

Izveštaj

Luka Arambašić

Jun 2023

Contents

1	Uvod	2
2	Opis modela	4
2.1	Klasifikacija	5
2.2	Klasterovanje	9
2.3	Pravila pridruživanja	10
3	Rezultati	11
4	Zaključak	22

1 Uvod

Projekat je radjen na *diamonds* dataset-u iz biblioteke *tensorflow-datasets*.

Projekat za cilj ima da se prikazu razlicite tehnike analize i preprocesiranja nad skupom podataka i demonstracije modela klasifikacije, klasterovanja i pravila pridruzivanja.

Opste o skupu podataka:

Skup podataka sadrzi fizicke attribute i cene 53 940 dijamanta.

Atributi(10):

- price: Cena u US dolarima. (326 USD–18,823 USD)
- cut: Kvalitet reza. (Fair, Good, Very Good, Premium, Ideal)
- color: Boja dijamanta. od D (najbolje) do J (najgore)
- clarity: Cistoca dijamanta. (I1 (najgore), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (najbolje))
- x: Duzina u mm. (0–10.74)
- y: Sirina u mm. (0–58.9)
- z: Dubina mm. (0–31.8)
- depth: $100 * z / \text{mean}(x, y)$. (43–79)
- table: Sirina vrha dijamanta u odnosu na najsiru tacku. (43–95)

Homepage: <https://ggplot2.tidyverse.org/reference/diamonds.html>

Source code: `tfds.structured.diamonds.Diamonds`

Versions: 1.0.0 (default): Initial release.

Download size: 2.64 MiB

Dataset size: 13.01 MiB

Klasifikacija: ovaj skup podataka se obicno koristi u svrhe problema linearne regresije i predvidjanja cene na osnovu atributa. Kako bismo problem regresije sveli na problem klasifikacije, mozemo kontinualni atribut cene podeliti na opsege cena koji ce nam ciniti klase za predvidjanje. S obzirom da vec imamo informaciju da je najvisa cena u skupu 18 823USD, napravicemo cetiri klase: $[0, 5000]$, $(5000, 10000]$, $(10000, 15000]$ i $15000+$

Klasterovanje: cilj je uociti da li u skupu podataka postoje grupacije instanci i izvuci dodatne informacije koje nam mogu pomoci u modelovanju.

Pravila pridruzivanja: cilj je uociti da li u podacima postoje neke pravilnosti koje se veoma cesto ponavljaju i uslovljavaju neki specificni rezultat.

2 Opis modela

Pretprocesovanje

S obzirom da je ovo jedan od popularnih skupova podataka koji se koristi u svrhe vezbanja masinskog učenja, sam skup podataka je već donekle preprocesovan i spreman za modelovanje. *Nedostajuće vrednosti* ne postoje. *Kodiranje kategorickih atributa* je već izvršeno i celobrojne vrednosti se već nalaze u kolonama atributa "clarity", "color" i "cut".

Inicijalno, svi atributi su imali prefiks "features/" koji smo odstranili radi lakšeg rukovanja skupom podataka.

Outlier-i su tehnikom *z-score* odstranjeni ako im je skor veći od 3. Prethodno je primenjeno na sve attribute osim atributa "price" koji želimo da predviđamo. Razlog za to je jer postoji mogućnost da su potencijalni outlier-i samo kombinacija visoko kvalitetnih atributa i da to utiče na odvajanje u ceni. Nakon što se otklone outlier-i za attribute, broj outlier-a se u ceni promenio sa 1206 na 1152 što najverovatnije ukazuje da je pretpostavka tačna.

Dalje, napravljena je nova kolona *class* koja sadrži jednu od četiri klase opsega cene za predviđanje. Nakon ovoga je uklonjena kolona *price* jer više nije od značaja.

Može se uočiti da skup podataka sadrži 3 atributa koji opisuju dimenziju dijamanta, *x*, *y* i *z*, kao i 2 atributa koji opisuju odnos tih dimenzija, *depth* i *table*. *Depth* nam daje neke podatke o obliku dijamanta ($100 * z / \text{mean}(x, y)$), a *table* nam daje podatke o odnosu gornje površine i najsireg dela dijamanta. Iz ovih atributa je napravljen jedan atribut koji će bolje uspeti da predstavi informaciju o dimenziji dijamanta, tako što su pomnožene vrednosti *x*, *y* i *z*. Taj atribut je nazvan *volume* i predstavlja opisan kvadar oko dijamanta. Posmatrajuci oblik dijamanta, bilo bi računski jako zahtevno izračunati stvarnu zapreminu. Bez obzira na taj nedostatak, nadamo se da će algoritmi uspeti da uhvate informaciju i da dobiju "sliku" o zapremini dijamanta. Istu zapreminu možemo dobiti za različite vrednosti *x*, *y* i *z*, a oblik dijamanta i odnos strana nisu zanemarljivi, biće ostavljeni atributi *depth*, *table*, *x*, *y*, *z* i prepustiti algoritmu da izabere šta mu najviše odgovara.

Svi atributi osim *carat* su normalizovani. *Carat* nećemo dirati jer svakako ne odskace mnogo od ostatka i većina je u opsegu od 0 do 1.

Na osnovu analize skupa uvidjeno je da postoji velika neuravnoteženost klase. Najzastupljenija klasa čini 70%, dok najmanje zastupljena čini svega 3%. S obzirom da imamo preko 50 000 instanci u skupu, 70% čini 35 000 instanci. Koriscenje tehnike poput SMOTE-a (Synthetic Minority Oversampling Technique) bi nam dovelo dataset na preko 140 000 instanci što bi veoma povećalo vreme treniranja modela. Takođe, instanci u klasi "15000+" ima svega 3% a ona sadrži najveći deo outliera i visoke cene, nešto što bismo smatrali "ekstremnim vrednostima". Oversamplovanje ovakve klase sa 1500 instanci na 35 000

dovodi u rizik da nam se smanji konzistentnost skupa podataka, pogotovo ako novogenerisane instance ne bi bile dobra reprezentacija inicijalne distribucije. Nadalje cemo koristiti tehniku *weighting*, tamo gde bude potrebno. Ona podrazumeva da kao argument nasem klasifikacionom modelu prosledimo tezinu klase kao merilo paznje koju ce joj posvetiti.

2.1 Klasifikacija

Modeli korisceni za klasifikaciju su: *DecisionTreeClassifier*, *KNeighborsClassifier*, *MLPClassifier*, *SVC*, *RandomForestClassifier* i *XGBClassifier*.

DecisionTreeClassifier Drveta odlucivanja su hijerarhijske strukture koje se sastoje od cvorova i grana. Svaki cvor predstavlja testiranje odredjene osobine, dok grane predstavljaju moguće vrednosti te osobine. Na osnovu rezultata testiranja, model se kreće kroz drveta sve do listova, gde se donose konacne odluke o klasifikaciji. Klasifikacija novih instanci se vrši prolaskom kroz model drveta, pri čemu se na svakom cvoru primenjuje odgovarajući test i prelazi se na sledeći cvor sve dok se ne stigne do lista. Na kraju se vrši klasifikacija na osnovu odluke u listu.

Prednosti su u tome što drveta odlucivanja lako razumljiva i mogu se vizualizovati u hijerarhijskoj strukturi. Takodje, pružaju meru značajnosti atributa, pokazujući koji atributi najviše utiču na predviđanja. Ove informacije mogu pomoći pri odabiru atributa. Drvo odlucivanja može modelirati kompleksne nelinearne veze između osobina i ciljne promenljive. Mogu uhvatiti interakcije i pragove, što ih čini pogodnim za probleme u kojima linearne metode možda neće dati dobre rezultate. Još jedna od pogodnosti ovog modela je to što je manje osetljiv na ekstremne vrednosti i nedostajuće vrednosti u poredjenju s nekim drugim algoritmima. Odluke se donose na osnovu serije podela, pa ekstremne vrednosti ili nedostajuće vrednosti ne utiču toliko na konacni rezultat. (Ranije je pomenuto da imamo outlier-e i da smo neke odlucili da zadržimo.)

Prvobitno je odradjen obican *test-train-split* da bismo uvideli kako se model ponasa nad nasim skupom podataka. Napravljene su pomocne funkcije *model_report* i *feature_importance* za vizualizaciju evaluacije modela i značajnosti atributa.

S obzirom da imamo neuravnoteženost klasa, a algoritam je osetljiv na tu neuravnoteženost, koristicemo parametar *class_weight* kako bismo dodelili tezine klasama. One su izracunate tako sto je za svaku klasu izracunat procenat koji zauzima u skupu a zatim reciprocnu vrednost toga pomnozimo sa 100.

Nakon ovoga, istreniran je prvi model i izvršena evaluacija modela i značajnosti atributa. Zatim je koriscen *GridSearchCV* kako bi bili pronadjeni najbolji parametri za model.

Tokom analize skupa podataka uvidjena je visoka korelacija nekih atributa, a i na osnovu analize značajnosti atributa tokom treniranja modela, ocigledno je da se tehnikom *PCA* (*Principal Component Analysis*) može smanjiti dimenzionalnost skupa podataka i možda napraviti bolja predikcija. Pretpostavka

se ispostavila tacnom jer kumulativna suma 4 komponente iznosi preko 0.95. Ponovo je primenjena pretraga parametara kako bismo nasli najbolji model i uporedili ga sa rezultatima ostalih modela.

KNeighborsClassifier Ovaj algoritam se oslanja na slicnost izmedju instanci kako bi donosio odluke o klasifikaciji. Klasifikacija se vrsi na osnovu bliskosti izmedju instance koja se predvidja i njenih K najblizih suseda. Bliskost se obicno meri putem euklidskog rastojanja ili drugih metrika udaljenosti. Takodje, ovaj algoritam se moze primeniti na razlicite vrste podataka i dobro se nosi sa sumovima i outlier-ima.

Motivacija za ovaj model lezi u njegovoj jednostavnosti ali i efikasnosti. Naime, algoritam ne zahteva pretpostavke o distribuciji podataka ili obliku veza izmedju osobina i ciljne promenljive. *KNeighborsClassifier* se oslanja na lokalnost podataka. Klasifikacija se vrsi na osnovu slicnosti izmedju instance koja se predvidja i njenih najblizih suseda. Ovo je posebno korisno u problemima gde su slicne instance sklonije pripadati istoj klasi. Pored ovoga, algoritam pruza informaciju o verovatnoci pripadnosti svakoj klasi. Skalabilnost je takodje jedna od prednosti, u smislu velicine trening skupa podataka. Moze se lako primeniti na velike skupove podataka bez gubitka performansi.

Inicijalno je koriscen obican *train-test-split* kako bismo uvideli ponasanje na skupu podataka. Od pomocnih funkcija implementirana je *model_report* za evaluaciju modela. Za ovaj model je od velikog znacaja skaliranje atributa s obzirom da se mere rastojanja izmedju vrednosti. U preprocesovanju je izvršena normalizacija vrednosti. Napravljeno je vise modela i primenje su tehnike *PCA* i *GridSearchCV* i svi modeli su upoređeni kako bi se nasao onaj sa najboljim rezultatima.

MLPClassifier Algoritam se zasniva na neuronskim mrežama, posebno vislojnim perceptronima, kako bi naucio kompleksne veze izmedju ulaznih osobina i ciljnih klasa. MLPClassifier koristi metodu propagacije unapred (feed-forward) kako bi preneo ulazne podatke kroz mrežu i generisao izlazne vrednosti. Zatim se koristi metoda propagacije unazad (backpropagation) kako bi se prilagodili parametri mreže na osnovu razlike izmedju predvidjenih izlaza i stvarnih vrednosti ciljnih klasa. Ovaj proces se iterativno ponavlja kako bi se minimizovala greska i optimizovali parametri mreže.

MLPClassifier je mocan alat za modelovanje kompleksnih veza izmedju ulaznih osobina i ciljnih klasa. Neuronske mreže imaju sposobnost nauciti i predstaviti nelinearne veze koje drugi linearni modeli ne mogu. S obzirom da su nam atributi uglavnom kontinualnog tipa, nadamo se da ce mreža uspeti da "uhvati" vezu izmedju atributa i klasa. Algoritam je takodje zadovoljavajuce otporan na sumove. Takodje, visoka prilagodljivost strukture same mreže je od velikog znacaja jer je mozemo prilagoditi nasem skupu podataka. Algoritam pokazuje visoke performanse kako sa kategorickim, tako i sa kontinualnim atributima.

Koristimo train-test-split kao i do sada kako bismo uvideli osnovne karak-

teristike modela. Od pomocnih funkcija tu su *model_report* i *plot_learning_curve* koja iscrtava krivu ucenja modela na trening skupu. Za *model_report* su korisceni *precision*, *recall* i *f1-score* kao i *accuracy*. Pored njih, funkcija i vizualizuje matricu konfuzije. Koriscene su tehnike *PCA* i *GridSearchCV*. U potrazi za najboljim parametrima isprobano je nekoliko jednoslojnih i viseslojnih arhitektura mreze kao i razlicite aktivacione funkcije. Nakon poredjenja svih modela i rezultata, ispostavlja se da se najbolje pokazala prvobitna konfiguracija parametara.

SVC Osnovna ideja SVC-a je pronaci hiper-ravan (u slucaju binarne klasifikacije) ili hiper-ravni (u slucaju viseklasne klasifikacije, kao u nasem slucaju) koje najbolje razdvajaju podatke razlicitih klasa. Hiper-ravni se biraju tako da maksimizuju rastojanje od najblizih tacaka svake klase, kojima se nazivaju potporni vektori. SVC koristi i tehniku zvanu "kernel trick" koja omogucava da se podaci preslikaju u prostor vece dimenzionalnosti radi bolje separacije.

Od pomocnih funkcija koriste se vec opisana *model_report*. Istrenirano je vise modela, primenjene su tehnike *PCA* i *GridSearchCV* i uporedjeni su dobijeni rezultati. S obzirom da nemamo jasno definisane grupacije u skupu podataka, razumljivo je zasto ovaj algoritam daje najgori performans. Odredjivanje hiper-ravni nije trivijalan posao. Sve u svemu, rezultati su i dalje zadovoljavajuci, mada je vreme treniranja skupo.

RandomForestClassifier Algoritam se zasniva na konceptu ensemble metoda i kombinuje vise stabala odlucivanja kako bi donosio odluke o klasifikaciji kombinovanjem predvidjanja svakog stabla. Odluka se obicno vrsi vecinskim glasanjem, gde se klasa koja je najcesca medju svim stablima smatra konacnim predvidjanjem. Svako stablo se trenira na razlicitom podskupu podataka dobijenih metodom slucajnog izbora. Takodje, tokom konstrukcije svakog stabla, vrsi se slucajni izbor atributa koji se koriste za donosenje odluka na svakom cvoru.

Ovaj algoritam ima nekoliko prednosti, ukljucujuci visoku tacnost predvidjanja, otpornost na preprilagodjavanje i sposobnost rukovanja sa velikim skupovima podataka. Paralelizacija treninga stabala omogucava brzu obradu velikog broja instanci ili osobina, cime se stedi vreme obuke i predikcije. Neuravnotezenost klasa ne predstavlja problem, jer se algoritam zasniva na stablima odlucivanja, te se mogu pripisati tezine klasama. Kombinacija vise stabala odlucivanja smanjuje varijabilnost i greske koje se mogu pojaviti u pojedinacnim stablima, sto vodi do pouzdanih rezultata klasifikacije. Kao i algoritam stabla odlucivanja, moze pružiti informacije o značajnosti atributa za klasifikaciju. Kombinacija vise stabala odlucivanja omogucava modelu da izgradi stabilne predikcije, uz minimalan uticaj suma ili nedostajucih vrednosti.

Od pomocnih funkcija se koriste *model_report* za evaluaciju modela, *plot_learning_curve* za vizualizaciju ucenja algoritma na trening skupu i *feature_importance* koja prikazuje znacaj atributa. Tezine klasa su identicno napravljene kao za stabla odlucivanja, reciprocnom vrednoscu procenta klase pom-

nozeno sa 100. Kao i za ostale, koriscene su tehnike PCA i GridSearchCV kako bismo evaluirali razlicite modele i kasnije ih uporedili.

Veoma zanimljiva informacija se moze uociti nakon prikazivanja znacaja atributa. Naime, novogenerisani atribut *volume* ima najveću ocenu znacaja sa preko 18%. Time se pokazalo da je pravljenje te kolone veoma dobro uticalo na skup podataka. Takodje, jedan od modela je zabeležio, za sada, najvisi skor evaluacije modela.

XGBClassifier Ovaj algoritam pripada porodici gradijentnog boostinga i koristi ekstremni gradijentni boosting (XGBoost) kao osnovni mehanizam za poboljšanje tacnosti klasifikacije.

XGBClassifier koristi *ensemble* pristup i takodje je zasnovan na kombinovanju više stabala odlucivanja, kako bi donosio odluke o klasifikaciji. Svako drvo odlucivanja se trenira iterativno, pri čemu se svaka iteracija fokusira na korekciju gresaka prethodne iteracije. Na taj način, model se postepeno poboljšava kako bi dao preciznije predviđanje.

Poznati po svom efikasnom upravljanju velikim skupovima podataka, visokim tacnostima predviđanja, robustnosti na šum i sposobnosti rukovanja sa razlicitim vrstama atributa, *Gradient booster*-i su najkorisceniji modeli za rešavanje realnih problema. Takodje, XGBClassifier moze pružiti informacije o znacajnosti atributa za klasifikaciju jer se zasniva na stablima odlucivanja.

Vec od najosnovnijeg modela se mogla uociti visoka performantnost. Jedina mana ovog algoritma je sto je vreme treniranja obicno dosta dugo, pogotovo kada se koristi GridSearchCV za pretragu parametara. Takodje, moze se uociti da PCA tehnika ne poboljšava rezultate. XGBClassifier je sposoban da uoci i najsitnije korelacije izmedju atributa i da ih modeluje na pravi način. Jos jednom, nakon analize znacajnosti atributa, pokazano je da je kreiranje atributa *volume* bila jako dobra ideja.

2.2 Klasterovanje

Zeleli bismo da znamo da li u našem skupu podataka postoje neka grupisanja instanci. Te informacije nam mogu pomoći u klasifikacionom problemu, na primer za određivanje klasa koje možemo predvideti. Ako su klasteri jasno definisani i ako bismo klase pravili po klasterima, verovatno je da bismo poboljšali predviđanja modela. Ono što nas najviše zanima je, da li je naša podela na 4 klase dobra ili možda postoji neka druga konfiguracija tih klasa koja bi dala bolje rezultate.

Na skupu podataka je isprobana tehnika PCA za smanjenje dimenzionalnosti i rezultat je dimenzije 4 što nam i nije od nekog značaja jer bismo želeli da možemo da vizualizujemo podatke na osnovu PCA komponenti.

Modeli korisni za klasterovanje su: *KMeans*, *KModes* i *GaussianMixture*

KMeans za cilj ima pronalazjenje K klastera, pri čemu je K unapred određen broj klastera koje želimo da formiramo. Koristimo ga zbog jednostavnosti implementacije i brzine izvršavanja. Međutim, treba imati na umu da KMeans može biti osetljiv na inicijalnu postavku centroida klastera.

Kako bismo to resili, uvešćemo pomoćnu funkciju *cluster_search* koja će nam pokrenuti više modela za zadati opseg $(0, k]$. Za meru evaluacije modela korist ćemo *silhouette_score*. Pored toga, funkcija nam crta i *silhouette diagram*.

KModes je algoritam za klasterovanje sličan *KMeans-u* koji je sposoban za obradivanje kategorickih podataka. Algoritam je proširenje k-means algoritma, koji je dizajniran za numeričke podatke. Umesto da koristi euklidsku udaljenost za merenje sličnosti između podataka, kmodes koristi metriku koja je specifična za kategoričke atribute. Cilj kmodes algoritma je grupisati podatke u klastera tako da se minimizuje ukupna udaljenost između podataka i modova (najčešće vrednosti) u svakom klasteru. Algoritam iterativno azurira klastera i konvergira ka stabilnom rasporedu.

S obzirom da u skupu podataka imamo 3 kategorička atributa, najviše se oslanjamo na rezultate ovog algoritma

GaussianMixture je algoritam za modelovanje verovatnoće i klasterovanje podataka koji se zasniva na Gausovim raspodelama. Ovaj algoritam se koristi za pronalazjenje skrivenih grupa ili klastera u podacima. GaussianMixture je superpozicija nekoliko Gausovih raspodela, pri čemu svaka komponenta predstavlja jedan klaster. Algoritam procenjuje parametre raspodele, kao što su srednje vrednosti (centroidi), matrice kovarijanse i težine komponentata. Kroz iterativni proces, algoritam pokušava da pronađe najbolju raspodelu komponentata koja najbolje odgovara podacima.

2.3 Pravila pridruzivanja

Apriori je algoritam za otkrivanje asocijativnih pravila u skupovima podataka. Algoritam funkcionise na principu pronalazenja cestih uzoraka u podacima. Asocijativna pravila su izrazi u obliku "A sledi B", gde A i B predstavljaju skupove stavki, a strelica predstavlja implikaciju. Ova pravila se zasnivaju na merama *support* i *confidence*, koje se koriste za procenu znacajnosti i pouzdanosti pravila.

Modelovanje i obrada skupa podataka su radjeni u IBM-ovom SPSS-u. S obzirom da ovaj algoritam radi samo sa kategorickim atributima, odbaceni su kontinualni atributi. Skup podataka je takve prirode da nije podoban za pravila pridruzivanja jer se uglavnom koristi za regresiju (Cilj je predvidjanje cene dijamanta). Ali, pokusacemo da uvidimo da li mozda postoje neka pravila izmedju kategorickih atributa *clarity*, *color*, *cut* sa klasama koje smo predvidjali u klasifikaciji. Instancama je ponovo dodeljen novi atribut *class* koji predstavlja opseg cene u kom se cena instance nalazi.

3 Rezultati

Rezultati klasifikacije

Od svih istreniranih modela selektovani su najbolji predstavnici i upoređeni su njihovi rezultati na našem skupu podataka. Metrike za evaluaciju modela koje su korišćene su: *accuracy*, *precision*, *recall* i *f1-score*. Na kraju, upoređena su vremena treniranja modela na skupu podataka radi ocene skalabilnosti.

Procenat tačnosti (*accuracy_score*):

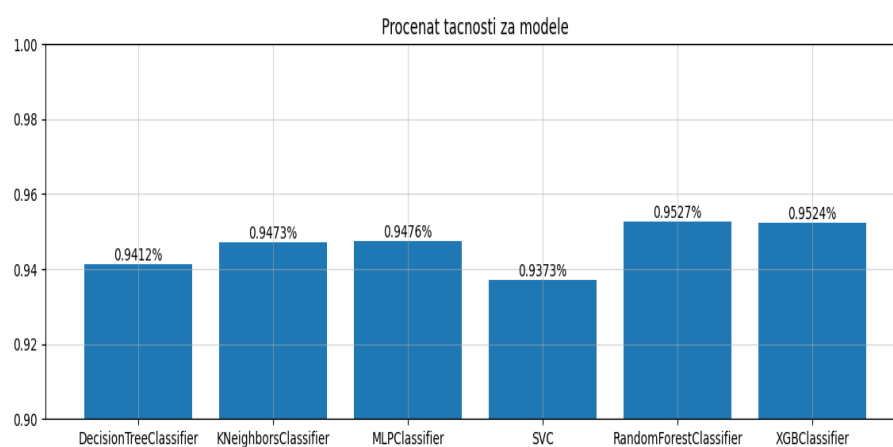


Figure 1: Accuracy_score za modele.

Svi modeli daju zadovoljavajuće rezultate. Može se uočiti da najbolje rezultate daju RandomForestClassifier i XGBClassifier, što je i očekivano od ensemble modela.

Preciznost je metrika koja predstavlja odnos tačno pozitivno klasifikovanih instanci prema zbiru tačno pozitivno i lažno pozitivno klasifikovanih instanci. Rezultati za preciznost (*precision_score*):

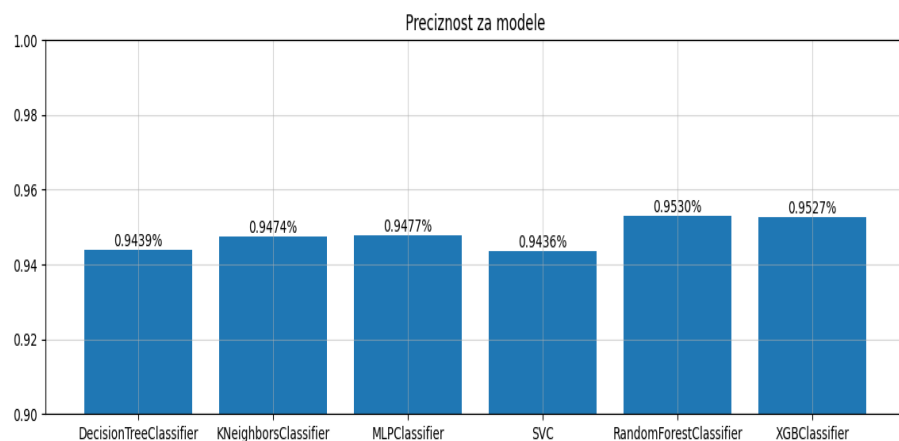


Figure 2: Precision_score za modele.

Odziv predstavlja odnos tačno pozitivno klasifikovanih instanci prema zbiru tačno pozitivno klasifikovanih i lažno negativno klasifikovanih instanci. Odziv meri sposobnost modela da pravilno identifikuje sve pozitivne instance. Rezultati za odziv (*recall_score*):

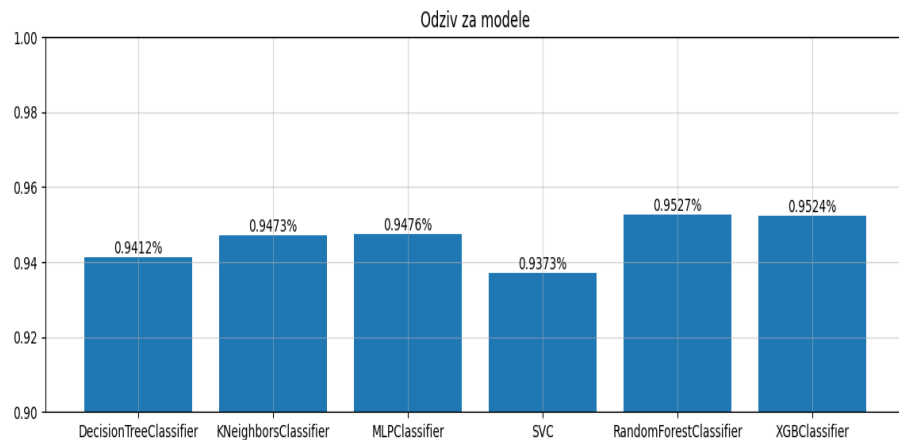


Figure 3: Recall_score za modele.

Mozemo primetiti da su accuracy_score i recall_score jednaki za svaki od 6 modela. Ako su rezultati preciznosti i odziva jednaki za model, to znaci da je model postigao istu performansu u pogledu tacnog identifikovanja pozitivnih

instanci i ukupne tacnosti svojih predvidjanja.

F1-skor je harmonična sredina preciznosti i odziva. On kombinuje informacije o tačnosti modela u klasifikaciji pozitivnih i negativnih instanci. Rezultati za F1-skor (*f1_score*):

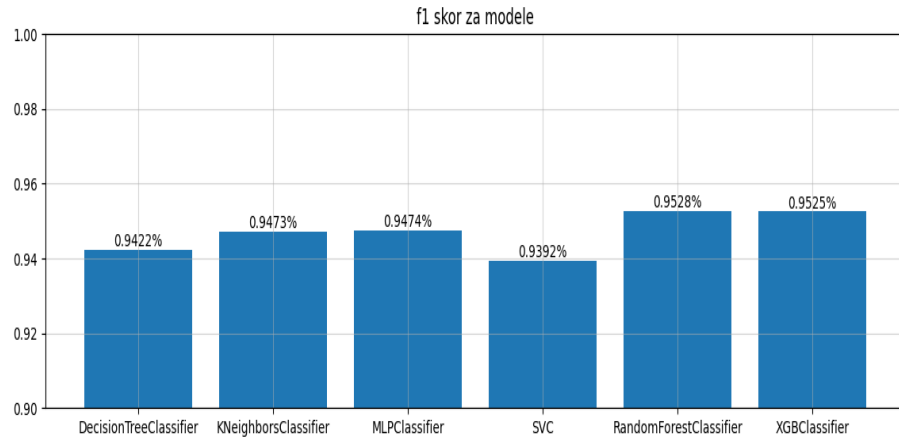


Figure 4: F1_score za modele.

Od svih modela **RandomForestClassifier** pokazuje najbolje rezultate. Za njim je odmah **XGBClassifier**. Ocekivano od *ensemble* modela. Ostali modeli daju zadovoljavajuće rezultate osim Bayesa. Razlog za to je najverovatnije visoka linearna korelacija izmedju kontinualnih atributa. Mada ni njegovi rezultati nisu mnogo losi.

S obzirom na generalno visoke performanse svih modela, ne bismo pogrešili da za nas problem izaberemo bilo koji od njih ali kada govorimo o realnim problemima u obzir se mora uzeti i vreme potrebno da se svaki model istrenira i njegova mogućnost za rad na jako velikim skupovima podataka. S toga smo izmerili vreme treniranja modela i uporedili im rezultate. Potrebno je odluciti da li je ta mala razlika u preciznosti pogađanja vredna u odnosu na razliku vremenskih cena modela.

Ako bismo birali prostije modele i dvoumili se, na primer, izmedju `KNeighborsClassifier`-a ili `MLPClassifier`-a, ova statistika nam u tome može pomoći:

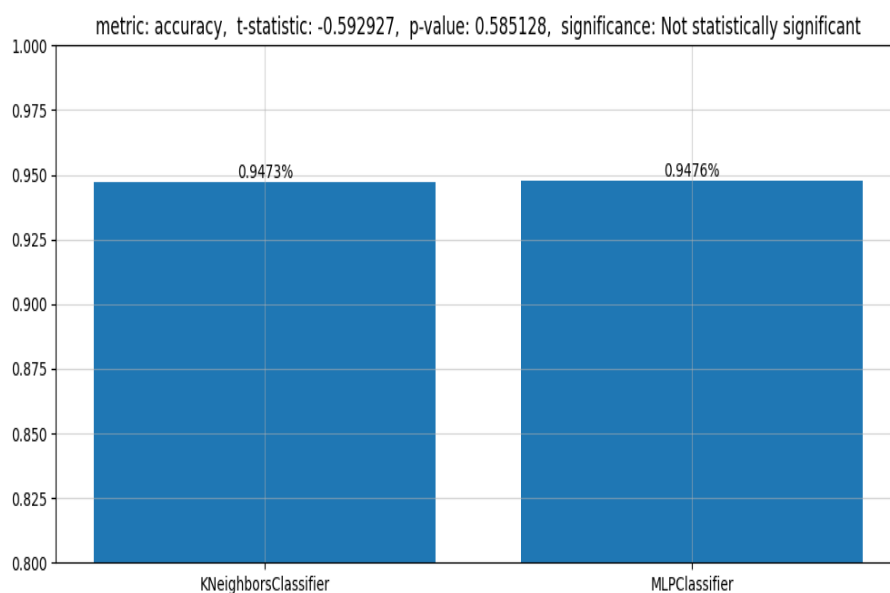
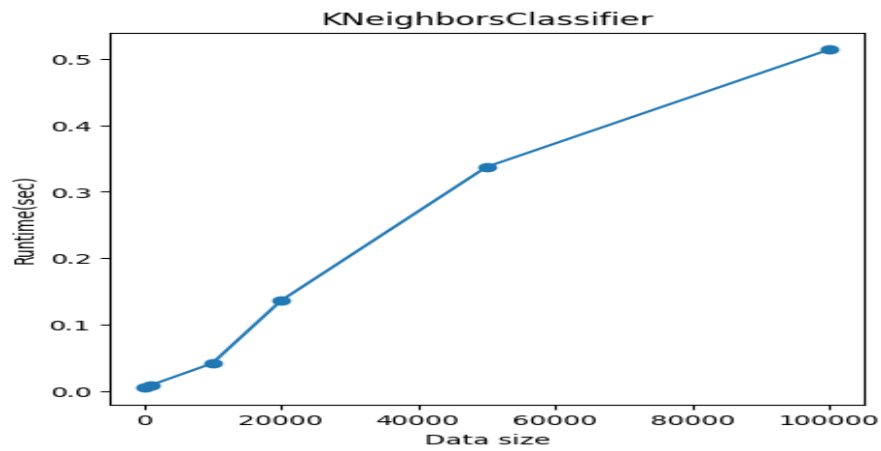


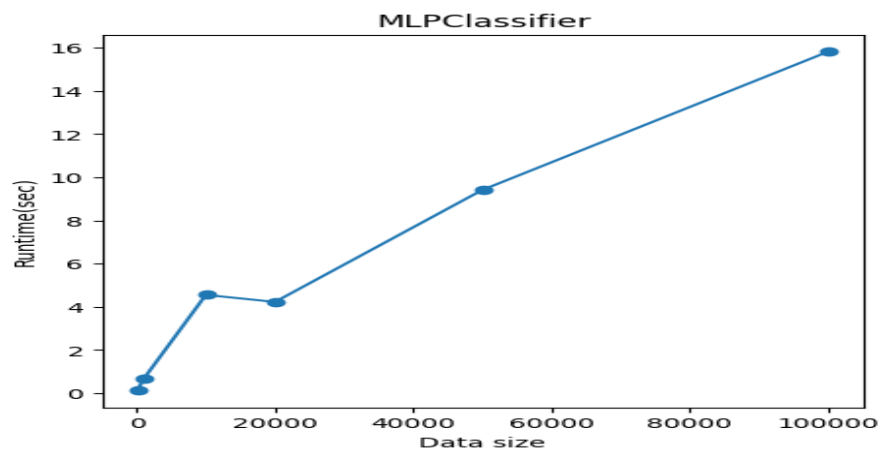
Figure 5: KNN vs MLP

`MLPClassifier` jeste model koji daje malo bolje rezultate. Ali možemo videti da ta razlika je tek 0.003% u tačnosti predviđanja. Da li je tih 0.0003% vredno vremenske cene potrebne da se istrenira cela neuronska mreža?

Pogledajmo sada vreme izvršavanja za KNeighborsClassifier:



Pogledajmo sada vreme izvršavanja za MLPClassifier:



Mozemo uociti da razlika u vremenu nije zanemarljivo mala. KNeighborsClassifier se na skupu podataka od 100 000 instanci istrenira za 0.5 sekundi dok je, za isti skup podataka, MLPClassifier-u potrebno cak 16 sekundi. Namece se pitanje sta bismo radili da imamo milion instanci u skupu podataka? U ovakvom slucaju bismo "zrtvovali" 0.0003% tacnosti zarad 32 puta brze treniranje modela.

Ako se ipak odlucimo za kompleksnije *ensemble* modele poput XGBClassifier-a ili RandomForestClassifier-a, trebalo bi uzeti u obzir da njihova visoka moc predvidjanja i modeliranje kompleksnih veza izmedju atributa, dolazi sa visokom vremenskom cenom potrebnom da se te veze uoce.

Ako uporedimo njihove ocene tacnosti videcemo da imamo slicnu situaciju, jedan model je bolji ali je takav inkrement statisticki neznacajan:

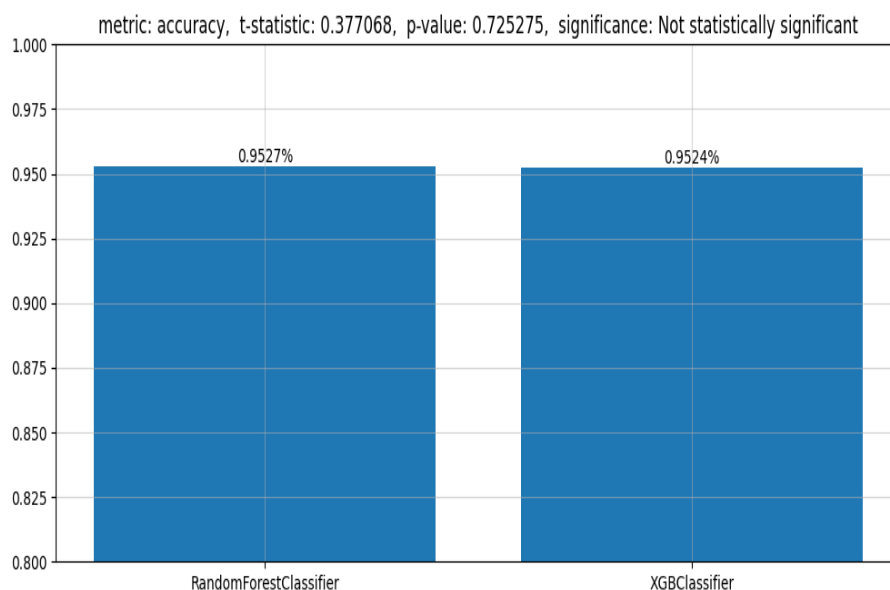
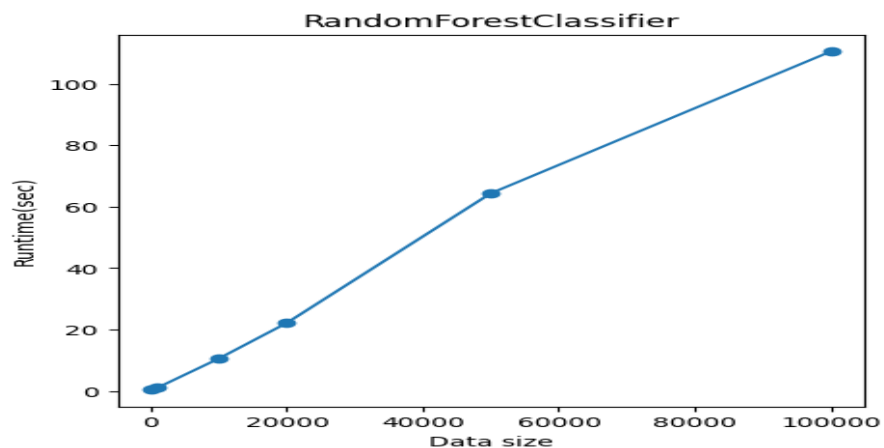


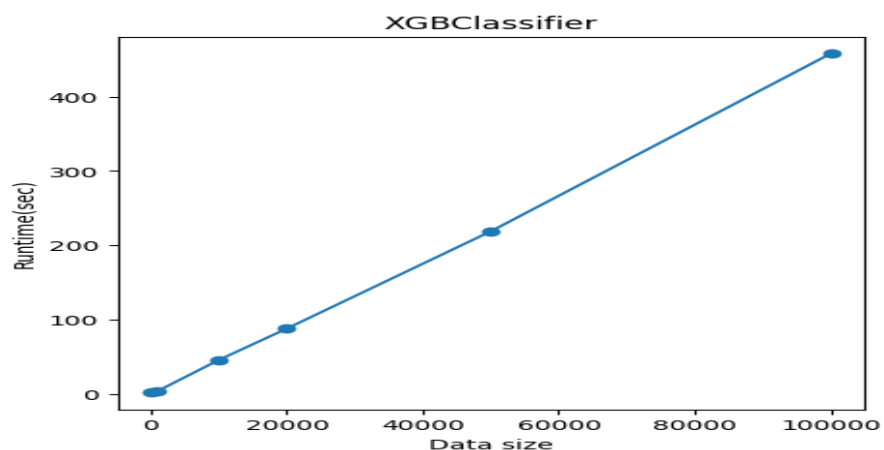
Figure 6: RFC vs XGB

Ponovo imamo istu razliku od 0.0003%. Ponovo nam se namece pitanje da li je to vredno razlike u vremenima potrebnim za treniranje modela. Od *ensemble* modela generalno ocekujemo duze vreme treniranja pa ta razlika moze biti od ogromnog znacaja.

Pogledajmo sada vreme izvršavanja za RandomForestClassifier:



Pogledajmo sada vreme izvršavanja za XGBClassifier:



Posle pazljivog razmatranja, uoceno je da je RandomForestClassifier cak 4 puta brzi od XGBClassifiera. Ta razlika na ovoj razmeri je veoma bitna. Svakako je RandomForestClassifier i tacniji model.

Nakon uzimanja u obzir svih metoda evaluacije modela koji su korisnici, ocigledno je zakljuciti da se najbolje pokazao **RandomForestClassifier**.

Rezultati klasterovanja

Metrike za evaluaciju modela klasterovanja koje su korišćene su: *Silhouette score*, *Dunn index*, *Rand index* i *Intra-cluster distances*. Razmatrani modeli korišćeni za klasterovanje su: *KMeans*, *KModes* i *GaussianMixture*.

Silueta skor je metrika koja meri koliko dobro je svaki objekat dodeljen odgovarajućem klasteru, uzimajući u obzir gustinu i razdvajanje klastera. Vrednosti Silhouette koeficijenta se kreću u opsegu od -1 do 1. Veće vrednosti silueta koeficijenta ukazuju na bolje klasterovanje, pri čemu vrednost bliska 1 ukazuje na dobro definisane i odvojene klastera, dok vrednost bliska -1 ukazuje na objekte koji su pogrešno dodeljeni klasterima. Silueta skor za modele:

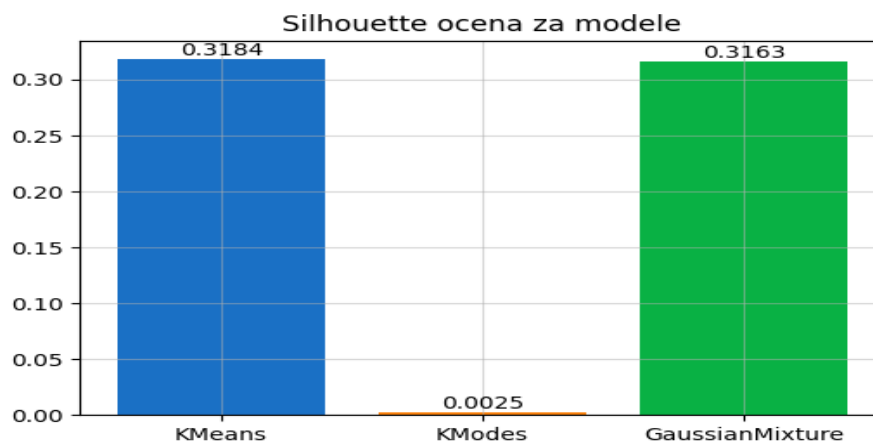


Figure 7: Silhouette score

Svakako ocene nisu zadovoljavajuće da bismo mogli da kažemo da imamo neke klastera. Takođe treba uzeti u obzir da su Kmeans i GaussianMixture osetljivi na kategoricke promenljive te daju ovakve rezultate. Najrealniju sliku klasterovanja daje KModes

Dunn index je metrika koja meri razdvajanje između klastera i gustocu objekata unutar klastera. Visoke vrednosti Dunn indeksa ukazuju na dobro odvojene klastera i dobru gustinu unutar klastera, dok niske vrednosti ukazuju na manje jasno definisane klastera i veću raspršenost objekata.

Dunn index za modele:

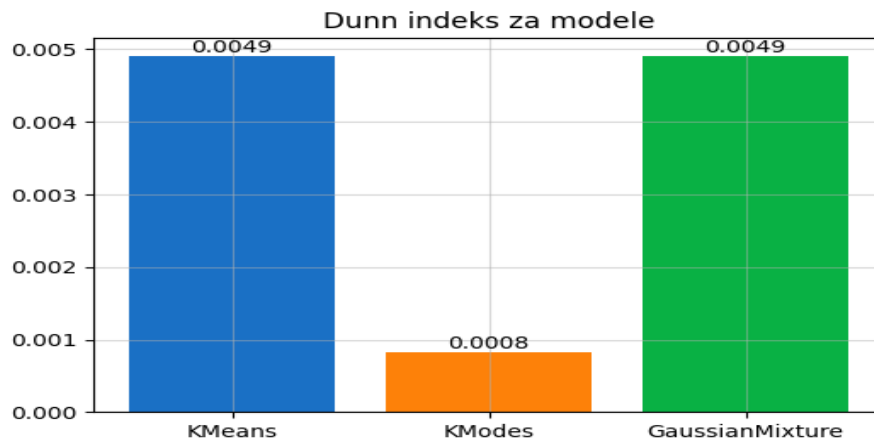
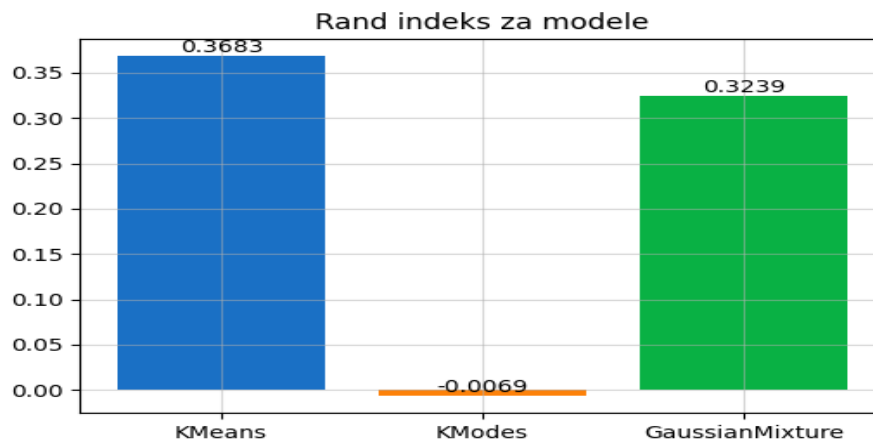


Figure 8: Dunn index

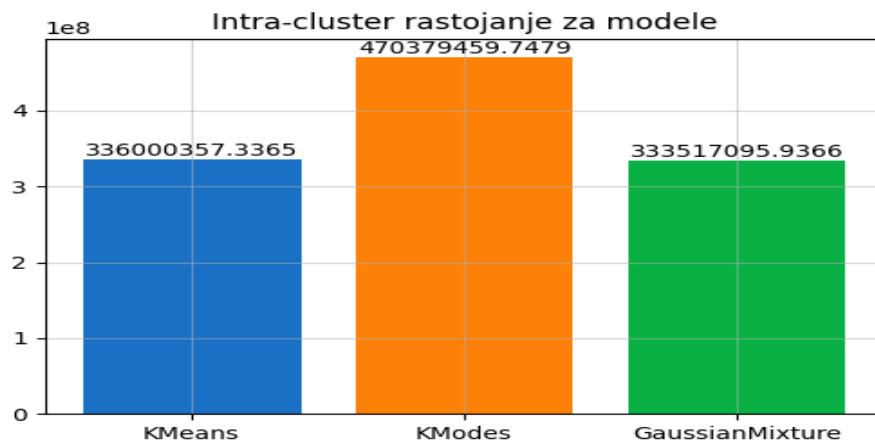
Lose ocene ukazuju na rasprsenost podataka i nisku gustinu "klastera". Vec nakon vizualne analize primenom *pairplot*-a, bilo je jasno da u skupu podataka nemamo klastera.

Rand indeks je metrika koja se koristi za merenje slicnosti izmedju predvidjenih klastera i stvarnih klasa (ukoliko su dostupne). Ova metrika ocenjuje koliko su tacno klasteri modela uskladjeni sa pravim klasama. Vrednosti *Rand indeksa* se kreću u opsegu od 0 do 1. Rand indeks za modele:



Lose ocene ukazuju na slabo podudaranje klastera i pravih klasa koje su korisne za svrhe klasifikacije.

Intra-klaster rastojanje je metrika koja se koristi za merenje prosečne udaljenosti između objekata unutar istog klastera. Ova metrika ocenjuje kompaktnost i gustinu klastera tako što računa prosečno rastojanje između svih parova objekata unutar istog klastera. Manje vrednosti intra-klaster rastojanja ukazuju na bolje klasterovanje, jer ukazuju na to da su objekti unutar klastera blizi jedni drugima. Intra-klaster rastojanje za modele:



Rezultat ukazuje na slabu gustinu unutar "klastera".

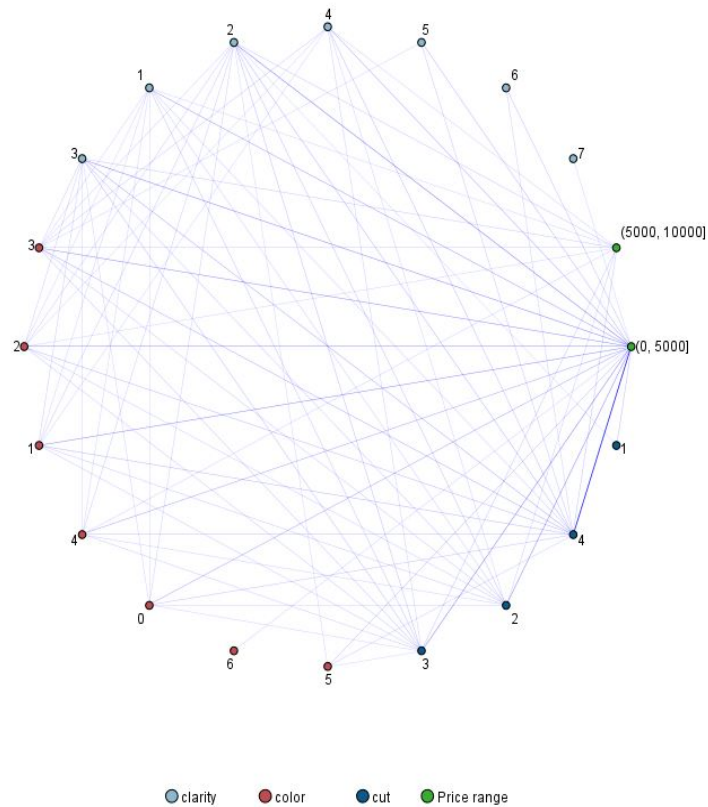
Na osnovu svih razmatranih evaluacija modela klasterovanja i nakon vizualne analize predviđenih klastera, lako možemo zaključiti da u skupu podataka ne postoje klasteri.

Rezultati pravila pridruzivanja

Apriori algoritam nam je pokazao sledece rezultate za kategoricke attribute *clarity*, *color*, *cut* i njihovu vezu sa *class* atributom:

Consequent	Antecedent	Support %	Confidence %	Lift
Price range = (0, 5...	color = 1	18.163	82.382	1.133
Price range = (0, 5...	color = 0	12.56	81.063	1.115
Price range = (0, 5...	cut = 4	39.954	76.869	1.057
Price range = (0, 5...	color = 2	17.69	75.959	1.045
Price range = (0, 5...	cut = 2	22.399	72.331	0.995
Price range = (0, 5...	clarity = 2	24.221	71.496	0.983
Price range = (0, 5...	clarity = 4	15.148	71.068	0.977
Price range = (0, 5...	color = 3	20.934	70.262	0.966

Izdvojeno je 8 pravila. Sva pravila predviđaju da se instance nalaze u prvom opsegu. Granice za *support* i *confidence* su 10% i 70%. Ako pogledamo i *lift*, mozemo zakljuciti da nam nijedno pravilo nije preterano interesantno, te na nasem skupu podataka nema uocljivih pravila od nekog velikog znacaja. Maksimalan support je nesto manje od 40% a maksimalan confidence je nesto vise od 82%. Jedino pravilo od nekog znacaja je prvo, da iz color=1 sledi pripadanje klasi (0, 5000]. Ni graf mreze nam ne pruza neke zanimljive podatke:



4 Zaključak

Nakon primenjivanja razlicitih tehnika istrazivanja podataka i masinskog učenja na skupu podataka *diamonds* iz biblioteke *tensorflow_datasets*, istrenirano je oko 10 modela i preispitane su karakteristike svakog od njih.

Sto se skupa podataka tice, podaci ispunjavaju kvantitativne i kvalitativne mere te se mogu koristiti za pouzdano predviđanje cenovnih klasa dijamanta, u opsegu do 20 000USD.

Klasifikacija nam je pokazala da nam je najbolji model za nas skup podataka **RandomForestClassifier**.

Klasterovanje nam je pokazalo da ne postoje nikakve grupacije na skupu podataka.

Pravila pridruzivanja su nam samo potvrdila ocigledno, da nisko kvalitetni dijamanti teze tome da kostaju manje.