

Otkrivanje uljeza u mrežnom prometu

Žada, Frane

Master's thesis / Diplomski rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Electrical Engineering and Computing / Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:168:156535>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2025-03-19**



Repository / Repozitorij:

[FER Repository - University of Zagreb Faculty of Electrical Engineering and Computing repository](#)



SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 501

OTKRIVANJE ULJEZA U MREŽNOM PROMETU

Frane Žada

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 501

OTKRIVANJE ULJEZA U MREŽNOM PROMETU

Frane Žada

Zagreb, lipanj 2024.

DIPLOMSKI ZADATAK br. 501

Pristupnik: **Frane Žada (0036508646)**

Studij: Računarstvo

Profil: Znanost o mrežama

Mentorica: izv. prof. dr. sc. Martina Antičić

Zadatak: **Otkrivanje uljeza u mrežnom prometu**

Opis zadatka:

U današnjem Internetu sve češće su prisutni napadi na komunikacijsku mrežu stoga je potrebno provoditi aktivnosti nadzora i analize mrežnog prometa kako bi se otkrile zlonamjerne aktivnosti u mreži. Klasifikacija mrežnog prometa u svrhu otkrivanja uljeza predstavlja ključan korak detekcije napada i provođenja postupaka za njegovo otklanjanje. Vaš je zadatak napraviti programsko rješenje za detekciju uljeza u mrežnom prometu u programskom jeziku Python. Predloženi sustav treba omogućiti odabir značajki, treniranje modela, optimizaciju parametara i evaluaciju, pri čemu je potrebno usporediti 3 različita algoritma klasifikacije mrežnog prometa. Performance svakog modela potrebno je analizirati koristeći različite metrike kao što su točnost, preciznost, vrijeme treniranja/testiranja i sl. te je na osnovu provedene usporedbe potrebno ocijeniti predložene modele za dani skup podataka.

Rok za predaju rada: 28. lipnja 2024.

Sadržaj

Uvod	1
1. Pregled relevantne literature	3
1.1. Teorijska osnova detekcije uljeza.....	3
1.2. Strojno učenje u detekciji uljeza.....	4
1.3. Izazovi i ograničenja.....	4
1.4. Budući pravci.....	5
2. Metodologija istraživanja	6
2.1. Skup podataka	6
2.1.1. Opis značajki	7
2.1.2. Analiza odabranog skupa podataka	8
2.1.3. Priprema podataka za strojno učenje	9
2.2. Odabrani modeli strojnog učenja.....	11
2.2.1. KNN	12
2.2.2. Logistička regresija	12
2.2.3. Stablo odlučivanja	13
2.2.4. Usporedba KNN-a, LR-a i DT-a	14
2.3. Alati i tehnologije	15
3. Treniranje modela.....	17
3.1. Funkcija za treniranje i evaulaciju modela	17
3.2. KNN	18
3.3. Logistička regresija	19
3.4. Stablo odlučivanja	21
3.5. Usporedba rezultata treniranja/testiranja modela	22
4. Evaluacija	25
4.1. Unakrsna validacija	25

4.2.	Točnost	26
4.3.	Matrica zabune	27
4.4.	Preciznost, odziv i F1-ocjena	31
4.5.	Usporedba binarne i višeklasne klasifikacije.....	35
5.	Rezultati i diskusija	39
5.1.	Analiza i interpretacija rezultata.....	40
5.2.	Izazovi tijekom implementacije i evaluacije modela	40
5.3.	Usporedba s relevantnim radovima u području.....	40
5.4.	Teorijska podloga i implikacije rezultata	41
	Zaključak	42
	Literatura	43
	Sažetak.....	44
	Summary.....	45
	Skraćenice.....	46

Uvod

U suvremenom digitalnom dobu, sigurnost informacijskih sustava postala je temeljna briga organizacija širom svijeta. Kako se mrežni napadi razvijaju u frekvenciji i sofisticiranosti, tako raste i potreba za naprednim metodama detekcije uljeza koji mogu učinkovito prepoznati i odvratiti potencijalne prijetnje. U ovom kontekstu, primjena metoda strojnog učenja u detekciji uljeza predstavlja obećavajući pristup, omogućujući razvoj dinamičnih, adaptivnih sustava sposobnih za identifikaciju složenih obrazaca ponašanja koji signaliziraju neautorizirani pristup ili zlonamjerne aktivnosti unutar mreže. Ovaj diplomski rad bavi se istraživanjem i evaluacijom različitih modela strojnog učenja s ciljem unapređenja efikasnosti detekcijskih sistema uljeza u mrežnom prometu.

Motivacija za odabir ove teme proizlazi iz rastuće svijesti o značaju kibernetičke sigurnosti u zaštiti osjetljivih informacija i infrastrukture. S obzirom na kontinuirani rast kibernetičkih prijetnji, postalo je imperativ razviti pouzdane mehanizme za njihovo prepoznavanje i neutralizaciju. Kroz praktični rad i istraživanja na području kibernetičke sigurnosti, poseban interes je usmjeren na mogućnosti koje strojno učenje nudi u borbi protiv mrežnih napada.

Cilj ovog rada je dvostruki: prvo, provesti analizu postojećih metoda detekcije uljeza u mreži s posebnim fokusom na primjenu strojnog učenja; i drugo, primijeniti i optimizirati odabrane modele strojnog učenja kako bi se postigla poboljšanja u preciznosti i efikasnosti detekcije. Ovo istraživanje uključuje prikupljanje i obradu relevantnih podataka, odabir i treniranje odgovarajućih modela strojnog učenja te analizu njihovih performansi kroz različite metrike.

Rad se susreće s višestrukim izazovima, uključujući one vezane za dostupnost i kvalitetu podataka, kao i komplikacija prilikom interpretacije modela strojnog učenja zbog njihove složenosti. Također, postoji potreba za pažljivim balansiranjem osjetljivosti i specifičnosti modela u cilju optimizacije njegove efikasnosti u detekciji, minimizirajući pritom broj lažno pozitivnih rezultata.

U strukturalnom smislu, diplomski rad je organiziran kako slijedi: nakon uvoda, prvo poglavlje pruža pregled relevantne literature, uključujući teorijsku osnovu detekcije uljeza, primjenu strojnog učenja u detekciji uljeza te izazove, ograničenja i buduće pravce u ovom području. Drugo poglavlje opisuje metodologiju istraživanja, uključujući prikupljanje i pripremu podataka te odabir modela i alata. Treće poglavlje se fokusira na treniranje modela,

dok četvrto poglavlje evaluira performanse modela koristeći unakrsnu validaciju, metrike kao što su točnost, matrica zabune, preciznost, odziv i F1-ocjena, te uspoređuje binarnu i višeklasnu klasifikaciju. Peto poglavlje analizira rezultate, izazove tijekom implementacije, uspoređuje ih s relevantnim istraživanjima te diskutira teoretske implikacije. Zaključak sumira ključne nalaze istraživanja i predlaže smjernice za budući rad na području detekcije uljeza koristeći metode strojnog učenja.

U okviru ovog diplomskog rada, cilj je ne samo razumjeti trenutno stanje u detekciji uljeza pomoću strojnog učenja, već i doprinijeti razvoju efikasnijih i pouzdanijih pristupa u borbi protiv kibernetičkih prijetnji.

1. Pregled relevantne literature

Proučavanje literature ključan je korak u razumijevanju trenutne istraživačke scene u bilo kojem znanstvenom području. U kontekstu detekcije uljeza, koristeći metode strojnog učenja, ovo poglavlje pruža pregled teorijskih osnova i istraživačkih rezultata koji su oblikovali ovu disciplinu. Razumijevanje kako su se razvijale tehnike detekcije i koje su se metode pokazale uspješnima (ili ne) neophodno je za daljnje napredovanje u ovom istraživačkom području.

1.1. Teorijska osnova detekcije uljeza

Prema Pradhanu i sur. (2020), detekcija uljeza može se definirati kao proces identificiranja pokušaja neautoriziranog pristupa, manipulacije ili uništavanja računalnih sustava, mrežnih resursa ili podataka. Prema Agoramoorthyju i suradnicima (2023), sustavi za detekciju uljeza (engl. *Intrusive Detection Systems*, IDS) klasificirani su u dvije glavne kategorije: IDS temeljen na potpisima i IDS temeljen na anomalijama. Prvi se oslanjaju na poznate obrasce zlonamjernih aktivnosti - kao što sugerira i sam naziv; vjerojatnije je da će se visoka učinkovitost postići u izvršavanju oznaka koje nisu nepoznate, ali one bez naznačenih uzoraka obično ne postižu učinkovit rezultat u usporedbi s onim poznatim oznakama. Sustavi temeljeni na anomalijama pak pokušavaju identificirati odstupanja od uobičajenog ponašanja korisnika ili mrežnih aktivnosti. Garcia-Teodoro i sur. (2009) opisuju kako "detekcija temeljena na anomalijama pomoću strojnog učenja može se prilagoditi novim, dosad neviđenim napadima učeći iz ponašanja mreže tijekom vremena." Oba pristupa imaju svoje prednosti i nedostatke, uključujući sposobnost prepoznavanja novih prijetnji, brzinu detekcije i stopu lažno pozitivnih rezultata. Stoga bi se trebao razmotriti sveobuhvatniji pristup tako da se mogu optimizirati i pogreške u obuci i pogreške u testiranju u smislu dvije gore navedene vrste. Konačno, integracijom IDS-a temeljenog na potpisu i IDS-a temeljenog na anomalijama dolazimo do hibridnog pristupa (engl. *hybrid*) koji omogućuje da se IDS-ovi, koji sadrže i ne sadrže oznake, mogu rigorozno testirati.

1.2. Strojno učenje u detekciji uljeza

Primjena strojnog učenja u detekciji uljeza predstavlja napredak u odnosu na tradicionalne metode, omogućavajući razvoj modela koji mogu naučiti iz podataka i poboljšavati svoje performanse s vremenom. Algoritmi strojnog učenja, posebno oni koji pripadaju kategoriji nadziranog i nenadziranog učenja, pokazali su značajan potencijal u identifikaciji složenih obrazaca ponašanja koji ukazuju na zlonamjerne aktivnosti.

Razne studije su istraživale primjenu različitih algoritama u detekciji uljeza, uključujući neuronske mreže, algoritme dubokog učenja, stabla odlučivanja i slučajne šume. Ovi modeli su se pokazali posebno korisnima u prepoznavanju sofisticiranih i do sada neviđenih napada, zahvaljujući njihovoj sposobnosti da nauče iz velikih količina podataka i prilagode se promjenjivim obrascima ponašanja. Primjeri uključivanja naprednih tehnika uključuju radove poput Kim i sur. (2016), koji istražuju upotrebu LSTM mreža (engl. *long short-term memory*) za identifikaciju složenih uzoraka u podacima mrežnog prometa, pokazujući izvanrednu sposobnost ovih modela da se nose s dinamičkim promjenama u ponašanju napadača. Posljednjih godina, s razvojem dubokog učenja, obuka velikog broja podataka kroz duboko učenje može učinkovito poboljšati točnost otkrivanja upada u mrežu. Često korišteni modeli dubinskog učenja uključuju mrežu dubokih uvjerenja (engl. *Deep Belief Network*, DBN), konvolucionarnu neuronsku mrežu (engl. *Convolutional Neural Network*, CNN) i rekurentnu neuronsku mrežu (engl. *Recurrent Neural Network*, RNN). Konkretno, istraživanje S. Zhenga i sur. (2021) naglašava ulogu konvolucijskih neuronskih mreža u detekciji uljeza, ističući njihovu sposobnost učenja karakteristika na različitim razinama iz opsežnih skupova podataka, što ukazuje na njihovu široku primjenjivost i potencijal u poboljšanju sigurnosnih mjera u sve složenijem mrežnom okruženju.

1.3. Izazovi i ograničenja

Unatoč obećavajućim rezultatima, primjena strojnog učenja u detekciji uljeza nije bez izazova. Jedan od ključnih problema je potreba za velikim i raznolikim skupovima podataka koji točno predstavljaju stvarne mrežne uvjete i napade. Osim toga, modeli strojnog učenja mogu biti podložni problemima kao što su prenaučenosť (engl. *overfitting*) i podnaučenosť (engl. *underfitting*), što može utjecati na njihovu generalizaciju i performanse u stvarnom svijetu. C. Lu (2022) ističe važnost nadogradnje metoda detekcije kako bi se adresirali nedostaci postojećih tehnika, posebno u kontekstu velikih i raspodijeljenih napada.

Pored toga, postoji pitanje interpretabilnosti modela. Složeni modeli, poput dubokih neuronskih mreža, mogu biti "crne kutije", čineći teškim razumijevanje kako su došli do određenih zaključaka. Ovo može otežati povjerenje korisnika u sustave detekcije i komplicirati postupak otklanjanja problema.

1.4. Budući pravci

S obzirom na brzi razvoj tehnologija i neprestane promjene u svijetu kibernetičkih prijetnji, kontinuirana istraživanja i inovacije ključni su za napredak u području detekcije uljeza. Budući rad bi mogao uključivati istraživanje novih modela strojnog učenja, poboljšanje kvalitete podataka i razvoj metoda za povećanje transparentnosti i interpretabilnosti modela. Također, važno je naglasiti potrebu za interdisciplinarnim pristupom, gdje stručnjaci iz područja kibernetičke sigurnosti i strojnog učenja surađuju kako bi razvili robusnije i učinkovitije sustave za detekciju uljeza.

Kroz pregled literature, jasno je da primjena strojnog učenja u detekciji uljeza nudi značajne mogućnosti za poboljšanje sigurnosti informacijskih sustava. Međutim, kako bi se ove tehnike učinkovito implementirale i iskoristio njihov puni potencijal, važno je adresirati postojeće izazove i kontinuirano raditi na inovacijama i istraživanjima u ovom dinamičnom polju.

2. Metodologija istraživanja

U ovom poglavlju detaljno je opisana metodologija koja se koristi u istraživanju s ciljem identifikacije i evaluacije učinkovitosti različitih klasifikacijskih algoritama strojnog učenja u detekciji uljeza. Osnovni cilj je primijeniti i optimizirati odabrane modele kako bi se postigla visoka preciznost u detekciji zlonamjernih aktivnosti, minimizirajući pritom broj lažno pozitivnih rezultata. Za postizanje ovog cilja, primijenjen je istraživački pristup opisan u nastavku ovog poglavlja.

2.1. Skup podataka

Za treniranje i testiranje modela korišten je javno dostupan skup podataka za detekciju uljeza - *LUFlow Network Intrusion Detection Data Set* (Ryan, 2024.), koji uključuje različite vrste mrežnih napada kombiniranih s legitimnim mrežnim prometom. LUFlow je skup podataka o mrežnom otkrivanju upada temeljen na protoku, koji sadrži robusne oznake zlonamjernog ponašanja. Telemetrija u LUFlow-u uključuje nove vektore napada prikupljene putem *honeypotova* unutar adresnog prostora Sveučilišta Lancaster. Oznake su dodane automatski i temelje se na čvrstim dokazima dobivenim usporedbom s vanjskim izvorima podataka o kibernetičkim prijetnjama (engl. *Cyber Threat Intelligence*, CTI). To omogućuje stalno snimanje, označavanje i objavljivanje telemetrije. Tokovi koji nisu identificirani kao zlonamjerni, ali odstupaju od normalnog profila telemetrije, označeni su kao izuzeci (engl. *outlier*) radi daljnje analize. Normalni promet, poput ssh i prometa baza podataka (engl. *database traffic*), također je uključen. Skup podataka kontinuirano se ažurira zahvaljujući automatskom označavanju, omogućujući praćenje novih obrazaca napada. Repozitorij je strukturiran prema godini i mjesecu snimanja telemetrije, npr. telemetrija iz rujna 2020. nalazi se u mapi 2020/09. Podaci za treniranje i testiranje učitavaju se iz CSV datoteka te su podvrgnuti određenim pretprocesiranjima kako bi se osigurala njihova kvaliteta i relevantnost za istraživačke potrebe. Proces pretprocesiranja uključuje čišćenje podataka, normalizaciju i selekciju značajki, čime se poboljšava učinkovitost i točnost konačnih modela.

2.1.1. Opis značajki

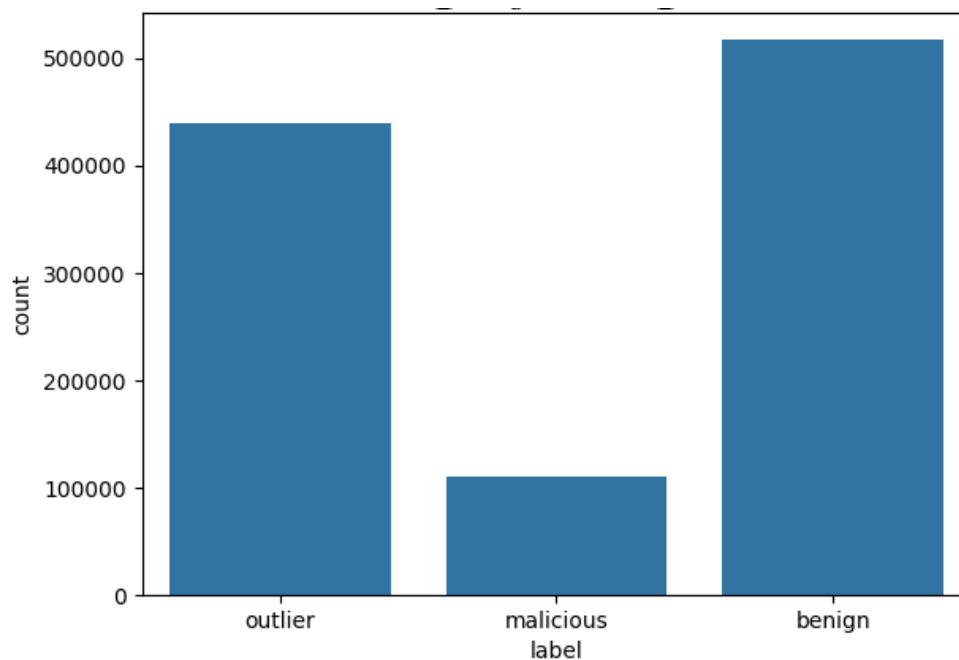
Telemetrijski podaci se prikupljaju putem Ciscovog alata *Joy*, koji zabilježava različita mjerenja vezana uz protok. Na temelju tih mjerenja projektirane su značajke, koje su detaljno opisane u nastavku (Tablica 2.1):

Tablica 2.1 Opis značajki

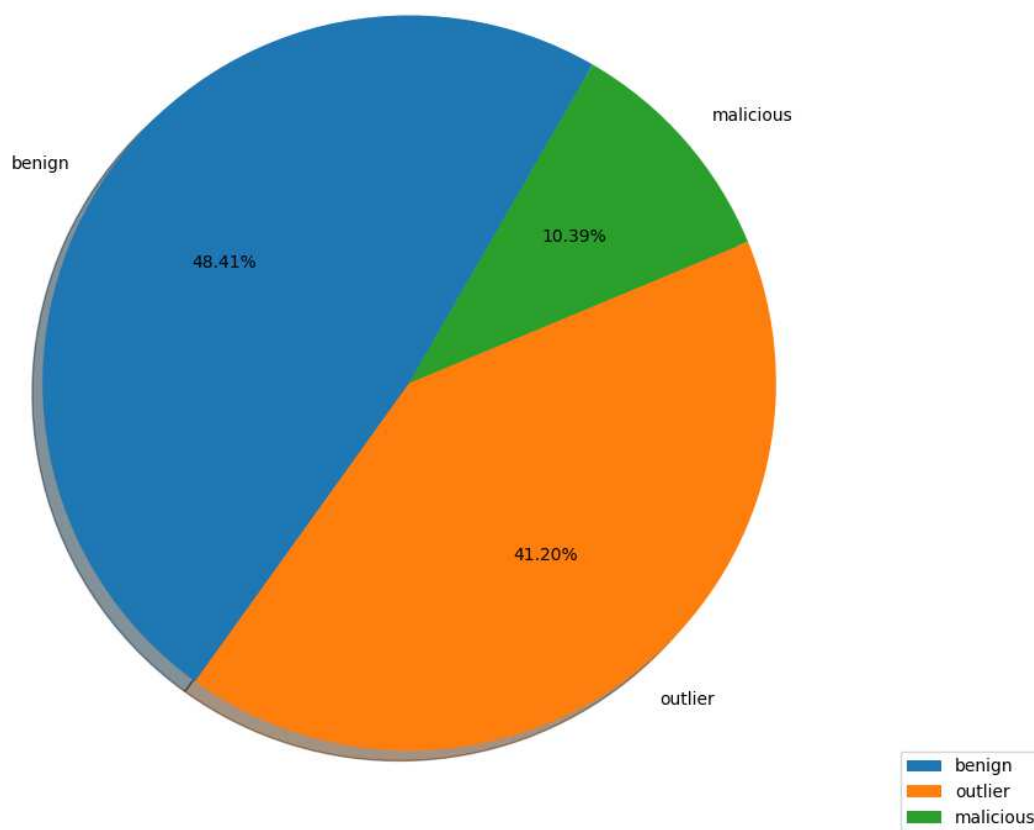
Ime	Opis	Tip podatka
src_ip	izvorna IP adresa	int64
src_port	broj izvornog porta	float64
dest_ip	odredišna IP adresa	int64
dest_port	broj odredišnog porta	float64
protocol	broj protokola, primjerice TCP je 6	int64
bytes_in	broj bajtova prenesenih od izvora do odredišta	int64
bytes_out	broj bajtova prenesenih od odredišta do izvora	int64
num_pkts_in	broj paketa od izvora do odredišta	int64
num_pkts_out	broj paketa od odredišta do izvora	int64
entropy	entropija u bitovima po bajtu podatkovnih polja	float64
total_entropy	ukupna entropija u bajtovima preko svih bajtova u podatkovnim poljima	float64
avg_ipt	srednja vrijednost vremena dolaska između paketa	float64
time_start	vrijeme početka toka u sekundama od epohe	int64
time_end	vrijeme završetka toka u sekundama od epohe	int64
duration	vrijeme trajanja protoka, s preciznošću u mikrosekundi	float64
label	oznaka toka – benigni (engl. <i>benign</i>), izuzetni (engl. <i>outlier</i>) ili zlonamjerni (engl. <i>malicious</i>)	object

2.1.2. Analiza odabranog skupa podataka

Analiza uključuje identificiranje naziva stupaca i njihovih tipova podataka, bilježenje dimenzija skupa podataka, provjeru postojanja praznih vrijednosti te analizu distribucije tipova mrežnih napada unutar skupa podataka. Ti koraci su ključni za razumijevanje strukture. Za ovo istraživanje korišteni su podaci iz svih CSV datoteka u 2022. godini. Navedeni podskup sastoji se od 1068376 redaka i 16 stupaca. Tijekom analize pronađen je određen broj redaka gdje nedostaju podaci te je umjesto praznih vrijednosti unesena 0 – *zero imputation*. Raspodjela napada (oznaka toka – engl. *label*) prikazana je u nastavku (Slika 2.1 i Slika 2.2) :



Slika 2.1 Raspodjela napada



Slika 2.2 Raspodjela svakog napada izražena u postocima

2.1.3. Priprema podataka za strojno učenje

U ovom potpoglavlju opisani su koraci za enkodiranje kategorijskih varijabli, selekciju značajki i pripremu podataka za treniranje modela strojnog učenja. Kroz ovaj proces, korištene su različite tehnike i biblioteke iz područja strojnog učenja kako bi optimizirali model i poboljšali njegove performanse. Kategorijske varijable su one koje sadrže diskretne vrijednosti, poput naziva gradova, boja ili kategorija proizvoda. Većina algoritama strojnog učenja zahtijeva numeričke ulazne podatke, stoga je potrebno kategorijske varijable pretvoriti u numerički format. Jedan od načina za to je korištenje *LabelEncoder*-a iz biblioteke *sklearn* te je kod prikazan u nastavku (Kod 2.1):

```
from sklearn.preprocessing import LabelEncoder

# Definicija funkcije za enkodiranje kategorijskih varijabli
def le(df):
    for col in df.columns:
```



```

if df[col].dtype == 'object':
    label_encoder = LabelEncoder()
    df[col] = label_encoder.fit_transform(df[col])

# Enkodiranje kategorijskih varijabli u skupu podataka
le(multiclass_data)

```

Kod 2.1 Enkodiranje kategorijskih varijabli

Sljedeći korak je odvajanje značajki (X) i ciljne varijable (Y) iz skupa podataka. Ciljna varijabla je ono što želimo predvidjeti, dok su značajke atributi koji će nam pomoći u predikciji (Kod 2.2):

```

# Odvajanje značajki i ciljne varijable iz skupa podataka za
treniranje
X_train = multiclass_data.drop(['label'], axis=1)
Y_train = multiclass_data['label']

```

Kod 2.2 Odvajanje značajki i ciljne varijable

Korištenjem tehnike rekurzivne eliminacije značajki (engl. *Recursive Feature Elimination*, RFE) s klasifikatorom *RandomForestClassifier*, odabrane su najvažnije značajke koje najviše doprinose predikciji (Kod 2.3):

```

# Inicijalizacija RandomForestClassifier-a
rfc = RandomForestClassifier()

# Korištenje RFE s RFC za odabir top 10 značajki
rfe = RFE(rfc, n_features_to_select=10)
rfe = rfe.fit(X_train, Y_train)

```

Kod 2.3 Selekcija najvažnijih značajki

RFE se ovdje koristi kao novi način ponovnog ispitivanja kvalitete prethodne metode odabira značajki (RFC-a). Ova metoda iterativno razmatra manje skupove uzoraka uz pomoć vanjskog procjenitelja koji dodjeljuje težine svakoj značajki odabranoj u svakom krugu. Prije početka sljedećeg kruga, značajka s najmanjom važnosti se uklanja. Proces se ponavlja dok se ne postigne željeni broj značajki. Selekcijom je odabrano sljedećih 10 značajki: *avg_ip_t*, *bytes_in*, *bytes_out*, *dest_port*, *num_pkts_in*, *src_port*, *time_end*, *time_start*, *total_entropy*, *duration*. Nakon selekcije skup podataka za treniranje reduciran je na te odabrane značajke, a potom su značajke standardizirane korištenjem *StandardScaler*-a kojim su uklonjene srednje vrijednosti i značajke su skalirane na jedinicu varijance. Ovaj

korak je ključan za mnoge algoritme strojnog učenja koji su osjetljivi na skale značajki. Kod je prikazan u nastavku (Kod 2.4):

```
from sklearn.preprocessing import StandardScaler

# Standardizacija značajki
scale = StandardScaler()
X_train = scale.fit_transform(X_train)
```

Kod 2.4 Standardizacija značajki

Na kraju, skup podataka podijeljen je na skupove za treniranje i testiranje. To omogućuje procjenu performansi modela na podacima koje model nije vidio tijekom treniranja (Kod 2.5):

```
from sklearn.model_selection import train_test_split

# Podjela skupa podataka na skupove za treniranje i
testiranje
x_train, x_test, y_train, y_test = train_test_split(X_train,
Y_train, train_size=0.70, random_state=2)

# Ispis oblika podijeljenih skupova podataka za provjeru
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

Kod 2.5 Podjela skupa podataka za treniranje/testiranje

Skup podataka podijeljen je na način da je 70 % podataka izdvojen za treniranje, a preostalih 30 % za testiranje. Tako skup podataka za treniranje sadrži 747863 redaka, dok skup za testiranje obuhvaća 320513 redaka.

2.2. Odabrani modeli strojnog učenja

Istraživanje obuhvaća nekoliko različitih modela strojnog učenja, uključujući K-najbližih susjeda (engl. *k-nearest neighbors*, KNN), logističku regresiju (engl. *Logistic Regression*, LR), i stablo odlučivanja (engl. *Decision Tree*, DT). Za svaki od modela provedena je analiza kako bi se identificirale prednosti i nedostaci u kontekstu detekcije uljeza (Kathiresan i suradnici (2022)).

2.2.1. KNN

KNN je jednostavan algoritam temeljen na instancama koji se koristi za klasifikaciju i regresiju. Model radi tako da za klasifikaciju pronalazi k najbližih točaka (susjeda) točki od interesa i dodjeljuje najčešću klasu među tim susjedima novoj točki.

- Izračunavanje udaljenosti - algoritam izračunava udaljenost između točke koju treba klasificirati i svih ostalih točaka u skupu podataka za treniranje. Uobičajene metričke udaljenosti uključuju Euklidsku, Manhattansku i Minkowskijevu udaljenost.
- Identifikacija susjeda - odabire k točaka iz skupa za treniranje koje su najbliže novoj točki.
- Glasovanje - algoritam dodjeljuje novu točku klasi koja je najčešća između k susjeda.

Prednosti:

- Jednostavan za razumijevanje i implementaciju.
- Nema faze treniranja, što ga čini brzim za treniranje.

Nedostaci:

- Računalno zahtjevan tijekom predikcije, posebno za velike skupove podataka.
- Performanse mogu opadati s visokom dimenzionalnošću podataka zbog problema prokletstva dimenzionalnosti.

2.2.2. Logistička regresija

Logistička regresija je statistička metoda za binarnu klasifikaciju koja modelira vjerojatnost binarnog ishoda na temelju jedne ili više prediktorskih varijabli. Koristi logističku funkciju za modeliranje binarne zavisne varijable.

- Logit funkcija - modelira vjerojatnost da određena ulazna točka pripada određenoj klasi.
- Treniranje - koeficijenti (β vrijednosti) se procjenjuju koristeći metode kao što je maksimalna vjerojatnost.
- Predikcija - daje vjerojatnost da određeni ulaz pripada određenoj klasi, obično koristeći prag od 0.5 za donošenje binarnih odluka.

Prednosti:

- Pruža vjerojatnosti, nudeći uvid u povjerenje predikcija.
- Jednostavan i učinkovit za binarne klasifikacijske probleme.
- Može se proširiti na multiklasnu klasifikaciju koristeći tehnike kao što je *one-vs-rest* koja je korištena u ovom radu

Nedostaci:

- Pretpostavlja linearnu vezu između nezavisnih varijabli i logaritamskih omjera.
- Nije pogodan za složene odnose između značajki.

2.2.3. Stablo odlučivanja

Stablo odlučivanja je model nalik stablu koji se koristi za klasifikaciju i regresiju. Dijeli podatke u podskupove na temelju vrijednosti ulaznih značajki, stvarajući grane koje predstavljaju odluke.

- Dijeljenje - na svakom čvoru, algoritam odabire značajku i prag koji rezultira najboljim podjelom, često koristeći kriterije poput Gini nečistoće (engl. *Gini impurity*), entropije ili smanjenja varijance.
- Rekurzija - proces se ponavlja rekurzivno za svaki čvor, stvarajući strukturu stabla.
- Listovi čvorovi - predikcije se donose na listovima čvorova, koji predstavljaju konačne oznake klasa.

Prednosti:

- Lako za razumijevanje i interpretaciju.
- Može rukovati i numeričkim i kategoriziranim podacima.
- Nelinearni odnosi između značajki ne utječu na performanse.

Nedostaci:

- Sklon pretreniranju, posebno kod dubokih stabala.
- Može biti nestabilan jer male promjene u podacima mogu dovesti do različitih podjela.

2.2.4. Usporedba KNN-a, LR-a i DT-a

Zajedničke točke:

- Sva tri algoritma mogu se koristiti za klasifikacijske zadatke.
- Imaju mehanizme za rukovanje numeričkim i kategoriziranim podacima (uz određenu pred obradu).
- Svaki algoritam ima parametre koji se mogu prilagoditi za poboljšanje performansi (npr. k u KNN, regularizacija u logističkoj regresiji, maksimalna dubina u stablu odlučivanja).

Razlike:

- Složenost i interpretabilnost:
 - KNN je jednostavan, ali manje interpretabilan u usporedbi sa stablom odlučivanja, koja pruža jasne putove odluka.
 - Logistička regresija nudi interpretabilne koeficijente koji pokazuju utjecaj svake značajke.
- Računalna učinkovitost:
 - Logistička regresija je općenito učinkovitija za treniranje i predikciju u usporedbi s KNN-om, koji može biti spor za velike skupove podataka zbog potrebe za izračunavanjem udaljenosti.
 - Stabla odlučivanja može biti računalno intenzivno za treniranje, ali je brzo za predikciju.
- Rukovanje nelinearnošću:
 - Stabla odlučivanja mogu prirodno rukovati nelinearnim odnosima, dok logistička regresija pretpostavlja linearnu vezu između značajki i logaritamskih omjera.
 - KNN implicitno može uhvatiti nelinearne odnose razmatrajući lokalne susjede.
- Pretreniranje:

- Stabla odlučivanja su sklona pretreniranju, posebno ako nisu obrezana ili ako im se dozvoli prevelika dubina.
- Logistička regresija teži boljoj generalizaciji, ali može podtrenirati ako je veza vrlo nelinearna.
- Performanse KNN-a ovise o izboru vrijednosti k i metričkoj udaljenosti; male k vrijednosti mogu dovesti do pretreniranja, dok velike k vrijednosti mogu podtrenirati.

Primjena:

- KNN je prikladan za male skupove podataka s jasno definiranim klasnim granicama i bez potrebe za interpretabilnošću.
- Logistička regresija je preferirana za binarne klasifikacijske probleme gdje je veza približno linearna i interpretabilnost je ključna.
- Stabla odlučivanja su korisna kada je važna interpretabilnost modela, a skup podataka ima nelinearne odnose i složene interakcije između značajki.

Kod 2.6 prikazuje inicijaliziranje svakog od opisanih modela :

```
# Decision Tree Model
dtt = DecisionTreeClassifier()

# KNN
knn = KNeighborsClassifier()

# LOGISTIC REGRESSION MODEL
lr = LogisticRegression()
```

Kod 2.6 Inicijalizacija modela

2.3. Alati i tehnologije

Za implementaciju modela i evaluaciju korišten je programski jezik Python i nekoliko ključnih biblioteka. Ove biblioteke omogućuju efikasno pretprocesiranje podataka, implementaciju različitih modela strojnog učenja, njihovu optimizaciju i vizualizaciju rezultata.

Kao glavni programski jezik korišten je Python zbog njegove jednostavnosti, fleksibilnosti i bogatstva biblioteka za strojno učenje. Glavne biblioteke korištene u ovom projektu su:

- Pandas za obradu i manipulaciju podataka
- Scikit-learn za implementaciju različitih modela strojnog učenja, evaluaciju i optimizaciju
- Optuna za optimizaciju hiperparametara modela
- Matplotlib i Seaborn za vizualizaciju podataka i rezultata
- Tabulate za prikaz rezultata u obliku tablica

Kroz ovu metodologiju, cilj je identificirati najučinkovitije modele za detekciju uljeza i doprinijeti općem razumijevanju primjene strojnog učenja u poboljšanju sigurnosti informacijskih sustava. Rezultati ovog istraživanja mogli bi pružiti vrijedne uvide za razvoj pouzdanijih sistema za detekciju uljeza, što je ključno u suzbijanju kibernetičkih prijetnji.

3. Treniranje modela

U ovom poglavlju detaljno su prikazani postupci treniranja i optimizacije različitih modela strojnog učenja, uključujući K-najbližih susjeda (KNN), logističku regresiju (LR) i stablo odlučivanja (DT).

3.1. Funkcija za treniranje i evaluaciju modela

Prvo definiramo funkciju `train_evaluate_model` koja trenira model, mjeri vrijeme treniranja i predikcije te evaluira performanse modela na trenirajućem i testnom skupu podataka (Kod 3.1).

```
import time

def train_evaluate_model(model, model_name):
    start_time = time.time()
    model.fit(x_train, y_train)
    end_time = time.time()
    training_time = end_time - start_time
    print(f"{model_name} Training time: {training_time:.4f}
seconds")

    start_time = time.time()
    model.predict(x_test)
    end_time = time.time()
    testing_time = end_time - start_time
    print(f"{model_name} Testing time: {testing_time:.4f}
seconds")

    train_score = model.score(x_train, y_train)
    test_score = model.score(x_test, y_test)
    print(f"{model_name} Training Score: {train_score:.4f}")
    print(f"{model_name} Test Score: {test_score:.4f}")

    return train_score, test_score, training_time,
testing_time
```

Kod 3.1 Funkcija za treniranje i evaluaciju modela

Opis funkcije:

- Treniranje modela: Funkcija započinje mjerenje vremena prije treniranja modela i zaustavlja mjerenje nakon završetka treniranja, čime se računa ukupno vrijeme treniranja.
- Predikcija: Nakon treniranja, funkcija mjeri vrijeme potrebno za predikciju na testnom skupu podataka.
- Evaluacija: Funkcija računa točnost modela na trenirajućem i testnom skupu podataka kako bi se ocijenile performanse modela.
- Povratne vrijednosti: Funkcija vraća rezultate u obliku n-torke koja sadrži točnost na trenirajućem skupu podataka, točnost na testnom skupu podataka, vrijeme treniranja i vrijeme predikcije.

Ova funkcija pruža strukturirani pristup za treniranje i evaluaciju različitih modela, omogućavajući nam usporedbu njihovih performansi na temelju vremena treniranja, vremena predikcije i točnosti.

3.2. KNN

U nastavku je opisano kako optimizirati parametre KNN klasifikatora korištenjem biblioteke Optuna, zatim treniranje optimiziranog modela te evaluacija njegovih performansi na trenirajućem i testnom skupu podataka. Optuna je biblioteka za optimizaciju hiperparametara koja koristi tehnike pretraživanja kako bi pronašla najbolje vrijednosti parametara za dani model. Potrebno je definirati funkciju cilja (engl. *objective function*) koja se koristi za evaluaciju različitih kombinacija hiperparametara tijekom optimizacije (Kod 3.2).

```
def objective_knn(trial):
    n_neighbors = trial.suggest_int('KNN_n_neighbors', 2, 16,
    log=False)
    model = KNeighborsClassifier(n_neighbors=n_neighbors)
    model.fit(x_train, y_train)
    accuracy = model.score(x_test, y_test)
    return accuracy

study_knn = optuna.create_study(direction='maximize')
study_knn.optimize(objective_knn, n_trials=10)
print("Best KNN parameters:", study_knn.best_trial.params)
```

```

KNN_model =
KNeighborsClassifier(n_neighbors=study_knn.best_trial.params[
'KNN_n_neighbors'])
KNN_train, KNN_test, KNN_train_time, KNN_test_time =
train_evaluate_model(KNN_model, "K-Nearest Neighbors")

```

Kod 3.2 KNN optimizacija

Ciljna funkcija `objective` uzima argument `trial` koji predstavlja pojedinačni pokušaj optimizacije. Unutar ciljne funkcije, koristeći `trial.suggest_int`, bira se vrijednost za parametar `n_neighbors` u rasponu od 2 do 16. Kreira se `KNeighborsClassifier` s odabranim `n_neighbors` i trenira se na skupu podataka za treniranje. Mjeri se točnost klasifikatora na testnom skupu podataka i ta vrijednost se vraća kao rezultat ciljne funkcije. Kreira se Optuna studija sa smjerom optimizacije postavljenim na `'maximize'`, što znači da želimo maksimizirati točnost. Studija se optimizira pozivanjem funkcije `study_KNN.optimize`, koja pokreće definiranu ciljnu funkciju `objective` tijekom 10 pokušaja (`n_trials=10`). Najbolji pokušaj optimizacije se ispisuje koristeći `study_KNN.best_trial`. Za treniranje i evaluacija optimiziranog KNN modela kreira se `KNeighborsClassifier` s najboljim `n_neighbors` parametrom pronađenim tijekom optimizacije. Model se trenira na skupu podataka za treniranje. Evaluira se točnost modela na trenirajućem i testnom skupu podataka te se rezultati ispisuju. Optimizacija hiperparametara pomoću Optuna pomaže u pronalaženju najboljih vrijednosti parametara, što može značajno poboljšati performanse modela u usporedbi s ručno podešenim parametrima. Kroz ovaj proces, osigurano je da KNN klasifikator koristi optimalne postavke za klasifikaciju podataka.

3.3. Logistička regresija

U ovom dijelu ćemo opisati postupak optimizacije hiperparametara modela logističke regresije pomoću biblioteke Optuna, kao i treniranje i evaluaciju optimiziranog modela. Pomoću Optune pretražuje se prostor hiperparametara kako bi se pronašla najbolja kombinacija za postizanje maksimalne točnosti na testnom skupu podataka. Slijedi kod za definiranje ciljne funkcije i pokretanje optimizacije (Kod 3.3) :

```

def objective_lr(trial):
    C = trial.suggest_loguniform('lr_C', 1e-6, 1e2)
    penalty = trial.suggest_categorical('lr_penalty', ['l1',
'12'])
    solver = 'liblinear' if penalty == 'l1' else 'lbfgs'
    tol = trial.suggest_loguniform('lr_tol', 1e-5, 1e-2)
    max_iter = trial.suggest_int('lr_max_iter', 100, 10000)
    model = LogisticRegression(C=C, penalty=penalty,
solver=solver, tol=tol, max_iter=max_iter, random_state=42,
multi_class="ovr")
    model.fit(x_train, y_train)
    accuracy = model.score(x_test, y_test)
    return accuracy

study_lr = optuna.create_study(direction='maximize')
study_lr.optimize(objective_lr, n_trials=10)
print("Best Logistic Regression parameters:",
study_lr.best_trial.params)

```

Kod 3.3 LR optimizacija

Ciljna funkcija `objective_lr` uzima argument `trial`, koji predstavlja pojedinačni pokušaj optimizacije. Unutar ciljne funkcije definiraju se hiperparametri koje želimo optimizirati:

- `C`: Regularizacijski parametar koji kontrolira jačinu regularizacije.
- `penalty`: Tip regularizacije (l1 ili l2).
- `solver`: Algoritam za optimizaciju (`liblinear` za l1 `penalty`, `lbfgs` za l2 `penalty`).
- `tol`: Tolerancija za kriterij zaustavljanja.
- `max_iter`: Maksimalan broj iteracija. Pretražuje se u rasponu od 100 do 10000.

Nakon definiranja modela s odabranim hiperparametrima, model se trenira na skupu podataka za treniranje, a točnost na testnom skupu se koristi kao metrika za optimizaciju. Optimizacija se pokreće pomoću `study_lr.optimize`, koja pokreće definiranu ciljnu funkciju tijekom 10 pokušaja (`n_trials=10`). Najbolji pronađeni parametri se ispisuju korištenjem `study_lr.best_trial.params`. Nakon optimizacije, model logističke regresije trenira se s najboljim pronađenim hiperparametrima i evaluira pomoću funkcije `train_evaluate_model` (Kod 3.4):

```

lg_model = LogisticRegression(
    C=study_lr.best_trial.params['lr_C'],
    penalty=study_lr.best_trial.params['lr_penalty'],
    solver='liblinear' if
study_lr.best_trial.params['lr_penalty'] == 'l1' else
'lbfgs',
    tol=study_lr.best_trial.params['lr_tol'],
    max_iter=study_lr.best_trial.params['lr_max_iter'],
    random_state=42, multi_class="ovr"
)
lg_train, lg_test, lg_train_time, lg_test_time =
train_evaluate_model(lg_model, "Logistic Regression")

```

Kod 3.4 Treniranje i evaluacija optimiziranog modela logističke regresije

3.4. Stablo odlučivanja

U ovom dijelu ćemo opisati postupak optimizacije hiperparametara stabla odlučivanja pomoću Optuna biblioteke. Cilj optimizacije je pronaći najbolju kombinaciju hiperparametara koja maksimizira točnost modela na testnim podacima. Nakon toga ćemo prikazati kako trenirati i evaluirati optimizirani model. Slijedi kod za definiranje ciljne funkcije i pokretanje optimizacije (Kod 3.5) :

```

def objective_dt(trial):
    dt_max_depth = trial.suggest_int('dt_max_depth', 2, 32,
log=False)
    dt_max_features = trial.suggest_int('dt_max_features', 2,
10, log=False)
    model = DecisionTreeClassifier(max_depth=dt_max_depth,
max_features=dt_max_features)
    model.fit(x_train, y_train)
    accuracy = model.score(x_test, y_test)
    return accuracy

study_dt = optuna.create_study(direction='maximize')
study_dt.optimize(objective_dt, n_trials=20)
print("Best Decision Tree parameters:",
study_dt.best_trial.params)

```

Kod 3.5 DT – ciljna funkcija

Ciljna funkcija `objective_dt` uzima argument `trial`, koji predstavlja pojedinačni pokušaj optimizacije. Unutar ciljne funkcije definiraju se hiperparametri koje želimo optimizirati:

- `max_depth`: Maksimalna dubina stabla. Pretražuje se u rasponu od 2 do 32.
- `max_features`: Maksimalan broj značajki koji se koristi pri svakom podjelu. Pretražuje se u rasponu od 2 do 10.

Nakon definiranja modela s odabranim hiperparametrima, model se trenira na skupu podataka za treniranje, a točnost na testnom skupu se koristi kao metrika za optimizaciju. Optimizacija se pokreće pomoću `study_dt.optimize`, koja pokreće definiranu ciljnu funkciju tijekom 20 pokušaja (`n_trials=20`). Najbolji pronađeni parametri se ispisuju korištenjem `study_dt.best_trial.params`. Nakon optimizacije, model stabla odlučivanja trenira se s najboljim pronađenim hiperparametrima i evaluira pomoću funkcije `train_evaluate_model` (Kod 3.6):

```
dt_model = DecisionTreeClassifier(  
    max_depth=study_dt.best_trial.params['dt_max_depth'],  
  
    max_features=study_dt.best_trial.params['dt_max_features']  
)  
dt_train, dt_test, dt_train_time, dt_test_time =  
train_evaluate_model(dt_model, "Decision Tree")
```

Kod 3.6 DT – treniranje optimiziranog modela

Optimizacija hiperparametara pomoću Optune osigurava da stablo odlučivanja koristi optimalne postavke za klasifikaciju podataka, što može značajno poboljšati njegove performanse u usporedbi s osnovnim modelom. Kroz ovaj proces, osigurano je da model stabla odlučivanja bude što učinkovitiji za odabrani skup podataka.

3.5. Usporedba rezultata treniranja/testiranja modela

Na kraju, prikazana je usporedba rezultata svih modela (KNN, logistička regresija i stablo odlučivanja) na trenirajućem skupu podataka. Tablica 3.1 prikazuje rezultate koji uključuju vrijeme treniranja, vrijeme testiranja, točnost pri treniranju i testiranju.

Tablica 3.1 Rezultati treniranja/testiranja

Model	Vrijeme treniranja (s)	Vrijeme testiranja	Treniranje	Testiranje
KNN	1.78846	58.1942	0.967739	0.960317
LR	2.57991	0.01419	0.816128	0.816276
DT	4.48883	0.03254	0.973915	0.970104

Vrijeme treniranja:

- Vrijeme treniranja za KNN model je 1.78846 sekundi, što je relativno brzo. KNN ne trenira model u tradicionalnom smislu, već samo pohranjuje podatke.
- Model koji koristi logističku regresiju ima vrijeme treniranja od 2.57991 sekundi. Ovo je duže od KNN-a, ali još uvijek prilično brzo. Treniranje modela uključuje optimizaciju parametara koristeći metode gradijentnog spuštanja, što može biti zahtjevnije ovisno o veličini skupa podataka.
- Model koji koristi stablo odlučivanja zahtijeva 4.48883 sekundi za treniranje, što je najduže među ovim modelima. Izgradnja stabla odlučivanja uključuje rekurzivnu podjelu podataka na temelju odabranih značajki, što može biti prilično računalno intenzivno.

Vrijeme testiranja:

- Vrijeme testiranja za KNN model je 58.1942 sekundi. Ovo je znatno duže nego kod ostalih modela jer KNN tijekom predikcije mora izračunati udaljenost od svake testne točke do svih trening točaka. To je računalno vrlo zahtjevno, posebno za velike skupove podataka.
- Logistička regresija ima vrijeme testiranja od 0.01419 sekundi. Predikcija za ovaj model je brza jer uključuje jednostavne matematičke operacije koristeći naučene parametre.
- Vrijeme testiranja za stablo odlučivanja je 0.03254 sekundi, što je također vrlo brzo. Predikcija u stablu odlučivanja uključuje slijed odluka kroz stablo, što je računarski jednostavno.

Točnost modela:

- KNN model pokazuje vrlo visoku točnost na setu za treniranje (0.967739) i testiranje (0.960317) što ukazuje na dobru generalizaciju, iako je vrijeme predikcije dugo.
- Logistička regresija ima točnost od 0.816128 na setu za treniranje i 0.816276 na setu za testiranje. Točnost je konzistentna između treninga i testiranja, što sugerira da model nije pretreniran ni podtreniran.
- Stablo odlučivanja pokazuje visoku točnost na setu za treniranje (0.973915) i testiranje (0.970104). Ovo također ukazuje na dobru generalizaciju i učinkovitost modela u klasifikaciji.

Svaki model ima svoje prednosti i nedostatke. KNN pokazuje visoku točnost, ali ima dugotrajno vrijeme predikcije. Logistička regresija je vrlo brza u predikciji, ali ima nešto nižu točnost. Stablo odlučivanja balansira između točnosti i brzine, pružajući visoku točnost uz brzo vrijeme predikcije. Izbor modela ovisi o specifičnim zahtjevima aplikacije, uključujući potrebu za brzim predikcijama ili visokom točnošću.

4. Evaluacija

Nakon odabira modela i procesa treniranja koristeći odabrani skup podataka modeli su optimizirani s ciljem postizanja najboljih mogućih performansi. Evaluacija modela napravljena je korištenjem standardnih metoda kao što su unakrsna validacija (engl. *cross validation*), matrica zabune (engl. *confusion matrix*), točnost (engl. *accuracy*), preciznost (engl. *precision*), odziv (engl. *recall*), i F1-ocjena (engl. *F1-score*).

4.1. Unakrsna validacija

Unakrsna validacija je metoda evaluacije modela koja osigurava da su rezultati modela pouzdani i da model dobro generalizira neviđene podatke. Korištenjem tehnike 10-dijelne unakrsne validacije (engl. *10-fold cross-validation*), skup podataka je podijeljen na 10 dijelova (engl. *fold*), pri čemu se model trenira na 9 dijelova, a testira na preostalom dijelu. Ovaj proces se ponavlja 10 puta, svaki put koristeći drugi dio kao testni set. Na kraju se računa prosječna točnost modela. Rezultati 10-dijelne unakrsne validacije za različite modele su prikazani u nastavku (Tablica 4.1):

Tablica 4.1 Unakrsna validacija

Model	Trenirajući skup	Testni skup
DT	0.97003	0.967998
LR	0.81612	0.81577
KNN	0.96009	0.95598

- Model stabla odlučivanja pokazao je visoku točnost s prosječnim rezultatom od 0.97003 i 0.967998 u 10-dijelnoj unakrsnoj validaciji. Ovo ukazuje da stablo odlučivanja ima izuzetno dobru sposobnost generalizacije na različite podskupove podataka, čineći ga pouzdanim izborom za klasifikaciju. Ovaj rezultat sugerira da model može precizno klasificirati nove podatke, održavajući konzistentne performanse kroz različite dijelove.

- Logistička regresija je postigla prosječnu točnost od 0.81612 i 0.81577. Iako je točnost nešto niža u usporedbi s modelima stabla odlučivanja i KNN modelima, logistička regresija pruža brze i efikasne predikcije. Ova metoda je korisna u aplikacijama gdje je brzina ključna, a rezultati pokazuju da model može zadržati stabilne performanse na različitim podskupovima podataka.
- KNN model postigao je visoku prosječnu točnost od 0.96009 i 0.95598. Ovo sugerira da KNN model također ima dobru sposobnost generalizacije. Visoka točnost kroz različite dijelove ukazuje na to da KNN model može učinkovito klasificirati podatke, iako je vrijeme predikcije dugo zbog prirode algoritma koji zahtijeva izračunavanje udaljenosti za svaku novu točku.

Rezultati 10-dijelne unakrsne validacije pružaju uvid u sposobnost generalizacije svakog modela. Stablo odlučivanja pokazalo je najvišu prosječnu točnost, slijedi ga KNN, dok je logistička regresija pokazala nešto nižu točnost. Unakrsna validacija osigurava da su rezultati modela pouzdani i da model može dobro generalizirati neviđene podatke, čime se smanjuje rizik od pretreniranja i podtreniranja.

4.2. Točnost

Točnost modela mjeri se kao omjer ispravno predviđenih ishoda u odnosu na ukupan broj slučajeva u skupu podataka (1).

$$Točnost = \frac{TP + TN}{TP + FP + FN + TN} \quad (1)$$

gdje su:

- TP (True Positive): Broj ispravno klasificiranih pozitivnih instanci.
- TN (True Negative): Broj ispravno klasificiranih negativnih instanci.
- FP (False Positive): Broj negativnih instanci koje su pogrešno klasificirane kao pozitivne.
- FN (False Negative): Broj pozitivnih instanci koje su pogrešno klasificirane kao negativne.

Ova metrika je posebno važna u domenama gdje su sve klase podjednako važne te pruža brzu indicaciju opće učinkovitosti modela. U prevedenoj analizi, točnost je izračunata kako

za trenirajući, tako i za testni skup podataka. Dobiveni rezultati su prikazani u nastavku (Tablica 4.2):

Tablica 4.2 Točnost

Model	Trenirajući skup	Testni skup
DT	0.97975	0.97074
LR	0.81613	0.81627
KNN	0.96774	0.96032

Stablo odlučivanja (DT) pokazuje točnost od 0.97975 na trenirajućem skupu i 0.97074 na testnom skupu. Ova visoka točnost sugerira da je model dobro prilagođen podacima te ima sposobnost dobre generalizacije nepoznatih podataka.

Logistička regresija (LR) ostvarila je točnost od 0.81613 na trenirajućem skupu i 0.81627 na testnom skupu. Konzistentnost točnosti između trenirajućeg i testnog skupa ukazuje na to da model nije ni pretreniran ni podtreniran, što je česta pojava u modeliranju.

KNN pokazuje točnost od 0.96774 na trenirajućem skupu i 0.96032 na testnom skupu. Visoka točnost na oba skupa ukazuje na to da model vrlo efikasno razlikuje različite klase, što ga čini pouzdanim izborom za praktičnu primjenu.

Ove brojke pokazuju kako različiti modeli rade pod različitim uvjetima i s različitim vrstama podataka. Izbor modela za specifičan problem trebao bi uzeti u obzir ove razlike u točnosti, ovisno o specifičnim zahtjevima projekta ili aplikacije.

4.3. Matrica zabune

Matrica zabune (engl. *confusion matrix*) je jedan od najvažnijih alata za evaluaciju performansi klasifikacijskih modela u strojnom učenju. Ona vizualno prikazuje točnost predikcija modela u usporedbi sa stvarnim vrijednostima, dajući nam detaljan uvid u to gdje model uspijeva, a gdje griješi. Matrica zabune je tablica s dvije dimenzije: stvarne klase i predviđene klase. U nastavku je prikazana struktura višeklasne matrice zabune (Tablica 4.3):

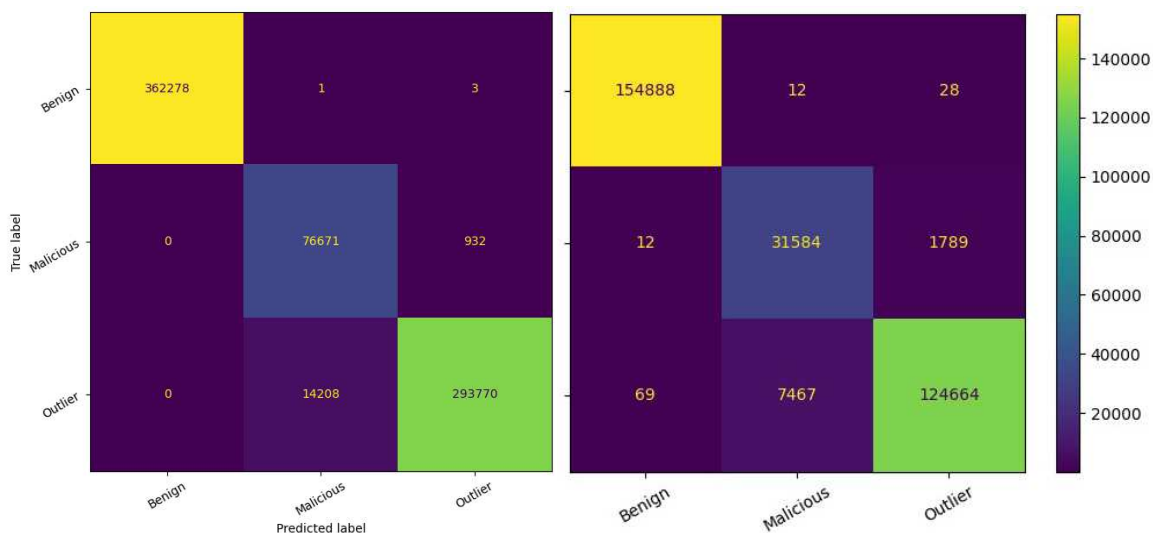
Tablica 4.3 Matrica zabune

	Predviđeno A (benign)	Predviđeno B (malicious)	Predviđeno C (outlier)
Stvarno A (benign)	Istinито A (TA)	Lažno AB	Lažno AC
Stvarno B (malicious)	Lažno BA	Istinито B (TB)	Lažno C
Stvarno C (outlier)	Lažno CA	Lažno CB	Istinито C (TC)

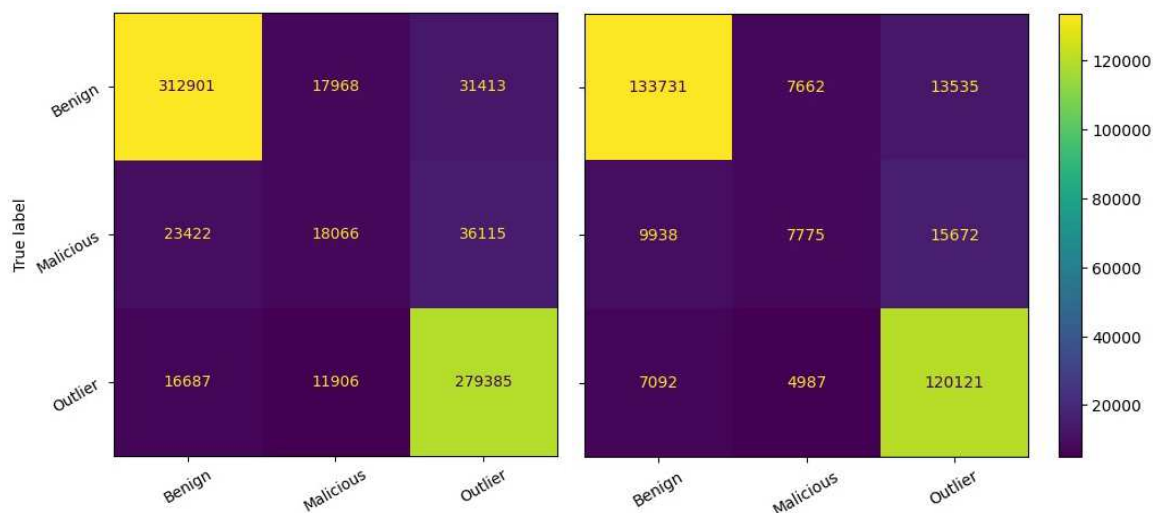
gdje je:

- Istinито A (TA): Broj instanci pravilno klasificiranih kao A.
- Lažno AB: Broj instanci koje su stvarno A, ali su pogrešno klasificirane kao B.
- Lažno AC: Broj instanci koje su stvarno A, ali su pogrešno klasificirane kao C.
- Lažno BA, Lažno BC, Lažno CA, Lažno CB: Slično kao gore, predstavljaju broj instanci koje su pogrešno klasificirane iz jedne klase u drugu.
- Istinито B (TB), Istinито C (TC): Broj instanci koje su pravilno klasificirane kao B ili C, respektivno.

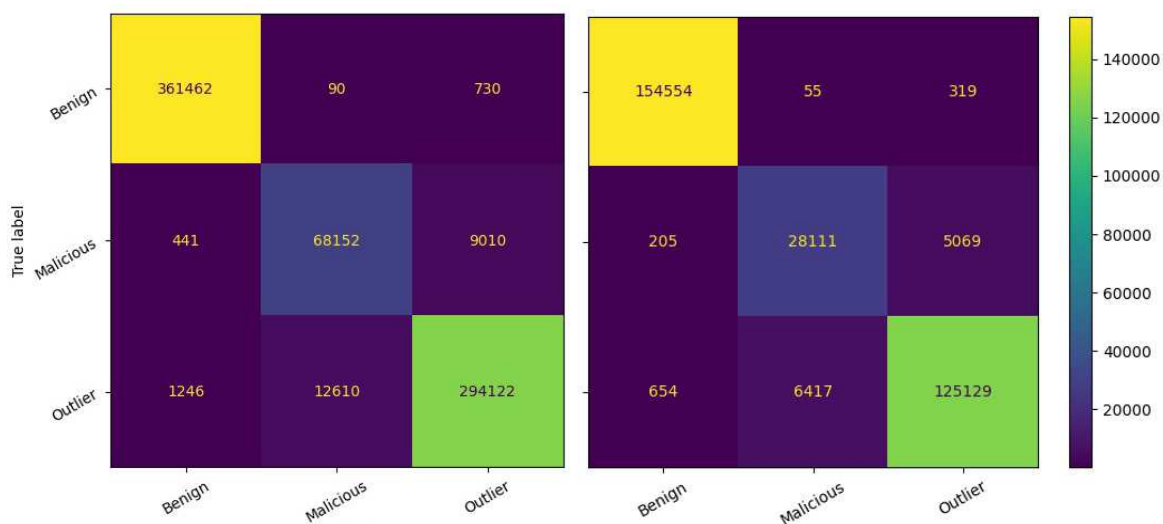
Na slikama Slika 4.1, Slika 4.2 i Slika 4.3 možemo vidjeti matrice zabune za analizirane modele na trenirajućem i testnom skupu:



Slika 4.1 Matrica zabune za trenirajući i testni skup - DT



Slika 4.2 Matrica zabune za trenirajući i testni skup – LR



Slika 4.3 Matrica zabune za trenirajući i testni skup – KNN

Stablo odlučivanja (DT):

- Matrica zabune za model DT na trenirajućem skupu podataka pokazuje izvrsne rezultate s vrlo visokim brojem točnih klasifikacija za sve klase (*benign*, *malicious*, *outlier*). Model uspijeva ispravno klasificirati većinu instanci, što je vidljivo iz visokih brojeva dijagonalnih elemenata matrice zabune. Ova visoka točnost na trenirajućem skupu ukazuje na dobru sposobnost modela da uči obrasce iz podataka.

- Na testnom skupu podataka, stablo odlučivanja također pokazuje vrlo dobre performanse, iako blago niže u usporedbi s trenirajućim skupom. I dalje se postiže visoka točnost klasifikacije za sve klase, što ukazuje na dobru generalizaciju modela.

KNN:

- Matrica zabune za model KNN na trenirajućem skupu podataka također pokazuje vrlo dobre rezultate, iako nešto lošije u usporedbi sa stablom odlučivanja. Model uspijeva pravilno klasificirati većinu instanci, ali postoji nešto veći broj pogrešnih klasifikacija (False Positives i False Negatives) u usporedbi sa stablom odlučivanja.
- Na testnom skupu podataka, performanse KNN modela su nešto slabije u usporedbi s trenirajućim skupom, ali još uvijek prilično dobre. Model i dalje pokazuje visoku točnost klasifikacije, ali je prisutno nešto više pogrešaka u usporedbi sa stablom odlučivanja. Ovi rezultati ukazuju na solidnu sposobnost generalizacije modela, ali i na moguće probleme s učinkovitosti kod velikih skupova podataka zbog dugotrajnog vremena predikcije.

Logistička regresija (LR):

- Matrica zabune za logističku regresiju pokazuje slabije rezultate u usporedbi s DT i KNN modelima, posebno na trenirajućem skupu podataka. Iako model uspijeva ispravno klasificirati veliki broj *benign* i *outlier* instanci, performanse na *malicious* instancama su znatno slabije, s visokim brojem pogrešnih klasifikacija.
- Na testnom skupu podataka, logistička regresija pokazuje sličan obrazac s relativno dobrom točnošću na *benign* i *outlier* instancama, ali vrlo lošim performansama na *malicious* instancama. Ova slabost može ukazivati na potrebu za daljnjim unapređenjem modela ili upotrebom složenijih tehnika za poboljšanje klasifikacije *malicious* instanci.

Na temelju analiza matrica zabune, može se zaključiti da stablo odlučivanja pokazuje najbolje performanse, kako na trenirajućem, tako i na testnom skupu podataka, s visokom točnošću klasifikacije za sve klase. KNN model također pokazuje dobre rezultate, ali uz nešto više krivih predikcija. Logistička regresija pokazuje najlošije performanse, posebno

kod klasifikacije *malicious* instanci te bi mogao zahtijevati dodatna poboljšanja za postizanje boljih rezultata. Ovi rezultati jasno ukazuju na prednosti i slabosti svakog modela, omogućujući bolji uvid u njihove sposobnosti i potencijalne primjene u stvarnim scenarijima.

4.4. Preciznost, odziv i F1-ocjena

Na kraju je generirano klasifikacijsko izvješće (engl. *classification report*) za svaki model na testirajućem i testnom skupu podataka. Ovo izvješće sadrži nekoliko ključnih metrika koje omogućuju detaljan uvid u performanse modela. Klasifikacijsko izvješće sadrži sljedeće metrike za svaku klasu:

Preciznost (engl. *precision*):

- Definicija: Preciznost je udio točno klasificiranih primjera (True Positives) u skupu pozitivno klasificiranih primjera (True Positives + False Positives) (2).

$$Preciznost = \frac{TP}{TP + FP} \quad (2)$$

- Značaj: Visoka preciznost znači da je model dobar u izbjegavanju lažno pozitivnih predviđanja.

Odziv (engl. *recall*)

- Definicija: Odziv je udio točno klasificiranih primjera (True Positives) u skupu svih stvarnih pozitivnih primjera (True Positives + False Negatives) (3).

$$Odziv = \frac{TP}{TP + FN} \quad (3)$$

- Značaj: Visok odziv znači da je model dobar u identificiranju svih stvarnih pozitivnih primjera.

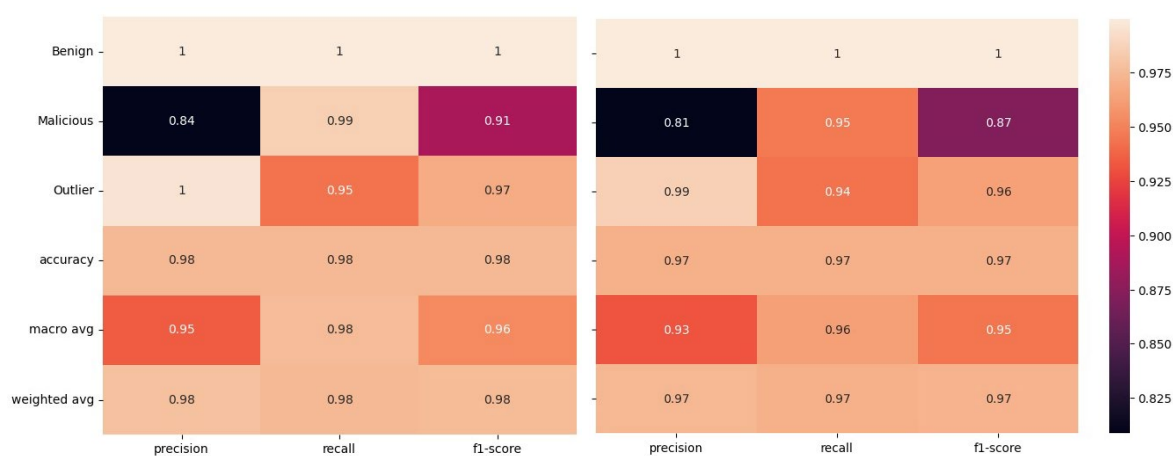
F1-ocjena (engl. *F1-score*):

- Definicija: F1-ocjena je harmonijska sredina preciznosti i odziva (4).

$$F1 - ocjena = 2 * \frac{odziv * preciznost}{odziv + preciznost} \quad (4)$$

- Značaj: F1-ocjena je korisna kada želimo uravnotežiti preciznost i odziv, posebno kada imamo nebalansirane klase.

Na slikama Slika 4.4, Slika 4.5 i Slika 4.6 možemo vidjeti klasifikacijska izvješća za odabrane modele.



Slika 4.4 Klasifikacijsko izvješće za trenirajući i testni skup – DT

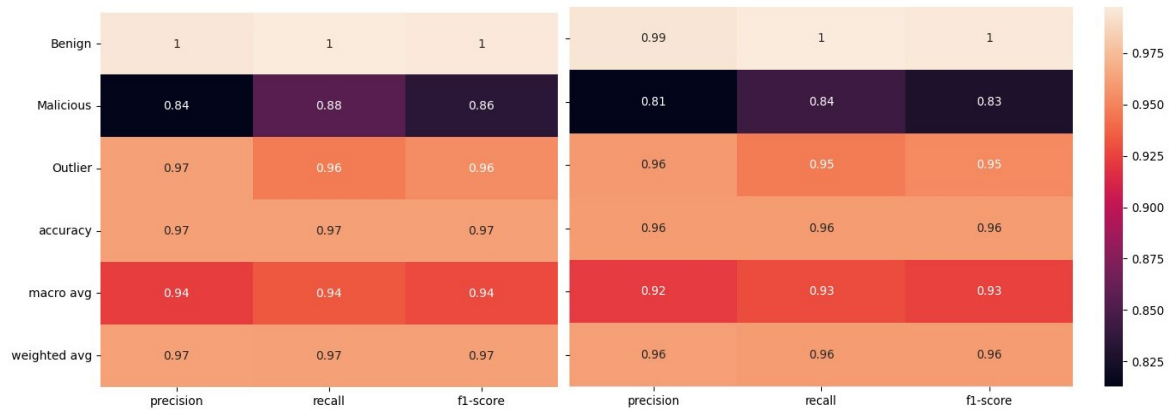
Trenirajući skup:

- **Benign:** Preciznost, odziv i F1-ocjena su savršene (1.00), što ukazuje na besprijekornu klasifikaciju ove klase.
- **Malicious:** Preciznost je 0.84, odziv 0.99, a F1-ocjena 0.91, što pokazuje vrlo dobre performanse, ali s nekim lažno pozitivnim predikcijama.
- **Outlier:** Preciznost je 1.00, odziv 0.95, a F1-ocjena 0.97, što ukazuje na visoku točnost s malim brojem lažno negativnih predikcija
- **Ukupna točnost:** 0.98, što pokazuje visoku točnost modela na trenirajućem skupu.

Testni skup:

- **Benign:** Performanse su identične trenirajućem skupu s preciznošću, odzivom i F1-ocjenom od 1.00.

- **Malicious:** Preciznost pada na 0.81, odziv na 0.95, a F1-ocjena na 0.87, što ukazuje na nešto slabije, ali i dalje visoke performanse.
- **Outlier:** Performanse su slične trenirajućem skupu s preciznošću od 0.99, odzivom 0.94 i F1-ocjenom 0.96.
- **Ukupna točnost:** 0.97, što pokazuje vrlo dobru generalizaciju modela.



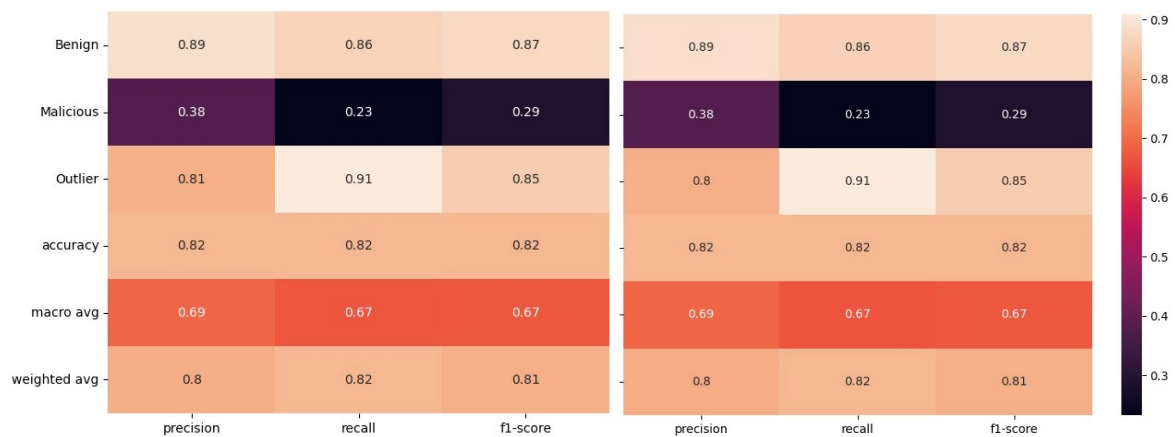
Slika 4.5 Klasifikacijsko izvješće za trenirajući i testni skup – KNN

Trenirajući skup:

- **Benign:** Perfektna klasifikacija s preciznošću, odzivom i F1-ocjenom od 1.00.
- **Malicious:** Preciznost 0.84, odziv 0.88, F1-ocjena 0.86, što ukazuje na visoku točnost.
- **Outlier:** Preciznost 0.97, odziv 0.96, F1-ocjena 0.96, što također pokazuje visoke performanse.
- **Ukupna točnost:** 0.97, vrlo visoka točnost.

Testni skup:

- Performanse su vrlo slične trenirajućem skupu s ukupnom točnošću od 0.96, što ukazuje na dobru generalizaciju i konzistentne performanse modela.



Slika 4.6 Klasifikacijsko izvješće za trenirajući i testni skup – LR

Trenirajući skup:

- **Benign:** Preciznost 0.89, odziv 0.86, F1-ocjena 0.87, što ukazuje na solidne performanse.
- **Malicious:** Preciznost 0.38, odziv 0.23, F1-ocjena 0.29, što ukazuje na poteškoće u klasifikaciji ove klase.
- **Outlier:** Preciznost 0.81, odziv 0.91, F1-ocjena 0.85, što pokazuje dobre performanse.
- **Ukupna točnost:** 0.82, što je dosta niže nego kod DT modela.

Testni skup:

- Performanse su gotovo identične trenirajućem skupu, što ukazuje na stabilnu, ali nižu točnost klasifikacije u odnosu na DT i KNN modele.

Modeli su pokazali različite razine performansi. Stablo odlučivanja (DT) i KNN model pokazuju visoke performanse s vrlo visokom ukupnom točnošću i dobrim generalizacijskim sposobnostima. Logistička regresija (LR) pokazuje niže performanse, posebno u klasifikaciji *malicious* klasa, što sugerira da bi se moglo razmotriti dodatno podešavanje ili korištenje alternativnih modela za poboljšanje performansi na ovoj klasi.

4.5. Usporedba binarne i višeklasne klasifikacije

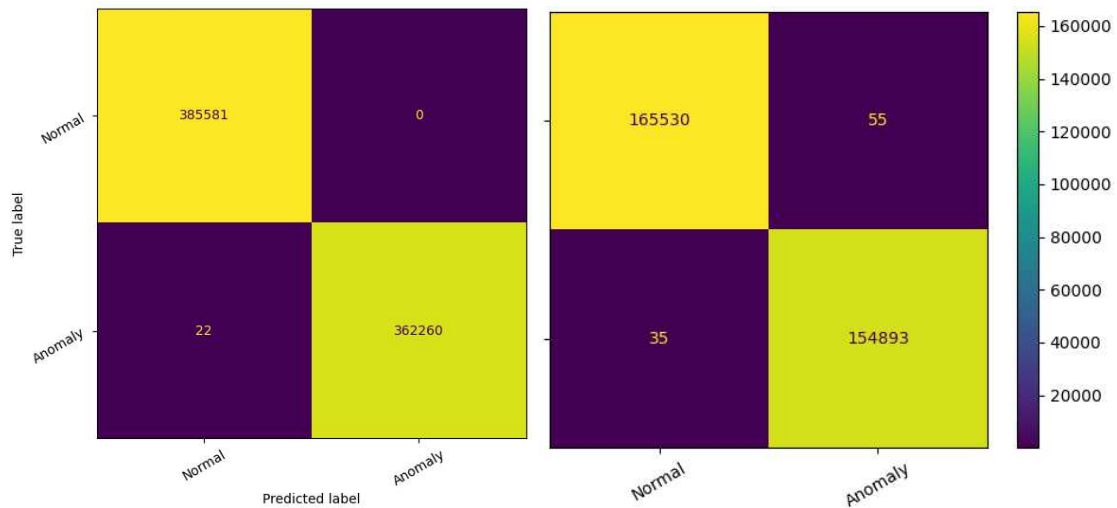
U ovom dijelu rada, opisana je prilagodba klasifikacijskog problema za binarnu klasifikaciju. Umjesto tri klase (*benign*, *malicious*, *outlier*), korištene su samo dvije klase: *normal* i *anomaly*, pri čemu su *malicious* i *outlier* objedinjeni u klasu *anomaly*.

Nakon treniranja modela na binarno klasificiranim podacima, dobiveni su sljedeći rezultati točnosti (Tablica 4.4) :

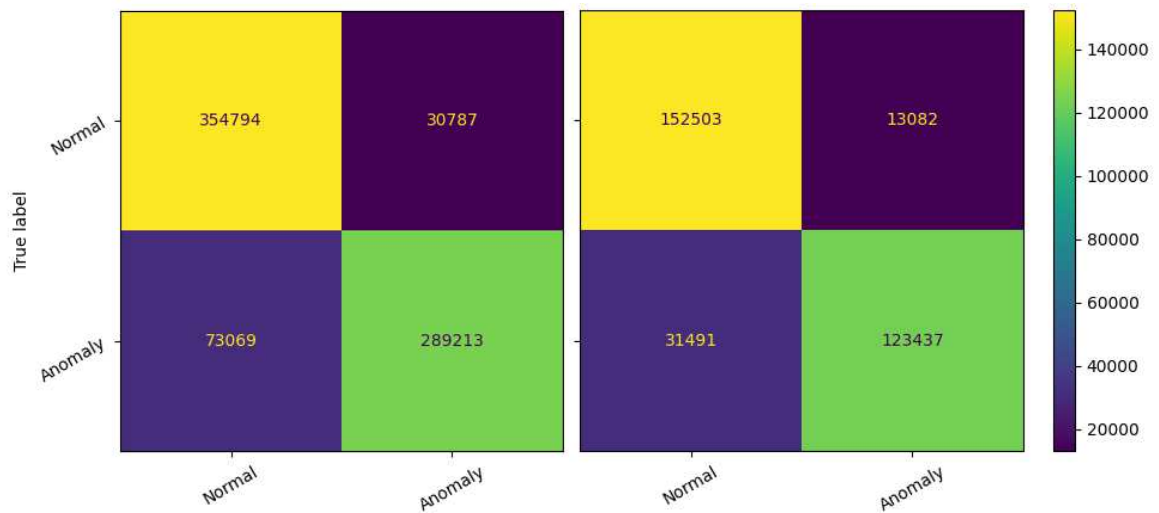
Tablica 4.4 Točnost

Model	Trenirajući skup	Testni skup
DT	0.99997	0.99972
LR	0.86113	0.86093
KNN	0.99852	0.99832

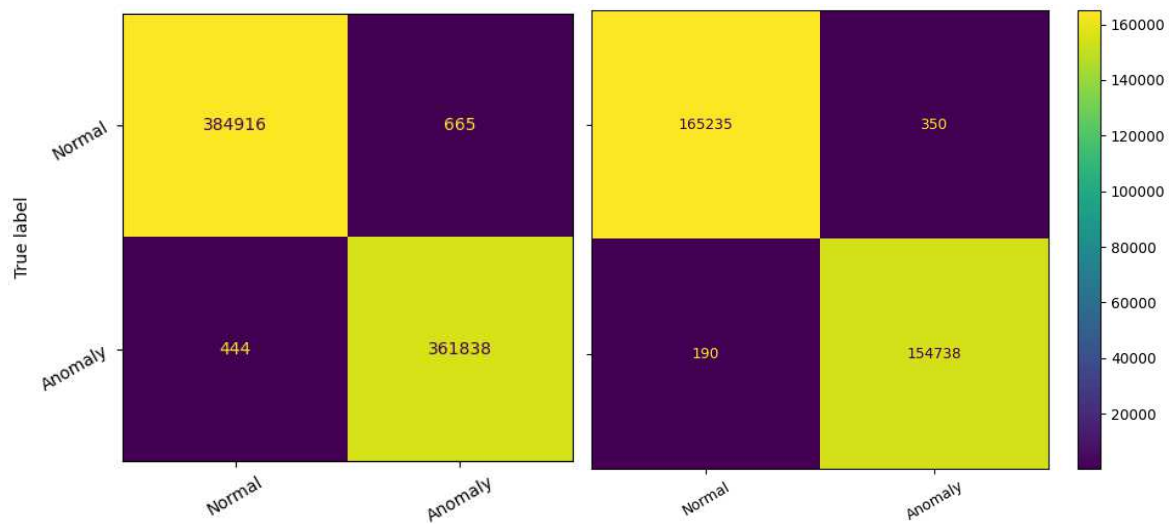
Matrice zabune prikazane su u nastavku (Slika 4.7, Slika 4.8 i Slika 4.9):



Slika 4.7 Matrica zabune za trenirajući i testni skup - DT

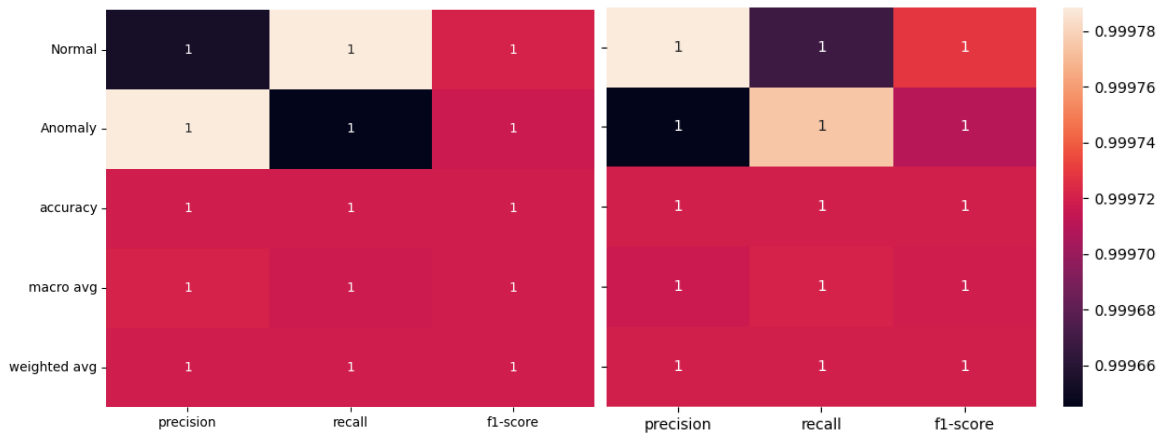


Slika 4.8 Matrica zabune za trenirajući i testni skup – LR

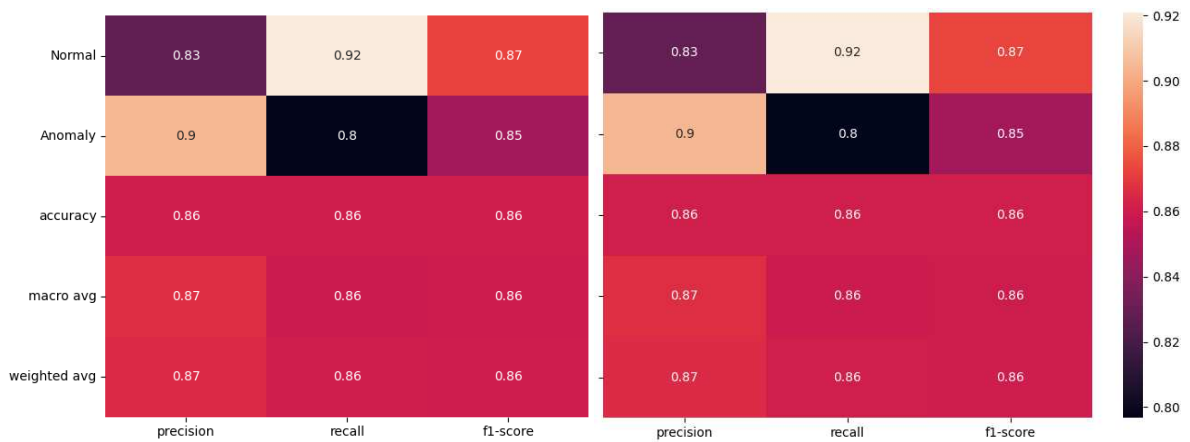


Slika 4.9 Matrica zabune za trenirajući i testni skup – KNN

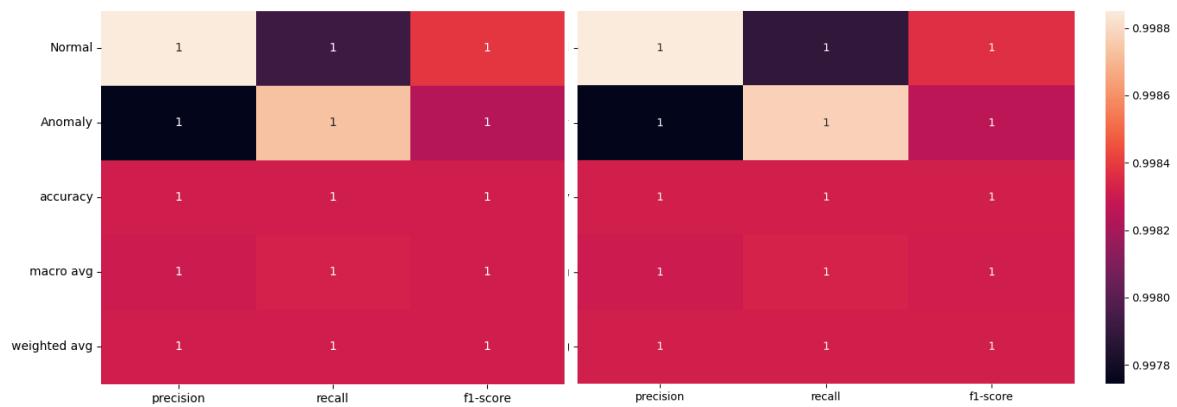
Klasifikacijska izvješća su prikazana na slikama u nastavku (Slika 4.10, Slika 4.11 i Slika 4.12):



Slika 4.10 Klasifikacijsko izvješće za trenirajući i testni skup – DT



Slika 4.11 Klasifikacijsko izvješće za trenirajući i testni skup – LR



Slika 4.12 Klasifikacijsko izvješće za trenirajući i testni skup – KNN

Usporedbom rezultata binarne i višeklasne klasifikacije, može se uočiti:

- **Točnost:** Binarna klasifikacija pokazuje blago višu točnost u odnosu na višeklasnu klasifikaciju za sve modele. Na primjer, stablo odlučivanja ima točnost od 0.99972 na testnom skupu u binarnoj klasifikaciji, dok je u višeklasnoj točnost bila 0.97974.
- **Matrica zabune:** Matrica zabune za binarnu klasifikaciju pokazuje manji broj pogrešnih klasifikacija u odnosu na višeklasnu. Ovo je očekivano jer binarna klasifikacija pojednostavljuje problem.
- **Preciznost, odziv, F1-ocjena:** Metrike preciznosti, odziva i F1-ocjene su također bolje u binarnoj klasifikaciji, što ukazuje na bolju sposobnost modela da razlikuje između normalnog i neuobičajenog ponašanja kada se smanji broj klasa.

Rezultati pokazuju da binarna klasifikacija može biti korisnija kada je cilj jednostavno detektirati anomalije bez detaljne kategorizacije. Višeklasna klasifikacija pruža detaljniji uvid, ali dolazi s povećanim izazovima i mogućnošću pogrešnih klasifikacija između sličnih klasa. Ovi rezultati sugeriraju da izbor između binarne i višeklasne klasifikacije ovisi o specifičnim potrebama aplikacije i raspoloživim resursima za treniranje i evaluaciju modela.

5. Rezultati i diskusija

U ovom poglavlju predstavljeni su rezultati analize koja ispituje kako različiti modeli strojnog učenja otkrivaju uljeze u sustavima. Za usporedbu performansi svakog modela korišten je niz ključnih mjera uspješnosti koje nam pomažu razumjeti koliko su oni efikasni u svojim zadacima. U nastavku je dano kratko objašnjenje svake od tih mjera u kontekstu detekcije uljeza:

- **Točnost (engl. *accuracy*)** - Ova mjera govori koliko često model ispravno identificira i prave i lažne uljeze. To je poput ocjenjivanja u školi; što je viši postotak točnosti, to bolje.
- **Vrijeme treniranja/testiranja** - Koliko brzo model uči i koliko brzo može donijeti odluku je ključno, posebno u situacijama gdje se broji svaka sekunda. Vrijeme treniranja govori koliko modelu treba da 'nauči' na podacima koje je dobio u periodu učenja, dok vrijeme testiranja pokazuje kako brzo može raditi kada ga stavimo na test.
- **Preciznost (engl. *precision*)** – Ova mjera pokazuje koliko možemo vjerovati modelu kada kaže da je nešto uljez. Ako model ima visoku preciznost, to znači da kada nas upozori na uljeza, vjerojatno ima pravo.
- **Odziv (engl. *recall*)** - Ova mjera se fokusira na sposobnost modela da prepozna stvarne uljeze. Visok odziv znači da model ne propušta mnogo stvarnih prijetnji, što je izuzetno važno za održavanje sigurnosti.
- **F1-ocjena** - Kombinira preciznost i odziv u jednu brojku, dajući uravnotežen pogled na to kako model radi. To je kao prosjek, ali fokusiran na ravnotežu između hvatanja uljeza i izbjegavanja lažnih alarma.

Kroz ove metrike, ne samo da vidimo kako se modeli ponašaju u idealnim uvjetima, već dobivamo uvid u to kako će se vjerojatno ponašati u stvarnim scenarijima, što je ključno za odluku o tome koji model koristiti za zaštitu sustava od neželjenih uljeza.

5.1. Analiza i interpretacija rezultata

Rezultati provedene studije pokazuju izvanredne performanse modela u detekciji uljeza na temelju analize mrežnih podataka. Stablo odlučivanja postiglo je izvanrednu točnost od 98 %, što sugerira da je model vrlo efikasan u identifikaciji zlonamjernih aktivnosti s minimalnim brojem lažno pozitivnih i lažno negativnih rezultata. KNN model također pokazuje visoku točnost od 97 %, dok logistička regresija ima točnost od svega 82 %.

Ovi rezultati potvrđuju da modeli strojnog učenja imaju izvanredan potencijal u detekciji uljeza u mrežnim sustavima. Stablo odlučivanja se ističe kao najučinkovitiji model u provedenom istraživanju, što može biti rezultat njegove sposobnosti modeliranja kompleksnih interakcija između značajki podataka.

Daljnja analiza rezultata pokazuje da su korišteni modeli također visoko precizni i osjetljivi, što je ključno za detekciju zlonamjernih aktivnosti uz minimiziranje lažnih upozorenja. F1-ocjena za sve modele također je visok, što ukazuje na dobru ravnotežu između preciznosti i odziva u detekciji uljeza.

5.2. Izazovi tijekom implementacije i evaluacije modela

Tijekom implementacije i evaluacije modela suočili smo se s nekoliko izazova. Jedan od njih bio je odabir optimalnih hiperparametara za svaki model, što je zahtijevalo detaljnu optimizaciju i ispitivanje. Kako bi se osigurala stabilnost i pouzdanost modela, ključno je bilo pravilno skaliranje i obrada podataka. Interpretacija rezultata i usporedba modela zahtijevali su analizu kako bi se razumjele njihove prednosti i nedostaci u kontekstu detekcije uljeza.

5.3. Usporedba s relevantnim radovima u području

Uspoređujući dobivene rezultate s relevantnim istraživanjima u području detekcije uljeza, primjećujemo da su performanse analiziranih modela usporedive ili čak bolje od nekih postojećih studija. Visoka točnost i preciznost, posebno kod stabla odlučivanja, sugeriraju da korišteni modeli mogu biti konkurentni u stvarnom svijetu primjene detekcije uljeza. Slične rezultate su zabilježili Abdaljabar i suradnici (2021) te Kumar i suradnici (2023), koji su također otkrili da stabla odlučivanja pružaju visoku točnost u detekciji uljeza.

Provedeno istraživanje također pruža dodatne uvide u prednosti i nedostatke različitih modela strojnog učenja u kontekstu detekcije uljeza. Ovi uvidi mogu biti korisni za istraživače i profesionalce u području kibernetičke sigurnosti kako bi razvili naprednije modele detekcije uljeza i poboljšali sigurnost mrežnih sustava.

5.4. Teorijska podloga i implikacije rezultata

Teorijska osnova dobivenih rezultata ukorijenjena je u naprednim konceptima strojnog učenja, s fokusom na nadzirano učenje i tehnike klasifikacije. Implementacija i usporedba različitih algoritama strojnog učenja, kao što su metoda K-najbližih susjeda (KNN), logistička regresija i stablo odlučivanja, omogućile su detaljnu analizu njihove učinkovitosti u kontekstu detekcije uljeza. Svaki od ovih modela pokazuje određene prednosti u rješavanju specifičnih izazova u identifikaciji zlonamjernih aktivnosti unutar mrežnih sustava. Implementacija modela stabla odlučivanja, koji se istaknuo svojom iznimnom točnošću, naglašava važnost razumijevanja složenih uzoraka interakcija među atributima podataka. Ovo modeliranje omogućava izrazitu efikasnost u detekciji uljeza i minimiziranju pogrešnih alarma, čime se potvrđuje teorijska prednost metode razgranatih odluka u kontekstu velikih i raznolikih podatkovnih skupova. Implikacije ovih rezultata su višestruke. Prvo, visoka točnost i robusnost ovih modela u detekciji uljeza pružaju vrijedan uvid u potencijal primjene strojnog učenja u razvoju efikasnijih sigurnosnih alata. Ovi rezultati mogu potaknuti daljnji razvoj i rafiniranje algoritama za detekciju uljeza, osobito u kontekstima gdje su konvencionalne metode neadekvatne ili previše sporo reagiraju na nove vrste prijetnji. Dodatno, provedena istraživanja potiču na razmatranje širih aplikacija ovih modela izvan tradicionalnih kibernetičkih okruženja, uključujući predikciju prijetnji u Internetu stvari (engl. *Internet of Things*, IoT), gdje sigurnosni izazovi postaju sve kompleksniji. Točnost i prilagodljivost modela stabla odlučivanja i algoritma KNN mogu se iskoristiti za razvoj personaliziranih sigurnosnih rješenja koja uzimaju u obzir specifične zahtjeve i okruženja korisnika.

Zaključak

Sigurnost informacijskih sustava postaje sve važnija s porastom učestalosti i sofisticiranosti mrežnih napada. U ovom diplomskom radu istražene su različite metode strojnog učenja kako bi se unaprijedila detekcija uljeza u mrežnom prometu. Fokus istraživanja bio je na analizi i evaluaciji postojećih tehnika te razvoju novog modela strojnog učenja koji bi mogao poboljšati preciznost i efikasnost detekcije.

U radu su korišteni različiti modeli strojnog učenja, uključujući K-najbliže susjede, logističku regresiju i stablo odlučivanja. Performanse modela evaluirane su korištenjem metrika kao što su točnost, preciznost, odziv i F1-ocjena. Rezultati su pokazali da je model stabla odlučivanja posebno učinkovit, s visokim stopama točnosti i preciznosti, što ukazuje na njegov potencijal u prepoznavanju zlonamjernih aktivnosti uz minimalan broj lažno pozitivnih i lažno negativnih rezultata.

Prilikom implementacije i evaluacije modela suočili smo se s brojnim izazovima, uključujući optimizaciju hiperparametara i interpretaciju rezultata. Pažljiva analiza i pristup bili su ključni za prevladavanje ovih izazova. Usporedba s relevantnim istraživanjima pokazala je da dobiveni rezultati konkuriraju postojećim metodama i pružaju dodatne uvide u prednosti i nedostatke različitih modela strojnog učenja.

Zaključno, ovo istraživanje potvrđuje da primjena strojnog učenja može značajno unaprijediti sigurnost informacijskih sustava kroz učinkovitu detekciju uljeza. Daljnja istraživanja i inovacije u ovom području bit će ključni za razvoj još pouzdanijih i učinkovitijih sustava koji mogu odgovoriti na sve složenije kibernetičke prijetnje.

Literatura

- [1] Pradhan, Manoranjan, Chinmaya Kumar Nayak, and Sateesh Kumar Pradhan. *Intrusion detection system (IDS) and their types*. Securing the internet of things: Concepts, methodologies, tools, and applications. IGI Global, 2020.
- [2] M. Agoramoorthy, A. Ali, D. Sujatha, M. R. T. F and G. Ramesh, *An Analysis of Signature-Based Components in Hybrid Intrusion Detection Systems*, 2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS), Chennai, India, 2023
- [3] Garcia-Teodoro, P., Diaz-Verdejo, J., Macia-Fernandez, G., & Vazquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*
- [4] Kim, J., Kim, J., Thu, H. L. T., & Kim, H. *Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection*. International Conference on Platform Technology and Service, 2016.
- [5] S. Zheng, *Network Intrusion Detection Model Based on Convolutional Neural Network*, 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 2021.
- [6] C. Lu, *Research on the technical application of artificial intelligence in network intrusion detection system*, 2022.
- [7] Ryan, M. *LUFlow Network Intrusion Detection Data Set*. Poveznica: <https://www.kaggle.com/datasets/mryanm/lufLOW-network-intrusion-detection-data-set/data>; pristupljeno 1.ožujka 2024.
- [8] V. Kathiresan, S. Karthik, P. Divya and D. P. Rajan, *A Comparative Study of Diverse Intrusion Detection Methods using Machine Learning Techniques*, 2022 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2022.
- [9] Z. H. Abdaljabar, O. N. Ucan and K. M. Ali Alheeti, *An Intrusion Detection System for IoT Using KNN and Decision-Tree Based Classification*, 2021 International Conference of Modern Trends in Information and Communication Technology Industry (MTICTI), Sana'a, Yemen, 2021.
- [10] A. Kumar and I. Sharma, *Machine Learning Based Model and Solution for Network Flow Attack Detection*, 2023 4th IEEE Global Conference for Advancement in Technology (GCAT), Bangalore, India, 2023.

Sažetak

Naslov: Otkrivanje uljeza u mrežnom prometu

Sažetak: U okviru ovog rada istražuje se primjena metoda strojnog učenja u detekciji uljeza u mrežnom prometu s ciljem unapređenja sigurnosti informacijskih sustava. Kroz pregled relevantne literature, metodologiju istraživanja, implementaciju različitih modela strojnog učenja te analizu dobivenih rezultata, pruža se sveobuhvatan uvid u trenutne mogućnosti u detekciji uljeza. Rezultati ukazuju na izvanredan potencijal modela poput stabla odlučivanja u identifikaciji zlonamjernih aktivnosti, pružajući korisne smjernice za razvoj efikasnijih sustava za sigurnost mrežnih sustava.

Ključne riječi: Strojno učenje, detekcija uljeza, mrežni promet, sigurnost informacijskih sustava, analiza rezultata.

Summary

Title: Network Traffic Intrusion Detection

Abstract: The thesis explores the application of machine learning algorithms for network traffic intrusion detection with the aim of enhancing the security of information systems. Through a review of relevant literature, research methodology, implementation of various machine learning models, and analysis of the obtained results, a comprehensive insight into the current state of the art in intrusion detection is provided. The results indicate the outstanding potential of models, such as decision trees, in identifying malicious activities, offering valuable guidance for the development of more efficient systems for network security.

Keywords: Machine learning, intrusion detection, network traffic, information systems security, results analysis

Skraćenice

IDS	<i>Intrusive Detection Systems</i>	sustav za otkrivanje upada
LSTM	<i>long short-term memory</i>	dugo kratkoročno pamćenje
DBN	<i>Deep Belief Network</i>	mreža dubokog uvjerenja
CNN	<i>Convolutional Neural Network</i>	konvolucijska neuronska mreža
RNN	<i>Recurrent Neural Network</i>	rekurentna neuronska mreža
CTI	<i>Cyber Threat Intelligence</i>	podaci o kibernetičkim prijetnjama
SSH	<i>Secure Shell</i>	mrežni protokol
CSV	<i>Comma-Separated Values</i>	format za pohranu tabličnih podataka
RFE	<i>Recursive Feature Elimination</i>	metoda za odabir značajki
RFC	<i>RandomForestClassifier</i>	skupna metoda učenja za klasifikaciju
KNN	<i>K-Nearest Neighbour</i>	algoritam za klasifikaciju
LR	<i>Logistic Regression</i>	algoritam za klasifikaciju
DT	<i>Decision Tree</i>	algoritam za klasifikaciju
IOT	<i>Internet of Things</i>	povezivanje uređaja putem interneta