Univerzitet u Zenici, Politehnički fakultet, Softversko inženjerstvo

Rad iz predmeta Poslovna inteligencija

Datum odbrane Februar, 2025

Predikcija ocjene filma na osnovu režisera

Ajla Brdarević[1], (autor), prof.dr. Esad Kadušić[2], (mentor), Asistent Emir Cogo[3], (mentor)
[1]Politehnički fakultet, Univerzitet u Zenici, Zenica, Bosna i Hercegovina (e-mail: ajla.brdarevic.20@size.ba)
[2]Politehnički fakultet, Univerzitet u Zenici, Zenica, Bosna i Hercegovina (e-mail: esad.kadusic@size.ba)
[3]Politehnički fakultet, Univerzitet u Zenici, Zenica, Bosna i Hercegovina (e-mail: emir.cogo@size.ba)

Odgovorni autor: Ajla Brdarević (e-mail: ajla.brdarevic.20@size.ba).

SAŽETAK Ovaj rad istražuje primjenu modela mašinskog učenja, konkretno Random Forest i Linear Regression, u svrhu predviđanja ocjena filmova unutar domena poslovne inteligencije. Korištenjem metapodataka filmova, uključujući žanrove, ocjene režisere, godinu produkcije i slično, razvijeni su modeli koji omogućavaju precizno predviđanje ocjena filmova na temelju ovih karakteristika. Analizirani podaci poslužili su za kreiranje alata koji omogućava procjenu kvalitete filma, što predstavlja značajnu primjenu poslovne inteligencije u filmskoj industriji. Evaluacija modela provedena je uz pomoć različitih metrike, uključujući Mean Squared Error (MSE), R2 Score, Mean Absolute Error (MAE) i Root Mean Squared Error (RMSE), koje su korištene za ocjenu tačnosti i preciznosti predviđanja. Ovi modeli omogućuju analizu velikih količina podataka, što je korisno za donosioce odluka u filmskoj industriji, jer im pomaže u procjeni potencijala filma još prije nego što postane dostupan široj publici. Glavni cilj istraživanja je pokazati kako poslovna inteligencija može koristiti analizu podataka i tehnike modeliranja za optimizaciju poslovnih odluka. Ovaj rad također istražuje izazove i mogućnosti u korištenju mašinskog učenja za predviđanje ocjena filmova, pružajući konkretne smjernice za buduće primjene u industriji zabave. Pokazuje razlike u rezultatima korištenjem linearne regresije i random forest regresije. Na kraju, istraživanje nudi potencijalne smjernice za dalji razvoj alata koji mogu unaprijediti poslovne procese u filmskoj industriji.

KLJUČNE RIJEČI Poslovna inteligencija, predviđanje ocjena filmova, Random Forest, Linear Regression, metapodaci filma, evaluacija modela.

I. UVOD

POSLOVNA INTELIGENCIJA (eng. Business Intelligence - BI) predstavlja skup tehnoloških procesa za prikupljanje, upravljanje i analizu organizacijskih podataka kako bi se dobili uvidi koji informišu poslovne strategije i operacije. [1] Zbog velike količine podataka i informacija koje imaju sve veći značaj u gotovo svim industrijama, te podatke je potrebno na pravilan način i sa jasnim ciljem analizirati. Korištenje raznih BI tehnika postalo je zbog toga neizostavno u industrijama kao što su marketing, zdravstvo ali i zabava. Prihodi na tržištu kina u svijetu, predviđaju se da će dostići 86,10 milijardi američkih dolara do 2025. godine, a penetracija korisnika će iznositi 22,4% u 2025. godini, te se predviđa da će porasti na 24,1% do 2029. godine. [2] Upravo ovo govori o značaju i zaradi filmske industrije, koja uz to zapošljava veliki broj ljudi, okuplja veliki broj gledalaca, te sa sobom nosi ogroman broj podataka.

U ovom radu istražena je primjena modela mašinskog učenja sa osnovnim ciljem koji BI donosi, za predviđanje ocjena filmova na temelju njihovih metapodataka. Cilj projekta je pokazati kako se tehnike poslovne inteligencije i analize podataka mogu iskoristiti za procjenu uspješnosti filmova na temelju raznih karakteristika. Korišteni su algoritmi linearne regresije i random forest regresije, čime se može ukazati na to koji od navedenih algoritama radi bolje sa datim podacima.

Provedeni rad uključuje prikupljanje, obradu i analizu podataka, razvoj i optimizaciju prediktivnih modela, te evaluaciju tih modela korištenjem odgovarajućih metrika. Rezultati pokazuju kako modeli poput Random Forest i Linear Regression mogu doprinijeti preciznijim procjenama, čime omogućavaju filmskoj industriji donošenje informisanijih odluka.

Ovaj rad pruža temelj za daljnju i detaljniju analizu

koja može biti od velikog značaja za filmsku industriju. Obzirom na kompleksnost i obim ove industrije, gdje se ulažu ogromni resursi i gdje veliki broj ljudi doprinosi realizaciji projekata, bolje razumijevanje faktora koji utiču na uspjeh filma može pomoći u optimizaciji ulaganja i postizanju većeg zadovoljstva publike. Dalje istraživanje na ovoj osnovi moglo bi uključivati složenije modele, širi obuhvat podataka i integraciju dodatnih izvora informacija kako bi se ostvarila što preciznija predviđanja i pružili dublji uvidi za strateško donošenje odluka.

A. SKRAĆENICE I AKRONIMI

Sljedeće skraćenice i akronimi se pojavljuju u ovom radu:

- **BI (Poslovna inteligencija)** – Tehnike i alati koji omogućavaju analizu podataka radi donošenja informisanih poslovnih odluka.
- **ML (Mašinsko učenje)** – Podskup vještačke inteligencije koji omogućava računarima učenje iz podataka i donošenje predikcija.
- **RMSE (Korijen srednje kvadratne greške)** – Metrička mjera za evaluaciju preciznosti modela.
- **R² (Koeficijent determinacije)** – Statistička mjera koja pokazuje koliko varijacije u zavisnoj varijabli objašnjava model.
- **RF (Random Forest)** – Algoritam mašinskog učenja baziran na ansamblu više odluka stabala za klasifikaciju i regresiju.

B. POSLOVNO PITANJE

U procesu razvoja projekta iz poslovne inteligencije, prvi korak ka identifikaciji i izradi rješenja uključuje definisanje poslovnog pitanja. Poslovno pitanje predstavlja osnovu na kojoj se gradi cijeli projekt jer usmjerava analizu podataka i oblikuje ciljeve koje projekt treba postići. U ovom projektu, poslovno pitanje je: **"Kako faktori poput ocjene režisera, broj ocjena filma, i žanr, utiču na predikciju ocjene filma?"** Pitanje je odabrano jer se filmovi ocjenjuju na osnovu različitih faktora, a cilj je istražiti u kojoj mjeri ovi faktori mogu predvidjeti konačnu ocjenu filma.

Ova analiza omogućava bolje razumijevanje povezanosti između različitih varijabli i ocjena filmova, čime se doprinosi razvoju prediktivnog modela koji može pomoći filmskim profesionalcima, režiserima, producentima, ali i ljubiteljima filma. Razmatranje ovog poslovnog pitanja pruža osnovu za prikupljanje podataka, njihovu analizu i razvoj modela za predviđanje ocjena filmova.

Dodatne implementacije, uzimanje u obzir novih varijabli, kao i upoređivanje rezultata s različitim modelima predikcije, omogućavaju dalja poboljšanja ovog pristupa. Ovaj temelj može poslužiti za istraživanje drugih poslovnih pitanja vezanih uz filmsku industriju, poput predikcije uspjeha filma na osnovu drugih faktora, kao što su glumačka postava ili datum premijere.

Slika 1: Prikaz baze podataka imdb$_{full}$

C. DATASET

Korišten je set podataka IMDB [3] koji sadrži brojne podatke o filmovima, glumcima, režiserima, ocjenama, te mnogim drugim podacima vezanim za filmove. Dostupno je nekoliko različitih baza podataka, sa više ili manje podataka, tabela ili kolona. Za potrebe odgovora na već spomenuto poslovno pitanje, sa ciljem predikcije ocjene filma, odabrana baza podataka je `imdb_full`, a tabele sa potrebnim podacima su sljedeće:

- `directors` (directorid, name, rate, gross, num)
- `movies` (movieid, title, year)
- `movies2directors` (movieid, directorid, genre)
- `ratings` (movieid, rank, votes, distribution)

Veze između tabela su:

- movies (movieid) povezana sa movies2directors (movieid)
- directors (directorid) povezana sa movies2directors (directorid)
- movies (movieid) povezana sa ratings (movieid)

Navedene veze su prikazane na slici 1.

D. KORIŠTENI ALGORITMI I MJERE

U radu je korišteno nekoliko različitih algoritama ML-a i mjera koje pokazuju tačnost prediktivnog modela:

- **Linearna regresija**: Linearna regresija je statistička tehnika koja se koristi za pronalaženje odnosa između varijabli. U kontekstu ML, linearna regresija je algoritam koji pronalazi odnos između karakteristika (feature) i oznake (label). [4] Slobodno rečeno, features su karakteristike koje opisuju film,

dok su label-i ocjena filma. Linearna regresija se koristi za predviđanje ocjena na osnovu raznih features-a, pomažući u identifikaciji značaja svakog faktora i njegovog doprinosa ukupnoj ocjeni filma.

- **Random forest regression**: Random Forest regresija je ML tehnika za predviđanje numeričkih vrijednosti. Kombinuje predikcije više stabala odluke kako bi smanjila prekomjerno prilagođavanje (overfitting) i poboljšala tačnost. [5] U kontekstu ovog rada, random forest algoritam je korišten za predikciju ocjena filmova na osnovu karakteristika kao što su režiser, žanr i sl. Ova metoda omogućava bolju generalizaciju rezultata i pruža robusnije predikcije.
- **MSE (Mean Squared Error)**: MSE je ključna metrike za ocjenu performansi prediktivnih modela. Mjeri prosječnu kvadratnu razliku između predviđenih i stvarnih ciljanih vrijednosti. Manja MSE vrijednost ukazuje na to da model bolje predviđa. [6]
- **R^2 (R-squared)**: R-squared je statistička mjera koja procjenjuje koliko dobro model odgovara podacima. Vrijednosti R^2 se kreću od 0 do 1, gdje 1 označava skoro savršeno podudaranje modela sa podacima. U ovom projektu, R^2 je korišten za validaciju i evaluaciju rezultata za predikcije sa random forest i linearnom regresijom. [7]
- **MAE (Mean Absolute Error)**: MAE mjeri prosječnu apsolutnu razliku između predviđene i stvarne vrijednosti. Koristi se kao mjera tačnosti prediktivnih modela, pomažući u procjeni prosječne razlike između stvarnih ocjena filmova i onih predviđenih modelom. [8]
- **RMSE (Root Mean Square Error)**: RMSE je mjera koja predstavlja korijen srednje kvadratne greške između predviđenih i stvarnih vrijednosti. Niža vrijednost RMSE-a ukazuje na bolju preciznost modela. [9]
- **Grid Search**: Grid Search je algoritam koji se koristi u ML-u za podešavanje hiperparametara. Sistematski ispituje svaku moguću kombinaciju ponuđenih vrijednosti hiperparametara kako bi pronašao najbolji model. [10]

E. ALATI

Korišten je skup tehnologija i alata koji su omogućili analizu, čišćenje podataka, transformaciju u potrebni tip podatka, te na kraju predikciju i validaciju rezultata uz prezentaciju.

1) DBeaver

Besplatan i open-source alat za manipulaciju podacima, eksportovanje podataka u potrebnim formatima, analizu podataka, te obradu podataka. Podržava više od osamdeset baza podataka [11]. Ovaj alat je dostupan za preuzimanje na https://dbeaver.io [12]. U kontekstu ovog rada korišten je za spajanje na IMDB bazu podataka putem Maria DB, eksportovanje tabela u CSV formatu i osnovnu analizu podataka.

2) Visual Paradigm

Cross-platform alat namijenjen menadžmentu IT sistema [13]. Koristi se za stvaranje ERD dijagrama i vizualno predstavljanje relacione sheme između tabela. Ovaj alat je pomogao pri daljnjoj analizi i razvijanju poslovnog pitanja. Dostupan je za preuzimanje na https://www.visual-paradigm.com/download/ [14].

3) Visual Studio Code

Popularno okruženje koje podržava rad sa ML i BI projektima [15]. Korišten je za finalno čišćenje, transformaciju podataka, predikciju i validaciju rezultata pomoću Python biblioteka. Dostupan je na https://code.visualstudio.com/Download [16].

4) Python

: Verzija 13.2.0 korištena za analizu, čišćenje, transformaciju podataka i predikciju. Korištene biblioteke uključuju:

- **pandas**: Biblioteka za rad sa skupovima podataka, analiza, čišćenje i manipulacija podacima [17].
- **numpy**: Biblioteka za rad s nizovima i linearnom algebrom [18].
- **seaborn**: Biblioteka za vizualizaciju podataka koja koristi matplotlib [19].
- **matplotlib**: Biblioteka za kreiranje statičkih, animiranih i interaktivnih vizualizacija [20].
  - **pyplot**: Koristi se za interaktivne dijagrame i generiranje dijagrama [21].
- **sklearn - scikit-learn**: Pruža algoritme za mašinsko učenje, uključujući metode za odabir modela i optimizaciju hiperparametara [22].
  - **model_selection**: Alati za odabir modela, unakrsnu validaciju i podešavanje hiperparametara [23].
    - **train_test_split**: Metoda za podjelu podataka u skupove za obuku i testiranje [24].
    - **gridSearchCV**: Optimizacija parametara estimatora putem unakrsne validacije [25].
  - **ensemble**: Metode za klasifikaciju, regresiju i detekciju anomalija [26].
    - **randomForestRegressor**: Random forest je meta-estimator koji prilagođava više regresora baziranih na stablu odluke na različitim podskupovima podataka i koristi prosjek za poboljšanje tačnosti predikcije i kontrolu prekomjernog prilagođavanja (over-fittinga). [27]
  - **linear_model**: Biblioteka za različite linearne modele [28]
    - **LinearRegression**: LinearRegression prilagođava linearni model s koeficijentima w

= (w1, . . . , wp) kako bi minimizirao zbir kvadrata ostataka između opaženih ciljeva u skupu podataka i ciljeva predviđenih linearnom aproksimacijom. [29]

- **preprocessing**: Metode za skaliranje, centriranje, normalizaciju, binarizaciju i druge tehnike obrade podataka. [30]
  - **standardScaler**: Standardizacija karakteristika uklanjanjem srednje vrijednosti i skaliranjem na jediničnu varijansu. [31]
- **metrics**: Funkcije ocjene, metričke performanse, metričke razlike parova i izračuni udaljenosti. [32]
  - **mean_squared_error**: Gubitak regresije zasnovan na srednjoj kvadratnoj pogrešci. [33]
  - **R2_score**: Funkcija ocjene regresije (koeficijent determinacije). [34]
  - **mean_absolute_error**: Regresijski gubitak srednje apsolutne greške. [35]
- **pipeline**: Alati za izgradnju kompozitnog estimatora kao lanca transformacija i estimatora. [36]
- **impute**: Transformatori za imputaciju nedostajućih vrijednosti. [37]
  - **simpleImputer**: Zamjena nedostajućih vrijednosti koristeći deskriptivnu statistiku (npr. srednja vrijednost, medijan ili najčešća vrijednost) za svaku kolonu, ili koristeći konstantnu vrijednost. [38]

5) Jupyter Notebook

: Jupyter (ranije poznat kao IPython Notebook) je open-source projekt koji omogućava jednostavno kombiniranje Markdown teksta i izvršivog Python koda na jednoj platformi koja se naziva notebook. Lako se implementira u Visaul Studio Code-u. [39] Korišten je zbog jednostavnosti kombinovanja koda te popratnih detaljnih objašnjenja i komentara koda. Također, pruža lakši i ljepši uvid šta svaki blok koda radi, šta ispisuje i slično.

II. PROCES IZRADE PROJEKTA

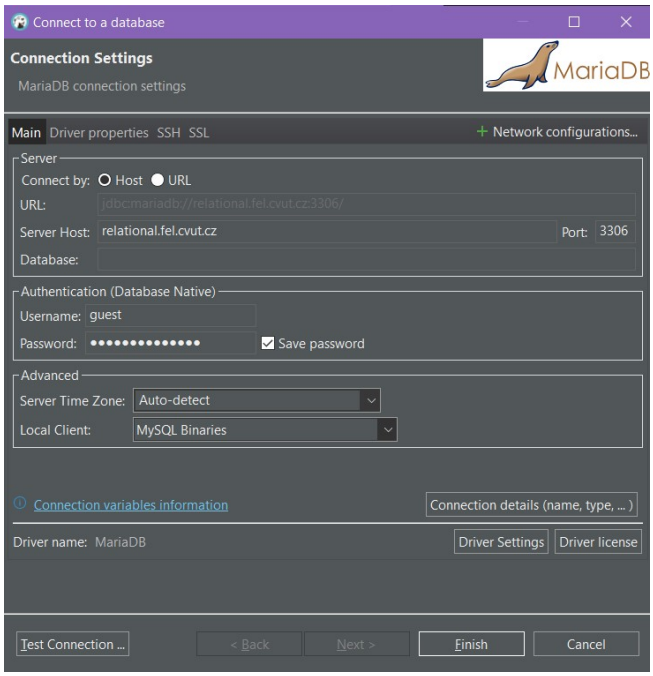
A. PREUZIMANJE DATASETA

Nakon pristupa IMDB [3], datasetu se pristupilo korištenjem DBeaver [11]. Pristupilo se spajanjem na novu MariaDB client konekciju u DBeaver okruženju. Na dataset se spojilo unošenjem sljedećih podataka koji su dostupni na IMDB [3]:

1) hostname: relational.fel.cvut.cz
2) port: 3306
3) username: guest
4) password: ctu-relational

Što je prikazano na slici 2.

U Database Navigator u DBeaver učitalo se nekoliko različitih baza podataka kao već spomenute



Slika 2: Konekcija na bazu podataka u DBeaver

Slika 3: Lista dostupnih baza podataka

midb_MovieLens, imdb_full, imdb_jis, imdb_small, te se odatle može pristupiti svim bazama podataka te njiovim tabelama. U ovom radu je korištena imdb_full baza podataka sa tabelama:

- directors (directorid, name, rate, gross, num)
- movies (movieid, title, year)
- movies2directors (movieid, directorid, genre)
- ratings (movieid, rank, votes, distribution)

Lista dostupnih baza se može vidjeti na slici q3

Tabele su preuzete tako što se odabrana tabela (u primjeru na slikama 4 i 5 korištena tabela directors), exportovala u CSV sa Export data.

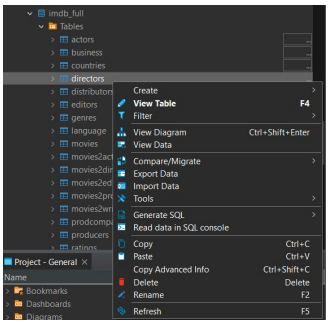
Zbog jednostavnosti rada sa CSV file-ovima u Python-u, podaci su spremljeni u CSV formatu, na primjeru na slici je prikazano za directors tabelu, no isti je proces i za sve ostale tabele.

Nakon odabira formata, CSV file se sprema na disk to jeste stvara se lokalna kopija podataka odabrane tabele.

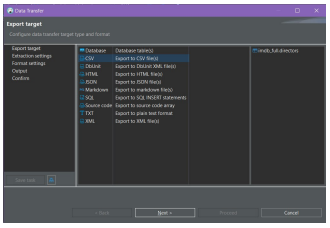
B. ANALIZA DOSTUPNIH PODATAKA

Pomoću DBeaver moguće je donekle analizirati podatke, ali s obzirom da se radi o realtivno velikom skupu podataka, mogao se jasno dobiti uvid samo u tipove podataka u kolonama tabela. Pa je tako:

- **Directors**:



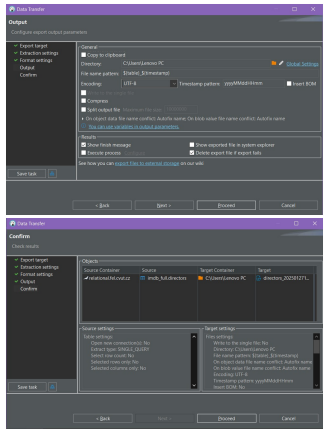
Slika 4: Exportovanje CSV file-a za directors tabelu



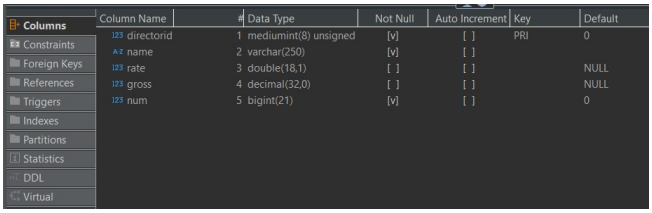
Slika 5: Exportovanje CSV file-a za directors tabelu

- Kolone:
  - Directorid (mediumint(8) unsigned)
  - Name (varchar(250))
  - Rate (double(18,1))
  - Gross (decimal(32,0))
  - Num (bigint(21))
- Constraints:
  - Directorid je primarni ključ
- References:
  - movies2directors

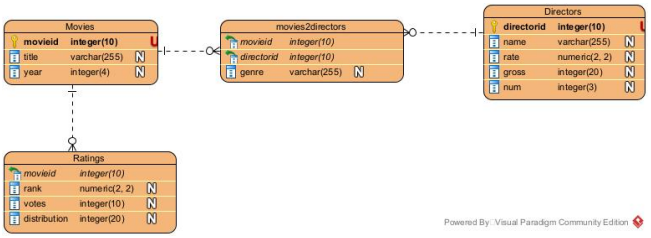
- **Movies**:
  - Kolone:
    - Movieid (mediumint(8) unsigned)
    - Title (varchar(400))
    - Year (varchar(100))



Slika 6: Exportovanje CSV file-a za directors tabelu



Slika 7: Tipovi podataka u tabeli directors



Slika 8: ERD dijagram imdb_full baze sa potrebnim tabelama

  - Constraints:
    - Movieid je primarni ključ
  - References:
    - movies2directors

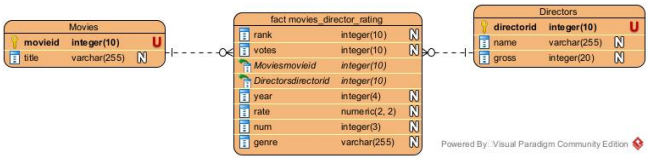
- **Movies2directors**:
  - Kolone:
    - Movieid (mediumint(8) unsigned)
    - Directorid (mediumint(8) unsigned)
    - Genre (varchar(15))
  - Constraints:
    - Movieid je primarni ključ
    - Directorid je primarni ključ
  - Foreign keys:
    - Movieid je strani ključ
    - Directorid je strani ključ

- **Ratings**:
  - Kolone:
    - Movieid (mediumint(8) unsigned)
    - Rank (char(4))
    - Votes (mediumint(8) unsigned)
  - Constraints:
    - Movieid je primarni ključ
  - Foreign keys:
    - References: movies2directors

Na slici 7 se može vidjeti primjer za tabelu directors, kako se mogu analizirati tipovi podataka.

C. ERD DIJAGRAM I STAR SHEMA

Kako bi se vizuelno jasnije prikazala veza između tabela, koristi se ERD Diagram u Visual Paradigm. To jeste, na slici 8 je prikazana relaciona shema tabela koje se koriste.



Slika 9: Star shema imdb_full baze sa potrebnim tabelama

Kako bi poslovno pitanje te cilj projekta bio jasniji, na slici 9 je prikazana star shema.

D. PRIPREMA ZA ČIŠĆENJE PODATAKA

Prije samog čišćenja podataka, potrebno je pozvati sve već navedene biblioteke.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import
    train_test_split, GridSearchCV
from sklearn.ensemble import
    RandomForestRegressor
from sklearn.linear_model import
    LinearRegression
from sklearn.preprocessing import
    StandardScaler
from sklearn.metrics import mean_squared_error,
     r2_score, mean_absolute_error
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
```

Kod 1: Učitavanje biblioteka

A nakon toga i učitati podatke, tj CSV file-ove koji su preuzeti kako je već navedeno.

```
directors = pd.read_csv('data/
    directors_202501211825.csv')
movies = pd.read_csv('data/movies_202501211826.
    csv')
movies2directors = pd.read_csv('data/
    movies2directors_202501211826.csv')
ratings = pd.read_csv('data/
    ratings_202501211826.csv')
```

Kod 2: Učitavanje podataka

E. ČIŠĆENJE PODATAKA

Kako bi podaci bili pogodni za obradu, potrebno je ili zamijeniti podatke koji fale (null), ili obrisati redove u tabelama gdje se takve vrijednosti pojavljuju. Kako se u projektu koristi Python, finalno čišćenje podataka se vršilo u Pythonu na sljedeći način.

1) Provjera i uklanjanje nedostajućih vrijednosti

U kodu 3 se uklanjaju sve redove sa nedostajućim podacima (NaN vrijednostima) iz četiri DataFrame-a: directors, movies, movies2directors i ratings, čime se osigurava da podaci budu kompletni za daljnju analizu.

```
directors.dropna(inplace=True)
movies.dropna(inplace=True)
movies2directors.dropna(inplace=True)
ratings.dropna(inplace=True)
```

Kod 3: Provjera i uklanjanje nedostajućih vrijednosti

`directors.dropna(inplace=True)`: Ova naredba uklanja sve redove iz DataFrame-a `directors` koji sadrže nedostajuće vrijednosti (NaN) u bilo kojem od svojih kolona. Parametar `inplace=True` znači da će promjena biti izvršena direktno na postojećem DataFrame-u, bez potrebe za stvaranjem novog objekta.

`movies.dropna(inplace=True)`: Ovdje se uklanjaju redovi koji sadrže nedostajuće vrijednosti iz DataFrame-a `movies`. Kao i prethodno, `inplace=True` osigurava da promjene budu izvršene na originalnom DataFrame-u.

`movies2directors.dropna(inplace=True)`: Ova naredba uklanja redove koji imaju nedostajuće vrijednosti iz DataFrame-a `movies2directors`, koji povezuje filmove s režiserima. Ovdje također koristi `inplace=True` za direktnu promjenu.

`ratings.dropna(inplace=True)`: Na kraju, redovi koji sadrže nedostajuće vrijednosti u DataFrame-u `ratings` se uklanjaju. Ovaj DataFrame sadrži ocjene filmova, a uklanjanjem redova s NaN vrijednostima osigurava da se analiziraju samo potpuni podaci.

2) Transformacija podataka u odgovarajući oblik

Finalna transformacija podataka se također vršila u Pythonu. Prvo se vrši konvertovanje podataka u odgovarajuće tipove podataka, što se može vidjeti u kodu 4. S obzirom da će se u nastavku koristiti linearna regresija i random forest regresija, potrebni je da podaci budu predstvaljeni kao numerički podaci, a ne tekstualni kako su podaci bili u orginalnom dataset-u.

```
ratings['rank'] = pd.to_numeric(ratings['rank
    '], errors='coerce')
ratings['votes'] = pd.to_numeric(ratings['votes
    '], errors='coerce')
ratings.dropna(subset=['rank', 'votes'],
    inplace=True)
```

Kod 4: Transformacija podataka u odgovarajući oblik

`ratings['rank'] = pd.to_numeric(ratings['rank'], errors='coerce')`: Ova naredba pokušava pretvoriti vrijednosti u koloni `rank` DataFrame-a `ratings` u numerički tip podataka. Ako postoji neka vrijednost koja ne može biti konvertirana u broj, umjesto nje bit će postavljena kao `NaN` zbog opcije `errors='coerce'`.

`ratings['votes'] = pd.to_numeric(ratings['votes'], errors='coerce')`: Slično kao prethodna naredba, ova linija konvertira vrijednosti u koloni `votes` u numerički tip podataka. Ako neka vrijednost nije brojčana, bit će postavljena kao `NaN`.

`ratings.dropna(subset=['rank', 'votes'], inplace=True)`: Nakon što su vrijednosti u kolonama `rank` i `votes` pretvorene u numeričke, ova naredba uklanja sve redove koji imaju `NaN` vrijednosti u bilo kojem od te dvije kolone. Parametar `subset=['rank', 'votes']` osigurava da

se uklone samo redovi koji imaju nedostajuće vrijednosti u tim specifičnim kolonama.

Drugi korak transformacije se oslanja na star shemu napravljenu u Visual Paradigm. Potrebno je spojiti tabele u jedan skup podataka - činjeničnu tabelu kao na kodu 5

```
movies_with_directors = pd.merge(
    movies2directors, directors[['directorid',
    'rate', 'num']], on='directorid', how='
    inner')
data = pd.merge(movies_with_directors, ratings
    [['movieid', 'rank', 'votes']], on='movieid
    ', how='inner')
data = pd.merge(data, movies[['movieid', 'year
    ']], on='movieid', how='inner')
```

Kod 5: Spajanje tabela

`movies_with_directors = pd.merge(movies2directors, directors[['directorid', 'rate', 'num']], on='directorid', how='inner')`: Ova naredba spaja DataFrame `movies2directors` s DataFrame-om `directors`, koristeći zajedničku kolonu `directorid`. Iz `directors` DataFrame-a zadržavaju se samo kolone `directorid`, `rate`, i `num`. Tip spajanja je `inner`, što znači da će se zadržati samo redovi koji imaju odgovarajući tj. isti `directorid` u oba DataFrame-a.

`data = pd.merge(movies_with_directors, ratings[['movieid', 'rank', 'votes']], on='movieid', how='inner')`: U ovoj naredbi, spajaju se prethodno dobijeni DataFrame `movies_with_directors` s DataFrame-om `ratings`. Koristi se zajednička kolona `movieid`, i zadržavaju se samo stupci `movieid`, `rank` i `votes` iz `ratings` DataFrame-a. Opet, se koristi `inner` spajanje, što znači da će se zadržati samo oni filmovi koji postoje u oba DataFrame-a.

`data = pd.merge(data, movies[['movieid', 'year']], on='movieid', how='inner')`: Na kraju, spajaju se DataFrame `data` s DataFrame-om `movies`, koristeći zajedničku kolonu `movieid`. Zadržavaju se samo stupci `movieid` i `year` iz `movies`. Ponovo se koristi `inner` spajanje kako bi zadržali samo one filmove koji su prisutni u oba DataFrame-a.

Kako je navedeno metode korištene za predikciju su algoritmi ML-a linearna regresija i random forest regresija. Prije prediktovanja, potrebno je odabrati relevantne kolone za analizu. Ovo se ujedno odnosi i na poslovno pitanje, to jeste uzima se u obzir koji feature-i i label-i su direktno ili indirektno navedeni poslovnim pitanjem.

```
data = data[['rate', 'num', 'votes', 'year', '
    genre', 'rank']]  # 'rank' je ciljana
    varijabla
```

Kod 6: Odabir feature-a i label-a

`data = data[['rate', 'num', 'votes', 'year', 'genre', 'rank']]`: Ova linija koda bira određene kolone iz DataFrame-a `data`. Odabrane kolone su: `rate`, `num`, `votes`, `year`, `genre` i `rank`. Kolona `rank` označava ciljnu varijablu koju želimo predvidjeti u modeliranju, dok ostale kolone predstavljaju karakteristike koje će se koristiti za predviđanje.

Nakon odabira relevantnih kolona, potrebno je pretvoriti i kategorijske kolone u numeričlke.

```
data = pd.get_dummies(data, columns=['genre'],
    drop_first=True)
```

Kod 7: Encoding za kategorijske podatke

`data = pd.get_dummies(data, columns=['genre'], drop_first=True)`: Ova linija koda koristi funkciju `pd.get_dummies` za pretvaranje kategorijske kolone `genre` u numeričke varijable pomoću one-hot encoding tehnike. Parametar `drop_first=True` osigurava da se ukloni prva kategorija u koloni `genre` kako bi se izbjegla multikollinearnost, što je važan korak u pripremi podataka za modele koji nisu otporni na višestruku korelaciju.

3) Korištenje metoda predikcije za generisanje rezultata

Prije početka korištenja metoda za predikcije, a nakon čišćenja i transformisanja podataka, potrebno je definisati ulazne i izlazne podatke.

```
X = data.drop(['rank'], axis=1)  # Ulazne
    varijable
y = data['rank']  # Ciljna varijabla
```

Kod 8: Ulazne varijable i ciljnja varijabla

`X = data.drop(['rank'], axis=1)`: Ova linija koda koristi metodu `drop` da bi uklonila kolonu `rank` iz skupa podataka i tako stvorila ulazne varijable (`X`). Ulazne varijable su sve varijable osim ciljne varijable. Parametar `axis=1` označava da se uklanja kolonu, a ne red.

`y = data['rank']`: Ovdje se ciljana varijabla (`rank`), koja predstavlja varijablu koju želimo predvidjeti, izolira kao `y`. Ova varijabla će biti predviđena od strane modela.

Nakon toga potrebno je izvršiti standardizaciju podataka.

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

Kod 9: Standardizacija podataka

`scaler = StandardScaler()`: Ovdje se kreira objekat `scaler` koji je instanca klase `StandardScaler` iz biblioteke `scikit-learn`. Ova klasa koristi metodu standardizacije koja centrirano transformiše podatke, čineći njihov srednji vrijednost 0 i standardnu devijaciju 1.

`X_scaled = scaler.fit_transform(X)`: Ova linija koda primjenjuje standardizaciju na ulazne varijable `X` pomoću metode `fit_transform`. Prvo se izračunaju parametri (srednja vrijednost i standardna devijacija) za svaku varijablu, a zatim se podaci transformiraju prema tim parametrima. Rezultat ove operacije je novi skup podataka `X_scaled` sa standardiziranim vrijednostima.

Prije početka treniranja modela, potrebno je dostupne podatke podijeliti na podatke koji će se koristiti za treniranje te podatke koji će se koristiti za testiranje.

```
X_train, X_test, y_train, y_test =
    train_test_split(X_scaled, y, test_size
    =0.2, random_state=42)
```

Kod 10: Podjela na trening i test skupove

`X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)`: Ovdje se koristi funkcija `train_test_split` iz `scikit-learn` biblioteke kako bi se podaci podijelili na trening i test skup.

`X_scaled` sadrži standardizirane ulazne varijable, dok `y` sadrži ciljne varijable (ocjene). Parametar `test_size=0.2` znači da će 20`random_state=42` osigurava da podjela bude reproduktivna (da se uvijek dobije ista podjela podataka kad god se kod pokrene sa istim parametrom). Rezultat ove funkcije su četiri varijable: `X_train`: Trening skup ulaznih varijabli. `X_test`: Test skup ulaznih varijabli. `y_train`: Trening skup ciljanih varijabli. `y_test`: Test skup ciljanih varijabli.

Kreiranje pipeline-a za modele u mašinskom učenju je vrlo važan korak, jer omogućava efikasno i organizovano upravljanje svim fazama procesa modeliranja, od preprocesiranja podataka do treniranja i evaluacije modela. Pipeline omogućava da se sve ove faze povežu u jedan tok rada, čineći cijeli proces ponovljivim i lakšim za održavanje.

```
rf_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='mean'))
    ,  # Zamjena nedostaju ih podataka sa
    srednjom vrijedno  u
    ('regressor', RandomForestRegressor(
    random_state=42))
])
```

Kod 11: Random Forest model u Pipeline-u

`rf_pipeline = Pipeline([ ... ])`: Ovdje se koristi `Pipeline` iz `scikit-learn` biblioteke kako bi se stvorio tok rada za modeliranje.

`Pipeline` omogućava da se više koraka (poput obrade podataka i treniranja modela) spoje u jedan objekt, što olakšava rad i implementaciju. U ovom pipeline-u su dva koraka:

`('imputer', SimpleImputer(strategy='mean'))`:

Ovaj korak koristi `SimpleImputer` za imputaciju (zamjenu) nedostajućih podataka u skupu podataka. Postavka `strategy='mean'` znači da će nedostajući podaci biti zamijenjeni srednjom vrijednošću odgovarajuće kolone. `('regressor', RandomForestRegressor(random_state=42))`:

Ovdje se koristi `RandomForestRegressor` kao model za regresiju. Parametar `random_state=42` osigurava da se model uvijek trenira na isti način sa istim početnim uvjetima (reproduktivnost rezultata).

Slično se radi i za linearnu regresiju.

```
lr_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='mean'))
    ,  # Zamjena nedostaju ih podataka sa
    srednjom vrijedno  u
    ('regressor', LinearRegression())
])
```

Kod 12: Linearna regresija model u Pipeline-u

`lr_pipeline = Pipeline([ ... ])`: Ovdje se također koristi `Pipeline` iz `scikit-learn` biblioteke za organiziranje toka rada koji uključuje korake obrade podataka i treniranja modela.

Pipeline se sastoji od dva koraka:

`('imputer', SimpleImputer(strategy='mean'))`: Ovaj korak koristi `SimpleImputer` za imputaciju nedostajućih podataka. Postavka `strategy='mean'` znači da će svi nedostajući podaci biti zamijenjeni srednjom vrijednošću odgovarajuće kolone, čime se osigurava da model ne bude ometan nedostajućim vrijednostima. `('regressor', LinearRegression())`:

Ovdje se koristi `LinearRegression` kao model za regresiju. Linearna regresija je metoda kojom se pronalazi najbolja linija koja minimizira grešku između stvarnih i predviđenih vrijednosti ciljne varijable.

Nakon svih ovih koraka je moguće početi sa treniranjem modela.

```
rf_pipeline.fit(X_train, y_train)
```

Kod 13: Treniranje modela - Random Forest

`rf_pipeline.fit(X_train, y_train)`: Ovdje se poziva `fit` metoda na `rf_pipeline` objektu. Ova metoda trenira `Random Forest` model unutar pipeline-a koristeći trening podatke:

- `X_train`: Ulazne varijable (atributi) za trening modela.
- `y_train`: Ciljna varijabla (predviđene vrijednosti) za trening modela.

Pipeline prvo izvršava imputaciju nedostajućih podataka, a zatim trenira `Random Forest Regressor` koristeći trenirane podatke.

A zatim je moguće testirati predikcije na skupu test podataka.

```
y_pred_rf = rf_pipeline.predict(X_test)
```

Kod 14: Predikcija na test skupu

`y_pred_rf = rf_pipeline.predict(X_test)`: Ovdje se koristi trenirani `rf_pipeline` za predviđanje ciljne varijable na testnim podacima.

- `X_test`: Ulazne varijable (atributi) za testne podatke.
- `y_pred_rf`: Predviđene vrijednosti ciljne varijable za testne podatke, koje generira `Random Forest` model.

Pipeline primjenjuje iste korake kao tijekom treniranja (imputacija nedostajućih podataka) i koristi trenirani model za predviđanje na testnim podacima.

Već spomenute tehnike za evaluaciju modela sada tek mogu pokazati da li je model dobro prediktovao, to jeste da li je dobro istreniran.

```
print("\n=== Evaluacija Random Forest modela
    ===")
print(f"Mean Squared Error: {mean_squared_error
    (y_test, y_pred_rf):.2f}")
print(f"R2 Score: {r2_score(y_test, y_pred_rf)
    :.2f}")
print(f"Mean Absolute Error: {
    mean_absolute_error(y_test, y_pred_rf):.2f
    }")
```

```
print(f"Root Mean Squared Error: {np.sqrt(
    mean_squared_error(y_test, y_pred_rf)):.2f
    }")
```

Kod 15: Evaluacija modela

`mean_squared_error(y_test, y_pred_rf)`: Izračunava srednju kvadratnu grešku (MSE), koja mjeri prosječnu kvadratnu razliku između stvarnih i predviđenih vrijednosti.

`r2_score(y_test, y_pred_rf)`: Izračunava koeficijent determinacije (R^2), koji daje mjeru koliko dobro model objašnjava varijaciju u podacima. Vrijednosti bliske 1 označavaju dobar model.

`mean_absolute_error(y_test, y_pred_rf)`: Izračunava srednju apsolutnu grešku (MAE), koja mjeri prosječnu apsolutnu razliku između stvarnih i predviđenih vrijednosti.

`np.sqrt(mean_squared_error(y_test, y_pred_rf))`: Izračunava korijen srednje kvadratne greške (RMSE), koji daje mjeru prosječne greške u istim jedinicama kao i ulazne varijable.

Ove metrike pomažu u procjeni performansi modela, gdje manji brojevi znače bolji model.

4) Prezentacija rezultata za Random Forest Regression

Ovaj graf prikazuje odnos između stvarnih (‘y_test‘) i predviđenih (‘y_pred_rf‘) ocjena korištenjem Random Forest modela.

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x=y_test, y=y_pred_rf, color='
    blue', edgecolor='black', alpha=0.6, s=100)
plt.plot([y_test.min(), y_test.max()], [y_test.
    min(), y_test.max()], color='red',
    linestyle='--', label="Savr eno
    predvi anje")
plt.xlabel("Stvarne ocjene", fontsize=14)
plt.ylabel("Predvi ene ocjene", fontsize=14)
plt.title("Random Forest: Stvarne vs.
    Predvi ene ocjene", fontsize=16)
plt.grid(True)
plt.legend()
plt.show()
```

Kod 16: Evaluacija modela

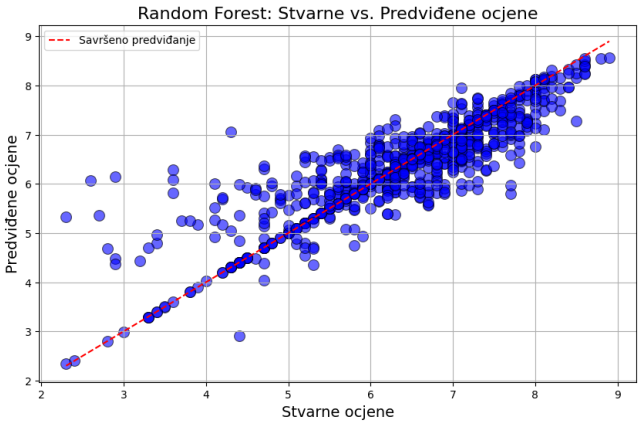
`sns.scatterplot()`: Na grafu su prikazane stvarne i predviđene ocjene kao tačke, s plavom bojom za tačke i crnim ivicama. Parametar `alpha=0.6` omogućava prozirnost tačaka, dok `s=100` postavlja veličinu tačaka.

Crvena linija: Linija označena kao "Savršeno predviđanje" (crvena, isprekidana) prikazuje idealan slučaj kada su predviđene i stvarne ocjene iste. Ova linija se koristi kao referenca za usporedbu tačaka scatter plota.

`plt.xlabel()` **i** `plt.ylabel()`: Ove funkcije postavljaju oznake za osovine, s fontovima veličine 14 za bolju čitljivost.

`plt.title()`: Naslov grafa postavljen je s fontom veličine 16, koji jasno označava da se graf odnosi na poređenje stvarnih i predviđenih ocjena s Random Forest modelom.

`plt.grid(True)`: Dodana mreža na grafu za lakše praćenje odnosa između tačaka.



Slika 10: Graf random forest stvartne vs predviđene ocjene

`plt.legend()`: Dodana legenda za označavanje crvene linije koja prikazuje savršeno predviđanje.

Graf na slici 10 omogućava vizualnu procjenu tačnosti predviđanja modela i pokazuje koliko su predviđene ocjene blizu stvarnim.

5) Validacija rezultata - poređenje sa Linear Regression modelom

Ovdje se evaluira izvedba Linear Regression modela koristeći razne metrike:

- **Mean Squared Error (MSE)**: Mjera prosječne kvadratne greške između stvarnih i predviđenih ocjena. Niža vrijednost ukazuje na bolju tačnost modela.
- **R2 Score**: Mjera koliko varijacije stvarnih podataka model može objasniti. Vrijednost 1 označava savršeno objašnjenje podataka, dok negativne vrijednosti označavaju loše predviđanje.
- **Mean Absolute Error (MAE)**: Prosječna apsolutna greška između stvarnih i predviđenih ocjena. Manje vrijednosti ukazuju na bolju preciznost modela.
- **Root Mean Squared Error (RMSE)**: Korijen iz MSE, koji daje uvid u prosječnu grešku u istim jedinicama kao i originalni podaci.

Ove metrike pomažu u razumijevanju tačnosti i performansi modela Linear Regression u predviđanju ocjena.

```
lr_pipeline.fit(X_train, y_train)
y_pred_lr = lr_pipeline.predict(X_test)
print("\n=== Evaluacija Linear Regression
    modela ===")
print(f"Mean Squared Error: {mean_squared_error
    (y_test, y_pred_lr):.2f}")
print(f"R2 Score: {r2_score(y_test, y_pred_lr)
    :.2f}")
print(f"Mean Absolute Error: {
    mean_absolute_error(y_test, y_pred_lr):.2f
    }")
print(f"Root Mean Squared Error: {np.sqrt(
    mean_squared_error(y_test, y_pred_lr)):.2f
```

```
    }")
```

Kod 17: Poređenje sa Linear Regression modelom

`lr_pipeline.fit(X_train, y_train)`: Model se trenira na skupu podataka za trening (`X_train` i `y_train`).

`y_pred_lr = lr_pipeline.predict(X_test)`: Predviđanja se generiraju za testni skup podataka (`X_test`).

Ispis koda 17:

Evaluacija Linear Regression modela
Mean Squared Error: 0.69
R2 Score: 0.50
Mean Absolute Error: 0.63
Root Mean Squared Error: 0.83

6) Vizualizacija poređenje

Ovaj graf prikazuje usporedbu distribucije stvarnih ocjena i predviđenih ocjena kroz dva modela:

- **Stvarne ocjene - plava linija** predstavljaju raspodjelu stvarnih ocjena u testnom skupu podataka (‘y_test‘).
- **Predviđene ocjene (Random Forest) - narandžasta linija** predstavljaju raspodjelu predviđenih ocjena koje je generisao Random Forest model (‘y_pred_rf‘).
- **Predviđene ocjene (Linear Regression) - zelena linija** predstavljaju raspodjelu predviđenih ocjena koje je generisao Linear Regression model (‘y_pred_lr‘).

```
plt.figure(figsize=(10, 6))
sns.kdeplot(y_test, label="Stvarne ocjene",
    fill=True, color='blue', alpha=0.4,
    linewidth=2)
sns.kdeplot(y_pred_rf, label="Predvi ene
    ocjene (Random Forest)", fill=True, color='
    orange', alpha=0.5, linewidth=2)
sns.kdeplot(y_pred_lr, label="Predvi ene
    ocjene (Linear Regression)", fill=True,
    color='green', alpha=0.6, linewidth=2)
plt.legend(fontsize=12)
plt.title("Pore enje distribucija", fontsize
    =16)
plt.grid(True)
plt.show()
```

Kod 18: Vizualizacija poređenja

`sns.kdeplot()`: Koristi se za izračunavanje i prikazivanje gustoće vjerovatnoće za svaki skup podataka (stvarne i predviđene ocjene).

`fill=True`: Ispunjava područje ispod linije gustoće.

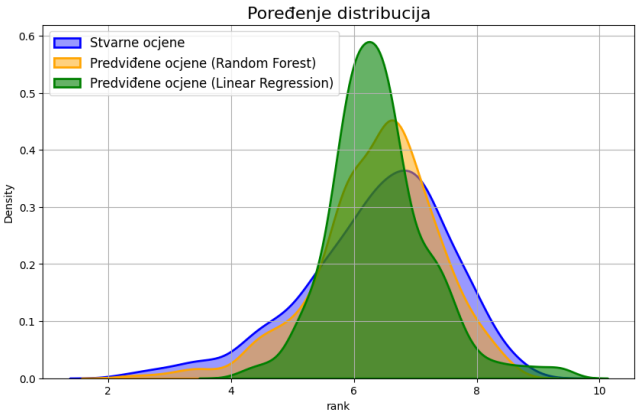
`alpha`: Određuje prozirnost boje za svaku liniju kako bi se lakše razlikovale.

`linewidth=2`: Povećava debljinu linije za bolju vidljivost.

`plt.legend()`: Dodaje legendu na grafu za lakše prepoznavanje linija.

`plt.grid(True)`: Dodaje mrežu na graf za bolju čitljivost.

Graf je korisna vizualizacija za usporedbu kako dobro različiti modeli (Random Forest i Linear Regression) predviđaju stvarne ocjene u odnosu na njihovu distribuciju.



Slika 11: Graf poređenja distribucija

Graf 11 omogućava usporedbu distribucija predviđenih i stvarnih ocjena te daje uvid u to koliko su predviđene ocjene za oba modela slične stvarnim ocjenama.

7) Pronalaženje optimalnih hiperparametara za random forest - grid search

U kodu 19 koristi se Grid Search za pronalaženje optimalnih hiperparametara za Random Forest Regressor unutar ‘rf_pipeline‘. Hiperparametri koji se optimiziraju su:

- **‘n_estimators‘**: Broj stabala u Random Forest modelu. Testiraju se dvije vrijednosti: 100 i 200.
- **‘max_depth‘**: Maksimalna dubina svakog stabla. Razmatraju se tri opcije: ‘None‘ (bez ograničenja dubine), 10 i 20.
- **‘min_samples_split‘**: Minimalan broj uzoraka potreban za podjelu čvora. Razmatraju se dvije opcije: 2 i 5.

```
param_grid = {
    'regressor__n_estimators': [100, 200],
    'regressor__max_depth': [None, 10, 20],
    'regressor__min_samples_split': [2, 5],
}
grid_search = GridSearchCV(rf_pipeline,
    param_grid, cv=5, scoring='
    neg_mean_squared_error', n_jobs=-1)
grid_search.fit(X_train, y_train)
```

Kod 19: Grid Search za Random Forest model

- `param_grid`: Definira vrijednosti koje će biti testirane za svaki hiperparametar.
- `cv=5`: Koristi 5-struku unakrsnu validaciju za procjenu performansi modela.
- `scoring='neg_mean_squared_error'`: Cilj je minimizirati negativnu srednju kvadratnu pogrešku (negativna jer je standard za `GridSearchCV` da maksimalizira rezultate).

- `n_jobs=-1`: Korištenje svih dostupnih procesora za paralelno izvođenje grid searcha, čime se ubrzava proces.
- `grid_search.fit(X_train, y_train)`: Pokreće Grid Search za treniranje modela koristeći različite kombinacije hiperparametara i traži onu koja daje najbolji rezultat prema negativnoj srednjoj kvadratnoj pogrešci (MSE).
- Ovaj proces omogućava optimizaciju performansi modela pronalaženjem najboljih parametara za Random Forest.
- Nakon izvršavanja Grid Search-a, najbolji parametri mogu biti pristupljeni putem `grid_search.best_params_`, a najbolji model putem `grid_search.best_estimator_`.

Nakon toga se ispisuju najbolji parametri, to jeste najbolji hiperparametri koje je GridSearchCV pronašao za Random Forest model. akon što se Grid Search završi, najbolji parametri mogu biti dohvaćeni pomoću 'grid_search.best_params_', što daje skup hiperparametara koji su dali najbolje rezultate prema definisanom kriteriju (u ovom slučaju, minimizacija negativne srednje kvadratne pogreške).

- `grid_search.best_params_`: Vraća optimalnu kombinaciju vrijednosti hiperparametara koja je dala najbolji rezultat u procesu pretrage.

```
param_grid = {
    'regressor__n_estimators': [100, 200],
    'regressor__max_depth': [None, 10, 20],
    'regressor__min_samples_split': [2, 5],
}
grid_search = GridSearchCV(rf_pipeline,
    param_grid, cv=5, scoring='
    neg_mean_squared_error', n_jobs=-1)
grid_search.fit(X_train, y_train)
```

Kod 20: Ispis najboljih parametara

Ispis rezultata koda 20:

GridSearchCV za Random Forest

Najbolji parametri: 'regressor__max_depth': 20, 'regressor__min_samples_split': 5, 'regressor__n_estimators': 200

Korištenjem ovih najboljih parametara možemo poboljšati performanse Random Forest modela i koristiti ga za predviđanja s boljim rezultatima.

8) Predikcija sa najboljim modelom

U ovom koraku koriste se najbolji parametri koji su dobijeni iz GridSearchCV za treniranje optimiziranog **Random Forest** modela. Nakon što se dobiju najbolji hiperparametri, predviđanja na testnom skupu podataka se vrše koristeći 'best_estimator_', koji sadrži model sa najboljim parametrima.

```
best_rf_model = grid_search.best_estimator_
y_pred_best_rf = best_rf_model.predict(X_test)
```

Kod 21: Predikcija sa najboljim modelom

- `grid_search.best_estimator_`: Dohvata najbolje trenirani model sa najboljim hiperparametrima.
- `best_rf_model.predict(X_test)`: Koristi optimizirani model za predviđanje ciljne varijable (`rank`) na testnom skupu podataka.

9) Evaluacija sa najboljim modelom

Nakon što se optimizira Random Forest model korištenjem GridSearchCV, evaluira se njegova učinkovitost na testnom skupu podataka. Evaluacija uključuje nekoliko ključnih mjera koje nam pomažu razumjeti kvalitetu predviđanja.

```
print("\n=== Evaluacija najboljeg Random Forest
     modela ===")
print(f"Mean Squared Error: {mean_squared_error
    (y_test, y_pred_best_rf):.2f}")
print(f"R2 Score: {r2_score(y_test,
    y_pred_best_rf):.2f}")
print(f"Mean Absolute Error: {
    mean_absolute_error(y_test, y_pred_best_rf)
    :.2f}")
print(f"Root Mean Squared Error: {np.sqrt(
    mean_squared_error(y_test, y_pred_best_rf))
    :.2f}")
```

Kod 22: Evaluacija sa najboljim modelom

- **Mean Squared Error (MSE)**: Mjera razlike između stvarnih i predviđenih vrijednosti. Niža vrijednost označava bolje performanse modela.
- **R2 Score (Koeficijent determinacije)**: Pokazuje koliko dobro model objašnjava varijaciju u podacima. Vrijednost bliža 1 znači bolju preciznost.
- **Mean Absolute Error (MAE)**: Mjera prosječne apsolutne greške između stvarnih i predviđenih vrijednosti.
- **Root Mean Squared Error (RMSE)**: Kvadratni korijen MSE koji također pokazuje veličinu greške, ali u istoj mjernoj jedinici kao originalni podaci.

Evaluacijski rezultati omogućuju nam da procijenimo koliko dobro optimizirani model predviđa ciljne varijable.

Ispis rezultata koda 22:

Evaluacija najboljeg Random Forest modela

Mean Squared Error: 0.33

R2 Score: 0.76

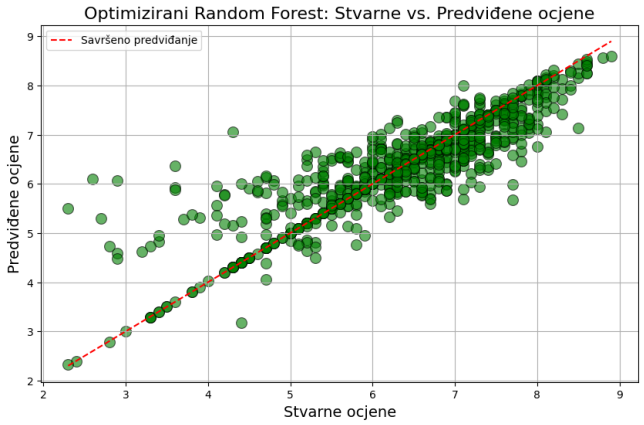
Mean Absolute Error: 0.35

Root Mean Squared Error: 0.58

10) Vizualizacija poboljšanja

Grafički prikaz poređenja stvarnih i predviđenih ocjena s optimiziranim Random Forest modelom daje vizualni uvid u to koliko su tačne prognoze u odnosu na stvarne vrijednosti.

- **Stvarne ocjene (x-osa)** naspram **predviđenih ocjena (y-osa)** na testnom skupu podataka.
- Crvena isprekidana linija označava "savršeno predviđanje" (gdje su stvarne i predviđene ocjene jednake).



Slika 12: Vizualizacija poboljšanja

- Razmještaj tačaka oko ove linije pokazuje koliko su predviđanja modela bliska stvarnim ocjenama.

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x=y_test, y=y_pred_best_rf,
    color='green', edgecolor='black', alpha
    =0.6, s=100)
plt.plot([y_test.min(), y_test.max()], [y_test.
    min(), y_test.max()], color='red',
    linestyle='--', label="Savr eno
    predvi anje")
plt.xlabel("Stvarne ocjene", fontsize=14)
plt.ylabel("Predvi ene ocjene", fontsize=14)
plt.title("Optimizirani Random Forest: Stvarne
    vs. Predvi ene ocjene", fontsize=16)
plt.grid(True)
plt.legend()
plt.show()
```

Kod 23: Vizualizacija poboljšanja

Grafikon 12 daje dobar vizualni prikaz preciznosti modela.

F. ANALIZA PRETHODNIH STUDIJA

BI i prediktivna analitika postali su ključni alati u filmskoj industriji, omogućavajući studijima donošenje odluka zasnovanih na podacima koje povećavaju šanse za uspjeh. Korištenjem velikih podataka, studiji mogu analizirati različite faktore koji utiču na performanse filma, od produkcije do marketinga i distribucije.

Ključne primjene prediktivne analitike u filmskoj industriji:

- **Prognoza prodajnih performansi**: Prediktivna analitika omogućava studijima da procijene potencijalne prodajne prihode analizom varijabli kao što su žanr, glumačka ekipa, režiser, datum izlaska i angažman na društvenim mrežama. Ova prognoza pomaže studijima da dodijele marketinške budžete i donesu informisane odluke o budućim projektima.
- **Optimizacija marketinških strategija**: Analizom demografskih podataka publike i obrazaca konzumacije medija, studiji mogu efikasnije prilagoditi svoje marketinške napore. Prediktivni modeli identifikuju najefikasnije platforme i strategije poruka za dosezanje ciljanih publika, čime se povećava vjerovatnoća uspjeha filma.
- **Povećanje angažmana gledatelja**: Streaming platforme koriste prediktivnu analitiku kako bi pružile personalizovane preporuke sadržaja na osnovu individualnih istorija gledanja. Ovaj pristup ne samo da poboljšava zadovoljstvo korisnika, već također povećava stope angažmana, što dovodi do veće lojalnosti kupaca.
- **Identifikacija novih prilika**: Napredni algoritmi mogu otkriti nove trendove i neiskorištena tržišta analizom opsežnih skupova podataka. Ova sposobnost omogućava studijima da se prilagode promjenjivim preferencijama publike i iskoriste nove prilike za rast unutar industrije. [40], [41], [42]

S obzirom na to da su u ovom istraživanju korištene dvije vrste modela za predikciju filmskih ocjena, linearna regresija i random forest, može se zaključiti da random forest model daje bolje rezultate u predikciji. Ovo nije samo specifično za naš rad, već je također potvrđeno u literaturi. Na primjer, kako je navedeno u radu "Movies Rating Prediction using Supervised Machine Learning Techniques" (autori: Siddique, A., Abid, M. K., Fuzail, M., Aslam, N, stranica 53, odjeljak 5. RESULTS AND DISCUSSION) [43], random forest model pokazuje bolje performanse u odnosu na druge modele poput SVM-a i KNN-a, što sugerira da random forest ne samo da nadmašuje linearne modele, već je i superiorniji u odnosu na druge popularne metode u oblasti mašinskog učenja.

Još jedan rad koji je istraživao upotrebu Random Forest modela pokazuje da je ovaj model postigao značajnu tačnost od 74% u predviđanju IMDb ocjena filmova, dok su drugi modeli poput Linearne Regresije i XGBoost-a imali visok trening rezultat, ali nisu dali precizna predviđanja. Istraživači su, kako bi poboljšali predviđanja, primijenili kategorisanje IMDb ocjena u intervale, što je omogućilo robusniji model i bolju interpretaciju podataka (IJFMR, 2023, str. 6). [44]

Korištenje BI, ML i AI u filmskoj industriji uglavnom se svodi na povećanje zarade, koja zavisi od različitih faktora, uključujući ocjene filmova, jer se zarada od filmova ostvaruje i mnogo godina nakon premijere. Filmovi imaju značajan utjecaj na ekonomiju, a uspjeh filma direktno je povezan s njegovim financijskim ishodom. Predviđanje uspjeha filmova postalo je ključno, a podaci sa stranica poput IMDb-a, koji pružaju informacije o filmu, glumcima, budžetima i recenzijama, važni su za primjenu ML tehnika u ovoj industriji. Na taj način, filmovi se bolje usklađuju s ukusima publike, što povećava šanse za financijski uspjeh. (Abid et al., 2023) [45]

Naravno, evo profesionalne verzije:

Predviđanje ocjene filma predstavlja izazovan zadatak jer ovisi o mnogim faktorima. Ocjene su često subjektivne i mogu varirati ovisno o individualnim preferenci-

jama, budući da neki ljudi mogu biti skloniji određenim žanrovima ili stilovima. Na društvenim mrežama, čak i prije premijere filma, može se osjetiti različit interes ili uzbuđenje u vezi s određenim filmom, što također može utjecati na konačnu ocjenu. Ponekad, visoka očekivanja prema filmu mogu rezultirati razočaranjem, iako je film objektivno dobar, s talentiranim režiserom i glumcima. Ipak, primjena metoda iz oblasti mašinskog učenja može biti korisna u predviđanju ocjena, što je potvrđeno u prethodnim istraživanjima. Na primjer, tokom Netflix Prize takmičenja 2009. godine, istraživači su koristili Singular Value Decomposition (SVD) kako bi predvidjeli ocjene filmova na temelju prethodnih ocjena korisnika (Stanford, 2019). [46] Ovo svakako pokazuje da su zaista različite tehnike, pristupi i ciljevi mogući koristeći ML i BI.

III. ZAKLJUČAK

Na temelju analize i usporedbe performansi dva prediktivna modela – linearne regresije i random foresta (RF) – u kontekstu predviđanja ocjena filmova temeljenih na različitim parametrima, rezultati jasno pokazuju da random forest pruža preciznije i pouzdanije predikcije u odnosu na linearnu regresiju. Iako linearna regresija ima prednost u jednostavnijim scenarijima s manje složenim podacima, random forest je pokazao veću sposobnost u modeliranju složenih i nelinearnih odnosa među parametarskim podacima, što je ključno u kontekstu predikcije ocjena filmova.

Prednost random foresta leži u njegovoj sposobnosti da razmatra veliki broj različitih kombinacija parametara i varijabli, čime može uočiti skrivena svojstva i uzorke koji bi bili zanemareni u linearnim modelima. Ovo je posebno korisno u poslovnoj inteligenciji, gdje preciznost u predikcijama može značajno utjecati na donošenje strateških odluka, kao što su preporuke filmova ili optimizacija marketinških kampanja.

Iako je linearna regresija dala razumljive rezultate za osnovne i jednostavne predikcije, random forest metoda je ponudila bolje rezultate u situacijama kada je potrebno modelirati složene, višedimenzionalne odnose. U konačnici, za poslovnu primjenu kao što je predviđanje ocjena filmova, random forest se pokazuje kao superiorniji alat, jer omogućava preciznije i robusnije predikcije, čime korisnicima omogućava bolji uvid u trendove i preference unutar filmske industrije.

IV. REFERENCE

REFERENCE

[1] IBM (2021). Business intelligence. [online] Ibm.com, dostupno na: https://www.ibm.com/think/topics/business-intelligence [pristupano: 19.01.2025.]

[2] Statista (2024). Cinema - Worldwide | Statista Market Forecast. [online] Statista, dostupno na: https://www.statista.com/outlook/amo/media/cinema/worldwide [pristupano: 19.01.2025.]

[3] Cvut.cz. (2025). CTU Relational | Dataset IMDb. [online] dostupno na: https://relational.fel.cvut.cz/dataset/IMDb [pristupano: 27.12.2024.]

[4] Google for Developers. (2024). Linear regression. [online], dostupno na: https://developers.google.com/machine-learning/crash-course/linear-regression [pristupano: 23.01.2025.]

[5] Dutta, A. (2019). Random Forest Regression in Python - GeeksforGeeks. [online] GeeksforGeeks, dostupno na: https://www.geeksforgeeks.org/random-forest-regression-in-python/ [pristupano: 23.01.2025.]

[6] encord.com. (n.d.). Mean Square Error (MSE) | Machine Learning Glossary | Encord | Encord. [online], dostupno na: https://encord.com/glossary/mean-square-error-mse/ [pristupano: 23.01.2025.]

[7] Onose, E. (2023). R Squared: Understanding the Coefficient of Determination. [online] Arize AI, dostupno na: https://arize.com/blog-course/r-squared-understanding-the-coefficient-of-determination/ [pristupano: 23.01.2025.]

[8] C3 AI. (n.d.). Mean Absolute Error. [online], dostupno na: https://c3.ai/glossary/data-science/mean-absolute-error/ [pristupano: 23.01.2025.]

[9] Deepchecks. (2024). What is Root Mean Square Error? Calculation Importance. [online], dostupno na: https://www.deepchecks.com/glossary/root-mean-square-error/ [pristupano: 23.01.2025.]

[10] dremio (n.d.). Grid Search | Dremio. [online] www.dremio.com, dostupno na: https://www.dremio.com/wiki/grid-search/ [pristupano: 23.01.2025.]

[11] DBeaver, dostupno na: https://dbeaver.com/docs/dbeaver/ [pristupano: 23.01.2025.]

[12] DBeaver, dostupno na: https://dbeaver.io [pristupano: 23.01.2025.]

[13] Visual Paradigm, dostupno na: https://www.visual-paradigm.com/support/documents/vpuserguide/12/13/5963_visualparadi.html [pristupano: 23.01.2025.]

[14] Visual Paradigm, dostupno na: https://www.visual-paradigm.com/download/ [pristupano: 23.01.2025.]

[15] Visual Studio Code, dostupno na: https://code.visualstudio.com/docs [pristupano: 23.01.2025.]

[16] Visual Studio Code, dostupno na: https://code.visualstudio.com/Download [pristupano: 23.01.2025.]

[17] W3Schools, "Pandas Introduction", [online], dostupno na: https://www.w3schools.com/python/pandas/pandas_intro.asp [pristupano: 21.01.2025.]

[18] W3Schools, "NumPy Introduction", [online], dostupno na: https://www.w3schools.com/python/numpy/numpy_intro.asp [pristupano: 21.01.2025.]

[19] Seaborn, dostupno na: https://seaborn.pydata.org [pristupano: 21.01.2025.]

[20] Matplotlib, dostupno na: https://matplotlib.org [pristupano: 21.01.2025.]

[21] matplotlib, "matplotlib.pyplot", [online], dostupno na: https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html [pristupano: 21.01.2025.]

[22] scikit-learn, dostupno na: https://scikit-learn.org/stable/getting_started.html [pristupano: 21.01.2025.]

[23] scikit-learn, "sklearn.model_selection", [online], dostupno na: https://scikit-learn.org/1.6/api/sklearn.model_selection.html#module-sklearn.model_selection [pristupano: 22.01.2025.]

[24] Scikit-learn, "sklearn.ensemble", [online], dostupno na: https://scikit-learn.org/dev/api/sklearn.ensemble.html [pristupano: 22.01.2025.]

[25] scikit-learn, dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.model_selection.GridSearchCV.html [pristupano: 22.01.2025.]

[26] scikit-learn, dostupno na: https://scikit-learn.org/1.6/api/sklearn.ensemble.html#module-sklearn.ensemble [pristupano: 22.01.2025.]

[27] scikit-learn, "RandomForestRegressor", [online], dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.ensemble.RandomForestRegressor.html [pristupano: 22.01.2025.]

[28] scikit-learn, dostupno na: https://scikit-learn.org/1.6/api/sklearn.linear_model.html#module-sklearn.linear_model [pristupano: 22.01.2025.]
[29] scikit-learn, dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.linear_model.LinearRegression.html#sklearn.linear_model.LinearRegression [pristupano: 22.01.2025.]
[30] scikit-learn, dostupno na: https://scikit-learn.org/1.6/api/sklearn.preprocessing.html#module-sklearn.preprocessing [pristupano: 22.01.2025.]
[31] scikit-learn, dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler [pristupano: 22.01.2025.]
[32] scikit-learn, dostupno na: https://scikit-learn.org/1.6/api/sklearn.metrics.html#module-sklearn.metrics [pristupano: 22.01.2025.]
[33] scikit-learn, dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.metrics.mean_squared_error.html [pristupano: 22.01.2025.]
[34] scikit-learn, dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.metrics.r2_score.html [pristupano: 22.01.2025.]
[35] scikit-learn, dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.metrics.mean_absolute_error.html [pristupano: 22.01.2025.]
[36] scikit-learn, dostupno na: https://scikit-learn.org/1.6/api/sklearn.pipeline.html#module-sklearn.pipeline [pristupano: 22.01.2025.]
[37] scikit-learn, dostupno na: https://scikit-learn.org/1.6/api/sklearn.impute.html#module-sklearn.impute [pristupano: 22.01.2025.]
[38] scikit-learn, dostupno na: https://scikit-learn.org/1.6/modules/generated/sklearn.impute.SimpleImputer.html#sklearn.impute.SimpleImputer [pristupano: 22.01.2025.]
[39] Visual Studio Code, "Jupyter Notebooks", [online], dostupno na: https://code.visualstudio.com/docs/datascience/jupyter-notebooks [pristupano: 23.01.2025.]
[40] filmgrail.com. (n.d.). Data Analytics in Film Industry: Audience Insights. [online] dostupno na: https://filmgrail.com/blog/data-analytics-in-film-industry-audience-insights/ [pristupano: 26.01.2025.]
[41] avianaglobal.com. (n.d.). Making Movies with Predictive Analytics. [online] dostupno na: https://avianaglobal.com/making-movies-with-predictive-analytics/ [pristupano: 26.01.2025.]
[42] quantzig.com. (n.d.). Predicting Movie Success: Data Analytics in the Film Industry. [online] dostupno na: https://www.quantzig.com/blog/predicting-movie-success-data-analytics-film-industry/ [pristupano: 26.01.2025.]
[43] Siddique, N.A., Kamran, M., None Muhammad Fuzail i Aslam, N.N. (2024). Movies Rating Prediction using Supervised Machine Learning Techniques. International Journal of Information Systems and Computer Technologies, 3(1), pp.40–56. doi:https://doi.org/10.58325/ijisct.003.01.0062 [pristupano: 26.01.2025.]
[44] IJFMR. (2023). Predicting IMDb ratings with Random Forest and other models dostupno na: https://www.ijfmr.com/papers/2023/6/8653.pdf [pristupano: 29.01.2025.]
[45] Shah, D., Swarda Mashere, Kumar, A., Chalse, S. i Pawar, R. (2023). Predicting Movie Success through Ratings Analysis: A Machine Learning Approach. [online] dostupno na: https://www.researchgate.net/publication/372314970_Predicting_Movie_Success_through_Ratings_Analysis_A_Machine_Learning_Approach [pristupano: 26.01.2025.]
[46] Stanford University. (2019). Related Work - Predicting Movie Ratings dostupno na: https://cs229.stanford.edu/proj2019aut/data/assignment_308832_raw/26260680.pdf [pristupano: 29.01.2025.]