Predviđanje plaća zaposlenika temeljem podataka koristeći metode strojnog učenja

Grgić, Nataša

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Nataša Grgić

Zagreb, 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Mentor: Student:

Izv. prof. dr. sc. Tomislav Stipančić, dipl. ing. Nataša Grgić

Zagreb, 2024.

Izjavljujem da sam ovaj rad izradila samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem se mentoru izv. prof. dr. sc. Tomislavu Stipančiću na pristupačnosti i pomoći tijekom izrade rada.

Najiskrenije zahvaljujem svojoj obitelji na neizmjernoj podršci, ljubavi i razumijevanju tijekom mog puta prema završetku ovog važnog poglavlja. Također, veliko hvala mojim prijateljima čiji su me poticaj, podrška i zajednički trenuci uljepšali i motivirali u ostvarivanju ciljeva.

Nataša Grgić

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne i diplomske ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum 23 -02- 2024	Prilog
Klasa: 602 – 04 / 24 – 06 / 1	
Ur.broj: 15 – 24 – 101	

ZAVRŠNI ZADATAK

Student: **Nataša Grgić** JMBAG: **0035231439**

Naslov rada na hrvatskom jeziku: **Predviđanje plaća zaposlenika temeljem podataka koristeći metode strojnog učenja**

Naslov rada na engleskom jeziku: **Predicting employee salaries based on data using machine learning methods**

Opis zadatka:

U svijetu podataka i velike količine informacija algoritmi umjetne inteligencije postaju sve značajniji. U ovisnosti o vrsti i strukturi podataka, metode strojnog učenja koriste se prilikom predviđanja, odlučivanja, klasifikacije, prepoznavanja i slično.
Koristeći nekoliko metoda za klasifikaciju primijenjenih na skupu podataka koji sadrži informacije o faktorima koji utječu na visinu plaće zaposlenika potrebno je analizirati mogućnosti predviđanja tih metoda te usporediti rezultate.
U sklopu rada potrebno je:

- pripremiti te analizirati podatke o plaćama zaposlenika u ovisnosti o utjecajnim faktorima da budu prikladni za korištenje od strane korištenih metoda
- primijeniti tri metode za klasifikaciju na zadanom skupu podataka koje bi se mogle koristiti da se utječe na točnost predviđanja plaće (npr. linear regression, , random forest i decision tree)
- načiniti analizu te usporediti rezultate primjene pojedinog klasifikatora.

U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan:
30. 11. 2023.

Zadatak zadao:
Izv. prof. dr. sc. Tomislav Stipančić

Datum predaje rada:
1. rok: 22. i 23. 2. 2024.
2. rok (izvanredni): 11. 7. 2024.
3. rok: 19. i 20. 9. 2024.

Predviđeni datumi obrane:
1. rok: 26. 2. – 1. 3. 2024.
2. rok (izvanredni): 15. 7. 2024.
3. rok: 23. 9. – 27. 9. 2024.

Predsjednik Povjerenstva:
Prof. dr. sc. Damir Godec

SADRŽAJ

POPIS SLIKA

POPIS OZNAKA

Oznaka	**Opis**
y	Ovisna varijabla jednadžbe
x	Neovisna varijabla jednadžbe
a	Nagib pravca
b	Odsječak na y - osi

SAŽETAK

Ovaj rad ima za cilj izgradnju modela predviđanja plaće zaposlenika koristeći odabrane ključne karakteristike i algoritme strojnog učenja. U području strojnog učenja koristi se tehnologija koja se oslanja na softverske programe, u ovom slučaju Python, kako bi postigla preciznije predviđanje ili detekciju te razvija računalne programe koji mogu koristiti podatke za autonomno učenje. Ovaj rad koristi klasifikacijske metode kao što su *decision trees, random forest* i *linear regression*. Također, provodi se usporedba različitih klasifikacijskih metoda kako bismo dobili uvid u to koja od njih pruža optimalno rješenje za postavljeni problem.

Ključne riječi: strojno učenje, algoritam, *Python,* model predviđanja, klasifikacija, plaća

SUMMARY

This paper aims to build a salary prediction model for employees using selected key features and machine learning algorithms. In the field of machine learning, the technology relies on software programs, in this case, Python, to achieve more accurate prediction or detection and develops computer programs that can use data for autonomous learning. This paper utilizes classification methods such as decision trees, random forest, and linear regression. Additionally, a comparison of different classification methods is conducted to gain insight into which one provides the optimal solution for the given problem.

Keywords: machine learning, algorithm, Python, prediction model, classification, salary

1. UVOD

U današnjem svijetu, kada je konkurencija na tržištu rada sve veća, zaposlenici su sve više motivirani da promijene poduzeće kako bi postigli što veću plaću. To predstavlja velike izazove za sama poduzeća koje žele zadržati svoje radnike, ali i izbjeći negativne posljedice koje bi mogle poremetiti mobilnost tvrtke.

Ova se tehnika može koristiti za analizu velikog broja različitih faktora koji utječu na cijenu rada kao što su demografija, razina obrazovanja, naziv radne pozicije, godine iskustva i dosadašnja plaća. U daljnjem će tekstu završnog rada ovi faktori biti pobliže objašnjeni.
Predviđanje plaće na osnovu strojnog učenja može imati brojne prednosti za kompanije. One mogu koristiti te informacije za donošenje odluka o zapošljavanju i unapređenju zaposlenika, samim time motiviraju svoje zaposlenike za daljni rad.

Kroz ovaj rad pokušat ću predvidjeti ljudsku plaću koristeći algoritme strojnog učenja kao što su: linearna regresija (eng. *linear regression*), nasumična šuma (eng. *random forest*) i stablo odlučivanja (eng. *decision tree*) te usporediti koji od tih algoritama daje najbolje rezultate.

2. Strojno učenje

Strojno učenje je grana umjetne inteligencije koja se bavi razvojem sustava i algoritama koji omogućuju računalima da uče i donose odluke na temelju podataka, bez eksplicitnog programiranja. Glavni cilj strojnog učenja je razvoj modela i tehnika koji omogućuju računalima da poboljšavaju svoje performanse s iskustvom, bez potrebe za specifičnim programiranjem za rješavanje određenih zadataka.

2.1. Učenje bez nadzora

Učenje bez nadzora (*eng. unsupervised learning*) je vrsta strojnog učenja u kojoj se algoritmi koriste za otkrivanje skrivenih uzoraka ili struktura iz neoznačenih podataka. U okviru takvog pristupa, model strojnog učenja pokušava sam pronaći sličnosti, razlike, uzorke i strukturu u podacima. Nije potrebno prethodno ljudsko posredovanje. Ovaj pristup često se primjenjuje za razdvajanje skupova podataka, identificiranje obrazaca, reduciranje dimenzionalnosti podataka te generiranje novih ideja ili hipoteza o podacima. Koristi se u širokom spektru aplikacija, od preporuka proizvoda i segmentacije slike do detekcije anomalija i komprimiranja podataka.

Algoritam učenja bez nadzora može se dodatno kategorizirati u dvije vrste problema, a to su klasterska i asocijacijska analiza.

Klasterska metoda (*eng. clustering*) sastoji se od grupiranja objekata u klastere tako da objekti s najvećim sličnostima ostaju u istoj grupi, dok imaju manje ili nimalo sličnosti s objektima iz drugih grupa. Analiza klastera otkriva zajedničke karakteristike među podacima i kategorizira ih prema prisutnosti ili odsutnosti tih zajedničkih karakteristika. Primjerice, u skupu podataka o voću, klasterizacija bi mogla grupirati voće poput jabuka, krušaka i šljiva u jedan klaster, dok bi banane i naranče bile u drugom, na temelju njihovih sličnosti u težini, slatkoći ili drugim značajkama. Ova tehnika omogućuje razumijevanje prirode podataka i identifikaciju skrivenih uzoraka ili grupa unutar njih.

Asocijacijska analiza (*eng. association*) je metoda učenja bez nadzora koja se koristi za pronalaženje odnosa između varijabli u velikoj bazi podataka. Ono određuje skup elemenata koji se pojavljuju zajedno u skupu podataka. Primjerice, u analizi kupovina u trgovini, postupak asocijacije bi identificirao učestalu pojavu kruha i mlijeka u istoj transakciji.

2.2. Učenje pod nadzorom

Učenje pod nadzorom (*eng. supervised learning*) je grana strojnog učenja gdje se algoritmu pružaju ulazni podaci zajedno s pripadajućim izlaznim oznakama ili rezultatima. Ovaj pristup omogućuje algoritmu da nauči prediktivne modele ili donese odluke na temelju primjera iz označenog skupa podataka. Nadzirani dio referira se na činjenicu da algoritmu pružamo "nadzor" putem označenih podataka, što mu omogućuje učenje i prilagođavanje cilju koji želimo postići. Primjeri učenja pod nadzorom uključuju klasifikaciju, gdje se cilj predviđanja odnosi na kategorijske izlazne varijable, te regresiju, gdje se cilj predviđanja odnosi na kontinuirane izlazne varijable.

Algoritmi regresije koriste se koriste za modeliranje odnosa između nezavisnih ulaznih varijabli i kontinuirane izlazne varijable. Cilj je naučiti funkciju koja može predvidjeti vrijednost izlazne varijable na temelju vrijednosti ulaznih varijabli. Primjeri algoritama regresije uključuju linearnu regresiju, polinomijalnu regresiju, regresijska stabla, podržane vektorske regresije (SVR) i neuralne mreže. Koriste se za predviđanje kontinuiranih varijabli, poput prognoze vremena, trendova na tržištu, predviđanje cijena i slično.

Algoritmi klasifikacije koriste se za klasifikaciju podataka u različite kategorije ili klase. Cilj je naučiti model koji može razlikovati između različitih klasa na temelju ulaznih varijabli. Primjeri algoritama klasifikacije uključuju logističku regresiju, nasumičnu šumu, stablo odlučivanja i druge. Ovi algoritmi se često koriste u raznim aplikacijama poput prepoznavanja uzoraka, klasifikacije slika, detekcije spam poruka i slično.

2.2.1. Linearna regresija

Linearna regresija (*eng. linear regression*) je statistička metoda koja se koristi za analizu odnosa između nezavisnih promjenljivih (također poznatih kao prediktori) i zavisne promjenljive varijable. Osnovna ideja linearne regresije je pronaći linearnu vezu između

nezavisnih varijabli i zavisne varijable, što omogućuje predviđanje vrijednosti zavisne varijable na temelju poznatih vrijednosti nezavisnih varijabli.
Najbolji način za shvatiti i opisati sam problem linearne regresije jest korištenjem matematičke jednadžbe pravca koja glasi:

$$y = ax + b \tag{1}$$

gdje je:

y – ovisna varijabla
x – neovisna varijabla
a – nagib pravca
b – odsječak na y-osi

Nagib pravca (a) predstavlja promjenu ovisne varijable (y) za promjenu neovisne varijable (x) za jednu jedinicu. Odsječak na y-osi (b) predstavlja vrijednost ovisne varijable (y) kada je neovisna varijabla (x) jednaka nuli.

U okviru ovog završnog rada ću koristiti skup podataka koji obuhvaća osnovne informacije o pojedincu, godine iskustva i dosadašnju plaću. Ključno pitanje koje ćemo istražiti jest: Kako predvidjeti novu plaću zaposlenika?

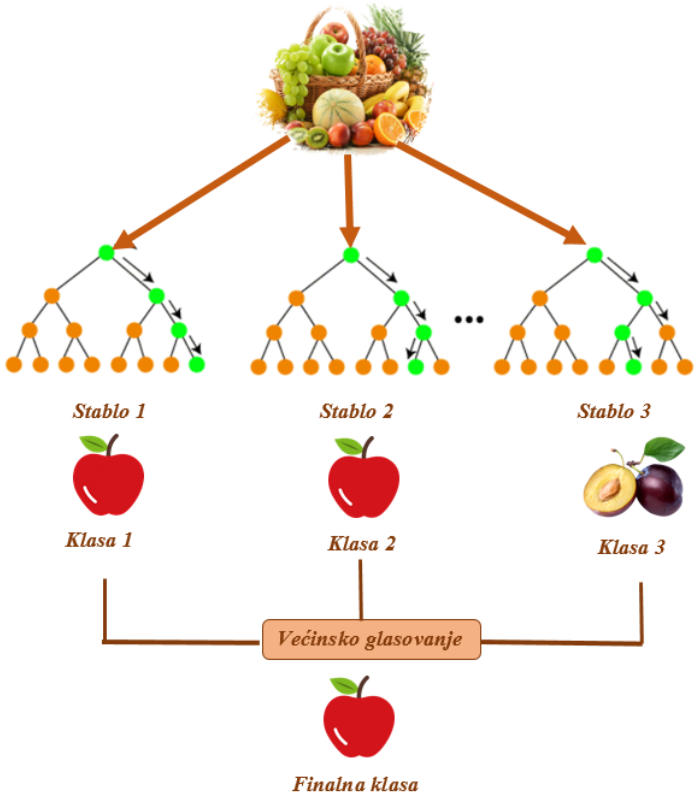
2.2.2. Nasumična šuma

Nasumična šuma (*eng. random forest*) je popularan algoritam strojnog učenja koji pripada superviziranoj tehnici učenja. Može se koristiti kako za klasifikacijske, tako i za regresijske probleme u strojnom učenju. Temelji se na konceptu ansambla učenja, što je proces kombiniranja više klasifikatora kako bi se riješio složen problem i poboljšala performansa modela.

Kao što samo ime sugerira, nasumična šuma je klasifikator koji sadrži nekoliko stabala odlučivanja na različitim podskupovima danog skupa podataka i uzima prosjek kako bi poboljšao prediktivnu točnost tog skupa podataka. Umjesto oslanjanja na jedno stablo

odlučivanja, slučajna šuma uzima predikciju iz svakog stabla i na temelju većine glasova predikcija predviđa konačni rezultat.

Veći broj stabala u šumi dovodi do veće točnosti i sprječava problem prenaučenosti.



Slika 1. Primjer nasumične šume

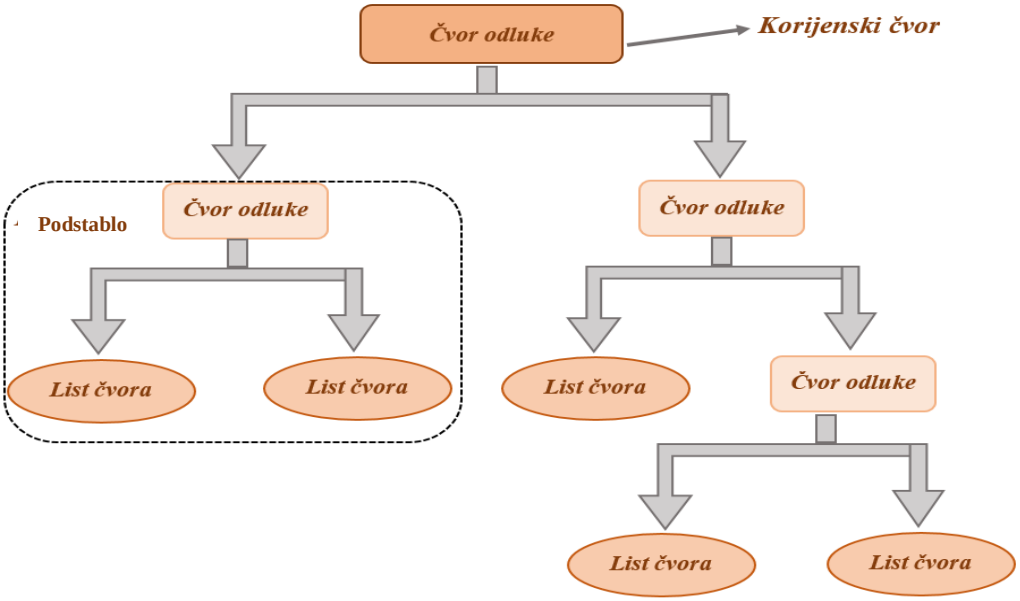
2.2.3. Stablo odlučivanja

Stablo odlučivanja (*eng. decision tree*) je tehnika koja se može koristiti i za klasifikacijske i za regresijske probleme, ali uglavnom se preferira za rješavanje klasifikacijskih problema. Ova dva tipa se obično zajednički nazivaju CART (*eng. Classification and Regression Tree*). Ovaj algoritam koristi se za stvaranje stabla kojim se predstavlja niz pravila odlučivanja koja se primjenjuju na ulazne podatke radi donošenja zaključaka o izlaznoj varijabli. To je klasifikator strukturiran kao stablo, gdje unutarnji čvorovi predstavljaju značajke skupa podataka, grane predstavljaju pravila odlučivanja, a svaki list čvora predstavlja ishod.

Cilj mu je postići optimalne odluke na kraju svakog čvora, što zahtijeva odgovarajući algoritam. Takav algoritam je poznat kao Huntov algoritam, koji sustavno donosi najbolje moguće odluke i rješava složene probleme razdvajajući ih na manje dijelove. Ključna ideja iza Huntovog algoritma leži u korištenju čistoće kao ključne metrike za procjenu homogenosti podataka u čvoru stabla. Čistoća čvora odražava koliko su svi podaci u tom čvoru pripadaju istoj klasi. Čistoća čvora odražava koliko su svi podaci u tom čvoru pripadaju istoj klasi. Što je veća čistoća čvora, to je podjela podataka efikasnija, jer se povećava informacijska dobit dobivena iz svakog čvora. Stoga, Huntov algoritam teži maksimiziranju čistoće tijekom podjele čvorova kako bi generirao stablo odlučivanja koje uspješno klasificira podatke.

Tijekom izrade stabla odlučivanja, algoritam odabire optimalne značajke za podjelu podataka koristeći određeni kriterij, kao što je Gini indeks. Gini indeks je metrika koja se koristi za procjenu nečistoće ili raznolikosti podataka u kontekstu stabla odlučivanja. On se koristi za odabir optimalnih značajki za podjelu podataka i mjeri koliko dobro ta podjela razdvaja različite klase unutar skupa podataka. Vrijednosti Gini indeksa variraju između 0 i 1, pri čemu niža vrijednost označava veću čistoću, odnosno bolju podjelu podataka. Kada je Gini indeks blizu 0, to ukazuje na to da su svi podaci u čvoru iste klase, dok vrijednost bliža 1 sugerira veću raznolikost podataka. Dakle, cilj stabla odlučivanja je minimizirati Gini indeks kako bi se postigla što bolja podjela podataka i generiralo efikasno stablo za klasifikaciju ili regresiju.

Stablo odlučivanja pruža prednosti u donošenju odluka zbog svoje jednostavnosti, transparentnosti i sposobnosti interpretacije rezultata, te se može prilagoditi različitim vrstama podataka i problemima. Ipak, postoje izazovi koji proizlaze iz korištenja stabla odlučivanja. Na primjer, algoritam može biti podložan preprilagodbi ako se koristi na kompleksnim ili velikim skupovima podataka, što može dovesti do manje preciznih rezultata. Također, stablo odlučivanja može biti osjetljivo na male promjene u podacima, što može uzrokovati značajne varijacije u generiranim stablima.

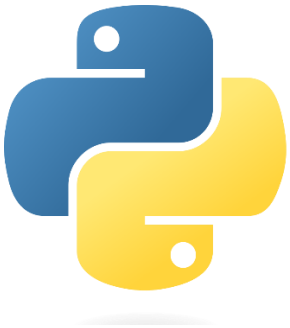


Slika 2. Primjer stabla odlučivanja

3. Baza podataka

3.1. Programski jezik Python

Python je interpretirani programski jezik visoke razine koji je postao vrlo popularan među programerima zbog svoje jednostavne sintakse, fleksibilnosti i moćnih značajki. Razvijen je kako bi olakšao proces programiranja te potiče čitljivost i jednostavnost koda. Široko je korišten u različitim područjima kao što su web razvoj, znanstveno računanje, analiza podataka, umjetna inteligencija i strojno učenje. Python se temelji na načelima jednostavnosti i praktičnosti te nudi širok raspon funkcionalnosti, uključujući dinamičko tipiziranje i automatsko upravljanje memorijom. Otkako je prvi put objavljen 1991. godine, koji je osmislio Guido van Rossum, Python je doživio izniman rast popularnosti i postao jedan od vodećih programskih jezika u svijetu, prepoznat po svojoj širokoj primjeni i korisnoj zajednici.



Slika 3. Logo programskog jezika Python

3.1.1. Demografija

Razlike u primanjima često se primjećuju između različitih demografskih skupina, pri čemu faktori poput rodne diskriminacije, rasne i etničke pripadnosti, te socioekonomskog statusa mogu igrati ključnu ulogu. Na primjer, istraživanja su pokazala da žene i manjine često ostvaruju manje prihode od svojih muških kolega za isti posao. Također, regije s većim standardom života obično imaju više plaće u usporedbi s ruralnim ili manje razvijenim područjima. Ovi demografski faktori važni su za razumijevanje nejednakosti u plaćama.

3.1.2. Razina obrazovanja

Općenito, osobe s višim stupnjem obrazovanja, kao što su diplomirani ili magistri, imaju tendenciju ostvarivanja većih primanja u usporedbi s onima s nižim stupnjem obrazovanja. Visoko obrazovani pojedinci često imaju bolje prilike za zapošljavanje u zahtjevnijim i bolje plaćenim pozicijama, te su često u mogućnosti stjecati specijalizirane vještine koje su tražene na tržištu rada. Osim toga, poslodavci često pridaju veću vrijednost visokoobrazovanim kandidatima, što rezultira većom ponuđenom plaćom. Stoga je ulaganje u obrazovanje ključno za osobni i profesionalni razvoj te ostvarivanje većih primanja u karijeri. Iako to nužno ne mora uvijek biti tako. Ponekad će osoba bez fakultetske diplome, ali s većom motivacijom i željom za učenjem i napretkom, postići bolju plaću od osobe s visokim stupnjem obrazovanja.

3.1.3. Naziv radne pozicije

Radna pozicija uvelike određuje visinu plaće pojedinca. Osobe na višim pozicijama, koje obično zahtijevaju veću odgovornost, stručnost i iskustvo, najčešće imaju veće prihode u usporedbi s onima na nižim pozicijama. Razlika u plaći često se primjećuje u hijerarhijskoj strukturi organizacije, pri čemu menadžeri, direktori i drugi visoki dužnosnici često ostvaruju znatno veće primanja od radnika na nižim razinama hijerarhije. Osim toga, kvalifikacije, radne obveze, potrebne vještine i specijalizacija za određenu poziciju igraju ključnu ulogu u određivanju razine plaće. Stoga je razumijevanje kako trenutna pozicija utječe na plaću bitno za pojedince prilikom planiranja karijere i pregovaranja o uvjetima zaposlenja.

3.1.4. Godine iskustva i plaća

Godine iskustva igraju ključnu ulogu u određivanju razine plaće radnika. Općenito, kako radnici stječu više iskustva tijekom karijere, njihova vještina, stručnost i vrijednost na tržištu rada obično rastu. Stoga stariji radnici, koji imaju veći broj godina iskustva, često ostvaruju veće prihode od svojih mlađih kolega s manje iskustva. Osim toga, radnici s dugogodišnjim iskustvom često imaju bolje pozicije, veću odgovornost i bolje prilike za napredovanje, što također može rezultirati povećanjem njihove plaće. Međutim, važno je napomenuti da utjecaj godina iskustva na plaću može varirati ovisno o industriji, zanimanju, obrazovanju i drugim

faktorima. Stoga je potrebno pažljivo analizirati kontekstualne čimbenike kako bismo bolje razumjeli kompleksnu dinamiku između godina iskustva i razine plaće radnika.

3.2. Učitavanje biblioteke

Korištenje već napisanih biblioteka omogućava programerima da izbjegnu pisanje koda "od nule" za svaku funkcionalnost koju žele implementirati, što značajno ubrzava razvojni proces. Osim toga, biblioteke često sadrže optimizirane implementacije algoritama i funkcija, što može poboljšati performanse i efikasnost programa. Učitavanje potrebne biblioteke prikazano je na slici 4.

```
1   # import needed libraries
2   import pandas as pd
3   import numpy as np
4   import matplotlib.pyplot as plt
5   import seaborn as sns
6   import warnings
7   warnings.filterwarnings('ignore')
8   from sklearn.model_selection import train_test_split, GridSearchCV
9   from sklearn.preprocessing import LabelEncoder
10  from sklearn.linear_model import LinearRegression
11  from sklearn.ensemble import RandomForestRegressor
12  from sklearn.tree import DecisionTreeRegressor
13  from sklearn.metrics import mean_squared_error,mean_absolute_error
```

Slika 4. Učitavanje biblioteke

Predviđanje plaće određuje se sa 6 ulaznih varijabli:

1) Godine zaposlenika
2) Spol zaposlenika
3) Razina obrazovanja
4) Naziv radnog mjesta zaposlenika
5) Godine iskustva zaposlenika
6) Plaća zaposlenika

	Age	Gender	Education Level	Job Title	Years of Experience	Salary
0	32.0	Male	Bachelor's	Software Engineer	5.0	90000.0
1	28.0	Female	Master's	Data Analyst	3.0	65000.0
2	45.0	Male	PhD	Senior Manager	15.0	150000.0
3	36.0	Female	Bachelor's	Sales Associate	7.0	60000.0
4	52.0	Male	Master's	Director	20.0	200000.0

Slika 5. Baza podataka

3.2.1. Pandas

Pandas je moćna biblioteka u programskom jeziku Python koja nudi mnoge funkcionalnosti za rad s podacima. Jedna od glavnih funkcionalnosti Pandasa je učitavanje podataka iz različitih izvora. Osim toga, Pandas pruža moćne alate za manipulaciju podacima, kao što su dodavanje, brisanje i uređivanje redaka i stupaca, preimenovanje indeksa i promjena tipova podataka. Također omogućava indeksiranje i selekciju podataka na različite načine, što olakšava pristup i manipulaciju podacima.

Slika 6. Logo biblioteke Pandas

3.2.2. Numpy

NumPy je moćna biblioteka koja omogućuje rad s višedimenzionalnim nizovima i matricama te pruža širok spektar matematičkih funkcija za manipulaciju tim podacima. Nudi efikasne algoritme za izvođenje različitih matematičkih operacija, uključujući statističke analize, linearnu algebru i generiranje slučajnih brojeva. Koristi kontinuirani blok memorije, što znači da su podaci pohranjeni jedan za drugim u memoriji. Ovo omogućava brz pristup elementima niza i efikasno izvršavanje operacija na podacima.

Slika 7. Logo biblioteke Numpay

3.2.3. Matplotlib

Matplotlib je Python biblioteka koja se koristi za vizualizaciju podataka, omogućujući korisnicima stvaranje raznolikih grafikona, dijagrama i drugih vizualnih prikaza. Ovaj alat je ključna za analizu i interpretaciju podataka, pružajući korisnicima fleksibilnost i kontrolu nad izgledom i stilom vizualizacija. Korisnici mogu prilagoditi svoje grafikone prema vlastitim potrebama i preferencijama zahvaljujući bogatom setu opcija koji Matplotlib nudi.

Slika 8. Logo biblioteke Matplotlib

3.2.4. Seaborn

Seaborn je biblioteka koja se oslanja na Matplotlib i nudi jednostavan i intuitivan način za kreiranje privlačnih i informativnih statističkih grafikona. Njegova popularnost leži u jednostavnosti korištenja, estetski ugodnim vizualizacijama i bogatim statističkim funkcionalnostima. Nudi širok raspon funkcionalnosti koje olakšavaju proces razumijevanja podataka. Neki od glavnih dijagrama koje može prikazati su distribucijski dijagrami, linearnu regresiju, toplinske mape, kategorizirane dijagrame i slično.

Slika 9. Logo biblioteke Seaborn

3.2.5. Warnings

Biblioteka Warnings u Pythonu omogućuje manipulaciju upozorenjima koja se generiraju tijekom izvršavanja programa. Ova biblioteka omogućuje nam kontrolu nad tim upozorenjima, omogućujući nam da ih prikažemo, filtriramo ili potisnemo prema našim potrebama. Kroz biblioteku možemo definirati različite razine upozorenja, primijeniti filtre prema specifičnim kriterijima ili čak prilagoditi funkcije koje će se pozvati kada se upozorenje pojavi. Ova biblioteka je korisna u situacijama gdje želimo kontrolirati način prikazivanja ili obrade upozorenja tijekom izvršavanja programa, što je posebno korisno u razvoju softvera ili analizi podataka.

3.3. Učitavanje podataka

CSV datoteka je vrsta datoteke koja se koristi za pohranu podataka u tabličnom formatu, gdje su podaci odvojeni zarezima. Naziv CSV dolazi od engleske skraćenice *"Comma-Separated Values"* (vrijednosti odvojene zarezima). One se koriste za razmjenu podataka između različitih programa, za uvoz i izvoz podataka, kao i za pohranu tabličnih podataka u čitljivom formatu. Za unos podataka iz te datoteke u naš DataFrame, koristimo funkciju read_csv koja je sposobna interpretirati takve datoteke.

```
# ucitavanje csv datoteke
salary_df = pd.read_csv('Salary_Data.csv')
print(salary_df.head())
```

Slika 10. Učitavanje podataka

Linija koda *„salary_df.head()“* koristi se za prikaz prvih pet redaka DataFramea *salary.df* . Ova metoda omogućava brz pregled početnog dijela podataka kako bismo dobili osnovni uvid

u strukturu i sadržaj DataFramea. Ukoliko želimo prikazati veći broj redaka, možemo proslijediti odgovarajući argument metodi *„head()“*. Ovo je korisna tehnika prilikom istraživanja i analize podataka u Pythonu, posebno kada radimo s velikim skupovima podataka.

```
    Age  Gender  ... Years of Experience    Salary
0  32.0    Male  ...                5.0   90000.0
1  28.0  Female  ...                3.0   65000.0
2  45.0    Male  ...               15.0  150000.0
3  36.0  Female  ...                7.0   60000.0
4  52.0    Male  ...               20.0  200000.0
```

Slika 11. Prvih 5 redaka DataFramea

Slika 12 prikazuje liniju koda koja se koristi za dobivanje detaljnih informacija o DataFrame-u *salary_df* u Pythonu. Ova metoda pruža opsežan pregled strukture i svojstava DataFrame-a, uključujući broj redaka i stupaca, nazive stupaca, tipove podataka u svakom stupcu te informacije o nepopunjenim vrijednostima. Ovakve informacije su korisne prilikom analize podataka, jer nam omogućuju bolje razumijevanje podataka s kojima radimo, kao i identifikaciju potencijalnih problema s podacima.

```
15    # ucitavanje csv datoteke
16    salary_df = pd.read_csv('Salary_Data.csv')
17    print(salary_df.head())

[5 rows x 6 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6704 entries, 0 to 6703
Data columns (total 6 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Age                  6702 non-null   float64
 1   Gender               6702 non-null   object
 2   Education Level      6701 non-null   object
 3   Job Title            6702 non-null   object
 4   Years of Experience  6701 non-null   float64
 5   Salary               6699 non-null   float64
dtypes: float64(3), object(3)
memory usage: 314.4+ KB
None
```

Slika 12. Osnovne informacije o DataFrameu

Analizu i obradu nepopunjenih vrijednosti u DataFrameu možemo vidjeti na slici 9. Linija koda *„print(salary df.isnull().sum))“* se koristi za brojanje nepopunjenih vrijednosti u svakom stupcu DataFramea *salary_df*. Nakon toga, linija koda *„salary df.dropna(inplace=True)“* koristi se za trajno uklanjanje redaka koji sadrže nepopunjene vrijednosti iz DataFramea. Konačno, linija koda *„print(salary df.describe())“* generira sažetak statističkih informacija o numeričkim stupcima u DataFrameu, pružajući uvid u raspodjelu podataka nakon uklanjanja nepopunjenih vrijednosti.

```
22    # otkrivanje nultih vrijednosti
23    print(salary_df.isnull().sum())
24    # izuzimanje nultih vrijednosti
25    salary_df.dropna(inplace=True)
26    # opis podataka
27    print(salary_df.describe())
```

Slika 13. Nepopunjene vrijednosti DataFramea

	Age	Years of Experience	Salary
count	6698.000000	6698.000000	6698.000000
mean	33.623022	8.095178	115329.253061
std	7.615784	6.060291	52789.792507
min	21.000000	0.000000	350.000000
25%	28.000000	3.000000	70000.000000
50%	32.000000	7.000000	115000.000000
75%	38.000000	12.000000	160000.000000
max	62.000000	34.000000	250000.000000

Slika 14. Osnovne statističke vrijednosti o podatcima

4. Analiza podataka

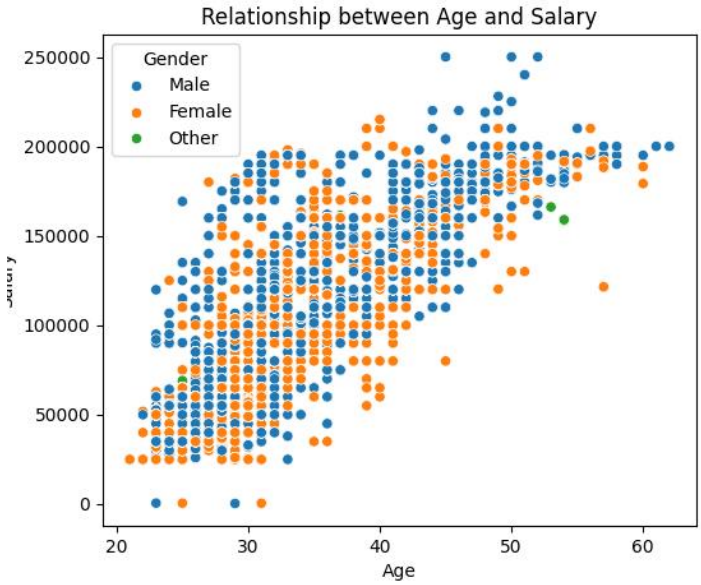
4.1. Grafički prikaz odnosa ulaznih parametara

4.1.1. Odnos između dobi zaposlenika i plaće

Analizirajući sliku 15, izrađuje se grafikon raspršenosti koji istražuje vezu između dobi zaposlenika i visine njihove plaće. Ovaj vizualni prikaz omogućuje promatranje kako se plaća mijenja s godinama iskustva, pružajući uvid u potencijalne trendove ili uzorke u podacima. Svaka točka na grafikonu, slike 16, predstavlja pojedinog zaposlenika, pri čemu je boja točke određena spolom, što dodatno omogućuje analizu razlika u plaćama između muškaraca i žena u različitim dobnim skupinama. Ova vizualizacija pruža intuitivno razumijevanje dinamike između dobi i plaće te omogućuje identifikaciju potencijalnih obrazaca ili anomalija u podacima. Postavljeni naslov "*Relationship between Age and Salary*" jasno odražava svrhu analize i naglašava ključni aspekt istraživanja. Analiza ovakvih veza može poslužiti kao temelj za daljnje istraživanje i donošenje informiranih odluka u kontekstu upravljanja ljudskim resursima, kao i razvoja strategija za kompenzaciju zaposlenika.

```
30    # odnos izmedu dobi zaposlenika i place
31    plt.figure(figsize=(6, 5))
32    sns.scatterplot(x='Age',y='Salary', data=salary_df,
33                    hue='Gender').set(title='Relationship between Age and Salary')
34    plt.show()
```

Slika 15. Linija koda za prikaz grafikona



Slika 16. Odnos između dobi zaposlenika i plaće

4.1.2. Odnos između iskustva zaposlenika i plaće

Na slici 17 linija koda kreira grafikon raspršenosti koji istražuje povezanost između godina iskustva zaposlenika i visine njihove plaće. Grafikon prikazuje svakog zaposlenika kao točku na grafu, gdje je položaj točke na osi x određen godinama iskustva, na osi y visinom plaće, a bojom točke označen spol zaposlenika. Ova vizualizacija omogućuje uvid u kako se plaća mijenja s godinama iskustva te kako ta promjena može varirati ovisno o spolu. Naslov "*Relationship between Experience and Salary*" jasno identificira temu grafikona i ističe ključni aspekt analize. Razumijevanje ove veze može pomoći u donošenju odluka vezanih uz politike plaća, poticanje profesionalnog razvoja zaposlenika te identifikaciju i rješavanje eventualnih nejednakosti u plaćama unutar organizacije.

```
36  # odnos izmedu iskustva zaposlenika i place
37  plt.figure(figsize=(6, 5))
38  (sns.scatterplot(x='Years of Experience', y='Salary', data=salary_df, hue='Gender')
39   .set(title='Relationship between Experience and Salary'))
40  plt.show()
```

Slika 17. Linija koda za prikaz grafikona

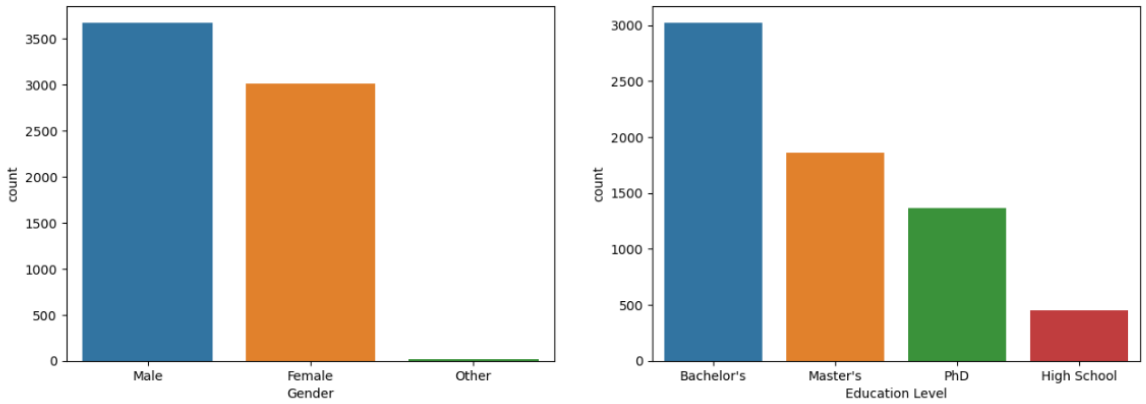
Slika 18. Odnos između iskustva zaposlenika i plaće

4.2. Raspodjela kategoričkih varijabli

Kategorizacija kategoričkih varijabli kao što su spol, obrazovanje ili zanimanje igra ključnu ulogu u predviđanju ljudske plaće iz nekoliko razloga. Prvo, analiza distribucije ovih varijabli omogućuje nam da identificiramo bitne faktore koji utječu na razinu plaće, što nam omogućuje prilagodbu strategija zapošljavanja ili plaćanja. Drugo, omogućuje nam personaliziranu analizu plaće prema različitim skupinama ljudi, što olakšava pravedniji pristup u procesima zapošljavanja i plaćanja. Treće, klasifikacija nam omogućuje da uključimo ove varijable u prediktivne modele, što rezultira poboljšanjem njihove preciznosti u predviđanju razlika u plaćama. Konačno, razumijevanje distribucije plaća prema kategoričkim varijablama pomaže nam u planiranju politika i strategija zapošljavanja te socijalnih politika, što doprinosi stvaranju pravednijeg i efikasnijeg tržišta rada.

```
42    # raspodjela kategorickih varijabli
43    fig, ax = plt.subplots( nrows: 1,  ncols: 2, figsize=(15, 5))
44    sns.countplot(x='Gender', data=salary_df, ax=ax[0])
45    sns.countplot(x='Education Level', data=salary_df, ax=ax[1])
```

Slika 19. Stvaranje grafikona



Slika 20. Raspodjela kategoričkih varijabli

Prvi grafikon otkriva da značajan dio zaposlenika čine muškarci, dok drugi grafikon pokazuje da većina zaposlenika ima završen preddiplomski studij.

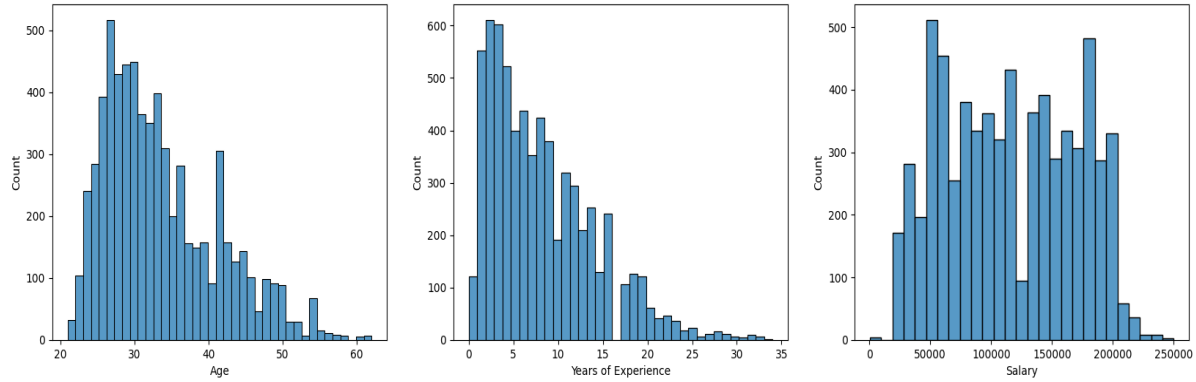
4.3. Distribucija atributa u DataFrameu

Proučavanje histograma i distribucije atributa važno je za razumijevanje kako su podaci raspoređeni u skupu podataka. Kod predviđanja ljudske plaće, razumijevanje distribucije atributa kao što su dob, godine radnog iskustva i plaća može pomoći u prepoznavanju uobičajenih obrazaca i trendova. Na primjer, normalna distribucija plaća može ukazivati na stabilnost u plaćama, dok nepravilna distribucija može ukazivati na prisutnost različitih skupina radnika s različitim primanjima. To je ključno za pravilno modeliranje i predviđanje plaća te donošenje odluka o politikama plaća i upravljanju ljudskim resursima.

```
47    # distribucija atributa u DataFrameu
48    fig, ax = plt.subplots( nrows: 1,  ncols: 3, figsize=(20, 5))
49    sns.histplot(salary_df['Age'], ax=ax[0])
50    sns.histplot(salary_df['Years of Experience'], ax=ax[1])
51    sns.histplot(salary_df['Salary'], ax=ax[2])
```

Slika 21. Stvaranje 3 histograma

Slika 18 prikazuje tri histograma koji prikazuju distribucije atributa iz DataFramea *salary_df*. Prva distribucija odnosi se na dob zaposlenika (*„Age“*), druga na godine radnog iskustva („*Years of experience*“), dok treća prikazuje distribuciju plaća („*Salary*“).



Slika 22. Histogrami

Grafikon 1 ističe da većina zaposlenika pripada dobnoj skupini od 23 do 37 godina, što naglašava mladu radnu snagu. Drugi grafikon prikazuje razine iskustva zaposlenika, pri čemu većina ima između 1 i 10 godina iskustva. Treći grafikon prikazuje distribuciju plaća, pri čemu većina zaposlenika zarađuje između 50.000 i 2.00.000.

4.4. Korelacijska matrica

Korelacijska matrica je pregledna tablica koja predstavlja sve moguće parove korelacija među varijablama unutar skupa podataka. Njezina svrha je pružiti uvid u stupanj povezanosti varijabli ili međusobnog utjecaja. Glavni cilj korelacijske matrice jest otkriti veze među varijablama i razumjeti njihove odnose, što je korisno u prediktivnoj analizi. Analizom korelacijske matrice možemo otkriti visoko korelirane varijable, razumjeti kako promjene u jednoj varijabli utječu na druge te identificirati varijable koje su najviše povezane s ciljnom varijablom.

```
53    # Korelacijska matrica
54    sns.heatmap(salary_df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
```

Slika 23. Stvaranje korelacijske matrice

Slika 24. Korelacijska matrica

4.5. Treniranje modela

Kada govorimo o treniranju modela, to implicira proces prilagodbe modela na osnovi skupa podataka koji uključuje ulazne karakteristike ili atribute, zajedno s odgovarajućim ciljnim vrijednostima ili etiketama. Ovaj proces učenja često uključuje iterativno podešavanje parametara modela kako bi se postigla optimalna prilagodba podacima, a time i poboljšala njegova sposobnost predviđanja ili klasifikacije.

```
56    x_train, x_test, y_train, y_test = train_test_split(test_size=0.25, random_state=42)
```

Slika 25. Prikaz funkcije za treniranje

Svaka metoda koristiti će funkciju *„fit()“* koja se koristi za treniranje modela. Prilikom poziva ove funkcije, model "uči" na podacima iz trening skupa. Varijable „x_train“ i „y_train“ koriste se za treniranje modela strojnog učenja, pri čemu „x_train“ sadrži ulazne značajke za trening, a „y_train“ odgovarajuće stvarne ciljne vrijednosti. Ove varijable pomažu modelu da nauči kako pravilno predviđati ciljne vrijednosti na temelju ulaznih značajki.

4.6. Točnost

Točnost (*eng. accuracy*) je mjera koliko je model strojnog učenja uspješan u ispravnom klasificiranju ili predviđanju ciljne varijable. Visoka točnost ukazuje na dobru sposobnost modela da prepozna obrasce i donese ispravne odluke na novim podacima. Točnost je ključni kriterij za procjenu uspješnosti modela.

```
59    # tocnost
60    y_pred_model=model.predict(x_test)
```

Slika 26. Kod za utvrđivanje točnosti

4.7. Hiperparametri

Hiperparametri su parametri koji kontroliraju ponašanje modela strojnog učenja, a postavljaju se ručno prije početka treninga. Oni igraju ključnu ulogu u prilagodbi modela specifičnim karakteristikama problema i postizanju optimalnih performansi.

Kod predviđanja plaće, hiperparametri su važni jer omogućuju fino podešavanje modela kako bi se postigle najbolje performanse u predviđanju plaće na temelju dostupnih podataka. Na primjer, za algoritme kao što su linearna regresija, stablo odluke ili random forest, hiperparametri poput dubine stabla, broja estimacija ili minimalnog broja uzoraka za razdvajanje mogu utjecati na točnost predikcija plaće. Pravilno postavljanje ovih hiperparametara može pomoći u izbjegavanju prenaučenosti ili podnaučenosti modela te poboljšati sposobnost modela da donese precizne predikcije o visini plaće na temelju različitih karakteristika pojedinca. U konačnici, optimalno podešeni hiperparametri ključni su za stvaranje pouzdanog modela koji može pružiti korisne informacije o očekivanoj visini plaće.

```
58    # Rječnik za podešavanje hiperparametara
59    model_params = {
60        'Linear_Regression': {
61            'model': LinearRegression(),
62            'params': {
63
64            }
65        },
66        'Decision_Tree': {
67            'model': DecisionTreeRegressor(),
68            'params': {
69                'max_depth': [2, 4, 6, 8, 10],
70                'random_state': [0, 42],
71                'min_samples_split': [1, 5, 10, 20]
72            }
73        },
74        'Random_Forest': {
75            'model': RandomForestRegressor(),
76            'params': {
77                'n_estimators': [10, 30, 20, 50, 80]
78            }
79        }
80    }
```

Slika 27. Podešavanje prametara

```
82  # Podešavanje hiperparametara kroz pretraživanje mreže cv
83  score = []
84
85  for model_name, m in model_params.items():
86      clf = GridSearchCV(m['model'], m['params'], cv=5, scoring='neg_mean_squared_error')
87      clf.fit(x_train, y_train)
88
89      score.append({
90          'Model': model_name,
91          'Params': clf.best_params_,
92          'MSE(-ve)': clf.best_score_
93      })
94  pd.DataFrame(score)
```

Slika 28. Postupak optimizacije hiperparametara

4.8. Klasifikacijske metode

Nakon što su formirani skupovi za obuku napravit će se skupovi za testiranje koji će se koristiti u tri metode strojnog učenja koje se uspoređuju. Rezultati modela strojnog učenja ocjenjuju se na temelju točnosti, što predstavlja postotak ispravno predviđenih vrijednosti u odnosu na ukupan broj primjera u testnom skupu podataka. Ova ocjena uspoređuje predviđene vrijednosti modela s stvarnim vrijednostima kako bi se procijenila njegova sposobnost predviđanja.

Varijable „x_test“ i „y_test“ koriste se za testiranje modela, gdje „x_test“ sadrži ulazne značajke za testni skup podataka, a „y_test“ odgovarajuće stvarne ciljne vrijednosti. Ove varijable pomažu u procjeni točnosti i performansi modela na novim, neviđenim podacima. Cilj je minimizirati razliku između predikcija i stvarnih vrijednosti kako bi model naučio pravilno predviđati ciljne vrijednosti na temelju ulaznih značajki. Nakon što je model treniran, evaluira se njegova točnost na testnom skupu podataka kako bi se procijenila njegova sposobnost predviđanja. Što je rezultat metode score() bliži 1, to je model precizniji u predviđanju ciljnih vrijednosti.

```
96   # Definiranje i treniranje modela Random forest
97   model = RandomForestRegressor(n_estimators=20)
98   model.fit(x_train, y_train)
99   model.score(x_test, y_test)
100
101  # Predikcija na testnom skupu
102  y_pred_rfr = model.predict(x_test)
```

```
0.9714296129632989
```

Slika 29. Random forest

```
105  # Definiranje i treniranje modela Decision tree
106  model = DecisionTreeRegressor(max_depth=10, min_samples_split=1, random_state=0)
107  model.fit(x_train, y_train)
108  model.score(x_test, y_test)
109
110  # Predikcija na testnom skupu
111  y_pred_dtr = model.predict(x_test)
```

```
0.9434787076679223
```

Slika 30. Decision tree

```
114    # Linear regression model
115    model = LinearRegression()
116    model.fit(x_train, y_train)
117    model.score(x_test, y_test)
118
119    # Predikcija na testnom skupu
120    y_pred_lr = model.predict(x_test)
```

```
0.8288218169608395
```

Slika 31. Linear regression

Iz slika 25, 26 i 27 očitavaju se točnosti svakog modela:

- Random forest – 0,97
- Decision tree – 0,94
- Linear regression – 0,83

Među tri modela, izgleda da je Random forest model najtočniji među ovim modelima, s točnošću od 97.14%. Model Stabla odlučivanja također pruža dobre rezultate s točnošću od 94.34%. S druge strane, model Linearne regresije ima najnižu ocjenu, što sugerira da možda nije u mogućnosti u potpunosti uhvatiti osnovne obrasce u podacima kao što to čine drugi modeli.

5. ZAKLJUČAK

Evaluacija performansi modela u kontekstu predviđanja plaća igra ključnu ulogu u osiguravanju točnih i pouzdanih procjena visine plaća na temelju dostupnih podataka. Preciznost u predviđanju plaća ima širok spektar implikacija u poslovnom okruženju i životima pojedinaca. Točne procjene plaća osiguravaju pravednije postupke zapošljavanja i plaćanja te pomažu u smanjenju nejednakosti u plaćama. Također, omogućuju tvrtkama bolje upravljanje svojim resursima, uključujući proračun plaća i strategije poticaja. Za zaposlenike, točne procjene plaća mogu povećati njihovo povjerenje u sustav nagrađivanja te potaknuti veću motivaciju i angažman na radnom mjestu.

Korištenjem metoda strojnog učenja, uključujući Random Forest, Decision Tree i Linear Regression, otkriveno je da Random Forest model pokazuje najvišu točnost od 97,14%, dok je Decision Tree model ostvario točnost od 94,34%, a Linear Regression model 83%. Rezultati sugeriraju da Random Forest model ima najbolju sposobnost predviđanja visine plaće na temelju dostupnih značajki, što ga čini preferiranim izborom za predikciju plaća u analizama sličnih skupova podataka. Iako su i Decision Tree i Linear Regression modeli pokazali solidne performanse, Random Forest model se istaknuo kao najpouzdaniji i najprecizniji. Ovi rezultati naglašavaju važnost primjene metoda strojnog učenja u predviđanju plaća i ukazuju na potencijalno široku primjenu Random Forest algoritma u praksi.

LITERATURA

[1] https://github.com/Subash0812/ML-projects/blob/main/Salary_prediction/salary_prediction.ipynbs preuzeto dana 16.02.2024.

[2] Understanding simple linear regression: Predicting salaries using Python, https://medium.com/@dancerworld60/understanding-simple-linear-regression-predicting-salaries-using-python-b6c3bf1f8f4 preuzeto dana 16.02.2024.

[3] Unsupervised machine learning, https://www.javatpoint.com/unsupervised-machine-learning preuzeto dana 16.02.2024.

[4] Supervised machine learning, https://www.javatpoint.com/supervised-machine-learning preuzeto dana 16.02.2024.

[5] Decision tree classification algorithm, https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm preuzeto dana 16.02.2024.

[6] Linear regression in machine learning, https://www.javatpoint.com/linear-regression-in-machine-learning preuzeto dana 16.02.2024.

[7] Random forest algorithm, https://www.javatpoint.com/machine-learning-random-forest-algorithm preuzeto dana 16.02.2024.

[8] What is a decision tree, https://towardsdatascience.com/what-is-a-decision-tree-22975f00f3e1 preuzeto dana 16.02.2024.

[9] Benšić M., Grahovec D., Mihalčić D.: Linearna regresija kroz primjer, 2023.

[10] Valtanen J.: Predicting the Annual Salary Costs of Finnish Companies with Machine Learning, Helsinki, 2021.

[11] Gopal K., Singh A., Kumar H., Sagar S.: Salary prediction using machine learning, India, 2021.

[12] Yanming Chen, Xinlong Li: Salary prediction based on the resume of the candidates, Singapur, 2023.

[13] Mukherjee T., Satyasaivani MS. B.: Employee's salary prediction, Visakhapatnam, 2022.

PRILOZI

I. Baza podataka

```python
# import needed libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error,mean_absolute_error

# ucitavanje csv datoteke
salary_df = pd.read_csv('Salary_Data.csv')
print(salary_df.head())

# informacije o DataFrameu
print(salary_df.info())

# otkrivanje nultih vrijednosti
print(salary_df.isnull().sum())

# izuzimanje nultih vrijednosti
salary_df.dropna(inplace=True)

# opis podataka
print(salary_df.describe())
```

II. Analiza podataka

```
# odnos izmedu dobi zaposlenika i place
plt.figure(figsize=(6, 5))
sns.scatterplot(x='Age',y='Salary', data=salary_df,
                hue='Gender').set(title='Relationship between Age and Salary')
plt.show()

# odnos izmedu iskustva zaposlenika i place
plt.figure(figsize=(6, 5))
(sns.scatterplot(x='Years of Experience', y='Salary', data=salary_df, hue='Gender')
 .set(title='Relationship between Experience and Salary'))
plt.show()

# raspodjela kategorickih varijabli
fig, ax = plt.subplots( nrows: 1,  ncols: 2, figsize=(15, 5))
sns.countplot(x='Gender', data=salary_df, ax=ax[0])
sns.countplot(x='Education Level', data=salary_df, ax=ax[1])

# distribucija atributa u DataFrameu
fig, ax = plt.subplots( nrows: 1,  ncols: 3, figsize=(20, 5))
sns.histplot(salary_df['Age'], ax=ax[0])
sns.histplot(salary_df['Years of Experience'], ax=ax[1])
sns.histplot(salary_df['Salary'], ax=ax[2])

# Korelacijska matrica
sns.heatmap(salary_df.corr(), annot=True)
```

III. Treniranje modela i klasifikacijske metode

```
x_train, x_test, y_train, y_test = train_test_split(test_size=0.25, random_state=42)

# Rječnik za podešavanje hiperparametara
model_params = {
    'Linear_Regression': {
        'model': LinearRegression(),
        'params': {

        }
    },
    'Decision_Tree': {
        'model': DecisionTreeRegressor(),
        'params': {
            'max_depth': [2, 4, 6, 8, 10],
            'random_state': [0, 42],
            'min_samples_split': [1, 5, 10, 20]
        }
    },
    'Random_Forest': {
        'model': RandomForestRegressor(),
        'params': {
            'n_estimators': [10, 30, 20, 50, 80]
        }
    }
}

# Podešavanje hiperparametara kroz pretraživanje mreže cv
score = []
```

```
for model_name, m in model_params.items():
    clf = GridSearchCV(m['model'], m['params'], cv=5, scoring='neg_mean_squared_error')
    clf.fit(x_train, y_train)

    score.append({
        'Model': model_name,
        'Params': clf.best_params_,
        'MSE(-ve)': clf.best_score_
    })
pd.DataFrame(score)

# Definiranje i treniranje modela Random forest
model = RandomForestRegressor(n_estimators=20)
model.fit(x_train, y_train)
model.score(x_test, y_test)

# Predikcija na testnom skupu
y_pred_rfr = model.predict(x_test)


# Definiranje i treniranje modela Decision tree
model = DecisionTreeRegressor(max_depth=10, min_samples_split=1, random_state=0)
model.fit(x_train, y_train)
model.score(x_test, y_test)

# Predikcija na testnom skupu
y_pred_dtr = model.predict(x_test)


# Linear regression model
model = LinearRegression()
model.fit(x_train, y_train)
model.score(x_test, y_test)

# Predikcija na testnom skupu
y_pred_lr = model.predict(x_test)
```