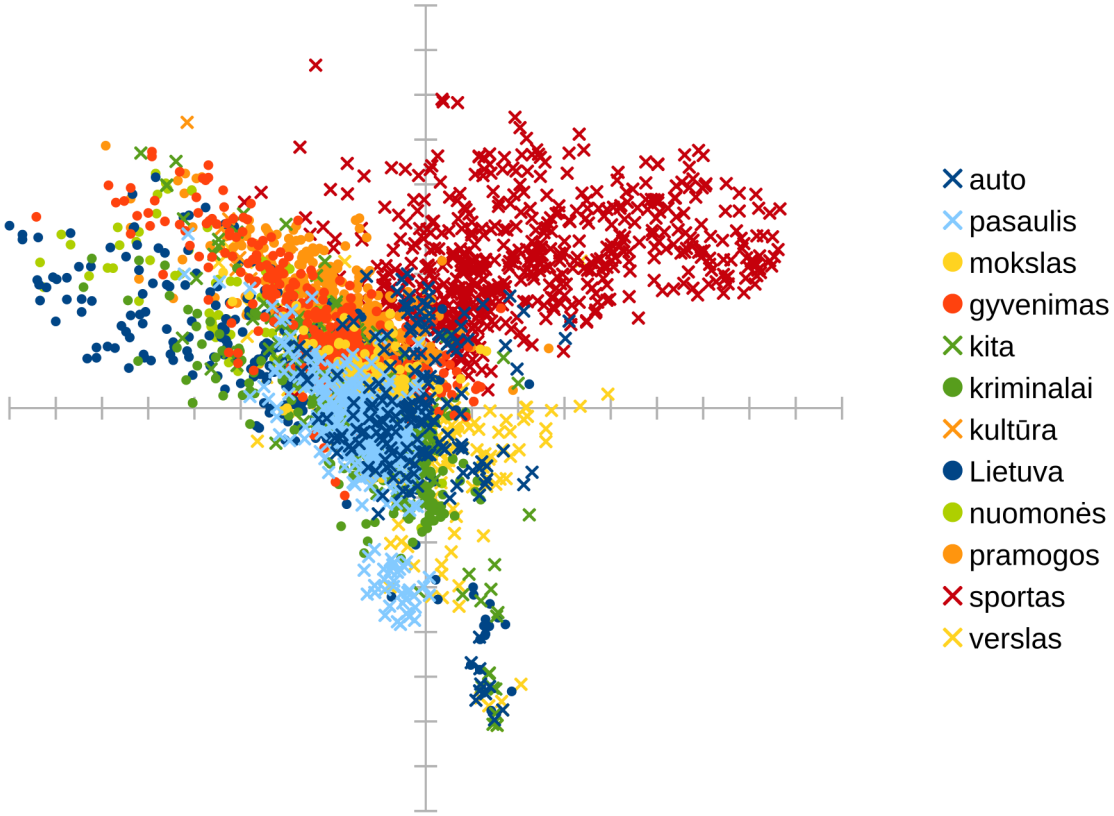
Naujienų portalai, kuriuose talpinami įvairios tematikos straipsniai, dažniausiai turi atskiras kategorijas ir rubrikas (pavyzdžiui, „sportas“ ar „verslas“), pagal kurias vartotojas gali atsirinkti jį dominantį turinį. Šios kategorijos buvo pasirinktos kaip priemonė įvardinti, apie ką rašoma straipsnyje. Reikia pastebėti, kad skirtinguose naujienų portaluose straipsniai suskirstomi į kategorijas skirtingai: atitinkamos kategorijos gali būti įvardijamos skirtingai (pavyzdžiui, „delfi.lt“ puslapyje kai kurios kategorijos įvardijamos anglų kalba) arba keletą rubrikų iš vieno portalo gali priklausyti vienai kategorijai (pavyzdžiui, viename puslapyje yra kategorija „sportas“, o kitame – atskiros rubrikos skirtingoms sporto šakoms).

Siekiant, kad skirtingai įvardintos kategorijos ir rubrikos kuo mažiau paveiktų klasterizavimo kokybinės įvertinimą, originalios straipsnių kategorijos buvo normalizuojamos į griežtai apibrėžtą kategorijų aibę. Įvertinus, kaip straipsniai suskirstomi įvairiuose portaluose, buvo išskirtos tokios 12 kategorijų:

- Lietuva (413 straipsnių; naujienos iš Lietuvos);
- Pasaulis (426 straipsniai; naujienos iš pasaulio);

- Kriminalai (321 straipsnis);
- Verslas (337 straipsniai);
- Automobiliai (170 straipsnių);
- Sportas (602 straipsniai);
- Mokslas (129 straipsniai);
- Nuomonės (99 straipsniai);
- Pramogos (526 straipsniai; straipsniai apie įžymybes, renginius, televiziją ir pnš.);
- Gyvenimas (306 straipsniai; įvairūs patarimai);
- Kultūra (56 straipsniai);
- Kita (187 straipsniai, kurie negali akivaizdžiai būti priskiriami kuriai nors iš anksčiau paminėtų kategorijų).

Reikia pastebėti, kad daugiausia straipsnių sudaro kategorija „sportas“, kuri apima 1/6 viso duomenų rinkinio. Tai atsispindi ir pavaizdavus kategorijas *2D* erdvėje (žiūrėti 4.1 paveikslėlį) pasitelkiant *principinę komponentų analizę* (*angl. Principal Component Analysis*, *PCA*). Dėl to, kad kategorija „sportas“ užima didelę dalį duomenų rinkinio, bei galimai dėl išskirtinio sporto straipsniuose vartojamo žodyno, ši kategorija akivaizdžiai išsiskiria ir yra išsidėsčiusi dešiniajame viršutiniame grafiko ketvirtyje (4.1 paveikslėlis, raudoni kryželiai), kai kitos kategorijos kur kas labiau susimaišiusios tarpusavyje.



4.1 pav. Straipsnių kategorijų pavaizdavimas *2D* erdvėje pasitelkiant *PCA*

Kodo fragmentas, kuris buvo panaudotas straipsnių kategorijų suvienodinimui, pateiktas skyriuje „Priedas A. Kategorijų suvienodinimo kodo fragmentas“ (47 puslapis).

4.4. Požymių atranka ir išgavimas

Požymių atrankai pasitelktas žodžių maišo (*angl. bag of words*) modelis. Tekstų apdorojimui pasitelkiami toliau pateikti žingsniai:

1. Teksto skaidymas į leksemas (žodžius);
2. Didžiųjų raidžių pakeitimas mažosiomis;
3. Kamienizavimas (dalyje eksperimentų);

4. Leksemų filtravimas pagal *IDF* (dalyje eksperimentų);
5. Leksemų rinkinio pavaizdavimas vektoriumi pritaikant skaitinius modelius (pavyzdžiui, *TF* ar *TF–IDF*);
6. Požymių vektoriaus normalizavimas.

Šiame darbe *TF* ir *IDF* paskaičiuojami pasitelkiant toliau pateiktas formules:

$$\mathrm{TF}(t,d)=f_{t,d}; \tag{4.1}$$

$$\mathrm{IDF}(t,D)=\log_{10}\frac{N}{n_t}, \tag{4.2}$$

kur $f_{t,d}$ – termino *t* pasikartojimo skaičius dokumente *d*, *N* – dokumentų skaičius rinkinyje *D,* ir n_t – skaičius dokumentų iš rinkinio *D*, kuriuose aptinkamas terminas *t*. Termino *TF–IDF* apskaičiuojamas sudauginant abi šias funkcijas.

4.5. Klasterizavimo procesas

Dokumentų klasterizavimas atliktas pasitelkiant „ELKI“ duomenų išgavimo (*angl. data mining*) karkasą [19]. Tyrimai atlikti su toliau pateiktais klasterizavimo metodais:

- k-vidurkių algoritmas [20]. Tyrimo metu išbandytas klasterizavimas keičiant klasterių skaičių ir pradinių centrų parinkimo metodus (atsitiktinis, tolimiausio taško bei *k-vidurkių++* [21]);
- dalijantis k-vidurkių algoritmas [8]. Bandymai atlikti su atitinkamais parametrais kaip ir k-vidurkių algoritmo atveju;
- hierarchinis klasterizavimas [22, 23]. Tyrimo metu išbandyti skirtingi klasterių tarpusavio panašumo įvertinimo metodai (vienos jungties, grupės vidurkio ir pilnos jungties, žiūrėti 2.2.1 skyrių). Hierarchijos pavertimui į plokščius klasterius pasirinktas *HDBSCAN* algoritmas [13]. Būtina pastebėti, kad darbe naudotas apibendrintas hierarchinio klasterizavimo algoritmas, kurio vykdymo laikas visiems jungčių metodams yra $O(n^3)$.

Atsižvelgiant į populiarumą mokslinėje literatūroje (žiūrėti 2.2.1.1 skyrių „Panašumo įverčiai“), atstumui tarp dokumentų įvertinti buvo pasirinktas kampo kosinuso panašumo įvertis.

4.6. Klasterių kokybės įvertinimas

Klasterizavimo kokybei įvertinti naudojamos toliau pateikiamos metrikos:

- grynumas (*angl. purity*);
- tikslumas (*angl. precision*);
- atkūrimas (*angl. recall*);
- *F1* balas;
- entropija;
- standartinis nuokrypis;
- absoliutaus nuokrypio mediana (*angl. median absolute deviation*).

Pirmos penkios iš aukščiau paminėtų metrikų apskaičiuojamos pasitelkiant žymes (*angl. label*). Kaip žymė imama kategorija, kuriai priskirtas straipsnis. Kokybiniai įverčiai tampa geresni, kai straipsniai iš tų pačių kategorijų priskiriami į tuos pačius klasterius.

Paskutinės dvi iš aukščiau paminėtų metrikų – standartinis nuokrypis ir absoliutaus nuokrypio mediana – skirtos įvertinti klasterių dydžių pasiskirstymams. Standartinis nuokrypis pasako, kiek labai klasterių dydžiai nutolę nuo vidurkio. Reikia pastebėti, kad standartinis nuokrypis yra gana jautrus išskirtiniams duomenų taškams (angl. *outlier*), t. y. vienas klasteris su kur kas mažesniu ar didesniu elementų skaičiumi nei kiti klasteriai žymiai paveiktų standartinio nuokrypio įvertį. Siekiant tai įvertinti, buvo papildomai pasirinkta absoliutaus nuokrypio medianos metrika, kuri yra mažiau jautri smarkiai išsiskiriančioms vertėms.

4.7. Klasterį apibūdinančių žodžių atranka

Darbo metu, siekiant geriau suprasti kokie straipsniai patenka į klasterius, buvo pasiūlyta metrika, kuri įvertina, kaip gerai pasirinktas žodis apibūdina klasterį. Ši metrika pavadinta *termino aktualumu* (*angl. term relevance*, *TR*) ir apskaičiuojama

$$\mathrm{TR}(t,c,D)=\frac{\mathrm{IDF}(t,D)}{1+\mathrm{IDF}(t,c)}, \tag{4.3}$$

kur $\mathrm{IDF}(t,D)$ ir $\mathrm{IDF}(t,c)$ – funkcijos, kurios paskaičiuoja termino *t* atvirkštinį dažnumą dokumentuose (*IDF*) atitinkamai visame straipsnių rinkinyje *D* ir pasirinktame klasteryje *c*.

4.8. Kombinuotas dviejų lygių klasterizavimas

Kai kurie naujienų agregatoriai (pavyzdžiui, „Google News“) siūlo dviejų lygių grupavimą: straipsniai, kurie aprašo tą patį įvykį, sugrupuojami kartu, o vėliau visos šios grupės suskirstomos į kategorijas („Lietuva“, „pasaulis“ ir t. t.). Šio darbo metu, jau atlikus dalį tyrimų, buvo nuspręsta realizuoti panašų dviejų lygių klasterizavimą. Atsižvelgus į klasterizavimo metodų eksperimento rezultatus buvo pasiūlytas toks sprendimas:

1. Visų pirma straipsniai suklasterizuojami pasitelkiant hierarchinį klasterizavimo algoritmą. Plokščių klasterių išskyrimui pasitelkiamas *HDBSCAN* algoritmas. Šiam algoritmui nustatomas žemas minimalaus klasterio dydžio parametras (išbandytos reikšmės – 2 ir 3 elementai), tikintis, kad tokiu atveju bus grąžinta daug mažų klasterių.
2. Straipsniai, pirmo etapo metu pakliuvę į tą patį klasterį, apjungiami į vieną dokumentą ir paduodami k-vidurkių metodui. Šio etapo metu parenkamas nedidelis klasterių skaičius (nuo 9 iki 13 klasterių).

Šiuo atveju, kai vieno iš etapų metu siekiama gauti daug mažų klasterių, hierarchinis klasterizavimas su *HDBSCAN* įgauna pranašumą prieš k-vidurkių algoritmą:

- k-vidurkių algoritmo atveju nėra aišku, kokį klasterių skaičių *k* parinkti. *HDBSCAN* atveju tuo rūpintis nereikia, nes klasterių skaičius parenkamas automatiškai priklausomai nuo minimalaus klasterio dydžio parametro;
- k-vidurkių algoritmo rezultatas labai priklauso nuo to, kaip parinkti pradiniai centrai. Turint omeny, kad šiuo atveju ieškoma didelio klasterių skaičiaus, išauga tikimybė, dalis centrų bus parinkti šalia vienas kito ir tai paveiks galutinius rezultatus. Hierarchiniai algoritmai šios problemos neturi;
- k-vidurkių algoritmo vykdymo laikas tiesiogiai priklauso nuo parametro *k* reikšmės. Šiam parametrui artėjant prie elementų skaičiaus *n*, vykdymo laikas nuo artimo tiesiniam $O(kn)$ artėja link kvadratinio $O(n^2)$.

4.9. Aparatinė įranga

Darbe aprašyti eksperimentai buvo atlikti pasitelkiant kompiuterį, kurio parametrai:

- Intel i3-2310M procesorius su 2,1 GHz taktiniu dažniu;
- 8 GB pagrindinės atminties.

5. EKSPERIMENTAI IR REZULTATŲ ANALIZĖ

Šiame skyriuje pateikiami darbo metu gauti rezultatai.

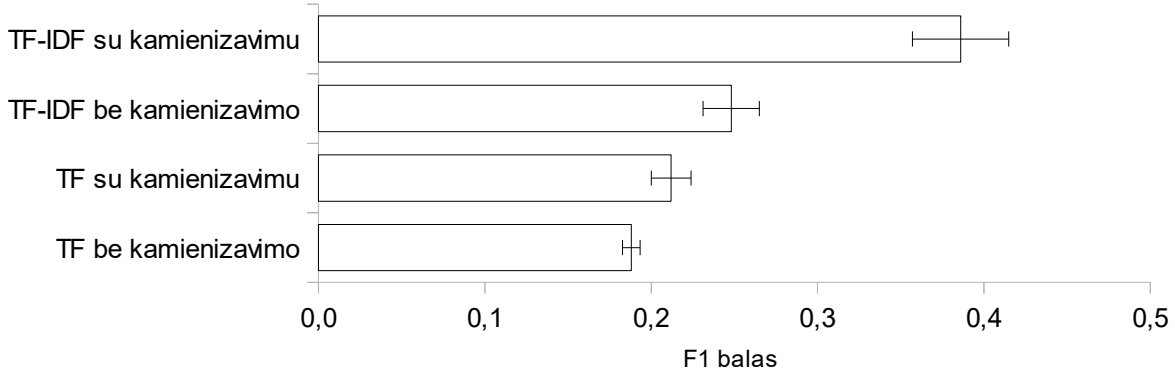
5.1. Požymių atrankos eksperimentų analizė

5.1.1. Termino dažnių metrikų ir kamienizavimo eksperimentas

Šio eksperimento metu pasirinkta ištirti, kokią įtaką klasterizavimo rezultatams daro žodžių kamienizavimas ir pasirinkimas tarp *TF* bei *TF–IDF*. Klasterizavimui pasirinktas k-vidurkių algoritmas remiantis gerais jo rezultatais klasterizavimo metodų tyrime (žiūrėti 5.2 skyrių „Klasterizavimo metodų tyrimo rezultatų analizė“). Tyrimo metu buvo išbandyti keletas skirtingų klasterių skaičių (k-vidurkių algoritmo *k* reikšmės nuo 9 iki 13), kiekvieną eksperimentą pakartojant 5 kartus su skirtingais atsitiktinai parinktais pradiniais centrais. Požymių atranka buvo vykdomi tokiais žingsniais:

1. Teksto skaidymas į žodžius;
2. Didžiųjų raidžių pakeitimas mažosiomis;
3. Kamienizavimas (pusėje testų šis žingsnis praleistas);
4. Pusėje testų naudojamas *TF*, kitoje pusėje – *TF-IDF*;
5. Gauto požymių vektoriaus normalizavimas.

5.1 paveikslėlyje pateikti tyrimo metu gautų *F1* įverčių vidurkiai bei standartiniai nuokrypiai. Geriausias rezultatas pasiektas pasitelkiant kamienizavimą ir *TF–IDF* ($\overline{\mathrm{F1}}=0{,}39$). Tai patvirtina intuiciją, kad kamienizavimas turėtų pašalinti triukšmą, sukeltą lietuvių kalbos morfologijos, ir kad *TF–IDF* turėtų sumažinti dažnai pasitaikančių ir sustiprinti retai pasitaikančių žodžių įtaką.



5.1 pav. Kamienizavimo ir dokumento pavaizdavimo modelių įtaka klasterizavimo kokybei

Taip pat galima pastebėti, kad su bet kuria kita požymių atrankos kombinacija *F1* įvertis stipriai krenta (nuo $\overline{\mathrm{F1}}=0{,}39$ iki $\overline{\mathrm{F1}}\leq 0{,}25$). Blogiausi rezultatai pasiekiami *TF–IDF* pakeitus į *TF* – kamienizavimas šiuo atveju daro minimalią įtaką ($\Delta\overline{\mathrm{F1}}=0{,}02$).

Lentelėje 5.1 pateiktos kitų įverčių reikšmės. *TF–IDF* su kamienizavimu pirmauja taip pat ir pagal grynumą bei entropiją, t. y. gauti klasteriai yra labiau homogeniški, juose labiau vyrauja straipsniai iš vienos kategorijos. Vienintelis atkūrimo įvertis yra mažesnis nei *TF–IDF* be kamienizavimo atveju, tačiau tai atsveria dvigubai didesnis nei kitose požymių atrankos kombinacijose tikslumo įvertis.

5.1 lentelė. Požymių atrankos eksperimento kokybiniai įverčiai

Požymių atrankos konfigūracija	Įverčių vidurkiai				
	***F1* balas**	**Tikslumas (*precision*)**	**Atkūrimas (*recall*)**	**Grynumas (*purity*)**	**Entropija**
TF-IDF su kamienizavimu	**0,39**	**0,33**	0,48	**0,52**	**0,58**
TF-IDF be kamienizavimo	0,25	0,16	**0,55**	0,39	0,72
TF su kamienizavimu	0,21	0,15	0,35	0,33	0,78
TF be kamienizavimo	0,19	0,13	0,36	0,26	0,85

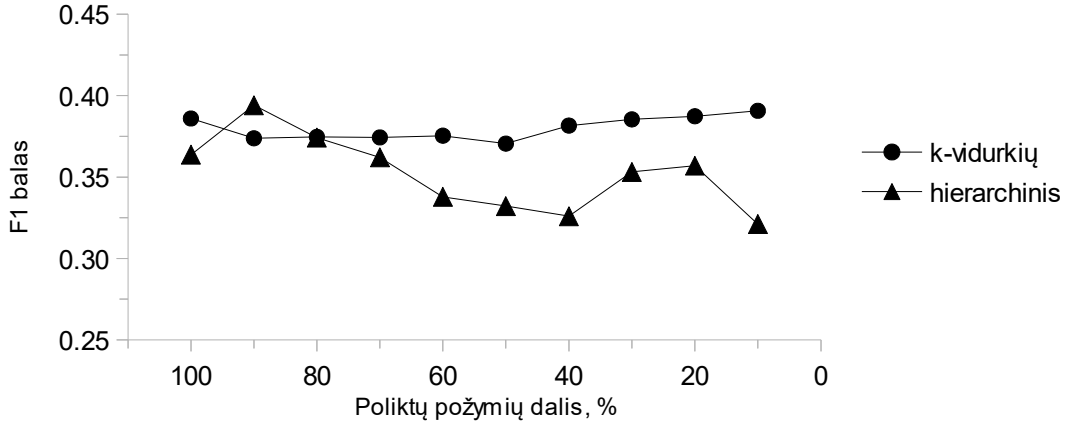
5.1.2. Požymių filtravimo pagal *IDF* eksperimentas

Požymių filtravimo eksperimento metu buvo tikrinama, kaip kinta klasterizavimo kokybė iš tekstų pašalinant dalį žodžių, turinčių didžiausią *IDF* įvertį (t. y. aptinkamų mažiausiame dokumentų skaičiuje). Tekstų apdorojimui pasitelkti tokie žingsniai:

1. Teksto skaidymas į leksemas (žodžius);
2. Didžiųjų raidžių pakeitimas mažosiomis;
3. Kamienizavimas;
4. Procentinės dalies leksemų pašalinimas, išrikiavus pagal *IDF* (nuo 0 % iki 90 %);
5. Požymių vektoriaus sudarymas pasitelkiant *TF–IDF*;
6. Gauto vektoriaus normalizavimas.

Straipsniams klasterizuoti, atsižvelgiant į gerus rezultatus (žiūrėti 5.2 skyrių „Klasterizavimo metodų tyrimo rezultatų analizė“), buvo pasirinkti k-vidurkių ir grupės vidurkio hierarchinis algoritmai. K-vidurkių algoritmo atveju buvo naudojami atsitiktinai parenkami pradiniai centrai. Eksperimentai leidžiami keletą kartų, parenkant vieną iš 5 skirtingų pradinių centrų išsidėstymų bei klasterių skaičių iš intervalo [9; 13] (iš viso 25 variantai kiekvienai filtravimo procentinei daliai). Hierarchinio klasterizavimo atveju taip pat algoritmas buvo leidžiamas keletą kartų, parenkant skirtingus *HDBSCAN* plokščio klasterizavimo išgavimo iš hierarchijos parametrus, bei atrenkant tuos testus, kurių metu buvo gauta nuo 9 iki 13 klasterių.

Paveikslėlyje 5.4 pateikti vidutiniai *F1* balo įverčiai, sugrupuoti pagal paliktų požymių procentinę dalį. K-vidurkių atveju įvertis kinta nežymiai, pradedant $\overline{F1_{1,0}} = 0{,}386$ ties 100 % paliktų požymių, mažiausią reikšmę $\overline{F1_{0,5}} = 0{,}371$ įgaunant ties 50 % paliktų požymių ir aukščiausią reikšmę $\overline{F1_{0,1}} = 0{,}391$ pasiekiant ties 10 % požymių.



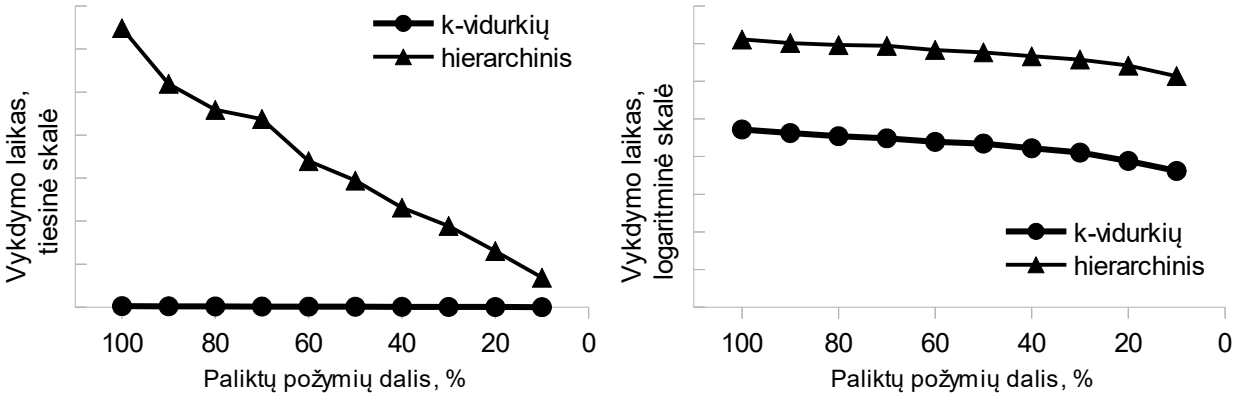
5.2 pav. *F1* balo priklausomybė nuo paliktų požymių procentinės dalies

Panašūs rezultatai pateikiami ir šaltinyje [3]. Šiame darbe buvo tiriamas k-vidurkių algoritmas ir skirtingos požymių atrankos metrikos, įskaitant ir *IDF*. Rezultatai parodė, kad kokybiniai įverčiai kinta nežymiai intervale tarp 100 % ir 10 % paliktų požymių. Klasterizavimo kokybė pradeda smarkiai kisti požymių skaičių mažinant nuo 10 % iki 2 %. Kito tyrimo metu, kuriame filtravimas pagal įvairias požymių atrankos metrikas buvo pritaikomas lietuviškiems ir rusiškiems tekstams, geriausi kokybiniai įverčiai buvo gauti ties 7 % paliktų požymių [10].

Rezultatai kitokie grupės vidurkio hierarchinio klasterizavimo atveju (žiūrėti 5.2 paveikslėlį). Vidutinis *F1* balas varijuoja kur kas labiau nei k-vidurkių algoritmo atveju. Nors keliuose taškuose *F1* balas pakyla (aukščiausia reikšmė $\overline{F1_{0,9}} = 0{,}394$ pasiekiama ties 90 %), tačiau pastebima mažėjimo tendencija (mažiausia reikšmė $\overline{F1_{0,1}} = 0{,}321$ ties 10 % paliktų požymių).

Palyginus abiejų metodų rezultatus ir kitų autorių tyrimus, akivaizdu, kad k-vidurkių metodo atveju klasterizavimo kokybė išlieka panaši (ar pagerėja) net pašalinant didžiąją dalį žodžių, kurie aptinkami mažame skaičiuje dokumentų. Panašu, kad k-vidurkių metodas yra savaime nejautrus retai pasitaikantiems žodžiams, galimai dėl to, nes apskaičiuojant klasterių centrus vidurkinami požymių vektoriai. To negalima pasakyti apie grupės vidurkio hierarchinį klasterizavimą, kuris jautriau reagavo į požymių erdvės pokyčius, o palaipsniui mažinant paliekamų požymių dalį rodė klasterizavimo kokybės prastėjimo tendenciją.

Sumažėjusi požymių erdvė atitinkamai sutrumpina algoritmų vykdymo laiką. Paveikslėlyje 5.3 pateiktas vykdymo laiko priklausomybės nuo požymių skaičiaus grafikas. Abiejų algoritmų vykdymo laikas mažėja panašia sparta, tačiau kadangi k-vidurkių algoritmas yra kur kas spartesnis už hierarchinio klasterizavimo algoritmą, tai vykdymo laiko atžvilgiu filtravimas naudingesnis hierarchiniam klasterizavimui.



5.3 pav. Vykdymo laiko priklausomybė nuo paliktų požymių skaičiaus (kairėje – tiesinė skalė, dešinėje – logaritminė skalė)

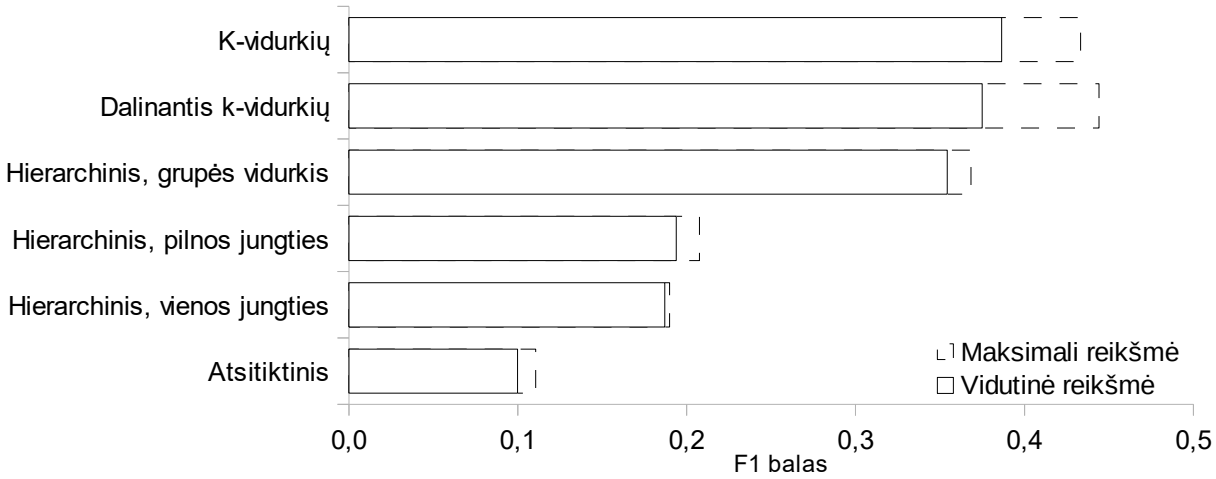
5.2. Klasterizavimo metodų tyrimo rezultatų analizė

Šio eksperimento metu buvo tiriami k-vidurkių, dalijantis k-vidurkių ir hierarchinis algoritmai. Abiejų k-vidurkių algoritmų atveju buvo išbandyti skirtingi klasterių skaičiai (nuo 9 iki 13 klasterių) bei skirtingos pradinių centrų parinkimo strategijos. Hierarchinio klasterizavimo atveju išbandytos 3 klasterių apjungimo strategijos: vienos jungties, grupės vidurkio ir pilnos jungties, o hierarchijos pavertimui į plokščius klasterius panaudotas *HDBSCAN* algoritmas. Taip pat, siekiant nustatyti apatines kokybinių įverčių ribas, buvo paleistas klasterizavimas, kai dokumentai atsitiktiniu būdu priskiriami į klasterius.

Požymių atrankai buvo pasitelkti tokie žingsniai:

1. Teksto skaidymas į žodžius;
2. Didžiųjų raidžių pakeitimas mažosiomis;
3. Kamienizavimas;
4. *TF–IDF* dokumento pavaizdavimo metrikos pritaikymas;
5. Požymių vektoriaus normalizavimas.

Paveikslėlyje 5.4 pateikti apibendrinti eksperimentų *F1* balų vidutinės ir maksimalios reikšmės. Į rezultatus įtraukti tik tie eksperimentai, kurių metu gautas klasterių skaičius buvo artimas kategorijų skaičiui – nuo 9 iki 13 klasterių. Atsitiktinio klasterizavimo atveju gautas $\overline{F1} = 0{,}1$. Prasčiausiai pasirodė vienos jungties ir pilnos jungties hierarchinio klasterizavimo metodai su atitinkamai $\overline{F1} = 0{,}19$. K-vidurkių algoritmas surinko $\overline{F1} = 0{,}39$, o dalijančioji versija surinko $\overline{F1} = 0{,}38$. Nuo jų nedaug atsiliko grupės vidurkio hierarchinis algoritmas su $\overline{F1} = 0{,}35$. Reikia pastebėti, kad patį aukščiausią įvertį $F1 = 0{,}45$ pavyko gauti su dalijančiu k-vidurkių algoritmu.



5.4 pav. Klasterizavimo metodų F*1* įverčių vidutinės ir maksimalios reikšmės

Lentelėje 5.2 pateikti apibendrinti klasterizavimo algoritmų tyrimo rezultatai. K-vidurkių algoritmas pirmauja pagal surinktą *F1* balą, tikslumą, grynumą ir entropiją. Grupės vidurkio hierarchinis algoritmas nors ir atsilieka pagal *F1* balą, tačiau turi aukštus tikslumo ir entropijos įverčius. Nepaisant to, naudojamam duomenų rinkiniui hierarchinių algoritmų vykdymo laikas daugiau nei 100 kartų didesnis už k-vidurkių algoritmų vykdymo laiką.

5.2 lentelė. Klasterizavimo metodų kokybiniai įverčiai

Klasterizavimo algoritmas	**Įverčių vidurkis**					
	***F1* balas**	**Tikslumas (*precision*)**	**Atkūrimas (*recall*)**	**Grynumas (*purity*)**	**Entropija**	**Vykdymo laikas, minutės**
K-vidurkių	**0,39**	**0,33**	0,48	**0,52**	**0,58**	1,48
Dalijantis k-vidurkių	0,38	0,29	0,53	0,48	0,62	1,04
Hierarchinis, grupės vidurkis	0,35	**0,33**	0,39	0,50	**0,58**	186,85
Hierarchinis, pilnos jungties	0,19	0,15	0,28	0,29	0,81	189,35
Hierarchinis, vienos jungties	0,19	0,11	**0,60**	0,28	0,81	189,28
Atsitiktinis	0,10	0,11	0,09	0,17	0,93	**0,00**

Lentelėje 5.3 pateiktos vidutinės kiekvieno algoritmo klasterių dydžių statistikos. Iš k-vidurkių, dalijančio k-vidurkių ir grupės vidurkio hierarchinio algoritmų, pastarasis turi mažiausias standartinio nuokrypio ir absoliutaus nuokrypio medianos reikšmes. Tai reiškia, kad šio algoritmo sudaromų klasterių dydžiai yra palyginus artimi vidurkiui, o ir rečiau pasitaiko klasterių su itin mažu ar itin dideliu elementų skaičiumi. K-vidurkių algoritmų atveju abi šios metrikos turi didesnes reikšmes. Įdomu tai, kad k-vidurkių algoritmas turi mažesnį standartinį nuokrypį, bet didesnę absoliutaus nuokrypio medianą, nei dalijantis k-vidurkių algoritmas. Iš to galima spręsti, kad dalijantis k-vidurkių algoritmas yra labiau linkęs sudaryti klasterius su išskirtinai dideliu ar išskirtinai mažu elementų skaičiumi.

5.3. lentelė. Klasterių dydžių statistika

Klasterizavimo algoritmas	Klasterio dydžio metrikų vidurkiai	
	Standartinis nuokrypis	Absoliutus nuokrypio mediana
K-vidurkių	286,47	194,86
Dalijantis k-vidurkių	356,74	150,90
Hierarchinis, grupės vidurkis	221,43	79,83
Hierarchinis, pilnos jungties	355,13	71,00
Hierarchinis, vienos jungties	784,73	21,75
Atsitiktinis	16,82	13,20

Tolimesniuose skyriuose detaliau aptariami kiekvienas iš klasterizavimo metodų.

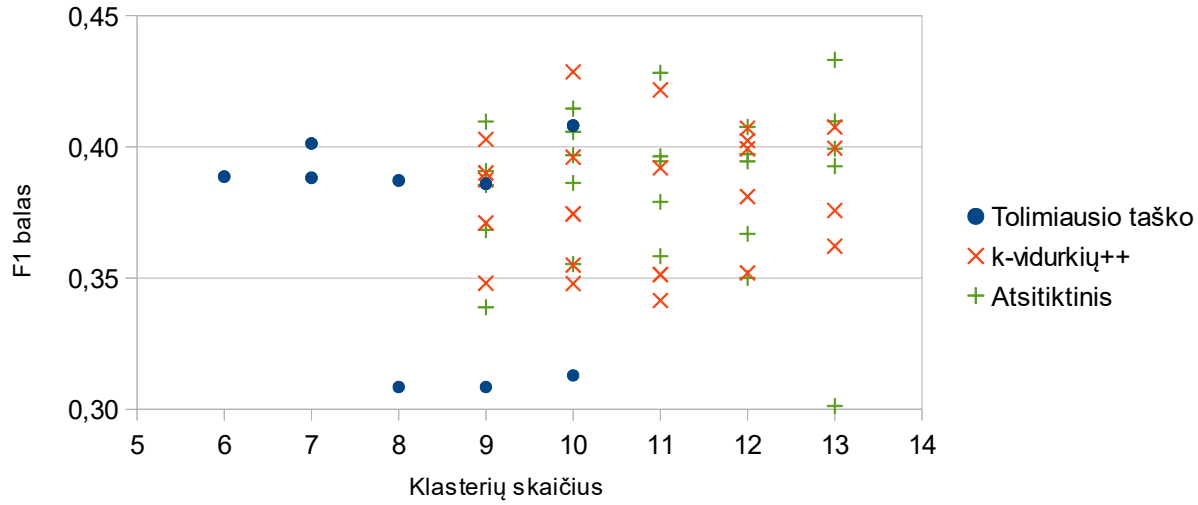
5.2.1. K-vidurkių algoritmo rezultatų analizė

K-vidurkių algoritmo tyrimo metu buvo išbandomi skirtingi klasterių skaičiai ir skirtingos pradinių centrų parinkimo metodikos. Rezultatai pateikti 5.4 lentelėje.

5.4 lentelė. K-vidurkių algoritmo kokybinių įverčių priklausomybė nuo pradinių centrų parinkimo metodo

Pradinių centrų parinkimo metodas	Įverčių vidurkis					
	F1 balas	Tikslumas (*precision*)	Atkūrimas (*recall*)	Grynumas (*purity*)	Entropija	Vykdymo laikas, sekundės
Atsitiktinis	**0,39**	**0,33**	0,48	**0,52**	**0,58**	**49,41**
Tolimiausio taško	0,38	0,28	**0,60**	0,45	0,63	107,65
k-vidurkių++	0,38	**0,33**	0,48	**0,52**	0,59	110,36

Nors ir *F1* balo vidurkiai gana panašūs, tačiau pagal kitus parametrus (išskyrus atkūrimą) tolimiausio taško metodas pasirodė prasčiau nei kiti išbandyti metodai. Tam galimai turi įtakos faktas, kad klasterizuojant su tolimiausio taško metodu būdavo grąžinamas mažesnis nei nustatyta klasterių skaičius. Paveikslėlyje 5.5 pateikti gauti klasterių kiekiai ir atitinkami *F1* balai. Nors eksperimentų metu buvo užduodamas nuo 9 iki 13 klasterių skaičius, kiekvienai iš šių reikšmių paleidžiant po 15 eksperimentų (3 pradinių centrų parinkimo metodai, kiekvienam iš jų pateikiant 5 skirtingas atsitiktinio parametro reikšmes), tolimiausio taško metodo atveju buvo grąžinama nuo 6 iki 10 klasterių.



5.5 pav. K-vidurkių algoritmo *F1* įverčių priklausomybė nuo klasterių skaičiaus ir pradinių centrų parinkimo metodo

Tiek atsitiktinis, tiek ir k-vidurkių++ pradinių centrų parinkimo metodai surinko itin artimus kokybinių įverčių rezultatus. Tačiau per pus mažesnis vykdymo laikas atsitiktinių klasterių centrų parinkimo atveju suteikia šiam metodui pranašumą.

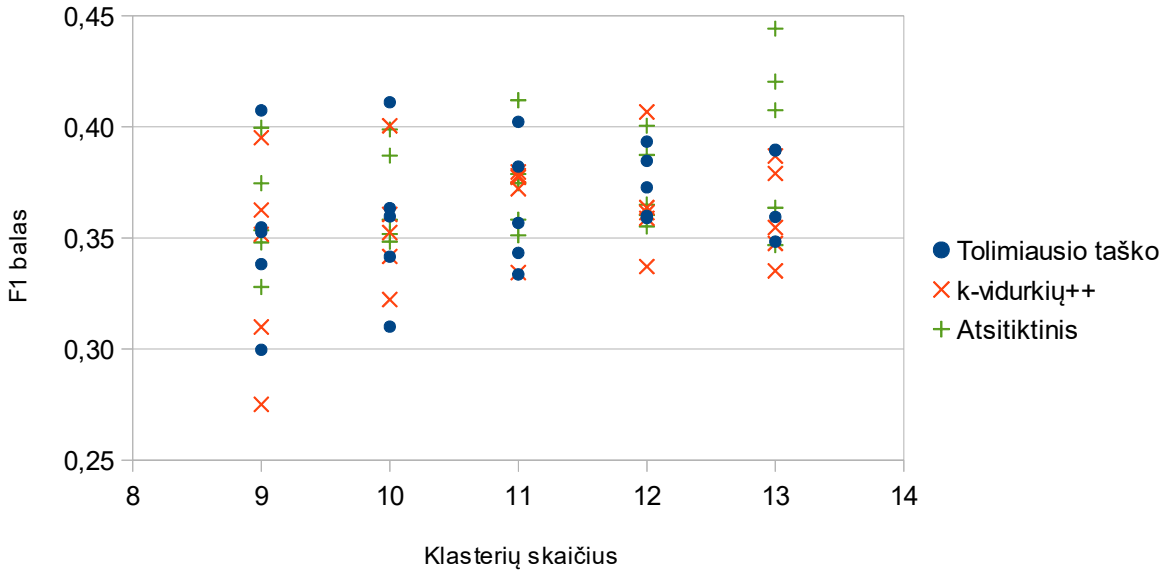
5.2.2. Dalijančio k-vidurkių algoritmo rezultatų analizė

Dalijančio k-vidurkių algoritmo atveju buvo išbandomi tie patys parametrai kaip ir su įprastu k-vidurkių algoritmu. Rezultatai pateikti 5.5 lentelėje. Šiuo atveju visi pradinių centrų parinkimo metodai parodė artimus rezultatus, tačiau vykdymo laiko atotrūkis tarp atsitiktinio pradinių centrų parinkimo ir kitų metodų siekia 3 ir daugiau kartų.

5.5 lentelė. Dalijančio k-vidurkių algoritmo kokybinių įverčių priklausomybė nuo pradinių centrų parinkimo metodo

Pradinių centrų parinkimo metodas	**Įverčių vidurkiai**					
	***F1* balas**	**Tikslumas (*precision*)**	**Atkūrimas (*recall*)**	**Grynumas (*purity*)**	**Entropija**	**Vykdymo laikas, sekundės**
Atsitiktinis	**0,38**	**0,29**	**0,53**	**0,48**	**0,62**	**23,92**
Tolimiausio taško	0,37	**0,29**	0,51	0,47	**0,62**	96,06
k-vidurkių++	0,36	0,28	0,50	0,47	0,63	66,59

Paveikslėlyje 5.6 pavaizduota *F1* balo priklausomybė nuo klasterių skaičiaus. Dalijančio k-vidurkių algoritmo atveju tolimiausio taško pradinių centrų parinkimo metodas nebekenčia nuo mažesnio nei užduota klasterių skaičiaus, kas ir atsispindi kokybiniuose įverčiuose.



5.6 pav. Dalijančio k-vidurkių algoritmo *F1* įverčių priklausomybė nuo klasterių skaičiaus ir pradinių centrų parinkimo metodo

5.2.3. Hierarchinio klasterizavimo rezultatų analizė

Tiriant hierarchinio klasterizavimo algoritmus buvo išbandyti trys klasterių apjungimo metodai: vienos jungties, pilnos jungties ir grupės vidurkio (žiūrėti 2.2.1 skyrių „Panašumu paremti klasterizavimo algoritmai“). Plokščių klasterių išgavimui iš hierarchijos buvo pasirinktas *HDBSCAN* algoritmas. Šio algoritmo veikimui būtinas parametras, nusakantis minimalų elementų skaičių klasteryje. Keičiant šį parametrą buvo reguliuojamas išgaunamų klasterių skaičių.

Lentelėje 5.6 pateikti atliktų bandymų rezultatai. Priklausomai nuo klasterių apjungimo metodo galima įžvelgti skirtingą minimalaus klasterio elementų skaičiaus parametro įtaką klasterių skaičiui: grupės vidurkio metodo atveju keičiant minimalų elementų skaičių tolygiai keičiasi ir gautų klasterių

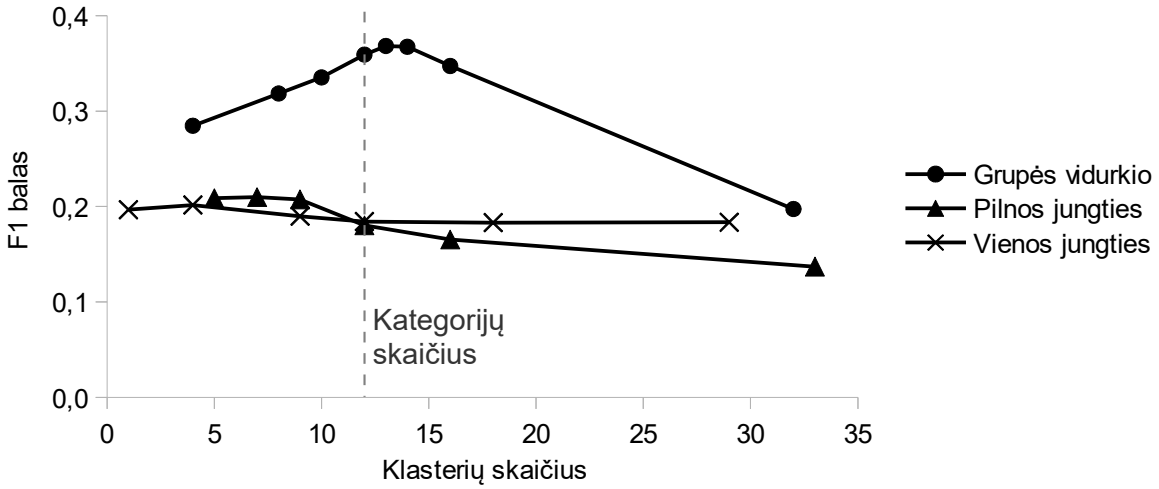
skaičius, kai kitų metodų atveju pastebimi didesni šuoliai tarp reikšmių pasikeitimo (smarkus parametro pakeitimas sukelia menką klasterių skaičiaus pasikeitimą ir atvirkščiai). Tai svarbu tuo atveju, kai norima išgauti konkretų klasterių skaičių: grupės vidurkio metodo atveju tai padaryti yra paprasčiau.

5.6 lentelė. Hierarchinio klasterizavimo bandymų rezultatai

Klasterių apjungimo metodas	***HDBSCAN* min. klasterio elementų skaičius**	**Klasterių skaičius**	***F1* balas**	**Tikslumas (*precision*)**	**Atkūrimas (*recall*)**	**Grynumas (*purity*)**	**Entropija**
Grupės vidurkio	300	4	0,29	0,18	0,64	0,33	0,73
	150	8	0,32	0,25	0,44	0,43	0,64
	140	10	0,34	0,28	0,42	0,46	0,60
	130	12	0,36	0,34	0,38	0,51	0,57
	120	13	**0,37**	**0,37**	0,37	0,52	0,56
	110	14	**0,37**	**0,37**	0,36	0,52	0,55
	100	16	0,35	0,34	0,35	0,54	0,54
	50	32	0,20	0,30	0,15	**0,59**	**0,50**
Vienos jungties	300; 150	1	0,20	0,11	**1,00**	0,17	0,93
	100	4	0,20	0,12	0,57	0,26	0,87
	50	9	0,19	0,11	0,64	0,28	0,82
	40	12	0,18	0,11	0,57	0,29	0,80
	30	18	0,18	0,11	0,51	0,32	0,77
Pilnos jungties	300; 150	5	0,21	0,15	0,37	0,25	0,86
	140; 130	7	0,21	0,15	0,35	0,27	0,84
	120	9	0,21	0,15	0,34	0,27	0,83
	110	12	0,18	0,16	0,22	0,31	0,78
	100	16	0,17	0,17	0,16	0,34	0,74
	50	33	0,14	0,24	0,10	0,43	0,63

Žvelgiant į klasterizavimo *F1* įverčius, akivaizdų pranašumą turi grupės vidurkio metodas, aukščiausią įvertį pasiekdamas ties klasterių skaičiumi, kuris artimas kategorijų skaičiui (12 kategorijų, aukščiausios įverčio reikšmės ties 12-14 klasterių). Šis metodas taip pat pirmauja ir pagal kitus įverčius, kai atsižvelgiama į šių įverčių polinkį augti didinant ar mažinant klasterių skaičių. Lentelėje 5.6 nepateikti vykdymo laiko duomenys, nes visų testų vykdymo laikas buvo panašus ir viršijo 3 valandas. Tiesa, šių testų metu buvo naudota bendrinė hierarchinio klasterizavimo algoritmo realizacija – priklausomai nuo klasterių apjungimo metodo galima būtų pritaikyti skirtingas optimizacijas, kurios sumažintų vykdymo laiką (daugiau informacijos 2.2.1 skyriuje).

Paveikslėlyje 5.7 pateikta *F1* balo priklausomybė nuo klasterių skaičiaus. Grafike akivaizdžiai matomi prasti vienos jungties ir pilnos jungties metodų rezultatai. Svarbu pastebėti tai, kad klasterių skaičiui artėjant prie kategorijų skaičiaus grupės vidurkio metodo atveju *F1* balas akivaizdžiai auga, kai tuo tarpu kitų metodų rezultatas mažėja.



5.7 pav. Hierarchinių algoritmų *F1* balo priklausomybė nuo klasterių skaičiaus

5.2.4. Kombinuoto dviejų lygių klasterizavimo tyrimo rezultatai

Šio eksperimento metu klasterizavimas buvo vykdomas dviem lygiais: visų pirma pasinaudojant hierarchinį klasterizavimą gaunama daug mažų klasterių, kurie vėliau suklasterizuojami pasitelkiant k-vidurkių algoritmą (detalesnis aprašymas 4.8 skyriuje).

Atliekant pirmo lygio klasterizavimą, požymių atrankai buvo panaudoti tokie patys žingsniai kaip ir klasterizavimo algoritmų tyrime (5.2 skyrius). Požymių atranka antro lygio klasterizavimui atlikta pasitelkiant tokius žingsnius:

1. Teksto skaidymas į žodžius;
2. Didžiųjų raidžių pakeitimas mažosiomis;
3. Kamienizavimas;
4. Mažų klasterių apjungimas į vieną dokumentą pritaikant *TF–IDF* arba *TF–TR* (žiūrėti 4.7 skyrių „Klasterį apibūdinančių žodžių atranka“);
5. Požymių vektoriaus normalizavimas.

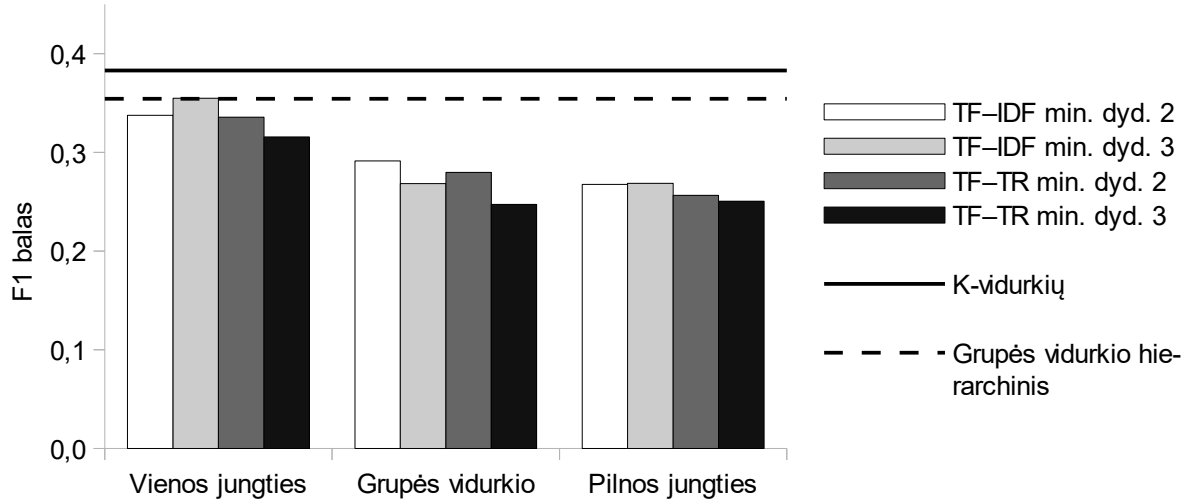
5.7 lentelėje pateikti eksperimento rezultatai. Dviejų lygių klasterizavimui nepavyko aplenkti vieno lygio k-vidurkių klasterizavimo. Geriausius rezultatus parodė dviejų lygių vienos jungties klasterizavimas, kurio vidutinis *F1* balas artimas vieno lygio grupės vidurkio hierarchiniam klasterizavimui. Priešingai nei vieno lygio hierarchinio klasterizavimo atvejų, prasčiausi rezultatai gauti kai pirmajame lygyje panaudotas grupės vidurkio ar pilnos jungties hierarchinis klasterizavimas.

Reikia pastebėti, kad pasirinkus vienos jungties kombinuotą klasterizavimą su minimaliu 3 elementų klasterio dydžiu, maksimalus gautas *F1* balas artimas maksimaliam k-vidurkių F1 balui (atitinkamai $F1_{TF-IDF}=0{,}42$, $F1_{TF-TR}=0{,}41$ ir $F1_{k-means}=0{,}43$). Palyginimui, maksimalūs įverčiai buvo mažesni, kai buvo parinkta 2 elementų minimalaus klasterio dydžio reikšmė. Galima to priežastis – parinkus mažesnį minimalų klasterio dydį padaugėja atvejų, kai nesusiję straipsniai sugrupuojami kartu, vietoj to, kad būtų priskiriami triukšmui.

5.7. lentelė. Dviejų lygių klasterizavimo rezultatai

Apjungimo metodas (*linkage*)	*HDBSCAN* min. klasterio dydis	Skaitinė pavaizd. metrika	Vid. *F1* balas	Maks. *F1* balas	Vid. tikslumas (*precision*)	Vid. atkūrimas (*recall*)	Vid. grynumas (*purity*)	Vid. entropija
K-vidurkių*			**0,39**	**0,43**	**0,33**	0,48	**0,52**	**0,58**
Vienos jungt.	3	TF-IDF	0,36	0,42	0,28	0,52	0,48	0,60
Grupės vidurkio hierarchinis*			0,35	0,37	**0,33**	0,39	0,50	**0,58**
Vienos jungt.	2	TF-IDF	0,34	0,38	0,25	**0,54**	0,47	0,62
Vienos jungt.	2	TF-TR	0,34	0,39	0,26	0,51	0,48	0,61
Vienos jungt.	3	TF-TR	0,32	0,41	0,23	0,52	0,45	0,63
Grupės vid.	2	TF-IDF	0,29	0,34	0,19	0,60	0,44	0,66
Grupės vid.	2	TF-TR	0,28	0,32	0,19	0,55	0,43	0,68
Pilnos jungt.	3	TF-IDF	0,27	0,29	0,17	0,62	0,39	0,70
Grupės vid.	3	TF-IDF	0,27	0,28	0,17	0,63	0,41	0,70
Pilnos jungt.	2	TF-IDF	0,27	0,36	0,18	0,53	0,41	0,67
Pilnos jungt.	2	TF-TR	0,26	0,29	0,17	0,51	0,39	0,70
Pilnos jungt.	3	TF-TR	0,25	0,27	0,15	0,68	0,37	0,72
Grupės vid.	3	TF-TR	0,25	0,27	0,16	0,60	0,37	0,72

5.8 paveikslėlyje pateiktos vidutinės *F1* balo reikšmės iš 5.7 lentelės. Žvelgiant į rezultatus akivaizdus vienos jungties hierarchinio klasterizavimo pranašumas. Taip pat svarbu pastebėti, kad rezultatai aukštesni, kai naudojamas paprastas *TF–IDF*, o ne darbe pasiūlytas *TF–TR*. To priežastis gali būti mažas klasterių dydis: esant dideliam straipsnių skaičiui klasteryje galima tiksliau nusakyti, kurie žodžiai būdingi tam klasteriui. Kai klasteris mažas, didėja tikimybė, kad atsitiktiniai žodžiai bus išrinkti kaip apibūdinantys klasterį.



5.8 pav. Dviejų lygių klasterizavimo vidutiniai *F1* įverčiai

5.3. Detalesnis žvilgsnis į eksperimentų rezultatus

Šio darbo metu buvo atlikta keletas eksperimentų. Per kai kuriuos iš jų skirtingai sukonfigūruoti klasterizavimo metodai buvo leidžiami virš šimto kartų, o bendras paleidimų skaičius viršija tūkstantį. Iš 50-ties geriausią *F1* balą turinčių testų visi išskyrus vieną naudojo klasikinį ar dalijantį

* Rezultatai iš vieno lygio klasterizavimo algoritmų eksperimento

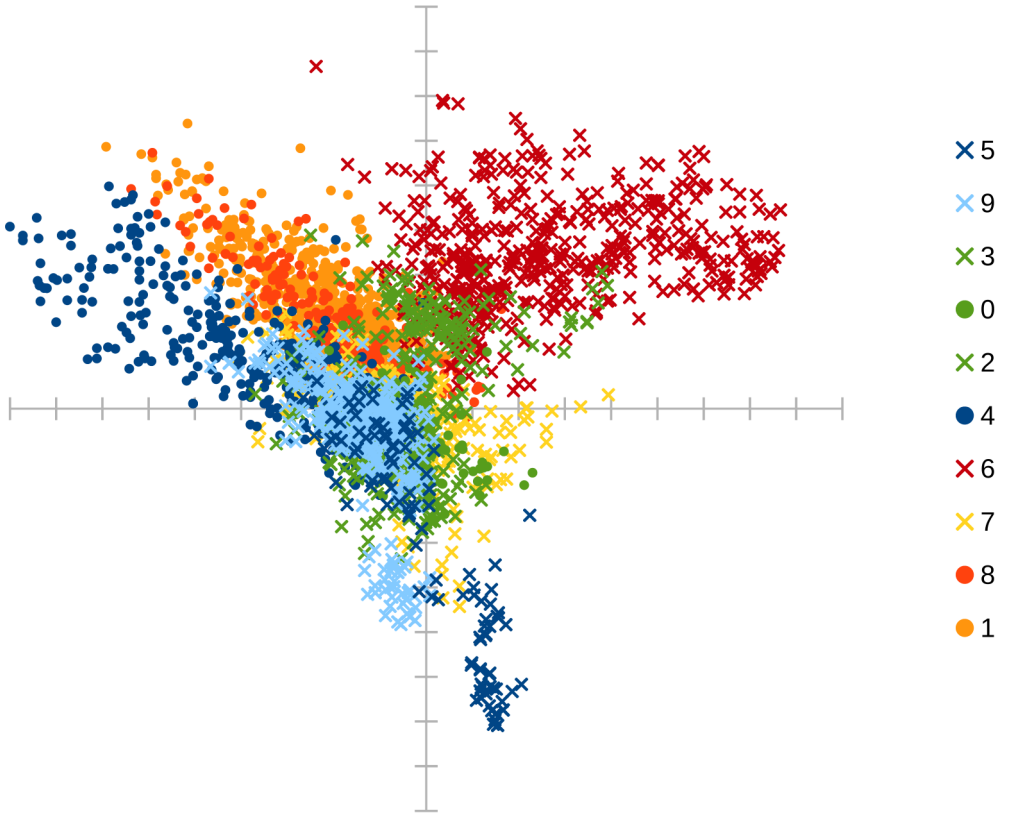
k-vidurkių algoritmus (likęs vienas – iš dviejų lygių klasterizavimo eksperimento). Didžioji dalis iš šių testų priklausė požymių filtravimo pagal *IDF* eksperimentui, daugiau nei pusė iš visų 50-ties – su 40 % ar mažiau paliktų požymių.

5.8 lentelėje pateikti pagal *TR* įvertį surikiuoti žodžiai iš testo su aukščiausiu *F1* įverčiu ($F1 = 0{,}468$). Šis testas buvo paimtas iš žodžių filtravimo eksperimento: k-vidurkių algoritmas, 20 % paliktų požymių. Sprendžiant pagal žodžius, ne visi klasteriai tiesiogiai atitinka kurią nors iš 12 kategorijų (pavyzdžiui, viename iš klasterių išskirti su Dakaro raliu susiję žodžiai). Tačiau iš šių žodžių nesunku nuspręsti, kokio turinio straipsniai vyrauja bet kuriame iš klasterių. Tiesa, peržvelgus į klasterius pakliuvusius straipsnius neišvengiamai galima rasti triukšmo – straipsnių, nesusijusių nei su aukščiausią *TR* įvertį turinčiais žodžiais, nei su kitais klasterio straipsniais (pavyzdžiui, vieno iš 5.8 lentelėje pateiktų klasterių dydis – 907 straipsniai. Akivaizdu, kad klasteris, kurį sudaro daugiau nei 1/4 viso duomenų rinkinio straipsnių, negali būti priskirtas vienai temai, ypač turint omeny, kad didžiausia originali kategorija „sportas“ sudarė 1/6 viso duomenų rinkinio).

5.8. lentelė. Aukščiausią *F1* balą turinčio testo klasterių žodžiai su aukščiausiomis *TR* reikšmėmis

Klasterio dydis	143	113	907	484	420
Žodžiai su aukščiausiu *TR* įverčiu	variklio dyzelinių gamintojų automobilių elektromobilių tesla motors fiat volkswagen lg	šalčio temperatūra laipsnių kritulių rajoniniai sniego provėžoti hidrometeorologijos plikledis prispausto	aktorė dainininkė muzikinę prodiuseris eurovizijos koncertinį meninės žanro scenoje režisierius	rungtynių ekipa žaidėjų turnyro taškų pergalę kamuolį treneris rezultatyvaus įvartį	karinių partijos sąjungininkų nimro šiitų obama narystės referendumas sirijos nato
Klasterio dydis	308	449	121	232	395
Žodžiai su aukščiausiu *TR* įverčiu	policijos ugniagesiai vpk neblaivus ambulatoriškai prom patrulių komisariato girtumas vairuojamas	bendrovės barelį indeksas brent įmonės valiutos wti holding akcininkų aplinkosaugos	dakaro ruožas ralio ekipažas vanagas juknevičius lenktynininkai trasos lenktynių benediktas	jums organizmą vitaminų astrologė cukraus horoskopas riebalų jus mitybos ožiaragis	šulinį saviečių tragedijos smurto mažamečius smurtaudamas sugyventinę sumetė nužudė ekspertizė

5.9 paveikslėlyje pateiktas didžiausią *F1* balą turinčio testo klasterių pasiskirstymas *2D* erdvėje pasitelkiant *PCA* algoritmą. Palyginus su originalių kategorijų pasiskirstymų (4.1 paveikslėlis, 29 puslapis), galima pastebėti panašumų. Visų pirma, dešiniajame viršutiniame ketvirtyje dominuoja raudoni kryželiai, atitinkantys sporto kategoriją. Taip pat grafike galima įžvelgti kategoriją „pasaulis“ (žydri kryželiai į kairę ir apačią nuo centro) ir kategoriją „verslas“ (geltoni kryželiai į dešinę nuo centro).



5.9 pav. Didžiausią *F1* balą turinčio testo klasterių pasiskirstymas *2D* erdvėje pasitelkiant *PCA*

6. IŠVADOS

1. Atliktas *TF-IDF* ir kamienizavimo tyrimas patvirtino intuiciją: aukščiausia klasterių kokybė gauta pasitelkus *TF-IDF* ir kamienizavimą ($\overline{\mathrm{F1}}=0{,}39$). Visais kitais atvejais gauti daugiau nei trečdaliu prastesni rezultatai ($\overline{\mathrm{F1}}\leq 0{,}25$, arba daugiau nei 1/3 mažesnis), iš kurių prasčiausiuose buvo naudojamas *TF*. Remiantis rezultatais galima daryti išvadą, kad apdorojant lietuvių kalbos tekstus svarbu tiek pašalinti nereikšminius žodžius (ką atlieka *TF–IDF*), tiek pašalinti dėl skirtingų to paties žodžio formų atsirandantį morfologinį triukšmą (ką atlieka kamienizavimas). Reikia pastebėti ir tai, kad kamienizavimas buvo mažai veiksmingas jį pritaikant kartu su *TF* ($\Delta\overline{\mathrm{F1}}=0{,}02$), tačiau kur kas veiksmingesnis kombinuojant su *TF–IDF* ($\Delta\overline{\mathrm{F1}}=0{,}14$). Tai parodo tiek morfologinio triukšmo pašalinimo svarbą, tiek tai, kad *TF* reikšminiams žodžiams, kuriems itin svarbus kamienizavimas, skiria menką įtaką.

   Žodžių filtravimo eksperimentas parodė, kad kai klasterizavimui naudojamas k-vidurkių algoritmas, galima pašalinti iki 90 % požymių erdvės be žymaus klasterių kokybės sumažėjimo (mažesnis nei 4 % *F1* balo pokytis). Grupių vidurkio hierarchinio klasterizavimo atveju mažinant požymių erdvę *F1* balas svyravo kur kas labiau ir rodė tendenciją mažėti (sumažinus požymių erdvę iki 10 % buvo gautas 12 % *F1* balo sumažėjimas), tačiau atitinkamai buvo kelis kartus sumažintas algoritmo vykdymo laikas.

2. Ištyrus skirtingus klasterizavimo algoritmus nustatyta, kad geriausius rezultatus pateikė k-vidurkių ir dalijantis k-vidurkių algoritmai (atitinkamai $\overline{\mathrm{F1}}=0{,}39$ ir $\overline{\mathrm{F1}}=0{,}38$). Nedaug nuo jų atsiliko grupės vidurkio hierarchinis klasterizavimas (surinkęs $\overline{\mathrm{F1}}=0{,}35$ bei aukštus tikslumo ir entropijos įverčius). Tiek vienos jungties, tiek pilnos jungties hierarchinio klasterizavimo metodai parodė kur kas prastesnius įverčius ($\overline{\mathrm{F1}}\leq 0{,}2$). Tai daro pastaruosius metodus netinkamus straipsnių klasterizavimui, bent jau kai siekiama gauti nedidelį klasterių skaičių.

   Vertinant gautų klasterių dydžių statistikas pastebėta, kad k-vidurkių algoritmai, lyginant su grupės vidurkio hierarchiniu klasterizavimu, turėjo didesnį klasterių dydžių standartinį nuokrypį. Iš to galima spręsti, kad pastarieji algoritmai yra labiau linkę sudaryti klasterius, kurie turėtų išskirtinai didelį ar išskirtinai mažą elementų skaičių.

   Darbe pasiūlytas dviejų lygių kombinuotas klasterizavimas parodė santykinai gerus rezultatus. Pasitelkus vienos jungties hierarchinį klasterizavimą buvo gautas rezultatas, artimas vieno lygio grupės vidurkio hierarchiniam klasterizavimui ($\overline{F1}=0{,}36$). Tiesa, dviejų lygių klasterizavimo vidutiniai *F1* įverčiai neprilygo vieno lygio k-vidurkių algoritmo rezultatams.

3. Įvertinus klasterizavimo algoritmų spartą bei klasterizavimo rezultatus, galima teigti, kad kuriant naujienų agregatorių ir siekiant sugrupuoti naujienų straipsnius į nedidelį grupių skaičių, tam geriau tinkamas k-vidurkių algoritmas. Rezultatai parodė, kad iš bandytų metodų k-vidurkių algoritmo sudaryti klasteriai labiausiai atitiko tai, kaip naujienas skirsto į kategorijas straipsnių autoriai. Taip pat testų metu k-vidurkių algoritmas veikė daugiau nei šimtą kartų sparčiau už hierarchinį klasterizavimą, kas leistų sutaupyti skaičiavimams skirtus resursus. Galiausiai, šis algoritmas buvo atsparus požymių erdvės filtravimui, kas dar labiau leidžia sumažinti vykdymo laiką. Tiesa, k-vidurkių algoritmo priklausomybė nuo atsitiktinai parenkamų pradinių centrų reikalauja priemonių, kurios leistų išrinkti geriausią klasterių paskirstymą iš kelių algoritmo paleidimų.

   Darbe pasiūlyto dviejų lygių klasterizavimo metodo tyrimas parodė, kad norint rasti daug mažų klasterių (t. y. sugrupuoti straipsnius, aprašančius tą patį įvykį), galimas pasirinkimas yra vienos jungties hierarchinis algoritmas, papildytas *HDBSCAN* hierarchijos suplokštinimo algoritmu su parinktu nedideliu minimaliu klasterio elementų skaičiaus parametru. Literatūroje pateikiami algoritmai, kurie leidžia vienos jungties hierarchinio klasterizavimo vykdymo laiką sumažinti nuo $O(n^3)$ iki $O(n^2)$, kas leistų vykdymo laiką labiau priartinti prie k-vidurkių algoritmo vykdymo

laiko. *HDBSCAN* automatiškai parenka optimalų klasterių skaičių. Šio algoritmo minimalaus klasterio dydžio parametras leidžia reguliuoti ribą, kiek mažiausiai reikia rasti panašių straipsnių, kad jie būtų sugrupuojami į klasterį taip sudarant „įvykį“. *HDBSCAN* algoritmas taip pat automatiškai atskiria „triukšmą“ – elementus, kurių sudaromi klasteriai nesiekia nustatyto minimalaus klasterio dydžio. Galimos įvairios strategijos, kaip panaudoti šį „triukšmą“ naujienų agregatoriuje: pateikti atskiroje kategorijoje, pašalinti ar suklasterizuoti atskirai nuo kitų straipsnių, taip siekiant rasti potencialiai besiformuojančius „įvykius“.

LITERATŪRA

1. AGGARWAL, Charu C. ir ZHAI, ChengXiang. A Survey of Text Clustering Algorithms. *Mining Text Data*. 2012, [žiūrėta 2015-11-15]. p. 77–128. ISBN 978-1-4614-3222-7. Prieiga per internetą: http://www.charuaggarwal.net/text-cluster.pdf
2. SLONIM, Noam ir TISHBY, Naftali. Document clustering using word clusters via the information bottleneck method. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval SIGIR 00* [interaktyvus], 2000 [žiūrėta 2015-11-02], p. 208–215. ISBN 1581132263. Prieiga per internetą: http://web.cs.iastate.edu/~honavar/infobottleneck.pdf
3. LIU, Tao ir kt. An evaluation on feature selection for text clustering. *Icml* [interaktyvus], 2003 [žiūrėta 2015-11-15], p. 488–495. ISBN 1577351894. Prieiga per internetą: http://www.aaai.org/Papers/ICML/2003/ICML03-065.pdf
4. DHILLON, Inderjit S. Co-clustering documents and words using bipartite spectral graph partitioning. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* [interaktyvus], 2001 [žiūrėta 2016-11-15], p. 269–274. ISSN 1-58113-391-X. Prieiga per internetą: http://www.cs.utexas.edu/users/inderjit/public_papers/kdd_bipartite.pdf
5. MANNING, Christopher D.; RAGHAVAN, Prabhakar ir SCHÜTZE, Hinrich. *Introduction to Information Retrieval* [interaktyvus]. 2008, [žiūrėta 2017-01-14]. 496 p. ISBN 0521865719. Prieiga per internetą: http://nlp.stanford.edu/IR-book/
6. CUTTING, Douglass R. ir kt. Scatter/Gather: a cluster-based approach to browsing large document collections. *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval* [interaktyvus], 1992 [žiūrėta 2016-02-16], p. 318–329. ISSN 01635840. Prieiga per internetą: http://portal.acm.org/citation.cfm?id=133214
7. HUANG, Anna. Similarity measures for text document clustering. *Proceedings of the sixth New Zealand computer science research student conference* [interaktyvus], 2008 [žiūrėta 2015-11-02], nr. April, p. 49–56. Prieiga per internetą: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.332.4480&rep=rep1&type=pdf
8. STEINBACH, Michael; KARYPIS, G. ir KUMAR, V. A Comparison of Document Clustering Techniques. *KDD workshop on text mining*, 2000, t. 400, p. 1–2. ISBN 9781424428748.
9. CIGANAITĖ, Greta; MACKUTĖ-VARONECKIENĖ, Aušra ir KRILAVIČIUS, Tomas. Text documents clustering. *Informacinės technologijos : 19-oji tarpuniversitetinė tarptautinė magistrantų ir doktorantų konferencija „Informacinė visuomenė ir universitetinės studijos“ (IVUS 2014) : konferencijos pranešimų medžiaga* [interaktyvus], 2014 [žiūrėta 2016-11-02] Prieiga per internetą: http://vddb.library.lt/fedora/get/LT-eLABa-0001:P.03~2014~D_20140424.ISSN_2029-249X.PG_90-93/DS.003.1.01.PAPER
10. MACKUTĖ-VARONECKIENĖ, Aušra ir KRILAVIČIUS, Tomas. Empirical Study on Unsupervised Feature Selection for Document Clustering. *Human Language Technologies – The Baltic Perspective*. 2014. p. 107–109. ISBN 9781614994428.
11. STREHL, Alexander; GHOSH, Joydeep ir MOONEY, Raymond. Impact of Similarity Measures on Web-page Clustering. *In Workshop on Artificial Intelligence for Web Search (AAAI 2000)*, 2000, p. 58–64. ISBN 1-57735-116-9.
12. ESTER, Martin ir kt. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, 1996, p. 226–231. ISSN 09758887.
13. CAMPELLO, Ricardo J. G. B.; MOULAVI, Davoud ir SANDER, Joerg. Density-Based Clustering Based on Hierarchical Density Estimates. *Advances in Knowledge Discovery and Data Mining*, 2013, p. 160–172. ISSN 16113349, 03029743.
14. BEIL, Florian; ESTER, Martin ir XU, Xiaowei. Frequent term-based text clustering. *KDD '02 Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, p. 436. ISSN 158113567X.

15. LU, Yue; MEI, Qiaozhu ir ZHAI, ChengXiang. Investigating task performance of probabilistic topic models: An empirical study of PLSA and LDA. *Information Retrieval*, 2011, t. 14, nr. 2, p. 178–203. ISSN 13864564.
16. ŽALINAUSKAS, Marius. *Individualiai klasifikuotų dokumentų klasterizavimo metodas*. [interaktyvus] 158 p. Prieiga per internetą: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2006~D_20060522_143851-15319/DS.005.0.02.ETD
17. KEBELYTĖ, Ernesta. *Lietuviško automatinio naujienų agregatoriaus prototipas*. [interaktyvus] 62 p. Prieiga per internetą: http://talpykla.elaba.lt/elaba-fedora/objects/elaba:8619299/datastreams/MAIN/content
18. STEPANOVIČ, Pavel ir KURASOVA, Olga. Tekstinių dokumentų panašumų paieška naudojant saviorganizuojančius neuroninius tinklus ir k vidurkių metodą. *Informacijos mokslai* [interaktyvus], 2013 [žiūrėta 2015-01-09], t. 65, p. 24–33. Prieiga per internetą: http://www.zurnalai.vu.lt/files/journals/163/articles/2058/public/24-33.pdf
19. ACHTERT, Elke; KRIEGEL, Hans-Peter ir ZIMEK, Arthur. ELKI: A Software System for Evaluation of Subspace Clustering Algorithms. *Scientific and Statistical Database Management*, 2008, p. 580–585. ISSN 3540694765.
20. MACQUEEN, J. B. Some Methods for classification and Analysis of Multivariate Observations. *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, 1967, t. 1, p. 281–297.
21. ARTHUR, David ir VASSILVITSKII, Sergei. k-means++: The Advantages of Careful Seeding. *Proc. of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms* [interaktyvus], 2007 [žiūrėta 2017-01-14] Prieiga per internetą: http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf
22. SNEATH, P. H. A. The Application of Computers to Taxonomy. *Journal of General Microbiology*, 1957, t. 17, nr. 1, p. 201–226. ISSN 0022-1287.
23. KAUFMAN, Leonard ir ROUSSEEUW, Peter J. Agglomerative Nesting (Program AGNES). *Finding Groups in Data*, 1990, p. 199–252. ISBN 9780470316801.

PRIEDAI

Priedas A. Kategorijų suvienodinimo kodo fragmentas

Žemiau pateikiamas kodo fragmentas „Scala“ kalboje, skirtas straipsnių kategorijų normalizavimui.

```
    private val delfi = PartialFunction[(String, String), String] {
      case ("auto", _) => Auto
      case ("gyvenimas", _) => Gyvenimas
      case ("krepsinis", _) => Sportas
      case ("mokslas", _) => Mokslas
      case ("pilietis", _) => Nuomones
      case ("sportas", _) => Sportas
      case ("veidai", _) => Pramogos
      case ("verslas", _) => Verslas

      case ("news", "crime") => Kriminalai
      case ("news", "lithuania") => Lietuva
      case ("news", "world") => Pasaulis
      case ("news", "ringas") => Nuomones

      case ("grynas", "gyvenimas") => Gyvenimas
      case ("grynas", _) => Kita

      case ("5braskes", "grazios") => Gyvenimas
      case ("5braskes", "linksmos") => Pramogos
      case ("5braskes", "konkursai") => Kita
      case ("5braskes", _) => Gyvenimas

      case ("video", "auto") => Auto
      case ("video", "mokslas-ir-gamta") => Mokslas
      case ("video", "pramogos") => Pramogos
      case ("video", "sportas") => Sportas
      case ("video", "verslas") => Verslas
      case ("video", "aktualijos") => Kita
      case ("video", "transliacijos") => Kita
      case ("video", "sveikata-tv") => Kita
      case ("video", "stilius") => Kita
      case ("video", "laidos") => Kita
      case e => logger.warn("delfi " + e); Kita
    }
    private val `15min` = PartialFunction[(String, String), String] {
      case ("24sek", _) => Sportas
      case ("deuce", _) => Sportas
      case ("sportas", _) => Sportas
      case ("gazas", _) => Auto
      case ("ikrauk", _) => Kita
      case ("mokslasit", _) => Mokslas
      case ("verslas", _) => Verslas

      case ("naujiena", "lietuva") => Lietuva
      case ("naujiena", "pasaulis") => Pasaulis
      case ("naujiena", "nusikaltimaiirnelaimes") => Kriminalai
      case ("naujiena", "nuomones") => Nuomones
      case ("naujiena", "komentarai") => Nuomones
      case ("naujiena", "kultura") => Kultura
      case ("naujiena", "sveikata") => Gyvenimas

      case ("laima", "laimos-veidai") => Pramogos
      case ("laima", _) => Gyvenimas

      case ("pasaulis-kiseneje", _) => Kita
      case ("zmones", _) => Pramogos
      case ("ji24", _) => Gyvenimas
      case e => logger.warn("15min " + e); Kita
    }
    private val alfa = PartialFunction[(String, String), String] {
      case ("auto", _) => Auto
      case ("gyvenimas", _) => Gyvenimas
      case ("it", _) => Mokslas
      case ("kriminalai", _) => Kriminalai
      case ("kultūra", _) => Kultura
      case ("lietuva", _) => Lietuva
      case ("pasaulis", _) => Pasaulis
```

```
          case ("nuomonės ir komentarai", _) => Nuomones
          case ("pramogos", _) => Pramogos
          case ("sportas", _) => Sportas
          case ("verslas", _) => Verslas
          case ("laisvalaikis", _) => Pramogos
          case e => logger.warn("alfa " + e); Kita
        }
```