VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
STUDIJŲ PROGRAMA: INFORMATIKA

Tekstų rekomendavimo algoritmai
Text Recommendation Algorithms

Baigiamasis magistro darbas

Atliko: Neringa Lukoševičiūtė
VU el. p.: neringa.lukoseviciute@mif.stud.vu.lt

Vadovas: doc. Arūnas Janeliūnas
Recenzentas: prof. dr. Aistis Raudys

Vilnius
2022

Santrauka

Šiame darbe yra aprašomas sukurtas tekstų rekomendavimo algoritmas, naudojantis ItemKNNCFCBF bei MostPopular algoritmų kombinaciją. Darbo tikslas - sukurti naują tekstų rekomendavimo algoritmą, tinkamą taikyti lietuvių kalba parašytų tekstų rekomendacijų gavimui, kai naudotojai yra mažai pažįstami. Tikslo įgyvendinimui buvo atsižvelgiama į jau egzistuojančių tekstų rekomendavimo algoritmų veikimo principus, išskiriant jų pliusus bei minusus. Pagal atsižvelgiamus kriterijus algoritmui buvo suformuluotos sąlygos, kuriomis algoritmas turi veikti bei apibrėžti pagrindiniai uždaviniai šioms sąlygoms įgyvendinti. Kuriant naują tekstų rekomendavimo algoritmą turiniu paremti metodai buvo kombinuojami su bendruoju filtravimu paremtais metodais, taip kuriant hibridinius metodus. Tokiu būdu yra išsprendžiamos aktualios rekomendavimo algoritmų problemos. Atliktas sukurto algoritmo efektyvumo vertinimas parodė, kad algoritmas tirtiems duomenims veikia tiksliau už kitus nagrinėtus tekstų rekomendavimo algoritmus. Atliekant eksperimentinį naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimą rezultatai buvo įvertinti preciziškumo, jautrumo, harmoniniu preciziškumo ir jautrumo vidurkio, preciziškumų vidurkių (MAP), normalizuotu diskontuotu suminiu naudingumu (NDCG), ploto po ROC kreive (AUC), naujumo ir dokumentų aprėpties reikšmėmis.

Raktiniai žodžiai: Rekomendavimo algoritmai, turiniu paremti rekomendavimo metodai, bendruoju filtravimu paremti rekomendavimo metodai, hibridiniai rekomendavimo metodai, tekstų rekomendavimas, teksto lemavimas, temų modeliavimas

Summary

This paper presents a new text recommendation algorithm that is a combination of ItemKNNCFCBF and MostPopular algorithms. The main objective of this research was to create an algorithm that would be capable of recommending texts written in Lithuanian, assuming that users' details are obscure. Other existing algorithms were taken into consideration to accomplish the aim. The main criteria and conditions for the algorithm were formulated, and the key goals were raised to satisfy these conditions. Content-based methods were combined with collaborative filtering methods to create a new text recommendation algorithm. This way, a hybrid method was created, and main recommendation algorithm problems were solved. The evaluation part proved that the created algorithm generates better results compared to other algorithms that were analyzed in this research. The effectiveness of text recommendation algorithms was evaluated in terms of Precision, Recall, F score, Mean Average Precision, Normalized Discounted Cumulative Gain, Area Under the ROC Curve, Novelty, and Coverage.

Keywords: Recommendation algorithms, content-based filtering, collaborative filtering, hybrid method, text recommendation, lemmatization, topic modelling

Turinys

Įvadas

Tradicinės žiniasklaidos žengimas į skaitmeninę erą, žmonių įpročių keitimasis ir taikymasis prie naujausių technologijų lėmė tai, kad didžioji dalis turinio yra skaitmenizuojama ir publikuojama internete. Kasdien talpinamas naujas turinys sparčiai didina internete saugomų duomenų kiekį, to pasekoje yra apsunkinama naudotojams aktualių duomenų paieška. Ši problema dažnai sprendžiama naudojant paieškos sistemas, kurios geba generuoti rezultatus, atsižvelgiant į naudotojo pateiktą įvestį. Ir, nors naudotojų naudojamos paieškos sistemos dažniausiai pakankamai tiksliai rekomenduoja reikiamą turinį, jos negali apdoroti visų naudotojų užklausų bei teikti labiau suasmenintą informaciją, atsižvelgiant į skirtingas aplinkybes, informacijos naujumą ir aktualumą ar ankstesnius naudotojo veiksmus konkrečioje turinio teikimo platformoje. Būtent šias problemas gali spręsti taikomi rekomendavimo algoritmai, kurie atsižvelgia į naudotojų poreikius iš anksto nenurodžius konkrečios įvesties bei teikia rekomendacijas pagal jau turimą informaciją. Pagrindinė šių algoritmų paskirtis - skirtingų produktų ar paslaugų rekomendavimas skirtingiems žmonėms, atrenkant tik konkrečiam žmogui galimai patinkančius produktus ar paslaugas.

Įprastai rekomendavimo algoritmai remiasi turimų duomenų grupavimu tarpusavyje pagal tam tikrus panašumo požymius (pavyzdžiui, turinio priklausymą bendroms temoms). Atsižvelgiant į juos, turinys naudotojams yra įvertinamas aktualumo skalėje bei rekomenduojamas naudotojui pagal tai, kas individualiam žmogui turėtų būti įdomu.

Literatūroje aprašomi rekomendavimo metodai yra skirstomi pagal pagrindinį jų veikimo principą. Vienas iš tokių principų - rekomendacijų generavimas atsižvelgiant į rekomenduojamų objektų turinį (t. y. turiniu paremti metodai). Visų pirma, norint taikyti šiuos metodus, rekomenduotiniems duomenims turi būti išskiriami pagrindiniai objekto požymiai. Pavyzdžiui, tekstų atveju - raktažodžiai, vaizdinės medžiagos atveju - pagrindinę vaizdinio temą apibūdinantys žodžiai, žanras ir pan. Tokių požymių išrinkimui gali būti naudojami Doc2vec algoritmas, TF-IDF (angl. *term frequency inverse document frequency*), esminių frazių parinkimo metodai. Kadangi požymių išrinkimas labai priklauso nuo analizuojamo turinio, šio tipo algoritmų veikimas bei tikslumas ypatingai priklauso nuo pasirinkto pagrindinių objekto požymių išrinkimo metodo. Pavyzdžiui, norint lietuvių kalba parašytiems tekstams išrinkti tinkamiausius raktažodžius, būtina naudoti tokį algoritmą, kuris turėtų lietuvių kalbos palaikymo funkciją bei gebėtų kuo tiksliau tokius geriausiai objektą apibūdinančius požymius išrinkti. Šie požymiai vėliau turi būti naudojami kaip įvestis rekomendacijų generavimui. Tam turiniu paremti rekomendavimo metodai gali remtis įvairiais klasifikavimo algoritmais (artimiausių kaimynų klasifikavimo, taisyklėmis paremto klasifikavimo ar kt.). Kaip pastebima literatūroje, šio tipo metodai dažnai susiduria su panašių objektų rekomendavimo ir mažos įverčių apimties problema. Taip yra todėl, kad esant mažai įverčių apimčiai, t. y. kai sistemoje nėra išsaugota pakankamai naudotojų įvertinimų, šio tipo metodai siūlys panašius objektus tik į tuos, kuriais naudotojas jau domėjosi. Pavyzdžiui, naudotojui susidomėjus sporto turiniu, algoritmas ir toliau rekomenduos tik su sportu susijusį turinį.

Siekiant išspręsti problemą, kai nėra įmanoma naudotojui sugeneruoti patikimų rekomendacijų dėl turiniu panašių objektų neradimo, literatūroje rekomenduojama naudoti bendruoju filtravimu paremtus rekomendavimo metodus. Šie metodai geba generuoti rekomendacijas nustatant atstu-

mą tarp produktų ar naudotojų, skaičiuojant pirkimo įvertinimo tikimybes ir kuriant sudėtingus spėjimo modelius. Įprastai rekomendacijų generavimui jie naudoja k – artimiausių kaimynų metodą, neuroninius tinklus ar kitus metodus. Apskritai bendruoju filtravimu paremti metodai remiasi kitais sistemai pažįstamais naudotojais, todėl tuo atveju, kai objektų skaičius yra labai didelis, o kiekvienas iš naudotojų susidomi tik keliais iš objektų, šie metodai neveikia efektyviai ir negali teikti pakankamai užtikrintų rekomendacijų, nes algoritmai nesugeba rasti kitų panašių naudotojų dėl gana tuščios naudotojų-objektų matricos. Šio tipo metodai taip pat gali susidurti su naujų naudotojų ir naujų objektų problemomis tais atvejais, kai sistema nežino pakankamai apie naują naudotoją (kuris dar nesusidomėjo pakankamu kiekiu objektų) arba naują objektą (kuriuo nesusidomėjo pakankamas kiekis naudotojų).

Norint išspręsti šias ir kitas literatūroje aprašomas problemas, yra siūloma taikyti hibridinį rekomendavimo principą, t. y. tokius metodus, kurie priima sprendimus remiantis kelių skirtingų rekomendavimo algoritmų rezultatais. Šie metodai geba apjungti tiek turiniu, tiek bendruoju filtravimu paremtus metodus ir generuoti rekomendacijas remiantis visų apjungtų algoritmų rezultatais. Įprastai tokiu būdu gali būti išsprendžiamos problemos, kurios yra aktualios taikant turiniu paremtus arba bendruoju filtravimu paremtus metodus atskirai. Tačiau yra labai svarbu tinkamai pasirinkti apjungiamus algoritmus, kad rezultatai būtų reikšmingi, o apjungimo laikas bei algoritmo sudėtingumas dėl to nenukentėtų. Būtent apjungimo bei apjungtų algoritmų veikimo laikas yra ypač aktualus hibridiniams rekomendavimo metodams, jei rekomendacijas norima teikti atsižvelgiant į naujausius turimus duomenis, kurie kasdien yra didėjantys. Šią problemą tyrė ir Xiaofeng Li bei Dong Li savo darbe [LL19], išskiriant realiu metu rekomendacijų teikimo, kurios yra apsunkinamos dėl bandomo apdoroti didelių duomenų kiekio, problemą.

Be jau išvardintų problemų aprašomame darbe yra sprendžiama ir paties turinio nagrinėjimo problema - prieš pateikiant turinį hibridinių ar turinių paremtų rekomendavimo algoritmų įvesčiai turi būti išskirti esminiai turinio bruožai (raktažodžiai). Raktažodžių identifikavimui aprašomame darbe buvo pritaikyti ir įvertinti 3 būdai - spaCy, TF-IDF ir Doc2Vec.

Dar viena tekstų rekomendavimo algoritmų problema - laiko dimensijos vertinimas. Kaip pastebėjo B. Fortuna ir kt. [FFM10], tekstai gali būti klasifikuojami pagal laiko aktualumą į ilgalaikius (moksliniai straipsniai, mokslinė literatūra) ir trumpalaikius (tokius, kas aktualu konkrečiam žmogui ar grupei žmonių, bet dažniausiai tik tam tikru laiko momentu). Norint rekomenduoti antrajai grupei priklausančius tekstus, būtina įvertinti ar rekomenduotinas tekstas bus aktualus ir rekomenduojamu metu.

Dalis šių problemų buvo nagrinėtos ir įvairiose moksliniuose straipsniuose - P. Valdiviezo-Diaz ir kt. [VOC+19] pateikia patobulintą bendruoju filtravimo algoritmu besiremiantį rekomendavimo metodą, kuris geba teikti rekomendacijas atsižvelgiant į naudojamą naivaus Bajeso metodą; W. Chuhan ir kt. [WWA+19] pasiūlė metodą, kaip teikti rekomendacijas, pritaikant neuroninius tinklus; Xiaofeng Li ir Dong Li [LL19] pasiūlė, kaip kombinuoti bendrojo filtravimo algoritmus su socialinių tinklų duomenimis siekiant kuo aukštesnio rekomendacijų tikslumo, išnaudojant kuo mažiau resursų ir išsprendžiant tuščių įvertinimų ir naujų naudotojų problemas. Taigi, tekstų rekomendavimo algoritmai, gebantys tiksliai nuspėti naudotojo poreikius ir sprendžiantys minėtus

iššūkius yra aktuali ir vis dar neišspręsta problema.

Šio darbo tikslas - sukurti naują tekstų rekomendavimo algoritmą, tinkamą taikyti lietuvių kalba parašytų tekstų rekomendacijų gavimui, kai naudotojai yra mažai pažįstami. Šiam tikslui įgyvendinti keliami tokie uždaviniai:

1. Atlikti analitinę jau egzistuojančių tekstų rekomendavimo algoritmų veikimo principų apžvalgą.
2. Rasti ir turimiems duomenims pritaikyti algoritmą, gebantį identifikuoti nagrinėjamo teksto raktinius žodžius.
3. Atlikti eksperimentinį naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimą.
4. Sukurti naują tekstų rekomendavimo algoritmą, gebantį generuoti lietuvių kalba parašytų tekstų rekomendacijas mažai pažįstamiems naudotojams.

Pasiekti magistro baigiamojo darbo rezultatai - sukurtas naujas tekstų rekomendavimo algoritmas, naudojantis ItemKNNCFCBF bei MostPopular algoritmų kombinaciją bei tinkamas taikyti lietuvių kalba parašytų tekstų rekomendacijų gavimui ir gebantis spręsti rekomendavimo uždavinį tokiomis sąlygomis:

- Mažai pažįstami naudotojai.
- Rekomendavimo algoritmo veikimui nereikalinga naudotojo įvestis.
- Rekomendacijų generavimas neturi reikalauti didelių skaičiavimo resursų.
- Į rekomendacijas turi būti įtraukiami ir nauji tekstai, apie kurių aktualumą nėra įmanoma nuspręsti iš kitų naudotojų, priklausančių tai pačiai naudotojų grupei.
- Rekomendacijos turi būti teikiamos atsižvelgiant į turinio aktualumą laiko prasme.

Sukurtas naujas algoritmas buvo palygintas su egzistuojančiais tekstų rekomendavimo algoritmais, o gauti efektyvumo vertinimo rezultatai parodė, jog naujasis algoritmas tirtiems duomenims veikė efektyviausiai. Algoritmų veikimas buvo vertintas apskaičiuojant preciziškumo, jautrumo, harmoninio preciziškumo ir jautrumo vidurkio, preciziškumų vidurkio, normalizuoto diskontuoto suminio naudingumo, ploto po ROC kreive, naujumo ir aprėpties reikšmes.

Šio rašto darbo 1 skyriuje pateikiama literatūros apžvalga - aprašomi turiniu, bendruoju filtravimu, hibridiniai ir kiti egzistuojantys rekomendavimo metodai, paaiškinami pagrindiniai literatūroje naudojami rekomendacinių sistemų įvertinimo matai bei įvardijamos dažnos rekomendavimo algoritmų problemos. 2 skyrius yra skirtas naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimo aprašymui - tirtos duomenų aibės ir duomenų paruošimo, raktažodžių išrinkimo metodų bei naudojamų tekstų rekomendavimo algoritmų efektyvumo vertinimo aprašymui. 3 skyriuje paaiškinama naujo algoritmo realizacija, apimanti duomenų aibės ir duomenų paruošimo aprašymą, realizavimo eigą bei algoritmo efektyvumo vertinimą. Rašto darbo pabaigoje pateikiami tyrimo metu gauti rezultatai bei išvados.

1. Literatūros apžvalga

Lyginant su tradicinėmis paieškos technikomis, rekomendacinės sistemos geba teikti labiau suasmenintus rezultatus bei apdoroti didelius duomenų kiekius [SGM17]. Taigi, rekomendacinės sistemos yra algoritmų, sąveikaujančių su dideliu duomenų kiekiu, visuma ir turimų žinių naudojimas naujų rekomendacijų gavimui. Kiekviena rekomendacinė sistema turi tris etapus - įvesties, generavimo ir išvesties. Anot A. Rapečkos [Rap15], tinkamų ir reprezentatyvių duomenų (naudojamų įvesties etape) išskyrimas iš didelės apimties duomenų (arba pirmasis etapas) yra vienas iš esminių efektyvaus rekomendacinės sistemos veikimo aspektų. Duomenų įvestis dažniausiai priklauso vienai iš pateiktų kategorijų [Rap15]:

- Skaitiniai vertinimai, išreiškiantys naudotojo nuomonę apie produktą, ir palikti sąmoningai (įvertinus produktus) arba nesąmoningai (palikus žymes pirkinių ar tinklalapių naršymo istorijose).
- Demografiniai duomenys.
- Turinio duomenys.

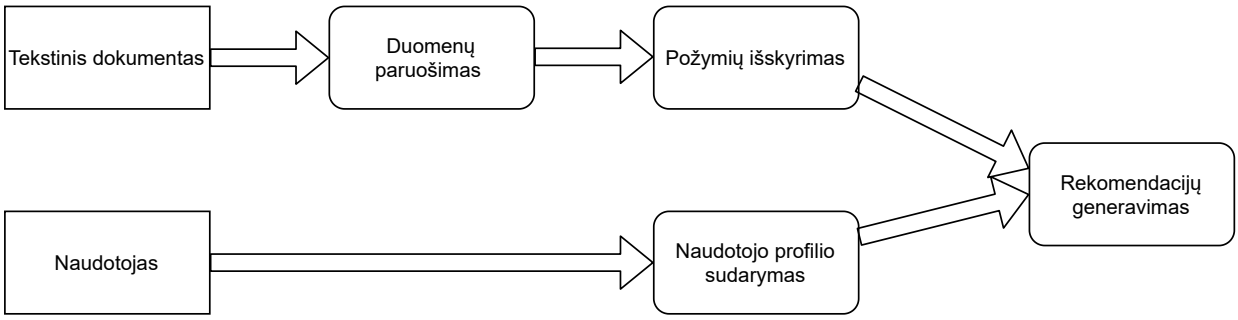
Kadangi aprašomame darbe siekiama teikti rekomendacijas mažai pažįstamiems naudotojams ir bandoma užtikrinti, kad rekomendavimo algoritmo veikimui nebūtų reikalinga naudotojo įvestis, galima remtis tik nesąmoningai paliktų skaitinių vertinimų arba teksto turinio duomenimis. Antrasis (generavimo) etapas yra glaudžiausiai susijęs su rekomendacinėje sistemoje naudojamu rekomendavimo algoritmu, kuris nulemia rekomendacijų generavimo principus ir veikia priklausomai nuo rekomendavimo metodo. Dažniausiai naudojami metodai [BWL+19]:

- Turiniu paremti.
- Bendruoju filtravimu paremti.
- Hibridiniai.

Toliau esančiuose poskyriuose analizuojamas kiekvienas iš šių metodų, išskiriant pagrindinius aspektus nagrinėjamai temai - tekstų rekomendavimui.

1.1. Turiniu paremti rekomendavimo metodai

Turiniu paremti rekomendavimo metodai atsižvelgia į naudotojo ankstesnius veiksmus sistemoje (t. y. atrenka panašius objektus į tuos, kuriais naudotojas jau domėjosi (skaitė)) ir sudaro naudotojo profilį [SML+14]. Be to, nagrinėjamam objektui yra išrenkami raktažodžiai ir skaičiuojamas panašumas tarp nagrinėjamo objekto ir naudotojo profilio. Galiausiai tekstai su didžiausiais panašumo koeficientais yra rekomenduojami naudotojams. 1 paveikslėlyje pateikiama turiniu paremtų rekomendavimo metodų struktūra. Visi jos etapai yra aprašomi toliau esančiuose skirsniuose.



1 pav. Turiniu paremtų rekomendavimo metodų struktūra

1.1.1. Duomenų paruošimas

Kaip pastebėjo L. Skorkovska [Sko12], duomenų paruošimas ir lemavimo taikymas turėtų būti taikomi prieš atliekant požymių išskyrimą, kadangi tokiu būdu yra išskiriami tikslesni dokumento požymiai, o analizuojamų požymių aibė gali sumažėti beveik per pusę. Dėl šios priežasties yra ženkliai paspartinamas sekančio etapo veikimas ir minimizuojama tikimybė, kad prie išskirtų dokumento požymių atsiras nereikšmingų (jungtukų, artikelių ar kitų naudingos informacijos nesuteikiančių) žodžių. Taigi, prieš analizuojant tekstinį dokumentą ir bandant jam išskirti esminius požymius, turimi duomenys yra paruošiami ir tinkamai apdorojami. Šio etapo pradžioje analizuojamas dokumentas yra išskaidomas į atskirus žodžius, pašalinant visus skyrybos ženklus bei aprašytus bereikšmius žodžius. Tokiam žodžių rinkiniui vėliau yra atliekamas teksto lemavimas (angl. *lemmatization*).

Lemavimo tikslas - sugrupuoti sudurtines žodžio formas, kad jas būtų galima analizuoti kaip vieną elementą, identifikuojamą pagal žodžio šaknį. Pavyzdžiui, veiksmažodis ”būti” yra žodžio bendratis, o veiksmažodžiai ”yra” ir ”buvo” yra atitinkamai esamojo ir būsimojo laiko veiksmažodžiai. Lemavimo dėka šie veiksmažodžiai yra apjungiami ir analizuojami kaip tas pats įvesties vienetas, o ne trys negimininingi veiksmažodžiai.

Lemavimo tikslumas tiesiogiai priklauso nuo to, ar bus nustatyta teisinga žodžio kalbos dalis. Kai ši yra nustatoma, taikomos įvairios taisyklės, būdingos konkrečiai kalbos daliai ir tokiu būdu nustatoma tikėtiniausia žodžio šaknis. Tam dažniausiai yra taikomos stochastinių (angl. *stochastic*) algoritmų taisyklės, besiremiančios tikimybėmis (analizuojamas žodis priskiriamas tikėtiniausiai kalbos daliai iš visų galimų). Tačiau galimos ir kitos variacijos, pavyzdžiui grubios jėgos, sufiksų nurėžimo, hibridinės taisyklės.

1.1.2. Požymių išskyrimas

Anot Xiaomei Bai [BWL+19], tekstai, net jeigu ir pateikiami tam tikroje struktūrinėje formoje, yra priskiriami nestruktūriniams duomenims. Taip yra todėl, kad teksto turinys struktūros neturi, be to, rašymo stilius ir pats turinys priklauso nuo teksto autoriaus. Autorius taip pat pabrėžia, kad struktūriški duomenys gali būti naudojami tiesiogiai, kas palengvina jų valdymą ir interpretaciją, tuo tarpu nestruktūriški duomenys prieš analizavimą turi būti transformuoti į struktūrinę formą. Taigi, siekiant realizuoti turiniu paremtą tekstų rekomendavimo algoritmą, tekstų turinį reikia struktūrizuoti - nagrinėjamam tekstui išskirti esminius požymius. Tam gali būti panaudotas

TF-IDF metodas, tam tikrų esminių frazių išrinkimas, Doc2vec algoritmas ar kiti metodai.

TF-IDF metodas. TF-IDF reikšmė yra statistinis rodmuo, įvertinantis žodžio svarbą tam tikrame dokumentų rinkinyje. Šis metodas remiasi 2 pagrindiniais aspektais [BWL+19]:

- Kuo dažniau žodis t pasitaiko tekste d, tuo t yra svarbesnis tekste d.
- Kuo žodžio t dažnis yra didesnis kituose dokumentuose, tuo žodžio t svarba yra mažesnė.

Taigi, laikantis šių aspektų, svarbiais žodžiais atrenkami tik tie, kurie dažniau pasitaiko mažesniame kiekyje tekstų. Taikant šį metodą žodžio t svoris w tekste d randamas pagal 1 formulę.

$$w_{td} = tf_{td} \cdot idf_{td} = tf_{td} \cdot \log \frac{n}{df_{td}} \tag{1}$$

čia tf_{td} - žodžio t dažnis tekste d, df_{td} - tekstų skaičius, kuriuose yra žodis t, n - visų tekstų skaičius.

Taikant šį metodą skaičiuojamas TF-IDF įvertis kiekvienam teksto žodžiui. Vėliau šie įverčiai sudaro ypatybių vektorius f kiekvienam nagrinėjamam tekstui. Šie vektoriai nurodo, kiek konkretus tekstas gali sudominti tam tikrą skaitytoją [PSA14]. Ypatybių vektorius apibrėžiamas taip, kaip nurodoma 2 apibrėžime.

$$f = (w_{t1}, w_{t2}, ..., w_{tm}) \tag{2}$$

čia m - skirtingų žodžių skaičius tekste, $t_k(k = 1, 2, ..., m)$ - visi žodžiai.

Šis metodas yra dažnai taikomas įvairių mokslinių straipsnių tyrinėtojų, kai norima realizuoti turiniu paremtą rekomendacinę sistemą [BWL+19]. Praktikoje dažnai yra taikomos šio metodo variacijos TF-IDF LSA arba TF-IDF LDA [Dzi19], [AOA+19].

LSA metodas. Latentinė semantinė analizė (LSA), dar žinoma kaip latentinis semantinis indeksavimas (LSI), yra metodas, skirtas paslėptiems tekstinių duomenų kontekstams rasti, remiantis TF-IDF ir ypatingųjų reikšmių dekompozicija (angl. *singular-value decomposition, SVD*). Šis metodas apskaičiuoja dokumentų panašumus sudarant žodžio-ištraukos matricą visoms ištraukoms (atvaizduojama stulpeliais) ir juose esantiems žodžiams (atvaizduojama eilutėmis). Matricos reikšmės yra užpildomos žodžių dažniais ištraukose arba TF-IDF statistikos reikšmėmis. Vėliau matricos skaičiui sumažinti ir informatyvesniems vektoriams gauti yra taikoma ypatingųjų reikšmių dekompozicija. Analizuojamų žodžių panašumas yra nustatomas apskaičiuojant normalizuotų vektorių sandaugą arba gautų vektorių dviejų eilučių kampo kosinuso reikšmę. Kuo rezultatas yra artimesnis vienetui, tuo žodžiai yra panašesni. Metodas remiasi prielaida, kad semantiškai panašūs žodžiai bus klasterizuojami kartu.

LDA metodas. Latentinio Dirichlė pasiskirstymo analizė (LDA) - projekcijos modelis, generuojantis dokumente vyraujančių žodžių aibę, besiremiant visų žodžių dažniais analizuojamame dokumente [AOA+19]. Šis metodas veikia pagal mokymo su mokytoju strategiją ir transformuoja žodžius iš visų žodžių multiaibės taip, kad klasių atskiriamumo kriterijaus reikšmė būtų optimali [Ras14]. Metodas remiasi prielaida, kad analizuojami dokumentai yra kelių temų mišinys, todėl kiekvienas dokumentas gali būti priskirtas kuriai nors iš temų. Kaip pastebėjo A. Rapečka [Rap15], pagrindinis LDA metodo privalumas - gebėjimas gerokai sumažinti duomenų požymių erdvę iki galimų klasių (temų) skaičiaus. Tokiu būdu yra pagreitinamas klasifikatoriaus mokymo laikas, kas yra itin svarbu, norint teikti tekstų rekomendacijas realiu metu, bandant apdoroti didelius duomenų kiekius. Reikia pastebėti, kad kai kurie tyrimai [AOA+19], [Rap15] pabrėžia, jog taikant šį metodą ir transformuojant didesnę požymių erdvę į mažesnę, atsiranda rizika prarasti svarbią informaciją nagrinėjamuose dokumentuose, todėl didžiausias tikslumas yra pasiekiamas taikant TF-IDF metodą, o ne kurią nors jo variaciją.

Esminių frazių išrinkimo metodas. Tekstiniuose dokumentuose (pavyzdžiui moksliniuose straipsniuose, naujienų portaluose) teksto požymių atrinkimui gali būti naudojamas ir esminių frazių išrinkimo metodas. Taikant šį metodą sudaromi esminių frazių sąrašai, apibūdinantys nagrinėjamo teksto turinį, išskiriant pagrindines tekste nagrinėjamas temas ir pateikiant trumpą teksto santrauką [BWL+19]. Pavyzdžiui, nagrinėjant mokslinius straipsnius, būtų sudaromi 3 sąrašai (vektoriai): $\vec{V}_{santrauka}$, $\vec{V}_{pavadinimas}$, $\vec{V}_{raktazodziai}$, esminėmis frazėmis apibūdinantys kiekvieną iš teksto dalių.

Doc2vec algoritmas. Doc2vec - dviejų sluoksnių neuroniniai tinklai, kurie atlieka dokumentų vektorizavimą. Šis modelis gaunamas apmokant dokumentus tam tikra dirbtinio neuroninio tinklo architektūra [LM14]. Neuroninis tinklas išmoksta požymių reikšmes iš viso požymių rinkinio taikant apmokymo be mokytojo strategiją bei pateikia nustatyto ilgio požymių vektorių kaip rezultatą. Vėliau šis rezultatas sudaro įvestį mašininio mokymo klasifikatoriui, kurio uždavinys yra suvesti dokumentą į reprezentatyvų vektorių. L. Stankevičiaus [Sta19] tyrimas parodė, kad optimizuotas Doc2vec algoritmas gali būti pranašesnis už įprastą TF-IDF naujienų straipsnių vektorizavimą, tirtiems lietuvių kalba parašytiems dokumentams.

1.1.3. Profilio sudarymas

Šio žingsnio tikslas - sudaryti naudotojo profilį, atsižvelgiant į ankstesnius konkretaus naudotojo veiksmus rekomendacinėje sistemoje. Kadangi aprašomame darbe siekiama užtikrinti, jog naudotojai gautų rekomendacijas be papildomos išankstinės įvesties, naudotojo profilį sudaryti galima remiantis tik nesąmoningai paliktų skaitinių vertinimų duomenimis. Tekstinių dokumentų vertinime dažnai naudojamas tokių duomenų atitikmuo - naudotojo užtruktas laikas skaitant konkretų dokumentą. Tokiu būdu filtravimo pagalba galima atmesti dokumentus, kuriuos naudotojas atidarė atsitiktinai arba dokumento turinys naudotojo nesudomino pakankamai, kad būtų skaitomas reikšmingą laiko tarpą.

1.1.4. Rekomendacijų generavimas

Šiame etape siekiama apjungti anksčiau minėtus etapų rezultatus - išskirtus tekstų požymius ir naudotojo profilį, taip nustatant naujus tinkamiausius tekstus, kurie turėtų dominti naudotoją. Naujų tekstų tinkamumo nustatymui gali būti taikomi tokie metodai, kaip artimiausių kaimynų klasifikavimo, taisyklėmis paremto (angl. *rule-based*) klasifikavimo ar kiti.

Artimiausių kaimynų klasifikavimas. Šis metodas rekomenduoja k panašiausių objektų naudotojui, atsižvelgiant į ankstesnius jo įverčius. Panašiausi objektai gali būti nustatomi, pavyzdžiui, taikant kosinuso panašumą (angl. *Cosine Similarity*). Kosinuso panašumas gali būti apskaičiuojamas pagal 3 formulę [Agg16].

$$\textit{panašumas}(X, Y) = \frac{\sum_{i=1}^{d} x_i y_i}{\sqrt{\sum_{i=1}^{d} x_i^2}\sqrt{\sum_{i=1}^{d} y_i^2}} \tag{3}$$

čia $X = (x_1...x_d)$, $Y = (y_1...y_d)$ - objektų poros, kuriose x_i ir y_i - i-ojo požymio svoriai, išskirti taikant, pavyzdžiui, TF-IDF metodą.

Tokiu būdu apskaičiavus panašumus, duomenys turi būti siejami su naudotojo profiliu, sukurtu ankstesniame etape. M. Kompan pritaikė šį algoritmą naujienų rekomendavimui [KB10] - apskaičiavus panašumo koeficientus, kiekvienam straipsniui atrenkami 10 panašiausių straipsnių ir surenkami duomenys apie tai, kokius straipsnius naudotojas jau perskaitė. Paskui sudaromas straipsnių sąrašas, sudarytas iš dviejų dalių - straipsnių, kurie buvo rekomenduoti bei perskaityti naudotojo ir straipsniai, kuriuos naudotojas perskaitė, bet kurie rekomenduoti anksčiau nebuvo. Galiausiai algoritmas atrinks tik tuos straipsnius, kurie turi panašius požymius į skaitytus straipsnius.

Taisyklėmis paremtas klasifikavimas. Šio tipo klasifikavimas, pritaikytas turiniu paremtuose metoduose, remiasi tam tikrų raktinių žodžių aibių buvimu teksto turinyje [Agg16]. Dėl to galima išskirti tokias taisykles, pagal kurias gali būti klasifikuojami objektai (tekstai):

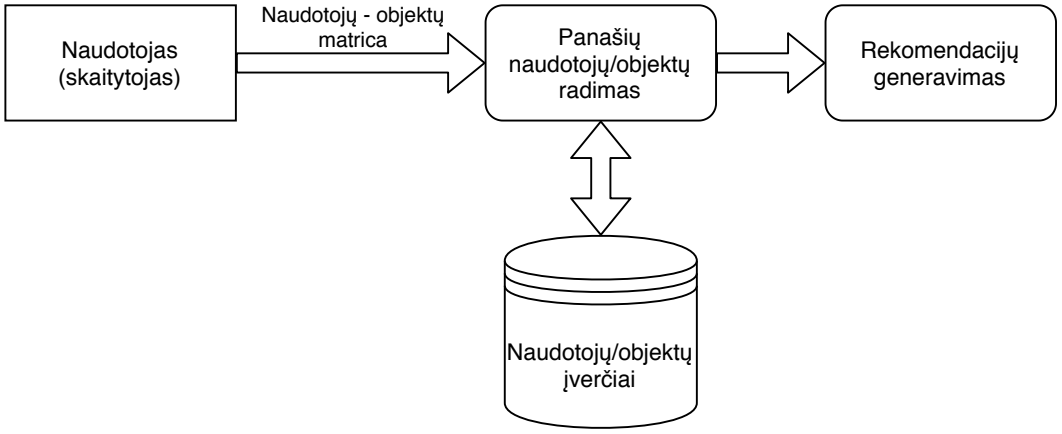
$$\begin{cases} \text{Objekte yra raktažodžių aibė A} \Rightarrow \text{Įvertis} = 1 \\ \text{Objekte yra raktažodžių aibė B} \Rightarrow \text{Įvertis} = 0 \end{cases}$$

Tokiu būdu, besiremiant ankstesniame žingsnyje sudarytu naudotojo profiliu, turėtų būti išnagrinėjami visi objektai, kuriais naudotojas domėjosi, atrenkant tuose tekstuose dominuojančius raktažodžius. Vėliau, siekiant generuoti naujas rekomendacijas, būtų atrenkami kiti panašūs tekstai, kuriuose tie patys raktažodžiai taip pat egzistuoja.

1.2. Bendruoju filtravimu paremti rekomendavimo metodai

Bendruoju filtravimu paremti rekomendavimo metodai generuoja rekomendacijas, atsižvelgiant į atstumus tarp rekomenduojamų objektų arba naudotojų, lyginant kiek jie yra panašūs vieni

su kitais. Šio tipo rekomendavimo algoritmai remiasi tuo, kad naudotojui siūloma rekomenduoti tuos objektus, kuriuos kiti panašūs naudotojai jau įvertino teigiamai [TVU+19]. Tokiu atveju, nustačius 2 panašius naudotojus, identifikuojami visi teigiamai įvertinti objektai. Jei kažkuris iš naudotojų objekto dar nėra įvertinęs, tai objektas yra rekomenduojamas, su prielaida, kad naudotojui rekomendacija bus tinkama, nes panašus naudotojas tą objektą įvertino teigiamai. Bendruoju filtravimu besiremiančių rekomendavimo algoritmų struktūra pateikta 2 paveikslėlyje.



2 pav. Bendruoju filtravimu besiremiančių rekomendavimo algoritmų struktūra

Objektų vertinimai gali būti renkami sąmoningai, paprašius naudotoją užpildyti apklausos formą ar tiesiog įvertinti objektą tam tikroje skalėje, arba nesąmoningai, tiesiog vertinant tai, kokiu turiniu naudotojas domėjosi. Kadangi aprašomame darbe siekiama sukurti algoritmą, gebantį veikti be jokios papildomos naudotojo įvesties, analizėje atsižvelgiama tik į nesąmoningai paliktus naudotojo įvertinimus.

Bendruoju filtravimu besiremiantys rekomendavimo algoritmai, visų pirma, turimoje duomenų aibėje bando įvertinti turimus duomenis apie naudotoją, taip surandant nagrinėjamo naudotojo (arba objekto) kaimynus - kitus panašiausius tos pačios sistemos naudotojus (arba objektus). Tam dažniausiai yra sukuriama naudotojų-objektų matrica, pavaizduojanti naudotojų įvertinimus sistemoje saugomiems objektams. Tokios matricos pavyzdys pateikiamas 1 lentelėje. Šioje matricoje naudojami 0 ir 1 skaitiniai vertinimai, nurodantys kiekvieno naudotojo įverčius objektams - 1, jei naudotojas objektu domėjosi ir 0, jei nesidomėjo. Ši matrica yra naudojama panašių naudotojų radimui, kurių įverčiais vėliau teikiamos rekomendacijos kitiems sistemos naudotojams.

1 lentelė. Naudotojų-objektų matricos pavyzdys

	Objektas1	Objektas2	...	ObjektasX
Naudotojas1	1	0	...	1
Naudotojas2	1	1	...	0
...	...	...	...	...
NaudotojasY	1	0	...	0

Šios matricos interpretacija ir naudojimas priklauso nuo pasirinkto bendruoju filtravimo besiremiančio rekomendavimo algoritmo [Agg16]:

- Naudotojais paremtas bendrasis filtravimas - naudotojui A panašių naudotojų įverčiai naudojami teikti rekomendacijas naudotojui A. Naudotojo A nuspėjami įverčiai apskaičiuojami pagal jam panašių naudotojų įverčių vidurkį kiekvienam objektui.

- Objektais paremtas bendrasis filtravimas - norint rekomenduoti objektą B, pirmasis žingsnis yra nustatyti aibę S objektų, kurie yra panašiausi į objektą B. Siekiant nuspėti objekto B įvertį bet kuriam naudotojui A, nustatomi naudotojo A įverčiai S aibės elementams. Šių įverčių vidurkis - nuspėjamas naudotojo A įvertis objektui B.

1.2.1. Naudotojais paremtas bendrasis filtravimas

Šio tipo filtravimas remiasi tuo, kad panašūs naudotojai įvertins tą patį objektą panašiais įverčiais. Jeigu naudotojai n ir u vertino panašiais įverčiais praeityje skaitytus tekstus, tuomet tikėtina, kad naudotojui n įvertinus naują tekstą i, naudotojo u įvertinimas tam pačiam tekstui bus panašus. Taigi, nuspėjamos įverčių reikšmės remiasi panašių naudotojų įverčiais. Panašūs naudotojai yra nustatomi atsižvelgiant į panašumus tarp naudotojų (t. y. remiantis eilutėmis naudotojų-objektų matricoje).

Taigi, taikant šio tipo algoritmus, visų pirma, siekiama rasti panašius naudotojus naudotojui u, kuriam planuojama generuoti rekomendacijas. Panašūs naudotojai randami skaičiuojant naudotojo panašumo įvertį su visais kitais naudotojais. Tam gali būti taikoma tokia panašumo formulė [BWL+19]:

$$\textit{panašumas}(u,n) = \frac{\sum_{i \subset CR_{u,n}} (r_{ui} - \overline{r}_u)(r_{ni} - \overline{r}_n)}{\sqrt{\sum_{i \subset CR_{u,n}} (r_{ui} - \overline{r}_u)^2} \sqrt{\sum_{i \subset CR_{u,n}} (r_{ni} - \overline{r}_u)^2}} \tag{4}$$

čia r - tekstų įverčiai, u - naudotojas, kuriam generuojama rekomendacija, n - kitas sistemos naudotojas, r_{ui} - naudotojo u įvertis objektui i, $\overline{r}$ - naudotojo u vidutinis įvertis visiems jo įvertintiems objektams. $CR_{u,n}$ - bendra naudotojų u ir n objektų aibė.

Žinant naudotojo u artimiausius (panašiausius) kaimynus, skaičiuojamas nuspėjamas įvertis, kurį suteiktų naudotojas u objektui i. Tam naudojama tokia formulė [BWL+19]:

$$\textit{nuspėjamasĮvertis}(u,i) = \overline{r}_u + \frac{\sum_{n \subset kaimynai(u)} \textit{panašumas(u, n)} \cdot (r_{ni} - \overline{r}_n)}{\sum_{n \subset kaimynai(u)} \textit{panašumas(u, n)}} \tag{5}$$

Naivus Bajeso metodas. Naivus Bajeso metodas taip pat gali būti naudojamas siekiant nustatyti nuspėjamą įvertį, kurį suteiks naudotojas u objektui i. Norint nustatyti nuspėjamą įvertį, iš pradžių reikia apibrėžti, kam lygi tikimybė, kad objektas i bus įvertintas y įverčiu bet kurio naudotojo [VOC+19]:

$$P(r_i = y) = \frac{\#\{u \in U \,|r_{u,i} = y\} + \alpha}{\#\{u \in U \,|r_{u,i} \neq \bullet\} + \#R \cdot \alpha} \tag{6}$$

čia U - naudotojų aibė, įvertinusių objektus aibėje I. $r_{u,i}$ - naudotojo u įvertis objektui i. $\bullet$ - įverčio nebuvimas, $\#R$ - galimų įverčių skaičius.

Taip pat turi būti apibrėžiama tikimybė, kad objektas j bus įvertintas k įverčiu, žinant, kad objektas i yra įvertintas įverčiu y [VOC+19]:

$$P(r_j = k|r_i = y) = \frac{\#\{u \in U|r_{u,j} = k \wedge r_{u,i} = y\} + \alpha}{\#\{u \in U|r_{u,j} \neq \bullet \wedge r_{u,i} = y\} + \#R \cdot \alpha} \tag{7}$$

Dėl to galima apibrėžti nuspėjamą įvertį $P(r_{u,i=y})$, kad naudotojas u įvertins objektą i įverčiu y, atsižvelgiant į anksčiau naudotojo u įvertintus kitus objektus [VOC+19]:

$$P(r_{u,i} = y) \propto P(r_i = y) \prod_{j \in I_u} P(r_j = r_{u,j}|r_i = y) \tag{8}$$

čia $I_u = \{i \in I|r_{u,i} \neq \bullet\}$ - naudotojo u įvertintų objektų aibė.

1.2.2. Objektais paremtas bendrasis filtravimas

Šio tipo filtravimas remiasi tuo, kad panašūs objektai yra vertinami panašiais įverčiais to paties naudotojo. Taigi tikėtina, kad naudotojas u naują tekstą t, kuris yra panašus į naudotojo u anksčiau įvertinus tekstus i, įvertins panašiais įverčiais, kaip ir įvertino objektus i. Taigi, nuspėjamos įverčių reikšmės remiasi to naudotojo, kuriam rekomenduojamas objektas, įverčiais, kurie buvo suteikti panašiems objektams. Panašūs objektai yra nustatomi atsižvelgiant į panašumus tarp objektų (t. y. remiantis stulpeliais naudotojų-objektų matricoje).

Panašumas tarp objektų gali būti nustatomas pritaikant tą pačią formulę, kaip ir nustatant panašumus tarp naudotojų (4 formulė). Pritaikant šią formulę randami k panašiausių elementų $i_1, i_2,...,i_k$ tiriamai objektų aibei. Vėliau bandoma rasti panašiausius objektus, atsižvelgiant į kitus naudotojo skaitytus tekstus, kuriais naudotojas turėtų susidomėti. Tam yra skaičiuojamas nuspėjamas įvertis $\hat{r}_{ut}$, priklausantis nuo naudotojo u ir analizuojamo objekto t [Agg16]:

$$\hat{r}_{ut} = \frac{\sum_{i \in Q_t(u)} \textit{panašumas(i, t)} \cdot r_{ui}}{\sum_{i \in Q_t(u)} |\textit{panašumas(i, t)}|} \tag{9}$$

čia $Q_t(u)$ - naudotojo u įvertintų k panašiausių elementų objektui t.

Naivus Bajeso metodas. Naivus Bajeso metodas taip pat gali būti naudojamas siekiant nustatyti nuspėjamą įvertį, kurį suteiks konkretus naudotojas tam tikram objektui, atsižvelgiant į kitus naudotojo įverčius panašiems objektams. Norint nustatyti nuspėjamą įvertį, iš pradžių reikia apibrėžti, kam lygi tikimybė, kad naudotojas u įvertins bet kurį objektą įverčiu y [VOC+19]:

$$P(r_u = y) = \frac{\#\{i \in I\,|r_{u,i} = y\} + \alpha}{\#\{i \in I\,|r_{u,i} \neq \bullet\} + \#R \cdot \alpha} \tag{10}$$

čia U - naudotojų aibė, įvertinusių objektus aibėje I. $r_{u,i}$ - naudotojo u įvertis objektui i. $\bullet$ - įverčio nebuvimas, $\#R$ - galimų įverčių skaičius.

Taip pat turi būti apibrėžiama tikimybė, kad naudotojas v įvertins objektą įverčiu k, žinant, kad naudotojas u įvertino objektą įverčiu y [VOC+19]:

$$P(r_v = k|r_u = y) = \frac{\#\{i \in I|r_{v,i} = k \wedge r_{u,i} = y\} + \alpha}{\#\{i \in I|r_{v,i} \neq \bullet \wedge r_{u,i} = y\} + \#R \cdot \alpha} \tag{11}$$

Dėl to galima apibrėžti nuspėjamą įvertį $P(r_{u,i=y})$, kad objektas i bus įvertintas įverčiu y naudotojo u, atsižvelgiant į ankstesnius objekto i įverčius [VOC+19]:

$$P(r_{u,i} = y) \propto P(r_u = y) \prod_{v \in U_i} P(r_v = r_{v,i}|r_u = y) \tag{12}$$

čia $U_i = \{u \in U|r_{u,i} \neq \bullet\}$ - naudotojų, įvertinusių objektą i, aibė.

1.2.3. Bendrojo filtravimo algoritmų palyginimas

Kaip pastebėjo C. Aggarwal [Agg16], nuspėjamų reikšmių skaičiavimas, tiek naudotojais, tiek objektais paremto bendrojo filtravimo atveju, reikalauja $O(k)$ laiko, kur k žymi analizuojamų kaimynų (atitinkamai kitų naudotojų arba kitų objektų) skaičių. Autoriai taip pat pastebi, kad neturint jokio naudotojų poaibio, o tiesiog ieškant panašumų lyginant visus turimus objektus, tai užims $O(k \cdot n)$ laiko. Priklausomai nuo turimų duomenų kiekio, skaičiavimai gali užimti sąlygiškai daug laiko, kas atsispindėtų realiu metu rezultatų generavime, todėl algoritmas ne visais atvejais gali veikti efektyviai. Dėl šios priežasties kai kurie bendruoju filtravimu paremti algoritmai remiasi ne visais sistemoje turimais naudotojais, bet daugiau tais, kurie turi artimesnį ryšį analizuojamam naudotojui. Tai gali būti pritaikoma pasitelkiant kokį nors bendrai naudotojų naudojamą socialinį tinklą, taip išrenkant naudotojų ryšius (draugus), remiantis tekstų autorių duomenimis ar panašiai. Kaip pastebėjo Xiaomei Bai [BWL+19], tam tikri tekstai gali būti populiarūs ne tik vienam analizuojamam naudotojui, bet ir kitiems to naudotojo pažįstamiems. Autorius nagrinėjo mokslinius straipsnius, todėl atkreipė dėmesį, jog panašumams gali daryti didelę įtaką straipsnio cituojamų šaltinių autoriai, t. y. skaitytojas, besidomintis konkrečiu straipsniu, tikėtina, kad labiau susidomės tų autorių straipsniais, kurie skaitomame tekste yra cituojami, negu bet kuriuo kitu, atsitiktinai parinktu, straipsniu.

C. Aggarwal [Agg16], lygindamas naudotojais ir objektais paremtus bendrojo filtravimo metodus, pastebi, jog objektais paremti metodai, visų pirma, dažniausiai yra tikslesni, nes remiasi tik konkretaus naudotojo, kuriam ir generuojama rekomendacija, įverčiais. Tačiau, nors panašūs objektai turėtų būti randami tiksliau, filtravimas gali neveikti taip, kaip tikimasi, jeigu nėra pakankamai žinoma apie naudotoją. Pavyzdžiui, naudotojui susidomėjus vienu tekstu, rekomendacijos bus teikiamos labai panašios į to teksto turinį, bet jei turinys nėra pakankamai išsamus arba aprėpiantis daug unikalių raktažodžių, rekomendacijos gali būti labai ribotos, t. y. per daug susijusios tik su konkrečiu turiniu, nors naudotoją gali dominti ir daug kitų temų.

C. Aggarwal [Agg16] taip pat pastebi ir vieną unikalų objektais paremto bendrojo filtravimo privalumą - gebėjimą naudotojui pateikti rekomendacijos teikimo priežastį. Tai nėra įgyvendina-

ma naudotojais paremtame bendrajame filtravime, nes kiti naudotojai dažniausiai yra anoniminiai asmenys arba tiesiog nėra galima teikti tokios informacijos dėl duomenų apsaugos.

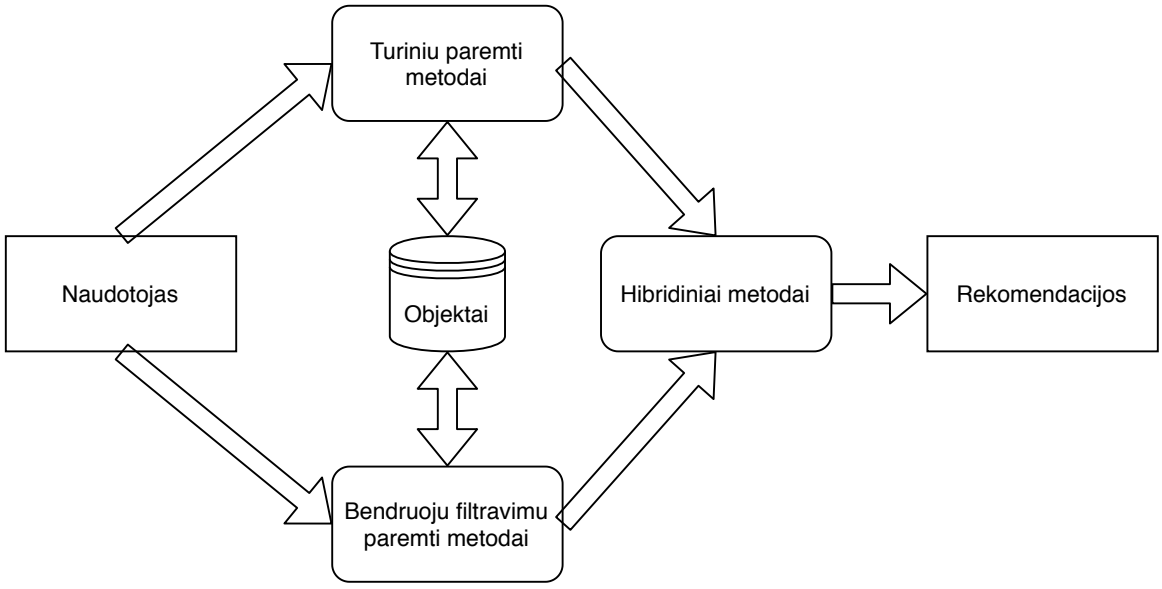
Galiausiai, objektais paremtas bendrasis filtravimas yra labiau pritaikomas norint teikti rekomendacijas realiu metu, nes naudotojų įverčiai yra labiau stabilūs. Taip yra todėl, kad:

1. Sistemose dažniausiai naudotojų yra žymiai daugiau, negu objektų. Net ir keli nauji nagrinėjamo naudotojo įvertinimai (arba keli nauji naudotojai) gali nulemti tai, kad visiškai pasikeis naudotojui panašiausi kaimynai, t. y. turės būti atliekami nauji skaičiavimai jiems nustatyti.

2. Tikėtina, kad naujų naudotojų skaičius didėja greičiau, lyginant su naujų objektų skaičiumi. Atsiradus naujiems naudotojams, objektais paremtu bendrojo filtravimo atveju, įverčiai drastiškai nesikeis, todėl tai nereikalaus daug naujų skaičiavimų, priešingai nei naudotojais paremtuose bendrojo filtravimo algoritmuose.

1.3. Hibridiniai rekomendavimo metodai

Kadangi ir turiniu, ir bendruoju filtravimu paremti rekomendavimo metodai turi tiek įvairių pliusų, tiek minusų, dažnai šių metodų naudojami algoritmai yra apjungiami, taip sukuriant hibridinius rekomendavimo metodus. Hibridinių metodų veikimo principo pavyzdys, kai kombinuojami turiniu paremti ir bendruoju filtravimu paremti metodai, pavaizduotas 3 paveikslėlyje.



3 pav. Hibridinių metodų veikimo principo pavyzdys

Kiti tyrinėtojai [TTS+16] pabrėžia, kad apjungus du ar daugiau rekomendavimo algoritmų yra pagerinamas rekomendacijų tikslumas ir pasiekiamas geresnis algoritmų veikimas. Literatūroje yra išskiriami 7 skirtingi algoritmų apjungimo metodai - svoriais paremtas (angl. *weighted*), perjungimu paremtas (angl. *switching*), mišrus (angl. *mixed*), atributų kombinavimo (angl. *feature combination*), pakopinis (angl. *cascade*), atributų papildymo (angl. *feature augmentation*) ir meta lygio (angl. *meta-level*).

1.3.1. Svoriais paremtas metodas

Šio tipo metodai remiasi tuo, kad apjungiamiems algoritmams yra įvedami svoriai arba tam tikri balsai, nurodantys kurie algoritmai daro didesnę įtaką (turi didesnę svorio (balso) reikšmę). Tokiu būdu pirmenybė yra suteikiama tiems objektams, kuriuos rekomenduotų daug naudotojų, tačiau atrinkti mažą svorį turinčių algoritmų, vietoje tų objektų, kuriuos rekomenduotų mažai naudotojų, tačiau atrinkti didelį svorį turinčių algoritmų.

P-Tango sistema [MCG+99] buvo vienas pirmųjų svoriais paremtų metodų realizacijų, kuri apjungė turiniu paremtus ir bendruoju filtravimu paremtus metodus, įvedant jiems svorius. Šis būdas vis dar išlieka vienas populiariausių [Çan17], tačiau rezultatai labai priklauso nuo tiriamos duomenų aibės, pavyzdžiui, S. Suriati tyrimas [SDT17] parodė, kad rekomendacijų rezultatai, gauti taikant hibridinius (svoriais paremtus) metodus nėra geresni, nei taikant turiniu paremtus ir bendruoju filtravimu paremtus metodus atskirai.

1.3.2. Perjungimu paremtas metodas

Šis metodas remiasi tuo, kad galutinės rekomendacijos yra teikiamos priklausomai nuo esamos situacijos. Taikant šio tipo metodus yra apibrėžiami tam tikri perjungimo kriterijai, nurodantys, kada metodas turi persijungti. Pavyzdžiui, taikant turiniu paremtus metodus, bet nepavykus sugeneruoti pakankamai tikslių rekomendacijų, rekomendacijos pergeneruojamos taikant kitą, bendruoju filtravimu paremtą metodą. Taigi, šio tipo metodai pagerina gaunamus rezultatus, bet rekomendacijų teikimą padaro sudėtingesnį, įvedant papildomus perjungimo kriterijus.

Vienas žymesnių šio metodo pritaikymų - DailyLearner naujienų rekomendavimo sistema, naudojanti perjungimu paremtus metodus tam, kad atsižvelgtų į laiko aktualumą naujienų rekomendavimo metu. Šioje sistemoje naudojamos technikos remiasi tuo, kad, visų pirma, bandoma pateikti einamuoju metu aktualias naujienas, remiantis turiniu paremtais metodais, kurie rezultatų generavimui naudoja artimiausių kaimynų klasifikavimą ir vektorinės erdvės modelį su TF-IDF svoriais. Jei šios technikos nesugeneruoja patikimų rezultatų, sistema persijungia į ilgesnio laiko tarpo modelį, kuris generuoja rezultatus atsižvelgiant į tai, kas konkrečiam naudotojui aktualu buvo ir praeityje (bet galbūt mažiau aktualu einamuoju metu). Pastarasis modelis naudoja Bajeso klasifikatorių, siekiant nustatyti tikimybes, ar tam tikros naujienos bus aktualios konkrečiam naudotojui.

1.3.3. Mišrus metodas

Šio tipo metodai remiasi kelių rekomendavimo algoritmų apjungimu vienu metu. Anot E. Çano [Çan17], šie metodai yra paprasčiausi hibridinių metodų apjungimo būdai, tačiau juos prasminga taikyti tada, kai yra įmanoma apjungi didelį kiekį rekomendavimo algoritmų vienu metu. Tas pats autorius taip pat pateikia kelis šių metodų taikymo pavyzdžius, susijusius su televizijos laidų rekomendacijomis. Dažniausiai jos, visų pirma, sudaro naudotojų profilius (pavyzdžiui, remiantis demografiniais duomenimis), kurie vėliau sudaro įvestį turiniu paremtiems rekomendavimo metodams. Šie metodai paprastai naudoja vektorinės erdvės modelį bei artimiausių kaimynų klasifikatorius, kad nustatytų rekomenduojamas televizijos laidas. Tuo pačiu metu naudojami ir bendruoju

filtravimu paremti metodai, bandantys surasti panašiausius naudotojus ir įvertinti, kiek konkreti laida turėtų patikti konkrečiam naudotojui. Galiausiai abiejų technikų rezultatai yra apjungiami į vieną, atrenkant geriausius rezultatus iš abiejų, vienu metu taikytų technikų.

1.3.4. Atributų kombinavimo metodas

Šio tipo metodai bando pagerinti rekomendacijų rezultatus, taikomam algoritmui pateikiant daugiau įvesties duomenų, gautų iš kitų rekomendavimo metodų. Pavyzdžiui, turiniu paremtiems rekomendavimo metodams gali būti perduodami naudotojų reitingai, kurie įprastai nėra naudojami.

D. Khattar ir kt. [KKG+18] pritaikė atributų kombinavimo metodą siekiant teikti naujienų rekomendacijas. Autoriai naudoja bendruoju filtravimu paremtus metodus naudotojų-objektų sąveikos nustatymui ir turiniu paremtus metodus esminių, konkretų naudotoją dominančių, sričių išskyrimui. Remiantis pastarųjų metodų rezultatais sudaromas naudotojo profilis, kurio duomenys sudaro įvestį neuroninio tinklo modeliui, atsakingam už rekomendacijų generavimą.

Kaip pastebėjo E. Çano [Çan17], atributų kombinavimas vertina naujus atributus tik kaip papildomą įvestį turiniu paremtiems metodams, bet nepasikliauna vien tik šiais duomenimis. Tokiu būdu sumažinamas duomenų jautrumas, o rezultatai, gaunami taikant šiuos metodus, pasižymi didesniu tikslumu.

1.3.5. Pakopinis metodas

Šis metodas naudoja kelias rekomendavimo technikas, palaipsniui vienos technikos rezultatus, kaip įvesties duomenis, perduodant kitai technikai. Rekomendavimo technikų tvarka yra svarbi - pirmoji technika yra naudojama rezultatų, su didesne paklaida, atrinkimui, o vėlesnė - tų pačių rezultatų rikiavimui pagal aktualumą ir konkrečiam naudotojui aktualiausių rezultatų parinkimą.

Tokie metodai dažnai naudojami teikiant rekomendacijas pasitelkus neuroniniais tinklais, kadangi neuroniniai tinklai reikalauja daug resursų, kurie didėja priklausomai nuo duomenų kiekio. Y. Song ir kt. [SEH16] pritaikė pakopinį metodą naujienų rekomendavimui, pasitelkiant bendrojo filtravimo metodo, tiesioginio sklidimo ir rekurentinių neuroninių tinklų kombinacijomis. Autoriai pastebėjo, kad žmonės, besidomintys naujienomis, turi tam tikras susidomėjimo sritis, kurios naudotoją domina ilgą laiko tarpą, trumpą laiko tarpą (trunkantį nuo dienos iki kelių savaičių) bei neapibrėžiamą laiko tarpą (tokios naujienos, kurios aktualios einamu metu daugeliui naudotojui). Pirmajai grupei priklausantys objektai yra labiau statiniai, jų raktažodžiai nėra linkę dažnai keistis. Dėl to, norint rekomenduoti pirmajai grupei priklausančius objektus, autoriai siūlo naudoti iš anksto apmokytą modelį, gebantį nustatyti konkrečiam naudotojui aktualias sritis. Tokiu būdu yra išfiltruojami tie objektai, kurie ilgame laiko tarpe naudotojui nebuvo aktualūs, todėl tolimesniuose etapuose, taikant neuroninius tinklus, tokie objektai net nebus analizuojami. Autoriai pastebi, kad toks būdas ženkliai padidina neuroninio tinklo efektyvumą.

Taigi, pakopiniai metodai pagerina rekomendavimo algoritmų veikimą, kadangi antroji technika nebeturi vertinti prastai pirmos technikos įvertintų objektų, kurie jau yra išfiltruoti ir nebūtų rekomenduojami bet kokiu atveju. Dėl šios priežasties antroji technika orientuosis tik į potencialias

rekomendacijas, todėl analizuos mažiau duomenų. Taigi, veiks efektyviau, nei, pavyzdžiui, svoriu paremti metodai.

1.3.6. Atributų papildymo metodas

Šio tipo metodai apjungia kelias rekomendavimo technikas taip, kad kurios nors technikos rezultatas būtų naudojamas kaip įvestis kitai technikai. Technika yra panaši į atributų kombinavimo, tačiau atributų papildymo technikoje naujasis atributas būna ne atskiras duomenų šaltinis, o kitos technikos rezultatas. Technika yra panaši ir į pakopinę techniką - abi jos veikia paeiliui, o technikų vykdymo eiliškumas yra svarbus. Šio tipo metodai gali padidinti rekomenduojamų objektų įvairovę bei pagerinti pačios rekomendacinės sistemos veikimą, be jokių papildomų sistemos pakeitimų [Çan17].

L. Xiaohui ir kt. [LM12] pritaikė šį metodą praktikoje, apjungiant bendrojo filtravimo ir klasterizavimo technikas. Autoriai aiškino algoritmo veikimą trimis etapais. Pirmajame etape taikomas klasterizavimo algoritmas, suskirstantis naudotojų profilius ir objektų duomenis į panašias grupes (klasterius). Antrajame etape objektais paremto bendrojo filtravimo metodai naudojami panašių klasterių radimui ir apjungimui. Trečiajame etape bendrojo filtravimo metodai naudojami galutinių rekomendacijų generavimui konkrečiam naudotojui. Autoriai pastebėjo, kad šis metodas išsprendžia panašių objektų rekomendavimo problemą, nors ir tai nežymiai atsiliepia ne tokiose tiksliose rekomendacijose.

1.3.7. Meta lygio metodas

Šio tipo metodai naudoja vienos rekomendavimo technikos modelį kaip įvestį kitai rekomendavimo technikai. Taigi, jie veikia panašiai, kaip atributų papildymo metodas. Atributų papildymo metodo atveju pirmoji technika naudojama atributų, kurie vėliau naudojami antroje technikoje, generavimui. Meta lygio metodų atveju visas pirmosios technikos modelis naudojamas kaip įvestis antrajai technikai. Šio metodo privalumas pasižymi tuo, kad antro lygio technika savaime gauna daugiau informacijos iš pirmojo lygmens technikos, pavyzdžiui, ne tik sugeneruotas rekomendacijas, bet ir jų panašumus, tikimybes ir pan.

1.4. Kiti rekomendavimo algoritmai

Be minėtų rekomendavimo metodų ir juose dažniausiai naudojamų rekomendavimo technikų kartais rekomendacinėse sistemose pritaikomi ir kiti algoritmai, kurie geba pasiekti pakankamai gerus rezultatus, priklausomai nuo testuojamos duomenų aibės, tačiau pagal veikimo principą jų negalima priskirti nei vienai iš anksčiau minėtų metodų grupių. A. Rapečkos tyrimas [RMD13] parodė, kad rekomendavimui, kai atsižvelgiama ne tik į objektus, bet ir jų kategorijas bei autorių populiarumą, vienas iš geriausius rezultatus generavusių rekomendavimo algoritmų buvo MostPopular. Iš kitos pusės, tas pats tyrimas parodė, kad turint daugiau duomenų apie naudotoją, tiksliau veikia k-artimiausių kaimynų metodas. Kiti tyrimai [SSZ+17], [Dzi19] parodė, kad geriausi rezultatai gali būti pasiekiami taikant neuroninius tinklus, kuriems taip pat reikia užtikrinto duomenų

kiekio. Toliau yra paaiškinami populiariausių objektų rekomendavimo ir globalių efektų algoritmai.

1.4.1. Populiariausių objektų rekomendavimo algoritmas

MostPopular algoritmas remiasi objektų reitingais - rekomenduojami tie objektai, kurie yra populiariausi tarp kitų naudotojų. Tekstų rekomendavimo atveju, naudotojui būtų rekomenduojami tie tekstai, kurie yra labiausiai skaitomi kitų naudotojų (turi didžiausią reitingą). Kiekvieno teksto įvertis būtų apskaičiuojamas pagal 13 formulę.

$$W_{MostPopular} = \frac{\text{Naudotojų, skaičiusių tekstą, skaičius}}{\text{Visų naudotojų skaičius}} \tag{13}$$

1.4.2. Globalių efektų algoritmas

Globalių efektų (angl. *GlobalEffects*) algoritmas remiasi tiek naudotojo, kuriam teikiamos rekomendacijos, įverčiais, tiek ir visų kitų naudotojų įverčiais visiems kitiems objektams. Algoritmo tikslas - normalizuoti šių įverčių reikšmes, kad jos labiau atspindėtų konkretaus naudotojo (arba konkretaus objekto) įverčius, atsižvelgiant į kitus naudotojus (arba objektus). Pavyzdžiui, jeigu analizuojamas naudotojas vidutiniškai objektus vertina balu 3, bet visų kitų naudotojų įverčių vidurkis - 3.5, tikėtina, kad naudotojas objektus vertina mažesniais balais, todėl rekomendavimo algoritmas, apskaičiuodamas nuspėjamo įverčio reikšmę, turėtų atsižvelgti tiek į naudotojo įverčių, tiek ir visų kitų naudotojų vidurkio reikšmes (šiuo atveju iš nuspėjamos reikšmės būtų atimtas skirtumas (0.5)).

1.5. Įvertinimo matai

Norint nustatyti, kaip efektyviai veikia tam tikras rekomendavimo algoritmas rekomenduojamiems objektams, yra taikomi įvairūs įvertinimo matai ([Rap15], [DFC+19], [Sta19] ir kt.). Toliau esančiuose skirsniuose yra aptariami preciziškumo (P), jautrumo (R), harmoninio preciziškumo ir jautrumo vidurkio (F), preciziškumų vidurkių (MAP), normalizuoto diskontuoto suminio naudingumo (NDCG), ploto po ROC kreive (AUC), naujumo ir dokumentų aprėpties įvertinimo matai. Kaip pastebėjo A. Rapečka, [Rap15], siekiant surasti efektyviausiai su konkrečiu duomenų rinkiniu veikiančius rekomendavimo metodus, jų veikimo rezultatus būtina vertinti ne pagal kurį nors atskirą įvertį, o pagal šių įverčių visumą. Dėl šios priežasties, atliekant eksperimentinį naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimą, yra skaičiuojamos visos šios įvertinimo matų reikšmės.

1.5.1. Preciziškumas (P)

Preciziškumas (angl. *precision*) įvertina taikyto metodo tikslumą, atsižvelgiant į teisingai rekomenduotų objektų skaičių (tai, kiek tekstų naudotojas iš tikrųjų perskaitė) ir visų rekomenduotų

objektų skaičių. Kuo didesnė preciziškumo reikšmė, tuo tiksliau veikia taikomas rekomendavimo algoritmas ir pati rekomendacinė sistema. Šis matas apibrėžiamas tokia formule:

$$\textit{Preciziškumas} = \frac{\textit{Teisingai rekomenduotų objektų skaičius}}{\textit{Visų rekomenduotų objektų skaičius}} \tag{14}$$

1.5.2. Jautrumas (R)

Jautrumas (angl. *recall*) parodo teisingai rekomenduotų objektų skaičiaus ir visų objektų, kurie turėjo būti rekomenduoti, skaičiaus santykį. Šis matas apibrėžiamas tokia formule:

$$\textit{Jautrumas} = \frac{\textit{Teisingai rekomenduotų objektų skaičius}}{\textit{Visų objektų, kurie turėjo būti rekomenduoti, skaičius}} \tag{15}$$

Kuo didesnė šio įverčio reikšmė, tuo rekomendavimo algoritmas geba tiksliau išskirti labiausiai aktualius objektus tam tikram naudotojui. Taigi, kuo didesnė jautrumo reikšmė, tuo tiksliau veikia ir pati rekomendacinė sistema.

1.5.3. Harmoninis preciziškumo ir jautrumo vidurkis (F)

Harmoninis preciziškumo ir jautrumo vidurkis remiasi tuo, kad preciziškumo ir jautrumo įverčiai gali prieštarauti vienas kitam - rekomenduojamų objektų skaičiui augant, jautrumo reikšmė gali didėti, o preciziškumo reikšmė mažėti. Dėl to šie matai yra apjungiami ir naudojama tokia harmoninio vidurkio reikšmė:

$$F = \frac{2 \cdot \textit{Preciziškumas} \cdot Jautrumas}{\textit{Preciziškumas} + Jautrumas} \tag{16}$$

Jei preciziškumo ir jautrumo reikšmės sutampa, tuomet F reikšmė bus lygi aritmetiniam vidurkiui, o kai reikšmės yra skirtingos, tuomet F reikšmė bus artimesnė mažesnei reikšmei. Apskritai, kuo ši reikšmė yra didesnė, tuo rekomendavimo algoritmas ir pati rekomendacinė sistema veikia tiksliau.

1.5.4. Preciziškumų vidurkis (MAP)

Preciziškumų vidurkis yra apskaičiuojamas iš pradžių nustatant vidutinį preciziškumą (angl. *Average Precision*):

$$AP = \frac{1}{m} \sum_{k=1}^{N} \textit{Preciziškumas}(R_k) \tag{17}$$

čia m - teisingai rekomenduotų objektų skaičius naudotojui u, N - visų rekomenduotų objektų skaičius, $\textit{Preciziškumas}(R_k)$ - preciziškumo reikšmė, apskaičiuota pagal 14 formulę, kur R_k nurodo eilę pirmųjų rekomenduotų objektų, iki kol buvo rekomenduotas objektas k.

Žinant vidutinį preciziškumą kiekvienam naudotojui, MAP reikšmė nustatoma pagal 18 formulę. Kaip pastebėjo A. Rapečka [Rap15], šis įvertis išsiskiria savo stabilumu, tačiau norint tiksliai įvertinti efektyvumą, reikia patikrinti didelę ir pakankamai diversifikuotą testavimo aibę.

$$MAP = \frac{1}{U} \sum_{k=1}^{U} AP(k) \tag{18}$$

čia U - naudotojų skaičius, AP - vidutinis preciziškumas.

1.5.5. Normalizuotas diskontuotas suminis naudingumas (NDCG)

Normalizuotas diskontuotas suminis naudingumas (*NDCG*, angl. *Normalized Cumulative Discounted Gain*) taikomas norint įvertinti naudotojui parekomenduoto objektų rinkinio kokybę, t. y. nustatyti, kaip tiksliai naudotojui buvo pateikta tai, ko jis tikisi. Vidutinis diskontuotas suminis naudingumas produktų rinkiniui J įvertinamas taikant tokią formulę:

$$DCG = \frac{1}{m} \sum_{u=1}^{m} \sum_{j=1}^{J} \frac{g_{uj}}{max(1, \log_b j)} \tag{19}$$

čia m - naudotojų skaičius, J - rekomenduotų objektų skaičius, j - objekto indeksas objektų aibėje, b - nekintanti reikšmė, paprastai imama nuo 2 iki 10, g_{uj} - nauda, kurią gauna naudotojas u iš objekto j.

Žinant vidutinį diskontuotą sumį naudingumą produktų rinkiniui, normalizuotas diskontuotas suminis naudingumas apskaičiuojamas diskontuotą sumį naudingumą padalijant iš maksimalaus galimo (20 formulė). Jei naudotojui atrodo, kad tam tikras tekstas (objektas) buvo parekomenduotas labai tiksliai, tai nauda, kurią gauna naudotojas iš objekto, turės didelę reikšmę. Jei rekomendacija visiškai neatitiko naudotojo lūkesčių, tai nauda bus lygi 0.

$$NDCG = \frac{DCG}{maxDCG} \tag{20}$$

1.5.6. Plotas po ROC kreive (AUC)

Plotas po ROC kreive (*AUC*, angl. *Area Under the ROC Curve*) parodo eilės nustatymo kokybę. ROC kreivė atvaizduoja teisingai rekomenduotų objektų dalies priklausomybę nuo neteisingai rekomenduotų (rekomenduotų, bet naudotojo nesudominusių) objektų dalies. Kuo ploto po ROC kreive įvertis yra didesnis (kuo arčiau 1), tuo rekomendavimo algoritmas veikia tiksliau ir siūlo tikslesnes rekomendacijas naudotojams.

1.5.7. Naujumas (Novelty)

Naujumas (angl. *Novelty*) įvertina rekomendacijų populiarumą. Kitaip nei kitos metrikos, naujumo reikšmė negali įvertinti rekomendacijų tikslumo ar to, kaip tiksliai rekomendacija atitiko naudotojo lūkesčius. Ši reikšmė tik atspindi, kiek pasiūlyta rekomendacija iki tol buvo nežinoma naudotojui. Taip pat svarbu pastebėti, jog nėra tikslo šios reikšmės nei maksimizuoti, nei minimizuoti. Aukštas naujumo įvertis parodo, kad algoritmas rekomenduoja mažiau žinomus dokumentus, taigi, naudotojui yra pateikiama įvairesnių tekstų. Žemas naujumo įvertis signalizuoja užtikrintesnes rekomendacijas, remiantis tuo, jog naudotojas panašiais tekstais jau domėjosi. Antruoju atveju kyla rizika, jog į rekomendacijas nebus įtraukti nauji dokumentai, apie kuriuos naudotojas iki tol nežinojo, tačiau galimai būtų susidomėjęs.

1.5.8. Dokumentų aprėptis (Coverage)

Dokumentų aprėptis (angl. *Coverage*) reprezentuoja visų galimų rekomendacijų procentinę dalį ir yra apibrėžiama tokia formule:

$$\textit{Dokumentų aprėptis} = \frac{n}{N} * 100 \tag{21}$$

čia n - rekomenduotų dokumentų skaičius, N - visų galimų rekomendacijų skaičius.

Nauji dokumentai visoje dokumentų aibėje sumažina aprėpties įverčio reikšmę, kadangi dokumentas, prieš jį rekomenduojant, turi būti įvertintas (perskaitytas) bent kelių kitų naudotojų. Taip pat, jeigu naudotojai dokumentą pradeda skaityti, bet juo nesusidomi, yra laikoma, jog naudotojo įvertis skaitytam dokumentui yra mažas. Norint rekomenduoti dokumentus, kurių įvertis tam tikro naudotojo (ar jam panašių naudotojų) yra vertinamas daugiau už nustatytą įvertį, kai kurie dokumentai į rekomendacijų aibę nebepatenka. Dėl šių priežasčių gaunama mažesnė dokumentų aprėpties įverčio reikšmė. Bendrai ši reikšmė yra mažesnė bendruoju filtravimu paremtuose algoritmuose ir signalizuoja apie tuščių įvertinimų arba naujų objektų problemas.

1.6. Rekomendavimo algoritmų problemos

Toliau esančiuose skirsniuose aptariamos aktualios rekomendavimo algoritmų problemos: tuščių įvertinimų, naujų naudotojų ir naujų objektų, pasitikėjimo rekomendacijomis ir privatumo, panašių objektų rekomendavimo ir mažos įverčių apimties, didėjančių duomenų bei laiko dimensijos vertinimo problemos.

1.6.1. Tuščių įvertinimų problema

Kaip pastebėjo X. Bai ir kt. [BWL+19], daugelis rekomendacinių sistemų veikia su prielaida, kad naudotojų skaičius yra žymiai didesnis už objektų skaičių, tačiau kartais gali būti atvirkščiai arba net populiariausi objektai gali būti įvertinti vos kelių naudotojų. Tokiu atveju yra susiduriama

su tuščių įvertinimų problema, kuri yra aktuali daugeliui bendruoju filtravimu paremtų rekomendacinių sistemų ir dėl kurios rekomendaciniai algoritmai negali veikti pakankamai tiksliai ir teikti užtikrintų rekomendacijų naudotojams. Ši problema iškyla tada, kai saugomų objektų skaičius yra labai didelis - kiekvienas naudotojas susidomi tik keliais ar keliolika iš jų, todėl naudotojų-objektų matrica vis tiek lieka gana tuščia. Tokiu atveju bendruoju filtravimu paremti metodai neveikia efektyviai, nes algoritmai nesugeba rasti kitų panašių naudotojų, kurių rezultatais remiantis būtų galima teikti naujas rekomendacijas.

1.6.2. Naujų naudotojų ir naujų objektų problemos

Naujiems naudotojams, kurie nėra įvertinę (skaitę) sistemoje saugomų objektų (tekstų), bendro filtravimo algoritmai nesugebės pakankamai tiksliai surasti panašių naudotojų. Ta pati problema nutiks ir bandant surasti panašius objektus naujam sistemoje užregistruotam tekstui, jei juo susidomėjo nedidelė dalis naudotojų. Tačiau abi problemos pasireiškia vienu metu tik tuo atveju, jeigu taikomas tik vienas, bendruoju filtravimu paremtas algoritmas. Vienas iš šiuo metu dažniausiai taikomų būdų naujų objektų problemos sprendimui - turinių paremtų metodų taikymas [DFC+19]. Turiniu paremti metodai išsprendžia naujų objektų problemą, kadangi jie analizuoja tik tekstų turinį, taigi, naujų objektų problema nebus didelis iššūkis bet kuriai rekomendacinei sistemai, jei sistemoje bus saugoma pakankamai objektų. Tačiau turiniu paremti metodai neišsprendžia naujų naudotojų problemos, kadangi jų veikimas grindžiamas naudotojo profilio sudarymu, kuriam sudaryti reikia žinoti naudotoją dominančias sritis. Šios problemos sprendimas dažnai yra paliekamas naudotojui, leidžiant jam pasirinkti jį dominančias sritis. Norint užtikrinti algoritmo veikimą be naudotojo įvesties, iš pradžių naujam naudotojui galima pasiūlyti populiariausius tekstus, pritaikant, pavyzdžiui, MostPopular algoritmą, paaiškintą 1.4.1 skirsnyje.

1.6.3. Pasitikėjimo rekomendacijomis ir privatumo problemos

Daugelis rekomendacinių sistemų veikia taip, kad generuojant rezultatus nebeįmanoma nustatyti pradinės priežasties, kodėl gaunama, kad vienas ar kitas objektas turėtų sudominti naudotoją. Taigi, naudotojai daugeliu atveju nežino, kaip rekomendacijos yra generuojamos, kaip yra sukurtos rekomendacinės sistemos ir kokiais naudotojų vertinimais jos remiasi. Kaip pastebi B. Kumar [KS16], kai kurios rekomendacinės sistemos apskritai bando surinkti kuo daugiau informacijos apie naudotoją, kad galėtų teikti kuo tikslesnes rekomendacijas, tačiau tai natūraliai sukelia naudotojų nepasitikėjimą rekomendacine sistema, nes naudotojai nėra linkę dalintis privačia informacija vien tam, kad gautų rekomendacijas. Taigi, rekomendacinė sistema neturėtų rinkti daugiau duomenų apie naudotoją, nei iš tikrųjų reikia. Norint pateikti naudotojui priežastis, dėl ko rekomenduojamas vienas ar kitas objektas, galima remtis objektais paremto bendrojo filtravimo metodu. Kaip ir minėta 1.2.3 skirsnyje, šio metodo privalumas - gebėjimas naudotojui pateikti rekomendacijos teikimo priežastį.

1.6.4. Panašių objektų rekomendavimo ir mažos įverčių apimties problema

Esant mažai įverčių apimčiai, t. y. kai sistemoje nėra išsaugota pakankamai naudotojų vertinimų, rekomendavimo algoritmai siūlo visiems tuos pačius panašius objektus, kuriuos naudotojas, ar kiti panašūs naudotojai yra įvertinę. A. Rapečka [Rap15] pamini 2 galimus sprendimo būdus:

- Panašumo tarp naudotojų slenksčio mažinimas, tokiu būdu būtų randama daugiau naudotojui panašių kitų sistemos naudotojų, tačiau dėl to sumažėja rekomendacijų tikslumas.
- Panašumo tarp naudotojų slenksčio ir rekomendacijų tikslumo didinimas, tačiau tokiu būdu kardinaliai sumažinamas rekomenduojamų objektų skaičius.

Taigi, norint išspręsti panašių objektų rekomendavimo problemą, dažniausiai nukenčia rekomendacijų tikslumas. Kaip pastebi B. Kumar [KS16], ši problema vis dar nėra išspręsta, nors, apjungus bendrojo filtravimo ir turiniu paremtus rekomendavimo metodus, problema sumažėja, nes didesnis dėmesys kreipiamas objekto turiniui, bet ne kitų naudotojų vertinimams. Šios problemos aktualumą rekomendavimo algoritmui galima įvertinti analizuojant naujumo įvertį, aptartą 1.5.7 skirsnyje.

1.6.5. Didėjančių duomenų problema

Xiaofeng Li [LL19] išskiria realiu metu rekomendacijų teikimo, kurios yra apsunkinamos dėl bandomo apdoroti didelių duomenų kiekio, problemą, aktualią taip pat ir tekstų rekomendavimo algoritmams, jei rekomendacijas norima teikti atsižvelgiant į naujausius turimus duomenis, kurie kasdien yra didėjantys. Kaip atkreipia dėmesį B. Kumar [KS16], tradicinės turiniu paremtos rekomendacinės sistemos bei bendruoju filtravimu paremtos rekomendacinės sistemos buvo kuriamos taip, kad teiktų rekomendacijas saugant tik nekintančius (statinius) duomenis. Tačiau šiais laikais, duomenims didėjant kasdien, būtina kurti algoritmus, gebančius apdoroti dinamines duomenų aibes. Tokie algoritmai dažniausiai taikomi hibridinėse rekomendacinėse sistemose.

1.6.6. Laiko dimensijos vertinimas

Kaip pastebėjo B. Fortuna ir kt. [FFM10], tekstai gali būti klasifikuojami pagal laiko aktualumą į ilgalaikius (moksliniai straipsniai, mokslinė literatūra) ir trumpalaikius (tokius, kas aktualu konkrečiam žmogui ar grupei žmonių, bet dažniausiai tik tam tikru laiko momentu). Norint rekomenduoti antrajai grupei priklausančius tekstus, būtina įvertinti, ar rekomenduotinas tekstas bus aktualus ir rekomenduojamu metu.

Kaip pastebėjo Y. Song ir kt. [SEH16], tekstų grupės, kurių aktualumas labiau priklauso nuo esamo laiko, gali būti skirstomi į:

- Konkrečiam naudotojui aktualias ilgą laiko tarpą. Tai, pavyzdžiui, pomėgiai, hobiai ar kiti panašaus turinio tekstai.
- Konkrečiam naudotojų aktualias trumpą laiko tarpą (trunkantį nuo dienos iki kelių savaičių). Tai dažniausiai įvairios aktualijos, su tam tikra istorija susijusios naujienos ir pan.

- Daugeliui naudotojų aktualias neapibrėžtą laiko tarpą. Tai naujienos, kurios aktualios einamuoju metu ir nėra aišku, kada daugeliui naudotojų jos taps nebeaktualios.

Ši problema galėtų būti sprendžiama taikant hibridinius metodus, realizuojant kelių algoritmų apjungimą. Pritaikius svoriais paremtą metodą būtų galima įvesti tam tikrus balsus, nurodančius, kuris algoritmas turi didesnį svorį. Tokiu būdu būtų sumažinama tikimybė, kad naudotojui bus rekomenduojama kažkas, kas nebėra aktualu konkrečiam naudotojui einamuoju metu.

Pritaikius perjungimu paremtą metodą būtų galima realizuoti rekomendacijų teikimą, atsižvelgiant į anksčiau minėtas tekstų grupes, t. y. rekomenduoti patikimiausius rezultatus iš vienos grupės, tačiau, jei tokių rekomendacijų nėra arba jos nepakankamai tikslios, perjungti metodą ir siūlyti rezultatus pasikliaunant kita technika. Tačiau tokiu būdu gali būti susiduriama su panašių objektų rekomendavimo problema, nes visi objektai būtų išskirti pagal svarbą atsižvelgiant tik į vieną grupę.

Atributų kombinavimo metodas galėtų užtikrinti, kad pirmoji naudojama technika įvertins konkretaus objekto aktualumą einamuoju metu, o šie rezultatai bus perduodami antrajai technikai, kuri apjungs rezultatus ir teiks rekomendacijas konkrečiam naudotojui, atsižvelgiant į pirmosios technikos rezultatus.

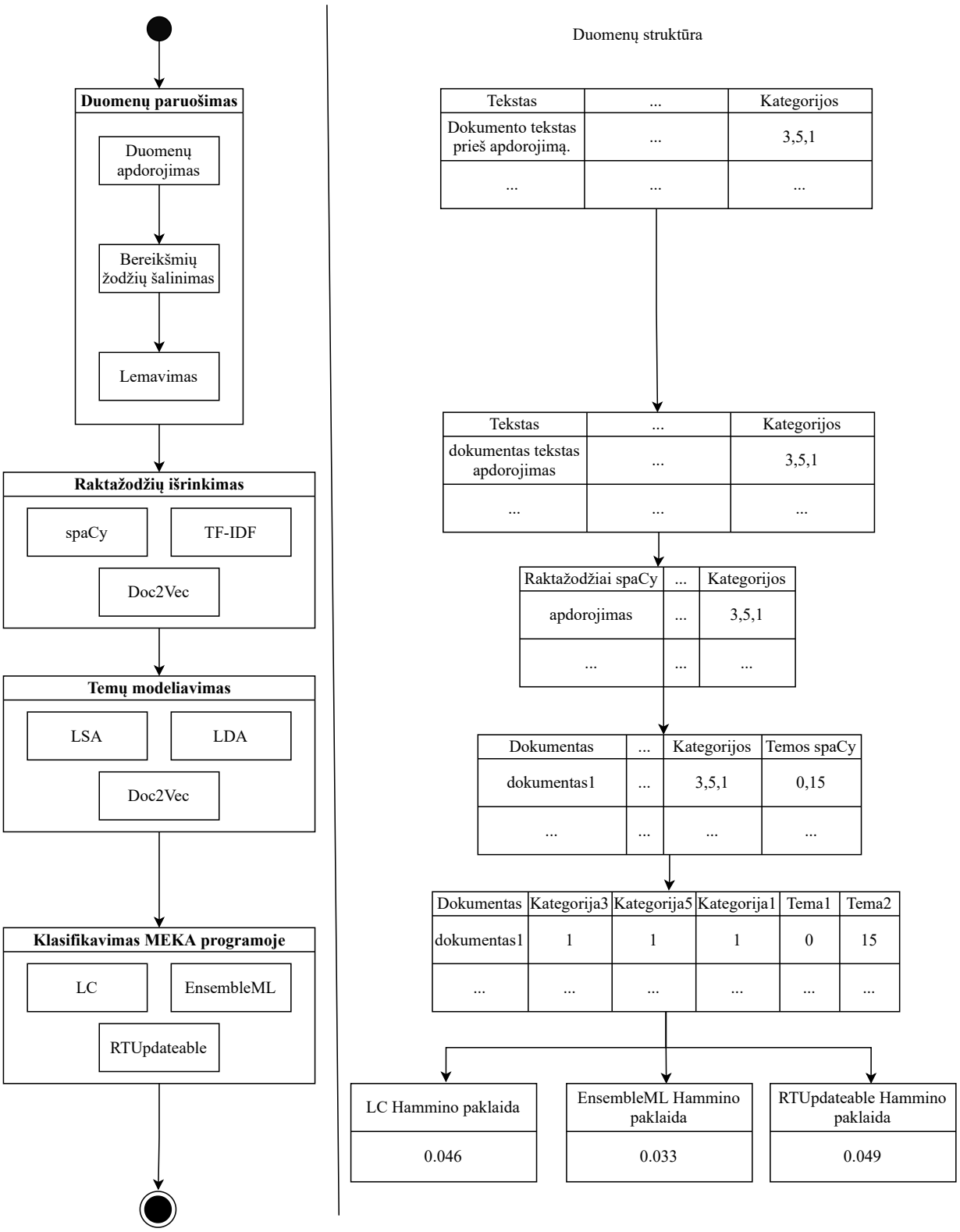
2. Naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimas

Visų pirma, norint įgyvendinti šio darbo išsikeltą tikslą, t. y. sukurti naują tekstų rekomendavimo algoritmą, tinkamą taikyti lietuvių kalba parašytų tekstų rekomendacijų gavimui, kai naudotojai yra mažai pažįstami, turi būti atliktas eksperimentinis naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimas. Šio tyrimo rezultatai padės pamatyti, kokie algoritmai turimai duomenų aibei galimai veikia tiksliau ir efektyviau.

Norint tyrimo metu vertinti turiniu paremtus rekomendavimo algoritmus arba hibridinius algoritmus, kurių vienas iš apjungiamų algoritmų remtųsi dokumento turinio duomenimis, turimų duomenų turinį reikia transformuoti į tam tikrą struktūrinę formą, išskiriant dokumento požymius, t. y. raktažodžius. Kaip aprašyta teoriniame turiniu paremtų rekomendavimo metodų aprašyme (1.1.1), norint sumažinti analizuojamų požymių aibę bei tiksliau išskirti dokumento požymius, duomenis reikėtų iš pradžių tinkamai apdoroti (lemuoti). Šie veiksmai yra aprašomi 2.1 poskyryje.

Siekiant nustatyti, kuris raktažodžių išrinkimo metodas tiriamiems duomenims veikia tiksliausiai, išskirti raktažodžiai turi būti grupuojami ir lyginami tarpusavyje, generuojant dominuojančias raktažodžių temas. Algoritmas, tiksliausiai išskiriantis tiriamų duomenų raktažodžius, bus naudojamas raktažodžių radimui toliau šiame darbe aprašomuose tyrimuose - naudojamų tekstų rekomendavimo algoritmų efektyvumo vertinime bei naujo algoritmo kūrime. Raktažodžių išrinkimo metodų vertinimas yra aprašomas 2.2 poskyryje.

Galiausiai, išskirti raktažodžiai yra naudojami kaip įvestis naudojamų tekstų rekomendavimo algoritmų efektyvumo vertinimui, kuris detaliau aprašytas 2.3 poskyryje. Duomenų paruošimo, raktažodžių išrinkimo, temų modeliavimo ir klasifikavimo pagrindiniai žingsniai pateikiami 4 paveikslėlyje. Dešinėje diagramos pusėje pateikiama pavyzdinė duomenų struktūra, reprezentuojanti turimų duomenų formą kiekviename etape.



4 pav. Duomenų paruošimas raktažodžių klasifikavimui ir vertinimui

2.1. Duomenų aibė ir duomenų paruošimas

Naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimui buvo remtasi turimais lietuvių kalba rašytais naujienų portalų duomenimis. Duomenų aibę iš viso sudarė 3653 tekstiniai dokumentai, jų kategorijų identifikacijų numeriai, parodantys, kuriai tekstų grupei (kategorijai)

tas tekstas priklauso bei 86609 nuasmenintų naudotojų įrašai, parodantys, kiek sekundžių vienas naudotojas domėjosi konkrečiu dokumentu, t. y. jį skaitė. Formatas, pavaizduojantis įvertinimų duomenų informaciją, pateikiamas 2 lentelėje.

2 lentelė. Įvertinimų duomenys

Naudotojo Id	Teksto Id	Skaitymo laikas, s
eU9Z2u1453...	A0DE427AB...	62

Siekiant išrinkti algoritmą, kuris tinkamiausiai gebėtų klasifikuoti nagrinėjamų dokumentų raktažodžius, buvo remtasi duomenų poaibiu, kurį sudarė ~20% visų duomenų aibės (739 tekstiniai dokumentai). Tekstinių duomenų formatas, parodantis nagrinėtus duomenis apie kiekvieną dokumentą, pateikiamas 3 lentelėje.

3 lentelė. Tekstų duomenys

Teksto Id	Tekstas	Kategorijos	Sukūrimo data
A0DE427AB...	...	1000,1023,1057	2022-01-01T03:30:00Z

Norint aprašomame darbe pritaikyti literatūros apžvalgoje aprašytus duomenų paruošimo ir lemavimo metodus, pasirinkta spaCy natūralios kalbos apdorojimo biblioteka. Šios bibliotekos oficiali dokumentacija [HMV+20] teigia, jog bibliotekos įrankiai gali būti naudojami apdorojant didelius tekstų kiekius, kuriant, pavyzdžiui, informacijos išrinkimo, natūralios kalbos apdorojimo, giliojo mokymo programas ir kt. Šiuo metu biblioteka palaiko daugiau nei 60 kalbų, tarp kurių yra ir lietuvių kalba. Be to, *spaCy* projektas yra atviro kodo sprendinys, kuris yra aktyviai tobulinamas daugiau nei 500 programuotojų.

Bibliotekoje aprašytam lietuvių kalbos palaikymo moduliui buvo pridėtos papildomos bereikšmių žodžių šalinimo ir lemavimo funkcijos:

- Bereikšmių žodžių šalinimo funkcijos tikslas - pašalinti iš anksto bibliotekoje apibrėžtus nereikšmingus žodžius (tokius kaip *aš, tu, ir, arba, ai, ak* ir pan.). Naudojamoje bibliotekos 3.0.0 versijoje lietuvių kalbos modelyje tokių žodžių yra aprašyta 1314.
- Lemavimo funkcijos tikslas - sugrupuoti sudurtines žodžio formas, kad jas būtų galima analizuoti kaip vieną elementą, identifikuojamą pagal žodžio bendraties formos šaknį. Pavyzdžiui, žodžiai *būti, yra, bus* šios funkcijos dėka bus apjungti ir analizuojami kaip tas pats įvesties vienetas (*būti*).

2.2. Raktažodžių išrinkimo metodų vertinimas

Raktažodžių išrinkimui iš paruoštų duomenų buvo realizuoti 3 metodai:

- spaCy bibliotekos sprendinys, surikiuojantis analizuojamo teksto žodžius ir grąžinantis svarbiausius raktažodžius, neatsižvelgiant į kitų dokumentų turinį.

- TF-IDF įverčio skaičiavimo metodas, įvertinantis kiekvieno žodžio svarbą visame dokumentų rinkinyje.
- Doc2Vec algoritmo taikymo metodas, besiremiantis dviejų sluoksnių neuroniniais tinklais, kurie atlieka dokumentų vektorizavimą, taikant apmokymo be mokytojo strategiją.

Šių metodų pagalba išrinkti raktažodžiai vėliau yra modeliuojami į juos siejančias temas. Tokiu būdu įgalinamas raktažodžių išrinkimo metodų vertinimas. Temų modeliavimui pasirinkti 3 literatūros apžvalgoje aprašyti metodai:

- LSA metodas, besiremiantis TF-IDF ir ypatingųjų reikšmių dekompozicija.
- LDA metodas, generuojantis dokumente vyraujančių žodžių aibę, besiremiant visų žodžių dažniais analizuojamame dokumente [AOA+19].
- Doc2Vec algoritmas, naudojantis dviejų sluoksnių neuroninius tinklus ir perduodantis gautą įvestį mašininio mokymo klasterizavimo metodui. Aprašomame darbe buvo pritaikytas k-vidurkių klasterizavimo metodas.

Temų modeliavimui aprašomame darbe pasirinktas *Gensim* sprendinys. Kaip teigia šios bibliotekos oficiali dokumentacija [ŘS10], bibliotekoje aprašyti metodai yra skirti ir gali būti naudojami temų modeliavimui, dokumentų indeksavimui bei panašumų nustatymui dirbant su didelėmis duomenų aibėmis. Šis projektas taip pat yra atviro kodo sprendinys, aktyviai tobulinamas beveik 400 programuotojų.

Kiekvienas iš aprašytų metodų sugrupuoja gautus dokumentus į jiems panašias temas. Taigi, kiekvienas iš dokumentų yra priskiriamas vienai arba daugiau dominuojančių grupių. Kadangi pradinėje duomenų aibėje yra žinoma informacija apie grupes, kurioms tekstas iš tikrųjų priklauso, ši problema gali būti sprendžiama kaip daugiažymio (angl. *multi-label*) klasifikavimo uždavinys.

Tokio uždavinio tikslas - kuo tiksliau nuspėti klasei priklausančias žymes, kurių gali būti 0 arba daugiau. Aprašomu atveju klasės reikšmė yra temų modeliavimo metodo priskirtos reikšmės (t. y. kategorijos, kurioms dokumentas priklauso), o žymės - kategorijų indeksai. Klasifikavimo metu pagal panašiems dokumentams (kurie priskirti toms pačioms/panašioms temoms) priskirtas kategorijas, yra bandoma nuspėti nagrinėjamo dokumento žymes.

Daugiažymio klasifikavimo uždaviniui spręsti pasirinkta MEKA [RRP+16] - atviro kodo Java karkasas, skirtas palengvinti praktinį daugiažymių klasifikatorių pritaikymą, įskaitant daugiažymio klasifikavimo uždavinio eksperimentų aprašymą, naujų algoritmų kūrimą ir jų vertinimą. Uždaviniui vertinti pasirinkti 3 klasifikatoriai:

- LC - vertinantis kiekvieną žymių kombinaciją kaip atskirą žymę.
- EnsenbleML - naudojantis kelis mokymosi algoritmus, kad pasiektų geresnį nuspėjamą našumą, nei būtų galima gauti naudojant bet kurį iš sudedamųjų mokymosi algoritmų atskirai.
- RTUpdateable - kiekvienai žymių kombinacijai priskiriantis tik vieną iš sudedamųjų žymių, kurios vėliau yra naudojamos kaip įvestis standartiniam klasifikatoriui.

Daugiažymio klasifikavimo uždaviniams vertinti yra naudojama Hammino paklaida (angl. *Hamming loss*). Šis įvertis parodo neteisingai nuspėtų žymių santykį ir yra apibrėžiamas tokia formule:

$$HL = \frac{1}{|N| \cdot |L|} \sum_{i=1}^{|N|} \sum_{j=1}^{|L|} Y_{i,j} \oplus Z_{i,j} \tag{22}$$

čia $Y_{i,j}$ yra žymių aibė, kurioms dokumentas iš tikrųjų priklauso, $Z_{i,j}$ - žymių spėjimų aibė, $\oplus$ - XOR funkcija.

Daugiažymio klasifikatoriaus tikslas - minimizuoti Hammino paklaidos reikšmę, t. y. kuo ši reikšmė yra mažesnė, tuo tiksliau veikia klasifikatorius. Hammino paklaidos reikšmės tirtiems (LC, EnsembleML, RTUpdateable) klasifikatoriams yra pateikiamos 4 lentelėje.

4 lentelė. Raktažodžių išrinkimo metodų vertinimas

Klasifikatorius	LDA			LSA			Doc2Vec		
	spaCy	TF-IDF	Doc2Vec	spaCy	TF-IDF	Doc2Vec	spaCy	TF-IDF	Doc2Vec
LC	0.033	0.032	0.032	0.046	0.043	0.041	0.041	0.036	0.036
EnsembleML	0.038	0.036	0.038	0.033	0.031	0.031	0.038	0.038	0.038
RTUpdateable	0.053	0.053	0.051	0.049	0.046	0.044	0.047	0.046	0.047

Žymėjimai:	Blogiausias rezultatas	Geriausias rezultatas

Iš gautų rezultatų matyti, jog spaCy metodas raktažodžių išrinkimui veikia mažiausiai tiksliai (visais tirtais atvejais buvo blogiausias rezultatas). Tarp TF-IDF ir Doc2Vec raktažodžių išrinkimo metodų didelių skirtumų nepastebėta, kadangi jų rezultatai varijuoja ir priklauso nuo taikomo temos modeliavimo metodo. Doc2Vec algoritmas temos modeliavimui yra mažiausiai tinkamas taikyti tirtiems duomenims - gautos Hammino paklaidos reikšmės nei vienam iš tirtų klasifikatorių nebuvo minimalios, lyginant su kitais metodais. Tuo tarpu raktažodžių išrinkimui tiek Doc2Vec, tiek TF-IDF metodai gali būti taikomi.

2.3. Naudojamų tekstų rekomendavimo algoritmų efektyvumo vertinimas

Turiniu paremtų algoritmų ir bendruoju filtravimu paremtų algoritmų efektyvumo vertinimui pasirinktas atviro kodo Python sprendinys, naudotas M. F. Dacrema ir kt. tyrime [DCJ19a]. Su šia biblioteka buvo įvertinti 3 turiniu ir 12 bendruoju filtravimu paremtų algoritmų, 1 hibridinis ir 3 kiti algoritmai. Papildomai buvo įtraukti ir su ta pačia biblioteka įvertinti 2 hibridiniai rekomendavimo algoritmai, nagrinėti kitų tyrinėtojų - LightFM [Kul15], CFeCBF [DFC+19], 2 mišrūs metodai (itemKNNCF + P3alpha, ItemKNNCF + pureSVD) ir 1 svoriais paremtas metodas (ItemKNNCFCBF*w).

Algoritmų vertinimui buvo naudoti Amazon WS resursai - Amazon SageMaker Studio Notebook su ml.t3.2xlarge virtualiu serveriu (angl. *instance*).

2.3.1. Turiniu paremti rekomendavimo algoritmai

Atliekant naudojamų tekstų rekomendavimo algoritmų efektyvumo vertinimą bei bandant įvertinti turiniu paremtų rekomendavimo algoritmų efektyvumą, naujų tekstų tinkamumo nustatymui pasirinktas artimiausių kaimynų klasifikavimo metodas. Kaip buvo minėta 1.1.4 skirsnyje, šis metodas panašiausių objektų radimui gali naudoti kelias panašumų nustatymo formules. Atliktame tyrime buvo pasirinktos 3 - kosinuso panašumas, Jaccard ir Euklido atstumai. Gauti tyrimo rezultatai yra pateikiami 5 lentelėje.

5 lentelė. Turiniu paremtų rekomendavimo algoritmų efektyvumo palyginimas

Algoritmas / Metrika	ItemKNN cosine	ItemKNN euclidean	ItemKNN jaccard
P	0.000829	0.000728	0.000802
R	0.013336	0.011812	0.012591
F	0.001560	0.001371	0.001507
MAP	0.004001	0.003443	0.003271
NDCG	0.006066	0.005334	0.005323
AUC	0.009703	0.008251	0.008714
Novelty	0.010116	0.010405	0.010445
Coverage	0.753353	0.851903	0.757460
Mokymo laikas, s	0.18	2.61	0.16
Testavimo laikas, s	4.52	4.17	4.50

Žymėjimai:	Blogiausias rezultatas	Geriausias rezultatas

Iš gautų rezultatų matyti, kad geriausi rezultatai tirtiems duomenims buvo gauti panašumų nustatymui taikant kosinuso panašumo nustatymo formulę. Rezultatai taip pat rodo, jog dokumentų aprėpties (Coverage) įvertis visose tirtose modifikacijose yra gana aukštas, kas ir yra būdinga turiniu paremtiems algoritmams. Kadangi visais trejais atvejais buvo naudotas tas pats artimiausių kaimynų klasifikavimo metodas, tik skirtingos jo modifikacijos, gauti rezultatai irgi yra labai panašūs, o matomi rezultatų skirtumai - minimalūs.

2.3.2. Bendruoju filtravimu paremti rekomendavimo algoritmai

Siekiant pamatuoti bendruoju filtravimų paremtų algoritmų efektyvumą, tirti duomenys buvo įvertinti 12 skirtingų algoritmų. Gauti palyginimų duomenys yra pateikiami 6 ir 7 lentelėse.

Kaip ir turiniu paremtų rekomendavimo algoritmų vertinime, rezultatai buvo maksimizuojami pagal preciziškumų vidurkio įvertį. Iš gautų rezultatų matyti, kad rezultatai tarpusavyje yra gana panašūs, bet aukščiausi daugelio įverčių rezultatai buvo gauti taikant IALS algoritmą. Tačiau šis algoritmas užtruko daugiausiai mokymosi laiko bei turėjo vieną žemiausių dokumentų aprėpties (Coverage) įvertį. Žema šio įverčio reikšmė parodo, kad duomenų aibėje yra daug naujų dokumentų, kurie negali būti rekomenduojami naudotojams dėl per žemų įvertinimų arba apskritai įvertinimų nebuvimo. Taigi, šis rodiklis įspėja apie galimas tuščių įvertinimų arba naujų objektų problemas.

Gauti rezultatai taip pat pagrindžia teoriniame skirsnyje (1.5.8) minėtą prielaidą, jog dokumentų aprėpties rodiklis paprastai yra mažesnis bendruoju filtravimu paremtuose algoritmuose.

6 lentelė. Bendruoju filtravimu paremtų rekomendavimo algoritmų efektyvumo palyginimas (1)

Algoritmas / Metrika	EASE R	ItemKNN	P3alpha	RP3beta	SLIM BPR	SLIM Elastic
P	0.015104	0.013177	0.015919	0.015360	0.015481	0.014969
R	0.235491	0.202333	0.248501	0.241937	0.242427	0.233261
F	0.028387	0.024743	0.029921	0.028886	0.029103	0.028133
MAP	0.081455	0.060199	0.086630	0.083847	0.085170	0.081577
NDCG	0.118554	0.093080	0.124619	0.122184	0.121919	0.118266
AUC	0.177161	0.145781	0.185784	0.182310	0.184392	0.177239
Novelty	0.015729	0.019570	0.015115	0.015572	0.015390	0.015712
Coverage	0.102108	0.138516	0.105393	0.113879	0.110047	0.097454
Mokymo laikas, s	4.89	0.23	0.48	0.51	137.98	119.51
Testavimo laikas, s	9.85	8.98	7.89	8.90	9.31	9.68

Žymėjimai:	Blogiausias rezultatas	Geriausias rezultatas

7 lentelė. Bendruoju filtravimu paremtų rekomendavimo algoritmų efektyvumo palyginimas (2)

Algoritmas / Metrika	IALS	Matrix Factorization Asy SVD	Matrix Factorization BPR	Matrix Factorization Funk SVD	NMF	Pure SVD
P	0.028597	0.027540	0.019287	0.024138	0.009580	0.011473
R	0.499461	0.485320	0.336093	0.429558	0.149366	0.178418
F	0.054097	0.052122	0.036481	0.045707	0.018005	0.021559
MAP	0.167973	0.166885	0.126172	0.143448	0.012968	0.050758
NDCG	0.243817	0.239038	0.174414	0.210028	0.041452	0.079844
AUC	0.356097	0.350224	0.258441	0.310251	0.063412	0.127950
Novelty	0.031032	0.035076	0.035008	0.033820	0.023733	0.016311
Coverage	0.014782	0.111963	0.014235	0.710375	0.013961	0.043800
Mokymo laikas, s	1327.11	230.28	167.08	381.88	110.12	4.15
Testavimo laikas, s	4.41	9.15	11.40	7.85	9.02	8.12

Žymėjimai:	Blogiausias rezultatas	Geriausias rezultatas

Apskritai bendruoju filtravimu paremti rekomendavimo algoritmai pasiekė ženkliai geresnius efektyvumo rezultatus tirtiems duomenims. Iš gautų rezultatų matyti, kad net prasčiausiai veikęs bendruoju filtravimu paremtas NMF algoritmas sugeneravo ženkliai geresnius rezultatus už bet kurią turiniu paremtų rekomendavimo algoritmų variaciją. Geriausiai veikusio IALS algoritmo maksimizuota preciziškumų vidurkio reikšmė (0.167973) buvo ženkliai didesnė už geriausią turiniu paremto rekomendavimo algoritmo preciziškumų vidurkio reikšmę (0.004001). Daugelis kitų rodiklių yra taip pat žymiai aukštesni bendruoju filtravimu paremtuose algoritmuose, tačiau turiniu paremti rekomendavimo algoritmai pasiekė ženkliai aukštesnius dokumentų aprėpties rezultatus. Taigi, turiniu paremti rekomendavimo algoritmai, nors ir naudojami atskirai tirtiems duomenims

negalėjo pateikti tikslių rekomendacijų, tikėtina, kad gali padėti išspręsti tuščių įvertinimų arba naujų objektų problemas, jeigu bus kombinuojami su bendruoju filtravimu paremtais rekomendavimo algoritmais.

2.3.3. Hibridiniai rekomendavimo algoritmai

Bandant įvertinti hibridinių rekomendavimo algoritmų efektyvumą tirtiems duomenims, buvo išbandyti 6 algoritmai. 2 iš jų buvo nagrinėti kitų tyrinėtojų - LightFM [Kul15] ir CFeCBF [DFC+19]. Pirmasis tirtiems duomenims veikė tiksliausiai, antrasis - mažiausiai tiksliai. Svarbu paminėti, kad geriausiai veikusio LightFM algoritmo preciziškumų vidurkio rodiklis yra labai panašus, kaip ir geriausiai veikusių bendruoju filtravimu paremtų rekomendavimo algoritmų (IALS, Matrix Factorization Asy SVD), tačiau LightFM algoritmo mokymo laikas tirtiems duomenims buvo atitinkamai 23 ir 4 kartus trumpesnis. Visi gauti hibridinių rekomendavimo algoritmų efektyvumo palyginimo rodikliai pateikiami 8 lentelėje.

8 lentelė. Hibridinių rekomendavimo algoritmų efektyvumo palyginimas

Algoritmas / Metrika	LightFM	CFeCBF	ItemKNN CFCBF	ItemKNN CFCBF*w	ItemKNNCF + P3alpha	PureSVD + ItemKNNCF
P	0.028200	0.000876	0.015360	0.015818	0.015825	0.013312
R	0.492986	0.014459	0.239508	0.247648	0.247782	0.205764
F	0.053348	0.001652	0.028868	0.029736	0.029749	0.025006
MAP	0.171130	0.004270	0.082961	0.085351	0.085350	0.061465
NDCG	0.244684	0.006567	0.120165	0.123587	0.123608	0.094548
AUC	0.363485	0.010658	0.180675	0.185762	0.185701	0.147115
Novelty	0.031268	0.010038	0.014732	0.014967	0.014965	0.019621
Coverage	0.012319	0.566658	0.504791	0.099370	0.099370	0.141254
Mokymo laikas, s	56.14	56.64	0.16	0.05	0.05	0.00
Testavimo laikas, s	15.70	4.18	4.28	4.28	4.35	4.62

Žymėjimai:	Blogiausias rezultatas	Geriausias rezultatas

Bendrai rezultatai taip pat rodo, kad kombinuojant turiniu ir bendruoju filtravimu paremtus algoritmus tarpusavyje yra gaunami daugeliu atveju geresni rezultatai. Pavyzdžiui, kombinuojant turiniu paremto rekomendavimo algoritmo, naudojančio artimiausių kaimynų klasifikavimą (ItemKNN), su bendruoju filtravimu paremto rekomendavimo algoritmo, naudojančio tą patį klasifikavimo metodą (CFCBF), buvo gauta 0.082961 vidutinio preciziškumo reikšmė. Taikant šiuos algoritmus atskirai, buvo gautos atitinkamai 0.004001 ir 0.060199 reikšmės. Taip pat matoma, kad taikant hibridinį metodą yra pasiekiamas ir geresnis dokumentų aprėpties rodiklis, t. y. yra daugeliu atveju išsprendžiamos tuščių įvertinimų ir naujų naudotojų problemos, kurios atsiranda taikant bendruoju filtravimu paremtą rekomendavimo algoritmą atskirai.

Taikant tą patį hibridinį algoritmą buvo ištirtas ir svoriais paremtas metodas (ItemKNNCFCBF*w). Algoritmui buvo apibrėžta taisyklė, nurodanti, kuris algoritmas iš dviejų sudedamųjų daro didesnę įtaką rekomendacijoms. Tačiau, kaip matoma iš gautų rezultatų,

svorio įvedimas ženkliai geresnių rezultatų sugeneruoti nepadėjo. Hibridinis metodas be papildomų svorių veikė panašiu efektyvumu ir generavo panašaus tikslumo rekomendacijas. Be to, ženkliai sumažėjo dokumentų aprėpties įvertis, signalizuojantis apie tuščių įvertinimų bei naujų objektų problemas. Kaip ir buvo minėta teoriniame svoriais paremtų metodų aprašyme (1.3.1) bei pagrįsta kitų tyrinėtojų darbuose [SDT17], rekomendacijų rezultatai, gauti taikant svoriais paremtus hibridinius metodus labai priklauso nuo tiriamos duomenų aibės ir ne visais atvejais būna geresni, nei taikant sudedamuosius algoritmus atskirai.

Iš gautų rezultatų taip pat matyti, kad algoritmų kombinavimas suvidurkino PureSVD + ItemKNNCF ir ItemKNNCF + P3alpha algoritmų rezultatus. Abiem atvejais, pritaikius pakopinį metodą, nurodyta, kad norima taikyti pirmąjį algoritmą, tačiau nesugeneravus patikimų rekomendacijų, algoritmas turi persijungti ir rekomenduoti dokumentus remiantis antruoju algoritmu. ItemKNNCF + P3alpha kombinacijos atveju buvo gauti nežymiai mažesni rekomendacijų tikslumo rodikliai (lyginant su P3alpha taikymu atskirai), o naujumo bei dokumentų aprėpties rodikliai išliko panašūs. PureSVD + ItemKNNCF kombinacijos atveju buvo gauti nežymiai geresni tiek rekomendacijų tikslumo, tiek naujumo bei dokumentų aprėpties rodikliai (lyginant su sudedamųjų algoritmų taikymu atskirai).

2.3.4. Kiti rekomendavimo algoritmai

Atliekant naudojamų tekstų rekomendavimo algoritmų efektyvumo vertinimą, taip pat buvo ištirti ir 3 kiti algoritmai, kurie pagal savo veikimo principą negali būti priskirti nei turiniu, nei bendruoju filtravimu paremtiems, nei hibridiniams metodams. Gauti kitų rekomendavimo algoritmų efektyvumo rezultatai yra pateikiami 9 lentelėje.

9 lentelė. Kitų rekomendavimo algoritmų efektyvumo palyginimas

Algoritmas / Metrika	GlobalEffects	MostPopular	Random
P	0.000020	0.029062	0.000317
R	0.000404	0.513183	0.005412
F	0.000038	0.055009	0.000598
MAP	0.000178	0.173315	0.001017
NDCG	0.000227	0.25063	0.0020241
AUC	0.000291	0.372339	0.003375
Novelty	0.081459	0.030688	0.019667
Coverage	0.005749	0.008486	1.000000
Mokymo laikas, s	0.02	0.02	0.02
Testavimo laikas, s	3.60	8.53	8.46

Žymėjimai:	Blogiausias rezultatas	Geriausias rezultatas

Kaip matyti iš gautų rezultatų, prasčiausias rekomendacijas tirtiems duomenims teikė globalių efektų rekomendavimo algoritmas. Taikant šį algoritmą buvo gauti žemiausi tikslumo rodikliai, lyginant su bet kuriuo kitu tirtu algoritmu, tačiau pasiektas aukščiausias naujumo įvertis, kas parodo, jog algoritmas rekomenduoja mažiau žinomus dokumentus, taigi naudotojui yra pateikiama

įvairesnių rekomendacijų (tekstų).

Gauti rezultatai taip pat parodė, kad tirtiems duomenims populiariausių tekstų rekomendavimas veikė beveik vienodai tiksliai, kaip ir geriausius rezultatus generavęs hibridinis (LightFM) algoritmas. Tačiau, nors ir populiariausių dokumentų rekomendavimas sudomina daugelį skaitytojų, šis algoritmas pasižymi gana mažu dokumentų aprėpties įverčiu.

Tirtas atsitiktinių rekomendacijų generavimo algoritmas, taip pat, kaip ir globalių efektų algoritmas, pasižymėjo labai žemais tikslumo rodikliais. Šis algoritmas iš kitų išsiskiria tuo, jog pasiekia maksimalų dokumentų aprėpties įvertį, kas signalizuoja apie tuščių įvertinimų bei naujų objektų problemų nebuvimą generuojamose rekomendacijose.

3. Naujo algoritmo realizavimas

Naujo algoritmo realizavimas susideda iš kelių pagrindinių dalių, aprašytų toliau esančiuose poskyriuose. 3.1 poskyryje paaiškinta tirta duomenų aibė bei duomenų paruošimas, o 3.2 poskyryje - algoritmo realizavimo eiga. Galiausiai, 3.3 poskyryje pateiktas algoritmo efektyvumo vertinimas, palyginantis naujo algoritmo rezultatus su 2 skyriuje gautais tirtiems duomenims geriausių rekomendavimo algoritmų rezultatais.

3.1. Duomenų aibė ir duomenų paruošimas

Naujo algoritmo realizavimui ir jo vertinimui buvo remtasi tais pačiais dokumentais, kaip ir naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrime. Duomenų aibę iš viso sudarė 3653 tekstiniai dokumentai bei 86609 nuasmenintų naudotojų įrašai, parodantys, kiek sekundžių vienas naudotojas domėjosi konkrečiu dokumentu.

Kadangi naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimas, paaiškintas 2 skyriuje, parodė, kad turimų duomenų raktažodžių išrinkimui gali būti naudojami TD-IDF arba Doc2Vec algoritmai, pasirinktas vienas iš jų - TD-IDF. Pritaikius šį algoritmą visiems turimiems tekstams buvo nustatyti pagrindiniai raktažodžiai, geriausiai apibūdinantys analizuojamą tekstą. Kaip ir naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimo atveju, prieš raktažodžių išrinkimą buvo atliktas duomenų paruošimas - tekstai buvo apdoroti juos struktūrizuojant, pašalinant bereikšmius žodžius bei visus gautus tekstų duomenis lemuojant.

Turimi skaitiniai duomenys taip pat turėjo būti atskirai apdoroti. Kadangi duomenys buvo saugomi sekundėmis, o skaitomų tekstų ilgiai yra skirtingi, šiuos skaičius reikėjo sureitinguoti, t. y. nustatyti, kokią vidutiniškai teksto dalį naudotojas perskaitė, tokiu būdu priskiriant naudotojo įvertį konkrečiam tekstui. Reitinguojant skaitymo laiką buvo remtasi, jog vidutinis skaitytojo skaitymo greitis galėtų būti apie 250 žodžių per sekundę. Kadangi greitis taip pat gali priklausyti nuo skaitytojo amžiaus bei įgūdžių [Mur20], nuspręsta laikyti, kad per sekundę naudotojas turėtų perskaityti maždaug 4 žodžius. Gauta reikšmė buvo dalinama iš visame tekste esančių žodžių skaičiaus, o gautas rezultatas laikomas naudotojo nesąmoningai paliktu įverčiu skaitytam tekstui.

3.2. Realizavimo eiga

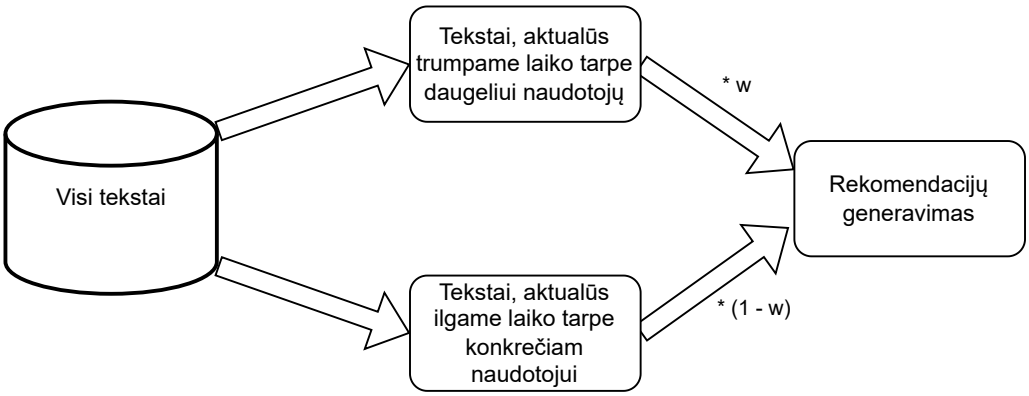
Paruošti, taip kaip aprašyti 3.1 poskyryje, duomenys, sudarė įvestį kuriamam algoritmui. Algoritmo apmokymui bei vertinimui visi duomenys buvo padalinti į 3 poaibius - apmokymo, testavimo ir validacijos, kur testavimo aibė sudarė 20% visos duomenų aibės, o iš likusios dalies 80% skirta apmokymo ir 20% validacijos aibėms.

Duomenų skaidymui į poaibius bei egzistuojančių algoritmų pritaikymui buvo panaudotas M. F. Dacrema [DCJ19a] tyrime pateiktas rekomendavimo algoritmų išeities kodas [DCJ19b]. To paties kodo pagalba visiems tirtiems algoritmams bei jų modifikacijoms atlikta optimalių parametrų paieška, padėjusi nustatyti parametrus, su kuriais algoritmas veikia tiksliausiai (t. y. pasiekia aukščiausią preciziškumų vidurkio reikšmę).

Naudojamų tekstų rekomendavimo algoritmų efektyvumo vertinimas, aprašytas 2 skyriuje, parodė, jog vienas efektyviausių algoritmų, generuojantis patikimas rekomendacijas, yra MostPopular algoritmas, rekomenduojantis skaitomiausius dokumentus. Šio algoritmo pliusas kuriamo metodo atžvilgiu yra toks, kad norint teikti rekomendacijas bet kuriam naudotojui, nėra būtina žinoti kuo konkretus naudotojas domisi, o algoritmo veikimui nėra reikalinga naudotojo įvestis. Tam, kad rekomendacijos būtų teikiamos atsižvelgiant į turinio aktualumą laiko prasme, kuriamam algoritmui nustatyta laiko riba (3 dienos), kuri užtikrina, kad šiuo algoritmu į rekomendacijas nebus įtraukti seniai parašyti tekstai. Taikant tokį algoritmą bus užtikrinama, jog mažai pažįstami naudotojai be jokios įvesties galės gauti 3 dienų populiariausias algoritmo atrinktas rekomendacijas.

Tam, kad algoritmas gebėtų teikti ir užtikrintesnes rekomendacijas, kurios būtų tikslesnės tais atvejais, kai algoritmui yra žinoma, kokiais straipsniais konkretus naudotojas yra linkęs domėtis, nutarta prijungti turiniu paremtą metodą. Tokiu būdu yra įgyvendinama dar viena kuriamam algoritmui iškelta veikimo sąlyga, teigianti, jog į rekomendacijas turi būti įtraukiami ir nauji tekstai, apie kurių aktualumą nėra įmanoma nuspręsti iš kitų naudotojų, priklausančių tai pačiai naudotojų grupei. Įtraukus šį metodą rekomendacijos bus teikiamos atsižvelgiant į panašumus tarp tokių tekstų, kokiais konkretus naudotojas domisi. Taigi, naudojant šį metodą, kuriamas algoritmas neturėtų susidurti su naujų objektų problema, nes bet koks naujas užregistruotas tekstas bus rekomenduojamas naudotojams atsižvelgiant į jo turinį, o ne į kitų naudotojų tekstui paliktus įvertinimus.

Galiausiai, kaip pastebėta literatūros apžvalgoje bei patvirtinta atlikus naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimą, turiniu paremti metodai generuoja geresnius rezultatus, kai yra kombinuojami su bendruoju filtravimu paremtais metodais. Dėl šios priežasties nuspręsta atsižvelgti ir į panašių naudotojų skaitomus tekstus. Taigi, remtis hibridinio algoritmo rezultatais, o ne vien turiniu ar bendruoju filtravimu paremtu algoritmu. Be to, taikant hibridinį metodą yra sumažinama ir tuščių įvertinimų problemos tikimybė - esant tuščiai naudotojų-objektų matricai ir algoritmui nesugebant rasti kitų panašių naudotojų, kurių rezultatais remiantis būtų galima teikti naujas rekomendacijas, algoritmas tiesiog remsis turinių paremto metodo rezultatais. 5 paveikslėlyje yra pavaizduojamas kuriamo algoritmo veikimo principas.



5 pav. Kuriamo algoritmo veikimo principas

Iš to matyti, kad algoritmas rekomendacijų generavimui turėtų remtis dviejų išeičių duomenimis. Pirma - tekstais, kurie yra aktualūs trumpame laiko tarpe daugeliui naudotojų (t. y. MostPopular algoritmu), antra - tekstais, kurie yra aktualūs ilgame laiko tarpe konkrečiam naudotojui

(t. y. ItemKNNCFCBF algoritmu). Daugeliu atveju antrasis algoritmas apims naudotojo hobius ar tiesiog kitas pamėgtas temas, kuriomis naudotojas domisi nuolatos. Tokiu būdu į rekomenduojamų duomenų aibę visai nepateks arba pateks tik labai nedidelė dalis tekstų, kurie naudotojui būtų neaktualūs rekomenduojamu metu. Taigi, kuriamas algoritmas yra mišriu metodu veikiantis hibridinis algoritmas, gebantis kaip įvestį priimti hibridinį ItemKNNCFCBF ir MostPopular algoritmus bei leidžiantis nustatyti šių algoritmų svorius. Žemiau yra pateikiamas šio algoritmo pseudokodas.

1 algoritmas. Mišrus, svoriais paremtas algoritmas

Įvestis: $rec\_1, rec\_2, w, users$
Išvestis: $item\_weights$
1: $item\_weights\_1 = rec\_1.compute\_item\_score(users)$
2: $item\_weights\_2 = rec\_2.compute\_item\_score(users)$
3: $item\_weights = item\_weights\_1 * w + item\_weights\_2 * (1 - w)$

Algoritmas apskritai užtikrina, kad rekomendacijų įverčių apskaičiavimui galėtų būti nurodomi bet kokie 2, vienas nuo kito nepriklausantys, rekomendavimo algoritmai. Algoritmo funkcija taip pat per parametrus turėtų gauti svorio įvertį, kurio reikšmė turėtų būti skaičius nuo 0 iki 1, ir kuris nustatytų pirmojo algoritmo svorį. Galiausiai, algoritmo funkcija turėtų gauti visų naudotojų, kurių rekomendacijų reikšmes norima apskaičiuoti, aibę.

Kiekvienas iš perduodamų rekomendavimo algoritmų taip pat turi savo apibrėžtą rekomendacijų generavimo ir rezultatų skaičiavimo funkciją (*compute_item_score*), kuri veikia nepriklausomai nuo kito perduodamo algoritmo:

- MostPopular algoritmo atveju ši funkcija atrinks populiariausius objektus, kuriais vėliau bus užpildomas kiekvieno naudotojo rekomendacijų profilis aibėje *users*.
- ItemKNNCFCBF algoritmo atveju ši funkcija remsis dvejais kitais algoritmais - bendruoju filtravimu ir turiniu paremtais metodais, kur panašumai tarp objektų bus apskaičiuojami taikant k-artimiausių kaimynų metodą. Šis algoritmas užtikrins, kad naudotojui bus pateiktos tinkamiausios rekomendacijos, atsižvelgiant į naudotojo iki tol skaitytus panašius tekstus, o jei tokių rekomendacijų sugeneruoti neįmanoma - į panašių naudotojų skaitomus tekstus.

3.3. Algoritmo efektyvumo vertinimas

Sukurto algoritmo vertinimui buvo apskaičiuoti tie patys įvertinimo matai, kaip ir naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrime. Visi gauti įverčiai yra palyginami 10 lentelėje.

Iš gautų rezultatų matyti, kad naujasis algoritmas turimiems duomenims teikia tikslesnes rekomendacijas, nei bet kuris kitas tirtas algoritmas. Visi tikslumo matai taikant naująjį algoritmą yra aukščiausi, nors kai kurie skiriasi ir labai nežymiai. Atkreiptinas dėmesys, jog naujojo algoritmo naujumo įvertis yra ženkliai aukštesnis, nei bet kurio kito tirto algoritmo. Žema šio įverčio reikšmė gali signalizuoti tai, kad algoritmas neįtraukia naujų dokumentų, apie kuriuos naudotojas

10 lentelė. Tirtiems duomenims geriausių rekomendavimo algoritmų efektyvumo palyginimas

Algoritmas / Metrika	ItemKNN cosine (CB)	P3alpha	IALS	LightFM	MostPopular	Naujas
P	0.000829	0.015919	0.028597	0.028200	0.029062	0.033450
R	0.013336	0.248501	0.499461	0.492986	0.513183	0.591997
F	0.001560	0.029921	0.054097	0.053348	0.055009	0.063323
MAP	0.004001	0.086630	0.167973	0.171130	0.173315	0.200378
NDCG	0.006066	0.124619	0.243817	0.244684	0.250631	0.289070
AUC	0.009703	0.185784	0.356097	0.363485	0.372339	0.424156
Novelty	0.010116	0.015115	0.031032	0.031268	0.030688	0.473086
Coverage	0.753353	0.105393	0.014782	0.012319	0.008486	0.144104
Mokymo laikas, s	0.18	0.48	1327.11	56.14	0.02	0.04
Testavimo laikas, s	4.52	7.89	4.41	15.70	8.53	3.04

Žymėjimai: Blogiausias rezultatas | Geriausias rezultatas

iki tol nežinojo, tačiau galbūt būtų susidomėjęs. Nors, kaip paminėta teoriniame naujumo aprašyme, pateiktame 1.5.7 skirsnyje, šio įverčio nėra prasmės nei maksimizuoti, nei minimizuoti, labai aukštas šio įverčio rezultatas parodo, kad algoritmas rekomenduoja mažiau žinomus dokumentus.

Taip pat, aukštas naujumo įvertis, kartu su vidutiniu (0.144104) dokumentų aprėpties įverčiu parodo, kad mažai tikėtina, jog algoritmas susiduria su panašių objektų rekomendavimo arba mažų įverčių apimties problemomis. Kaip ir minėta, aukštas naujumo įvertis parodo, kad algoritmas rekomenduoja mažiau žinomus dokumentus, o šiuo atveju vidutinė dokumentų aprėpties reikšmė užtikrina, kad rekomendacijos teikiamos su pakankama dokumentų įverčių apimtimi. Vidutinė dokumentų aprėpties reikšmė taip pat signalizuoja apie tuščių įvertinimų problemos nebuvimą. Manytina, kad taip yra dėl to, nes į rekomendacijas yra įtraukiami turiniu paremti metodai, kurie pasižymi itin aukštu dokumentų aprėpties rodikliu.

Be to, sukurtame algoritme apjungtas populiariausių tekstų rekomendavimo algoritmas užtikrina, kad net ir nauji naudotojai, kurie nėra įvertinę (skaitę) pakankamai objektų (tekstų), galės gauti patikimas rekomendacijas. Pakankamai aukšti tikslumo rezultatai patvirtina, kad naujasis algoritmas nesusiduria su naujų naudotojų problema. Naujo algoritmo mokymo ir testavimo laikai taip pat yra labai maži, o apjungti algoritmai nėra sudėtingi ar reikalaujantys daug skaičiavimo resursų, taigi tikėtina, kad algoritmas veiktų sklandžiai ir taikant rekomendacijas realiu metu.

Iš gautų palyginimo rezultatų taip pat matyti, jog prasčiausiai veikia turiniu ir bendruoju filtravimu paremti rekomendavimo algoritmai, kai jie yra taikomi atskirai. Populiariausių objektų rekomendavimo algoritmas generuoja panašius tikslumo rezultatus, kaip ir hibridinis LightFM algoritmas, o šių abiejų algoritmų rezultatai yra panašūs ir naujam sukurtam algoritmui.

Rezultatai ir išvados

Šiame darbe buvo siekiama sukurti naują tekstų rekomendavimo algoritmą, tinkamą taikyti lietuvių kalba parašytų tekstų rekomendacijų gavimui, kai naudotojai yra mažai pažįstami. Šiam tikslui įgyvendinti buvo:

- Atlikta analitinė jau egzistuojančių tekstų rekomendavimo algoritmų veikimo principų apžvalga.
- Turimi duomenys buvo apdoroti spaCy bibliotekoje esančiais įrankiais, pritaikant skyrybos ženklų, bereikšmių žodžių šalinimą bei žodžių lemavimą.
- Gautiems duomenims išskirti raktažodžiai naudojant spaCy, TF-IDF ir Doc2Vec algoritmus.
- Išskirtiems raktažodžiams pritaikytas temų modeliavimo principas, o gautus rezultatus įvertinus LSA, LDA ir Doc2Vec metodais nustatyta, kad tirtiems duomenims:
  - Raktažodžių išskyrimui TF-IDF ir Doc2Vec algoritmai veikia tiksliau, nei spaCy.
  - Doc2Vec algoritmas temos modeliavimui buvo mažiausiai tinkamas dėl nei karto negautos mažiausios Hammino paklaidos reikšmės visais tirtais atvejais.
- Atliktas naudojamų tekstų rekomendavimo algoritmų efektyvumo tyrimas, kurio įvestį sudarė TD-IDF metodu išskirti tekstų raktažodžiai. Atlikto tyrimo rezultatai parodė, kad:
  - Turiniu paremti rekomendavimo algoritmai veikė mažiausiai tiksliai tirtiems duomenims, tačiau turėjo aukščiausius dokumentų aprėpties įverčius.
  - Kai kurie bendruoju filtravimu paremti rekomendavimo algoritmai sugeneruoja netgi geresnius arba labai panašius rezultatus, kaip ir hibridiniai rekomendavimo algoritmai.
  - Pats rekomendavimo algoritmų apjungimas, t. y. hibridinių metodų kūrimas dažniausiai gali padėti suvidurkinti kombinuojamų algoritmų rezultatus, taip išsprendžiant, pavyzdžiui, tuščių įvertinimų ir naujų naudotojų problemas.

Šio darbo rezultatas - sukurtas mišrus, svoriais paremtas rekomendavimo algoritmas, naudojantis ItemKNNCFCBF bei MostPopular algoritmų kombinaciją. Atliktas algoritmo efektyvumo vertinimas parodė, jog algoritmas tirtiems duomenims veikia tiksliau už kitus tirtus tekstų rekomendavimo algoritmus. Gauti rezultatai buvo įvertinti apskaičiuojant ir lyginant preciziškumo (P), jautrumo (R), harmoninio preciziškumo ir jautrumo vidurkio (F), preciziškumų vidurkio (MAP), normalizuoto diskontuoto suminio naudingumo (NDCG), ploto po ROC kreive (AUC), naujumo ir aprėpties reikšmes.

Literatūra

[Agg16] Charu C. Aggarwal. *Content-Based Recommender Systems*. *Recommender Systems: The Textbook*. Springer International Publishing, Cham, 2016, p.p. 139–166. ISBN: 978-3-319-29659-3. DOI: 10.1007/978-3-319-29659-3_4. URL: https://doi.org/10.1007/978-3-319-29659-3_4.

[AOA+19] A. Ayodele Adebiyi, Olawole Ogunleye, O. Marion Adebiyi ir Olatunji J. Okesola. A Comparative Analysis of TF-IDF, LSI and LDA in Semantic Information Retrieval Approach for Paper-Reviewer Assignment. *Journal of Engineering and Applied Sciences*, 14(10):3378–3382, 2019. URL: https://ir.tech-u.edu.ng/433/.

[BWL+19] X. Bai, M. Wang, I. Lee, Z. Yang, X. Kong ir F. Xia. Scientific Paper Recommendation: A Survey. *IEEE Access*, 7:9324–9339, 2019.

[Çan17] Erion Çano. Hybrid Recommender Systems: A Systematic Literature Review. *Intelligent Data Analysis*, 21:1487–1524, 2017-11. DOI: 10.3233/IDA-163209.

[DCJ19a] Maurizio Ferrari Dacrema, Paolo Cremonesi ir Dietmar Jannach. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. Toine Bogers, Alan Said, Peter Brusilovsky ir Domonkos Tikk, redaktoriai, *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*, p.p. 101–109. ACM, 2019. DOI: 10.1145/3298689.3347058. URL: https://doi.org/10.1145/3298689.3347058.

[DCJ19b] Maurizio Ferrari Dacrema, Paolo Cremonesi ir Dietmar Jannach. DeepLearning RS Evaluation. https://github.com/MaurizioFD/RecSys2019_DeepLearning_Evaluation/, 2019.

[DFC+19] Yashar Deldjoo, Maurizio Ferrari Dacrema, Mihai Gabriel Constantin, Hamid Eghbal-zadeh, Stefano Cereda, Markus Schedl, Bogdan Ionescu ir Paolo Cremonesi. Movie genome: alleviating new item cold start in movie recommendation. *User Modeling and User-Adapted Interaction*, 2019-02. ISSN: 1573-1391. DOI: 10.1007/s11257-019-09221-y. URL: https://doi.org/10.1007/s11257-019-09221-y. Source: https://github.com/MaurizioFD/CFeCBF.

[Dzi19] Robert Dzisevič. Trumpo teksto klasifikacija naudojant neuroninius tinklus. lit, Vilnius, 2019.

[FFM10] Blaž Fortuna, Carolina Fortuna ir Dunja Mladenić. Real-Time News Recommender System. Tom. 6323, p.p. 583–586, 2010-08. DOI: 10.1007/978-3-642-15939-8_38.

[HMV+20] Matthew Honnibal, Ines Montani, Sofie Van Landeghem ir Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python, 2020. DOI: 10.5281/zenodo.1212303. URL: https://doi.org/10.5281/zenodo.1212303.

[KB10] Michal Kompan ir Maria Bielikova. Content-Based News Recommendation. Tom. 61, p.p. 61–72, 2010-09. DOI: 10.1007/978-3-642-15208-5_6.

[KKG+18] Dhruv Khattar, Vaibhav Kumar, Manish Gupta ir Vasudeva Varma. Neural Content-Collaborative Filtering for News Recommendation. *NewsIR@ECIR*, 2018.

[KS16] Balraj Kumar ir Neeraj Sharma. Approaches, Issues and Challenges in Recommender Systems: A Systematic Review. *Indian Journal of Science and Technology*, 9, 2016-12. DOI: 10.17485/ijst/2015/v8i1/94892.

[Kul15] Maciej Kula. Metadata Embeddings for User and Item Cold-start Recommendations. Toine Bogers ir Marijn Koolen, redaktoriai, *Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 16-20, 2015*. Tom. 1448 *CEUR Workshop Proceedings*, p.p. 14–21. CEUR-WS.org, 2015. URL: http://ceur-ws.org/Vol-1448/paper4.pdf.

[LL19] Xiaofeng Li ir Dong Li. An Improved Collaborative Filtering Recommendation Algorithm and Recommendation Strategy. *Mobile Information Systems*, 2019, 2019.

[LM12] X. Li ir T. Murata. Multidimensional clustering based collaborative filtering approach for diversified recommendation. *2012 7th International Conference on Computer Science Education (ICCSE)*, p.p. 905–910, 2012.

[LM14] Quoc Le ir Tomas Mikolov. Distributed Representations of Sentences and Documents. Eric P. Xing ir Tony Jebara, redaktoriai, *Proceedings of the 31st International Conference on Machine Learning*, tom. 32 numeris 2 *Proceedings of Machine Learning Research*, p.p. 1188–1196, Bejing, China. PMLR, 2014-22–24 Jun. URL: http://proceedings.mlr.press/v32/le14.html.

[MCG+99] Tim Miranda, Mark Claypool, Anuja Gokhale, Tim Mir, Pavel Murnikov, Dmitry Netes ir Matthew Sartin. Combining content-based and collaborative filters in an online newspaper. *In Proceedings of ACM SIGIR Workshop on Recommender Systems*. Citeseer, 1999.

[Mur20] Mantas Murauskas. Teksto skaitymo metodų tyrimas. lit, Kaunas, 2020.

[PSA14] Simon Philip, Peter Shola ir Ovye Abari. Application of Content-Based Approach in Research Paper Recommendation System for a Digital Library. *International Journal of Advanced Computer Science and Applications*, 5, 2014-10. DOI: 10.14569/IJACSA.2014.051006.

[Rap15] Aurimas Rapečka. *Rekomendacinių sistemų socialiniuose tinkluose efektyvumo didinimas*. Disertacija, Vilnius University, 2015.

[Ras14] Sebastian Raschka. Linear Discriminant Analysis bit by bit, 2014-08. DOI: 10.13140/2.1.3196.6084.

[RMD13] Aurimas Rapečka, Virginijus Marcinkevičius ir Gintautas Dzemyda. Rekomendacinės sistemos algoritmų veikimo elektroninio knygyno duomenų bazėje analizė. *Informacijos mokslai*, 65:45–55, 2013-saus. DOI: 10.15388/Im.2013.0.2056. URL: https://www.zurnalai.vu.lt/informacijos-mokslai/article/view/2056.

[RRP+16] Jesse Read, Peter Reutemann, Bernhard Pfahringer ir Geoff Holmes. MEKA: A Multi-label/Multi-target Extension to WEKA. *Journal of Machine Learning Research*, 17(21):1–5, 2016. URL: http://jmlr.org/papers/v17/12-164.html.

[ŘS10] Radim Řehůřek ir Petr Sojka. Software Framework for Topic Modelling with Large Corpora. English. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, p.p. 45–50, Valletta, Malta. ELRA, 2010-05. http://is.muni.cz/publication/884893/en.

[SDT17] S. Suriati, Meisyarah Dwiastuti ir Tulus Tulus. Weighted hybrid technique for recommender system. *Journal of Physics: Conference Series*, 930:012050, 2017-12. DOI: 10.1088/1742-6596/930/1/012050.

[SEH16] Yang Song, Ali Elkahky ir Xiaodong He. Multi-Rate Deep Learning for Temporal Recommendation. P.p. 909–912, 2016-07. DOI: 10.1145/2911451.2914726.

[SGM17] Ritu Sharma, Dinesh Gopalani ir Yogesh Meena. Concept-Based Approach for Research Paper Recommendation. *PReMI*, 2017.

[Sko12] Lucie Skorkovská. Application of Lemmatization and Summarization Methods in Topic Identification Module for Large Scale Language Modeling Data Filtering. Tom. 7499, 2012-09. ISBN: 978-3-642-32789-6. DOI: 10.1007/978-3-642-32790-2_23.

[SML+14] J. Sun, J. Ma, Z. Liu ir Y. Miao. Leveraging Content and Connections for Scientific Article Recommendation in Social Computing Contexts. *The Computer Journal*, 57(9):1331–1342, 2014.

[SSZ+17] Jiangbo Shu, Xiaoxuan Shen, Xingchi Zhou, Baolin Yi ir Zhaoli Zhang. A content-based recommendation algorithm for learning resources. *Multimedia Systems*, 2017-03. DOI: 10.1007/s00530-017-0539-8.

[Sta19] Lukas Stankevičius. Lietuviškų naujienų grupavimas pasitelkiant dokumentų vektorizavimus. lit, Kaunas, 2019.

[TTS+16] Anastasios Tsolakidis, Evangelia Triperina, Cleo Sgouropoulou ir Nikos Christidis. Research Publication Recommendation System based on a Hybrid Approach. *PCI '16*, 2016.

[TVU+19] Priyank Thakkar, Krunal Varma, Vijay Ukani, Sapan Mankad ir Sudeep Tanwar. Combining User-Based and Item-Based Collaborative Filtering Using Machine Learning. Suresh Chandra Satapathy ir Amit Joshi, redaktoriai, *Information and Communication Technology for Intelligent Systems*, p.p. 173–180, Singapore. Springer Singapore, 2019. ISBN: 978-981-13-1747-7.

[VOC+19] P. Valdiviezo-Diaz, F. Ortega, E. Cobos ir R. Lara-Cabrera. A Collaborative Filtering Approach Based on Naïve Bayes Classifier. *IEEE Access*, 7:108581–108592, 2019.

[WWA+19] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang ir Xing Xie. Npa: Neural news recommendation with personalized attention. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, p.p. 2576–2584, 2019.