Sveučilište Sjever

Završni rad br. 493/EL/2021

Analiza utjecaja obrade podataka na uspješnost klasifikacijskih algoritama strojnog učenja

David Stević, 1487/336

Varaždin, rujan 2021.

Sveučilište Sjever

Odjel za elektrotehniku

Završni rad br. 493/EL/2021

Analiza utjecaja obrade podataka na uspješnost klasifikacijskih algoritama strojnog učenja

Student

David Stević 1487/336

Mentor

Emil Dumić, izv. prof. dr. sc.

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR-42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL: Odjel za elektrotehniku

STUDIJ: preddiplomski stručni studij Elektrotehnika

PRISTUPNIK: David Stević | JMBAG: 0336015579

DATUM: 26.8.2021 | KOLEGIJ: Signali i sustavi

NASLOV RADA: Analiza utjecaja obrade podataka na uspješnost klasifikacijskih algoritama strojnog učenja

NASLOV RADA NA ENGL. JEZIKU: Effects of data preprocessing on the performance of machine learning classification models

MENTOR: Emil Dumić | ZVANJE: izv. prof. dr. sc.

ČLANOVI POVJERENSTVA:
1. doc. dr. sc. Ladislav Havaš, predsjednik
2. dr. sc. Tomislav Horvat, član
3. izv. prof. dr. sc. Emil Dumić, član
4. doc. dr. sc. Dunja Srpak, zamjenski član
5.

Zadatak završnog rada

BROJ: 493/EL/2021

OPIS:

U ovom radu će biti analizirana uspješnost nekoliko klasifikacijskih algoritama strojnog učenja u ovisnosti o predobradi ulaznih podataka.
Strojno učenje je u današnje vrijeme bitna grana umjetne inteligencije, temeljena na računalnim algoritmima koji koriste modele zasnovane na skupovima podataka. Općenito se strojno učenje može podijeliti na nadzirano, nenadzirano i pojačano, ovisno o vrsti informacije koja je dostupna algoritmu za učenje. Nadzirano učenje se može podijeliti na klasifikacijske i regresijske modele. Modeli strojnog učenja se koriste u mnogim primjenama, pogotovo tamo gdje kao ulaz imamo veliku količinu podataka.
U praktičnom dijelu rada će biti ispitana uspješnost klasifikacijskih algoritama nadziranog strojnog učenja (poput logističke regresije, metode potpornih vektora i neuronskih mreža) u ovisnosti o npr. standardizaciji, smanjenju ili povećanju broja uzoraka za balansiranje klasa, smanjenju broja značajki, eliminaciji kolinearnih značajki, različitim načinima kodiranja oznaka klasa, itd. Ispitivanje će se provesti nad nekim skupovima realnih podataka, posebno za trening, validaciju i test, preuzetih sa javno dostupnih stranica poput www.kaggle.com. Uspješnost klasifikacijskih algoritama će se usporediti nad testnim skupom podataka pomoću različitih mjera poput točnosti, F1 mjere, preciznosti i odziva, izračunatih iz matrice zabune.

ZADATAK URUČEN: 26.8.2021. | POTPIS MENTORA: Emil Dumić

Sažetak

U ovom radu analizirana je uspješnost nekoliko klasifikacijskih algoritama strojnog učenja u ovisnosti o predobradi ulaznih podataka. U praktičnom dijelu rada ispitana je uspješnost klasifikacijskih algoritama nadziranog strojnog učenja, a to su logistička regresija, k-najbližih susjeda, algoritam stabla odluke, neuronske mreže i stroj potpornih vektora, u ovisnosti o standardizaciji, smanjenju ili povećanju broja uzoraka za balansiranje klasa, smanjenju broja značajki, eliminaciji kolinearnih značajki, različitim načinima kodiranja oznaka klasa. Ispitivanje se provodi nad nekim skupovima realnih podataka, posebno na skupu podataka za učenje i skupu podataka za ispitivanje, preuzetih s javno dostupnih stranica poput www.kaggle.com. Uspješnost klasifikacijskih algoritama uspoređuje se nad skupom podataka za ispitivanje pomoću različitih mjera poput točnosti, F1 mjere, preciznosti i odziva, izračunatih iz matrice zabune.

Ključne riječi: strojno učenje, Logistička regresija, K najbližih susjeda, Stablo odluke, Stroj potpornih vektora, Neuronske mreže, Metode predobrade, Mjere uspiješnosti

Summary

This paper analyzes the success of several classification algorithms of machine learning depending on the pretreatment of input data. In the practical part of the paper, the performance of classification algorithms of supervised machine learning was examined, these are logistical regression, k-nearest neighbors, decision tree algorithm, neural networks and support vector machine, depending on standardization, reducing or increasing the number of patterns for balancing classes, reducing the number of features, eliminating cholinear features, different ways of encoding class labels. The test shall be carried out on some real datasets, in particular the learning dataset and the test dataset, downloaded from publicly available sites such as www.kaggle.com. The success of classification algorithms is compared over the data set for testing using various measures such as accuracy, F1 measure, precision and response, calculated from the confusion matrix.

Keywords: machine learning, Logistical regression, K nearest neighbors, Decision tree, Support vector machine, Neural networks, Pretreatment methods, Success measures

Popis korištenih kratica

Gini indeks

Sadržaj

1. Uvod

Algoritmi strojnog učenja koriste statistiku, te poznate podatke kako bi pronašli uzorke u bazi podataka. Ti uzorci mogu biti riječi, brojevi čak i slike. Sve što se može spremiti u digitalnom obliku može se koristiti kako bi predvidio ishod ili predvidjelo ponašanje nekog uzorka. Strojno učenje koristi mnogo današnjih servisa i društvenih mreža, kao što su Youtube, Facebook i Google. Svaki od tih servisa sakuplja podatke o korisnicima, kakve videe gledaju, kakve oglase čitaju i pretražuju i na kakve slike reagiraju i komentiraju. Svaki taj sakupljen podatak se koristi u učenju modela algoritmima strojnog učenja kako bi korisniku predložio što bi mu se moglo sljedeće svidjeti.

Utjecaj na uspješnost modela ima obrada podataka. Podatci koji se koriste za učenje algoritama strojnog učenja treba obraditi prije samog učenja zbog nevažećih i nepostojećih vrijednosti i nekodiranih vrijednosti značajki. Nebalansiran (engl. *imbalanced*) skup podataka, odnosno onaj u kojem je velika razlika u brojevima predstavnika pojedinih klasa, međuovisne značajke i neskalirani skup podataka mogu se otkloniti metodama obrade podataka.

U ovom radu uspoređuje se uspješnost naučenih modela različitih algoritama strojnog učenja naučenih podatcima podloženih različitim metodama predobrade, a uspješnost tih mjera određuje se pomoću mjera točnosti (engl. *accuracy*), odziva (engl. *recall*), preciznosti (engl. *precision*) i F1-ocjene (engl. *F1-score*) koje predstavljaju mjere uspješnosti u strojnom učenju.

2. Nadzirani klasifikacijski algoritmi

Nadzirani klasifikacijski algoritmi koriste se u većini praktičnog strojnog učenja. Nadzirano strojno učenje je učenje gdje imamo ulaznu vrijednost (X) i izlaznu vrijednost (Y) na kojima koristimo algoritme kako bi mapirali funkciju od ulaza do izlaza. Model se uči na označenim (engl. *labeled*) podatcima, odnosno podatcima kojima je pridružena oznaka klase, a zatim se taj model koristi za predviđanje klase kojoj pripada neki neoznačeni primjer. [1]

2.1. k– najbližih susjeda

Algoritam k-najbližih susjeda (engl. *k-nearest neighbors*) je algoritam strojnog učenja koji sprema cijeli skup podataka i na temelju njih predviđa klasu neviđenih primjera. Učinkovite implementacije ovog algoritma koriste složene strukture podataka za pohranu skupa primjera.

Predviđanje ovim algoritmom svodi se na pretraživanje zapamćenog skupa podataka kojim se traži K zapamćenih primjera koji su najsličniji, odnosno najmanje udaljeni od danog neviđenog primjera. Klasa koja se najčešće pojavljuje u tih k primjera odabire se kao klasa neviđenog primjera.

Skup primjera često je prikazan kao točke u n-dimenzionalnom prostoru R^n, pa se kao mjera udaljenosti između dva vektora u tom prostor može koristiti Euklidska udaljenost (1):

$$d(x,y) = \sqrt{\sum_{r=1}^{n}\left(a_{rx} - a_{ry}\right)^2} \qquad (1)$$

Za koju je:

n – broj različitih značajki u skupu podataka,

a_{rx}, a_{ry} – vrijednost varijable r vektora x, odnosno vektora y.

Manhattan udaljenost se također može koristiti kao mjera udaljenosti, kako je prikazano (2):

$$d = (x,y) = \sum_{r}^{N} a_{rx} - a_{ry} \vee \qquad (2)$$

Mogu se koristiti i druge mjere udaljenosti, kao što je Minkowski udaljenost.

Algoritam k-najbližih susjeda može se koristiti za probleme regresije, gdje se predviđanje temelji na srednjoj vrijednosti ili medijanu za K - najsličnijih slučajeva.

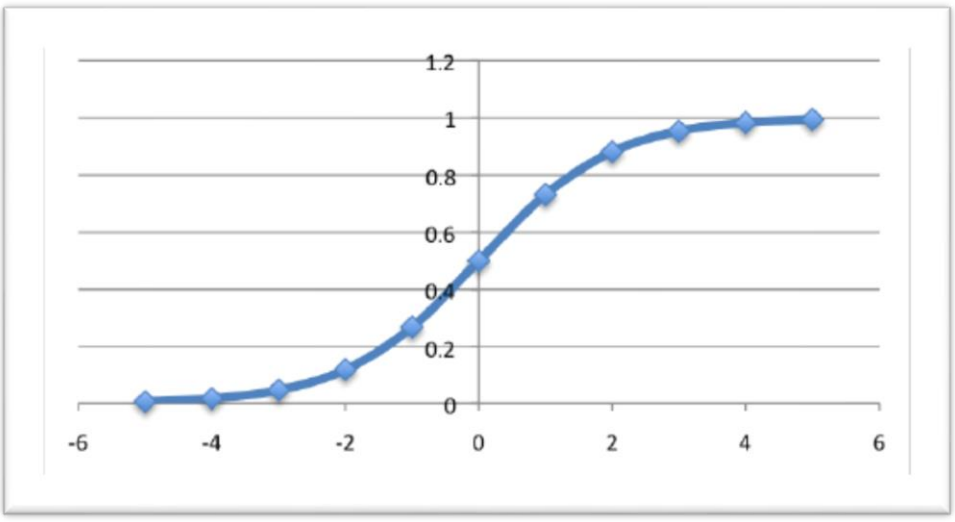
Algoritam dobro radi s malim brojem ulaznih varijabli (p), no problem se stvara kada je broj ulaza dosta velik. Svaka ulazna varijabla može se smatrati dimenzijom p-dimenzionalnog

ulaznog prostora. Dodatkom značajki povećava se dimenzija hiperprostora, a samim time se povećava i udaljenost između primjera u prostoru, pa će time svi primjeri biti na velikoj udaljenosti jedan od drugog. Taj problem algoritma k-najbližih susjeda naziva se prokletstvo dimenzionalnosti (engl. *curse of dimensionality*). [2]

2.2. Logistička regresija

Algoritam logističke regresije (engl. *logistic regression*) koristi se za binarne klasifikacijske probleme, koji su problemi s dvije vrijednosti klase, npr. da ili ne, 0 ili 1. Također se koristi za određivanje vjerojatnosti pripadnosti uzoraka nekom razredu.

Logistička funkcija procjenjuje vjerojatnost dobivenog rezultata, a zatim definira najbliži razred (+ ili -) toj dobivenoj vrijednosti vjerojatnosti. Ako primijenimo algoritam logističke regresije i dobivamo rezultat veći od 0.5, to bi značilo da ulaz pripada pozitivnoj klasi pa se ulazu dodjeljuje oznaka klase 1, a da smo dobili vrijednost manju od 0.5 to bi značilo da je ulaz pripada negativnoj klasi, pa ulazu dodjeljuje vrijednost 0. Logistička funkcija ima krivulju oblika slova S koja može sadržavati svaki realni broj i označiti ga u vrijednosti između 0 i 1, ali nikad ne poprima krajnje vrijednosti. [3] Na slici 2.1 grafički je prikazana logistička funkcija.



Slika 2.1 Grafički prikaz logističke funkcije [1]

Logistička funkcija opisana je i prikazana formulom (3):

$$P(t) = \frac{1}{1+e^{-t}} \quad (3)$$

2.3. Algoritam stabla odluke

Algoritam stabla odluke (engl. *decision tree*) koristi se za klasifikaciju ili kod prediktivnih problema modeliranja kod kojih se predviđa vrijednost binarne ciljne značajke. Izrada modela kreće od korijena stabla. Gledaju se moguće podjele na temelju jedne značajke. Kako bi se odabrala najbolja podjela mjeri se smanjenje varijabilnosti distribucije krajnje značajke u granama ispod u usporedbi s granama iznad tj. s granom na kojoj se radi podjela. Početna podjela u korijenu predstavlja podjelu ulaza prostora na dva potprostora s paralelnom granicom jednoj ulaznoj značajki. I tako se ponavlja dalje dok sve grane ne postanu „čiste“ ili tako dugo do kad se ne zadovolji neki zaustavni kriterij. [4]
Evaluacijska funkcija korištena za prijelom je Gini indeks (IG), definiran prema formuli (4) (Apte, 1997):

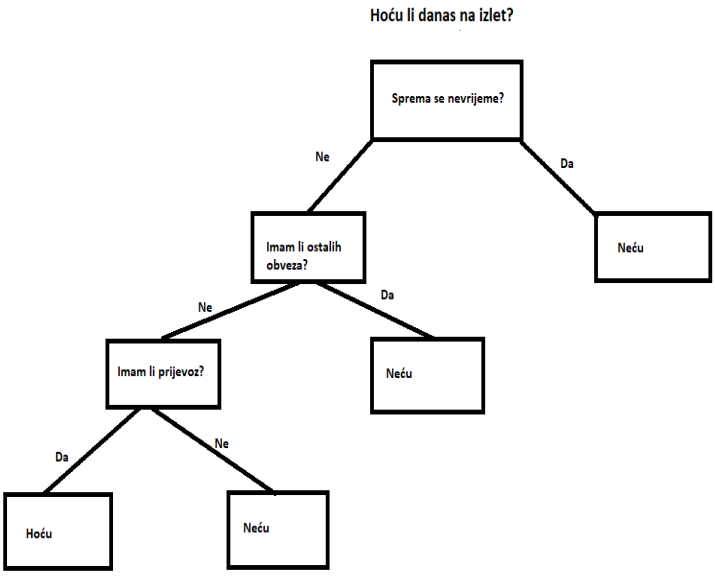
$$I_G(t) = 1 - \sum_{i=1}^{m} p_i^2 \quad (4)$$

t - trenutni čvor

p_i– vjerojatnost klase i u čvoru t

m – broj klasa u modelu

Najbolje grananje određuje se za svaki atribut u svakom čvoru, a pobjednik se izabire s pomoću Gini indeksa. [5]



Slika 2.2 Jednostavni primjer stabla odlučivanja (autorski rad) [2]

Slika 2.2 prikazuje primjer stabla odlučivanja kojim želimo saznati koja je vjerojatnost da danas idemo na izlet. Prvo se pitamo sprema li se nevrijeme? Time dobivamo podjelu stabla na dva čvora. Polovica početnog skupa grana se u čvor s odgovorom da, a druga polovica se grana u ne.

Onaj čvor koji završava nakon odgovora nazivamo terminalni čvor. Onaj dio algoritma koji je preostao se nastavlja tako dugo do kad ne dođemo do nekog mogućeg rješenja. U našem slučaju je to kad nemamo ostalih obveza, te kad imamo prijevoz. [6]

2.4. Stroj potpornih vektora

Stroj potpornih vektora jedan je od najpopularnijih algoritama. Bili su najpopularniji 90. godina prošloga stoljeća, kada su i razvijeni. Oni se i dan danas koriste za algoritme visokih performansi s malim ugađanjem. Maksimizacija margine (engl. *margin*) je krajnji cilj optimizacije u ovom algoritmu. Margina se računa kao okomica udaljenost od crte pa do samo najbližih točaka. Te točke su relevantne u definiranju linija i u izgradnji klasifikatora. Te se točke nazivaju vektori potpore (engl. *support vectors*). U praksi su stvarni podaci neuredni i ne mogu se savršeno odvojiti hiperravninom (engl. *hyperplane*). Ograničenje maksimiziranja margine linije koja razdvaja klase mora biti ublaženo, a naziv za to je klasifikator mekih margina (engl. *soft margin*). Ta promjena u nekim točkama podataka o obuci dopušta prekršiti liniju razdvajanja. Uvodi se dodatni skup koeficijenata koji daju marginu u svakoj dimenziji.

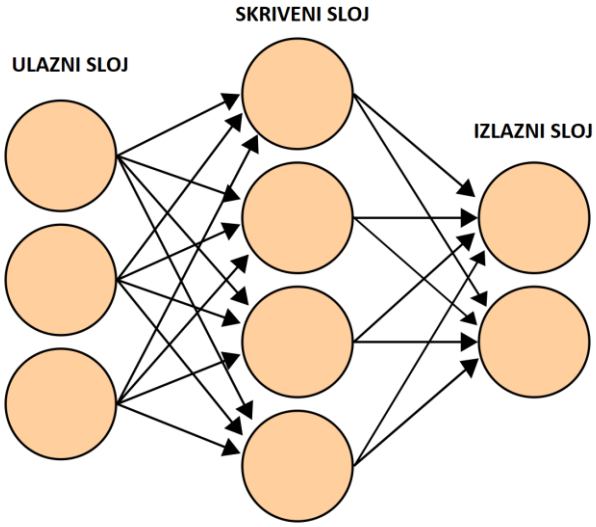
2.5. Neuronske mreže

Neuronske mreže rađene su po primjeru bioloških neurona, odnosno ljudskog mozga. U našem mozgu informacije se izmjenjuju iz jednog u drugi neuron. Po tom primjeru neuronske mreže funkcioniraju tako da bi što brže odradile neki račun. Neuronske mreže dijele se na dvije vrste, a to su: feedforward i feedback. Feedforward govori o tome da informacija prolazi od ulaza do izlaza te ne sadrži povratnu petlju, dok u feedback-u informacija može putovati u oba smjera te sadrži povratnu petlju. Feedforward mreže dalje se dijele na višeslojnu i jednoslojnu mrežu. U jednoslojnoj se mreži ulazni sloj povezuje s izlaznim slojem, dok se višeslojna mreža sastoji od više slojeva koji se nazivaju skriveni slojevi između ulaznog i izlaznog sloja.

Neuronsku mrežu čine čvorovi koji su nalik čvorovima u mozgu. Svaka veza u mreži ima svoju težinu. Kada su ulazi u čvorove $x1, x2, x3$ ... i odgovarajući utezi su $w1, w2, w3, \ldots$ neto ulaz (y) sličan je sljedećem primjeru (5).

$$y = x1 * w1 + x2 * w2 + x3 * w3 + \cdots \quad (5)$$

Kad se primjene aktivacijske funkcije na neto ulaz, ona daje izlaz Y = F (y). Nakon toga se procjenjuje izlaz. Težine (engl. *weights*) se podešavaju ako se procijenjeni izlaz razlikuje od izlaza kojeg smo htjeli. Tako se ponavlja tako dugo do kad se ne postignu željeni rezultati. [8] Slika 2.3 prikazuje feedforward mrežu.



Slika 2.3 Pojednostavljeni prikaz feedforward umjetne neuronske mreže [13]

3. Mjere uspješnosti

Mjere uspješnosti se koriste kako bi se odredilo je li naučen model ujedno i dobar model. Postoje razne mjere uspješnosti koje se koriste ovisno o ograničenjima koja konačan model treba zadovoljiti.

3.1. Matrica zabune

Matrica zabune (engl. *confusion matrix*) grafički je prikaz predviđenih vrijednosti klasa i pravih vrijednosti klasa koji pomaže pri računaju mjera uspješnosti modela strojnog učenja. U njemu se označuju ispravno i neispravno predviđene vrijednosti kako bi se dao dojam o ispravnosti rada naučenog modela. Matrica zabune grafički je prikazana na slici 3.1.

		PRAVA VRIJEDNOST	
		Pozitivna (P)	Negativna (N)
PREDVIĐENA VRIJEDNOST	Pozitivna (P)	Istinito pozitivna (engl. *true positive, TP*)	Lažno pozitivna (engl. *false positive, FP)*
	Negativna (N)	Lažno negativna (engl. *false negative, FN*)	Istinito negativna (engl. *true negative, TN*)

Slika 3.4 Matrica zabune

3.2. Točnost

Točnost (engl. *accuracy)* mjera je uspješnosti koja je omjer ispravno klasificiranih ulaza u odnosu na sve ulaze, a prikazana je formulom (6).

$$točnost = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

3.3. Odziv

Odziv (engl. *recall*) mjera je uspješnosti koja je omjer ulaza ispravno klasificiranih kao pozitivni i ukupnog broja pozitivnih ulaza, prikazan formulom (7). Njeno povećanje ukazuje da je došlo do smanjenja broja primjera koji su krivo svrstani u negativnu klasu, odnosno klasu označenu s 0.

$$odziv = \frac{TP}{TP + FN} \tag{7}$$

3.4. Preciznost

Preciznost (engl. *precision*) mjera je uspješnosti koja je omjer ulaza ispravno klasificiranih kao pozitivni i ukupnog broja ulaza koji su klasificirani kao pozitivni.Preciznost je prikazana pormulom (8) Njeno povećanje ukazuje da je došlo do smanjenja broja primjera koji su krivo svrstani u pozitivnu klasu, odnosno klasu označenu s 1.

$$preciznost = \frac{TP}{TP + FP} \tag{8}$$

3.5. F1-ocjena

F1-ocjena (engl. F1-score) mjera je uspješnosti koja je kombinacija preciznosti i odziva, a prikazana je formulom (9).

$$f1 - ocjena = \frac{2 * preciznost * odziv}{preciznost + odziv} \tag{9}$$

4. Metode predobrade

Za predobradu podataka koriste se razne metode koje ovise o ulaznom skupu podataka i algoritmu koji se uči. Neki algoritmi normalno funkcioniraju bez predobrade, a kod drugih je predobrada nužna za ispravan rad i interpretaciju rezultata.

Algoritmi kojima je potrebna predobrada su oni algoritmi koji imaju previše nedostajućih vrijednosti ili varijabli, koji nemaju smisla, ili imaju nekonzistentne podatke.

4.1. Smanjenje broja značajki

Smanjenje broja značajki (engl. *feature selection*) odnosi se na odabir podskupa značajki iz izvornog skupa značajki kako bi se smanjila složenost modela i time povećala računska učinkovitost modela i smanjila pogreška generalizacije (engl. *generalization*) koja može nastati zbog suvišnih značajki. Postoje dva uobičajena pristupa smanjenju broja značajki, gdje omotač (eng. *wrapper*) koristi predviđeni algoritam učenja za procjenu korisnosti značajki, dok filtar (eng. *filter*) procjenjuje značajke prema heuristikama na temelju općih karakteristika podataka. Smatra se da pristup omotača daje bolje podskupove značajki, no sporije je od filtra. [9]

4.2. Uklanjanje koreliranih značajki

Modeli se često koriste za ispitivanje hipoteza, gdje ispituju statističku značajnost učinka značajke na izlaz modela. Visoka kolinearnost među značajkama znači da varijable u kolinearnom skupu dijele značajne količine informacija. Male promjene u skupu podataka mogu snažno utjecati na rezultate pa je model nestabilan (velika varijacija), a teško je procijeniti relativnu važnost varijabli. Zato se važne i utjecajne značajke mogu algoritmu strojnog učenja činiti neznačajnima zbog povećane pogreške uzrokovane varijacijom u nestabilnom modelu. [10]

Korelirane značajke općenito ne poboljšavaju modele, ali utječu na određene modele na različite načine i u različitoj mjeri. Za linearne modele, multikolinearnost (engl. *multicolinearity*) može dati rješenja koja se jako razlikuju i mogu biti brojčano nestabilna.

Slučajne šume (eng. *random forests*) mogu biti dobre u otkrivanju interakcija između različitih značajki, ali visoko korelirane značajke mogu prikriti te interakcije.

4.3. Poduzorkovanje i preuzorkovanje

Metode preuzorkovanja (engl. *oversampling*) dupliciraju ili stvaraju nove sintetičke primjere u manjinskoj klasi, dok metode poduzorkovanja (engl. *undersampling*) brišu ili spajaju primjere

u većinskoj klasi. Najjednostavnija metoda poduzorkovanja je slučajno poduzorkovanje (engl. *random undersampling*). Ta metoda ne koristiti nikakav algoritam da bi riješila problem, te joj je cilj balansirati klasnu distribuciju slučajnim odabirom primjera negativne klase koje će ukloniti iz skupa za učenje. Tako se i u preuzorkovanju koristi slučajno preuzorkovanje (engl. *random oversampling*). Ova metoda je slična kao i slučajno poduzorkovanje samo što slučajno odabire primjere iz pozitivne klase, odnosno kopije pozitivnih stavi primjera u skup za učenje. Ovom metodom ne gubimo nikakve podatke budući da ih ne izbacujemo.

4.4. Kodiranje klasa (eng. class encoding)

Strojno učenje zahtijeva da sve ulazne varijable budu numeričke. To znači da ako podaci sadrže kategoričke podatke koji su predstavljeni nominalnom vrijednošću, odnosno vrijednošću koja nije kvantitativne, oni bi se trebali kodirati u brojeve da bismo mogli učiti i procijeniti model. Ordinalne značajke su one koje označavaju poredak, na primjer „malen, srednji, velik“ koji si kodiraju rastućim brojevima, na primjer s 0, 1 i 2. Nominalne značajke su one koje ne predstavljaju kvantitativnu vrijednost, na primjer značajka boje koja bi poprimila vrijednosti „crvena, žuta i plava“. Te vrijednosti su kodirane pomoću One-hot encoding metode. Ako bi se boje kodirale samo s „0, 1 i 2“, određeni algoritmi mogu biti pristrani prema većoj brojevnoj vrijednosti, u ovom slučaju plavoj boji.

5. Programski kôd

Učenje i ispitivanje modela, predobrada podataka i izračun mjera uspješnosti obavljeno je pomoću programskog jezika Python. Sve metode potrebne za izračun mjera uspješnosti, implementacije modela i metode predobrade podataka uzete su iz biblioteke `scikit-learn` programskog jezika Python.
Učitavanje skupova podataka is csv formata obavljeno je na sljedeći način:

```
clients = pd.read_csv('clients.csv')
```

gdje je pd biblioteka `pandas`, a `read_csv` metoda pomoću koje se čita skup podataka u csv formatu.

5.1. Predobrada podataka

Za predobradu korištene su implementacije unutar sljedećih biblioteka:

- `sklearn.preprocessing.StandardScaler`
- `sklearn.ensemble.ExtraTreesClassifier`
- `sklearn.feature_selection.SelectFromModel`
- `imblearn.over_sampling.RandomOverSampler`
- `imblearn.under_sampling.RandomUnderSampler`
-

Uklanjanje duplih vrijednosti obavljeno je pomoću metode:

```
clients.drop_duplicates()
```

Provjera ima li nepostojećih vrijednosti obavljena je pomoću koda:

```
clients.isnull().values.any()
```

čije izvršavanje vraća vrijednost False ako nema nepostojećih vrijednosti, a True ako ima.

Kodiranje klasa za ordinalne značajke obavljeno je pomoću mapiranja u pythonu, gdje su vrijednosti ordinalnim značajkama pridružene u odgovarajućem rastućem redoslijedu. U originalnom skupu podataka mapirane su nove vrijednosti odgovarajuće značajke.

```
map = {"Incomplete secondary education" : 0, "Secondary
education" : 1, "Secondary special education" : 2, "Incomplete
higher education" : 3, "Higher education" : 4, "PhD degree" : 5}
```

```
X["education"] = X["education"].replace(map)
```

Kôd 5.1 Kodiranje klasa za ordinalne značajke

Kodiranje klasa za nominalne značajke obavljeno je pomoću metode `get_dummies` unutar python biblioteke `pandas`.

```
X = pd.get_dummies(X, columns = ['month', 'sex', 'product_type',
'region','family_status','phone_operator','is_client','having_ch
ildren_flg'], drop_first =True)
```

Kôd 5.2 Kodiranje klasa za nominalne značajke

Standardiziranje podataka obavljeno je pomoću metode `fit_transform` unutar razreda `StandardScaler`.

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Kôd 5.3 Standardizacija podataka

Selekcija značajki napravljena je pomoću razreda `SelectFromModel` koji služi za odabir informativnih značajki pomoću razreda `ExtraTreesClassifier` koji predstavlja kombinaciju više stabla odluke. Razred odabire optimalnu kombinaciju značajki tako da uči modele postupno izbacujući neinformativne značajke.

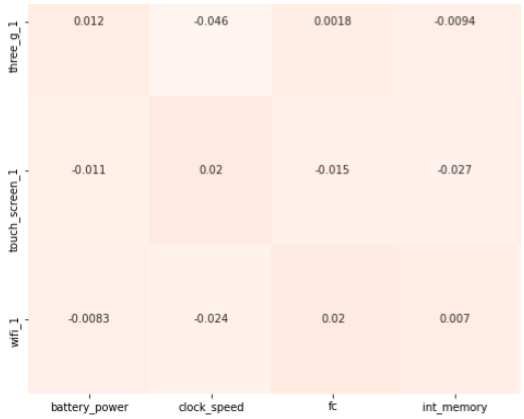
```
clf = ExtraTreesClassifier(n_estimators=50, random_state=0)
clf = clf.fit(X, y)
mod = SelectFromModel(clf, prefit=True)
X_selected = mod.transform(X)
```

Kôd 5.4 Selekcija značajki

Za određivanje korelacije značajki upotrebljena je metoda `corr()` iz biblioteke `pandas` koja određuje korelacije između svakog para značajki. Iz skupa podataka izbačene su značajke koje su imale korelaciju veću od 0.7.

```
plt.figure(figsize=(120,120))
cor = pd.DataFrame(X).corr()
sns.heatmap(cor, annot=True, cmap=plt.cm.Reds)
plt.show()
```

Kôd 5.5 Određivanje korelacije značajki



Slika 5.5 Isječak tablice korelacija

Na slici 5.1. prikazan je isječak tablice korelacija.

Poduzorkovanje je napravljeno pomoću razreda `RandomUnderSampler`. Metoda `fit_` unutar razreda `RandomUnderSampler` obavlja poduzorkovanje većinskog razreda tako da nasumično odabire broj primjeraka većinske klase koji je jednak broju primjeraka manjinske klase.

```
rus = RandomUnderSampler(random_state=0)
X_undersample, y_undersample = rus.fit_resample(X, y)
```

Kôd 5.6 Poduzorkovanje podataka

Preuzorkovanje je napravljeno pomoću razreda `RandomOverSampler`. Metoda `fit_` unutar razreda `RandomOverSampler` obavlja preuzorkovanje manjinskog razreda tako da nasumično odabire broj primjeraka manjinske klase koji je jednak broju primjeraka većinske klase.

```
ros = RandomOverSampler(random_state=0)
X_oversample, y_oversample = ros.fit_resample(X, y)
```

Kôd 5.7 Preuzorkovanje podataka

5.2. Mjere uspješnosti

Za izračun mjera uspješnosti korištene su implementacije unutar sljedećih biblioteka:

- `sklearn.metrics.accuracy_score`
- `sklearn.metrics.precision_score`
- `sklearn.metrics.recall_score`

- `sklearn.metrics.F1_score`

Računanje mjera uspješnosti napravljeno je na sljedeći način:

```
    accuracy_score(y_test, y_pred)
    recall_score(y_test,y_pred,average='macro',zero_division=0)
    precision_score(y_test,         y_pred,         average='macro',
zero_division=0)
    F1_score(y_test, y_pred, average='macro')
```

Kôd 5.8 Izračun mjera uspješnosti

Gdje su u `y_test` spremljene ispravne vrijednosti klasa skupa za ispitivanje, a `y_pred` vrijednosti klasa koje su predviđene za taj skup podataka.

5.3. Implementacije modela

Za modele korištene su implementacije unutar sljedećih biblioteka:

- `sklearn.linear.LogisticRegression`
- `sklearn.tree.DecisionTreeClassifier`
- `sklearn.neighbors.KNeighborsClassifier`
- `sklearn.neural_network.MLPClassifier`
- `sklearn.svm.SVC`

Svaka od navedenih implementacija ima sljedeće metode:

```
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Kôd 5.9 Učenje i predviđanje modela

Metoda `fit` služi za učenje modela pomoću skupa podatka za učenje, a metoda `predict` služi za predviđanje vrijednosti klasa pomoću naučenog modela i skupa podataka za ispitivanje.

Podatci su razdvojeni na skup za učenje i skup za ispitivanje. 30% početnih podataka pripada skupu za ispitivanje, a ostatak podataka pripada skupu podataka za učenje. Razdvajanje je napravljeno pomoću metode `train_test_split` iz biblioteke `sklearn.model_selection`. Opcija `stratify=y` označuje da je razdvajanje skupa podataka na dva skupa obavljeno tako da je omjer razreda očuvan. To znači da ako imamo skup podataka od 100 ulaza, od kojih 50 pripada klasi 1, a 50 klasi 0, onda ćemo dobiti skup za učenje s 70 ulaza koji ima 35 pripadnika klasi 1 i 35 pripadnika klasi 0, i skup za ispitivanje koji ima 30 ulaza i 15 pripadnika klasi 1, a 15 pripadnika klasi 0.

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=0, stratify=y)
```

Kôd 5.10 Razdvajanje podataka na skup za učenje i skup za ispitivanje

Kod koji se koristi izračun mjera uspješnosti i prikaz konačnih rezultata ukomponiran je u dvije metode. Metoda `scoring` prikazana u kodu 5.11 prima model koji želimo učiti, skupove podataka i pripadne oznaka klase, ime metode predobrade koja se koristi i tablicu u koju se stavljaju rezultati modela naučenog na skupu podataka obrađenom danom metodom predobrade.

```
def scoring(model, X, y, preprocessing_method, compare_table) :
    X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.3, random_state=0, stratify=y)


     compare_table[preprocessing_method]                          =
[accuracy_score(y_test, y_pred),
           recall_score(y_test,y_pred,average='macro',zero_divis
      ion=0),
           precision_score(y_test,    y_pred,    average='macro',
      zero_division=0),
           F1_score(y_test, y_pred, average='macro')]
```

Kôd 5.11 Metoda za izračun mjera uspješnosti

Metoda `preprocessing` prima model na kojem želimo isprobati uspješnost metoda predobrade, a onda ona uzastopno poziva metodu `scoring` kako bi napravila tablicu u kojoj je uspoređena uspješnost.

```
def preprocessing(model) :
    compare_table  =  pd.DataFrame(index=['Accuracy',  'Recall',
'Precision', 'F1 Score'])

    ## bez predobrade
    scoring(model, X, y, "No preprocessing", compare_table)

    ## standardizacija
    scoring(model,      X_scaled,      y,      "Standardization",
compare_table)

    ## ekstrakcija značajki
    scoring(model,    X_selected,    y,    "Feature    selection",
compare_table)

    ## undersampling
    scoring(model,          X_undersample,          y_undersample,
"Undersampling",   compare_table)

    ## oversampling
    scoring(model,  X_oversample,  y_oversample,  "Oversampling",
compare_table)

    ## rezultat usporedbe
```

```
compare_table['Best Score'] = compare_table.idxmax(axis=1)
print(compare_table.to_markdown() + '\n')
```

Kôd 5.12 Metoda za konstruiranje tablice za usporedbu mjera uspješnosti

Poziv metode `preprocessing` koja prikazuje rezultate prikazan je u kodu 5.13.

```
# logistička regresija
preprocessing(LogisticRegression(max_iter=100))

# k-najbližih susjeda
preprocessing(KNeighborsClassifier())

# algoritam stabla odluke
preprocessing(DecisionTreeClassifier(random_state=0))

# neuronske mreže
preprocessing(MLPClassifier(hidden_layer_sizes=(100,100,100
), random_state=0))

# stroj potpornih vektora
preprocessing(SVC())
```

Kôd 5.13 Poziv metode preprocessing za pojedine modele

Konačan prikaz rezultata za dani skup podataka prikazan je na slici 5.2..

```
----------------------
Mobile dataset results.
----------------------

Performance metrics comparison after preprocessing for logistic regression classifier:
|           |   No preprocessing |   Standardization |   Feature selection | Best Score      |
|:----------|-------------------:|------------------:|--------------------:|:----------------|
| Accuracy  |               0.62 |              0.96 |            0.846667 | Standardization |
| Recall    |               0.62 |              0.96 |            0.846667 | Standardization |
| Precision |               0.616504 |              0.960308 |            0.845708 | Standardization |
| F1 Score  |               0.617491 |              0.960097 |            0.845712 | Standardization |


Performance metrics comparison after preprocessing for k nearest neigbors classifier:
|           |   No preprocessing |   Standardization |   Feature selection | Best Score      |
|:----------|-------------------:|------------------:|--------------------:|:----------------|
| Accuracy  |               0.905 |              0.485 |            0.79     | No preprocessing |
| Recall    |               0.905 |              0.485 |            0.79     | No preprocessing |
| Precision |               0.905662 |              0.513297 |            0.786842 | No preprocessing |
| F1 Score  |               0.905005 |              0.485082 |            0.78766  | No preprocessing |


Performance metrics comparison after preprocessing for decision tree classifier:
|           |   No preprocessing |   Standardization |   Feature selection | Best Score      |
|:----------|-------------------:|------------------:|--------------------:|:----------------|
| Accuracy  |               0.813333 |              0.815 |            0.746667 | Standardization |
| Recall    |               0.813333 |              0.815 |            0.746667 | Standardization |
| Precision |               0.816952 |              0.818573 |            0.74541  | Standardization |
| F1 Score  |               0.814576 |              0.816174 |            0.745989 | Standardization |


Performance metrics comparison after preprocessing for neural network classifier:
|           |   No preprocessing |   Standardization |   Feature selection | Best Score      |
|:----------|-------------------:|------------------:|--------------------:|:----------------|
| Accuracy  |               0.453333 |              0.886667 |              0.25   | Standardization |
| Recall    |               0.453333 |              0.886667 |              0.25   | Standardization |
| Precision |               0.419995 |              0.897041 |              0.0625 | Standardization |
| F1 Score  |               0.365626 |              0.887658 |              0.1    | Standardization |


Performance metrics comparison after preprocessing for support vector machine classifier:
|           |   No preprocessing |   Standardization |   Feature selection | Best Score      |
|:----------|-------------------:|------------------:|--------------------:|:----------------|
| Accuracy  |               0.951667 |              0.861667 |            0.841667 | No preprocessing |
| Recall    |               0.951667 |              0.861667 |            0.841667 | No preprocessing |
| Precision |               0.952111 |              0.865066 |            0.84044  | No preprocessing |
| F1 Score  |               0.951691 |              0.862542 |            0.840183 | No preprocessing |
```

Slika 5.6 Prikaz rezultata

Svaka od navedenih implementacija ima parametre koji su automatski postavljeni na pretpostavljene vrijednosti. Podešavanjem parametara u implementaciji model se bolje prilagođava podatcima.

Implementacija logističke regresije ima velik broj parametara koji se koriste kod penalizacije, optimizacije, skaliranja broja razreda i drugih problema koji se nastoje riješiti logističkom regresijom.

```
LogisticRegression(penalty='l2',*,dual=False,tol=0.0001, C=1.0,
fit_intercept=True,intercept_scaling=1,class_weight=None,random_state
```

```
=None,solver='lbfgs', max_iter=100, multi_class='auto',
verbose=0,warm_start=False, n_jobs=None, l1_ratio=None)
```

Kod 5.14 Parametri implementacije logističke regresije

`Penalty` je parametar koji se koristi za određivanje norme kod kažnjavanja unutar algoritma logističke regresije. Moguće vrijednosti za ovaj parametar su `l1` koji označuje l1 regularizaciju, `l2` koji koristi l2 regularizaciju, `elasticnet` koji koristi elastičnu regularizaciju mreže i `none` parametar koji označuje algoritam bez norme za kažnjavanja. Kao pretpostavljena vrijednost ovog parametra koristi se l2 regularizacija.

`Dual` je parametar koji se koristi kad je vrijednost parametra `solver` jednaka `liblinear` i kad se koristi l2 regularizacija. Pretpostavljena vrijednost za ovaj parametar je `False`.

`Tol` je parametar koji predstavlja toleranciju na kriterij za zaustavljanje. On određuje koliko konačna vrijednost kriterija za zaustavljanje smije odstupati od granice. Pretpostavljena vrijednost ovog parametra je 1e-4.

`C` je parametar koji označuje inverz snage regularizacije. Manja vrijednost ovog parametra označuje jaču regularizaciju. Pretpostavljena vrijednost ovog parametra je jedan.

`Fit_intercept` je parametar koji određuje konstantu (engl. *bias*) koja se dodaje decizijskoj (engl. *decision*) funkciji. Pretpostavljena vrijednost ovog parametra je True što znači da se dodaje konstanta decizijskoj funkciji.

`Intercept_scaling` označuje parametar koji se koristi kad je `solver` parametar postavljen na `liblinear` i `fit_intercept` parametar postavljen na `True`. U tom slučaju `c` postaje „sintetička“ značajka s konstantnom vrijednosti koja je jednaka postavljenoj `intercept_scaling` vrijednosti.

`Class_weight` je parametar koji klasama pridružuje težine. Ako parametar nije postavljen na vrijednosti `dict` ili `balanced`, svim klasama pridružuje se težina jedan. Kad je ovaj parametar postavljen na `balanced`, onda se težine automatski inverzno proporcionalno prilagođavaju ovisno o frekvencijama pojedinih razreda. Pretpostavljena vrijednost za ovaj parametar je `None`.

`Random_state` parametar se koristi kad je vrijednost parametra `solver` postavljena na `sag`, `saga` ili `liblinear` kako bi se podatci izmiješali. Pretpostavljena vrijednost ovog parametra je `None`, odnosno podatci su ostavljeni u istom redoslijedu u kojem su bili.

Parametar `solver` označuje algoritam koji se koristi kod optimizacije kod logističke regresije. Kao opcije su dani `newton-cg`, `liblinear`, `lbfgs` i `saga` algoritmi kod optimizacije. `newton-cg` je algoritam koji koristi linearnu metodu konjugiranih gradijenata (engl. *linear conjugate gradient method*) za optimizaciju. `liblinear` je algoritam koji koristi

koordinatni spust (engl. *coordinate descent*) za optimizaciju. `lbfgs` je algoritam čiji je naziv skraćenica od „Limited Memory Broyden – Fletcher – Goldfarb – Shanno algorithm“ koji predstavlja aproksimaciju Broyden – Fletcher –Goldfarg Shanno algoritma za rješavanje nelinearnih optimizacijskih problema koristeći ograničenu memoriju računala. `saga` algoritam je algoritam inkrementalnog gradijentnog spusta s brzom linearnom konvergencijom. `sag` je algoritam koji koristi stohastički gradijentni spust. Pretpostavljena vrijednost za ovaj parametar je `lbfgs`.

`Max_iter` je parametar koji predstavlja granicu za broj iteracija za konvergenciju algoritama određenih parametrom `solver`. Pretpostavljena vrijednost ovog parametra je 100 iteracija.

`Multi_class` je parametar koji određuje način na koji će se algoritam logističke regresije koristiti u slučaju više od dvije klase. Ako je odabrana opcija `ovr`, to označuje „one-versus-all“ odnosno za svaku klasu se uči jedan model. Kad je vrijednost parametra `auto`, onda se odabire `ovr`, ako je binaran problem ili `solver=liblinear`, a `multinomial` se odabire u svim ostalim slučajevima.

`Verbose` je parametar koji se koristi kad je vrijednost parametra `solver` postavljena na `liblinear` ili `lbfgs`. On označuje detaljnost postupka kojim se dolazi do konačnog modela.

`N_jobs` predstavlja broj procesorskih jezgri koji se koristi kod paralelizacije kad je `multi_class=ovr`, odnosno kad se za jedan problem uči više modela.

`L1_ratio` je parametar koji se koristi kad je `penalty=elastic_net`. On je parametar koji se koristi za miješanje kod elastične regularizacijske mreže. Pretpostavljena vrijednost ovog parametra je `None`.

Implementacija k-najbližih susjeda sastoji se od parametara koji se koriste za izračun udaljenosti, određivanje broja susjeda i paralelizaciju izračunavanja.

```
Class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, *,
weights='uniform', algorithm='auto', leaf_size=30, p=2,
metric='minkowski', metric_params=None, n_jobs=None, **kwargs)
```

Kôd 5.15 Parametri implementacije k-najbližih susjeda

`n_neighbors` je parametar koji predstavlja broj susjeda koji se koriste za klasifikaciju primjera. Pretpostavljena vrijednost ovog parametra je pet.

`Weights` označuje funkciju koja se koristi kod predviđanja. `uniform` predstavlja da je svim točkama pridružena ista težina. `distance` označuje da se točkama pridružuje težina koja je

inverzna udaljenosti od točke. `callable` označuje da korisnik može sam odrediti težinsku funkciju koja će se koristiti. Pretpostavljena vrijednost ovog parametra je `uniform`.

`Algorithm` je parametar koji označuje koji algoritam se koristi za izračun najbližih susjeda. `ball_tree` označuje da se za izračun koristi `BallTree` algoritam, `kd_tree` označuje da se za izračun koristi `KDTree` algoritam, `brute` označuje da će se `brute-force` odrediti najbliži susjedi, odnosno izračunati će se udaljenost svake točke od primjera i odrediti najbliže točke. Ako je vrijednost postavljena na `auto`, onda će algoritam sam odrediti najprikladniji algoritam za određivanje susjeda. Pretpostavljena vrijednost ovog parametra je `auto`.

`leaf_size` je parametar koji se prosljeđuje kad je `algorithm` parametar jednak `ball_tree` ili `kd_tree`. Ovaj parametar može utjecati na brzinu rada i količinu memorije. Pretpostavljena vrijednost ovog parametra je 30.

`P` označuje potenciju za Minkowski metriku za izračun udaljenosti. Kad je `p=1`, onda je to Manhattan udaljenost, a kad je `p =2` onda je to Euklidska udaljenost. Pretpostavljena vrijednost za ovaj parametar je dva.

`Metric` je mjera udaljenosti koja se koristi za izgradnju stabla. Pretpostavljena vrijednost za ovaj parametar je `minkowski`.

`Metric_params` čuva dodatne parametre koji su potrebni kod izračuna udaljenosti pomoću zadane mjere udaljenosti.

`N_jobs` je broj poslova koji se obavljaju kod izračuna udaljenosti primjera i svih točaka u skupu za učenje. Ako je parametar postavljen na `None`, onda znači da samo jedan procesor radi kod izračuna. Kad je vrijednost `-1` to znači da se koriste svi procesori. Pretpostavljena vrijednost za ovaj parametar je `None`.

Implementacija algoritma stabla odluke sadrži parametre koji se odnose na kriterij računanja pri razdvajanju čvorova, određivanja dubine stabla i broja listova i drugih karakteristika konačnog rješenja.

```
class sklearn.tree.DecisionTreeClassifier(*, criterion='gini',
splitter='best', max_depth=None, min_samples_split=2,
min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None,
random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, class_weight=None, ccp_alpha=0.0
```

Kôd 5.16 Parametri implementacije stabla odluke

`Criterion` predstavlja funkciju za određivanje čvorova u stablu. Za vrijednost `gini` koristi se gini indeks kao mjera za informacijsku dobit (engl. *information gain*), a za vrijednost

`entropy` koristi se entropija za informacijsku dobit. Pretpostavljena vrijednost za ovaj parametar je `gini`.

`Splitter` je parametar koji označuje strategiju za odabir sljedećeg razdvajanja u čvoru. `best` označuje da se odabire najbolja podjela, a `random` znači da se odabire nasumična najbolja podjela u čvora.

`Max_depth` je parametar koji označuje najveću dubinu stabla. Ako je vrijednost parametra `None`, onda se čvorovi šire do kad svi listovi nemaju djece ili do kad svi listovi ne sadrže broj djece koji je manji od vrijednosti parametra `min_samples_split`.

`Min_samples_split` označuje minimalni broj primjera na koje se razdvaja čvor. Pretpostavljena vrijednost za ovaj parametar je sva.

`Min_samples_leaf` označuje minimalni broj primjera koji treba biti u listu. Pretpostavljena vrijednost ovog parametra je jedan.

`Min_weight_fraction_leaf` predstavlja minimalnu težinsku frakciju sume svih težina koje trebaju biti u čvoru list. Svi primjeri imaju jednaku težinu kad `sample_weight` parametar nije određen. Pretpostavljena vrijednost ovog parametra je 0.0.

`Max_features` je parametar koji predstavlja broj značajki koje se gledaju kod traženja najbolje podjele u čvoru. Pretpostavljena vrijednost za ovaj parametar je `None`.

`Min_impurity_decrease` znači da će se čvor podijeliti ako je smanjenje „nečistoće“ (engl. *impurity*)veće ili jednako ovoj vrijednosti. Pretpostavljena vrijednost ovog parametra je 0.0.

`Min_impurity_split` je granica za zaustavljanje rasta stabla. Čvor će se podijeliti ako je „nečistoća“ čvora iznad ove granice.

`Class_weight` je parametar koji klasama pridružuje težine. Ako parametar nije postavljen na vrijednosti `dict` ili `balanced`, svim klasama pridružuje se težina jedan. Kad je ovaj parametar postavljen na `balanced`, onda se težine automatski inverzno proporcionalno prilagođavaju ovisno o frekvencijama pojedinih razreda. Pretpostavljena vrijednost za ovaj parametar je `None`.

`Ccp_alpha` je parametar koji se koristi podrezivanja stabla s minimalnom cijenom. Pretpostavljena vrijednost ovog parametra je 0.0.

Implementacija neuronske mreže sadrži parametre koji određuju aktivacijsku funkciju, stopu učenja, veličinu skrivenih slojeva i druge parametre koji se koriste kod izgradnje neuronske mreže.

```
lass sklearn.neural_network.MLPClassifier(hidden_layer_sizes=100,
activation='relu', *, solver='adam', alpha=0.0001, batch_size='auto',
learning_rate='constant', learning_rate_init=0.001, power_t=0.5,
max_iter=200, shuffle=True, random_state=None, tol=0.0001,
verbose=False, warm_start=False, momentum=0.9,
nesterovs_momentum=True, early_stopping=False,
validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-08,
n_iter_no_change=10, max_fun=15000)
```

Kôd 5.17 Parametri implementacije neuronske mreže

Parametar `hidden_layer_size` predstavlja dimenzije neuronske mreže. Pretpostavljena vrijednost je `hidden_layers_sizes = (100,)` što znači da imamo jedan skriveni sloj od 100 neuron i jedan ulazni i izlazni sloj.

`Activation` je parametar koji predstavlja aktivacijsku funkciju u skrivenom sloju. `identity` predstavlja no–op aktivacijsku funkciju, `logistic` predstavlja logističku sigmoidalnu funkciju, `tanh` predstavlja hiperbolnu tan funkciju, `relu` predstavlja ReLu aktivacijsku funkciju (engl. *rectified linear unit function*). Pretpostavljena vrijednost ovog parametra je `relu`.

`Solver` je parametar koji predstavlja funkciju za optimizaciju težina. Vrijednost `lbfgs` predstavlja optimizator unutar kvatzi-Newton metodama (engl. *quasi-Newton methods*). `sgd` označuje stohastički gradijentni spust. `adam` označuje optimizator baziran na stohastičkom gradijentnom spustu. Pretpostavljena vrijednost ovog parametra je `adam`.

Parametar `alpha` je regularizacijski parametar za L2 regularizaciju. Pretpostavljena vrijednost ovog parametra je 0.0001.

Parametar `batch_size` predstavlja veličinu skupova (engl. *minibatches*) za stohastičke optimizatore. Pretpostavljena vrijednost je `auto` koji u slučaju kad je parametar `solver=lbfgs` ne koristi ovaj parametar.

`Learning_rate` predstavlja stopu učenja (engl. *learning rate*) kod ažuriranja težina. `constant` predstavlja konstantnu stopu učenja, `invscaling` predstavlja stopu učenja koja se postupno smanjuje svakim prolazom kroz skup podataka. `adaptive` drži stopu učenja konstantnom dok se gubitak kod učenja smanjuje. Pretpostavljena vrijednost ovog parametra je `constant`.

`Learning_rate_init` predstavlja inicijalnu stopu učenja. Pretpostavljena vrijednost ovog parametra je 0.001.

`Power_t` je eksponent za inverzno skaliranje stope učenja. Koristi se kad je `learning_rate=invscaling` i `solver=sgd`. Pretpostavljena vrijednost ovog parametra je 0.5.

`Max_iter` predstavlja maksimalni broj iteracija. `Solver` iterira do konvergencije ili do ovog broja iteracija. Pretpostavljena vrijednost ovog parametra je 200.

`Shuffle` je parametar koji dok je postavljen na `True` ispremiješa ulazne primjere. Koristi se kad je `solver=adam` ili `solver=sgd`.

`Random_state` je parametar koji određuje nasumičan broj za inicijalizaciju težina i početne konstante. Pretpostavljena vrijednost ove konstante je `None`.

`Tol` je parametar koji predstavlja toleranciju kod optimizacije. Kad se gubitak ne poboljšava barem za `tol` ili nakon `n_iter_no_change` broj iteracija, osim kad je `learning_rate=adaptive`, pretpostavi se da je konvergencija postignuta i prestaje učenje modela. Pretpostavljena vrijednost ovog parametra je 1e-4.

Parametar `verbose` se koristi kad se žele prikazati poruke o napretku na standardni izlaz. Pretpostavljena vrijednost ovog parametra je `False`.

Parametar `warm_start` se postavlja na `True` kad želimo da se rezultat posljednjeg poziva metode `fit` iskoristi za inicijalizaciju. Kad je postavljen na `False`, novim pozivom metode `fit` briše se prethodni rezultat.

Parametar `momentum` predstavlja moment (engl. *momentum*) za ažuriranje kod gradijentnog spusta. Pretpostavljena vrijednost parametra je 0.9, a koristi se kad je `solver=sgd`.

Parametar `nesetrovs_momentum` se postavlja kad se želi koristiti Nesterov momentum, a to je moguće jedino kad je `solver=sgd` i parametar `momentum` postavljen na broj veći od 0. Pretpostavljena vrijednost ovog parametra je `True`.

Parametar `early_stopping` se postavlja na `True` kad želimo terminirati učenje modela ako se rezultat validacije ne popravlja. Automatski se 10% podataka za učenje stavi na stranu i time se provjerava rezultat validacije. Pretpostavljena vrijednost ovog parametra je `False`.

Parametar `validation_fraction` označuje koliki dio podataka za učenje se uzima za validaciju kad je `early_stopping` parametar postavljen na `True`. Pretpostavljena vrijednost ovog parametra je 0.1.

Parametar `beta_1` se koristi kad je `solver=adam`. Pretpostavljena vrijednost ovog parametra je 0.1. On se koristi kod procjene prvog vektora momenta.

Parametar `beta_2` se koristi kad je `solver=adam`. Pretpostavljena vrijednost ovog parametra je 0.999. On se koristi kod procjene drugog vektora momenta.

Parametar `epsilon` predstavlja vrijednost numeričke stabilnosti za `solver=adam`. Pretpostavljena vrijednost ovog parametra je 1e-8.

Parametar `n_iter_no_change` predstavlja maksimalan broj epoha koji može proći bez poboljšanja koje se ograničava parametrom `tol`. Koristi se kad `solver=sgd` ili `solver=adam`. Pretpostavljena vrijednost ovog parametra je deset.

Parametar `max_fun` se koristi kad `solver=lbfgs`. On predstavlja maksimala broj poziva funkcije gubitka. Pretpostavljena vrijednost ovog parametra je 15000.

Stroj potpornih vektora sadrži parametre koji određuju jezgru koja se koristi u algoritmu i druge parametre koji pomažu pri izračunu.

```
class sklearn.svm.SVC(*, C=1.0, kernel='rbf', degree=3,
gamma='scale', coef0=0.0, shrinking=True, probability=False,
tol=0.001, cache_size=200, class_weight=None, verbose=False,
max_iter=- 1, decision_function_shape='ovr', break_ties=False,
random_state=None)
```

Kôd 5.18 Parametri implementacije stroja potpornih vektora

`C` je regularizacijski parametar. Snaga regularizacije je obrnuto proporcionalna ovom parametru. Pretpostavljena vrijednost ovog parametra je jedan.

Parametar `kernel` je jezgra koja se koristi u algoritmu stroja potpornih vektora. Kad je parametar postavljen na `poly` koristi se polinomna jezgra, kad je postavljen na `linear` se koristi linearna jezgra, vrijednost `rbf` predstavlja Gaussovu radijalnu baznu funkciju (engl. *radial basis function*), `sigmoid` predstavlja sigmoidalnu jezgru, `precomputed` opcionalnu funkciju, a `callable` jezgru koja se kreira iz podataka. Pretpostavljena vrijednost ovog parametra je `rbf`.

`Degree` je parametar koji predstavlja stupanj polinomne jezgre. Vrijednost je važeća samo ako je parametar `kernel` postavljen na `poly`. Pretpostavljena vrijednost za ovaj parametar je tri.

Parametar `gamma` je jezgreni koeficijent za polinomnu, `rbf` ili `sigmoidalnu` jezgru. Kad je parametar postavljen na `auto`, vrijednost ovog parametra je $\frac{1}{n_{features}}$, a kad je `gamma=scale`, onda se kao vrijednost koristi $\frac{1}{n_{features} * X.var(\ )}$. Pretpostavljena vrijednost ovog parametra je `scale`.

`Coef0` je nezavisni koeficijent u jezgrenoj funkciji koji se koristi kod polinomne i sigmoidalne jezgre. Pretpostavljena vrijednost ovog parametra je nula.

`Shrinking` je parametar koji se postavlja na `True` ako se želi koristiti „shrinking“ heuristika. Pretpostavljena vrijednost ovog parametra je `True`.

`Probability` je parametar koji omogućuje procjenjivanje na temelju vjerojatnosti. Pretpostavljena vrijednost ovog parametra je `False`.

`Tol` je parametar koji predstavlja toleranciju na kriterij za zaustavljanje. Pretpostavljena vrijednost ovog parametra je 1e-3.

`Cache_size` je parametar koji definira vrijednost jezgrene predmemorije u megabajtima. Pretpostavljena vrijednost ovog parametra je 200.

`Class_weight` je parametar koji klasama pridružuje težine. Ako parametar nije postavljen na vrijednosti `dict` ili `balanced`, svim klasama pridružuje se težina jedan. Kad je ovaj parametar postavljen na `balanced`, onda se težine automatski inverzno proporcionalno prilagođavaju ovisno o frekvencijama pojedinih razreda. Pretpostavljena vrijednost za ovaj parametar je `None`.

Parametar `verbose` se koristi kad se žele prikazati poruke o napretku na standardni izlaz. Pretpostavljena vrijednost ovog parametra je `False`.

`Max_iter` je parametar koji predstavlja granicu za broj iteracija algoritama određenih parametrom `solver`. Pretpostavljena vrijednost ovog parametra je -1, što označuje da nema granice na broj iteracija.

`Decision_function_shape` određuje hoće li decizijska funkcija biti u `ovr` obliku, odnosno one-vs-rest ili u `ovo` obliku, odnosno ove-vs-one obliku. Pretpostavljena vrijednost ovog parametra je `ovr`.

`Break_ties` je parametar koji kad je `True` i `decision_function_shape` parametar je `ovr` i broj klasa je veći od dva, onda se kod neodlučnosti između dvije klase primjeru pridružuje klasa ovisno o intervalu povjerenja decizijske funkcije. Pretpostavljena vrijednost ovog parametra je `False`.

`Random_state` je parametar koji kontrolira generiranje nasumičnih brojeva za ispremještanje podataka kod procjene vjerojatnosti. Pretpostavljena vrijednost ovog parametra je `None`.

6. Utjecaj parametara modela na rezultate

Mijenjanjem parametara algoritma moguće je dobiti modele koji su različite uspješnosti iako su naučeni na istim podatcima. Ispitan je utjecaj par parametara svakog modela na rezultate mjera uspješnosti.

6.1. Parametri logističke regresije

U ovom poglavlju su detaljnije opisani parametri logističke regresijeM.

6.1.1. Hiperparametar solver

Rezultati svih algoritama su slični kad su naučeni na neobrađenim podatcima, standardiziranim podatcima i podatcima na kojima je obavljena selekcija značajki. Mijenjanje parametra `solver` nije uzrokovalo razlike među vrijednostima mjera uspješnosti. Kod poduzorkovanja su algoritmi `newton_cg` i `liblinear` imali jednake rezultate koji su bili bolji od onih dobivenih modelima koji su koristili algoritme `lbfgs` i `saga` za optimizaciju. Kod naduzorkovanja najbolje rezultate dao je model koji je koristio `newton-cg` za optimizaciju.

Logistic regression: solver = newton_cg

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.882012	0.880077	0.880077	0.661017	0.731734	No preprocessing
Recall	0.505199	0.504108	0.496725	0.661017	0.731757	Oversampling
Precision	0.56847	0.543359	0.442607	0.661202	0.732177	Oversampling
F1 Score	0.484462	0.483666	0.468107	0.66092	0.731619	Oversampling

Logistic regression: solver = liblinear

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.880077	0.88588	0.661017	0.718648	No preprocessing
Recall	0.5	0.504108	0.5	0.661017	0.718681	Oversampling
Precision	0.44294	0.543359	0.44294	0.661202	0.71947	Oversampling
F1 Score	0.469744	0.483666	0.469744	0.66092	0.718404	Oversampling

Logistic regression: solver = lbfgs

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.880077	0.88588	0.59322	0.631407	No preprocessing
Recall	0.5	0.504108	0.5	0.59322	0.63138	Oversampling
Precision	0.44294	0.543359	0.44294	0.593328	0.631697	Oversampling
F1 Score	0.469744	0.483666	0.469744	0.593103	0.631175	Oversampling

Logistic regression: solver = saga

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.880077	0.88588	0.567797	0.492912	No preprocessing
Recall	0.5	0.504108	0.5	0.567797	0.492515	Undersampling
Precision	0.44294	0.543359	0.44294	0.644608	0.484106	Undersampling
F1 Score	0.469744	0.483666	0.469744	0.501615	0.415349	Undersampling

Slika 6.7 Prikaz mjera uspješnosti ovisno o parametru solver

Slika 6.1 prikazuje mjere uspješnosti ovisno o parametru solver.

6.1.2. Hiperparametar penalty

Rezultati mjera uspješnosti koje je dao model koji je koristio l2 mjeru za kažnjavanje i onaj koji nije koristio mjeru kažnjavanje dali su jednake rezultate s obzirom na sve metode predobrade. To znači da l2 regularizacija nije imala utjecaj na metode predobrade. Ti modeli su u usporedbi s modelom koji je koristio l1 mjeru za kažnjavanje imali lošije rezultate kod mjera uspješnosti za modele učene na podatcima koji su bili poduzorkovani i preuzorkovani.

Logistic regression: penalty = l1

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.880077	0.880077	0.882012	0.669492	0.721919	Feature selection
Recall	0.496725	0.504108	0.497817	0.669492	0.721946	Oversampling
Precision	0.442607	0.543359	0.442718	0.66954	0.722482	Oversampling
F1 Score	0.468107	0.483666	0.468654	0.669468	0.721759	Oversampling

Logistic regression: penalty = l2

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.880077	0.88588	0.59322	0.631407	No preprocessing
Recall	0.5	0.504108	0.5	0.59322	0.63138	Oversampling
Precision	0.44294	0.543359	0.44294	0.593328	0.631697	Oversampling
F1 Score	0.469744	0.483666	0.469744	0.593103	0.631175	Oversampling

Logistic regression: penalty = none

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.880077	0.88588	0.59322	0.631407	No preprocessing
Recall	0.5	0.504108	0.5	0.59322	0.63138	Oversampling
Precision	0.44294	0.543359	0.44294	0.593328	0.631697	Oversampling
F1 Score	0.469744	0.483666	0.469744	0.593103	0.631175	Oversampling

Slika 6.8 Prikaz mjera uspješnosti ovisno o parametru penalty

Na slici 6.2 prikazana je mjera uspješnosti ovisno o parametru penalty.

6.2. Parametri k-najbližih susjeda

6.2.1. Hiperparametar n_neighbors

Za model učen na neobrađenom skupu podataka se povećanjem broja susjeda povećava točnost i odziv, a smanjuju se preciznost i F1-ocjena. Kod modela naučenog na standardiziranom skupu najbolje rezultate dao je kojemu je parametar `n_neighbors = 10`. Daljnim povećanjem broja susjeda došlo je do smanjenja preciznosti i F1 ocjene. Selekcijom značajki i povećanjem broja susjeda povećava se točnost i odziv, a smanjuju se preciznost i F1-ocjena. Kod poduzorkovanja i učenja modela na tom skupu se povećanjem parametra `n_neighbors` povećavaju vrijednosti svih mjera uspješnosti. Kod učenja modela na preuzorkovanom skupu se povećavanjem parametra `n_neighbours` smanjuju vrijednosti svih mjera uspješnosti.

K-nearest neighbors: n_neighbors = 5

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.870406	0.882012	0.870406	0.466102	0.78735	Standardization
Recall	0.498649	0.549497	0.498649	0.466102	0.787551	Oversampling
Precision	0.492801	0.666854	0.492801	0.465294	0.832451	Oversampling
F1 Score	0.479778	0.561529	0.479778	0.462978	0.779965	Oversampling

K-nearest neighbors: n_neighbors = 10

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.882012	0.887814	0.882012	0.516949	0.667394	Standardization
Recall	0.497817	0.508475	0.497817	0.516949	0.667485	Oversampling
Precision	0.442718	0.943798	0.442718	0.518484	0.672281	Standardization
F1 Score	0.468654	0.486893	0.468654	0.506711	0.665093	Oversampling

K-nearest neighbors: n_neighbors = 20

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.88588	0.88588	0.525424	0.63904	No preprocessing
Recall	0.5	0.5	0.5	0.525424	0.639041	Oversampling
Precision	0.44294	0.44294	0.44294	0.525541	0.639041	Oversampling
F1 Score	0.469744	0.469744	0.469744	0.524878	0.63904	Oversampling

K-nearest neighbors: n_neighbors = 100

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.88588	0.88588	0.559322	0.604144	No preprocessing
Recall	0.5	0.5	0.5	0.559322	0.604252	Oversampling
Precision	0.44294	0.44294	0.44294	0.55939	0.608478	Oversampling
F1 Score	0.469744	0.469744	0.469744	0.559195	0.600294	Oversampling

Slika 6.9 Prikaz mjera uspješnosti ovisno o parametru n_neighbours

Na slici 6.3 prikazana je mjera uspješnosti ovisno o parametru n_neighbours

6.3. Parametri algoritma stabla odluke

6.3.1. Hiperparametar criterion

Na neobrađenom skupu podataka i standardiziranom skupu podataka bolje je rezultate dao model naučen algoritmom kojemu je parametar `criterion` postavljen na `entropy`. Na skupu podataka nad kojim je napravljeno poduzorkovanje, skupu podataka nad kojim je napravljeno preuzorkovanje i na skupu podataka nad kojim je napravljena selekcija značajki bolji modeli su oni naučeni algoritmom kojemu je parametar `criterion` postavljen na `gini`. Kod oba slučaja najbolje modeli su oni naučeni na preuzorkovanim podatcima. Na slici 6.4 prikazana je mjera uspješnosti ovisno o parametru criterion.

Decision tree: criterion = gini

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.816248	0.816248	0.812379	0.70339	0.924755	Oversampling
Recall	0.564059	0.564059	0.591407	0.70339	0.924829	Oversampling
Precision	0.559706	0.559706	0.575346	0.703448	0.932872	Oversampling
F1 Score	0.561681	0.561681	0.581509	0.703369	0.924409	Oversampling

Decision tree: criterion = entropy

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.816248	0.820116	0.804642	0.542373	0.912759	Oversampling
Recall	0.586208	0.595774	0.579657	0.542373	0.912847	Oversampling
Precision	0.574449	0.58271	0.564223	0.542816	0.923735	Oversampling
F1 Score	0.579297	0.588154	0.569752	0.541187	0.912203	Oversampling

Slika 6.10 Prikaz mjera uspješnosti ovisno o parametru criterion

6.3.2. Hiperparametar splitter

Na neobrađenom skupu podataka i standardiziranom skupu podataka bolje je rezultate dao model naučen algoritmom kojemu je parametar `splitter` postavljen na `best`. Na skupu podataka nad kojim je napravljeno poduzorkovanje, skupu podataka nad kojim je napravljeno preuzorkovanje i na skupu podataka nad kojim je napravljena selekcija značajki bolji modeli su oni naučeni algoritmom kojemu je parametar `splitter` postavljen na `random`. Kod oba slučaja najbolje modeli su oni naučeni na preuzorkovanim podatcima. Prikaz mjera uspješnosti ovisno o parametru splitter možemo vidjeti na slici 6.5.

Decision tree: splitter = best

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.816248	0.816248	0.812379	0.70339	0.924755	Oversampling
Recall	0.564059	0.564059	0.591407	0.70339	0.924829	Oversampling
Precision	0.559706	0.559706	0.575346	0.703448	0.932872	Oversampling
F1 Score	0.561681	0.561681	0.581509	0.703369	0.924409	Oversampling

Decision tree: splitter = random

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.829787	0.829787	0.758221	0.661017	0.925845	Oversampling
Recall	0.571701	0.571701	0.516542	0.661017	0.925919	Oversampling
Precision	0.573894	0.573894	0.511632	0.66176	0.933722	Oversampling
F1 Score	0.572759	0.572759	0.509915	0.660627	0.925516	Oversampling

Slika 6.11 Prikaz mjera uspješnosti ovisno o parametru splitter

6.4. Parametri neuronske mreže

6.4.1. Hiperparametar hidden_layer_sizes

Kod modela naučenih na neobrađenim podatcima povećanjem broja slojeva i neurona u slojevima pada točnost, ali se preciznost, odziv i F1-ocjena povećavaju.

Kod modela naučenih na standardiziranim podatcima najbolje rezultate dao je model učen algoritmom kojemu je parametar `hidden_layer_sizes=(100,100,100)`, odnosno neuronska mreža s tri skrivena sloja od kojih svaki sadrži 100 neurona. Kod modela naučenih na skupu podataka nad kojim je napravljena selekcija značajki broj slojeva i neurona nije imao utjecaj na vrijednosti mjera uspješnosti. Kod modela naučenih na poduzorkovanim podatcima povećanje slojeva i broja neurona povećale su mjere uspješnosti. Kod modela naučenih na preuzorkovanim podatcima najbolje rezultate za točnost, preciznost i odziv dao je model s dva skrivena sloja od 50 neurona. Najbolji rezultat za odziv dao je model naučen algoritmom koji ima pet skrivenih slojeva od kojih svaki ima 200 neurona. Prikaz mjera uspješnosti ovisno o parametru hidden_layer_sizes vidi se na slici 6.6.

Neural network: hidden_layer_sizes = (50, 50)

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.682785	0.856867	0.88588	0.483051	0.514722	Feature selection
Recall	0.466583	0.55007	0.5	0.483051	0.515251	Standardization
Precision	0.481769	0.584711	0.44294	0.44537	0.753599	Oversampling
F1 Score	0.462442	0.558548	0.469744	0.375336	0.366116	Standardization

Neural network: hidden_layer_sizes = (100, 100, 100)

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.847195	0.88588	0.5	0.500545	No preprocessing
Recall	0.5	0.559378	0.5	0.5	0.5	Standardization
Precision	0.44294	0.580426	0.44294	0.25	0.250273	Standardization
F1 Score	0.469744	0.566574	0.469744	0.333333	0.333576	Standardization

Neural network: hidden_layer_sizes = (200, 200, 200, 200, 200)

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.83559	0.856867	0.88588	0.567797	0.500545	Feature selection
Recall	0.523296	0.55007	0.5	0.567797	0.500797	Undersampling
Precision	0.532993	0.584711	0.44294	0.620408	0.501012	Undersampling
F1 Score	0.525253	0.558548	0.469744	0.514795	0.472623	Standardization

Slika 6.12 Prikaz mjera uspješnosti ovisno o parametru hidden_layer_sizes

6.5. Parametri stroja potpornih vektora

6.5.1. Hiperparametar kernel

Kod modela naučenih na neobrađenim podatcima najbolje rezultate za odziv, preciznost i F1 ocjenu daju modeli naučeni algoritmom kojem je `kernel=poly`, algoritam kojemu je `kernel`

= rbf i algoritam kojemu je kernel = sigmoid. Kod modela naučenih na standardiziranim podatcima najbolje rezultate za preciznost, odziv i F1 ocjenu daje algoritam kojemu je kernel=poly. Kod modela naučenih na skupu podataka nad kojim je napravljena selekcija značajki najbolje rezultate za odziv, preciznost i F1 ocjenu daju modeli naučeni algoritmom kojem je kernel=poly, algoritam kojemu je kernel = rbf i algoritam kojemu je kernel = sigmoid. Kod modela naučenih na preuzorkovanim podatcima najbolje rezultate za sve mjere uspješnosti daje algoritam kojemu je kernel=linear. Kod modela naučenih na poduzorkovanim podatcima najbolje rezultate za sve mjere uspješnosti daje algoritam kojemu je kernel=linear.

Mjera uspješnosti o parametru kernel prikazana je na slici 6.7.

Support vector machine: kernel = linear

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.878143	0.88588	0.883946	0.652542	0.649945	Standardization
Recall	0.495633	0.5	0.498908	0.652542	0.650056	Undersampling
Precision	0.442495	0.44294	0.442829	0.660326	0.656422	Undersampling
F1 Score	0.467559	0.469744	0.469199	0.648273	0.646386	Undersampling

Support vector machine: kernel = poly

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.883946	0.88588	0.516949	0.508179	No preprocessing
Recall	0.5	0.506291	0.5	0.516949	0.508688	Undersampling
Precision	0.44294	0.610246	0.44294	0.587798	0.56864	Standardization
F1 Score	0.469744	0.485265	0.469744	0.394872	0.371197	Standardization

Support vector machine: kernel = rbf

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.88588	0.88588	0.584746	0.558342	No preprocessing
Recall	0.5	0.5	0.5	0.584746	0.558284	Undersampling
Precision	0.44294	0.44294	0.44294	0.592419	0.558971	Undersampling
F1 Score	0.469744	0.469744	0.469744	0.575944	0.557025	Undersampling

Support vector machine: kernel = sigmoid

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.883946	0.88588	0.559322	0.524537	No preprocessing
Recall	0.5	0.498908	0.5	0.559322	0.524526	Undersampling
Precision	0.44294	0.442829	0.44294	0.584046	0.524535	Undersampling
F1 Score	0.469744	0.469199	0.469744	0.524341	0.524491	Oversampling

Slika 6.13 Prikaz mjera uspješnosti ovisno o parametru kernel

6.5.2. Hiperparametar degree

Kod modela naučenih na neobrađenim podatcima najbolje rezultate za sve mjere uspješnosti dali su algoritmi kojima je polinomna jezgra imala stupanj tri i stupanj pet. Kod modela naučenih na standardiziranim podatcima najbolje rezultate za odziv, preciznost i F1 mjeru dao je algoritam kojemu je polinomna jezgra stupnja pet, a najbolju točnost dao je algoritam kojemu je polinomna jezgra stupnja tri. Kod modela naučenih na skupu podataka nad kojim je napravljena selekcija značajki svi algoritmi dali su jednake mjere uspješnosti neovisno o stupnju polinomne jezgre.

Kod modela naučenih na preuzorkovanim podatcima najbolji model dao je algoritam kojemu je polinomna jezgra stupnja pet. Kod modela naučenih na poduzorkovanim podatcima najbolji model dao je algoritam kojemu je polinomna jezgra stupnja pet. Prikaz mjere uspješnosti ovisno o parametru kernel vidi se na slici 6.8.

Support vector machine: kernel = poly, degree = 3

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.883946	0.88588	0.516949	0.508179	No preprocessing
Recall	0.5	0.506291	0.5	0.516949	0.508688	Undersampling
Precision	0.44294	0.610246	0.44294	0.587798	0.56864	Standardization
F1 Score	0.469744	0.485265	0.469744	0.394872	0.371197	Standardization

Support vector machine: kernel = poly, degree = 5

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.882012	0.88588	0.542373	0.522356	No preprocessing
Recall	0.5	0.512582	0.5	0.542373	0.522854	Undersampling
Precision	0.44294	0.610894	0.44294	0.761062	0.64031	Undersampling
F1 Score	0.469744	0.499293	0.469744	0.421148	0.396319	Standardization

Support vector machine: kernel = poly, degree = 10

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.883946	0.882012	0.88588	0.525424	0.515812	Feature selection
Recall	0.498908	0.505199	0.5	0.525424	0.516309	Undersampling
Precision	0.442829	0.56847	0.44294	0.6139	0.595459	Undersampling
F1 Score	0.469199	0.484462	0.469744	0.411052	0.389583	Standardization

Slika 6.14 Prikaz mjera uspješnosti ovisno o parametru kernel

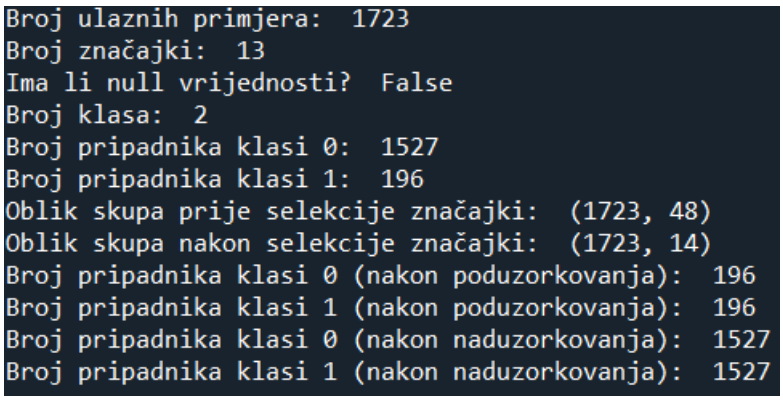
7. Analiza rezultata

Za svaki skup podataka provjereno je ima li nepostojećih vrijednosti. U slučaju da su se pojavile nepostojeće vrijednosti, one bi se morale zamijeniti zamjenskim vrijednostima kako bi algoritmi ispravno funkcionirali. Sve ordinalne i nominalne značajke su kodirane kako bi ih algoritam ispravno interpretirao. Ponavljajući ulazi su uklonjeni kako algoritmi ne bi bili pristrani prema tim vrijednostima. Ostale upotrebljene metode predobrade birane su ovisno o pojedinom skupu podataka. Za svaki skup podataka primjenjene su različite metode predobrade. Za svaki naučeni model određene su mjere uspješnosti pomoću kojih se uspoređuje uspješnost algoritma ovisno o vrsti predobrade podataka. Svi modeli naučeni su na 70% ulaznih podataka, a ispitani pomoću preostalih 30%. Kod učenja modela korištene su pretpostavljene vrijednosti parametara implementacije, a kod implementacije neuronske mreže odabrana je neuronska mreža s tri skrivena sloja od kojih svaki ima po sto neurona.

7.1. Klasifikacija dobrih klijenata

U skupu podataka 'clients.csv' imamo 13 značajki i jednu oznaku klase. Oznaka klase je u skupu podataka označena s 'bad_clients_target'. Značajka 'month' označuje mjesec u kojemu je klijent obavio kupnju u firmi. Značajka 'credit_amount' označuje količinu pozajmice, dok 'loan terms' označuje uvjete koji su potrebni da bi se pozajmica dobila. Ostale značajke se tiču osobnih karakteristika klijenta, a to su npr: Dob, spol, razina obrazovanja, za što namjeravaju podići kredit, broj djece, područje na kojem žive te na kraju mjesečna primanja svakog od klijenta.

Svaki redak ovog skupa podataka označen je s 1 ako je klijent loš, a s 0 ako je klijent dobar. Ovaj skup podataka primjer je nebalansiranog skupa podataka, gdje je veći broj ulaza označenih s 1, a manji broj označen s 0. Budući da su prisutne samo dvije klase, ovo je primjer binarne klasifikacije (engl. *binary classification*). Na slici 7.1. prikazane su informacije o skupu podataka za klasifikaciju dobrih i loših klijenata. Na slici 7.2. prikazan je primjer pojedinih razreda za skup podataka za klasifikaciju istih.



Slika 7.15 Informacije o skupu podataka za klasifikaciju dobrih i loših klijenata

	month	credit_amount	credit_term	age	sex	education	product_type	having_children_flg	region	income	family_status	phone_operator	is_client	bad_client_target
1	1	19000	6	20	male	Secondary special education	Household appliances	1	2	17000	Another	3	1	0
23	1	14000	10	25	female	Secondary special education	Cell phones	0	2	24000	Another	1	1	1

Slika 7.16 Prikaz primjera pojedinih razreda za skup podataka za klasifikaciju dobrih i loših klijenata

Na slici 7.3 prikazani su rezultati mjera uspješnosti za model logističke regresije. U prvom stupcu prikazani su rezultati mjera uspješnosti modela naučenog na neobrađenim podatcima. Vrijednost točnosti je 0.88588, a to znači da je model na skupu podataka za ispitivanje ispravno klasificirao više od 88% primjera. Iako je ova vrijednost visoka, vrijednosti preciznosti, odziva i F1-ocjena su niže, a to je uzrokovano učenjem na neuravnoteženom skupu podataka. Učenjem na takvom skupu dobiven je model koji je pristran većinskoj klasi, pa će u više slučajeva primjeru dati oznaku klase 0 nego 1. Budući da je skup podataka za ispitivanje također nebalansiran i većinska klasa je 0, a model je pristran prema klasi 0 i svrstati će većinu primjera u klasu 0, vrijednost točnosti će biti visoka. Kod nebalansiranih skupova podataka treba obratiti pozornost i na druge mjere kako ne bi došlo do neispravnih zaključaka o uspješnosti modela.Niska vrijednost preciznosti ukazuje da model primjere označene s 1 pogrešno svrstava u 0 u većini slučajeva, odnosno pristran je prema većinskoj klasi 0, a niska vrijednost odziva ukazuje da kad model svrsta primjer u klasu 1, on u većini slučajeva griješi. Vrijednost odziva od 0.5 znači da je 50% pozitivnih primjera iz skupa podataka za ispitivanje koji su trebali biti označeni s 1 neispravno označeno s 0, što znači da je 50% dobrih klijenata pogrešno svrstano pod loše klijente. Ovaj model nije dobar ako je firmi koja koristi model bitno da klijente ne označi krivo kao loše, jer bi to moglo značiti bespotreban gubitak klijenata. Vrijednost preciznosti je 0.44294, pa je time oko 66% primjera označenih s 1 pogrešno označeno. To znači da 66% loših klijenata nije izbačeno, što bi također moglo narušiti poslovanje firme. Vrijednost F1-ocjene, koja je kombinacija preciznosti i odziva također nije veća od 0.5. Kod standardiziranog skupa podataka i skupa podataka na kojemu je obavljeno smanjenje broja značajki, koji su u drugom i trećem

stupcu, dobiveni su slični rezultati kao i kod skupa koji nije obrađen, uz malo povećanje preciznosti, odziva i F1-ocjene. Kod poduzorkovanja i preuzorkovanja došlo je do smanjenja točnosti modela s obzirom na neobrađen skup podataka, gdje je kod poduzorkovanja vrijednost točnosti pala na manje od 0.6. To znači da je model ispravno klasificirao samo 60% ulaza, što nije puno veće od klasifikacije koju bi obavio model koji slučajno pogađa oznake klasa, gdje bi točnost u prosjeku bila 50%. Budući da je skup podataka nebalansiran, treba obratiti pozornost i na druge mjere.

Vrijednosti odziva, preciznosti i F1-mjere su se povećale kod skupova podataka na kojima je obavljeno poduzorkovanje i preuzorkovanje. Ako je za odabir modela bitna vrijednost točnosti, najbolji rezultat dao je model učen na neobrađenom skupu podataka, a ako su bitne ostale mjere, najbolje rezultate dalo je preuzorkovanje.

Performance metrics comparison after preprocessing for logistic regression classifier:

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.880077	0.88588	0.59322	0.631407	No preprocessing
Recall	0.5	0.504108	0.5	0.59322	0.63138	Oversampling
Precision	0.44294	0.543359	0.44294	0.593328	0.631697	Oversampling
F1 Score	0.469744	0.483666	0.469744	0.593103	0.631175	Oversampling

Slika 7.17 Prikaz mjera uspješnosti modela logističke regresije na skupu podataka za klasifikaciju dobrih i loših klijenata

Na slici 7.4 prikazani su rezultati mjera uspješnosti za algoritam k-najbližih susjeda. Učenje modela na neobrađenom skupu podataka dalo je slične rezultate kao i za model logističke regresije. Vrijednost točnosti je visoka, ali su vrijednosti odziva, preciznosti i F1-ocjene nisko što ukazuje na pristranost modela prema većinskoj klasi. Kod k-najbližih susjeda došlo je do takve situacije jer se klasa primjera određuje prema k najbližih primjera. Budući da je općenito više pripadnika klase 0 nego 1, veća je mogućnost da je klasa 0 većinska u k najbližih primjera. Standardizacija je poboljšala vrijednost mjera odziva i preciznosti, odnosno broj primjera koji su pogrešno označeni s 0 i broj primjera koji su pogrešno klasificirani kao 1 se smanjio. Smanjenje broja značajki nije imalo utjecaj na uspješnost modela, dok je kod poduzorkovanja došlo do gubitka informacije budući da su se vrijednosti svih mjera uspješnosti smanjile i dobiven je model koji ispravno klasificira u samo 50% slučajeva. Najbolje rezultate dalo je preuzorkovanje, gdje smanjenje točnosti i povećanje ostalih mjera ukazuju na smanjenje pristranosti prema većinskoj klasi. Iako selekcija značajki nije poboljšala uspješnost modela, smanjila je kompleksnost, a time i povećala brzinu učenja modela.

Performance metrics comparison after preprocessing for k nearest neigbors classifier:

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.870406	0.882012	0.870406	0.466102	0.78735	Standardization
Recall	0.498649	0.549497	0.498649	0.466102	0.787551	Oversampling
Precision	0.492801	0.666854	0.492801	0.465294	0.832451	Oversampling
F1 Score	0.479778	0.561529	0.479778	0.462978	0.779965	Oversampling

Slika 7.18 Prikaz mjera uspješnosti modela k-najbližih susjeda na skupu podataka za klasifikaciju dobrih i loših klijenata

Na slici 7.5 prikazani su rezultati mjera uspješnosti za algoritam stabla odluke. Za model učen na neobrađenim podatcima točnost je visoka, dok su ostale mjere niske, što ukazuje na pristranost prema većinskoj klasi. Standardizacija podataka nije imala utjecaj na mjere uspješnosti. Selekcija značajki malo je poboljšala vrijednosti odziva, preciznosti i F1-ocjene. Kod poduzorkovanja točnost se smanjila, a vrijednosti odziva, preciznosti i f-ocjene su se povećale. To ukazuje da algoritam više nije pristran prema većinskoj klasi. Najbolji model dao je algoritam učen na preuzorkovanim podatcima jer su se vrijednosti svih mjera uspješnosti povećale i veće su od 0.9.

Performance metrics comparison after preprocessing for decision tree classifier:

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.816248	0.816248	0.812379	0.70339	0.924755	Oversampling
Recall	0.564059	0.564059	0.591407	0.70339	0.924829	Oversampling
Precision	0.559706	0.559706	0.575346	0.703448	0.932872	Oversampling
F1 Score	0.561681	0.561681	0.581509	0.703369	0.924409	Oversampling

Slika 7.19 Prikaz mjera uspješnosti modela algoritma stabla odluke na skupu podataka za klasifikaciju dobrih i loših klijenata

Slika 7.20 prikazuje mjere uspješnosti za modele učene neuronskom mrežom. Za model učen na neobrađenim podatcima točnost je visoka, dok su ostale mjere niske, što ukazuje na pristranost prema većinskoj klasi. Standardizacija podataka umanjila je prenaučenost modela pa su se vrijednosti odziva i preciznosti povećale. Selekcija značajki nije imala utjecaj na mjere uspješnosti. Kod poduzorkovanja je došlo do učenja modela koji nije dobro prilagođen neviđenim podatcima, jer su se vrijednosti svih mjera uspješnosti smanjile na vrijednosti ispod 0.5. Preuzorkovanjem su se duplicirali primjeri manjinske klase što je u ovom slučaju dovelo do prenaučenosti modela. Najveću točnost daje model nastao učenjem na neobrađenim podatcima, a najbolju preciznost, odziv i F1-ocjenu dao je model učen na standardiziranim podatcima.

Performance metrics comparison after preprocessing for neural network classifier:

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.847195	0.88588	0.5	0.500545	No preprocessing
Recall	0.5	0.559378	0.5	0.5	0.5	Standardization
Precision	0.44294	0.580426	0.44294	0.25	0.250273	Standardization
F1 Score	0.469744	0.566574	0.469744	0.333333	0.333576	Standardization

Slika 7.20 Prikaz mjera uspješnosti modela neuronske mreže na skupu podataka za klasifikaciju dobrih i loših klijenata

Slika 7.21 prikazuje mjere uspješnosti za modele učene strojem potpornih vektora. Model učen na neobrađenim podatcima ima visoku točnost, ali su vrijednosti odziva, preciznosti i F1-ocjene niske, što bi ukazivalo na model pristran s obzirom na većinsku klasu. Standardizacija i selekcija značajki nisu imale utjecaj na mjere uspješnosti, a poduzorkovanje i preuzorkovanje dali su modele sa smanjenom točnosti s obzirom na neobrađeni skup podataka i povećanim vrijednostima preciznosti, odziva i F1-mjere. Učenjem strojem potpornih vektora dobiveni su modeli koji nemaju visoke vrijednosti mjera uspješnosti, najveću točnost ima model učen na neobrađenim podatcima, a najveću preciznost, odziv i F1-ocjenu ima model učen na preuzorkovanim podatcima.

Performance metrics comparison after preprocessing for support vector machine classifier:

	No preprocessing	Standardization	Feature selection	Undersampling	Oversampling	Best Score
Accuracy	0.88588	0.88588	0.88588	0.584746	0.558342	No preprocessing
Recall	0.5	0.5	0.5	0.584746	0.558284	Undersampling
Precision	0.44294	0.44294	0.44294	0.592419	0.558971	Undersampling
F1 Score	0.469744	0.469744	0.469744	0.575944	0.557025	Undersampling

Slika 7.21 Prikaz mjera uspješnosti modela stroja potpornih vektora na skupu podataka za klasifikaciju dobrih i loših klijenata

7.2. Klasifikacijski robota na temelju razgovora

Svaki redak ovog skupa podataka predstavlja odgovor robota koji se sastoji od deset brojeva i oznake robota koji je dao odgovor. Ovaj skup je balansiran (engl. *balanced*) skup podataka koji se sastoji od ukupno 500 000 redaka. Ovo je primjer višeklasne klasifikacije jer klasa može poprimiti vrijednosti 1, 2, 3, 4 ili 5. Na slici 7.8 prikazane su informacije o skupu podataka za klasifikaciju robota, a na slici 7.9 prikazan je primjer pojedinih razreda za skup podataka za klasifikaciju robota.

```
Broj ulaznih primjera:  500000
Broj značajki:  10
Ima li null vrijednosti?  False
Broj klasa:  5
Broj pripadnika klasi 0:  100000
Broj pripadnika klasi 1:  100000
Broj pripadnika klasi 2:  100000
Broj pripadnika klasi 3:  100000
Broj pripadnika klasi 4:  100000
Oblik skupa prije selekcije značajki:  (500000, 10)
Oblik skupa nakon selekcije značajki:  (500000, 4)
```

Slika 7.22 Informacije o skupu podataka za klasifikaciju robota

	source	num1	num2	num3	num4	num5	num6	num7	num8	num9	num10
5	0	4	9	5	4	8	6	6	7	7	2
6	1	1487	1491	1498	1503	1512	1514	1518	1522	1524	1528
7	2	18850	18850	113100	452400	452400	1809600	3619200	3619200	7238400	50668800
8	3	8962	8967	8972	8977	8982	8987	8992	8997	9002	9007
9	4	2870	14350	71750	358750	1793750	8968750	44843750	224218750	1121093750	5605468750

Slika 7.23 Prikaz primjera pojedinih razreda za skup podataka za klasifikaciju robota

Standardizacija je značajno poboljšala model učen algoritmom logističke regresije čiji su rezultati prikazani na slici 7.10. Selekcija značajki je malo povećala vrijednosti mjera uspješnosti, ali je smanjenjem značajki došlo do bržeg učenja modela, a to je korisno za ovaj skup podataka budući da se sastoji od 500 000 ulaza. Kod predobrade podataka uklanjanjem kolinearnih značajki došlo je do smanjenja vrijednosti mjera uspješnosti. Do toga je došlo zato što su nakon uklanjanja kolinearnih značajki ostale samo dvije značajke, a na početku ih je bilo deset, pa je došlo do gubitka informacije u postupku. Standardizacija se pokazala kao metoda predobrade koja je imala najveći utjecaj na mjere uspješnosti.

Performance metrics comparison after preprocessing for logistic regression classifier:

	No preprocessing	Standardization	Feature selection	Colinear features removed	Best Score
Accuracy	0.265287	0.549493	0.276413	0.199947	Standardization
Recall	0.265287	0.549493	0.276413	0.199947	Standardization
Precision	0.130215	0.533249	0.134793	0.0999917	Standardization
F1 Score	0.172918	0.533367	0.177939	0.133314	Standardization

Slika 7.24 Prikaz mjera uspješnosti modela logističke regresija na skupu podataka za klasifikaciju robota

Najbolje rezultate za model učen pomoću algoritma k-najbližih susjeda na ovim podatcima dala je selekcija značajki, a prikaz rezultata je na **Pogreška! Izvor reference nije pronađen.** 7.10. Kod algoritma k-najbližih susjeda selekcija značajki je korisna budući da je to algoritam

koji je baziran na računanju udaljenosti među primjerima, pa se smanjenjem dimenzije smanjuje kompleksnost računa. Kod standardizacije učen je model koji loše predviđa jer je standardizacijom došlo do smanjenja „razlike“ među značajkama najbližih susjeda budući da je kod takvog algoritama korisno imati jako „razdvojene“ razrede. Uklanjanje kolinearnih značajki neznatno je smanjilo vrijednosti mjera uspješnosti. Na slici 7.11 nalazi se prikaz mjera uspješnosti modela k-najbližih susjeda na skupu podataka za klasifikaciju robota

Performance metrics comparison after preprocessing for k nearest neigbors classifier:

	No preprocessing	Standardization	Feature selection	Colinear features removed	Best Score
Accuracy	0.9113	0.886753	0.92758	0.90158	Feature selection
Recall	0.9113	0.886753	0.92758	0.90158	Feature selection
Precision	0.936399	0.915802	0.945105	0.916588	Feature selection
F1 Score	0.907376	0.87931	0.925482	0.900385	Feature selection

Slika 7.25 Prikaz mjera uspješnosti modela k-najbližih susjeda na skupu podataka za klasifikaciju robota

Standardizacija, selekcija značajki i uklanjanje kolinearnih značajki nisu imale znatan utjecaj na vrijednosti mjera uspješnosti za model učen algoritmom stabla odluke prikazan na Slika 7.26. Najbolje rezultate dao je model učen na neobrađenim podatcima, ali su mjere selekcije značajki ili uklanjanja kolinearnih značajki pojednostavile model budući da algoritam ne prima informacije koje njemu neće biti korisne, pa ih ne mora obraditi i interpretirati.

Performance metrics comparison after preprocessing for decision tree classifier:

	No preprocessing	Standardization	Feature selection	Colinear features removed	Best Score
Accuracy	0.973567	0.97166	0.9721	0.971367	No preprocessing
Recall	0.973567	0.97166	0.9721	0.971367	No preprocessing
Precision	0.976597	0.97506	0.975243	0.974239	No preprocessing
F1 Score	0.973454	0.971523	0.971972	0.971256	No preprocessing

Slika 7.26 Prikaz mjera uspješnosti modela algoritma stabla odluke na skupu podataka za klasifikaciju robota

Najbolji model učen neuronskom mrežom dao je onaj učen na standardiziranim podatcima, a rezultati mjera uspješnosti prikazani su na Slika 7.27. Vrijednosti mjera uspješnosti značajno su se povećale kod primjene standardizacije. Selekcija značajki nije imala povećala je vrijednosti svih mjera uspješnosti, dok je uklanjanje kolinearnih značajki pogoršalo model.

Performance metrics comparison after preprocessing for neural network classifier:

	No preprocessing	Standardization	Feature selection	Colinear features removed	Best Score
Accuracy	0.40002	0.780827	0.603467	0.4	Standardization
Recall	0.40002	0.780827	0.603467	0.4	Standardization
Precision	0.325003	0.774296	0.701675	0.25	Standardization
F1 Score	0.280044	0.72792	0.537308	0.28	Standardization

Slika 7.27 Prikaz mjera uspješnosti modela neuronske mreže na skupu podataka za klasifikaciju robota

Na Slika 7.28 prikazani su rezultati mjera uspješnosti za modele učene algoritmom stroja potpornih vektora. Selekcija značajki nije imala znatan utjecaj na mjere uspješnosti, dok je kod standardizacije i uklanjanja kolinearnih značajki došlo do poboljšanja modela.

Performance metrics comparison after preprocessing for support vector machine classifier:

	No preprocessing	Standardization	Feature selection	Colinear features removed	Best Score
Accuracy	0.658127	0.770527	0.642107	0.6757	Standardization
Recall	0.658127	0.770527	0.642107	0.6757	Standardization
Precision	0.613514	0.765476	0.642869	0.677227	Standardization
F1 Score	0.614378	0.76732	0.593514	0.670292	Standardization

Slika 7.28 Prikaz mjera uspješnosti modela stroja potpornih vektora na skupu podataka za klasifikaciju robota

7.3. Klasifikacija cijena mobilnih uređaja

U skupu podataka 'test.csv' imamo 21 značajku.Značajka 'battery_power' označuje totalnu energiju koju baterija može uskladištiti u jedinici vremena (mili amper sati). 'blue' značajka označava imali uređaj bluetooth ili ne. 'clock_speed' označava brzinu kojom mikroprocesor izvršava zadatke. Značajka 'dual_sim' prikazuje da li uređaj podržava dvije sim kartice ili ne. Značajke koje se također koriste su: mega pikseli prednje kamere (fc), podržava li 4G mrežu (four_g), interna memorija (int_memory), dimenzije i težina uređaja (m_dep, moblie_wt), broj jezgri procesora, ima li zaslon na dodir(touch_screen), ima li WiFI, i cijena uređaja (price_range).

Svaki redak ovog skupa podataka označen je s oznakom 0, 1, 2 ili 4. Te oznake predstavljaju raspon cijene mobitela, gdje je 1 u rasponu cijena s najmanjom vrijednosti, a 4 s najvećom. Budući da je jednak broj ulaza za svaku klasu, ovo je primjer balansiranog skupa podataka.

Budući da su prisutne četiri klase, ovo je primjer višeklasne klasifikacije (engl. *multiclass classification*). Na slici 7.15 možemo vidjeti informacije o skupu podataka za klasifikaciju cijena mobilnih uređaja, dok se na slici 7.16 nalazi prikaz primjera pojedinih razreda za skup podataka klasifikacije cijena mobilnih uređaja.

```
Broj ulaznih primjera:  2000
Broj značajki:  20
Ima li null vrijednosti?  False
Broj klasa:  4
Broj pripadnika klasi 0:  500
Broj pripadnika klasi 1:  500
Broj pripadnika klasi 2:  500
Broj pripadnika klasi 3:  500
Oblik skupa prije selekcije značajki:  (2000, 20)
Oblik skupa nakon selekcije značajki:  (2000, 2)
```

Slika 7.29 Informacije o skupu podataka za klasifikaciju cijena mobilnih uređaja

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	pc	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi	price_range
8	1445	1	0.5	0	0	0	53	0.7	174	7	14	386	836	1099	17	1	20	1	0	0	0
4	1821	1	1.2	0	13	1	44	0.6	141	2	14	1208	1212	1411	8	2	15	1	1	0	1
2	563	1	0.5	1	2	1	41	0.9	145	5	6	1263	1716	2603	11	2	9	1	1	0	2
10	769	1	2.9	1	0	0	9	0.1	182	5	1	248	874	3946	5	2	7	0	0	0	3

Slika 7.30 Prikaz primjera pojedinih razreda za skup podataka za klasifikaciju cijena mobilnih uređaja

Slika 7.31 prikazuje rezultate mjera uspješnosti za modele učene algoritmom logističke regresije. Standardizacija i ekstrakcija značajki su u ovom slučaju dalje jako dobre modele s obzirom na model učen na neobrađenim podatcima, jer su vrijednosti točnosti, preciznosti, odziva i F1-ocjene visoke. Najbolji model je onaj učen na standardiziranim podatcima.

Performance metrics comparison after preprocessing for logistic regression classifier:

	No preprocessing	Standardization	Feature selection	Best Score
Accuracy	0.62	0.96	0.846667	Standardization
Recall	0.62	0.96	0.846667	Standardization
Precision	0.616504	0.960308	0.845708	Standardization
F1 Score	0.617491	0.960097	0.845712	Standardization

Slika 7.31 Prikaz mjera uspješnosti modela logističke regresije na skupu podataka za klasifikaciju cijena mobilnih uređaja

Slika 7.32 prikazuje rezultate mjera uspješnosti modela naučenih algoritmom k-najbližih susjeda. Model učen na neobrađenim podatcima je najbolji model jer su vrijednosti mjera uspješnosti najviše. Kod standardizacije učen je model koji loše predviđa jer je standardizacijom došlo do smanjenja „razlike“ među značajkama, a time i razlike među razredima, koje je onda teško svrstati pomoću algoritma kao što je k-najbližih susjeda budući da je kod takvog algoritama korisno imati jako „razdvojene“ razrede. Selekcijom značajki dobiven je model koji je brži jer k-najbližih susjeda smanjenjem broja značajki pamti manje informacije, ali je narušena uspješnost modela s obzirom na neobrađeni skup.

Performance metrics comparison after preprocessing for k nearest neigbors classifier:

	No preprocessing	Standardization	Feature selection	Best Score
Accuracy	0.905	0.485	0.79	No preprocessing
Recall	0.905	0.485	0.79	No preprocessing
Precision	0.905662	0.513297	0.786842	No preprocessing
F1 Score	0.905005	0.485082	0.78766	No preprocessing

Slika 7.32 Prikaz mjera uspješnosti modela k-najbližih susjeda na skupu podataka za klasifikaciju cijena mobilnih uređaja

Slika 7.33 prikazuje rezultate mjera uspješnosti za modele učene algoritmom stabla odluke. Najbolje rezultate u ovom slučaju daje model učen na skupu podataka koji nije obrađen, a vrijednosti svih mjera uspješnosti su visoke. Standardizacija podataka nije imala velik utjecaj na uspješnost, a kod selekcije značajki došlo je do smanjenja vrijednosti svih mjera zbog gubitka informacije uzrokovane selekcijom.

Performance metrics comparison after preprocessing for decision tree classifier:

	No preprocessing	Standardization	Feature selection	Best Score
Accuracy	0.813333	0.815	0.746667	Standardization
Recall	0.813333	0.815	0.746667	Standardization
Precision	0.816952	0.818573	0.74541	Standardization
F1 Score	0.814576	0.816174	0.745989	Standardization

Slika 7.33 Prikaz mjera uspješnosti modela algoritma stabla odluke na skupu podataka za klasifikaciju cijena mobilnih uređaja

Slika 7.34 prikazuje rezultate mjera uspješnosti modela naučenih pomoću neuronske mreže. Model učen na neobrađenim podatcima loš je klasifikator budući da ima male vrijednosti mjera uspješnosti. Niska točnost kod selekcije značajki ukazuje da model ispravno klasificira samo 47% podataka, odziv vrijednosti 0.473333 znači da je od ukupnog broja pripadnika klase 1 samo 47.3333% ispravno označeno s 1, dok vrijednost preciznosti od 0.35 znači da je samo 35% primjera koje je model označio kao 1 dobro svrstao. Standardizacija podataka pokazala se kao metoda predobrade koja je dala model gdje su vrijednosti mjera uspješnosti visoke, odnosno model dobro generalizira.

Performance metrics comparison after preprocessing for neural network classifier:

	No preprocessing	Standardization	Feature selection	Best Score
Accuracy	0.631667	0.901667	0.473333	Standardization
Recall	0.631667	0.901667	0.473333	Standardization
Precision	0.633511	0.902486	0.350748	Standardization
F1 Score	0.604922	0.901783	0.401074	Standardization

Slika 7.34 Prikaz mjera uspješnosti modela neuronske mreže na skupu podataka za klasifikaciju

cijena mobilnih uređaja

Slika 7.35 prikazuje rezultate mjera uspješnosti za modele učene pomoću algoritma stroja potpornih vektora. Standardizacija i selekcija značajki imaju manje vrijednosti jer je primjenom tih postupaka došlo do gubitka informacije koja bi bila korisna pri učenju modela. Najbolje rezultate je u ovom slučaju dao model učen na neobrađenom skupu podataka.

Performance metrics comparison after preprocessing for support vector machine classifier:

	No preprocessing	Standardization	Feature extraction	Best Score
Accuracy	0.951667	0.861667	0.841667	No preprocessing
Recall	0.951667	0.861667	0.841667	No preprocessing
Precision	0.952111	0.865066	0.84044	No preprocessing
F1 Score	0.951691	0.862542	0.840183	No preprocessing

Slika 7.35 Prikaz mjera uspješnosti modela stroja potpornih vektora na skupu podataka za klasifikaciju cijena mobilnih uređaja

8. Zaključak

Kod modela logističke regresije, standardizacija je u sva tri slučaja poboljšala uspješnost modela. Selekcija značajki ili nema utjecaj ili poboljša uspješnost, ali smanji kompleksnost modela i ubrzava učenje algoritma. Kod učenja modela algoritmom logističke regresije na nebalansiranom skupu podataka preuzorkovanje se pokazalo kao najbolje rješenje, dok je kod poduzorkovanja došlo do gubitka informacije i time pogoršanja uspješnosti modela. Ista stvar se dogodila i kod uklanjanja kolinearnih značajki.

Standardizacija podataka pokazala se kao loš odabir za metodu predobrade kod algoritma k-najbližih susjeda. Kod standardizacije učen je model koji loše predviđa jer je standardizacijom došlo do smanjenja „razlike“ među značajkama, a time i razlike među razredima, koje je onda teško svrstati pomoću algoritma kao što je k-najbližih susjeda budući da je kod takvog algoritama korisno imati jako „razdvojene“ razrede. Selekcija značajki ili nema utjecaj na mjere uspješnosti, ili pogoršava model k-najbližih susjeda. Poduzorkovanje je narušilo rad modela naučenog algoritmom k-najbližih susjeda, dok je preuzorkovanje znatno povećalo mjere uspješnosti. Selekcija značajki je u jednom slučaju jako narušila mjere uspješnosti, dok je u druga dva poboljšala model, a time i smanjila kompleksnost i vrijeme učenja modela. Uklanjanjem kolinearnih značajki došlo je do gubitka informacije i pogoršanja modela.

Standardizacija nema značajan utjecaj na rad algoritma stabla odluke zato što su algoritmi zasnovani na stablima neosjetljivi su na skalu značajki. Stablo odluke dijeli čvor na temelju jedne značajke koja povećava homogenost čvora. Na ovu podjelu značajke ne utječu druge značajke.[11]

Selekcija značajki pokazala je različite rezultate s obzirom na skup podataka, u jednom slučaju je poboljšala, u jednom pogoršala, a u trećem nije imala utjecaja na mjere uspješnosti. Općenito stabla odluke sama biraju optimalne značajke u postupku učenja, pa metode selekcije nemaju velik utjecaj. Kod nebalansiranog skupa podataka najbolje je rezultate dalo preuzorkovanje, ali je i poduzorkovanje dalo bolje rezultate. Ove metode spriječile su prenaučenost stabla odluke koje bi se inače prilagodilo većinskoj klasi. Uklanjanje kolinearnih značajki nije imalo velik utjecaj na mjere uspješnosti. To proizlazi zbog činjenice da stabla odluke "ne gledaju" odnose među značajkama, već stablo odluke dijeli čvor na temelju jedne značajke, pa na tu podjelu druge značajke nemaju utjecaja [11].

Kod algoritma neuronske mreže se standardizacija pokazala kao najbolja metoda predobrade. Sve ostale metode su pokazale nepredvidiv utjecaj na mjere ili su samo pogoršale modele. Do pogoršanja modela došlo je zbog gubitka informacije.

Za algoritam stroja potpornih vektora standardizacija je u jednom slučaju poboljšala mjere uspješnosti. Selekcija značajki imala je loš utjecaj na mjere uspješnosti. Kod nebalansiranog skupa podataka su se poduzorkovanje i preuzorkovanje pokazale kao najbolje metode predobrade. Uklanjanje kolinearnih značajki poboljšalo je rad modela. Za algoritam stroja potpornih vektora ne postoji metoda predobrade koja uvijek "pomaže" u radu modela, već je rad modela i odabir metoda predobrade ovisan o skupu podataka.

Zbog selekcije značajki može doći do smanjene uspješnosti kod algoritama logističke regresije stabla odluke i algoritma k-najbližih susjeda zbog gubitka informacije i interakcije značajki do koje dolazi zbog brisanja značajki. Kod nekih modela selekcija značajki nije pokazala utjecaj na mjere uspješnosti, ali je ubrzala rad algoritma i smanjila kompleksnost modela. Iako kod učenja linearnih algoritama kao što je logistička regresija nije poželjno imati kolinearne značajke, brisanje takvih značajki može dovesti do pogoršanja rada algoritma u slučaju kad je većina značajki korelirana, jer bi brisanjem došlo do gubitka informacija koje su potrebne za donošenje zaključaka o skupu podataka. Za nebalansiran skup podataka, metoda preuzorkovanja uspješno je povećala manjinski razred pa je zato došlo do povećanja vrijednosti mjera uspješnosti. Metoda poduzorkovanja je kod većine modela uzrokovala gubitak informacije koje su bile informativne kod učenja, pa su dobiveni modeli lošiji. Standardizacija se pokazala dobra za modele koji su osjetljivi na skalu, kao što su logistička regresija i neuronska mreža [12].

Kod obrade podataka prije učenja modela u obzir se moraju uzeti karakteristike skupa podataka i algoritmi koje učimo kako bi se odredilo koje metode obrade treba upotrijebiti. Obrada različitih skupova podataka zahtjeva različite pristupe. Iako se obradom podataka nastoji smanjiti kompleksnost i povećati uspješnost modela koji učimo, u obzir treba uzeti i gubitak informacije i interakcije među značajkama do kojeg dolazi upotrebom pojedinih metoda obrade podataka, a time i pada vrijednosti mjera uspješnosti.

9. Literatura

[1] Jason Brownlee Master Machine Learning Algorithms Discover How They Work and Implement Them From Scratch, Supervised, Unsupervised and Semi-Supervised Learning, 2016.

[2] Jason Brownlee Master Machine Learning Algorithms Discover How They Work and Implement Them From Scratch, K-Nearest Neighbors 98 22.1 KNN Model Represen, 2016.

[3] Jason Brownlee Master Machine Learning Algorithms Discover How They Work and Implement Them From Scratch, Logistic Regression, 2016.

[4] Jason Brownlee Master Machine Learning Algorithms Discover How They Work and Implement Them From Scratch, Classification and Regression Trees, 2016.

[5] Stabla odlučivanja, Marijana Zekić-Sušac,2017. http://www.efos.unios.hr/sustavi-poslovne-inteligencije/wp-content/uploads/sites/192/2017/10/P4_Stabla-odlucivanja-2017.pdf

Datum pristupa dokumentu: 8.8.2021.

[6] Primjena stabla odlučivanja na skupu podataka iz obrazovanja, Žitković Bruno, 2020, datum pristupa dokumenta: 18.8.2021. https://repozitorij.foi.unizg.hr/islandora/object/foi:5818,

[7] Jason Brownlee Master Machine Learning Algorithms Discover How They Work and Implement Them From Scratch, Supervised, Support Vector Machines, 2016.

[8] Što je strojno učenje?,Osnove strojnog učenja Tutorial za strojno učenje Edureka !, datum pristupa dokumentu: 21.8.2021. https://hr.strephonsays.com/difference-between-machine-learning-and-neural-networks

[9] Mark A. Hall and Lloyd A. Smith, University of Waikato, Feature Selection For Machine Learning: Comparing a Correlation-based Filter Approach to the Wrapper, https://www.aaai.org/Library/FLAIRS/1999/flairs99-042.php , datum pristupa dokumentu: 21.8.2021.

[10] Collinearity: A review of methods to deal with it and a simulation study evaluating their performanc, Carsten F. Dormann , Jane Elith, Sven Bacher , Carsten Buchmann , Gudrun Carl , Gabriel Carré, Jaime R. García Marqu é z, Bernd Gruber, Bruno Lafourcade, Pedro J. Leitão, Tamara Münkemüller, Colin McClean, Patrick E. Osborne, Björn Reineking, Boris Schröder, Andrew K. Skidmore,Damaris Zurelland, Sven Lautenbach, 2013, datum pristupa dokumentu: 9.8.2021.

[11] Feature Scaling for Machine Learning: Understanding the Difference Between Standardization and Normalization : Tree-based Algorithms, Bhandari A., https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/, 2020, datum pristupa dokumentu: 24.8.2021

[12] Feature Scaling for Machine Learning: Understanding the Difference Between Standardization and Normalization : Gradient Descent Based Algorithms, Bhandari A., https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/, 2020, datum pristupa dokumentu: 24.8.2021

[13] Umjetna neuronska mreža, slika 2.3 https://hr.wikipedia.org/wiki/Umjetna_neuronska_mre%C5%BEa , 2018, datum pristupa dokumentu: 10.9.2021.

10. Popis slika

11. Prilozi

SKUPOVI PODATAKA:

https://www.kaggle.com/iabhishekofficial/mobile-price-classification

https://www.kaggle.com/msk1097/classification-of-robots-from-their-conversation

https://www.kaggle.com/podsyp/is-this-a-good-customer

Sveučilište Sjever

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, DAVID STEVIĆ *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom ANALIZA UTJECAJA OBRADE PODATAKA NA USPJEŠNOST KLASIFIKACIJSKIH ALGORITAMA STROJNOG UČENJA *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)

_____
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, DAVID STEVIĆ *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom ANALIZA UTJECAJA OBRADE PODATAKA NA USPJEŠNOST KLASIFIKACIJSKIH ALGORITAMA STROJNOG UČENJA *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)

_____
(vlastoručni potpis)