

Paramount TV Shows and Movies

Samostalni projekat za predmet “Istraživanje Podataka 1”

Student: Mila Lukić, 222/2018 Asistent: Stefan Kapunac Profesor: Nenad Mitić

Sadržaj

Uvod.....	3
Skup Podataka.....	3
Titles datoteka.....	4
Credits Datoteka.....	6
Pretprocesiranje.....	6
Čišćenje podataka - Titles datoteka.....	7
Čišćenje podataka - Credits datoteka.....	8
Klasifikacija.....	9
KNN Algoritam.....	9
Pretprocesiranje.....	10
Transformacija Podataka.....	10
Odabir Karakteristika.....	10
Skaliranje.....	10
Treniranje modela i rezultati.....	10
Tačnost Modela.....	11
Matrica Konfuzije.....	11
Optimizacija KNN Algoritma.....	12
GridSearchCV.....	12
Bagging Classifier.....	14
Stabla odlučivanja.....	15
Stablo odlučivanja bez balansiranja klasa.....	16
Optimizacija Stabla Odlučivanja uz GridSearchCV.....	16
Balansirani model.....	17
Nebalansirani model.....	18
Balansiranje klasa pomoću kombinacije oversamplinga i undersamplinga - SMOTEENN.....	19
Random Forest Classifier.....	20
Poređenje modela pomoću ROC krive.....	21
Klasterovanje.....	23
Algoritam K sredina.....	23
Pretprocesiranje.....	23
Rezultati.....	24
Algoritam Sakupljajućeg Klasterovanja.....	24
Rezultati.....	25
Poređenje modela.....	26
Dominantno Nisko Rangiran klaster.....	26
Dominantno Srednje Rangiran klaster.....	27
Dominantno Visoko Rangiran klaster.....	27
Pravila Pridruživanja - SPSS.....	28
Apriori Algoritam.....	28

Pretprocesiranje - Python.....	28
Pretprocesiranje - SPSS.....	29
Pokretanje algoritma.....	32
Rezultati.....	33
Zaključak.....	34

Uvod

Svrha ovog projekta je demonstriranje algoritama:

- Klasifikacije (KNN - K Next Neighbours, Algoritmi stabala odlučivanja)
- Klasterovanja (Algoritam K-sredina, Algoritam Sakupljajućeg Klasterovanja)
- Pravila Pridruživanja u SPSS-u (Apriori Algoritam)

Skup podataka sa kojim je rađeno je "Paramount TV Shows and Movies", koji se može naći na sledećem linku:

<https://www.kaggle.com/datasets/victorsoeiro/paramount-tv-shows-and-movies>

Algoritmi klasifikacije i klasterovanja, kao i pretprocesiranje skupa podataka rađeni su u okruženju Jupyter Notebook, dok je Apriori algoritam (Pravila pridruživanja) rađen u IBM SPSS Modeleru.

Kod ovog projekta može se naći u repozitorijumu predmeta Istraživnje Podataka 1 za 2023. godinu:

https://github.com/MATF-istrazivanje-podataka-1/2023_Data_Mining_Artificial_Characters_Dataset

Skup Podataka

Kao što je već pomenuto, ovaj projekat je rađen nad skupom podataka "Paramount TV Shows and Movies".

Ovaj skup podataka se sastoji od dve datoteke:

1. titles.csv
2. credits.csv

U nastavku ovog poglavlja opisaću obe datoteke, a zatim i podeliti detalje o tome kako sam ih menjala kako bi bile pogodnije za primenu algoritama klasifikacije, klasterovanja i pravila pridruživanja.

Titles datoteka

Ovaj skup podataka sadrži 2825 jedinstvenih naslova Paramountovih filmova i serija, kao i 15 kolona (atributa) koje nam ga bolje opisuju:

- **id**: ID naslova na JustWatch platformi
- **title**: Naslov filma, odnosno serije
- **show_type**: Indikator da li je u pitanju film ili serija - moguće vrednosti su SHOW i MOVIE
- **description**: Kratak opis naslova
- **release_year**: Godina premijere filma, odnosno serije
- **age_certification**: Naznaka o preporučenoj starosnoj dobi gledaoca
- **runtime**: Dužina epizode serije, odnosno dužina filma
- **genres**: Lista žanrova
- **production_countries**: Lista zemalja u kojima je rađeno na filmu
- **seasons**: Broj sezona (ukoliko je u pitanju SHOW tip)
- **imdb_id**: ID naslova na IMDBu
- **imdb_score**: Ocena na IMDBu
- **imdb_votes**: Glasovi na IMDBu
- **tmdb_popularity**: Popularnost na TMDBu
- **tmdb_score**: Ocena na TMDBu

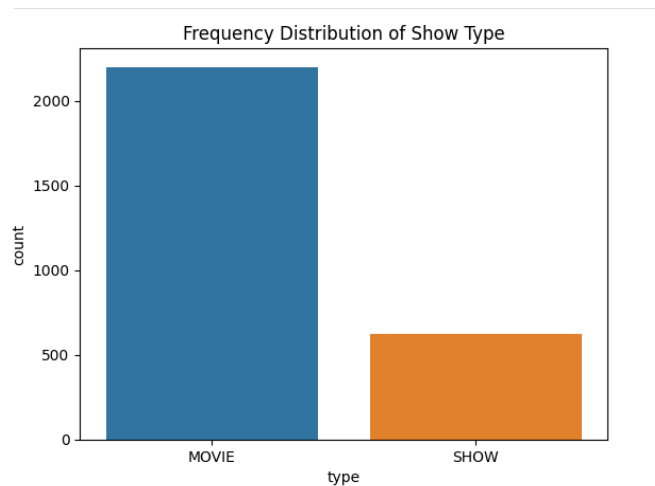
Tokom eksplorativne analize ovog skupa, saznali smo sledeće:

- U skupu postoji 110 različitih godina premijera filmova, u rasponu od 1912. do 2022. godine.
- Broj sezona se kreće od 1 do 49, sa ukupno 31 jedinstvenim brojem sezona. Ovo polje je NaN ukoliko se radi o filmu, čime bi trebalo da se pozabavi u preprocesiranju.
- Srednje vrednosti svih mera ocene u ovom skupu nam pokazuju da će nam biti neophodno skaliranje ovih atributa ukoliko ih budemo koristili u budućnosti:

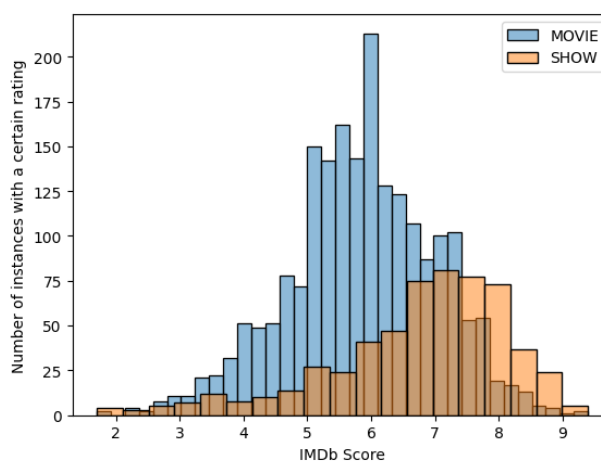
Out[9]:

	Attribute	Mean	Median	Mode
0	imdb_score	6.042863	6.1000	5.6
1	imdb_votes	22238.755454	548.0000	10.0
2	tmdb_popularity	13.043838	2.6205	0.6
3	tmdb_score	6.045528	6.1000	6.0

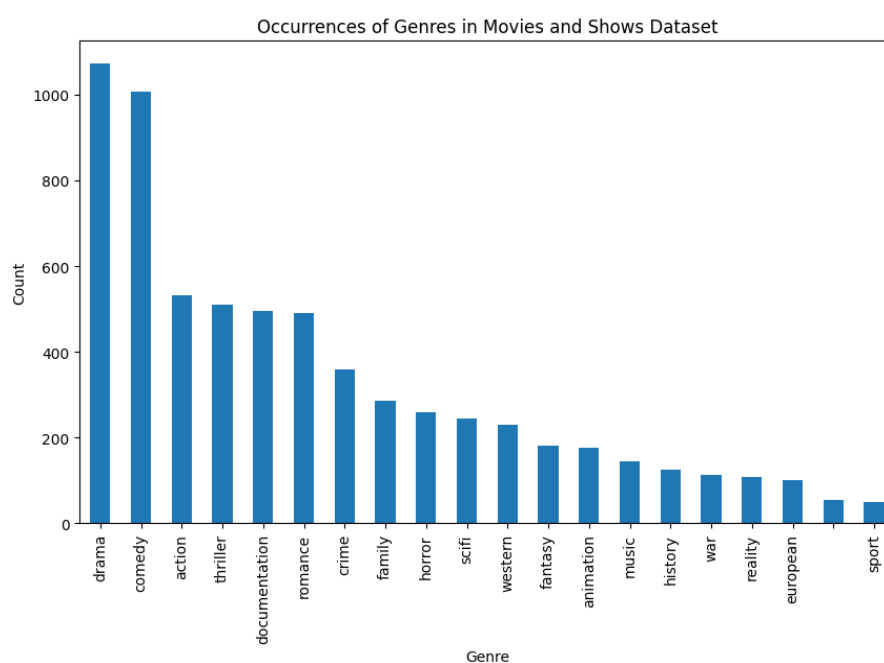
- Kada je u pitanju odnos između filmova i serija u skupu, bavimo se prilično nebalansiranim klasama, što ćemo takođe pokušati da ispravimo:



- Serije generalno imaju bolje ocene na IMDBu u odnosu na filmove: iako filmova ima više ukupno, imamo veći broj serija sa ocenom između 7.5 i 10:



- Drame, komedije i akcija su ubedljivo najčešći žanrovi u ovom skupu podataka



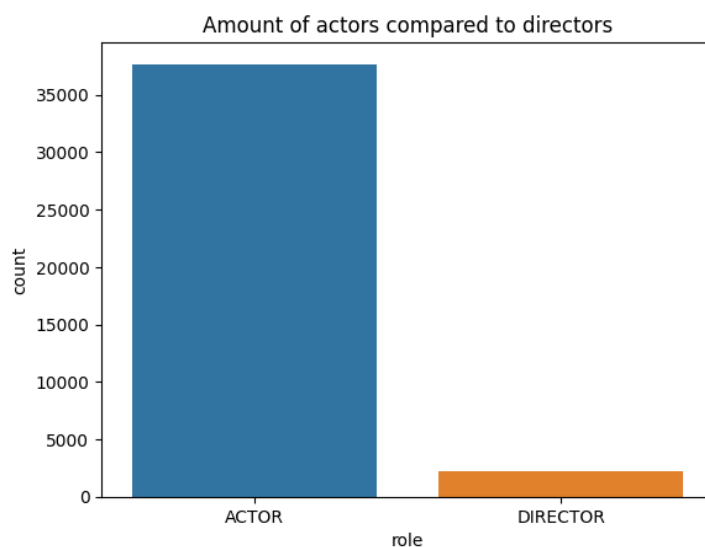
Credits Datoteka

Ovaj skup podataka sadži 39842 glumaca i režisera koji su radili na gorepomenutim naslovima, kao i 5 atributa koji nam ih bolje opisuju:

- person_ID: ID osobe na JustWatch platformi
- id: ID naslova na JustWatch platformi (preko ovoga se mogu povezati sa Titles datotekom)
- name: Ime glumca, odnosno režisera
- character_name: Ime lika kojeg glumac glumi u naslovu
- role: Indikator da li je u pitanju glumac ili režiser (moguće vrednosti su ACTOR i DIRECTOR)

Tokom eksplorativne analize ovog skupa, saznali smo sledeće:

- Ponovo imamo posla sa nebalansiranim klasama: glumaca ima mnogo više nego režisera:



- Prosečna IMDB ocena za 10 najbolje rangiranih glumaca se kreće između 7.3 i 8.1, dok se prosečna IMDB ocena za 10 najbolje rangiranih režisera kreće između 6.6 i 7.6

Pretprocesiranje

Pretprocesiranje podataka podrazumeva:

- Odabir relevantnih atributa za određeni model
- Prenošnje jednog tipa podatka u drugi
- Čišćenje podataka
- Redukciju i transformaciju podataka

Ovi koraci nisu univerzalni, i u nastavku ćemo raditi pojedinačna, specifična pretprocesiranja za svaki algoritam koji budemo implementirali.

Međutim, postoje neki koraci koje bismo ponavljali u svakom pojedinačnom pretprocesiranju, tako da su oni odrađeni unapred, odmah nakon eksplorativne analize.

Čišćenje podataka - Titles datoteka

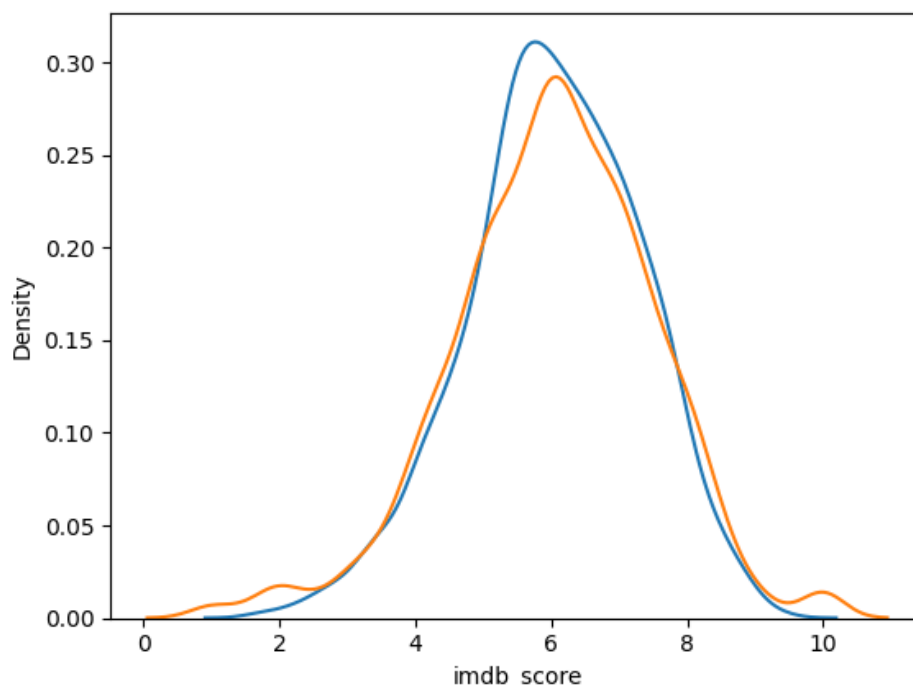
Ovo su neke od akcija koje smo preduzeli da bismo “sredili” ovaj skup podataka:

- Kako svi filmovi imaju NaN za broj sezona, postavili smo taj broj na 0.0
- Izbacili smo kolonu “Age Certification”, jer nas nije interesovala za dalji rad, a imala je i mnogo nedostajućih vrednosti
- Izbacili smo sve instance koje nemaju “imdb_id”, kao i “description” - nije ih bilo puno, pa nam nije pravilo preveliku razliku.

Najveći izazov predstavljao je obradu nedostajućih vrednosti kada su u pitanju IMDB ocene - ovo je atribut koji nam je u budućnosti ključan, tako da ne bismo smeli da imamo nedostajuće vrednosti u ovoj koloni.

Pretpostavili smo da je IMDB ocena u opštem slučaju približno slična TMDB oceni nekog filma, odnosno serije. Ukoliko bi se ovo ispostavilo kao tačno, mogli bismo da popunimo nedostajuće vrednosti IMDB ocena njihovim odgovarajućim TMDB ocenama.

Korelacija između IMDB ocene i TMDB ocene je prilično visoka (oko 0.6, dok je idealna korelacija 1.0). Pored toga, treba proveriti i da li je raspodela ove dve promenljive slična. To smo uradili korišćenjem grafika:



Kako imamo visoku korelaciju i gotovo identičnu raspodelu, možemo zameniti nedostajuće IMDB ocene odgovarajućim TMDB ocenama. Instance koje nismo mogli da popunimo na ovaj način smo popunili prosečnom IMDB ocenom.

Čišćenje podataka - Credits datoteka

Jedine nedostajuće vrednosti sa kojim smo se susreli u ovom skupu podataka bile su 3113 nedostajućih imena likova. Nakon što smo utvrdili da su u pitanju većinski instance režisera (koji, logično, nemaju ime lika u filmu jer ne glume), samo smo popunili te nedostajuće vrednosti vrednošću "director".

Preostale instance smo prosto izbacili iz skupa jer ih nije bilo mnogo (913).

Klasifikacija

Klasifikacija je problem određivanja ciljne funkcije f koja preslikava skup atributa X u neku od unapred određenih oznaka klasa y .

Ciljna funkcija se obično naziva klasifikacionim modelom.

Prilikom klasifikacije, podaci se dele na trening skup, skup za proveru i test skup. Nekada se skup za proveru (validacioni skup) izostavlja.

Trening skup služi za obucavanje modela koji ćemo koristiti za klasifikaciju.

Test skup služi da proverimo koliko je naš model dobar na podacima koje nikad pre nije video

Skup za proveru služi za odabir različitih parametara modela ili modela uopšte.

U ovom projektu, skup za proveru nije korišćen.

Kako bismo predstavili KNN algoritam i algoritam stabala odlučivanja, odlučili smo da klasifikujemo glumce i režisere u jednu od tri klase:

- Visoko Rangirani (High Rated) - glumci/režiseri čija je prosečna IMDB ocena između 7 i 10
- Srednje Rangirani (Medium Rated) - glumci/režiseri čija je prosečna IMDB ocena između 4 i 7
- Nisko Rangirani (Low Rated) - glumci/režiseri čija je prosečna IMDB ocena između 0 i 4

Prosečna IMDB ocena je za svakoga izračunata na osnovu svih filmova u kojima su glumili/režirali.

KNN Algoritam

KNN - K-Nearest Neighbors (K Najbližih Suseda) Algoritam je jedan od najpoznatijih algoritama klasifikacije.

Za ovaj algoritam neophodni su nam:

- Skup sacuvanih slogova (trening skup)
- Metrika kojom merimo rastojanje između instanci (obično se koristi Euklidsko rastojanje)
- Broj suseda - k

Opis algoritma:

1. Izračunati rastojanje test instance do svih instanci iz trening skupa
2. Odrediti k najbližih suseda
3. Glasanjem utvrditi klasu test instance

Pretprocesiranje

Kao što je već napomenuto, iako smo izvršili “uopšteno” pretprocesiranje, neophodno je i uraditi “specifično” pretprocesiranje za svaki algoritam.

Transformacija Podataka

Prva stvar koju smo morali uraditi da bismo uspešno izvršili zadatak klasifikacije bila je spajanje dva skupa podataka. Ovo smo uradili “merge” funkcijom pomoću atributa “id” koji predstavlja vezu između ova dva skupa.

Zatim smo za svaku osobu (svaki person_id) izračunali prosečne vrednosti svakog numeričkog atributa, to jest “release_year”, “runtime”, “seasons” i “imdb_score”.

Iskoristili smo prosečnu IMDB ocenu da napravimo novi atribut, “imdb_score_bin”, koji klasifikuje svaku instancu glumca/režisera kao visoko, srednje i nisko rangiranog (objašnjeno u uvodu).

Odabir Karakteristika

Za izgradnju modela, izabrali smo prosečnu godinu premijere filmova na kojim je određen glumac/režiser radio, prosečan broj sezona, kao i dužinu trajanja filma/epizode serije.

Cilj nam je da na osnovu ovih atributa, naš KNN model uspešno klasifikuje određenog glumca ili režisera kao visoko, srednje ili nisko rangiranog.

Skaliranje

Nakon odvajanja karakteristika koje ćemo iskoristiti za treniranje i testiranje modela, skalirali smo ih koristeći MinMaxScaler.

MinMaxScaler se koristi za proces normalizacije - svođenje atributa na interval [0, 1]

Treniranje modela i rezultati

Nakon pretprocesiranja, KNN model smo istrenirali na trening skupu, i zatim razmotrili koliko precizne rezultate dobijamo.

Dve glavne metrike koje smo pratili bile su Tačnost (Accuracy) i matrica konfuzije (confusion matrix)

Tačnost Modela

Tačnost nekog modela računa se kao procenat korektno klasifikovanih instanci.

Ovaj model je dao **tačnost** 92% na test skupu.

Classification report for model KNeighborsClassifier on test data

	precision	recall	f1-score	support
High-Rated	0.84	0.82	0.83	1764
Low-Rated	0.64	0.66	0.65	336
Medium-Rated	0.94	0.95	0.95	7527
accuracy			0.92	9627
macro avg	0.81	0.81	0.81	9627
weighted avg	0.92	0.92	0.92	9627

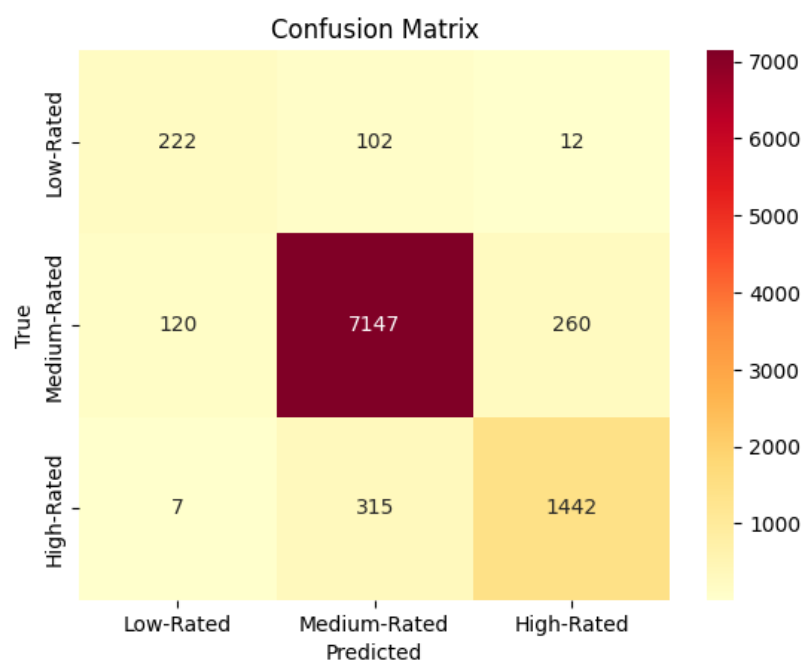
Matrica Konfuzije

Vertikalna osa matrice konfuzije predstavlja realnu klasu određenih instanci, dok horizontalna predstavlja klasu koja im je dodeljena od strane KNN klasifikatora.

Polje M_{ij} matrice konfuzije predstavlja broj instanci klase i koje je klasifikator procenio kao klasu j .

Idealan slučaj bi bio da matrica konfuzije ima što veće brojeve na dijagonali, dok su preostala polja matrice 0.

Ovo je matrica konfuzije za naš model:



Možemo primetiti da najveće vrednosti zaista imaju polja na dijagonali matrice. Međutim, postoje optimizacije koje možemo primeniti da ovi rezultati budu još bolji.

Optimizacija KNN Algoritma

GridSearchCV

GridSearchCV je tehnika za podešavanje hiperparametara u mašinskom učenju, koja uključuje traženje najboljeg skupa hiperparametara koji rezultuju optimalnom performansom modela.

U GridSearchCV-u, skup hiperparametara i njihove odgovarajuće vrednosti su unapred definisani, a algoritam procenjuje performansu modela za svaku kombinaciju ovih hiperparametara.

Procena se obično vrši pomoću unakrsne validacije, gde se skup podataka deli na nekoliko preklapljenih delova, a model se trenira na jednom delu podataka i testira na preostalom delu. Ovo pomaže u izbegavanju preprilagođavanja i pruža pouzdaniju procenu performanse modela.

GridSearchCV izvršava iscrpnu pretragu svih mogućih kombinacija hiperparametara i vraća kombinaciju koja daje najbolju performansu na validacionim podacima. Metrika performanse koja se koristi za evaluaciju može biti specificirana od strane korisnika i može se razlikovati u zavisnosti od konkretnog problema.

Pre pozivanja GridSearchCV-a, definisali smo jednu mapu:

```
params_grid = {'n_neighbors': range(10, 50, 5),
               'weights': ['uniform', 'distance'],
               'p': [1, 2]}
```

Ova mapa nam daje opcije za sledeće hiperparametre za KNN klasifikator:

- 'n_neighbors' sa opsegom od 10 do 50 (isključujući 50) sa korakom 5 (Ovo predstavlja koji broj K ćemo pokušavati da uzmemo)
- 'weights' sa opcijama:
 - "uniform" znači da će svaki sused imati podjednak uticaj na klasifikaciju novog uzorka.
 - "distance" znači da će bliži susedi imati veću težinu, dok će udaljeni susedi imati manju težinu. Ova opcija uzima u obzir udaljenost suseda prilikom klasifikacije i može biti korisna kada su bliži susedi relevantniji za klasifikaciju.
- 'p' sa opcijama:
 - 1 - za rastojanje između suseda koristiće se Menhetn rastojanje
 - 2 - za rastojanje između suseda koristiće se Euklidsko rastojanje

Nakon pokretanja GridSearchCV-a, dobijamo sledeće “idealne” hiperparametre:

```
{'n_neighbors': 45, 'p': 2, 'weights': 'distance'}
```

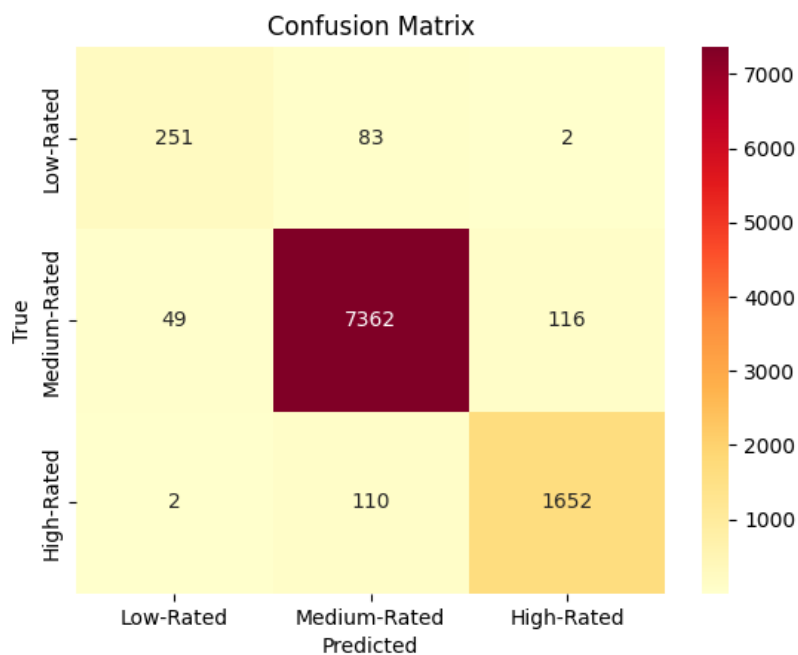
kao i procenu za tačnost: 0.9582394574680647

Rezultati KNN algoritma uz GridSearchCV optimizaciju hiperparametara:

Classification report for model KNeighborsClassifier on test data

	precision	recall	f1-score	support
High-Rated	0.93	0.94	0.93	1764
Low-Rated	0.83	0.75	0.79	336
Medium-Rated	0.97	0.98	0.98	7527
accuracy			0.96	9627
macro avg	0.91	0.89	0.90	9627
weighted avg	0.96	0.96	0.96	9627

Tačnost algoritma sa optimizovanim hiperparametrima je 96%



Primećujemo i da je matrica konfuzije bolja od prethodne.

Bagging Classifier

Bagging klasifikator, takođe poznat kao bootstrap agregacija, je vrsta algoritma za učenje ansambla koji kombinuje više modela drveća odlučivanja radi poboljšanja ukupne performanse zadatka klasifikacije.

Tokom faze predviđanja, svaki model drveća odlučivanja daje sopstveni rezultat klasifikacije, a konačna klasifikacija se određuje agregiranjem rezultata svih pojedinačnih modela. Ovo agregiranje se može obaviti tako što se uzima većinsko glasanje, gde se klasifikacija sa najviše glasova bira kao konačno predviđanje.

Kao parametre za ovaj klasifikator smo takođe koristili optimizovane parametre dobijene GridSearchCV-em.

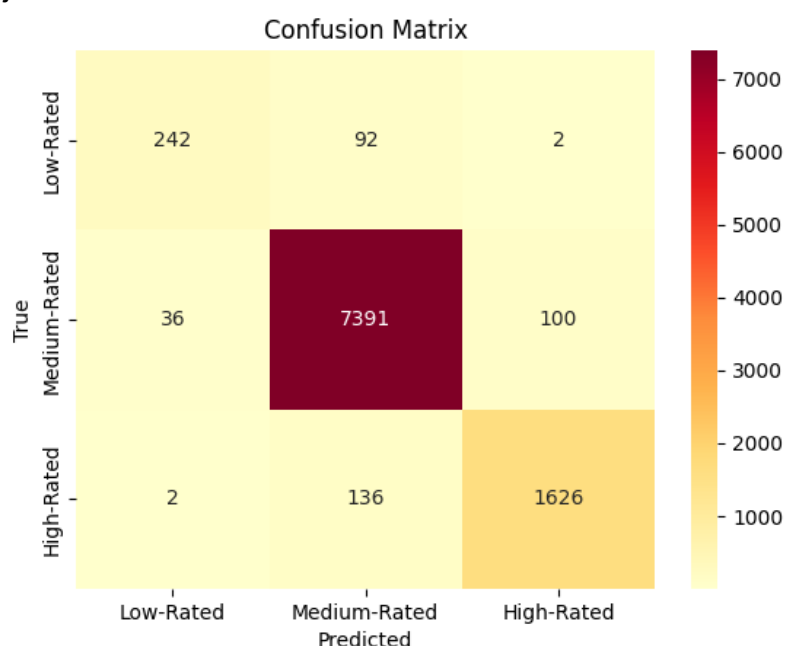
Rezultati:

Classification report for model BaggingClassifier on test data

	precision	recall	f1-score	support
High-Rated	0.94	0.92	0.93	1764
Low-Rated	0.86	0.72	0.79	336
Medium-Rated	0.97	0.98	0.98	7527
accuracy			0.96	9627
macro avg	0.93	0.87	0.90	9627
weighted avg	0.96	0.96	0.96	9627

Tačnost je takođe 96%, kao i kod obične GridSearchCV optimizacije.

Matrica konfuzije:



Ova matrica konfuzije je slična prethodnoj. Nakon što obradimo sve algoritme klasifikacije, uporedićemo ih i pomoću ROC krive, što će nam pomoći pri proceni najboljeg klasifikatora.

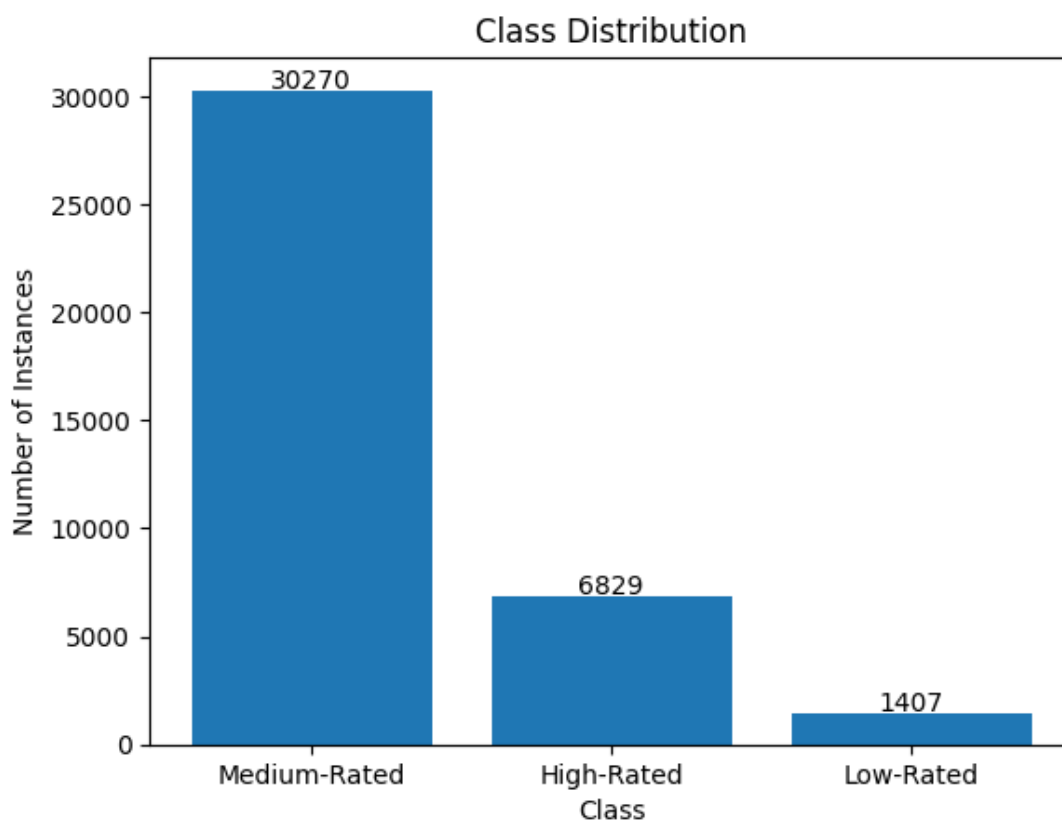
Stabla odlučivanja

Stabla odlučivanja su popularna metoda za rešavanje problema klasifikacije. U pitanju je vrsta algoritma nadgledanog učenja koji uči da donosi odluke konstruišući model odluka u obliku drveta i njihovih mogućih posledica.

U stablu odlučivanja, svaki čvor predstavlja odluku, a svaka grana predstavlja mogući ishod te odluke. Drvo se konstruiše rekurzivno particionišući podatke u podskupove na osnovu vrednosti ulaznih atributa, sa ciljem minimiziranja neke mere nečistoće ili entropije. Pri svakom čvoru, algoritam bira karakteristiku koja najbolje razdvaja podatke u različite klase i deli podatke na odgovarajući način.

Pretprocesiranje skupa podataka za ovaj model izvršeno je identično kao pretprocesiranje za KNN model.

Jedna razlika u pretprocesiranju je bila u tome što smo proverili balansiranost klasa. Primećujemo da klasa Medium-Rated glumaca i režisera ima značajno više instanci od druge dve klase.



Pristupićemo ovom problemu na dva načina: sa balansiranjem klasa i bez balansiranja klasa.

Stablo odlučivanja bez balansiranja klasa

Koristimo istu podjelu test i trening setova kao u prethodnom primeru (da bismo kasnije mogli da realno poredimo modele).

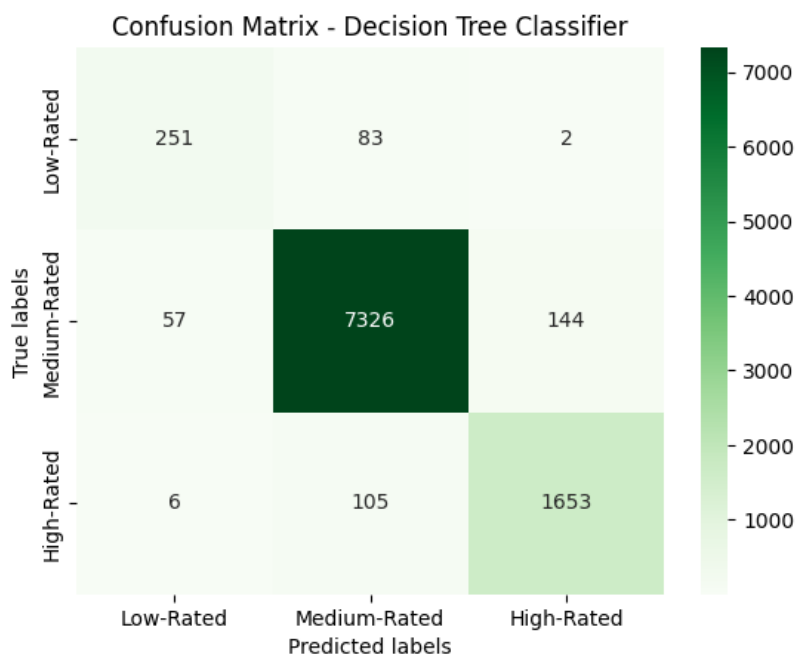
Rezultati klasifikacije ovog algoritma:

Classification report for model DecisionTreeClassifier on test data

	precision	recall	f1-score	support
High-Rated	0.92	0.94	0.93	1764
Low-Rated	0.80	0.75	0.77	336
Medium-Rated	0.97	0.97	0.97	7527
accuracy			0.96	9627
macro avg	0.90	0.89	0.89	9627
weighted avg	0.96	0.96	0.96	9627

Vidimo da je tačnost 96%, dakle identična kao kod prethodno optimizovanog KNN algoritma.

Matrica konfuzije nam takođe daje odlične rezultate:



Optimizacija Stabla Odlučivanja uz GridSearchCV

Baš kao i za KNN, koristićemo GridSearchCV da podešavamo hiperparametre koji dovode do optimalnog rezultata ovog modela.

Pre pozivanja GridSearchCV-a, definisali smo mapu, kao i u KNN-u:

```
params = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [x for x in range(3, 30)],
    'max_features': [x for x in range(3, 40)]
}
```

Ova mapa nam daje opcije za sledeće hiperparametre za KNN klasifikator:

- 'criterion' sa opcijama:
 - gini - mere nečistoće se izračunavaju računanjem verovatnoće netačne klasifikacije nasumično izabranog podatka u tom čvoru.
 - entropy - mere nečistoće se računaju na osnovu verovatnoća pojavljivanja različitih klasa u čvoru.
- 'max_depth' sa nizom brojeva od 3 do 29 - određuje dubinu drveta
- 'max_features' sa nizom brojeva od 3 do 39 - određuje količinu ulaznih atributa koju uzimamo u obzir pri podeli čvora

Pozvali smo GridSearchCV za dva različita modela - jedan od njih sam balansira klase (class_weight = 'balanced'), dok drugi to ne radi.

Balansirani model

Nakon pokretanja GridSearchCV-a, dobijamo sledeće "idealne" hiperparametre:

```
{'criterion': 'gini', 'max_depth': 27, 'max_features': 38}
```

kao i procenu za tačnost: 0.9354201232746939

Izgleda da balansiranje klasa nije pomoglo, jer je ova procena lošija od prethodne, neoptimizovane procene.

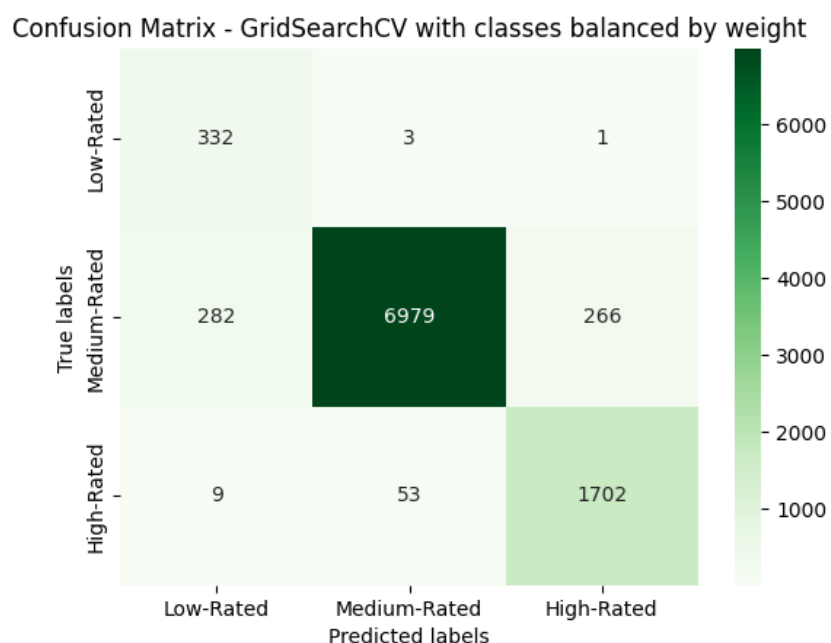
Rezultati algoritma stabla odlučivanja uz GridSearchCV optimizaciju hiperparametara sa balansiranjem klasa:

Classification report for model DecisionTreeClassifier on testing data

	precision	recall	f1-score	support
High-Rated	0.86	0.96	0.91	1764
Low-Rated	0.53	0.99	0.69	336
Medium-Rated	0.99	0.93	0.96	7527
accuracy			0.94	9627
macro avg	0.80	0.96	0.85	9627
weighted avg	0.95	0.94	0.94	9627

Vidimo da je tačnost malo lošija.

Matrica konfuzije nam takođe pokazuje manje precizne rezultate:



Nebalansirani model

Nakon pokretanja GridSearchCV-a, dobijamo sledeće “idealne” hiperparametre:

```
{'criterion': 'entropy', 'max_depth': 29, 'max_features': 38}
```

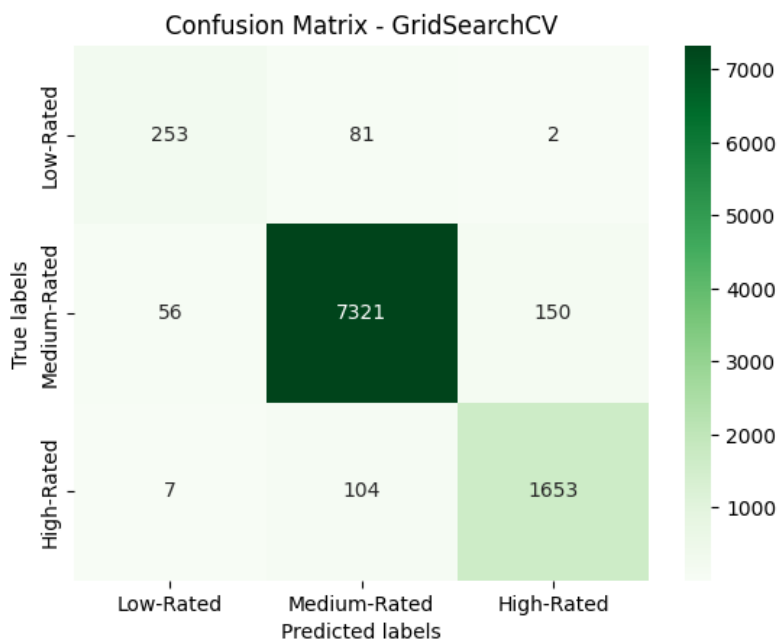
kao i procenu za tačnost: 0.9532531628113345

Rezultati algoritma stabla odlučivanja uz GridSearchCV optimizaciju hiperparametara bez balansiranja klasa:

Classification report for model DecisionTreeClassifier on test data

	precision	recall	f1-score	support
High-Rated	0.91	0.94	0.93	1764
Low-Rated	0.80	0.75	0.77	336
Medium-Rated	0.98	0.97	0.97	7527
accuracy			0.96	9627
macro avg	0.90	0.89	0.89	9627
weighted avg	0.96	0.96	0.96	9627

Matrica konfuzije nam pokazuje da je ovo najbolji rezultat stabala odlučivanja do sada:



Balansiranje klasa pomoću kombinacije oversamplinga i undersamplinga - SMOTEENN

SMOTEEN je kombinacija tehnika sintetičkog oversamplovanja manjinske klase (SMOTE) i uređivanja najbližih suseda (ENN) koje se koriste za balansiranje neizbalansiranih klasa u problemima klasifikacije.

SMOTE stvara sintetičke uzorke manjinske klase interpoliranjem između postojećih uzoraka. Ovo pomaže u balansiranju raspodele klasa i izbegava preprilagođavanje većinskoj klasi. Međutim, može stvoriti i šumovite uzorke ako se sintetički uzorci generišu preblizu postojećim uzorcima.

ENN uklanja uzorke iz većinske klase koji su previše blizu uzorcima manjinske klase. Ovo pomaže uklanjanju šumovitih uzoraka i poboljšava kvalitet uzoraka većinske klase. Međutim, može takođe ukloniti informativne uzorke i dovesti do nedopasnosti.

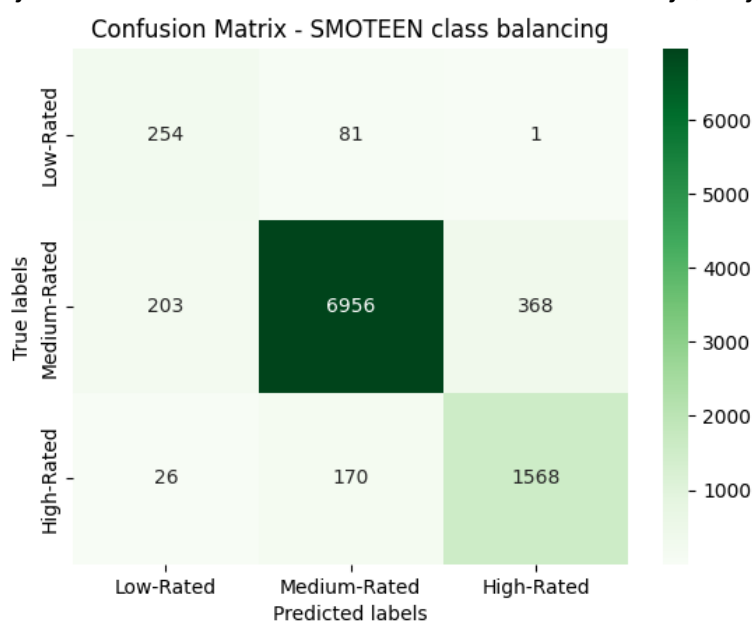
SMOTEEN kombinuje ove dve tehnike tako što prvo primenjuje SMOTE za oversamplovanje manjinske klase, a zatim koristi ENN za čišćenje rezultirajućeg skupa podataka uklanjanjem šumovitih uzoraka kako iz manjinske tako i iz većinske klase koji su blizu jedni drugima. Ovo pomaže u balansiranju klasa i poboljšava kvalitet skupa podataka.

Rezultati ove klasifikacije:

Classification report for model DecisionTreeClassifier on test data

	precision	recall	f1-score	support
High-Rated	0.81	0.89	0.85	1764
Low-Rated	0.53	0.76	0.62	336
Medium-Rated	0.97	0.92	0.94	7527
accuracy			0.91	9627
macro avg	0.77	0.86	0.80	9627
weighted avg	0.92	0.91	0.92	9627

Matrica konfuzije je slična GridSearchCV rezultatima bez balansiranja, ali je tačnost lošija:



Random Forest Classifier

Random Forest Classifier je algoritam ansambla. On gradi više drveća odlučivanja na nasumičnim podskupovima skupa podataka, a zatim kombinuje njihova predviđanja kako bi napravio konačno predviđanje.

Algoritam nasumično bira podskup karakteristika za svako drvo odlučivanja kako bi smanjio preprilagođavanje trening podacima. Takođe koristi bootstrap uzorkovanje za stvaranje više verzija skupa podataka, što mu omogućava da uhvati više informacija iz skupa podataka i smanji varijansu modela.

Isprobala sam ovaj algoritam jer se često koristi za probleme sa neizbalansiranim klasama.

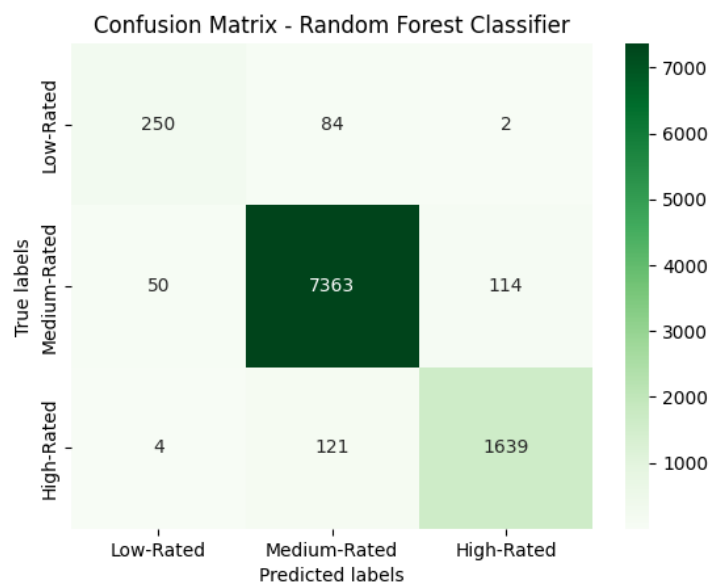
Rezultati algoritma:

Classification report for model RandomForestClassifier on test data

	precision	recall	f1-score	support
High-Rated	0.93	0.93	0.93	1764
Low-Rated	0.82	0.74	0.78	336
Medium-Rated	0.97	0.98	0.98	7527
accuracy			0.96	9627
macro avg	0.91	0.88	0.90	9627
weighted avg	0.96	0.96	0.96	9627

Tačnost ovog algoritma je 96%

Matrica konfuzije:



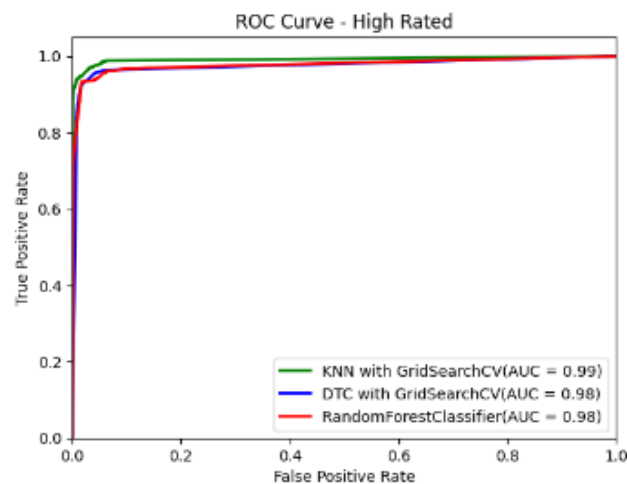
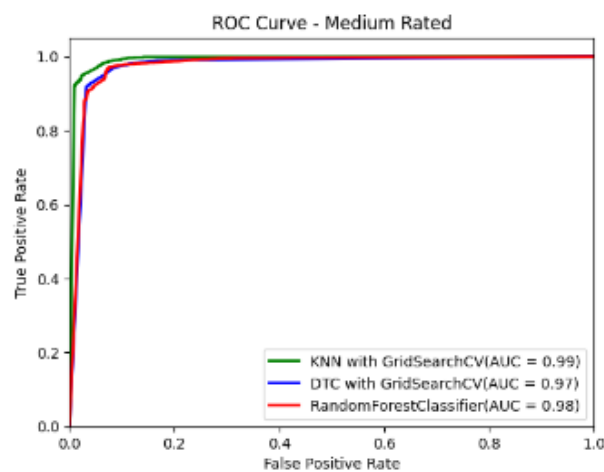
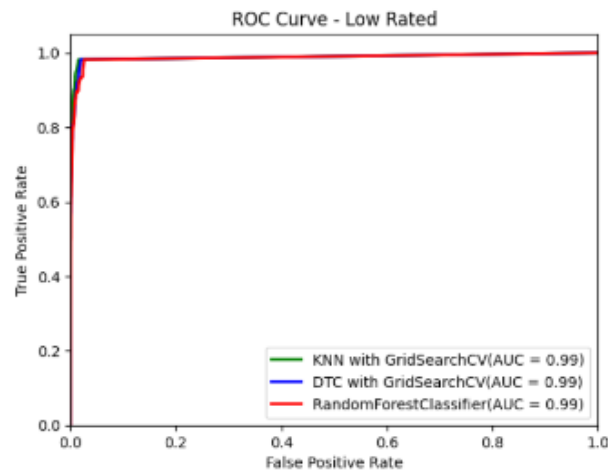
Poređenje modela pomoću ROC krive

Receiver Operating Characteristic - ROC kriva je grafički prikaz odnosa između procenta lažno pozitivnih (False Positive Rate - FPR) i procenta tačno pozitivnih (True Positive Rate - TPR) instanci. Ona prikazuje TPR u odnosu na FPR za različite pragove klasifikacije.

- Procenat lažno pozitivnih (FPR) predstavlja odnos između broja lažno pozitivnih predviđanja i ukupnog broja stvarno negativnih uzoraka. Definiše se kao $FP:(FP+TN)$, gde je FP broj lažno pozitivnih, a TN broj stvarno negativnih.
- Procenat tačno pozitivnih (TPR), takođe poznat kao osetljivost ili odziv, predstavlja odnos između broja tačno pozitivnih predviđanja i ukupnog broja stvarno pozitivnih uzoraka. Definiše se kao $TP:(TP+FN)$, gde je TP broj tačno pozitivnih, a FN broj lažno negativnih.

AUC (Area Under the Curve) je metrika koja predstavlja ukupnu performansu modela u razlikovanju između pozitivnih i negativnih klasa. Ona meri površinu ispod ROC krive, sa vrednostima u rasponu od 0 do 1, gde vrednost 0.5 predstavlja slučajan klasifikator, a vrednost 1 predstavlja savršen klasifikator. Uopšteno, što je veća vrednost AUC, to je bolja performansu modela.

U ovom projektu, poredila sam performanse svih modela sa 3 različite ROC krive, od kojih svaka ocenjuje pojedinačnu klasu.



Zaključujemo da najbolje performanse ima KNN sa GridSearchCV optimizacijom.

Klasterovanje

Klasterovanje je tehnika koja se koristi kako bi se grupisali slični objekti ili podaci na osnovu njihovih karakteristika. Cilj klasterovanja je identifikovanje prirodnih grupa ili obrazaca unutar skupa podataka, pri čemu su objekti unutar iste grupe sličniji jedni drugima nego objektima u drugim grupama.

U klasterovanju, algoritmu nisu pružene unapred definisane oznake ili kategorije. Umesto toga, algoritam analizira podatke i dodeljuje objekte klasterima na osnovu njihove sličnosti. Sličnost između objekata određuje se uzimajući u obzir karakteristike ili atributa. Često korišćeni algoritmi klasterovanja uključuju K-sredina, hijerarhijsko klasterovanje i DBSCAN (klasterovanje na osnovu gustine prostornih podataka sa šumom).

U ovom projektu, klasterovaćemo naslove filmova, odnosno serija na osnovu njihovih ocena (Visoka, Srednja, Niska).

Algoritam K sredina

K-sredina (K-means) je jedan od najčešće korišćenih algoritama klasterovanja u mašinskom učenju i analizi podataka. Cilj mu je da podeli dati skup podataka na k klastera, pri čemu svaka tačka podataka pripada klasteru sa najbližim srednjim vrednostima (centrom). Cilj K-sredina algoritma je da minimizuje zbir kvadratnih udaljenosti između tačaka podataka i njihovih odgovarajućih klaster centara.

Pretprocesiranje

Kao što je već napomenuto, iako smo izvršili "uopšteno" pretprocesiranje, neophodno je i uraditi "specifično" pretprocesiranje za svaki algoritam.

Za ovaj algoritam, želimo da uključimo više atributa. Međutim, potrebno nam je da ti atributi budu predstavljeni brojevima. Zbog toga ćemo enkodirati žanrove i tip naslova pre nego što izdvojimo attribute.

Žanrovi su predstavljeni preko liste, što otežava rad sa njima. Umesto liste, promenili smo tu kolonu tako da predstavlja samo glavni (prvi) žanr.

Zatim smo enkodirali žanrove i tip koristeći Label Encoder.

Klase za klasterovanje smo napravili na isti način kao i klase za klasifikaciju.

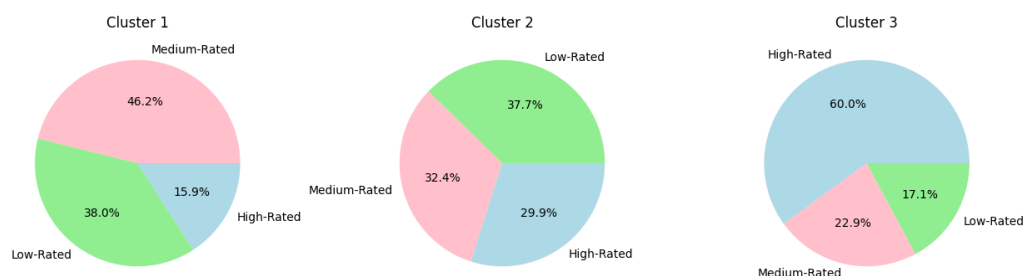
Atributi koje smo izabrali za treniranje modela su:

- 'runtime'
- 'release_year'
- 'seasons'
- 'encoded_type'
- 'encoded_genre'

Ove attribute smo skalirali korišćenjem Standard Scalera.

Rezultati

Ovako izgledaju klasteri nakon podele algoritmom K sredina:



Vidimo da klasteri nisu u potpunosti čisti, ali:

1. Prvi klaster ima najveći procenat Medium Rated klase
2. Drugi klaster ima najveći procenat Low Rated klase
3. Treći klaster ima najveći procenat High Rated klase

Još jedna metrika na koju ćemo obratiti pažnju je silueta koeficijent.

Ona meri koliko dobro svaki objekat odgovara svom sopstvenom klasteru u odnosu na druge klasterne.

Silueta koeficijent za ovaj model: 0.39665762880499467

Algoritam Sakupljajućeg Klasterovanja

Sakupljajuće klasterovanje (Agglomerative Clustering) je algoritam hijerarhijskog klasterovanja koji se koristi za grupisanje sličnih tačaka podataka u klasterima.

Ovo je pristup odozdo prema gore (bottom-up), gde svaka tačka podataka počinje kao sopstveni klaster, a zatim se iterativno spajaju klasteri na osnovu njihove sličnosti sve dok se ne dostigne željeni broj klastera.

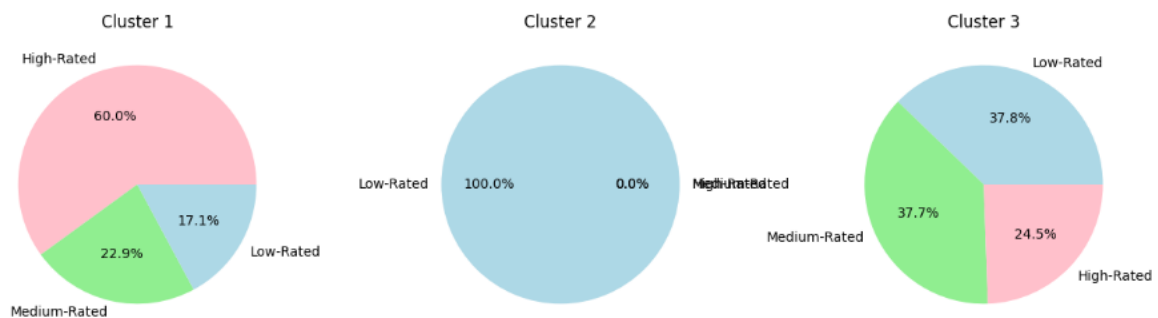
Pretprocesiranje skupa podataka i priprema za ovaj algoritam je identična kao za algoritam K sredina.

Rezultati

Nakon prvog poziva ovog algoritma, ovim kodom:

```
model = AgglomerativeClustering(n_clusters=3, linkage='single',
compute_distances=True)
```

Ovo su bili dobijeni rezultati:

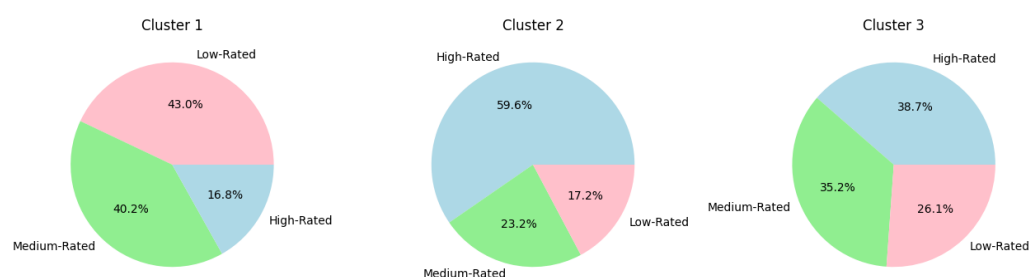


Klaster broj 2 je sadržao samo jednu instancu u sebi - verovatno neki autlajer. Zbog toga je bilo neophodno ispraviti "linkage" opciju:

```
model = AgglomerativeClustering(n_clusters=3, linkage='complete',
compute_distances=True)
```

"Complete" opcija nam omogućava bezbedniji rad sa autlajerima.

Nakon ove izmene, ovo su bili rezultati:



Silueta koeficijent za ovaj model: 0.4191515352464625

Poređenje modela

Upoređivaćemo modele na osnovu silueta koeficijenta i dominantnih klastera za svaku klasu.



Dominantno Nisko Rangiran klaster

Prvo ćemo uporediti klaster 2 algoritma K sredina i klaster 1 algoritma sakupljajućeg klasterovanja:

	Low Rated	Medium Rated	High Rated
K Means	37.7%	32.4%	29.9%
Agglomerative	43%	40.2%	16.8%

Iako model sakupljajućeg klasterovanja ima veći procenat nisko ocenjenih naslova, razlika između nisko i srednje ocenjenih naslova je veća kod ovog modela.

Sa druge strane, model K sredina ima skoro duplo veći procenat visoko ocenjenih naslova, što smatram većim problemom.

Algoritam sakupljajućeg klasterovanja je efikasniji u ovom slučaju.

Dominantno Srednje Rangiran klaster

Sada ćemo uporediti klaster 1 algoritma K sredina i klaster 3 algoritma sakupljajućeg klasterovanja:

	Low Rated	Medium Rated	High Rated
K Means	38%	46.2%	15.9%
Agglomerative	26.1%	35.2%	38.7%

U modelu sakupljajućeg klasterovanja zapravo ne postoji dominantni klaster srednje ocenjenih naslova. Poredili smo ga sa Klasterom 3 zato što je Klaster 2 dominantniji u visoko ocenjenim naslovima.

U ovom slučaju, algoritam K sredina je efikasniji.

Dominantno Visoko Rangiran klaster

Sada ćemo uporediti klaster 3 algoritma K sredina i klaster 2 algoritma sakupljajućeg klasterovanja:

	Low Rated	Medium Rated	High Rated
K Means	17.1%	22.9%	60%
Agglomerative	17.2%	23.2%	59.6%

Model K sredina je malo efikasniji.

Uopšteno, algoritam K-sredina ima bolje performanse. Ima bolju distribuciju visoko i srednje ocenjenih naslova, iako algoritam sakupljajućeg klasterovanja ima bolju distribuciju nisko ocenjenih naslova, kao i malo viši koeficijent siluete(0.4191515352464625 u odnosu na K sredina: 0.39665762880499467).

Pravila Pridruživanja - SPSS

Koristimo ovu metodu kada imamo veliki broj atributa i želimo da odredimo koji od tih atributa su međusobno povezani.

Pravila pridruživanja nam upravo služe da nađemo veze između podataka.

Ove veze su nam značajne jer time dobijamo bolje razumevanje podataka

Apriori Algoritam

Jedan od najpoznatijih algoritama za izdvajanje pravila pridruživanja je Apriori algoritam.

Apriori algoritam u fazi generisanja čestih skupova stavki koristi osobine podrške kako bi se smanjio broj skupova stavki za koje je potrebno izračunati podršku da bi se odredilo da li je skup stavki čest.

IBM SPSS Modeler nam omogućava laku implementaciju ovog algoritma.

Pretprocesiranje - Python

Radi lakšeg rada sa podacima, veći deo pretprocesiranja odrađen je u Pythonu. Podaci koje smo odlučili da izdvojimo za ovaj algoritam su:

- release_year
- imdb_score
- encoded_type
- encoded_genre

	release_year	imdb_score	encoded_type	encoded_genre
0	1926	8.2	0	1
1	1940	7.8	0	3
2	1945	7.3	0	16
3	1936	4.0	0	4
4	1916	7.7	0	9













Promenljiva encoded_type je LabelEncoder-om enkodirana tako da 0 predstavlja film (MOVIE), a 1 seriju (SHOW)

Promenljiva `encoded_genre` je `LabelEncoder`-om enkodirana na sledeći način:

Genre: ['action'], Encoded genre: 1
 Genre: ['animation'], Encoded genre: 2
 Genre: ['comedy'], Encoded genre: 3
 Genre: ['crime'], Encoded genre: 4
 Genre: ['documentation'], Encoded genre: 5
 Genre: ['drama'], Encoded genre: 6
 Genre: ['family'], Encoded genre: 7
 Genre: ['fantasy'], Encoded genre: 8
 Genre: ['history'], Encoded genre: 9
 Genre: ['horror'], Encoded genre: 10
 Genre: ['music'], Encoded genre: 11
 Genre: ['reality'], Encoded genre: 12
 Genre: ['romance'], Encoded genre: 13
 Genre: ['scifi'], Encoded genre: 14
 Genre: ['sport'], Encoded genre: 15
 Genre: ['thriller'], Encoded genre: 16
 Genre: ['war'], Encoded genre: 17
 Genre: ['western'], Encoded genre: 18

Pretprocesiranje - SPSS

Nakon pretprocesiranja u Pythonu, ovi podaci su učitani u SPSS Modeler. To je izgledalo ovako:

Field	Measurement	Values	Missing	Check	Role
 release_year	 Continuous	[1912,2022]		None	 Both
 imdb_score	 Continuous	[1.7,9.4]		None	 Both
 encoded_type	 Flag	1/0		None	 Both
 encoded_ge...	 Nominal	0,1,2,3,4,5...		None	 Both

Nakon čitanja svih vrednosti i postavljanja njihovih uloga na “Both”, odrađeno je binovanje.

Binovanje se koristi pri radu sa neprekidnim promenljivim, konvertujući ih u “binove” - intervale (kao što smo mapirali ove vrednosti u intervale pri klasterovanju i klasifikaciji). Ovo nam omogućuje pogodan format za budući rad sa podacima.

IMDB_score smo podelili na 3 intervala, kao i do sada, dok smo release_year podelili na intervale širine 40.

	release_year	imdb_score	encoded_type	encoded_genre	imdb_score_BIN	release_year_BIN
1	1926	8.200	0	1	3	1
2	1940	7.800	0	3	3	2
3	1945	7.300	0	16	3	2
4	1936	4.000	0	4	1	2
5	1916	7.700	0	9	3	1
6	1946	7.300	0	16	3	2
7	1925	7.500	0	10	3	1
8	1932	7.100	0	16	3	2
9	1928	7.800	0	6	3	1
10	1921	8.300	0	6	3	1

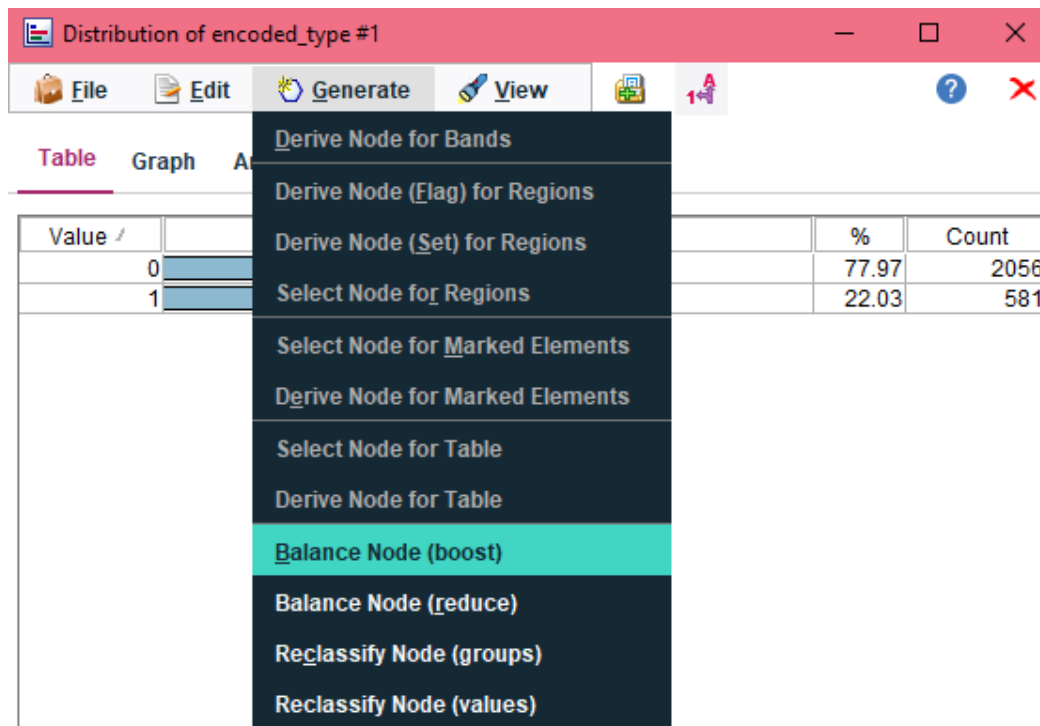
Sledeći korak koji je trebalo obaviti je balansiranje klasa. Pogledali smo kako naše klase izgledaju korišćenjem "Output" opcije:

Field	Sample Graph	Measurement	Min	Max	Mean	Std. Dev	Skewness	Unique	Valid
release_year		Continuous	1912	2022	1989.415	31.588	-0.722	--	2637
imdb_score		Continuous	1.700	9.400	6.043	1.266	-0.268	--	2637
encoded_type		Flag	0	1	--	--	--	2	2637
encoded_genre		Nominal	1	18	--	--	--	18	2637
imdb_score_BIN		Nominal	1	3	--	--	--	3	2637
release_year_B...		Nominal	1	4	--	--	--	4	2637

Prvo smo izbalansirali encoded_type opciju, da bismo izjednačili broj filmova i serija, jer tu imamo najveću razliku:

Distribution of encoded_type #1			
File	Edit	Generate	View
Table	Graph	Annotations	
Value	Proportion	%	Count
0		77.97	2056
1		22.03	581

Izabrali smo balansiranje generisanjem dodatnih uzoraka za manje reprezentovanu klasu:















Nakon toga, izbalansirali smo i imdb_score_BIN, i ovo je bio naš konačni skup podataka:

Field	Sample Graph	Measurement	Min	Max	Mean	Std. Dev	Skewness	Unique	Valid
release_year		Continuous	1912	2022	1999.637	26.059	-1.458	--	6880
imdb_score		Continuous	1.700	9.400	5.629	1.813	-0.152	--	6880
encoded_type		Flag	0	1	--	--	--	2	6880
encoded_genre		Nominal	1	18	--	--	--	18	6880
imdb_score_BIN		Nominal	1	3	--	--	--	3	6880
release_year_B...		Nominal	1	4	--	--	--	4	6880

* Indicates a multimode result * Indicates a sampled result

Pokretanje algoritma

Nakon pretprocesiranja, pokrenuli smo apriori algoritam sa sledećim opcijama:

Consequents:	<div>  imdb_score_BIN  encoded_type  encoded_genre  release_year_BIN </div>	 
Antecedents:	<div>  encoded_type  encoded_genre  release_year_BIN  imdb_score_BIN </div>	 

U listu Consequents se dodaju stavke koje mogu da se pojave u glavi pravila, a u listu Antecedents se dodaju stavke koje mogu da se pojave u telu pravila (Oblik pravila pridruživanja je telo \rightarrow glava, odnosno antecedents \rightarrow consequents). Stavili smo sve attribute u obe liste.

Model name:
 ☒ Auto
 ☐ Custom

☒ Use partitioned data

Minimum antecedent support (%):

Minimum rule confidence (%):

Maximum number of antecedents:

☒ Only true values for flags

Optimize:
 ☒ Speed
 ☐ Memory

Podrska (Support) : $\text{sup}(X \rightarrow Y) = \sigma(X \cup Y) / N$ određuje koliko se cesto pravilo pojavljuje u transakcijama skupa podataka

Pouzdanost (Condence): $\text{conf}(X \rightarrow Y) = \sigma(X \cup Y) / \sigma(X)$ određuje koliko se cesto stavke iz Y pojavljuju u transakcijama koje sadrze stavke iz X.

Rezultati

Ovo je izlaz apriori algoritma za ove parametre:

Consequent	Antecedent	Support %	Confidence %
encoded_type	encoded_genre = 12 release_year_BIN = 3	4.51	100.0
encoded_type	encoded_genre = 12 imdb_score_BIN = 1	5.188	100.0
encoded_type	encoded_genre = 12	9.845	99.85
encoded_type	encoded_genre = 12 release_year_BIN = 4	5.335	99.724
encoded_type	encoded_genre = 2	4.348	95.593

Možemo interpretirati ova pravila:

1. encoded_type -> encoded_genre = 12 and release_year_BIN = 3

- Kada nam je tip naslova SHOW(seriya), velika je verovatnoća da će biti praćena sa encoded_genre = 12 (REALITY) i release_year_BIN = 3 (Okolo 1990-2000tih godina)

2. encoded_type -> encoded_genre = 12 and imdb_score_BIN = 1

- Kada nam je tip naslova serija, velika je verovatnoća da bude "reality" žanra i da mu IMDB ocena bude u najnižem rasponu.

3. encoded_type -> encoded_genre = 12

- Kada nam je tip naslova serija, velika je verovatnoća da bude "reality" žanra

4. encoded_type -> encoded_genre = 12 and release_year_BIN = 4

- Kada nam je tip naslova SHOW(seriya), velika je verovatnoća da će biti praćena sa encoded_genre = 12 (REALITY) i release_year_BIN = 4 (poslednjih 20ak godina)

5. encoded_type -> encoded_genre = 2

- Kada nam je tip naslova serija, velika je verovatnoća da bude "animation" žanra

Zaključak je da je većina naslova koji su serije upada u "reality" žanr, dok im je ocena ili u najnižem ili najvišem rasponu, a sve su imale premijere u poslednjih tridesetak godina.

Zaključak

Kada je u pitanju klasifikacija, algoritam koji je pokazao najbolje rezultate na ovom skupu je KNN algoritam sa GridSearchCV optimizacijom, sa tačnošću 96% i AUC-om 99.

Najbolje rezultate algoritama klasterovanja dao je algoritam K sredina, sa najboljim razdvajanjem visoko, srednje i nisko ocenjenih naslova.

Zaključak Apriori algoritma u IBM SPSS modeleru je da je većina naslova koji su serije upada u "reality" žanr, dok im je ocena ili u najnižem ili najvišem rasponu, a sve su imale premijere u poslednjih tridesetak godina.

Link do projekta:

https://github.com/MATF-istrazivanje-podataka-1/2023_Data_Mining_Artificial_Characters_Dataset/tree/main