AKADEMIJA STRUKOVNIH STUDIJA ŠUMADIJA ODSEK

TRSTENIK

Predmet: Mašinsko učenje

Tema: Poređenje Random Forest i Decision Tree algoritma na istom datasetu.

Profesor: Marija Mojsilović

Student: Lazar Birtaševic M17/25

Trstenik 2026

SADRŽAJ

1. Uvod

Mašinsko učenje predstavlja granu veštačke inteligencije koja se bavi razvojem algoritama i statističkih modela koji omogućavaju računarskim sistemima da uče iz podataka i donose odluke bez eksplicitnog programiranja za svaku specifičnu situaciju. U današnje vreme, mašinsko učenje ima široku primenu u raznim oblastima kao što su medicina, finansije, marketing, prepoznavanje slika, obrada prirodnog jezika i mnoge druge.

Jedan od bitnijih zadataka u mašinskom učenju jeste klasifikacija, odnosno svrstavanje podataka u unapred definisane kategorije na osnovu njihovih karakteristika. Za rešavanje problema klasifikacije razvijeni su brojni algoritmi, među kojima se posebno ističu Decision Tree (stablo odlučivanja) i Random Forest (nasumična šuma).

Cilj ovog seminarskog rada jeste praktična implementacija i poređenje ova dva popularna algoritma za klasifikaciju na dobro poznatom Iris datasetu. Kroz ovaj rad biće demonstrirano kako se pristupa problemu klasifikacije, od pripreme razvojnog okruženja, preko učitavanja i obrade podataka, do treniranja modela i evaluacije njihovih performansi. Posebna pažnja biće posvećena analizi rezultata i vizualizaciji, kako bi se jasno prikazale prednosti i nedostaci svakog od algoritama.

Rad je organizovan tako da najpre pruži teorijsku osnovu o korišćenim algoritmima, zatim detaljno opisuje korišćeni dataset, a potom korak po korak prikazuje celokupan proces implementacije. Na kraju, rezultati su analizirani i izvedeni su odgovarajući zaključci.

2. Teorijska osnova

2.1 Decision Tree

Decision Tree ili stablo odlučivanja predstavlja jedan od najintuitivnijih i najlakše interpretabilnih algoritama mašinskog učenja. Ovaj algoritam kreira model u obliku stabla, gde svaki unutrašnji čvor predstavlja test nad nekim atributom, svaka grana predstavlja ishod tog testa, a svaki list (terminalni čvor) predstavlja klasnu oznaku ili konačnu odluku.

Princip rada Decision Tree algoritma zasniva se na rekurzivnom particionisanju prostora atributa. Algoritam počinje od korena stabla i na svakom nivou bira atribut koji najbolje razdvaja podatke prema ciljnoj varijabli. Za merenje kvaliteta razdvajanja najčešće se koriste sledeće mere:

Gini indeks nečistoće - meri verovatnoću pogrešne klasifikacije nasumično izabranog elementa ako bi bio klasifikovan prema distribuciji klasa u tom čvoru. Gini indeks se računa po formuli:

$$\text{Gini} = 1 - \sum_{i=1}^{n} \pi^2$$

Gde je π verovatnoća da element pripada klasi i.

Entropija - meri količinu informacije (ili nesigurnosti) u skupu podataka. Računa se po formuli:

$$\text{Entropija} = -\sum_{i=1}^{n} \pi^2 log2(\pi)$$

Information Gain - predstavlja smanjenje entropije nakon razdvajanja skupa podataka prema određenom atributu.

Prednosti Decision Tree algoritma uključuju:

- Jednostavnost za razumevanje i interpretaciju
- Mogućnost vizualizacije stabla
- Ne zahteva normalizaciju podataka
- Može raditi sa numeričkim i kategoričkim podacima

Nedostaci Decision Tree algoritma su:

- Sklonost preprilagođavanju (overfitting)
- Nestabilnost - male promene u podacima mogu dovesti do potpuno drugačijeg stabla
- Može kreirati pristrasna stabla ako su neke klase dominantne

2.2 Random Forest

Random Forest predstavlja ansambl metodu koja kombinuje više stabala odlučivanja kako bi postigla bolje performanse i veću robusnost. Ovaj algoritam je razvio Leo Breiman 2001. godine i od tada je postao jedan od najpopularnijih algoritama za klasifikaciju i regresiju.

Princip rada Random Forest algoritma zasniva se na dve ključne tehnike:

Bagging (Bootstrap Aggregating) - svako stablo u šumi trenira se na različitom bootstrap uzorku originalnog skupa podataka. Bootstrap uzorak se kreira nasumičnim izborom uzoraka sa ponavljanjem iz originalnog skupa, pri čemu je veličina bootstrap uzorka jednaka veličini originalnog skupa. Na ovaj način svako stablo vidi malo drugačiju verziju podataka.

Nasumični izbor atributa - prilikom svakog razdvajanja čvora, algoritam ne razmatra sve atribute, već samo nasumično izabrani podskup atributa. Ovo dodatno povećava raznolikost među stablima u šumi.

Konačna predikcija Random Forest algoritma dobija se glasanjem (voting) svih stabala u šumi. Za klasifikaciju, klasa sa najviše glasova postaje konačna predikcija.

Prednosti Random Forest algoritma:

- Značajno smanjena sklonost preprilagođavanju u odnosu na pojedinačno stablo
- Veća stabilnost i robusnost
- Može proceniti važnost atributa
- Dobro radi sa velikim brojem atributa
- Može raditi sa nedostajućim vrednostima

Nedostaci Random Forest algoritma:

- Manja interpretabilnost u odnosu na pojedinačno stablo
- Veća računska složenost
- Zahteva više memorije za skladištenje svih stabala

3. Opis dataseta

Za potrebe ovog seminarskog rada korišćen je Iris dataset, jedan od najpoznatijih i najčešće korišćenih skupova podataka u oblasti mašinskog učenja. Ovaj dataset je kreirao britanski statisticar i biolog Ronald Fisher 1936. godine u svom radu "The use of multiple measurements in taxonomic problems".

Iris dataset sadrži merenja 150 cvetova irisa, pri čemu svaki cvet pripada jednoj od tri vrste:

1. Iris setosa - 50 uzoraka
2. Iris versicolor - 50 uzoraka
3. Iris virginica - 50 uzoraka

Za svaki cvet izmerena su četiri numerička atributa:

1. SepalLengthCm - dužina čašičnog listića u centimetrima
2. SepalWidthCm - širina čašičnog listića u centimetrima
3. PetalLengthCm - dužina kruničnog listića u centimetrima
4. PetalWidthCm - širina kruničnog listića u centimetrima

Ovaj dataset je idealan za demonstraciju algoritama klasifikacije iz nekoliko razloga, relativno je mali i jednostavan za razumevanje, klase su balansirane (jednak broj uzoraka u svakoj klasi), nema nedostajućih vrednosti, atributi su numerički i ne zahtevaju složeno pretprocesiranje, Iris setosa je linearno separabilna od ostale dve vrste, dok su Iris versicolor i Iris virginica delimično preklapajuće, što predstavlja interesantan izazov za klasifikaciju

4. Priprema razvojnog okruženja

4.1 Preuzimanje dataseta

Prvi korak u realizaciji projekta bio je preuzimanje Iris dataseta. Dataset je preuzet sa platforme Kaggle, koja predstavlja jednu od najvećih online zajednica za nauku o podacima i mašinsko učenje. Kaggle nudi ogroman broj javno dostupnih skupova podataka koji se mogu besplatno preuzeti i koristiti u edukativne i istraživačke svrhe.

Dataset je preuzet u CSV (Comma-Separated Values) formatu, koji predstavlja jednostavan tekstualni format za skladištenje tabelarnih podataka. CSV format je izabran jer je:

- Univerzalno podržan od strane raznih alata i programskih jezika
- Lako čitljiv i za ljude i za računare
- Kompaktan i jednostavan za manipulaciju

Preuzeta datoteka Iris.csv sadrži 151 red (1 zaglavlje + 150 uzoraka) i 6 kolona (Id, SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm, Species).

4.2 Kreiranje virtuelnog okruženja (venv)

Pre početka pisanja koda, kreirao sam virtuelno okruženje (virtual environment) za projekat. Virtuelno okruženje u Python-u predstavlja izolovano okruženje koje omogućava instalaciju paketa specifičnih za dati projekat, bez uticaja na globalni Python sistem ili druge projekte.

Korišćenje virtuelnog okruženja ima brojne prednosti:

Izolacija zavisnosti - svaki projekat može imati svoje verzije biblioteka, nezavisno od drugih projekata. Na primer, jedan projekat može koristiti scikit-learn verziju 1.0, dok drugi koristi verziju 0.24, bez ikakvih konflikata.

Reproduktivnost - virtuelno okruženje omogućava da se tačno dokumentuju sve zavisnosti projekta, što olakšava pokretanje projekta na drugim računarima ili od strane drugih korisnika.

Čist globalni prostor - instaliranje paketa u virtuelno okruženje ne zagađuje globalni Python prostor, što smanjuje rizik od konflikata i problema sa kompatibilnošću.

Jednostavno upravljanje - virtuelno okruženje se može lako kreirati, aktivirati, deaktivirati i obrisati.

Za kreiranje virtuelnog okruženja korišćen je modul venv koji dolazi kao deo standardne Python biblioteke. Kreiranje okruženja izvršeno je sledećom komandom u terminalu:

```
python -m venv venv
```

Ova komanda kreira novi folder pod imenom venv u kome se nalazi kompletna kopija Python interpretera i potrebnih alata za upravljanje paketima.

Nakon kreiranja, virtuelno okruženje je aktivirano komandom:

```
source venv/bin/activate
```

Nakon aktivacije, sve naredne instalacije paketa biće izvršene unutar ovog izolovanog okruženja.

4.3 Instalacija potrebnih biblioteka

Za realizaciju projekta bilo je potrebno instalirati nekoliko Python biblioteka koje pružaju funkcionalnosti za rad sa podacima, mašinsko učenje i vizualizaciju. Instalacija je izvršena pomoću pip alata, koji predstavlja standardni upravitelj paketa za Python.

pandas - biblioteka za manipulaciju i analizu podataka. Pruža strukture podataka kao što su DataFrame i Series, koje omogućavaju efikasan rad sa tabelarnim podacima. Instalirana komandom:

```
pip install pandas
```

numpy - fundamentalna biblioteka za numeričko računanje u Python-u. Pruža podršku za višedimenzionalne nizove i matrice, kao i veliki broj matematičkih funkcija. Instalirana komandom:

```
pip install numpy
```

matplotlib - biblioteka za kreiranje statičkih, animiranih i interaktivnih vizualizacija. Predstavlja osnovu za većinu vizualizacija u Python ekosistemu. Instalirana komandom:

```
pip install matplotlib
```

seaborn - biblioteka za statističku vizualizaciju podataka, izgrađena na matplotlib-u. Pruža viši nivo apstrakcije i lepše podrazumevane stilove. Instalirana komandom:

```
pip install seaborn
```

scikit-learn - najpopularnija biblioteka za mašinsko učenje u Python-u. Pruža implementacije brojnih algoritama za klasifikaciju, regresiju, klasterovanje, kao i alate za preprocesiranje podataka i evaluaciju modela. Instalirana komandom:

```
pip install scikit-learn
```

5. Implementacija

5.1 Učitavanje i pregled podataka

Nakon pripreme razvojnog okruženja, pristupilo se pisanju Python koda. Na početku su uvezene sve potrebne biblioteke:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.preprocessing import LabelEncoder
```

Svaki od ovih import-a ima svoju specifičnu namenu:

- **pandas** se koristi za učitavanje i manipulaciju tabelarnim podacima
- **numpy** pruža podršku za numeričke operacije
- **matplotlib.pyplot** i seaborn koriste se za vizualizaciju
- **os** omogućava rad sa fajl sistemom (kreiranje foldera za slike)
- Iz **sklearn** su uvezeni modeli, metrike i pomoćne funkcije

Zatim je kreiran folder za čuvanje generisanih grafika:

```
SLIKE_FOLDER = 'slike'
if not os.path.exists(SLIKE_FOLDER):
    os.makedirs(SLIKE_FOLDER)
```

Dataset je učitan korišćenjem pandas funkcije read_csv:

```
df = pd.read_csv('Iris.csv')
```

Nakon učitavanja, izvršen je osnovni pregled dataseta koji je pokazao:

- Broj uzoraka: 150
- Broj atributa: 4 (bez kolona Id i Species)
- Nedostajuće vrednosti: 0
- Distribucija klasa je ujednačena sa po 50 uzoraka za svaku vrstu irisa

Ovi podaci potvrđuju da je dataset čist i spreman za dalju obradu bez potrebe za dodatnim preprocesiranjem ili čišćenjem podataka.

5.2 Priprema podataka

Priprema podataka za treniranje modela obuhvatila je nekoliko koraka:

Definisanje atributa i ciljne varijable:

```
atributi = ['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']

X = df[atributi]

y = df['Species']
```

Promenljiva X sadrži matricu atributa (features) dimenzija 150x4, dok promenljiva y sadrži ciljnu varijablu (target), odnosno vrste irisa.

Enkodiranje klasa:

```
encoder = LabelEncoder()
y = encoder.fit_transform(y)
```

LabelEncoder transformiše tekstualne oznake klasa u numeričke vrednosti:

Iris-setosa → 0

Iris-versicolor → 1

Iris-virginica → 2

Ova transformacija je neophodna jer algoritmi mašinskog učenja rade sa numeričkim vrednostima.

Podela na trening i test skup:

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

Dataset je podeljen na trening skup (80% podataka - 120 uzoraka) i test skup (20% podataka - 30 uzoraka). Parametar stratify=y obezbeđuje da distribucija klasa bude očuvana u oba skupa, odnosno da odnos između klasa u trening i test skupu bude isti kao u originalnom datasetu. Parametar random_state=42 obezbeđuje reproduktivnost rezultata - svaki put kada se kod pokrene, podela će biti identična.

5.3 Treniranje modela

Decision Tree model:

```
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
y_pred_dt = dt_model.predict(X_test)
```

Decision Tree model je kreiran sa podrazumevanim parametrima, što znači da koristi Gini indeks kao kriterijum za razdvajanje i da nema ograničenje dubine stabla. Model je treniran na trening skupu korišćenjem metode fit(), a zatim su generisane predikcije na test skupu korišćenjem metode predict().

Random Forest model:

```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
```

Random Forest model je kreiran sa 100 stabala (n_estimators=100), što predstavlja uobičajenu vrednost koja daje dobar balans između performansi i računske složenosti. Veći broj stabala generalno daje bolje i stabilnije rezultate, ali povećava vreme treniranja i predviđanja.

5.4 Evaluacija modela

Za evaluaciju modela korišćeno je nekoliko metrika i pristupa:

Tačnost na test skupu:

```
acc_dt = accuracy_score(y_test, y_pred_dt)
acc_rf = accuracy_score(y_test, y_pred_rf)
```

Tačnost (accuracy) predstavlja procenat tačno klasifikovanih uzoraka i računa se kao:

$$Tačnost = \frac{Broj\ tačnih\ predikcija}{Ukupan\ broj\ uzroka}$$

Cross-Validation:

```
cv_dt = cross_val_score(dt_model, X, y, cv=5)
cv_rf = cross_val_score(rf_model, X, y, cv=5)
```

5-fold cross-validation predstavlja pouzdaniji način evaluacije modela. Kod ove tehnike, dataset se deli na 5 jednakih delova (foldova). Model se trenira na 4 folda, a testira na preostalom foldu. Ovaj proces se ponavlja 5 puta, tako da svaki fold jednom bude test skup. Konačna ocena je prosek ocena iz svih 5 iteracija.

Cross-validation daje bolju procenu generalizacione sposobnosti modela jer koristi sve podatke i za treniranje i za testiranje, eliminiše zavisnost od jedne specifične podele podataka.

Matrica konfuzije:

```
confusion_matrix(y_test, y_pred_dt)

confusion_matrix(y_test, y_pred_rf)
```

Matrica konfuzije prikazuje broj tačnih i netačnih predikcija za svaku klasu. Dijagonalni elementi predstavljaju tačne predikcije, dok elementi van dijagonale predstavljaju greške.

Važnost atributa:

```
dt_model.feature_importances_

rf_model.feature_importances_
```

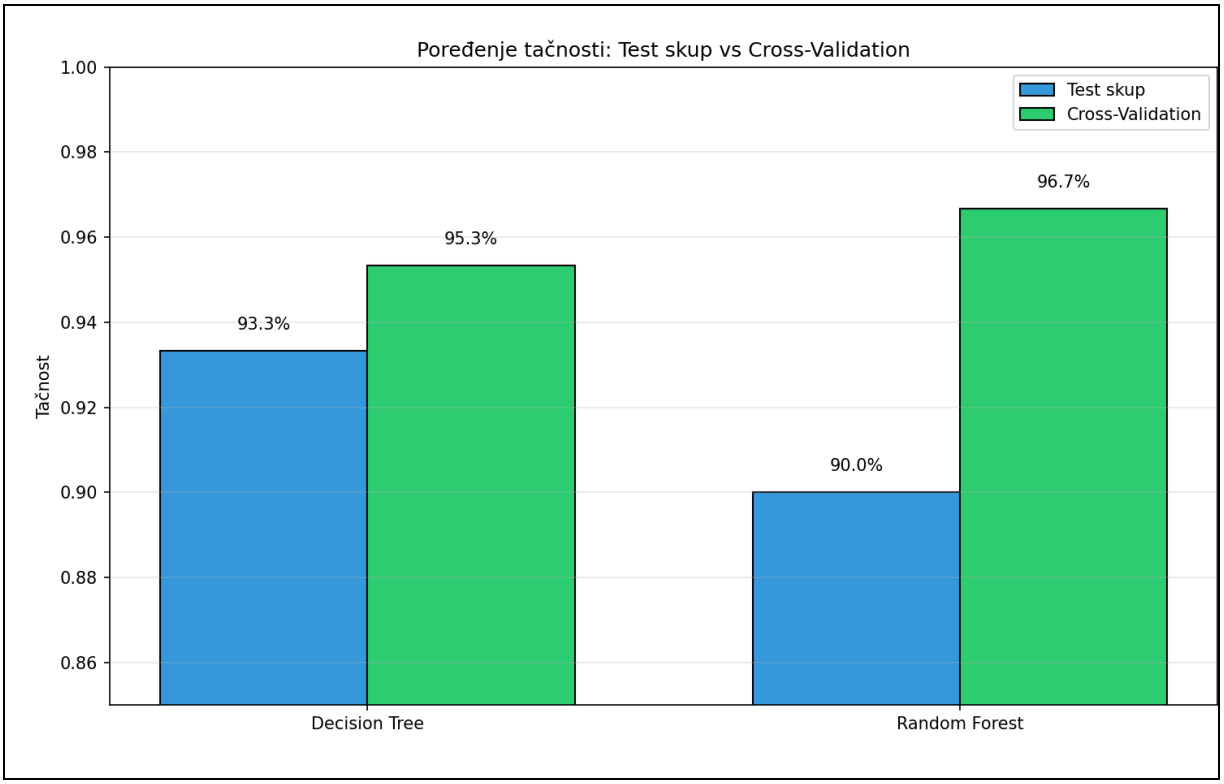
Oba modela automatski računaju važnost svakog atributa za klasifikaciju. Ova informacija je korisna za razumevanje koji atributi najviše doprinose donošenju odluka.

5.5 Vizualizacija rezultata

Za bolje razumevanje rezultata kreirano je šest različitih vizualizacija:

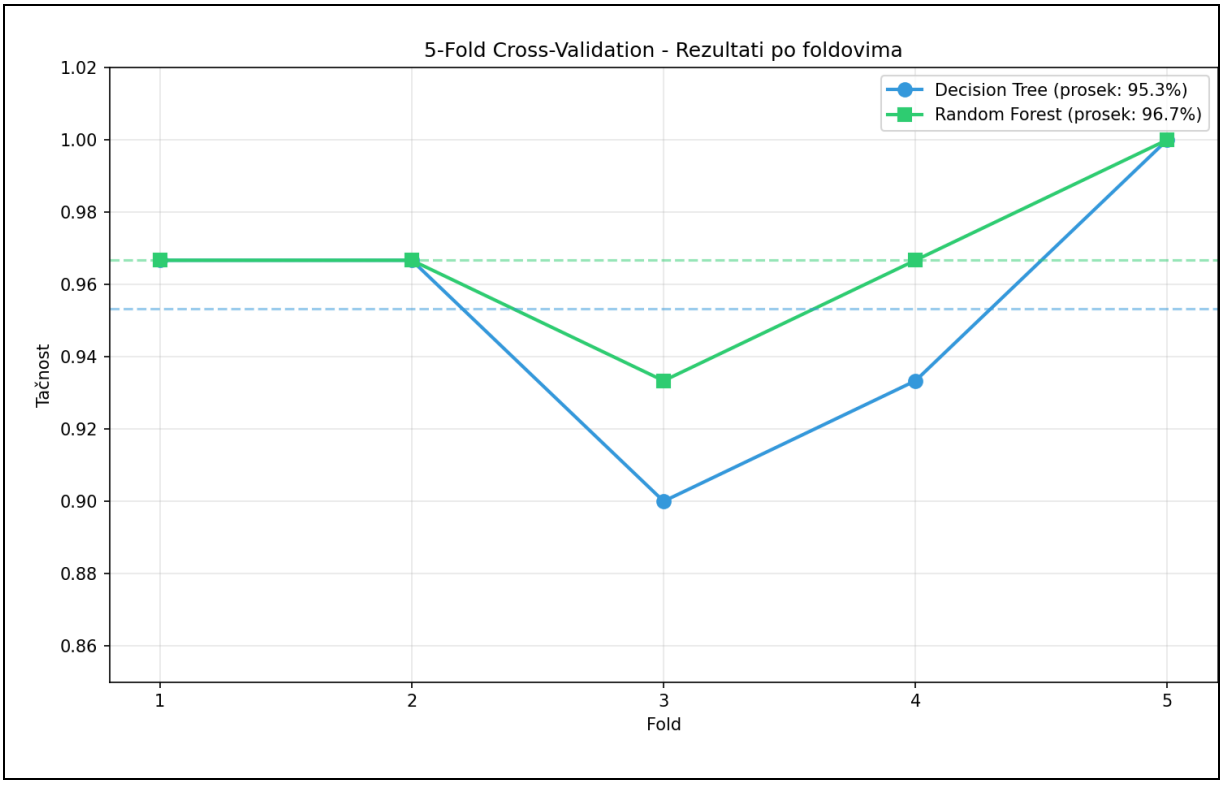
Grafik 1: Poređenje tačnosti

Stupčasti grafik koji prikazuje tačnost oba modela na test skupu i cross-validation. Omogućava brzo vizuelno poređenje performansi.



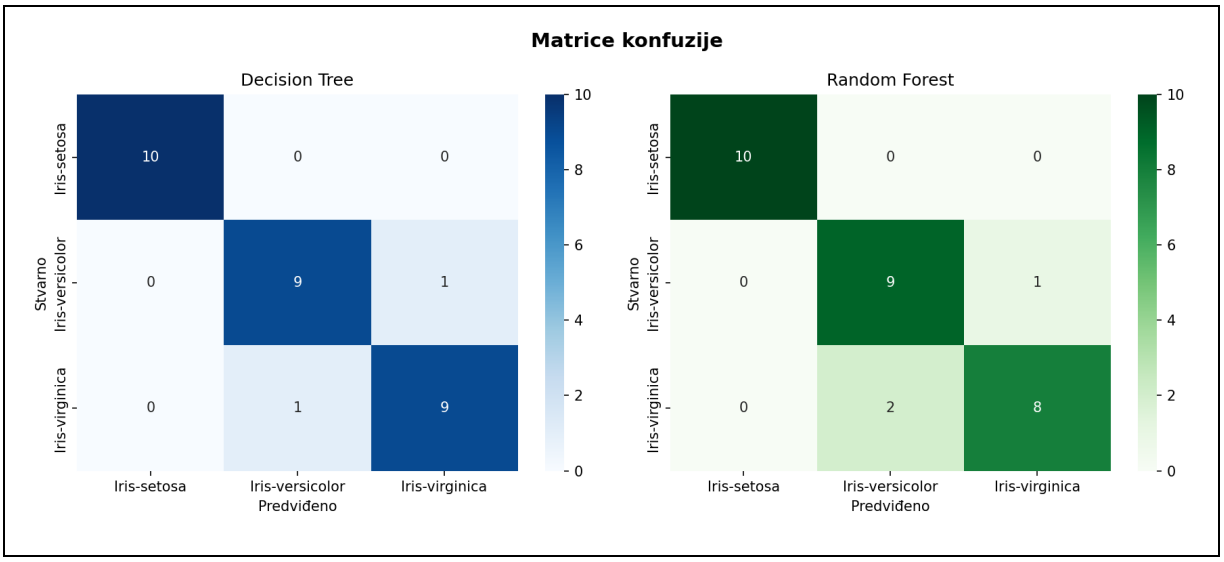
Grafik 2: Cross-Validation rezultati po foldovima

Linijski grafik koji prikazuje tačnost svakog modela u svakom od 5 foldova. Prikazuje i prosečnu vrednost kao horizontalnu isprekidanu liniju. Ovaj grafik pokazuje stabilnost modela - manji varijabilitet između foldova ukazuje na stabilniji model.



Grafik 3: Matrice konfuzije

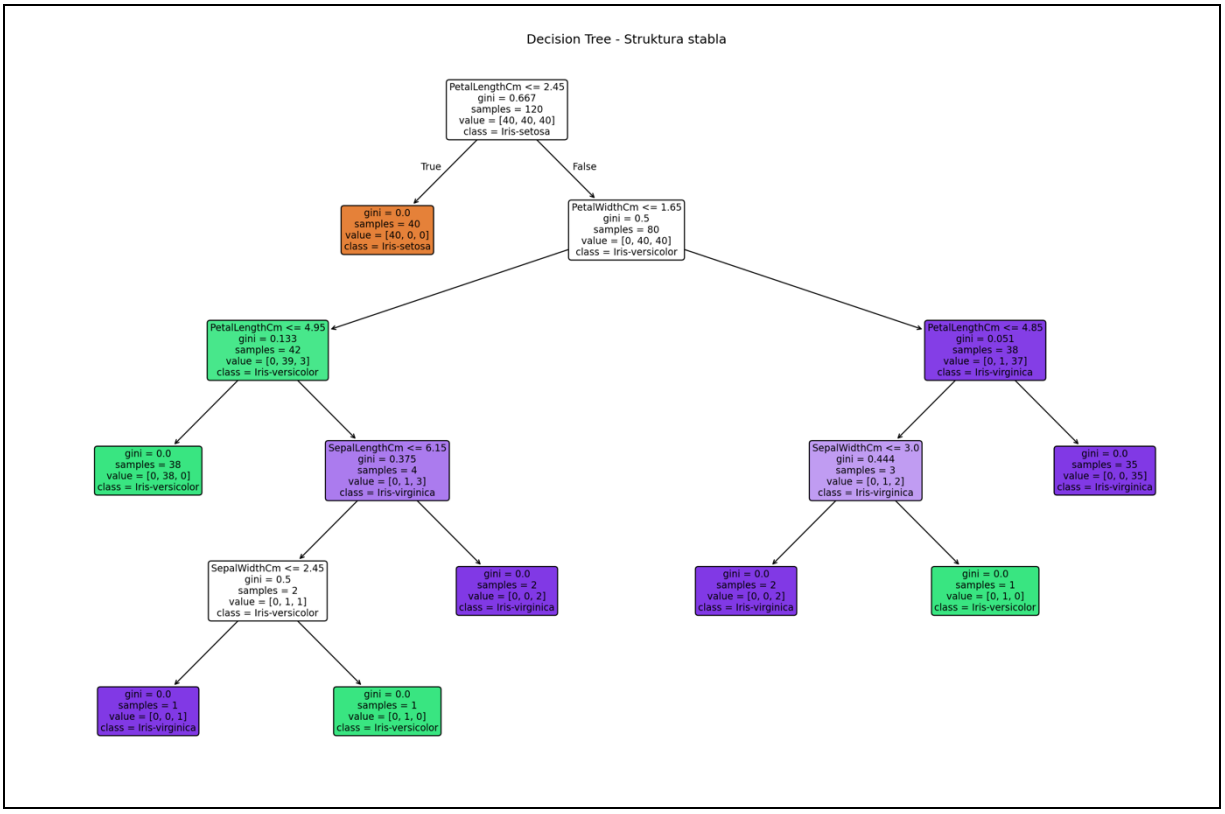
Toplotne mape (heatmaps) koje vizualizuju matrice konfuzije za oba modela. Tamnije boje označavaju veći broj uzoraka. Ovaj prikaz omogućava brzo uočavanje obrazaca grešaka.



Grafik 4: Struktura Decision Tree stable

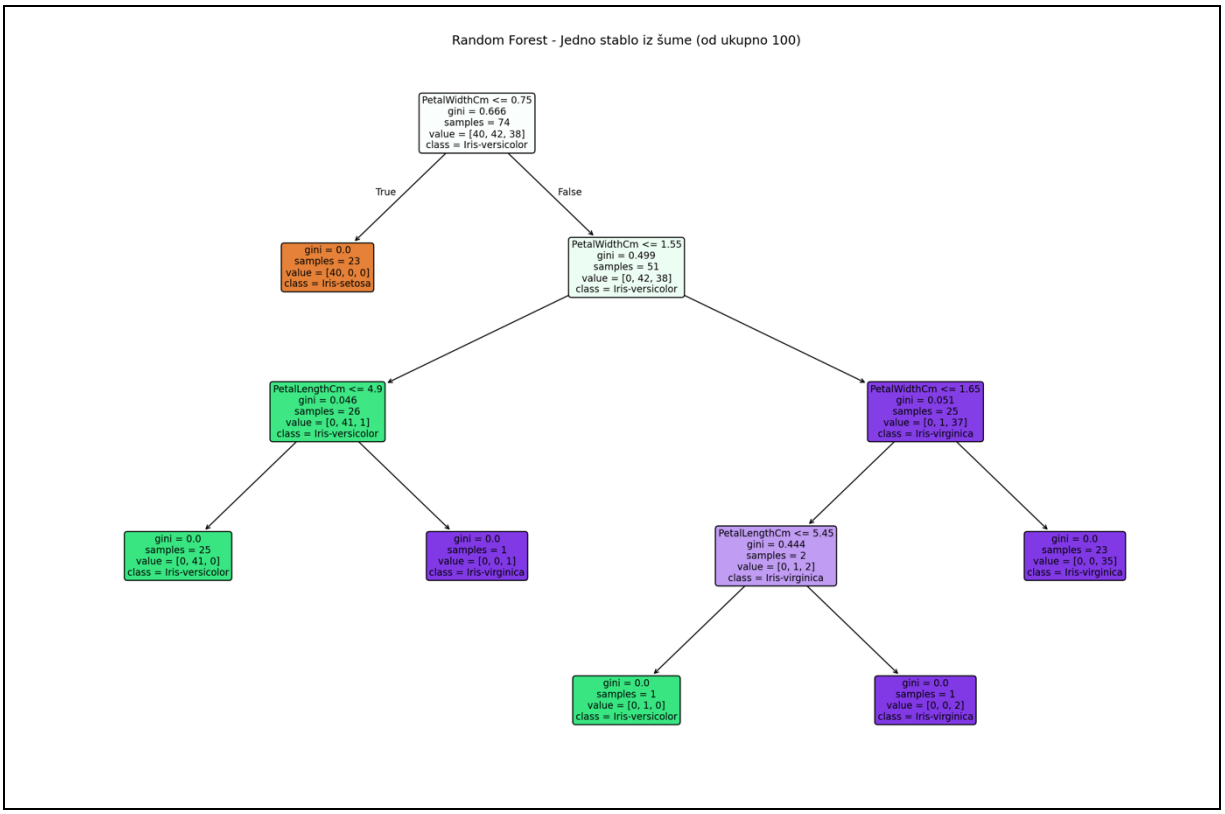
Vizualizacija kompletnog stabla odlučivanja. Za svaki čvor prikazani su:

- Uslov za razdvajanje
- Gini indeks
- Broj uzoraka
- Distribucija klasa
- Dominantna klasa



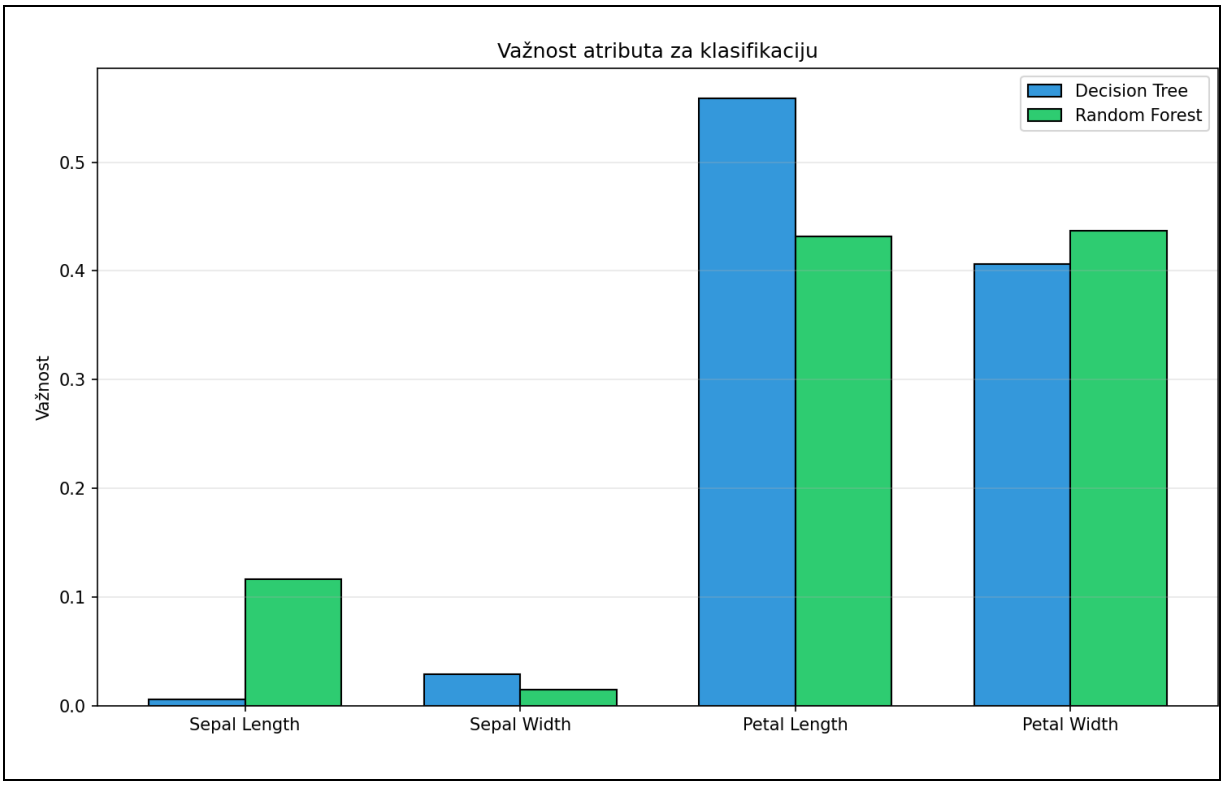
Grafik 5: Jedno stablo iz Random Forest-a

Vizualizacija jednog od 100 stabala koja čine Random Forest. Prikazuje kako pojedinačno stablo donosi odluke, uz napomenu da konačna odluka Random Forest-a zavisi od glasanja svih stabala.



Grafik 6: Važnost atributa

Stupčasti grafik koji upoređuje važnost svakog atributa prema oba modela. Omogućava razumevanje koji atributi najviše utiču na klasifikaciju.



6. Rezultati i analiza

6.1 Tačnost na test skupu

Rezultati evaluacije na test skupu pokazali su sledeće:

- Decision Tree tačnost: 93.33%
- Random Forest tačnost: 90.00%

Interesantno je primetiti da je u ovom konkretnom slučaju Decision Tree postigao bolju tačnost na test skupu od Random Forest-a. Ovo se može objasniti činjenicom da je test skup relativno mali (30 uzoraka) i da specifična podela podataka možda favorizuje jednostavniji model.

6.2 Cross-Validation rezultati

5-fold cross-validation je dao pouzdanije ocene:

- Decision Tree: 95.33% (±3.40%)
- Random Forest: 96.67% (±2.11%)

Ovi rezultati pokazuju da Random Forest ima:

Višu prosečnu tačnost (96.67% naspram 95.33%)

Manju standardnu devijaciju (2.11% naspram 3.40%), što ukazuje na veću stabilnost

Cross-validation rezultati su pouzdaniji pokazatelj stvarnih performansi modela jer koriste sve podatke i eliminišu zavisnost od jedne specifične podele. Rezultati potvrđuju teorijsku prednost Random Forest-a u smislu veće stabilnosti i robusnosti.

6.3 Analiza matrica konfuzije

Decision Tree matrica konfuzije:

```
[[10  0  0]
 [ 0  9  1]
 [ 0  2  8]]
```

Decision Tree je savršeno klasifikovao sve uzorke Iris-setosa. Napravio je jednu grešku klasifikujući Iris-versicolor kao Iris-virginica, i jednu grešku klasifikujući Iris-virginica kao Iris-versicolor. Ukupno 2 greške od 30 uzoraka.

Random Forest matrica konfuzije:

```
[[10  0  0]
 [ 0  9  1]
 [ 0  2  8]]
```

Random Forest je takođe savršeno klasifikovao sve uzorke Iris-setosa. Napravio je jednu grešku klasifikujući Iris-versicolor kao Iris-virginica, i dve greške klasifikujući Iris-virginica kao Iris-versicolor. Ukupno 3 greške od 30 uzoraka.

Obe matrice potvrđuju da je Iris-setosa lako razdvojiva od ostale dve vrste, dok postoji određeno preklapanje između Iris-versicolor i Iris-virginica, što je poznata karakteristika ovog dataseta.

6.4 Analiza važnosti atributa

Atribut	**Decision Tree**	**Random Forrest**
SepalLengthCm	0.006	0.116
SepalWidthCm	0.029	0.015
PetalLengthCm	0.559	0.431
PetalWidthCm	0.406	0.437

Oba modela se slažu da su atributi vezani za latice (Petal) značajno važniji od atributa vezanih za čašične listiće (Sepal). Ovo je u skladu sa biološkom realnošću - vrste irisa se lakše razlikuju po karakteristikama latica.

Primetne razlike između modela:

- Decision Tree daje veću važnost PetalLengthCm (0.559), dok Random Forest ujednačenije vrednuje PetalLengthCm (0.431) i PetalWidthCm (0.437)
- Random Forest dodeljuje značajno veću važnost SepalLengthCm (0.116 naspram 0.006)

Ove razlike proizilaze iz načina na koji modeli računaju važnost atributa. Decision Tree bazira važnost na konkretnim razdvajanjima u jednom stablu, dok Random Forest agregira informacije iz 100 stabala, što daje uravnoteženiju sliku.

6.5 Vizuelna analiza stabala

Vizualizacija Decision Tree stabla pokazuje da model koristi hijerarhijsku strukturu odluka. Prvo razdvajanje se vrši na osnovu PetalLengthCm, što odmah izdvaja Iris-setosa (svi uzorci sa PetalLengthCm ≤ 2.45 pripadaju ovoj vrsti). Dalja razdvajanja koriste kombinaciju PetalWidthCm, PetalLengthCm i SepalLengthCm za razlikovanje Iris-versicolor i Iris-virginica.

Vizualizacija jednog stabla iz Random Forest-a pokazuje sličnu logiku, ali sa različitim pragovima i redosledom razdvajanja. Ovo je očekivano s obzirom da svako stablo u šumi trenira na različitom bootstrap uzorku podataka i razmatra nasumični podskup atributa pri svakom razdvajanju.

7. Zaključak

U ovom seminarskom radu uspešno je izvršeno poređenje Decision Tree i Random Forest algoritama na Iris datasetu. Kroz implementaciju u Python programskom jeziku, koristeći biblioteku scikit-learn, demonstrirani su svi koraci tipičnog projekta mašinskog učenja: od pripreme okruženja i učitavanja podataka, preko treniranja modela, do evaluacije i vizualizacije rezultata.

Ključni zaključci ovog rada su:

1. Oba algoritma postižu visoku tačnost na Iris datasetu, sa cross-validation rezultatima od 95.33% za Decision Tree i 96.67% za Random Forest. Ovo potvrđuje da je Iris dataset relativno jednostavan za klasifikaciju.

2. Random Forest pokazuje veću stabilnost sa manjom standardnom devijacijom u cross-validation rezultatima (2.11% naspram 3.40%). Ovo je u skladu sa teorijskim očekivanjima, s obzirom da ansambl metode generalno daju stabilnije rezultate.

3. Decision Tree pruža bolju interpretabilnost - struktura stabla se može vizualizovati i lako razumeti, dok Random Forest, kao kombinacija 100 stabala, ne pruža tako direktan uvid u proces donošenja odluka.

4. Atributi vezani za latice (Petal) su najvažniji za klasifikaciju vrsta irisa, što su potvrdila oba modela. Ovo je biološki smisleno i pokazuje da modeli uče relevantne obrasce iz podataka.

5. Iris-setosa je lako separabilna od ostale dve vrste, dok postoji određeno preklapanje između Iris-versicolor i Iris-virginica, što se ogleda u greškama klasifikacije oba modela.

Za praktičnu primenu, izbor između ova dva algoritma zavisi od specifičnih zahteva projekta. Ako je prioritet interpretabilnost i razumevanje procesa odlučivanja, Decision Tree je bolji izbor. Ako je prioritet maksimalna tačnost i stabilnost, Random Forest je preporučljiv, uz napomenu da zahteva više računarskih resursa.

Virtuelno okruženje (venv) pokazalo se kao dragocen alat za izolaciju projekta i upravljanje zavisnostima, što je posebno važno u profesionalnom radu gde se često radi sa više projekata koji mogu zahtevati različite verzije biblioteka.

Ovaj rad demonstrira da čak i na relativno jednostavnom datasetu kao što je Iris, postoji prostor za detaljnu analizu i poređenje različitih pristupa mašinskom učenju. Stečena znanja i metodologija mogu se primeniti na složenije probleme klasifikacije u budućem radu.

8. Literatura

1. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python.
2. McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference, 51-56.
3. Scikit-learn dokumentacija. https://scikit-learn.org/stable/documentation.html
4. Pandas dokumentacija. https://pandas.pydata.org/docs/
5. Kaggle - Iris Species Dataset. https://www.kaggle.com/datasets/uciml/iris