KAUNO TECHNOLOGIJOS UNIVERSITETAS
MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

Darius Aliulis

LATENTINIO DIRICHLĖ PASKIRSTYMO MODELIO TAIKYMAS INTERNETO REKLAMOS VARTOTOJŲ SEGMENTAVIMUI

Baigiamasis magistro projektas

Vadovas
doc. dr. Vytautas Janilionis

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS
MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

LATENTINIO DIRICHLĖ PASKIRSTYMO MODELIO TAIKYMAS INTERNETO REKLAMOS VARTOTOJŲ SEGMENTAVIMUI

Baigiamasis magistro projektas
Taikomoji matematika (kodas 621G10003)

Vadovas
doc. dr. Vytautas Janilionis

Recenzentas
dr. Tomas Ruzgas

Projektą atliko
Darius Aliulis

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS
MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS
Darius Aliulis
Taikomoji matematika (kodas 621G10003)

Baigiamojo projekto „Latentinio Dirichlė paskirstymo modelio taikymas interneto reklamos vartotojų segmentavimui“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

2015 m. birželio mėn. 5 d.
Kaunas

Patvirtinu, kad mano, **Darius Aliulis,** baigiamasis darbas tema „Latentinio Dirichlė paskirstymo modelio taikymas interneto reklamos vartotojų segmentavimui“ yra parašytas visiškai savarankiškai, o visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena darbo dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymu nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

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

_____ _____

(studento vardas ir pavardė, įrašyti ranka) (parašas)

TURINYS

Įžanga. 8

1. Literatūros apžvalga. 9
   - 1.1. Reklama internete. 9
   - 1.2. Interneto reklamos vartotojų segmentavimo uždavinys. 11
   - 1.3. Atviro kodo didžiųjų duomenų rinkinių analizės programinės įrangos lyginamoji analizė. 14
   - 1.4. Darbo tikslas ir uždaviniai. 22

2. Tyrimo metodai. 23
   - 2.1. Latentinis Dirichlė paskirstymo modelis interneto reklamos vartotojų segmentavimui. 23
   - 2.2. Latentinio Dirichlė paskirstymo modelio realizacijos aspektai. 28
   - 2.3. Segmentavimo kokybės įvertinimo metrikos. 29

3. Tyrimų rezultatai ir jų aptarimas. 33
   - 3.1. Interneto reklamos vartotojų segmentavimo uždavinio sprendimo metodika. . 33
   - 3.2. Metodikos ir programinių priemonių taikymas vartotojų segmentavimo uždavinio sprendimui. 33

Išvados. 49

Literatūros sąrašas. 50

Priedai. 55
   - 1 priedas. Tyrimų grafikai. 55
   - 2 priedas. Programų tekstai. 119
     - sparkevaluation/src/main/scala/SegmentEvaluation.scala. 119
     - btusersegmentation/bidmach_scripts/lda_crossvalidation.ssc. 126
     - for-use-in-spark-shell.scala. 129
     - sparsify-user-tokens-interactive.scala. 142
     - btusersegmentation/config.json. 143
     - btusersegmentation/job_params.json. 144
     - btusersegmentation/btlda.py. 149
     - btusersegmentation/core.py. 154
     - btusersegmentation/plotting/rscripts/clear_env.R. 158
     - btusersegmentation/plotting/rscripts/config_plot.R. 158
     - btusersegmentation/plotting/rscripts/config.R. 159
     - btusersegmentation/plotting/rscripts/data_info.R. 160
     - btusersegmentation/plotting/rscripts/fn_fname_helpers.R. 160
     - btusersegmentation/plotting/rscripts/fn_gather_helpers.R. 163

btusersegmentation/plotting/rscripts/fn_gather_tex_plots.R..... 166
btusersegmentation/plotting/rscripts/fn_plot_ads_hist2d.R..... 168
btusersegmentation/plotting/rscripts/fn_plot_aggrs.R..... 171
btusersegmentation/plotting/rscripts/fn_plot_helpers.R..... 176
btusersegmentation/plotting/rscripts/gather_results_bidmach.R..... 178
btusersegmentation/plotting/rscripts/gather_results_spark.R..... 179
btusersegmentation/plotting/rscripts/gather_self_source.R..... 180
btusersegmentation/plotting/rscripts/gather_tex_plot_ads_hist2d.R.. 180
btusersegmentation/plotting/rscripts/gather_tex_plot_aggr.R..... 181
btusersegmentation/plotting/rscripts/plot_ads_hist2d.R..... 182
btusersegmentation/plotting/rscripts/plot_aggrs.R..... 184
btusersegmentation/plotting/rscripts/setup.R..... 185
btusersegmentation/plotting/rscripts/static_ad_revenue_growth.R.... 186
btusersegmentation/plotting/rscripts/util.R..... 187

Aliulis, D. User Segmentation for Online Advertising using Latent Dirichlet Allocation. *Master's work in applied mathematics* / supervisor assoc. prof. dr. Vytautas Janilionis; Kaunas University of Technology, Faculty of Mathematics and Natural Sciences, Department of Applied Mathematics.
Kaunas, 2015. 54p.

SUMMARY

Online advertising is a fast growing multi-billion industry. Its revenue has totaled 30.7 billion Euro in Europe and 49.5 billion US dollars in USA in 2014. When to compared to other forms of adverting, online is the most effective means to reach the right customer, but the problem of serving the right ad to the right customer remains. Behavioral targeting addresses this problem and user segmentation is an essential part of it.

Behavioral targeting for online advertising has gained more attention from academic researchers in 2009. The early works employed classical clustering methods for user segmentation. Later it has been shown that topic models are able to capture the semantics of user behavior and outperform the classical methods. However, the experiments were done using relatively small sample sizes and small numbers of segments, therefore it is of interest to conduct thorough research into the effects of user segmentation for behavioral targeting.

In this paper Latent Dirichlet Allocation model is used to segment users for online advertising. Tools needed for data preparation, modeling experiments and evaluation of segmentation effectiveness are implemented using Big Data technologies. User segmentation and evaluation of results are carried out on a real-world dataset containing user search query and click-through logs.

SANTRUMPOS

API – taikomųjų programų kūrimo sąsaja (angl. *Application Programming Interface*).

CPU – centrinis procesorius (angl. *Central Processing Unit*).

CTR – interneto reklamos paspaudimų ir parodymų santykis (angl. *Click-Through Rate*).

DSL – specialios paskirties programavimo kalba (angl. *Domain Spefic Language*).

ETL – duomenų surinkimo, transformavimo ir įkėlimo procesas (angl. *Extract Transform Load*).

F – segmentavimo F-matas (angl. *F-measure*).

GPU – grafinis procesorius (angl. *Graphics Processing Unit*).

HDFS – Hadoop paskirstyta failų sistema (angl. *Hadoop Distributed File System*).

ID – identifikacinis numeris arba simbolių rinkinys.

LDA – Latentinis Dirichlė paskirstymo modelis (angl. *Latent Dirichlet Allocation*).

MCMC – Monte-Karlo Markovo grandinės metodas (angl. *Markov Chain Monte-Carlo*).

P – segmentavimo tikslumas (angl. *Precision*).

R – segmentavimo atkuriamumas (angl. *Recall*).

REPL – interaktyvi komandinės eilutės vartotojo sąsaja (angl. *Read Eval Print Loop*).

SDDMM – atrinktoji matricų daugybos operacija (angl. *Sampled Dense-Dense Matrix Multiplication*).

SGD – stochastinis gradientinis nusileidimas (angl. *Stochastic Gradient Descent*).

SQL – struktūrizuota užklausų kalba (angl. *Structured Query Language*).

ĮŽANGA

Interneto reklama yra sparčiai auganti multimilijardinė industrija. Iš jos gaunamos pajamos Europoje per pastaruosius 9 metus išaugo nuo 6.6 iki 30.7 mlrd. eurų [1], o JAV – nuo 16.9 iki 49.5 mlrd. JAV dolerių [2]. Lyginant su kitomis reklamos rūšimis, interneto reklama leidžia efektyviausiai pasiekti reikiamą vartotoją, tačiau pagrindinė problema yra aktualiausios reklamos parinkimas konkrečiam vartotojui. Vienas iš patikimiausių būdų tai pasiekti yra vartotojų elgsena internete pagrįsta reklama, kurioje svarbiausias uždavinys yra vartotojų segmentavimas.

Moksliniuose darbuose vartotojų elgsena pagrįstos reklamos efektyvumas išsamiau pradėtas nagrinėti 2009 m. Juose vartotojų segmentavimui naudojami klasikiniai klasterizavimo metodai. Vėlesnėje literatūroje parodoma, kad teksto turinio analizės modeliai, gebantys panaudoti vartotojų elgsenos semantinę informaciją, yra pranašesni už klasikinius metodus. Šiuolaikinės interneto reklamos uždaviniai priskiriami didžiųjų duomenų rinkinių kategorijai, tačiau literatūroje aprašytuose tyrimuose naudojamos sąlyginai mažos imtys ir parenkami nedideli segmentų kiekiai, todėl aktualu atlikti išsamesnius teksto turinio analizės modelių tyrimus.

Šiame darbe interneto reklamos vartotojų segmentavimo uždavinio sprendimui sudarytas latentinis Dirichlė paskirstymo modelis. Didžiųjų duomenų rinkinių analizės programinėmis priemonėmis sukurta programinė įranga, reikalinga latentinio Dirichlė paskirstymo modelio taikymui ir segmentavimo efektyvumo tyrimui. Panaudojant realius vartotojų interneto paieškos užklausų, reklamų parodymų ir paspaudimų duomenis atliktas vartotojų segmentavimas ir įvertintas jo efektyvumas.

Darbo tematika perskaityti pranešimai konferencijose:

- konferencijoje „Matematika ir matematikos dėstymas 2015“ – KTU;
- XIII studentų konferencijoje „Matematika ir gamtos mokslai: teorija ir taikymas“ – KTU;

ir paskelbta publikacija konferencijos pranešimų medžiagoje:

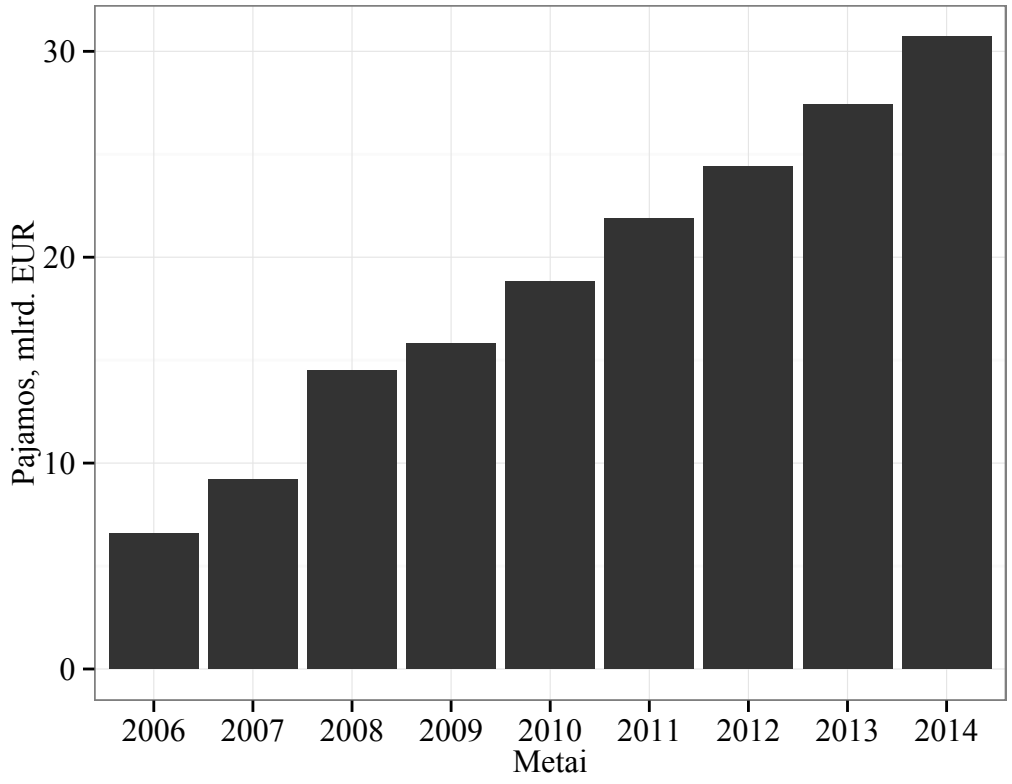
- Aliulis D., Janilionis V. User Segmentation for Internet Advertising using Latent Dirichlet Allocation with Online Learning. XIII studentų konferencijos pranešimų medžiaga. Kauno technologijos universitetas. Kaunas: Technologija, 2015. ISBN 9786090211342. p. 41 – 42.

1. LITERATŪROS APŽVALGA

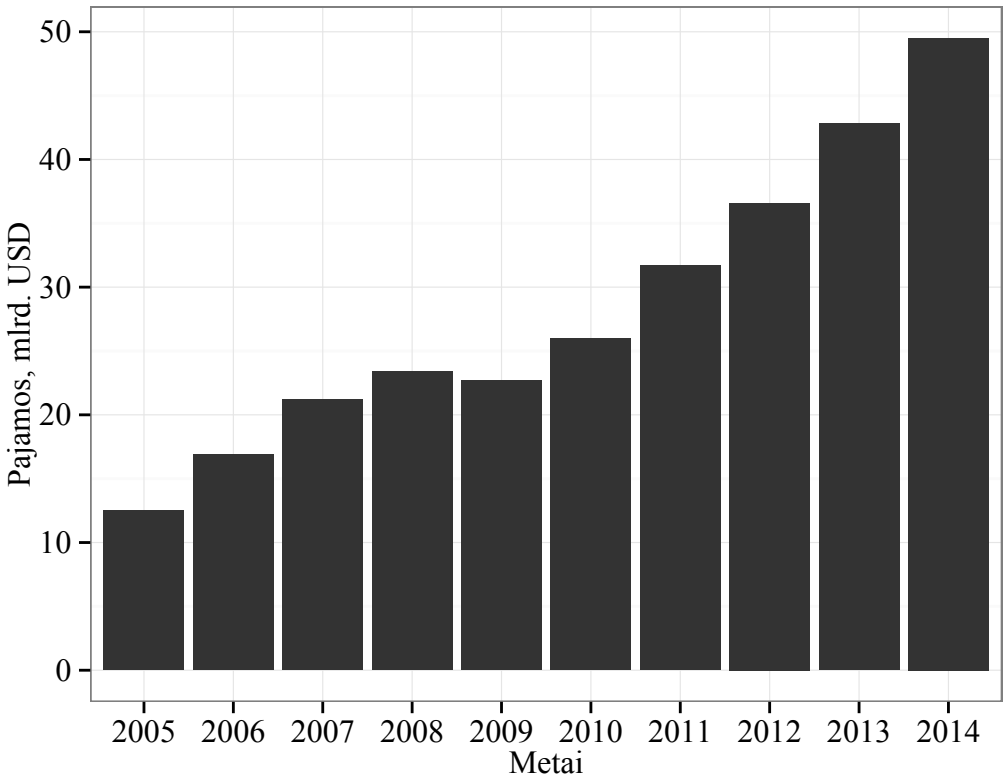
Šioje dalyje pateikta interneto reklamos vartotojų segmentavimo problemą, apžvelgti kitų autorių darbai šia tematika, latentinis Dirichlė paskirstymo modelis, dažniausiai vartotojų segmentavimui taikomi klasterizavimi metodai. Taip pat aptartos didžiųjų duomenų rinkinių (angl. *Big Data*) analizei tinkamos atviro kodo programinės priemonės ir suformuluoti šio darbo uždaviniai.

1.1. REKLAMA INTERNETE

Per pastarąjį dešimtmetį internetas reklamos teikėjams tapo viena svarbiausių terpių pasiekti pasiekti potencialius prekių ir paslaugų vartotojus. Pagal interaktyvios reklamos biuro (IAB – angl. *Interactive Advertising Bureau*) pateikiamas ataskaitas [1] [2], interneto reklama jau eilę metų yra multimilijardinė industrija ir yra sparčiausiai augantis reklamos sektorius tiek Europoje, tiek JAV. 2014m. Europoje išlaidos interneto reklamai, siekė 30.7 mlrd. eurų, t.y. 3.3 mlrd. eurų (11.6 %) daugiau negu 2013 m. (žr. 1.1 pav.), o JAV 2014 m. išlaidos siekė 49.45 mlrd. JAV dolerių, t.y. 6.67 mlrd. (15.6 %) daugiau negu 2013 m (žr. 1.2 pav.).



1.1 pav. Pajamos iš interneto reklamos 27 Europos šalyse pagal metus [1].



1.2 pav. Pajamos iš interneto reklamos JAV pagal metus [2].

Reklama internete leidžia pasiekti didelius kiekius potencialių klientų, tačiau tai nėra pirminė šio reklamos pateikimo terpės augimo priežastis. Reklama internete turi kur kas daugiau potencialo pasiekti reikiamus vartotojus, t.y. tokius, kurie bus suinteresuoti reklamos turiniu ir bus labiausiai linkę įsigyti reklamuojamą produktą, paslaugą ar atlikti kitą reklamos davėjo siekiamą veiksmą.

Potencialius klientus internete galima pasiekti keletu būdų.

Reklama paieškoje (angl. *Sponsored Search*) [3] pagrįsta tekstinės informacijos rodymu kartu su rezultatais paieškos sistemose, tokiose kaip Google, Bing, Yahoo! ir kt. Paieškos sistemos tikslas yra pateikti vartotojui aktualią reklamą, kurią jis galbūt paspaus ir tokiu būdu generuos pajamas paieškos sistemai.

Kontekstinė reklama (angl. *Contextual advertising*) [4] yra kita reklamos rūšis, pateikiama reklaminių skydelių (angl. *banner*), vaizdinės medžiagos ir kitomis formomis pagal interneto svetainės turinį.

1.2. INTERNETO REKLAMOS VARTOTOJŲ SEGMENTAVIMO UŽDAVINYS

Pagrindinis interneto reklamos siekis yra vartotojams pateikti kuo aktualesnę reklamą ir taip maksimizuoti tikimybę, kad vartotojas atliks reklamos davėjo norimą veiksmą. Nors ir galima pasiekti kiekvienam vartotojui sukuriant modelį, prognozuojantį kaip jis pasielgs pateikus reklamą atitinkame kontekste, tokios prognozės yra gana nepatikimos ir pasižymi didele sklaida, kadangi siekiami vartotojo veiksmai yra ypatingai reti įvykiai, o vartotojų aibė yra itin didelė (šimtais milijonų vartotojų) ir pasižymi nepaprastai dideliu kintamumu, t.y. kadangi interneto rekmlamoje vartotojai identifikuojami pagal interneto naršyklės slapukus (angl. *cookies*), kiekvieną dieną internete sutinkama daug naujų vartotojų ir daug iš anksčiau sutiktų yra nebegaliojantys [5]. Dėl šių priežasčių siekiama panašius vartotojus suskirstyti į segmentus, kurių skaičius yra keletą eilių mažesnis negu vartotojų skaičius.

Interneto reklamos vartotojų segmentavimo uždavinio sprendimo darbų apžvalga

Vartotojų segmentavimas siekiant padidinti reklamos pateikimo sistemų efektyvumą pastaraisiais metais sulaukia daug tyrėjų dėmesio, tačiau verta pastebėti, kad akademinė bendruomenė į vartotojų elgsena pagrįsto reklamų nukreipimo ir vartotojų segmentavimo tyrimus įsitraukė ganėtinai neseniai.

Dažnai cituojamas kaip pirmasis literatūroje sutinkamas darbas [6], įvertinantis vartotojų segmentavimu padidinamą reklamos efektyvumą yra atliktas „Microsoft“ tyrėjų paskelbtas 2009m. Šį darbą apžvelgsime detaliau, kadangi jame sukurta segmentavimo įvertinimo metodika ir vartotojų reprezentavimo rezultatais remiasi vėliau paskelbti darbai. Darbe taikomi 3 klasterizavimo metodai: K-vidurkių [7], konkrečiai neįvardijamas vienas iš trijų programinės įrangos pakete CLUTO [8] esančių metodų ir aproksimacinis artimiausių kaimynų metodas [9]. Naudojami komercinės interneto paieškos sistemos savaitės laikotarpio 2008 m. duomenys. Rinkinyje yra 6 426 633 unikalių vartotojų ir 335 170 unikalių reklamų. Po išskirčių pašalinimo paliekama 17 901 unikalių reklamų. Interneto paieškos sistemos vartotojai reprezentuojami keliais būdais: jų atliktomis paieškos užklausomis ir aplankytais interneto tinklalapiais, abiem atvejais panaudojant trumpo laikotarpio (1 dienos) ir ilgo laikotarpio (7 dienų) istorinius duomenis. Įvedamos vartotojų segmentavimo efektyvumo įvertinimo metrikos P, R, F ir paspaudimų entropija kurios naudojamos ir kituose vėliau paskelbtuose darbuose. Gautais rezultatais pagrindžiamas vartotojų segmentavimo taikymas interneto reklamoje: vartotojai, paspaudę tą pačią reklamą priklausomai nuo reprezentavimo metodo gali būti daugiau negu 90% panašesni negu vartotojai spaudę kitas reklamas. Parodoma, kad vartotojų reprezentavimas atliktomis interneto užklausomis yra efektyvesnis, negu reprezentavimas aplankytais tinklapiais, trumpo laikotarpio duomenų naudojimas abiems atvejais yra efektyviau už ilgo laikotarpio duomenų naudojimą segmentavimui. Po segmentavimo į 160 segmentų CTR rodiklio santykinis pagerėjimas, taikant K-vidurkių ir vienu iš CLUTO klasterizavimo metodą yra 670%, o taikant aproksimacinį artimiausių kaimynų metodą 1000%. Darbe atliekamas išsamus paieškos sistemos vartotojų segmentavimo efektyvumo tyrimas interneto reklamos kontekste, tačiau

vienas iš pagrindinių pastebėjimų yra tai, kad tyrimai atliekami tik segmentų kiekiams 20, 40, 80, 160. Efektyviausios metodikos taikymo atvejais santykinis CTR pagerėjimas tarp didesnio ir mažesnio segmentų kiekio yra žymus ir galima numatyti, kad jį padidinus būtų gaunami dar geresni rezultatai. Suprantama, kad didesnio segmentų kiekio parinkimas reikalauja daugiau skaičiavimų, o autorių naudota programinė įranga CLUTO [8] buvo parašyta 2002 m., o darbe cituojamas K-vidurkių metodo tyrimo ir programinės realizacijos straipsnis [7] paskelbtas 2002 m.

Vienas iš pirmųjų darbų, kuriame interneto reklamos vartotojų segmentavimui panaudojami vartotojų semantiniai duomenys yra 2009 m. paskelbtas [10] darbas. Skirtingai nuo tuo metu įprastų vartotojų segmentavimo metodų, kuriuose nenaudojama vartotojų elgsenos internete semantika, autoriai pasiūlo tikimybinį latentinį vartotojų segmentavimo modelį (PLSUS – angl. *Probabilistic Latent Semantic User Segmentation*). Detalių apie programinę realizaciją autoriai nepateikia. Tyrimui autoriai naudoja neįvardintos komercinės paieškos sistemos istorinius duomenis. Eksperimentai atliekami su dviem duomenų rinkiniais, kuriuose yra atitinkamai 120 000 ir 150 000 įrašų, kurių kintamieji yra vartotojo – interneto naršyklės slapuko (angl. cookie) ID, vartotojo pateikta paieškos užklausa, vartotojui parodytos reklamos ID ir dvejetainis kintamasis, aprašantis ar reklama buvo paspausta. Pirmasis duomenų rinkinys segmentuojamas į 5 ir 10 segmentų, antrasis į 10 ir 20 segmentų. Segmentavimo įvertinimui naudojamos santykinio CTR pagerėjimo, P, R ir F metrikos, kurios buvo pasiūlytos [6]. Darbe parodoma, kad autorių pasiūlytas modelis, pasitelkinatis vartotjų elgsenos semantiką pasiekia geresnį tikslumą, negu klasikiniai metodai. Autoriai pastebi ir [10] darbe naudojamos geriausio segmento parinkimo strategijos aspektą, į kurį reikia itin atsižvelgti vertinant segmentavimo efektyvumą. Šį aspektą detaliau aptarsime 2.3. skyriuje, tačiau verta paminėti, kad vėliau paskelbti darbai [11] ir [12] į šį aspektą neatsižvelgia. Darbe gauti rezultatai, kad su 5 ir 10 segmentų PLSUS modeliu gautas santykinis CTR pagerėjimas yra 78.78% ir 135.83%, o [6] naudotais klasikiniais klasterizavimo metodais šio rodiklio reikšmės yra atitinkamai ne didesnės už 64.44% ir 129.54%. Santykinis CTR pagerėjimas naudojant kitą duomenų rinkinį ir segmentuojant į 10 ir 20 segmenų yra 130.36% ir 265.49% PLSUS modeliui bei ne didesnės negu 74.47% ir 120.76% klasikiniams klasterizavimo metodams. Nors darbe parodomas autorių pasiūlyto modelio efektyvumas lyginant su klasikiniais klasterizavimo metodais, tačiau eksperimentai atliekami su nedideliais duomenų kiekiais, o vartotojai segmentuojami į akivaizdžiai per mažai segmentų. Kaip to priežastį autoriai nurodo techninius ribojimus skaičiavimų prasme.

Literatūroje cituojamas kaip pirmas darbas, kuriame interneto reklamos vartotojų segmentavimui taikomas latentinis Dirichlė paskirstymo modelis (LDA) yra [11], paskelbtas 2010 m. Tradiciniai vartotojų segmentavimo metodai, pagrįsti klasikiniais klasterizavimo metodais, tokiais kaip K-vidurkių, hierarchinio klasterizavimo ir kt. paprastai požymiams naudoja vartotojų užklausose esančius paieškos raktažodžius. Tokio metodo problema, kad vartotojai, kurių raktažodžiai yra semantiškai susiję, tačiau nėra tapatūs, po segmentavimo nepateks į tą patį segmentą. Eksperimentams autoriai naudoja populiariausios Kinijos elektroninės parduotuvės lankytojų užklausų istorinius duomenis. Duomenų rinkinyje yra daugiau negu 6.5 mln. unikalių vartotojų, tačiau autoriai tyrimams parenka atsitiktinę 300 000 vartotojų imtį, kurie pateikė daugiau negu 4 užklausas ir paspaudė bent vieną jiems parodytą reklamą. Segmentavimo įvertinimui naudojamos [6] pa-

siūlytos santykinio CTR pagerėjimo, P, R, F ir paspaudimų metrikos. Rezultatai palyginami su trimis kitais segmentavimo metodais: K-vidurkių klasterizavimu, hierarchiniu klasterizavimu ir tikimybiniu latentinės semantinės analizės modeliu (PLSA – angl. *Probabilistic Latent Semantic Analysis*) [10]. Šie metodai lyginami parenkant skirtingus segmentų kiekius K: 20, 30, 40, 50 ir 60. Didžiausias LDA pranašumas prieš kitus modelius gaunamas kai segmentų kiekis yra 30 ir nepaisant to, kad santykinis CTR pagerėjimas didėjant segmentų kiekiui auga beveik tiesiškai, autoriai daugiau dėmesio skiria segmentų kiekio $K = 30$ atvejui ir lygina LDA rezultatus su skirtingais Gibbs ėmimo [13] parametrų įvertinimo metodo iteracijų kiekiais. Reikia pastebėti, kad autoriai neįvertina galimų apgaulingų reklamos paspaudimų, kuriuos atlieka kompiuterių programos, o ne žmonės. Šis reiškinys yra itin aktuali interneto reklamos problema [14]. Autoriai [11] darbe atlieka pilno paketo skaičiavimus (angl. *batch-processing*), todėl suprantama, kad didesnių K reikšmių parinkti neleidžia techniniai apribojimai. Šiuo metu yra atlikta daug LDA modelio modifikacijų tyrimų tekstų analizės srityje. Panaudojant atitinkamas LDA modelio modifikacijas jį būtų galima praktiškai taikyti ir interneto reklamos vartotojų uždavinio sprendimui.

Kitas iš keleto literatūroje sutinkamų darbų, kuriame interneto reklamos vartotojų segmentavimui taikomas LDA modelis yra [12]. Skirtingai negu [11] darbe, segmentų sudarymui naudojami duomenys yra paieškos užklausos ID, paspaustos reklamos ID ir vartotojo – interneto naršyklės slapuko (angl. *cookie*) – ID, o ne vartotojų interneto paieškos užklausų žodžiai. Tai yra inovatyvus metodas, kuriuo modeliuojamas ryšys tarp vartotojo atliktos užklausos ir paspaustos reklamos. Konkrečiau, užuot LDA modelį taikę duomenų rinkiniui, kur stebėjimai yra vartotojai, o požymiai yra jų užklausų žodžiai, darbe požymiai yra vartotojai, o stebėjimai parenkami vienu iš trijų būdų: reklamos, paieškos užklausos, paieškos užklausos ir paspaustos reklamos pora. Segmentavimo efektyvumo įvertinimui naudojamos [6] pasiūlytos santykinio CTR pagerėjimo, P, R ir F metrikos. LDA modelio parametrų įvertinimui naudojamas Gibbs ėmimo metodas, o įvertinimas atliekamas pilno-paketo būdu. Tyrimams naudojamas vienas iš nedaugelio atvirai prieinamų interneto reklamos duomenų rinkinių [15]. Po filtravimo duomenų rinkyje lieka 438 485 unikalios užklausos, 574 541 unikalūs vartotojai ir 25 683 unikalios reklamos. Segmentuojama į 20, 40, 60 ir 80 segmentų. Duomenų rinkinys padalinamas į dvi dalis, kuriose yra atitinkamai po 90% ir 10% reklamų. Modelis sudaromas panaudojant 90% reklamų duomenų, įvertinamas šio segmentavimo efektyvumas, tuomet sudarytas modelis naudoajmas likusių 10% reklamų segmentavimui ir įvertinama koks efektyvumas pasiekiamas anksčiau nematytoms reklamoms. Kaip ir darbuose [6], [10], [11], santykinis CTR pagerėjimas akivaizdžiai didėja su didesnėmis segmentų kiekio reikšmėmis. Nors autoriai pasiūlo inovatyvų LDA modelio sudarymo metodiką, tokiu būdu sudarytas modelis negali segmentuoti naujų vartotojų segmentų, tik iki tol nematytas reklamas. Interneto reklamoje vartotojų kintamumas yra itin didelis, kadangi vartotojai yra identifikuojami pagal interneto naršyklės slapukus (angl. *cookies*), kurie gali būti bet kada ištrinti [5]. Aptikus naują vartotoją ir norint jį priskirti segmentui, reikėtų visą modelio sudarymo procesą atlikti iš naujo, kadangi naujas vartotojas reikštų naują požymį. Su šia esmine problema nesisudriama kuomet požymiams naudojami vartotojų užklausų žodžiai kaip tai daroma kituose apžvelgtuose darbuose. Tokiu atveju blogiausias scenarijus yra anksčiau nematyto užklausos žodžio neįvertinimas priskiriant vartotoją

į segmentą, ko galima išvengti modelio sudarymui naudojant pakankamo dydžio žodyną.

1.3. ATVIRO KODO DIDŽIŲJŲ DUOMENŲ RINKINIŲ ANALIZĖS PROGRAMINĖS ĮRANGOS LYGINAMOJI ANALIZĖ

Daugelio interneto reklamos uždavinių duomenų kiekis yra itin didelis ir yra priskiriamas didžiųjų duomenų rinkinių kategorijai. Ne išimtis yra ir interneto reklamos vartotojų segmentavimo uždavinys, todėl jo sprendimui reikalingos atitinkamos programinės priemonės. Šiame skyriuje apžvelgiame atviro kodo dižiųjų duomenų analizės programinę įrangą. Išsamiau aptariame šio darbo programinei realizacijai panaudotas *Spark SQL* ir *BIDData* bibliotekas.

Hadoop yra *Java* programavimo kalba parašytas atviro kodo (angl. *open source*) programinės įrangos projektas, skirtas didžiųjų duomenų rinkinių saugojimui ir apdorojimui. *Hadoop* buvo sukurtas duomenų apdorojimui paskirstant užduotį į tūkstančius serverių su ypatingai dideliu atsparumu techniniams gedimams. Vienos skaičiavimų mašinos sugedimo tikimybė yra labai maža, tačiau kuomet klasteryje yra tūkstančiai serverų, tikimybė, kad ilgai trunkančių skaičiavimų metu iš jų nė vienas nesuges gerokai sumažėja. Sistemos naudojimui nereikia ypatingos techninės įrangos – brangiai kainuojančių galingų serverių. *Hadoop* kompiuterių klasteriai yra sudaromi iš eilinių kompiuterių ar serverių. Tokių klasterių atsparumas nesėkmėms, pvz. vienos iš skaičiavimo mašinų gedimui, pagrįstas jų išsprendimu dar programiniame sluoksnyje, nesikliaujant technine įranga. Projektas yra sudarytas iš dviejų pagrindinių dalių: duomenų talpinimo *Hadoop* paskirstyta failų sistema (angl. *Hadoop Distributed File System*) [16] ir duomenų apdorojimo dalies *MapReduce* [17]. Hadoop suskaido failų į didelius blokus ir paskirsto juos į klasterio mazgus. *Hadoop* klasteris turi pagrindinį mazgą ir daugelį antrinių mazgų. Pagrindiniame mazge saugoma identifikavimo, veikimo ir duomenų informacija (*JobTracker*, *TaskTracker*, *NameNode* ir *DataNode*). O antriniuose mazguose saugoma veikimo ir duomenų informacija (*TaskTracker* ir *DataNode*). Apdorojant duomenis, Hadoop *MapReduce* pagal duomenų rūšį, suskirsto ir persiunčia mazgų paketo programinį kodą paskirstytam apdorojimui.

Mahout terminas Hindu kalboje reiškia ant dramblio jojantį žmogų. *Hadoop* šiuo atveju dramblys, o *Mahout* [18] [19] – vienas iš daugelio projektų kuris gali „sėdėti“ ant šio dramblio. Projektas yra mašinų mokymo (angl. *Machine-Learning*) algoritmų rinkinys, naudojantis *MapReduce* paradigmą ir iš esmės papildantis *Hadoop*. Patalpinus didelį kiekį duomenų į HDFS failų sistemą, *Mahout* suteikia vietinio rėžimo ir keletą paskirstyto rėžimo duomenų analizės metodus. *Mahout* projekto tiesioginis tikslas yra kuo greičiau didžiulį duomenų rinkinį paversti į didelės apimties informaciją. Projektas parašytas *Java* kalba, tačiau pastaruoju metu pradėta vystyti *Scala* kalba paremta DSL mašinų mokymo sričiai ir mėginama skačiavimams vietoje *MapReduce* naudoti *Spark* platformą, tačiau projekto aktyvumas yra mažesnis, negu *Spark MLlib* mašinų mokymo modulio.

Vowpal Wabbit yra projektas, orientuotas į greitą mašinų mokymo modelių apmoky-

mą. Projekte optimizavimui naudojami stochastinio gradientinio nusileidimo, jungtinio gradiento, Broyden–Fletcher–Goldfarb–Shanno ir kt. metodai. *Vowpal Wabbit* pagal nutylėjimą palaiko mokymąsi ir optimizavimą internete [20], [21]. Tai yra kurį laiką žinoma metodika, pastaruoju metu dažnai taikoma didžiųjų duomenų kiekių analizėje. Projektas taipogi naudoja požymių maišos metodą (angl. *feature hashing*) [22], o tai leidžia sumažinti duomenų paruošimo laiko sąnaudas ir paspartina apmokymo procesą. *Vowpal Wabbit* realizuotas C++ programavimo kalba ir dėka mokymosi internete ir duomenų persiuntimo tinklu, juo galima analizuoti neribotus duomenų kiekius.

Apache Spark [23], [24], [25] yra bendro pobūdžio klasteriams skirta ir mazgų klaidos atspari skaičiavimo plaforma, kurios API palaiko *Java*, *Scala*, *Python* ir *R* kalbas. Ši platforma palaiko bedruosius užduočių vykdymo grafus ir interacinius skaičiavimus, todėl tinka kurti didelio masto mašinų mokymo taikomosioms programoms.

Spark MLlib yra *Apache Spark* paskirstų skaičiavimų mašinų mokymo biblioteka. Ji yra plačiausiai išvystyta mašinų mokymo biblioteka, skirta skaičiavimams klasteriuose [23]. Biblioteka skirta didelio masto modeliavimui ir išnaudoja duomenų ir modelių paralelizmą. *Spark MLlib* sudaro greitos standartinių mokymo algoritmų programinės realizacijos, kuriomis gali būti sprendžiami klasifikavimo, regresijos, rekomendacijų pateikimo, klasterizavimo ir dimensijos sumažinimo uždaviniai. Bibliotekoje realizuotos įvairių statiskų skaičiavimo, tiesinės algebros ir optimizavimo priemonės. *Spark MLLib* yra parašyta *Scala* programavimo kalba ir suteikia API *Java*, *Scala* ir *Python* programavimo kalbomis.

Glaudi intergracija su *Apache Spark* turi keletą svarbių privalumų. Pirma, *Apache Spark* buvo suprojektuotas iteratyvių skaičiavimų palaikymui, o tai leidžia kurti efektyvias didelio masto mašino mokymo programines realizacijas, kadangi dauguma mašinų mokymo algoritmų yra iteratyūs, o pažanga *Apache Spark* spartos prasme paspartina ir *Spark MLLib* nekeičiant bibliotekos kodo. Antra, *Apache Spark* populiarumas atviro kodo bendruomenėje lemia greitą *Spark MLLib* bibliotekos plėtojimą ir prigijimą, o biblioteką vysto daugiau negu 140 žmonių iš įvairių organizacijų. Trečia, *Spark MLLib* yra tik viena iš aukšto abstrakcijos lygio *Apache Spark* bibliotekų. Kito šios ekosistemos bibliotekos ir *spark.ml* paketo API derinys suteikia programuotojams platų įrankių pasirinkimą, kurie gerokai supaprastina praktinių mašinų mokymo sistemų kūrimą.

Spark SQL. Didžiųjų duomenų rinkinių taikomosiose programose dažniausiai tenka derinti keletu duomenų apdorojimo technikų, duomenų šaltinių ir jų talpinimo formatų. Viena pirmųjų sistemų, skirtų tokioms darbų apkrovoms, *MapReduce* [17], [26] suteikė galingą bet žemo abstrakcinio lygio procedūrinio programavimo sąsają. Tokios sistemos programavimas buvo keblus procesas, reikalaujantis daug neautomatizuoto optimizavimo iš programuotojo siekiant aukšto užduočių įvykdymo našumo lygio. Kuriant daugialypes naujas sistemas buvo siekiama sukurti kuo produktyvesnę darbo aplinką vartotojui pasiūlant reliacines sąsajas darbui su didžiaisiais duomenų rinkiniais. Tokiose sistemos kaip *Pig* [27], *Hive* [28], *Dremel* [29] ir *Shark* [30] suteikia vartotojui galimybę rašyti deklaratyvias užklausas, o tai leidžia iki tam tikro lygio atlikti automatizuotą vykdymo laiko optimizavimą.

Nors reliacinių sistemų populiarumas rodo, kad vartotojai dažnai renkasi deklaratyvių užklausų rašymą, tačiau reliaciniu modeliu pagrįsti metodai dažnai yra nepakankami didžiųjų duomenų rinkinių taikomosioms programoms. Visų pirma, įvairiems duomenų šaltiniams, kurie gali būti pusiau arba visai nestruktūrizuoti, vartotojai dažnai siekia atlikti ETL [31], [32], o tai reikalauja rankinio programinio kodo rašymo. Visų antra, vartotojai nori atlikti sudėtingą duomenų analizę taikant mašinų mokymo (angl. *machine learning*) ir grafų apdorojimo metodus, kuriuos yra sudėtinga realizuoti reliacinėse sistemose. Pastebėta, kad praktikoje didelę dalį duomenų apdorojimo kanalų (angl. *data pipelines*) būtų idealu realizuoti kombinuojant reliacines užklausas ir sudėtingus procedūrinius algoritmus. Tačiau dažnai dvi sistemų klasės – reliacinė ir procedūrinė – išlieka visiškai atskirtos, o tai verčia vartotojus rinktis vieną iš dviejų paradigmų.

Siekiant apjungti abu sistemų modelius Berkeley universiteto laboratorijose buvo sukurta *Spark SQL* [33] technologija, vienas iš esminių *Apache Spark* sistemos [24] komponentų.

Spark SQL apjungia reliacinį ir procedūrinį modelius dvejomis savo ypatybėmis. Pirma, *Spark SQL* duomenų karkaso (angl. *Data Frame*) API, kuris geba atlikti reliacines operacijas ir išoriniuose duomenų šaltiniuose, ir *Spark* integruotuose paskirtytuose duomenų rinkiniuose. Šis API yra panašus plačiai naudojamai duomenų karkaso koncepcijai *R* statistiniame pakete [34], [35], tačiau operacijos yra vertinamos „tingiu“ būdu *lazily*, t.y. veiksmai yra atliekami tik tuomet, kad prireikia jais aprašyto rezultato, o ne reikšmės priskyrimo metu ir tai leidžia atlikti reliacines optimizacijas. Antra, siekiant palaikyti platų įvairių duomenų šaltinių ir algoritmų spektrą didžiųjų duomenų rinkinių kontekste, *Spark SQL* naudojamas inovatyvus išplėtojamas optimizatorius (angl. *extensible optimizer*) pavadinimu *Catalyst*. Jo dėka galima lengvai įvesti pridėti naujo tipo duomenų šaltinių, optimizavimo taisyklių, ir duomenų tipų, taikomų tokiose srityse kaip mašinų mokymas (angl. Machine Learning).

Duomenų karkasų API suteikia plačią reliacinės ir procedūrinės paradigmų integracija *Spark* taikomosiose programose. Duomenų karkasai yra struktūrizuotų įrašų rinkiniai, kuriais galima manipuliuoti naudojant *Spark* procedūrinį API arba reliacinius API, kurie įgalina efektyvesnį įvykdymo optimizavimą. Karkasai gali būti sukuriami tiesiai iš *Spark* integruotų paskirstytų duomenų rinkinių *Java* arba *Python* objektų, suteikiant galimybę naudoti reliacinį modelį vartotojų parašytose *Spark* programose. Kiti *Spark* komponentai, tokie kaip mašinų mokymo biblioteka *MLlib* [23], taip pat priima ir kuria duomenų karkasus. Duomenų karkasai, daugelyje situacijų, yra daug patogesnė ir efektyvesnė abstrakcija negu *Spark* procedūrinis API.Pavyzdžiui, jie leidžia lengviau apskaičiuoti daugialypius suvestinius rodiklius (multiple aggregates) vienu duomenų rinkinio nuskaitymu naudojant SQL, kas yra sudėtinga išreikšti įprastu funkciniu API.Duomenų karkasai automatiškai išsaugo duomenis stulpelių formatu, kuris yra kompaktiškesnis negu *Java* ar *Python* objektai. Galiausiai, skirtingai nei *R* ar *Python* esančių duomenų karkasų API, *Spark SQL* karkasų operacijos prieš įvykdymą apdorojamos *Catalyst* reliacinio optimizatoriaus.

Siekiant palaikyti didelę duomenų šaltinių ir analitinių užduočių įvairovę *Spark SQL* sistemoje buvo sukurtas išplečiamas užklausų optimizatorius *Catalyst*. Jis naudoja *Scala* programavimo kalbos ypatybes, tokias kaip šablono pritaikymas (angl. *pattern-matching*), kad aprašyti su-

komponuojamas taisykles. Tai įgalina naudoti bendrą programavimo karkasą (angl. *framework*) įvykdymo medžių transformavimui, kurie bus naudojami vykdant analizę, įvykdymo planavimą ir programinio kodo generavimą vykdymo metu. Naudojant tokį programavimo karkasą, *Catalyst* gali būti išplečiamas: naujais duomenų šaltiniais, tokiais kaip pusiau struktūrizuoti duomenys JSON (angl. *JavaScript Object Notation*) formatu ir „išmaniasias“ duomenų talpyklas, pvz. *HBase* [36]; vartotojo aprašytomis funkcijomis; vartotojų aprašytais duomenų tipais konkrečioms taikymo sritims, tokios kaip mašinų mokymas. Funkcinės kalbos yra žinomos kaip puikiai tinkančios kompiliatorių kūrimui [37], taigi nesunku suprasti kodėl šiomis kalbomis buvo sukurtas ir išplėtojamas optimizatorius, įgalinantis nesunkiai praplėsti *Spark SQL*.

Spark SQL sistema, išleista 2014 m. gegužę, yra vienas labiausiai plėtojamų *Spark* komponentų. 2015 m. *Apache Spark* yra aktyviausias atviro kodo didžiųjų duomenų kiekių apdorojimo projektas, prie kurio per pastaruosius metus prisijungė per 400. *Spark SQL* jau yra pilnai naudojamas itin didelio masto gamybinėse aplinkose. Pavyzdžiui, viena iš didžiųjų interneto kompanijų naudoja šią sistemą duomenų srautų sudarymui ir užklausų vykdymui 8000 mazgų kompiuterių klasteryje su daugiau kaip 100 petabaitų duomenų. Kiekviena individuali užklausa naudoja dešimtis terabaitų [33]. Daugelis vartotojų naudoja *Spark SQL*ne tik SQL užklausoms bet taikomosiose *Spark* programose kombinuodami SQL su procedūriniu apdorojimu. Pavyzdžiui, 2/3 „Databricks Cloud“ klientų, naudoja *Spark SQL* kartu su kitomis programavimo kalbomis. Kaip teigiama [33], *Spark SQL* veikia iki 10 kartų greičiau, negu tokio paties funkcionalumo programinis kodas, parašytas gryna *Spark* API ir yra konkurencinga sistema lyginant su tik SQL naudojančiomis sistemomis, pagrįstomis *Hadoop* [16] technologija [33].

BIDData [38], [39], [40] yra tiriamajai duomenų analizės programinės įrankos rinkinys, skirtas tiriamąjai duomenų analizei. Leisti greitai pažinti duomenų rinkinį, formuluoti hipotezes ir jas patikrinti yra šios sistemos paskirtis.

Kuo greičiau įvykdoma ši tiriamoji stadija, tuo tikslesnį modelį galima sukurti per ribotą laiką. Realių duomenų analizės uždavinių sprendimo rezultatą dažnai sudaro keletas paprastesnių modelių. Sudėtingi modeliai reikalauja išsamaus parametrų derinimo, o tai paprastai daroma išmėginant daug variantų ir pilnai prižiūrint visą procesą. *BIDData* leidžia šį procesą atlikti tiek rankiniu, tiek automatizuotu būdu.

Kitas *BIDData* tikslas greito vystymo ir modelių derinimo realiu laiku palaikymas komercinio taikymo kontekste, t.y. leisti greitu ir paprastu būdu prototipus perkelti į gamybinę aplinką. Šio perkeitimo iš prototipų į gamybinius modelius metu dažnai prarandamos modelių tikslumas, siekiant tenkinti spartumo reikalavimus. Viena iš *BIDData* savybių yra itin didelė sparta, todėl sukurti prototipai gali net pranokti gamybinės aplinkos spartos reikalavimus.

BIDData yra skirtas pavieniams vartotojams arba nedideliems klasteriams, tačiau šios sistemos sparta naudojant vieną kompiuterį daugumoje mašinų mokymo uždavinių pranoksta klasterių programinės įrangos realizacijas, o toks naudojimo modelis tinka daugumai varotojų elgsenos uždavinių sprendimui [40]. *BIDData* sistemą sudaro: (1) duomenų sluoksnis, kuris balansuoja disko

vietą, CPU ir GPU paspartinimą, (2) *BIDMat* – interaktyvią analizę palaikanti matricų biblioteka, kuri apjungia CPU, GPU optimizacija ir skaičiavimų branduolius; (3) *BIDMach* – mašinų mokymo sistema su itin efektyviais optimizatoriais [38]; (4) Drugelio maiša (angl. *Butterfly mixing*) yra komunikacijos strategija, paslepianti dažnų modelio atnaujimų sukeliamą vėlavimą; (5) programinės įrangos projektavimo principai, padidinantys iteratyvius atnaujinimus naudojančių modelių apmokymo spartą.

BIDMat yra *BIDData* [40] projekto greitų skaičiuojamų matricų biblioteka, sukurta siekiant išpildyti specifinius projekto reikalavimus. Vienas iš projekto siekių yra galimybė išreikšti mašinų mokymo algoritmus aukšto lygio įprastiniais pažįstamais matematiniais objektais (pvz. matricomis), o žemesnio lygio operacijų vykdymas turėtų būti paslėptas nuo vartotojo. Plačiai paplitęs tokių įrankių kaip *Matlab R*, *Python* mašinų mokymo biblioteka *SciPy* ir kt. naudojimas tik patvirtina matricų sluoksnio svarbą mašinų mokymo įrankių rinkinyje. *BIDMat* tenkina šiuos reikalavimus:

Interaktyvumas suteikia vartotojui galimybę atlikti pakartotinius skaičiavimus, tikrinti modelius, pertvarkyti ir sudaryti sudėtingą analizės atlikimo eigą pažingsniui. Tai analitikui leidžia kur kas paprasčiau pajausti, įsigilinti ir suprasti duomenis. *BIDData* projektas parašytas *Scala* kalba, kuri turi yra efektyvią REPL sąsają.

Natūrali sintaksė. Programos kodas, atrodantis kaip ir matematinė formuluotė, kurią jis ir realizuoja yra lengviau vystomas, taisomas ir palaikomas. Jis paprastai buna ir gerokai trumpesnis. *Scala* kalboje yra realizuotos įprastinės matematinės operacijos, o naujus operatorius leidžiama aprašyti naudojant daugumą neabėcėlinių ir neskaitinių simbolių bei *unicode* koduotės matematinius simbolius, pvz. $+@, *!, *|, \bullet, \circ, \otimes, \vee, \wedge, ...$

CPU ir GPU paspartinimas. Daugeliui mašinų mokyme naudojamų skaičiavimų GPU yra gerokai našesnis, negu CPU [39], [40], [38]. *BIDMach* GPU pagreitinimą naudoja keletu būdų: GPU operatyviojoje atmintyje saugomoms matricoms ir paspartinant skaičiavimus ir operatoriams, naudojantiems GPU duomenims, saugomiems bendroje operatyviojoje atmintyje. Dėka *BIDMach* bendrųjų matricų abstrakcijos, realizuotiems algoritmams galima pakeisti skaičiavimo rėžimą tarp CPU ir GPU nekeičiant paties algoritmo kodo.

Lygiagrečių skaičiavimų paprastumas. *BIDMat* matricų skaičiavimai naudoja gijas, todėl programuotojui daugeliu atvejų nereikia rūpintis skaičiavimų lygiagretinimu. Prireikus lygiagretumo palaikymo ne *BIDMat* funkcijoms, o programuotojo rašomam kodui, galima naudoti itin grakščią ir nesudėtingą *Scala* aktorių abstrakciją, kurios dėka lygiagrečias gijas gali naudoti nebūtinai ekspertų lygio programuotojai [40].

Pakartotinis panaudojimas. Kadangi *Scala* naudoja *Java* virtualią mašiną (angl. *Java Virtual Machne*) ir gali iškviesti *Java* klases, programose gali naudoti didžiulę jau parašyto kodo bazę: (1) biblioteką GPU panaudojimui [41], (2) pagalbines bibliotekas specifinių uždavinių sprendimui, tokių natūralios kalbos apdorojimas, (3) failų įvesties-išvesties HDF5 [42] formatu *hdf5-java* biblioteką arba HDFS [16] formatu panaudojant *Hadoop*, (4) JMPI [43] pranešimų perdavimo biblioteką ir (5) skaičiavimus klasteriuose panaudojant *Hadoop*, [18], *Spark*, [25], [24], [44], *Hyracks* [45]

ir kt.

BIDMat realizuota keletas specialių branduolių (funkcijų arba operacijų), reikalingų užtikrinti įvykdymo spartą sprendžiant konkrečius uždavinius. Tai yra nestandartinės matricinės operacijos, itin pagreitinančios tam tikrų mašininio algoritmų skaičiavimus. Detaliau pateikiame tris iš jų:

SDDMM. Atrinktoji matricų daugybos operacija užrašoma **(1.1)**

$$P = A *_s B = (AB) \circ (S > 0) \tag{1.1}$$

kur A ir B yra atitinkamai $m \times p$ ir $p \times n$ tankios matricos, S yra $m \times n$ reta matrica, o $\circ$ yra Ademaro sandauga, $S > 0$ yra matrica, kurios elementai įgyja reikšmes 1 nenulinių S elementų vietose ir nulines reikšmes kitu atveju. P taip pat yra $m \times n$ reta matrica, kuriose nenulinių elementų indeksai sutampa su nenulinių S elementų indeksais, o šių elementų reikšmės yra AB sandaugos rezultatas. SDDMM operacijos daugiausiai skaičiavimo resursų užimta faktorinės analizės algoritmuose: alternuojančių mažiausiausių kvadratų (angl. *Alternating Least Squares*), išretintos faktorinės analizės (angl. *Sparse Factor Analysis*), latentinio Dirichlė paskirstymo, gamma Puasono. Kiekvienu atveju, S yra duomenų matrica (požymiai $\times$ vartotojai; požymiai $\times$ dokumentai) ir paprastai ji yra labai išretinta. Tiesioginiu būdu dauginti A ir B matricas yra labai nepraktiška, nes tik nedidelė dalis P elementų bus nelygūs nuliui.

Kraštiniai operatoriai (angl. *Edge Operators*). Dauguma matricų bibliotekų veikmams, atliekamiems pagal atitinkamus elementus (angl. *element-wise*), palaiko skaliarinius argumentus, pvz. $C = A \otimes B$, kur $\otimes$ yra $+$, $-$, $\circ$ ar kitas operatorius. Praktikoje dažnai tenka prie konkretiems konkrečių stulpelių ar eilučių pridėti atitinkamas konstantas. *BIDMat* tai realizuojama kraštiniais operatoriais. Kai A yra $m \times n$ matrica, yra leistini pagal atitinkamus elementus atliekami veiksmai su matrica B, kuri gali būti:

- $m \times n$ **matrica**, tuomet atliekamas įprastas veiksmas pagal atitinkamus elementus.
- $m \times 1$ **vektorius stulpelis**, tuomet atliekamas $\otimes$ veiksmas su kiekviena A eilute ir atitinkamu B. elementu.
- $1 \times n$ **vektorius eilutė**, tuomet atliekamas $\otimes$ veiksmas su kiekvienu A stulpeliu ir atitinkamu B elementu.
- 1×1 **skaliaras**, tuomet $\otimes$ su B ir visais A elementais.

Kraštiniai operatoriai paprastai veikia dvigubai greičiau negu kitos realizacijos [40] ir leidžia rašyti kompaktiškesnį ir lengviau suprantamą kodą.

Multivektoriniai operatoriai. Vienas tikslų iš *BIDData* projekto tikslų yra leisti apmokyti keletą modelių vienu metu ir modelių mišinius (angl. *ensembles*). Atliekant tokias užduotis daug efektyviau su keletu tokio paties dydžio vektorių atlikti matricinį veiksmą, negu veiksmus su atskirais vektoriais. Tarkime, reikia sudauginti skaliarą a ir m-vektorių b. Tai galima atlikti su n vektorių panaudojant $n \times 1$ A matricą, kurios elementai yra kiekvieno modelio a reikšmės, ir $m \times n$

multivektorių B, kurio stulpeliai yra b vektoriai. *BIDMach* tai užrašoma

$$C = A*@B \tag{1.2}$$

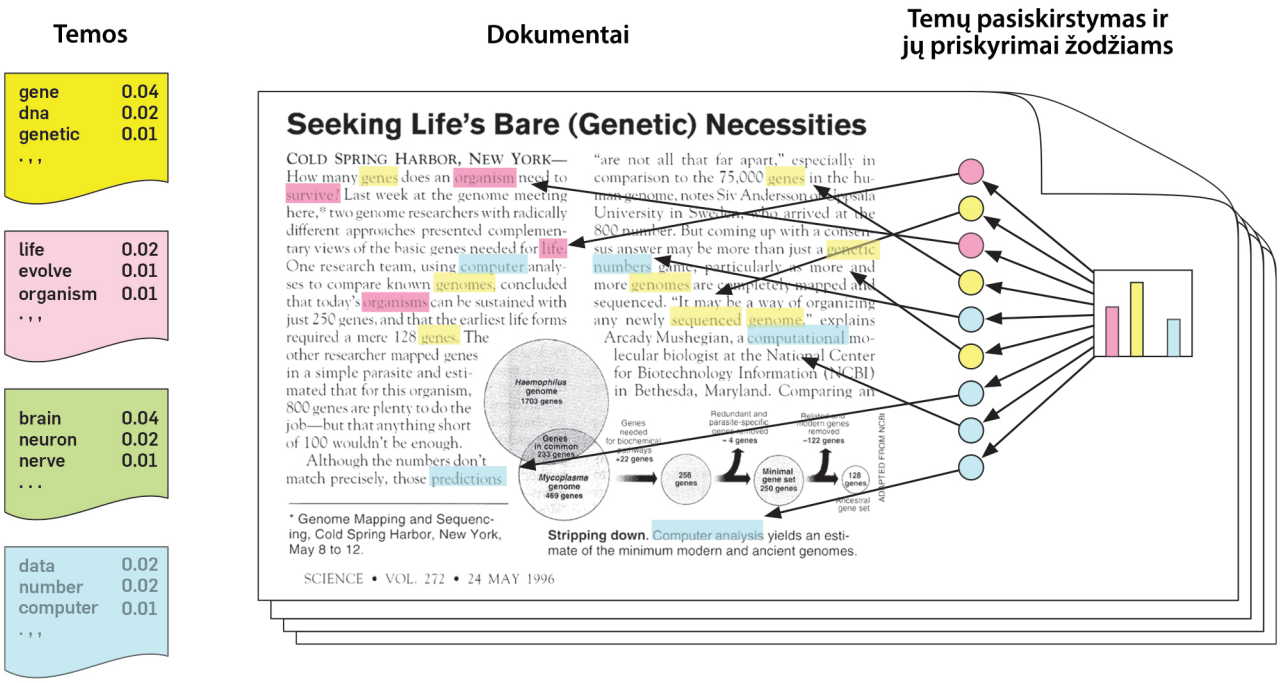
kadangi $*@$ yra kraštiniu operatoriumi pagal atitinkamus elementus atliekamas veiksmas. Kraštinių operatorių naudojimas reiškia, kad daugeliu atvejų vieno ir daugelio modelių vienu metu apmokymo *Scala* programų tekstai yra identiški.

BIDMach yra mašinų mokymo įrankių rinkinys sukurtas *BIDMat* pagrindu. Ketinama *BIDMach* išplėtoti į bendro pobūdžio mašinų mokymo įrankių rinkinį, tačiau šiuo metu susikoncentruota į didelio masto vartotojų elgsenos analitiką plačiausiai naudojamais modeliais: regresijai, faktorių modeliais, klasterizavimui ir kai kuriems modelių mišinių metodams (angl. *ensemble methods*). Pagrindiniai *BIDMach* elementai yra:

Dažno atnaujinimo parametrų įvertinimo metodai. Stochastiniai gradientiniai optimizavimo metodai ir MCMC – ypač Gibbs ėmimo – metodai išpopuliarėjo kaip du praktiškiausi parametrų įvertinimo gavimo metodai didžiųjų duomenų rinkinių kontekste. *BIDMach* realizuoti bendrieji SGD optimizavimo metodai ir Gibbs ėmimas. Abu metodai atlieka dažnus globalių modelių atnaujinimus, idealiu atveju kiekvienam stebėjimui, tačiau praktikoje tai atliekami mini-paketams (angl. *mini-batches*). Kai skaičiavimai atliekami klasteryje norint palaikyti optimalius mini-paketinius atnaujimus daug resursų pareikalauja komunikacija tarp klasterio mazgų, todėl *BIDMach* realizuotas naujas metodas, pavadintas *drugelio maiša* (angl. *Butterfly Mixing*) [46], [47], [39]. Siekiant pagerinti skaičiavimus klasteriuose vystomi žemesnio abstrakcijos lygio branduoliai [40], pvz. GPU paremtu parametriniai atsitiktinių skaičių generatoriai, kurie ženkliai paspartina Gibbs ėmimo metodus.

Keleto modelių apmokymas vienu metu ir modelių mišiniai. *BIDMat* biblioteka suteikia žemesnio abstrakcijo lygio bazinius elementus, ledžiandžius apmokyti keletą modelių vienu duomenų rinkinio nuskaitymu. Tai išnaudojama *BIDMach*, kuri palengvina aukštensio abstrakcijos lygio uždavinus: (1) parametrų tyrinėjimas, kai skirtingos to paties modelio realizacijos paleidžiamos su skirtingais parametrų rinkiniais, (2) parametrų reguliavimas (angl. *parameter tuning*), kai parinkami skirtingi pradiniai parametrų rinkiniai, kurie periodiškai pakeičiami dinaminiu būdu pagal gaunamus tikslo funkcijos įverčius ir paiešką vykdant siaurose lokalių optimalių taškų aplinkose; (3) k-lypis kryžminis patikrinimas, kai k modelių apmokomi lygiagrečiai, kiekvieno apmokymui nenaudojant n/k duomenų, (4) modelių mišinių metodai (angl. *ensemble methods*), kai lygiagrečiai apmokoma daug atskirų modelių. Pažangūs modelių mišinių metodai, tokie kaip super-mokymas [48] naudoja kelias iš šių technikų vienu metu.

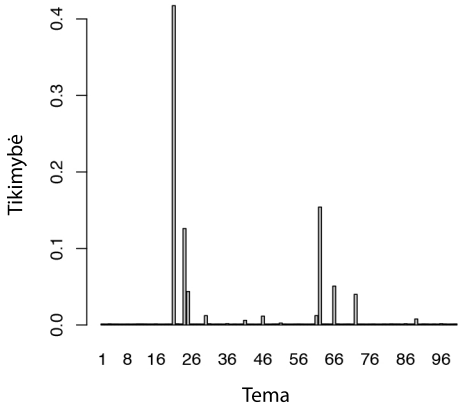
LDA modelis yra realizuotas *Mahout*, *Spark MLLib*, *Vowpal Wabbit* ir *BIDMach* bibliotekose. Pateikiame šio modelio veikimo iliustraciją [49].



1.3 pav. Latentinio Dirichlė paskirstymo modelio dokumentų generavimo iliustracija [49].

Generuojančio LDA modelio idėja pateikiama 1.3 pav. Pirmiausia padaroma prielaida, kad viso duomenų rinkinio mastu egzistuoja temų rinkinys, kurių kiekviena yra apibūdinama visų žodyno žodžių multinominiais pasiskirstymais (kairėje). Tuomet priimama prielaida, kad rinkinyje esantys dokumentai yra sugeneruoti toliau aprašomu būdu. Kiekvienas dokumentas turi savo temų multinominį pasiskirtymą (stulpelinė diagrama dešinėje). Kiekvienas dokumento žodis yra priskirtas konkrečiai temai (spalvoti apskritimai), kuri yra viso dokumentų rinkinio temų pasiskirstymo realizacija, o pats žodis yra tos temos pasiskirstymo realizacija. 1.3 pav. pateikti pasiskirtymai yra iliustravaus pobūdžio ir nėra išgauti iš realaus dokumentų rinkinio [49].

Rezultatai, gauti LDA pritaikius 17 000 žurnalo *Science* straipsnių ir temų kiekį parinkus 100 pateikiami 1.4 pav. Kairėje pavaizduotas 1.3 pav. pateikto straipsnio temų skirstinys, o dešinėje pateikiamia po 15 dažniausių žodžių iš keturių dažniausiai sutiktų temų [49].



"Genetics"	"Evolution"	"Disease"	"Computers"
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

1.4 pav. LDA modelio taikymo realiam dokumentų rinkiniui rezultatai [49].

1.4. DARBO TIKSLAS IR UŽDAVINIAI

Darbo tikslas: Sukurti interneto reklamos vartotojų segmentavimo modelį ir ištirti jo efektyvumą.

Uždaviniai:

1) Atlikti interneto reklamos vartotojų segmentavimui taikomų metodų, modelių ir programinių priemonių apžvalgą.
2) Sudaryti latentinio Dirichlė paskirstymo matematinį modelį interneto reklamos vartotojų segmentavimui.
3) Pasirinktomis didžiųjų duomenų rinkinių analizės priemonėmis sukurti programinę įrangą Latentinio Dirichlė paskirstymo modelio taikymui interneto reklamos segmentavimo uždavinio sprendimui ir gautų rezultatų įvertinimui.
4) Atlikti interneto reklamos vartotojų segmentavimo efektyvumo tyrimą, kai naudojami jų elsenos internete duomenys.

2. TYRIMO METODAI

Šioje dalyje sudaromas LDA modelis interneto reklamos vartotojų segmentavimui, aptariami jo praktiniai skaičiavimų aspektai ir apibrėžiamos segmentavimo efektyvumo vertinimo metrikos.

2.1. LATENTINIS DIRICHLĖ PASKIRSTYMO MODELIS INTERNETO REKLAMOS VARTOTOJŲ SEGMENTAVIMUI

Prieš sudarydami LDA modelį aprašome jame naudojamus tikimybinius skirstinius, o 2.1 lentelėje pateikiame šioje dalyje naudojamus žymėjimus ir jų paaiškinimus.

2.1 lentelė

LDA modelio žymėjimai

Žymėjimas	Paaiškinimas
D	interneto reklamos vartotojų skaičius
K	interneto reklamos vartotojų segmentų skaičius
V	paieškos užklausų žodyno žodžių skaičius
N	visų vartotojo pateiktų paieškos užklausų žodžių skaičius
$1 \leq d \leq D$	vartotojų indeksas
$1 \leq k \leq K$	vartotojų segmento indeksas
$1 \leq v \leq V$	paieškos užklausų žodyno žodžio indeksas
$1 \leq n \leq N$	vartotojo paieškos užklausų žodžio indeksas
α	Dirichlė skirstinio, iš kurio generuojami K ilgio tikimybių vektoriai θ_d, parametras
η	Dirichlė skirstinio, iš kurio generuojami V ilgio tikimybių vektoriai β_k, parametras
$\theta_{d,k}$	tikimybė, kad d-asis vartotojas priklauso k-ajam segmentui
$\beta_{k,v}$	tikimybė, kad v-asis žodyno žodis pasirodys k-ojo segmento vartotojų užklausose
$w_{d,n}$	d-ojo vartotojo n-asis visų pateiktų paieškos užklausų žodis
$1 \leq z_{d,n} \leq K$	d-ojo vartotojo n-ojo užklausų žodžio segmento indeksas, sugeneruotas iš $\mathrm{Multi}(\theta_d)$
$\mathbf{w}_d$	d-ojo vartotojo visų pateiktų paieškos užklausų žodžių $w_{d,n}$ vektorius

Dirichlė skirstinys yra itin svarbus LDA modelio komponentas [50]. k-dimensinis Dirichlė atsitiktinis dydis θ įgija reikšmes iš $(k-1)$ simplekso. k ilgio vektorius θ yra iš $(k-1)$ simplekso, jeigu jis tenkina sąlygas:

1) $\theta_i \geq 1$, kur $i \in \{1, \ldots, k\}$,
2) $\sum_{i=1}^{k} \theta_i = 1$.

Dirichlė skirstinio tankis k-dimensiam simpleksui **(2.1)**:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \theta_1^{\alpha_1 - 1} \cdots \theta_k^{\alpha_k - 1}, \qquad \textbf{(2.1)}$$

kur parametras α yra k ilgio teigiamų komponenčių vektorius, o $\Gamma(x)$ yra Gama funkcija **(2.2)**

$$\Gamma(x) = \int_0^\infty e^{-t} tkx - 1\, dt. \tag{2.2}$$

Dirichlė skirstinys yra patogus tuo, kad jis priklauso eksponentinų skirstinių šeimai ir yra jungtinis multinomimiam skirstiniui **(2.3)**, t.y. Bajeso tikimybių kontekste Dirichlė skirstinys yra apriorinis skirstinys aposterioriniam multinominiam skirstiniui [51].

Multinominis skirstinys yra binominio skirstinio apibendrinimas. Vykdant n nepriklausomų bandymų kiekvieno iš jų baigtis gali būti vienos iš k klasės įvykio pasirodymas, kur kiekviena klasė turi savo fiksuotą pasirodymo tikimybę, o visų klasių tikimybių suma lygi vienetui. Multinominis skirstinys aprašo konkrečios klasių pasirodymų kombinacijos tikimybę. Jo tikimybių masės funkcija

$$p(x_1, \dots, x_k; p_1, \dots, p_k) = \frac{\Gamma(\sum_i x_i + 1)}{\prod_i \Gamma(x_i + 1)} \prod_{i=1}^{k} p_i^{x_i} \tag{2.3}$$

kur x_i yra i-tosios klasės pasirodymo kiekis, n yra bandymų kiekis, p_i yra i-osios klasės pasirodymo tikimybė, k yra klasių kiekis, $i \in \{1, \dots, k\}$ yra klasės indeksas.

Modelio sudarymas. LDA yra generuojantis tikimybinis dokumentų rinkinio modelis. Sudarydami LDA modelį vartotojų segmentavimo uždaviniui remiamės idėja, kad interneto reklamos vartotojai apibūdinami atsitiktiniais paslėptų interesų (segmentų) tikimybinių vektoriais, o kiekvienas segmentas charakterizuojamas žodžių tikimybių vektoriumi. Tuomet LDA modeliu aprašomas interneto reklamos vartotojų paieškos užklausų generuojantis procesas aprašomas tokiu būdu:

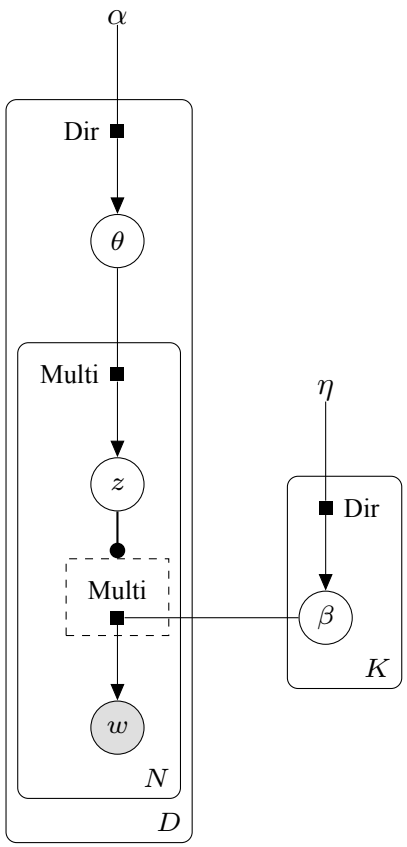
1) Kiekvienam vartotojų segmentui $k = 1, \dots, K$:
   (a) sugeneruojamas segmentą apibūdinantis V ilgo atsitiktinis tikimybių vektorius $\beta_k \sim \mathrm{Dir}(\eta)$.
2) Kiekvienam vartotojui, kuris apibrėžiamas jo pateiktų interneto paieškos užklausų žodžių vektoriumi $\mathbf{w}_d, d \in \{1, \dots, D\}$:
   (a) sugeneruojamas atsititkinis visų vartotojo pateiktų interneto paieškos užklausų žodžių skaičius $N \sim \mathrm{Puas}(\xi)$;
   (b) sugeneruojamas atsitiktinis K ilgio vartotojo interesų (priklausymo segmentams) tikimybių vektorius $\theta_d \sim \mathrm{Dir}(\alpha)$.
   (c) Kiekvienam vartotojo pateiktų užklausų žodžiui $w_n, n \in \{1, \dots, N\}$:
      i. sugeneruojamas intereso žodžio priskyrimas interesui $z_{d,n} \sim \mathrm{Multi}(\theta_d)$, $z_{d,n} \in \{1, \dots, K\}$;
      ii. sugeneruojamas žodis iš $z_{d,n}$ intereso (segmento) $w_{d,n} \sim \mathrm{Multi}(\beta_{z_{d,n}})$, $w_{d,n} \in \{1, \dots, V\}$.

LDA modeliui taikoma keletas jį supaprastinančių prielaidų:

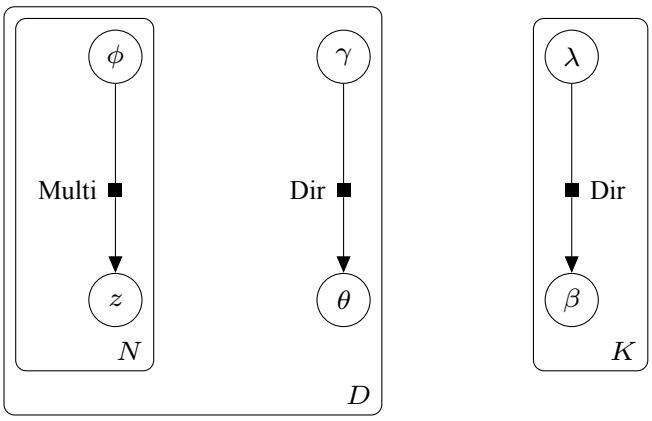
1) Dirichlė skirstinio dimensija k (tuo pačiu ir užklausų žodžių priskyrimų interesams (segmentams) skirstinio z) dimensija yra žinoma ir nekintanti.

2) Žodžių tikimybės parametrizuojamos $k \times V$ matrica β, kur $\beta_{ij} = p(w^j = 1|z^i = 1)$. Ši matrica nėra žinoma ir reikia rasti jos įvertį.
3) Puasono skirstinio prielaida nėra kritiškai svarbi tolesniems etapams, todėl dokumentų ilgiais gali būti pasirenkami kiti dydžiai. Verta pastebėti, kad N yra nepriklausomas nuo kitų generuojančių kintamųjų (θ ir $\mathbf{z}$), todėl tai yra pagalbinis kintamasis ir toliau aprašant modelį nelaikysime, kad N yra atsitiktinis dydis.

Tikimybius grafinius modelius įprasta pateikti orientuotų grafų grafų notacija [52]. Šia notacija LDA modelis pateikiamas 2.1 pav., o 2.2 pav. pateikiamas jo variacinis skirstinys.



2.1 pav. LDA modelis pateikiamas orientuotų faktorinių grafų notacija [52]. Rodyklės žymi kintamųjų priklausomybę, stačiakampiai žymi kartotinumą, balti skrituliai žymi paslėptuosius kintamuosius, pilkas apskritimas žymi stebimą kintamąjį.



2.2 pav. LDA modelio variacinio skirstinio pateikimas orientuotų faktorinių grafų notacija [52].

LDA modelio pilnoji tikimybė:

$$p(\theta, \beta, \mathbf{z}, \mathbf{w}|\alpha, \eta) = \prod_{k=1}^{K} p(\beta_k|\eta) \left(\prod_{d=1}^{D} p(\theta_d|\alpha) \left(\prod_{n=1}^{N} p(z_{d,n}|\theta_d) p(w_{d,n}|z_{d,n}, \beta_{z_{d,n}}) \right) \right) \quad \textbf{(2.4)}$$

LDA modelio aposteriorinis skirstinys [53]

$$p(\theta, \beta, \mathbf{z}|\mathbf{w}) = \frac{p(\theta, \beta, \mathbf{z}, \mathbf{w}|\alpha, \eta)}{p(\mathbf{w}|\alpha, \eta)} \quad \textbf{(2.5)}$$

kur $p(\mathbf{w}|\alpha, \beta) = \int \int \sum_{\mathbf{z}} p(\theta, \beta, \mathbf{z}, \mathbf{w}|\alpha, \eta)\, d\theta\, d\beta$,
t.y. stebimo dokumentų rinkinio pasirodymo tikimybė su bet kuriuo LDA modeliu. Kitaip tariant, reikėtų sumumuoti jungtinį skirstinį įvertinant visus galimas dokumentų rinkinio žodžių priskyrimo temoms variantus. Vardiklio tiesiogiai apskaičiuoti negalima [53]. Detalesni išvedimai pateikiami [54], [55], [56].

Variacinis Bayeso metodas.

LDA modelio tikrojo aposterinio skirstinio kintamieji yra susieti, tačiau vidutinio lauko variacinio skirstinio (angl. *mean field variational distribution*) kintamieji yra nepriklausomi ir kiekvienas yra susietas su skirtingu variaciniu parametru:

$$q(\theta, \mathbf{z}, \beta) = \prod_{k=1}^{K} q(\beta_k|\lambda_k) \prod_{d=1}^{D} \left(q(\theta_d|\gamma_d) \prod_{n=1}^{N} q(z_{d,n}|\phi_{d,n}) \right) \quad \textbf{(2.6)}$$

Kiekvienas paslėptasis kintamas yra aprašomas jo rūšies skirstiniu: temos β_k, $k \in \{1, \dots, K\}$ aprašomos V-dimensiniu Dirichlė skirstiniu λ_k; temų pasiskirstymai θ_d, $d \in \{1, \dots, D\}$ aprašomi K-dimensiu Dirichlė skirstiniu γ_d; temų priskyrimai žodžiams $z_{d,n}$ apra6omi K-dimensiniu multinominiu skirstiniu $\phi_{d,n}$. Tuomet stengiamasi parinti tokius variacinių parametrų įverčius, kad jie

minimizuotų Kullback-Leibler (KL) atstumą iki tikrojo aposterionio skirstinio:

$$(\gamma^*, \phi^*, \lambda^*) = \arg\min_{\gamma,\phi,\lambda} KL\left(q(\theta, \mathbf{z}, \beta) \| p(\theta, \beta|\mathbf{w})\right) \tag{2.7}$$

Detalesnė informacija apie KL atstumo metriką pateikta [57].

Nors tikslo funkcija negalima tiksliai apskaičiuoti, tačiau ją galima apskaičiuoti tiek, kad jos reikšmė būtų konstanta nebepriklausytų nuo variacinių parametrų, o ši konstanta yra stebėjimų log-tikėtinumo funkcijos reikšmė konkrečiam modeliui [53].

Tikėtinumo funkcija

$$\begin{aligned}\mathcal{L} = &\sum_{k=1}^{K} \mathrm{E}[\log p(\beta_k|\eta)] + \sum_{d=1}^{D} \mathrm{E}[\log p(\theta_d|\alpha)] + \sum_{d=1}^{D}\sum_{n=1}^{N} \mathrm{E}[\log p(z_{d,n}|\theta_d)] \\ &+ \sum_{d=1}^{D}\sum_{n=1}^{N} \mathrm{E}[\log p(w_{d,n})|z_{d,n}, \beta] - \sum_{n=1}^{N}\sum_{k=1}^{k} \phi_{nk} \log \phi_{nk}\end{aligned} \tag{2.8}$$

Detaliau apie šios funkcijos apskaičiavimą rašoma [56], [58], [55]. Apie variacinius metodus plačiau rašoma [59], [60], [61].

Tikslo funkcija optimizuojama parametrų didėjimo tvarka ir iteratyviai optimizuojant kiekvieną variacinį parametrą.

[53] **Variacinio metodo iteracija**

1) Kiekvienai temai $k,\ k \in \{1, \dots, K\}$ ir žodyno žodžiui $v,\ v \in \{1, \dots, V\}$:

$$\lambda_{k,v}^{(t+1)} = \eta + \textstyle\sum_{d=1}^{D} \sum_{n=1}^{N} 1(w_{dn} = v)\phi_{n,k}^{(t)} \tag{2.9}$$

2) Kiekvienam dokumentui $d,\ d \in \{1, \dots, D\}$:

   (a) Atnaujinti γ_d:

$$\gamma_{d,k}^{(t+1)} = \alpha_k + \textstyle\sum_{n=1}^{N} \phi_{d,nk}^{(t)}. \tag{2.10}$$

   (b) Kiekvienam žodžiui $n\ n \in \{1, \dots, N\}$:

      i. Atnaujinti $\phi_{d,n}$:

$$\phi_{d,n,k}^{(t+1)} \propto \exp\left\{\Psi(\gamma_{d,k}^{(t+1)}) + \Psi(\lambda_{k,w_n}^{(t+1)}) - \Psi(\textstyle\sum_{v=1}^{V} \lambda_{k,v}^{(t+1)})\right\} \tag{2.11}$$

kur Ψ yra digama funkcija, t.y. pirmoji funkcijos $\log\Gamma$ išvestinė.

LDA modelio parametrų įverčiai gaunami iš **(2.12)**, **(2.13)** ir **(2.14)**:

$$\hat{\beta}_{k,v} = \frac{\lambda_{k,v}}{\sum_{v'=1}^{V} \lambda_{k,v'}}. \tag{2.12}$$

$$\hat{\theta}_{d,k} = \frac{\gamma_{d,k}}{\sum_{k'=1}^{K} \gamma_{d,k'}}. \tag{2.13}$$

$$\hat{z}_{d,n,k} = \phi_{d,n,k}. \tag{2.14}$$

2.2. LATENTINIO DIRICHLĖ PASKIRSTYMO MODELIO REALIZACIJOS ASPEKTAI

Vartotojų interneto užklausų žodžiai aprašomi pagal vektorių erdvės modelį [62]. Žodis w yra bazinis diskrečių duomenų vienetas, priklausantis žodynui ir indeksuojamas $\{1, ..., V\}$. Žodis aprašomas V ilgio vektoriumi, kurio viena komponentė lygi 1, o kitos lygios 0. Tokiu būdu v-tasis žodyno žodis yra vektorius w, kuriame $w^v = 1$ ir $w^u = 0, u \neq v$.

Toliau pateikti variacinio Bayeso metodo realizacijos aspektai [53]:

Tarpinių reikšmių saugojimas. Skaičiavimų prasme intensyviausia dalis yra Ψ funkcijos skaičiavimas, todėl rekomenduojama $\mathrm{E}[\log \beta_{k,w}]$ ir $\mathrm{E}[\log \theta_{d,k}]$ reikšmes perskaičiuoti tik pasikeitus atitinkamoms variacinių parametrų reikšmėms.

Lizdinis skaičiavimas. Praktiniam taikymui pirmiausia skaičiuojami dokumento parametrai $\gamma_{d,k}$ ir $\phi_{n,d,k}$ kol jų reikšmės sukonverguoja ir tik tuomet atnaujinami temų parametrai $\lambda_{k,v}$ [58].

Pakartotiniai variacinio parametro ϕ atnaujinimai. Variacinio Bajeso metodo iteracijoje **(2.11)** išraiška yra skaičiuojama kiekvienam pasikartojančiam žodžiui w_n. Kiekvienam jo pasikartojimui atliekama po identišką parametro atnaujinimą, todėl verta parametro atnaujinimus daryti tik kiekvienam skirtingam dokumento žodžiui. Tokiu būdu **(2.11)** atnaujinimas užrašomas

$$\gamma_{d,k}^{(t+1)} = \alpha_k + \sum_{v=1}^{V} n_{d,v} \phi_{d,v}^{(t)} \tag{2.15}$$

kur $n_{d,v}$ yra žodyno žodžio v pasikartojimų skaičius dokumente d.

[53] teigiama, kad dėl **(2.15)** vidutinio lauko variacinis Bajeso metodo algoritmas yra pranašesnis už kitus metodus skaičiavimų prasme, todėl jį pasitelkus galima apdoroti itin didelius duomenų rinkinius. Užsienio autorių atliktame tyrime [63] buvo lyginami skirtingi LDA modelio parametrų įvertinimo metodai bei pastebėta, kad kiti autoriai pateikdami metodų modifikacijas ir pagerindami anksčiau paskelbtus rezultatus naudodavo skirtingas α ir η paremetrų reikšmes. [63] parodoma, kad LDA modelio rezultatai gerokai labiau priklauso nuo apriorinių Dirichlė skirstinių parametrų α ir η, negu nuo konretaus metodo, šiame darbe netyrinėjame skirtingų parametrų įvertimo metodų ir naudojame [58] pateiktą variacinio Bajeso metodo modifikaciją.

Adaptyvus laiko variacinis Bajeso metodas. [58] pateikiamas adaptyvus (angl. *online*) variacinis Bajeso metodas. Nors [56] pateiktas metodas turi pastovius atminties resursų reikalavimus

ir empiriškai konverguoja greičiau, negu Gibbs ėmimas [64], jo kiekvienoje jo iteracijoje reikia nuskaityti visą duomenų rinkinį. Dėl to jo skaičiavimai dideliems duomenų rinkiniams užimtų daug laiko. Taip pat, jo taikymas netinka kontekstui, kur nauji duomenys gaunami pastoviai ir reikia reguliariai atnaujinti modelio parametrus prisitaikant prie naujų duomenų.

$$\rho_t \triangleq (\tau_0 + t)^{-\kappa}, \quad \kappa \in (0.5, 1], \quad \tau_0 \geq 0 \tag{2.16}$$

$$\boldsymbol{\lambda} = (1 - \rho_t)\boldsymbol{\lambda} + \rho_t \widetilde{\boldsymbol{\lambda}} \tag{2.17}$$

Mini-paketai. Stochastinių metodų taikyme parametrų įverininimui (angl. *stochastic learning*) dažnai naudojama metodika prieš atnaujinant paremetrų reikšmes yra imti keletą stebėjimų, o ne vieną [65], [66].

$$\tilde{\lambda}_{kw} = \eta + \frac{D}{S} \sum_{s} n_{tsk} \phi_{tskw}, \quad S > 1 \tag{2.18}$$

kur n_{ts} yra s-asis dokumentas t-jame mini-pakete (angl. *mini-batch*). Variacinių parametrų ϕ_{ts} ir $gamma_{ts}$ įvertinimas šiam dokumentui nesikeičia ir tam naudojamos **(2.15)**, **(2.10)** išraiškos. Parinkus $S = D$ ir $\kappa = 0$ parametrų atnaujinimas būtų daromas pereinant visą duomenų rinkinį.

Matricinė išraiška iš [39].

$$F_{ij} = \exp(\Psi(\gamma_{ij})) \tag{2.19}$$

kur γ_{ij} yra variacinis temų parametras i-ajai temai ir j-ajam dokumentui.

$$\gamma_{ij} = \alpha_i + \sum_{w=1}^{M} \beta_{iw} C_{wj} / \sum_{i=1}^{k} \beta_{iw} F_{id} \tag{2.20}$$

kur C_{wj} yra w žodyno žodžio pasikartojimų skaičius j-ajame dokumente. Daumuma C_{ij} reikšmių yra nulinės, kadangi C yra labai išretinta matrica. M yra žodyno dydis, k yra temų skaičius. Matricinė **(2.20)** išraiška:

$$\gamma' = \alpha + F \circ \left(\beta \cdot \frac{C}{\beta^{\top} \cdot F} \right) \tag{2.21}$$

kur $\circ$ yra Ademaro sandauga, o trupmena C iš $\beta^{\top} \cdot F$ žymi matricų elementų dalybą pagal atitinkamus elementus.

2.3. SEGMENTAVIMO KOKYBĖS ĮVERTINIMO METRIKOS

Yra keletas būdų vertinti interneto reklamos vartotojų segmentavimo kokybę: galima imti klasterizavimo uždaviniuose dažnai naudojamas įvairias atstumo metrikas ir vertinti atstumus tarp klasterių ir tarp stebėjimų priklausančių tam pačiam klasteriui.

Jeigu klasterizavimo uždavinys yra didesnio uždavinio sudėtinė dalis – mūsų atveju vartotojų segmentavimo tikslas yra pateikti vartotojui jo interesus atitinkančias reklamas – klasterizavimo kokybę patariama vertinti jo rezulatus tiesiogiai susiejant su šiuo didesniu uždaviniu [53].

Kadangi [15] pateikiamas ne tik vartotojų atliktų užklausų duomenų rinkinys, bet jiems rodytų reklamų ir jų paspaudimų duomenys, segmentavimą yra tikslinga vertinti pagal jo rezultatų įtaką reklamų paspaudimams. Konkrečiau, įvertinsime, kokie reklamų paspaudimų rezultatai būtų, jeigu atitinkamos reklamos būtų rodomos suinteresuotiems segmentams priklausantiems vartotojams.

Toliau apibrėžiame metrikas, kurias naudosime interneto reklamos vartotojų efektyvumo vertinimui. [6] darbe pateikiamose naudojama dvejetainė funkcija $\delta(u_{ij})$, kurios reikšmė yra 1, jeigu j-asis vartotojas paspaudė i-ąją reklamą ir 0 kitu atveju. Mūsų turimų duomenų atveju, vartotojui ta pati reklama galėjo būti parodyta keletą kartų, kurią jis galėjo atitinkamą kiekį kartų. Todėl apsibrėšime keletą funkcijų, kurias naudosime nežymiai modifikuose metrikose [6] iš darbo.

Tarkime, $A = \{a_1, a_2, \dots, a_n\}$ yra n kardinalumo reklamų aibė duomenų rinkinyje. Vartotojai, kuriems buvo parodyta reklama a_i priklauso aibei $U_i = \{u_{i1}, u_{i2}, \dots, u_{m_i}\}$. Tada $\varphi(u_{ij})$ reikšmė yra i-osios reklamos parodymų skaičius j-ajam vartotojui, o $\delta(u_{ij})$ reikšmė yra i-sios reklamos paspaudimų skaičius, kurios atliko j-asis vartotojas.

Pritaikę LDA modelį paskirstome vartotojus į K segmentų. Tuomet funkcija

$$G(U_i) = \{g_1(U_i), g_2(U_i), \dots, g_K(U_i)\}, \quad i = 1, 2, \dots, n \tag{2.22}$$

kur $g_k(U_i)$ yra U_i vartotojai, priklausantys k-ajam segmentui, apibrėžia U_i vartotojų paskirstymą po segmentavimo. Tuomet k-ąjį vartotojų segmentą galima apibrėžti

$$g_k = \bigcup_{i=1}^{n} g_k(U_i) \tag{2.23}$$

Reklamos paspaudimų ir parodymų santykio rodiklis (toliau darbe CTR – *angl.* Click-Through Rate) yra dažnai naudojamas charaktistika įvertinant interneto reklamos našumą. Reklamos a_i CTR reikšmė yra jos paspaudimų ir parodymų santykis

$$CTR(a_i) = \frac{\sum_{j=1}^{m_i} \delta(u_{ij})}{\sum_{j=1}^{m_i} \varphi(u_{ij})} \tag{2.24}$$

kur m_i yra vartotojų, kuriems buvo parodyta reklama a_i, skaičius.

Po interneto reklamos vartotojų segmentavimo, reklamos a_i CTR rodiklis g_k vartotojų segmentui pažymime

$$CTR(a_i|g_k) = \frac{\sum_{u_{ij} \in g_k(U_i)} \delta(u_{ij})}{\sum_{u_{ij} \in g_k(U_i)} \varphi(u_{ij})}. \tag{2.25}$$

Reklamos a_i santykinis CTR rodiklio pagerėjimas,

Santykinis CTR pagerėjimas reklamai a_i apibrėžiamas

$$\Delta(a_i) = \frac{CTR(a_i|g_*(a_i)) - CTR(a_i)}{CTR(a_i)} \tag{2.26}$$

kur $g_*(a_i) = \arg\max_{g_k} CTR(a_i|g_k), \quad k = 1, 2, \ldots, K.$

Nors CTR rodiklio pagerėjimas suteikia daug informacijos apie segmentavimo efektyvumą, norint pateikti įtikinamas išvadas vien juo pasikliauti negalima. Segmentas g_k, tenkinantis $CTR(a_i|g_k) > CTR(a_i)$ parodo, kad šio segmento vartotojai yra labiau suinteresuoti a_i reklama, negu kiti vartotojai. Tačiau tai negarantuoja, kad į g_k segmentą buvo priskirta kiek galima daugiau vartotojų, kurie galima paspaus a_i [6].

Toliau pateikiamos klasifikavimo metodams taikomos efektyvumo vertinimo metrikos, kurios pritaikomos interneto reklamos vartotojų segmentavimo įvertinimui.

Jeigu reklamos paspaudimai įvardinami kaip teigiami stebėjimai, o reklamos parodymai be paspaudimų kaip neigiami, CTR rodiklio pagerėjimas atitinka tikslumą **(2.27)**, tačiau nieko nepasako apie atkuriamumą **(2.28)**.

Tikslumas (P) interneto vartotojų segmentavimo kontekste apibūdina į kokią dalį reklamos a_i parodymų k-ojo segmento vartotojams jie sureaguos paspaudimais. Tai atitinka CTR rodiklį g_k vartotojų segmentui:

$$P(a_i|g_k) = CTR(a_i|g_k). \tag{2.27}$$

Atkuriamumas (R) parodo kokia dalis iš reklamos a_i paspaudimų atlieka g_k segmentui priskirti vartotojai:

$$R(a_i|g_k) = \frac{\sum_{u_{ij} \in g_k(U_i)} \delta(u_{ij})}{\sum_{j=1}^{m_i} \delta(u_{ij})}. \tag{2.28}$$

Kuo didesnė R reikšmė, tuo geriau vartotojų segmentas g_k padengia a_i reklamą spaudžiančius vartotojus. Tikslumą ir atkuriamumą apjungia F-matas [67].

F-matas (F) yra tikslumo ir atkuriamumo harmoninis vidurkis:

$$F(a_i|g_k) = \frac{2 \cdot P(a_i|g_k) \cdot R(a_i|g_k)}{P(a_i|g_k) + R(a_i|g_k)} \tag{2.29}$$

Geriausią segmentą reklamai a_i galima vertinti ne tik pagal jo CTR rodiklį (tikslumą), bet ir pagal atkuriamumą bei F-matą. Tuomet geriausias segmentas reklamai a_i gali parenkamas pagal vieną iš metrikų:

$$g_*^{(P)}(a_i) = \arg\max_{g_k} P(a_i|g_k), \quad k = 1, 2, \ldots, K \tag{2.30}$$

$$g_*^{(R)}(a_i) = \arg\max_{g_k} R(a_i|g_k), \quad k = 1, 2, \dots, K \tag{2.31}$$

$$g_*^{(F)}(a_i) = \arg\max_{g_k} F(a_i|g_k), \quad k = 1, 2, \dots, K \tag{2.32}$$

Darbuose [6], [12], [11] neatsižvelgiama į tai, jog geriausią segmentą parenkant pagal tikslumą, reklamai galima parinkti segmentą, kuriame yra vienas vartotojas, jam reklama buvo parodyta vieną kartą ir jis ją paspaudė. Tuomet segmento CTR rodiklis (tikslumas) bus lygus 1 ir jeigu reklmos CTR rodiklis yra mažas, santykinis CTR pagerėjimas $\Delta(a_i)$ bus itin didelis [10].

Todėl įvedami apribojimai, kurių netenkinantys segmentai negali būti parinkti geriausiais reklamai a_i. Toliau apibėžiamas [10] pasiūlytas apribojimas, kuriuo atmetami segmentai su mažiau negu $1/K$ vartotojų.

Įvedus apribojimus geriausias segmentas, parinktas be apribojimų, žymimas $g_*^{(\tau,1)}(a_i)$, kur $\tau \in \{P, R, F\}$.

Apribojimas pagal segmento vartotojų skaičių.

$$\begin{gathered} g_*^{(\tau,2)}(a_i) = \arg\max_{g_k} \tau(a_i|g_k), \quad k = 1, 2, \dots, K, \quad g_k \in \tilde{g}_k, \\ \tilde{g}_k = \{g_k|\, |g_k(U_i)|/m_i \geq 1/K\} \end{gathered} \tag{2.33}$$

kur $\tau \in \{P, R, F\}$, $|g_k(U_i)|$ yra reklamos a_i vartotojų skaičius.

Verta pastebėti, kad [10] vartotojai segmentuojami į 5, 10, ir 20 segmentų ir nebuvo patikrinta ar šis apribojimas yra naudingas kai segmentų kiekis K yra dešimtis kartų didesnis.

Jeigu reklamoms geriausiais segmentai parinkami segmentai su keletu varotojų ir vienetine tikslumo reikšme, vidutinės tikslumo ir santykinio CTR pagerėjimo reikšmės ženkliai didėja, tačiau gaunama maža atkuriamumo reikšmė. Toliau pateikiamas apribojimas tiesiogiai susiejamas su atkuriamumu.

Apribojimas pagal atkuriamumą.

$$\begin{gathered} g_*^{(\tau)}(a_i) = \arg\max_{g_k} \tau(a_i|g_k), \quad k = 1, 2, \dots, K, \quad g_k \in \tilde{g}_k, \\ \tilde{g}_k = \left\{g_k|\, R(g_k) \geq \overline{R(G(U_i))}\right\} \end{gathered} \tag{2.34}$$

kur $\tau \in \{P, R, F\}$, $\overline{R(G(U_i))}$ yra reklamos a_i vartotojų segmentų atkuriamumo aritmetinis vidurkis.

3. TYRIMŲ REZULTATAI IR JŲ APTARIMAS

Sudarytą LDA modelį šioje dalyje taikome [15] duomenų rinkiniui ir sprendžiame interneto vartotojų segmentavimo uždavinį.

3.1. INTERNETO REKLAMOS VARTOTOJŲ SEGMENTAVIMO UŽDAVINIO SPRENDIMO METODIKA

Duomenų rinkinys ir jo paruošimas analizei

Tyrimui atlikti panaudoti [15] pateikti realūs interneto vartotojų paieškos užklausų, jiems pateiktų reklamų ir jų paspaudimų duomenys. Duomenys yra anonimizuoti ir interneto reklamos vartotojų užklausų žodžių reikšmės pateikiamos skaičiais, todėl negalima taikyti teksto analizės metodų.

Atliktas duomenų filtravimas ir pagal žemiau pateikiamas sąlygas pašalinti stebėjimai:

- Neidentifikuoti vartotojai.
- Vartotojai, paspaudę reklamas daugiau negu 30 kartų, kurių CTR > 0.9.
- Vartotojai, pateikę mažiau negu 4 užklausas ir nepaspaudę nė vienos reklamos.
- Reklamos, kurias paspaudė mažiau negu 3 vartotojai.

Po šių stebėjimų pašalinimo duomenų rinkinyje palikta:

- 66 462 376 reklamų parodymų ir paspaudimų įrašai.
- $D = 5\ 647\ 787$ skirtingi vartotojai.
- $n = 105\ 745$ skirtingos reklamos.

Kryžminis patikrinimas.

Sudaryto LDA modelio paramerų įvertinimui taikomas 5-lypis kryžminis patikrinimas: modelis apmokomas su 80% duomenų, apmokytu modeliu segmentai priskiriami likusiam 20% vartotojų. Taip kartojama 5 kartus, kad kiekvienam vartotojui segmentas būtų priskirtas modeliu, kurio parametrai buvo įvertinti panaudojant duomenų pjūvį, kuriame šio vartotojo nebuvo.

Vartotojų segmentavimas. Panaudojant apmokytą modelį įvertinamas d-tojo vartotojo interesų tikimybių vektorius $\theta_{\cdot,d}$. d-asis vartotojas prikiriamas segmentui g_k, $k \in \{1, 2, \dots, K\}$, kur $k = \arg\max_k \theta_{k,d}$, segmentui, kurio tikimybė didžiausia.

3.2. METODIKOS IR PROGRAMINIŲ PRIEMONIŲ TAIKYMAS VARTOTOJŲ SEGMENTAVIMO UŽDAVINIO SPRENDIMUI

Toliau atliekame vartotojų segmentavimą ir jo efektyvumo tyrimą.

Tyrimo eiga. LDA modelis apkomomas panaudojant kryžminio patikrinimo metodą. Interneto vartotojų pasieškos užklausų duomenų rinkinys padalinamas į 5 imtis ir apmokomi 5 modeliai kiekvieno iš jų apmokymui naudojant 4 iš 5 imčių. Kiekvieną kartą vienai likusiai vartotojų imčiai panaudojant apmokytą modelį priskiriami segmentų indeksai k, $k \in \{1, 2, ... K\}$.

Naudojami segmentų kiekiai $K \in 10, 20, 30, 45, 70, 110, 160, 240, 350, 800$.

Atlikus vartotojų segmentavimą įvertinamas jo efektyvumas. Tam naudojamos santykinio CTR pagerėjimo, tikslumo, atkuriamumo ir F-mato metrikos. Kiekvienai iš šių metrikų vidurkių pateikiami atskiri grafikai su 95% pasikliovimo lygmens pasikliautinaisias intervalais. Vidutinio patikimimo ir vidutinės F-mato reikšmės juoda linijagrafikuose palyginimui pateikiamos vidutinės šių metrikų reikšmės, gautos neatliekiant vartotojų segmentavimo. Atkuriamumo reikšmė neatlikus vartotojų segmentavimo yra 1 (100%).

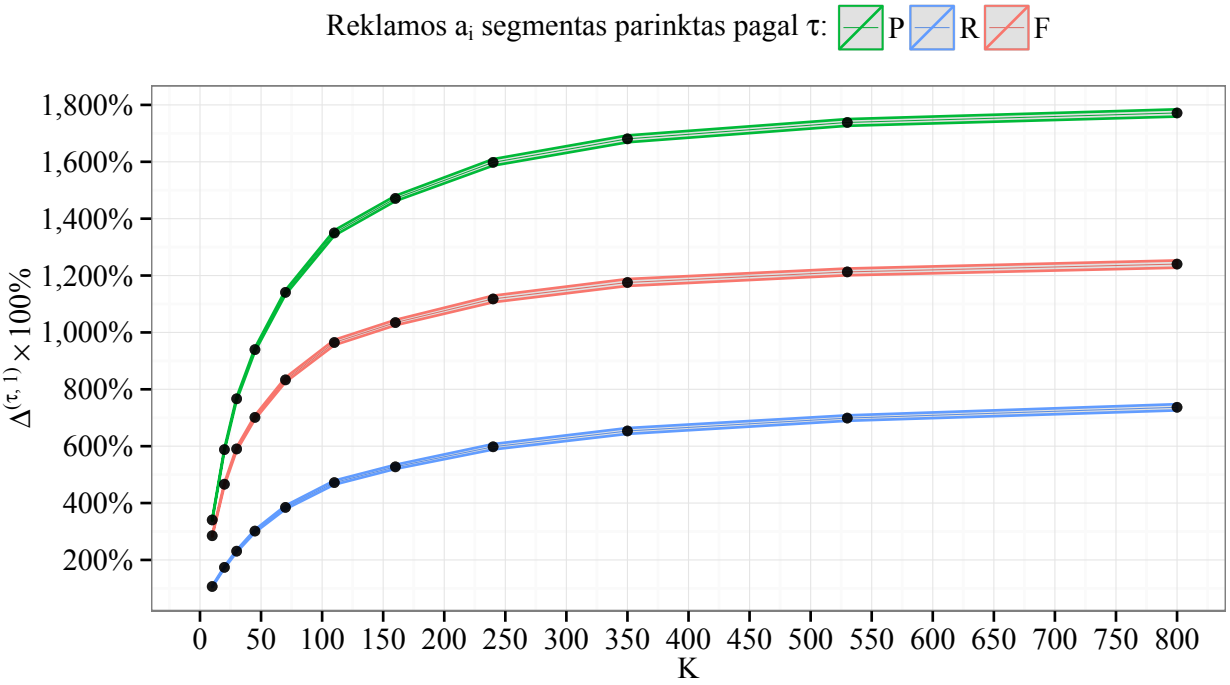
Geriausi segmentai su kiekvienu K kiekvienai reklamai a_i, $i \in \{1, 2, ... n\}$ parenkami pagal dižiausias tikslumo atkuriamumo arba F-mato reikšmes.

Taip pat pateikiami reklamų dvimačių pasiskirtymų pagal atkuriamumo ir tikslumo metrikas grafikai, kai geriausias segmentas kiekvienai iš jų parenkami pagal tikslumo metiką ir pagal F-mato metriką. Grafikai braižomi segmentų kiekiams $K \in 530, 800$. Kadangi duomenų rinkinyje yra 105 745 reklamos, pasiskirstymų vizualizuojami reklamų tankio įverčiai, gauti branduolinio tankio įvertiniu, o ne braižomos sklaidos diagramos. Kartu pateikiamos atkuriamumo ir tikslumo marginalios histogramos.

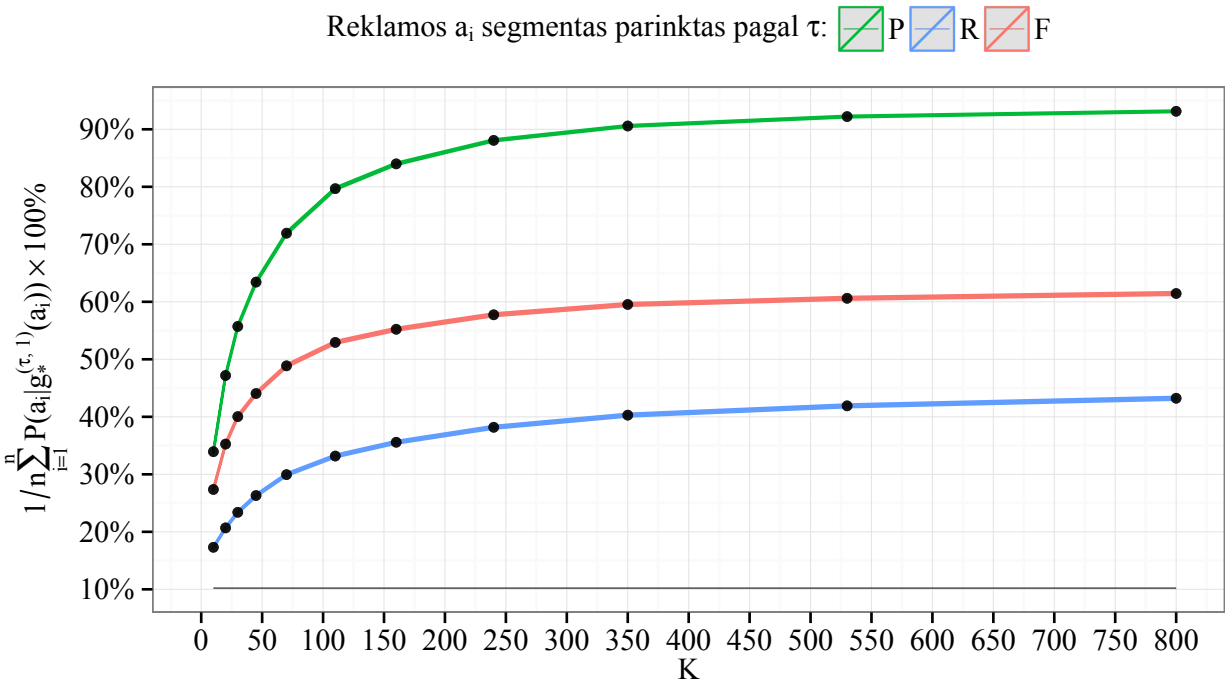
Geriausi segmentai parenkami:

- **be apribojimų** – geriausias segmentas kiekvienai reklamai parenkamas iš visų jos segmentų;
- **taikant apribojimą pagal segmento varototjų skaičių;**
- **taikant apribojimą pagal atkuriamumą (R).**

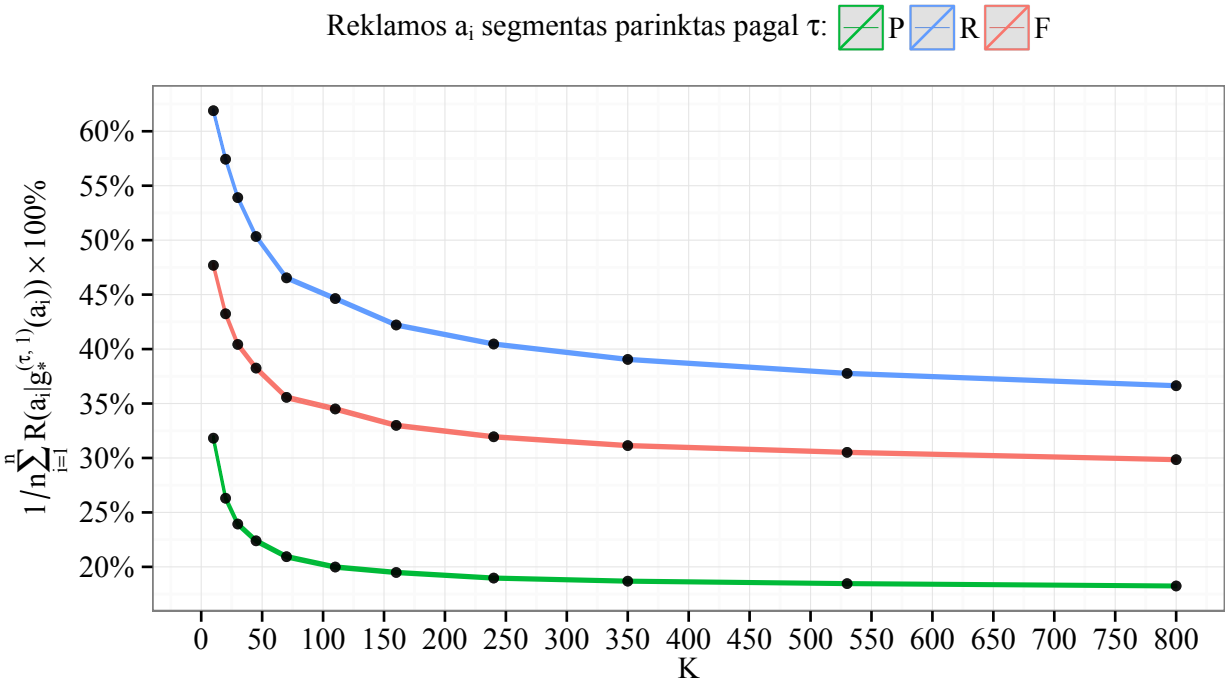
Toliau pateikiami šio tyrimo rezultatai.



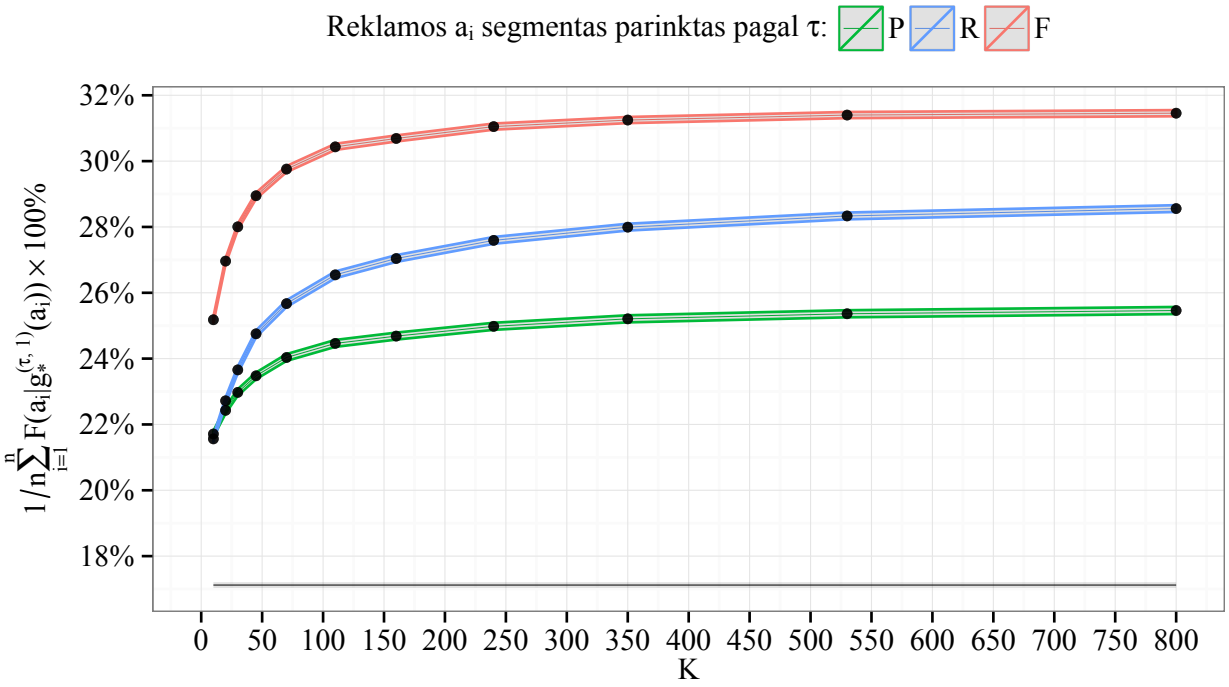
3.1 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K be apribojimų.



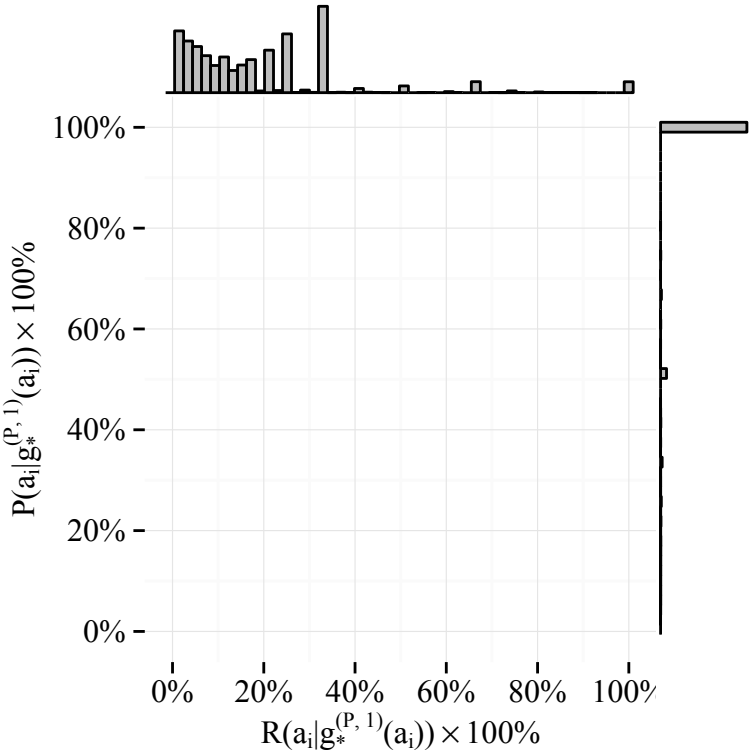
3.2 pav. Vidutinio tikslumo (P) priklausomybė nuo segmentų kiekio K be apribojimų.



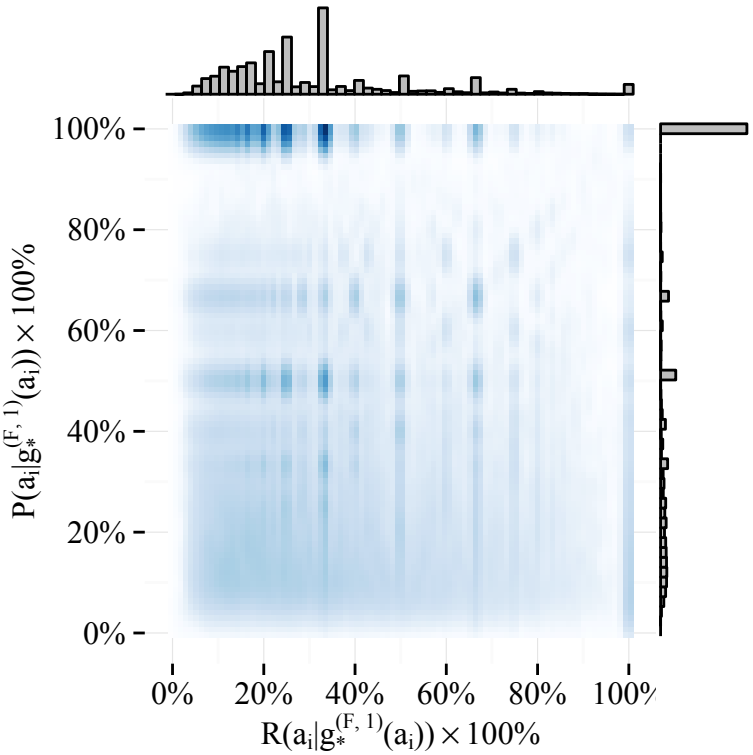
3.3 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K be apribojimų.



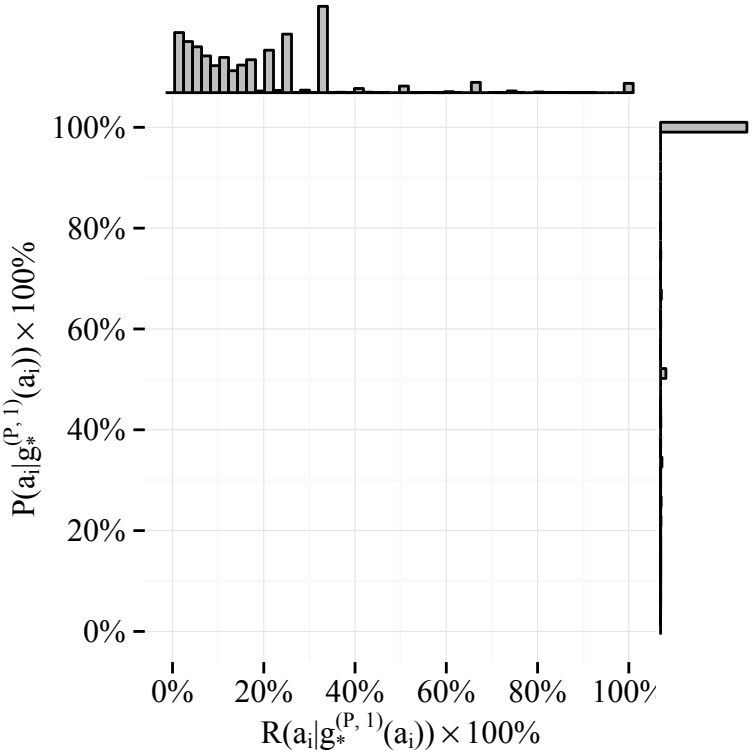
3.4 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K be apribojimų.



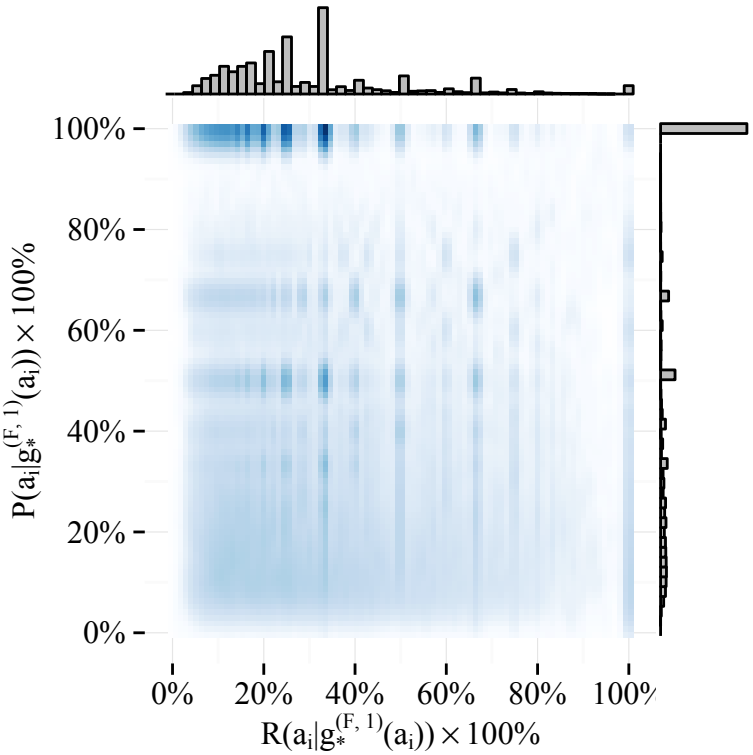
3.5 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 530$**.**



3.6 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 530$**.**



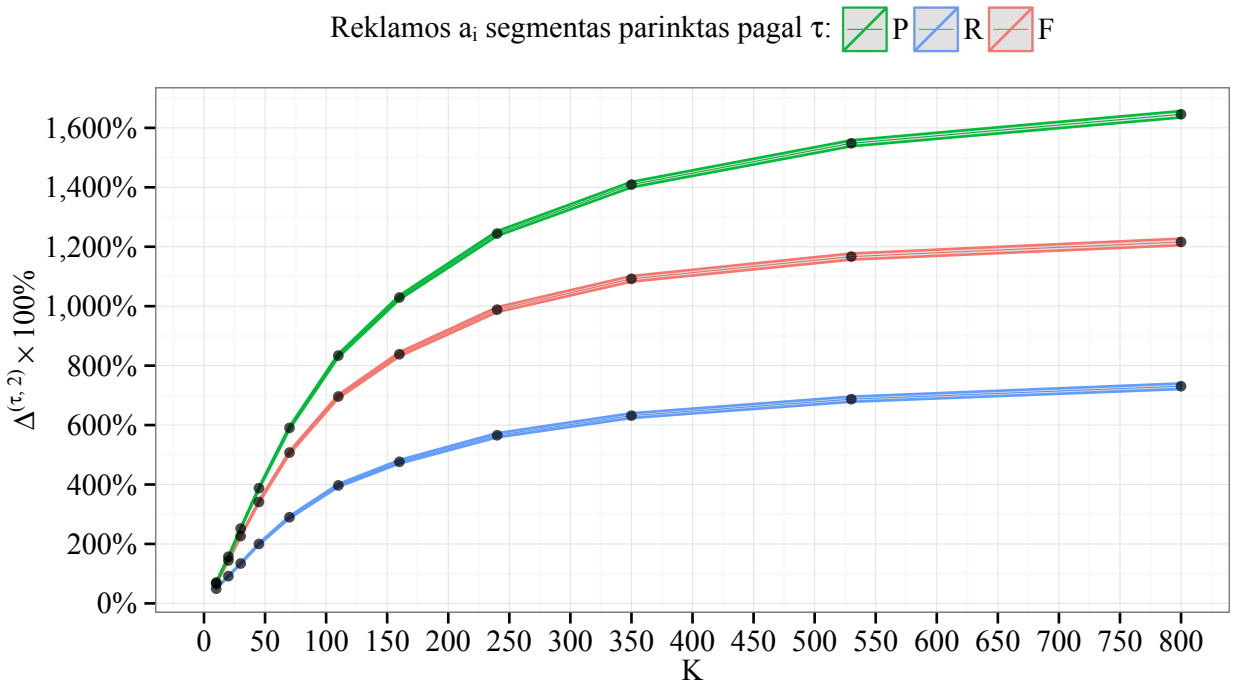
3.7 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 800$**.**



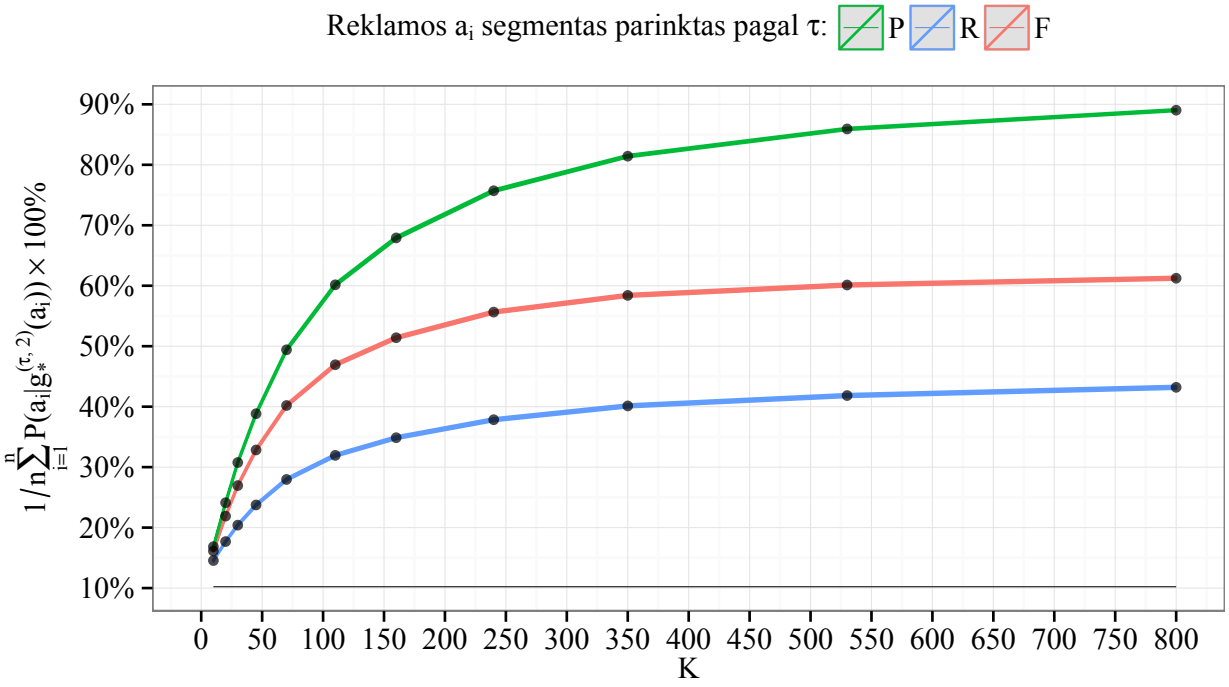
3.8 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 800$**.**

Pateiktuose 3.1 – 3.8 paveiksluose matoma, kad geriausią segmentą parenkant be apribojimų

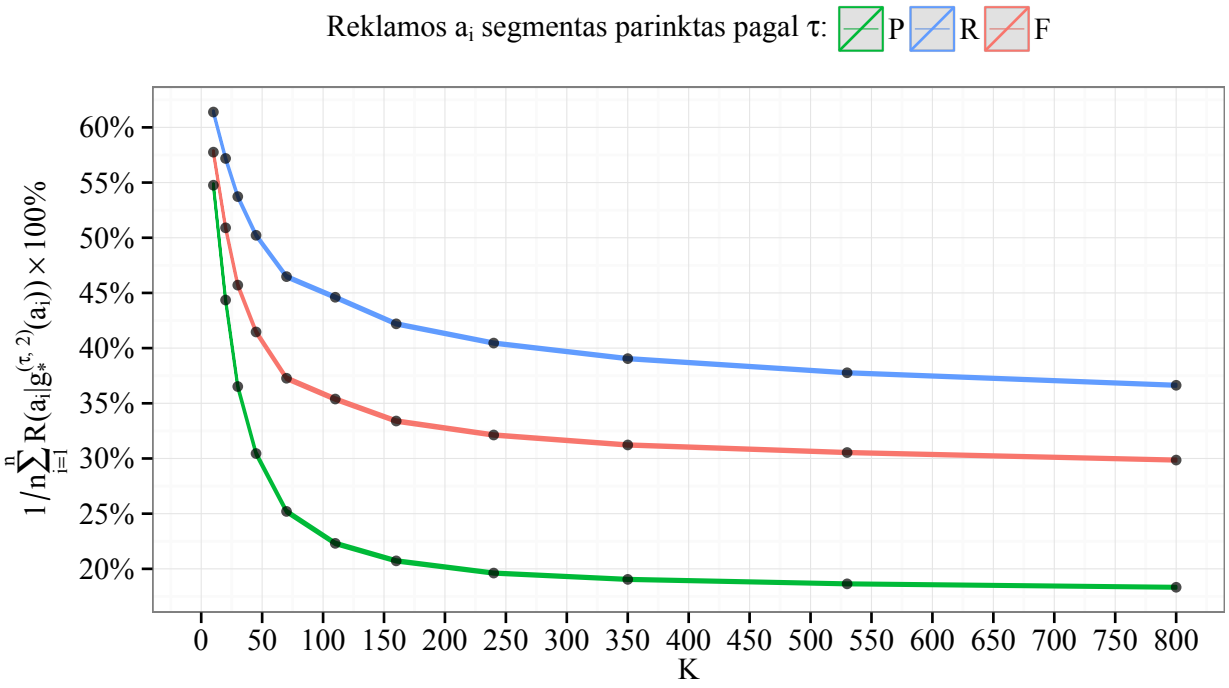
pagal tikslumą jau su $K = 230$ vidutinis CTR pagerėjimas su 95% pasikliovimo lygmeniu patenka į intervalą [1586.55%; 1609.42%], tačiau vidutinė atkuriamumo reikšmė yra mažesnė negu 20%. Marginaliose histogramose matoma, kad beveik daugumos segmentų tiklumo reikšmė yra 100% arba arti 100%, tačiau taip pat matoma, kodėl gaunamos toks atkuriamumas – nemažos dalies segmentų atkuriamumo reikšmė mažesnė už 10%.



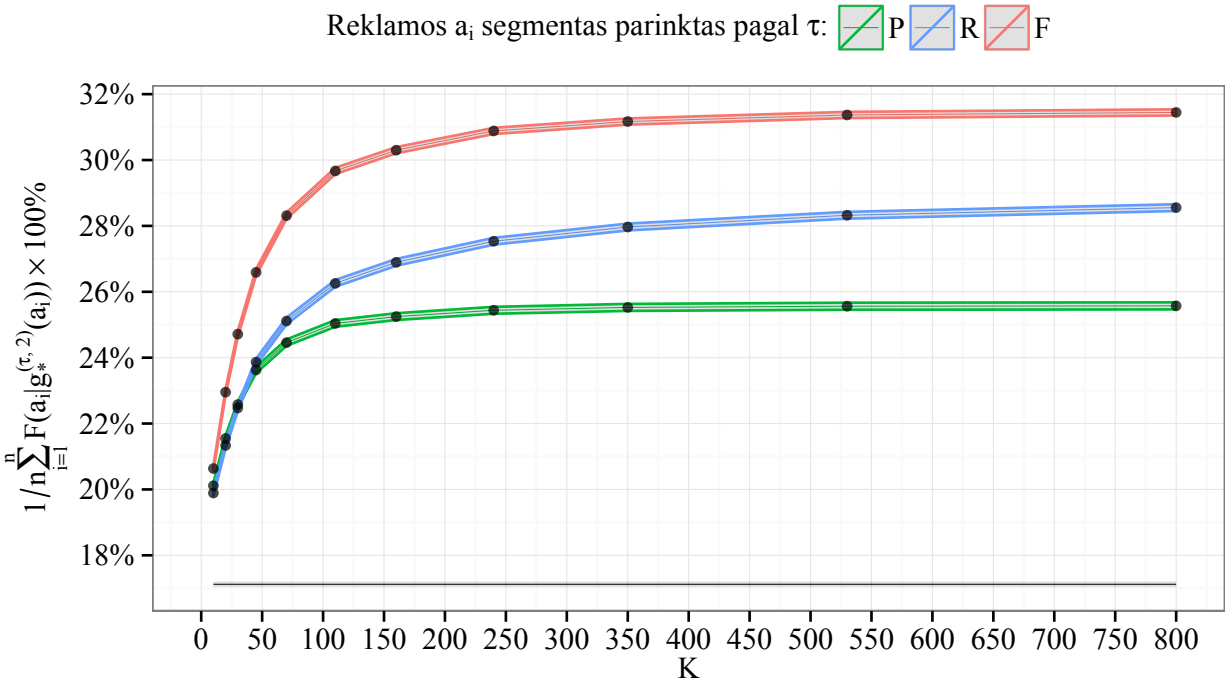
3.9 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K (su apribojimu pagal segmento vartotojų skaičių).



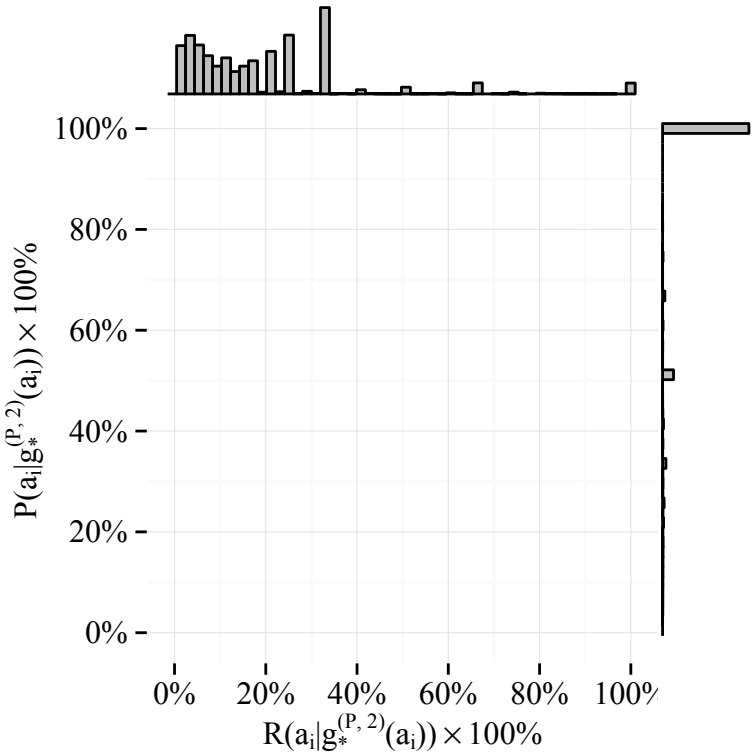
3.10 pav. Vidutinio tikslumo (P) priklausomybė nuo segmentų kiekio K„ kai taikomas apribojimas pagal segmento vartotojų skaičių.



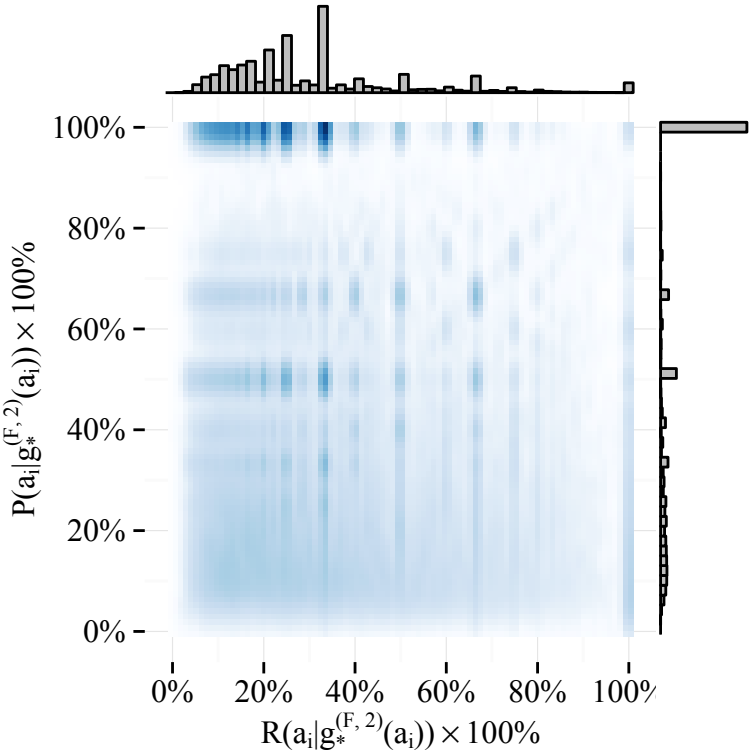
3.11 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



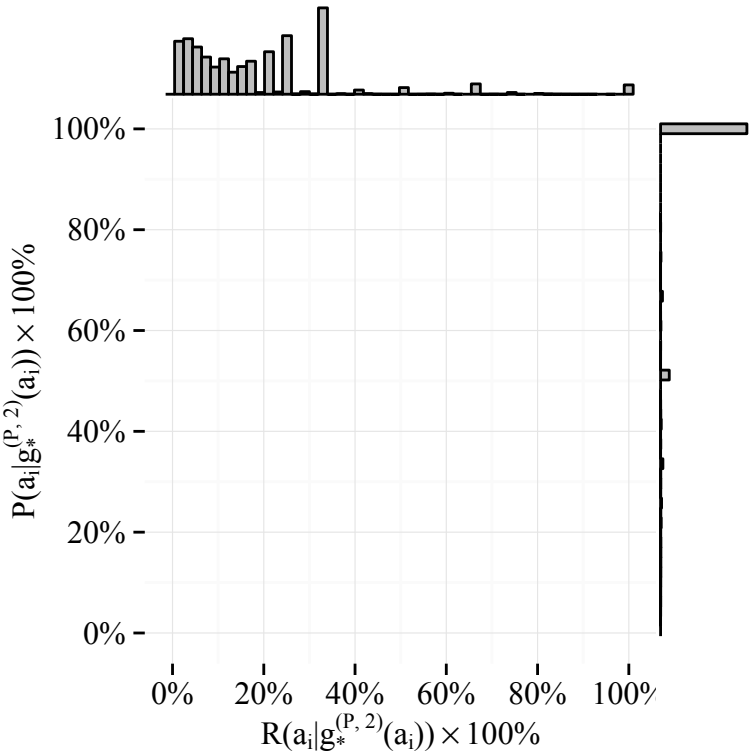
3.12 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



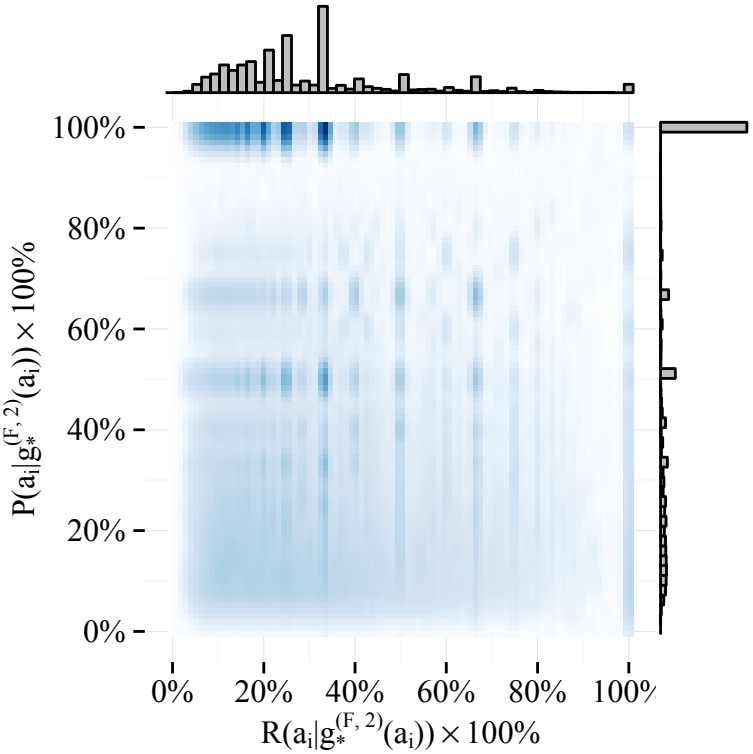
3.13 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 530$.



3.14 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 530$.

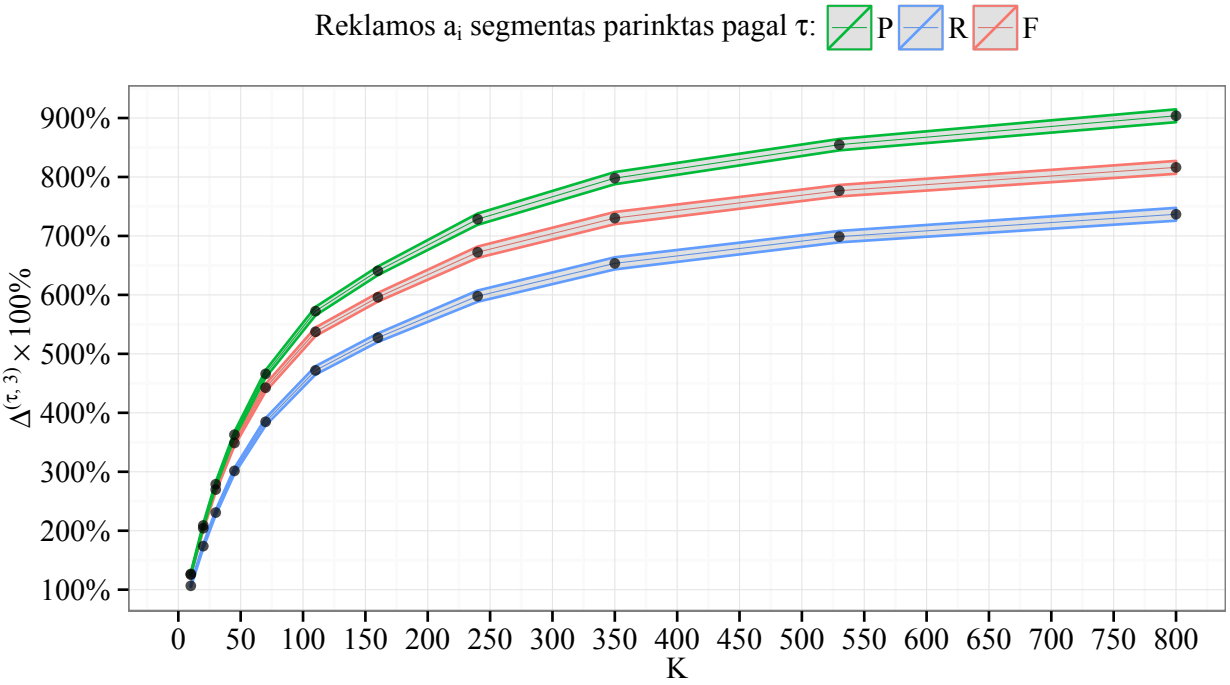


3.15 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 800$.

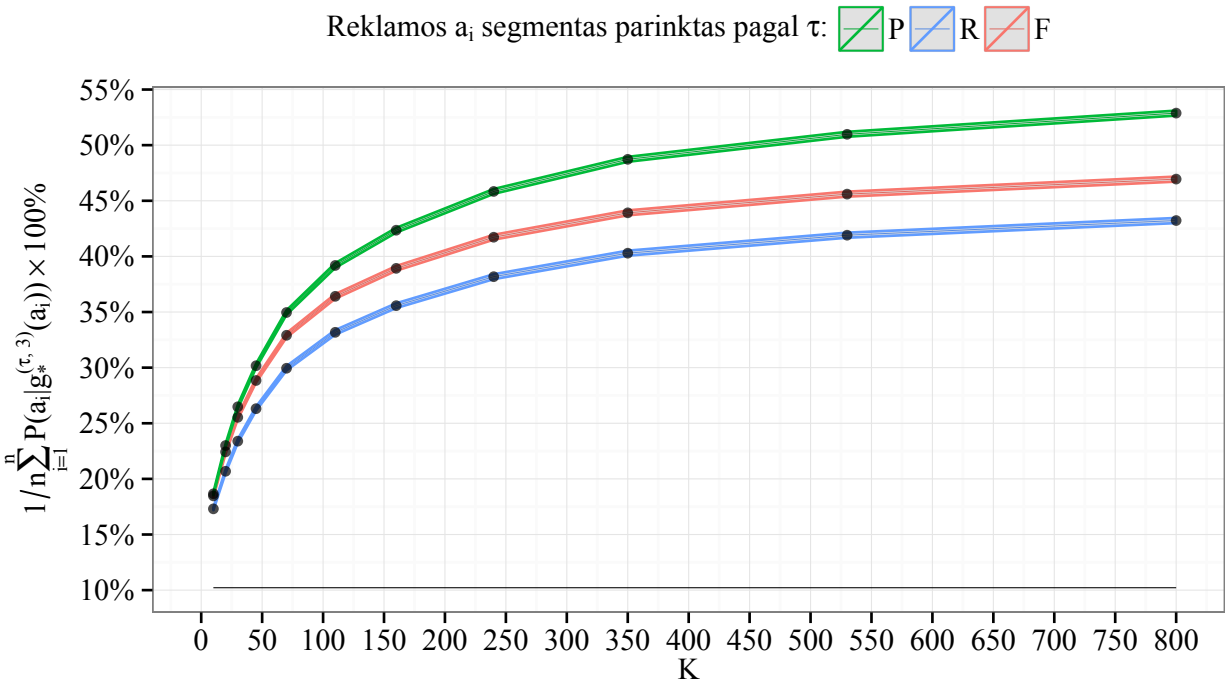


3.16 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 800$**.**

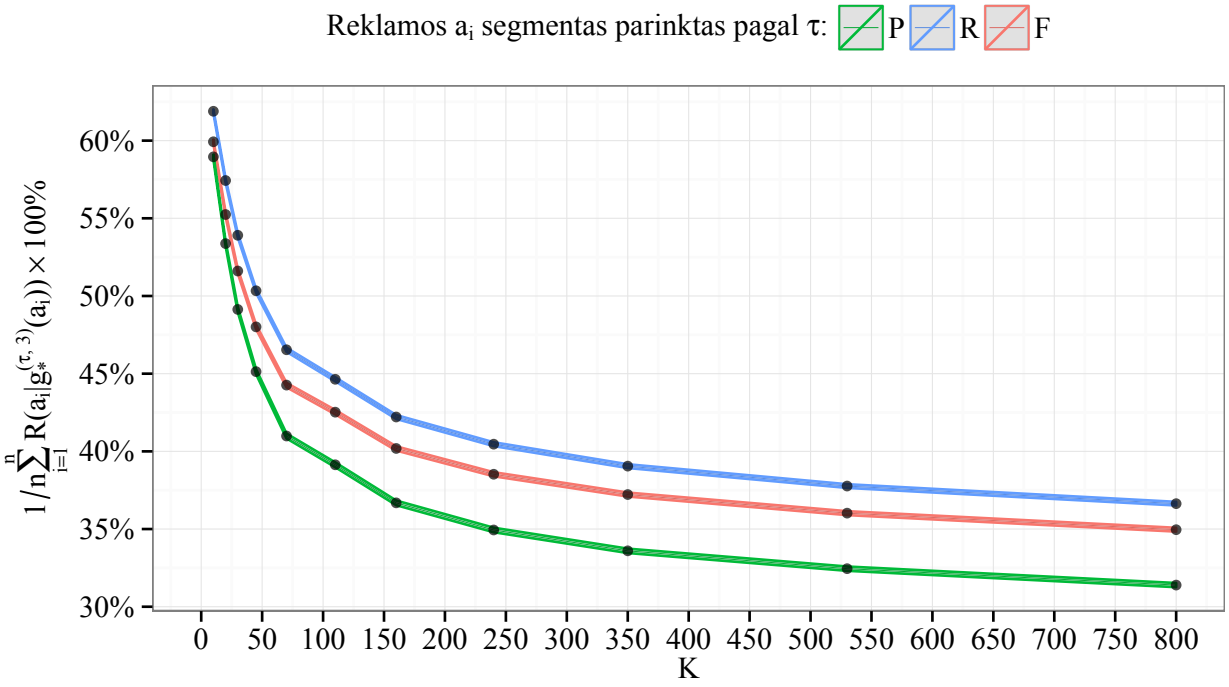
Paveiksluose 3.9 – 3.16 matoma, kad kai segmentų skaičius K yra didesnis, negu vartotojų skaičius, kuriems buvo parodyta reklama a_i, rezultatai yra panašūs į gaunamus, kuomet geriausi segmentai parenkami be apribojimų.



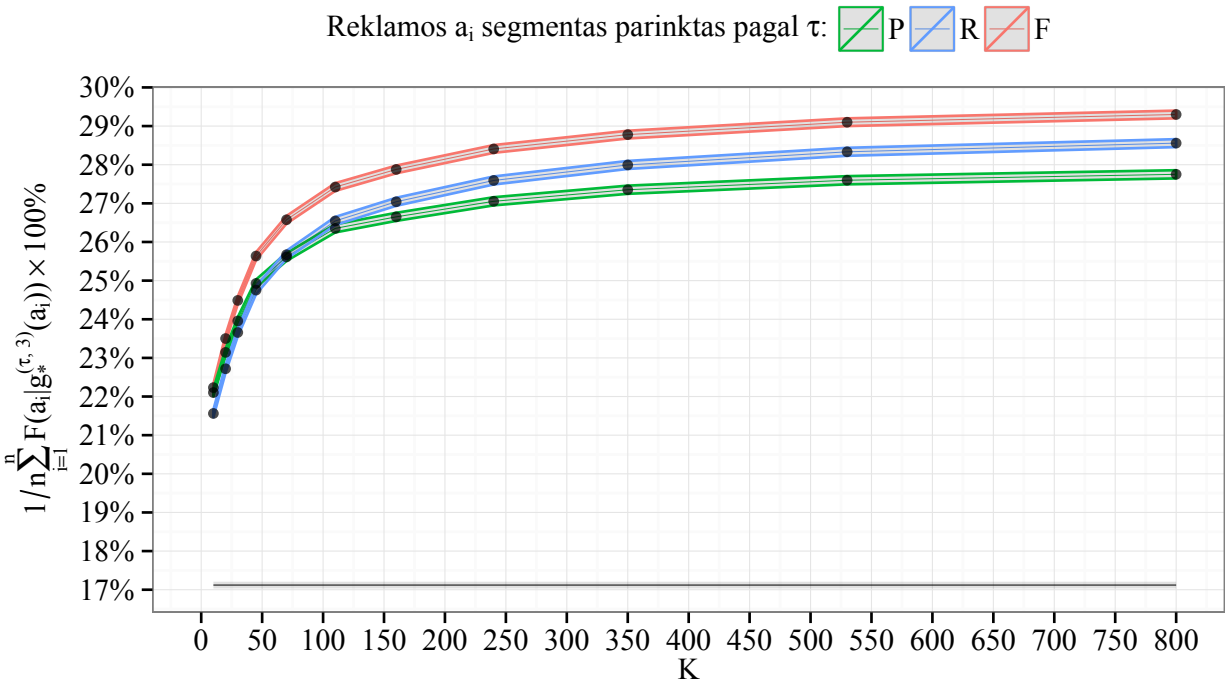
3.17 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



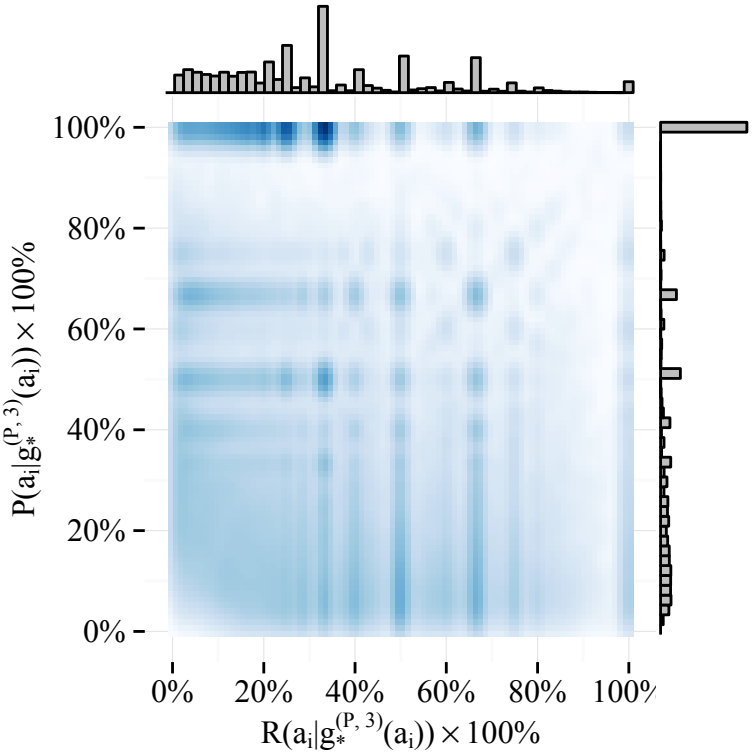
3.18 pav. Vidutinio tikslumo (P) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



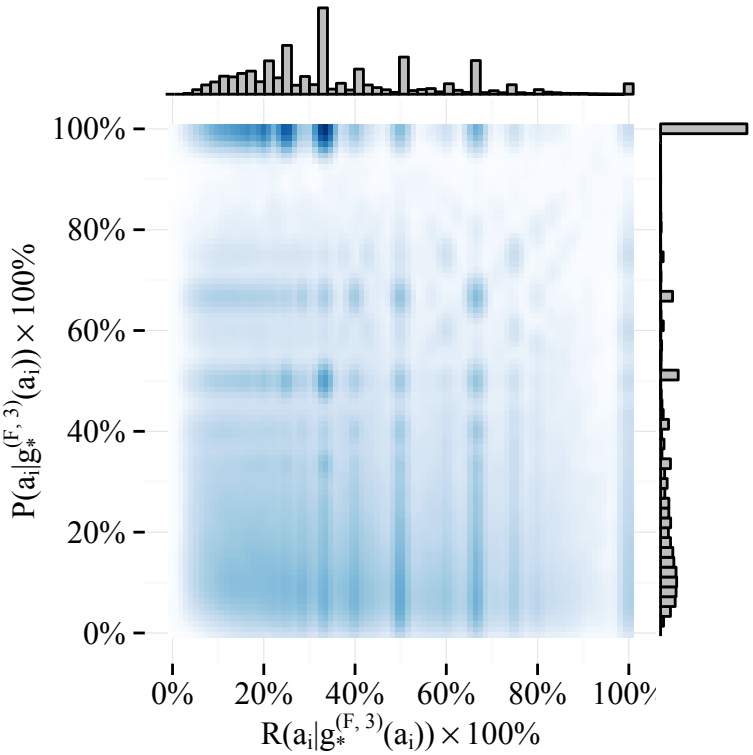
3.19 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



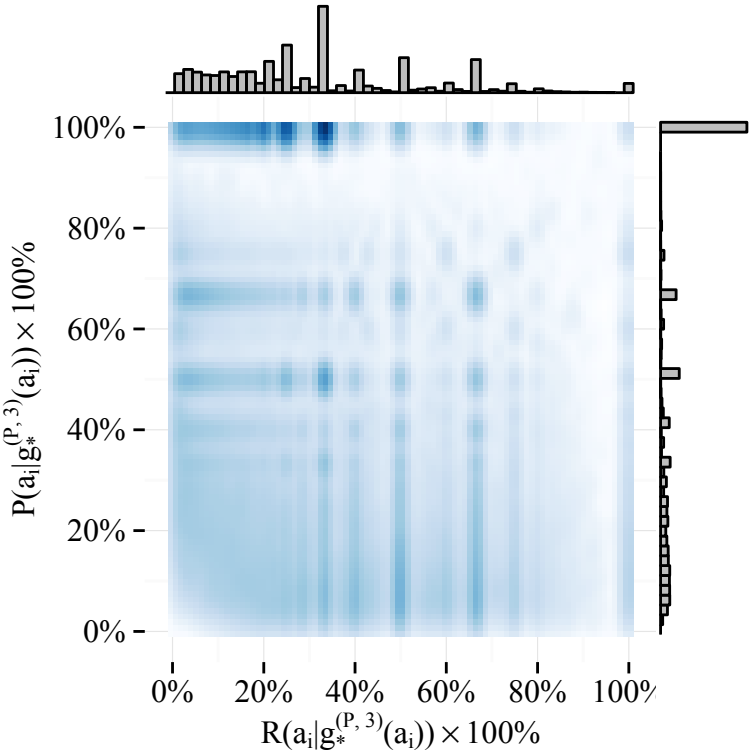
3.20 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



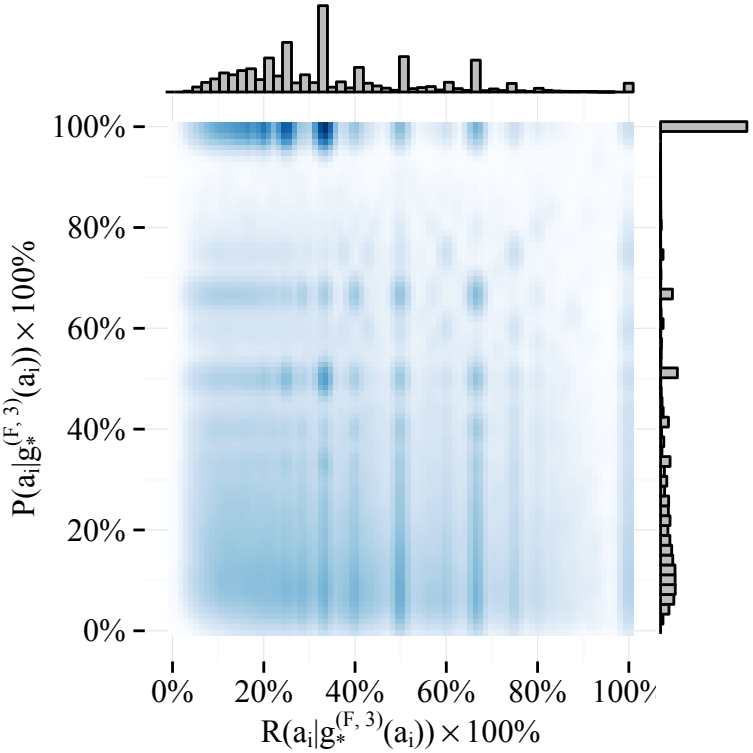
3.21 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 530$**.**



3.22 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai kai, taikomas apribojimas pagal atkuriamumą (R), $K = 530$**.**



3.23 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 800$**.**



3.24 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 800$**.**

Paveiksluose 3.17 – 3.24 pateikiami rezultatai yra gauti taikant apribojimą pagal atkuriamu-

mą. Kai $K = 800$ ir segmentai parenkami pagal tikslumo metriką, vidutinė atkuriamumo reikšmė su 95% pasikliovimo lygmeniu patenka į intervalą [31.26%; 31.53%], o vidutinis tikslumo reikšmė patenka į intervalą [52.67%; 53.10%].

Pastebėta, kad netaikant apribojimų ir geriausią segmentą parenkant pagal F-matą, kai $K = 800$ su 95% pasikliovimo lygmeniu vidutinė tikslumo reikšmė patenka į intervalą [61.22%; 61.67%], atkuriamumo reikšmė patenka į intervalą [29.73%; 29.96%]. Tokiu būdu parenkant geriausią segmentą reklamoms, lyginant su segmento paėmimu pagal tikslumą taikant apribojimą pagal atkuriamumą, pirmuoju atveju daugiau reklamų turi mažesnę atkuriamumo reikšmę, bet didesnę tikslumo reikšmę.

Nors interpretuojant rezultatus dažnai F-matas yra mažiau naudinga metrika, negu atsikirai tikslumas ir atkuriamumas iterpretuojami kartu, pastebėta, kad F-matas yra pakankamai gera metrika geriausio segmento parinkimui.

IŠVADOS

1) Atlikus vartotojų elgsena internete pagrįstos reklamos literatūros analizę, nustatyta, kad vartotojų segmentavimo uždavinio sprendimui dažnai taikomi klasikiniai klasterizavimo metodai, tačiau geresni rezultatai pasiekiami teksto turinio analizės modeliais.
2) Sudarytas latentinio Dirichlė paskirstymo modelis, leidžiantis pagal vartotojų atliktų interneto paieškos užklausų žodžius įvertinti vartotojo interesus, kurie naudojami interneto reklamos vartotojų segmentavimui.
3) Atlikus didžiųjų duomenų rinkinių analizės programinės įrangos analizę nustatyta, kad yra didelis pasirinkimas, tačiau skirtinga programinė įranga efektyviai sprendžia skirtingas problemas, todėl darbe panaudotos kelios programinės priemonės. Latentinio Dirichlė paskirstymo modelio, įvertinančio vartotojo interesus, realizacijai pasirinkta *BIDMach* biblioteka dėl efektyvaus operatyvios atminties panaudojimo ir didelės modeliavimo spartos. Paskirstytiems skaičiavimams ir interaktyvumo palaikymui pasirinkta *Apache Spark* platforma, kurią naudojant buvo atliktas pirminis duomenų apdorojimas ir sukurta segmentavimo efektyvumo įvertinimo taikomoji programa. Tyrimų automatizavimui panaudota *Python* programavimo kalba, o segmentavimo efektyvumo rezultatų analizei – statistinis paketas *R*.
4) Panaudojus sukurtą modelį, programines priemones ir realius anonimizuotus interneto vartotojų paieškos užklausų, reklamų parodymų ir paspaudimų duomenis atliktas tyrimas ir nustatyta, kad didinant segmentų skaičių didėja vidutinis reklamų santykinio paspaudimų ir parodymų rodiklis, vidutinė tikslumo metrikos reikšmė didėja greičiau, negu mažėja vidutinė atkuriamumo metrikos reikšmė, todėl rekomenduojama parinkti kuo didesnį segmentų kiekį atsižvelgiant į toleruotiną atkuriamumo praradimą. Pavyzdžiui, su 95% pasikliautinumo lygmeniu vidutinė tikslumo metrikos reikšmė 160 segmentų yra intervale [42.14%; 42.57%], 800 segmentų – [52.67%; 53.10%], vidutinė atkuriamumo metrikos reikšmė 160 segmentų yra intervale [36.54%; 36.83%], 800 segmentų – [31.26%; 31.53%], kai segmentai parenkami nustatant mažiausios R metrikos reikšmės apribojimą. Tikslumo metrikos reikšmė nepritaikius vartotojų segmentavimo yra intervale [10.15%; 10.28%].

LITERATŪROS SĄRAŠAS

1 IABEUROPE.EU. IAB Europe AdEx Benchmark 2014 results [interaktyvus]. Brussels, 2015 [žiūrėta 2015-06-02]. Prieiga per internetą: ⟨http://www.iabeurope.eu/research-and-papers/iab-europe-adex-benchmark-2014-resul⟩.

2 IAB.NET. IAB Internet Advertising Revenue Report conducted by PricewaterhouseCoopers (PWC) [interaktyvus]. New York, 2015 [žiūrėta 2015-06-02]. Prieiga per internetą: ⟨http://www.iab.net/AdRevenueReport⟩.

3 FAIN, D. C.; PEDERSEN, J. O. Sponsored search: A brief history. *Bulletin of the American Society for Information Science and Technology*. 2006, t. 32, nr. 2, psl. 12–13. ISSN 1550-8366.

4 BRODER, A.; FONTOURA, M.; JOSIFOVSKI, V.; RIEDEL, L. A Semantic Approach to Contextual Advertising. *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Amsterdam, The Netherlands: ACM, 2007, psl. 559–566. SIGIR '07. ISBN 978-1-59593-597-7.

5 GEYIK, S. C.; DASDAN, A.; LEE, K. User Clustering in Online Advertising via Topic Models. *CoRR*. 2015, t. abs/1501.06595.

6 YAN, J.; LIU, N.; WANG, G. ir kt. How Much Can Behavioral Targeting Help Online Advertising? *Proceedings of the 18th International Conference on World Wide Web*. Madrid, Spain: ACM, 2009, psl. 261–270. WWW '09. ISBN 978-1-60558-487-4.

7 TAPAS, K.; MOUNT, D.; NETANYAHU, N. ir kt. An efficient k-means clustering algorithm: analysis and implementation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*. 2002, t. 24, nr. 7, psl. 881–892. ISSN 0162-8828.

8 RASMUSSEN, M.; DESHPANDE, M.; KARYPIS, G. ir kt. wCLUTO: A Web-Enabled Clustering Toolkit. *Plant Physiology*. 2003, t. 133, psl. 511.

9 INDYK, P.; MOTWANI, R. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*. Dallas, Texas, JAV: ACM, 1998, psl. 604–613. STOC '98. ISBN 0-89791-962-9.

10 WU, X.; YAN, J.; LIU, N. ir kt. Probabilistic Latent Semantic User Segmentation for Behavioral Targeted Advertising. *Proceedings of the Third International Workshop on Data Mining and Audience Intelligence for Advertising*. Paris, France: ACM, 2009, psl. 10–17. ADKDD '09. ISBN 978-1-60558-671-7.

11 TU, S.; LU, C. Topic-Based User Segmentation for Online Advertising with Latent Dirichlet Allocation. CAO, L.; ZHONG, J.; FENG, Y. (Hrsg.). *Advanced Data Mining and Applications*. 2010, psl. 259–269. Lecture Notes in Computer Science. ISBN 978-3-642-17312-7.

12 GONG, X.; GUO, X.; ZHANG, R.; HE, X.; ZHOU, A. Search Behavior Based Latent Semantic User Segmentation for Advertising Targeting. *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. 2013, psl. 211–220.

13 GRIFFITHS, T. L.; STEYVERS, M. Finding scientific topics. *Proceedings of the National Academy of Sciences*. 2004, t. 101, nr. suppl 1, psl. 5228–5235.

14 WILBUR, K. C.; ZHU, Y. Click Fraud. *Marketing Science*. 2009, t. 28, nr. 2, psl. 293–308. ISSN 1526-548X.

15 CUP, K. *Data - KDD Cup 2012, Track2*. 2012.

16 SHAFER, J.; RIXNER, S.; COX, A. The Hadoop distributed filesystem: Balancing portability and performance. *Performance Analysis of Systems Software (ISPASS), 2010 IEEE International Symposium on*. 2010, psl. 122–133.

17 DEAN, J.; GHEMAWAT, S. MapReduce: Simplified Data Processing on Large Clusters. *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6*. San Francisco, CA: USENIX Association, 2004, psl. 10–10. OSDI'04.

18 LI, X.; GROSSMAN, M.; KAELI, D. Mahout on Heterogeneous Clusters Using HadoopCL. *Proceedings of the 2Nd Workshop on Parallel Programming for Analytics Applications*. San Francisco, CA, JAV: ACM, 2015, psl. 9–16. PPAA 2015. ISBN 978-1-4503-3405-1.

19 *Introducing Apache Mahout*.

20 SHALEV-SHWARTZ, S. Online Learning and Online Convex Optimization. *Found. Trends Mach. Learn.* 2012, t. 4, nr. 2, psl. 107–194. ISSN 1935-8237.

21 SHALEV-SHWARTZ, S. *Online Learning: Theory, Algorithms, and Applications*. 2007.

22 WEINBERGER, K.; DASGUPTA, A.; LANGFORD, J.; SMOLA, A.; ATTENBERG, J. Feature Hashing for Large Scale Multitask Learning. *Proceedings of the 26th Annual International Conference on Machine Learning*. Montreal, Quebec, Canada: ACM, 2009, psl. 1113–1120. ICML '09. ISBN 978-1-60558-516-1.

23 MENG, X.; BRADLEY, J. K.; YAVUZ, B. ir kt. MLlib: Machine Learning in Apache Spark. *CoRR*. 2015, t. abs/1505.06807.

24 ZAHARIA, M.; CHOWDHURY, M.; DAS, T. ir kt. Resilient Distributed Datasets: A Fault-tolerant Abstraction for In-memory Cluster Computing. *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*. San Jose, CA: USENIX Association, 2012, psl. 2–2. NSDI'12.

25 ZAHARIA, M.; CHOWDHURY, M.; FRANKLIN, M. J.; SHENKER, S.; STOICA, I. Spark: Cluster Computing with Working Sets. *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*. Boston, MA: USENIX Association, 2010, psl. 10–10. HotCloud'10.

26 DEAN, J.; GHEMAWAT, S. MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM*. 2008, t. 51, nr. 1, psl. 107–113. ISSN 0001-0782.

27 OLSTON, C.; REED, B.; SRIVASTAVA, U.; KUMAR, R.; TOMKINS, A. Pig Latin: A Not-so-foreign Language for Data Processing. *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. Vancouver, Canada: ACM, 2008, psl. 1099–1110. SIGMOD '08. ISBN 978-1-60558-102-6.

28 THUSOO, A.; SARMA, J. S.; JAIN, N. ir kt. Hive: A Warehousing Solution over a Map-reduce Framework. *Proc. VLDB Endow.* 2009, t. 2, nr. 2, psl. 1626–1629. ISSN 2150-8097.

29 MELNIK, S.; GUBAREV, A.; LONG, J. J. ir kt. Dremel: Interactive Analysis of Web-scale Datasets. *Proc. VLDB Endow.* 2010, t. 3, nr. 1-2, psl. 330–339. ISSN 2150-8097.

30 ENGLE, C.; LUPHER, A.; XIN, R. ir kt. Shark: Fast Data Analysis Using Coarse-grained Distributed Memory. *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. Scottsdale, Arizona, JAV: ACM, 2012, psl. 689–692. SIGMOD '12. ISBN 978-1-4503-1247-9.

31 SELLIS, T. K.; SIMITSIS, A. ETL Workflows: From Formal Specification to Optimization. *Proceedings of the 11th East European Conference on Advances in Databases and Information Systems*. Varna, Bulgaria: Springer-Verlag, 2007, psl. 1–11. ADBIS'07. ISBN 3-540-75184-X, 978-3-540-75184-7.

32 MAJCHRZAK, T. A.; JANSEN, T.; KUCHEN, H. Efficiency Evaluation of Open Source ETL Tools. *Proceedings of the 2011 ACM Symposium on Applied Computing*. TaiChung, Taiwan: ACM, 2011, psl. 287–294. SAC '11. ISBN 978-1-4503-0113-8.

33 ARMBRUST, M.; XIN, R. S.; LIAN, C. ir kt. Spark SQL: Relational Data Processing in Spark. *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. Melbourne, Victoria, Australia: ACM, 2015, psl. 1383–1394. SIGMOD '15. ISBN 978-1-4503-2758-9.

34 ELLNER, S. P. Review of R, Version 1.1.1. *Bulletin of the Ecological Society of America*. 2001, t. 82, nr. 2, psl. 127–128.

35 RIBEIRO Jr., P. J.; BROWN, P. E. Some words on the R project. *The ISBA Bulletin*. 2001, t. 8, nr. 1, psl. 12–16.

36 VORA, M. Hadoop-HBase for large-scale data. *Computer Science and Network Technology (ICCSNT), 2011 International Conference on*. 2011, psl. 601–605.

37 WADLER, P. Monads for Functional Programming. *Advanced Functional Programming, First International Spring School on Advanced Functional Programming Techniques-Tutorial Text*. London, UK, UK: Springer-Verlag, 1995, psl. 24–52. ISBN 3-540-59451-5.

38 CANNY, J.; ZHAO, H. Bidmach: Large-scale learning with zero memory allocation. *BigLearn Workshop, NIPS*. 2013.

39 ZHAO, H.; CANNY, J. F. *High Performance Machine Learning through Codesign and Rooflining*. 2014.

40 CANNY, J.; ZHAO, H. Big data analytics with small footprint: Squaring the cloud. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2013, psl. 95–103.

41 YAN, Y.; GROSSMAN, M.; SARKAR, V. JCUDA: A Programmer-Friendly Interface for Accelerating Java Programs with CUDA. SIPS, H.; EPEMA, D.; LIN, H.-X. (Hrsg.). *Euro-Par 2009 Parallel Processing*. 2009, psl. 887–899. Lecture Notes in Computer Science. ISBN 978-3-642-03868-6.

42 FOLK, M.; HEBER, G.; KOZIOL, Q.; POURMAL, E.; ROBINSON, D. An Overview of the HDF5 Technology Suite and Its Applications. *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*. Uppsala, Sweden: ACM, 2011, psl. 36–47. AD '11. ISBN 978-1-4503-0614-0.

43 MORIN, S.; KOREN, I.; KRISHNA, C. M. JMPI: Implementing the Message Passing Standard in Java. *Proceedings of the 16th International Parallel and Distributed Processing Symposium*. Washington, DC, JAV: IEEE Computer Society, 2002, psl. 191–. IPDPS '02. ISBN 0-7695-1573-8.

44 GOPALANI, S.; ARORA, R. Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means. *International Journal of Computer Applications*. 2015, t. 113, nr. 1, psl. 8–11.

45 BORKAR, V.; CAREY, M.; GROVER, R.; ONOSE, N.; VERNICA, R. Hyracks: A Flexible and Extensible Foundation for Data-intensive Computing. *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*. Washington, DC, JAV: IEEE Computer Society, 2011, psl. 1151–1162. ICDE '11. ISBN 978-1-4244-8959-6.

46 ZHAO, H.; CANNY, J. Butterfly mixing: Accelerating incremental-update algorithms on clusters. *SIAM Conf. on Data Mining*. 2013.

47 ZHAO, H.; CANNY, J. F. Communication-Efficient Distributed Stochastic Gradient Descent with Butterfly Mixing. *University of California, Berkeley*. 2012.

48 LAAN, M. van der; ROSE, S. *Targeted learning: causal inference for observational and experimental data*. 2011.

49 BLEI, D. M. Probabilistic Topic Models. *Commun. ACM*. 2012, t. 55, nr. 4, psl. 77–84. ISSN 0001-0782.

50 FRIGYIK, B.; KAPILA, A.; GUPTA, M. R. Introduction to the Dirichlet distribution and related processes. *Department of Electrical Engineering, University of Washignton, UWEETR-2010-0006*. 2010.

51 GELMAN, A. *Bayesian data analysis*. 1995.

52 DIETZ, L. *Directed Factor Graph Notation for Generative Models*. 2010.

53 BLEI, D. M.; LAFFERTY, J. D. Topic models. *Text mining: classification, clustering, and applications*. 2009, t. 10, psl. 71.

54 REED, C. *Latent Dirichlet Allocation: Towards a Deeper Understanding*. 2012.

55 NAKAJIMA, S.; SATO, I.; SUGIYAMA, M.; WATANABE, K.; KOBAYASHI, H. Analysis of Variational Bayesian Latent Dirichlet Allocation: Weaker Sparsity Than MAP. *Advances in Neural Information Processing Systems*. 2014, psl. 1224–1232.

56 BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 2003, t. 3, psl. 993–1022. ISSN 1532-4435.

57 BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, JAV: Springer-Verlag New York, Inc., 2006. ISBN 0387310738.

58 HOFFMAN, M.; BLEI, D. M.; BACH, F. Online learning for latent dirichlet allocation. *Advances in Neural Information Processing Systems*. 2010, t. 23, psl. 856–864.

59 JORDAN, M. I.; GHAHRAMANI, Z.; JAAKKOLA, T. S.; SAUL, L. K. An Introduction to Variational Methods for Graphical Models. *Mach. Learn.* 1999, t. 37, nr. 2, psl. 183–233. ISSN 0885-6125.

60 WAINWRIGHT, M.; JORDAN, M. A Variational Principle for Graphical Models. *New Directions in Statistical Signal Processing*. 2005.

61 WAINWRIGHT, M. J.; JORDAN, M. I. Graphical Models, Exponential Families, and Variational Inference. *Found. Trends Mach. Learn.* 2008, t. 1, nr. 1-2, psl. 1–305. ISSN 1935-8237.

62 SALTON, G.; WONG, A.; YANG, C. S. A Vector Space Model for Automatic Indexing. *Commun. ACM*. 1975, t. 18, nr. 11, psl. 613–620. ISSN 0001-0782.

63 WALLACH, H. M.; MIMNO, D. M.; MCCALLUM, A. Rethinking LDA: Why Priors Matter. BENGIO, Y.; SCHUURMANS, D.; LAFFERTY, J.; WILLIAMS, C.; CULOTTA, A. (Hrsg.). *Advances in Neural Information Processing Systems 22*. 2009, psl. 1973–1981.

64 ASUNCION, A.; WELLING, M.; SMYTH, P.; TEH, Y. W. On Smoothing and Inference for Topic Models. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. Montreal, Quebec, Canada: AUAI Press, 2009, psl. 27–34. UAI '09. ISBN 978-0-9749039-5-8.

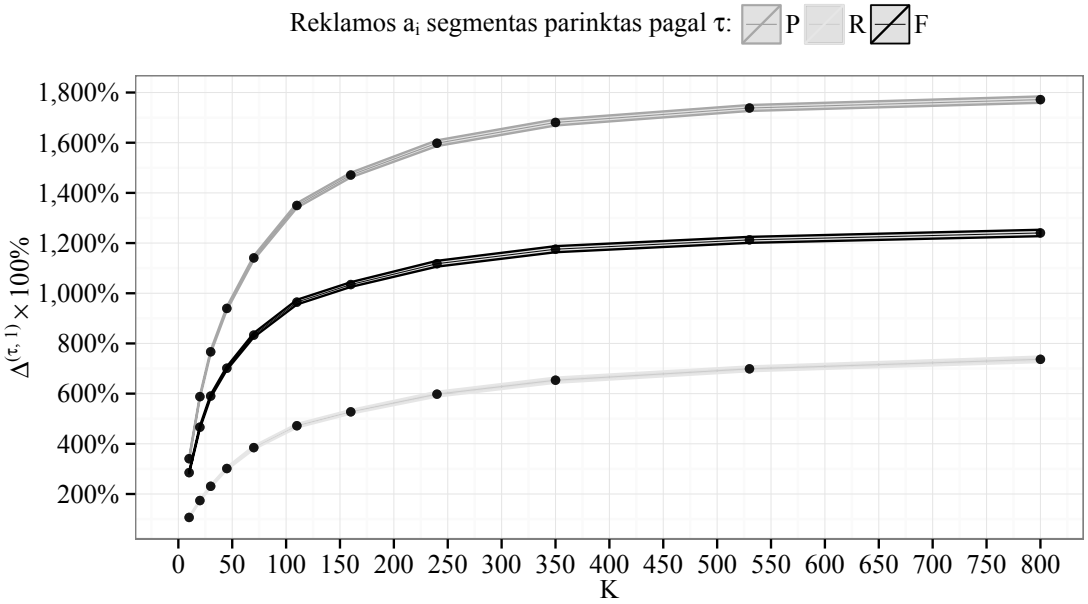
65 BOUSQUET, O.; BOTTOU, L. The tradeoffs of large scale learning. *Advances in neural information processing systems*. 2008, psl. 161–168.

66 LIANG, P.; KLEIN, D. Online EM for Unsupervised Models. *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Boulder, Colorado: Association for Computational Linguistics, 2009, psl. 611–619. NAACL '09. ISBN 978-1-932432-41-1.

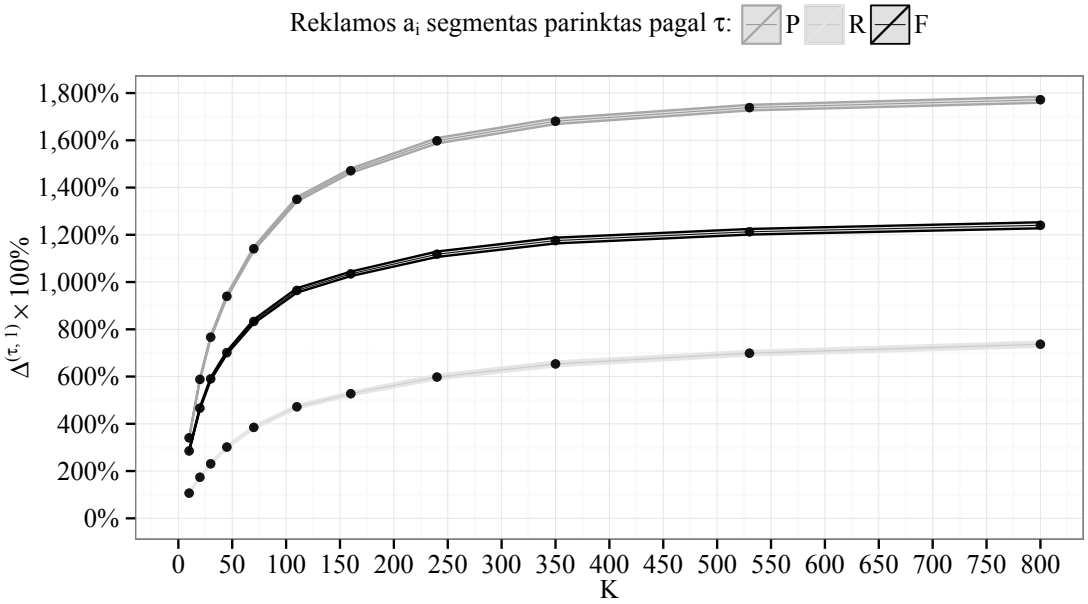
67 HRIPCSAK, G.; ROTHSCHILD, A. S. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American Medical Informatics Association*. 2005, t. 12, nr. 3, psl. 296–298.

PRIEDAI

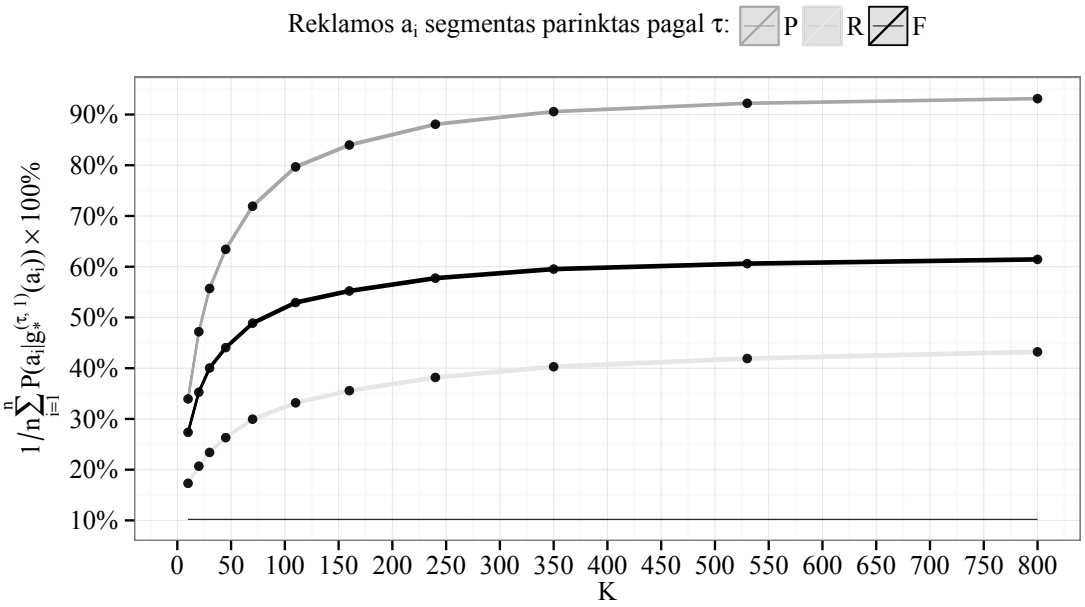
1 priedas. TYRIMŲ GRAFIKAI



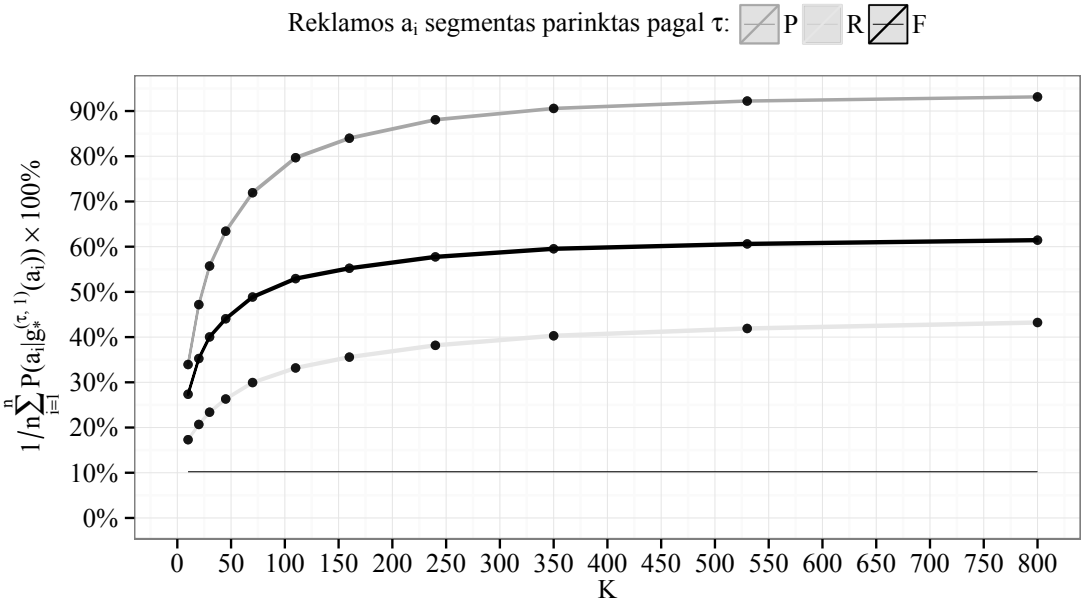
0.1 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K be apribojimų.



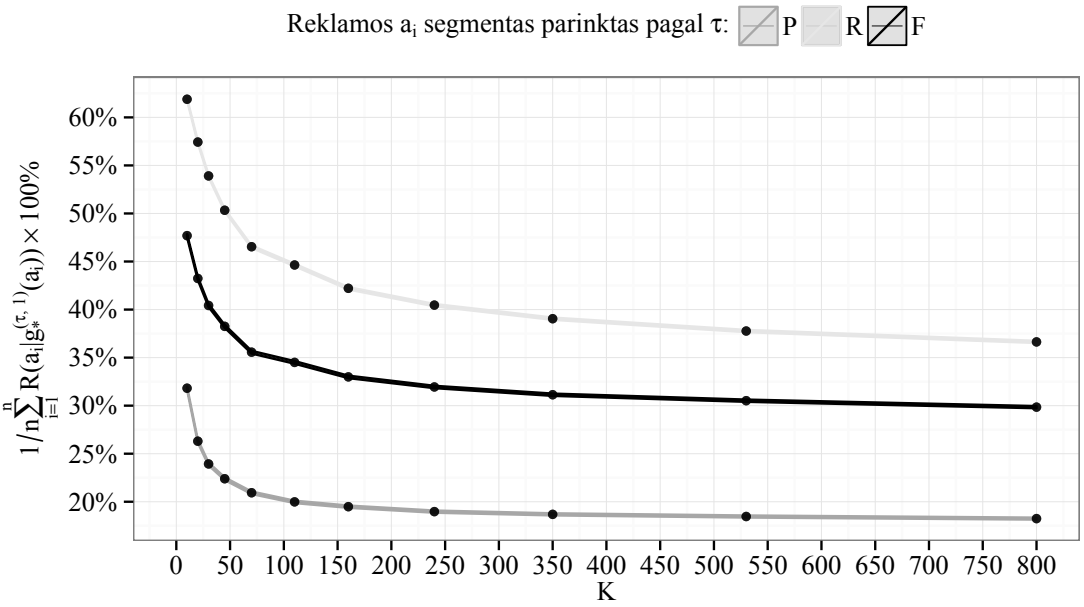
0.2 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K be apribojimų.



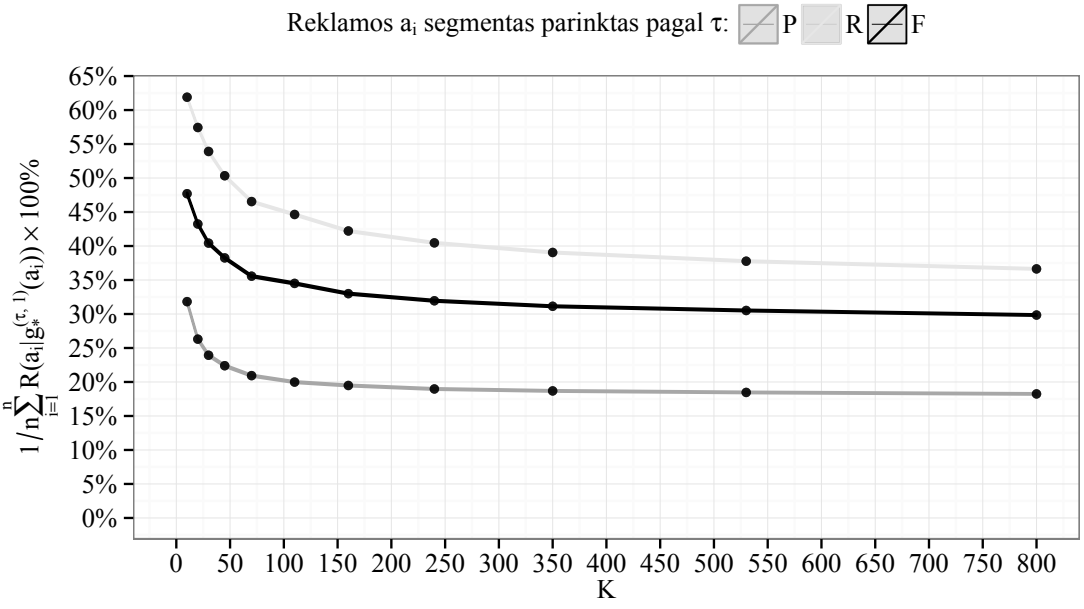
0.3 pav. Vidutinio tikslumu (P) priklausomybė nuo segmentų kiekio K be apribojimų.



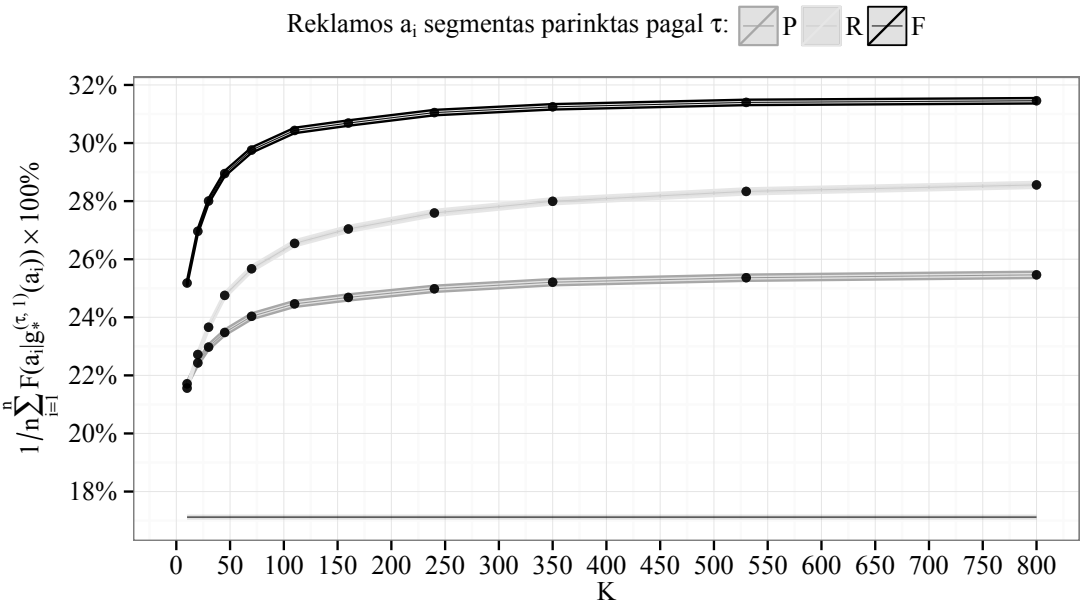
0.4 pav. Vidutinio tikslumu (P) priklausomybė nuo segmentų kiekio K be apribojimų.



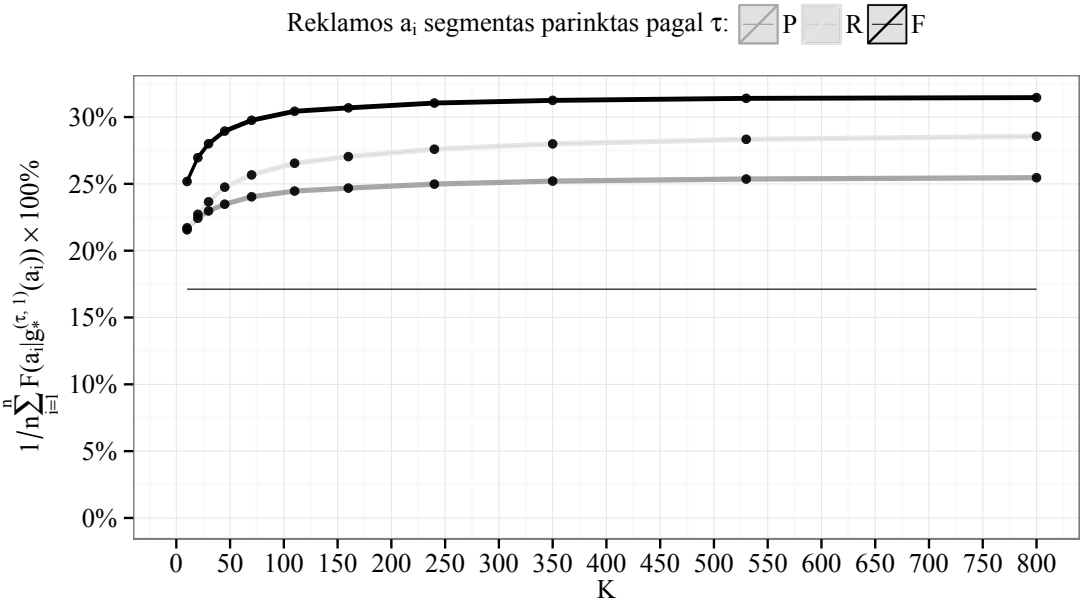
0.5 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K be apribojimų.



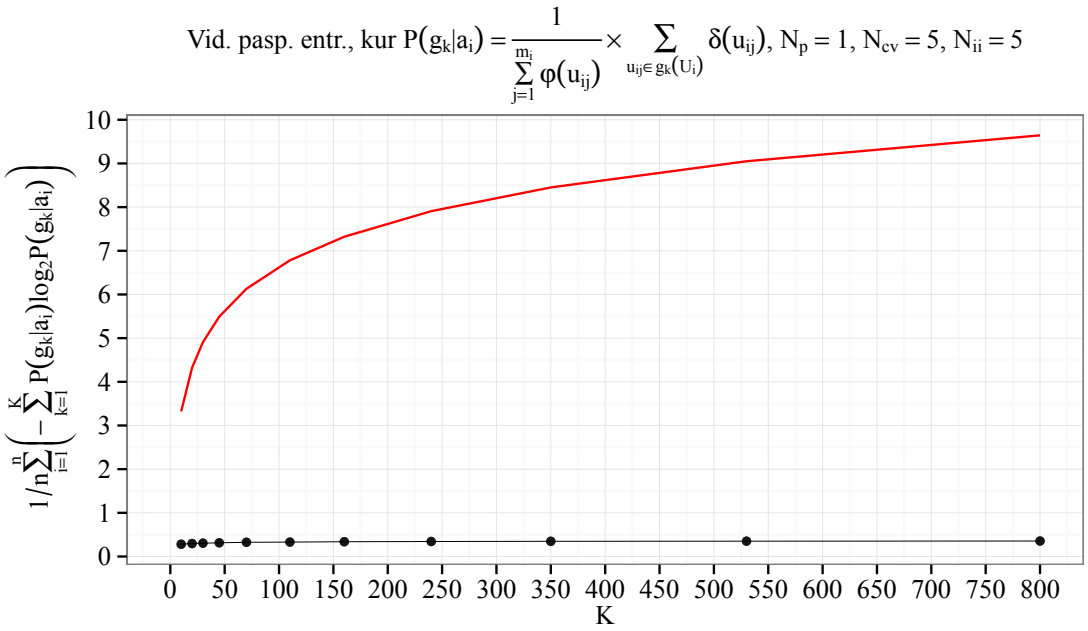
0.6 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K be apribojimų.



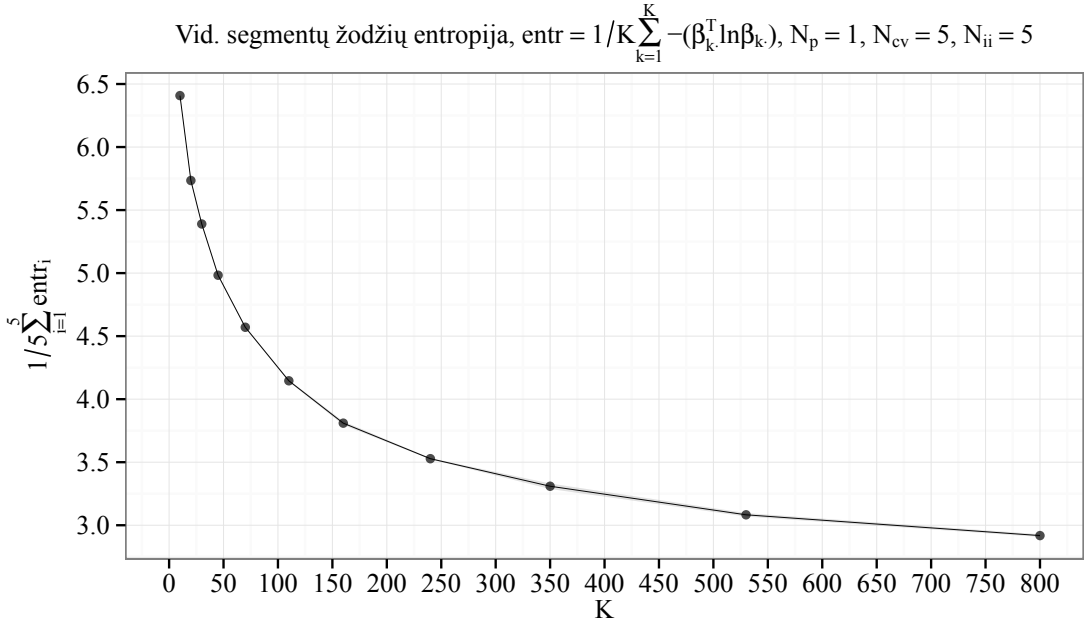
0.7 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K be apribojimų.



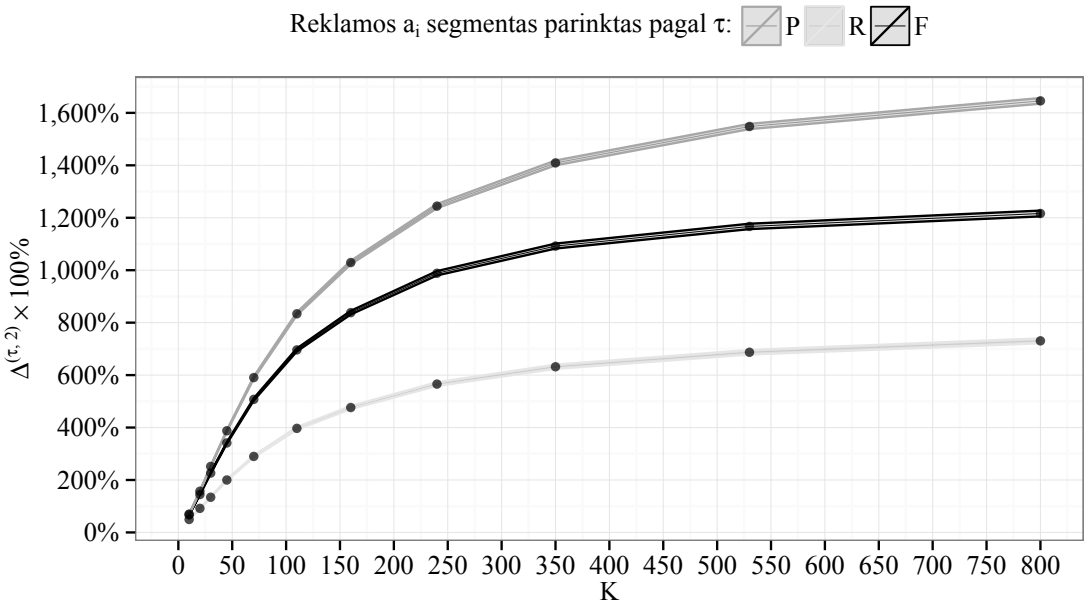
0.8 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K be apribojimų.



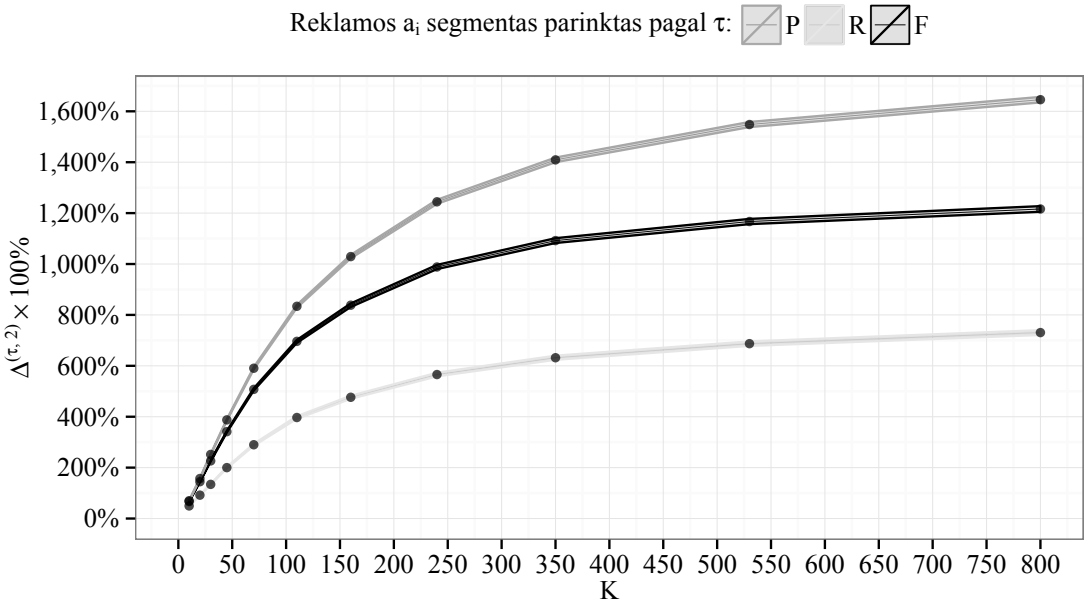
0.9 pav. Vidutinė paspaudimų entropija.



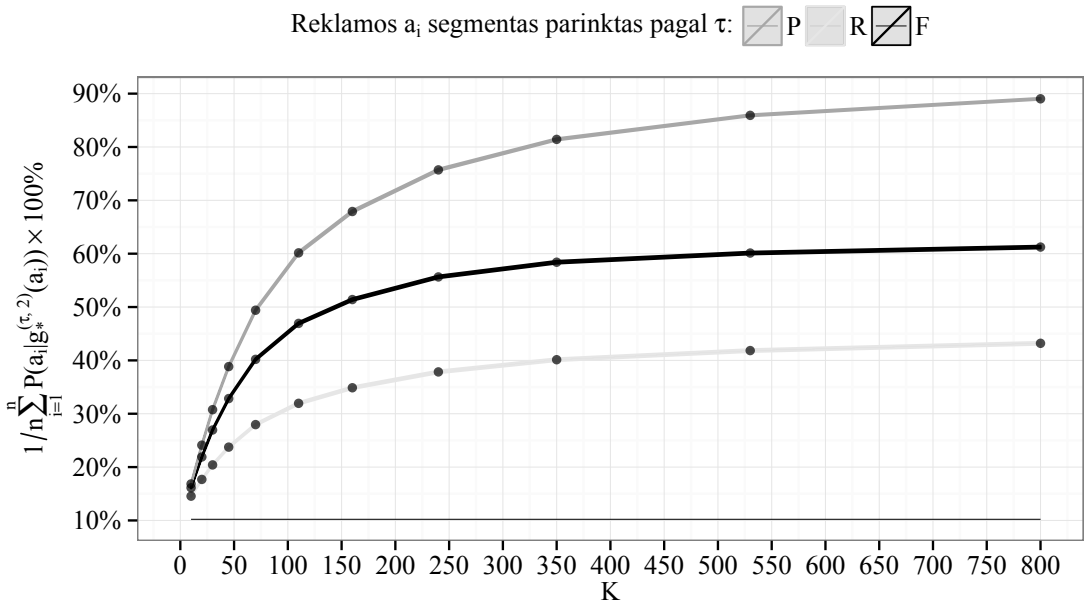
0.10 pav. Vidutinė segmentų žodžių entropija.



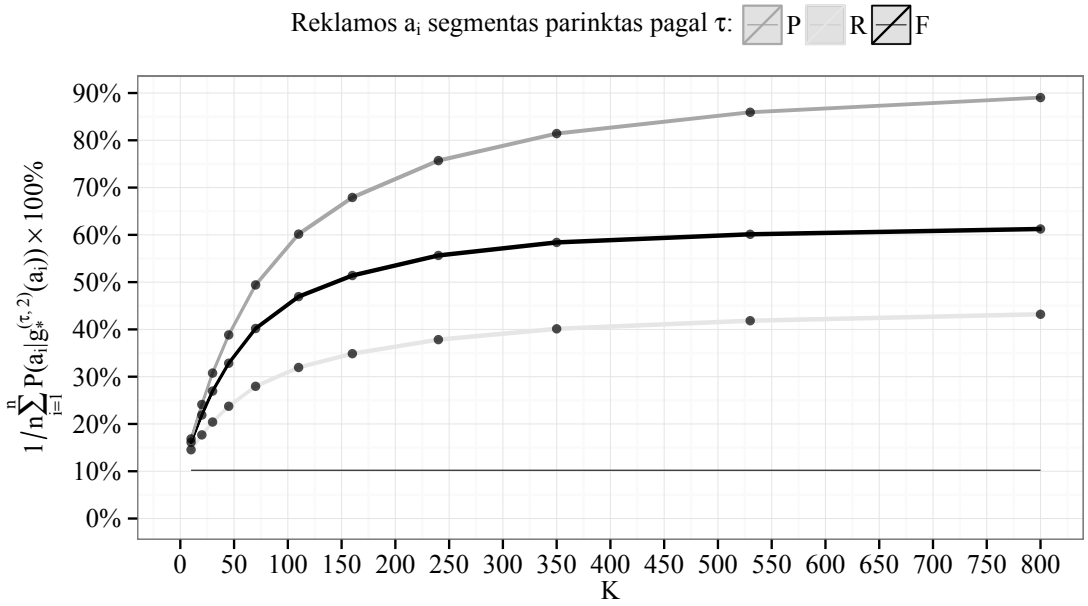
0.11 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



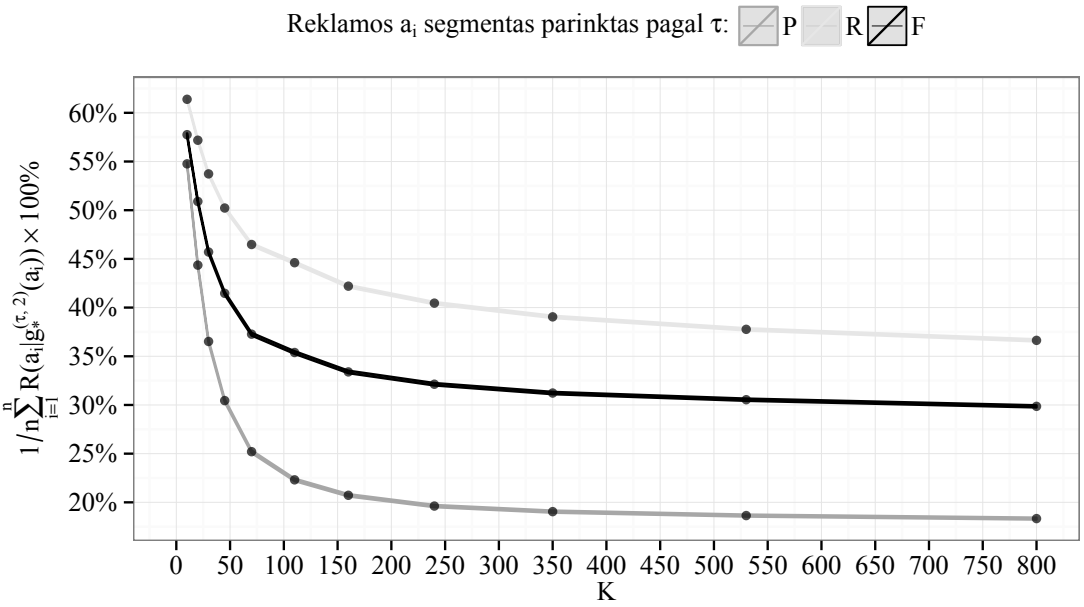
0.12 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



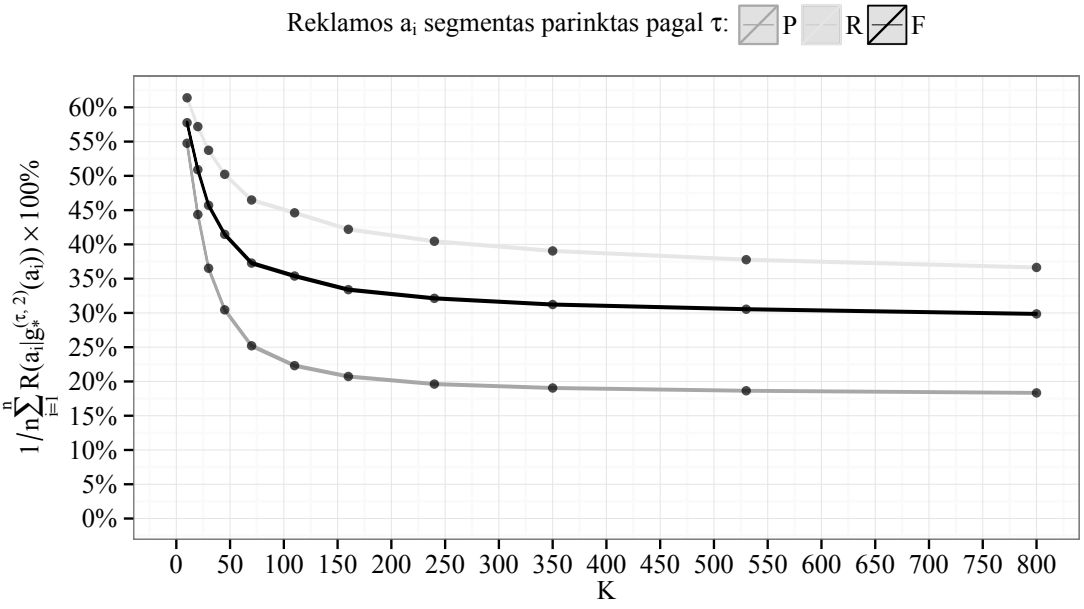
0.13 pav. Vidutinio tikslumu (P) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



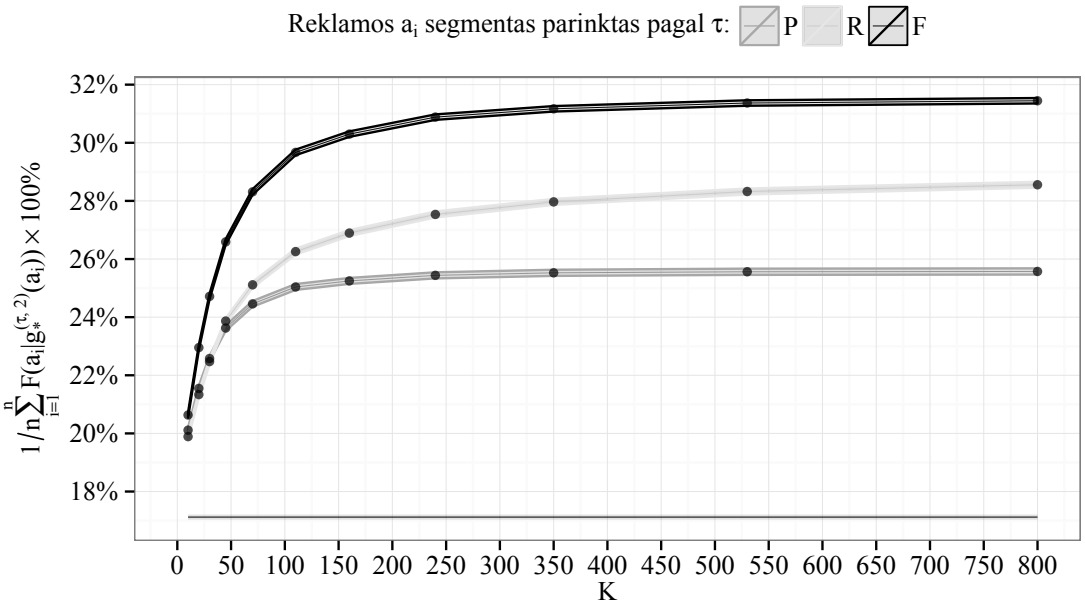
0.14 pav. Vidutinio tikslumu (P) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



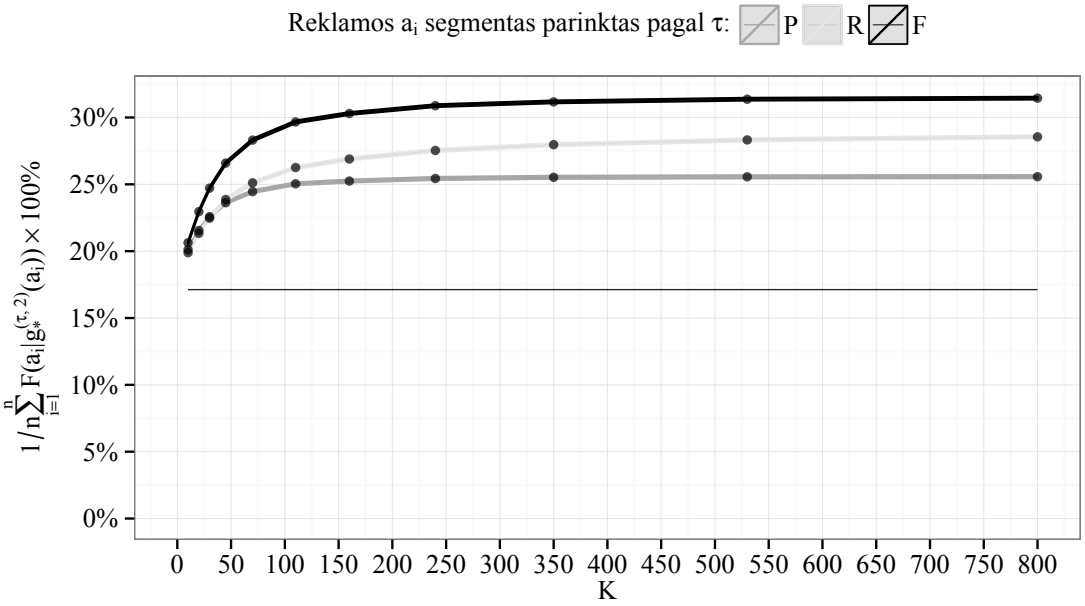
0.15 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



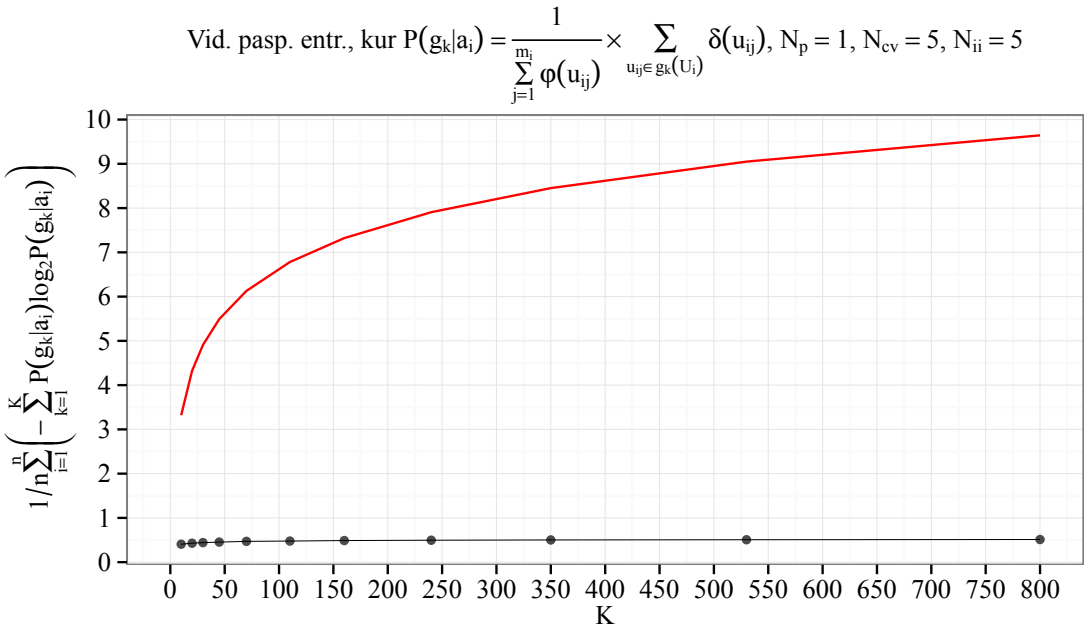
0.16 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



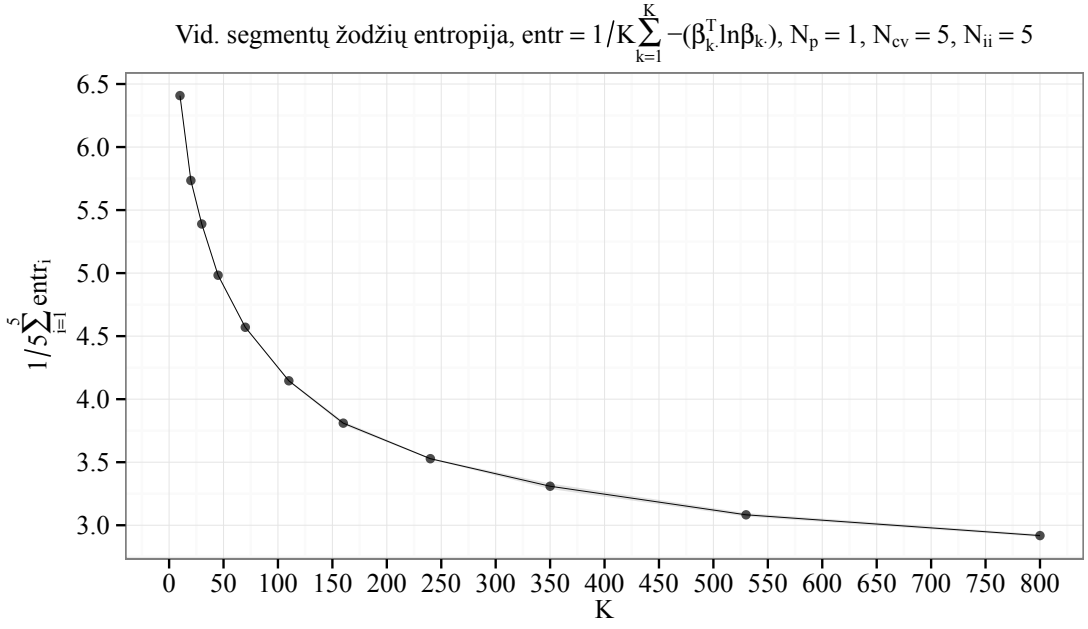
0.17 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



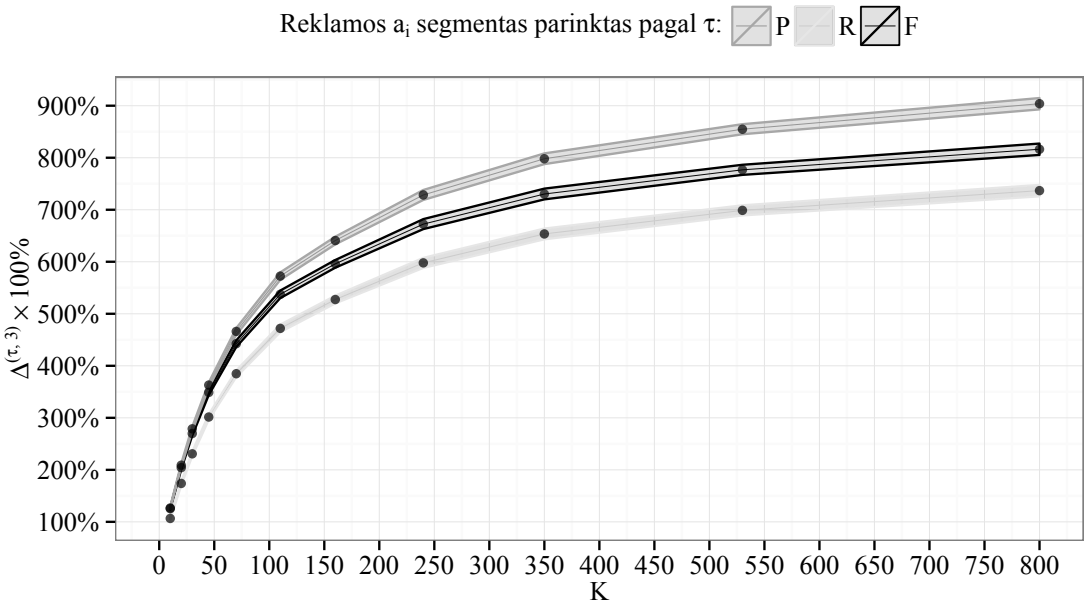
0.18 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal segmento vartotojų skaičių.



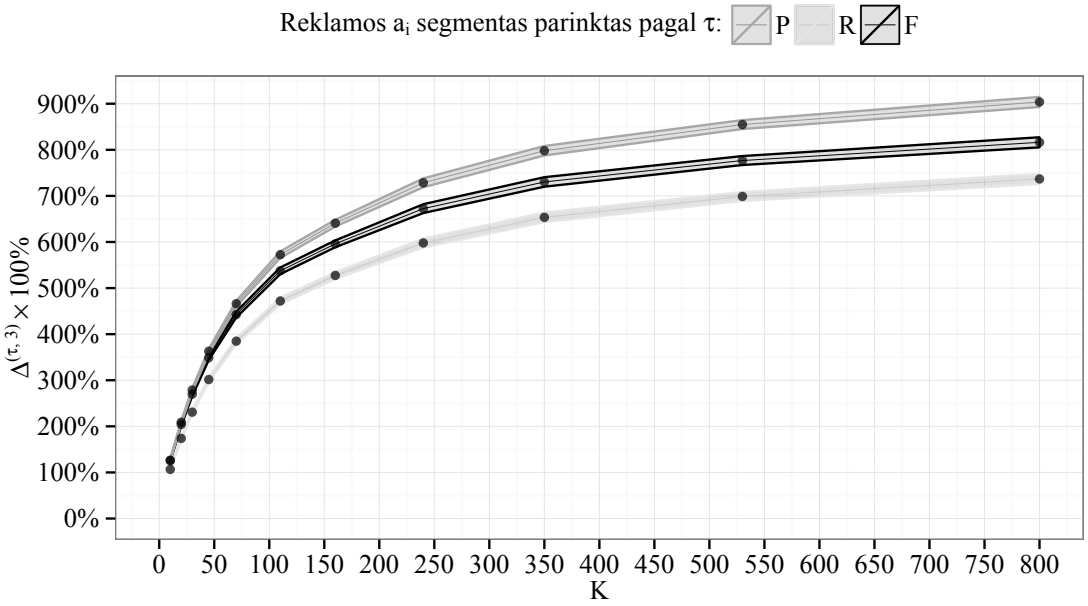
0.19 pav. Vidutinė paspaudimų entropija.



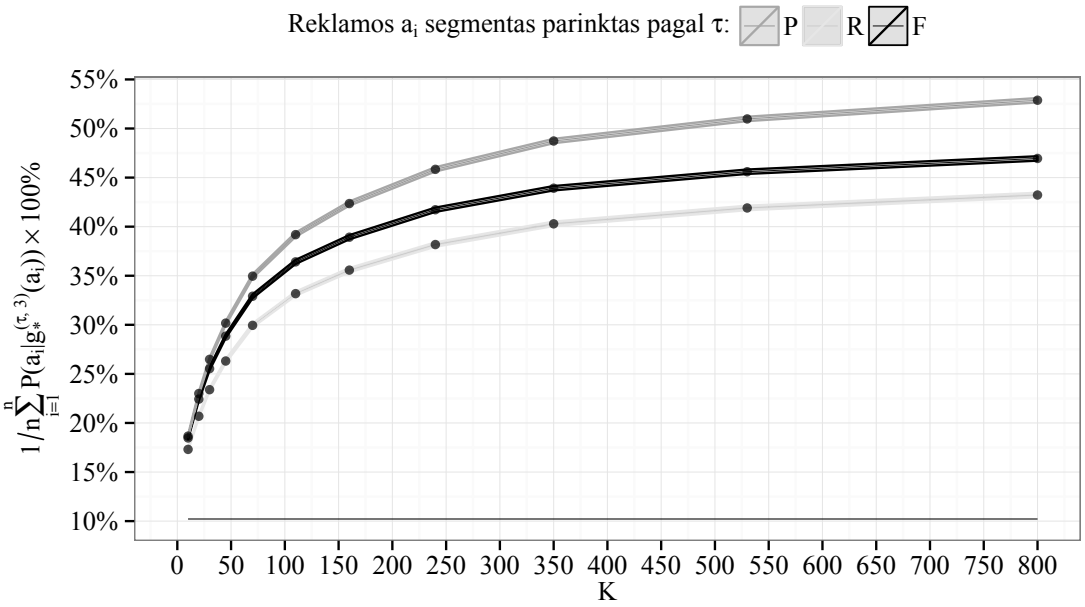
0.20 pav. Vidutinė segmentų žodžių entropija.



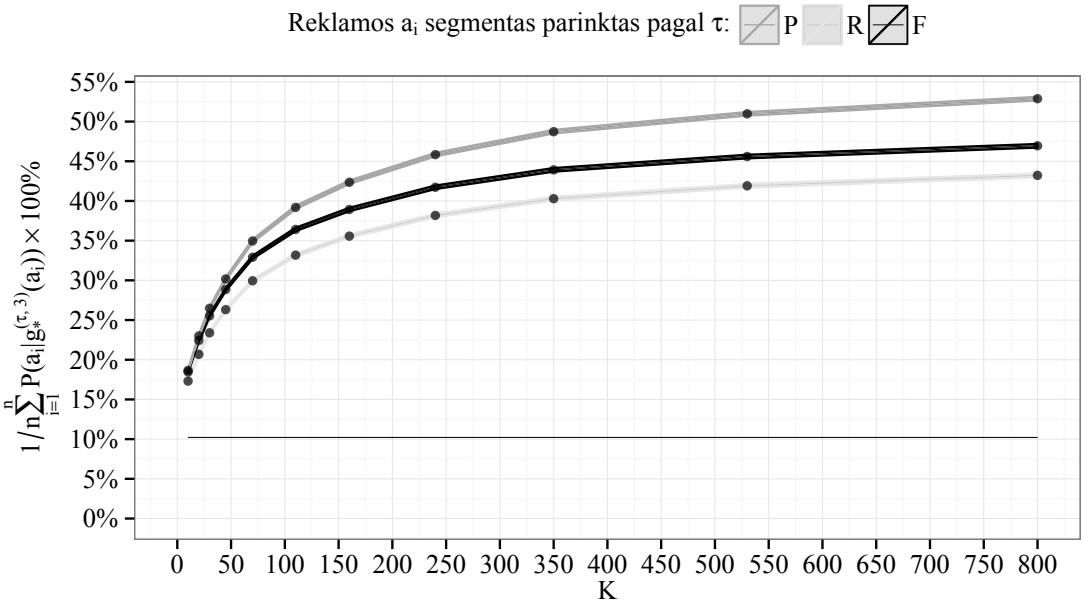
0.21 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



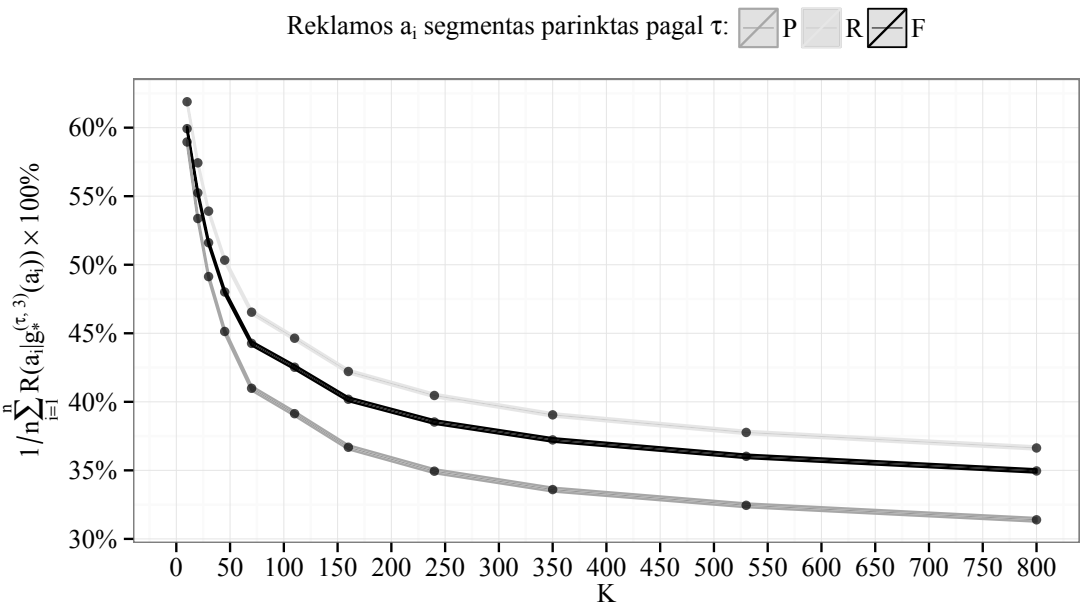
0.22 pav. Vidutinio santykinio CTR pokyčio (Δ) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



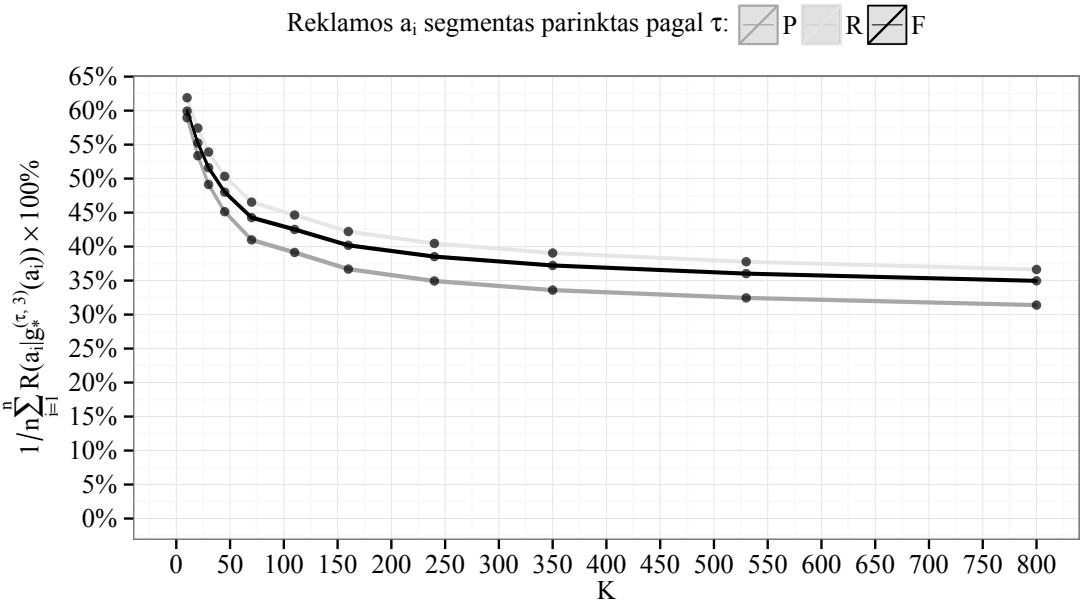
0.23 pav. Vidutinio tikslumu (P) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



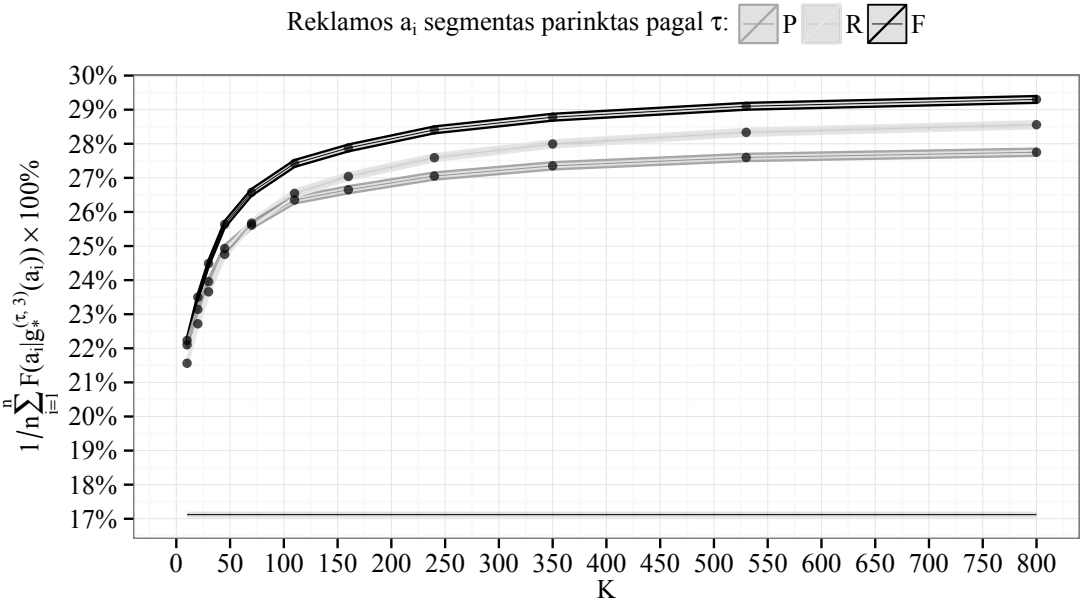
0.24 pav. Vidutinio tikslumu (P) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



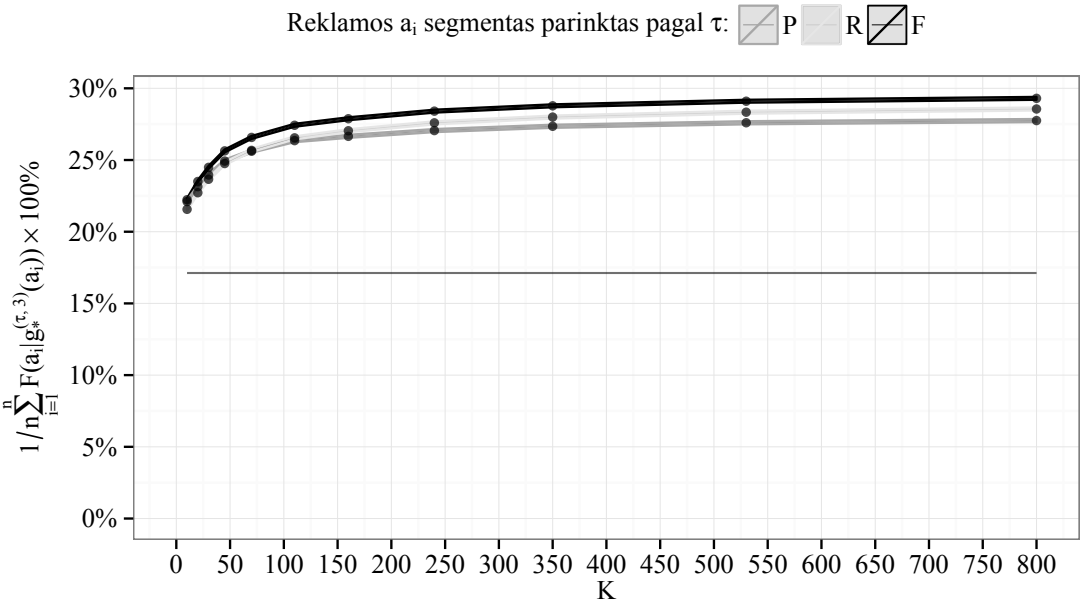
0.25 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



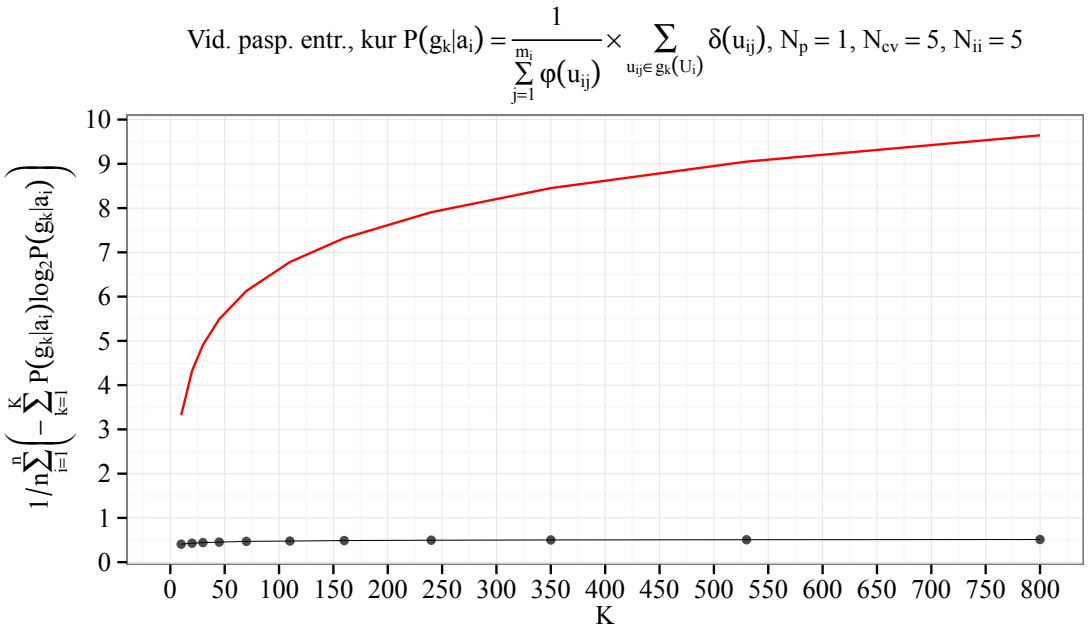
0.26 pav. Vidutinio atkuriamumo (R) priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



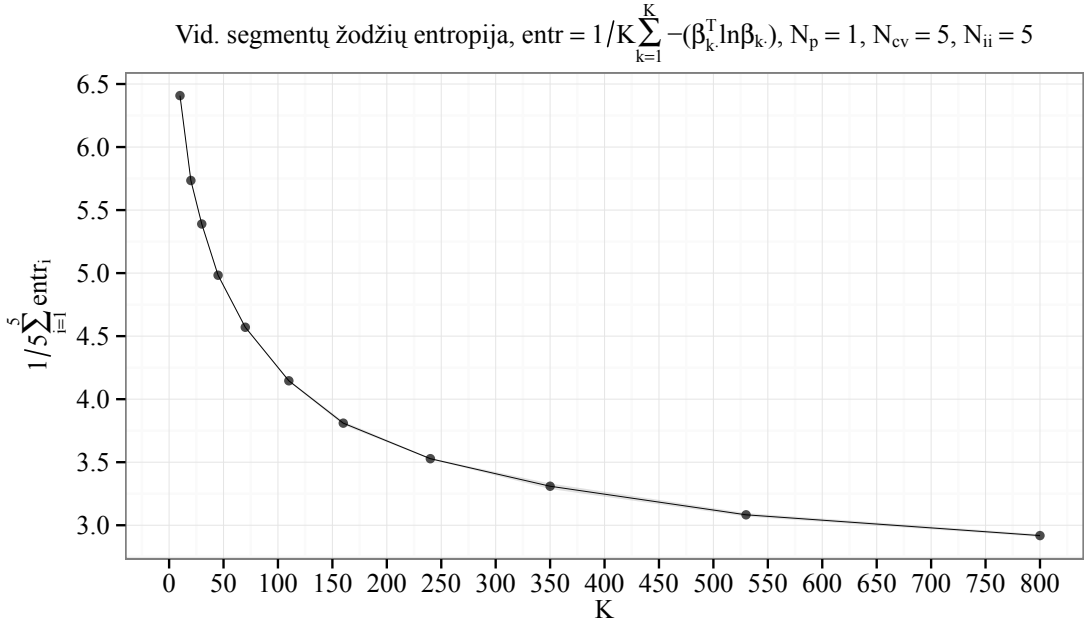
0.27 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



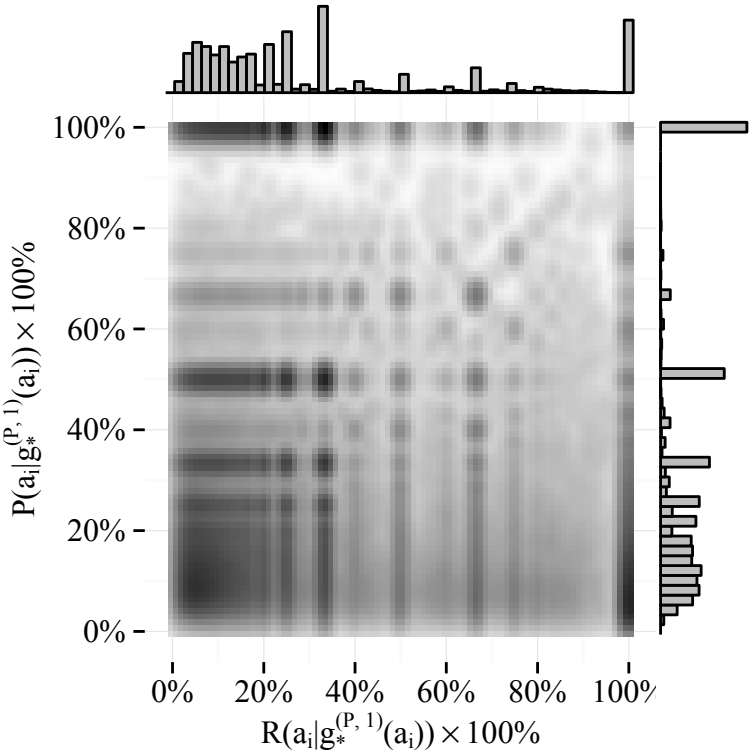
0.28 pav. Vidutinės F-mato reikšmės priklausomybė nuo segmentų kiekio K, kai taikomas apribojimas pagal atkuriamumą (R).



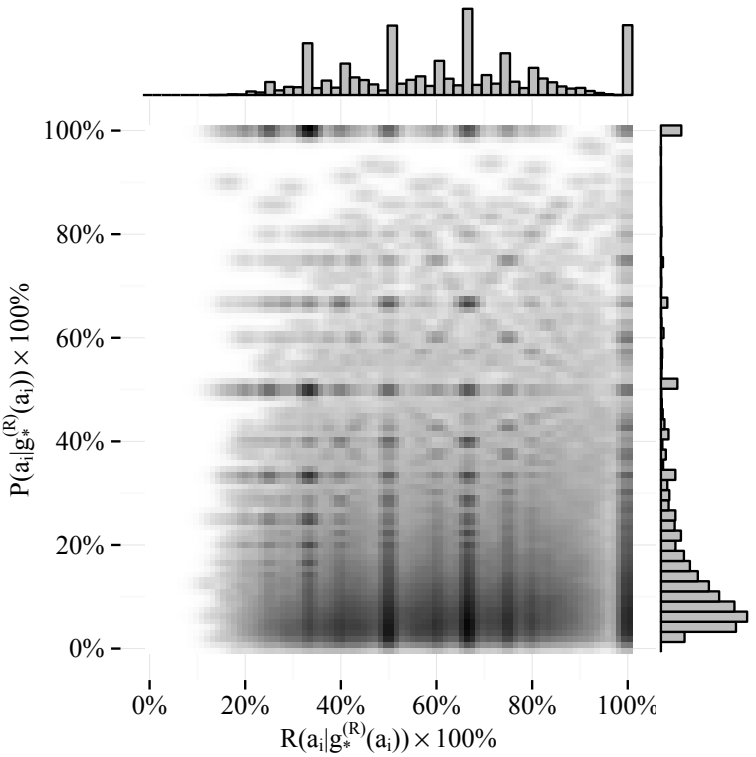
0.29 pav. Vidutinė paspaudimų entropija.



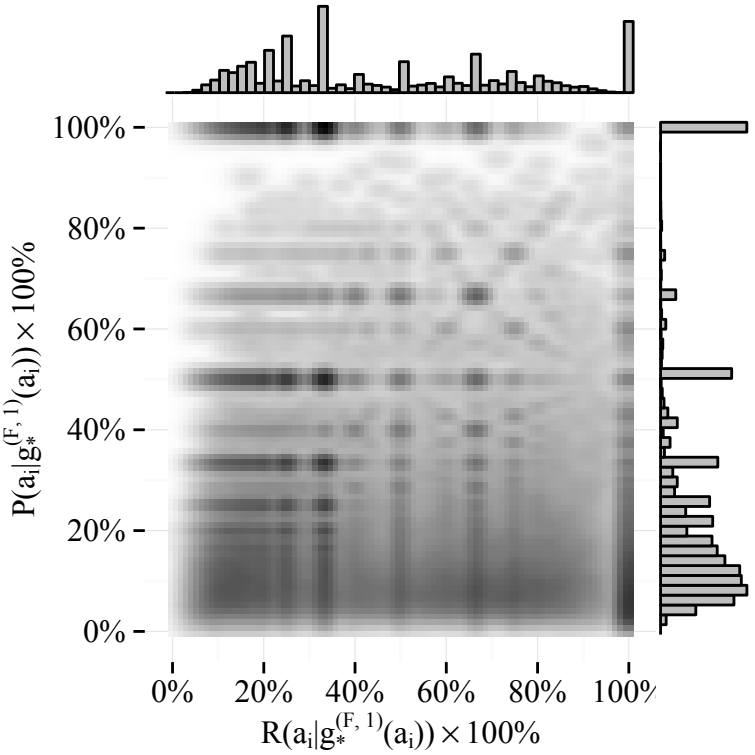
0.30 pav. Vidutinė segmentų žodžių entropija.



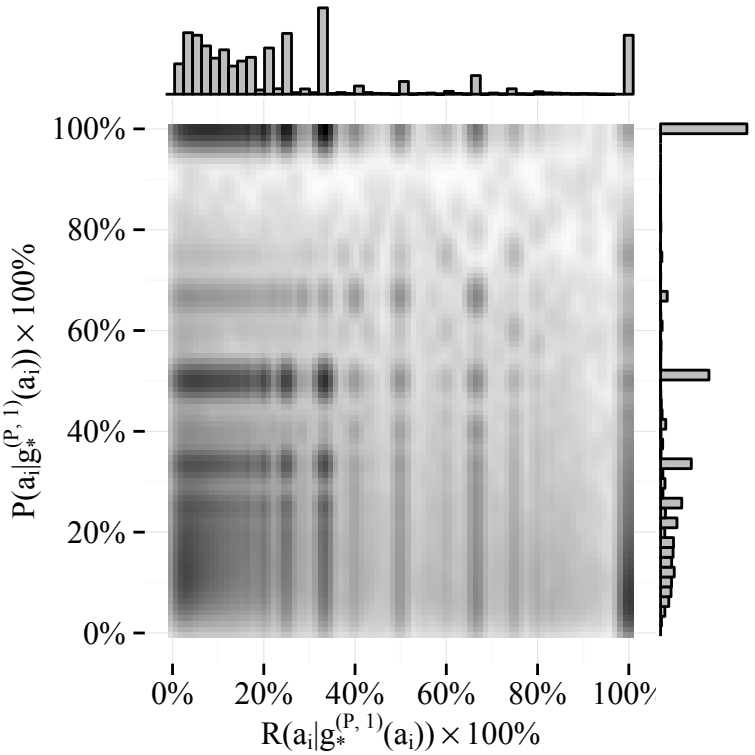
0.31 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 10$**.**



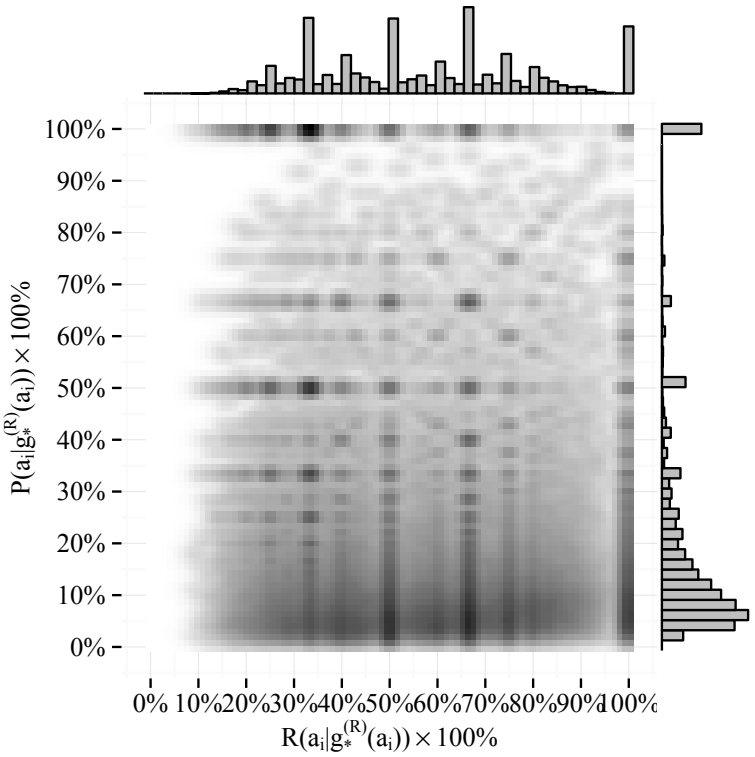
0.32 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 10$**.**



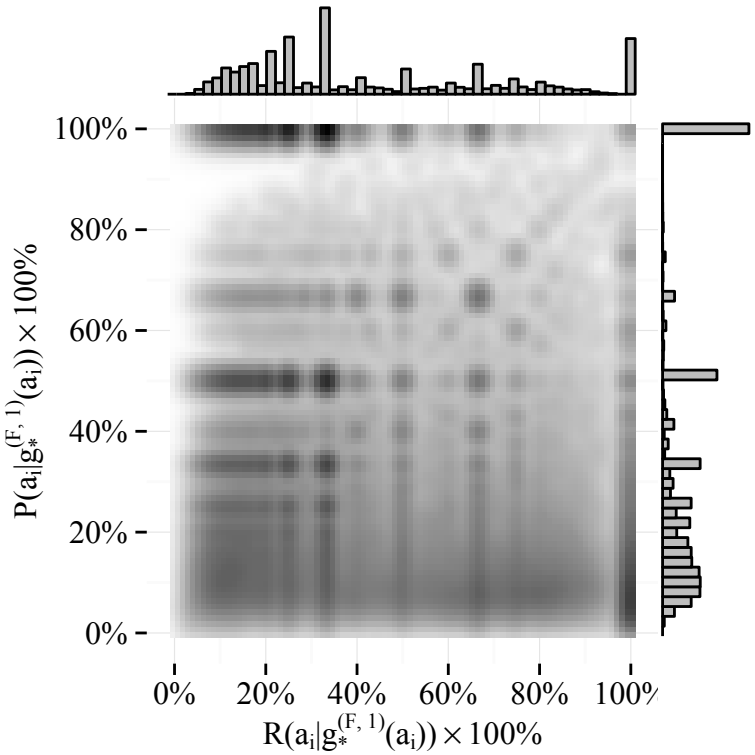
0.33 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 10$**.**



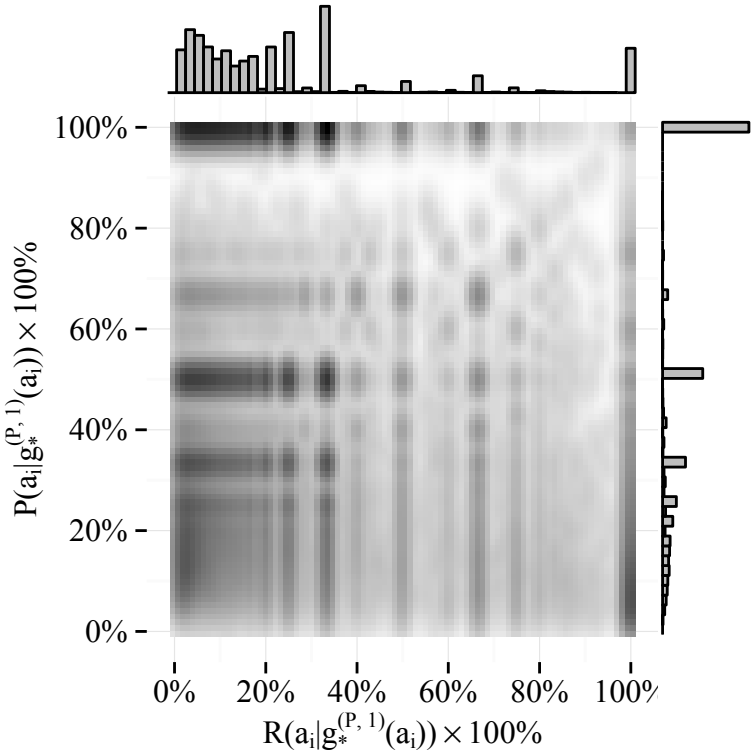
0.34 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 20$**.**



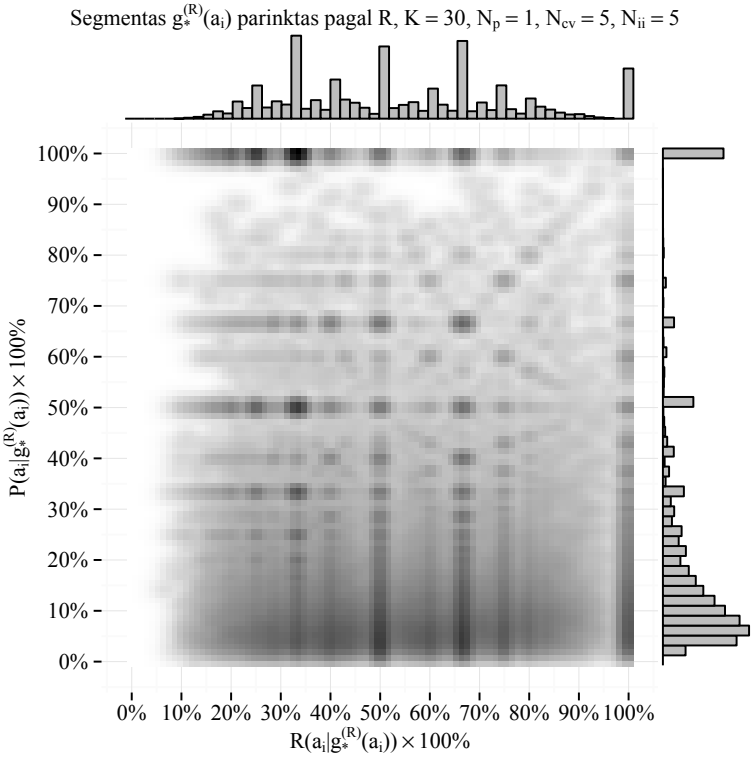
0.35 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 20$**.**



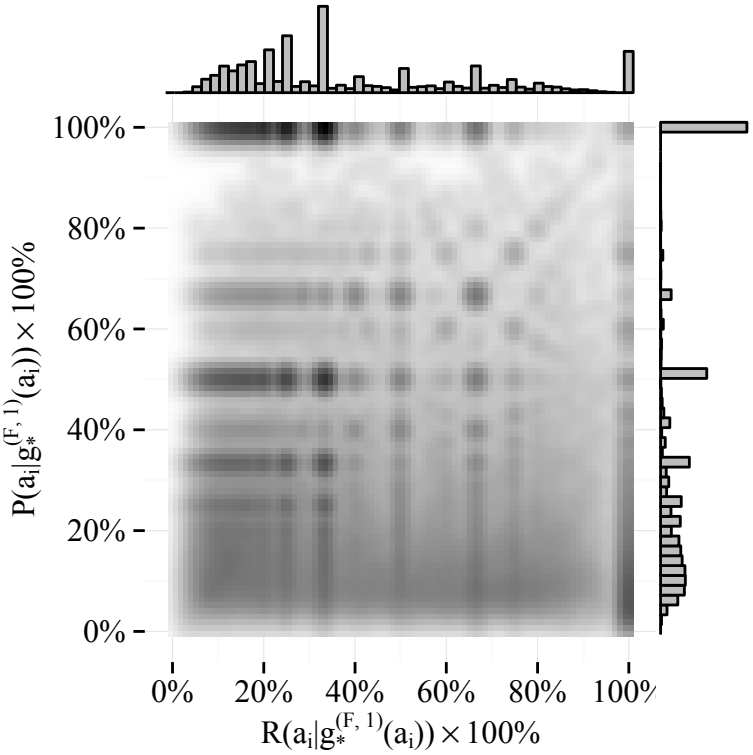
0.36 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 20$**.**



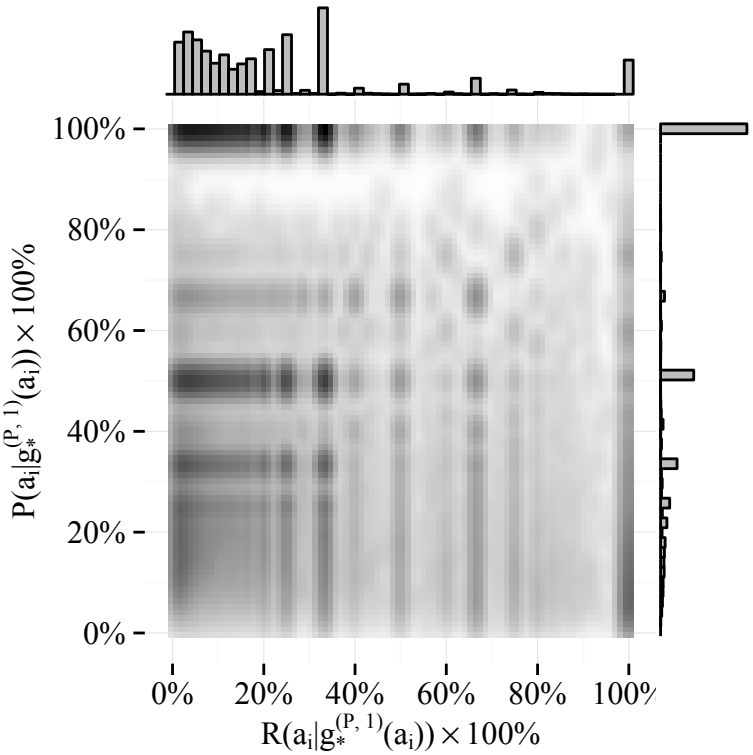
0.37 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 30$**.**



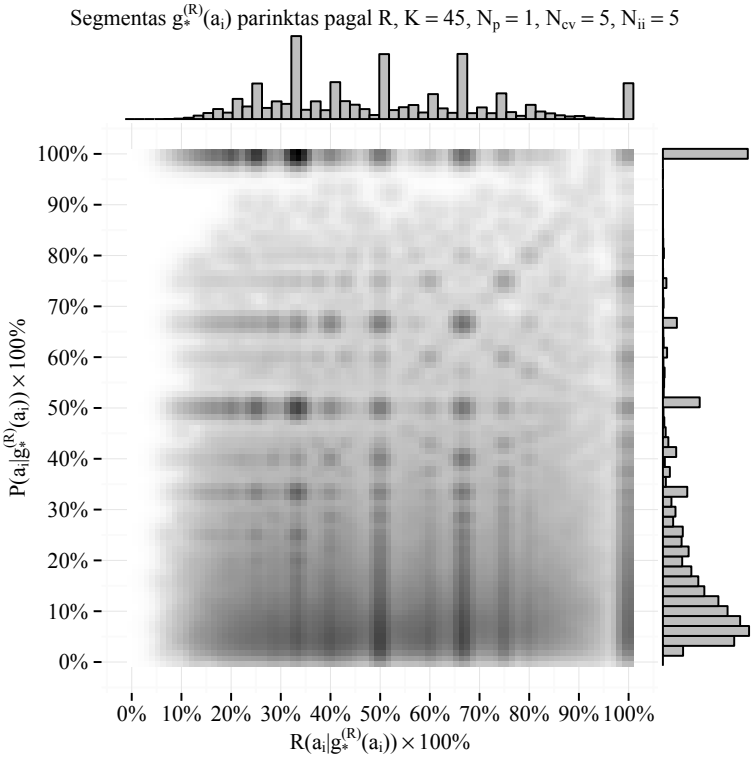
0.38 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 30$**.**



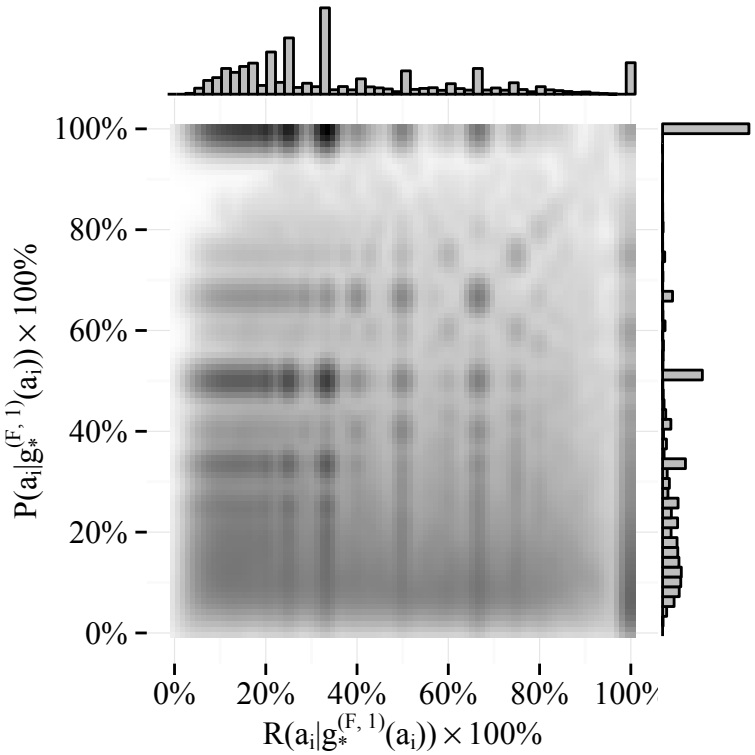
0.39 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 30$**.**



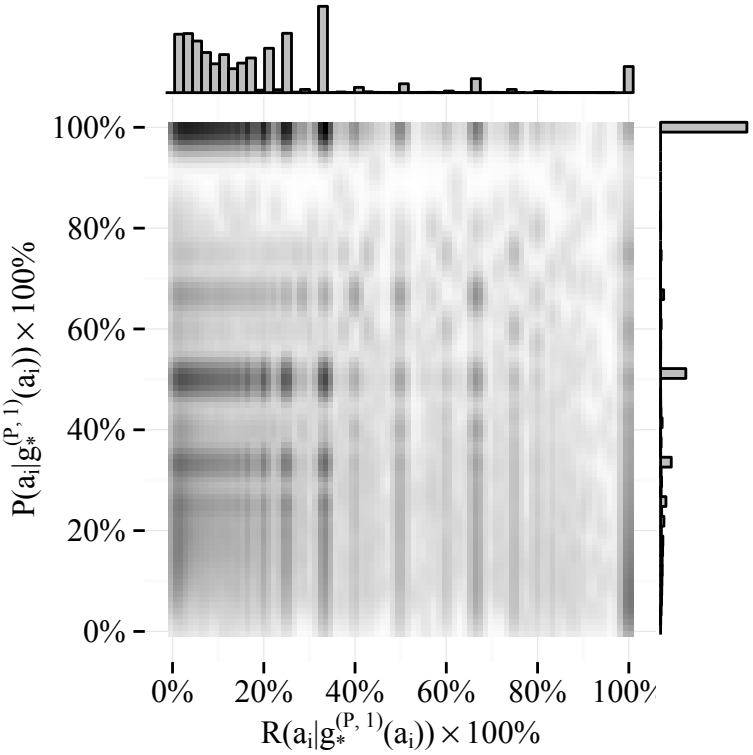
0.40 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 45$**.**



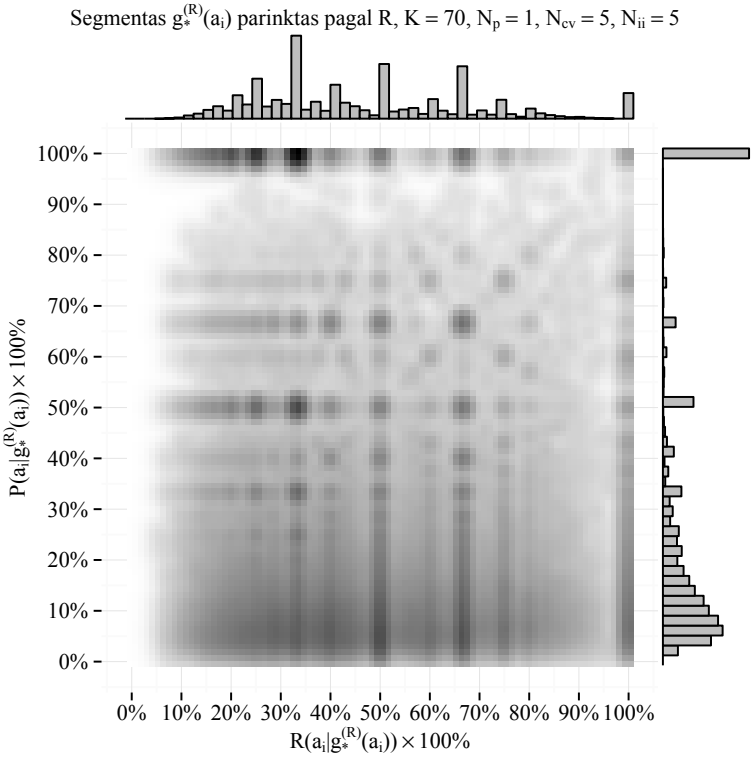
0.41 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 45$.



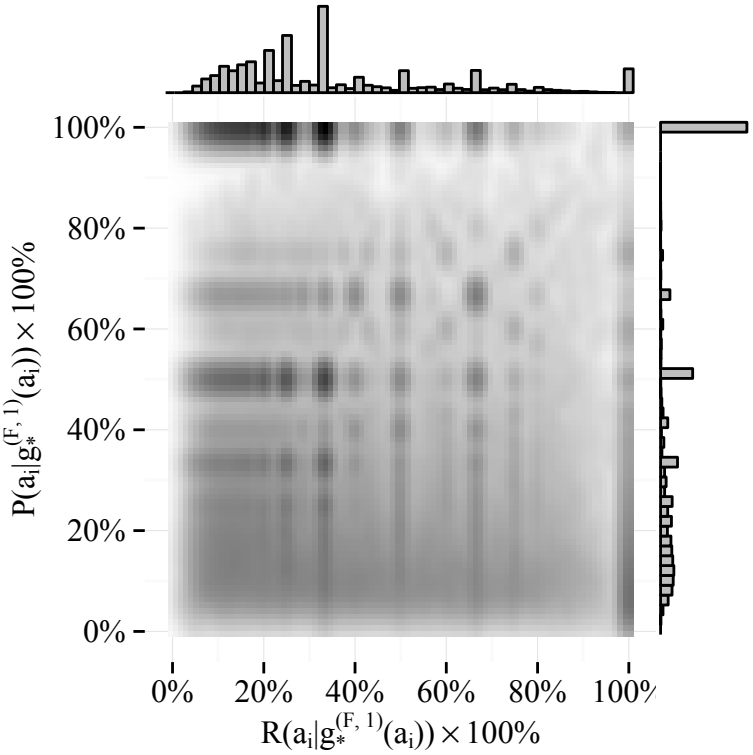
0.42 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 45$.



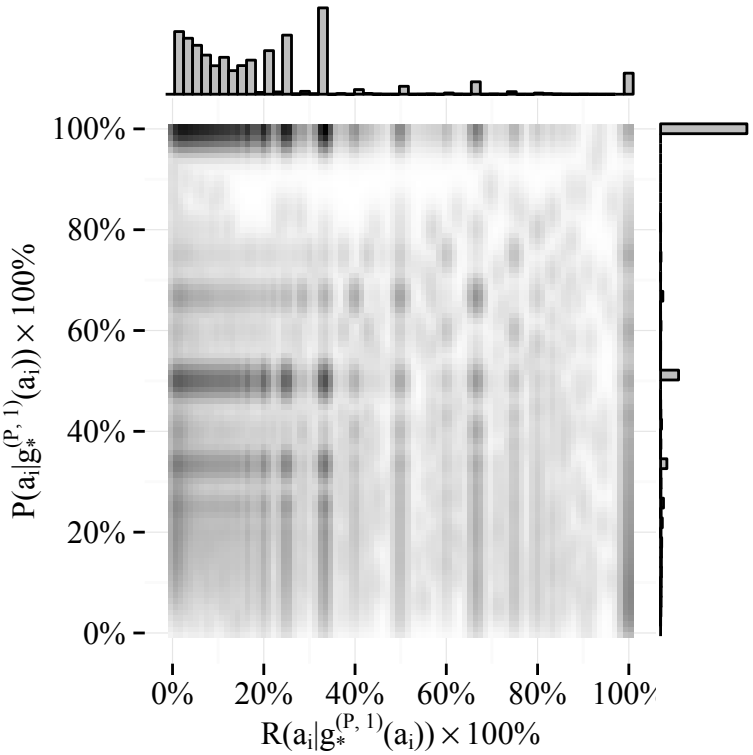
0.43 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 70$**.**



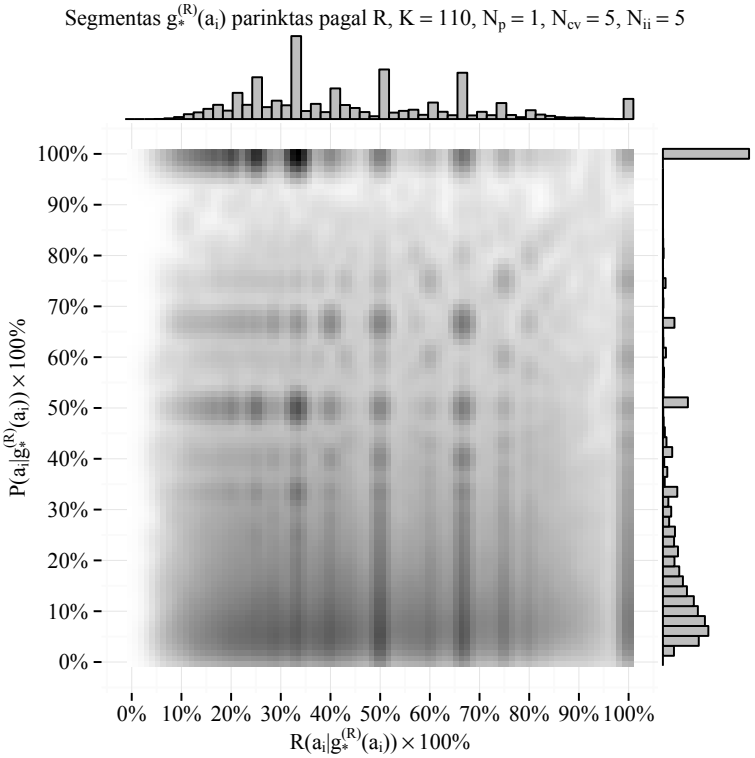
0.44 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 70$**.**



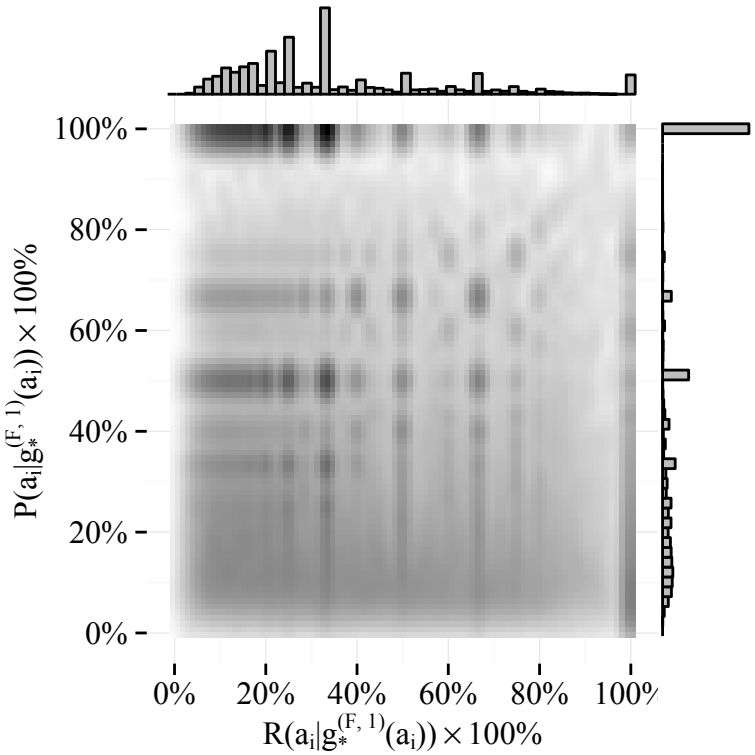
0.45 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 70$**.**



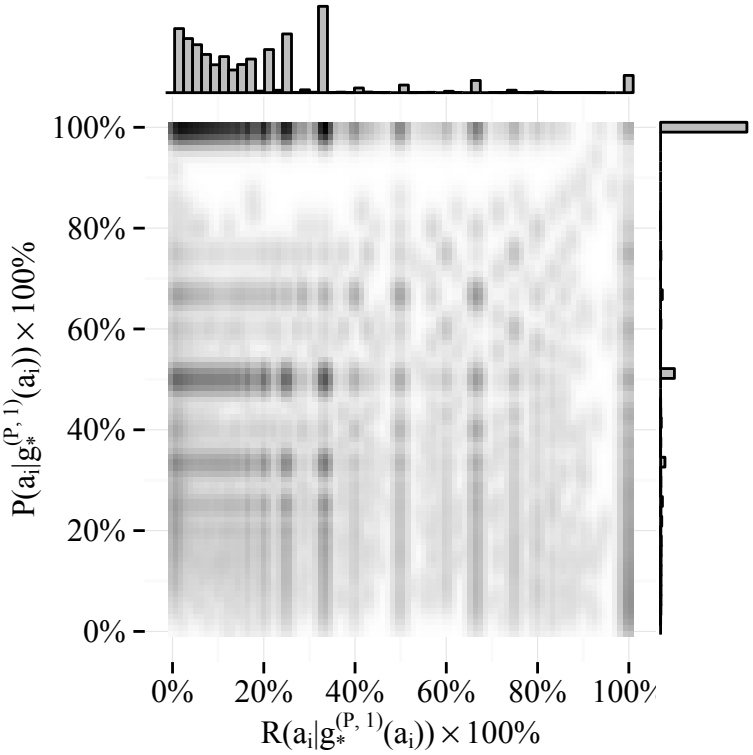
0.46 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 110$**.**



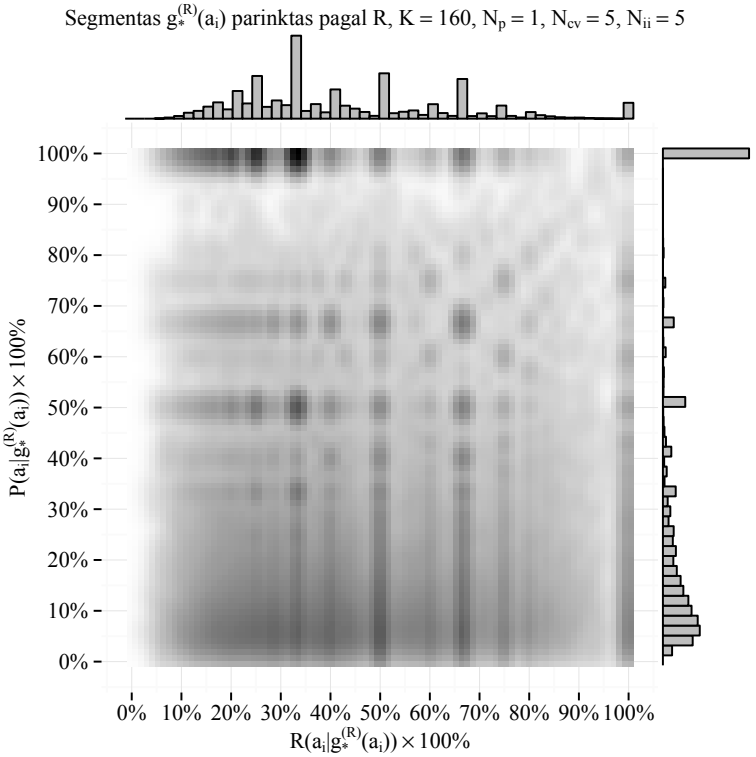
0.47 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 110$**.**



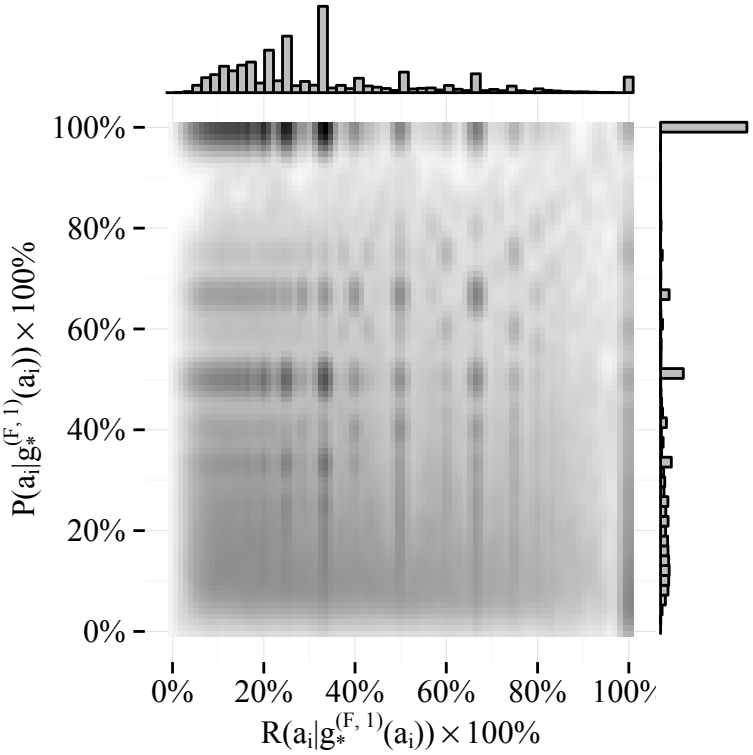
0.48 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 110$**.**



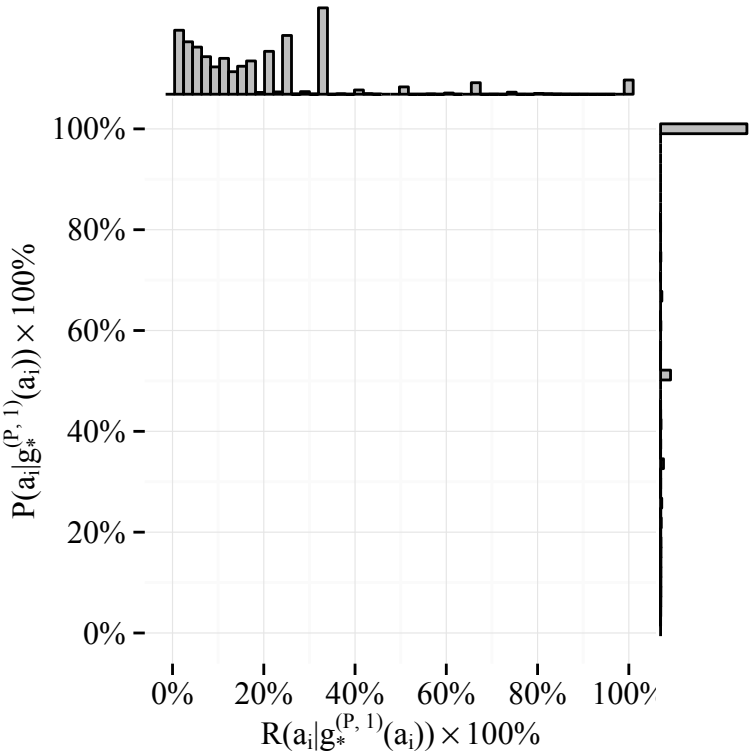
0.49 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 160$**.**



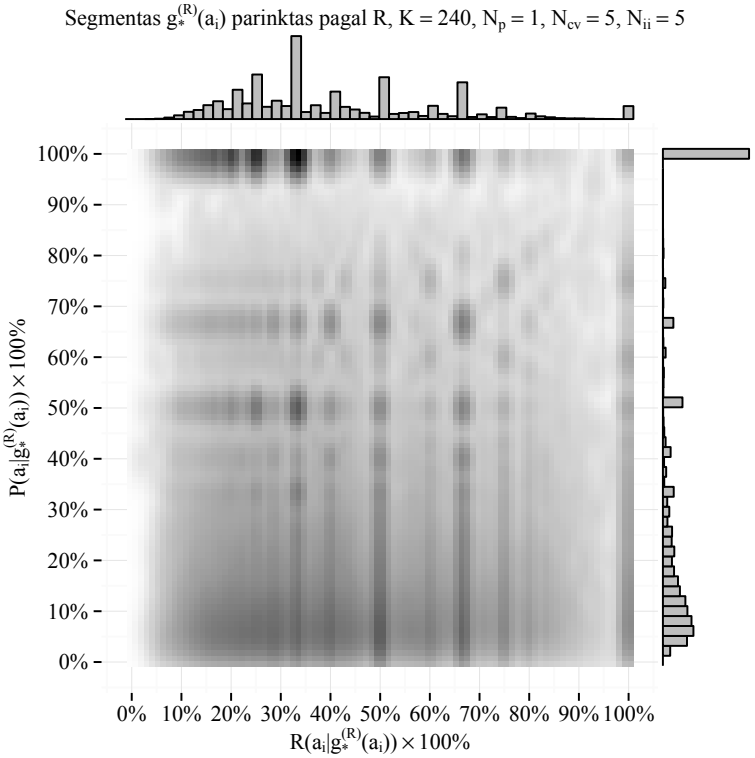
0.50 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 160$**.**



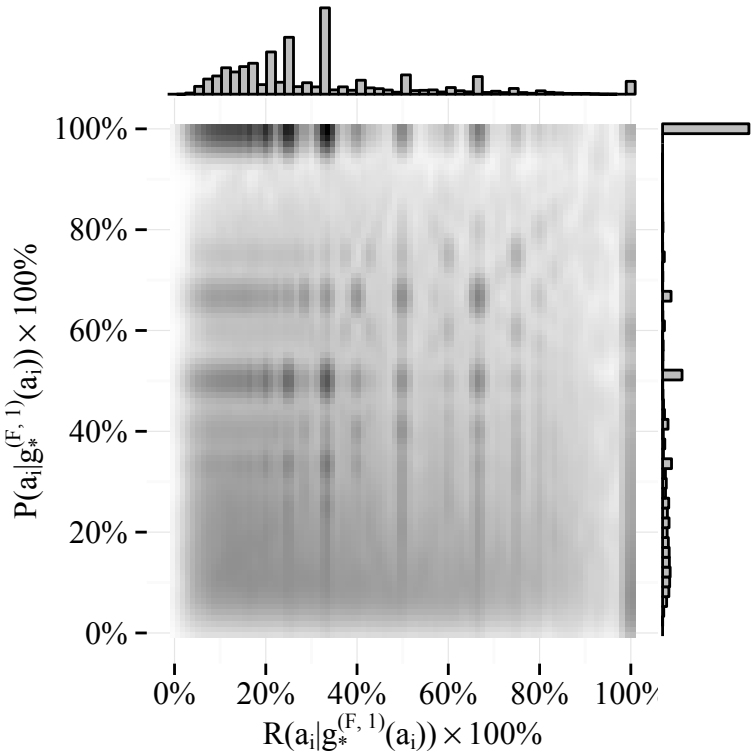
0.51 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 160$**.**



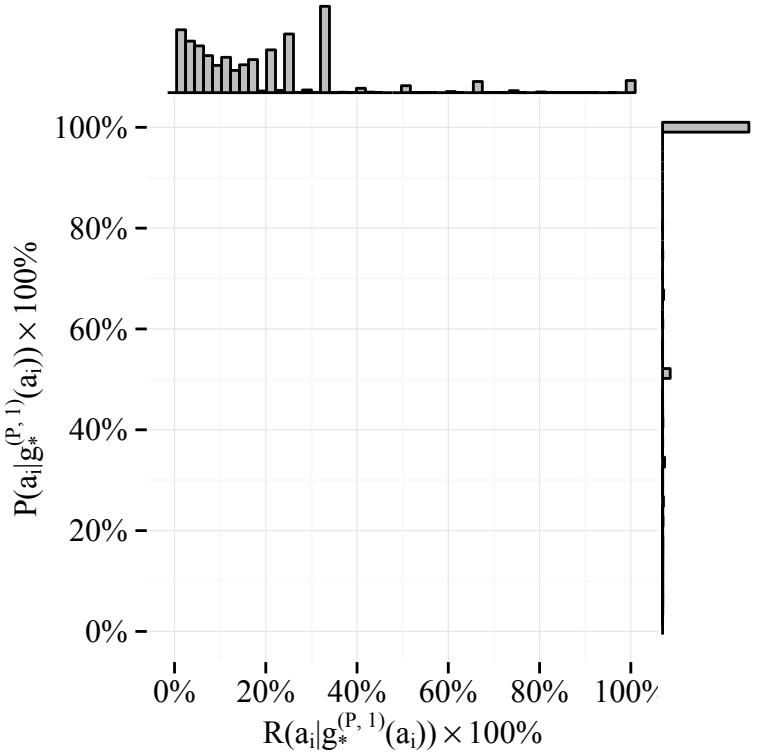
0.52 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 240$**.**



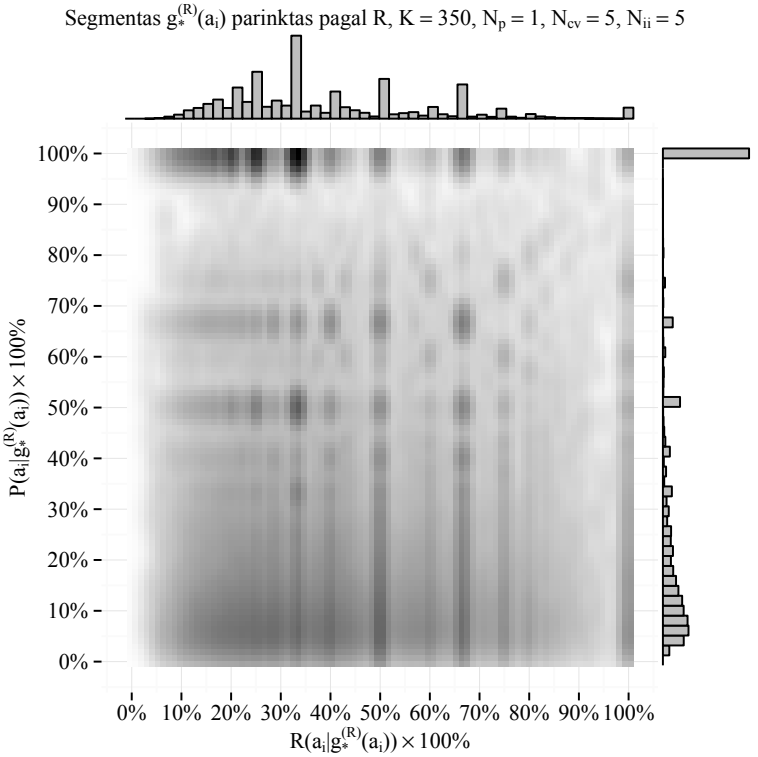
0.53 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 240$**.**



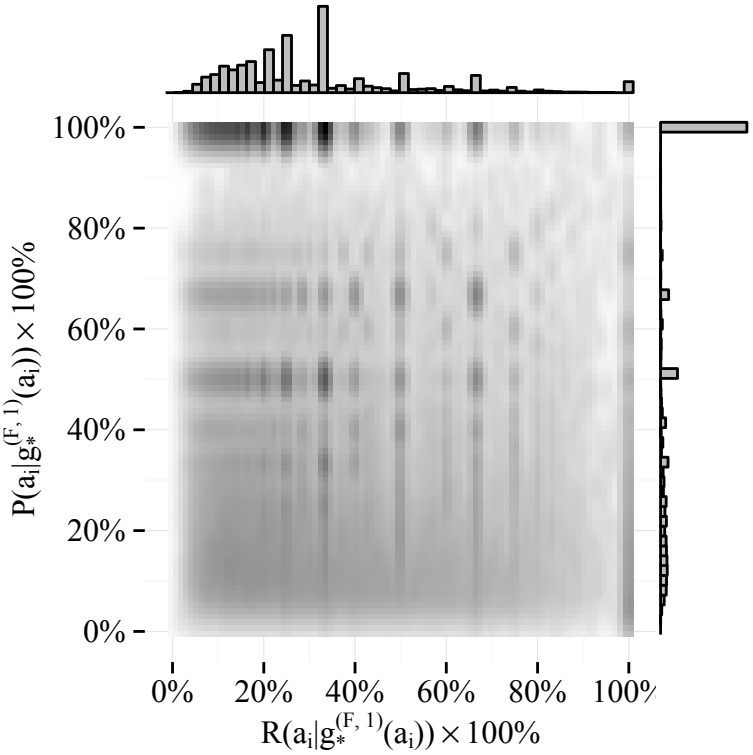
0.54 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 240$**.**



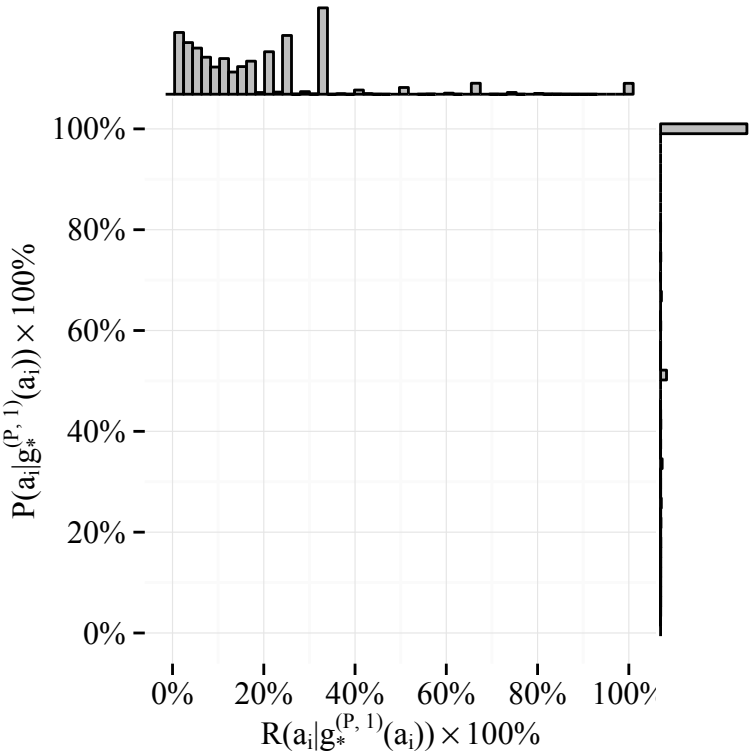
0.55 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 350$**.**



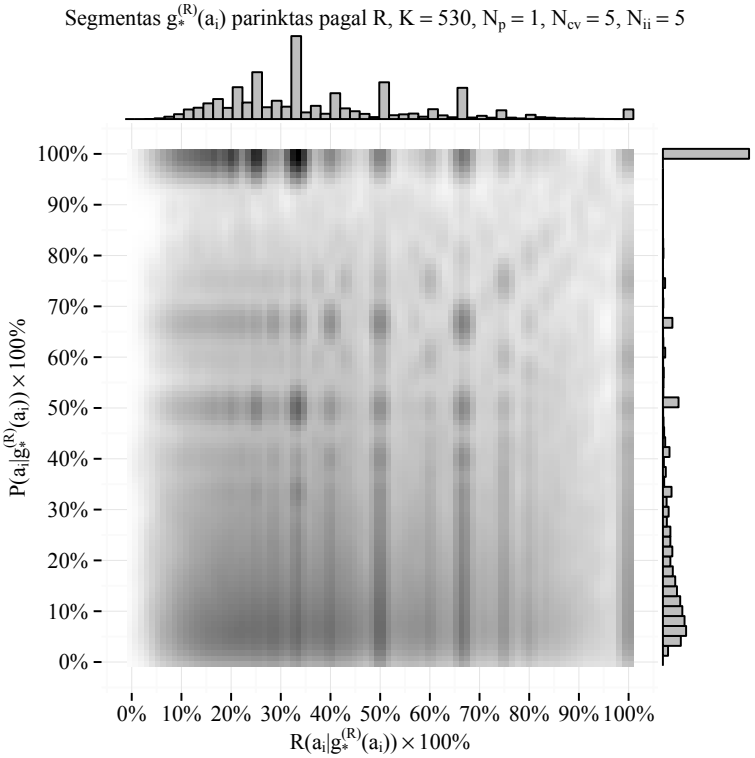
0.56 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 350$**.**



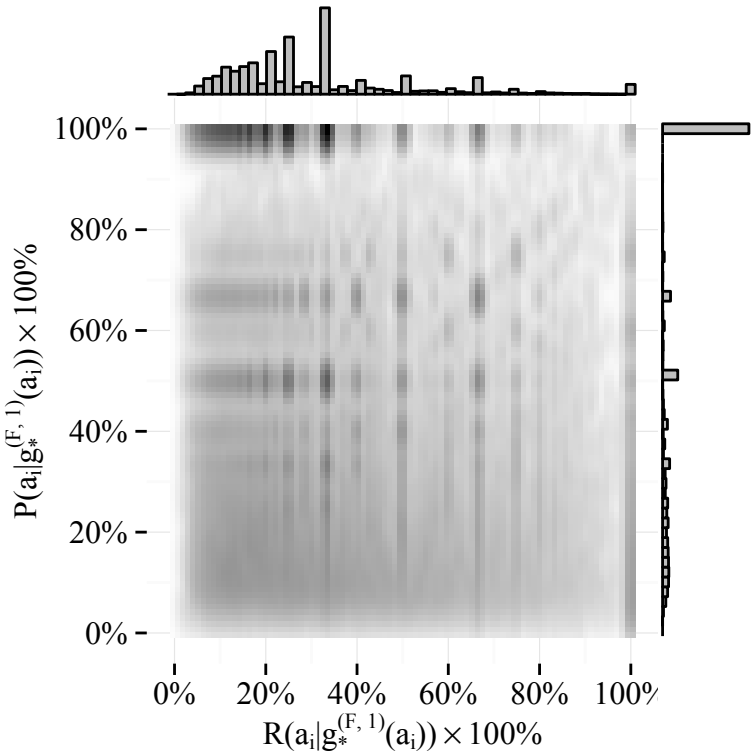
0.57 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 350$**.**



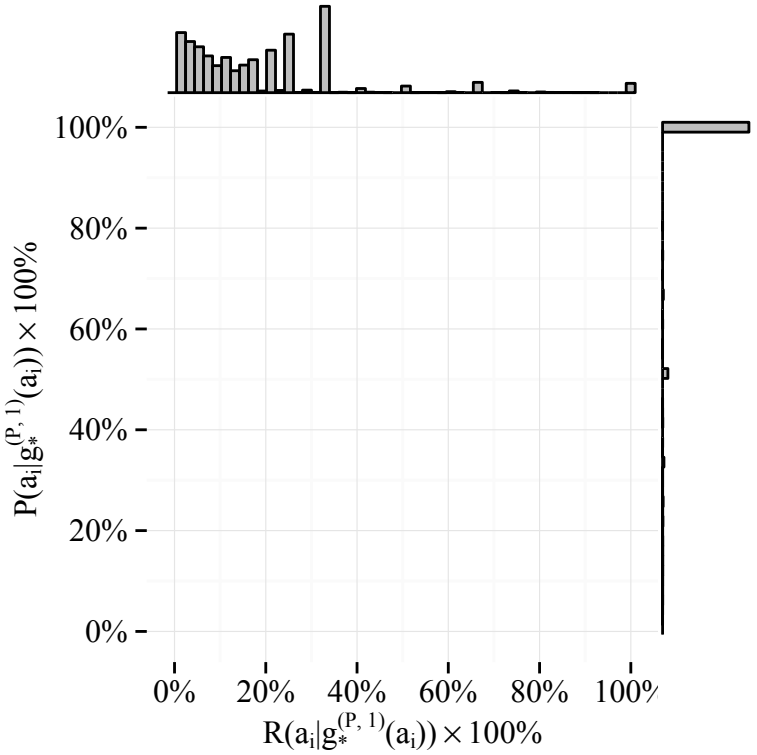
0.58 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 530$**.**



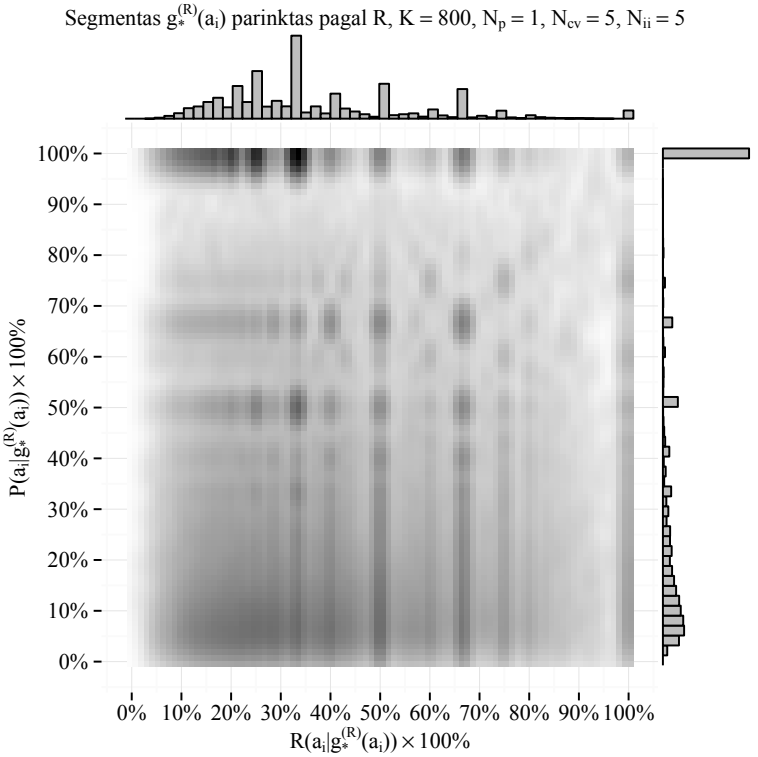
0.59 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 530$**.**



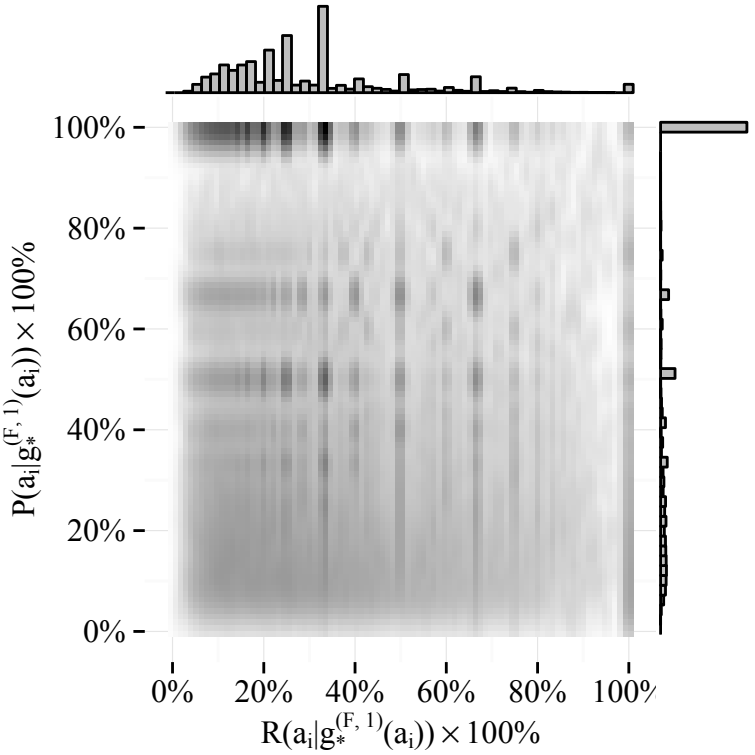
0.60 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 530$**.**



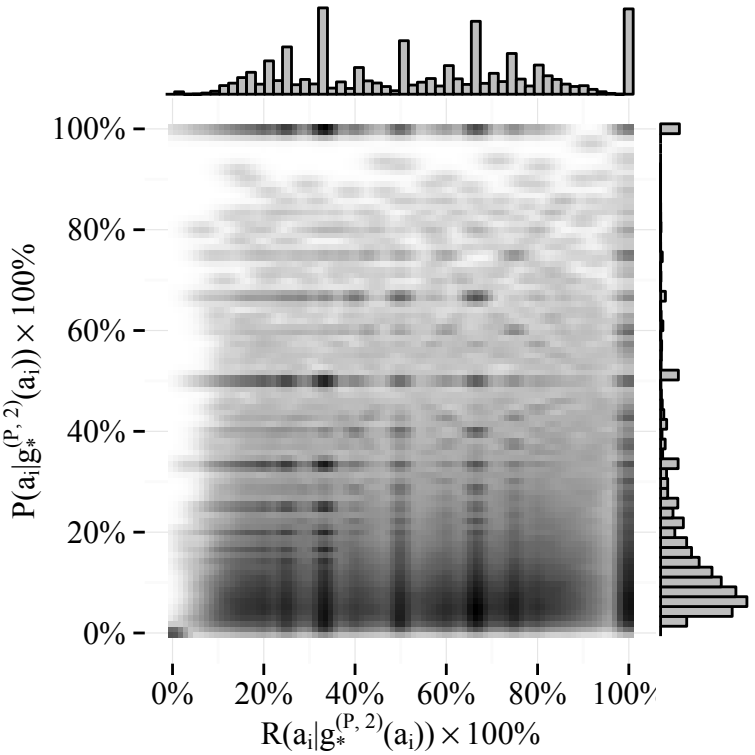
0.61 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai be apribojimų, $K = 800$.



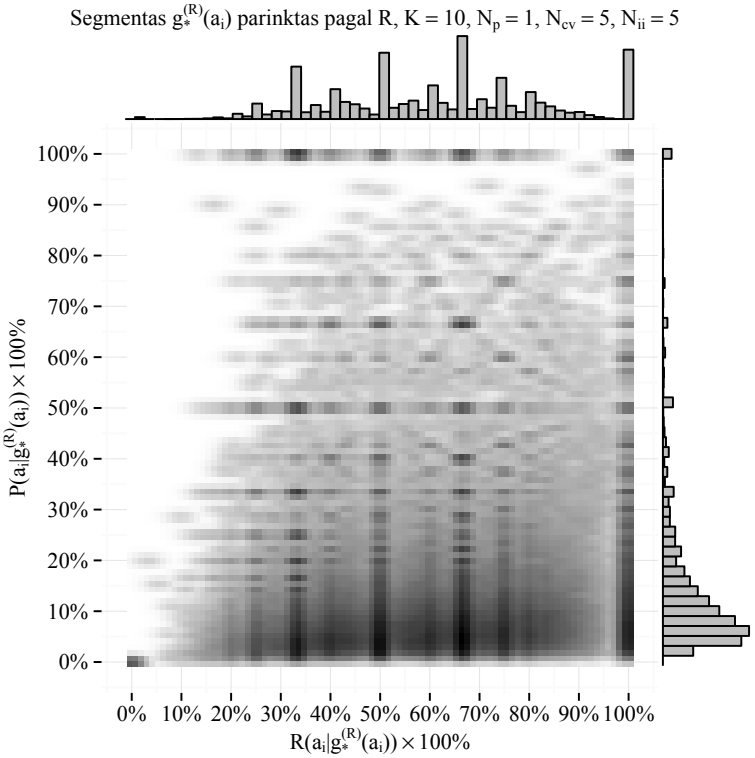
0.62 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai be apribojimų, $K = 800$.



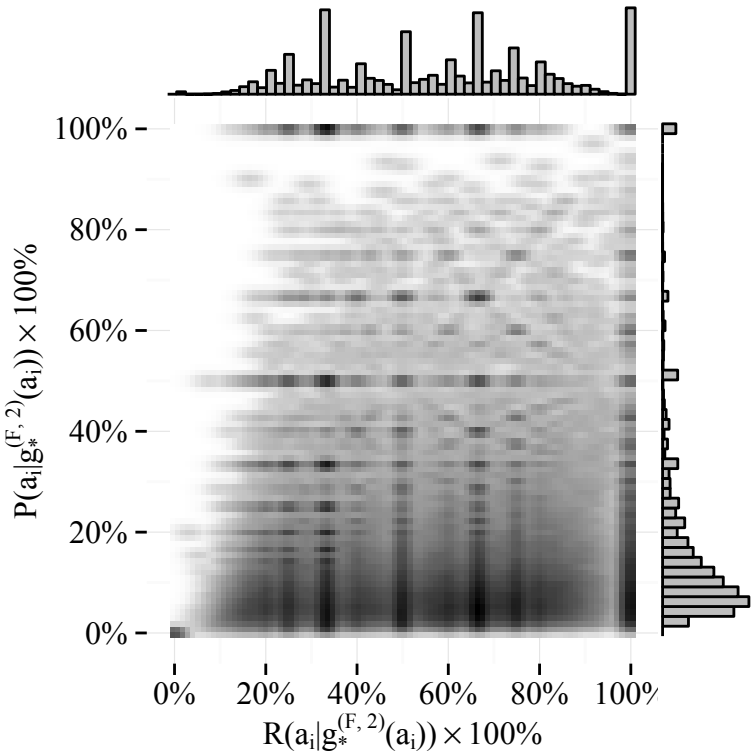
0.63 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai be apribojimų, $K = 800$**.**



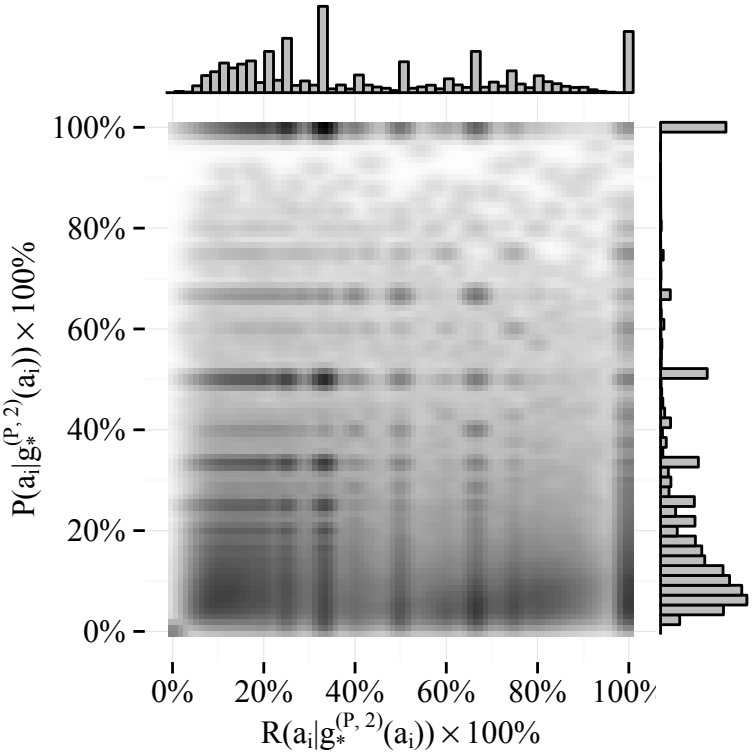
0.64 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 10$**.**



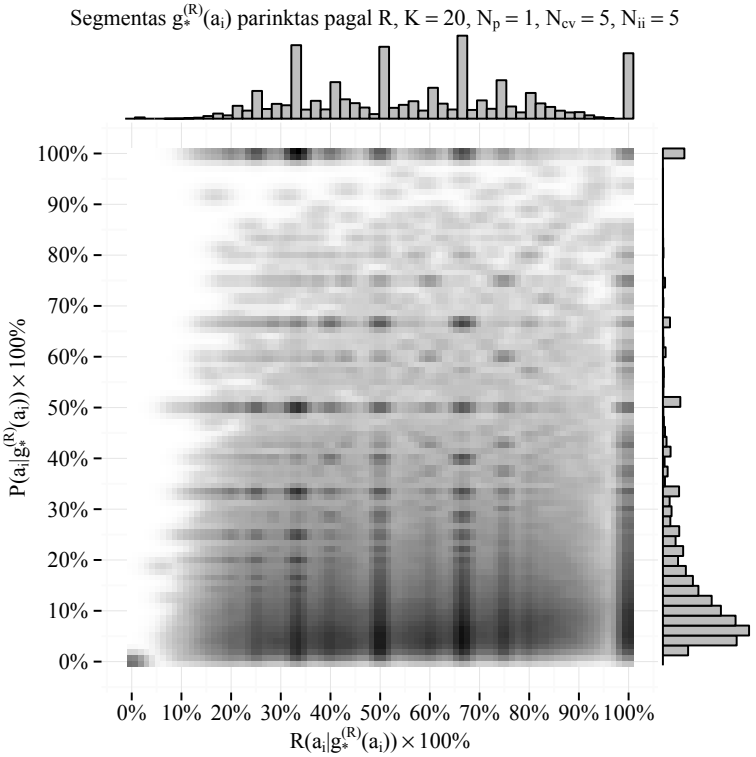
0.65 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 10$**.**



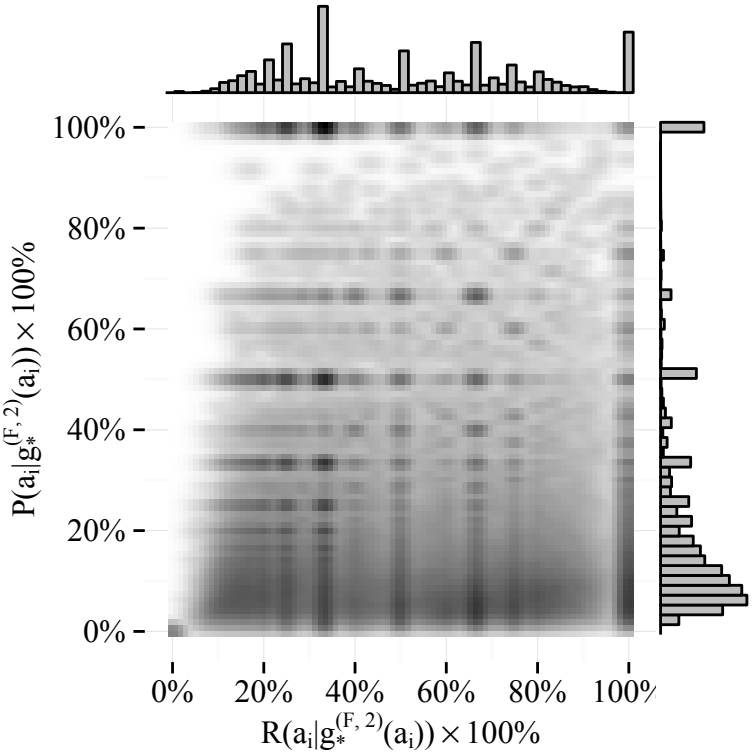
0.66 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 10$**.**



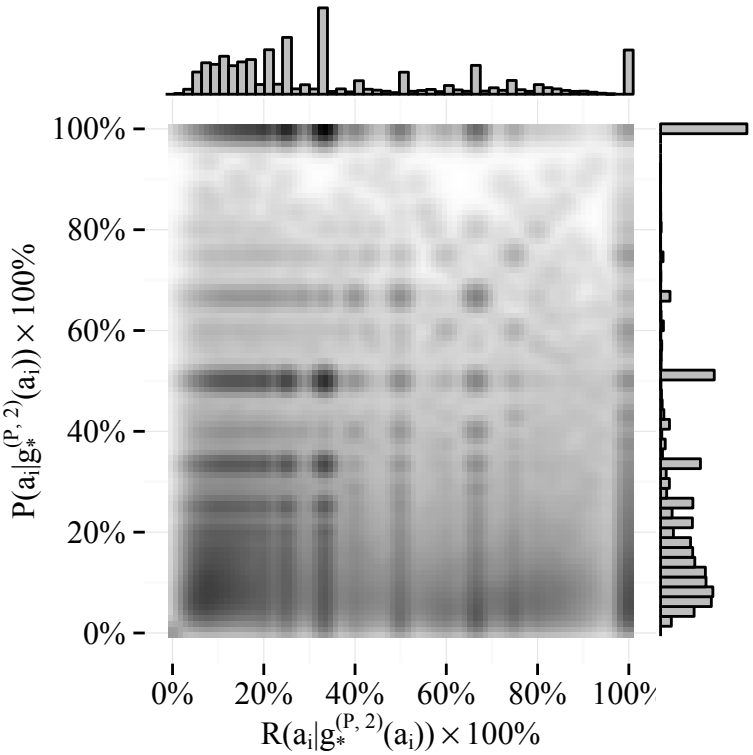
0.67 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 20$.



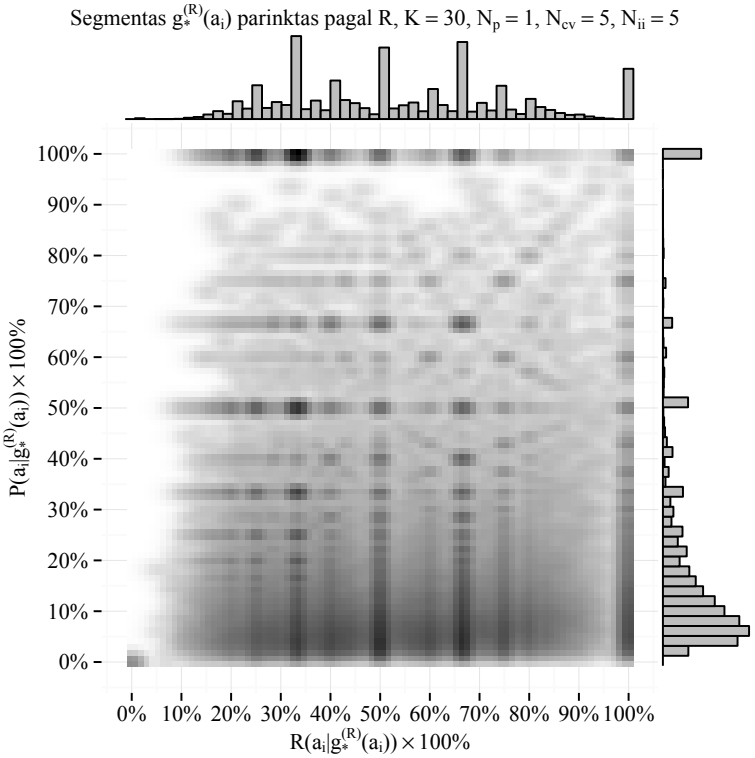
0.68 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 20$.



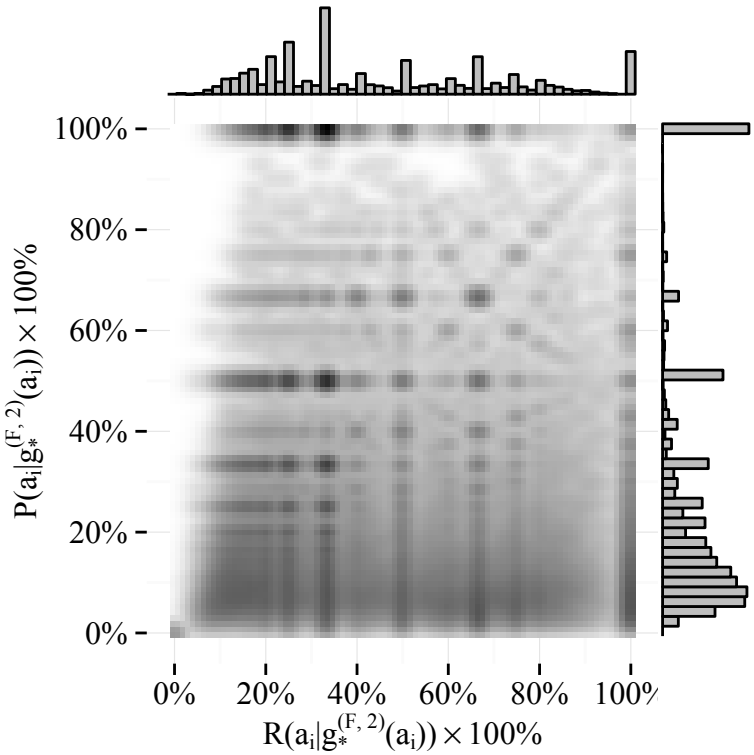
0.69 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 20$.



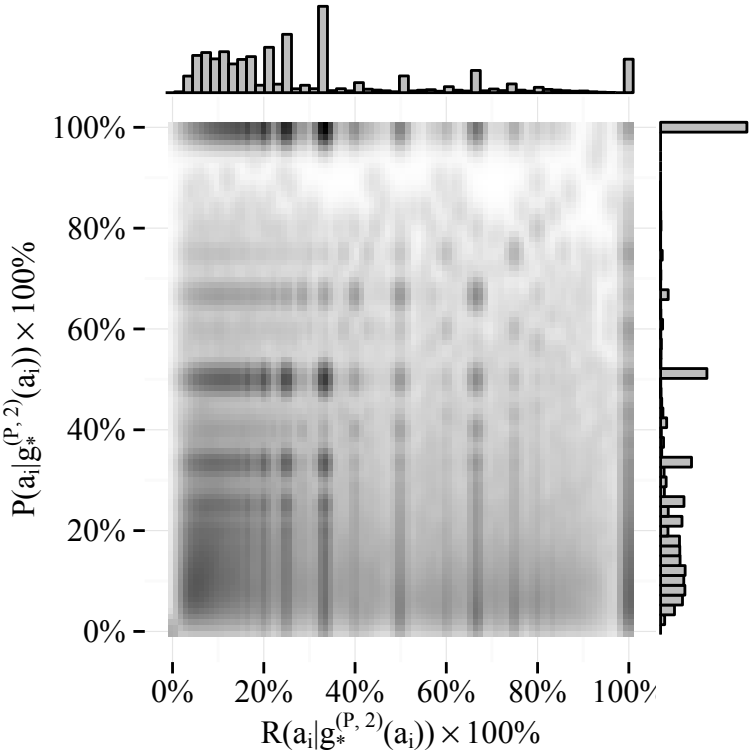
0.70 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 30$.



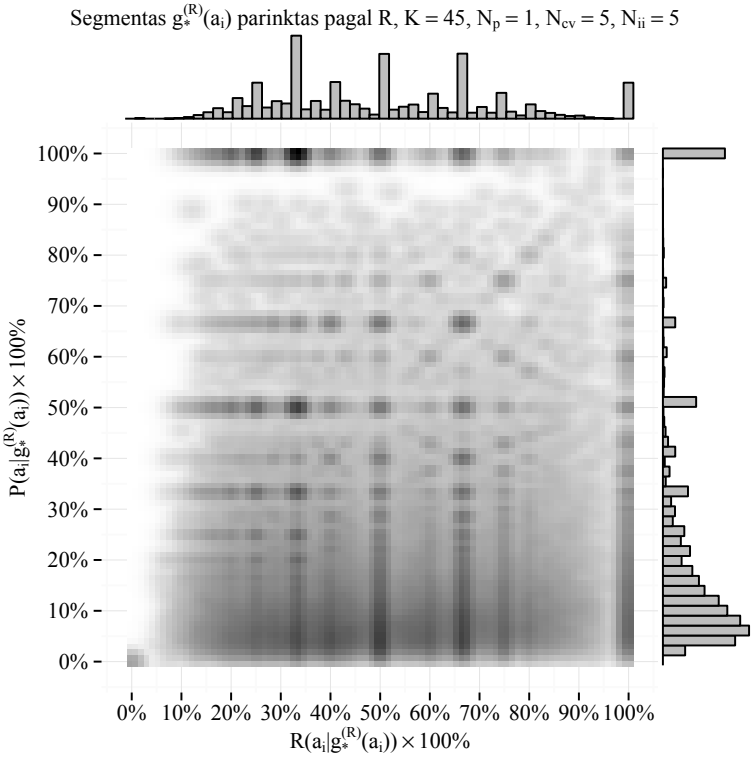
0.71 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 30$.



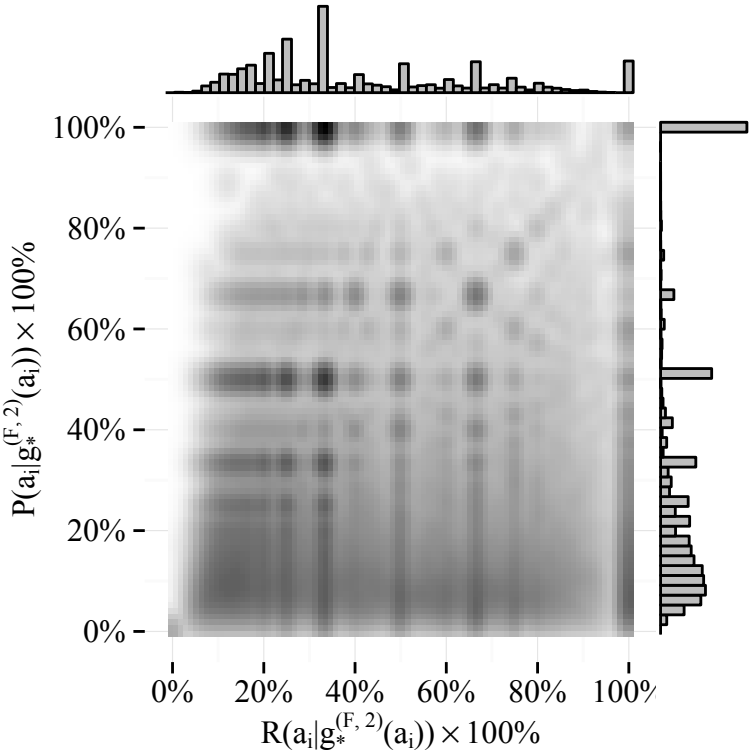
0.72 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 30$.



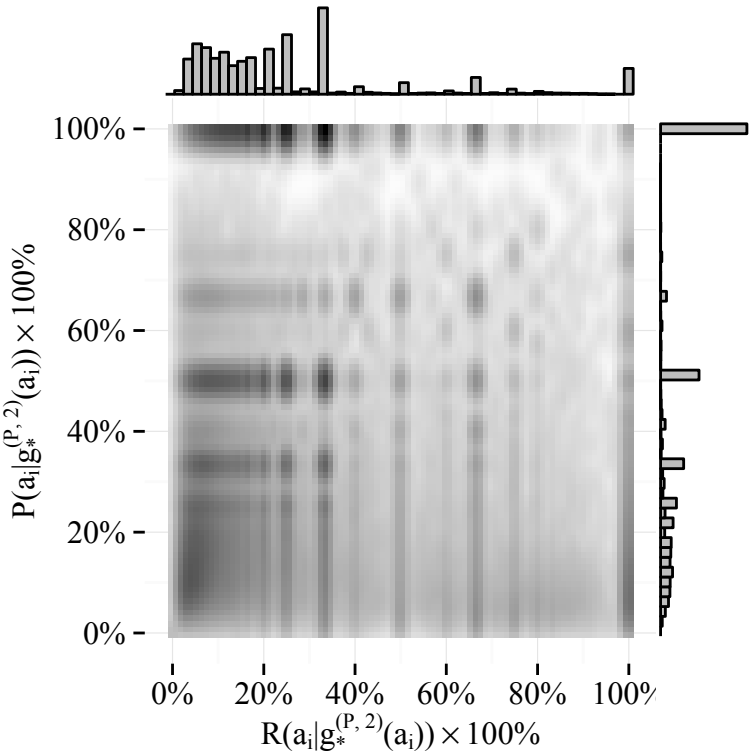
0.73 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 45$.



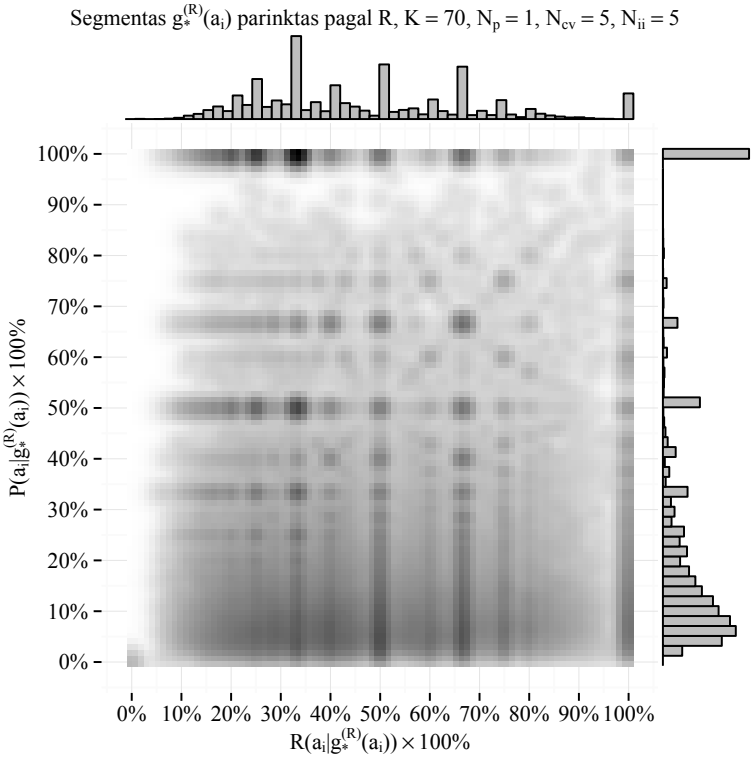
0.74 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 45$.



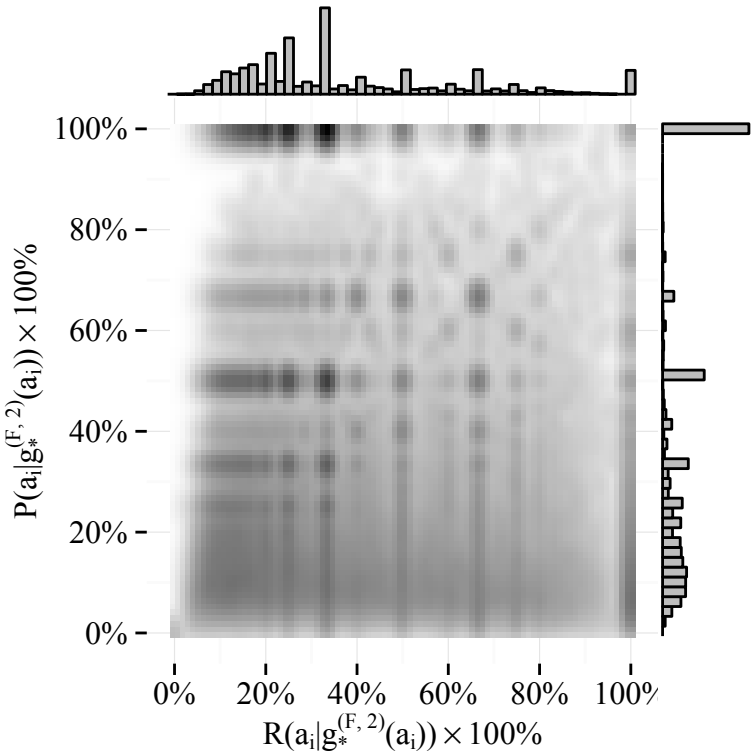
0.75 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 45$.



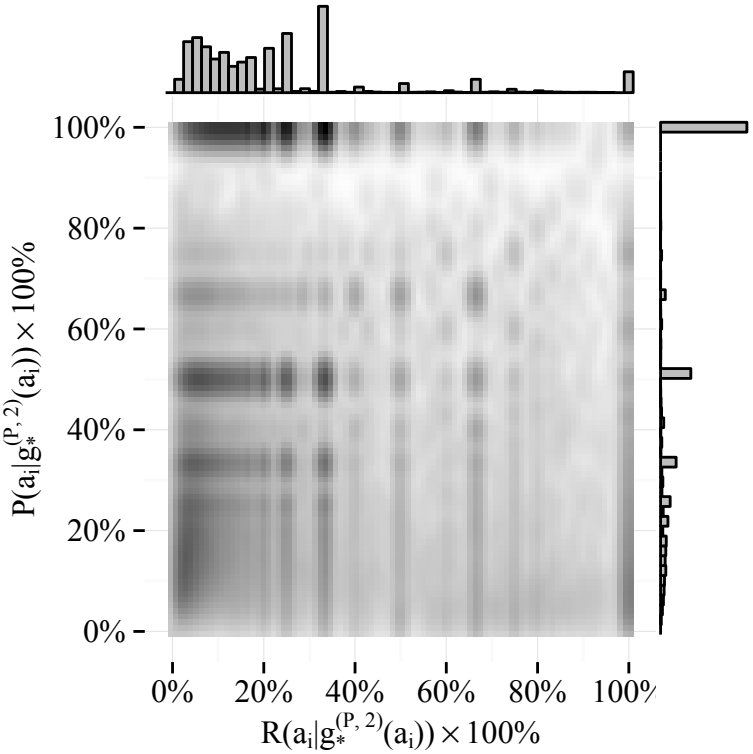
0.76 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 70$.



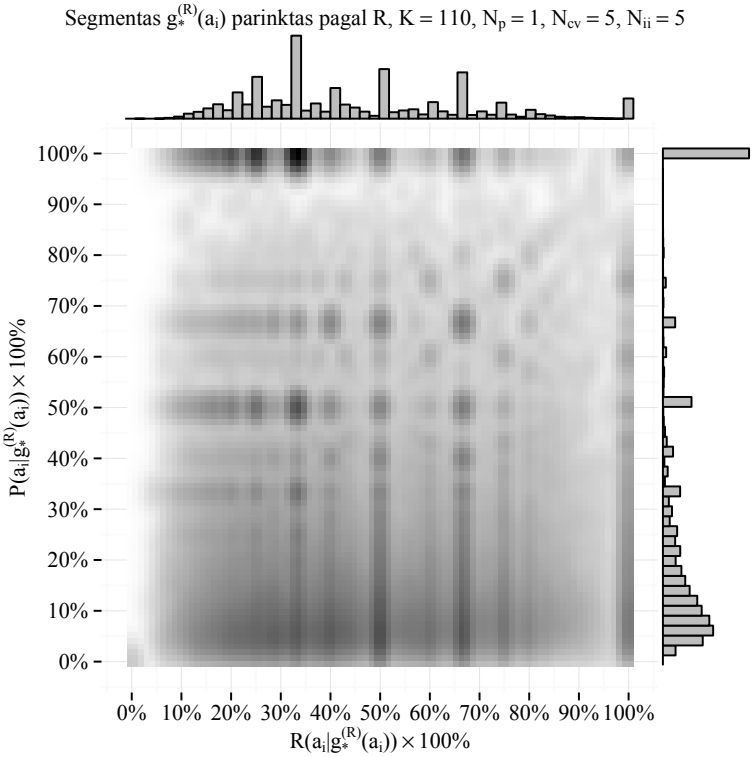
0.77 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 70$**.**



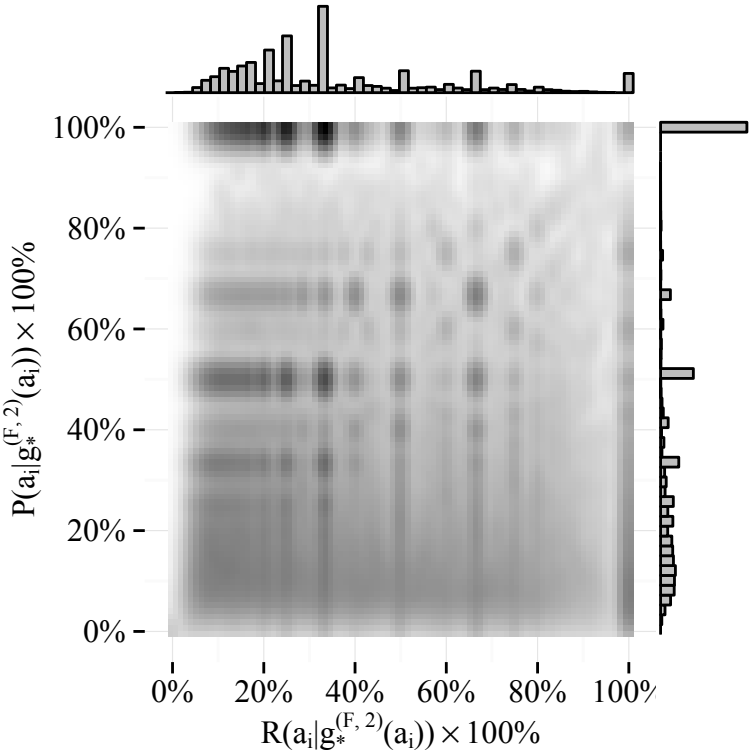
0.78 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 70$**.**



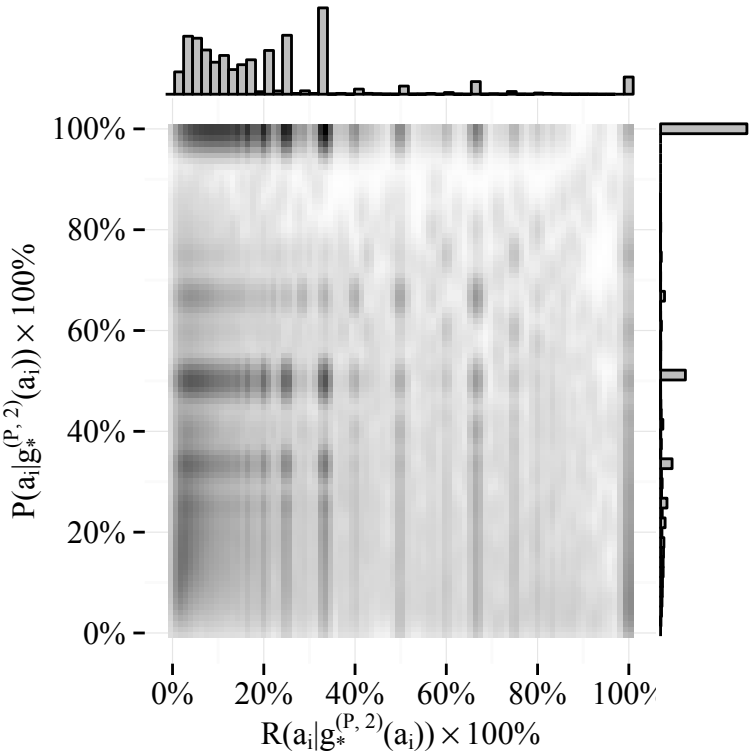
0.79 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 110$**.**



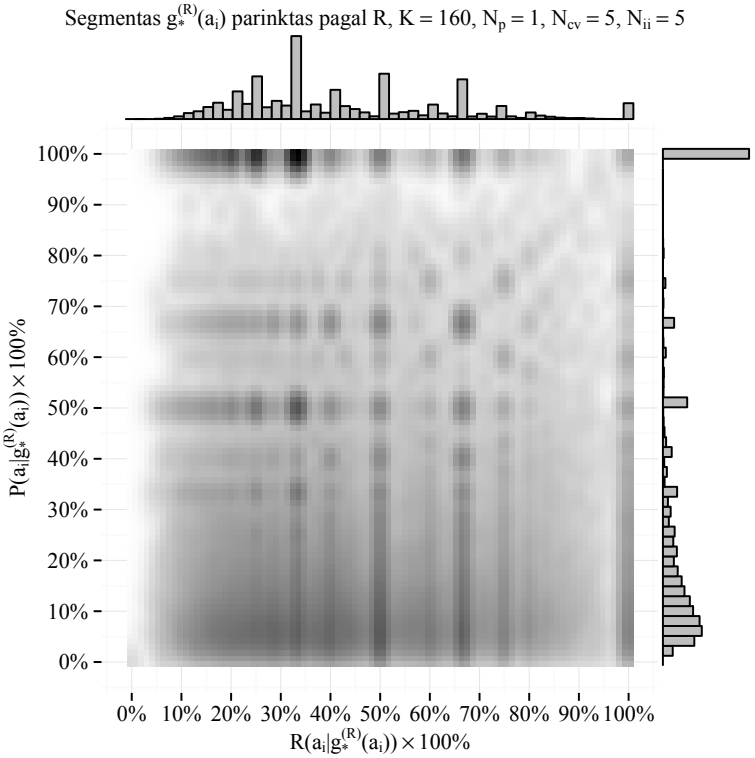
0.80 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 110$**.**



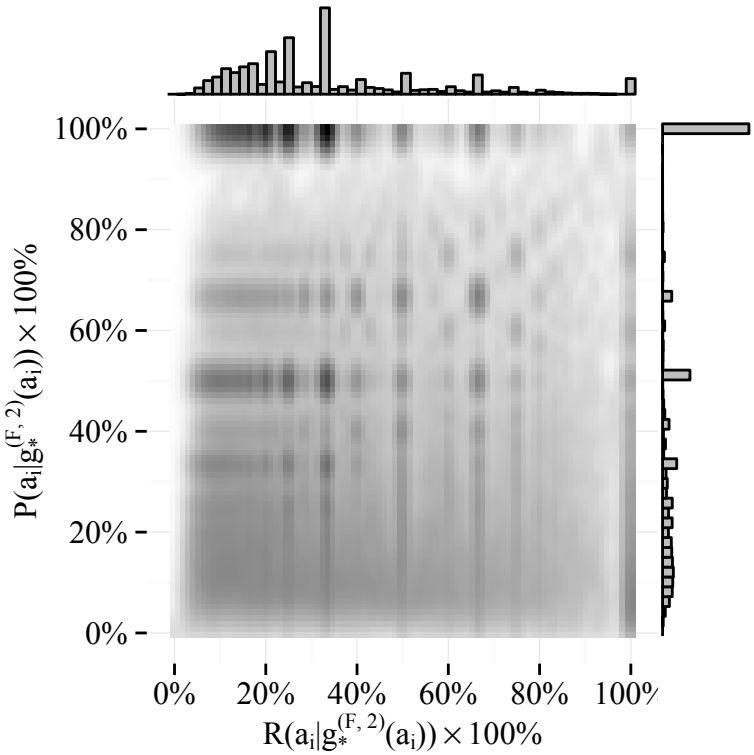
0.81 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 110$**.**



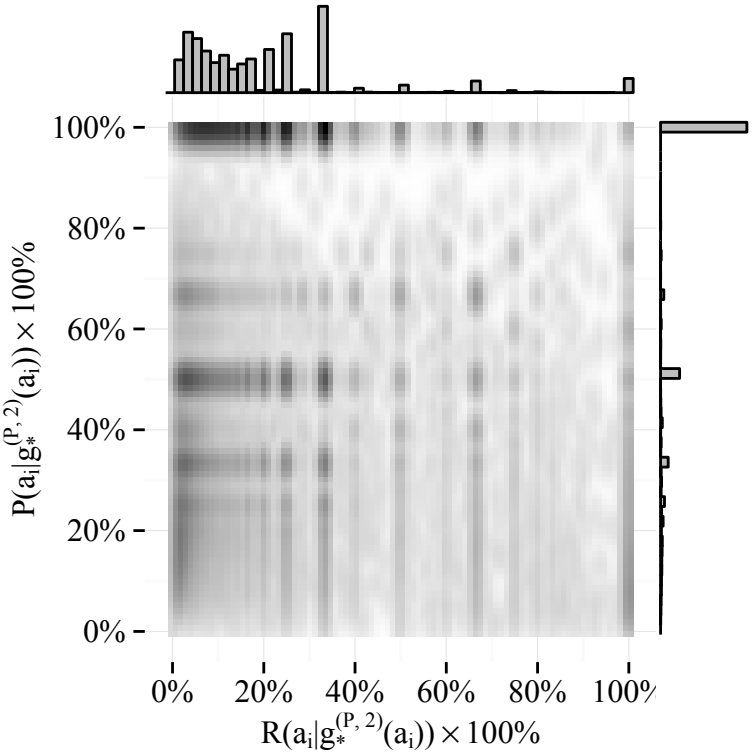
0.82 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 160$**.**



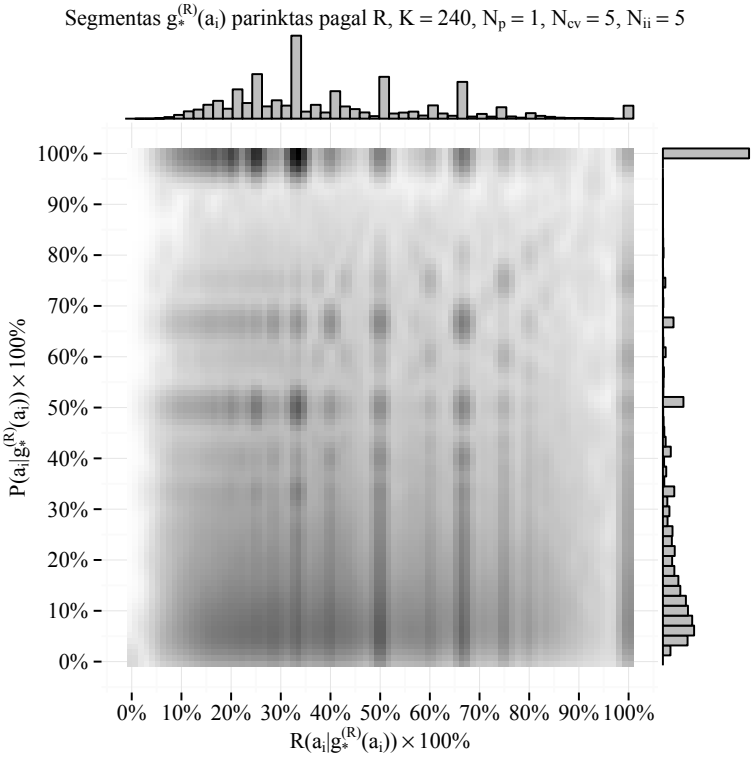
0.83 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 160$**.**



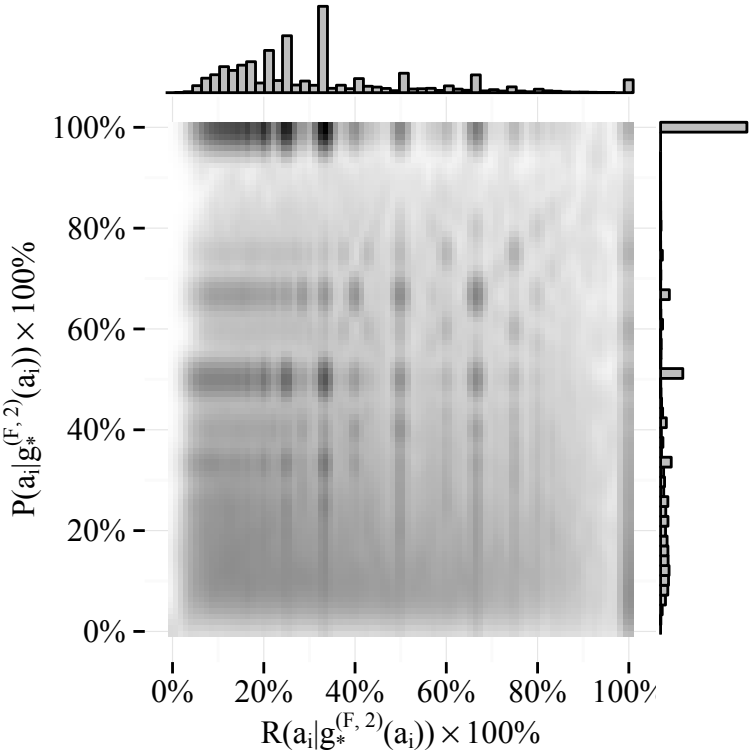
0.84 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 160$**.**



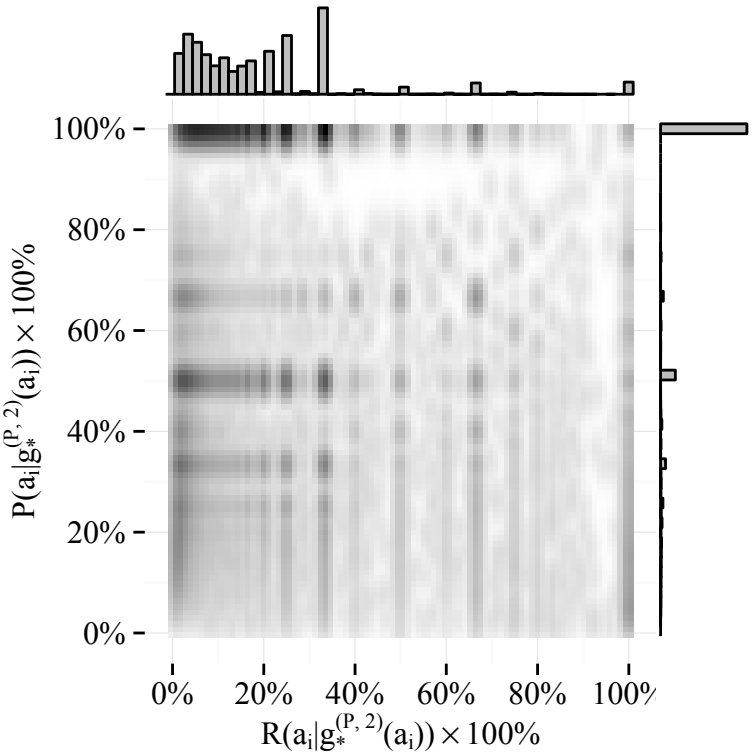
0.85 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 240$**.**



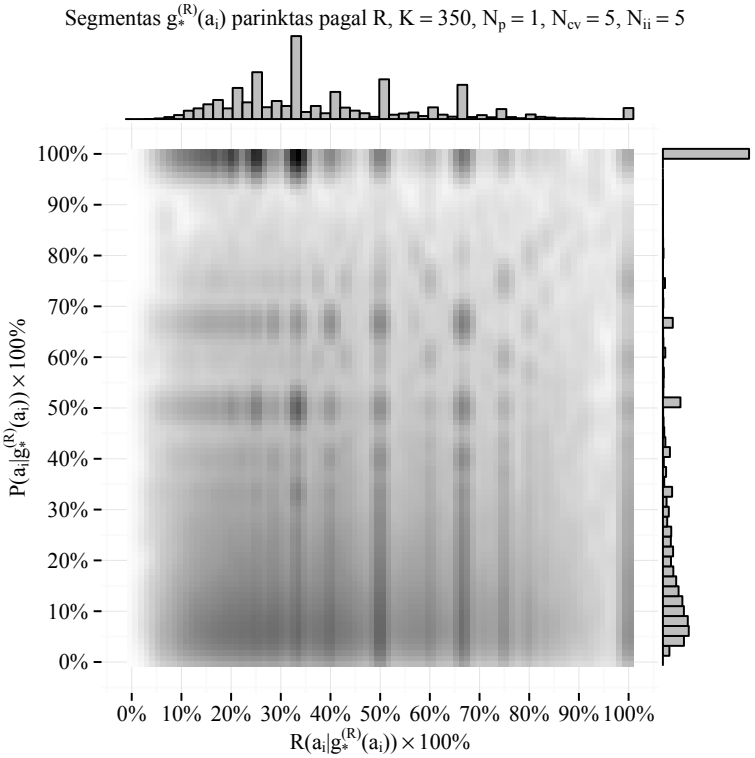
0.86 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 240$**.**



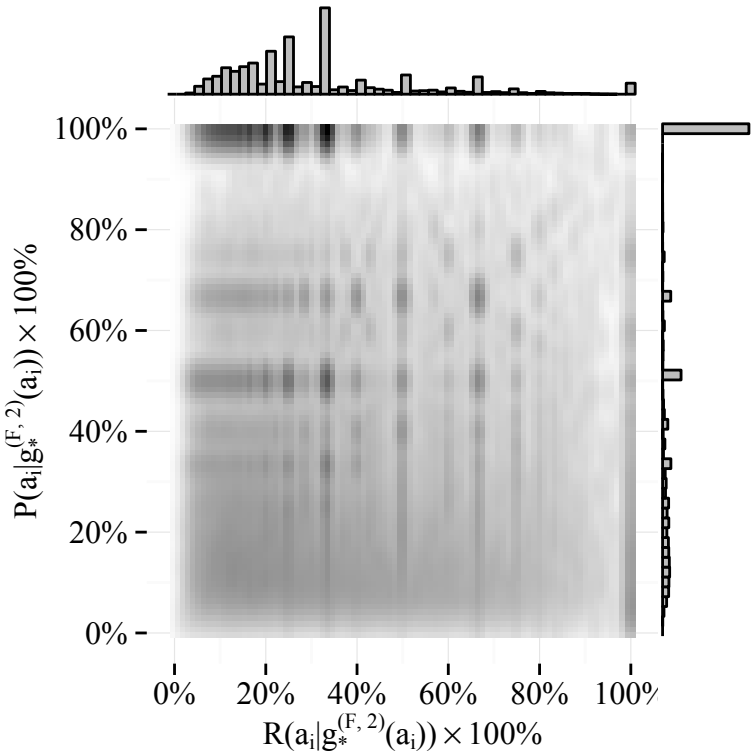
0.87 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 240$**.**



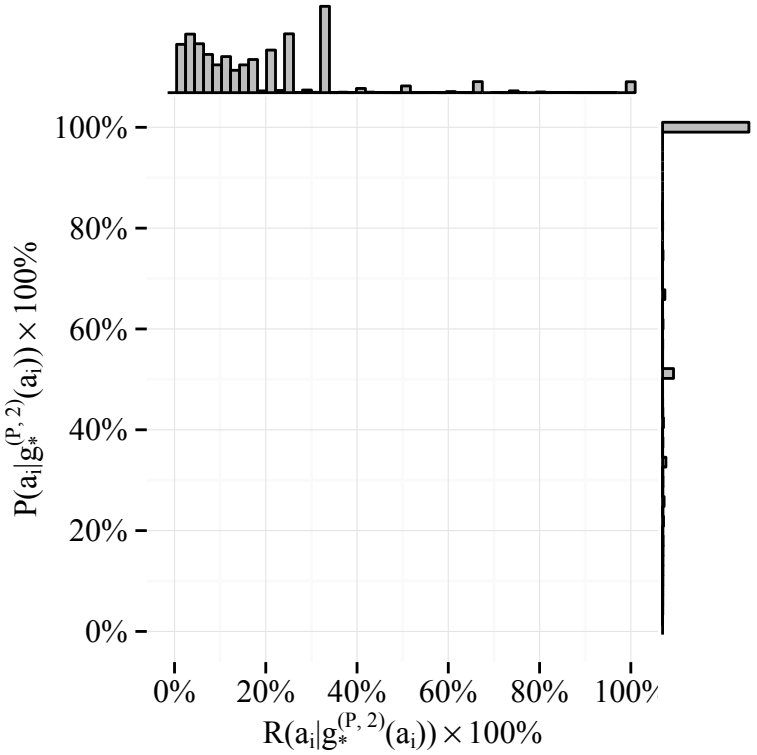
0.88 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 350$**.**



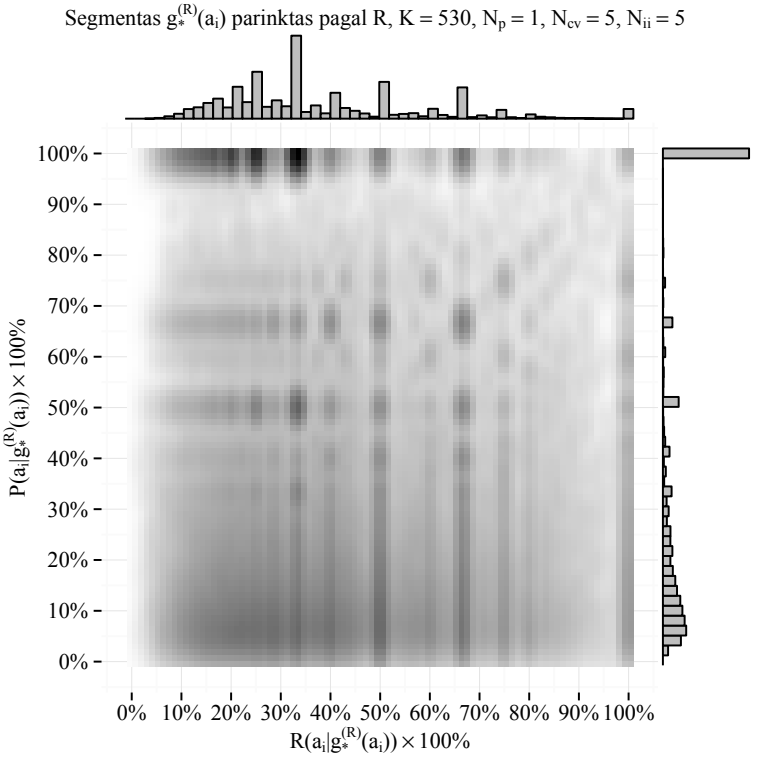
0.89 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 350$**.**



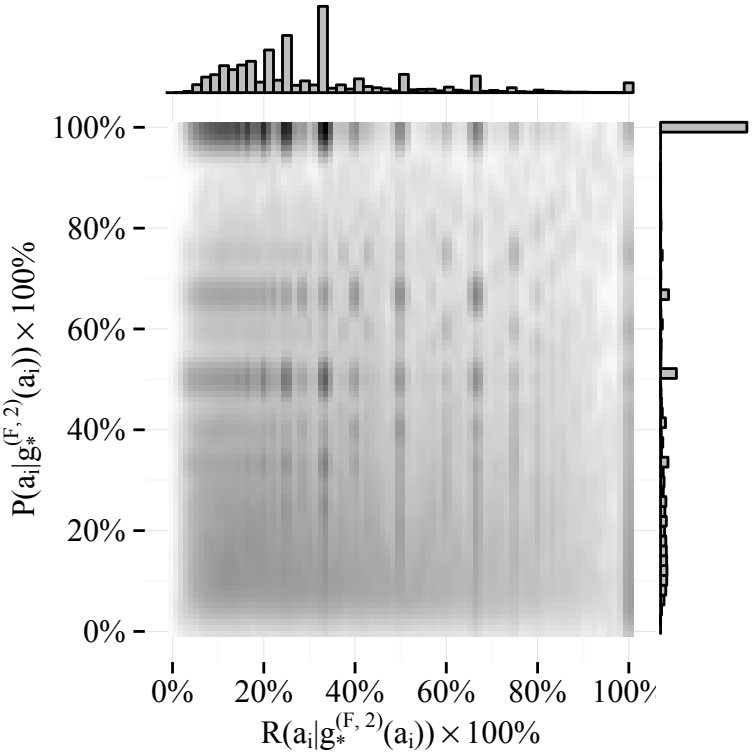
0.90 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 350$**.**



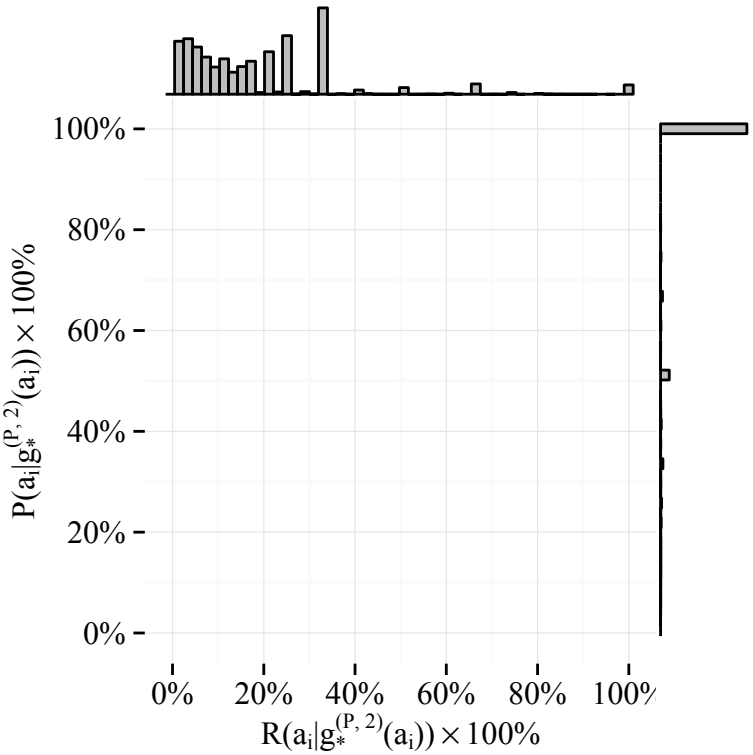
0.91 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 530$**.**



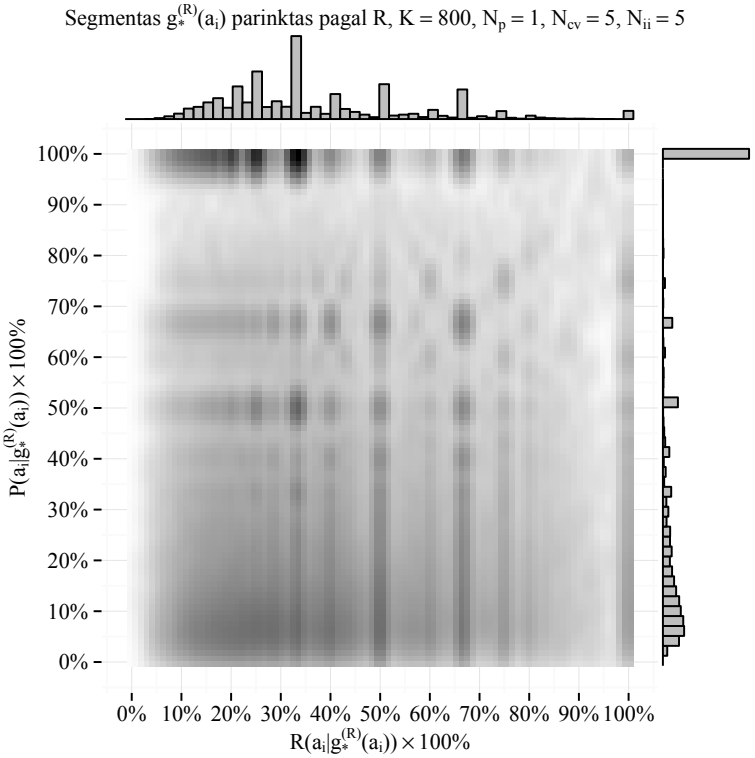
0.92 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 530$**.**



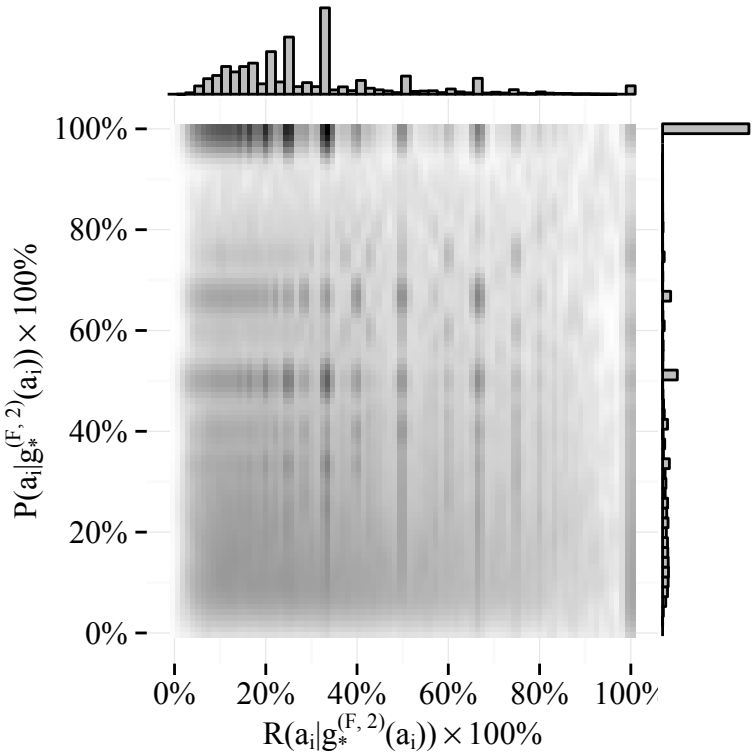
0.93 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 530$**.**



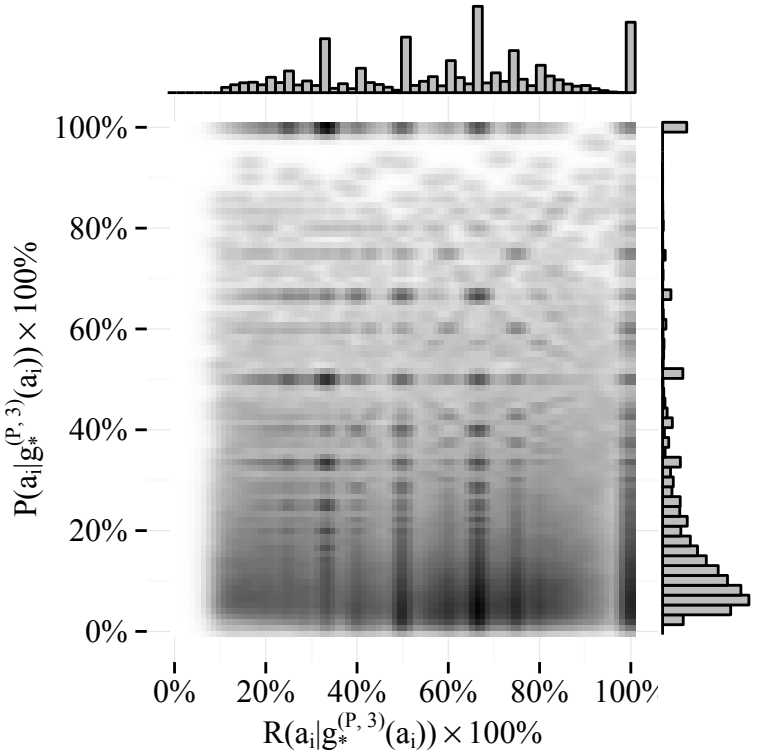
0.94 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 800$**.**



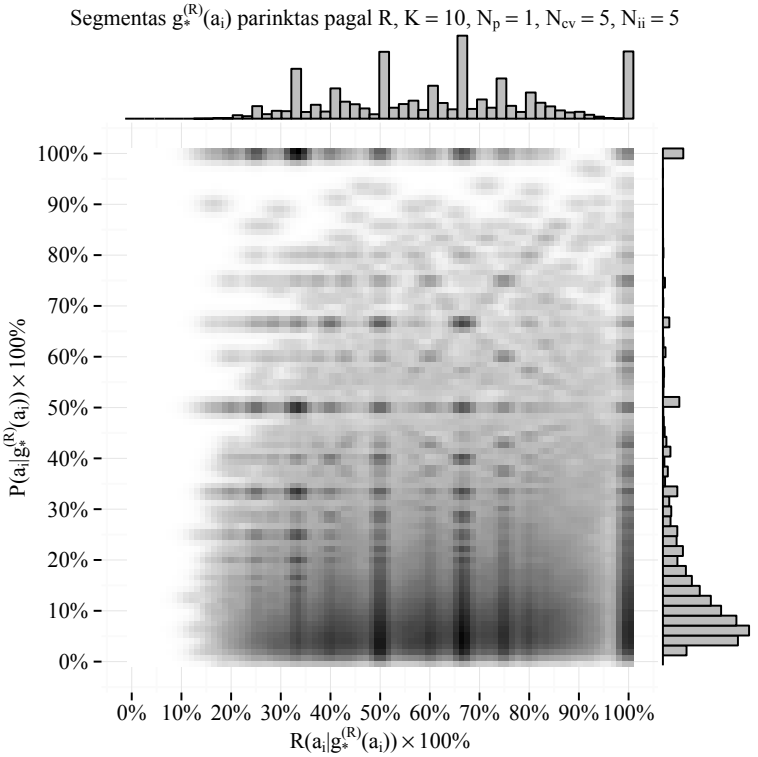
0.95 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 800$**.**



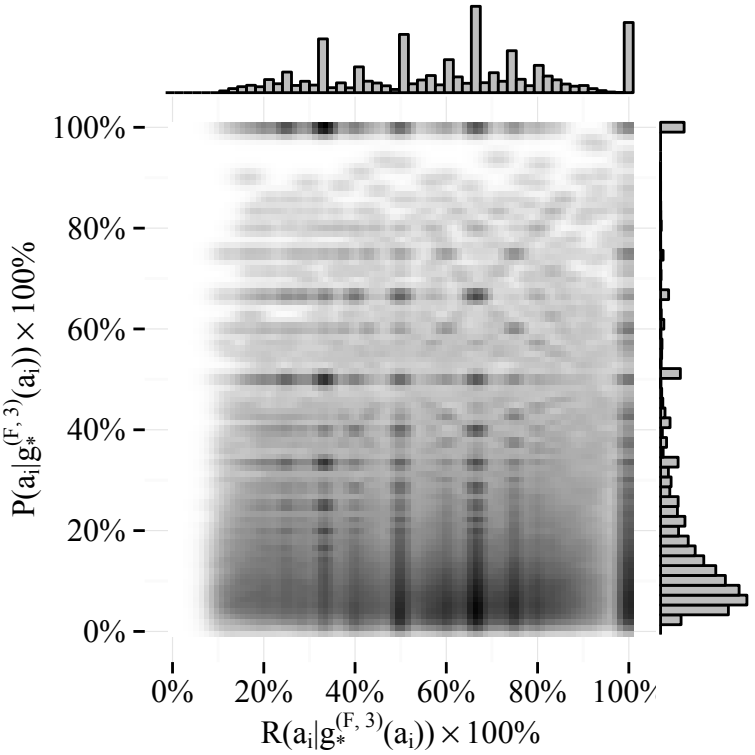
0.96 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal segmento vartotojų skaičių, $K = 800$**.**



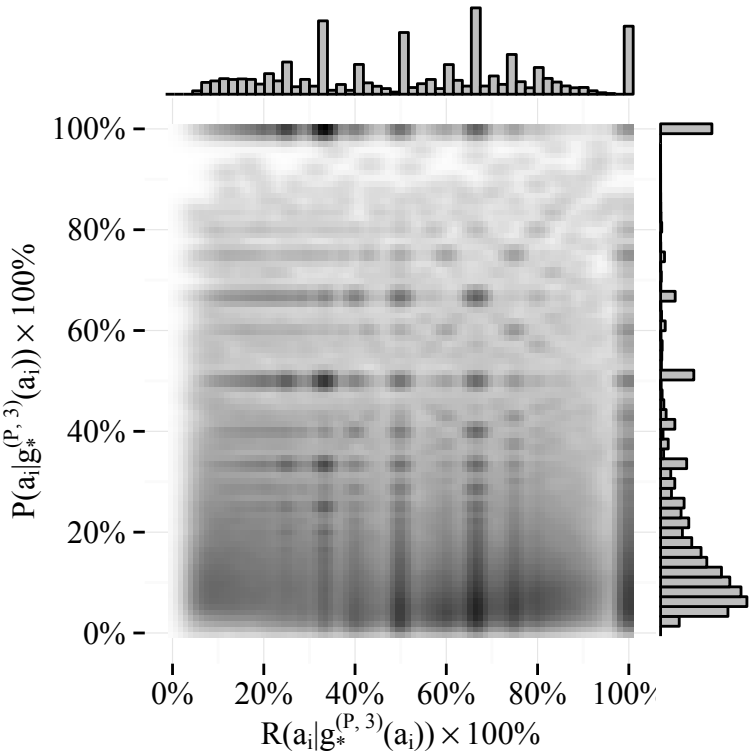
0.97 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 10$**.**



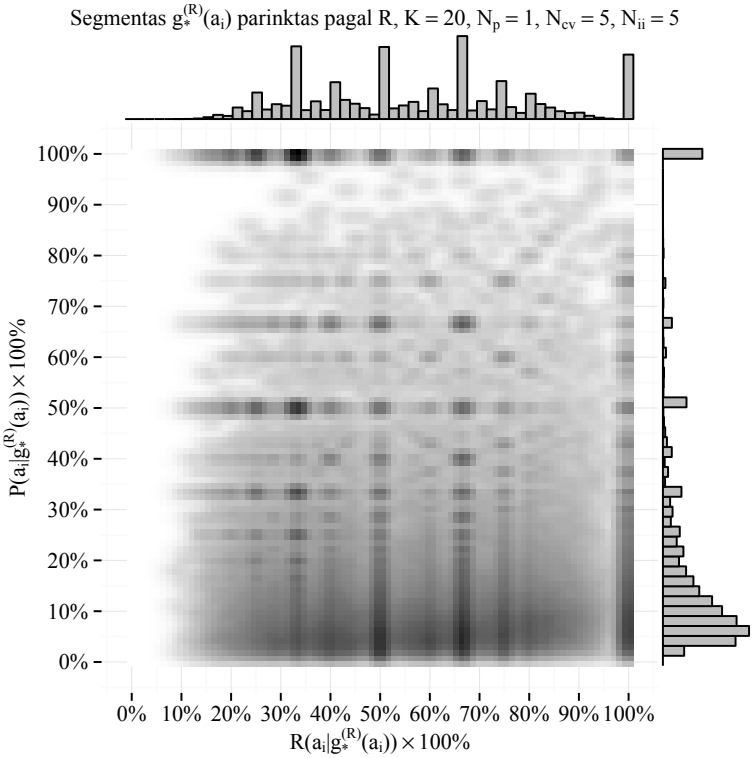
0.98 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 10$**.**



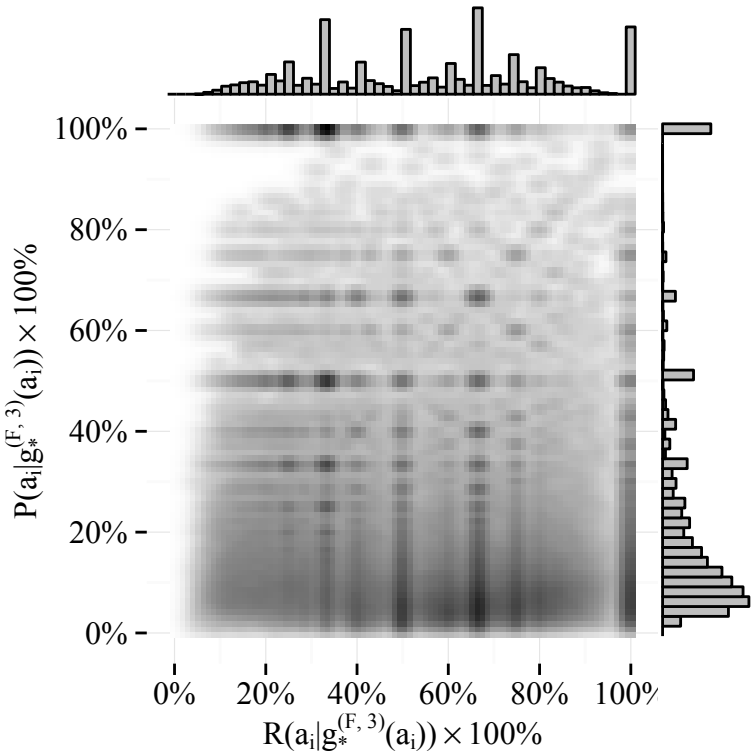
0.99 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 10$**.**



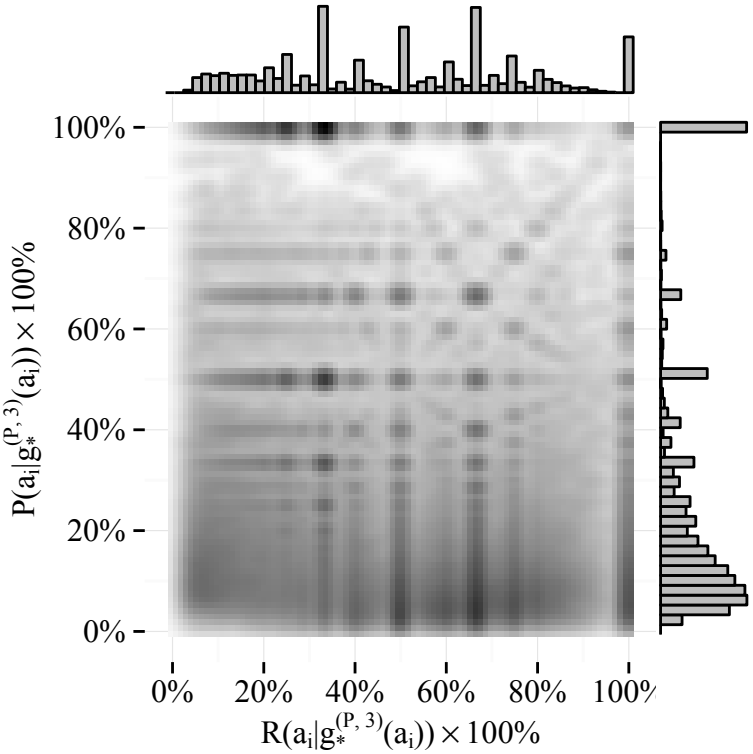
0.100 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 20$**.**



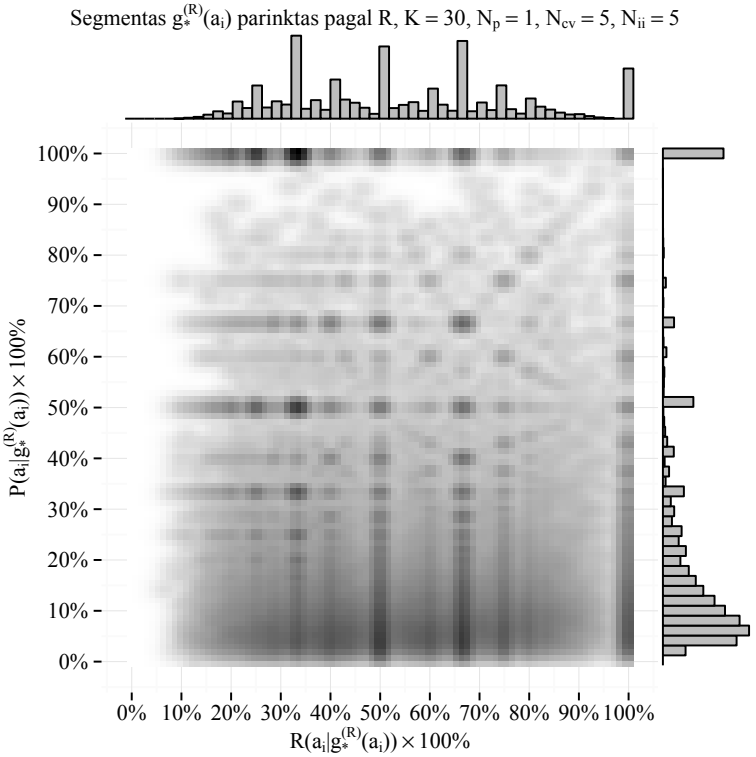
0.101 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 20$**.**



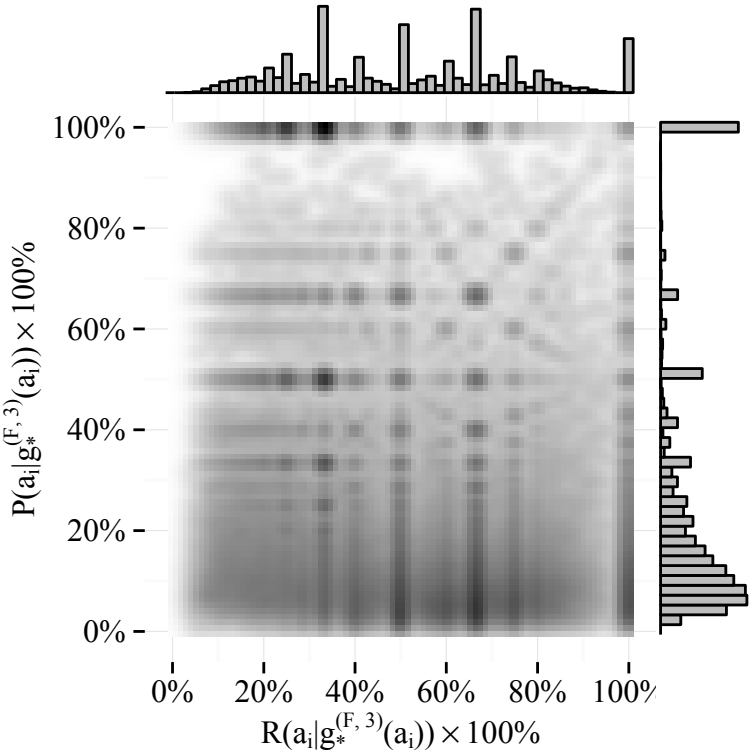
0.102 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 20$**.**



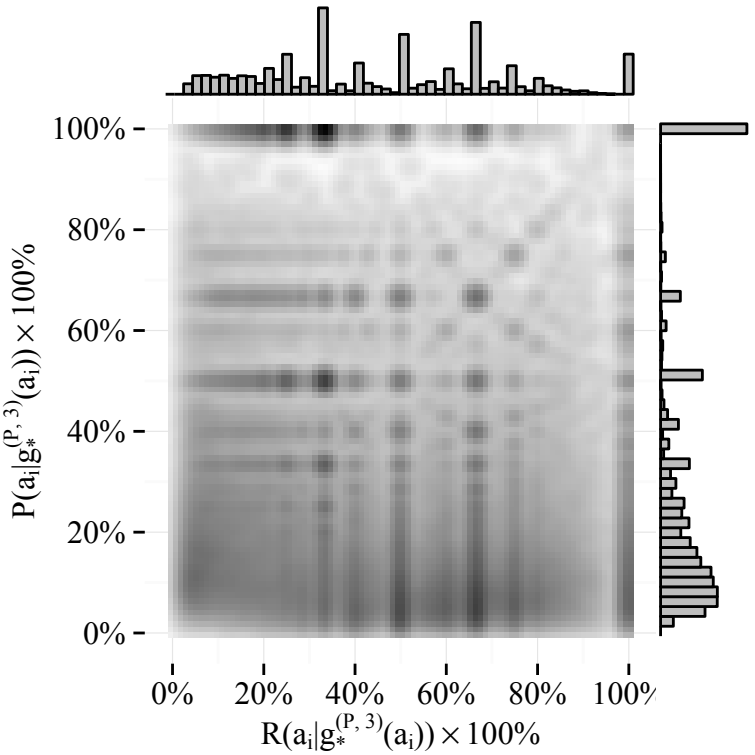
0.103 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 30$**.**



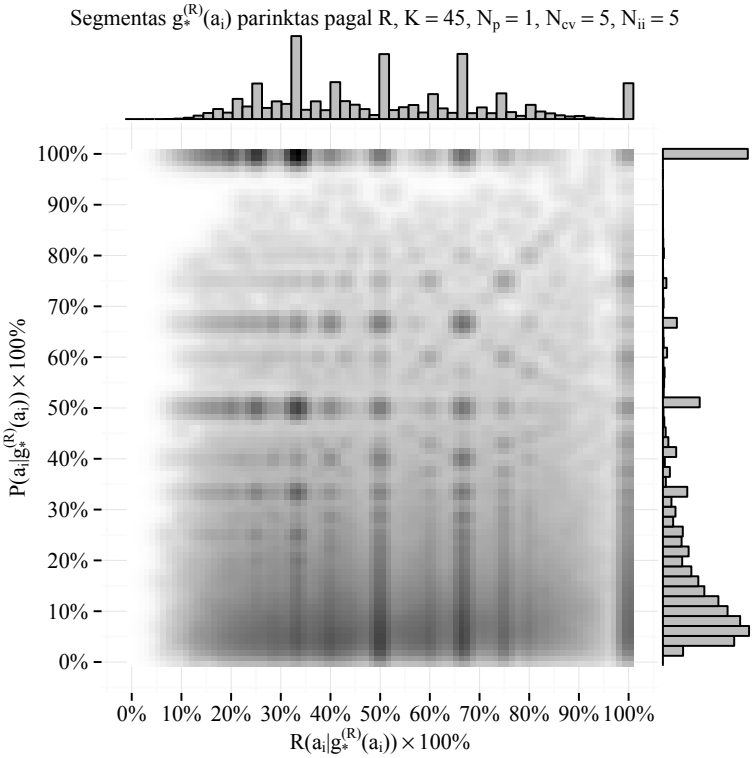
0.104 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 30$**.**



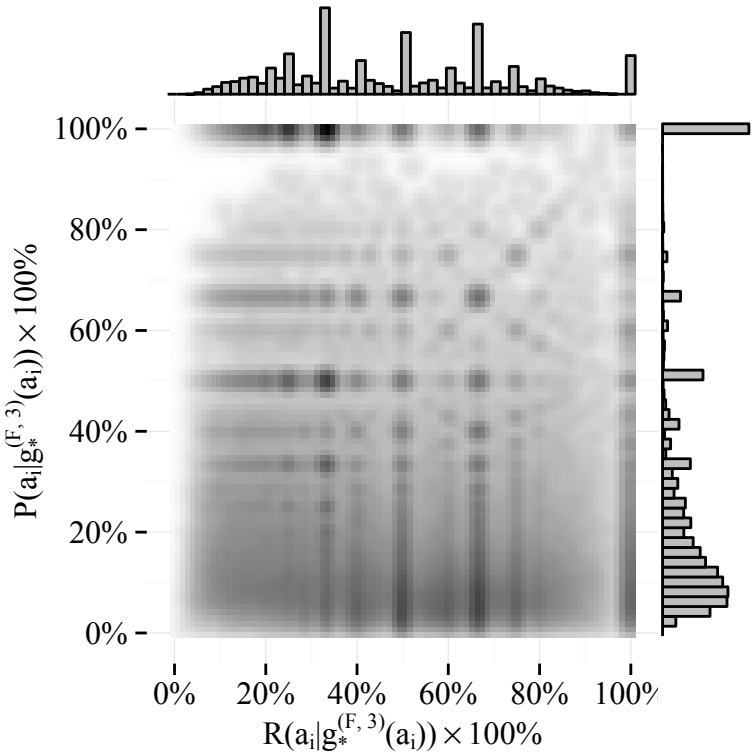
0.105 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 30$**.**



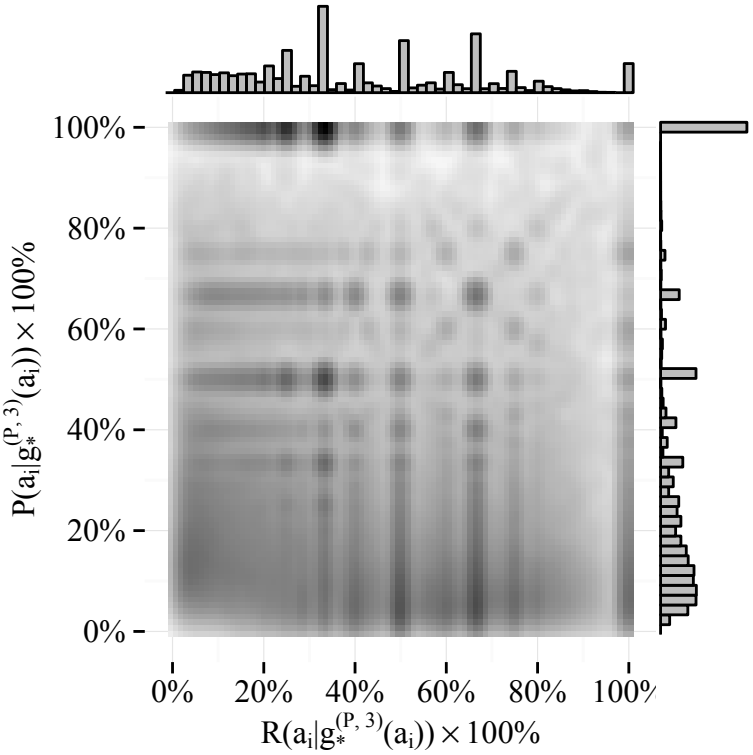
0.106 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 45$**.**



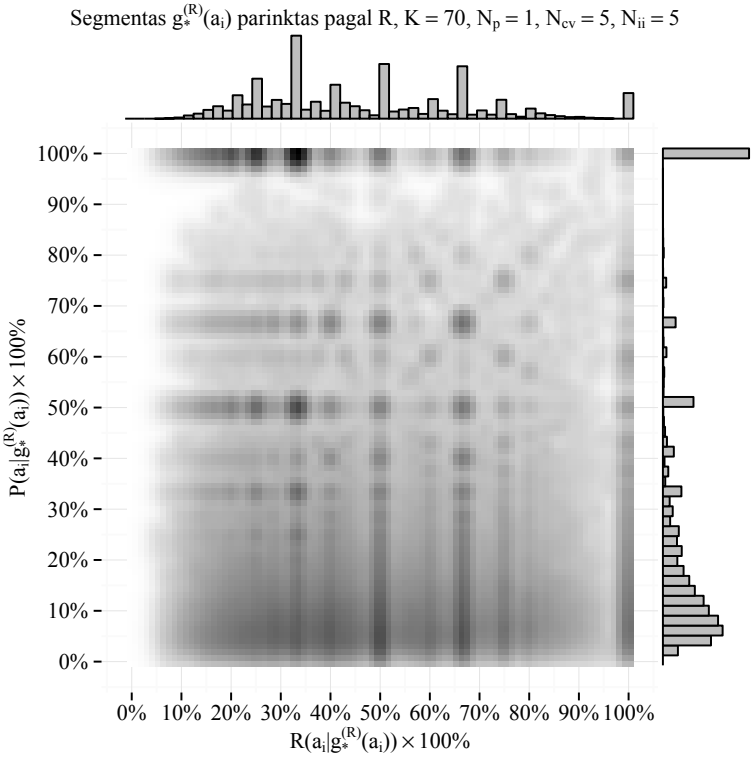
0.107 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 45$**.**



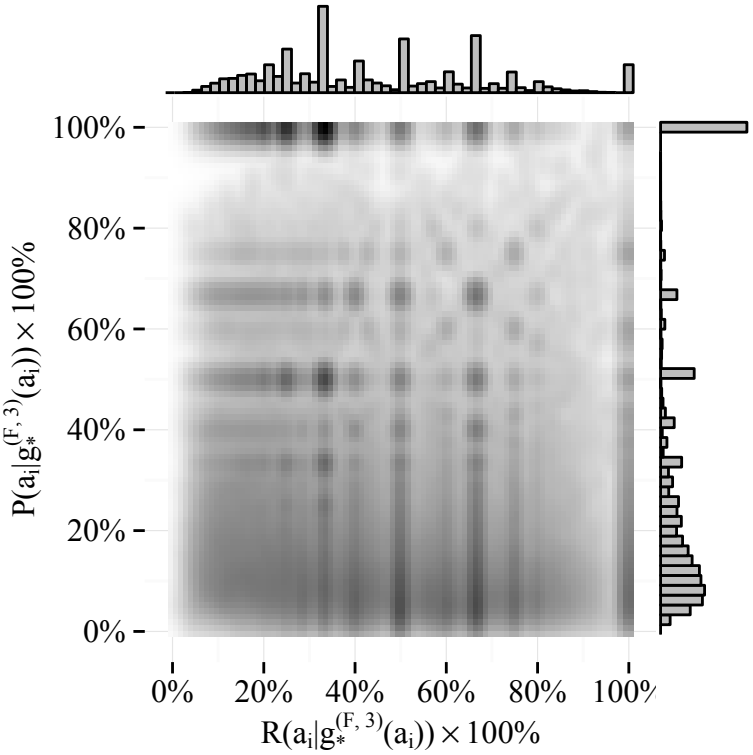
0.108 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 45$**.**



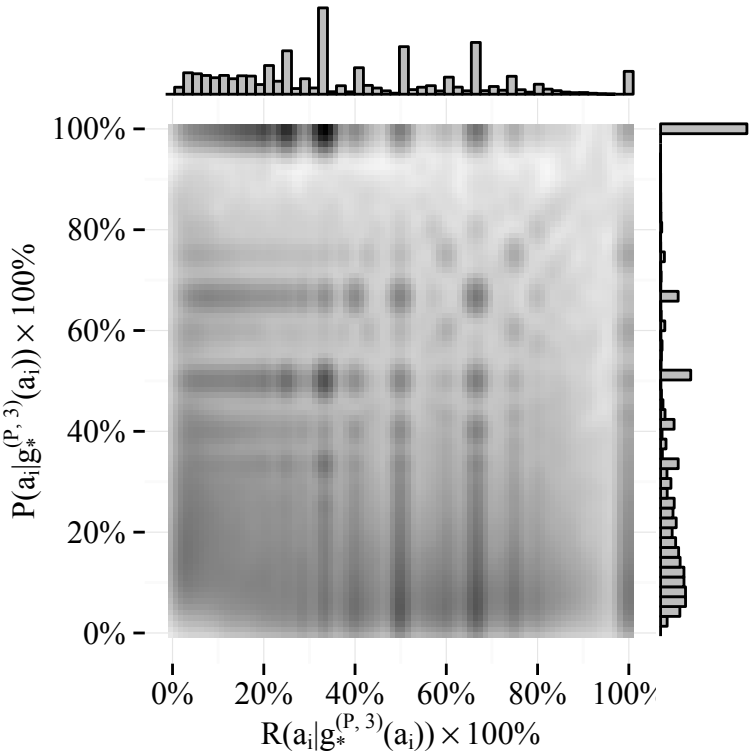
0.109 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 70$**.**



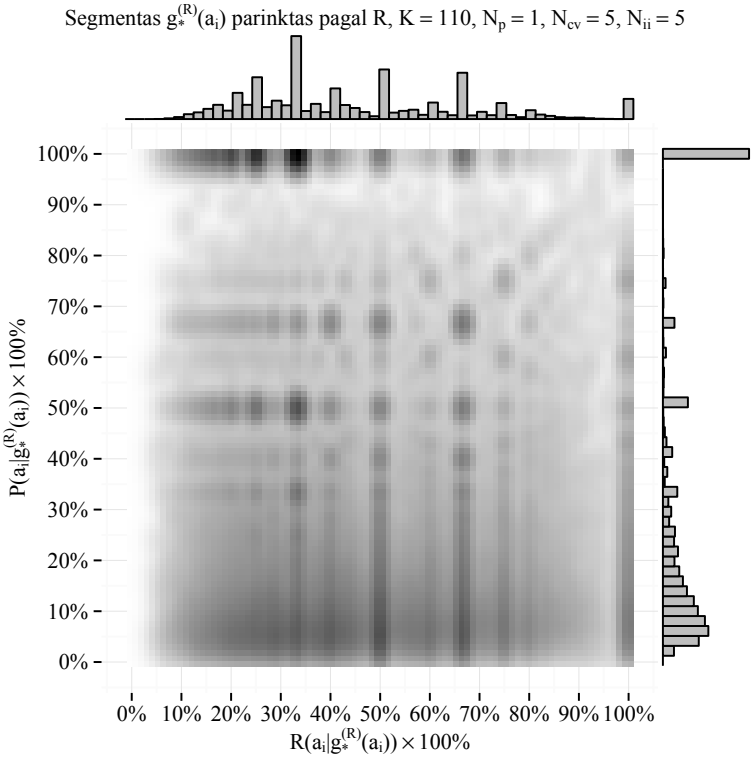
0.110 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 70$**.**



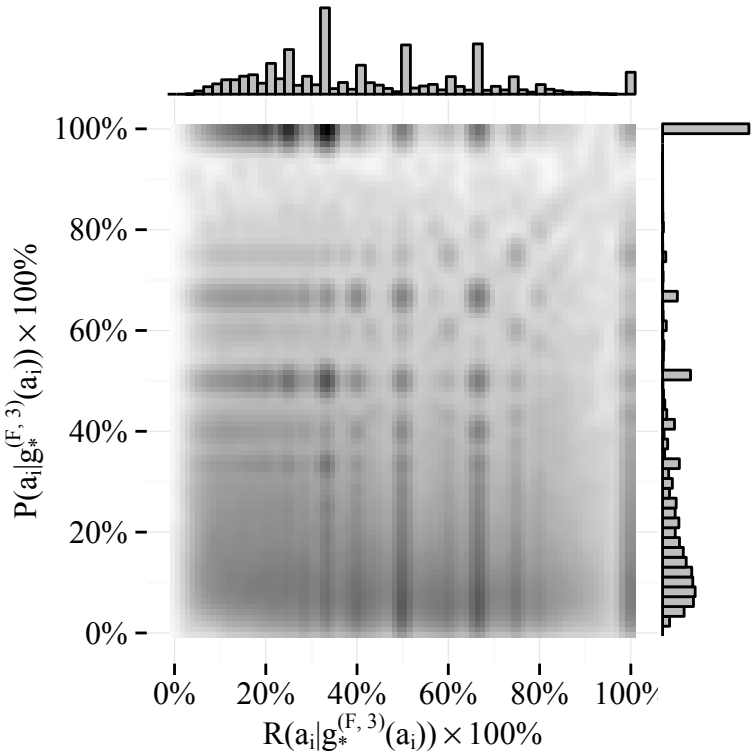
0.111 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 70$**.**



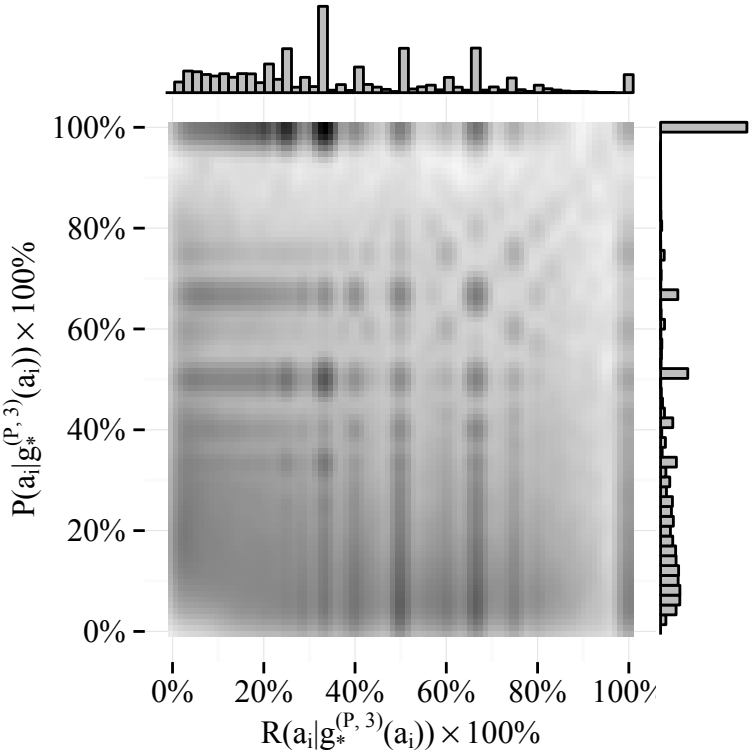
0.112 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 110$**.**



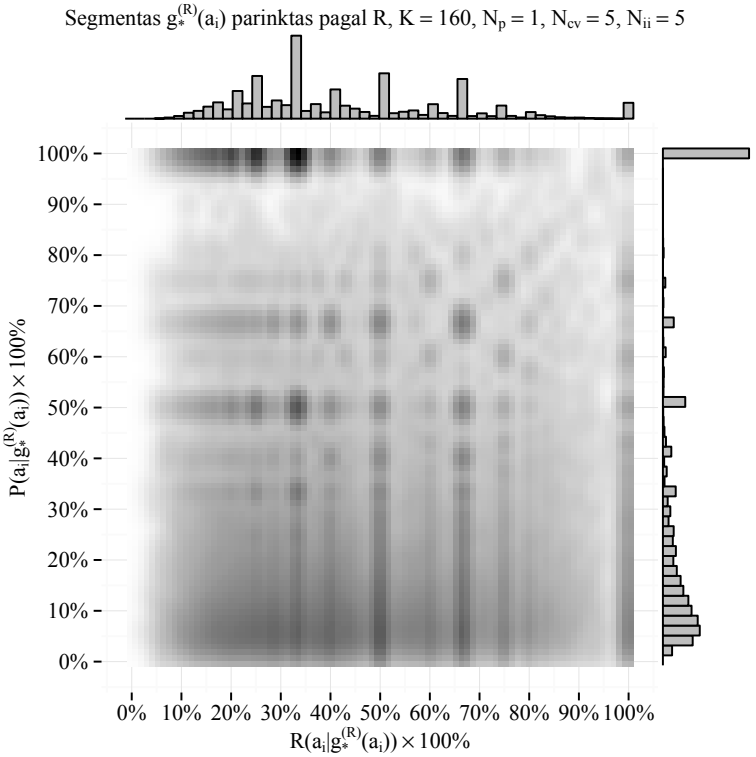
0.113 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 110$**.**



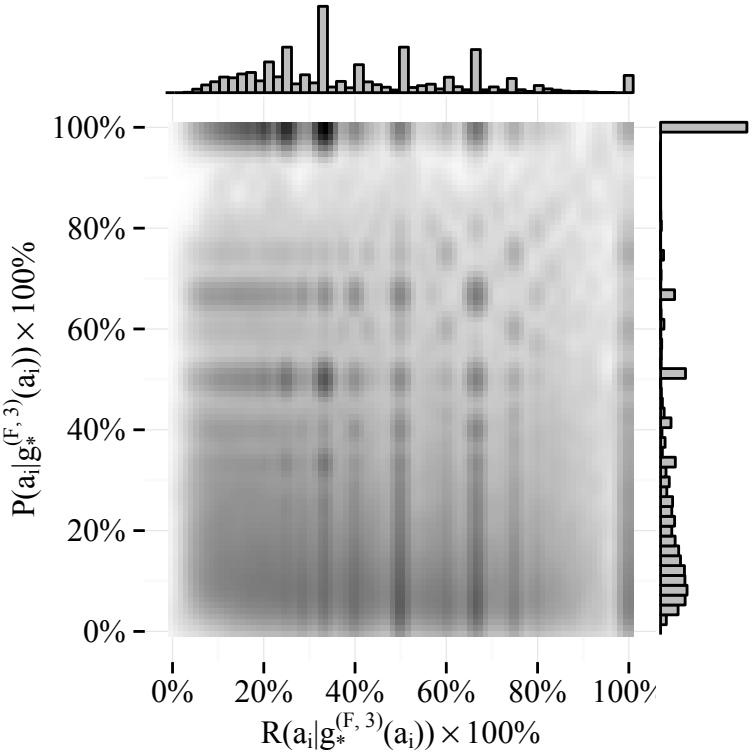
0.114 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 110$**.**



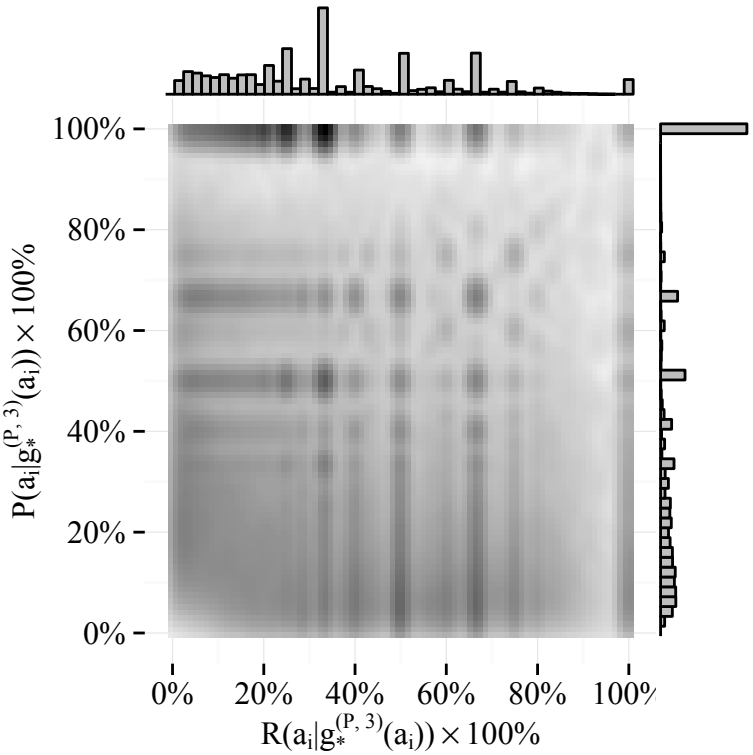
0.115 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 160$**.**



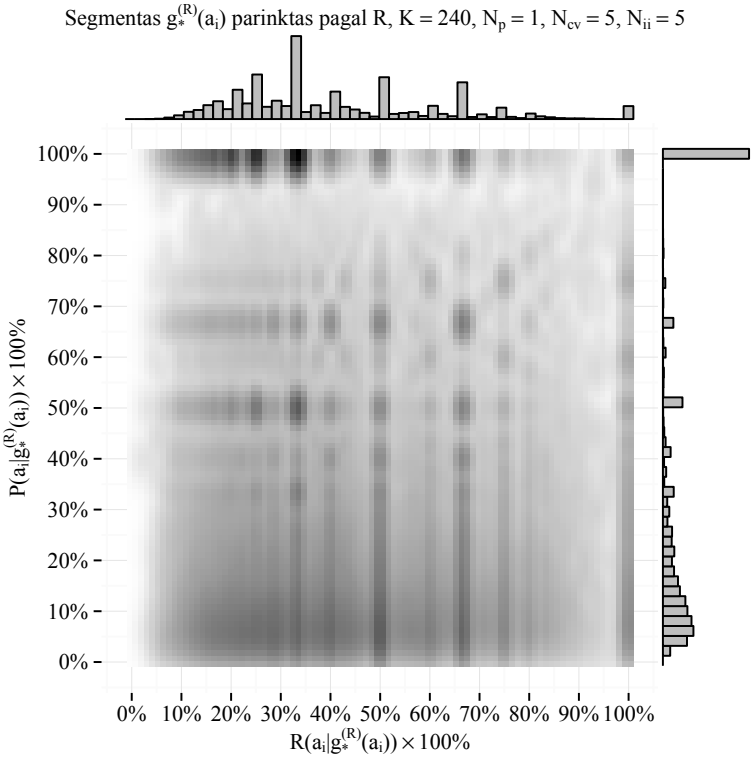
0.116 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 160$**.**



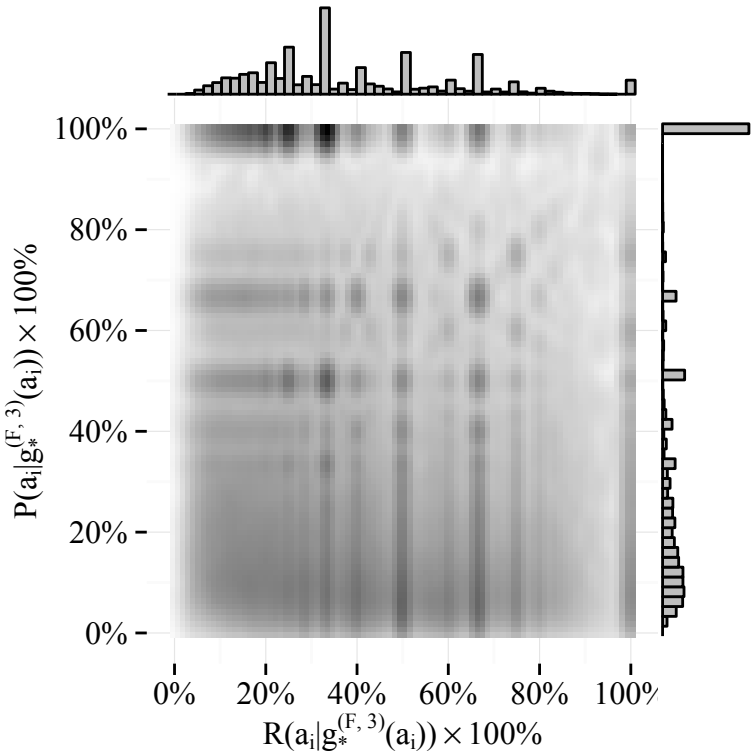
0.117 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 160$**.**



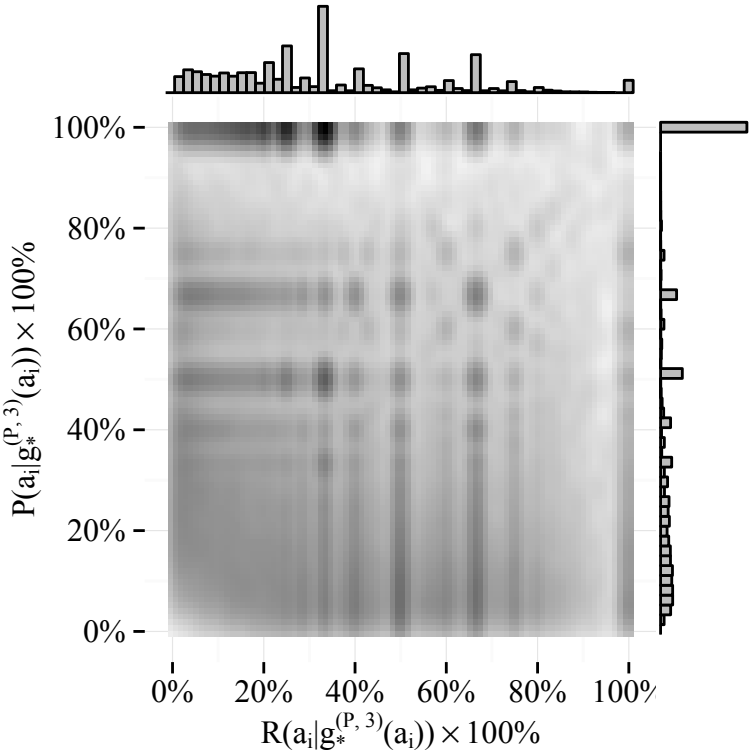
0.118 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 240$**.**



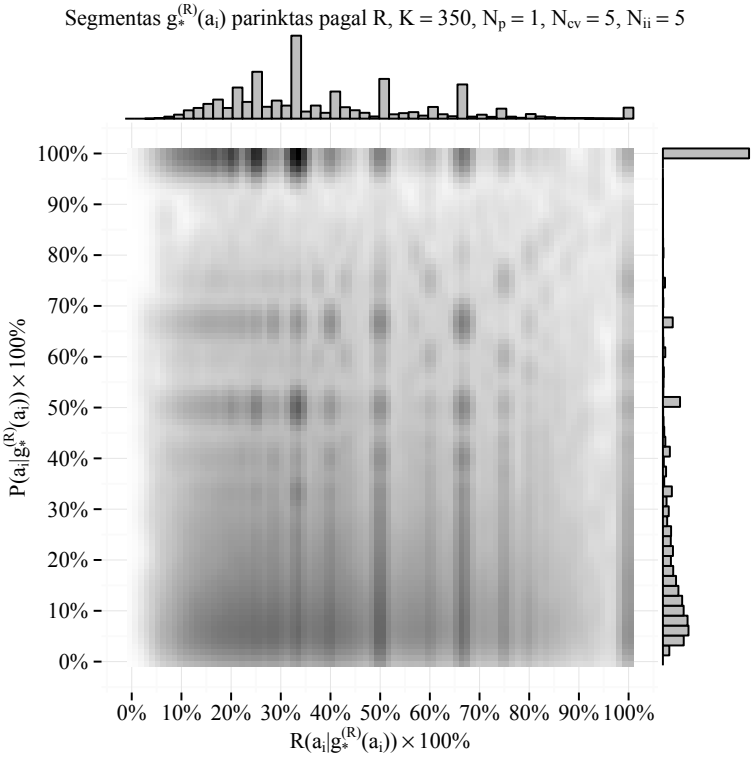
0.119 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 240$**.**



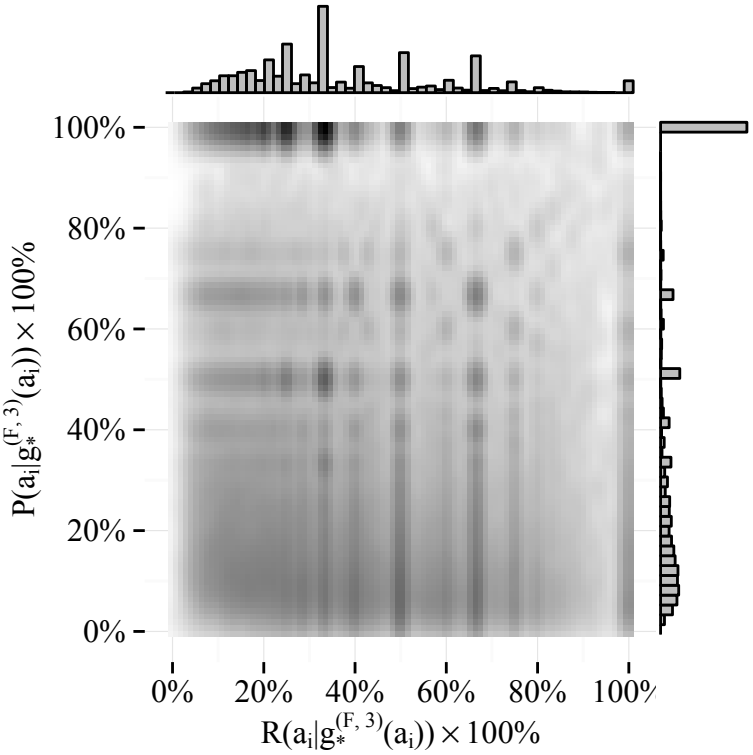
0.120 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 240$**.**



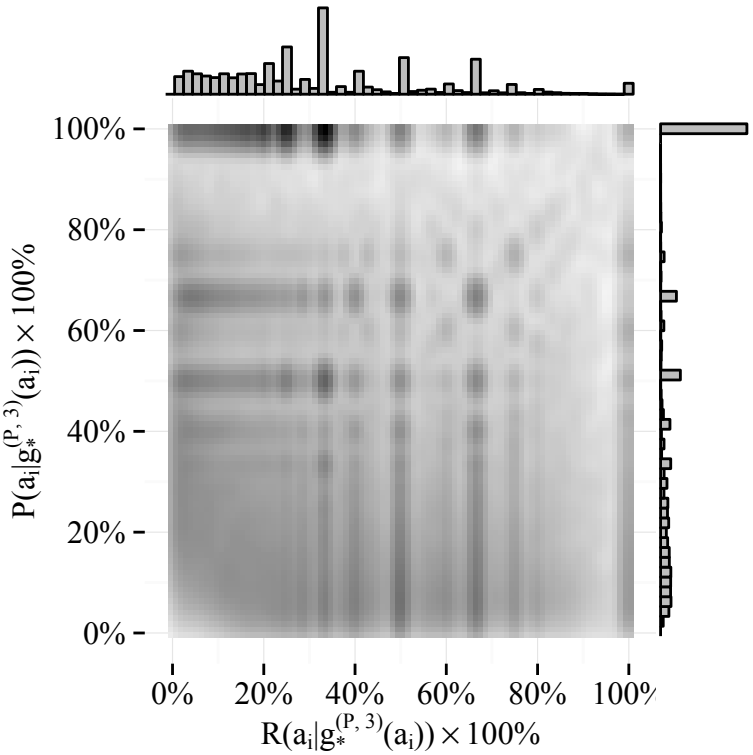
0.121 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 350$**.**



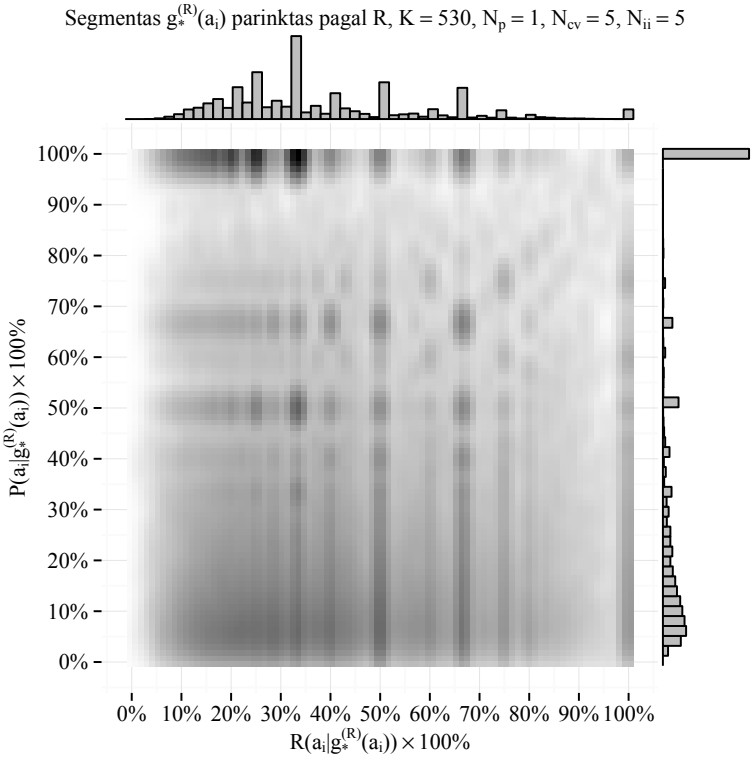
0.122 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 350$**.**



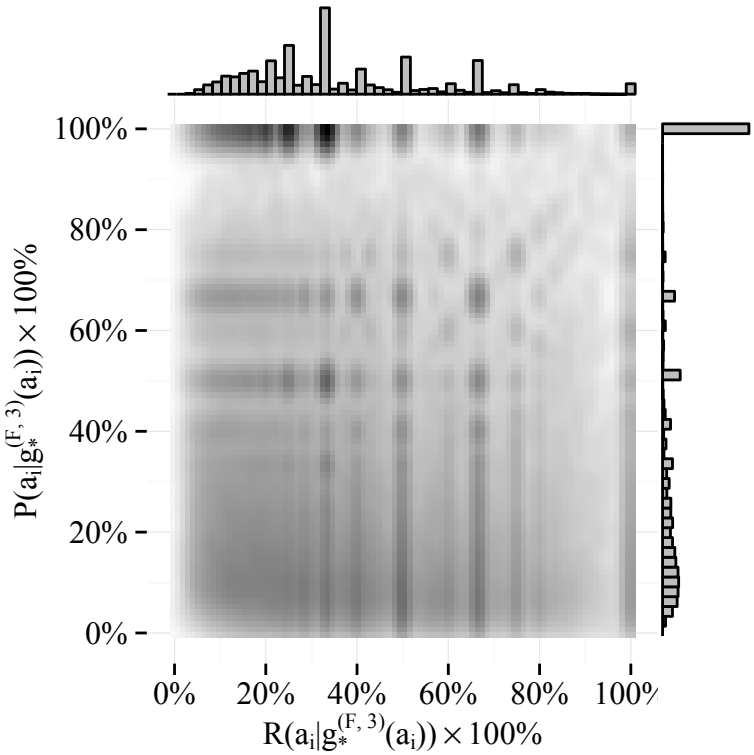
0.123 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 350$**.**



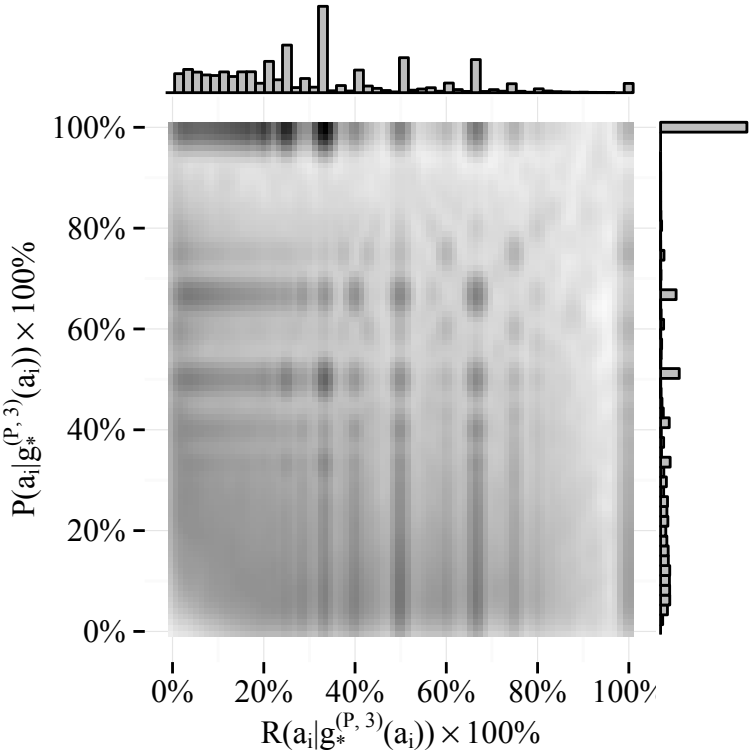
0.124 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 530$**.**



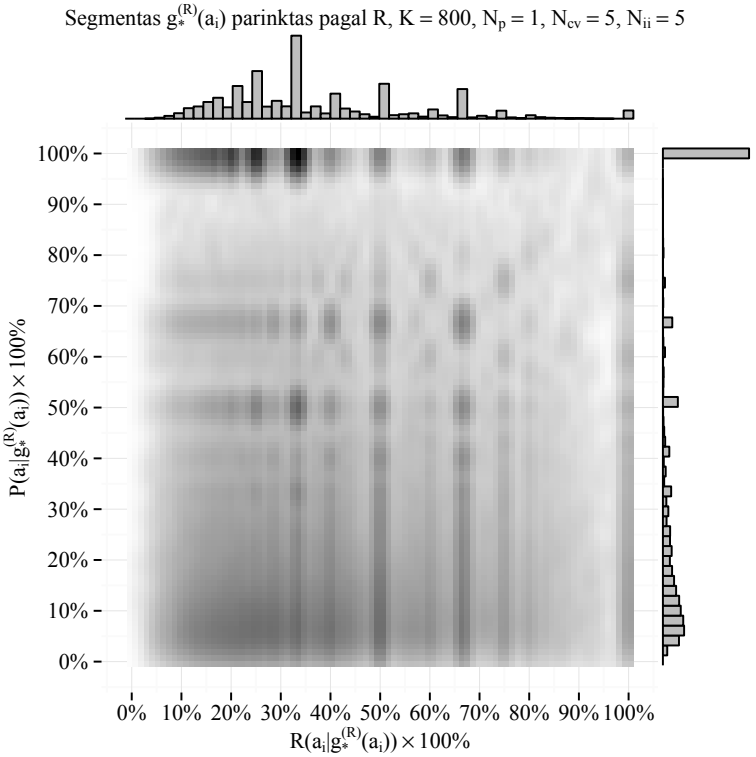
0.125 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 530$**.**



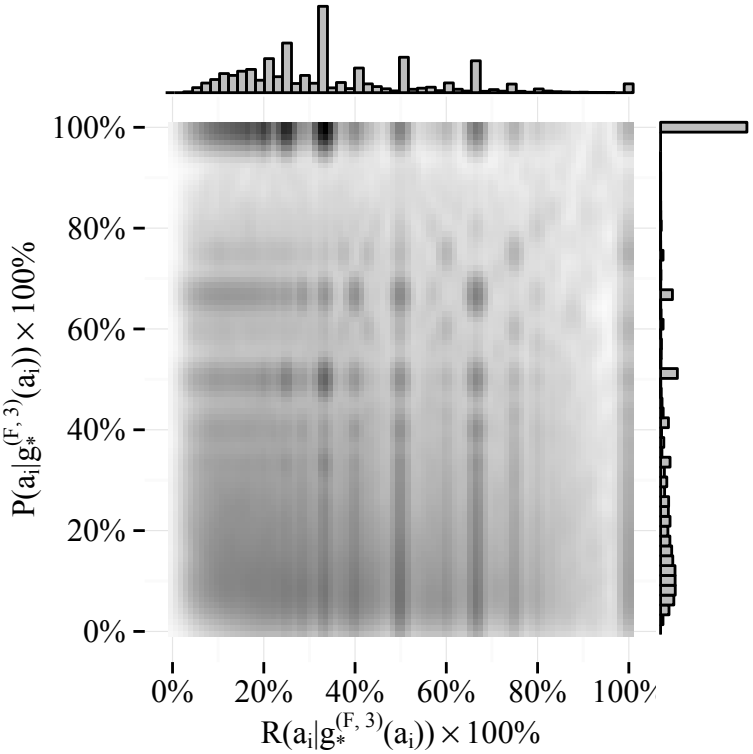
0.126 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 530$**.**



0.127 pav. Geriausio segmento parinkimo pagal tikslumą (P) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 800$**.**



0.128 pav. Geriausio segmento parinkimo pagal atkuriamumą (R) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 800$**.**



0.129 pav. Geriausio segmento parinkimo pagal F-matą (F) rezultatai, kai taikomas apribojimas pagal atkuriamumą (R), $K = 800$**.**

2 priedas. PROGRAMŲ TEKSTAI

sparkevaluation/src/main/scala/SegmentEvaluation.scala

```
1  import org.apache.spark.SparkConf
2  import org.apache.spark.SparkContext
3  import org.apache.spark.SparkContext._
4  import org.apache.spark.sql.functions._
5  import org.apache.spark.rdd.RDD
6  import org.apache.spark.sql.SQLContext
7  import org.apache.spark.sql.types._
8  import org.apache.spark.sql.DataFrame
9  import org.apache.spark.mllib.linalg.Vector
10 import java.nio.file.{Paths, Files}
11
12 case class CtrEvalRow(
13   click: Long, impression: Long, adId: Long,
14   userId: Long, bmUserId: Long)
15 case class DeltaCtrByAdRow(adId: Long, deltaCtr: Double)
16
17 case class CtrByAdRow(
18   adId: Long, ctr: Double,
19   nClick: Long, nImpression: Long, nUser: Long)
```

```
20
21  case class BidmachUserSegmentRow(bmUserId: Long, segmentId: Int)
22
23  case class MetricsByAdBySegmentRow(
24    adId: Long, segmentId: Int,
25    nUserAd: Long, nUserSeg:Long,
26    nClickAd: Long, nImpressionAd:Long, ctrAd: Double,
27    nClickSeg: Long, nImpressionSeg:Long, ctrSeg: Double,
28    deltaCtr: Double, precision: Double,
29    recall: Double, fMeasure: Double, clickEntropyComponent: Double)
30
31  object Main {
32
33    val projectDir = "/home/hm/math_master_ktu/magistro_baigiamasis_darbas/" +
34      "project/"
35    val dataDir = projectDir + "data/kdd/"
36    val histDataDir = projectDir + "preprocesskdd/r_plots/data_hist/"
37
38    val dirs = Array("ctr", "ctr_eval", "query", "keyword", "ctr_by_user",
39                     "query_counts",
40                     "n_query_by_user_wo_click", "n_click_n_user_by_ad")
41    val pathMap = dirs.zip(dirs.map(x => dataDir + x)).toMap
42
43    def main(args: Array[String]) {
44
45      val conf = new SparkConf()
46        .setAppName("SegmentationEvaluation")
47        .setMaster("local[4]")
48        .set("spark.driver.memory", "13g")
49      val sc = new SparkContext(conf)
50
51      val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)
52
53      import sqlContext.implicits._
54      import sqlContext.sql
55
56      val bidmachSegmentFile = args(0)
57      val dirByPrecision = args(1)
58      val dirByRecall = args(2)
59      val dirByFMeasure = args(3)
60      val dirAdClickEntropy = args(4)
61      val nSegment = args(5).toDouble
62      val dirMetricsByAdBySegment = args(6)
63      val dirNSegPerAd = args(7)
64
65      def saveMetrics(): Unit = {
66        val metricsByAdBySegment:DataFrame = if (
67            Files.exists(Paths.get(dirMetricsByAdBySegment))
68          ) {
69            loadMetricsByAdBySegment(dirMetricsByAdBySegment)
```

```
70          } else {
71            val ctr = loadCtrEval(dataDir + "ctr_eval.tsv")
72            val userSegment = loadBidmachUserSegment(bidmachSegmentFile)
73            val ctrByAdBySegment = getCtrByAdBySegment(ctr, userSegment)
74            val ctrByAd = loadCtrByAd(dataDir + "ctr_by_ad.tsv")
75            val metrics = getMetricsByAdBySegment(ctrByAdBySegment, ctrByAd)
76            saveDf(dirMetricsByAdBySegment, metrics)
77            loadMetricsByAdBySegment(dirMetricsByAdBySegment)
78          }
79
80        // fms
81        //, COUNT(DISTINCT segmentId) AS nSeg
82        // fma
83        //, AVG(nUserSeg) AS avgNUserSeg
84        // fmau
85        //, AVG(nUserSeg) AS avgNUserSeg
86        // fmc
87        //, AVG(nClickSeg) AS avgNClickSeg
88        // fmi
89        //, AVG(nImpressionSeg) AS avgNImpressionSeg
90        // fmck
91        //, AVG(nClickSeg) AS avgNClickSeg
92        // fmf
93        //, AVG(fMeasure) AS avgfMeasureSeg
94        // fmrsd
95        // AVG(recall) AS avgRecSeg
96        // STD(recall) AS stdRecSeg
97        metricsByAdBySegment.registerTempTable("metrics")
98        val nSegPerAd = sqlContext.sql("""
99              SELECT
100             adId
101             , COUNT(DISTINCT segmentId) AS nSeg
102             , AVG(nUserSeg) AS avgNUserSeg
103             , AVG(nClickSeg) AS avgNClickSeg
104             , AVG(nImpressionSeg) AS avgNImpressionSeg
105             , AVG(fMeasure) AS avgFMeasureSeg
106             , AVG(recall) AS avgRecSeg
107             , STD(recall) AS stdRecSeg
108             FROM metrics
109             GROUP BY adId
110         """).cache()
111       saveDf(dirNSegPerAd, nSegPerAd)
112       nSegPerAd.registerTempTable("n_seg_per_ad")
113
114       // fmu
115       //WHERE metrics.nUserSeg / metrics.nUserAd >= 1 / {nSegment}
116       // fms
117       //WHERE metrics.nUserSeg / metrics.nUserAd >= 1 / n_seg_per_ad.nSeg
118       // fma
119       //WHERE metrics.nUserSeg >= n_seg_per_ad.avgNUserSeg
```

```
120       // fmc
121       //WHERE metrics.nClickSeg  >= n_seg_per_ad.avgNClickSeg
122       // fmi
123       //WHERE metrics.nImpressionSeg  >= n_seg_per_ad.avgNImpressionSeg
124       // fmau
125       //WHERE metrics.nUserSeg / metrics.nUserAd >= n_seg_per_ad.avgNUserSeg
126       // fmck
127       // WHERE metrics.nClickSeg >= metrics.nClickAd / n_seg_per_ad.nSeg
128       // fmik
129       // WHERE metrics.nImpressionSeg >= (
130       //    metrics.nImpressionAd / n_seg_per_ad.nSeg)
131       // fmf
132       // WHERE metrics.nfMeasureSeg >= metrics.avgFMeasureSeg
133       // fmrsd
134       //WHERE metrics.nClickSeg  >= (
135       //    n_seg_per_ad.avgNClickSeg - n_seg_per_ad.stdNClickSeg)
136       val filteredMetrics = sqlContext.sql(s"""
137     SELECT metrics.*
138     FROM metrics
139     INNER JOIN n_seg_per_ad ON n_seg_per_ad.adId = metrics.adId
140     WHERE metrics.recall  >= n_seg_per_ad.avgRecSeg - n_seg_per_ad.stdRecSeg
141         """).cache()
142
143       saveDf(dirByPrecision, getMetricsByAdByBestSegmentFromMetric(
144                filteredMetrics, "precision"))
145       saveDf(dirByRecall, getMetricsByAdByBestSegmentFromMetric(
146                filteredMetrics, "recall"))
147       saveDf(dirByFMeasure, getMetricsByAdByBestSegmentFromMetric(
148                filteredMetrics, "fMeasure"))
149       saveDf(dirAdClickEntropy, getClickEntropyByAd(
150                loadMetricsByAdBySegment(dirMetricsByAdBySegment)))
151     }
152
153     def loadCtrByAd(path: String): DataFrame = {
154       sc.textFile(path).
155         map(x => x.split("\\t")).
156         map(x => CtrByAdRow(
157               x(0).toLong, x(1).toDouble,
158               x(2).toLong, x(3).toLong, x(4).toLong)).
159         toDF()
160     }
161
162     def loadCtrEval(
163       path: String = pathMap("ctr_eval")): DataFrame = {
164       sc.textFile(path).
165         map(x => x.split("\\t").map(_.toLong)).
166         map(x => CtrEvalRow(x(0), x(1), x(2), x(3), x(4))).
167         toDF()
168     }
169
```

```
170     def loadMetricsByAdBySegment(path: String): DataFrame = {
171       sc.textFile(path).
172         map(x => x.split("\\t")).
173         map(x => MetricsByAdBySegmentRow(
174               x(0).toLong, x(1).toInt,
175               x(2).toLong, x(3).toLong,
176               x(4).toLong, x(5).toLong, x(6).toDouble,
177               x(7).toLong, x(8).toLong, x(9).toDouble,
178               x(10).toDouble, x(11).toDouble,
179               x(12).toDouble, if (x(13) == "null") 0.0 else x(13).toDouble,
180               x(14).toDouble)).
181         toDF()
182     }
183
184     def getCtrByAdBySegment(
185       ctr: DataFrame,
186       userSegment: DataFrame): DataFrame =
187     {
188       ctr.registerTempTable("ctr")
189       userSegment.registerTempTable("user_segment")
190       val ctrWithSegment = sqlContext.sql("""
191             SELECT
192             ctr.adId
193             , ctr.click
194             , ctr.impression
195             , user_segment.segmentId
196             , ctr.bmUserId
197             FROM ctr
198             INNER JOIN user_segment ON user_segment.bmUserId = ctr.bmUserId
199         """)
200       ctrWithSegment.registerTempTable("ctr_w_seg")
201
202       val ctrByAdBySegment =  sqlContext.sql("""
203             SELECT
204             adId
205             , segmentId
206             , SUM(click) AS nClick
207             , SUM(impression) AS nImpression
208             , SUM(click) / SUM(impression) AS ctr
209             , COUNT(DISTINCT bmUserId) AS nUser
210             FROM ctr_w_seg
211             GROUP BY adId, segmentId
212         """)
213       ctrByAdBySegment
214     }
215
216     def getMetricsByAdBySegment(
217       ctrByAdBySegment: DataFrame,
218       ctrByAd: DataFrame
219     ): DataFrame = {
```

```
220       ctrByAdBySegment.registerTempTable("ctr_by_ad_by_seg")
221       ctrByAd.registerTempTable("ctr_by_ad")
222       val metricsByAdBySegmentWithAdCtr = sqlContext.sql("""
223             SELECT
224             ctr_by_ad_by_seg.adId
225             , ctr_by_ad_by_seg.segmentId
226             , ctr_by_ad.nUser AS nUserAd
227             , ctr_by_ad_by_seg.nUser AS nUserSeg
228             , ctr_by_ad.nClick AS nClickAd
229             , ctr_by_ad.nImpression AS nImpressionAd
230             , ctr_by_ad.ctr AS ctrAd
231             , ctr_by_ad_by_seg.nClick AS nClickSeg
232             , ctr_by_ad_by_seg.nImpression AS nImpressionSeg
233             , ctr_by_ad_by_seg.ctr AS ctrSeg
234             , (ctr_by_ad_by_seg.ctr - ctr_by_ad.ctr) / ctr_by_ad.ctr
235             AS deltaCtr
236             , ctr_by_ad_by_seg.ctr
237             AS precision
238             , ctr_by_ad_by_seg.nClick / ctr_by_ad.nClick
239             AS recall
240             , (2 * (ctr_by_ad_by_seg.ctr) *
241                 (ctr_by_ad_by_seg.nClick / ctr_by_ad.nClick)) /
242                 (ctr_by_ad_by_seg.ctr +
243                 (ctr_by_ad_by_seg.nClick / ctr_by_ad.nClick))
244             AS fMeasure
245             , (ctr_by_ad_by_seg.nClick / ctr_by_ad.nImpression)
246             AS clickEntropyComponent
247             FROM ctr_by_ad_by_seg
248             INNER JOIN ctr_by_ad ON ctr_by_ad.adId = ctr_by_ad_by_seg.adId
249         """)
250       metricsByAdBySegmentWithAdCtr
251     }
252
253     def getClickEntropyByAd(
254       metricsByAdBySegment: DataFrame
255     ): DataFrame = {
256       metricsByAdBySegment.registerTempTable("metrics_by_ad_by_segment")
257       val clickEntropyByAd = sqlContext.sql("""
258             SELECT
259             adId
260             , -SUM(clickEntropyComponent * LOG(2, clickEntropyComponent))
261             AS adClickEntropy
262             FROM metrics_by_ad_by_segment
263             GROUP BY adId
264         """)
265       clickEntropyByAd
266     }
267
268     def getMetricsByAdByBestSegmentFromMetric(
269       metricsByAdBySegment: DataFrame,
```

```
270       metric: String
271     ): DataFrame = {
272
273       val otherMetric:String = metric match {
274           case "precision" => "recall"
275           case _ => "precision"
276         }
277
278       def compareSegments(
279         r1: MetricsByAdBySegmentRow,
280         r2: MetricsByAdBySegmentRow
281       ): MetricsByAdBySegmentRow = {
282         val r1M1 = getMetric(r1, metric)
283         val r2M1 = getMetric(r2, metric)
284         val r1M2 = getMetric(r1, otherMetric)
285         val r2M2 = getMetric(r2, otherMetric)
286
287         val r = (r1M1, r2M1) match {
288             case (r1M1, r2M1) if (r1M1 > r2M1) => r1
289             case (r1M1, r2M1) if (r1M1 < r2M1) => r2
290             case (r1M1, r2M1) if (r1M1 == r2M1) => (r1M2, r2M2) match {
291               case (r1M2, r2M2) if (r1M2 >= r2M2) => r1
292               case (r1M2, r2M2) if (r1M2 < r2M2) => r2
293             }
294           }
295         r
296       }
297
298       val metricsByAdByBestSegmentFromMetric: RDD[MetricsByAdBySegmentRow] =
299         metricsByAdBySegment.rdd.map(
300           x => (x.getLong(0),
301                 MetricsByAdBySegmentRow(
302                   x.getLong(0), x.getInt(1),
303                   x.getLong(2), x.getLong(3),
304                   x.getLong(4), x.getLong(5), x.getDouble(6),
305                   x.getLong(7), x.getLong(8), x.getDouble(9),
306                   x.getDouble(10), x.getDouble(11),
307                   x.getDouble(12), x(13).asInstanceOf[Double], x.getDouble(14))
308           )).
309           reduceByKey(compareSegments).map(x => x._2)
310       metricsByAdByBestSegmentFromMetric.toDF
311     }
312
313     def getMetric(r: MetricsByAdBySegmentRow, m:String):Double = m match {
314         case "precision" => r.precision
315         case "recall" => r.recall
316         case "fMeasure" => r.fMeasure
317       }
318
319
```

```
320     def loadBidmachUserSegment(path: String): DataFrame = {
321       sc.textFile(path).
322         map(x => x.split("\\t")).
323         map(x => BidmachUserSegmentRow(x(0).toLong, x(1).toInt)).
324         toDF()
325     }
326 
327     def saveDf(path: String, df: DataFrame, delim: String = "\t",
328                nPartitions: Int = 0): Unit = {
329       val rdd = if (nPartitions > 0) df.rdd.repartition(nPartitions) else df.rdd
330       rdd.map(_.mkString(delim)).saveAsTextFile(path)
331     }
332 
333     val t0 = System.nanoTime()
334     saveMetrics()
335     val t1 = System.nanoTime()
336     println("\n\n#####################################")
337     println(s"# elapsed: ${(t1 - t0) / 1e9} s")
338     println("# SPARK EVALUATION DONE!")
339     println("#####################################\n")
340 
341     sc.stop()
342     System.exit(0)
343   }
344 
345 }
```

btusersegmentation/bidmach_scripts/lda_crossvalidation.ssc

```
1  val hasExppsi = true
2  var weps = 1e-10f
3  var alpha = 0.001f
4  
5  val nUsersTotal = 5647787
6  val nUsersPerFold = floor(nUsersTotal / nFolds)
7  val foldIndicesMat = (((0 until nFolds) * nUsersPerFold) on
8                        (((1 until nFolds) * nUsersPerFold) \ nUsersTotal))
9  val idFromTest = foldIndicesMat(0,iFold).toInt
10 val idUntilTest = (foldIndicesMat(1,iFold)).toInt
11 
12 class xopts extends Learner.Options with
13     MatDS.Opts with
14     LDA.Opts with
15     IncNorm.Opts with
16     Perplexity.Opts with
17     CosineSim.Opts with
18     Top.Opts with
19     L1Regularizer.Opts with
```

```
20      L2Regularizer.Opts
21
22  val opts = new xopts
23
24  // (1) feature type, 0=binary, 1=linear
25  opts.featType = 1;
26  opts.putBack = 0; // -1 vs 0 ?
27  opts.batchSize = miniBatchSize
28  opts.npasses = nPasses
29  opts.dim = nSegments
30  opts.uiter = nIter
31  opts.perpnmats = 2
32  opts.topnmats = 2
33  opts.cosnmats = 2
34  opts.r1nmats = 2
35  opts.r2nmats = 2
36
37  val ds = if (iFold == 0) {
38      new MatDS(
39        Array(
40          loadSMat(dataDir + "usertokens.smat.lz4")(
41            ?,idUntilTest until nUsersTotal)), opts)
42  } else if (iFold == (nFolds - 1)) {
43      new MatDS(
44        Array(
45          loadSMat(dataDir + "usertokens.smat.lz4")(
46            ?,0 until idFromTest)), opts)
47  } else {
48      new MatDS(
49        Array(
50          loadSMat(dataDir + "usertokens.smat.lz4")(
51            ?,(0 until idFromTest) \ (idUntilTest until nUsersTotal))), opts)
52  }
53  opts.autoReset = false
54
55  val mixinsAllowedNames = Array("Perp", "Top", "Cos", "L1", "L2")
56  def matchMixin(mx:String):BIDMach.mixins.Mixin = mx match {
57    case "Perp" => new Perplexity(opts)
58    case "Top" => new Top(opts)
59    case "Cos" => new CosineSim(opts)
60    case "L1" => new L1Regularizer(opts)
61    case "L2" => new L2Regularizer(opts)
62    }
63
64  val mixinsObjs = mixinsNames.
65    filter(x => mixinsAllowedNames.contains(x)).
66    map(x => matchMixin(x))
67  val mixins = if (mixinsObjs.length > 0) mixinsObjs else null
68
69  val nn = new Learner(                // make a learner instance
```

```
70      ds,                                 // datasource
71      new LDA(opts),                      // the model (a LDA model)
72      mixins,                             // list of mixins or regularizers
73      new IncNorm(opts),                  // the optimization class to use
74      opts)                               // pass the options to the learner as well
75  nn.train
76
77  val meanEnt = mean(-(nn.modelmat dotr ln(nn.modelmat)))
78  saveFMat(meanEntropyFilename, FMat(meanEnt))
79
80  saveFMat(logLikelihoodFilename, FMat(nn.results))
81
82  def predictVB(wordsPerSegment:Mat, queryTokens:Mat, nInter:Int=5):Mat = {
83    var segmentsPerUser = ones(wordsPerSegment.nrows, queryTokens.ncols).set(1f)
84    var iter = 0;
85    while (iter < nInter) {
86      iter += 1
87      val preds = DDS(wordsPerSegment, segmentsPerUser, queryTokens)
88      val dc = queryTokens.contents
89      val pc = preds.contents
90      max(weps, pc, pc)
91      pc ~ dc / pc
92      val newSegmentsPerUser = segmentsPerUser *@ (wordsPerSegments * preds) +
93        alpha
94      if (hasExppsi) exppsi(newSegmentsPerUser, newSegmentsPerUser)
95      segmentsPerUser <-- newSegmentsPerUser
96    }
97    segmentsPerUser
98  }
99
100
101 val datamat = loadSMat(
102     dataDir + "usertokens.smat.lz4")(?,idFromTest until idUntilTest)
103
104 val nResParts = 50
105 val nUsersTest:Int = size(datamat)._2
106 val nPerPart:Int = (nUsersTest / nResParts).toInt
107 var iResPart:Int = 0
108
109 while (iResPart < nResParts) {
110   var iFrom:Int = iResPart * nPerPart
111   var iUntil:Int = (if (iResPart == (nResParts - 1)) {nUsersTest}
112                     else {iFrom + nPerPart})
113
114   var idFrom:IMat = iFrom + idFromTest
115   var idUntil:IMat = iUntil + idFromTest
116
117   var datamatIndices:Range = (iFrom until iUntil)
118   var userIds:IMat = datamatIndices + idFromTest
119   println(s"size userIndices: ${size(datamatIndices)._2}")
```

```
120   println(s"iFrom: ${iFrom}, idFrom: ${idFrom}")
121   println(s"iUntil: ${iUntil}, idUntil: ${idUntil}")
122   val userSegments = predictVB(nn.modelmat, datamat(?,datamatIndices), nIter)
123   val segmentI = sortdown2(FMat(userSegments))._2
124
125   val filename = f"${jobName}_user_segment_" +
126     f"cv${iFold + 1}%02dof${nFolds}%02d_" +
127     f"part${iResPart + 1}%03dof${nResParts}%03d.txt"
128
129   saveIMat(dataSaveDir + "user_segment_parts/" + filename,
130            (userIds on segmentI(0, ?)).t)
131   iResPart += 1
132
133   println("Wrote datamatIds/userIds from " +
134            s"${iFrom}/${idFrom} until ${iUntil}/${idUntil}\n")
135 }
136
137 System.exit(0)
```

for-use-in-spark-shell.scala

```
1  import org.apache.spark.sql.DataFrame
2  import org.apache.spark.rdd.RDD
3  import org.apache.spark.mllib.linalg.Vector
4  import java.io.FileWriter
5
6  val isSample = false
7  val projectDir = "/home/hm/math_master_ktu/magistro_baigiamasis_darbas/project/"
8  val dataDir = projectDir + "data/kdd/"
9  val histDataDir = projectDir + "preprocesskdd/r_plots/data_hist/"
10
11 val dirs = Array("ctr", "ctr_eval", "query", "keyword", "ctr_by_user",
12                  "query_counts",
13                  "n_query_by_user_wo_click", "n_click_n_user_by_ad")
14 val paths = dirs.zip(dirs.map(x => dataDir + x)).toMap
15 val samplePaths = paths.map(x => (x._1, x._2 + "_sample"))
16 val pathMap = if (isSample) samplePaths else paths
17 val samplePct = 0.001
18
19 case class QueryRow(queryId: Long, tokens: String)
20
21 case class KeywordRow(keywordId: Long, tokens: String)
22
23 case class CtrByUserRow(
24   userId: Long, nClick: Long, nImpression: Long, userCtr: Double)
25
26 case class QueryCountRow(queryId: Long, nQuery: Long)
27
```

```
28  case class UniqueQueryCountByUserRow(userId: Long, nUniqueQuery: Long)
29
30  case class ClickCountByUserRow(userId: Long, nClick: Long)
31
32  case class ClickCountUserCountByAd(adId: Long, nClick: Long, nUser: Long)
33
34
35  def splitCsvRow(s: String): Array[String] = {
36    s.split(",")
37  }
38
39  val ds = if (iFold == 0) {
40      new MatDS(Array(loadSMat(dataDir + "usertokens.smat.lz4")(
41                          ?,iToTest + 1 until nUsersTotal)), opts)
42  } else if (iFold == (nFolds - 1)) {
43      new MatDS(Array(loadSMat(dataDir + "usertokens.smat.lz4")(
44                          ?,(0 until (iFromTest - 1)) \
45                            ((iToTest + 1) until nUsersTotal))), opts)
46  } else {
47      new MatDS(Array(loadSMat(dataDir + "usertokens.smat.lz4")(
48                          ?,iToTest + 1 until nUsersTotal)), opts)
49  }
50
51  case class CtrRow(
52    click: Long, impression: Long, adId: Long, queryId: Long,
53    keywordId: Long, userId: Long)
54
55  def loadCtr(path: String = pathMap("ctr")): DataFrame = {
56    sc.textFile(path).
57      map(x => splitCsvRow(x).map(_.toLong)).
58      map(x => CtrRow(x(0), x(1), x(2), x(3), x(4), x(5))).
59      toDF()
60  }
61
62  case class CtrEvalRow(click: Long, impression: Long, adId: Long,
63                        userId: Long, bmUserId: Long)
64
65  def loadCtrEval(path: String = pathMap("ctr_eval")): DataFrame = {
66    sc.textFile(path).
67      map(x => x.split("\\t").map(_.toLong)).
68      map(x => CtrEvalRow(x(0), x(1), x(2), x(3), x(4))).
69      toDF()
70  }
71
72  def loadQuery(path: String = pathMap("query")): DataFrame = {
73    sc.textFile(path).map(splitCsvRow).map(x => QueryRow(x(0).toLong, x(1))).
74      toDF()
75  }
76
77  def loadKeyword(path: String = pathMap("keyword")): DataFrame = {
```

```
78    sc.textFile(path).map(splitCsvRow).map(x => KeywordRow(x(0).toLong, x(1))).
79      toDF()
80  }
81
82  def loadCtrByUser(
83    path: String = pathMap("ctr_by_user")): DataFrame =
84  {
85    sc.textFile(path).map(x => splitCsvRow(x)).
86      map(x => CtrByUserRow(
87        x(0).toLong, x(1).toLong, x(2).toLong, x(3).toDouble)).
88      toDF()
89  }
90
91  def getQueryCounts(ctrDf: DataFrame): DataFrame = {
92    ctrDf.cache()
93    ctrDf.registerTempTable("ctr")
94    val queryCountDf = sqlContext.sql(
95        "SELECT queryId, COUNT(*) as nQuery " +
96        "FROM ctr " +
97        "GROUP BY queryId"
98      )
99    ctrDf.unpersist()
100   queryCountDf
101 }
102
103 def loadQueryCounts(path: String = pathMap("query_counts")): DataFrame = {
104   sc.textFile(path).map(x => splitCsvRow(x).map(_.toLong)).
105     map(x => QueryCountRow(x(0), x(1))).
106     toDF()
107 }
108
109 def getClickCountByUser(ctrDf: DataFrame): DataFrame = {
110   ctrDf.cache()
111   ctrDf.registerTempTable("ctr")
112   val nClickByUserDf = sqlContext.sql(
113       "SELECT userId, SUM(click) FROM ctr GROUP BY userId")
114   nClickByUserDf
115 }
116
117 def loadClickCountByUser(path: String): DataFrame = {
118   sc.textFile(path).map(x => splitCsvRow(x).map(_.toLong)).
119     map(x => ClickCountByUserRow(x(0), x(1))).
120     toDF()
121 }
122
123 def getQueryCountsByUserWithoutClicks(
124   ctrDf: DataFrame,
125   ctrByUserDf: DataFrame): DataFrame =
126 {
127   ctrDf.cache()
```

```
128   ctrDf.registerTempTable("ctr")
129   val userWithoutClickDf = ctrByUserDf.where(ctrByUserDf("nClick") === 0).
130     select(ctrByUserDf("userId"), ctrByUserDf("nClick"))
131   userWithoutClickDf.cache()
132   userWithoutClickDf.registerTempTable("user_wo_click")
133   val queryCountDf = sqlContext.sql(
134       "SELECT " +
135       " ctr.userId" +
136       ", COUNT(DISTINCT ctr.queryId) as nQuery " +
137       "FROM ctr " +
138       "INNER JOIN user_wo_click ON user_wo_click.userId = ctr.userId " +
139       "GROUP BY ctr.userId"
140     )
141   userWithoutClickDf.unpersist()
142   queryCountDf
143 }
144
145 def loadQueryCountsByUserWoClicks(
146   path: String = pathMap("n_query_by_user_wo_click")): DataFrame =
147 {
148   sc.textFile(path).map(x => splitCsvRow(x).map(_.toLong)).
149     map(x => UniqueQueryCountByUserRow(x(0), x(1))).
150     toDF()
151 }
152
153 def getClickCountUserCountByAd(ctrDf: DataFrame): DataFrame = {
154   ctrDf.cache()
155   ctrDf.registerTempTable("ctr")
156   sqlContext.sql(
157     "SELECT " +
158       "adId, " +
159       "SUM(click) AS nClick, " +
160       "COUNT(DISTINCT userId) AS nUser " +
161       "FROM ctr " +
162       "GROUP BY adId"
163   )
164 }
165
166 def loadClickCountUserCountByAd(
167   path: String = pathMap("n_click_n_user_by_ad")): DataFrame =
168 {
169   sc.textFile(path).map(x => splitCsvRow(x).map(_.toLong)).
170     map(x => ClickCountUserCountByAd(x(0), x(1), x(2))).
171     toDF()
172 }
173
174 def saveDf(path: String, df: DataFrame, delim: String = "\t",
175            nPartitions: Int = 0): Unit = {
176   val rdd = if (nPartitions > 0) df.rdd.repartition(nPartitions) else df.rdd
177   rdd.map(_.mkString(delim)).saveAsTextFile(path)
```

```
178 }
179
180 def getHistogram(
181   df: DataFrame,
182   colName: String,
183   nBuckets: Int = 30): (Array[Double], Array[Long]) =
184 {
185   val values: RDD[Double] = df.withColumn("tmp", df(colName).cast("Double")).
186     select('tmp).rdd.map(row => row.getDouble(0)).cache()
187   values.histogram(nBuckets match {
188     case x if x < 1 => 30
189     //case x if x < 1 => math.sqrt(values.count()).ceil.toInt
190     case _ => nBuckets
191   })
192 }
193
194 def writeLines(path: String, lines: Array[String]): Unit = {
195   val fw = new FileWriter(path)
196   lines.foreach(line => fw.write(line + "\r\n"))
197   fw.close()
198 }
199
200 def saveHistogramData(
201   path: String,
202   histogram: (Array[Double], Array[Long]),
203   xLabel: String,
204   yLabel: String = "frequency"): Unit =
205 {
206   val histData: Array[(Double, Long)] = histogram._1.zip(histogram._2)
207   val lines: Array[String] = Array(s"$xLabel,$yLabel") ++
208     histData.map(x => s"${x._1},${x._2}")
209   writeLines(path, lines)
210 }
211 // ============================================================================
212 // rm id retrieval Queries
213 // ============================================================================
214 def getIdsUserHighCtr(ctrByUser: DataFrame): DataFrame = {
215   ctrByUser.registerTempTable("ctr_u")
216   val highCtrUsers = sqlContext.sql("""
217     SELECT userId
218     FROM ctr_u
219     WHERE (nClick > 30 AND userCTR = 1)
220       OR (nClick > 50 AND userCTR >= 0.95)
221       OR (nClick > 100 AND userCTR >= 0.9)
222     """)
223   highCtrUsers
224 }
225
226 def getIdsUserWoClick(nQueryByUserWoClick: DataFrame): DataFrame = {
227   nQueryByUserWoClick.select(nQueryByUserWoClick("userId"))
```

```
228 }
229
230 def getIdsUserWoClick(nQueryByUserWoClick: DataFrame,
231                       nQueryGT: Option[Int]): DataFrame =
232 {
233   nQueryByUserWoClick.where(nQueryByUserWoClick("nUniqueQuery") > nQueryGT).
234     select(nQueryByUserWoClick("userId"))
235 }
236
237 def getIdsUserWoClick(nQueryByUserWoClick: DataFrame,
238                       nQueryLT: Option[Int]): DataFrame =
239 {
240   nQueryByUserWoClick.where(nQueryByUserWoClick("nUniqueQuery") < nQueryLT).
241     select(nQueryByUserWoClick("userId"))
242 }
243
244 def getIdsUserWoClick(nQueryByUserWoClick: DataFrame,
245                       nQueryLT: Int,
246                       nQueryGT: Int): DataFrame =
247 {
248   nQueryByUserWoClick.registerTempTable("n_query_by_user_wo_click")
249   sqlContext.sql(s"""
250     SELECT userId
251     FROM n_query_by_user_wo_click
252     WHERE nUniqueQuery < $nQueryLT
253       OR nUniqueQuery > $nQueryGT
254     """)
255 }
256
257 def getIdUserWoClick4to100(nQueryByUserWoClick: DataFrame): DataFrame = {
258   getIdsUserWoClick(nQueryByUserWoClick, 4, 100)
259 }
260
261 def getIdAdClickLT3UserLT3(nClickNUserByAd: DataFrame): DataFrame = {
262   nClickNUserByAd.registerTempTable("ns_by_ad")
263   sqlContext.sql("""
264 SELECT adId
265 FROM ns_by_ad
266 WHERE nClick < 3 OR nUser < 3
267   """)
268 }
269
270 def getIdAdClickGE3UserGE3(nClickNUserByAd: DataFrame): DataFrame = {
271   nClickNUserByAd.registerTempTable("ns_by_ad")
272   sqlContext.sql("""
273 SELECT adId
274 FROM ns_by_ad
275 WHERE nClick >= 3 AND nUser >= 3
276   """)
277 }
```

```
278
279 // id removal / keeping
280 case class rmUserIdRow(userId: Long)
281 case class keepAdIdRow(adId: Long)
282
283 def loadRmUserId(path: String): DataFrame = {
284   sc.textFile(path).map(x => rmUserIdRow(x.toLong)).toDF()
285 }
286
287 def loadKeepAdId(path: String): DataFrame = {
288   sc.textFile(path).map(x => keepAdIdRow(x.toLong)).toDF()
289 }
290
291 // def filterCtr(): DataFrame = {
292 //   val ctr = loadCtr(dataDir + "ctr_op").cache()
293 //   ctr.registerTempTable("ctr")
294
295 //   val highCtrUser = loadRmUserId(dataDir + "rm_user_high_ctr")
296 //   val fewQueryWoClickUser = loadRmUserId(
297 //       dataDir + "rm_user_wo_click_nquery_lt_4_gt_100")
298 //   val rmUser = highCtrUser.unionAll(fewQueryWoClickUser).cache()
299 //   rmUser.registerTempTable("rm_user")
300
301 //   val keepAd = loadKeepAdId(dataDir + "keep_ad_n_click_ge_3_n_user_ge_3").
302 //     cache()
303 //   keepAd.registerTempTable("keep_ad")
304
305 //   sqlContext.sql("""
306 //     SELECT ctr.*
307 //     FROM ctr
308 //     LEFT OUTER JOIN rm_user ON rm_user.userId = ctr.userId
309 //     INNER JOIN keep_ad ON keep_ad.adId = ctr.adId
310 //     WHERE rm_user.userId IS NULL
311 //   """)
312 // }
313
314 // def mungeCtr(): DataFrame = {
315 //   val ctr = loadCtr(dataDir + "ctr_op.txt").cache()
316 //   ctr.registerTempTable("ctr")
317
318 //   val rmUser = loadRmUserId(dataDir + "rm_user.txt").cache()
319 //   rmUser.registerTempTable("rm_user")
320
321 //   val keepAd = loadKeepAdId(dataDir + "keep_ad.txt").
322 //     cache()
323 //   keepAd.registerTempTable("keep_ad")
324
325 //   sqlContext.sql("""
326 //     SELECT ctr.*
327 //     FROM ctr
```

```
328 //     LEFT OUTER JOIN rm_user ON rm_user.userId = ctr.userId
329 //     LEFT OUTER JOIN keep_ad ON keep_ad.adId = ctr.adId
330 //     WHERE rm_user.userId IS NULL AND keep_ad.adId IS NOT NULL
331 //   """)
332 // }
333
334 def removeUsers(): DataFrame = {
335   val ctr = loadCtr(dataDir + "ctr_op.txt")
336   ctr.registerTempTable("ctr")
337
338   val rmUser = loadRmUserId(dataDir + "rm_user.txt")
339   rmUser.registerTempTable("rm_user")
340
341   sqlContext.sql("""
342     SELECT ctr.*
343     FROM ctr
344     LEFT OUTER JOIN rm_user ON rm_user.userId = ctr.userId
345     WHERE rm_user.userId IS NULL
346   """)
347 }
348
349 def removeAds(): DataFrame = {
350   val ctr = loadCtr(dataDir + "ctr_rm_user")
351   ctr.registerTempTable("ctr")
352
353   val keepAd = loadKeepAdId(dataDir + "keep_ad.txt")
354   keepAd.registerTempTable("keep_ad")
355
356   sqlContext.sql("""
357     SELECT ctr.*
358     FROM ctr
359     LEFT OUTER JOIN keep_ad ON keep_ad.adId = ctr.adId
360     WHERE keep_ad.adId IS NOT NULL
361   """)
362 }
363
364 // ================================================================================
365 // User Data Preparation
366 // ================================================================================
367 case class UserTokensRow(userId: Long, tokens: String)
368
369 def getUserQuerytokens(ctrDf: DataFrame, queryDf: DataFrame): DataFrame = {
370   ctrDf.registerTempTable("ctr")
371   queryDf.registerTempTable("query")
372   sqlContext.sql("""
373     SELECT ctr.userId, query.tokens
374     FROM ctr
375     LEFT JOIN query ON query.queryId = ctr.queryId
376   """)
377 }
```

```
378
379 // def getMergedQuerytokensByUser(userTokens: DataFrame): DataFrame = {
380 //   val groupedByUser = userTokens.rdd.groupBy(x => x(0))
381 //   groupedByUser.
382 //     map(x => UserTokensRow(x._1.asInstanceOf[Number].longValue,
383 //                            x._2.map(r => r(1)).mkString("|"))).
384 //     toDF()
385 // }
386
387 def getMergedTokensByUser(userTokens: DataFrame): DataFrame = {
388   userTokens.rdd.
389     map(row => (row(0), row(1))).
390     reduceByKey(_ + "|" + _).
391     map(row => UserTokensRow(row._1.asInstanceOf[Number].longValue,
392                              row._2.toString)).
393     toDF()
394 }
395
396 def loadUserTokens(path: String): DataFrame = {
397   sc.textFile(path).
398     map(splitCsvRow).
399     map(x => UserTokensRow(x(0).toLong, x(1))).
400     toDF()
401 }
402
403 // ================================================================================
404 // CTR Evaluations
405 // ================================================================================
406 case class UserSegmentRow(userId: Long, segmentId: Int)
407
408 def loadUserSegments(path: String): DataFrame = {
409   sc.textFile(path).
410     map(x => splitCsvRow(x)).
411     map(x => UserSegmentRow(x(0).toLong, x(1).toInt)).
412     toDF()
413 }
414
415 // --- used for MLLib LDA
416 def getUserSegment(segmentDistribution: Vector): Int = {
417   segmentDistribution.toArray.zipWithIndex.maxBy(_._1)._2
418 }
419
420 def getUserSegments(segmentDistributions: RDD[(Long, Vector)]): DataFrame = {
421   segmentDistributions.
422     map(x => UserSegmentRow(x._1, getUserSegment(x._2))).
423     toDF()
424 }
425 def loadUserSegments(nSegment: Int, nIter: Int): DataFrame = {
426   loadUserSegments(
427     dataDir + s"user_segment_s${nSegment}_i${nIter}_sample_100k")
```

```
}
// ---

// merging BIDMACH userSegment with KDD user
case class BidmachUserSegmentRow(bmUserId: Long, segmentId: Int)
case class UserIdMapSparseRow(bmUserId: Long, userId: Long)

val bidmachSegmentDir = projectDir + "data/kdd/bidmach/"
def loadBidmachUserSegment(path: String): DataFrame = {
  sc.textFile(path).
    map(x => x.split("\\t")).
    map(x => BidmachUserSegmentRow(x(0).toLong, x(1).toInt)).
    toDF()
}

// def getUserSegmentFromBIDMach(
//   userIdMapSparse: DataFrame, bidMachUserSegment: DataFrame): DataFrame =
// {
//   userIdMapSparse.registerTempTable("user_id_map_sparse")
//   bidMachUserSegment.registerTempTable("bidmach_user_segment")
//   sqlContext.sql("""
//     SELECT
//     u_map.userId
//     , bm_u_s.segmentId
//     FROM user_id_map_sparse as u_map
//     JOIN bidmach_user_segment as bm_u_s ON bm_u_s.rowId = u_map.rowId
//     """)
// }

// def getMergedUserSegmentFromBIDMach(
//   pathBidMachSegment: String
// ): DataFrame = {
//   val userSegment = loadBIDMachUserSegment(pathBidMachSegment)
//   val userIdMap = loadUserIdMapSparse(dataDir + "user_id_map_sparse")
//   getUserSegmentFromBIDMach(userIdMap, userSegment)
// }

case class CtrByAdRow(adId: Long, ctr: Double,
                      nClick: Long, nImpression: Long, nUser: Long)

def getCtrByAd(ctr: DataFrame): DataFrame = {
  ctr.registerTempTable("ctr")
  sqlContext.sql("""
    SELECT
    adId
    , SUM(click) / SUM(impression) AS ctr
    , SUM(click) AS nClick
    , SUM(impression) AS nImpression
    , COUNT(DISTINCT bmUserId) AS nUser
    FROM ctr
```

```
478     GROUP BY adId
479     ORDER BY adId
480   """)
481 }
482
483 def loadCtrByAd(path: String): DataFrame = {
484   sc.textFile(path).
485     map(x => x.split("\\t")).
486     map(x => CtrByAdRow(
487           x(0).toLong, x(1).toDouble,
488           x(2).toLong, x(3).toLong, x(4).toLong)).
489     toDF()
490 }
491
492
493 case class DeltaCtrByAdRow(adId: Long, deltaCtr: Double)
494
495 // def getDeltaCtrByAd(ctrByAd: DataFrame,
496 //                     ctrBestSegmentByAd: DataFrame): DataFrame =
497 // {
498 //   sqlContext.dropTempTable("ctr")
499
500 //   ctrByAd.registerTempTable("ctr")
501 //   ctrBestSegmentByAd.registerTempTable("ctr_segment")
502 //   sqlContext.sql("""
503 //     SELECT ctr.adId, (ctr_segment.ctr - ctr.ctr) / ctr.ctr AS delta_ctr
504 //     FROM ctr
505 //     LEFT JOIN ctr_segment ON ctr_segment.adId = ctr.adId
506 //     GROUP BY ctr.adId
507 //   """)
508 // }
509
510 def loadDeltaCtrByAd(path: String): DataFrame = {
511   sc.textFile(path).
512     map(x => splitCsvRow(x)).
513     map(x => DeltaCtrByAdRow(x(0).toLong, x(1).toDouble)).
514     toDF()
515 }
516
517 case class DeltaCtrRow(deltaCtr: Double)
518
519 // def getDeltaCtr(deltaCtrByAd: DataFrame): DataFrame = {
520 //   sqlContext.dropTempTable("delta_ctr")
521
522 //   deltaCtrByAd.registerTempTable("delta_ctr")
523 //   sqlContext.sql("SELECT AVG(delta_ctr.delta_ctr FROM delta_ctr)")
524 // }
525
526 def loadDeltaCtr(path: String): DataFrame = {
527   sc.textFile(path).
```

```
528     map(x => DeltaCtrRow(x.toDouble)).
529     toDF()
530 }
531
532 def getCtrBestByAdBySegment(
533   ctr: DataFrame,
534   userSegment: DataFrame): DataFrame =
535 {
536   ctr.registerTempTable("ctr")
537   userSegment.registerTempTable("user_segment")
538   val ctrWithSegment = sqlContext.sql("""
539     SELECT
540     ctr.adId
541     , ctr.click
542     , ctr.impression
543     , user_segment.segmentId
544     FROM ctr
545     INNER JOIN user_segment ON user_segment.bmUserId = ctr.bmUserId
546   """).cache()
547   ctrWithSegment.registerTempTable("ctr_w_seg")
548
549   val timeCtrByAdBySegment = System.nanoTime()
550   val ctrByAdBySegment =  sqlContext.sql("""
551     SELECT
552     adId
553     , segmentId
554     , SUM(click) / SUM(impression) AS ctr
555     FROM ctr_w_seg
556     GROUP BY adId, segmentId
557   """).cache()
558   ctrByAdBySegment.registerTempTable("ctr_by_ad_by_seg")
559   println(s"${(System.nanoTime() - timeCtrByAdBySegment) / 1e9}" +
560             " s elapsed for timeCtrByAdBySegment")
561
562   val ctrBestSegmentByAd = sqlContext.sql("""
563     SELECT
564     adId
565     , MAX(ctr) AS CTR
566     FROM ctr_by_ad_by_seg
567     GROUP BY adId
568     """)
569   ctrBestSegmentByAd
570 }
571
572 def getDeltaCtrByAd(
573       ctrBestByAdBySegment: DataFrame,
574       ctrByAd: DataFrame): DataFrame =
575     {
576       ctrBestByAdBySegment.registerTempTable("ctr_best_by_ad_by_seg")
577       ctrByAd.registerTempTable("ctr_by_ad")
```

```
578       val deltaCtrByAd = sqlContext.sql("""
579             SELECT
580             ctr_by_ad.adId
581             , IF(ctr_by_ad.ctr > 0,
582                 ((ctr_best_by_ad_by_seg.ctr - ctr_by_ad.ctr) / ctr_by_ad.ctr),
583                 0) AS delta_ctr
584             FROM ctr_by_ad
585             JOIN ctr_best_by_ad_by_seg
586             ON ctr_best_by_ad_by_seg.adId = ctr_by_ad.adId
587         """)
588       deltaCtrByAd
589     }
590
591 def getDeltaCtrByAd2(
592   ctrBestByAdBySegment: DataFrame,
593   ctrByAd: DataFrame): DataFrame =
594 {
595   ctrBestByAdBySegment.registerTempTable("ctr_best_by_ad_by_seg")
596   ctrByAd.registerTempTable("ctr_by_ad")
597   val deltaCtrByAd = sqlContext.sql("""
598     SELECT
599     ctr_by_ad.adId
600     , IF(ctr_by_ad.ctr > 0,
601         ((ctr_best_by_ad_by_seg.ctr - ctr_by_ad.ctr) / ctr_by_ad.ctr),
602         0) AS delta_ctr
603     FROM ctr_by_ad
604     JOIN ctr_best_by_ad_by_seg ON ctr_best_by_ad_by_seg.adId = ctr_by_ad.adId
605   """)
606   deltaCtrByAd
607 }
608
609
610 def getDeltaCtr(deltaCtrByAd: DataFrame): DataFrame = {
611   deltaCtrByAd.registerTempTable("delta_ctr_by_ad")
612   val deltaCtr = sqlContext.sql("""
613     SELECT AVG(delta_ctr)
614     FROM delta_ctr_by_ad
615     WHERE delta_ctr != 0
616   """)
617   deltaCtr
618 }
619
620
621 def saveEvaluationDf(ldaDesc: String): Unit = {
622   val ctr = loadCtrEval(dataDir + "ctr_eval.tsv")
623   val userSegment = loadBidmachUserSegment(
624       bidmachSegmentDir + s"cv1of5_user_segment_${ldaDesc}.txt")
625
626   val timeCtrBestByAdBySegment = System.nanoTime()
627   val ctrBestByAdBySegment = getCtrBestByAdBySegment(ctr, userSegment).cache()
```

```
628   println(s"${(System.nanoTime() - timeCtrBestByAdBySegment) / 1e9} " +
629             "timeCtrBestByAdBySegment")
630
631   val timeDeltaCtrByAd = System.nanoTime()
632   val deltaCtrByAd = getDeltaCtrByAd(
633       ctrBestByAdBySegment,
634       loadCtrByAd(dataDir + "ctr_by_ad.tsv")).cache()
635   println(s"${(System.nanoTime() - timeDeltaCtrByAd) / 1e9} timeDeltaCtrByAd")
636
637   val timeSaveDeltaCtr = System.nanoTime()
638   saveDf(dataDir + s"cv1of5_bidmach_delta_ctr_${ldaDesc}",
639          getDeltaCtr(deltaCtrByAd))
640   println(s"${(System.nanoTime() - timeSaveDeltaCtr) / 1e9} timeSaveDeltaCtr")
641 }
642
643
644 val descr = "s200_p1_mb100k"
645
646 val t0 = System.nanoTime()
647 saveEvaluationDf(descr)
648 val t1 = System.nanoTime()
649 println(s"elapsed: ${((t1 - t0) / 1e9)} s")
650 println("DONE!")
```

sparsify-user-tokens-interactive.scala

```
1  import scala.collection.mutable
2  import org.apache.spark.mllib.linalg.{Vector, Vectors}
3  import org.apache.spark.rdd.RDD
4  import org.apache.spark.sql.DataFrame
5
6  sc.setCheckpointDir("/home/hm/checkpoints_spark")
7  val dataDir = "/home/hm/math_master_ktu/magistro_baigiamasis_darbas/" +
8    "project/data/kdd/"
9
10
11 val tokenized: RDD[(Long, Seq[Long])] = sc.textFile(
12     dataDir + "user_tokens").
13   map({ line => val idTokens = line.split(",")
14         (idTokens(0).toLong, idTokens(1).split("\\|").map(_.toLong).toSeq) })
15
16 val termCounts: Array[(Long, Long)] =
17   tokenized.map(_._2).flatMap(_.map(_ -> 1L)).
18     reduceByKey(_ + _).collect().sortBy(-_._2)
19
20 val numStopwords = 0
21 val vocabArray: Array[Long] =
22   termCounts.takeRight(termCounts.size - numStopwords).map(_._1)
```

```
val vocab: Map[Long, Int] = vocabArray.zipWithIndex.toMap

val tokenizedWithIndex: RDD[((Long, Seq[Long]), Long)] = {
  tokenized.zipWithIndex
}

val userMapToSparse: RDD[(Long, Long)] = {
  tokenizedWithIndex.sortBy(x => x._2)
}

val userMapToSparseStr: RDD[String] = {
  userMapToSparse.map(x => s"${x._2},${x._1._1}")
}
userMapToSparseStr.saveAsTextFile(dataDir + "user_id_map_sparse")

val documentsToSparse: RDD[(Long, Long, Double)] = {
  tokenizedWithIndex.map { case ((id, tokens), rowId) =>
    val counts = new mutable.HashMap[Long, Double]()
    tokens.foreach { term =>
      if (vocab.contains(term)) {
        val idx = vocab(term)
        counts(idx) = counts.getOrElse(idx, 0.0) + 1.0
      }
    }
    counts.toSeq.map(x => (rowId, x._1, x._2))
  }.flatMap(x => x).
    sortBy(x => (x._1, x._2))
}

val documentsToSparseStr: RDD[String] = {
  documentsToSparse.map(x => s"${x._1},${x._2},${x._3}")
}
documentsToSparseStr.saveAsTextFile(dataDir + "user_tokens_sparse")
```

btusersegmentation/config.json

```
{
  "path_project": [
    "/home", "hm", "math_master_ktu",
    "magistro_baigiamasis_darbas", "project/"
  ],
  "bidmach": {
    "bin": ["/home", "hm", "BIDMach_1.0.2-full-linux-x86_64", "bidmach"],
    "path_scripts": ["btusersegmentation", "bidmach_scripts/"],
    "path_user_tokens": ["data","bidmach", "user_tokens/"],
    "path_results_lda": ["results_btusersegmentation", "bidmach/"],
    "script_cv": "lda_crossvalidation.ssc"
  },
```

```
  "spark": {
    "path_apps": ["spark_apps/"],
    "jar": [
      "/home", "hm", "math_master_ktu",
      "magistro_baigiamasis_darbas", "project", "spark_apps",
      "sparkevaluation", "target", "scala-2.10",
      "sparkevaluation_2.10-1.0.jar"
    ],
    "submit_params": ["--master", "local[4]", "--driver-memory", "13g"],
      "path_output_base": ["results_btusersegmentation", "spark/"]
  }
}
```

btusersegmentation/job_params.json

```
{
  "ldaSegmentationRuns": [
    {
      "nSegment": 10, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 20, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 30, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 45, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
```

```
    },
    {
      "nSegment": 70, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 110, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 160, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 240, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 350, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
      "nSegment": 530, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },
    {
```

```
      "nSegment": 800, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": true, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "g2",
      "prefix_spark_metrics": "g2"
    },

    {
      "nSegment": 10, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 20, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 30, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 45, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 70, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 110, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
```

```
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 160, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 240, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 350, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 530, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 800, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmu",
      "prefix_spark_metrics": "fmu"
    },

    {
      "nSegment": 10, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
```

```
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 20, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 30, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 45, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 70, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 110, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 160, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
```

```
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 240, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 350, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 530, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    },
    {
      "nSegment": 800, "nDatasetPass": 1, "mbSize": 100000, "nCvFold": 5,
      "idxsCvFold": [1,2,3,4,5], "idxsRun": [1], "mixins": [""],
      "nIter": 5, "has_bm": false, "has_spark": true, "isDone": true,
      "prefix_bm": "g2",
      "prefix_spark_output": "fmc",
      "prefix_spark_metrics": "fmu"
    }
  ]
}
```

btusersegmentation/btlda.py

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
"""
Run Lattent Dirichlet Allocation simulations.
Model training with BIDMach
Results evaluation with Apache Spark
"""

```

```
import os
import time
from subprocess import call
import logging
from core import (
    here,
    PATH_BM_BIN,
    PATH_BM_USER_TOKENS,
    PATH_TMP_BIDMACH_VALS,
    PATH_BM_CV_SCRIPT,
    CMD_SPARK_BASE,
    load_job_params,
    mkstr_jobname,
    mkstr_bm_jobname,
    mkstr_bm_job_fold_dir,
    mkstr_cvinfo,
    ensure_dir,
    mkstr_mean_entropy_cv_fname,
    mkstr_log_likelihood_cv_fname,
    save_bidmach_params,
    mkstr_user_segments_merged,
    mkstr_mean_entropy_merged,
    mkstr_log_likelihood_merged,
    mkstr_spark_precision_dir,
    mkstr_spark_recall_dir,
    mkstr_spark_fmeasure_dir,
    mkstr_spark_clickentropy_dir,
    mkstr_spark_metrics_by_ad_by_seg_dir,
    mkstr_spark_n_seg_per_ad_dir,
    cat_and_rm_spark_results_parts
)


logging.basicConfig(format='%(asctime)s::%(levelname)s::%(message)s',
                    datefmt='%Y-%m-%d %H:%M:%S',
                    filename=os.path.join(here, "log", "btlda.log"),
                    level=logging.DEBUG)


def main():
    job_params = load_job_params()

    for job_info in job_params["ldaSegmentationRuns"]:
        if job_info["isDone"]:
            continue

        logging.info("STARTING JOB --- {}".format(mkstr_jobname(job_info)))

        for i_fold in job_info["idxsCvFold"]:

```

```
59              bm_job_fold_dir = mkstr_bm_job_fold_dir(i_fold, job_info)
60              bm_job_fold_segment_parts_dir = os.path.join(
61                  bm_job_fold_dir, "user_segment_parts")
62              ensure_dir(bm_job_fold_segment_parts_dir)
63
64              tmp_bidmach_vals = {
65                  "nIter": job_info["nIter"],
66                  "mixinsNames": job_info["mixins"],
67                  "iFold": i_fold - 1,
68                  "nSegments": job_info["nSegment"],
69                  "miniBatchSize": job_info["mbSize"],
70                  "nPasses": job_info["nDatasetPass"],
71                  "nFolds": job_info["nCvFold"],
72                  "cvInfo": mkstr_cvinfo(i_fold, job_info["nCvFold"]),
73                  "jobName": mkstr_bm_jobname(job_info),
74                  "dataDir": PATH_BM_USER_TOKENS,
75                  "dataSaveDir": bm_job_fold_dir + os.path.sep,
76                  "meanEntropyFilename": mkstr_mean_entropy_cv_fname(
77                      i_fold, job_info),
78                  "logLikelihoodFilename": mkstr_log_likelihood_cv_fname(
79                      i_fold, job_info),
80              }
81              save_bidmach_params(tmp_bidmach_vals)
82              cmd_bm = [
83                  PATH_BM_BIN,
84                  PATH_TMP_BIDMACH_VALS,
85                  PATH_BM_CV_SCRIPT
86              ]
87              if job_info["has_bm"]:
88
89                  logging.info(
90                      "STARTING JOB --- {} -- BIDMACH - FOLD {:02d}".format(
91                          mkstr_jobname(job_info), i_fold))
92
93                  call(cmd_bm)
94
95                  logging.info(
96                      "FINISHED JOB --- {} -- BIDMACH - FOLD {:02d}".format(
97                          mkstr_jobname(job_info), i_fold))
98
99          if job_info["has_bm"]:
100
101             logging.info(
102                 "STARTING JOB --- {} -- BIDMACH - CAT RM".format(
103                     mkstr_jobname(job_info)))
104
105             user_segment_parts_dirs = [
106                 os.path.join(mkstr_bm_job_fold_dir(i, job_info),
107                              "user_segment_parts", "*")
108                 for i in range(1, job_info["nCvFold"] + 1)]
```

```
109             fname_user_segments_merged = mkstr_user_segments_merged(job_info)
110             cmd_cat_user_segment = " ".join(
111                 ["cat", " ".join(user_segment_parts_dirs), ">",
112                     fname_user_segments_merged])
113             if sum([len(os.listdir(os.path.dirname(p))) > 0 for
114                     p in user_segment_parts_dirs]) == job_info["nCvFold"]:
115                 os.system(cmd_cat_user_segment)
116                 while not os.path.exists(fname_user_segments_merged):
117                     time.sleep(2)
118                 os.system("rm {}".format(" ".join(user_segment_parts_dirs)))
119
120             mean_entropy_files = [
121                 mkstr_mean_entropy_cv_fname(i, job_info)
122                 for i in range(1, job_info["nCvFold"] + 1)]
123             fname_mean_entropy_merged = mkstr_mean_entropy_merged(job_info)
124             cmd_cat_mean_entropy = " ".join(
125                 ["cat", " ".join(mean_entropy_files), ">",
126                     fname_mean_entropy_merged])
127             if sum([os.path.exists(p) for
128                     p in mean_entropy_files]) == job_info["nCvFold"]:
129                 os.system(cmd_cat_mean_entropy)
130                 while not os.path.exists(fname_mean_entropy_merged):
131                     time.sleep(2)
132
133             log_likelihood_files = [
134                 mkstr_log_likelihood_cv_fname(i, job_info)
135                 for i in range(1, job_info["nCvFold"] + 1)]
136             fname_log_likelihood_merged = mkstr_log_likelihood_merged(job_info)
137             cmd_cat_log_likelihood = " ".join(
138                 ["cat", " ".join(log_likelihood_files), ">",
139                     fname_log_likelihood_merged])
140             if sum([os.path.exists(p) for
141                     p in log_likelihood_files]) == job_info["nCvFold"]:
142                 os.system(cmd_cat_log_likelihood)
143                 while not os.path.exists(fname_log_likelihood_merged):
144                     time.sleep(2)
145
146             logging.info(
147                 "FINISHED JOB --- {} -- BIDMACH - CAT RM".format(
148                     mkstr_jobname(job_info)))
149
150         path_spark_precision = mkstr_spark_precision_dir(job_info)
151         path_spark_recall = mkstr_spark_recall_dir(job_info)
152         path_spark_fmeasure = mkstr_spark_fmeasure_dir(job_info)
153         path_spark_clickentropy = mkstr_spark_clickentropy_dir(job_info)
154         path_spark_ctr_by_ad_by_segment = mkstr_spark_metrics_by_ad_by_seg_dir(
155             job_info)
156         path_spark_n_seg_per_ad = mkstr_spark_n_seg_per_ad_dir(job_info)
157         cmd_spark = CMD_SPARK_BASE + [
158             mkstr_user_segments_merged(job_info),
```

```
159             path_spark_precision,
160             path_spark_recall,
161             path_spark_fmeasure,
162             path_spark_clickentropy,
163             str(job_info["nSegment"]),
164             path_spark_ctr_by_ad_by_segment,
165             path_spark_n_seg_per_ad
166         ]
167         if job_info["has_spark"]:
168 
169             logging.info("STARTING JOB --- {} -- SPARK".format(
170                 mkstr_jobname(job_info)))
171 
172             call(cmd_spark)
173 
174             logging.info("FINISHED JOB --- {} -- SPARK".format(
175                 mkstr_jobname(job_info)))
176 
177             logging.info("STARTING JOB --- {} -- SPARK - CAT RM".format(
178                 mkstr_jobname(job_info)))
179 
180             cat_and_rm_spark_results_parts(path_spark_precision)
181             cat_and_rm_spark_results_parts(path_spark_recall)
182             cat_and_rm_spark_results_parts(path_spark_fmeasure)
183             cat_and_rm_spark_results_parts(path_spark_clickentropy)
184 
185             logging.info("FINISHED JOB --- {} -- SPARK - CAT RM".format(
186                 mkstr_jobname(job_info)))
187 
188         logging.info("FINISHED JOB --- {}".format(mkstr_jobname(job_info)))
189 
190 if __name__ == "__main__":
191     logging.info(
192         "\n"
193         "==============================\n"
194         "======= STARTING RUN ========\n"
195         "==============================\n"
196         )
197     main()
198     logging.info(
199         "\n"
200         "==============================\n"
201         "======= FINISHED RUN ========\n"
202         "==============================\n"
203         )
```

btusersegmentation/core.py

```
import json
import os
from collections import Iterable
import time


here = os.path.abspath(os.path.dirname(__file__))
with open(os.path.join(here, "config.json")) as f:
    config = json.load(f)

PATH_BM_BIN = os.path.join(*config["bidmach"]["bin"])
PATH_BM_RESULTS = os.path.join(*(
    config["path_project"] +
    config["bidmach"]["path_results_lda"]))
PATH_BM_USER_TOKENS = os.path.join(*(
    config["path_project"] +
    config["bidmach"]["path_user_tokens"]))
PATH_TMP_BIDMACH_VALS = os.path.join(here, "tmp_bidmach_vals.ssc")
PATH_BM_CV_SCRIPT = os.path.join(*(
    config["path_project"] +
    config["bidmach"]["path_scripts"] +
    [config["bidmach"]["script_cv"]]))

CMD_SPARK_BASE = (
    ["spark-submit"] + config["spark"]["submit_params"] +
    [os.path.join(*config["spark"]["jar"])])

PATH_SPARK_RESULTS = os.path.join(*(
    config["path_project"] +
    config["spark"]["path_output_base"]))


def load_job_params():
    with open(os.path.join(here, "job_params.json")) as f:
        return json.load(f)


def ensure_path(path):
    dirname = os.path.dirname(path)
    ensure_dir(dirname)


def ensure_dir(dirname):
    if not os.path.exists(dirname):
        os.makedirs(dirname)


```

```
48  def cast_primitive_to_scala(v):
49      return '"{}"'.format(v) if type(v) is str else v
50
51
52  def cast_list_to_scala(v):
53      return "Array({})".format(", ".join(
54          [cast_primitive_to_scala(e) for e in v]))
55
56
57  def cast_dict_to_scala(v):
58      raise NotImplemented
59
60
61  def is_iterable(obj):
62      return not isinstance(obj, str) and isinstance(obj, Iterable)
63
64
65  # or do explicit generation in btlda.py
66  def cast_value(v):
67      if is_iterable(v):
68          if isinstance(v, list):
69              return cast_list_to_scala(v)
70          elif isinstance(v, dict):
71              return cast_dict_to_scala(v)
72      else:
73          return cast_primitive_to_scala(v)
74
75
76  def save_bidmach_params(params):
77      with open(PATH_TMP_BIDMACH_VALS, "w") as f:
78          f.writelines(
79              ["val {} = {}\r\n".format(
80                  k, cast_value(v))
81               for k, v in params.items()])
82
83
84  def mkstr_bm_job_dir(job_info):
85      return os.path.join(
86          PATH_BM_RESULTS,
87          "{}_cvf{:02d}".format(mkstr_bm_jobname(job_info), job_info["nCvFold"]))
88
89
90  def mkstr_bm_job_fold_dir(i_fold, job_info):
91      return os.path.join(
92          mkstr_bm_job_dir(job_info), "{}_cv{:02d}of{:02d}".format(
93              mkstr_bm_jobname(job_info), i_fold, job_info["nCvFold"]))
94
95
96  def mkstr_jobname(job_info):
97      return "mxs{}_i{:04d}_s{:04d}_p{:02d}_mb{:.0f}k".format(
```

```
 98          "-".join(job_info["mixins"]),
 99          job_info["nIter"],
100          job_info["nSegment"],
101          job_info["nDatasetPass"],
102          job_info["mbSize"] / 1000)
103
104
105  def mkstr_bm_jobname(job_info):
106      return "{}{}".format(
107          "{}_".format(job_info["prefix_bm"]) if job_info["prefix_bm"] else "",
108          mkstr_jobname(job_info))
109
110
111  def mkstr_cvinfo(i_fold, n_fold):
112      return "cv{:02d}of{:02d}".format(i_fold, n_fold)
113
114
115  def mkstr_user_segments_merged(job_info):
116      return os.path.join(
117          mkstr_bm_job_dir(job_info),
118          "{}_cvf{:02d}_user_segment.txt".format(
119              mkstr_bm_jobname(job_info), job_info["nCvFold"]))
120
121
122  def mkstr_mean_entropy_cv_fname(i_fold, job_info):
123      return os.path.join(
124          mkstr_bm_job_fold_dir(i_fold, job_info),
125          "{}_{}_mean_entropy.txt".format(
126              mkstr_jobname(job_info),
127              mkstr_cvinfo(i_fold, job_info["nCvFold"])))
128
129
130  def mkstr_mean_entropy_merged(job_info):
131      return os.path.join(
132          mkstr_bm_job_dir(job_info), "{}_cvf{:02d}_mean_entropy.txt".format(
133              mkstr_bm_jobname(job_info), job_info["nCvFold"]))
134
135
136  def mkstr_log_likelihood_cv_fname(i_fold, job_info):
137      return os.path.join(
138          mkstr_bm_job_fold_dir(i_fold, job_info),
139          "{}_{}_log_likelihood.txt".format(
140              mkstr_bm_jobname(job_info),
141              mkstr_cvinfo(i_fold, job_info["nCvFold"])))
142
143
144  def mkstr_log_likelihood_merged(job_info):
145      return os.path.join(
146          mkstr_bm_job_dir(job_info), "{}_cvf{:02d}_log_likelihood.txt".format(
147              mkstr_bm_jobname(job_info), job_info["nCvFold"]))
```

```
148
149
150 def mkstr_spark_out_jobname(job_info):
151     return "{}{}".format(
152         "{}_".format(job_info["prefix_spark_output"]) if
153         job_info["prefix_spark_output"]
154         else "",
155         mkstr_jobname(job_info))
156
157
158 def mkstr_spark_metrics_jobname(job_info):
159     return "{}{}".format(
160         "{}_".format(job_info["prefix_spark_metrics"]) if
161         job_info["prefix_spark_metrics"]
162         else "",
163         mkstr_jobname(job_info))
164
165
166 def mkstr_spark_base_out_dir(job_info):
167     base_res = "{}_cvf{:02d}".format(
168         mkstr_spark_out_jobname(job_info),
169         job_info["nCvFold"])
170     return os.path.join(
171         PATH_SPARK_RESULTS,
172         base_res,
173         "{}_byad_".format(base_res))
174
175
176 def mkstr_spark_precision_dir(job_info):
177     return mkstr_spark_base_out_dir(job_info) + "precision"
178
179
180 def mkstr_spark_recall_dir(job_info):
181     return mkstr_spark_base_out_dir(job_info) + "recall"
182
183
184 def mkstr_spark_fmeasure_dir(job_info):
185     return mkstr_spark_base_out_dir(job_info) + "fmeasure"
186
187
188 def mkstr_spark_clickentropy_dir(job_info):
189     return mkstr_spark_base_out_dir(job_info) + "clickentropy"
190
191
192 def mkstr_spark_n_seg_per_ad_dir(job_info):
193     return mkstr_spark_base_out_dir(job_info) + "n_segment"
194
195
196 def mkstr_spark_metrics_by_ad_by_seg_dir(job_info):
197     base_res = "{}_cvf{:02d}".format(
```

```
198         mkstr_spark_metrics_jobname(job_info),
199         job_info["nCvFold"])
200     return os.path.join(
201         PATH_SPARK_RESULTS,
202         base_res,
203         "{}_byad_byseg_metrics".format(base_res))
204
205
206 def cat_and_rm_spark_results_parts(path_spark_results):
207     pattern_spark_results_parts = os.path.join(
208         path_spark_results, "*")
209     os.system(" ".join([
210         "cat", pattern_spark_results_parts, ">",
211         path_spark_results + ".txt"]))
212     # should not be needed, but just in case ;)
213     while not os.path.exists(path_spark_results + ".txt"):
214         time.sleep(2)
215     os.system("rm {}".format(pattern_spark_results_parts))
```

btusersegmentation/plotting/rscripts/clear_env.R

```
1 rm(list=ls(all.names=TRUE))
```

btusersegmentation/plotting/rscripts/config_plot.R

```
1  ################################################################################
2  # Plots configuration
3  ################################################################################
4  font_family_title_report <- 'Times New Roman'
5  font_family_report <- 'Times New Roman'
6  # font_family_report <- 'CM Roman CE'
7  font_family_title_slides <- 'sans'
8  font_family_slides <- 'sans'
9  # font_family_slides <- 'CM Sans'
10 line_width <- 1
11 size_label <- 12
12 size_title <- 12
13 size_main <- 12
14 size_point <- 2
15 size_line <- 0.05
16 point_alpha <- 0.7
17 ribbon_alpha <- 0.15
18
19 nxbreaks <- 15
20 nybreaks <- 10
21
```

```
grey_start <- 0
grey_end <- 0.9

plot_ads_hist2d_width <- 10
plot_ads_hist2d_height <- 10
plot_ads_hist2d_scale <- 1

plot_aggr_width <- 15
plot_aggr_height <- 9
plot_aggr_scale <- 1.2

has_plot_embed_font <- TRUE

plot_aggr_x_end_limit <- 800
```

btusersegmentation/plotting/rscripts/config.R

```
################################################################################
# Paths configuration
################################################################################
path_job_params <- file.path(
  path.expand("~"), "math_master_ktu", "magistro_baigiamasis_darbas",
  "project", "btusersegmentation", "job_params.json")

dir_data <- file.path("..", "data")

dir_report <- file.path(
  path.expand("~"), "math_master_ktu", "magistro_baigiamasis_darbas",
  "report")

dir_report_img <- file.path(dir_report, "img")

dir_report_img_bw <- file.path(dir_report_img, "bw")

dir_slides_img <- file.path(
  path.expand("~"), "math_master_ktu", "magistro_baigiamasis_darbas",
  "slides", "img")

dir_spark_results <- file.path(
  path.expand("~"), "math_master_ktu", "magistro_baigiamasis_darbas",
  "project", "results_btusersegmentation", "spark")

dir_bidmach_results <- file.path(
  path.expand("~"), "math_master_ktu", "magistro_baigiamasis_darbas",
  "project", "results_btusersegmentation", "bidmach")

################################################################################
# Results filtering configuration
```

```
32 ##############################################################################
33 .metrics <- c("precision", "recall", "fmeasure")
```

btusersegmentation/plotting/rscripts/data_info.R

```
1 n_ads <- 105745
```

btusersegmentation/plotting/rscripts/fn_fname_helpers.R

```
1  source("config.R")
2  require("jsonlite")
3
4  .run_cols <- c("prefix_spark_output", "prefix_bm", "mixins",
5                 "nIter", "nDatasetPass", "mbSize",
6                 "nCvFold", "isDone")
7  .run_job_cols <- c(.run_cols, "nSegment")
8
9  prefix <- function(prefix_str) {
10   ifelse(!(is.null(prefix_str)) && prefix_str != "",
11          sprintf("%s_", prefix_str),
12          "")
13 }
14
15 get_job_params <- function() {
16   job_params <- fromJSON(path_job_params)$ldaSegmentationRuns[, .run_job_cols]
17   job_params$mixins <- sapply(job_params$mixins,
18                               function(x) paste(x, collapse="-"))
19   job_params[job_params$isDone == TRUE, ]
20 }
21
22 get_runs <- function(job_params) {
23   runs <- unique(job_params[, .run_cols])
24   runs
25 }
26
27 # get_res_run_name <- function(run_info) {
28 #   sprintf(
29 #     "mxs%s_%sp%02d_mb%3dk_cvf%02d",
30 #     run_info$mixins,
31 #     ifelse(run_info$nIter == 5, "", sprintf("i%04d_", run_info$nIter)),
32 #     run_info$nDatasetPass,
33 #     run_info$mbSize / 1000, run_info$nCvFold)
34 # }
35
36 get_run_name <- function(run_info) {
37   sprintf(
```

```
38      "%smxs%s_i%04d_p%02d_mb%3dk_cvf%02d",
39      prefix(run_info$prefix_spark_output),
40      run_info$mixins,
41      run_info$nIter,
42      run_info$nDatasetPass,
43      run_info$mbSize / 1000, run_info$nCvFold)
44  }
45
46  get_bm_job_name <- function(job_info) {
47    sprintf(
48      "%smxs%s_i%04d_s%04d_p%02d_mb%3dk_cvf%02d",
49      prefix(job_info$prefix_bm),
50      job_info$mixins,
51      job_info$nIter,
52      job_info$nSegment,
53      job_info$nDatasetPass,
54      job_info$mbSize / 1000, job_info$nCvFold)
55  }
56
57  get_run_jobs <- function(run, job_params) {
58    run_jobs <- merge(job_params, run)
59    run_jobs <- run_jobs[order(run_jobs$nSegment), .run_job_cols]
60    run_jobs
61  }
62
63
64  # get_res_job_name <- function(job_info) {
65  #   if (job_info$prefix == "g1" &&
66  #         job_info$mixins %in% c("", "Perp") &&
67  #         job_info$nIter == 5 &&
68  #         job_info$mbSize == 100000 &&
69  #         job_info$nCvFold == 5) {
70  #     iter_part <- ""
71  #   } else if (job_info$nIter == 20) {
72  #     iter_part <- sprintf("i%04dd_", job_info$nIter)
73  #   } else {
74  #     iter_part <- sprintf("i%04d_", job_info$nIter)
75  #   }
76  #   sprintf(
77  #     "%smxs%s_%ss%04d_p%02d_mb%3dk_cvf%02d",
78  #     ifelse(job_info$prefix == "g1", "", sprintf("%s_", job_info$prefix)),
79  #     job_info$mixins,
80  #     iter_part,
81  #     job_info$nSegment, job_info$nDatasetPass,
82  #     job_info$mbSize / 1000, job_info$nCvFold)
83  # }
84
85  get_res_spark_job_name <- function(job_info) {
86    sprintf(
87      "%smxs%s_i%04d_s%04d_p%02d_mb%3dk_cvf%02d",
```

```
    prefix(job_info$prefix_spark_output),
    job_info$mixins,
    job_info$nIter,
    job_info$nSegment,
    job_info$nDatasetPass,
    job_info$mbSize / 1000, job_info$nCvFold)
}

get_res_bm_job_name <- function(job_info) {
  sprintf(
    "%smxs%s_i%04d_s%04d_p%02d_mb%3dk_cvf%02d",
    prefix(job_info$prefix_bm),
    job_info$mixins,
    job_info$nIter,
    job_info$nSegment,
    job_info$nDatasetPass,
    job_info$mbSize / 1000, job_info$nCvFold)
}

get_job_name <- function(job_info) {
  sprintf(
    "%smxs%s_i%04d_s%04d_p%02d_mb%3dk_cvf%02d",
    prefix(job_info$prefix_spark_output),
    job_info$mixins,
    job_info$nIter,
    job_info$nSegment,
    job_info$nDatasetPass,
    job_info$mbSize / 1000, job_info$nCvFold)
}

get_fname_aggr_metrics <- function(run_info) {
  filename <- paste0(get_run_name(run_info), "_aggr_metrics.csv")
  file.path(dir_data, filename)
}

get_fname_aggr_click_entropy <- function(run_info) {
  filename <- paste0(get_run_name(run_info), "_aggr_click_entropy.csv")
  file.path(dir_data, filename)
}

get_fname_aggr_beta_entropy <- function(run_info) {
  filename <- paste0(get_run_name(run_info), "_aggr_beta_entropy.csv")
  file.path(dir_data, filename)
}
```

btusersegmentation/plotting/rscripts/fn_gather_helpers.R

```
source("config.R")
source("fn_fname_helpers.R")
require("jsonlite")
require("Rmisc")

##############################################################################
# Spark - All Metrics By Best Segment [which is chosen by some metric]
##############################################################################
get_fname_spark_metric <- function(metric, job_info) {
  job_name <- get_res_spark_job_name(job_info)
  fname <- file.path(
    dir_spark_results, job_name, paste0(job_name,"_byad_", metric, ".txt"))
  fname
}

read_table_spark_metric <- function(metric, job_info) {
  cc1 = c(
    "integer", "integer", "numeric", "numeric", "numeric",
    "numeric", "numeric")
  cn1 = c(
    "ad_id", "segment_id", "delta_ctr", "precision", "recall",
    "fmeasure", "clickentropycomponent")

  cc2 = c(
    "integer", "integer",
    "integer", "integer", "numeric",
    "integer", "integer", "numeric",
    "numeric", "numeric", "numeric", "numeric", "numeric")
  cn2 = c(
    "ad_id", "segment_id",
    "n_click_ad", "n_impression_ad", "ctr_ad",
    "n_click_seg", "n_impression_seg", "ctr_seg",
    "delta_ctr", "precision", "recall", "fmeasure", "clickentropycomponent")

  cc3 = c(
    "integer", "integer",
    "integer", "integer",
    "integer", "integer", "numeric",
    "integer", "integer", "numeric",
    "numeric", "numeric", "numeric", "numeric", "numeric")
  cn3 = c(
    "ad_id", "segment_id",
    "n_user_ad", "n_user_seg",
    "n_click_ad", "n_impression_ad", "ctr_ad",
    "n_click_seg", "n_impression_seg", "ctr_seg",
    "delta_ctr", "precision", "recall", "fmeasure", "clickentropycomponent")

```

```
48
49    has_only_metrics <- (
50      job_info$prefix_spark_output == "g1" &&
51      job_info$mixins == "" &&
52      job_info$nIter == 5 &&
53      job_info$nDatasetPass %in% c(1,2) &&
54      job_info$mbSize == 100000 &&
55      job_info$nCvFold == 5)
56    if (has_only_metrics == TRUE) {
57      cc <- cc1
58      cn <- cn1
59    } else if (job_info$prefix_spark_output == "g2") {
60      cc <- cc2
61      cn <- cn2
62    }else {
63      cc <- cc3
64      cn <- cn3
65    }
66
67    fname <- get_fname_spark_metric(metric, job_info)
68    df <- read.table(fname, col.names=cn, colClasses=cc)
69    if (has_only_metrics) {
70      df$ctr_ad <- df$precision / (df$delta_ctr + 1)
71    }
72    df$fmeasure_ad <- 2 * df$ctr_ad / (df$ctr_ad + 1)
73    df
74  }
75
76  get_aggr_row_metrics <- function(metric, job_info) {
77    df <- read_table_spark_metric(metric, job_info)
78    ci_deltactr <- CI(df$delta_ctr)
79    ci_precision <- CI(df$precision)
80    ci_recall <- CI(df$recall)
81    ci_fmeasure <- CI(df$fmeasure)
82
83    ci_precision_ad <- CI(df$ctr_ad)
84    ci_fmeasure_ad <- CI(df$fmeasure_ad)
85
86    row <- data.frame(
87      mxs=job_info$mixins,
88      p=job_info$nDatasetPass,
89      cvf=job_info$nCvFold,
90      K=job_info$nSegment,
91      taken_by_metric=metric,
92      deltactr_cu=ci_deltactr["upper"],
93      deltactr_mean=ci_deltactr["mean"],
94      deltactr_cl=ci_deltactr["lower"],
95      precision_cu=ci_precision["upper"],
96      precision_mean=ci_precision["mean"],
97      precision_cl=ci_precision["lower"],
```

```
    recall_cu=ci_recall["upper"],
    recall_mean=ci_recall["mean"],
    recall_cl=ci_recall["lower"],
    fmeasure_cu=ci_fmeasure["upper"],
    fmeasure_mean=ci_fmeasure["mean"],
    fmeasure_cl=ci_fmeasure["lower"],
    precision_ad_cu=ci_precision_ad["upper"],
    precision_ad_mean=ci_precision_ad["mean"],
    precision_ad_cl=ci_precision_ad["lower"],
    fmeasure_ad_cu=ci_fmeasure_ad["upper"],
    fmeasure_ad_mean=ci_fmeasure_ad["mean"],
    fmeasure_ad_cl=ci_fmeasure_ad["lower"],
    row.names=NULL)
  row
}

################################################################################
# Spark - Mean Click Entropy
################################################################################
get_fname_click_entropy <- function(job_info) {
  job_name <- get_res_spark_job_name(job_info)
  fname <- file.path(
    dir_spark_results, job_name,
    paste0(job_name,"_byad_clickentropy.txt"))
  fname
}

read_table_click_entropy <- function(job_info) {
  cc = c("integer", "numeric")
  cn = c('ad_id', "click_entropy")
  fname <- get_fname_click_entropy(job_info)
  df <- read.table(fname, col.names=cn, colClasses=cc)
  df
}

get_aggr_row_click_entropy <- function(job_info) {
  df <- read_table_click_entropy(job_info)
  ci_click_entropy <- CI(df$click_entropy)

  row <- data.frame(
    mxs=job_info$mixins,
    p=job_info$nDatasetPass,
    cvf=job_info$nCvFold,
    K=job_info$nSegment,
    click_entropy_cu=ci_click_entropy["upper"],
    click_entropy_mean=ci_click_entropy["mean"],
    click_entropy_cl=ci_click_entropy["lower"],
    row.names=NULL)
  row
}
```

```
148
149 ##############################################################################
150 # Bidmach - Mean Beta Entropy
151 ##############################################################################
152 get_fname_beta_entropy <- function(job_info) {
153   job_name <- get_res_bm_job_name(job_info)
154   fname <- file.path(
155     dir_bidmach_results, job_name,
156     paste0(job_name,"_mean_entropy.txt"))
157   fname
158 }
159
160 read_table_beta_entropy <- function(job_info) {
161   cc <- c("numeric")
162   cn <- c("mean_beta_entropy")
163   fname <- get_fname_beta_entropy(job_info)
164   df <- read.table(fname, col.names=cn, colClasses=cc)
165   df
166 }
167
168 get_aggr_row_beta_entropy <- function(job_info) {
169   df <- read_table_beta_entropy(job_info)
170   ci_beta_entropy <- CI(df$mean_beta_entropy)
171
172   row <- data.frame(
173     mxs=job_info$mixins,
174     p=job_info$nDatasetPass,
175     cvf=job_info$nCvFold,
176     K=job_info$nSegment,
177     beta_entropy_cu=ci_beta_entropy["upper"],
178     beta_entropy_mean=ci_beta_entropy["mean"],
179     beta_entropy_cl=ci_beta_entropy["lower"],
180     row.names=NULL)
181   row
182 }
```

btusersegmentation/plotting/rscripts/fn_gather_tex_plots.R

```
1  source("fn_plot_aggrs.R")
2  source("fn_plot_ads_hist2d.R")
3
4  mk_tex_figurewidestn <- function(file_plot, caption_plot) {
5    sprintf("\\ktufigurewidestn{%s}{%s}", file_plot, caption_plot)
6  }
7
8  mk_tex_figuren <- function(file_plot, caption_plot, width) {
9    sprintf("\\ktufiguren{%s}{%s}{%s}", file_plot, width, caption_plot)
10 }
```

```
11
12  map_caption_metric_chosen <- list(
13    "precision"="tikslumą (P)",
14    "recall"="atkuriamumą (R)",
15    "fmeasure"="F-matą (F)")
16
17  map_tex_caption_penalty <- list(
18    "g2"=" be apribojimų",
19    "fmu"=", kai taikomas apribojimas pagal segmento vartotojų skaičių",
20    "fmc"=", kai taikomas apribojimas pagal atkuriamumą (R)")
21
22  map_tex_caption_aggr_metric <- list(
23    "deltactr"=paste0(
24      "Vidutinio santykinio CTR pokyčio ($\\Delta$) ",
25      "priklausomybė nuo segmentų kiekio $K$"),
26    "precision"=paste0(
27      "Vidutinio tikslumu (P) ",
28      "priklausomybė nuo segmentų kiekio $K$"),
29    "recall"=paste0(
30      "Vidutinio atkuriamumo (R) ",
31      "priklausomybė nuo segmentų kiekio $K$"),
32    "fmeasure"=paste0(
33      "Vidutinės F-mato reikšmės ",
34      "priklausomybė nuo segmentų kiekio $K$")
35    )
36
37  mk_tex_caption_metric_chosen <- function(metric, info) {
38    paste0("Geriausio segmento parinkimo pagal ",
39           get(metric, map_caption_metric_chosen), " rezultatai",
40           get(info$prefix_spark_output, map_tex_caption_penalty))
41  }
42
43  mk_tex_caption_aggr_metric <- function(metric, info) {
44    paste0(get(metric, map_tex_caption_aggr_metric),
45           get(info$prefix_spark_output, map_tex_caption_penalty), ".")
46  }
47
48  mk_tex_caption_mixins <- mk_title_run_mixins
49
50  mk_tex_caption_run <- function(run_info) {
51     paste0(""#,#"$N_p = ", run_info$nDatasetPass, "$, ",
52  #          "$N_{cv} = ", run_info$nCvFold, "$, ",
53  #          "$N_{ii} = ", run_info$nIter, "$",
54  #          mk_tex_caption_mixins(run_info)
55  )
56  }
57
58  mk_tex_caption_job <- function(job_info) {
59    paste0("$K = ", job_info$nSegment, "$.")
60  }
```

```
61
62  mk_tex_caption_click_entropy <- function(run_info) {
63    paste0("Vidutinė paspaudimų entropija.",#, ", ",
64           mk_tex_caption_run(run_info))
65  }
66
67  mk_tex_caption_beta_entropy <- function(run_info) {
68    paste0("Vidutinė segmentų žodžių entropija.", # ", ",
69           mk_tex_caption_run(run_info))
70  }
71
72  mk_tex_caption_ads_hist2d <- function(metric, job_info) {
73    paste0(mk_tex_caption_metric_chosen(metric, job_info), ", ",
74            mk_tex_caption_job(job_info))
75  }
76
77  get_fname_tex_plots_aggr <- function() {
78    file.path(dir_report, "appendix_plots_aggr.tex")
79  }
80
81  get_fname_tex_plots_ads_hist2d <- function() {
82    file.path(dir_report, "appendix_plots_ads_hist2d.tex")
83  }
```

btusersegmentation/plotting/rscripts/fn_plot_ads_hist2d.R

```
1   source("fn_plot_aggrs.R")
2   require("ggplot2")
3   require("gridExtra")
4   require("ggExtra")
5   require("MASS")
6   require("scales")
7
8   n_breaks <- 5
9
10  get_fname_plot_ads_hist2d <- function(metric, job_info) {
11    paste(get_job_name(job_info), "hist2d",metric, sep="_")
12  }
13
14  get_scale_ads_hist2d_bw <- function() {
15    scale_fill_gradientn(colours = colorRampPalette(c("white", "grey0"))(256))
16  }
17
18  get_scale_ads_hist2d_blues <- function() {
19    scale_fill_gradientn(colours = colorRampPalette(c("white", blues9))(256))
20  }
21
22  mk_ylab_metric_hist2d <- function(metric, by_metric, info) {
```

```
  bquote(.(mk_expr_metric_prf(metric, by_metric, info))%*%"100%")
}

get_plot_ads_hist2d_only <- function(
  df_ads_metrics, metric, job_info, is_bw=FALSE, is_for_slides=FALSE,
  n=100, div_h=1
) {
  scale_fill <- get_scale_ads_hist2d_blues()
  if (is_bw == TRUE) {
    scale_fill <- get_scale_ads_hist2d_bw()
  }
  # TODO: calculate h in tryCatch? and use instead:
  # ##stat_bin2d(bins=100) + guides(fill=FALSE) +
#   message(get_job_name(job_info), metric)
#   message("begin calc inside")
#   message("precision ", bandwidth.nrd(df_ads_metrics$precision))
#   message("recall ", bandwidth.nrd(df_ads_metrics$recall))
#   message("end calc inside")

  h_vec <- c(
    ifelse(var(df_ads_metrics$recall) > 0.02,
           bandwidth.nrd(df_ads_metrics$recall),
           0.01),
    ifelse(var(df_ads_metrics$precision) > 0.02,
           bandwidth.nrd(df_ads_metrics$precision),
           0.01))

  p <- ggplot(df_ads_metrics) + aes(x=recall, y=precision) +
    stat_density2d(
      geom="tile", aes(fill=..density..^0.25), contour=FALSE,
      h=h_vec/div_h,n=n) +
    scale_fill +
    theme_minimal() + theme(legend.position="none") +
    scale_x_continuous(breaks=pretty_breaks(n_breaks), labels=percent,
                       limits=c(-0.02, 1.02), expand=c(0.02, 0.02)) +
    scale_y_continuous(breaks=pretty_breaks(n_breaks), labels=percent,
                       limits=c(-0.02, 1.02), expand=c(0.02, 0.02)) +
    xlab(mk_ylab_metric_hist2d("recall", metric, job_info)) +
    ylab(mk_ylab_metric_hist2d("precision", metric, job_info))

  if (is_for_slides == TRUE) {
    font_family_title <- font_family_title_slides
    font_family <- font_family_slides
  } else {
    font_family_title <- font_family_title_report
    font_family <- font_family_report
  }
  p <- p +theme(plot.title=element_text(family=font_family_title, size=size_title)) +
    theme(axis.title=element_text(family=font_family, size=size_label)) +
    theme(axis.text=element_text(family=font_family, size=size_label)) +
```

```
    theme(legend.text=element_text(family=font_family, size=size_label)) +
    theme(legend.title=element_text(family=font_family_title, size=size_label))
  p
}

add_marginals <- function(plot) {
  g <- ggExtra::ggMarginal(plot, type="histogram", size=7, binwidth=0.02)
  g
}

mk_title_ads_hist2d <- function(metric, job_info) {
  bquote("Segmentas"~.(mk_expr_gbest(metric, job_info))~"parinktas"~"pagal"~
           .(get(metric, map_metrics_abbr))*","~
           .(mk_title_job(job_info)))
}

get_title_ads_hist2d <- function(metric, job_info, is_for_slides=FALSE) {
  font_family <- font_family_title_report
  if (is_for_slides == TRUE) {
    font_family <- font_family_title_slides
  }
  textGrob(mk_title_ads_hist2d(metric, job_info), gp=gpar(
    fontsize=size_title, fontfamily=font_family))
}

get_plot_ads_hist2d <- function(df_ads_metrics, metric, job_info, ...) {
  p <- get_plot_ads_hist2d_only(
    df_ads_metrics, metric, job_info, is_bw=FALSE, is_for_slides=FALSE, ...)
  g <- add_marginals(p)
  gg <- arrangeGrob(
    arrangeGrob(g, ncol=1, nrow=1),
#     main=get_title_ads_hist2d(metric, job_info, is_for_slides=FALSE),
    ncol=1, nrow=1)
  gg
}

get_plot_ads_hist2d_bw <- function(df_ads_metrics, metric, job_info, ...) {
  p <- get_plot_ads_hist2d_only(
    df_ads_metrics, metric, job_info, is_bw=TRUE, is_for_slides=FALSE, ...)
  g <- add_marginals(p)
  gg <- arrangeGrob(
    arrangeGrob(g, ncol=1, nrow=1),
#     main=get_title_ads_hist2d(metric, job_info, is_for_slides=FALSE),
    ncol=1, nrow=1)
  gg
}

get_plot_ads_hist2d_slides <- function(df_ads_metrics, metric, job_info, ...) {
  p <- get_plot_ads_hist2d_only(
    df_ads_metrics, metric, job_info, is_bw=FALSE, is_for_slides=TRUE, ...)
```

```
123   g <- add_marginals(p)
124   gg <- arrangeGrob(
125     arrangeGrob(g, ncol=1, nrow=1),
126 #     main=get_title_ads_hist2d(metric, job_info, is_for_slides=TRUE),
127     ncol=1, nrow=1)
128   gg
129 }
130 
131 save_plot_hist2d <- function(plot, path_filename, is_embeded=FALSE, ...) {
132   ggsave(path_filename, plot, ...)
133   if (is_embeded == TRUE) {
134     embed_fonts(path_filename)
135   }
136 }
```

btusersegmentation/plotting/rscripts/fn_plot_aggrs.R

```
1  source("config_plot.R")
2  source("fn_plot_helpers.R")
3  
4  require("ggplot2")
5  require("scales")
6  require("ggplot2")
7  require("gridExtra")
8  require("Rmisc")
9  require("scales")
10 require("Rttf2pt1")
11 require("extrafontdb")
12 require("extrafont")
13 
14 Sys.setlocale(category="LC_CTYPE", locale="lt_LT.utf8")
15 ##############################################################################
16 # Metric plots
17 ##############################################################################
18 get_fname_plot_metric <- function(metric, run_info) {
19   paste(get_run_name(run_info), metric, sep="_")
20 }
21 
22 get_fname_plot_metric_y0 <- function(metric, run_info) {
23   paste(get_fname_plot_metric(metric, run_info), "y0",sep="_")
24 }
25 
26 map_metrics_abbr <- list("precision"="P", "recall"="R", "fmeasure"="F")
27 map_metrics_lt <- list(
28   "precision"="Tikslumas", "recall"="Atkuriamumas", "fmeasure"="F-matas")
29 map_gbest_penalty <- list("g2"="1", "fmu"="2", "fmc"="3")
30 
31 legend_name <- bquote(
```

```
#   "segmentas"~g["*"]^{"(%.%)"} * (a[i])~"parinktas"~"pagal")
  "Reklamos"~a[i]~"segmentas"~"parinktas"~"pagal"~tau*":")
legend_breaks <- c("precision", "recall", "fmeasure")
legend_labels <- c("P", "R", "F")

add_bw_metrics <- function(plot) {
  plot + scale_colour_grey(
    name=legend_name, breaks=legend_breaks, labels=legend_labels,
    start=grey_start, end=grey_end)
}


mk_gbest_penalty <- function(info) {
  map_gbest_penalty[[info$prefix_spark_output]]
}

mk_expr_gbest <- function(by_metric, info) {
  if (is.null(by_metric)) {
    by_metric <- bquote(tau)
  } else {
    by_metric <- get(by_metric, map_metrics_abbr)
  }
  bquote(g["*"]^{"("*.(by_metric)*",~.(mk_gbest_penalty(info))*")"}*
           "("*a[i]*")")
}

mk_expr_metric_prf <- function(metric, by_metric=NULL, info=NULL) {
  bquote(
    .(get(metric, map_metrics_abbr))*"("*a[i]*"|"*
      .(mk_expr_gbest(by_metric, info=info))*")")
}

mk_title_metric_prf <- function(metric, info) {
  bquote(.(get(metric, map_metrics_lt))~(.(get(metric, map_metrics_abbr))))
}

mk_ylab_metric_prf <- function(metric, info) {
  bquote(
    1/n * sum(.(mk_expr_metric_prf(metric, info=info)),i==1,n)%*%"100%")
}

mk_ylab_metric <- function(metric, info) {
  if (metric == "deltactr") {
    return(bquote(
      Delta^{"("*tau*","~.(mk_gbest_penalty(info))*")"} %*% "100%"))
  }
  mk_ylab_metric_prf(metric, info)
}

mk_title_metric <- function(metric, info) {
```

```
  if (metric == "deltactr") {
    return(bquote("Vid."~"santykinis"~"CTR"~"pokytis"~(Delta)))
  }
  mk_title_metric_prf(metric)
}

mk_title_metric_full <- function(metric, info) {
  bquote(.(mk_title_metric(metric, info))*","~.(mk_title_run(info)))
}

get_plot_base <- function(df_metrics, y_from=NA) {
  p_base <- ggplot(data=df_metrics) +
    scale_x_continuous(
      breaks=pretty_breaks(nxbreaks),
      limits=c(0, plot_aggr_x_end_limit)) +
    scale_y_continuous(
      breaks=pretty_breaks(nybreaks),
      limits=c(y_from, NA), labels=percent) +
    xlab(expression(K)) +
    theme_bw() +
    guides(alpha=FALSE) +
    theme(legend.position = "top") +
    scale_colour_discrete(
      name=legend_name, breaks=legend_breaks, labels=legend_labels)
  p_base
}

get_plot_metric <- function(plot_base, metric, run_info) {
  plot_metric <- plot_base +
    geom_line(
      mapping=aes_string(
        x="K", y=paste0(metric, "_mean"), color="taken_by_metric"),
      size=size_line) +
    geom_ribbon(
      mapping=aes_string(
        x="K", ymin=paste0(metric, "_cl"), ymax=paste0(metric, "_cu"),
        color="taken_by_metric"),
      alpha=ribbon_alpha) +
    geom_point(
      mapping=aes_string(
        x="K", y=paste0(metric, "_mean"), group="taken_by_metric"),
      size=size_point, alpha=point_alpha) +
#     ggtitle(mk_title_metric_full(metric, run_info)) +
    xlab(expression(K)) +
    ylab(mk_ylab_metric(metric, info=run_info))

  if (metric %in% c("precision", "fmeasure")) {
    plot_metric <- plot_metric +
      geom_ribbon(
        mapping=aes_string(
```

```
          x="K",
          ymin=paste0(metric, "_ad_cl"),
          ymax=paste0(metric, "_ad_cu")),
        alpha=ribbon_alpha) +
      geom_line(
        mapping=aes_string(
          x="K", y=paste0(metric, "_ad_mean")),
        size=size_line)
  }
  plot_metric
}

################################################################################
# click entropy plot
################################################################################
get_fname_plot_click_entropy <- function(run_info) {
  paste0(get_run_name(run_info), "_click_entropy")
}

mk_expr_click_entropy_component <- function() {
  bquote(P(g[k]*"|"*a[i]) == frac(1, sum(varphi(u[ij]), j==1, m[i]))%*%
           sum(delta(u[ij]), u[ij] %in% g[k](U[i])))
}

mk_expr_click_entropy <- function() {
  bquote(-sum(P(g[k]*"|"*a[i])*log[2]*P(g[k]*"|"*a[i]), k==1, K))
}

mk_title_click_entropy <- function() {
  bquote(
    "Vid."~"pasp."~"entr."*","~"kur"~
      .(mk_expr_click_entropy_component()))
}

mk_title_click_entropy_full <- function(run_info) {
  bquote(.(mk_title_click_entropy())*","~.(mk_title_run(run_info)))
}

mk_ylab_click_entropy <- function() {
#   bquote(frac(1, .(n_ads))*sum(entr[i], i==1, .(n_ads)))
  #bquote(frac(1, n)*sum(entr[i], i==1, n))
  bquote(1 / n * sum(bgroup("(",.(mk_expr_click_entropy()),")"), i==1, n))
}


get_plot_click_entropy <- function(df_click_entropy, run_info) {
  df_max_enp <- data.frame(
    K=df_click_entropy$K, max_enp=log(df_click_entropy$K, 2))
  p_entr <- ggplot() +
    geom_line(data=df_max_enp, aes(x=K, y=max_enp), color="red") +
```

```
182     geom_line(
183       data=df_click_entropy, aes(x=K, y=click_entropy_mean),
184       size=size_line) +
185     geom_ribbon(
186       data=df_click_entropy,
187       aes(x=K, ymin=click_entropy_cl, ymax=click_entropy_cu),
188       alpha=ribbon_alpha) +
189     geom_point(
190       data=df_click_entropy,
191       aes(x=K, y=click_entropy_mean),
192       size=size_point, alpha=point_alpha) +
193     theme_bw() +
194     scale_x_continuous(breaks=pretty_breaks(nxbreaks),
195                        limits=c(0, plot_aggr_x_end_limit)) +
196     scale_y_continuous(breaks=pretty_breaks(nybreaks)) +
197     xlab(bquote(K)) +
198     ylab(mk_ylab_click_entropy()) +
199     ggtitle(mk_title_click_entropy_full(run_info)) +
200     guides(color=FALSE, alpha=FALSE)
201   p_entr
202 }
203
204 ##############################################################################
205 # beta entropy plot
206 ##############################################################################
207 get_fname_plot_beta_entropy <- function(run_info) {
208   paste0(get_run_name(run_info), "_beta_entropy")
209 }
210
211 mk_title_beta_entropy <- function() {
212   bquote(
213     "Vid."~"segmentų"~"žodžių"~"entropija," ~
214       "entr" == 1 / K *
215       sum("-("*beta[k %.% ""]^T * ln * beta[k %.% ""]*")", k == 1, K))
216 }
217
218 mk_title_beta_entropy_full <- function(run_info) {
219   bquote(.(mk_title_beta_entropy())*","~.(mk_title_run(run_info)))
220 }
221
222 mk_ylab_beta_entropy <- function(run_info) {
223   n_cv <- run_info$nCvFold
224   bquote(1 / .(n_cv) * sum("entr"[i], i == 1, .(n_cv)))
225 }
226
227 get_plot_beta_entropy <- function(df_beta_entropy, run_info) {
228   p_entr <- ggplot() +
229     geom_line(
230       data=df_beta_entropy, aes(x=K, y=beta_entropy_mean),
231       size=size_line) +
```

```
    geom_ribbon(
      data=df_beta_entropy,
      aes(x=K, ymin=beta_entropy_cl, ymax=beta_entropy_cu),
      alpha=ribbon_alpha) +
    geom_point(
      data=df_beta_entropy,
      aes(x=K, y=beta_entropy_mean),
      size=size_point, alpha=point_alpha) +
    theme_bw() +
    scale_x_continuous(breaks=pretty_breaks(nxbreaks),
                       limits=c(0, plot_aggr_x_end_limit)) +
    scale_y_continuous(breaks=pretty_breaks(nybreaks)) +
    xlab(bquote(K)) +
    ylab(mk_ylab_beta_entropy(run_info)) +
    ggtitle(mk_title_beta_entropy_full(run_info)) +
    guides(color=FALSE, alpha=FALSE)
  p_entr
}
```

btusersegmentation/plotting/rscripts/fn_plot_helpers.R

```
source("config.R")
require("stringr")

mk_title_run_mixins <- function(run_info) {
  if (run_info$mixins == "") {
    return("")
  }
  paste0(", b.: ", str_replace_all(run_info$mixins, "-", ", "))
}

# mk_gbest_part <- function(run_info) {
#   if (run_info$gbest == "2") {
#     return("")
#   }
#   paste0(", g = ", run_info$gbest)
# }

mk_title_run <- function(run_info) {
  bquote(N[p]==.(run_info$nDatasetPass)*","~
           N[cv]==.(run_info$nCvFold)*","~
           N[ii] == .(run_info$nIter)*
#            .(mk_gbest_part(run_info))*
           .(mk_title_run_mixins(run_info)))
}

mk_title_job <- function(job_info) {
  bquote(K == .(job_info$nSegment)*","~.(mk_title_run(job_info)))
```

```
28 }
29
30 # from R package Hmisc -- not loading it for only one function
31 capitalize <- function(string) {
32   capped <- grep("^[^A-Z]*$", string, perl = TRUE)
33   substr(string[capped], 1, 1) <- toupper(substr(string[capped], 1, 1))
34   string
35 }
36
37 add_fonts <- function(plot, font_family, font_family_title) {
38   plot +
39     theme(plot.title=element_text(family=font_family_title, size=size_title)) +
40     theme(axis.title=element_text(family=font_family, size=size_label)) +
41     theme(axis.text=element_text(family=font_family, size=size_label)) +
42     theme(legend.text=element_text(family=font_family, size=size_label)) +
43     theme(legend.title=element_text(family=font_family_title, size=size_label))
44 }
45
46 add_fonts_slides <- function(plot) {
47   add_fonts(plot, font_family_slides, font_family_title_slides)
48 }
49
50 add_fonts_report <- function(plot) {
51   add_fonts(plot, font_family_report, font_family_title_report)
52 }
53
54 mk_fname_plot_pdf <- function(fname) {
55   paste0(fname, ".pdf")
56 }
57
58 mk_fname_plot_png <- function(fname) {
59   paste0(fname, ".png")
60 }
61
62 mk_fname_plot_report <- function(fname) {
63   file.path(dir_report_img, mk_fname_plot_pdf(fname))
64 }
65
66 mk_fname_plot_report_bw <- function(fname) {
67   file.path(dir_report_img_bw, mk_fname_plot_pdf(fname))
68 }
69
70 mk_fname_plot_slides_pdf <- function(fname) {
71   file.path(dir_slides_img, mk_fname_plot_pdf(fname))
72 }
73
74 mk_fname_plot_slides_png <- function(fname) {
75   file.path(dir_slides_img, mk_fname_plot_png(fname))
76 }
77
```

```
78  add_bw <- function(plot) {
79    plot +
80      scale_color_grey(start = grey_start, end = grey_end) +
81      scale_fill_grey(start = grey_start, end = grey_end)
82  }
83
84  save_plot <- function(
85    plot, filename, fn_add_bw=add_bw,
86    is_embeded=FALSE, ...
87  ) {
88    # save to report img
89    plot_report <- add_fonts_report(plot)
90    fname_report <- mk_fname_plot_report(filename)
91    ggsave(fname_report, plot_report, ...)
92
93    # save to report img/bw
94    plot_report_bw <- fn_add_bw(plot_report)
95    fname_report_bw <- mk_fname_plot_report_bw(filename)
96    ggsave(fname_report_bw, plot_report_bw, ...)
97
98    # save to slides as pdf
99    plot_slides <- add_fonts_slides(plot)
100   fname_slides_pdf <- mk_fname_plot_slides_pdf(filename)
101   ggsave(fname_slides_pdf, plot_slides, ...)
102
103   # save to slides as png
104   ggsave(mk_fname_plot_slides_png(filename), plot_slides, ...)
105   if (is_embeded == TRUE) {
106     embed_fonts(fname_report)
107     embed_fonts(fname_report_bw)
108     embed_fonts(fname_slides_pdf)
109   }
110 }
```

btusersegmentation/plotting/rscripts/gather_results_bidmach.R

```
1  source("clear_env.R")
2  source("config.R")
3  source("fn_gather_helpers.R")
4  require('Rmisc')
5
6  job_params <- get_job_params()
7  runs <- get_runs(job_params)
8
9  for (i in seq_len(nrow(runs))) {
10   run_info <- runs[i, ]
11   run_jobs <- get_run_jobs(run_info, job_params)
12
```

```
  df_aggr_beta_entropy <- data.frame()
  for (j in seq_len(nrow(run_jobs))) {
    job_info <- run_jobs[j, ]

    row <- get_aggr_row_beta_entropy(job_info)
    df_aggr_beta_entropy <- rbind(df_aggr_beta_entropy, row)
  }
  fname_aggr_beta_entropy <- get_fname_aggr_beta_entropy(run_info)
  write.csv(df_aggr_beta_entropy, fname_aggr_beta_entropy, row.names=FALSE)
}
```

btusersegmentation/plotting/rscripts/gather_results_spark.R

```
source("clear_env.R")
source("config.R")
source("fn_fname_helpers.R")
source("fn_gather_helpers.R")
require('Rmisc')

job_params <- get_job_params()
runs <- get_runs(job_params)

filter_prefix <- c("fmrsd")
filter_mixins <- c("")
filter_nIter <- c(5)
filter_nDatasetPass <- c(1)

is_filtered_prefix <- TRUE
is_filtered_mixins <- TRUE
is_filtered_nIter <- TRUE
is_filtered_nDatasetPass <- TRUE

runs <- runs[
  (!is_filtered_prefix) | (runs$prefix_spark_output %in% filter_prefix) &
    (!is_filtered_mixins | (runs$mixins %in% filter_mixins)) &
    (!is_filtered_nIter | (runs$nIter %in% filter_nIter)) &
    (!is_filtered_nDatasetPass |
       (runs$nDatasetPass %in% filter_nDatasetPass)), ]

for (i in seq_len(nrow(runs))) {
  run_info <- runs[i, ]
  run_jobs <- get_run_jobs(run_info, job_params)

  df_aggr_metrics <- data.frame()
  df_aggr_click_entropy <- data.frame()
  for (j in seq_len(nrow(run_jobs))) {
    job_info <- run_jobs[j, ]

```

```
36      for (metric in .metrics) {
37        row_metrics <- get_aggr_row_metrics(metric, job_info)
38        df_aggr_metrics <- rbind(df_aggr_metrics, row_metrics)
39      }
40      row_click_entropy <- get_aggr_row_click_entropy(job_info)
41      df_aggr_click_entropy <- rbind(df_aggr_click_entropy, row_click_entropy)
42    }
43    fname_aggr_metrics <- get_fname_aggr_metrics(run_info)
44    write.csv(df_aggr_metrics, fname_aggr_metrics, row.names=FALSE)
45
46    fname_aggr_click_entropy <- get_fname_aggr_click_entropy(run_info)
47    write.csv(df_aggr_click_entropy, fname_aggr_click_entropy,
48              row.names=FALSE)
49  }
```

btusersegmentation/plotting/rscripts/gather_self_source.R

```
1   source("fn_fname_helpers.R")
2   require("stringr")
3
4   mk_tex_srcr <- function(fcaption, fpath) {
5     sprintf("\\ktusrcr{%s}{%s}{R}", gsub("_", "\\\\_", fcaption), fpath)
6   }
7
8   here <- getwd()
9   rscript_fnames <- list.files()
10  tex_lines <- as.character(lapply(
11    rscript_fnames,
12    function(x) {
13      mk_tex_srcr(file.path("btusersegmentation", "plotting", "rscripts", x),
14                  file.path(here, x))
15    }))
16
17
18  printer <- file(file.path(dir_report, "appendix_source_r.tex"),"w")
19  writeLines(tex_lines, con=printer, sep="\n\n")
20  close(printer)
```

btusersegmentation/plotting/rscripts/gather_tex_plot_ads_hist2d.R

```
1   source("clear_env.R")
2   source("fn_plot_aggrs.R")
3   source("fn_plot_ads_hist2d.R")
4   source("config.R")
5   source("config_plot.R")
6   source("fn_fname_helpers.R")
```

```
source("fn_gather_tex_plots.R")

job_params <- get_job_params()
runs <- get_runs(job_params)

tex_lines <- c()
for (i in seq_len(nrow(runs))) {
  run_info <- runs[i, ]
  run_jobs <- get_run_jobs(run_info, job_params)

  for (j in seq_len(nrow(run_jobs))) {
    job_info <- run_jobs[j, ]

    for (metric in .metrics) {
      fname_plot_ads_hist2d <- get_fname_plot_ads_hist2d(metric, job_info)
      file_plot_plot_ads_hist2d_bw <- mk_fname_plot_report_bw(
        fname_plot_ads_hist2d)
      caption_plot_ads_hist2d <- mk_tex_caption_ads_hist2d(metric, job_info)
      tex_plot_ads_hist2d <- mk_tex_figuren(
        file_plot_plot_ads_hist2d_bw, caption_plot_ads_hist2d,
        sprintf("%dcm", 10))
      tex_lines <- c(tex_lines, tex_plot_ads_hist2d)
    }
  }
}

printer <- file(get_fname_tex_plots_ads_hist2d(),"w")
writeLines(tex_lines, con=printer, sep="\n\n")
close(printer)
```

btusersegmentation/plotting/rscripts/gather_tex_plot_aggr.R

```
source("clear_env.R")
source("fn_plot_aggrs.R")
source("config.R")
source("fn_fname_helpers.R")
source("fn_gather_tex_plots.R")

job_params <- get_job_params()
runs <- get_runs(job_params)

tex_lines <- c()
for (i in seq_len(nrow(runs))) {
  run_info <- runs[i, ]
  run_jobs <- get_run_jobs(run_info, job_params)

  for (metric in c("deltactr", .metrics)) {
    fname_plot_metric <- get_fname_plot_metric(metric, run_info)
```

```
    file_plot_metric_bw <- mk_fname_plot_report_bw(fname_plot_metric)
    caption_plot_metric <- mk_tex_caption_aggr_metric(metric, run_info)
    tex_plot_metric <- mk_tex_figuren(
      file_plot_metric_bw, caption_plot_metric, sprintf("%dcm", 15))
    tex_lines <- c(tex_lines, tex_plot_metric)

    fname_plot_metric_y0 <- get_fname_plot_metric_y0(metric, run_info)
    file_plot_metric_y0_bw <- mk_fname_plot_report_bw(fname_plot_metric_y0)
    caption_plot_metric <- mk_tex_caption_aggr_metric(metric, run_info)
    tex_plot_metric_y0 <- mk_tex_figuren(
      file_plot_metric_y0_bw, caption_plot_metric, sprintf("%dcm", 15))
    tex_lines <- c(tex_lines, tex_plot_metric_y0)
  }

  fname_plot_click_entropy <- get_fname_plot_click_entropy(run_info)
  file_plot_click_entropy <- mk_fname_plot_report_bw(fname_plot_click_entropy)
  caption_click_entropy <- mk_tex_caption_click_entropy(run_info)
  tex_plot_click_entropy <- mk_tex_figuren(
    file_plot_click_entropy, caption_click_entropy, sprintf("%dcm", 15))
  tex_lines <- c(tex_lines, tex_plot_click_entropy)

  fname_plot_beta_entropy <- get_fname_plot_beta_entropy(run_info)
  file_plot_beta_entropy <- mk_fname_plot_report_bw(fname_plot_beta_entropy)
  caption_beta_entropy <- mk_tex_caption_beta_entropy(run_info)
  tex_plot_beta_entropy <- mk_tex_figuren(
    file_plot_beta_entropy, caption_beta_entropy, sprintf("%dcm", 15))
  tex_lines <- c(tex_lines, tex_plot_beta_entropy)
}

printer <- file(get_fname_tex_plots_aggr(),"w")
writeLines(tex_lines, con=printer, sep="\n\n")
close(printer)
```

btusersegmentation/plotting/rscripts/plot_ads_hist2d.R

```
source("clear_env.R")
source("config.R")
source("config_plot.R")
source("fn_gather_helpers.R")
source("fn_plot_helpers.R")
source("fn_plot_ads_hist2d.R")

require('Rmisc')

div_h <- 2

filter_prefix <- "g2"
filter_mixins <- c("")
```

```
filter_nIter <- c(5)
filter_nSegment <- c(10)
filter_nDatasetPass <- c(1)
plotted_metrics <- c("fmeasure")

is_filtered_prefix <- FALSE
is_filtered_mixins <- TRUE
is_filtered_nIter <- TRUE
is_filtered_nSegment <- FALSE
is_filtered_nDatasetPass <- TRUE

job_params <- get_job_params()
runs <- get_runs(job_params)

for (i in seq_len(nrow(runs))) {
  run_info <- runs[i, ]
  run_jobs <- get_run_jobs(run_info, job_params)

  run_jobs <- run_jobs[
    (!is_filtered_prefix) | (run_jobs$prefix_spark_output %in% filter_prefix) &
      (!is_filtered_mixins | (run_jobs$mixins %in% filter_mixins)) &
      (!is_filtered_nIter | (run_jobs$nIter %in% filter_nIter)) &
      (!is_filtered_nSegment | (run_jobs$nSegment %in% filter_nSegment)) &
      (!is_filtered_nDatasetPass |
         (run_jobs$nDatasetPass %in% filter_nDatasetPass)), ]

  df_aggr_metrics <- data.frame()
  df_aggr_click_entropy <- data.frame()
  for (j in seq_len(nrow(run_jobs))) {
    job_info <- run_jobs[j, ]

    for (metric in plotted_metrics) {
      df_ads_metrics <- read_table_spark_metric(metric, job_info)

      p <- get_plot_ads_hist2d(
        df_ads_metrics, metric, job_info, div_h=div_h)
      p_bw <- get_plot_ads_hist2d_bw(
        df_ads_metrics, metric, job_info, div_h=div_h)
      p_slides <- get_plot_ads_hist2d_slides(
        df_ads_metrics, metric, job_info, div_h=div_h)

      name <- get_fname_plot_ads_hist2d(metric, job_info)
      fname <- mk_fname_plot_report(name)
      fname_bw <- mk_fname_plot_report_bw(name)
      fname_slides <- mk_fname_plot_slides_pdf(name)
      fname_slides_png <- mk_fname_plot_slides_png(name)

      save_plot_hist2d(p, fname,
                       units="cm",
                       scale=plot_ads_hist2d_scale,
```

```
                        width=plot_ads_hist2d_width,
                        height=plot_ads_hist2d_height,
                        is_embeded=has_plot_embed_font)
      save_plot_hist2d(p_bw, fname_bw,
                        units="cm",
                        scale=plot_ads_hist2d_scale,
                        width=plot_ads_hist2d_width,
                        height=plot_ads_hist2d_height,
                        is_embeded=has_plot_embed_font)
      save_plot_hist2d(p_slides, fname_slides,
                        units="cm",
                        scale=plot_ads_hist2d_scale,
                        width=plot_ads_hist2d_width,
                        height=plot_ads_hist2d_height,
                        is_embeded=has_plot_embed_font)
      save_plot_hist2d(p_slides, fname_slides_png,
                        units="cm",
                        scale=plot_ads_hist2d_scale,
                        width=plot_ads_hist2d_width,
                        height=plot_ads_hist2d_height,
                        is_embeded=FALSE)

    }
  }
}
```

btusersegmentation/plotting/rscripts/plot_aggrs.R

```
source("clear_env.R")
source("fn_plot_aggrs.R")
source("config.R")
source("fn_fname_helpers.R")

require("ggplot2")
require("gridExtra")
require("Rmisc")
require("scales")
Sys.setlocale(category="LC_CTYPE", locale="lt_LT.utf8")
require("Rttf2pt1")
require("extrafontdb")
require("extrafont")

loadfonts()


job_params <- get_job_params()
runs <- get_runs(job_params)

```

```
for (i in seq_len(nrow(runs))) {
  run_info <- runs[i, ]

  # metrics
  fname_aggr_metrics <- get_fname_aggr_metrics(run_info)
  df_aggr_metrics <- read.csv(fname_aggr_metrics)
  for (metric in c("deltactr", .metrics)) {
    p_base <- get_plot_base(df_aggr_metrics)
    p_metric <- get_plot_metric(p_base, metric, run_info)
    fname_plot_metric <- get_fname_plot_metric(metric, run_info)
    save_plot(p_metric, fname_plot_metric, fn_add_bw=add_bw_metrics,
              units="cm", scale=plot_aggr_scale,
              is_embeded=has_plot_embed_font,
              width=plot_aggr_width, height=plot_aggr_height)

#     p_base_y0 <- get_plot_base(df_aggr_metrics, y_from=0)
#     p_metric_y0 <- get_plot_metric(p_base_y0, metric, run_info)
#     fname_plot_metric_y0 <- get_fname_plot_metric_y0(metric, run_info)
#     save_plot(p_metric_y0, fname_plot_metric_y0, fn_add_bw=add_bw_metrics,
#               units="cm", scale=plot_aggr_scale,
#               is_embeded=has_plot_embed_font,
#               width=plot_aggr_width, height=plot_aggr_height)
  }

#   # click entropy
#   fname_aggr_click_entropy <- get_fname_aggr_click_entropy(run_info)
#   df_aggr_click_entropy <- read.csv(fname_aggr_click_entropy)
#   p_click_entropy <- get_plot_click_entropy(df_aggr_click_entropy, run_info)
#   fname_plot_click_entropy <- get_fname_plot_click_entropy(run_info)
#   save_plot(p_click_entropy, fname_plot_click_entropy,
#             units="cm", scale=plot_aggr_scale, is_embeded=has_plot_embed_font,
#             width=plot_aggr_width, height=plot_aggr_height)
#
#   # beta entropy
#   fname_aggr_beta_entropy <- get_fname_aggr_beta_entropy(run_info)
#   df_aggr_beta_entropy <- read.csv(fname_aggr_beta_entropy)
#   p_beta_entropy <- get_plot_beta_entropy(df_aggr_beta_entropy, run_info)
#   fname_plot_beta_entropy <- get_fname_plot_beta_entropy(run_info)
#   save_plot(p_beta_entropy, fname_plot_beta_entropy,
#             units="cm", scale=plot_aggr_scale, is_embeded=has_plot_embed_font,
#             width=plot_aggr_width, height=plot_aggr_height)
}
```

btusersegmentation/plotting/rscripts/setup.R

```
required_cran_packages <- c(
  "MASS", "Rmisc", "ggplot2", "gridExtra", "ggExtra", "scales", "Rttf2pt1",
  "extrafont", "extrafontdb", "jsonlite")
```

```
4  for (package in required_cran_packages) {
5    if (require(package) == FALSE) {
6      install.packages(package)
7    }
8  }
9
10 font_import()
```

btusersegmentation/plotting/rscripts/static_ad_revenue_growth.R

```
1  source("clear_env.R")
2  source("fn_plot_aggrs.R")
3  source("config.R")
4  source("fn_fname_helpers.R")
5  source("fn_plot_helpers.R")
6
7  require("ggplot2")
8  require("gridExtra")
9  require("Rmisc")
10 require("scales")
11 Sys.setlocale(category="LC_CTYPE", locale="lt_LT.utf8")
12 require("Rttf2pt1")
13 require("extrafontdb")
14 require("extrafont")
15
16 loadfonts()
17
18 plot_ad_revenue_by_year <- function(df, ylab, title=NULL) {
19   p <- ggplot(df, aes(x=year, y=spend)) +
20     geom_bar(stat="identity") +
21     scale_x_discrete(breaks=df$year) +
22     xlab("Metai") +
23     ylab(ylab) +
24     theme_bw()
25   if (!is.null(title)) {
26     p <- p + ggtitle(title)
27   }
28   p
29 }
30
31 df_europe <- read.csv(
32   "../data/ad-spend-2014-europe.csv",
33   colClasses=c("factor", "numeric"))
34
35 df_us <- read.csv(
36   "../data/ad-spend-2014-us.csv",
37   colClasses=c("factor", "numeric"))
38
```

```
39
40 p_eu <- plot_ad_revenue_by_year(df_europe, "Pajamos, mlrd. EUR")
41
42 p_us <- plot_ad_revenue_by_year(df_us, "Pajamos, mlrd. USD")
43
44 fname_p_eu <- file.path("static", "ad_revenue_eu")
45 fname_p_us <- file.path("static", "ad_revenue_us")
46
47 plot_ad_revenue_width <- 14
48 plot_ad_revenue_height <- 11
49 plot_ad_revenue_scale <- 1
50
51 save_plot(
52   p_eu, fname_p_eu,
53   is_embeded=TRUE,
54   units="cm",
55   width=plot_ad_revenue_width,
56   height=plot_ad_revenue_height,
57   scale=plot_ad_revenue_scale)
58
59 save_plot(
60   p_us, fname_p_us,
61   is_embeded=TRUE,
62   units="cm",
63   width=plot_ad_revenue_width,
64   height=plot_ad_revenue_height,
65           scale=plot_ad_revenue_scale)
```

btusersegmentation/plotting/rscripts/util.R

```
1 clear <- function() rm(list=ls(all.names=TRUE))
```