UNIVERZITET U ZENICI
POLITEHNIČKI FAKULTET

PREDVIĐANJE PREŽIVLJAVANJA NA TITANICU UPOTREBOM KLASIFIKACIJSKIH METODA

PROJEKAT

Predmet: Rudarenje podataka

Profesor: Doc.dr.sci. Adnan Dželihodžić

Studenti: Mujo Kalabuzić, Nejra Rizvić

Zenica, Juni 2025.

Sadržaj

1. UVOD

U oblasti mašinskog učenja (ML), klasifikacija predstavlja jedan od ključnih zadataka koji podrazumijeva svrstavanje podataka u unaprijed definisane kategorije. Klasifikacioni algoritmi se obučavaju na označenim podacima kako bi naučili paterne koji im omogućavaju da donose tačne odluke o novim, nepoznatim podacima.

“Classification in machine learning models is a predictive modeling process by which machine learning models use classification algorithms to predict the correct label for input data.

A classification model is a type of machine learning model that sorts data points into predefined group called classes.” [1] (Klasifikacija u mašinskom učenju je proces prediktivnog modeliranja kojim modeli mašinskog učenja koriste algoritme klasifikacije kako bi predvidjeli ispravnu oznaku za ulazne podatke. Klasifikacijski model je vrsta modela mašinskog učenja koji sortira tačke podataka u unaprijed definisane grupe koje se nazivaju klase)

U ovom projektu, zadatak klasifikacije se primjenjuje na jedan dosta poznat, historijski slučaj - potonuće Titanika. Osnovni cilj jeste kreiranje ML modela koji će izvršiti predviđanje da li je putnik preživio ili nije preživio ovu tragediju. Ovo predstavlja klasičan binarni klasifikacioni problem, gdje na osnovu informacija o putnicima i upotrebom različitih metoda (Logistička Regresija i Random Forest) predviđamo preživljavanje putnika na brodu. Projekat obuhvata obradu i analizu podataka, treniranje modela kao i evaluaciju istog.

Također, budući da će se kreirati dva ML modela, na kraju samog projekta će se izvršiti uporedna analiza istih, pri ćemu će fokus biti na samoj prilagodbi modela podacima, kao i evaluaciji njihove tačnosti.

2. DATASET

Za potrebe ovog projekta, upotrebljen je već prethodno dostupni dataset, a koji uključuje sve potrebne informacije o putnicima, kako bi se mogao kreirati, odnosno trenirati ML model koji će izvršiti potrebnu klasifikaciju.

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owe	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. Jo	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss.	female	26	0	0	STON/O2. 3101	7.925		S
4	1	1	Futrelle, Mrs. Jac	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr. Jame	male		0	0	330877	8.4583		Q
7	0	1	McCarthy, Mr. Tir	male	54	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Master.	male	2	3	1	349909	21.075		S
9	1	3	Johnson, Mrs. O	female	27	0	2	347742	11.1333		S
10	1	2	Nasser, Mrs. Nic	female	14	1	0	237736	30.0708		C
11	1	3	Sandstrom, Miss	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, Miss. El	female	58	0	0	113783	26.55	C103	S
13	0	3	Saundercock, Mr	male	20	0	0	A/5. 2151	8.05		S
14	0	3	Andersson, Mr. A	male	39	1	5	347082	31.275		S
15	0	3	Vestrom, Miss. H	female	14	0	0	350406	7.8542		S
16	1	2	Hewlett, Mrs. (Ma	female	55	0	0	248706	16		S
17	0	3	Rice, Master. Eu	male	2	4	1	382652	29.125		Q
18	1	2	Williams, Mr. Cha	male		0	0	244373	13		S

Slika 2.1 Train dataset

Slika iznad, predstavlja dostupni dataset, koji je upotrebljen za samo treniranje modela mašinskog učenja. Svaki red unutar ovog dataseta, predstavlja jednog putnika i uključuje izlaznu (label - ciljnu) vrijednost *Survived*, koja označava da li je putnik preživio. Također, imamo još nekoliko kolona:

- PassengerId - jedinstven identifikator putnika
- Pclass - klasa karte - može biti važna kolona, upravo zbog samih različitih klasa (1,2,3) i možemo reći prednosti putnika na osnovu istih
- Name - ime putnika
- Sex - spol putnika - možemo reći dosta bitan kolona u ovom slučaju
- Age - starost putnika - bitna karakteristika, djeca i žene su imali veću šansu da budu spašeni
- SibSp - broj braće/sestara ili supružnika na brodu - može biti bitna kolona koja pokazuje da li je putnik sam ili na porodičnom putovanju
- Parch - broj roditelja ili djece na brodu - kolona slična prethodnoj
- Ticket - broj karte

- Fare - cijena karte
- Cabin - broj kabine
- Embarked - Luka ukrcavanja - C = Cherbourg, Q = Queenstown, S = Southampton

Također, možemo vidjeti na slici, da za određene kolone postoje nedostajuće vrijednosti, što će biti objašnjeno u narednim dijelovima (priprema podataka za modele ipynb file sekcija).

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292		Q
893	3	Wilkes, Mrs. Jam	female	47	1	0	363272	7		S
894	2	Myles, Mr. Thom	male	62	0	0	240276	9.6875		Q
895	3	Wirz, Mr. Albert	male	27	0	0	315154	8.6625		S
896	3	Hirvonen, Mrs. A	female	22	1	1	3101298	12.2875		S
897	3	Svensson, Mr. Jo	male	14	0	0	7538	9.225		S
898	3	Connolly, Miss. K	female	30	0	0	330972	7.6292		Q
899	2	Caldwell, Mr. Alb	male	26	1	1	248738	29		S
900	3	Abrahim, Mrs. Jo	female	18	0	0	2657	7.2292		C
901	3	Davies, Mr. John	male	21	2	0	A/4 48871	24.15		S
902	3	Ilieff, Mr. Ylio	male		0	0	349220	7.8958		S
903	1	Jones, Mr. Charl	male	46	0	0	694	26		S
904	1	Snyder, Mrs. Joh	female	23	1	0	21228	82.2667	B45	S
905	2	Howard, Mr. Ben	male	63	1	0	24065	26		S

Slika 2.2 Test dataset

Slika iznad predstavlja test dataset, a koji bi mogao biti upotrebljen za finalnu predikciju modela, odnosno, predviđanje vrijednosti *Survived* za putnike u datasetu. Također, bitno je naglasiti da se u ovom datasetu nalaze različiti putnici, ali su struktura i ostale kolone identične, osim *Survived* kolone koja će biti prediktovana.

PassengerId	Survived
892	0
893	1
894	0
895	0
896	1
897	0
898	1
899	0
900	1
901	0

Slika 2.3 Submission

Slika iznad predstavlja izlaznu vrijednost samog modela, pri čemu možemo vidjeti prediktovanu vrijednost *Survived*, za putnike iz testnog skupa. Budući da je sam dataset preuzet,

izlaz modela bi trebao biti upravo na osnovu ovog fajla, pri čemu će se u ovom slučaju samo izvršavati kreiranje, testiranje i evaluacija modela.

3. OPIS PROBLEMA

U ovom projektu razmatra se problem klasifikacije, konkretno, predviđanja da li je određeni putnik preživio potonuće Titanika ili ne. Na osnovu dostupnih podataka o svakom putniku, kao što su spol, starost, klasa karte, broj članova porodice na brodu, cijena karte i luka u kojoj se ukrcao putnik, cilj je trenirati modele koji će moći donijeti što tačniju, precizniju odluku o ishodu preživljavanja.

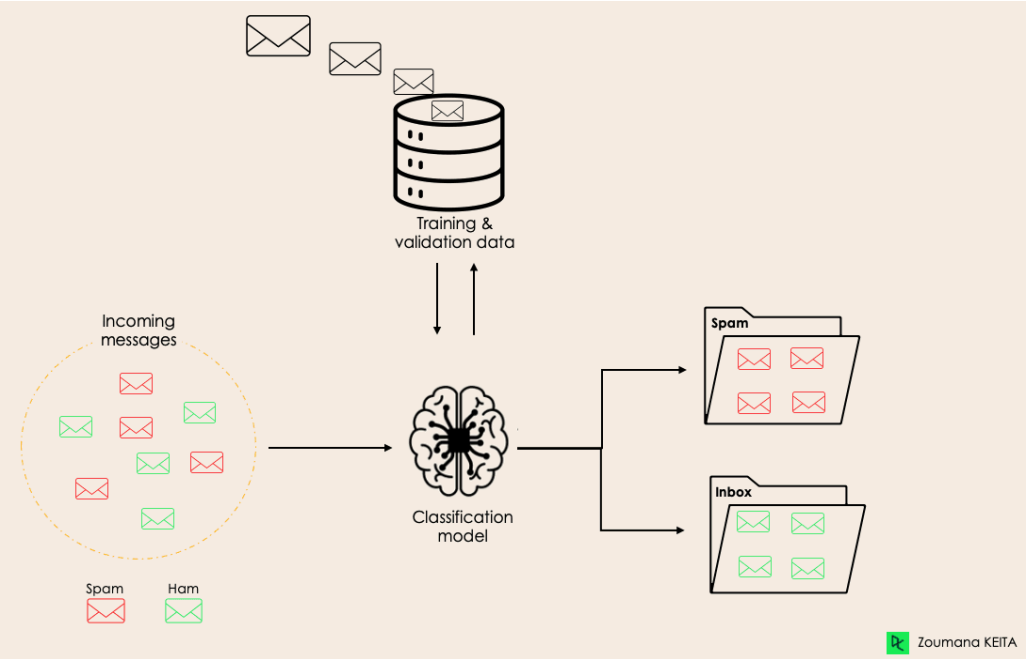
U ovom slučaju možemo reći da je problem binarne prirode, jer je izlazna vrijednost (ciljna vrijednost - label) *Survived* predstavljena kroz dvije moguće klase: 0 (nije preživio) i 1 (preživio). Važno je pravilno razumjeti i pripremiti podatke, upravo jer se u njima mogu pronaći korisni paterni i veze između različitih osobina, informacija putnika i njihovih ishoda. Sam problem nije samo tehnički, jer zahtijeva i razumijevanje konteksta podataka. Na primjer, spol i klasa putovanja imali su značajan uticaj na šanse preživljavanja, što se mora dodatno uzeti u obzir tokom same analize podataka za model.

Ovaj zadatak ne samo da provjerava tehničku primjenu klasifikacijskih metoda, već zahtijeva obradu i analizu podataka, izbor relevantnih osobina te razumijevanje samog problema, kako bi se razvili modeli koji dobro generalizuju i nepoznate podatke.

4. OPIS ALGORITMA

Klasifikacija u mašinskom učenju je proces prediktivnog modeliranja kojim modeli mašinskog učenja koriste algoritme klasifikacije kako bi predvidjeli ispravnu oznaku/label za ulazne podatke. Kako modeli uče analizirati i klasificirati podatke u svojim skupovima podataka za trening, oni postaju vještiji u identificiranju različitih tipova podataka, ali i davanju preciznijih predviđanja. [1]

Klasifikacija spada u grupu supervised metoda mašinskog učenja, gdje modeli pokušavaju predvidjeti ciljnu vrijednost na osnovu ulaznih podataka. Dakle, model se u potpunosti uči koristeći trening podatke, onda se evaluira na testnim podacima prije same upotrebe na realnim podacima. [2]



Slika 4.1 Spam Klasifikacija [3]

Primjer klasifikacije bi bio na slici, pri čemu model može predvidjeti da li je email spam ili ne. Dakle, imamo ulazne podatke, u ovom slučaju emailove, koji će biti upotrebljeni za trening i test klasifikacijskog modela, nakon čega će se izvršiti potrebna evaluacija, odnosno upotreba kreiranog modela nad nepoznatim podacima. [2]

U slučaju ovog projekta, rješava se problem klasifikacije ali kreiranjem dva modela, pri čemu će se na osnovu metrika uporediti način na koji su se modeli prilagodili samim podacima. Također, klasifikacijski algoritmi, a koje će biti objašnjeni u nastavku, su:

- Random Forest
- Logistic Regression

Također, bitno je naglasiti da je izbor modela uglavnom bio na osnovu prednosti samih algoritama. Na primjer Logistička Regresija je predstavljala dobru početnu tačku za rješavanje ovog problema, upravo jer predstavlja jednostavan model koji pokazuje kako svaki podaci utiču na odluku. Random Forest, možemo reći kompleksniji model koji dobro radi i sa komplikovanim podacima i kada između njih ima nelinearna veza, kombinuje više stabala i na kraju glasa za najbolji rezultat.

4.1 Random Forest

Za potrebe rješavanja ovog problema, jedan od upotrebljenih je Random Forest algoritam. Na osnovu same analize podataka, možemo reći da bi ovaj algoritam mogao dati

dobre rezultate, upravo zbog samih odnosa i pripreme podataka koji će biti upotrebljeni za trening i kreiranje modela.

```
rf_grid = GridSearchCV(
       estimator=RandomForestClassifier(random_state=42),
      param_grid=param_grid,
      cv=5, # K-fold cross-validation
      scoring='recall', # Metrika za evaluaciju
     n_jobs=-1, # Koristi sve dostupne CPU jezgre
     verbose=2
)
```

Na osnovu samog koda iznad, možemo vidjeti nekoliko bitnih stvari za dobijanje ovog modela:

- Upotreba GridSearchCV - za hiperparametre (jedna od naprednijih metoda za poboljšanje algoritma)
- Fokus bi trebao biti recall metrika, upravo zbog odlučnosti da model uhvati što više pozitivnih rezultata
- param_grid - vrijednosti hiperparametara, u ovom slučaju bitno za model upravo jer to uključuje npr. koliko stabala, koliko duboka stabla itd.
- cv = 5 - unakrsna validacija, da bi sam model bolje generalizovao

```
rf_grid.fit(X_train, y_train)

best_rf_model = rf_grid.best_estimator_

rf_predictions = best_rf_model.predict(X_val)
```

Treniranje modela se izvršilo na osnovu prethodno podijeljenih podataka na trening i test, konkretno, ovi podaci su nakon podjele upotrebljeni i za samu evaluaciju modela. Na osnovu GridSearchCV-a smo dobili najbolju kombinaciju hiperparametara, odnosno najbolji model na osnovu proslijeđenih vrijednosti, a pomoću kojeg smo poslije izvršili predikciju.

4.2 Logistic Regression

Za početni pregled podataka, ali da bi se izvršila i uporedba modela sa prethodno objašnjenim, izabrao se algoritam Logističke Regresije. Algoritam koji predviđa vjerovatnoću pripadnosti jednoj od dvije klase, što u ovom slučaju predstavlja labela *Survived* (binarna klasifikacija). Dosta lakše i brže se trenira u odnosu na prethodni algoritam, koristi linearnu kombinaciju ulaza i vraća vrijednost koja predstavlja vjerovatnoću klase.

```
lr_grid = GridSearchCV(
       estimator=LogisticRegression(),
      param_grid=lr_param_grid,
      cv=5, # K-fold cross-validation
     scoring='accuracy', # Metrika za evaluaciju
    n_jobs=-1, # Koristi sve dostupne CPU jezgre
    verbose=2
)
```

Na osnovu prethodnog koda, možemo vidjeti identičnu postavku za kreiranje modela, pri čemu se ponovno koristi GridSearchCV za pretragu hiperparametara za ovaj model. Također, upotreba unakrsne validacije za preciznu procjenu performansi modela, odnosno metrike tačnosti - procenta tačno klasifikovanih primjera.

```
lr_grid.fit(X_train, y_train)

best_lr_model = lr_grid.best_estimator_

best_lr_model = best_lr_model.predict(X_val)
```

Prethodni kod prestavlja treniranje modela, upotrebom prethodno podijeljenih podataka na trening i test, a na osnovu kojih će se izvršavati i potrebna evaluacija. Izvršava se izbor najboljeg modela (estimatora) na osnovu automatske pretrage hiperparametara, kao i finalna predikcija modela.

5. METODE UPOTREBLJENE ZA POBOLJŠANJE ALGORITMA

Za sam razvoj modela koji će predviđati preživljavanje putnika na Titaniku, upotrebljeno je nekoliko različitih tehnika za poboljšanje tačnosti, generalizacije modela. Uključujemo nekoliko metoda:

- *Feature Engineering - FamilySize, IsAlone*

```
df_train['FamilySize'] = df_train['SibSp'] + df_train['Parch'] + 1

df_test['FamilySize'] = df_test['SibSp'] + df_test['Parch'] + 1


df_train['IsAlone'] = 0

df_train.loc[df_train['FamilySize'] == 1, 'IsAlone'] = 1


df_test['IsAlone'] = 0

df_test.loc[df_test['FamilySize'] == 1, 'IsAlone'] = 1
```

U ovom slučaju, obje kolone predstavljaju kombinaciju već postojećih kolona, da bi se dobile nove informacije. Ove kolone daju dodatni kontekst modelu, npr. putnici koji putuju sami, imaju manju šansu za preživljavanje u Titanic datasetu.

- *Mapiranje kategorija i One-Hot Encoding*

```
sex_map = {'male': 0, 'female': 1}

df_train['Sex'] = df_train['Sex'].map(sex_map)

df_test['Sex'] = df_test['Sex'].map(sex_map)


df_train = pd.get_dummies(df_train, columns=['Embarked'])

df_test = pd.get_dummies(df_test, columns=['Embarked'])
```

Pretvaranje npr kolone Sex, koja ima vrijednosti male/female u numeričke ulaze. Dosta korisne tehnike, upravo jer upotrebljeni modeli zahtijevaju numeričke ulaze.

- *Popunjavanje nedostajućih vrijednosti - Age, Fare, Embarked*

```
age_median = df_train['Age'].median()

df_train['Age'] = df_train['Age'].fillna(age_median)
```

```
df_test['Age'] = df_test['Age'].fillna(age_median)


embarked_mode = df_train['Embarked'].mode()[0]
df_train['Embarked'] = df_train['Embarked'].fillna(embarked_mode)
df_test['Embarked'] = df_test['Embarked'].fillna(embarked_mode)


fare_median = df_train['Fare'].median()
df_test['Fare'] = df_test['Fare'].fillna(fare_median)
```

U ovom slučaju dosta bitna tehnika, pri čemu se koristio median zbog toga što nije osjetljiv na ekstremne vrijednosti, također, ovo omogućava modelu da radi sa punim datasetom.

- Hiperparametri modela

```
param_grid = {
    'n_estimators': [100, 500, 1000], # Broj stabala u šumi
    'max_depth': [None, 10, 20, 30], # Maksimalna dubina stabla
    'min_samples_split': [2, 5], # Minimalan broj uzoraka potreban za podjelu čvora
    'min_samples_leaf': [1, 2, 4], # Minimalan broj uzoraka u listu
    'max_features': ['sqrt', 0.3], # Broj karakteristika koje se koriste za podjelu čvora
    'class_weight': [None, 'balanced'], # Težine klasa za nerazmjerno raspoređene klase
    'bootstrap': [True, False], # Da li koristiti bootstrap uzorkovanje
}


# Inicijalizacija GridSearchCV sa RandomForestClassifier
rf_grid = GridSearchCV(estimator=RandomForestClassifier(random_state=42),
                                param_grid=param_grid,
                                cv=5, # K-fold cross-validation
                                scoring='recall', # Metrika za evaluaciju
                                n_jobs=-1, # Koristi sve dostupne CPU jezgre
                                verbose=2)
```

Dosta bitna napredna tehnika koja automatski pronalazi najbolje kombinacije hiperparametra za modele, dakle, na ovaj način će se sistemski testirati više opcija uz cross-validation. Pri čemu možemo vidjeti u kodu iznad da je ovo primjer za Random Forest algoritam, ali da su i

proslijeđeni za automatsko pronalaženje dosta bitni faktori algoritma, poput broja stabala, koliko su stabla duboka, da li se koristi bootstraping uzoraka, minimalan broj uzoraka u listu itd.

```
lr_param_grid = {
    'C': [0.01, 0.1, 1, 10, 100], # Inverzna regularizacija
    'penalty': ['l1', 'l2'], # Regularizacija
    'solver': ['liblinear'], # Solver za optimizaciju
    'class_weight': ['balanced'], # Težine klasa za nerazmjerno raspoređene klase
    'max_iter': [1000] # Maksimalan broj iteracija
}


# Inicijalizacija GridSearchCV sa LogisticRegression
lr_grid = GridSearchCV(estimator=LogisticRegression(),
                                param_grid=lr_param_grid,
                                cv=5, # K-fold cross-validation
                                scoring='accuracy', # Metrika za evaluaciju
                                n_jobs=-1, # Koristi sve dostupne CPU jezgre
                                verbose=2)
```

Kod iznad predstavlja ponovnu upotrebu GridSearchCV, ali ovaj put za algoritam Logističke Regresije, pri čemu možemo vidjeti različite hiperparametre poput izbora regularizacije, solvera koji je upotrebljen u ovom slučaju, maksimalan broj iteracija, inverzna vrijednost jačine regularizacije.

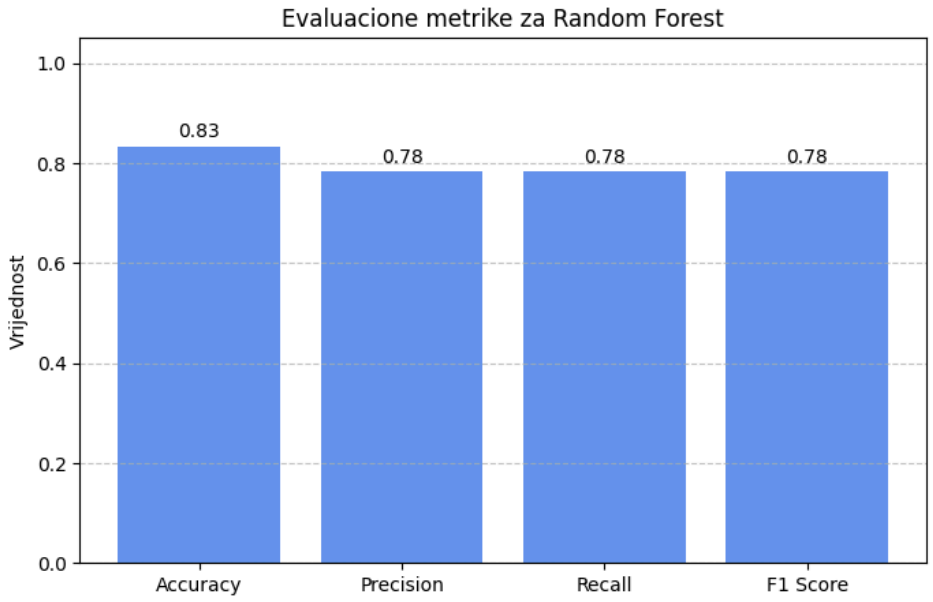
Bitno je naglasiti da će GridSearchCV da isproba više kombinacija hiperparametara i da pronađe onu koja daje najbolji model, pri čemu u zavisnosti od samog broja kombinacija trening modela se izvršava sporo/brzo.

6. REZULTATI I NJIHOVA INTERPRETACIJA

Nakon samog kreiranja i treniranja modela, izvršena je evaluacija istih, pri čemu treba napomenuti, upotrebljen je testni skup, odnosno osnovne metrike za analizu performansi kreiranog modela. Upotrebljene metrike su:

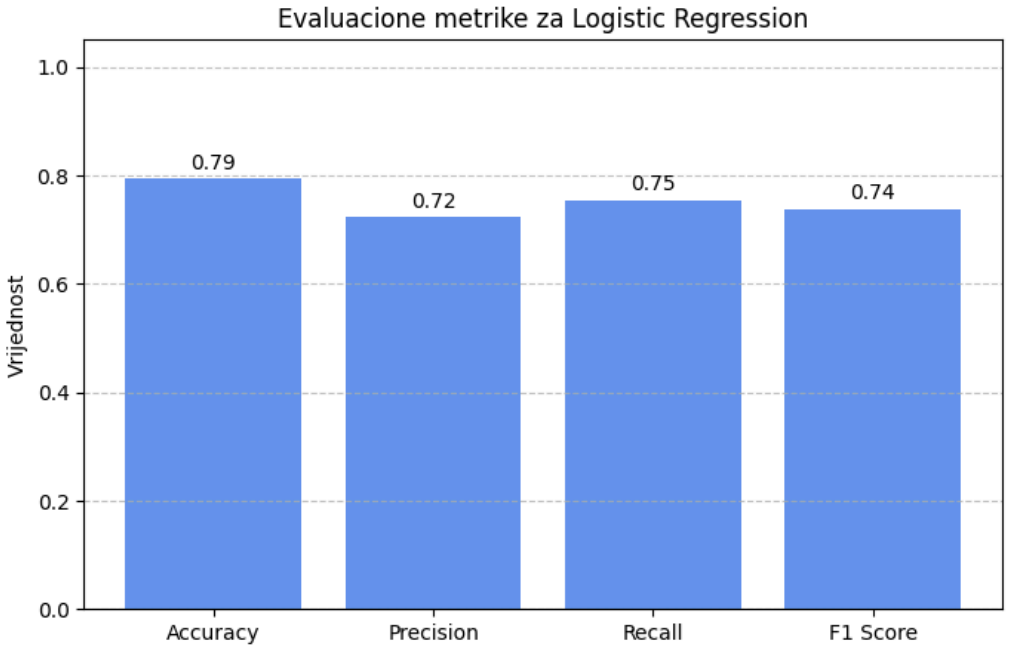
- Accuracy - Tačnost - ukupna preciznost, koliko ukupno predikcije je bilo tačno
- Precision - Preciznost - tačnost pozitivnih predikcija, koliko od onoga što je model označio kao pozitivno zaista jeste pozitivno
- Recall - Odziv - tačnost hvatanja pravih pozitivnih, koliko od stvarnih pozitivnih je model uspješno pronašao
- F1 Score - sredina preciznosti i odziva u jednoj mjeri
- Confusion Matrix - pokazuje koliko uzoraka iz svake stvarne klase je model svrstao u koju predviđenu klasu
- Classification Report - sve metrike po klasama

Za prikaz metrika upotrebljeni su chartovi, heatmap (Classification Report i Confusion Matrix).



Slika 5.1 Metrike za Random Forest

Slika iznad, predstavlja prikaz osnovnih metrika, odnosno evaluacije modela RandomForest-a, koji će poslije biti u uporedbi sa drugim modelom.



Slika 5.2 Metrike za Logističku Regresiju

Slika iznad predstavlja vizulani prikaz vrijednosti osnovnih metrika evaluacije za prethodno kreirani model Logističke Regresije.

Na osnovu prethodno prikazanih metrika možemo uporediti modele:

- RandomForest generalno bolji po svim metrikama, što je i očekivano jer se ipak radi o složenijem modelu
- Bolja ukupna tačnost kod RandomForest modela, razlika u 4% može biti značajna u praksi
- RandomForest ima bolju preciznost i recall, pa možemo reći da je balansiraniji u tačnim pozitivnim predikcijama i otkrivanju stvarnih pozitivnih slučajeva, odnosno ovaj model bolje pronalazi stvarne preživjele i daje manji broj lažno pozitivnih slučajeva

7. ZAKLJUČAK

U ovom projektu, cilj je bio razviti dva osnovna klasifikacijska modela, koji će rješavati problem predikcije preživljavanja na Titaniku. U ovom slučaju su izabrani algoritmi RandomForest i Logistic Regression, pri čemu je sam dataset već prethodno bio dostupan. Prije samog treniranja modela, izvršila se analiza podataka, te na koji način bi određeni podaci mogli da utiču na modele. Nakon analiziranja podataka, bilo je potrebno izvršiti pripremu istih kako bi se dobili podaci koje će sam model moći koristiti, ali i koji će biti i realni u odnosu na same izmjene u istim (pri tom se misli na dodavanje vrijednosti, uklanjanje kolona…). Upotrebom naprednijih metoda, modeli su dovedeni do određene granice tačnosti, što možemo vidjeti i na osnovu različitih metrika. Sam RandomForest u ovom slučaju ima prednost u odnosu na model Logističke Regresije, naravno, na osnovu samih metrika. Međutim, slobodno možemo reći da iako su modeli pokazali zadovoljavajuće i solidne performanse, još uvijek postoji prostor za buduća poboljšanja, kao što je dodavanje dodatnih varijabli (izdvajanje novih na osnovu postojećih npr kategorija dob), različito podešavanje hiperparametara. Također, konkretno za sam problem, moglo bi se razmotriti i upotreba drugih algoritama, kao i dobavljanje većeg dataseta.

LITERATURA

[1] IBM, “What is classification in machine learning?”, [Na internetu], (15.10.2024.), Dostupno: https://www.ibm.com/think/topics/classification-machine-learning, [Pristupano: 02.06.2025]

[2] Datacamp, “Classification in Machine Learning: An Introduction”, [Na internetu], (08.08.2024), Dostupno: https://www.datacamp.com/blog/classification-machine-learning, [Pristupano: 03.06.2025]

[3] Spam Klasifikacija, [Slika 4.1] (08.08.2024.), Dostupno: https://media.datacamp.com/legacy/image/upload/v1663850410/Machine_learning_classification_illustration_for_the_email_a993b8df37.png, [Pristupano: 03.06.2025]