Praktiskais darbs - Eirovīzija

Pēteris Račinskis pr20015

05/01/22

Piebilde: praktiskā darba nosacījumos nav norādīts sagaidāmais atskaites apjoms. Kursa aprakstā tas tiek saukts par kursa darbu, kas liecina, ka tiek gaidīts aptuveni 15-30lpp. garš dokuments ar atbilstošu ieguldītā darba un informācijas apjomu. Atvainojos, ja iesūtītā atskaite ir ievērojami plānāka, nekā tiek gaidīts - taču kaut kāds darbs ir padarīts, vairākas dienas laika ir iztērētas, personīgais sesijas grafiks liek virzīties tālāk pie citiem uzdevumiem, un, ja pastāv kaut neliela iespēja uzlabot gala vērtējumu, uzskatu, ka ir vērts iesūtīt pat potenciāli nepietiekamu rezultātu.

Visi koda faili, attēli, datu korpusi: GitHub - DIA_PD

1. Ievads

1.1. Uzdevums

Datizraces uzdevumi var būt dažādi. Viens no veidiem, kā tie var radikāli atšķirties pēc būtības, ir pirms pētījuma sākšanas pastāvošā skaidrība par rezultāta formu. Ja kādreiz sākam ar datu kopu, par ko nekas nav zināms, un mēģinām gūt vispārīgus priekštatus par starp tās elementiem pastāvošajām sakarībām, citreiz jau no paša sākuma ir laba izpratne par to, ko vēlamies sasniegt, kādi ir datu kopu veidojošie mehānismi un kādus skaitļus varētu redzēt iegūto aprakstošo modeļu parametros. Šis uzdevums visai pārliecinoši pieder otrajai kategorijai. Prasīts atbildēt uz ļoti specifisku jautājumu. Tāpēc tā vietā, lai sāktu ar ļoti vispārīgu datizraces metožu lietojumu, varam pielāgot vai izstrādāt darba rīkus tieši viena jautājuma atbildēšanai. Turklāt jautājums uzdots par datu kopas ģenerējošo procesu - eirovīzijas dziesmu konkursu - nevis par kādu konkrētu, specifiskā veidā strukturētu korpusu, kas ļauj patstāvīgi izvēlēties maksimāli piemērotu informācijas avotu, ierobežot definīcijas apgabalu pēc saviem ieskatiem, u.t.t.

Intuitīvi uzreiz rodas priekštats, kas domāts ar terminiem "kaimiņu būšana" un "objektīvāks novērtējums". Taču ar to nepietiek, lai iegūtu kaut kādu šo fenomenu skaitlisku izteiksmi. Nepieciešams definēt "objektīvu novērtējumu" un attiecīgi - novirzes no tāda. Viens veids, kā to darīt, varētu būt ieviest latentu dziesmu "popularitātes" mēru. Tādu var modelēt, iztēlojoties "demokrātisku" visu skatītāju balsošanu par, viņuprāt, labāko konkursa kārtas dalībnieku dziesmu:

$$i \in \{1, 2, ..., K\} = [K] \tag{1}$$

$$N_i - \text{balsis par dziesmu}; N - \text{balsis kopā} \tag{2}$$

$$q_i = \frac{N_i}{N}; \sum_{i \in [K]} q_i = 1 \tag{3}$$

$$s_i \in [10] \cup 0 - \text{punktu skaits (score)}; s_i \sim P(s_i = x \mid q_i, K) \tag{4}$$

$$q_i \leq q_j \Rightarrow s_i \leq s_j \tag{5}$$

$$s_i, s_j \neq 0 \Rightarrow s_i \neq s_j \tag{6}$$

kur $P(x \mid q_i, K)$ ir sadalījums, kas apraksta katra iespējamā diskrētā novērtējuma (punktu skaita) varbūtību, pieņemot, ka dziesmas "demokrātiskā" balsojuma varbūtība ir q_i. Šķiet, ka šis sadalījums varētu kaut kādas formas binomiālais, (vai arī kaut kas krietni sarežģītāks), taču tā precīzā analītiskā forma nav svarīga tālākiem aprēķiniem. Svarīgi piebilst, ka eirovīzijas vērtējumu sistēmā parasti punkti pieder kopai $[8] \cup \{0, 10, 12\}$, un

šīm skaitliskām vērtībām ir nozīme, rēķinot gala rezultātu (punkti tiek skaitīti kopā), taču katras dalībvalsts vērtējuma piešķiršanas procesā šiem skaitļiem ir tikai ordināla nozīme, t.i., $10 < 12$; $\forall i \in [8] : i < 10$. Tāpēc var pieņemt, ka $s_i \in [10]$ un vajadzības gadījumā izmantot pārveidojumu $\hat{s}_i = f(s_i); f(9) = 10; f(10) = 12; f(x \neq 9, 10) = x$.

Tad sagaidāmo punktu skaitu, ja balsojums notiek tikai vienreiz un sakrīt ar "objektīvo" novērtējumu (vienā valstī, visās kopā), var izteikt kā:

$$E[s_i^1] = \sum_{s \in [10]} P(s \mid q_i, K) * s \tag{7}$$

bet, ja balsojums tiek atkārots vairākas reizes un tiek skaitīta to svērto variantu summa (K' ir balsojošo dalībvalstu skaits, kas daudzreiz ir tāds pats, kā uzstājošos dalībvalstu skaits, bet ne vienmēr, jo nesenākos konkursos ieviesta pusfinālu sistēma un visas valstis var balsot finālā):

$$E[\hat{s}_i^{K'}] = \sum_{j \in [K']} \sum_{s \in [10]} P(s \mid q_i, K) * f(s) \tag{8}$$

Ja interesē nevis dziesmas rezultāts konkursa uzvarētāja noteikšanai, bet tās vidējā ordinālā pozīcija katras balsotājvalsts vērtējumā, pārveidojumu $f(x)$ var (varētu pat teikt, ka nepieciešams) atmest. Pilnīgi korekts novērtējums šis nav tāpat, jo pazudušas ir visas vērtības, kas nav starp 10 lielākajām:

$$E[s_i^{K'}] = \sum_{j \in [K']} \sum_{s \in [10]} P(s \mid q_i, K) * s = K' * E[s_i^1] \tag{9}$$

Kā redzams, rezultāts nav atkarīgs no katras dalībvalsts un ir "objektīvs". Ieviest nobīdes nacionālajos balsojumos varētu ar svariem:

$$E[s_i^{K'}] = \sum_{j \in [K']} E[s_i^1] * w_{ji}, \sum_{i \in [K]} w_{ji} = 1 \Rightarrow E[s_i^{K'}] = E[s_i^1] \tag{10}$$

un tad "neobjektivitāti" varētu potenciāli labot, lai atjaunotu sagaidāmo vērtību, reizinot svarus ar korekcijas koeficientiem, kas iegūti, dalot svarus ar vienmērīgam sadalījumam (pār vērtējamām dalībvalstīm, nevis balsojošajām) atbilstošajiem:

$$c_{ji} = \frac{w_K^0}{w_{ji}} = \frac{\frac{1}{K}}{w_{ji}} = \frac{1}{K * w_{ji}} \tag{11}$$

un visu kopā apkopot korekciju matricā:

$$C = \begin{bmatrix} c_{11} & \dots & c_{1K'} \\ \vdots & \ddots & \vdots \\ c_{K1} & \dots & c_{KK'} \end{bmatrix} \tag{12}$$

Lai šo korekciju matricu pielietotu rezultātu labošanai, punktu matricu (bez $f(s)$ pārveidojuma) pa elementiem reizina ar korekciju matricu:

$$S = \begin{bmatrix} s_{11} & \dots & s_{1K'} \\ \vdots & \ddots & \vdots \\ K1 & \dots & s_{KK'} \end{bmatrix} \tag{13}$$

$$S' = C \circ S \tag{14}$$

Kas tad īsti ir iegūts, un kā no tā aprēķināt konkursa rezultātu? Jāatceras, ka s_i ir punktu skaits, kas iegūts, pēc slēptā mainīgā q_i kārtojot konkursa dalībniekus un piešķirot punktus 10 labākajiem. Ir izdarīts pieņēmums, ka katra valsts vispirms ieguvusi šos punktu skaitus no vienādiem varbūtību sadalījumiem, tad pareizinājusi ar svariem. Koriģējot, atgriezta pēdējā darbība, un iegūti punktu skaiti, kādi tie būtu pirms šīs fiktīvās svēršanas operācijas, pēc fiktīvas demokrātiskas balsošans un punktu piešķiršanas.

Protams, ka realitātē process ir citāds: svari netiek pielietoti punktiem, tā vietā jau punktu skaitu sadalījums ir kropļots - precīzākus rezultātus varētu iegūt, rēķinot korekcijas saņemto balsu skaitiem, ja tie būtu zināmi (un nebūtu žūrijas komponentes, kas visu šo "demokrātijas" modeli padara par vienkārši nederīgu). Turklāt visas vērtības, kas nav bijušas augstākajā desmitniekā no datu kopas vienkārši ir izgrieztas (vienādas ar 0). Tāpēc jau uzreiz var pateikt, ka no matemātiska viedokļa, ar šādu matricu nav iespējams atjaunot slēpto q_i sadalījumu. Taču no statistikas kursa pagājušajā semestrī zināms, ka t.s. "rangu" metodes, kas strādā ar kārtas skaitļiem, nevis skaitliskām vērtībām tiešā veidā, parasti uzvedas vismaz virpsusēji līdzīgi nepārtrauktajām, un bieži vien ir algoritmiski vienkāršākas (pat ja kaut ko par tām pierādīt mēdz būt grūtāk), tāpēc var pastāvēt zināma cerība, ka pat matemātiski nekorektas un nepilnīgas, no datu korpusiem ar iztrūkumiem iegūtas korekcijas varētu darboties vismaz pareizajā virzienā.

Attiecīgi tiek izvirzīts sekojošs korekcijas modelis: tā kā iegūtas ir "atjaunotās" rangu sagaidāmās vērtības, tās drīkst vienkārši pārkārtot - piešķirt kārtas skaitļus no mazākās uz lielāko - un izdarīt korekciju $f(s)$, lai svērtu kārtas skaitļus summās starp balsojošajām valstīm.

$$S'' = S' \text{ rangos 1-10 (pārējie = 0), pa kolonnām} \tag{15}$$

$$\hat{S}'' = f(S'') \tag{16}$$

$$\hat{s}_i = \sum_{j \in [K']} \hat{s}_{ij} \tag{17}$$

Jāmin, ka šādi nav iespējams izšķirt tieši "kaimiņu būšanu" - ko varbūt gribētos rakstu-rot kā tīri etniskas tuvības vai ģeopolitisku interešu sakritības motivētu nobīdi balosjumu rezultātos. No citiem faktoriem, kas arī atšķiras valstu starpā - kulturālas noslieces, demogrāfiskie sadalījumi, konkursa popularitāte, u.t.t. - radušās nobīdes skaitliski izskatītos tāpat.

1.2. Datu kopas

Kā jau minēts iepriekš, uzdevums ir par fenomenu, nevis tā radītu konkrētu datu kopu. Tāpēc iespējams ne tikai brīvi pēc saviem ieskatiem pārveidot vienu datu kopu, bet apzināti meklēt un izvēlēties jau maksimāli atbilstoši noformētu. Nav arī dots stingrs uzstādījums, ka obligāti jāstrādā ar visu konkursa vēsturi. Laika gaitā ir notikušas daudzas noteikumu un organizatoriskas izmaiņas, kas var apgrūtināt dažādu periodu rezultātu salīdzināšanu.

Par datu kopu izvēlēta *Eurovision Song Contest Dataset* (pieejama *GitHub* repozitorijā), kur jau atrodams korpuss *votes.csv*. Tajā katrā rindā dots notikuma gads, atbilstošā stadija (fināls; pusfināls; pirmais vai otrais pusfināls gados, kad ir divi), vērtējošā valsts un punktus saņemošā valsts.

Pietiek vien atvērt *Wikipedia* rakstu par konkursa balsošanas kārtību, lai kristu nelielā panikā. Garākais (un, Latvijas iedzīvotājiem, interesantākais) posms ar samērā noturīgu balsošanas kārtību ir 1980-2015, tāpēc tālāk tieši ar to arī pārsvarā strādāts.

2. Metodes

2.1. Modeļi

Sadaļā 1.1. aprakstīts, kā varētu izskatīties korekcijas matrica, un radīta aptuvena nojausma par procesu, kas šādu matricu varētu ģenerēt, taču tā nav konstruktīva - svari w_{ji} *a priori* nav zināmi, tos nepieciešams noteikt empīriski. Līdz šim arī aplūkots tikai gadījums ar vienu q_i "kvalitātes vērtību" sadalījumu. Dažādās konkursa kārtās šie sadalījumi var radikāli atšķirties. Vienam konkursa etapam korekcijas varētu vienkārši algebriski izrēķināt, taču, ja vēlamies noteikt korekcijas matricu C noteikt globāli, jāveido modeļa šablons un jāapmāca.

Var krietni palauzīt galvu, domājot par veidiem, kā to izdarīt. Pirmā ideja, kas varētu rasties, varētu būt vienkārši ņemt punktus tiešā veidā no datu kopas un apmācīt klasifikatoru formā (vērtējošā valsts, vērtējamā valsts) $\rightarrow$ vērtējums (kur vērtējums vai teksta mainīgai, vai diskrētu vērtību vektors, kur 1 apzīmē konkrētu punktu skaitu). Lai iegūtu sagaidāmās vērtības novērtējumu, var izmantot daudzu klasifikatoru īpatnību - modeļa ģenerētais izejas vektors var tikt normalizēts uz varbūtību sadalījumu, nevis vienu konkrētu klasi. Šīs varbūtības tad var reizināt ar atbilstošajām punktu vērtībām, lai iegūtu matemātisko cerību. Problēma ar šādu pieeju ir tāda, ka bieži vien pieejami pavisam nedaudzi novērtējumi no vienas valsts uz otru. Par spīti šim trūkumiem, mēģinājumi uztrenēt klasifikatora šablonu datu kopai tika veikti, bet vienīgais, kas sniedza cilvēkam saprotamus rezultātus, bija neironu tīkls. Iegūtās sagaidāmās vērtības var tikt izmantotas kā distances starp dalībvalstīm vai par estimatoru, no kā tālāk rēķināt korekcijas. Citi izmēģinātie modeļu šabloni bija SVM un *NaiveBayes*, taču ar tiem semantiski saprotamas skaitliskas vērtības gūt neizdevās, un tiem atbilstošais kods ir izkomentēts.

Tā kā neviens no tipveida modeļu šabloniem nešķita ideāli piemērots tieši šim uzdevumam, pirmais algoritms, kas tika realizēts, bija paša rakstīts. Tā vietā, lai pakāpeniski optimizētu šablonu, iterējot pār datu kopas elementiem, datu kopu var mēģināt reducēt uz formu, kur rezultāts atrodams algebriski. Šajā konkrētajā gadījumā tas darīts, nosakot vidējās empīriskās vērtības piešķirto punktu skaitam katram valstu pārim katrā virzienā, pārveidojot tās par divdimensionālu varbūtību sadalījumu un pielīdzinot rezultātu vienmērīgajam sadalījumam.

To dara, atrodot divus korekcijas koeficientus: r_i, kas vienādo rindu varbūtību summas (savā ziņā kompensējot q_i) un c_{ij}, kas vienādo varbūtības koriģētās matricas kolonnās un reizē ir arī korekcijas matricas vērtības.

$$p_i = \sum_{j \in [K']} p_{ij} \tag{18}$$

$$p_i * r_i = \frac{1}{K} \tag{19}$$

$$r_i = \frac{1}{p_i * K} \tag{20}$$

$$p'_{ij} = p_{ij} * r_i \tag{21}$$

$$p'_{ij} * c_{ij} = \frac{\sum_{i \in [K]} p'_{ij}}{K} \tag{22}$$

$$c_{ij} = \frac{\sum_{i \in [K]} p'_{ij}}{p'_{ij} K} \tag{23}$$

$$w_{ij} = \frac{1}{c_{ij}K} \tag{24}$$

Praktiski tika konstatēts, ka, iespējams, kvalitatīvākus rezultātus sniedz korekcijas kvadrātsakne, par ko diskutēts pie rezultātiem:

$$c'_{ij} = \sqrt{c_{ij}} \tag{25}$$

$$S' = C' \circ S \tag{26}$$

Par šī algoritma matemātisko pareizību pārliecības nav nekādas - pirmais kompensācijas solis koriģē q_i, balstoties uz stipri kropļotiem vērtējumiem, un svari tad tiek rēķināti no šī nekorektā starprezultāta. Taču vismaz virspusēji šķiet, ka iegūie rezultāti varētu būt noderīgi.

Iespējams, ka varētu šo procesu pilnveidot, ieviešot iterāciju un cerot uz konverģenci, atkārtoti veicot korekcijas aprēķinu koriģētiem datiem, lai gan šādas funkcionalitātes ieviešanai laika nav pieticis. Galvenais šķērslis ir šaubas par to, kā pareizi kombinēt soļos iegūtās korekcijas matricas - vai tas vispār ir pamatojams. Taču zināms, ka līdzīgas metodes izmanto citur datizracē, lai cīnītos ar t.s. *distribution shift* jeb atšķirību starp modeļa ģenerētiem sadalījumiem un reāliem.

2.2. Datu priekšapstrāde

Klasifikatoru šablonu izmantošanai nepieciešams datu kopu "vektorizēt", t.i., diskrētas klases ieejas datu kopā izteikt kā vektorus ar vienu nenulles elementu. Atkarībā no konkrētā algoritma realizācijas, var būt nepieciešams rezultātu kolonnu vai nu izteikt kā tādu pašu vektoru, vai viendimensionālu sarakstu ar teksta elementiem. Šīs manipulācijas veic *Python* skripts *vectorize.py*, pieejams kopā ar visiem pirmkoda failiem un dažādiem datu apstrādes starpproduktiem projekta repozitorijā.

Lai veiktu aprēķinus pēc algebriskās metodes, jāveic smagnējāki pārveidojumi. Kopu ar kortežiem formā (t, i, j, s) nepieciešams izteikt kā tensoru ar elementiem s_{ijt}, kur t - konkursa etaps. Ja konkursa etapu dalībnieki un balsotāji vienmēr būtu tie paši, šis būtu triviāls uzdevums. Taču sarežģījumus rada fakts, ka neeksistējošas vērtības un nulles vērtības nav viens un tas pats. Ne katra dalībvalsts piedalās katrā etapā un ne katra dalībvalsts, kas balso, arī piedalās konkursā. Tāpēc neeksistējošas vērtības nepieciešams marķēt atsevišķi. Izvēlētais risinājums ir tās marķēt ar -1, un pēc tam datu apstrādē īpaši apstrādāt šādi marķētus ierakstus, kur nepieciešams. Vērtībām 10,12 tiek ieviesta jau iepriekš aprakstītā korekcija uz 9,10.

Šīs operācijas veiktas repozitorijā pieejamajā skriptā *datagen.py*. Principā priekšapstrādes solī iespējams arī veikt vidējo vērtību rēķināšanu, kas droši vien arī būtu ātrāk nekā faktiski realizētajā tensora ģenerēšanas pieejā, taču tīri praktiski apsvērumu vadībā tika izlemts visas netriviālās datu manipulācijas atstāt atsevišķi: darba autors daudz labāk pazīstams ar n-dimensionālu homogēnu datu bloku skaitļošanas un lineārās algebras bibliotēku *numpy* nekā ar 2-dimensionālu heterogēnu datu korpusu apstrādes bibliotēku *pandas*. Izstrādes procesā vieglāk iteratīvi izmainīt visu procesu, strādājot ar jau gatavu datu tensoru. Lai nebūtu katru reizi jāveic tensora veidošana, kas aizņem gandrīz 40 sekundes, starprezultāts tiek saglabāts failā. Datu pārveidošanu ir iespējams paātrināt daudzkārtīgi, taču tad jāizmanto cita darbību secība, ko grūtāk atkļūdot. Reizi dažās stundās zaudēt 40s šķiet mazāks zaudējums, nekā pavadīt vairākas stundas, pārrakstot jau pietiekami labi strādājošu kodu.

Šī ir arī laba vieta tekstā, kur norādīt, ka šādam datu formātam matrica ar vērtējošajām un vērtejamām valstīm vienmēr ir kvadrātiska, t.i., $K = K'$. Iztrūkstošās šķautnes ir atbilstoši marķētas ar vērtībām ārpus parastā definīcijas apgabala..

2.3. Modeļu aprēķins

Universālo klasifikatoru apmācībai izmantota bibliotēka *scikit-learn*, kas ļauj ar ļoti lakoniskām definīcijām pielietot dažādus visai jaudīgus datizraces algoritmus. Ja nav vēlmes vai nepieciešamības īpaši iedziļināties modeļu hiperparametru optimizācijā, šādu rīku izmantot ir vienkāršāk nekā daudz jaudīgākas bibliotēkas kā *tensorflow*. Kods atrodams skriptā *model-classifier.py*. Tā kā šī programma rakstīta vēlāk, tā vietām izmanto funkcijas no zemāk aprakstītās.

Algebriskā modeļa aprēķins tiek veikts citā *Python* skriptā - *model.py*. Pirms iespējams aprēķināt vidējās vērtības pār etapiem, nepieciešams noteikt katra balsojuma virziena (varētu teikt, orientēta grafa šķautnes) kopskaitu. To veic funkcija *coincidence_count*, kas atgriež matricu ar vērtībām formā n_{ij} - reižu skaits, kad j-tā valsts balsojusi konkursā, kurā uzstājas i-tā. Šo parametru izmanto funkcija *clear_dataset*, kas atsijā valstis pēc sliekšņa kritērija - ja maksimālais n_{ij} ir mazāks par slieksni, no matricas tiek izmestas i-tās rindas un i-tās kolonnas. Tādējādi tiek apkarota traucējoša parādība - valstīm, kas piedalījušās konkursā tikai dažas reizes, korekcijas koeficienti var būt krietni lielāki vai mazāki, nekā visām pārējām, jo lielo skaitļu likumam nav pieticis elementu, lai izlīdzinātu iegūto punktu skaitus un tuvotos sagaidāmajai vērtībai. Piešķirto punktu summu atrod funkcija *coincidence_total*. Vidēji j-tās valsts i-tajai piešķirto novērtējumu tad atrod *edge_average*.

Nākamais solis ir iegūto vidējo vērtību pārveidošana par varbūtību sadalījumu. Sākumā tas netika darīts, un rezultāts bija grūtības ar izsekošanu vērtību semantiskajai nozīmei un pareizu algoritma realizāciju. Strādājot ar varbūtībām, daudzas potenciālas programmatiskas kļūdas var novērst, vienkārši sekojot līdzi tam, ka rindu, kolonnu vai kopējā varbūtību summa matricā ir vienāda ar 1 (atkarībā no veicamajām darbībām). Papildus tiek veikts arī matemātiski pilnīgi nepamatots taču praktiski pašsaprotams solis - visām vidējo vērtējumu matricas vērtībām tiek pieskaitīta konstante α, kas ir funkcijas parametrs. Tas tiek darīts, jo citādi nogrieztām vērtībām tiek aprēķināts svars 0 un korekcijas koeficients $+\infty$, kas korekcijas mērķiem neder. To var uztvert kā zaudētās informācijas ekstrapolāciju - valstis, kas ne reizi nav saņēmušas rangu 1-10, tik un tā gandrīz noteikti kādas balsis saņemtu, ja tiktu veikta izlase no balsis veidojošā sadalījuma. Visu augstāk minēto veic funkcija *normalized_score*.

Tālāk, 2.1. sadaļā aprakstīto korekcijas matricas aprēķinu veic funkcija *corrections*. Strādājot ar *numpy*, mēdz rasties nepieciešamība veikt skalāras operācijas starp vektoriem, matricām un dažādu dimensiju tensoriem - un tad var pieļaut grūti atrodamas kļūdas, nepareizā secībā savietojot to dažādās dimensijas. Tāpēc funkcija *expand* gluži vienkārši vienreiz atrisina šo problēmu, lai vēlāk programmētājam par to nebūtu jādomā. *weights* un *distance_measure* atgriež attiecīgi noteikto svaru un simetrisko distanču matricas informatīviem mērķiem. Šīs pašas funkcijas izmanto, lai apstrādātu neironu tīkla ģenerētās matemātiskās cerības.

Lai korekciju piemērotu datu kopai, izveidotas funkcijas *apply_correction_matrix* un *rank_corrected*. Pirmā pareizina datu kopu ar korekcijām, saglabājot iztrūkstošo datu vērtības, otrā veic diezgan piņķerīgo koriģētu vērtību pārkārtošanas procedūru, ko nevar pilnībā vektorizēt (t.i, uzreiz nodot aprēķina formulu visam tensoram).

Subjektīvam novērtējumam ar tiktāl minēto pietiek - var aprēķināt matricas koeficientus, pārveidot tos par distancēm, pārbaudīt iegūtās vērtības skaitliski vai veikt ar tām dimensiju redukciju. Taču kā spriest par korekciju "pareizību"? Cik lielā mērā tās patiešām kaut ko uzlabo? Tas, protams, ir filozofisks jautājums, un droši vien pietiekami sarežģīts arī no matemātikas viedokļa. Īpaši neiedziļinoties varētu izmantot intuīciju un pieņemt, ka, ja kaut kādā veidā kompensēsim slēptus svarus, mēģinot vienādot ar tiem iegūtos rezultātus, tad šo te iegūto rezultātu dispersijai vajadzētu mazināties. Funkcijas *split_dataset* un *pre_post_variance* realizē "slinko krosvalidāciju". Proti, viena sadala datu

kopu nejauši izvēlētās apakškopās, otra aprēķina dispersiju katram tensora "slānim" un atgriež vidējās vērtības - pirms un pēc korekcijas. Lai gūtu krosvalidācijai līdzīgu efektu, šo procedūru var atkārtot daudzas reizes. Padodot par argumentu visu treniņa kopu abās ailēs, iespējams novērtēt dispersijas izmaiņas visai kopai.

Visbeidzot, iegūtos rezultātus izvada failos - distanču, svaru, korekciju matricas; valstu indeksiem atbilstošos kodus; datu tensora formu (lai varētu pareizi to atjaunot citur). Uz konsoles tiek izvadīti "krosvalidācijas" rezultāti visiem datiem, 10 neujaušiem dalījumiem, 10 tuvākās distances, 10 tālākās un Latvijai - 5 tuvākās, 5 tālākās. Repozitorijā atrodas arī skripts *mds.R*, kur bez īpašām pārmaiņām pārkopēts kods no 2.1. mājas darba, lai pārbaudītu, vai ar MDS un isoMDS var iegūt vizuāli uzskatāmus rezultātus.

3. Rezultāti

Tā kā neironu tīkls apmācīts uz atšķirīgas formas datu kopas, ar to nav mēģināts rēķināt "krosvalidāciju". Bet analoģiski salīdiznāt iespējams abu modeļu radītos distances mērus.

```
10 closest distances:
('al', 'mk', 0.22886635031781632)
('lv', 'lt', 0.2512198730393583)
('si', 'hr', 0.2521401049055594)
('md', 'ro', 0.26430728020439137)
('hr', 'ba', 0.2726166129790104)
('mk', 'hr', 0.2755331516315524)
('mt', 'lu', 0.28044880967003194)
('ee', 'fi', 0.2827490808148247)
('cy', 'gr', 0.28962514416189455)
('ge', 'lt', 0.28970839212072375)
10 farthest distances:
('am', 'az', 7.740737501095516)
('tr', 'rs', 7.175960311203305)
('it', 'az', 5.818060775489824)
('tr', 'lv', 5.534297314958385)
('ee', 'rs', 5.180505784590336)
('tr', 'cy', 5.177250070995729)
('rs', 'az', 5.048095521545731)
('ch', 'ua', 5.030904666344915)
('ie', 'am', 4.982109374997082)
('mt', 'md', 4.707881300893598)
```

Att. 1: Klasifikatora distances

```
10 closest distances:
('al', 'mk', 0.23084138280959632)
('md', 'ro', 0.2553724339552601)
('lv', 'lt', 0.26341819970254643)
('mk', 'hr', 0.2833631361564437)
('mt', 'lu', 0.29174612073616324)
('rs', 'mk', 0.29614555238299944)
('ge', 'lt', 0.29783346799023414)
('mk', 'ba', 0.30526297244399436)
('lv', 'ee', 0.30827908788517033)
('ge', 'am', 0.30927560451988034)
10 farthest distances:
('tr', 'rs', 10.116266674038648)
('it', 'az', 8.736756047253964)
('am', 'az', 8.531587575450875)
('yu', 'ru', 8.044583443739626)
('yu', 'ua', 7.830527043879655)
('lu', 'ru', 7.750715525851016)
('lu', 'ua', 7.542802458802074)
('yu', 'az', 7.445466102755876)
('yu', 'rs', 7.40618490729717)
('lu', 'az', 7.177469453216153)
```

Att. 2: Vidējo vērtību matricas distances

Redzams, ka tuvības starp Balkānu un Baltijas valstīm abi modeļi atrod ļoti līdzīgas (skaitlisko vērtību sakritība gan ir nejauša, nozīme ir secībai - mainot parametru α var iegūt dažādas absolūtās vērtības), taču atšķiras valstis, ko modeļi iedala "tālajā galā".

```
Distances to Latvia:
('lv', 'lt', 0.2512198730393583)
('lv', 'ee', 0.31562955956837224)
('lv', 'ie', 0.5504004353971912)
('lv', 'ge', 0.5985488627911816)
('lv', 'dk', 0.6237675011361759)
...
('lv', 'rs', 3.4309126376899775)
('lv', 'al', 3.8441120439583703)
('lv', 'bg', 3.979162247421961)
('lv', 'ba', 4.602940728155071)
('lv', 'tr', 5.534297314958385)
```

Att. 3: Klasifikatora distances - Latvijai

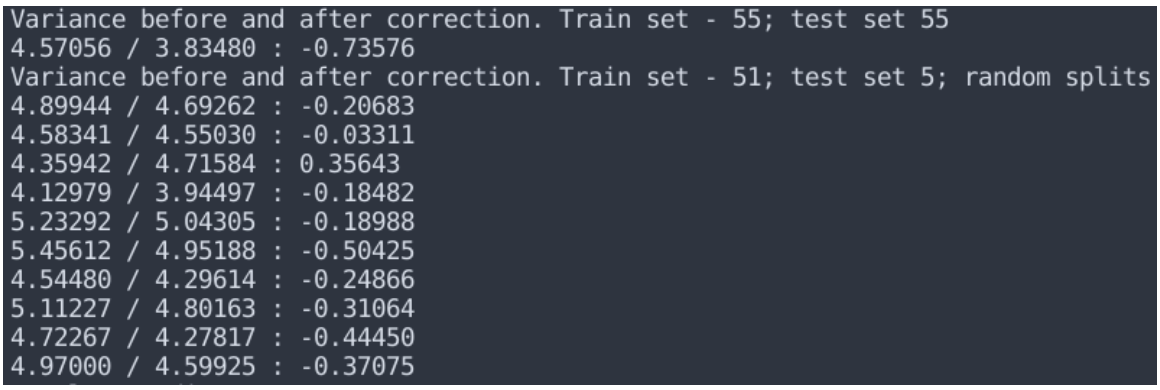
```
Distances to Latvia:
('lv', 'lt', 0.26341819970254643)
('lv', 'ee', 0.30827908788517033)
('lv', 'ie', 0.6109779325132721)
('lv', 'ge', 0.6167449582216571)
('lv', 'no', 0.6556400104442234)
...
('lv', 'bg', 4.535719241029227)
('lv', 'al', 5.313071620296169)
('lv', 'tr', 5.4204298335344)
('lv', 'lu', 5.572481768036559)
('lv', 'yu', 5.881886482136249)
```

Att. 4: Vidējo vērtību matricas distances - Latvijai

Droši vien pārsteidzošākais secinājums, ja var ticēt iegūtajiem rezultātiem, ir tas, ka pie gandrīz jebkuriem parametriem, viena no stiprākajām "kaimiņu būšanām" ir tieši Latvijai un Lietuvai. Protams, Balkāni veido cieši saistītu grafu, taču savstarpējā favorītisma ziņā laikam tomēr iepaliek. Tāpat saraksta augšgalā gandrīz vienmēr ir Rumānija ar Moldovu un Albānija ar (Ziemeļ)Maķedoniju.

```
Variance before and after correction. Train set - 55; test set 55
4.57056 / 4.22202 : -0.34854
Variance before and after correction. Train set - 51; test set 5; random splits
4.89944 / 8.90881 : 4.00936
4.58341 / 8.06198 : 3.47857
4.35942 / 9.53019 : 5.17077
4.12979 / 5.81298 : 1.68319
5.23292 / 8.25463 : 3.02171
5.45612 / 7.39811 : 1.94199
4.54480 / 7.22218 : 2.67738
5.11227 / 7.24402 : 2.13175
4.72267 / 7.67969 : 2.95702
4.97000 / 7.53223 : 2.56224
```

Att. 5: Dispersijas izmaiņas - reizinot ar c_{ij}



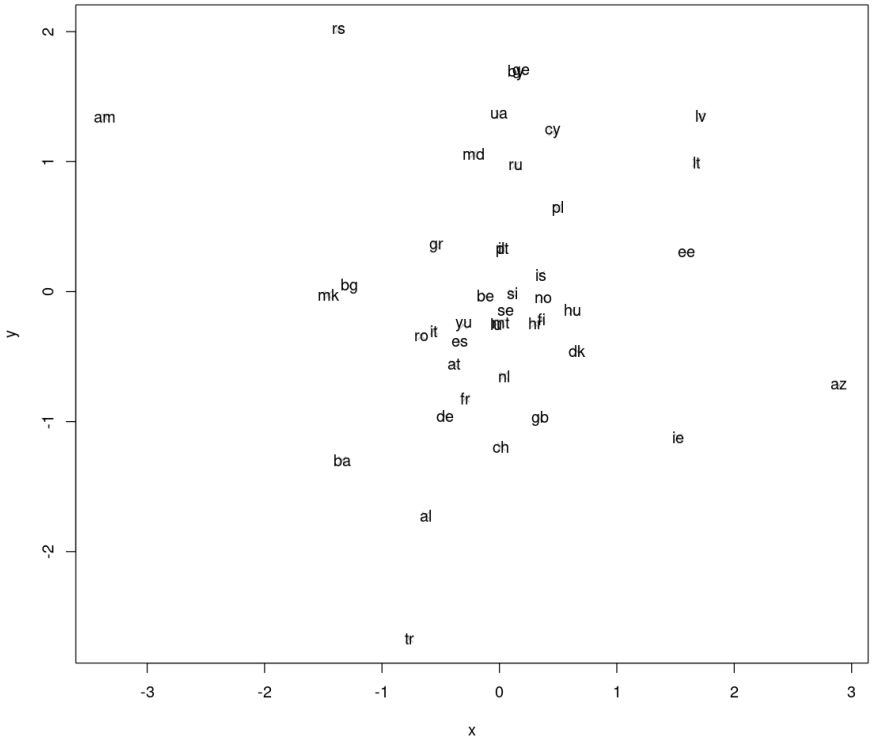
Att. 6: Dispersijas izmaiņas - reizinot ar $\sqrt{c_{ij}}$

Salīdzinot datu kopu dispersijas (vidējās vērtība korpusam kopumā no 2-dimensionālo sadalījumu dispersijām katra etapa ietvaros), redzams, ka, vienkārši reizinot ar korekcijām, pārkārtojot pašu treniņa kopu un pārrēķinot rangus, dispersija samazinās, taču to pašu darot ar datiem, kas neietilpst treniņa sadalī-jumā, dispersija pieaug. Savukārt reizinot ar korekcijas kvadrātsakni, dispersija krītas abos gadījumos - arī datiem, kas treniņa kopā nav iekļauti. Pastāv divas visai ticamas iespējas:

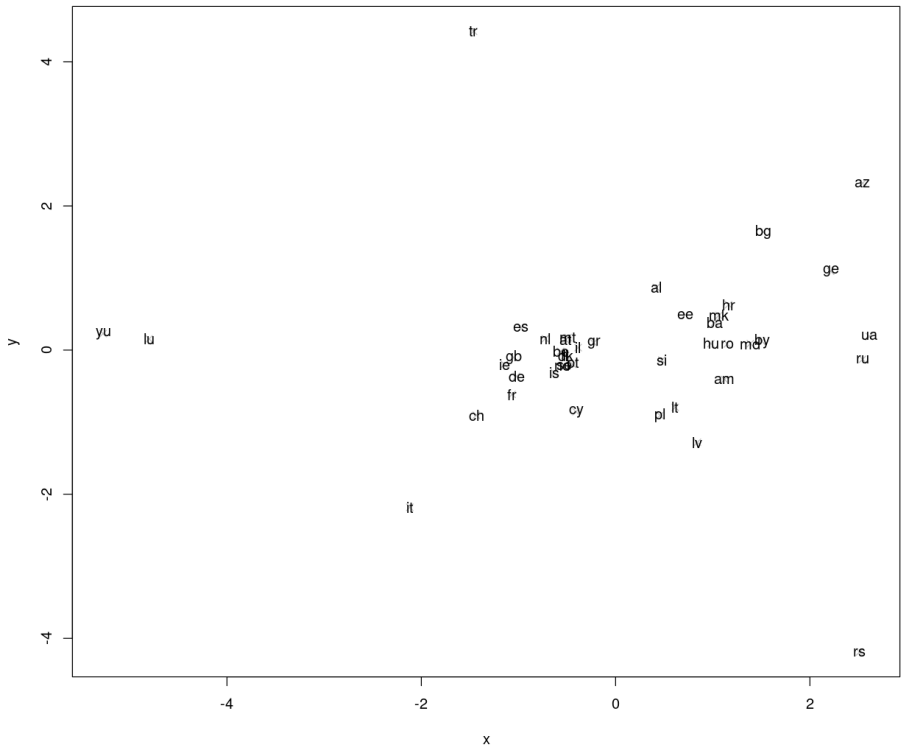
- Ir kāda vienkārša (vai ne tik vienkārša) matemātiska sakarība, kas determinēti nosaka, ka tā jābūt, bet patstāvīgā darba autors savos neveiklajos un kļūdpilnajos aprēķinos to vienkārši nav pamanījis. Tam varētu būt saistība ar divkāršo koeficientu rēķināšanu - vispirms vienādojot rindu varbūtību summas, tad līmeņojot tās lokāli;

- Notiek kaut kas analoģisks *overfitting* - koriģējot ar pilnajām vērtībām tiek izdarīts par daudz, rezultāti tiek samudžināti un kropļoti. Tieši šī intuīcija ir tas, kas iedvesmoja veikt šādu (autoraprāt) matemātiski nepamatotu pārveidojumu, turklāt pat pirms dispersiju salīdzināšanas. Laimīgas sagadīšanās rezultātā šī korekcija patiešām izrādījās noderīga. Tas ir, ja šāds dispersiju izmaiņas novērtējums vispār kaut ko nozīmē - kas nebūt nav garantēts.

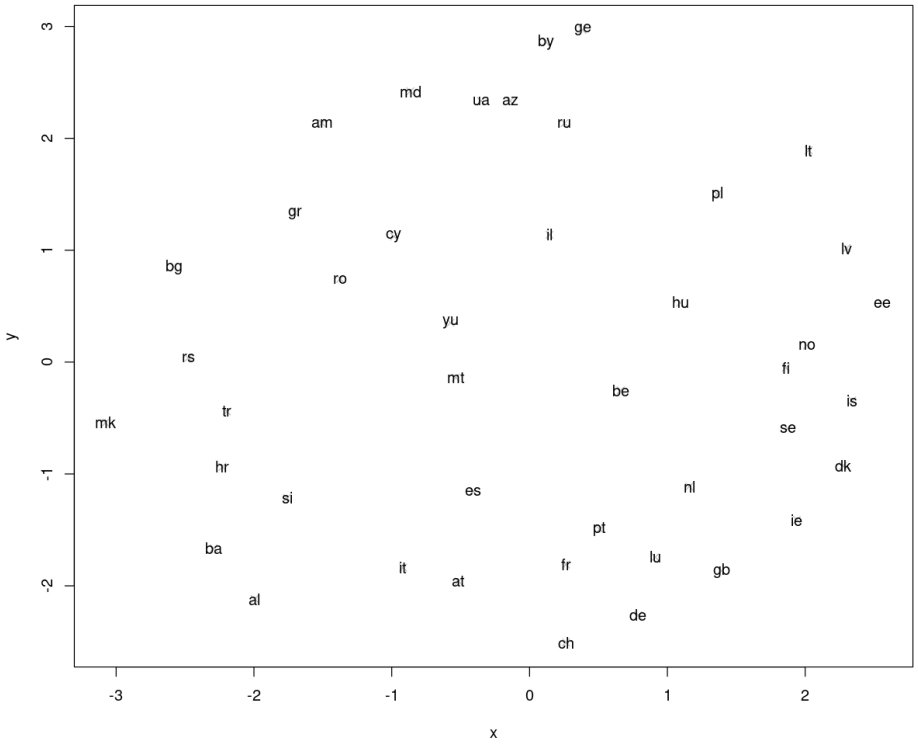
Izmēģināti trīs dažādi vizualizācijas paņēmieni - *MDS*, *isoMDS* un *t-SNE*. Trešais nekādus saprotamus rezultātus sniegt nespēja, tāpēc netiek iekļauts. Pilna izmēra attēli atrodami projekta repozitorijā "tex/" direktorijā, kur ir arī šīs atskaites LaTeXpirmokds.



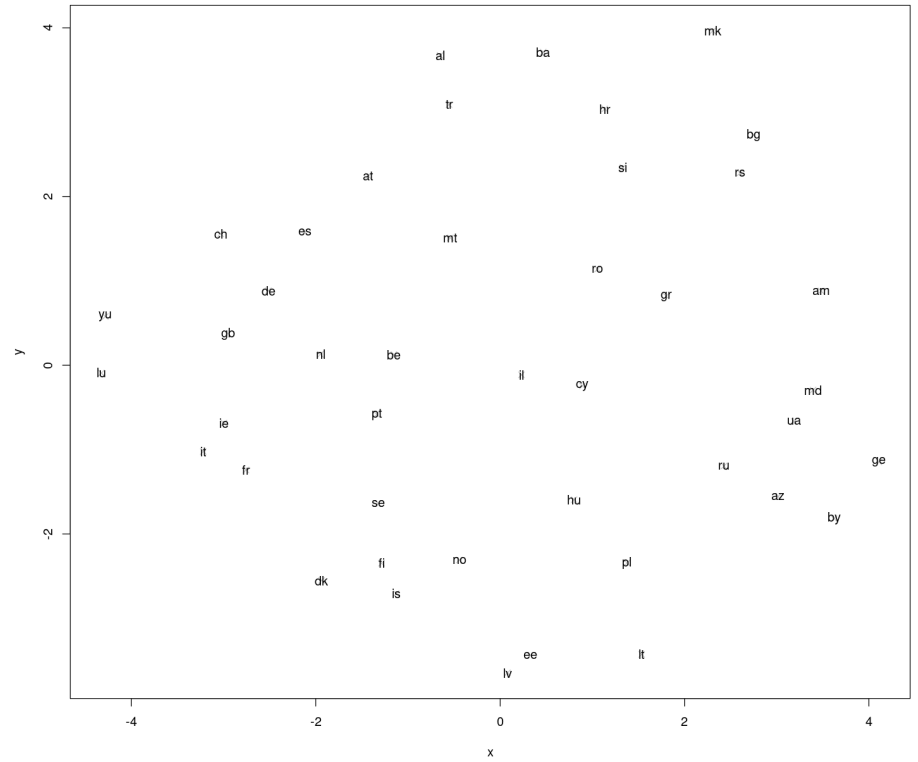
Att. 7: MDS vizualizācija klasifikatora distancēm



Att. 8: MDS vizualizācija klasifikatora distancēm



Att. 9: isoMDS vizualizācija klasifikatora distancēm



Att. 10: isoMDS vizualizācija klasifikatora distancēm

Redzams, ka MDS gadījumā atrastā projekcija ir visnotaļ nevienmērīga, ar klasteriem tuvu koordinātu sākumpunktam un izlēcējiem. *isoMDS* iegūst daudz vienmērīgāku izkliedi. Abos gadījumos tomēr ir redzams, kā, piemēram, Balkānu, Skandināvijas, Austrumeiropas (tās šaurajā definīcijā) vai Baltijas valstis grupējas.

4. Secinājumi

Darba gaitā tika diezgan detalizēti iepazīta konkrēta datu kopa - Eirovīzijas dziesmu konkursa rezultāti (precīzāk, laika periodā starp 1980. un 2015. gadu, kad vērtēšana notika pēc vienāda principa vai vismaz pietiekami nemainīga principa), izvirzīta hipotēze par ģenerējošo modeli t.s. "kaimiņu būšanas" fenomenam, piedāvāta metode tā koriģēšanai. Praktiski tika izstrādāti skripti gan tipveida klasifikatora ģeneratora pielietojumam, gan datu kopas elementārai algebriskai reducēšanai uz formu, kurā aprēķins būtu izsakāms analītiski.

Rezultātā tika iegūti "kaimiņu būšanas" novērtējumi no diviem radikāli atšķirīgiem modeļiem, kas tomēr daudzējādā ziņā ir līdzīgi. To ģenerēto skaitlisko vērtību un vizualizāciju pārbaude atklāj likumsakarības, kas sakrīt ar cilvēka intuitīvo izpratni par meklējamās parādības būtību. Piedāvātajai korekcijas metodei tika arī izstrādāts kvantitatīvs novērtējums, taču par tā nozīmīgumu ir grūti spriest, jo, tāpat kā pati aprēķinu secība, kas noved pie modeļa, tas nav nekā dziļi matemātiski pamatots.