

BIOLAB AND COLLABORATORS

UVOD V RUDARJENJE BE- SEDIL

BIOLAB

Copyright © 2021 Biolab and Collaborators

PUBLISHED BY BIOLAB

TUFTE-LATEX.GOOGLECODE.COM

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

First printing, November 2021

Kazalo

Delotoki v Orangeu 5

Priprava besedil 8

Kontekst 11

Vreča besed 12

Hierarhično razvrščanje v skupine 13

Hierarhično razvrščanje besedil 15

Klasifikacija 18

Logistična regresija 19

Ocenjevanje modelov 20

Obogatitev besed 21

Npovedovanje 23

Analiza sentimenta 24

Geo označevanje 26

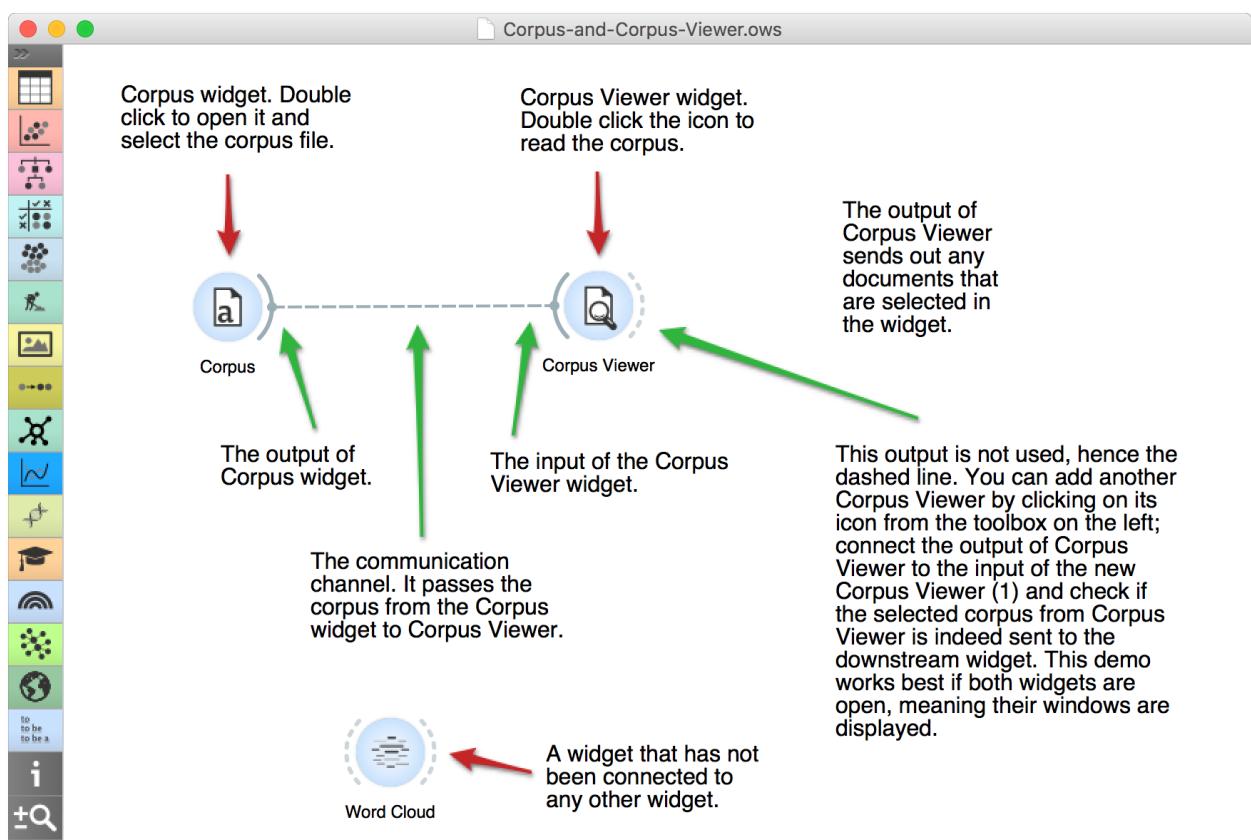
Moji podatki 28

Literatura 29

Stvarno kazalo 30

Delotoki v Orangeu

DELOTOKI V ORANGEU so sestavljeni iz komponent, ki berejo, procesirajo in prikazujejo podatke. Te komponente imenujemo gradniki oz gradniki. Na desni je prazen prostor, t.i. platno. Nanj polagamo gradnike. Gradniki v Orangeu komunicirajo preko komunikacijskih kanalov. Izhod iz enega gradnika je uporabljen kot vhod za drug gradnik.

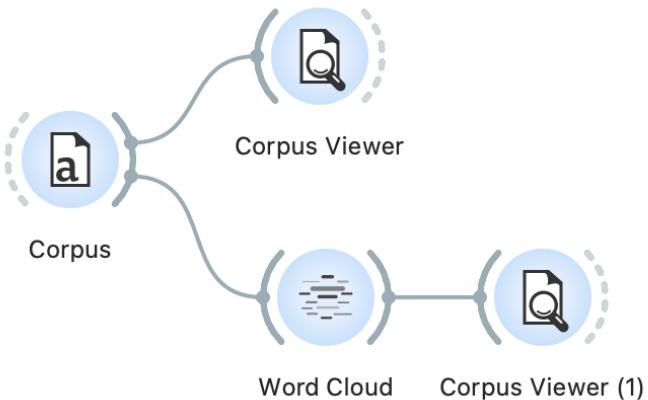


Delotoke sestavljamo tako, da polagamo gradnike na platno in jih povezujemo. Povezavo ustvarimo tako, da potegnemo črto od izhodnega v vhodni gradnik. Izhodi gradnika so na desni, vhodi pa na levi strani. V zgornjem delotoku gradnik *Corpus* pošilja podatke v gradnik *Corpus Viewer*.

Slika zgoraj kaže preprost delotok z dvema povezanimi gradnikoma in enim gradnikom brez povezav. Izhodi gradnika so na desni strani, vhodi pa na levi.

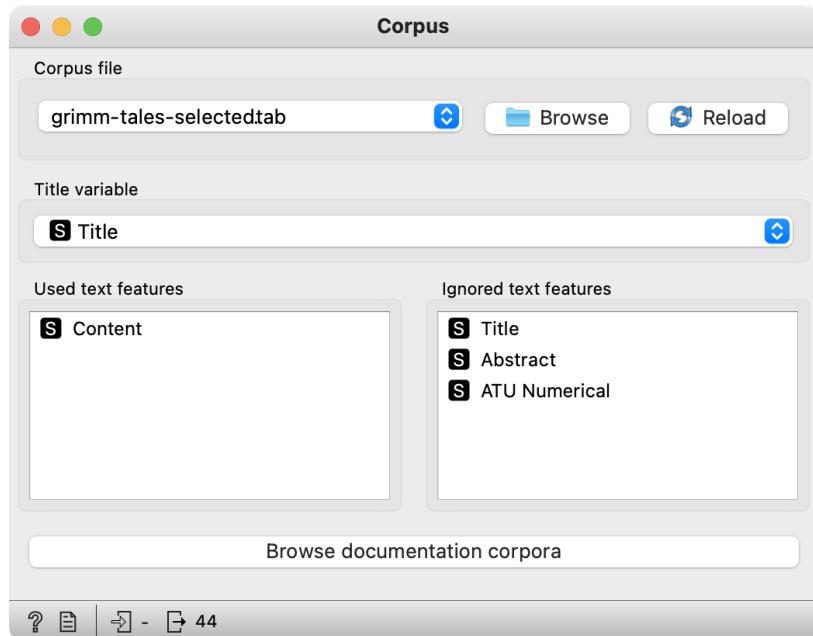
Pričnite z gradnjo delotoka, ki vsebuje gradnik *Corpus*, dva gradnika *Corpus Viewer* in gradnika *Word Cloud*:

Delotok z gradnikom *Corpus* bere podatke iz računalnika in jih pošlje v gradnika *Corpus Viewer* in *Word Cloud*. *Corpus Viewer* prikaže besedila v iskalniku, *Word Cloud* pa izriše najpogosteje besede. Dokumenti, ki vsebujejo izbrano besedo iz gradnika *Word Cloud*, so prikazani v gradniku *Corpus Viewer* (1).



Gradnik *Corpus* bere podatke iz lokalnega diska. Odprite *Corpus* tako, da dvakrat kliknete na ikono. Dodatek Text že vsebuje nekaj prednaloženih korpusov. Iz teh ("Browse") izberite *Grimm-tales-selected.tab*, korpus z izbranimi Grimmovimi pravljicami.

Oranjevi delotoki se pogosto pričnejo z gradnikoma File ali *Corpus*. Korpus Grimmovih pravljic vsebuje 44 dokumentov. Polje "Used text features" na levi pove, katere stolpce bomo smatrali kot del besedila, medtem ko polje na desni vsebuje dodatne informacije (naslov, povzetek, itd.).



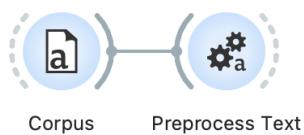
Odprite *Word Cloud*. *Word Cloud* (oblak besed) prikaže pogostost besed v dokumentih, kjer so pogosteje besede prikazane sorazmerno večje. Izberite besedo v oblaku in jo pošiljte v *Corpus Viewer* (1). Sedaj lahko pregledate samo dokumente, ki vsebujejo izbrano besedo.



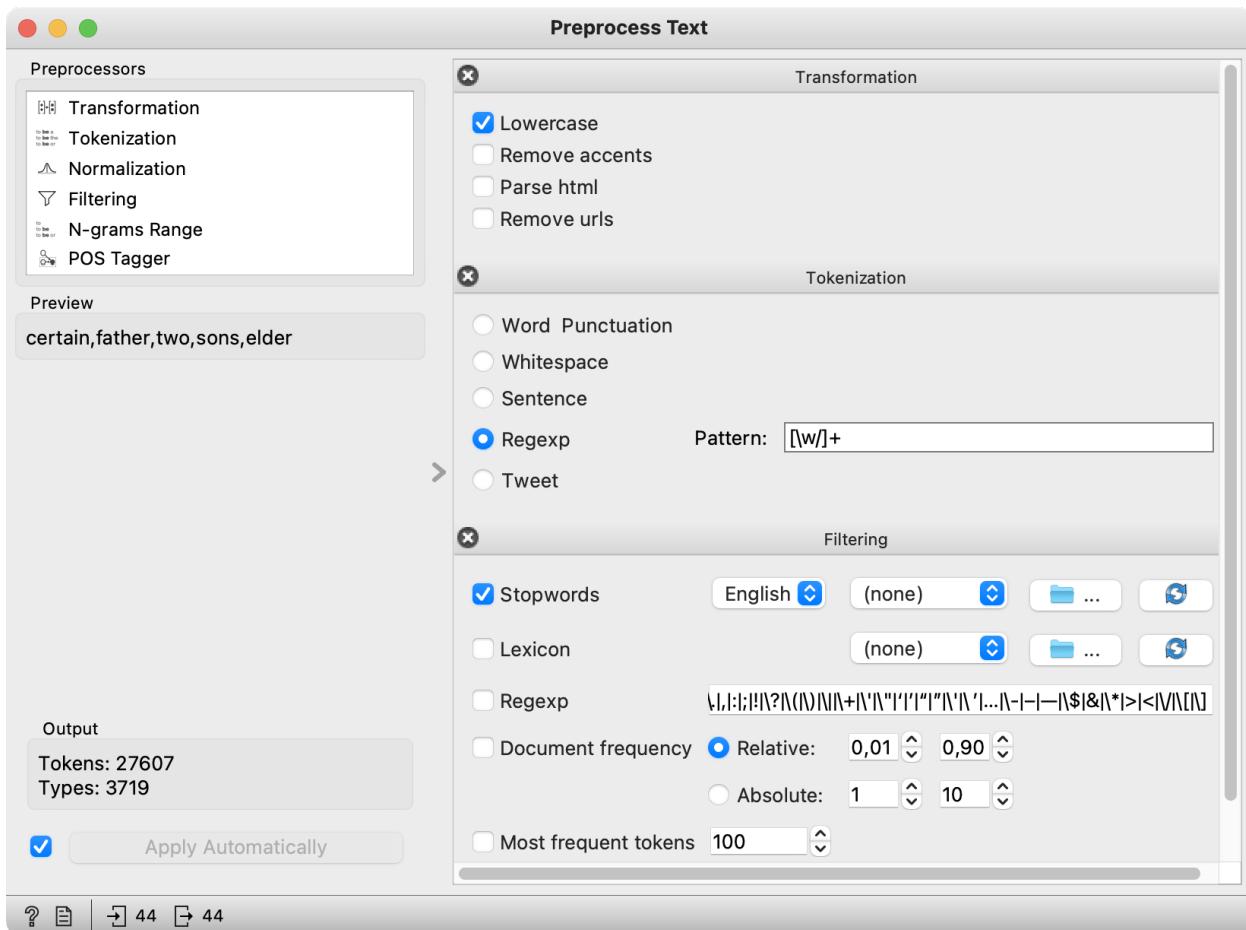
Počakajmo malo! Ta oblak besed je živa groza! Vidimo lahko celo kopico semantičnih smeti. Ali obstaja kakšen način, da to nekako uredimo?

Seveda! Odstraniti moramo vse delčke, ki ne vsebujejo nikakršne informacije, točneje ločila in odvečne besede (členke, pomožne glagole, veznike).

Priprava besedil



Word Cloud je preprosto prikazal vse besede in simbole, ki obstajajo v besedilu. Ampak običajno to ni to, kar hočemo. Ponavadi želimo prikazati zgolj pomenske enote, torej semantično bogate besede. Zato potrebujemo predprocesiranje.



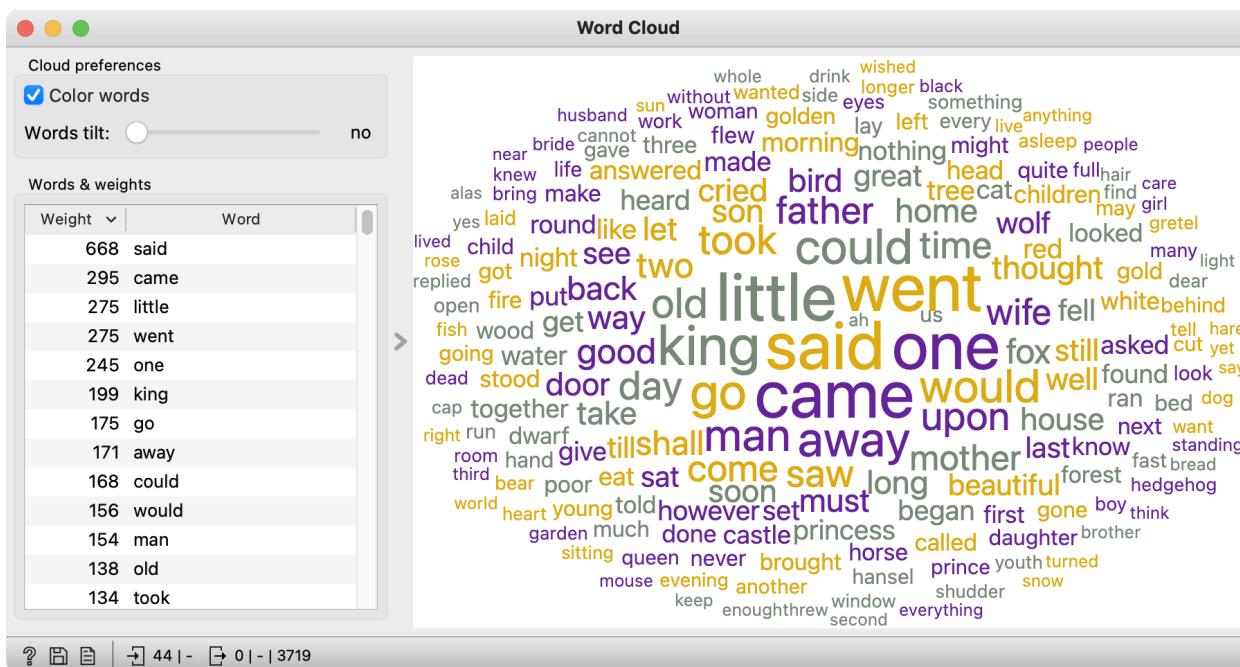
Preprocesiranje definira kaj je pomembno v podatkih. Je Ždravnik "enako kot ždravnik"? Naj upoštevamo besede kot so "in", "ali", "ko" ali naj jih izpustimo? Ali želimo upoštevati besedi "živel" in "živi" kot isto besedo? Preprocesiranje definira osnovne enote analize.

Token je osnovna enota naše analize. Lahko je beseda, besedna zveza, stavek... S predprocesiranjem definiramo osnovne enote za analizo.

V gradniku *Preprocess Text* smo vse besede pretvorili v male črke, vsako besedo smo obravnavali kot svojo *menoto (token)* in odstranili ločila, na koncu pa smo odstranili tudi nepomenske besede (npr. 'in', 'da', 'čeprav'). Takšno predprocesiranje ustvari sledeče enote:

"To je vzorčni stavek." → "vzorčni", "stavek"

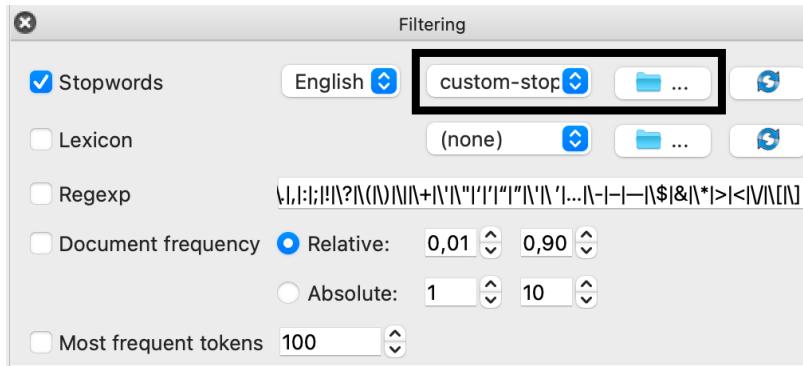
Rezultate predprocesiranja si lahko pogledamo v gradniku *Word Cloud*, kjer vidimo najpogosteje enote. S to vizualizacijo lahko identificiramo odvečne besede in nepravilnosti.



Ker ne želimo upoštevati besed brez pomena, smo že odstranili nekaj nepomenskih besed. Ampak morda generično filtriranje ni dovolj za našo analizo.

V tem primeru vedno lahko naložimo seznam besed po meri. Odprite urejevalnik besedil in ustvarite seznam nepomenskih besed oz. besed, ki jih želite odstraniti. Vsako besedo zapišite v svojo vrstico in shranite dokument v obliki *.txt*.

Rezultate predprocesiranja vidimo v gradniku Word Cloud. Dve najpogosteji besedi sta "would" in "could". Če se odločimo, da ti dve besedi nista primerni za našo analizo, ju moramo odstraniti. To lahko storimo s filtriranjem po meri.



Dober urejevalnik besedil je Sublime, lahko pa uporabite tudi WordPad ali Word.

Seznam besed naložite s klikom na ikono z mapo poleg opcije *Stopwords* v razdelku *Filtering*.

Filtriramo lahko tudi besede, ki so preredke ali prepogoste. Redke besede se pojavijo običajno le v nekaj dokumentih, medtem ko so prepogoste besede presplošne ali pa nimajo pomena (stopwords). Da bi ohranili le besede, ki zares predstavljajo naš korpus dokumentov, uporabimo filtriranje Document frequency (Pogostost v besedilu). Če nastavimo vrednosti na 0,1 and 0,9, bomo obdržali le tiste besede, ki se pojavijo v več kot 10 % in manj kot 90 % dokumentov.

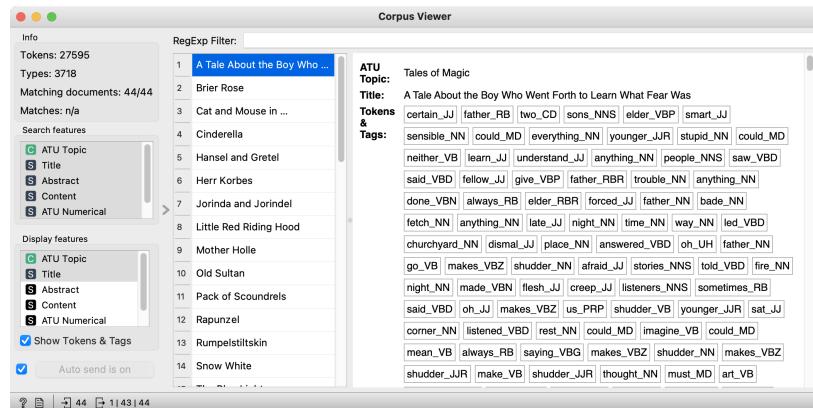
Predprocesiranje je ključ do uspešne analize besedil. Omenili smo le nekaj tehnik, sami pa lahko preizkusite še druge, na primer:

- *normalizacija (Normalization)* pretvori vse besede v korene oz. osnovne oblike (na primer sinovi v sin)
- *n-grami* so večje enote, na primer bigrami (par zaporednih besed) in trigrami (trojke besed)
- *oblikoskladenjsko označevanje (POS tagging)* označi vsako enoto s njenim oblikoskladenjskim vlogo (sinovi → samostalnik, množina, oznaka = NNS)

Pred kratkim smo za slovenščino dodali korenjenje z orodjem UDPipe

Za razlagovo POS oznak glejte:
<http://nl.ijs.si/imp/msd/html-sl/>

Na sliki vidite gradnik Corpus Viewer, s katerim si lahko pogledamo naslove, besedila dokuemntov in enote na katere je preprocesirane razbilo besedilo. V našem primeru imamo poleg enot prikazane tudi POS oznake.

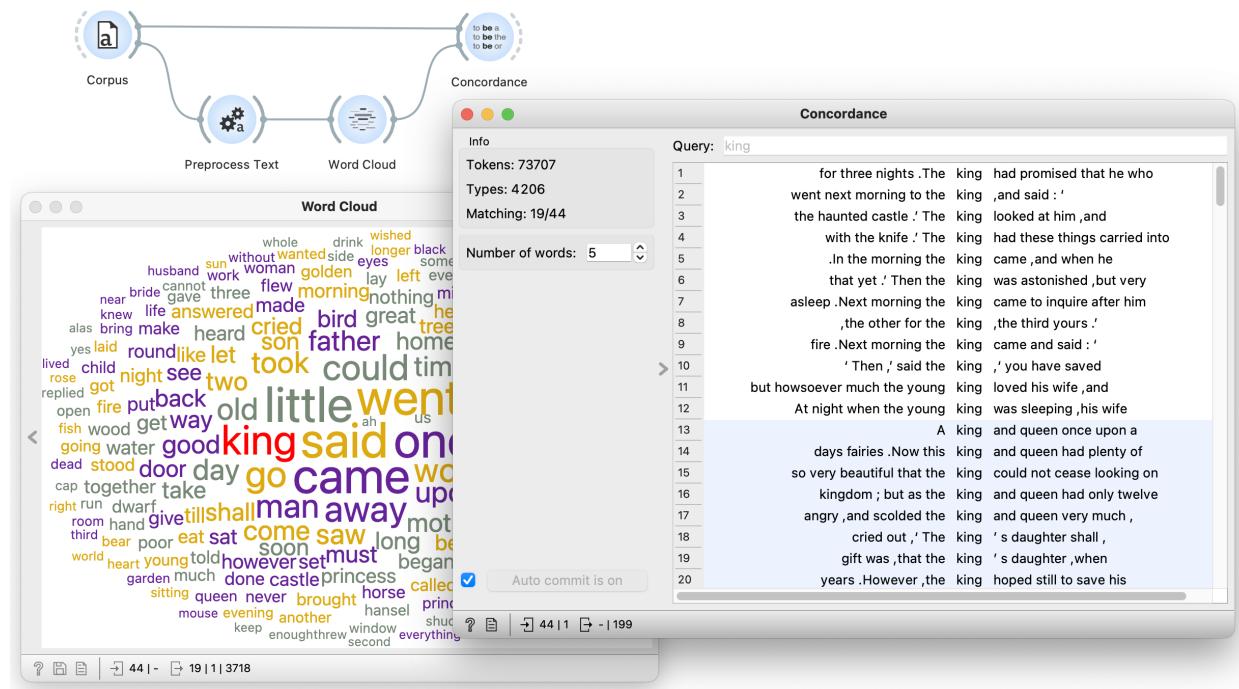


Kontekst

Sedaj smo pripravili korpus in čas je, da ga prikažemo. En način prikaza je oblak besed, ki ga že poznamo. *Word Cloud* nam prikaže pogostost besed. Pogostejša kot je beseda, z večjimi črkami bo zapisana.

Še vedno pa ne vemo, kako se besede uporabljajo v besedilu. Na primer 'oh' je lahko maločrkovna verzija besede OH (kemijska spojina hidroksid), preprost vzklik 'Oh!' ali pa kratica za ameriško zvezno državo Ohio.

Da bi preverili kontekst posamezne besede, lahko uporabimo gradnik Concordance. *Concordance* na pokaže besedilo okrog izbrane besede. Pozvežite *Concordance* z gradnikom *Corpus*. Tako *Concordance* dobi vhodno besedilo. Besedo lahko poiščemo z iskalnikom na vrhu gradnika ali pa jo izberemo v gradniku *Word Cloud*.



V tem primeru smo izbrali besedo "king" v oblaku besed in preverili njen kontekst v gradniku Concordance.

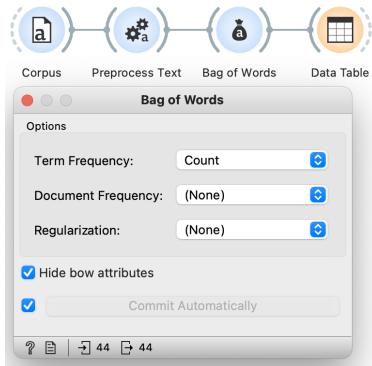
Vizualizacije v Orangeu so narejene tako, da podpirajo izbor podmnožic. Odkrivanje zanimivih podmnožic in raziskovanje njihovih podobnosti je ključni del odkrivanja znanj iz podatkov.

Dokumente, ki vsebujejo izbrano besedo, si pogledamo tako, da izberemo dokumente v gradniku Concordance in jih pošljemo v Corpus Viewer za podrobnejšo analizo.

Vreča besed

Sedaj imamo predprocesirano besedilo, s pravimi enotami, ampak še vedno pa ne moremo odkriti nikakršnih vzorcev v besedilu. Za to potrebujemo številke in preprost način, kako spremenimo besedila v številske vektorje je... da preštejemo besede v vsakem dokumentu!

	this	is	an	example	another	apple
"This is an example"	1	1	1	1	0	0
"Another example"	0	0	0	1	1	0
"This is another apple"	1	1	0	0	1	1



Gradnik *Bag of Words* ustvari tabelo z besedami v stolpcih in dokumenti v vrsticah. Vrednosti so pojavitve besed v vsakem dokumentu.

Besede lahko preprosto preštejemo (TF ali term frequency) ali pa besede utežimo glede na to, kako pogosto se pojavijo v dokumentih (IDF ali inverse document frequency). S TF-IDF bodo pogoste besede imele nizko vrednost, saj se pojavijo velikokrat pri velikem deležu dokumentov, medtem ko bodo imele pomembne besede visoko vrednost, saj se pojavijo pogosto v majhnem deležu dokumentov.

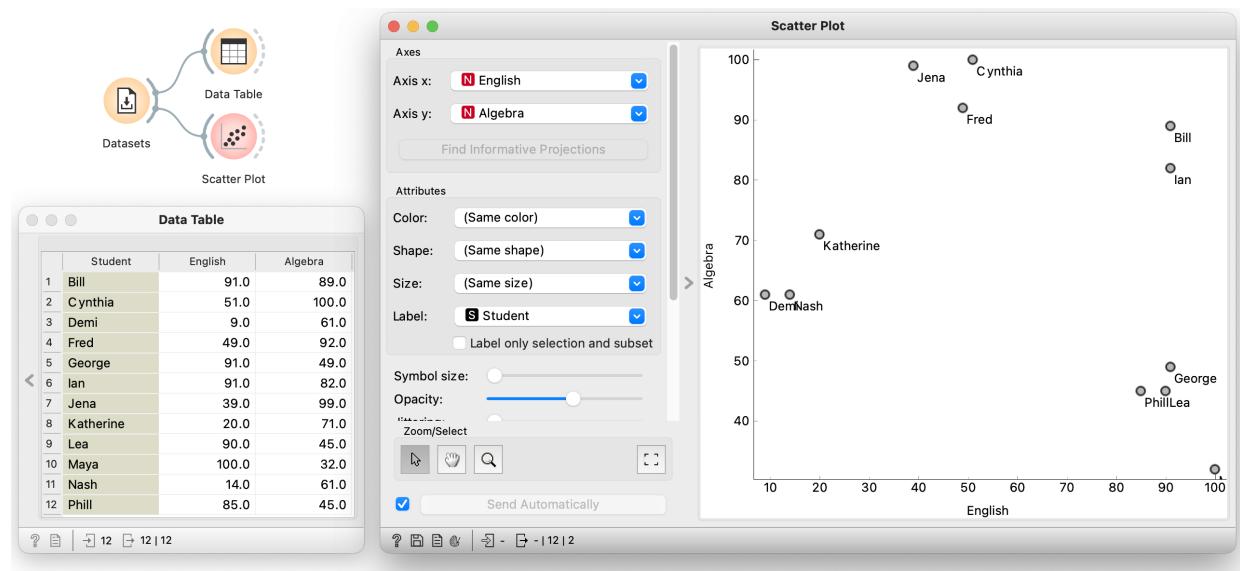
Podatke pošilje v gradnik *Bag of Words* in od tam naprej v *Data Table*. Vidimo nov stolpec, ki vsebuje pojavitve besed za vsak dokument. Sedaj imamo številke in končno lahko pričnemo z analizo!

Data Table		
bow-feature hidden include skip-normalization title	ATU Topic	Title
1	Tales of Magic	A Tale About...
2	Tales of Magic	Brier Rose
3	Animal Tales	Cat and Mou...
4	Tales of Magic	Cinderella
5	Tales of Magic	Hansel and ...
6	Animal Tales	Herr Korbes
7	Tales of Magic	Jorinda and ...
8	Tales of Magic	Little Red Ri...
9	Tales of Magic	Mother Holle
10	Animal Tales	Old Sultan
11	Animal Tales	Pack of Sco...
12	Tales of Magic	Rapunzel
13	Tales of Magic	Rumpelstilts...
14	Tales of Magic	Snow White
15	Tales of Magic	The Blue Light
16	Animal Tales	The Bremen ...
17	Animal Tales	The Crumbs ...
18	Animal Tales	The Dog and...
19	Tales of Magic	The Elves an...
20	Tales of Magic	The Fisher...
(...)		

Hierarhično razvrščanje v skupine

Ena od nalog rudarjenja besedil je iskanje zanimivih skupin dokumentov. Torej radi bi odkrili dokumente, ki so si podobni mes sabo.

Poglejmo si preproste podatke z dvema stolpcema (glejte opombo) in jih prikažimo v gradniku *Scatter Plot*. Koliko skupin imamo? Kaj predstavlja različne skupine? Kateri primeri sodijo v posamezno skupino?



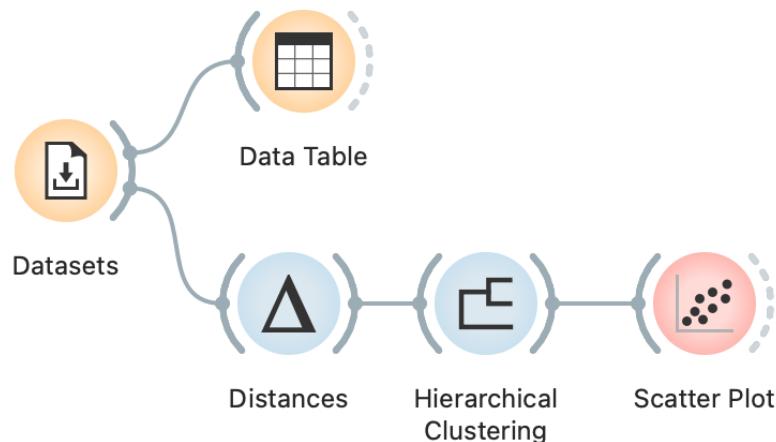
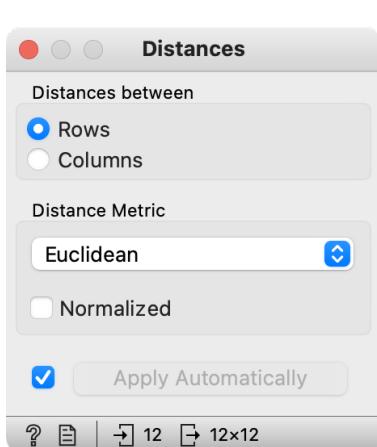
Kaj sploh pomeni "podobno"? Študenti so opisani s številskimi spremenljivkami, torej s ocenami pri predmetu. Ena od mer podobnosti je *evklidska razdalja*, ki preprosto izmeri razdaljo med dvema študentoma (točkama) v prostoru, kot bi to storili z metrom.

Sedaj definirajmo še postopek za razvrščanje v skupine. Recimo, da začnemo z vsakim dokumentom v svoji skupini, nato pa v vsakem koraku združimo skupini, ki sta si najbolj podobni. Razdaljo med skupinami izračunamo kot povprečje razdalj med posameznimi elementi skupine. Tak postopek imenujemo hierarhično razvrščanje v skupine.

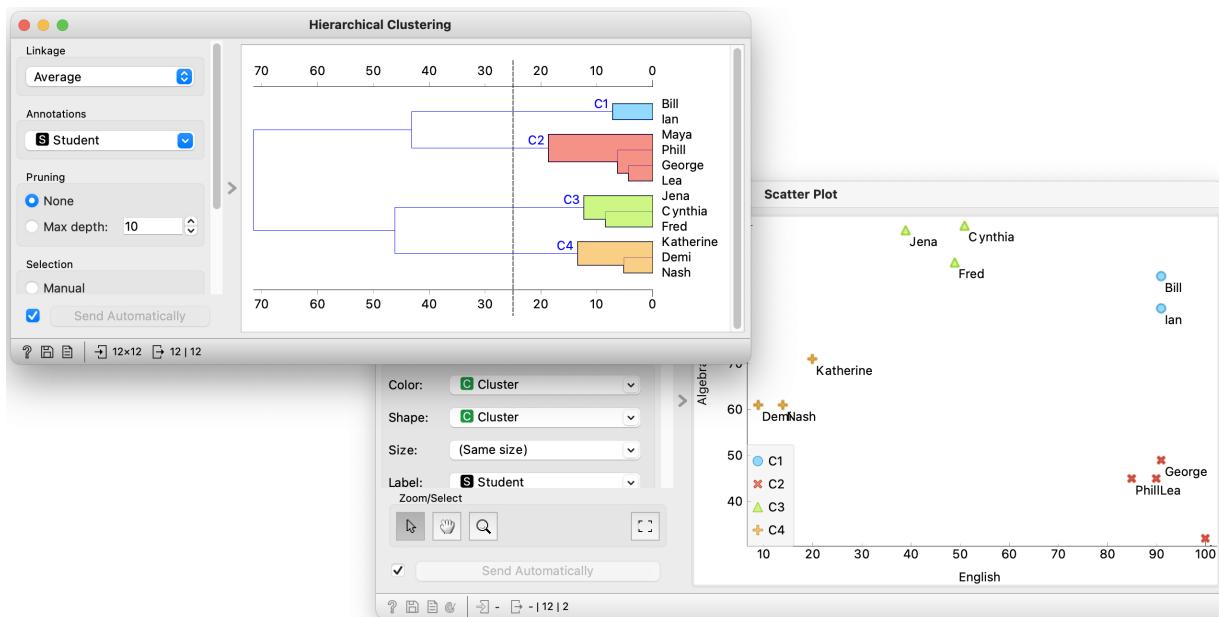
Razvščanje v skupine bomo predstavili s preprostimi podatki o študentih in njihovih ocenah pri angleščini in matematiki. Podatki so dostopni v Datasets widgetu.

Načinov merjenja razdalj med skupinami je več. Način, ki smo ga opisali, se imenuje *povprečna razdalja (average linkage)*. Lahko bi računali tudi *razdaljo med najbližnjima točkama v skupini (single linkage)* ali pa med točkama, ki sta si *najbolj oddaljeni (complete linkage)*.

Rezultate razvrščanja v skupine na primeru naših študentov si lahko pogledamo v sledečem delotoku:



Naložite podatke z gradnikom *Datasets*, izračunajte razdalje z gradnikom *Distances*, uporabite *Hierarchical Clustering* in si poglejte rezultate v gradniku *Scatter Plot*. Gradnik Hierarchical Clustering omogoča, da hierarhijo skupin odrežemo pri določeni meri podobnosti in tako definiramo skupine.



Hierarhično razvrščanje besedil

Vrnimo se h Grimmovim pravljicam in ustvarimo naslednji delotok:



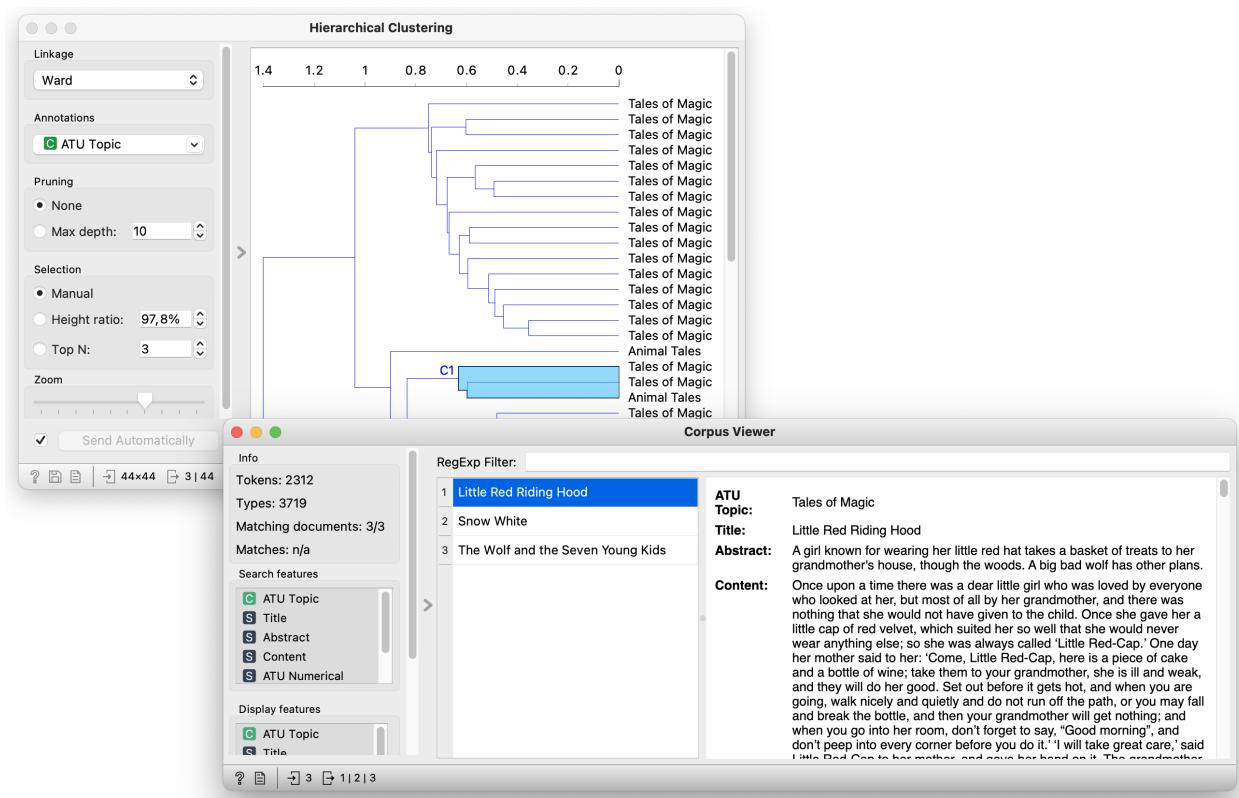
Gradnik *Hierarchical Clustering* prikaže gruče v obliki dendrograma. Hierarchical Clustering povežite z gradnikom *Corpus Viewer* in odprite oba gradnika. Izberite gručo v dendrogramu in v gradniku *Corpus Viewer* poglejte, kateri dokumenti pripadajo izbrani skupini.

Raziščite različne gruče. Zakaj so nekatere magične pravljice (Tales of Magic) pomešane z živalskimi pravljicami (Animal Tales)? Kaj imajo skupnega?

Enak delotok lahko preizkusite tudi na drugih podatkih, na primer na bookexcerpt.tab, ki vsebuje izvlečke knjig za odrasle in otroke.

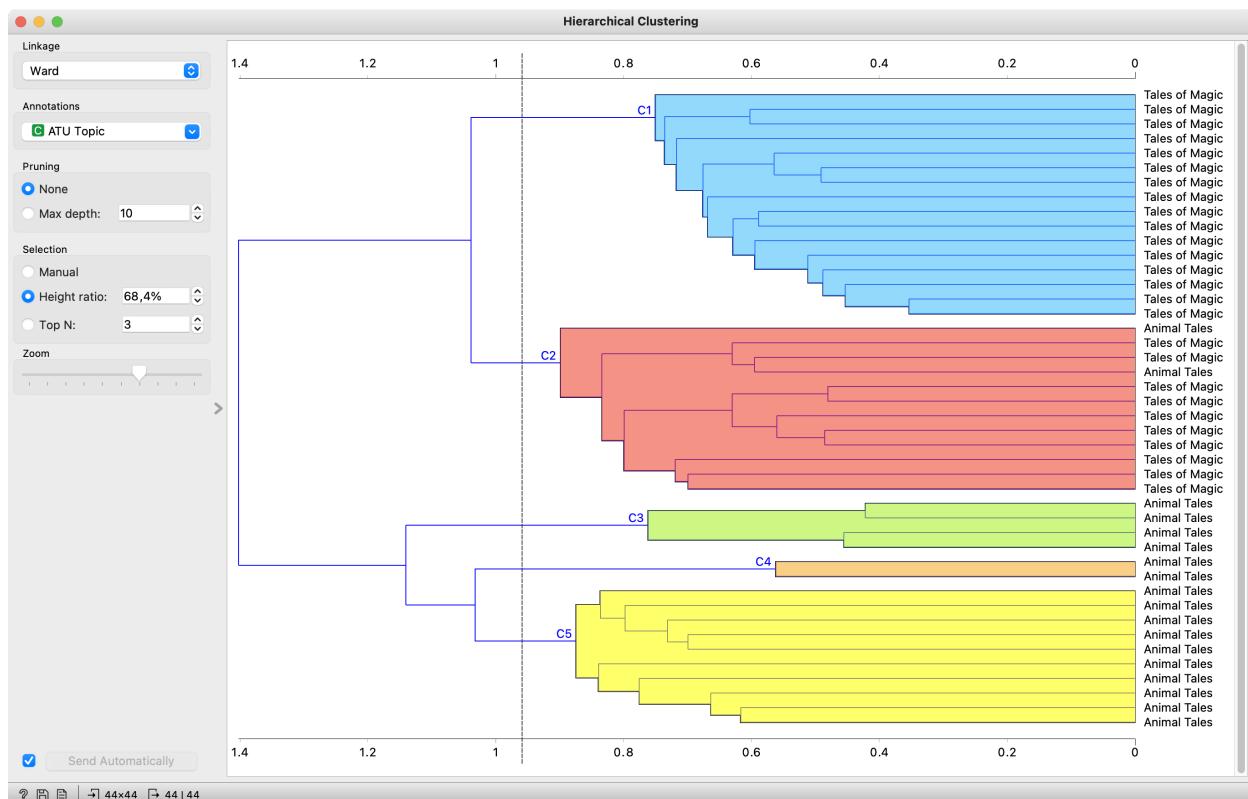
V tem primeru smo za razliko od prejšnjega uporabili *kosinusno razdaljo*. Pogostost besed iz vreč besed je predstavljena z vektorji, ki kažejo vsak v svojo smer glede na vsebino posameznega dokumenta. Kosinusna razdalja je kot med temi vektorji.

Beseda dendrogram je sestavljena iz grških besede dendro "drevo" in gramma "risba" in pomeni hierarhično vizualizacijo v obliki drevesa.

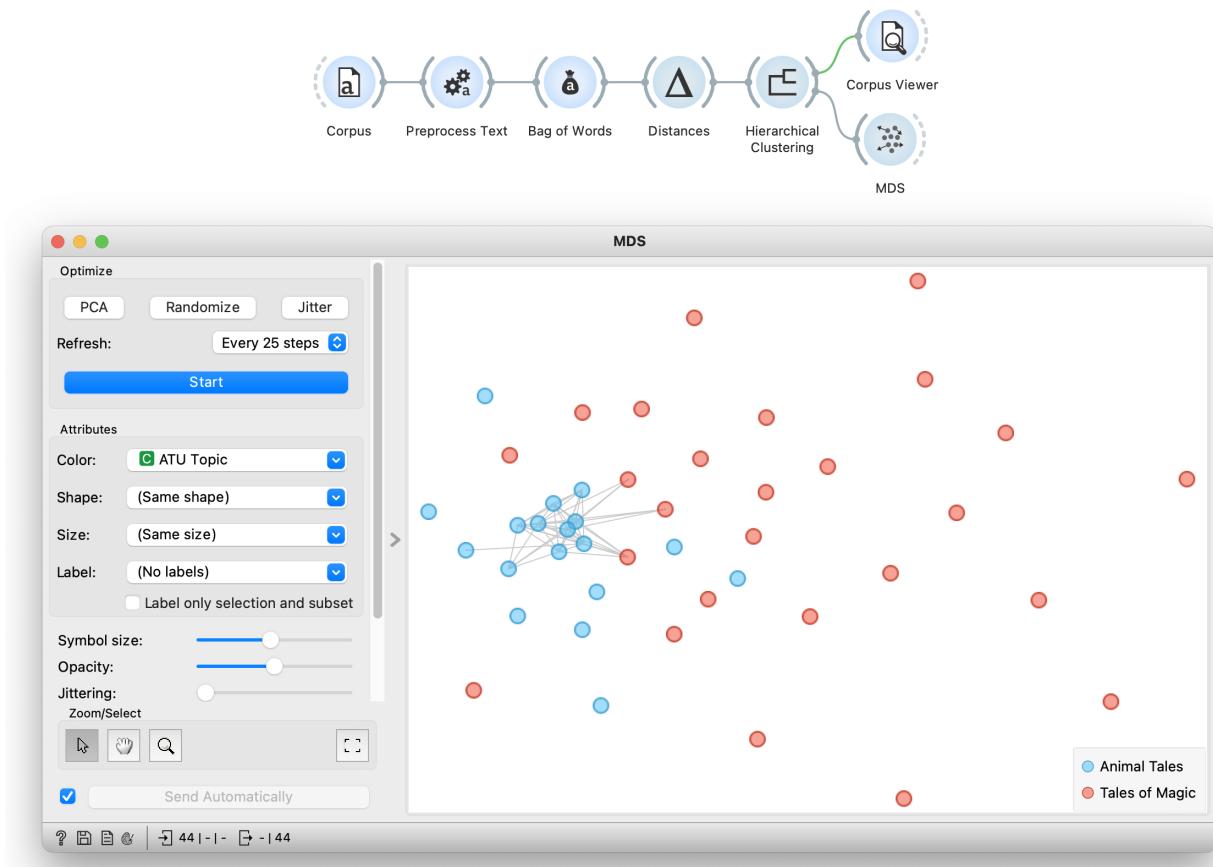


Hierarhično razvrščanje zgradi hierarhijo dokumentov, mi pa se moramo odločiti, kje zamejimo podobnost znotraj skupine. Mejo podobnosti nastavimo tako, da na ravnili zgoraj povlečemo črto desno ali levo in s tem zamejimo skupine.

Odločili smo se za pet skupin, saj po tem razdalja med skupinami kar precej naraste. Primerjajte pet skupin s širimo, šestimi ali sedmimi. Gruče, ki jih odkrijemo, hkrati sovpadajo z oznako tipa Aarne-Thompson (ATU Topic).



Kako blizu pa so si v resnici živalske pravljice iz tretje in te iz četrte skupine? Ali ne bi bilo bolj zanimivo pogledati dokumente v ravnini, kjer bi se podobni dokumenti nahajali skupaj, različni pa narazen? Taka vizualizacija se imenuje večrazsežnostno lestvičenje oziroma *Multidimensional Scaling (MDS)*.



Magične pravljice tvorijo eno skupino, živalske pa drugo - tako kot smo pričakovali. Zanimivo, magične pravljice so si med sabo bolj podobne kot živalske (bolj so povezne). Razščite podobne pravljice tako, da jih izberete v vizualizaciji in jih pogledate v gradniku Corpus Viewer.

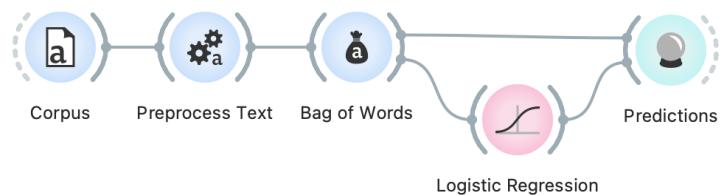
ni vec res

Klasifikacija

Aarne and Thompson sta folklorista, ki uvedla sistem klasifikacije ljudskih pravljic glede na podlagi motiva. Sistem je v uporabi od leta 1810 do danes. U v ATU pomeni Uther, ki je leta 2004 zadnji posodobil indeks.

Omenili smo že tip Aarne-Thompson (ATU). To je indeks folklornih motivov in naše pravljice so že označene z visokonivojskim (žanr) in srednjenvojskim motivom (podžanr).

Ali bi morda lahko napovedali ATU tip na podlagi vsebine pravljice? Pa poglejmo.



Za začetek potrebujemo ciljno spremenljivko. To je spremenljivka, ki jo želimo napovedati, v našem primeru tip ATU. Potrebujemo tudi številsko reprezentacijo besedila - vrečo besed, ki nam jo je izračunal gradnik Bag of Word.

Sedaj bomo zgradili napovedni model. Model na podlagi enot (besed) napove ciljno vrednost (tip ATU). Vsak model potrebuje tudi postopek, ki definira, kako model upošteva besede. V našem primeru bo to *logistična regresija*.

V gradniku *Predictions* vidimo stolpec, kjer so napovedane vrednosti po postopku logistične regresije. Izgleda, da je naš model pravilno napovedal večino pravljic.

The screenshot shows the Weka interface for 'Predictions'. On the left, a sidebar lists 'Show probabilities for' categories: 'Animal Tales' and 'Tales of Magic'. The main area displays a table titled 'Predictions' with the following data:

	Logistic Regression	ATU Topic	Title	Abstract
1	0.00 : 1.00 → Tales of M...	Tales of Magic	A Tale About...	A simple boy...
2	0.00 : 1.00 → Tales of M...	Tales of Magic	Brier Rose	An offended ...
3	1.00 : 0.00 → Animal Tales	Animal Tales	Cat and Mou...	A mouse live...
4	0.00 : 1.00 → Tales of M...	Tales of Magic	Cinderella	The familiar ...
5	0.00 : 1.00 → Tales of M...	Tales of Magic	Hansel and ...	A poor wood...
6	1.00 : 0.00 → Animal Tales	Animal Tales	Herr Korbes	A hen and a ...
7	0.00 : 1.00 → Tales of M...	Tales of Magic	Jorinda and ...	A witch lures...
8	0.00 : 1.00 → Tales of M...	Tales of Magic	Little Red Ri...	A girl known ...
9	0.00 : 1.00 → Tales of M...	Tales of Magic	Mother Holle	A widow spo...

Below the table, a summary row shows performance metrics:

Model	AUC	CA	F1	Precision	Recall
Logistic Regres...	1.000	1.000	1.000	1.000	1.000

At the bottom, navigation icons include a question mark, a file icon, a right arrow, and a status bar showing '44 | 44 | 1x44'.

Logistična regresija

V zgornjem delotoku smo uporabili logistično regresijo, priljubljeno metodo strojnega učenja. Pogosto se uporablja v rudarenju besedil zaradi hitrosti in napovedne točnosti. Kako pa deluje?



Pri postopku besede glasujejo. Na primer, beseda 'fox' v besedilu glasuje, da je pravljica živalskega tipa. Enako glasujejo mačke, ptiči in volkovi, ampak manj močno (črte v vizualizaciji so krajše). Lisica je očitno najboljši namig, da je pravljica živalska.

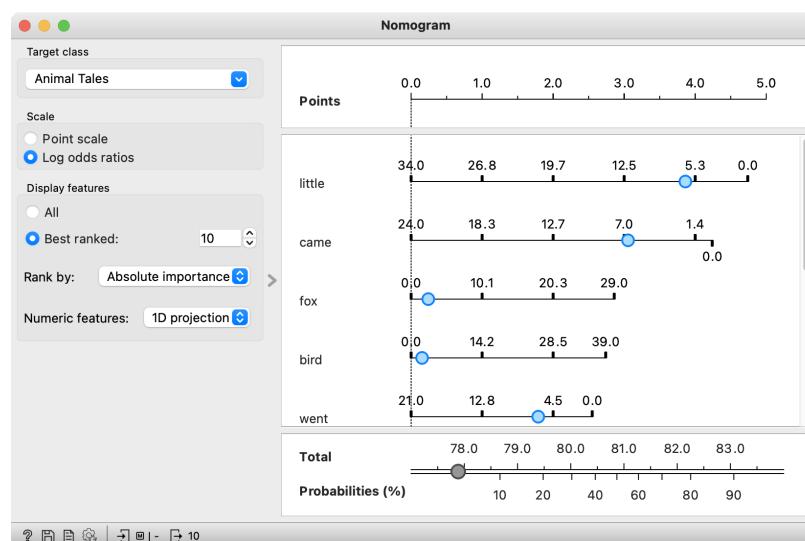
Beseda "little" glasuje obratno. Ravno tako beseda "came" (opposite, kako imajo te besede ničlo na desni strani črte?). Na drugi strani če se v pravljici pojavi več lisic, bolj verjetno bo pravljica živalska.

Vsaka beseda prispeva h končni oceni. Če v pravljici ni besede "majhen", potem živalskim dodeli 5 točk. In če je v besedilu 29 lisic, potem model dodeli 3 točk za živalske pravljice.

Jasno je dejanska metoda nekoliko bolj zapletena, saj poskuša najti pravilno ravnotežje med utežmi glasov in mejami odločitve. Ampak to vsebuje kot volk strašno linearno algebro, tako da se raje ne bomo podali na to pot.

Vizualizacija se imenuje nomogram in prikaže točke (glasove) za najboljših 6 spremenljivk za ciljno vrednost, ki je izbrana levo zgoraj

[preveri tfidf](#)

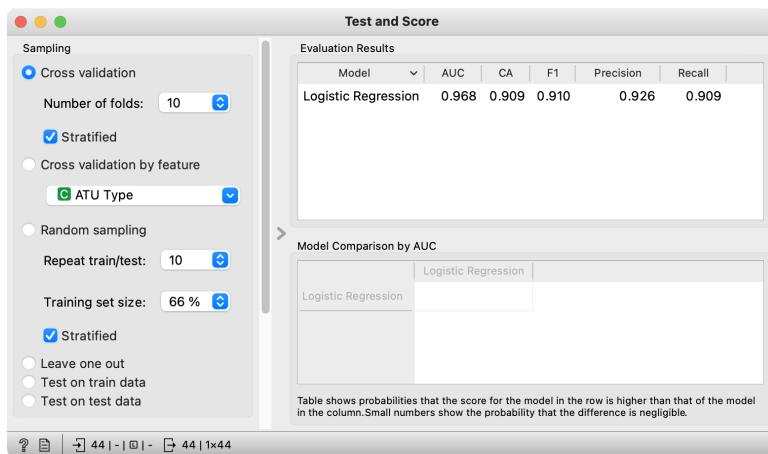


V gradniku Nomogram lahko interaktivno opazujete odločitve modela. Povlecite modre točke levo ali desno tako, da bo končna vsota točk (Total) kar največja.

Ocenjevanje modelov

Pogledali smo si logistični model – lisice, ptice, kralje. Shema glasovanja je izgledala smiselna. Pred tem pa smo videli napovedi modela. Tudi te so bile točne. Kako pa bi lahko izračunali točnost modela?

Morda lahko izračunamo zgolj razmerje pravilno napovedanih pravljic? Taka mera se imenuje *napovedna točnost (classification accuracy)*. Na primer, če smo pravilno napovedali 40 pravljic izmed 44, bo naša napovedna točnost $40/44$ oziroma 91%.

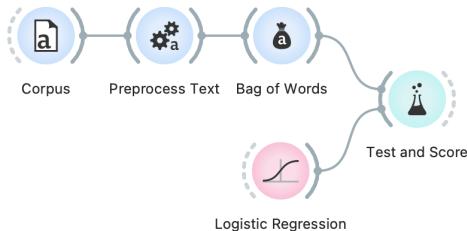


AUC je ena izmed boljših mer točnosti. Na kratko, bližje ko je mera vrednosti 1, boljša je točnost modela.

Gradnik za računanje točnosti modela se imenuje *Test & Score*. Potrebuje dva vhoda: podatke za testiranje modela in napovedni postopek.

“Ni ravno smiselno, da vprašamo model, ali je Zlatolaska živalska pravljica, če smo modelu že pred tem povedali, da je magična.”

Kaj nismo tega storili zgoraj v gradniku Predictions? Prav zares in zato so bile napovedi tako dobre. Modelov se nikoli ne sme preverjati na podatkih, ki smo jih jim dali za učenje.



Tokrat logistična regresija ne potrebuje vhodnih podatkov. Namesto tega bo gradnik podal postopek za gradnjo modela. Nato bo Test & Score v več ponovitvah uporabil postopek na različnih podmnožicah podatkov. V vsaki ponovitvi bo naučil model na izbrani podmnožici in uporabil podatke izven množice za testiranje. Ni ravno smiselno, da vprašamo model, ali je Zlatolaska živalska pravljica, če smo modelu že pred tem povedali, da je magična.

Obogatitev besed

Sedaj o modelu vemo že veliko, ampak kaj ne bi bilo super pogledati, za katere pravljice se je model zmotil in za katere je imel prav? Gradnik *Confusion Matrix* omogoča prav to!

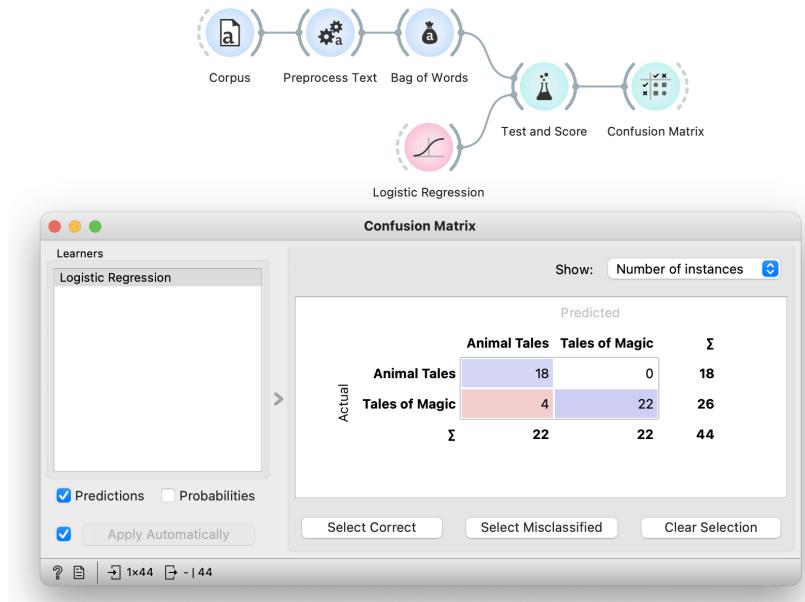


Tabela vsebuje pravilno napovedane dokumente v diagonali (modra) in nepravilno napovedane dokumente izven diagonale (rdeča). Vidimo, da so bile 4 živalske pravljice napovedane kot magične in 3 magične kot živalske.

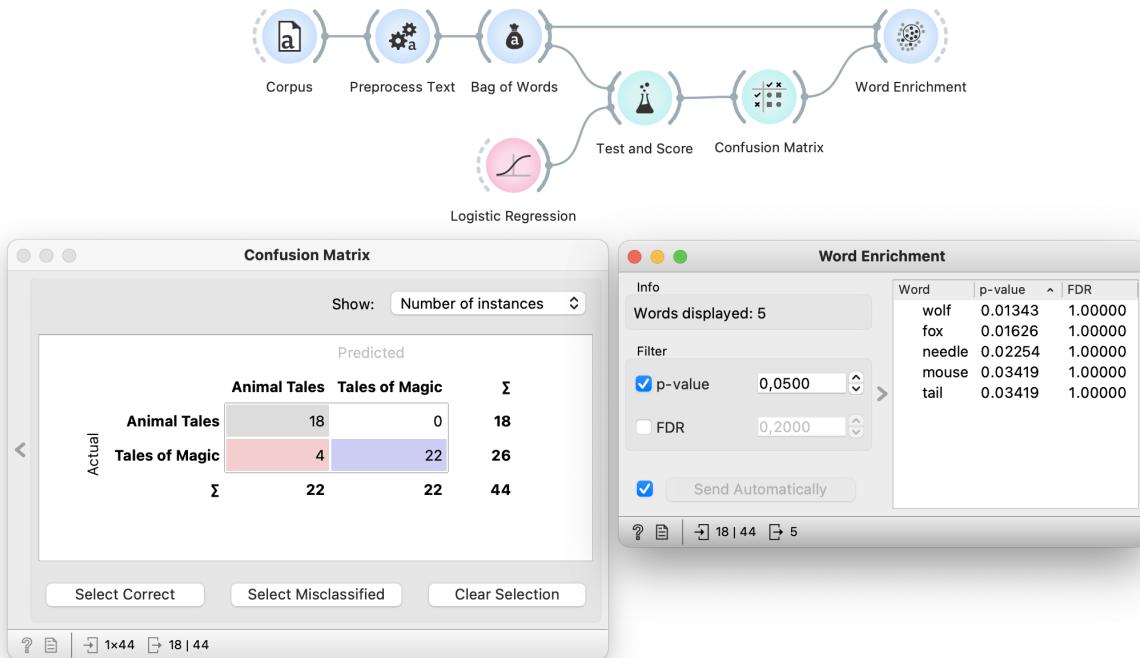
Ampak zakaj? Kaj je tako drugačnega na teh pravljicah, da jih model ni uspel razvrstiti v pravi razred?

Da preverimo napačno klasificirane dokumente, lahko uporabimo *Corpus Viewer*. Ali pa še bolje, pogledamo, katere besede so značilne za katero podmnožico!

Obogatitev besed primerja podmnožico dokumentov s celotnim korpusom in odkrije besede, ki statistično signifikantno zaznamujejo izbrano podmnožico.

Posamezne napačno klasificirane pravljice pogledamo s klikom na polje v tabeli.

Vse napačno klasificirane dokumente lahko izberemo z gumbom 'Select Misclassified'. To bo na izhod poslalo vse napačno klasificirane pravljice, ki jih nato pogledamo v gradniku *Corpus Viewer*.



Word Enrichment deluje na kakršni koli podmnožici. V gradniku Corpus Viewer poiščite vse dokumente, ki vsebujejo besedo queen. Sedaj pošljite izbor v Word Enrichment. Ne pozabite povezati celotnega korpusa iz gradnika Bag of Words - Word Enrichment potrebuje celoten korpus za primerjavo.

Besedi wolf in fox najbolj zaznamujta pravilno napovedane živalske pravljice. Seznam je nekoliko daljši za pravilno napovedane maščne pravljice – king, beautiful, man, itd. Rezultati so zelo podobni tistim iz nomograma. Pravzaprav je to samo drugačen način, kako raziščemo model!

Torej ko boste naslednjič videli lisico v besedilu, ste lahko precej prepričani, da gre za živalsko pravljico! :)

Npovedovanje

todo: preveri kaj delas drugace?

Analiza sentimenta

Kaj pa kakšni novi podatki? Na primer z omrežja Twitter? V građniku *Corpus* kliknite na spustni meni in izberite *election-2016-tweets.tab*. Ta korpus vsebuje 6000 tvitov Hillary Clinton in Donalda Trumpa v času predvolilne kampanje leta 2016.

Najprej moramo podatke predobdelati. Uporabili bomo poseben razčlenjevalnik enot (tokenizer) Tweet, ki je bil naučen na miljonih tvitov.

Tokrat bomo imeli kar veliko unikatnih besed in stvari lahko postanejo precej počasne. Število besed lahko zmanjšamo tako, da uporabimo tehniko *Document frequency*. Nižjo mejo bomo nastavili na 10 - tako bomo obdržali vse besede, ki se pojavijo v več kot 10 tvitih. Super, sedaj imamo znosnih 1000 enot.

Tokrat si bomo podatke pogledali z vidika čustev, ki jih tviti izražajo. Dodajte gradnik *Sentiment Analysis*. Ta deluje tako, da besede iz dokumentov primerja s slovarjem pozitivnih in negativnih besed in na koncu izračuna oceno, kakšno čustvo zaznamuje tvit - pozitivno, negativno ali nevtralno.

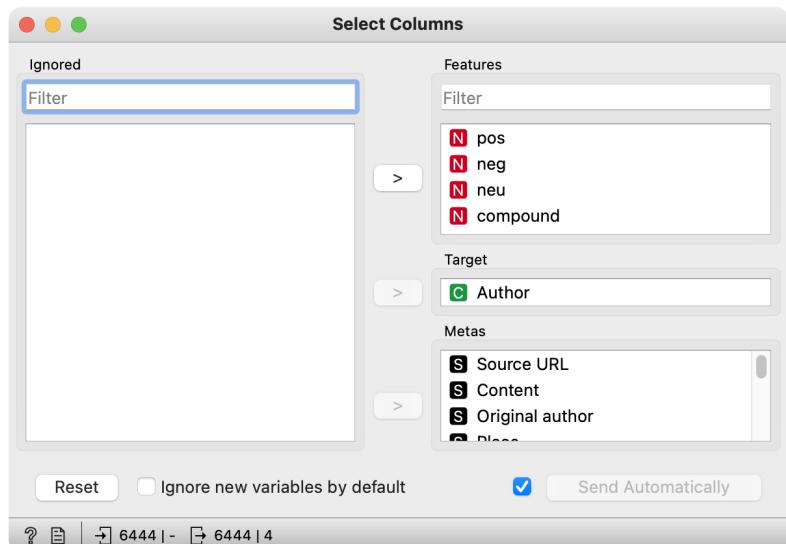
Preden pa pogledamo rezultat, bomo morali še nekaj spremenljivk umakniti iz pogleda (glejte sliko). Uporabite gradnik *Select Columns* in premaknite vse dodatne informacije o tvitih v polje meta attributes.

Tviti pogosto zahtevajo posebno predprocesiranje. Najprej odstranite vse povezave (remove url), ker so običajno nezanimive za analizo. Nato uporabite prednastavljeni Tweet razčlenjevalnik, ki obdrži enote kot so '@omemba', '#oznaka' and emotikoni :). Za konec uporabite še filter Regexp, ki odstrani ločila.

Nastavite:

Razčlenjevalnik Tweet se uporablja za tvite, prednastavljeni razčlenjevalnik Regexp pa za običajno besedilo.

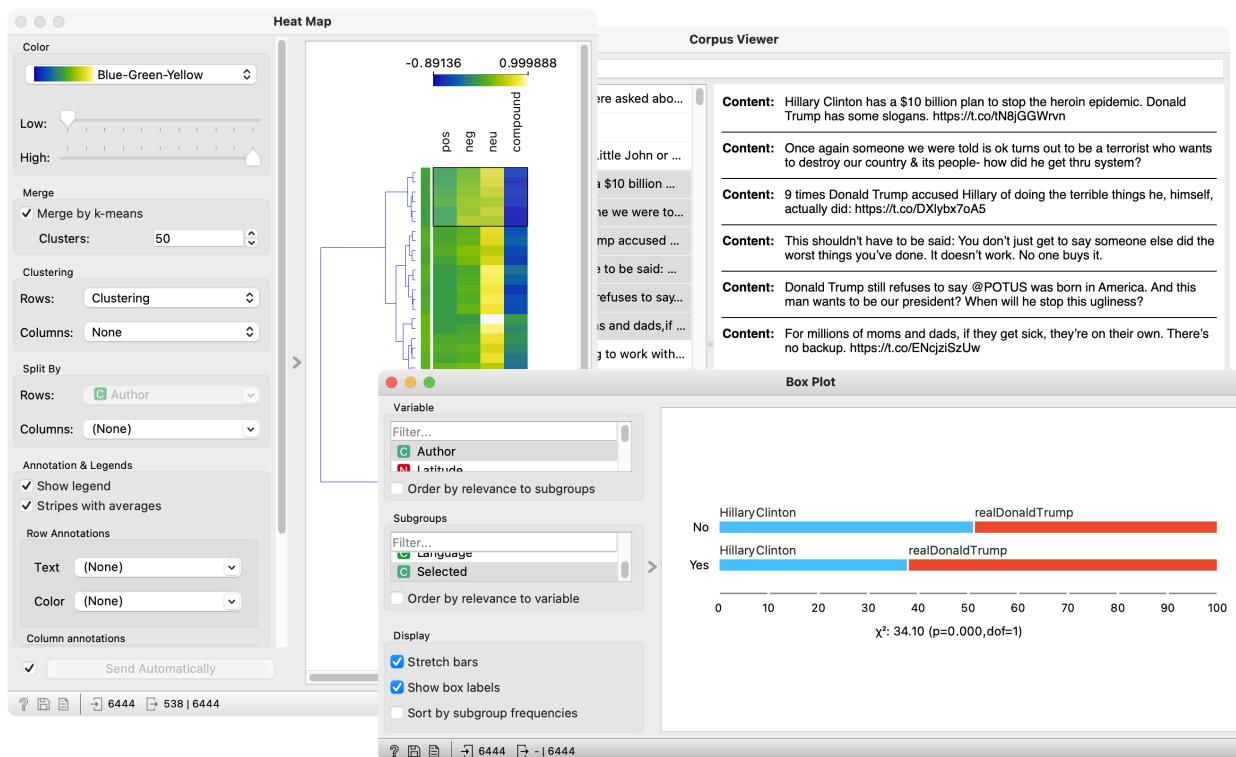
Nastavite pogostost v dokumentih na 10. To bo odstranilo enote, ki se pojavijo v manj kot 10 tvitih.



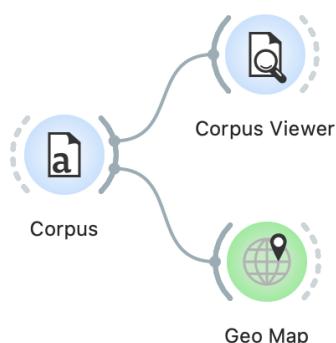
Nato dodajmo gradnik *Heat Map*, ki prikazuje številske vrednosti stolpcov z barvno shemo. Modra polja imajo nizko vrednost, rumena pa visoko. Z drugimi besedami, kjer je vrednost spremenljivke compound morda, je tvit negativen, kjer pa je rumena, je pozitiven.



V prikazu lahko označimo negativne tvite in jih pogledamo v *Corpus Viewerju* ali pa pogledamo, kdo jih je napisal z gradnikom *Box Plot*.



Geo označevanje



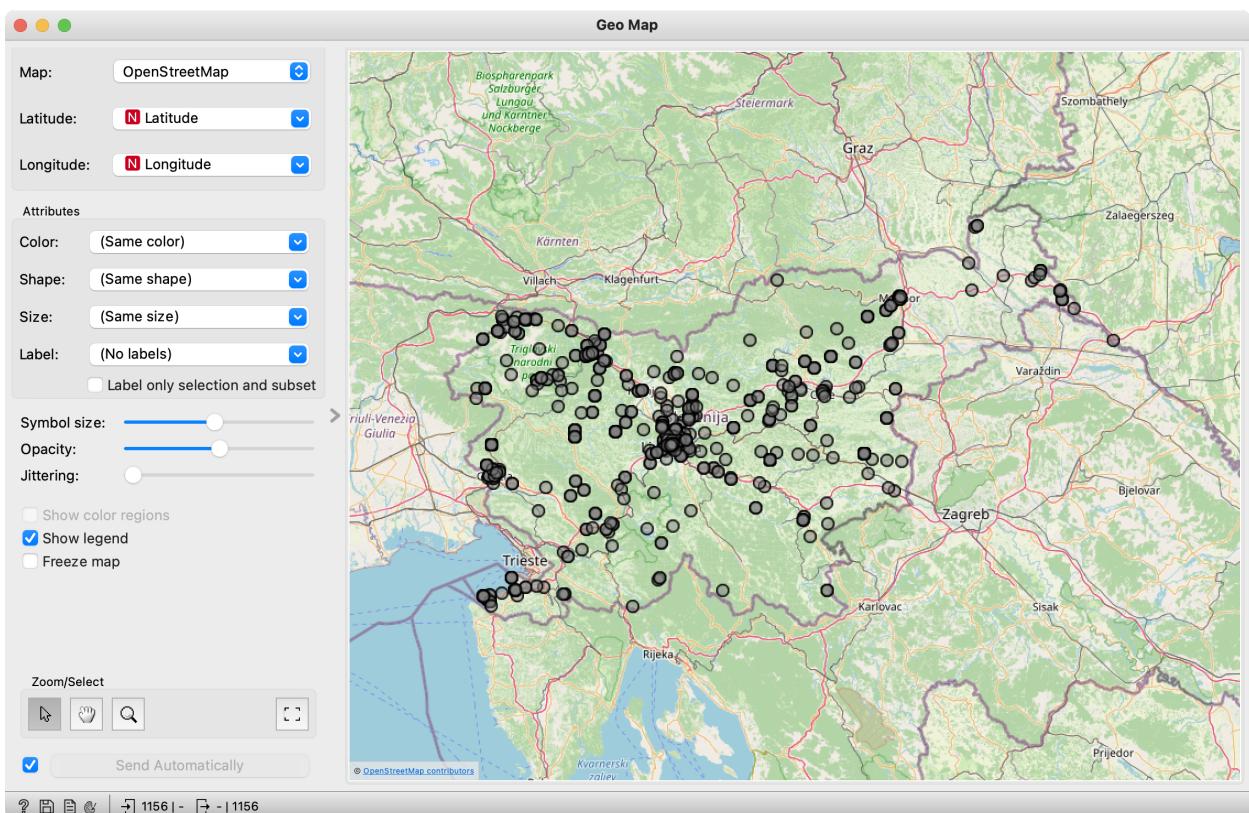
Ste vedeli, da nekateri tviti vsebujejo oznako lokacije? Prav zares, nekateri imajo oznako geografske dolžine in širine, ki kaže na to, kje je bil tvit objavljen! Za konec tečaja si bomo pogledali, kako raziskati geolocirane tvite!

Z omrežja smo pridobili podatke za prvih nekaj dni leta 2017. Vsi tviti so bili objavljeni v Sloveniji, napisani pa so v različnih jezikih!

Prenesite slo-geo-tweets.tab z <http://file.biolab.si/text/slo-geo-tweets.tab> in poglejte podatke v gradniku *Corpus Viewer*.

Ta korpus je precej poln informacij! Ne samo da imamo besedilo, število všeckov in čas objave, imamo tudi lokacijo in slike. Kakšno obilje informacij! Vsak tvit bomo prebrali posebej in...

Seveda ne bomo naredili tega! Do sedaj že vemo, da obstajajo hitrejši načini raziskovanja podatkov. Lahko jih na primer prikažemo na zemljevidu. Povežite Corpus z *Geo Map* - gradnik bo avtomatsko našel stolpca latitude in longitude in prikazal podatke.



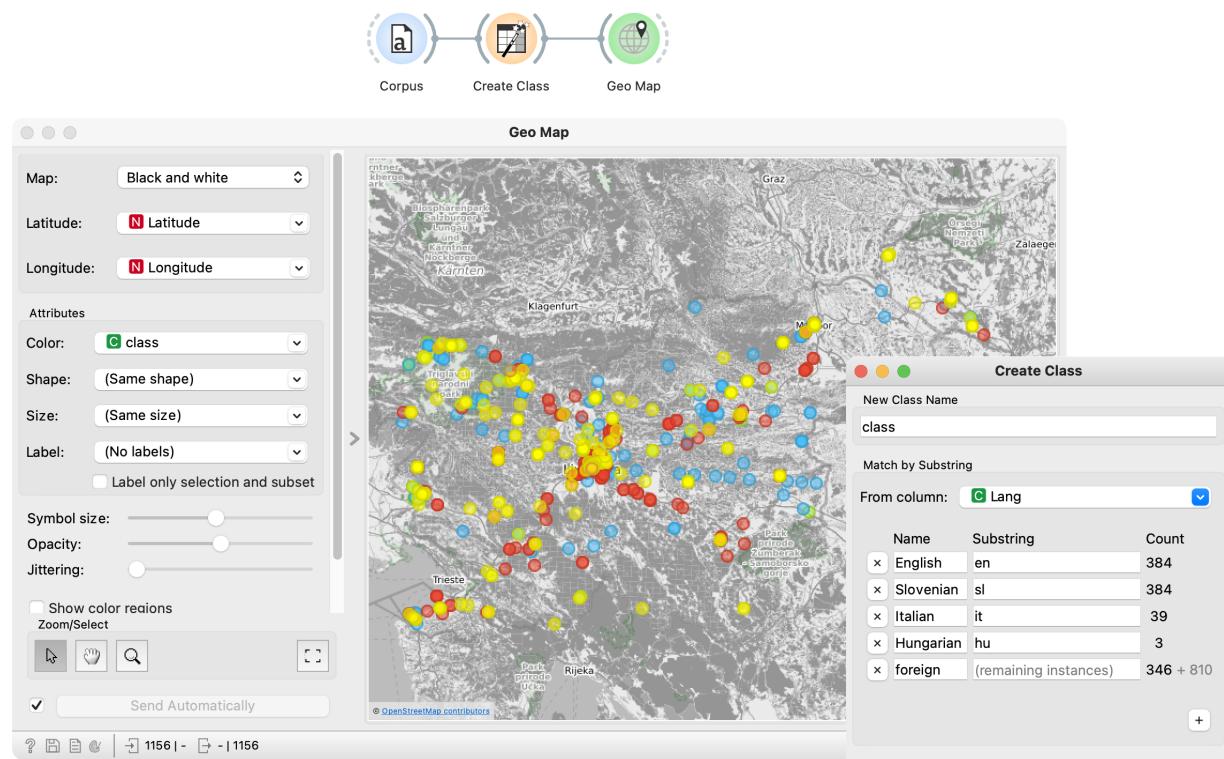
Dobro, sedaj vemo, da večina tvitov prihaja iz Slovenije in da so bolj pogosti v severo-zahodnem delu države.

Zanimiv podatek, ki smo ga pridobili, je, v katerem jeziku so bili

tviti napisani. Ampak jezikov je ogromno! Zato moramo poenostaviti interpretacijo.

Corpus povežite z gradnikom *Create Class*. Create Class ustvari novo spremenljivko iz obstoječih. Vrednosti bomo definirali v gradniku. Na primer, označiti želimo vse angleške tvite, zato bomo uporabili oznako 'English', če ima jezik vrednost 'en'. Za slovenščino bomo uporabili 'Slovenian' za jezik 'sl' in tako naprej. Za vse druge jezike bomo uporabili oznako 'foreign'.

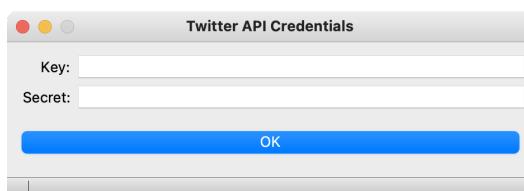
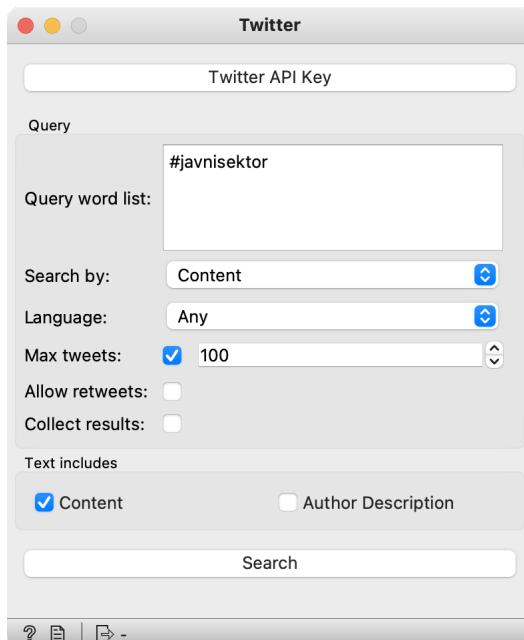
Sedaj povežite Geo Map s Create Class. Naši geolocirani tviti bodo obarvani glede na jezik. To je že precej bolj zanimivo!



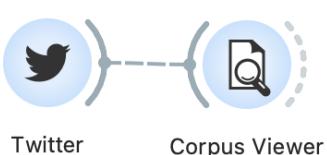
Moji podatki

Grimove pravljice, volitve 2016 in slovenski geolocirani tviti so že pripravljeni podatki. Dodatek Text lahko nalaga podatke tudi iz drugih virov: Twitterja, Guardiana, New York Timesa in Wikipedie!

Spodaj je primer, kako uporabiti gradnik Twitter.



Za uporabo gradnika Twitter boste morali ustvariti svoj API ključ. Pojdite na <https://apps.twitter.com/> in ustvarite novo aplikacijo. S tem boste prejeli tudi svoj lasten API ključ. Vpišite podatke v okno Twitter API Key in pričnite z uporabo gradnika.



Odprite gradnik *Twitter*. Prenesli bomo sto angleških tvitov z oznako #javnisektor. Iskalni ukaz smo vnesli v polje *Query word list* in nastavili jezik na angleščino.

Ko pritisnemo gumb *Search* bo gradnik poiskal podatke in jih dal na izhod. Povežite Twitter z gradnikom Corpus Viewer in poglejte, kaj smo prenesli.

Twitter (in drugi podobni gradniki) omogočajo, da sami ustvarite nabor podatkov in jih uporabite za analizo. Preizkusite katerega izmed delotokov na novih podatkih!

Literatura

Stvarno kazalo

license, [2](#)