UNIVERZITET U SARAJEVU

ELEKTROTEHNIČKI FAKULTET

RAČUNARSTVO I INFORMATIKA

IZVJEŠTAJ

PRVI PROJEKTNI ZADATAK

MAŠINSKO UČENJE

STUDENTI:

- Begović Amila (2031/18608)
- Leka Elma (2035/18661)
- Panjeta Eldar (2016/18711)

SARAJEVO

11. DECEMBAR, 2022

SADRŽAJ

SPECIFIKACIJA PROJEKTA

Projekat se radi u timu. Svaki tim se sastoji od 3-5 studenta. Sastavi timova određeni su u koordinaciji sa predmetnim asistentom.
Implementacija: korištenjem Python googlecolab okruženja.
Prije početka rada na projektu u potpunosti pročitati specifikaciju projekta jer su taskovi međusobno povezani. Specifikacija projekta prati gradivo predmeta (predavanja i vježbe).

ZADATAK 1

Definicija zadatka 1.

1.1. Analizirati dobijeni set podataka sa kreditnim informacijama– dimenzije, broj i namjenu varijabli.

1.2. Primjenom odgovarajućih metoda potrebno je analizirati pojedinačno sve varijable:

- Analizitati tipove varijabli, distribuciju, deskriptivnu statistiku (srednju vrijednost, standardnu devijaciju,…) korelaciju između parova varijabli,….
- Vizualizirajte proces istraživanja podataka za svaku varijablu i jasno argumentujte zapažanja i zaključke o svakoj varijabli.

1.3. Izvršiti odgovarajuće transformacije podataka, te uklanjanje nepodobnih vrijednosti da bi dobili podatke koji su spremni za korištenje u većini metoda klasifikacije.

Analiza i transformacija dataset-a

Korištene metode

Za početnu analizu dataseta korišteno je nekoliko metoda, čije se korištenje i implementacija može vidjeti u colab-u pod poglavljem “Analiza svih podataka prije transformacije”. Metode koje su korištene za analizu:

- `shape` - govori nam s kojim brojem redova i kolona radimo.
- `dtypes` - metoda koja nam govori o kojoj vrsti podataka je riječ u svakoj koloni.
- `describe` - metoda koja nam daje deskriptivnu statistiku za sve kolone unutar dataseta
- `unique` - metoda koja nam daje sve moguće vrijednosti za kolonu
- Za vizualizaciju podataka smo koristili boxplot-ove, histogram-e, te vizuelnu korelaciju između parova varijabli.

Analiza podataka prije transformacije

Unutar početnog dataset-a smo imali 30 kolona i 100022 reda. Pomoću prethodno korištenih metoda smo zaključili da dosta kolona imaju nepodobne vrijednosti i da se trebaju transformisati. Posebno je od koristi bila metoda `unique` preko koje smo mogli vidjeti o kojim je podacima tačno riječ u koloni i porediti ih sa tipom podataka u kolonama.

Postojalo je nekoliko kolona koje bi po imenu, i vrijednostima podataka unutar kolone, trebale da budu numeričkog tipa, a one su ustvari objekt tipa. Sve te kolone je trebalo pronaći i transformisati u numeričke kolone. Razlog za transformaciju u numeričke vrijednosti je u tome što se numeričke varijable mogu vizualizirati preko boxplot-a i tako je lakši pronalazak outlier-a.

Neke kolone smo razdvojili u dvije ili više kolona, da bismo se riješili nepodobnih vrijednosti, ali i izvukli više informacija iz kolona. Npr. kolona Name_and_Age je imala grupisane vrijednosti za ime i broj godina. Tu kolonu smo razdvojili na dvije i uočili da Age ima nepodobnih vrijednosti koje smo dalje transformisali. Također, sada je bilo moguće da se Age vizualizira.

Analiza i transformacija podataka po kolonama

Unutar colab-a smo kreirali poglavlje “Analiza i tranformacija podataka po kolonama” u kojima smo odvojili na podpoglavlja svaku analizu i transformaciju koju smo radili. Tu se vidi za svaku transformisanu kolonu objašnjenje, kod za transformaciju i analiza podataka nakon transformacije (vizualizacija, deskriptivna statistika, jedinstvene vrijednosti).

Ukratko objašnjen rad nad svakoj transformisanoj koloni se može vidjeti na narednoj tabeli (Napomena: svaka kolona u tabeli odgovara podpoglavlju u colab okruženju, gdje se može pronaći kod za transformaciju i analiza nakon transformacije. Ukoliko kolone ne postoji u tabeli, nad njom nije rađena nikakva transformacija):

TRANSFORMACIJA	OPIS
Transformacija kolona Unnamed: 0, Unnamed: 0.1, Unnamed: 0.1.1	Unutar dataset-a se nalaze 3 Unnamed kolone. Iz razloga što su "unnamed" ne možemo izvući nikakve informacije o njima, te ćemo ih uklonuti iz dataset-a.
Transformacija kolone "Month”	Unutar kolone "Month" nalaze se vrijednosti za mjesec u obliku naziva mjeseca, ali i u obliku rednog broja. Potrebno je transformisati podatke da imaju isti oblik. Mi smo uzeli da transformišemo redni broj u naziv mjeseca, iz razloga što ima manje kolona koje trebamo transformisati, te je naziv mjeseca intuitivniji od rednog broja.
Transformacija kolone "Name_and_Age”	Unutar kolone "Name_and_Age" se nalaze informacije o imenu i godinama korisnika unutar dataset-a. Te informacije kad su zajedno ne možemo koristiti ni za kakve proračune. Mi smo tu kolonu podijelili na dvije: Name i Age. Te smo primijetili da neke godine nisu "tačne". Tj. broj godina ne može biti negativan broj, a kako je maksimalan broj godina koji je dostignut: 122 godine i 164 dana, za gornju granicu smo stavili 123 godine, te sve redove koji ne spadaju unutar intervala [0, 123) smo zamijenili sa srednjom vrijednošću godina. Nakon svih potrebnih transformacija izbacili smo i kolonu Name_and_Age
Transformacija kolone "SSN"	Unutar kolone "SSN" nalaze se vrijednosti za social security number. One su jedinstvene za sve kolone i podaci nam ne daju nikakve relevantne informacije. Te kolonu možemo obrisati.
Transformacija kolone "Occupation”	Unutar kolone "Occupation" nalaze se vrijednosti različog formata. Formati: lowercase, UPPERCASE, CamelCase. Mi smo dataset transformisali u Titel case. Unutar dataset se nalazi i vrijednost '_______'. Tu smo vrijednost pretvorili u 'No occupation'.
Transformacija kolone "Annual_Income"	Unutar kolone "Annual_Income" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju donju crticu na kraju

	vrijednosti. Te donje crtice smo uklonuli, te vrijednosti prebacili u float. Kolona sadrži dosta jedinstvenih vrijednosti, pa smo kolonu dodatno transformisali na način da smo kolonu grupisali po Customer_ID, te za svaku grupu zamijenili vrijednost Annual Incoma sa mean vrijednošću, a prije toga smo NaN vrijednosti zamijenili sa mean vrijednošću.
Transformacija kolone "Monthly_Inhand_Salary"	Unutar kolone "Monthly_Inhand_Salary" nalazi se dosta outliera. Njih ćemo riješiti grupisanjem po CustomerId, i određivanjem mode za grupu.
Transformacija kolone "Num_Bank_Accounts"	Unutar kolone "Num_Bank_Accounts" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju donju crticu na kraju vrijednosti. Te donje crtice smo uklonuli, te vrijednosti prebacili u float. Kolona sadrži dosta jedinstvenih vrijednosti, pa smo kolonu dodatno transformisali na način da smo kolonu grupisali po Customer_ID, te za svaku grupu zamijenili vrijednost Annual Incoma sa mean vrijednošću. Pošto ova kolona predstavlja količinu, brojeve smo prebacili u cijele.
Transformacija kolone "Num_Credit_Card"	Unutar kolone "Num_Credit_Card" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju donju crticu na kraju vrijednosti. Te donje crtice smo uklonuli, te vrijednosti prebacili u float. Kolona sadrži dosta jedinstvenih vrijednosti, pa smo kolonu dodatno transformisali na način da smo kolonu grupisali po Customer_ID, te za svaku grupu zamijenili vrijednost Num_Credit_Card sa mean vrijednošću. Pošto ova kolona predstavlja količinu, brojeve smo prebacili u cijele.
Transformacija kolone "Interest_Rate"	Unutar kolone "Interest_Rate" nalazi se dosta outliera. Njih ćemo riješiti grupisanjem po CustomerId, i određivanjem mean za grupu.
Transformacija kolone "Num_of_Loan"	Unutar kolone "Num_of_Loan" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju dodatan tekst uz vrijednost. Taj tekst smo uklonuli i kolonu prebacili u float vrijednosti. NaN vrijednosti smo zamijenili sa 0, dok smo se riješili nekih outliera mijenjanjem vrijednosti kolona tako što smo grupisali kolone po Customer_ID, te izračunali mean vrijednosti. Pošto ova kolona predstavlja količinu, brojeve smo prebacili u pozivine cijele.
Transformacija kolone "Type_of_Loan"	Unutar kolone "Type_of_Loan" nalaze se vrijednosti koji su stringovi koji govore koje sve vrste zajmova postoje za taj red, ali od tih podataka ne možemo izvući nikakve korektne informacije. Pa smo iz tog razloga kreirali kolone koje sadrže informaciju da li taj određeni zajam postoji za određeni red. Nakon izvlačenja novih kolona obrisali smo staru kolonu Type of Loan
Transformacija kolone "Num_of_Delayed_Payment"	Unutar kolone "Num_of_Delayed_Payment" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju dodatan tekst uz vrijednost. Taj tekst smo uklonuli i kolonu prebacili u float vrijednosti. U koloni postoje i negativne i NaN vrijednosti. Kolonu smo grupisali po CustomerId i uzeli srednju vrijednost Num_of_Delayed_Payment za grupu da bismo transformisali te vrijednosti.

Transformacija kolone "Changed_Credit_Limit"	Unutar kolone "Changed_Credit_Limit" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju dodatan tekst uz vrijednost. Taj tekst smo uklonuli i kolonu prebacili u float vrijednosti. Unutar kolone je postojala i vrijednost _. Tu smo vrijednost prvo transformisali u NaN, da bismo kolonu mogli prebaciti u float. Nakon čega smo vrijednosi NaN, transformisali u mean vrijednost kolone.
Transformacija kolone "Num_Credit_Inquiries"	Unutar kolone "Num_Credit_Inquiries" nalazi se dosta outliera. Njih ćemo riješiti grupisanjem po CustomerId, i određivanjem mean za grupu.
Transformacija kolone "Credit_Mix"	Unutar kolone "Credit_Mix" nalaze se 4 različite vrijednosti. Sve su intuitivne, sem vrijednosti '_'. Tu smo vrijednost zamijenili sa "Unknown"
Transformacija kolone "Outstanding_Debt"	Unutar kolone "Outstanding_Debt" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju dodatan tekst uz vrijednost. Taj tekst smo uklonuli i kolonu prebacili u float vrijednosti.
Transformacija kolone "Credit_History_Age"	Unutar kolone "Credit_History_Age" nalaze se vrijednosti koje su stringovi, ali predstavljaju broj godina i mjeseci. S tom informacijom u obliku stringa, ne možemo raditi ništa. Te smo vrijednosti prebacili u broj mjeseci pomoću pomočne metode.
Transformacija kolone "Payment_of_Min_Amount"	Unutar kolone "Payment_of_Min_Amount" nalaze se kategoricke varijable. Postoje dvije YES i NO varijable, te True i NM varijable. True var ćemo prebaciti u YES, a NP u No.
Transformacija kolone "Total_EMI_per_month"	Unutar kolone "Total_EMI_per_month" nalazi se dosta outliera. Njih ćemo riješiti grupisanjem po CustomerId, i određivanjem mean za grupu.
Transformacija kolone "Amount_invested_monthly"	Unutar kolone "Amount_invested_monthly" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju dodatan tekst uz vrijednost. Taj tekst smo uklonuli i kolonu prebacili u float vrijednosti. Kolonu smo grupisali po CustomerId i uzeli srednju vrijednost Num_of_Delayed_Payment za grupu da bismo transformisali vrijednosti.
Transformacija kolone "Payment_Behaviour"	Unutar kolone "Payment_Behaviour" nalaze se vrijednosti koje su stringovi, koji predstavljaju vrstu payment behaviour. Nakon toga smo kreirali 5 novih kolona koje sadrže informaciju o vrsti payment behaviour. Nakon izvlačenja novih kolona obrisali smo staru kolonu Payment_Behaviour.
Transformacija kolone "Monthly_Balance"	Unutar kolone "Monthly_Balance" nalaze se vrijednosti koje su stringovi, iz razloga što neke vrijednosti imaju dodatan tekst uz vrijednost. Taj tekst smo uklonuli i kolonu prebacili u float vrijednosti. Kolonu smo grupisali po CustomerId i uzeli srednju vrijednost Num_of_Delayed_Payment za grupu da bismo transformisali vrijednosti.

Analiza podataka nakon transformacije

Nakon sveukupne transformacije opisane u prethodnom poglavlju, obavili smo opet analizu cijelih podataka, koja se u colabu urađena pod poglavljem “Analiza podataka nakon transformacije”. Nakon transformacije u dataset-u je ostao isti broj redova, ali sad imam nešto više kolona, tj. 39. Dosta kolona smo prebacili iz tipa object u neki numerički tip, pa nakon transformacije sada imamo 17 numeričkih kolona.

Obavili smo vizualizaciju za dataset nakon transformacije podataka i prikazali sveukupnu deskriptivnu statistiku za svaku varijablu, iako smo to uradili i prilikom transformacije.

Kao zadnji korak u prvom zadatku, očišćeni dataset smo spremili u novi csv fajl koji će se koristiti kao polazni izvor podataka u zadatku 2.

ZADATAK 2

Definicija zadatka 2.

Potrebno je izgraditi klasifikacijski model na osnovu kreditnih podataka pojedinca za klasifikaciju kreditnog rezultata.

2.1. Radi se grupno. (2 bod) Upoznali smo se sa metodama mašinskog učenja za klasifikaciju/predikciju podataka: drvo odlučivanja, Bayes, SVM mašina, neuralne mreže, k-nn, regresija,… Odaberite skup algoritama (broj algoritama treba da odgovara broju članova tima) pogodan za izgradnju navedenog klasifikacijskog modela. U grupnom dogovoru svakom članu tima se dodjeljuje po jedan algoritam.

2.2. Radi se individualno. (8 bod) Za svaki algoritam, odgovorni član tima (iz tačke 2.1.) vrši izgradnju klasifikacijskog modela što uključuje sljedeće:

- Analizirajte ponovo podatke i odlučite da li treba dodatna priprema seta podataka za primjenu konkretnog algoritma.
- Opišite i implementirajte proces izgradnje klasifikacijskog modela za odgovarajući problem (podjelu seta podataka na trening, testni i validacijski set podataka).
- Evaluirajte model pomoću konfuzijske matrice (tačnost, specifičnost, osjetljivost, kappa statistika, Mean Absolute Error za regresiju, itd). Koristiti metode unakrsne validacije za dodatnu evaluaciju modela.
- Analizirajte potencijalni uticaj nebalansiranosti podataka na izgrađeni model i na osnovu toga primijeniti metode za oversampling i/ili undersampling.
- Uradite tuning parametara.

Detaljno dokumentujte i diskutujete urađeno u zadatku 2.2.

2.3. Radi se grupno. (5 bod). Analizirajte i diskutujte performanse svakog modela (zašto je neki model dao bolje rezultate na osnovu neke metrike). Nakon toga predložite i implementirajte ensamble model klasifikacije za problem koji će doprinijeti boljim performansama. Validirajte predloženi ensamble model klasifikacije. Napomena: Prilikom izgradnje modela formirati odgovarajući pipeline. U cilju poboljšanja performansi nemojte zaboraviti da je proces izgradnje modela mašinskog učenja iterativan proces.

Odabir klasifikacijskih modela

Pošto je bilo potrebno da svaki član tima odabere po jedan klasifikacijski model, mi smo se odlučili za Drvo odlučivanja, KNN i Native Bayes.

Zašto smo odabrali Drvo odlučivanja kao jedan od klasifikatora?

Vrlo jednostavan, ali efikasan klasifikator koji olakšava cinjenicom da postoje razne vizualizacije drveta s ciljem boljeg percipiranja podataka te je veoma brz klasifikator koji dobro radi sa velikim podacima kao što je slučaj sa ovim datasetom.

Mana klasifikatora je u tome da je podložan overfittingu i može se desiti da ima baš puno grana i konačno teško je postići značajnu optimizaciju.

Zašto smo odabrali KNN algoritam kao jedan od klasifikatora?

KNN je poprilično intuitivan i jednostavan algoritam. Može se lahko razumjeti kako on funkcioniše. Glavni razlog zašto smo odabrali ovaj algoritam jeste zato što KNN odlično radi sa višeklasnim problemima a naš data-set jeste upravo takav. Osim toga KNN je algoritam koji konstantno evolvira i ima nekoliko mogućih varijanti parametara koje možemo koristiti.

Međutim, KNN ima i određene mane. PrIje svega, čitav dataset je bilo potrebno pretvoriti u numeričke podatke te ukoliko se ta transformacija ne obavi kako treba moguće je izgubiti mnogo korisnih informacija. Osim toga algoritam je poprilično spor i osjetljiv na outlier-e, tako da se oni moraju riješiti prije treniranja.

Zašto smo odabrali Naive Bayes algoritam kao jedan od klasifikatora?

Naive Bayes ima dosta klasifikatora, te je moguće kreirati i analizirati razne modele veoma jednostavno. Pored toga, klasifikator je veoma brz i može dati bolje rezultate nego ostali modeli, te ne zahtjeva puno podataka.

Mana algoritma je da podatke smatra samostalnim, pa njegove estimacije ne treba smatrati previše ozbiljnim.

Drvo odlučivanja

Radio/la: Leka Elma

Analiza podataka

Prije početka rada neophodno je izlistati tipove podataka i jedinstvene vrijednosti finalnog dataseta kreiranog u prvom zadatku. Ukoliko analiziramo pomenute podatke i osobine možemo doći do zaključka da korišteni dataset i dalje nije pogodan za kreiranje Decision Tree stoga ce u nastavku biti izvršene moge izmjene nad podacima gdje će se oni svrstavati u određene kategorije (binove). Razlog odabira takvog pristupa jeste činjenica da se radi o općepoznatim terminima koji se mogu svrstati u određene okvire te nema potrebe da se posmatraju kao pojedinačne instance brojeva. Sama kreacija binova doprinosi poboljšanju tačnosti Decision Tree.

Transformacija

Prva izmjena učinjena nad kodom jeste da su ID, Customer_ID i Name dropane iz razloga sto ne doprinose datasetu, osim toga posjeduju veliki broj različitih vrijednosti koje se ne bi trebale pojavljivati u drvu odlučivanja.

Potrebno je takođre da izvršimo izmjenu kategoričkih podataka na način da ih pretvorimo u jedinstvene numeričke s tipom int. Ovakav pristup “ručnog” enkodiranja je izabran iz razloga što ne postoji mnogo kategoričkih podataka, a osim toga prilikom kreacije binova zadale su se okvirne vrijednosti koje su analogne kategorijama te da bi se zadrzala uniformnost ostavlja se takav pristup.

U nastavku će biti opisan način svrstavanja u binove, kako je riječ o općepoznatim vrijednostima moguće ih je bilo svrstati u logične skupine s tim da su označene brojevima 0,1,2 … koji su analogni kategorijama za Low, Average… i drugima.

Annual Income posjeduje veliki broj različitih vrijednosti, no kako je riječ o primanjima na nivou godine, može se svrstati u 6 okvirnih grupa a to su Very Low, Low, Average, Above Average, High i Very High gdje se ove grupe odnose na visinu primanja.

Monthly Inhand Salary posjeduje veliki broj različitih vrijednosti, no kako je riječ o primanjima na nivou mjeseca, može se svrstati u 6 okvirnih grupa, a to su Very Low, Low,

Average, Above Average, High i Very High gdje se ove grupe odnose na visinu mjesečnih primanja.

Interest Rate posjeduje veliki broj različitih vrijednosti, no kako je riječ okamati na nivou godine, može se svrstati u 6 okvirnih grupa, a to su Very Low, Low, Average, Above Average, High i Very High gdje se ove grupe odnose na visinu kamate.

Changed Credit Limit posjeduje veliki broj različitih vrijednosti, no kako je riječ o mogućem posuđivanjem, može se svrstati u 5 okvirnih grupa, a to su Very Low, Low, Average, High i Very High gdje se ove grupe odnose na visinu novca koju može posuditi.

Outstanding Debt posjeduje veliki broj različitih vrijednosti, no kako je riječ o dugu, može se svrstati u 6 okvirnih grupa, a to su Very Low, Low, Average, Above Average, High i Very High gdje se ove grupe odnose na visinu novca koju osoba/kompanija duguje.

Credit Utilization Ratio posjeduje veliki broj različitih vrijednosti, no kako je riječ o dugu, može se svrstati u 6 okvirnih grupa, a to su Very Low, Low, Average, Above Average, High i Very High gdje se ove grupe odnose na visinu novca koju osoba/kompanija duguje.

Age posjeduje veliki broj različitih vrijednosti, no kako je riječ o godinama, može se svrstati u 3 okvirne grupe, a to su Young Adult, Adult i Senior gdje se ove grupe odnose na broj godina.

Monthly Balance posjeduje veliki broj različitih vrijednosti, no kako je riječ o minimalnoj vrijednosti koju treba posjedovati na računu, može se svrstati u 5 okvirnih grupa, a to su Very Low, Low, Average, High i Very High gdje se ove grupe odnose na visinu novca.

Amount invested monthly posjeduje veliki broj različitih vrijednosti, no kako je riječ o vrijednosti novca koja se investira, može se svrstati u 5 okvirnih grupa, a to su Very Low, Low, Average, High i Very High gdje se ove grupe odnose na visinu novca.

Total EMI per month posjeduje veliki broj različitih vrijednosti, no kako je riječ o fiksnoj vrijednosti novca koja uplaćuje mjesečno pozajmivaču, može se svrstati u 3 okvirne grupe, a to su Low, Average, High gdje se ove grupe odnose na visinu novca.

Credit History Age posjeduje veliki broj različitih vrijednosti, no kako je riječ o broju godina koliko je račun otvoren, može se svrstati u 4 okvirne grupe, a to su Young Age, Average Age, Older, Abnormaly Old gdje se ove grupe odnose na broj godina.

Num Bank Accounts posjeduje veliki broj različitih vrijednosti, no kako je riječ o broju računa koji su otvoreni, može se svrstati u 5 okvirnih grupa, a to su Low, Average, High i Very High gdje se ove grupe odnose na broj računa.

Num Credit Card posjeduje veliki broj različitih vrijednosti, no kako je riječ o broju kreditnih kartica, može se svrstati u 4 okvirnih grupa, a to su Low, Average, High i Very High gdje se ove grupe odnose na broj računa.

Num of Loan posjeduje veliki broj različitih vrijednosti, no kako je riječ o broju pozajmnica, može se svrstati u 3 okvirne grupe, a to su Normal, Average i High gdje se ove grupe odnose na broj računa.

Delay from due date posjeduje veliki broj različitih vrijednosti, no kako je riječ o kašnjenju uplate od željenog datuma, može se svrstati u 3 okvirnih grupa, a to su No Delay, Average Delay and Late Delay gdje se ove grupe odnose na broj dana zakašnjenja.

Num of Delayed Payment posjeduje veliki broj različitih vrijednosti, no kako je riječ o broju dana zakasnjenja, može se svrstati u 4 okvirne grupe, a to su Low Num of Delays, Mediun Num of Delays, Average Num of Delays and High Num of Delays gdje se ove grupe odnose na broj provjera.

Važno je napomenuti da je i labela klase enkodirana na način da je 0 za Poor, 1 za Standard i 2 za Good.

Evaluacija modela

Nakon pripreme i analize podataka pristupa se treniranju klasifikatora te se potom model evaluira. Evaluacija podrazumijeva nekoliko karakteristika koje se provjeravaju te koje će biti prikazane i objašnjene u nastavku. U nastavku prikazana evaluacija tiče se **** modela no evaluacija će biti rađena i nakon tuning psrametara ali i nakon oversamplinga i undersamplinga

Osjetljivost ili True positive Rate ili recall govori koliko je pozitivnih instanci korektno model zaista ispravno prepoznao, cilj je imati što višu vrijednost.

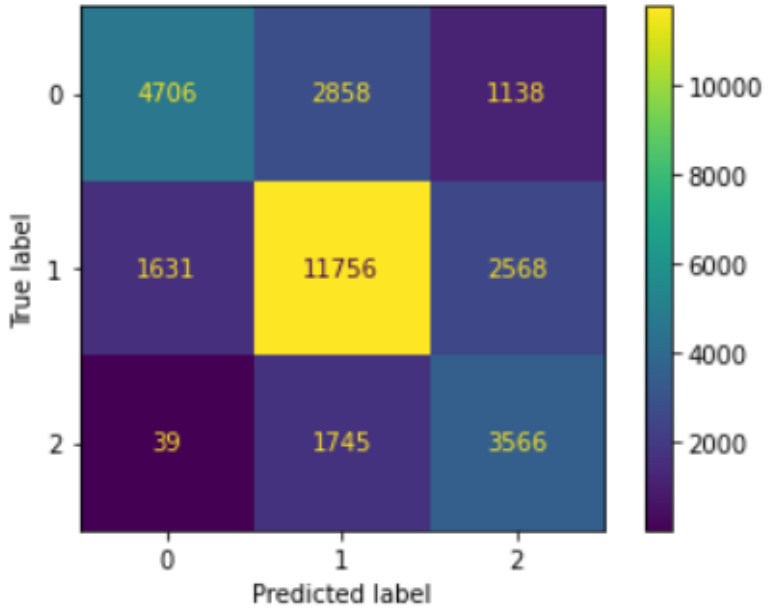
Specifičnost govori koliko je mogel True Negative ispravno prepoznao, također i za ovu vrijednost je cilj da bude što viša.

Preciznost govori o svim instancama koje je model kao tačne u odnosu sa onima koje su zaista tačne i prepoznate kao takve.

F1 mjera za cilj ima prikazati balansiranost između osjetljivosti i preciznosti.

Kappa statistika ocjenjuje pouzdanost “rater-a”, te se koristi i za ocjenjivanje tačnosti s tim kada je tačnost veća i kappa statstika će biti veća.

```
Konfuzijska matrica:
[[ 4706  2858  1138]
 [ 1631 11756  2568]
 [   39  1745  3566]]
Tacnost:
0.6674442630052988
Stepen greske:
0.3325557369947012
Osjetljivost:
0.6480531961063871
Specificnost:
0.8145811565648722
Preciznost:
0.6490267238026151
Opoziv (recall):
0.6480531961063871
F-score:
0.6389587339575692
Kappa statistika:
0.4505923154971434
```



Kao još jedan dio evaluacije i analize rezultata možemo pogledati izvještaj. Bitno je analizirati klasifikacijski izvještaj.

Prediction govori o procentu tačno pozitivnih u odnosu na sve pozitivne. Na osnovu prediction mozemo zaključiti da je od očekivanih Credit Score za Poor samo 74% zaista bilo Poor, analogno 69% za Standard i konačno 54% Good.

Recall govori o odnosu tačno pozitivnih u odnosu na ukupan broj zaista pozitivnih. Recall govori da je 55% Credit Scora označenih zaista sa Poor bilo prepoznato od strane modela kao Poor. Tako analogno 79% za Standard i 54% za Good.

F1 score govori o odnosu precision i recall te se može izraziti sa 2 * (Precision * Recall) / (Precision + Recall)

Konačno Support nam govori koliko Credit Scora pripada kojoj skupini, jasno je da najviše pripada Standard dok najmanje pripada Good. Upravo će to biti razlog zbog kojeg će se primjeniti overfitting i underfitting na dataset.

Za kross validaciju se koristila Stratified Folds zajedno sa pipelinom.

Tuning parametara

Za tuning parametara se koristi GridSearchCV pri čemu se za svaki parametar odredi okvir vrijednosti koje će testirati za pronalazak najbolje kombinacije parametara. Odabrani raspon za parametre je:

```
'max_depth':range(10,20,5),'min_samples_leaf':range(5,150,50),
'min_samples_split':range(5,150,50), 'criterion':["entropy","gini"].
```

Nakon što su se odrediti najbolji hiperparametri trenira se klasifikator i dobivena tačnost je:

Tacnost: 0.6874729229846369

Oversampling

Oversampling će se odraditi pomocu SMOTE. Posebnu pažnju treba obratiti na činjenicu da se oversampling radi nad trening podacima. Razlog za ovo jeste činjenica da bi se moglo desiti da ukoliko overfittamo čitav skup podataka mnogo sintetičkih podataka uđe u trening skup te klasifikator nam da lažnu nadu za potencijali uspijeh.

Oversampling je doveo do sljedeće količine podataka:

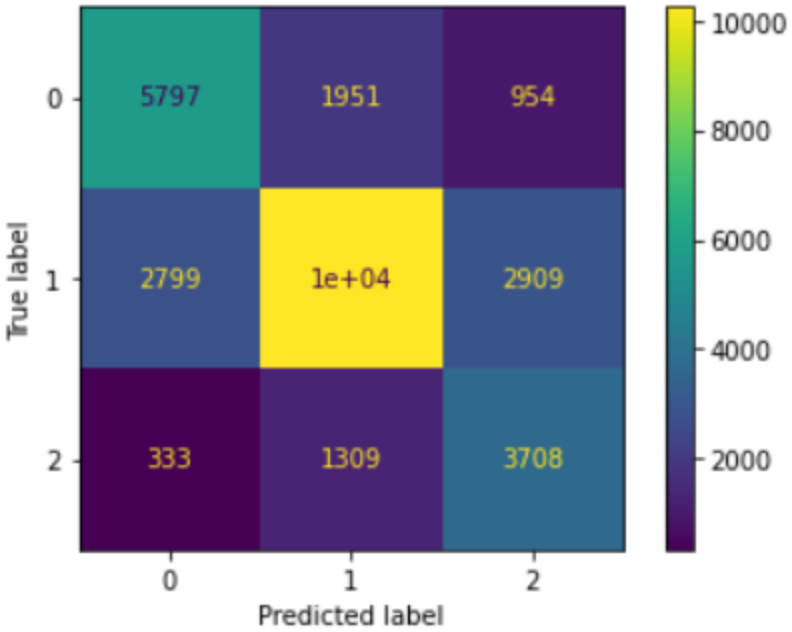
```
Broj instanci u originalnom datasetu:  (70015,)
Broj instanci u datasetu nakon oversamplinga:  (111681,)
Resampled dataset shape Counter({2: 37227, 1: 37227, 0: 37227})
```

Nakon oversamplinga radi se konačno i tuning podataka za oversampled trening podatke. Dobiva se sljedeće:

```
Najbolja tačnost: 0.72332842523096
Najbolji estimator:
DecisionTreeClassifier(max_depth=15, min_samples_leaf=5,
min_samples_split=5)
```

Za dobivene hiperparametre se trenirao klasifikator i dobili su se sljedeći podaci:

```
Konfuzijska matrica:
[[ 5797  1951   954]
 [ 2799 10247  2909]
 [  333  1309  3708]]
Tacnost:
0.6582464091711934
Stepen greske:
0.34175359082880663
Osjetljivost:
0.6671655399061582
Specificnost:
0.8214424347446231
Preciznost:
0.6325466915312933
Opoziv (recall):
0.6671655399061582
F-score:
0.6423830094149409
Kappa statistika:
0.45700389935710006
```



Može se primjetiti da očekivana tačnost nakon tuninga i dobivena sa istim hiperparametrima nije ona koju smo očekivali, zbog toga se radi i Stratified cross validacija nakon koje možemo vidjeti da su podaci zaista kao što je i očekivano:

```
Fold:  1, Accuracy: 0.687
Fold:  2, Accuracy: 0.687
Fold:  3, Accuracy: 0.672
Fold:  4, Accuracy: 0.711
Fold:  5, Accuracy: 0.736
Fold:  6, Accuracy: 0.749
Fold:  7, Accuracy: 0.757
Fold:  8, Accuracy: 0.753
Fold:  9, Accuracy: 0.758
Fold: 10, Accuracy: 0.758

Cross-Validation accuracy: 0.727 +/- 0.033
```

Konačni zaključak je da smo zaista uspjeli dobiti željeno: zahvaljujući tuningu i oversamplingu podataka tačnost modela se povećala sa 0.669 +/- 0.006 na 0.727 +/- 0.033.

K-nearest neighbor (KNN)

Radio/la: Panjeta Eldar

Analiza podataka

Prvobitna analiza podataka je izvršena u prvom projektnom zadatku. Otkrivene su i otklonjene nepodobne vrijednosti. Također, bilo je dosta podataka sa neispravnom strukturom te su i takvi podaci transformisani ili otklonjeni. Kreiran je novi dataset.

Za KNN algoritam, potrebno je da svi podaci budu numeričke vrijednosti. Ovo znači, da je neophodno sve kategoričke vrijednosti pretvoriti u numeričke.

Transformacija podataka

Korištenjem funkcija dtypes i unique možemo saznati koje kolone imaju kategoričke vrijednosti i koje su moguće vrijednosti unutar tih kolona. Kolone koje je potrebno transformisati su: Month, Occupation, Credit_Mix te Payment_of_Min_Amount. Ove transformacije su urađene u sklopu funkcije transformNumericUpgraded. Kolona Occupation je pretvorena u nekoliko novih kolona koje imaju True (1) ili False (0) vrijednosti, dok su vrijednosti iz ostalih kolona intuitivno pretvorene u numeričke vrijednosti. Također, obrisane su kolone Name, ID i Customer_ID jer nam ne daju bitne informacije za predikciju. Ispod je prikazana funkcija koja izvršava ove transformacije.

```
def transformNumericUpgraded(df):
  # kolone Customer_ID, ID, Name možemo obrisati
  df = df.drop(['Customer_ID', 'ID', 'Name'], axis=1)

  # pretvaramo kolonu Month u numericku
  df.loc[df["Month"] == "January", "Month"] = 1
  df.loc[df["Month"] == "February", "Month"] = 2
  df.loc[df["Month"] == "March", "Month"] = 3
  df.loc[df["Month"] == "April", "Month"] = 4
  df.loc[df["Month"] == "May", "Month"] = 5
  df.loc[df["Month"] == "June", "Month"] = 6
  df.loc[df["Month"] == "July", "Month"] = 7
  df.loc[df["Month"] == "August", "Month"] = 8
  df['Month'] = df['Month'].astype('int')

  # Kreiranje kolona za svaki tip posla posebno
  df['Teacher'] = np.where(df['Occupation'] == 'Teacher', True, False)
  df['Writer'] = np.where(df['Occupation'] == 'Writer', True, False)
  df['Lawyer'] = np.where(df['Occupation'] == 'Lawyer', True, False)
```

```
df['Scientist'] = np.where(df['Occupation'] == 'Scientist', True, False)
df['Musician'] = np.where(df['Occupation'] == 'Musician', True, False)
df['Manager'] = np.where(df['Occupation'] == 'Manager', True, False)
df['Entrepreneur'] = np.where(df['Occupation'] == 'Entrepreneur', True, False)
df['Journalist'] = np.where(df['Occupation'] == 'Journalist', True, False)
df['Accountant'] = np.where(df['Occupation'] == 'Accountant', True, False)
df['Mechanic'] = np.where(df['Occupation'] == 'Mechanic', True, False)
df['Architect'] = np.where(df['Occupation'] == 'Architect', True, False)
df['Engineer'] = np.where(df['Occupation'] == 'Engineer', True, False)
df['Developer'] = np.where(df['Occupation'] == 'Developer', True, False)
df['Doctor'] = np.where(df['Occupation'] == 'Doctor', True, False)
df['No occupation'] = np.where(df['Occupation'] == 'No occupation', True, False)
df = df.drop(['Occupation'], axis=1)

# pretvaramo kolonu Credit_Mix u numericku
df.loc[df["Credit_Mix"] == "Unknown", "Credit_Mix"] = 0
df.loc[df["Credit_Mix"] == "Good", "Credit_Mix"] = 3
df.loc[df["Credit_Mix"] == "Standard", "Credit_Mix"] = 2
df.loc[df["Credit_Mix"] == "Bad", "Credit_Mix"] = 1
df['Credit_Mix'] = df['Credit_Mix'].astype('int')

# pretvaramo kolonu Payment_of_Min_Amount u numericku
df.loc[df["Payment_of_Min_Amount"] == "No", "Payment_of_Min_Amount"] = 0
df.loc[df["Payment_of_Min_Amount"] == "Yes", "Payment_of_Min_Amount"] = 1
df['Payment_of_Min_Amount'] = df['Payment_of_Min_Amount'].astype('int')

return df
```

Kreiranje modela

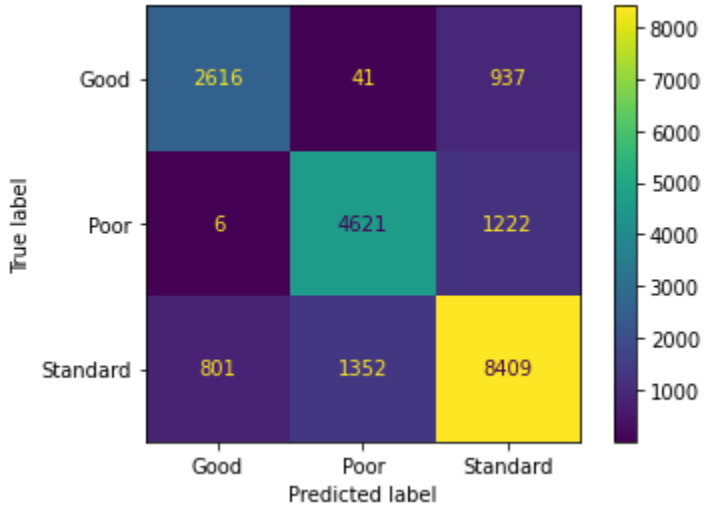
Model je kreiran sa standardnim (default) parametrima za KNN:

- Weights: uniform
- Algorithm: auto
- Leaf_size: 30
- Metric: Minkowsky

Te je za k odabrana vrijednost 5. Podaci su podijeljeni na sljedeći način: 80% za treniranje i 20% za testiranje.

Model 1 - bez skaliranih podataka

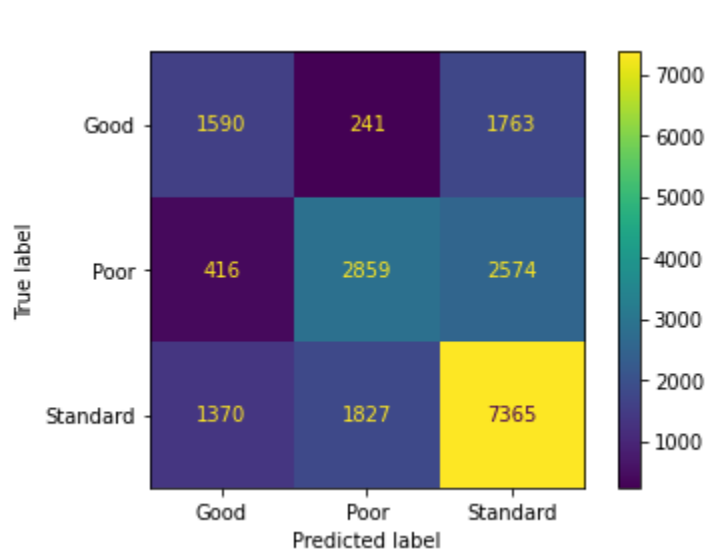
Prvi model kreiran je sa običnim (neskaliranim podacima). Ovaj model je dao tačnost od 78%. U nastavku je prikazana konfuzijska matrica i određene metrike koje govore o kvaliteti modela.




```
Tacnost:
0.7821044738815296
Stepen greske:
0.2178955261184704
Osjetljivost:
0.7713618039486043
Specificnost:
0.8745957313988143
Preciznost:
0.7761065665559443
Opoziv (recall):
0.7713618039486043
F-score:
0.7735362295275855
Kappa statistika:
0.6383208935217849
```

Model 2 - podaci skalirani min-max transformacijom

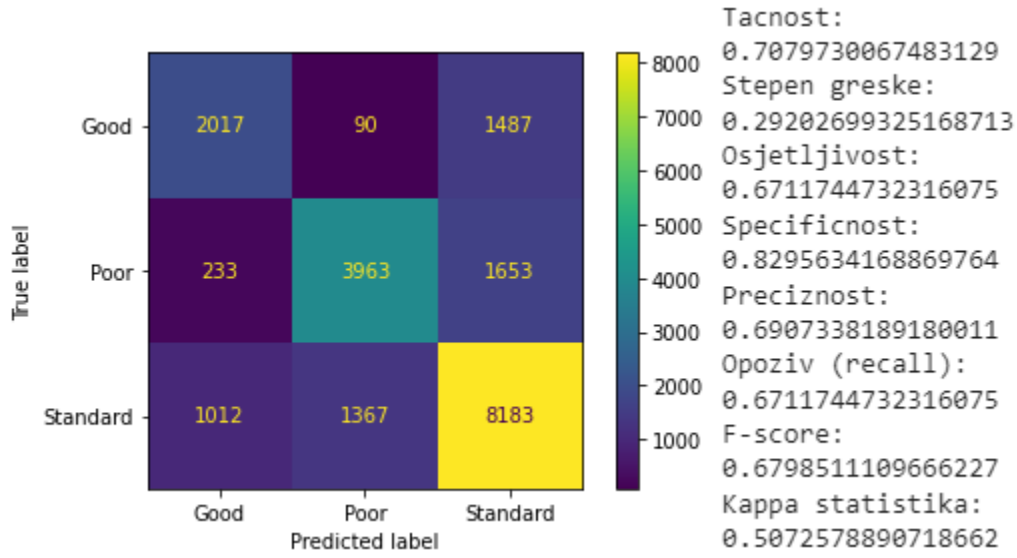
Drugi model kreiran je nad skaliranim podacima. Korištena je min-max transformacija. Na ovaj način je pokušano dobiti veću tačnost, međutim tačnost se značajno smanjila.




```
Tacnost:
0.5905523619095226
Stepen greske:
0.40944763809047735
Osjetljivost:
0.5428388755091845
Specificnost:
0.761934027798179
Preciznost:
0.560207709328753
Opoziv (recall):
0.5428388755091845
F-score:
0.5494902737990889
Kappa statistika:
0.3046489661748617
```

Model 3 - podaci skalirani z-score transformacijom

Treći model kreiran je nad skaliranim podacima koristeći z-score transformaciju. I u ovam slučaju, tačnost je smanjena.



Primjećeno je da skaliranje podataka daje dosta lošije rezultate. Jedan od razloga zašto se ovo dešava može biti to da su određene kolone “vrijednije” od drugih pri predikciji, pa skalirane svih kolona daje lošije rezultate jer su onda svi istog ranga (za min-max transformaciju).

Modeli 4 i 5 - undersampling i oversampling metode

Modeli 4 i 5 su kreirani nad podacima koji su smanjeni (undersampling) i povećani (oversampling), respektivno. Rezultati ponovo nisu sjajni. Korištenjem undersampling metode sa strategijom majority, dobijena je tačnost od 63%. Na slici ispod je prikazan broj redova za treniranje prije i poslije primjene undersampling metode te dobijena tačnost.

```
(array(['Good', 'Poor', 'Standard'], dtype=object), array([14238, 23159, 42620]))
(array(['Good', 'Poor', 'Standard'], dtype=object), array([14238, 14238, 14238]))
0.6350412396900775
```

Korištenjem uversampling metode sa strategijom minority, dobijena je tačnost od 75%. Na slici ispod je prikazan broj redova za treniranje prije i poslije primjene oversampling metode te dobijena tačnost.

```
(array(['Good', 'Poor', 'Standard'], dtype=object), array([14238, 23159, 42620]))
(array(['Good', 'Poor', 'Standard'], dtype=object), array([42620, 42620, 42620]))
0.7560109972506873
```

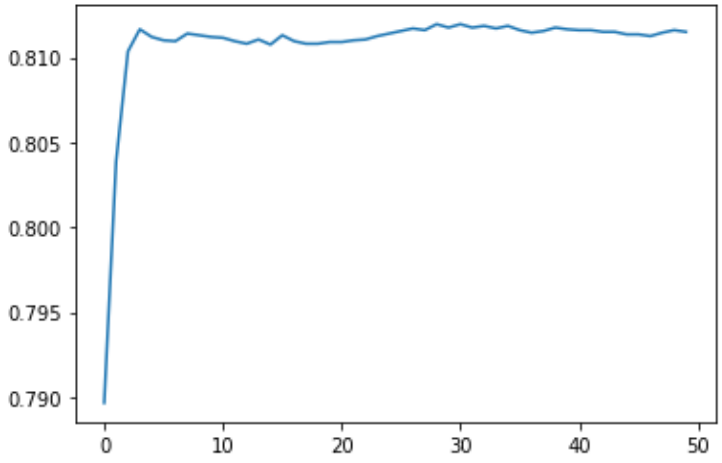
Nadogradnja modela

U nastavku će biti prikazana pokušaj nadogradnje modela 1 jer je on dao najveću tačnost. Zbog dugog izvršavanja procesa treniranja, tuning parametara je izvršen ručno i nisu uzimane sve moguće kombinacije parametara jer kada bi se to radilo, proces bi predugo trajao.

Nakon treniranja nekoliko modela sa različitim postignuti su sljedeći rezultati:

	weights	algorithm	n_neighbors	metric	tačnost
Model 1	distance	auto	5	minkowsky	0.78365
Model 2	distance	ball_tree	5	minkowsky	0.80775
Model 3	distance	kd_tree	5	minkowsky	0.80775
Model 4	distance	brute	5	minkowsky	0.78365
Model 5	distance	ball_tree	5	cityblock	0.81035
Model 6	distance	ball_tree	5	euclidean	0.80775
Model 7	distance	ball_tree	5	manhattan	0.81035
Model 8	distance	ball_tree	5	l1	0.81035
Model 9	distance	ball_tree	5	l2	0.80775

Modeli 5, 7 i 8 daju najbolju tačnost. Ostale je još da se izvrši tuning parametra n_neighbors. Ostali parametri će biti kao i u modelu 8. Na slici ispod možemo vidjeti graf tačnosti u zavisnosti od n, gdje n ima vrijednosti od 1 do 101 ali samo neparni brojevi (59 vrijednosti). Najveća tačnost iznosi 0.81195, što je približno 82%. Ova tačnost je dobijena za n_neighbors = 61.

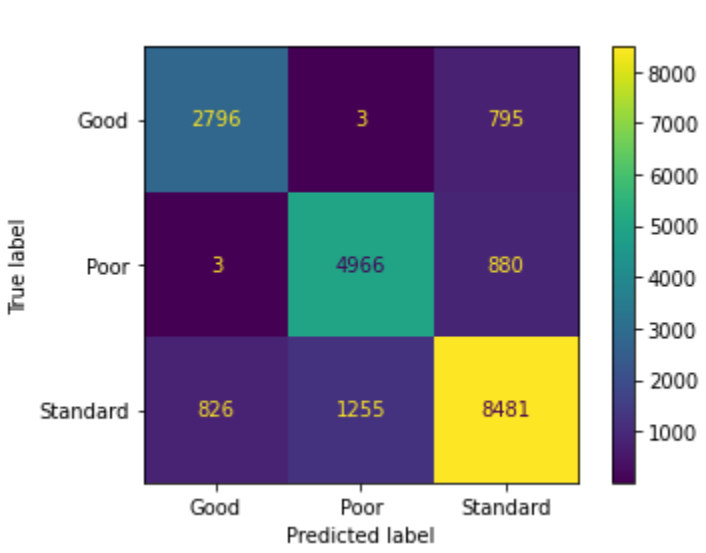


Tako da sada možemo reći da model ima najveću tačnost sa sljedećim parametrima:

- Weights: distance
- Algorithm: ball_tree
- Leaf_size: 30 (default)
- Metric: cityblock / manhattan / l1

Dok je n_neighnors (k) = 61.

Ostale mjere kvaliteta i konfuzijska matrica su dati ispod




```
Tacnost:
0.8119470132466883
Stepen greske:
0.18805298675331172
Osjetljivost:
0.8099900722750714
Specificnost:
0.894412706668787
Preciznost:
0.8014207951791662
Opoziv (recall):
0.8099900722750714
F-score:
0.8053310106951738
Kappa statistika:
0.6909288190464222
```

Cross validacija

Da bi pokazali da je tuningom parametara zaista povećana tačnost modela, izvršena je cross validacija prije i poslije tuninga parametara (gdje je broj fold-ova 30) sa neskaliranim podacima.

Prije tuninga

```
Fold:  1, Accuracy: 0.780
Fold:  2, Accuracy: 0.746
Fold:  3, Accuracy: 0.780
Fold:  4, Accuracy: 0.747
Fold:  5, Accuracy: 0.760
Fold:  6, Accuracy: 0.775
Fold:  7, Accuracy: 0.778
Fold:  8, Accuracy: 0.772
Fold:  9, Accuracy: 0.761
Fold: 10, Accuracy: 0.782
Fold: 11, Accuracy: 0.781
Fold: 12, Accuracy: 0.747
Fold: 13, Accuracy: 0.773
Fold: 14, Accuracy: 0.769
Fold: 15, Accuracy: 0.773
Fold: 16, Accuracy: 0.768
Fold: 17, Accuracy: 0.766
Fold: 18, Accuracy: 0.775
Fold: 19, Accuracy: 0.754
Fold: 20, Accuracy: 0.761
Fold: 21, Accuracy: 0.781
Fold: 22, Accuracy: 0.761
Fold: 23, Accuracy: 0.744
```

```
Fold: 24, Accuracy: 0.782
Fold: 25, Accuracy: 0.765
Fold: 26, Accuracy: 0.779
Fold: 27, Accuracy: 0.782
Fold: 28, Accuracy: 0.750
Fold: 29, Accuracy: 0.756
Fold: 30, Accuracy: 0.752


Cross-Validation accuracy:
0.767 +/- 0.013
```

Poslije tuninga

```
Fold:  1, Accuracy: 0.805
Fold:  2, Accuracy: 0.820
Fold:  3, Accuracy: 0.816
Fold:  4, Accuracy: 0.815
Fold:  5, Accuracy: 0.800
Fold:  6, Accuracy: 0.816
Fold:  7, Accuracy: 0.808
Fold:  8, Accuracy: 0.808
Fold:  9, Accuracy: 0.816
Fold: 10, Accuracy: 0.811
Fold: 11, Accuracy: 0.818
Fold: 12, Accuracy: 0.802
Fold: 13, Accuracy: 0.802
Fold: 14, Accuracy: 0.801
Fold: 15, Accuracy: 0.804
Fold: 16, Accuracy: 0.800
Fold: 17, Accuracy: 0.804
Fold: 18, Accuracy: 0.810
Fold: 19, Accuracy: 0.801
Fold: 20, Accuracy: 0.817
Fold: 21, Accuracy: 0.816
Fold: 22, Accuracy: 0.793
Fold: 23, Accuracy: 0.808
Fold: 24, Accuracy: 0.811
Fold: 25, Accuracy: 0.797
Fold: 26, Accuracy: 0.811
Fold: 27, Accuracy: 0.807
Fold: 28, Accuracy: 0.802
Fold: 29, Accuracy: 0.801
Fold: 30, Accuracy: 0.801


Cross-Validation accuracy:
0.807 +/- 0.007
```

Sada se jasno vidi da je tačnost nakon tuninga povećana, i to sa 0.767 +/- 0.013 na 0.807 +/- 0.007.

Naive Bayes

Radio/la: Begović Amila

Analiza podataka

Prvi korak pri kreiranju za bilo kojeg modela za treniranje podataka, jeste analiza podataka. Prvobitna analiza podataka je obavljena u zadatku 1 ovog projektnog zadatka, kao i transformacija za rješavanje nepodobnih vrijednosti. Nakon svih analiza i transformacija originalnog dataseta, kreiran je novi csv fajl transformisanog dataseta, te se isti koristi kao početni dataset.

Struktura dataset-a je većinski tipa ‘number’, ali sadrži i nekoliko kolona tipa ‘object’. Naivni Bayes, koji će biti rađen u ovom dijelu, može raditi striktno s podacima tipa ‘number’, te je iz tog razlog apotrebno transformisati dataset, da bi odgovarao modelu.

Ali prilikom istraživanja Naivnog Bayesa, došli smo do informacije da Naivni Bayes, najbolje radi sa kategoričkim podacima. Iz tog razloga se u ovom projektu obradio NB i nad numeričkim i nad kategoričkim podacima da bi se mogla vidjeti razlika.

Transformacija

Transformacija za Bayes-a je urađena na dva načina. Kreirana su dva različita dataset-a za treniranje i testiranje modela. Prvi model koji je kreiran je numerički, dok je drugi kategorički.

Prva transformacija

Unutar dataset-a imamo sedam kategoričkih kolona, tipa stirng. Te kolone trebamo transformisati u tip int. Prva transformacija koja je obavljenja jeste OneHotEncoding za kolone Month, Occupation, Credit_Mix i Payment_of_Min_Amount. Nakon enkodiranja, sve nepotrebne kolone su obrisane. Pored navedenih, tu spadaju i Customer_ID, ID i Name. Zadnja transformacija je obavljena nad kolonom Credit_Score. To je kategorička kolona nad kojom smo radili predikciju. Kolona sadrži tri različite vrijednosti (Good, Standard i Bad), te nismo mogli obaviti enkodiranje, da se ne bi kreirale dodatne kolone. Kolonu smo transformisali ručno, na način da smo svakoj varijabli unutar kolone dodijelili numeričku vrijednosti (Good = 3, Standard = 2, Bad = 1).

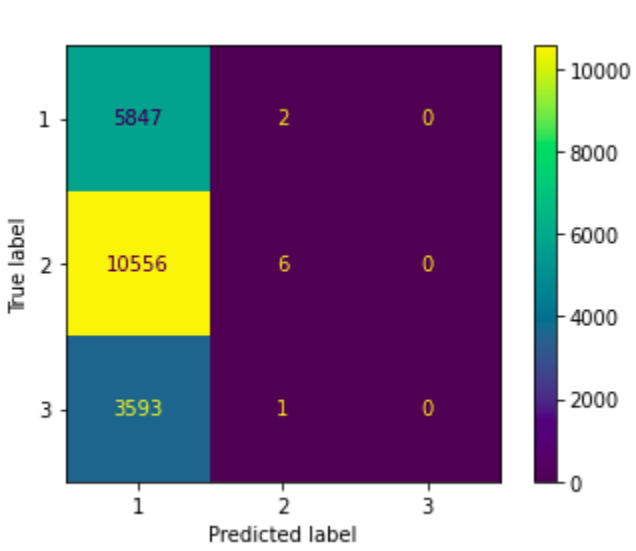
Druga transfromacija

Unutar dataset-a postoji dosta numeričkih kolona, koje smo pretvorili u kategoričke tipa int. Te smo sve kategoričke kolone enkodirali kao u prethodnom primjeru. Sve nepotrebne kolone su obrisane, a kolona labele je transformisana na isti način kao i u prethodnom poglavlju. Nakon svih transformacija u tom novom dataset-u sada postoji 119 kolona.

Kreiranje modela koristeći numeričke podatke

Model bez skaliranja sa Gaussian klasifikatorom

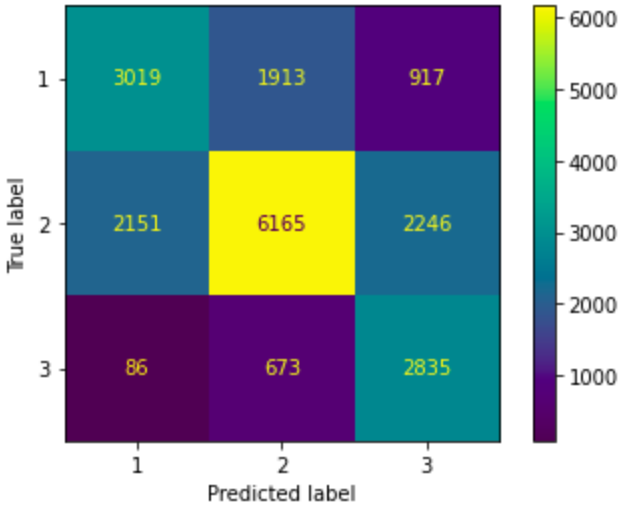
Za prvo kreirani model nije obavljeno nikakvo skaliranje i korišten je Gaussian klasifikator za kreaciju podataka. Taj model je dao najmanje izračunatu predikciju sa 29.26% tačnosti.




```
Tacnost:
0.29257685578605347
Stepen greske:
0.7074231442139465
Osjetljivost:
0.33340871181180326
Specificnost:
0.666725598107116
Preciznost:
0.31969171612100195
Opoziv (recall):
0.33340871181180326
F-score:
0.15120060304348532
Kappa statistika:
0.00013280316934050074
```

Model bez skaliranja sa Bernoulli klasifikatorom

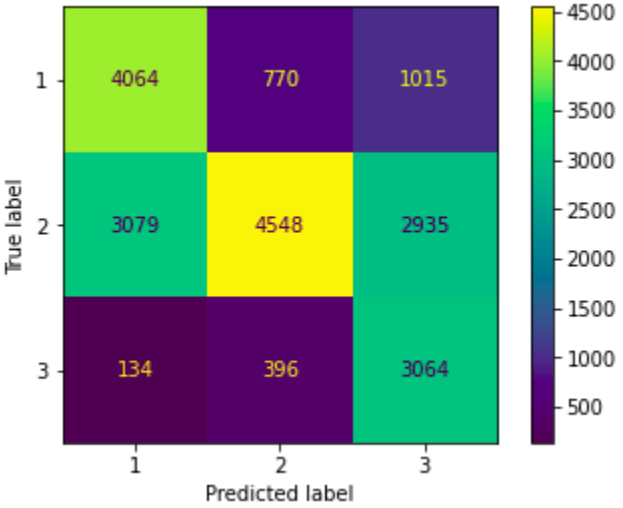
Drugi kreirani model također nije imao skaliranja i kreiran je sa Bernoulli klasifikatorom. Taj model je dao najveću izračunatu tačnost od 60.08%.




```
Tacnost:
0.6007998000499875
Stepen greske:
0.39920019995001255
Osjetljivost:
0.629555856254998
Specificnost:
0.7917949689633094
Preciznost:
0.5838465466117936
Opoziv (recall):
0.629555856254998
F-score:
0.5910889054937205
Kappa statistika:
0.3746499602989056
```

Model sa min-max skaliranjem sa Gaussian klasifikatorom

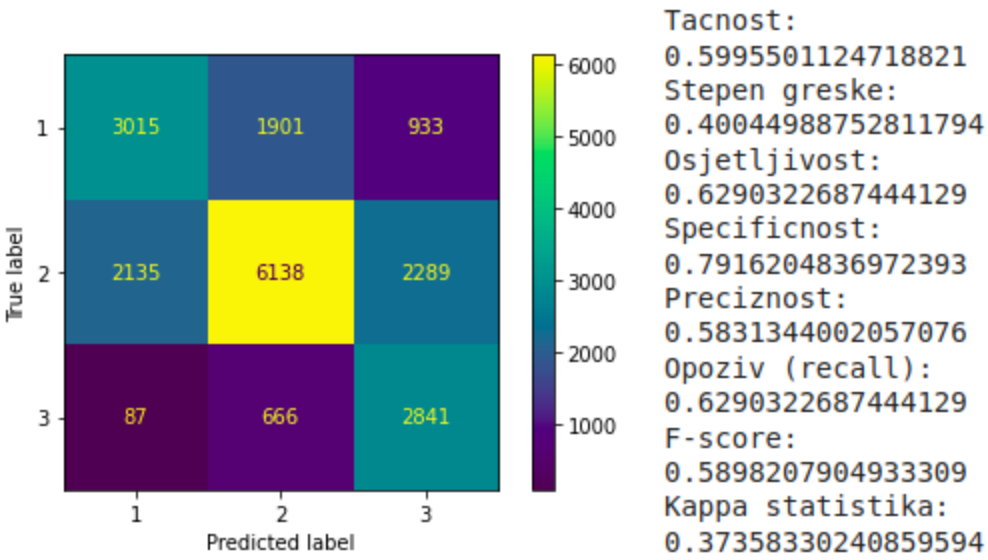
Podaci za predikciju su transformisani koristeći min-max transformaciju, za ovaj model. Te se koriteći Gaussian klasifikator uspjela postići preciznost od 58.36%, što je skoro duplo bolje od prvo kreiranog modela.




```
Tacnost:
0.5836540864783804
Stepen greske:
0.41634591352161965
Osjetljivost:
0.6593172967206966
Specificnost:
0.8029530590925011
Preciznost:
0.5970840997517303
Opoziv (recall):
0.6593172967206966
F-score:
0.5852553021799224
Kappa statistika:
0.3875943910168931
```

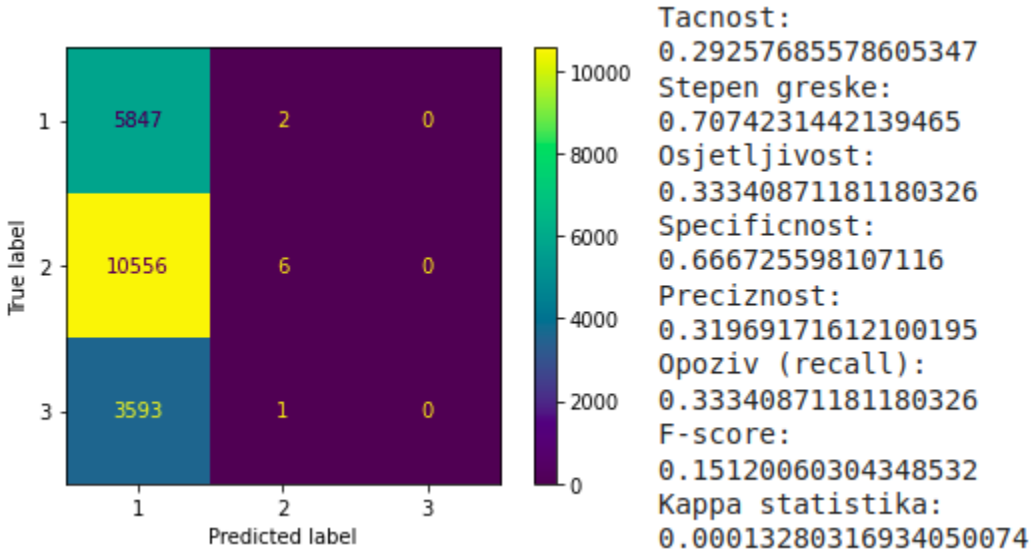
Model sa min-max skaliranjem sa Bernoulli klasifikatorom

Podaci za predikciju su transformisani koristeći min-max transformaciju, za ovaj model. Te se koriteći Bernoulli klasifikator uspjela postići nešto manja preciznost od drugog kreiranog modela, sa 59.95%.



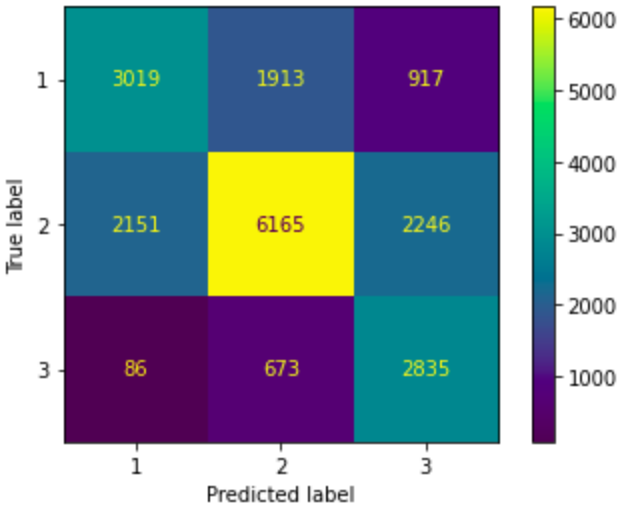
Model transformisan preko z-skaliranja sa Gaussian klasifikatorom

Ukoliko za Gaussian klasifikator koristimo z-skaliranje za podatke, može se primijetiti da je tačnost ostala ista kao i za model bez skaliranja, tj model je imao 29.26% tačnost.



Model transformisan preko z-skaliranja sa Bernoulli klasifikatorom

Ukoliko za Bernoulli klasifikator koristimo z-transformaciju za podatke, može se primijetiti da je tačnost ostala ista kao i za model bez skaliranja, tj model je imao 60.08% tačnost.

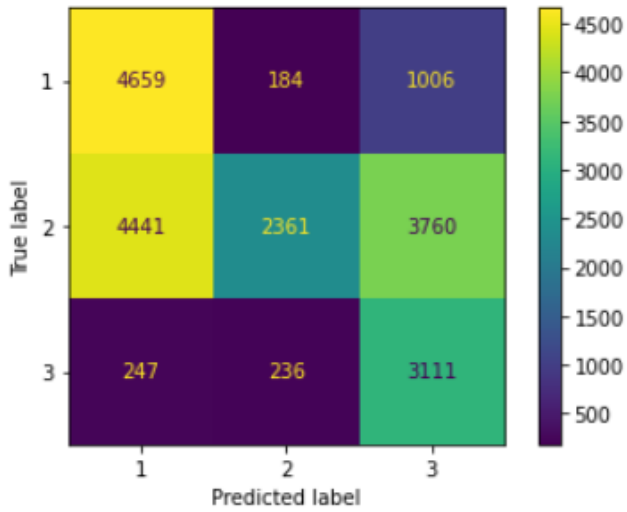



```
Tacnost:
0.6007998000499875
Stepen greske:
0.39920019995001255
Osjetljivost:
0.629555856254998
Specificnost:
0.7917949689633094
Preciznost:
0.5838465466117936
Opoziv (recall):
0.629555856254998
F-score:
0.5910889054937205
Kappa statistika:
0.3746499602989056
```

Kreiranje modela koristeći kategoričke podatke

Model bez skaliranja sa Gaussian klasifikatorom

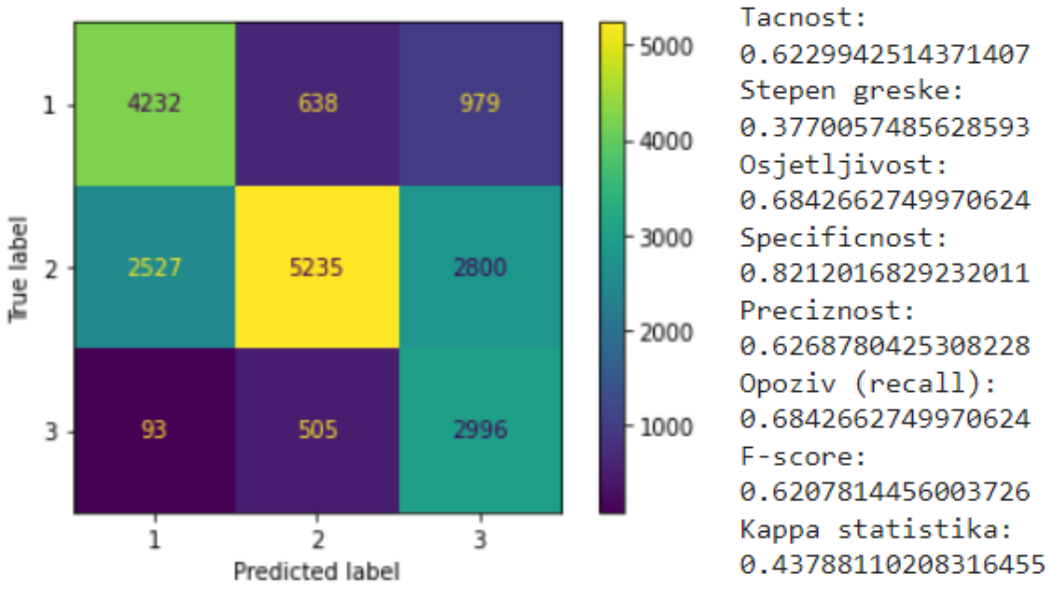
Za prvo kreirani model nije obavljeno nikakvo skaliranje i korišten je Gaussian klasifikator za kreaciju podataka. Taj model je dao predikciju od 50.64% tačnosti.




```
Tacnost:
0.5064233941514621
Stepen greske:
0.4935766058485379
Osjetljivost:
0.6285643253226533
Specificnost:
0.7779802158139112
Preciznost:
0.5807904012887753
Opoziv (recall):
0.6285643253226533
F-score:
0.5031641354299586
Kappa statistika:
0.3137685278311575
```

Model bez skaliranja sa Bernoulli klasifikatorom

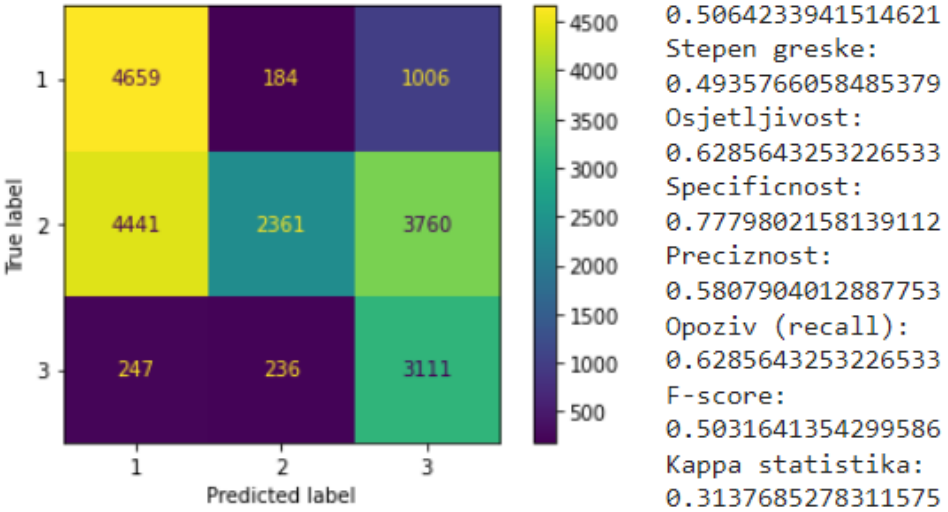
Drugi kreirani model također nije imao skaliranja i kreiran je sa Bernoulli klasifikatorom. Taj model je dao najveću izračunatu tačnost od 62.30%.



Tacnost:
0.6229942514371407
Stepen greske:
0.3770057485628593
Osjetljivost:
0.6842662749970624
Specificnost:
0.8212016829232011
Preciznost:
0.6268780425308228
Opoziv (recall):
0.6842662749970624
F-score:
0.6207814456003726
Kappa statistika:
0.43788110208316455

Model sa min-max skaliranjem sa Gaussian klasifikatorom

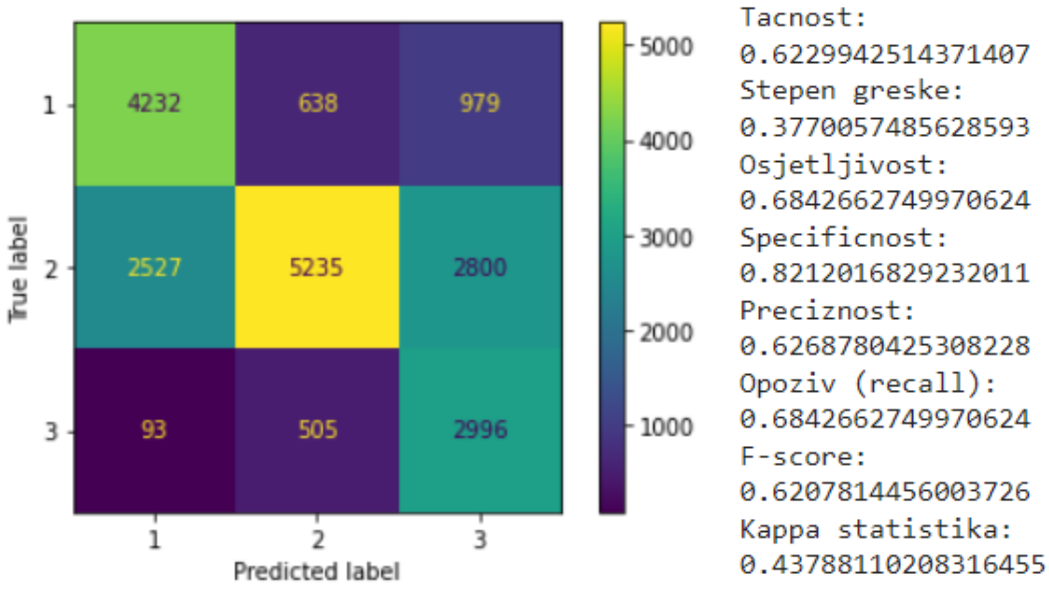
Podaci za predikciju su transformisani koristeći min-max transformaciju, za ovaj model. Te je koriteći Gaussian klasifikato preciznost ostala ista sa 50.64%.



Tacnost:
0.5064233941514621
Stepen greske:
0.4935766058485379
Osjetljivost:
0.6285643253226533
Specificnost:
0.7779802158139112
Preciznost:
0.5807904012887753
Opoziv (recall):
0.6285643253226533
F-score:
0.5031641354299586
Kappa statistika:
0.3137685278311575

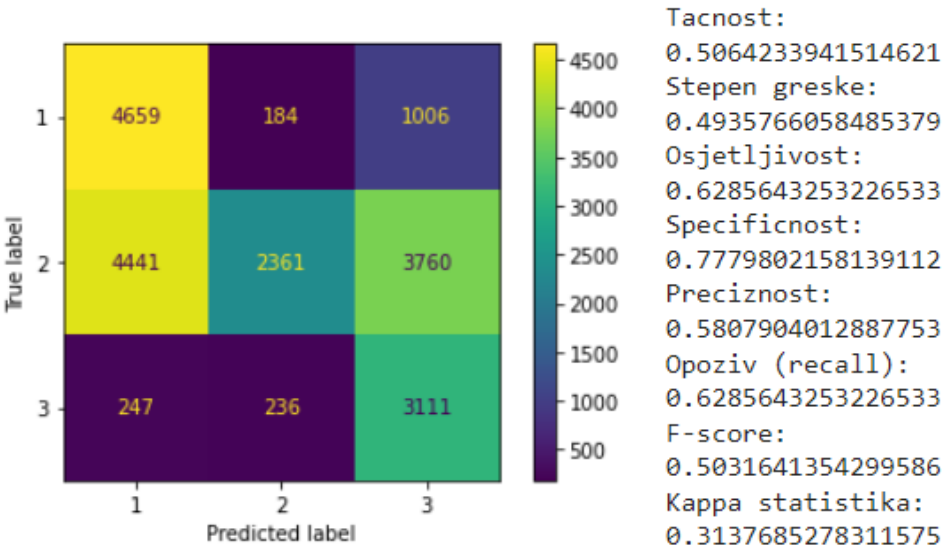
Model sa min-max skaliranjem sa Bernoulli klasifikatorom

Podaci za predikciju su skalirani koristeći min-max skaliranje, za ovaj model. Te se koristeći Bernoulli klasifikator uspjela postići ista preciznost kao i za drugi model.



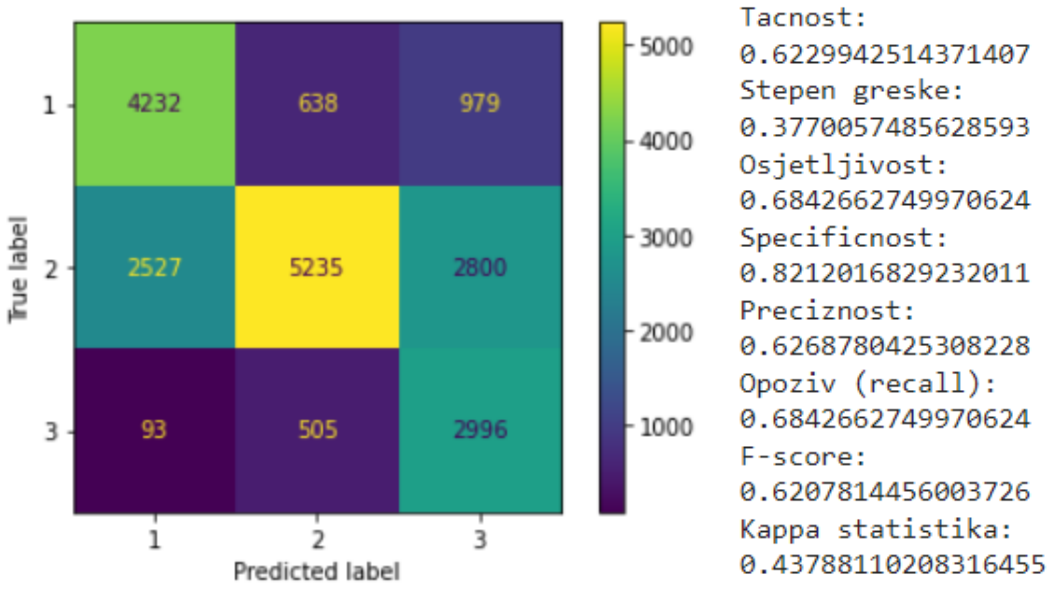
Model transformisan preko z-skaliranja sa Gaussian klasifikatorom

Podaci za predikciju su skalirani koristeći z-skaliranje, za ovaj model. Te je koriteći Gaussian klasifikator preciznost ostala ista sa 50.64%.



Model transformisan preko z-skaliranja sa Bernoulli klasifikatorom

Ukoliko za Bernoulli klasifikator koristimo z-transformaciju za podatke, može se primijetiti da je tačnost ostala ista kao i za model bez skaliranja, tj model je imao 62.30% tačnost.



Zaključci pri kreiranju modela

Prilikom analize modela da se zaključiti da koristeći kategoričke podatke, klasifikatori ne daju različite vrijednosti ukoliko se primijeni skaliranje podataka, te smo koristeći kategoričke podatke uspjeli postići najbolju vrijednost za BernoulliNB klasifikator, od 62.30%.

Za numeričke podatke se može primijetiti da GaussianNB klasifikator daje baš loše rezultate, sa samo 30% tačnosti, ukoliko podaci nisu skalirani sa min-max skaliranjem.

Evaluacija modela

U prethodnim izračunavanjima su odrađene po 3 transformacije podataka nad 2 klasifikatora, za svaki kreirani dataset, tj. ukupno je kreirano 12 modela. Najbolju tačnost je imao model sa BernulliNB klasifikatorom, sa tačnošću od 62.30%, treniran nad kategoričkim podacima. Najbolji model s GaussianNB klasifikatorom je imao 58.4% tačnosti i to model sa min-max transformacijom nad numeričkim podacima. U ovom poglavlju pokušati unaprijediti oba modela

Oversampling model-a sa min-max skaliranjem sa Gaussian klasifikatorom

Oversampling smo uradili sa dvije strategije. Prva strategija sa random oversamplingom strategijom: {1:25000, 2:43000, 3:30000} je dala manju tačnost za 2%. Tj. dobili smo tačnost od 56.56%. Dok je strategija “minority” dala tačnost 56.35%

Undersampling model-a sa min-max sklarianjem sa Gaussian klasifikatorom

Undersampling smo uradili sa dvije strategije. Prva strategija sa random undersampling strategijom: {1:15500, 2:37000, 3:11000} je dala malo veću tačnost. Tj. dobili smo tačnost od 59.12%. Dok je strategija "majority" dala tačnost 58.36%.

Oversampling model-a sa z-skaliranjem sa Bernoulli klasifikatorom

Oversampling smo uradili sa dvije strategije. Prva strategija sa random oversamplingom strategijom: {1:25000, 2:430000, 3:20000} je dala veću tačnost za nešto više 1%. Tj. dobili smo tačnost od 63.68%. Dok je strategija "minority" dala tačnost 60.70%

Undersampling model-a sa z-skaliranjem sa Bernoulli klasifikatorom

Undersampling smo uradili sa dvije strategije. Prva strategija sa random undersampling strategijom: {1:25000, 2:430000, 3:20000} je dala malo manju tačnost. Tj. dobili smo tačnost od 60.70%, kao i strategija "majority".

Zaključak

Zaključak da je za naivnog bayesa, najbolji kreirani model je model sa BernulliNB klasifikatorom, sa tačnošću od 62.30%, treniran nad kategoričkim podacima. Dok se sa oversamplingom ta tačnost uspjela povećati na 63.68%

Također, prvobitno istraživanje je dalo očekivane razultate, da kategorički podaci bolje odgovaraju Native Bayesu od numeričkih. Dok razlog zašto je BernulliNB bolji od GaussianNB leži u tome što ima dosta kolona tipa 'boolean', a ti podaci najbolje odgovaraju BernulliNB klasifikatoru.

Ensambl model

U prethodnom poglavlju su obrađena tri različita validacijska modela, drvo odlučivanja, KNN i naive bayes. KNN model, uz sve njegove mane, je dao najbolji rezultat predikcije odnosno najveću tačnost. Razlog ovoga, može biti to jer KNN je izuzetno efektivan za velike setove podataka a pogotovo za nelinearne podatke jer KNN algoritam ne koristi "pretpostavke" o podacima. Dosta podataka su već bili numerički podaci pa nije bilo potrebe za puno transformacija kojima bi se potencijalno izgubile neke informacije.

Naredni korak u zadataku 2, jeste kreiranje ensambl modela.

Voting klasifikator je model koji se obučava na ensamblu brojnih modela i predviđa izlaz (klasu) na osnovu njihove najveće vjerovatnoće izabrane klase kao izlaza. On jednostavno agregira predikcije svakog klasifikatora koji su proslijeđeni u Voting klasifikator i predviđa klasu izlaza na osnovu najveće većine glasova. Ideja je umjesto stvaranja posebnih modela i pronalaženja tačnosti za svaki od njih, mi kreiramo jedan model koji trenira po ovim modelima i predviđa izlaz na osnovu njihove kombinovane većine glasova za svaku izlaznu klasu. Postoje dvije načina glasanja a to su: hard voting i soft voting. Voting klasifikator najbolje odgovara našim modelima jer podržava sva tri modela.

Za naš projektni zadatak smo kreirali jedan Voting model, ali smo model testirali/trenirali tri puta na različitim podacima.

Priprema podataka za kreiranje modela

Početna priprema se sastoji u tome da kreiramo modele sa parametrima koji su se pokazali najbolji u prethodnom poglavlju. Kreacija modela je prikazana na slijedećoj slici:

```
[32] # modeli
     dt = DecisionTreeClassifier(criterion="gini", random_state=100, max_depth=15, min_samples_leaf=5, min_samples_split=55)
     dtPipeline = Pipeline(steps=[('smote',SMOTE()), ('ss',StandardScaler()), ('dt', dt)])

[33] knn = KNeighborsClassifier(weights = 'distance', algorithm = 'ball_tree', metric='l1', n_neighbors=61)
     knnPipeline = Pipeline(steps=[('knn',knn)])

[34] nb = BernoulliNB()
     nbPipeline = Pipeline(steps=[('nb',nb)])
```

Težina estimatora je podijeljena tako da je najbolji klasifikator, tj KNN, dobio najveću težinu 3, a Naive Bayes koji je imao najmanju tačnost, dobio težinu 1.

```
# estimatori
estimatori = [('dt', dt), ('knn', knn), ('nb', nb)]
weights = [2, 3, 1]
```

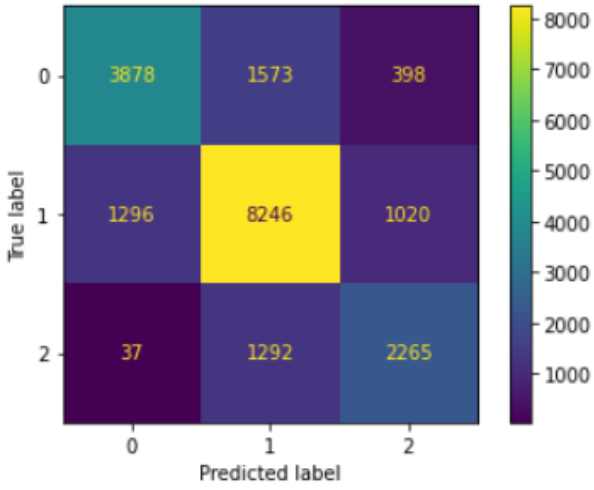
Nakon pripreme početnih podataka, ensambl model je kreiran na slijedeći način:

```
model = VotingClassifier(estimators=estimatori, voting='soft', weights=weights)
```

Prvi ensambl model

Prvi kreirani model je model koji je treniran i testiran nad podacima, koji su kreirani za model: Drvo odlučivanja. Podaci u dataset-u su kategorički, kreirani ručno bez enkodiranja. Podaci najbolje odgovaraju za model Drvo odlučivanja.

Rezultati kreiranog esnambl modela:

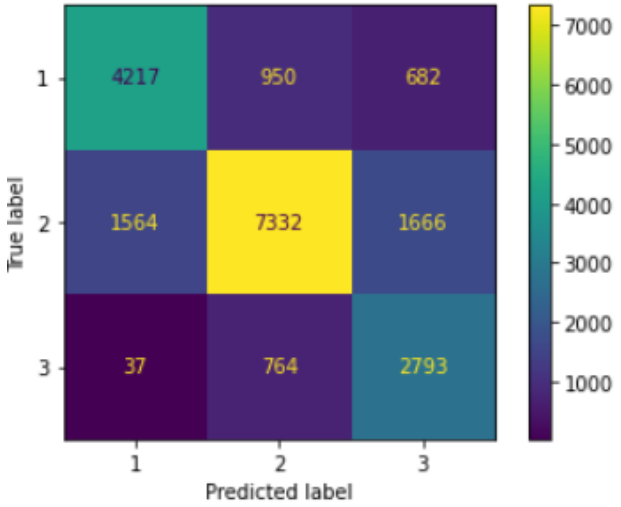



```
Tacnost:
0.7192701824543865
Stepen greske:
0.28072981754561355
Osjetljivost:
0.691319898591074
Specificnost:
0.8386767261505432
Preciznost:
0.7004433917827209
Opoziv (recall):
0.691319898591074
F-score:
0.6949072995837943
Kappa statistika:
0.5301795131332062
```

Drugi ensambl model

Drugi kreirani model je model koji je treniran i testiran nad podacima, koji su kreirani za model: Naive Bayes. Podaci u dataset-u su kategorički, ali umjesto da su kreirani ručno, podaci su kodirani i najbolje odgovaraju za model Naive Bayes.

Rezultati kreiranog ensambl modela:

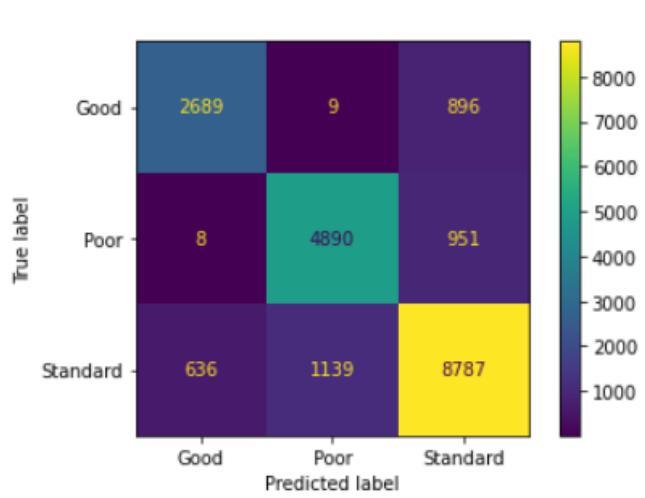



```
Tacnost:
0.7169207698075482
Stepen greske:
0.28307923019245185
Osjetljivost:
0.7307643998634031
Specificnost:
0.8541060665766737
Preciznost:
0.6928743439056534
Opoziv (recall):
0.7307643998634031
F-score:
0.7034159760274642
Kappa statistika:
0.5507105163385386
```

Treći ensambl model

Prvi kreirani model je model koji je treniran i testiran nad podacima, koji su kreirani za model: KNN. Podaci u ovom dataset-u su numerički i najbolje odgovaraju modelu KNN.

Rezultati kreiranog ensambl modela:




```
Tacnost:
0.8180954761309672
Stepen greske:
0.18190452386903277
Osjetljivost:
0.805392162126907
Specificnost:
0.8946890176318778
Preciznost:
0.8143211087329414
Opoziv (recall):
0.805392162126907
F-score:
0.8094161710794783
Kappa statistika:
0.6974163407448812
```

Zaključak za ensambl modele

Kao što se može vidjeti, kreirana su tri modela. Svaki model ima različitu tačnost, ali najbolji ensambl model je treći. Model je postigao tačnost od 81.81%. Razlog zašto je taj model

imao skoro 10% bolju tačnost od ostala dve, leži u tome da podaci korišteni za izgradnju tog modela najbolje odgovaraju KNN klasifikatoru, a kako je KNN samostalno postigao najbolji razultat, tačnost je očekivana.

Svi modeli su također spremljeni u sopstvene .joblib fajlove.