UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET
ODSJEK ZA AUTOMATIKU I ELEKTRONIKU

Algoritmi za generisanje sintetičkih podataka za finansije

ZAVRŠNI RAD

- PRVI CIKLUS STUDIJA -

Student:
Demir Hasičić

Mentor:
Vanr. prof. dr Amila Akagić

Sarajevo,
juli 2024.

Sažetak

U ovom radu su predstavljeni različiti alati za generisanje sintetičkih podataka, ističući njihov značaj i primjenu. Posebna pažnja posvećena je primjeni ovih podataka na detekciju prevarnih transakcija (*fraud detection*). Kroz rad su prikazani i različiti modeli mašinskog učenja korišteni za klasifikaciju, uz dodatne metode za unapređenje njihove učinkovitosti u detekciji prevara.

Ključne riječi: sintetički podaci, mašinsko učenje, detekcija prevara

Abstract

This paper presents various tools for generating synthetic data, highlighting their significance and application. Special attention is given to the use of these data in detecting fraudulent transactions (fraud detection). The study also showcases different machine learning models used for classification, along with additional methods to enhance their effectiveness in fraud detection.

Keywords: synthetic data, machine learning, fraud detection

Elektrotehnički fakultet, Univerzitet u Sarajevu
Odsjek za automatiku i elektroniku
Vanr. prof. dr Amila Akagić
Sarajevo, juli 2024.

Postavka zadatka završnog rada I ciklusa:
Algoritmi za generisanje sintetičkih podataka za finansije

Generisanje sintetičkih podataka bavi se stvaranjem podataka koji oponašaju stvarne podatke, a da pritom ne narušavaju privatnost ili povjerljivost originalnih podataka. Razvojem naprednih generativnih modela poput GAN-ova (Generative Adversarial Networks), moguće je kreirati sintetičke podatke koji su vrlo slični stvarnim podacima. Ovi sintetički podaci mogu se koristiti za treniranje modela mašinskog učenja bez potrebe za pristupom osjetljivim informacijama.

U ovom radu potrebno je prezentovati generisanje sintetičkih podataka, istaknuti njihov značaj, te istražiti njihovu primjenu u detekciji prevarnih transakcija (*fraud detection*). Rad obuhvata generisanje sintetičkih datasetova sa balansiranim brojem prevarnih i neprevarnih transakcija kako bi se poboljšala efikasnost modela mašinskog učenja. Također, prikazani su različiti modeli mašinskog učenja korišteni za klasifikaciju, sa osvrtom na njihov učinak.

Polazna literatura:

- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. arXiv preprint arXiv:1406.2661.
- Patki, N., Wedge, R., & Veeramachaneni, K. (2016). The Synthetic Data Vault. IEEE International Conference on Data Science and Advanced Analytics (DSAA), 399-410.
- Dal Pozzolo, A., Caelen, O., Boracchi, G., Alippi, C., & Bontempi, G. (2018). Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy. IEEE Transactions on Neural Networks and Learning Systems, 29(8), 3784-3797.

Vanr. prof. dr Amila Akagić

Univerzitet u Sarajevu
Elektrotehnički fakultet
Odsjek za automatiku i elektroniku

Izjava o autentičnosti radova

Završni rad
I ciklusa studija

Ime i prezime: Demir Hasičić
Naslov rada: Algoritmi za generisanje sintetičkih podataka za finansije
Vrsta rada: Završni rad Prvog ciklusa studija
Broj stranica: 73

Potvrđujem:

- da sam pročitao dokumente koji se odnose na plagijarizam, kako je to definirano Statutom Univerziteta u Sarajevu, Etičkim kodeksom Univerziteta u Sarajevu i pravilima studiranja koja se odnose na I i II ciklus studija, integrirani studijski program I i II ciklusa i III ciklus studija na Univerzitetu u Sarajevu, kao i uputama o plagijarizmu navedenim na web stranici Univerziteta u Sarajevu;
- da sam svjestan univerzitetskih disciplinskih pravila koja se tiču plagijarizma;
- da je rad koji predajem potpuno moj, samostalni rad, osim u dijelovima gdje je to naznačeno;
- da rad nije predat, u cjelini ili djelimično, za stjecanje zvanja na Univerzitetu u Sarajevu ili nekoj drugoj visokoškolskoj ustanovi;
- da sam jasno naznačio prisustvo citiranog ili parafraziranog materijala i da sam se referirao na sve izvore;
- da sam dosljedno naveo korištene i citirane izvore ili bibliografiju po nekom od preporučenih stilova citiranja, sa navođenjem potpune reference koja obuhvata potpuni bibliografski opis korištenog i citiranog izvora;
- da sam odgovarajuće naznačio svaku pomoć koju sam dobio pored pomoći mentora i akademskih tutora/ica.

Sarajevo, datum juli 2024.

Potpis:

_____

Demir Hasičić

Sadržaj

Popis slika

Popis tabela

Poglavlje 1

Uvod

1.1 Obrazloženje teme

Sintetički podaci su korisni u savremenim istraživanjima i industrijskim primjenama. Njihov značaj leži u mogućnosti kreiranja velikih skupova podataka koji oponašaju stvarne podatke, a da pritom ne ugrožavaju privatnost i povjerljivost originalnih informacija. Sintetički podaci se koriste zbog nekoliko razloga: smanjenje troškova i vremena potrebnog za prikupljanje stvarnih podataka, omogućavanje istraživanja u okruženjima gdje stvarni podaci nisu dostupni ili su pravno ograničeni, te za testiranje i validaciju modela mašinskog učenja bez rizika od izlaganja osjetljivih informacija.

Međutim, postoje ograničenja zbog kojih se sintetički podaci moraju koristiti. Prvo, prikupljanje stvarnih podataka može biti skupo i vremenski zahtjevno. Drugo, pravni i etički problemi često ograničavaju pristup stvarnim podacima. Konačno, stvarni podaci mogu biti nebalansirani, što predstavlja izazov za treniranje modela mašinskog učenja.

Modeli mašinskog učenja imaju poteškoće u učenju na nebalansiranim datasetovima jer su skloni favoriziranju većinske klase. Zbog toga je ključno generisati balansirane datasetove kako bi se poboljšala efikasnost i tačnost ovih modela. Problem detekcije prevarnih transakcija (*fraud detection*) je odabran zbog svoje praktične važnosti i složenosti. U 21. stoljeću broj korisnika kartica značajno se povećao, što je dovelo do porasta krađe novca i prevarnih transakcija. Ova pojava predstavlja ozbiljan problem, te je ključno imati modele mašinskog učenja koji mogu prepoznati prevarne transakcije s visokom tačnošću i blagovremeno obavijestiti korisnika.

Prema izvještajima, broj prevarnih transakcija raste svake godine, što naglašava važnost efikasnih modela za njihovo prepoznavanje. Na primjer, grafikon u nastavku prikazuje gubitke novca koji su posljedica prevarnih transakcija.

Slika 1.1: Trend rasta gubitaka novca zbog prevarnih transakcija u periodu 2010.-2020. [1]

Ovaj grafik jasno pokazuje rastući problem koji se tiče prevarnih transakcija, što dodatno opravdava potrebu za pouzdanim i preciznim modelima mašinskog učenja u ovoj oblasti.

1.2 Struktura rada

U ovoj sekciji bit će prikazan pregled svakog poglavlja u radu, ističući glavne doprinose i povezanost između poglavlja.

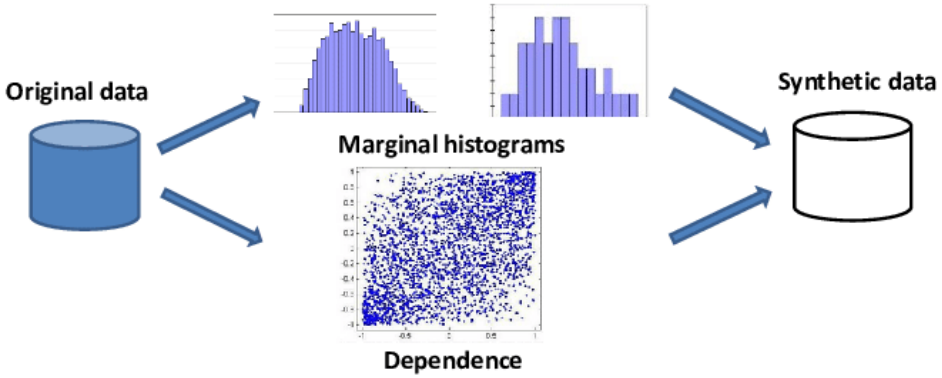
- Poglavlje **Uvod** pruža osnovne informacije o značaju sintetičkih podataka u modernim istraživanjima i industriji. Objašnjava se motivacija za korištenje sintetičkih podataka, kao i osnovni ciljevi rada.
- Poglavlje **Generisanje sintetičkih podataka i pregled postojećih alata** opisuje različite pristupe i alate za generisanje sintetičkih podataka. Također, izvršen je i pregled alata koji se koriste za generisanje sintetičkih podataka.
- Poglavlje **Korištenje sintetičkih podatka u svrhu detekcije prevarnih (fraud) transakcija** fokusira se na praktičnu primjenu sintetičkih podataka za detekciju prevara. Detaljno se opisuje problem detekcije prevara, metode evaluacije kvalitete modela, i treniranje modela na početnom neuravnoteženom datasetu. Dakle, posvećena je pažnja treniranju različitih modela mašinskog učenja, te analiza njihovih performansi.
- Poglavlje **Generisanje sintetičkih podataka** opisuje proces stvaranja sintetičkih podataka korištenjem SDV-a. Ovo uključuje kreiranje modela za sintetičke podatke, generisanje datasetova sa različitim omjerima prevarnih transakcija, i miješanje i spremanje sintetičkih podataka.

- Poglavlje **Treniranje modela na kombinovanim datasetovima** predstavlja proces treniranja modela na datasetovima koji kombinuju stvarne i sintetičke podatke. Vrši se evaluacija i analiza performansi različitih modela mašinskog učenja.
- Poglavlje **Zaključak** predstavlja sažetak ključnih tema i zaključaka iz pojedinih poglavlja.

Poglavlje 2

Generisanje sintetičkih podataka i pregled postojećih alata

Generisanje sintetičkih podataka predstavlja proces kreiranja umjetnih podataka koji oponašaju statističke karakteristike stvarnih podataka [5, 6]. U prethodnom poglavlju opisan je značaj sintetičkih podataka, te njihove primjena. Ovo poglavlje detaljnije će opisati postojeće alate i metode za generisanje sintetičkih podataka zasnovane na vještačkoj inteligenciji.



Slika 2.1: Generisanje sintetičkih podataka [2]

2.1 Pristupi generisanju sintetičkih podataka

Postoji nekoliko pristupa generisanju sintetičkih podataka, od kojih svaki ima svoje specifične metode i primjene. Ovi pristupi mogu se klasifikovati u nekoliko kategorija, kao što su: statistički pristupi, modeli zasnovani na pravilima, probabilistički modeli i neuronske mreže, poput Generative Adversarial Networks (GAN) i Recurrent Neural Networks (RNN) [7, 8].

2.1.1 Statistički pristupi za generisanje sintetičkih podataka

Statistički pristupi zasnivaju se na korištenju različitih statističkih modela i metoda za kreiranje novih podataka. Ovi modeli detaljno analiziraju distribucije, korelacije i druge statističke odnose unutar stvarnih podataka, a zatim koriste te informacije za generisanje sintetičkih podataka koji zadržavaju ključne karakteristike stvarnih podataka. Primjeri ovog pristupa su Monte Carlo simulacije i metode zasnovane na regresiji [5, 6].

Monte Carlo simulacije

Monte Carlo simulacije koriste slučajne uzorke za simulaciju i analizu složenih sistema. Ove metode mogu generisati podatke bazirane na unaprijed definisanim statističkim distribucijama [9, 10].

Metode zasnovane na regresiji

Metode zasnovane na regresiji koriste regresione modele za predviđanje novih podataka na osnovu postojećih uzoraka. Ove metode analiziraju odnose između zavisnih i nezavisnih promjenljivih kako bi kreirale modele koji mogu generisati nove podatke [11].

2.1.2 Modeli bazirani na pravilima

Modeli bazirani na pravilima definišu skup pravila i logiku koja se koristi za generisanje podataka. Ovi modeli često uključuju složene simulacije interakcija između različitih entiteta. Primjeri ovog pristupa su Agent-based modeling (ABM) i Rule-based systems [12, 13, 14].

Agent-based modeling

Agent-based modeling (ABM) simulira ponašanje i interakcije autonomnih agenata kako bi generisao podatke. ABM je posebno koristan za simulaciju složenih sistema, kao što su ekonomski ili društveni sistemi [12, 13, 14].

Rule-based systems

Rule-based systems koriste unaprijed definisana pravila za generisanje podataka, što omogućava preciznu kontrolu nad karakteristikama generisanih podataka [13].

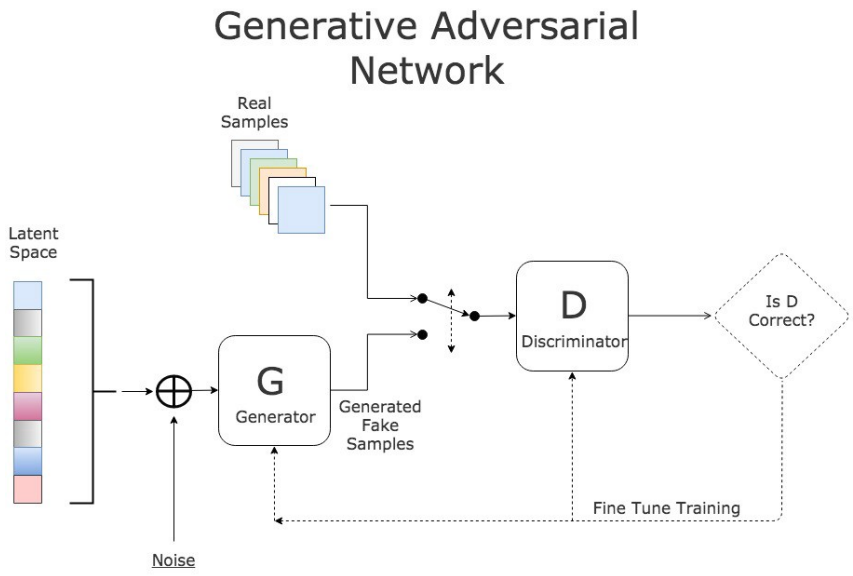
2.1.3 Neuronske mreže

Već pomenute neuronske mreže koje se koriste u svrhu generisanja sintetičkih podataka su GAN i RNN [15].

Generative adversarial networks (GAN)

GAN su napredne neuronske mreže koje uključuju dva modela: generator i diskriminator. Generator kreira sintetičke podatke, dok diskriminator pokušava da razlikuje sintetičke od stvarnih podataka. Kroz iterativni proces, oba modela se međusobno unapređuju, što rezultira visoko kvalitetnim sintetičkim podacima [15]. Arhitektura GAN-a evoluirala je kroz vrijeme, pa tako razlikujemo:

- **Vanilla GAN**: Osnovna struktura GAN-a koja koristi generator i diskriminator za generisanje novih podataka [15].
- **Conditional GAN (cGAN)**: Proširuje osnovni GAN model dodavanjem uslovnih informacija koje usmjeravaju generisanje podataka [16].
- **CTGAN**: Specijalizovani GAN za generisanje tabularnih podataka, koji uključuje i kontinuirane i kategorijske varijable [17].



Slika 2.2: Struktura GAN-a [3]

Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNN), a posebno njihove LSTM (Long Short-Term Memory) varijante, koriste se za generisanje sekvencijalnih podataka, poput vremenskih serija ili tekstova. Ovi modeli su jedinstveni po tome što mogu učiti i generisati sekvence podataka koje zavise od prethodnih stanja, što ih čini posebno korisnim za aplikacije gde je redoslijed informacija ključan [18].

2.1.4 Variational Autoencoders (VAE)

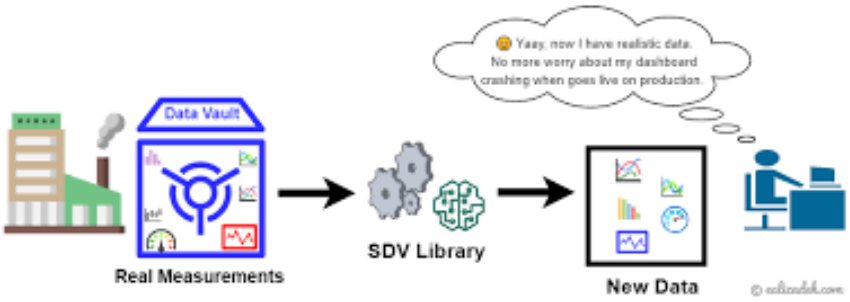
Variational Autoencoders (VAEs) predstavljaju vrstu generativnih modela koji kombinuju karakteristike autoenkodera sa probabilističkim modelima. Dizajnirani su tako da uče latentne reprezentacije podataka, koje se potom koriste za kreiranje novih uzoraka. Ova sposobnost čini VAEs izuzetno korisnim za generisanje sintetičkih podataka koji vjerno odražavaju karakteristike originalnog skupa podataka [19].

2.2 Pregled alata za generisanje sintetičkih podataka

U ovom poglavlju bit će prikazani alati koji su razvijeni za generisanje sintetičkih podataka, koji mogu poslužiti za različite svrhe. Ovi alati se oslanjaju na različite metodologije, od jednostavnih pravila do složenih generativnih modela poput GAN-ova i VAE-a. Svaki alat donosi specifične mogućnosti i prilagođava se različitim potrebama. U nastavku će biti prikazani alati: Synthetic data vault (SDV), Faker i Gretel Synthetics.

2.2.1 Synthetic Data Vault (SDV)

Synthetic Data Vault (SDV) je složen alat koji koristi kombinaciju generativnih modela i statističkih tehnika za stvaranje sintetičkih podataka. SDV uključuje nekoliko ključnih komponenti koje zajednički rade na generisanju podataka koji zadržavaju ključne karakteristike originalnih podataka [20].



Slika 2.3: Synthetic Data Vault [4]

Primjene SDV-a

SDV je razvijen kako bi prevazišao izazove vezane za privatnost, sigurnost i dostupnost stvarnih podataka. Koristi se u raznim industrijama i istraživačkim domenima. Sintetički podaci generisani pomoću SDV-a koriste se za obuku modela mašinskog učenja, testiranje softvera, analizu podataka i simulacije. Neke ključne primjene uključuju:

- Finansije i detekcija prevara
- Zdravstvo
- E-trgovina
- Simulacije i prediktivna analitika

[21]

Glavne Komponente SDV-a

SDV se sastoji od sljedećih komponenti[20] :

- **Data Profiler**: Analizira originalne podatke kako bi identifikovao njihove statističke karakteristike, distribucije i relacije između atributa. Ove karakteristike se koriste za treniranje generativnih modela.
- **Modeler**: Koristi generativne modele kao što su CTGAN, CopulaGAN, Gaussian Mixture Models (GMM) i Bayesian Networks za modeliranje podataka.
- **Sampler**: Nakon treniranja modela, ova komponenta generiše nove sintetičke podatke.
- **Evaluator**: Procjenjuje kvalitet sintetičkih podataka poredeći ih sa originalnim podacima kako bi osigurao da sintetički podaci adekvatno reprezentuju stvarne podatke.

Povezanost Sistema

Efikasno funkcionisanje SDV-a zavisi od povezanosti njegovih komponenti, a proces generisanja sintetičkih podataka izgleda ovako: [20]

- **Data Profiler do Modeler-a**: Data Profiler prenosi statističke karakteristike i distribucije podataka Modeler komponenti, koja koristi ove informacije za treniranje generativnih modela .

- **Modeler do Sampler-a**: Nakon treniranja, Modeler prenosi trenirane modele Sampler komponenti, koja koristi modele za generisanje novih sintetičkih podataka.
- **Sampler do Evaluator-a**: Sampler dostavlja generisane sintetičke podatke Evaluator-u, koji upoređuje sintetičke podatke sa originalnim podacima kako bi procijenio njihov kvalitet.
- **Evaluator do Data Profiler-a i Modeler-a**: Evaluator pruža povratne informacije Data Profiler i Modeler komponentama za dodatno fino podešavanje modela i poboljšanje kvaliteta sintetičkih podataka.

Modeliranje

Različiti generativni modeli imaju svoje prednosti i nedostatke, a izbor pravog modela zavisi od karakteristika podataka i ciljeva generisanja sintetičkih podataka. Generativni modeli korišteni u SDV-u su:

- **Conditional Tabular GAN (CTGAN)** je varijanta GAN-a (Generative Adversarial Network) posebno dizajnirana za generisanje tabularnih podataka. CTGAN se bavi izazovima koji nastaju prilikom rada sa tabularnim podacima koji sadrže različite tipove varijabli, kao što su kontinuirane i kategorijske varijable [17]. Glavne komponente CTGAN-a su:
  - **Generator**: Kreira sintetičke podatke nastojeći da što bolje oponaša distribuciju stvarnih podataka.
  - **Diskriminator**: Za zadatak ima da razlikuje sintetičke podatke od stvarnih, podstičući generator da proizvodi realističnije podatke.
  - **Trening proces**: Trening je iterativan proces u kojem se generator i diskriminator naizmjenično unapređuju kroz više iteracija.
- **CopulaGAN** koristi koncept *kopula* iz statistike za modeliranje zavisnosti među varijablama, što omogućava bolju kontrolu nad korelacijama i distribucijama podataka [22]. Komponente CopulaGAN-a su:
  - **Copula modeli**: Modeluju zavisnosti između varijabli pomoću kopula funkcija, omogućavajući generisanje podataka sa složenim međuzavisnostima [22].
  - **GAN okvir**: Vrši integraciju kopula modela u GAN okvir za generisanje sintetičkih podataka [17].
- **Gaussian Mixture Models (GMM)** se koriste za modeliranje podataka pomoću mješavine više Gaussovih distribucija. Ovaj pristup je koristan za generisanje podataka sa multimodalnim distribucijama [23]. Glavni dijelovi GMM-a su:
  - **EM algoritam**: Koristi Expectation-Maximization (EM) algoritam za procjenu parametara GMM-a [24].
  - **Generisanje podataka**: Nakon treniranja, koristi procijenjene parametre za generisanje novih podataka [23].
- **Bayesian Networks (Bayesove mreže)** koriste probabilističke grafičke modele za predstavljanje zavisnosti između varijabli. Komponente Bayesovih mreža uključuju: [25]

  - **Učenje strukture**: Identifikovanje zavisnosti između varijabli kroz učenje strukture mreže.
  - **Učenje parametara**: Procjena vjerovatnoća na osnovu podataka.
  - **Generisanje podataka**: Korištenje naučene strukture i parametara za generisanje novih podataka.

2.2.2 Faker

Faker je popularan alat za generisanje lažnih podataka u razne svrhe, poput testiranja softvera, populacije baza podataka i simulacije. Faker se koristi za kreiranje raznih tipova podataka, uključujući imena, adrese, brojeve telefona, datume, tekstove i druge podatke [26].

Primjene Fakera

Faker se najčešće koristi u sljedećim scenarijima:

- **Testiranje softvera**: Generisanje testnih podataka za razvoj i testiranje aplikacija.
- **Populacija baza podataka**: Punjenje baza podataka s realistično lažnim podacima za testiranje i razvoj.
- **Simulacije**: Kreiranje lažnih podataka za simulacije i analize.

Glavne Karakteristike Fakera

Faker nudi širok spektar funkcionalnosti, uključujući: [26]

- **Raznovrsni podaci**: Faker može generisati raznovrsne tipove podataka, od jednostavnih tekstualnih do složenih numeričkih i datuma.
- **Jednostavnost korištenja**: Alat je jednostavan za upotrebu i integraciju u razne projekte.
- **Fleksibilnost**: Faker omogućava prilagođavanje generisanih podataka prema specifičnim potrebama korisnika.

2.2.3 Gretel Synthetics

Gretel Synthetics je napredni alat za generisanje sintetičkih podataka, koji koristi mašinsko učenje kako bi stvorio podatke visoke kvalitete. Koristi se za obuku modela, testiranje softvera i zaštitu privatnosti podataka [27].

Glavne Karakteristike Gretel Synthetics

Gretel Synthetics se izdvaja po: [27]

- **Visok kvalitet podataka**: Generiše podatke koji zadržavaju ključne karakteristike stvarnih podataka.
- **Zaštita privatnosti**: Osigurava da sintetički podaci ne otkrivaju osjetljive informacije.
- **Jednostavna integracija**: Može se lako integrisati s postojećim alatima i procesima.

Generisanje sintetičkih podataka postalo je ključna tehnologija u mnogim industrijama, omogućavajući razvoj i testiranje aplikacija bez rizika po privatnost korisnika. Alati kao što su SDV, Faker i Gretel Synthetics nude različite pristupe i metode za stvaranje kvalitetnih sintetičkih podataka, prilagođenih specifičnim potrebama i zahtjevima. Korištenje ovih alata može značajno unaprijediti procese razvoja, testiranja i analize u različitim domenima.

Poglavlje 3

Korištenje sintetičkih podatka u svrhu detekcije prevarnih (*fraud*) transakcija

U prethodnom poglavlju dat je prikaz alata koji se koriste za generisanje sintetičkih podataka, dok će se u ovom poglavlju predstaviti konkretna primjena jednog od navedenih alata, u ovom slučaju SDV-a, za detekciju prevarnih transakcija.

3.1 Opis problema detekcije prevarnih (*fraud*) transakcija

Detekcija prevarnih transakcija je ključna za zaštitu korisnika i financijskih institucija od gubitaka. Međutim, zbog visoke neuravnoteženosti podataka, ovaj problem je prilično izazovan. Upravo zbog navedene neuravnoteženosti može doći do problema sa treniranjem modela, jer većina modela mašinskog učenja teži ka učenju dominantne klase (u ovom slučaju validnih transakcija), što može rezultirati lošim performansama u detekciji stvarnih prevara. Korištenjem sintetičkih podataka omogućava se prevazilaženje navedenog problema, jer je moguće kreirati sintetički dataset koji će imati dovoljan broj prevarnih transakcija, kako bi se model mogao naučiti da ih bolje prepoznaje.

3.1.1 Početni dataset

Za potrebe ovog rada bit će korišten dataset koji predstavlja transakcije kreditnih kartica europskih korisnika u septembru 2013. godine. Dataset sadrži transakcije koje su se dogodile tokom dva dana, s 492 prevarne transakcije od ukupno 284,807 transakcija. Dataset je vrlo neuravnotežen, s pozitivnom klasom (prevare) koja čini samo 0.172% svih transakcija. Dataset sadrži samo numeričke ulazne varijable koje su rezultat PCA transformacije. Zbog problema povjerljivosti, originalne značajke nisu dostupne. Značajke V1, V2, ... V28 su glavne komponente dobivene PCA-om, dok značajke 'Time' i 'Amount' nisu transformirane PCA-om. Značajka 'Class' je ciljna varijabla koja ima vrijednost 1 u slučaju prevare i 0 u suprotnom [28, 29, 30, 31, 32]. Principal Component Analysis (PCA) je tehnika smanjenja dimenzionalnosti koja se koristi za transformaciju velikog skupa varijabli u manji skup glavnih komponenti, dok se zadržava što je moguće više informacija iz originalnog skupa podataka [33]. Pošto je dataset transformisan pomoću PCA, nije potrebno vršiti predprocesiranje podataka [34]. Struktura opisanog dataseta prikazana je na slici 3.1.

	Time	V1	V2	V3	V4	V5	V6	V7	\
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	

	V8	V9	V10	V11	V12	V13	V14	\
0	0.098698	0.363787	0.090794	-0.551600	-0.617801	-0.991390	-0.311169	
1	0.085102	-0.255425	-0.166974	1.612727	1.065235	0.489095	-0.143772	
2	0.247676	-1.514654	0.207643	0.624501	0.066084	0.717293	-0.165946	
3	0.377436	-1.387024	-0.054952	-0.226487	0.178228	0.507757	-0.287924	
4	-0.270533	0.817739	0.753074	-0.822843	0.538196	1.345852	-1.119670	

	V15	V16	V17	V18	V19	V20	V21	\
0	1.468177	-0.470401	0.207971	0.025791	0.403993	0.251412	-0.018307	
1	0.635558	0.463917	-0.114805	-0.183361	-0.145783	-0.069083	-0.225775	
2	2.345865	-2.890083	1.109969	-0.121359	-2.261857	0.524980	0.247998	
3	-0.631418	-1.059647	-0.684093	1.965775	-1.232622	-0.208038	-0.108300	
4	0.175121	-0.451449	-0.237033	-0.038195	0.803487	0.408542	-0.009431	

	V22	V23	V24	V25	V26	V27	V28	\
0	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	
1	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	
2	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	
3	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	
4	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	

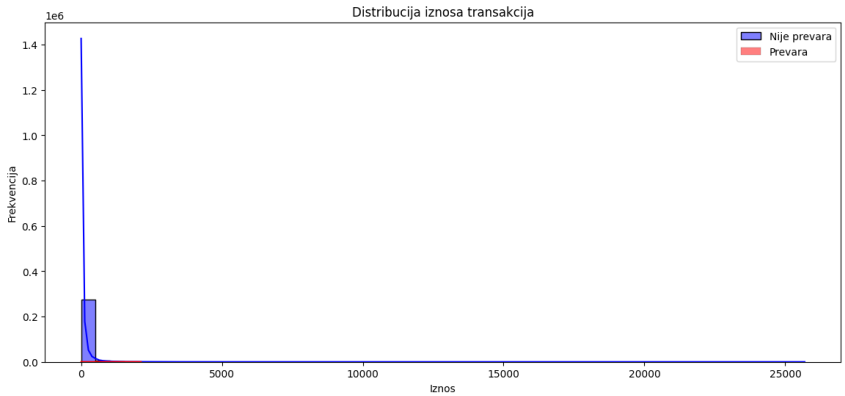
	Amount	Class	hour	Hour	Sat
0	149.62	0	0.0	0.0	0.0
1	2.69	0	0.0	0.0	0.0
2	378.66	0	1.0	1.0	1.0
3	123.50	0	1.0	1.0	1.0
4	69.99	0	1.0	1.0	1.0

Slika 3.1: Početni dataset

Karakteristika	**Opis**	**Tip**
V1 - V28	Glavne komponente dobivene PCA-om	Numerički
Time	Vrijeme transakcije	Numerički
Amount	Iznos transakcije	Numerički
Class	Oznaka klase (0 = validna, 1 = prevara)	Kategorijalni

Tabela 3.1: Osnovni opis početnog dataseta

Vizualizacija raspodjela transakcija prikazana je na sljedećoj slici 3.2



Slika 3.2: Distribucija iznosa transakcija

3.1.2 Metode evaluacije kvalitete modela

Kao što je već ranije rečeno, problem detekcije prevarnih transakcija najčešće leži u činjenici da modeli koji bi se trenirali na nebalansovanim datasetovima, gdje je broj validnih transakcija dosta veći, ne bi bili u mogućnosti kvalitetno prepoznati prevarne transakcije. U ovom dijelu napravit će se evaluacija modela treniranog na ovakvom datasetu, te se prikazati njegovi nedostaci pomoću nekoliko metrika kao što su precision (preciznost), recall (odziv), F1 score, AUPRC (Površina ispod Precision-Recall Krive), te ROC-AUC (Površina ispod Receiver Operating Characteristic Krive).

Precision (preciznost)

Preciznost je omjer tačno predviđenih pozitivnih slučajeva i ukupno predviđenih pozitivnih slučajeva. Govori o tome koliko je zapravo bilo tačnih prevara od ukupnog broja predviđenih prevara [35].

$$\text{Preciznost} = \frac{TP}{TP+FP} \tag{3.1}$$

TP = True Positive (ispravno predviđeni pozitivni slučajevi)
FP = False Positive (neispravno predviđeni pozitivni slučajevi)

Recall (odziv)

Recall (odziv) je omjer tačno predviđenih pozitivnih slučajeva i svih stvarnih pozitivnih slučajeva. Predstavlja broj ispravno predvidjelih slučajeva prevare od ukupnog broja stvarnih prevarnih transakcija [36].

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3.2}$$

TP = True Positive
FN = False Negative (stvarni pozitivni slučajevi koji su pogrešno predviđeni kao negativni)

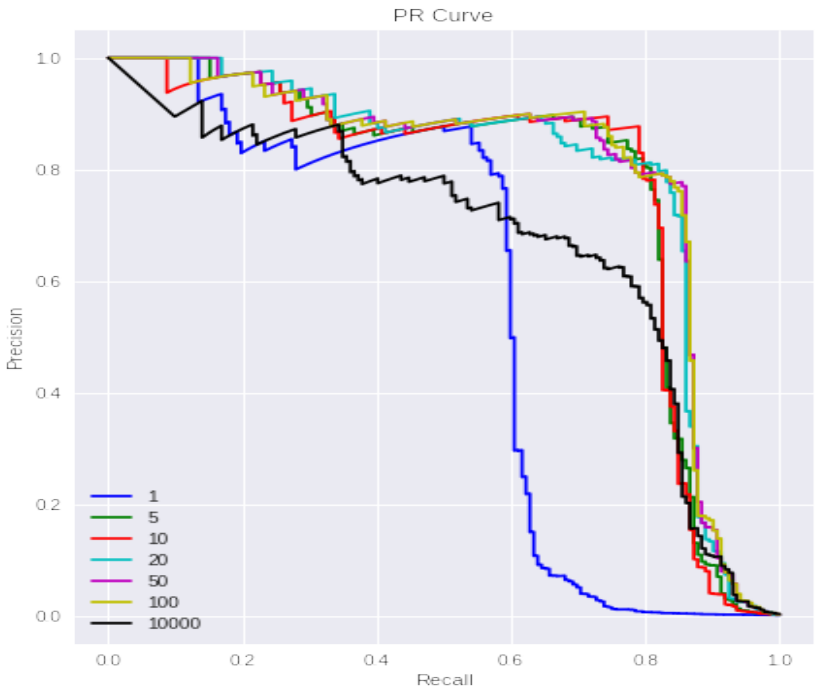
F1 score

F1 score predstavlja harmonijsku sredinu preciznosti i odziva. Ova metrika uzima u obzir i lažno pozitivna i lažno negativna predviđanja. Posebno je korisna kada je potrebno postići balans između preciznosti i odziva [37].

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.3}$$

AUPRC (Površina ispod Precision-Recall krive)

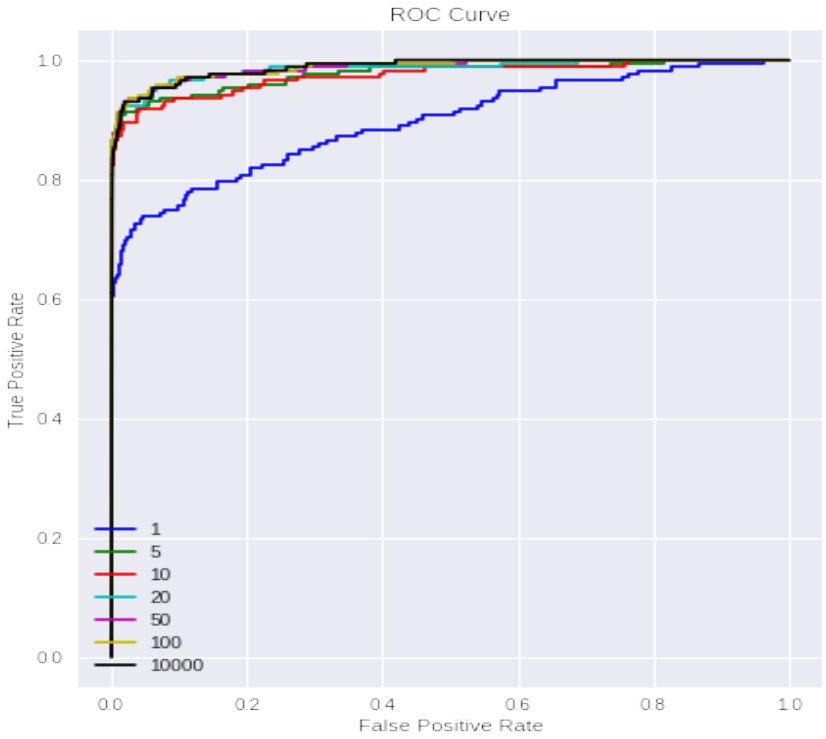
Površina ispod Precision-Recall krive (AUPRC) sumira odnos između tačno pozitivnih i pozitivno predviđenih vrijednosti za model koristeći različite pragove vjerovatnoće. Veći AUPRC ukazuje na bolje performanse modela. Precision-Recall kriva je posebno korisna za neuravnotežene skupove podataka jer se fokusira na performanse u odnosu na pozitivnu klasu što je u ovom slučaju bitno, jer su kao pozitivna klasa označene prevarne transakcije [38].



Slika 3.3: Primjer Precision-Recall krive

ROC-AUC (Površina ispod Receiver Operating Characteristic krive)

ROC-AUC je površina ispod ROC krive, koja prikazuje True Positive Rate (Recall) u odnosu na False Positive Rate (1 - Specifičnost) pri različitim pragovima. Pruža agregatnu mjeru performansi modela kroz sve moguće pragove klasifikacije [39].



Slika 3.4: Primjer ROC krive

U slučajevima neuravnoteženih skupova podataka, gdje je jedna klasa (najčešće negativna klasa) znatno zastupljenija od druge (pozitivne klase), standardne metrike poput tačnosti i ROC-AUC mogu biti varljive, pa je AUPRC (Area Under the Precision-Recall Curve) češće vjerodostojnija metrika. Razlozi zbog kojih je AUPRC često bolja metrika su:

- **Osjetljivost na neuravnoteženost**: ROC-AUC može biti varljiva jer FPR (False Positive Rate) uključuje veliki broj negativnih instanci. Kada je negativna klasa dominantna, čak i mali broj lažno pozitivnih može izgledati dobro na ROC krivoj. Suprotno tome, AUPRC se fokusira na pozitivne klase i pruža bolji uvid u performanse modela u identifikaciji stvarno pozitivnih slučajeva [40, 41].
- **Fokus na relevantne metrike**: AUPRC direktno mjeri preciznost i odziv, što su ključne metrike kod neuravnoteženih podataka. Visoka preciznost i odziv su kritični za dobru performansu modela, naročito kada je broj pozitivnih slučajeva mali [41].
- **Praktična upotreba**: U mnogim stvarnim aplikacijama, kao što su detekcija prevara, medicinska dijagnoza i slično, cilj je minimizirati lažno pozitivne i lažno negativne predikcije. AUPRC pomaže da se bolje balansira između ta dva aspekta [40, 41].

Ukratko, AUPRC je često najbolja metrika za evaluaciju modela na neuravnoteženim skupovima podataka jer daje jasniji i precizniji prikaz sposobnosti modela da prepoznaje pozitivne slučajeve bez obzira na neuravnoteženost podataka.

3.2 Treniranje modela nad početnim datasetom

Za treniranje nad početnim datasetom odabrani su modeli: Random Forest Classifier (RFC), Neuronske mreže (MLP), XGBoost (XGB), Logistička Regresija (LR) i Light GBM (LGB).

Logistička regresija (LR)

Logistička regresija je osnovna metoda klasifikacije koja se često koristi zbog svoje jednostavnosti i interpretabilnosti. Ova metoda koristi logističku funkciju, poznatu i kao sigmoidna funkcija, za modeliranje vjerojatnosti binarne zavisne varijable [11].
Matematički, logistička regresija modelira vjerojatnost pripadanja pozitivnoj klasi, koristeći sigmoidnu funkciju:

$$P(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b) = \frac{1}{1 + \exp(-(\mathbf{w}^\top \mathbf{x} + b))}$$

gdje je σ sigmoidna funkcija, $\mathbf{w}$ vektor težina, a b je pomak (intercept).

Cilj logističke regresije je maksimizirati log-likelihood funkciju koja se dobije uzimanjem prirodnog logaritma vjerojatnosti posmatranih podataka:

$$\ell(\mathbf{w}, b) = \sum_{i=1}^{n} [y_i \log P(y_i = 1 \mid \mathbf{x}_i) + (1 - y_i) \log(1 - P(y_i = 1 \mid \mathbf{x}_i))]$$

gdje y_i predstavlja stvarne oznake klase, a $\mathbf{x}_i$ ulazne podatke za primjer i.
Maksimizacijom ove funkcije nalazimo optimalne parametre $\mathbf{w}$ i b koji maksimiziraju vjerojatnost posmatranih podataka [42].

Random Forest Classifier (RFC)

Random Forest je ansambl metoda koja koristi više stabala odlučivanja i agregira njihove rezultate radi poboljšanja tačnosti i kontroliranja overfittinga. Ova metoda kreira niz nezavisnih stabala odlučivanja na različitim podskupovima podataka i koristi prosjek njihovih predikcija za donošenje konačne odluke [43].
Za svaki uzorak $\mathbf{x}_i$, predikcija je prosjek predikcija svih B stabala u ansamblu:

$$\hat{y}_i = \frac{1}{B} \sum_{b=1}^{B} f_b(\mathbf{x}_i)$$

gdje je f_b predikcija b-tog stabla. Ovaj pristup pomaže u smanjenju varijanse modela i povećava ukupnu stabilnost i tačnost predikcija [43].

XGBoost (XGB)

XGBoost je model baziran na gradijentnom pojačanju koji iterativno gradi modele fokusirajući se na teško klasificirane instance. Ovakav pristup mu omogućava da bolje prepoznaje složene obrasce u podacima i efikasno se nosi s problemom overfittinga [44].
Matematički, XGBoost optimizira sljedeću funkciju cilja:

$$\mathscr{L}(\theta) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

gdje je l funkcija gubitka koja mjeri razliku između predikcija $\hat{y}_i$ i stvarnih vrijednosti y_i, a Ω je regularizacijski termin za kontrolu složenosti modela. Predikcija u t iteraciji je data sa:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)$$

gdje je f_t funkcija stabla odlučivanja u t iteraciji [44].

Neuronske mreže (MLP Klasifikator)

Neuronske mreže su napredne metode mašinskog učenja koje mogu modelirati složene nelinearne odnose u podacima. MLP (Multilayer Perceptron) klasifikator je vrsta potpuno povezane neuronske mreže koja se sastoji od više slojeva neurona, uključujući ulazni sloj, jedan ili više skrivenih slojeva i izlazni sloj. Svaki neuron u skrivenom sloju primjenjuje nelinearnu aktivacijsku funkciju kako bi naučio složene obrasce u podacima [7].

Matematički, proces u MLP klasifikatoru može se opisati sljedećim koracima:

1. **Ulazni sloj**: Ulazne značajke $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ se prosljeđuju kroz mrežu.

2. **Skriveni sloj**: Svaki neuron u skrivenom sloju prima ulaz iz svih neurona prethodnog sloja. Aktivacija a_j neurona j u skrivenom sloju se računa kao:

$$a_j = f\left(\sum_{i=1}^{n} w_{ji} x_i + b_j\right)$$

   gdje su w_{ji} težine povezane s ulazima, b_j su pristranosti (bias) neurona, a f je aktivacijska funkcija, često ReLU (Rectified Linear Unit) ili sigmoidna funkcija.

3. **Izlazni sloj**: Aktivacije iz skrivenog sloja prosljeđuju se izlaznom sloju. Aktivacija o_k neurona k u izlaznom sloju se računa kao:

$$o_k = g\left(\sum_{j=1}^{m} w_{kj} a_j + b_k\right)$$

gdje su w_{kj} težine povezane sa skrivenim slojem, b_k su pristranosti neurona u izlaznom sloju, a g je aktivacijska funkcija, često sigmoidna ili softmax funkcija za klasifikacione zadatke.

4. **Funkcija greške (Loss function)**: Razlika između predviđenih vrijednosti i stvarnih oznaka klase računa se pomoću funkcije greške, često korištena binarna unakrsna entropija (binary cross-entropy) za binarnu klasifikaciju:

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

gdje su y_i stvarne oznake klase, $\hat{y}_i$ predviđene vrijednosti, a N je broj primjera u datasetu [7].

5. **Unazadna propagacija (Backpropagation)**: Proces treniranja koristi algoritam unazadne propagacije za minimiziranje funkcije greške. Težine i pristranosti se ažuriraju koristeći gradijentni spust (gradient descent):

$$w_{ji} \leftarrow w_{ji} - \eta \frac{\partial \text{Loss}}{\partial w_{ji}}$$

$$b_j \leftarrow b_j - \eta \frac{\partial \text{Loss}}{\partial b_j}$$

gdje je η brzina učenja (learning rate).

Ovaj matematički pristup omogućava MLP klasifikatoru da nauči složene nelinearne obrasce u podacima, čineći ga moćnim alatom za klasifikacione zadatke [7].

LightGBM (LGBM)

LightGBM je framework za gradient boosting koji koristi histogram-based algoritam za brzo treniranje. LightGBM optimizira funkciju cilja sličnu XGBoost-u [45]:

$$\mathscr{L}(\theta) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

gdje je l funkcija gubitka, a Ω regularizacijski termin. LightGBM koristi histogram-based algoritam za efikasno binning feature-a:

$$\text{Histogram}(x) = \sum_i I(x_i \in \text{bin}_j)$$

gdje je bin_j j-ti bin [45].

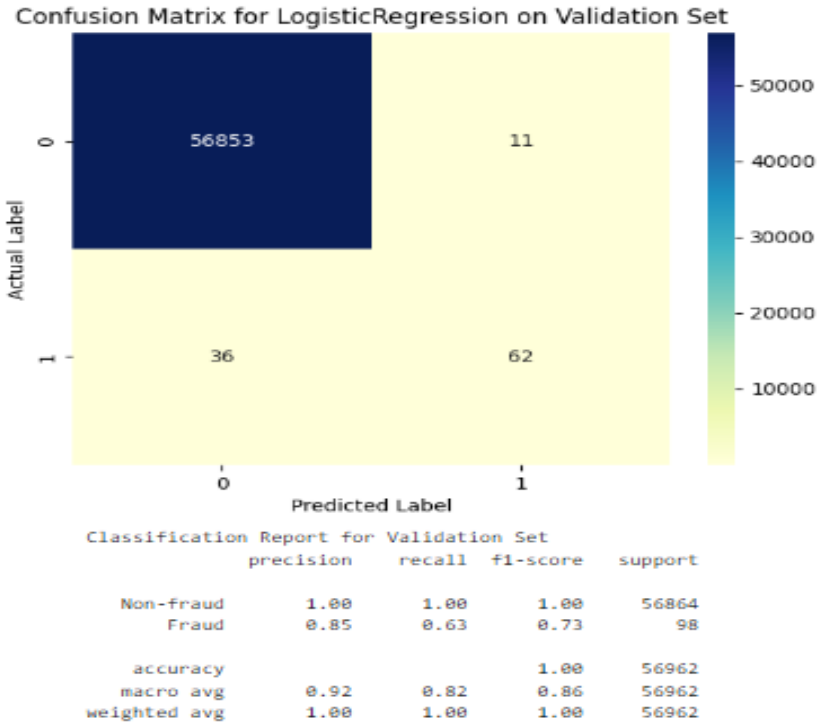
Svaki od navedenih modela će odvojeno biti trenirani na dva skupa podataka, na početnom datasetu, te datasetovima koji su nastali spajanjem početnog dataseta sa sintetički generisanim datasetovima. U nastavku će biti prikazana usporedba ovih modela.

3.2.1 Rezultati treniranja modela nad početnim datasetom

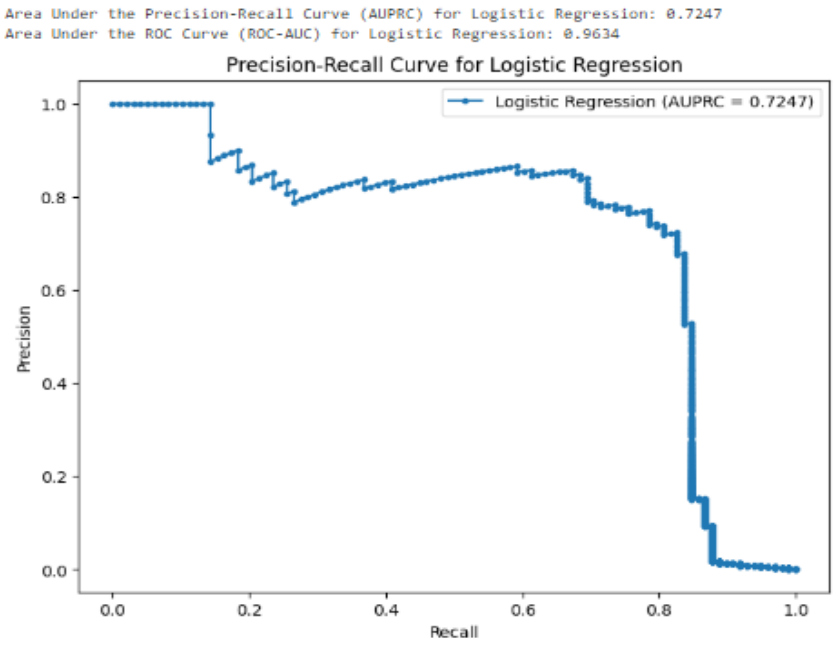
Ovaj dio rada prikazuje rezultate treniranja svakog modela na početnom, neuravnoteženom skupu podataka. Evaluacija modela će se provesti koristeći prethodno navedene metrike, s posebnim naglaskom na AUPRC kao najvažniju metriku.

Logistička regresija (LR)

U nastavku prikazan je rezultat testiranja modela, te vrijednosti precision, recall, F1 score, AUPRC i ROC-AUC, kao i grafik na kojem je prikazan AUPRC.



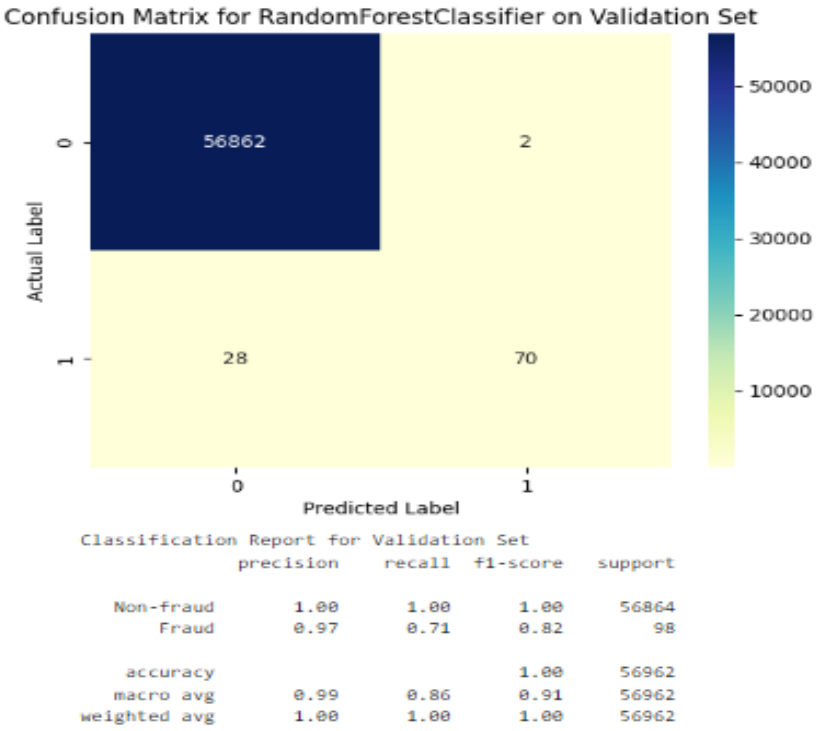
Slika 3.5: Rezultat klasifikacije logističke regresije



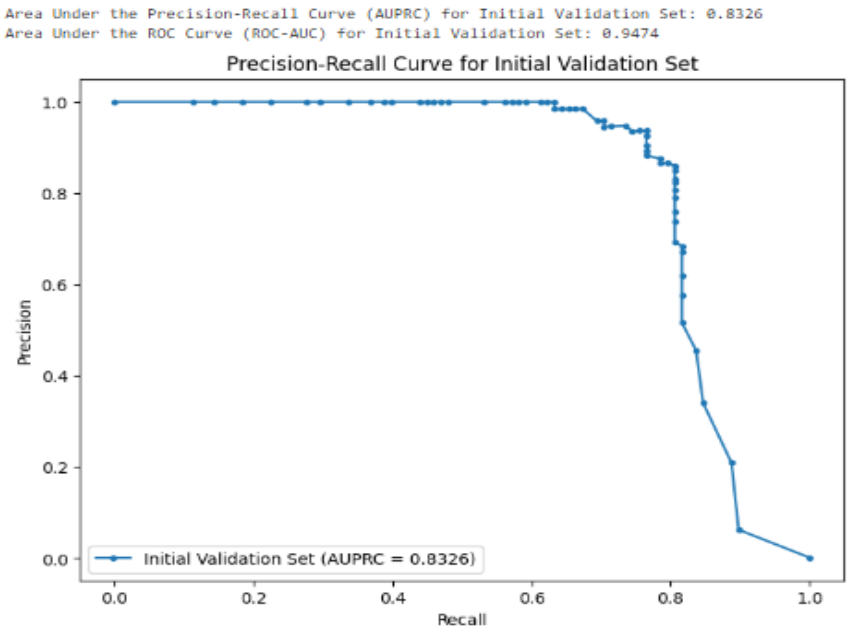
Slika 3.6: Precision-Recall kriva logističke regresije

Random Forrest Classifier (RFC)

Ovaj model je prepoznatljiv po svojoj otpornosti na overfitting i učinkovitosti u radu s neuravnoteženim datasetovima, te daje bolje rezultate u poređenju sa logističkom regresijom [46, 47]. Rezultati su prikazani na slikama 3.7 i 3.8.



Slika 3.7: Rezultat klasifikacije RFC-a

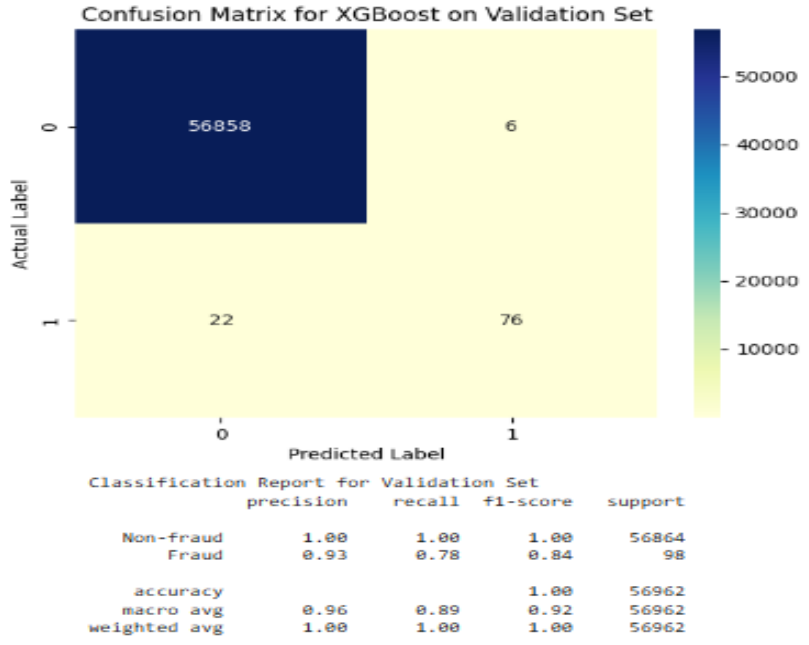


Slika 3.8: Precision-Recall kriva RFC-a

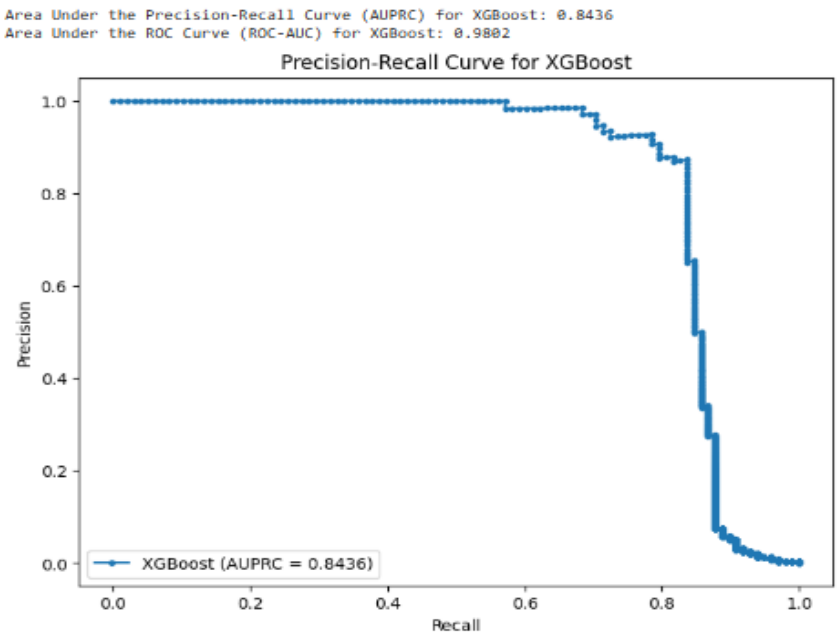
Moguće je uočiti da je RFC, zbog činjenice da je dobro prilagođen radu sa neuravnoteženim datasetovima, ostvario bolji rezultat.

XGBoost (XGB)

Ovaj model je poznat po svojoj izuzetnoj performansi u mnogim sferama mašinskog učenja, a rezultati treniranja su prikazani na slici 3.9.



Slika 3.9: Rezultat klasifikacije XGB-a

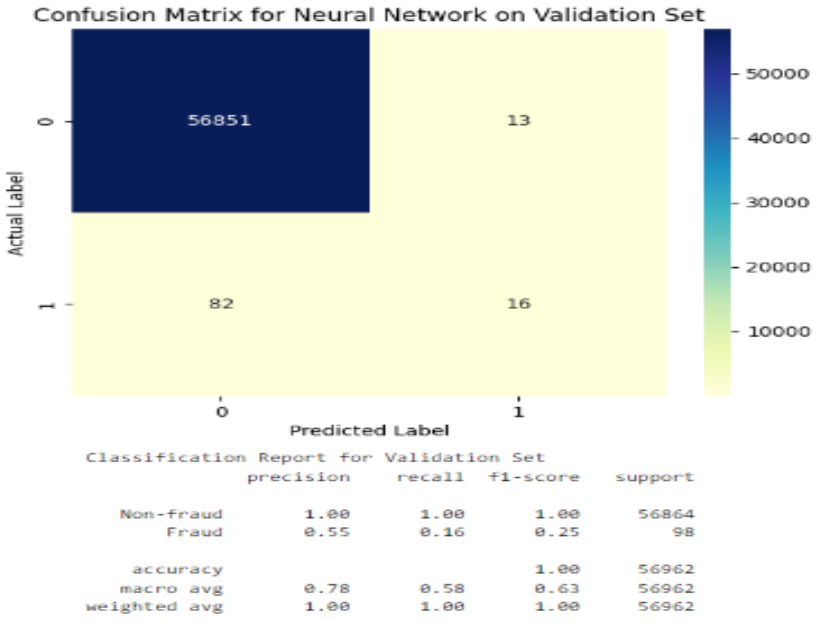


Slika 3.10: Precision-Recall kriva XGB-a

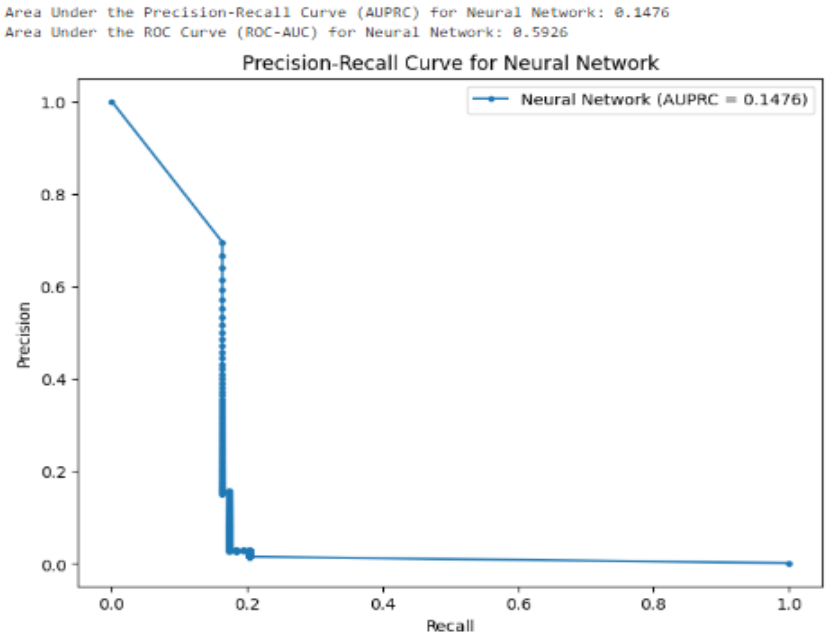
Slike prikazuju da je XGBoost izuzetno efikasan pri radu s neuravnoteženim datasetovima, te je ovaj model ostvario solidne rezultate na početnom datasetu.

Neuronske Mreže (MLPClassifier)

Neuronske mreže predstavljaju napredniji model klasifikacije koji se koristi za složenije zadatke. Rezultati modela su prikazani na slikama 3.11 i 3.12.



Slika 3.11: Rezultat klasifikacije MLP-a

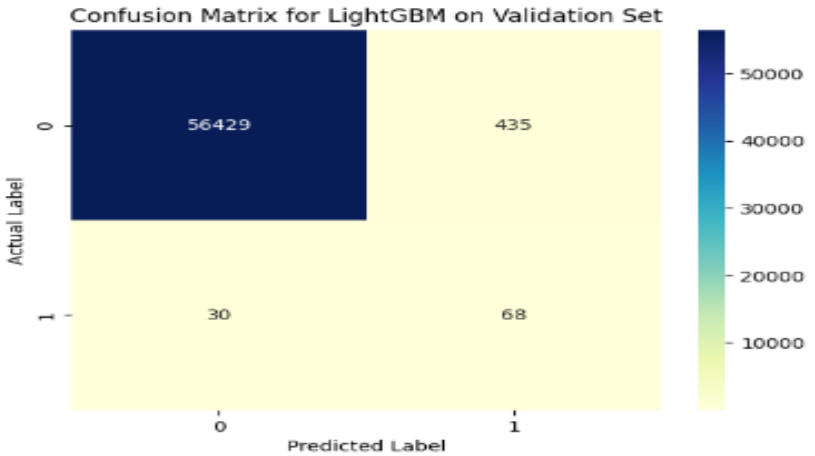


Slika 3.12: Precision-Recall kriva MLP-a

Međutim, iako kvalitetan i efikasan model klasifikacije, zbog lošeg balansa podataka, MLP klasifikator je ostvario prilično loše rezultate, jer nije prilagođen nebalansovanim datasetovima.

LightGBM (LGB)

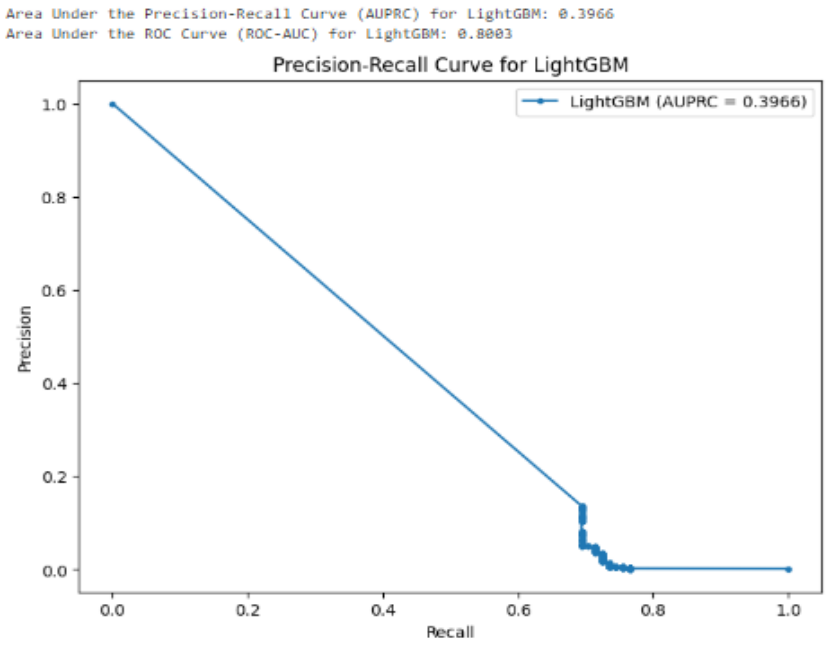
LightGBM je sljedeći model čiji će se rezultati prikazati. Ovaj model se pokazao izuzetno brzim i efikasnim za velike datasetove, te je za njegovo izvršavanje trebalo dosta manje vremena, u poređenju sa ostalim modelima. Rezultati su prikazani na slikama 3.13 i 3.14.



Classification Report for Validation Set

	precision	recall	f1-score	support
Non-fraud	1.00	0.99	1.00	56864
Fraud	0.14	0.69	0.23	98
accuracy			0.99	56962
macro avg	0.57	0.84	0.61	56962
weighted avg	1.00	0.99	0.99	56962

Slika 3.13: Rezultat klasifikacije LightGBM-a



Slika 3.14: Precision-Recall kriva LightGBM-a

LGBM, kao i MLP, na inicijalnom datasetu postiže jako loše rezultate u detekciji prevarnih transakcija.

3.2.2 Analiza dobijenih rezultata

Analizom dobijenih rezultata, uzimajući u obzir da je ustanovljeno da će se AUPRC smatrati najpouzdanijom metrikom za ovakve datasetove, jasno je da su XGB i RFC modeli ostvarili najbolje rezultate.

Tabela 3.2: Performanse modela treniranih na početnom datasetu

Model	Preciznost	Odziv	F1-Skor	AUPRC	ROC-AUC
LR	0.72	0.69	0.71	0.5939	0.9024
RFC	0.97	0.71	0.82	0.8263	0.9268
XGB	0.93	0.78	0.84	0.8436	0.9802
NN	0.23	0.53	0.32	0.4834	0.8209
LGB	0.14	0.69	0.23	0.3966	0.8003

Kao što tabela 3.2 pokazuje, XGB model je postigao najbolji rezultat sa AUPRC vrijednošću od 0.8436, dok je RFC model ostvario AUPRC vrijednost od 0.8326. Ove vrijednosti sugerišu da oba modela imaju dobru sposobnost prepoznavanja prevarnih transakcija u neuravnoteženim datasetovima.
Detaljna analiza svakog od modela pokazuje da XGB model ima bolji balans između preciznosti i odziva, što ga čini pogodnijim za detekciju prevara u situacijama gdje je važno minimizirati

broj lažno negativnih rezultata. S druge strane, RFC model također pokazuje visoku efikasnost, ali sa nešto nižim odzivom, što može rezultirati većim brojem propuštenih prevara. Ostali modeli su pokazali znatno lošije rezultate.
S obzirom na ove rezultate, može se zaključiti da su ansambl metode, poput XGB-a i RFC-a, izuzetno efikasne u detekciji prevarnih transakcija na neuravnoteženim datasetovima. Daljnje istraživanje fokusirat će se na poboljšanje performansa ovih, ali i ostalih modela upotrebom sintetičkih podataka.

3.3 Generisanje sintetičkih podataka

SDV (Synthetic Data Vault) omogućava stvaranje vjerodostojnih sintetičkih podataka koji zadržavaju karakteristike originalnog skupa podataka. Cilj korištenja sintetičkih podataka je povećati broj prevarnih (*fraud*) transakcija kako bi modeli mogli bolje prepoznati takve transakcije [5].

Da bi se dobili vjerodostojni sintetički podaci koji se mogu koristiti u daljoj analizi, potrebno je izvršiti nekoliko koraka:

3.3.1 Kreiranje modela za sintetičke podatke

Prvi korak je kreiranje modela za generisanje sintetičkih podataka korištenjem klase *CTGANSynthesizer* iz SDV biblioteke. Model se trenira na originalnom datasetu kako bi naučio njegove karakteristike. *CTGANSynthesizer* je klasa koja koristi model zasnovan na GAN-ovima (Generative Adversarial Networks) za generisanje sintetičkih podataka [6]. Ova metoda je efikasna u učenju kompleksnih distribucija podataka. Za potrebe ove analize korišten je *CTGANSynthesizer* sa 100 epoha, bez zaokruživanja, te sa prikazom informacija o napretku procesa tokom treninga.

3.3.2 Generisanje sintetičkih podataka sa različitim omjerima prevarnih transakcija

Nakon treniranja modela, generisani su različiti sintetički skupovi podataka sa raznovrsnim omjerima prevarnih transakcija. Ovo omogućava testiranje i treniranje modela na različitim scenarijima balansiranja podataka:

- **Dataset sa 50% prevarnih transakcija**: Cilj je napraviti dataset gdje polovinu transakcija čine prevarne transakcije. Koriste se uvjeti za uzorkovanje podataka kako bi se osiguralo da je broj prevarnih i validnih transakcija jednak.
- **Dataset sa 0.17% prevarnih transakcija**: Cilj je simulirati originalni neuravnoteženi dataset ali sa sintetičkim podacima. Koriste se uvjeti za uzorkovanje podataka kako bi se osiguralo da samo 0.17% transakcija čine prevarne transakcije, kao što je slučaj i sa originalnim datasetom. Svrha ovog dataseta je da služi kao krajnji testni set za modele.
- **Dataset sa omjerom 10:1 (validne : prevarne) transakcije**: Cilj je napraviti dataset gdje je omjer validnih i prevarnih transakcija 10:1, kako bi se testirali modeli na različitim omjerima validnih i prevarnih transakcija, te izvukao zaključak o tome koji omjer je najpogodniji.

3.3.3 Sintetički dataset 1

Prvi sintetički dataset je generisan na osnovu polaznog dataseta, sa istim omjerom validnih i prevarnih transakcija. Cilj je da se ovaj dataset koristi kao dataset na kojem se modeli nakon treniranja mogu dodatno testirati. Na sljedećoj slici 3.15, prikazana je evaluacija kvalitete podataka ovog dataseta.

Slika 3.15: Evaluacija kvalitete prvog sintetičkog dataseta

Evaluating Column Shapes: Ova faza evaluacije ispituje distribuciju vrijednosti u svakoj koloni sintetičkog skupa podataka i upoređuje je sa distribucijom u stvarnom skupu podataka. Column Shapes Score: Rezultat ove evaluacije je 94.48%, što znači da je distribucija vrijednosti u sintetičkim kolonama veoma slična distribuciji u stvarnim kolonama u 94.48% slučajeva. Evaluating Column Pair Trends: Ova faza evaluacije ispituje odnose između parova kolona u sintetičkom skupu podataka i upoređuje ih sa odnosima u stvarnom skupu podataka. Column Pair Trends Score: Rezultat ove evaluacije je 93.91%, što znači da su odnosi između parova kolona u sintetičkim podacima veoma slični odnosima u stvarnim podacima u 93.91% slučajeva. Overall Score (Average): Sveukupni rezultat je prosjek rezultata iz obje evaluacije, što iznosi 94.2%. Ovaj rezultat predstavlja ukupnu sličnost između sintetičkog i stvarnog skupa podataka. Visoki rezultati (iznad 90%) za obje evaluacije i sveukupni rezultat sugerišu da su sintetički podaci veoma slični stvarnim podacima. To znači da su sintetički podaci uspješno replicirali distribucije i odnose iz stvarnog skupa podataka, što ih čini pogodnim za daljnju analizu.

3.3.4 Sintetički dataset 2

Drugi dataset koji je generisan predstavljat dataset u kojem je broj validnih i prevarnih transakcija približno jednak. Razlika između originalnog i ovog dataseta prikazana je na sljedećoj slici 3.16

Slika 3.16: Usporedba originalnog i dataseta 2

S druge strane kvaliteta podataka ovog dataseta i nije toliko visoka kao originalnog, što je donekle i očekivano jer se broj transakcija uveliko razlikuje. Ovakva kvaliteta podataka sintetičkog dataseta može uzrokovati probleme, jer postoji značajan nivo odstupanja, posebno u distribuciji pojedinačnih kolona, što može uticati na preciznost modela treniranih na sintetičkim podacima. Dakle, modeli trenirani na ovim podacima mogu imati smanjenu efikasnost u stvarnim scenarijima zbog tih odstupanja.

```
Generating report ...

(1/2) Evaluating Column Shapes: |██████████| 31/31 [00:03<00:00,  7.99it/s]|
Column Shapes Score: 80.47%

(2/2) Evaluating Column Pair Trends: |██████████| 465/465 [01:10<00:00,  6.56it/s]|
Column Pair Trends Score: 86.65%

Overall Score (Average): 83.56%
```

Slika 3.17: Evaluacija kvalitete drugog sintetičkog dataseta

3.3.5 Sintetički dataset 3

Ovaj dataset je napravljen tako da bi se minimizirala greška u vidu razlike originalnih i sintetičkih podataka, ali i da bi se povećao broj prevarnih transakcija u svrhu boljeg treniranja modela.

Slika 3.18: Usporedba originalnog i dataseta 3

```
Generating report ...

(1/2) Evaluating Column Shapes: |██████████| 31/31 [00:08<00:00,  3.53it/s]|
Column Shapes Score: 93.89%

(2/2) Evaluating Column Pair Trends: |██████████| 465/465 [01:26<00:00,  5.39it/s]|
Column Pair Trends Score: 90.26%

Overall Score (Average): 92.08%
```

Slika 3.19: Evaluacija kvalitete trećeg sintetičkog dataseta

Tabela 3.3: Usporedba sintetičkih datasetova

Sintetički dataset	Column Shapes (%)	Column Pair Trends(%)	Overall Score (%)
Dataset 1	94.48	93.91	94.2
Dataset 2	80.47	86.65	83.56
Dataset 3	93.89	90.26	92.08

3.4 Miješanje i spremanje sintetičkih podataka

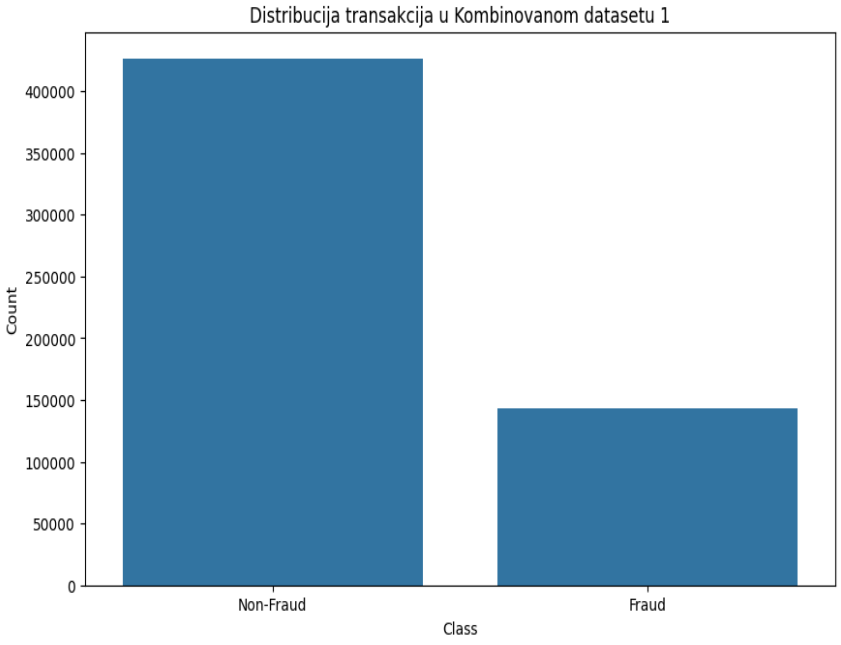
Miješanje sintetičkih i stvarnih podataka za treniranje modela omogućava bolju pripremljenost modela za prepoznavanje prevara u stvarnom svijetu. Prednosti ovog pristupa uključuju:

- **Bolja generalizacija**: Modeli trenirani na kombinovanom skupu podataka imaju tendenciju da bolje generaliziraju na nove, neviđene podatke, jer su trenirani na širem spektru primjera [7].
- **Veća preciznost i odziv**: Dodavanjem sintetičkih prevarnih transakcija, model može postići veću preciznost i osjetljivost, što je kritično za detekciju prevara [48].
- **Smanjenje overfittinga**: Više podataka može smanjiti rizik od overfittinga, jer model ima više primjera za učenje [11].
- **Dodavanje raznovrsnosti podacima**: Povećava se ukupan broj instanci i dodaje raznovrsnost podacima, što sve zajedno doprinosi robusnijem i preciznijem modelu za detekciju prevarnih transakcija [49].

Dakle, korištenjem ranije generisanih sintetičkih datasetova kreirat će se tri različita kombinovana (merged) dataseta, prvi će biti kreiran povezivanjem početnog dataseta i sintetičkog dataseta 1, zatim drugi dataset će biti kreiran povezivanjem početnog dataseta i dataseta 2, te treći povezivanjem početnog dataseta i dataseta 3.

3.4.1 Kombinovani dataset 1

Kombinovani dataset 1 je kreiran kombinovanjem početnog dataseta sa sintetičkim datasetom 2, gdje je broj prevarnih transakcija povećan kako bi bio približno jednak broju validnih transakcija. Prikaz raspodjele klasa ovog dataseta dat je na slici 3.20.

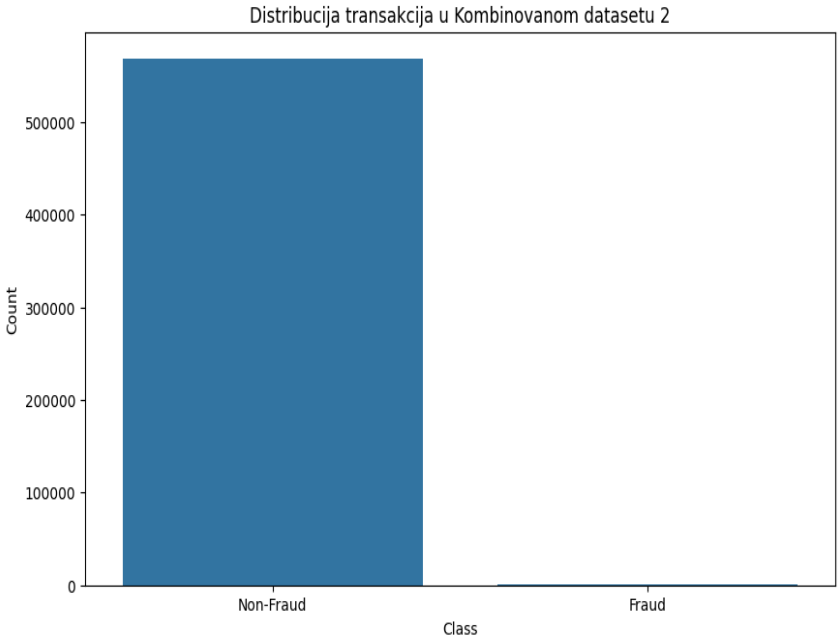


Slika 3.20: Distribucija transakcija u Kombinovanom datasetu 1

3.4.2 Kombinovani Dataset 2

Kombinovani dataset 2 je kreiran kombinovanjem početnog dataseta sa sintetičkim datasetom 1, gdje je broj prevarnih transakcija zadržan na istom nivou kao i u početnom datasetu. Zbog

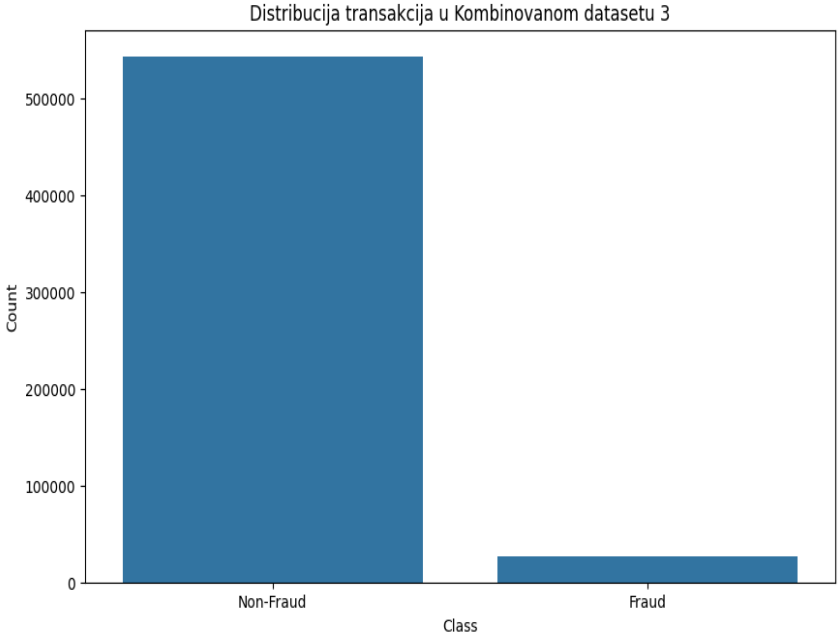
toga ovaj dataset ima sličan omjer prevarnih i validnih transakcija kao i početni dataset. Prikaz raspodjele klasa ovog dataseta dat je na slici 3.21.



Slika 3.21: Distribucija transakcija u Kombinovanom datasetu 2

3.4.3 Kombinovani Dataset 3

Kombinovani dataset 3 je kreiran kombinovanjem početnog dataseta sa sintetičkim datasetom 3, gdje je broj prevarnih transakcija povećan, ali ne u tolikoj mjeri kao kod kombinovanog dataseta 1. Prikaz raspodjele klasa ovog dataseta dat je na slici 3.22.



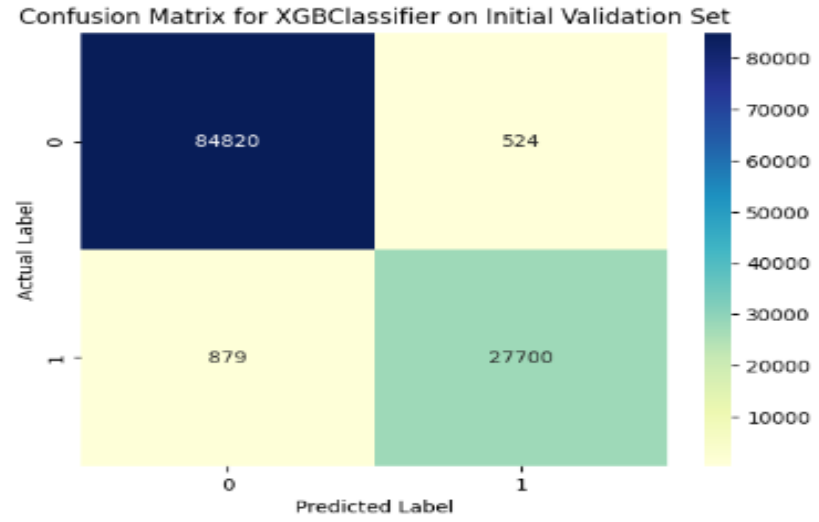
Slika 3.22: Distribucija transakcija u Kombinovanom datasetu 3

3.5 Treniranje modela na kombinovanim datasetovima

Nakon kreiranja kombinovanih datasetova, potrebno je trenirati i testirati modele koji su ranije prikazani. Svaki model je treniran koristeći 80% podataka za treniranje i 20% podataka za testiranje. Zbog poboljšane preglednosti, u nastavku će biti prikazan najbolji model za svaki od novokreiranih (kombinovanih) datasetova. Na kraju podpoglavlja bit će priložena tabela sa svim modelima i svim datasetovima.

3.5.1 Treniranje i testiranje modela za kombinovani dataset 1

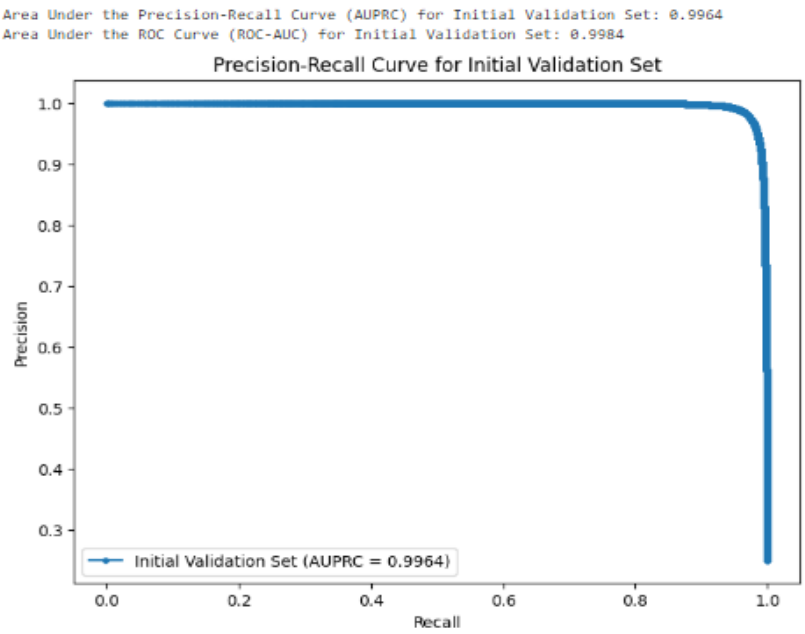
Model sa najboljim rezultatima za ovaj dataset je **XGBoost (XGB)**. XGBoost je pokazao superiorne performanse u odnosu na ostale modele zbog svoje sposobnosti da se nosi s kompleksnim obrascima u podacima, čak i za velike datasetove poput ovog, te zbog toga pruža visoku tačnost.



Classification Report for Initial Validation Set

	precision	recall	f1-score	support
Non-fraud	0.99	0.99	0.99	85344
Fraud	0.98	0.97	0.98	28579
accuracy			0.99	113923
macro avg	0.99	0.98	0.98	113923
weighted avg	0.99	0.99	0.99	113923

Slika 3.23: Rezultat klasifikacije XGB-a na Kombinovani dataset 1

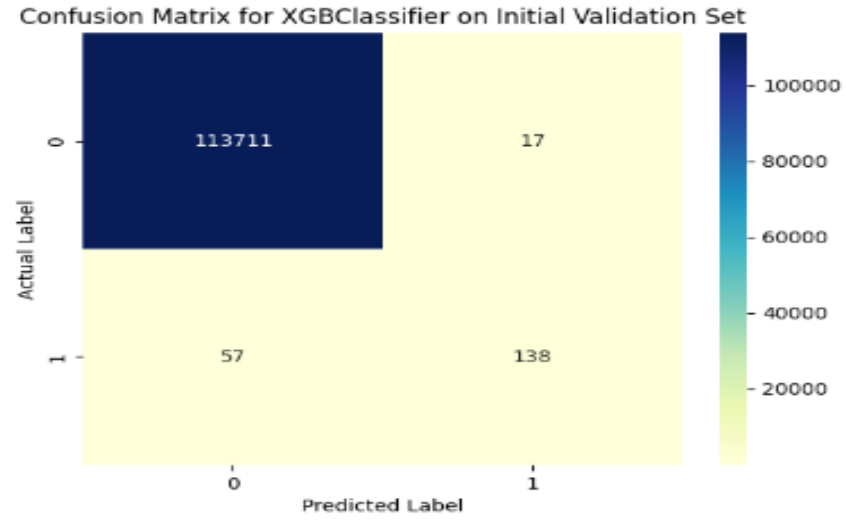


Slika 3.24: Precision-Recall kriva XGB-a za Kombinovani dataset 1

Moguće je primijetiti da je treniranje modela nad kombinovanim datasetom dalo daleko veću preciznost nad testnim podacima u odnosu na treniranje nad početnim datasetom, što ide u prilog korištenju sintetičkih podataka za detekciju prevarnih transakcija. XGBoost je uspio uhvatiti raznolike obrasce i osigurati visoku preciznost i recall, čime se smanjuje broj lažno pozitivnih i lažno negativnih predikcija.

3.5.2 Treniranje i testiranje modela za kombinovani dataset 2

Model sa najboljim rezultatima za ovaj dataset je ponovo **XGBoost (XGB)**. Međutim, performanse ovog modela su dosta slabije u odnosu na kombinovani dataset 1, što se može pripisati manjem broju prevarnih transakcija u datasetu.

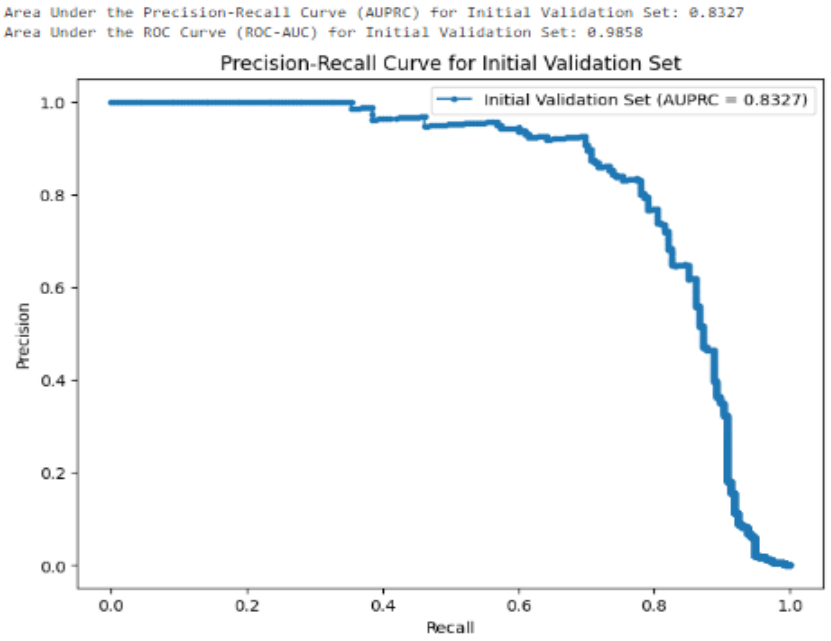



```
Classification Report for Initial Validation Set
              precision    recall  f1-score   support

   Non-fraud       1.00      1.00      1.00    113728
       Fraud       0.89      0.71      0.79       195

    accuracy                           1.00    113923
   macro avg       0.94      0.85      0.89    113923
weighted avg       1.00      1.00      1.00    113923
```

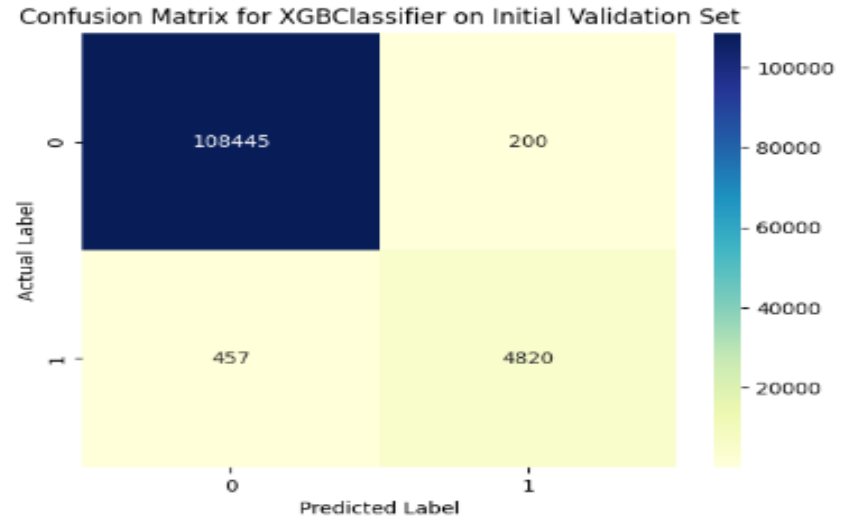
Slika 3.25: Rezultat klasifikacije XGB-a na Kombinovani dataset 2



Slika 3.26: Precision-Recall kriva XGB-a za Kombinovani dataset 2

3.5.3 Treniranje i testiranje modela za kombinovani dataset 3

Model sa najboljim rezultatima za ovaj dataset je ponovo **XGBoost (XGB)**. Ovaj model je pokazao bolje performanse u odnosu na treniranje nad kombinovanim datasetom 2, ali ne i u odnosu na treniranje modela nad kombinovanim datasetom 1.

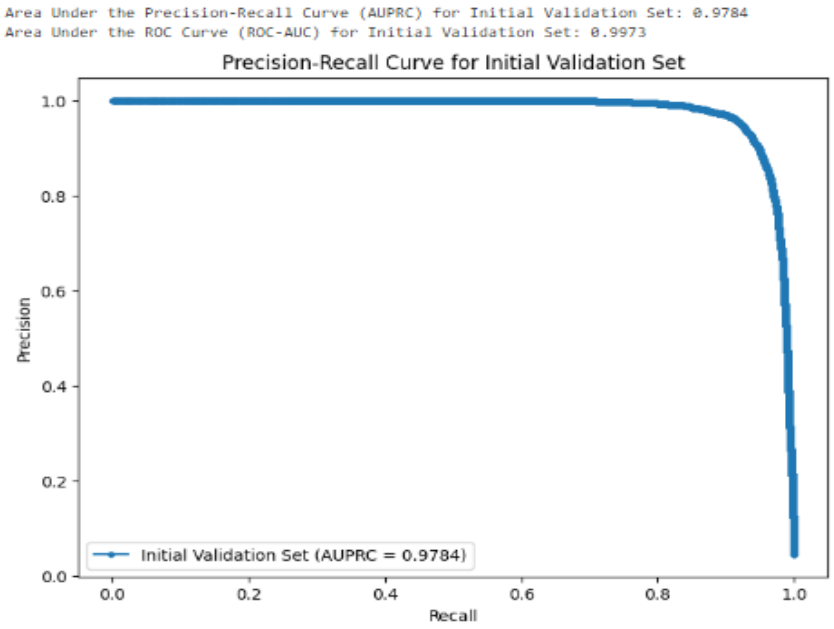



```
Classification Report for Initial Validation Set
              precision    recall  f1-score   support

   Non-fraud       1.00      1.00      1.00    108645
       Fraud       0.96      0.91      0.94      5277

    accuracy                           0.99    113922
   macro avg       0.98      0.96      0.97    113922
weighted avg       0.99      0.99      0.99    113922
```

Slika 3.27: Rezultat klasifikacije XGB-a na kombinovani dataset 3



Slika 3.28: Precision-Recall kriva XGB-a za Kombinovani dataset 3

Na osnovu prikazanih rezultata, može se zaključiti da je veći broj prevarnih transakcija u odnosu na početni dataset uvijek poželjan za svrhe treniranja modela, kako bi model postao bolji u prepoznavanju prevarnih transakcija [48]. U nastavku je prikazana tabela 3.4 performansi svih modela nad svakim od kombinovanih datasetova.

Tabela 3.4: Tabela performansi modela na kombinovanim datasetovima

Model	Dataset	Preciznost	Odziv	F1-Skor	AUPRC	ROC-AUC
LR	Kombinovani dataset 1	0.95	0.92	0.93	0.9734	0.985
LR	Kombinovani dataset 2	0.61	0.58	0.60	0.5295	0.896
LR	Kombinovani dataset 3	0.81	0.83	0.82	0.8754	0.968
RFC	Kombinovani dataset 1	0.98	0.96	0.97	0.9940	0.996
RFC	Kombinovani dataset 2	0.92	0.67	0.77	0.8071	0.9495
RFC	Kombinovani dataset 3	0.97	0.89	0.93	0.9692	0.9922
XGB	Kombinovani dataset 1	0.99	0.97	0.98	0.9964	0.9984
XGB	Kombinovani dataset 2	0.89	0.71	0.79	0.8327	0.9858
XGB	Kombinovani dataset 3	0.96	0.91	0.94	0.9784	0.9973
NN	Kombinovani dataset 1	0.97	0.92	0.94	0.9853	0.9919
NN	Kombinovani dataset 2	0.31	0.50	0.39	0.3712	0.8698
NN	Kombinovani dataset 3	0.93	0.87	0.90	0.9351	0.9896
LGB	Kombinovani dataset 1	0.98	0.96	0.97	0.9957	0.9981
LGB	Kombinovani dataset 2	0.27	0.56	0.37	0.3926	0.7451
LGB	Kombinovani dataset 3	0.96	0.91	0.94	0.9768	0.9970

3.6 Analiza dobijenih rezultata nad kombinovanim datasetovima

Tabela 3.4 prikazuje performanse različitih modela na testnom skupu različitih kombinovanih datasetova. Performanse su evaluirane pomoću pet ranije navedenih metrika: preciznost, odziv, F1-skor, AUPRC (Average Precision-Recall Curve) i ROC-AUC (Receiver Operating Characteristic - Area Under Curve).

3.6.1 Logistička Regresija (LR)

- **Kombinovani dataset 1**: Model postiže visoke vrijednosti svih metrika, sa preciznošću od 0.95, odzivom od 0.92, i F1-skorom od 0.93. AUPRC i ROC-AUC su također visoki, što ukazuje na dobru sposobnost modela da razlikuje između prevarnih i neprevarnih transakcija.
- **Kombinovani dataset 2**: Performanse modela značajno opadaju na ovom datasetu, sa preciznošću od 0.61 i F1-skorom od 0.60. Zaključuje se da model ima poteškoća sa balansiranjem neuravnoteženih podataka bez dodatne pomoći sintetičkih prevarnih transakcija.

- **Kombinovani dataset 3**: Model pokazuje bolje rezultate u odnosu na kombinovani dataset 2, sa preciznošću od 0.81 i F1-skorom od 0.82, ali i dalje nije tako dobar kao na Kombinovani dataset 1.

3.6.2 Random Forest Classifier (RFC)

- **Kombinovani dataset 1**: RFC pokazuje izvanredne performanse sa preciznošću od 0.98, F1-skorom od 0.97, AUPRC od 0.9940 i ROC-AUC od 0.996. Ovi rezultati sugerišu da je RFC veoma učinkovit u prepoznavanju prevarnih transakcija kada su podaci uravnoteženi.
- **Kombinovani dataset 2**: Performanse opadaju na kombinovanom datasetu 2, sa preciznošću od 0.92 i F1-skorom od 0.77. Iako su rezultati bolji nego kod LR, RFC i dalje ima probleme sa neuravnoteženosti podataka.
- **Kombinovani dataset 3**: RFC pokazuje bolje rezultate u odnosu na kombinovani dataset 2, sa preciznošću od 0.97 i F1-skorom od 0.93.

3.6.3 XGBoost (XGB)

- **Kombinovani dataset 1**: XGBoost pokazuje najbolje performanse među svim modelima na Kombinovani dataset 1, sa preciznošću od 0.99, F1-skorom od 0.98, AUPRC od 0.9964 i ROC-AUC od 0.9984. Ovo pokazuje da je XGBoost veoma učinkovit u prepoznavanju prevara kada su podaci uravnoteženi.
- **Kombinovani dataset 2**: Performanse opadaju, sa preciznošću od 0.89 i F1-skorom od 0.79. Iako su rezultati i dalje dobri, vidi se značajno smanjenje efikasnosti modela u odnosu na kombinovani dataset 1.
- **Kombinovani dataset 3**: XGBoost postiže bolje rezultate u odnosu na kombinovani dataset 2, sa preciznošću od 0.96 i F1-skorom od 0.94, ali nije tako dobar kao na kombinovanom datasetu 1.

3.6.4 Neuralna Mreža (NN)

- **Kombinovani dataset 1**: Neuralna mreža pokazuje visoke performanse sa preciznošću od 0.97 i F1-skorom od 0.94, AUPRC od 0.9853 i ROC-AUC od 0.9919.
- **Kombinovani dataset 2**: Performanse opadaju drastično na kombinovanom datasetu 2, sa preciznošću od 0.31 i F1-skorom od 0.39.
- **Kombinovani dataset 3**: Neuralna mreža pokazuje poboljšanje u odnosu na Kombinovani dataset 2, sa preciznošću od 0.93 i F1-skorom od 0.90.

3.6.5 LightGBM (LGB)

- **Kombinovani dataset 1**: LightGBM pokazuje visoke performanse, sa preciznošću od 0.98 i F1-skorom od 0.97, AUPRC od 0.9957 i ROC-AUC od 0.9981.
- **Kombinovani dataset 2**: Performanse opadaju drastično kao i kod neuralnih mreža, sa preciznošću od 0.27 i F1-skorom od 0.37, što pokazuje da LightGBM također ima poteškoća sa neuravnoteženim podacima.

- **Kombinovani dataset 3**: LightGBM pokazuje poboljšanje u odnosu na Kombinovani dataset 2, sa preciznošću od 0.96 i F1-skorom od 0.94, AUPRC od 0.9768 i ROC-AUC od 0.9970, ali i dalje nije na nivou kao na kombinovani dataset 1.

Na osnovu rezultata prikazanih u tabeli 3.4, može se zaključiti da svi modeli pokazuju bolje performanse kada se treniraju na kombinovanim datasetovima sa većim brojem prevarnih transakcija. XGBoost se pokazao kao najbolji model na svim datasetovima, sa najvišim vrijednostima svih metrika, posebno na kombinovanom datasetu 1. Random Forest Classifier također pokazuje vrlo dobre performanse, posebno na uravnoteženijim datasetovima. Neuronske mreže i LightGBM pokazuju poboljšanja kada se povećava broj prevarnih transakcija, ali imaju poteškoća sa izrazito neuravnoteženim podacima.

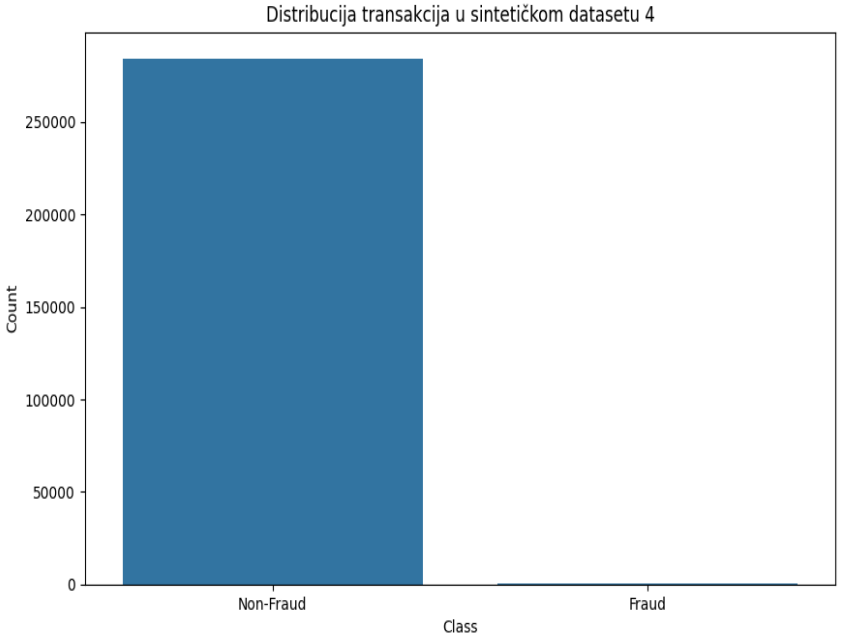
Korištenje sintetičkih podataka se pokazalo kao efikasan pristup za poboljšanje performansi modela, posebno u situacijama kada postoji neuravnoteženost između klasa. Veći broj prevarnih transakcija u kombinovanim datasetovima omogućava modelima da bolje nauče obrasce prevara, što rezultira boljim performansama u detekciji prevarnih transakcija [12, 50, 51].

3.7 Testiranje modela na neviđenom datasetu

Testiranje modela na neviđenom datasetu predstavlja ključni korak u procesu evaluacije performansi modela mašinskog učenja. U prethodnim fazama rada, modeli su trenirani i validirani na datasetovima koji su bili dostupni tokom procesa razvoja. Međutim, da bi se procijenila prava sposobnost modela da generalizira i da bude primjenjiv na stvarne podatke, neophodno je testiranje na potpuno novom skupu podataka koji model nikada nije vidio [7, 11, 35].

U ovom poglavlju fokusirat ćemo se na testiranje modela korištenjem sintetičkog dataseta 4. Ovaj dataset je generisan sa sličnim karakteristikama kao i originalni dataset, ali je potpuno nepoznat treniranim modelima. Cilj testiranja na ovako generisanom datasetu je da se procijeni koliko dobro modeli mogu prepoznati obrasce i detektovati prevarne transakcije kada su suočeni sa novim, nepoznatim podacima. Ovo testiranje je od izuzetne važnosti jer pruža uvid u stvarnu efikasnost modela u realnim uslovima.

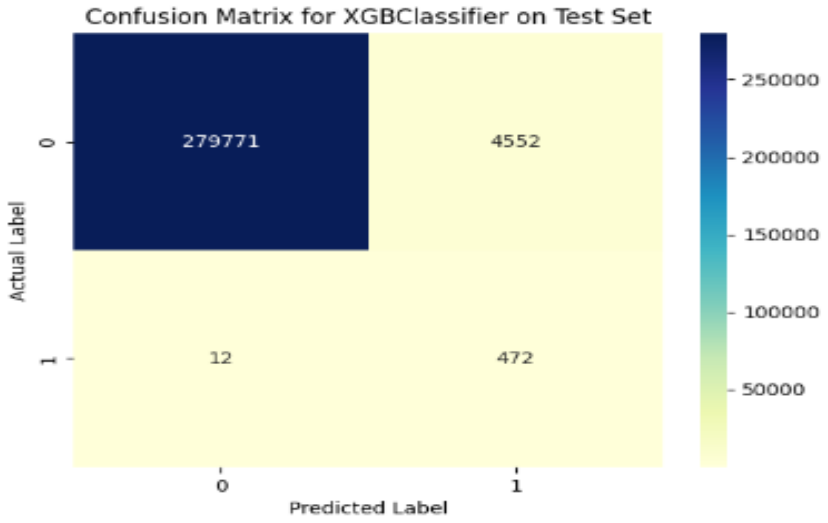
Distribucija podataka u sintetičkom datasetu je identična kao u početnom, te je prikazana na sljedećoj slici.



Slika 3.29: Distribucija transakcija u sintetičkom datasetu 4

Očekivano, pošto je broj validnih transakcija daleko veći, broj prevarnih transakcija je jako teško vidljiv.

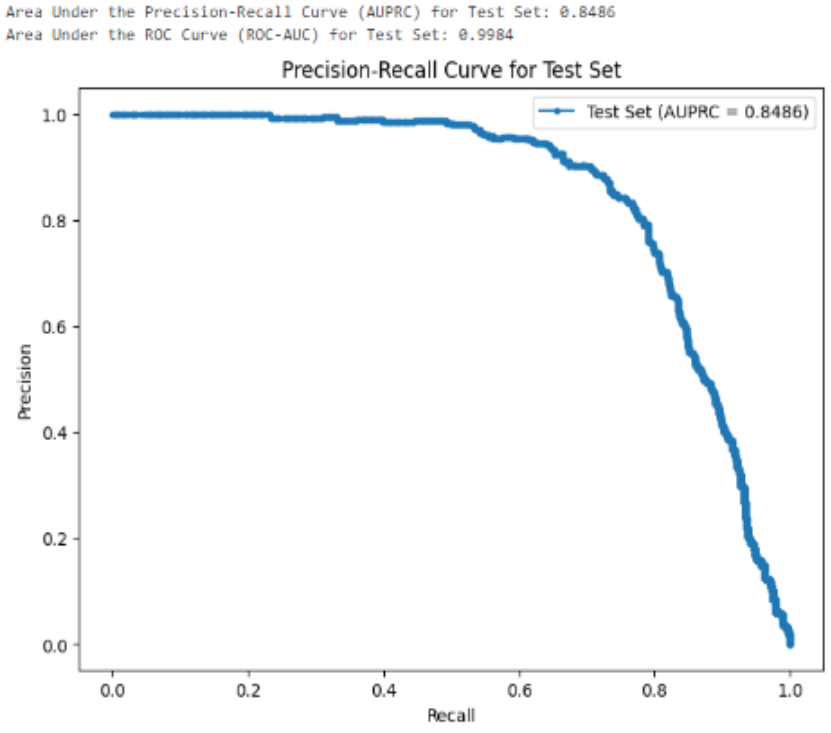
U prethodnom dijelu svaki od modela je treniran i testiran na različitim datasetovima, te je od svih modela XGB pokazao najbolje rezultate. Upravo zbog toga, potrebno je na kraju ovaj model testirati na novom, sintetičkom datasetu koji je gotovo identičan polaznom.



Classification Report for Test Set

	precision	recall	f1-score	support
Non-fraud	1.00	0.98	0.99	284323
Fraud	0.09	0.98	0.17	484
accuracy			0.98	284807
macro avg	0.55	0.98	0.58	284807
weighted avg	1.00	0.98	0.99	284807

Slika 3.30: Rezultat klasifikacije XGB-a na sintetički dataset 4



Slika 3.31: Precision-Recall kriva XGB-a za sintetički dataset 4

Analizom dobijenih rezultata, uviđa se da je performans modela nad novim datasetom prilično dobar, međutim dosta niži u odnosu na performans modela nad validacijskim setom kombinovanog dataseta.

Potencijalni razlozi za opadanje kvalitete modela su [11]:

- Veličina skupa podataka: Ako je sintetički dataset 4 znatno veći ili manji od validacijskog skupa, to može utjecati na performanse modela zbog različite reprezentativnosti uzoraka. U ovom slučaju je 5 puta veći.
- Kvaliteta sintetičkih podataka može varirati. Iako je uvršen trud za generisanje sintetičkih podataka koji su što sličniji stvarnim podacima, sintetički podaci možda ne reflektuju sve nijanse i obrasce originalnog skupa podataka.
- Kombinovani dataset i sintetički dataset 4 mogu imati različite distribucije podataka. Iako je dataset 4 sintetički generisan da bi imao slične karakteristike kao originalni skup podataka, moguće je da postoje suptilne razlike koje utječu na performanse modela [35].

3.7.1 Analiza rezultata modela na testnom skupu (3:1 omjer)

Ukoliko bi se pokušao eliminirati neki od navedenih problema, kako bismo imali još bolji uvid u performanse modela, moguće je testirati model na uravnoteženiji skup podataka ekstraktovan iz sintetičkog dataseta 4. Dakle moguće je sintetički dataset podijeliti tako da ima isti omjer validnih i prevarnih transakcija kao i testni set kombinovanog dataseta 1. Pošto je omjer u testnom skupu kombinovanog dataseta 1 približno 3:1, u korist validnih transakcija, isti omjer je generisan i za sintetički dataset 4, pa je dobijen sljedeći rezultat 3.5.

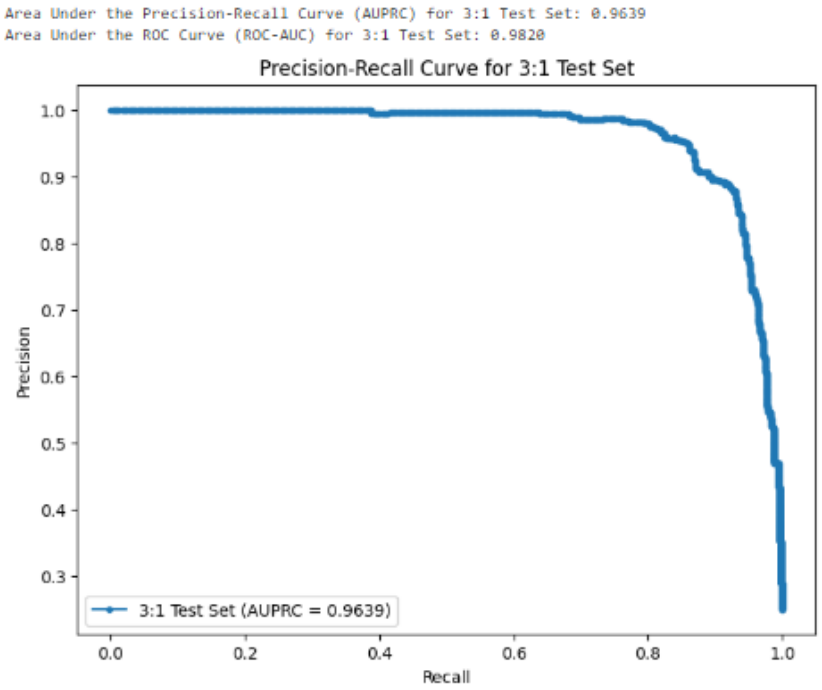
Klasa	**Preciznost**	**Odziv**	**F1-skor**	**Podrška**
Validne transakcije	1.00	0.95	0.97	1452
Prevarne transakcije	0.86	0.99	0.92	484
Ukupno	**0.96**	**0.96**	**0.96**	**1936**

Tabela 3.5: Izvještaj o klasifikaciji za testni skup (3:1 omjer)

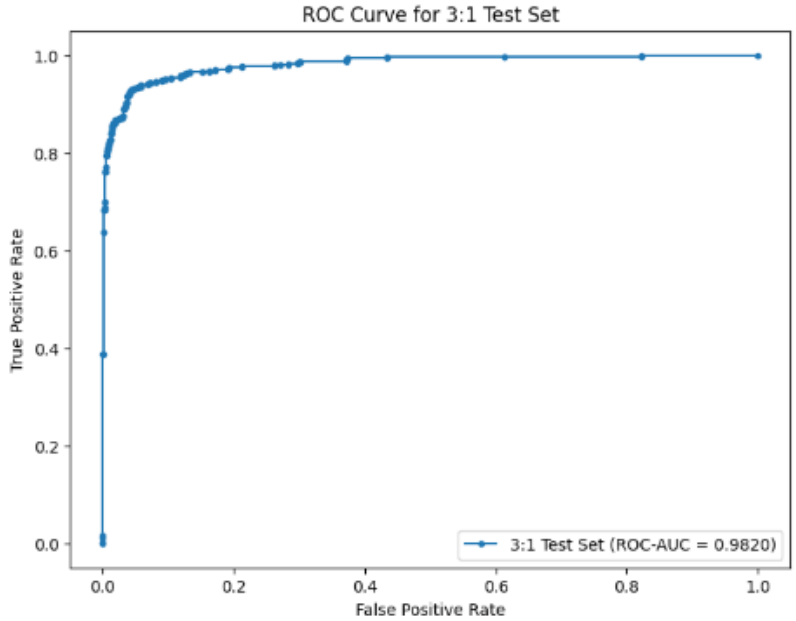
Rezultati pokazuju da model postiže visoku preciznost, odziv i F1-skor za obje klase, s ukupnom tačnošću od 96%. Detaljnija analiza metrika za pojedine klase je sljedeća :

- **Validne transakcije (klasa 0)**:
  - **Preciznost:** 1.00 - To znači da je model vrlo uspješan u tačnom identificiranju validnih transakcija.
  - **Odziv:** 0.95 - To znači da model ispravno identificira 95% svih stvarnih validnih transakcija.
  - **F1-skor:** 0.97 - Kombinacija preciznosti i odziva ukazuje na izbalansiranu učinkovitost modela za ovu klasu.
- **Prevarne transakcije (klasa 1)**:
  - **Preciznost:** 0.86 - To znači da od svih transakcija koje model identificira kao prevarne, 86% su stvarno prevarne.
  - **Odziv:** 0.99 - To znači da model ispravno identificira 99% svih stvarnih prevarnih transakcija.
  - **F1-skor:** 0.92 - Kombinacija preciznosti i odziva ukazuje na visoku učinkovitost modela za ovu klasu.

Ovi rezultati pokazuju da model ima iznimno visoku sposobnost prepoznavanja prevarnih transakcija (visok odziv), što je ključno za primjenu u stvarnim financijskim sustavima gdje je identificiranje svih prevara od vitalne važnosti. Istovremeno, model održava visoku preciznost za validne transakcije, smanjujući broj lažno pozitivnih predikcija.



Slika 3.32: Precision-Recall kriva za testni skup (3:1 omjer)

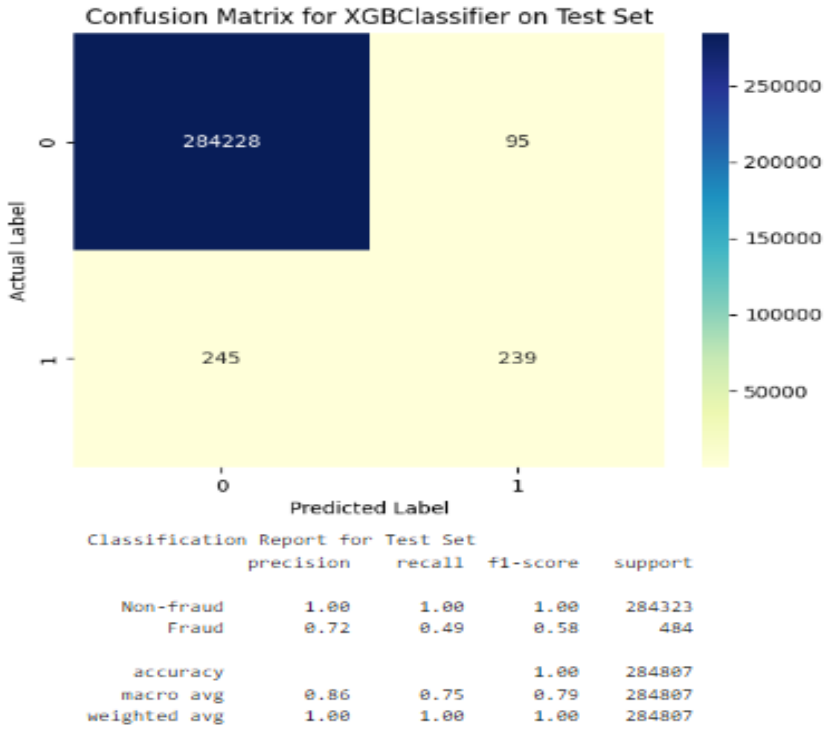


Slika 3.33: ROC kriva za testni skup (3:1 omjer)

Iz rezultata prikazanih na slici 3.32 (Precision-Recall kriva) i slici 3.33 (ROC kriva), vidljivo je da model ima izvanredne performanse s visokim vrijednostima AUPRC (Površina ispod

Precision-Recall krivulje) i ROC-AUC (Površina ispod ROC krivulje), što dodatno potvrđuje njegovu učinkovitost u detekciji prevarnih transakcija.

Ukoliko bi se isto ponovilo za XGB model treniran na početnom datasetu, dobili bi se sljedeći rezultati.

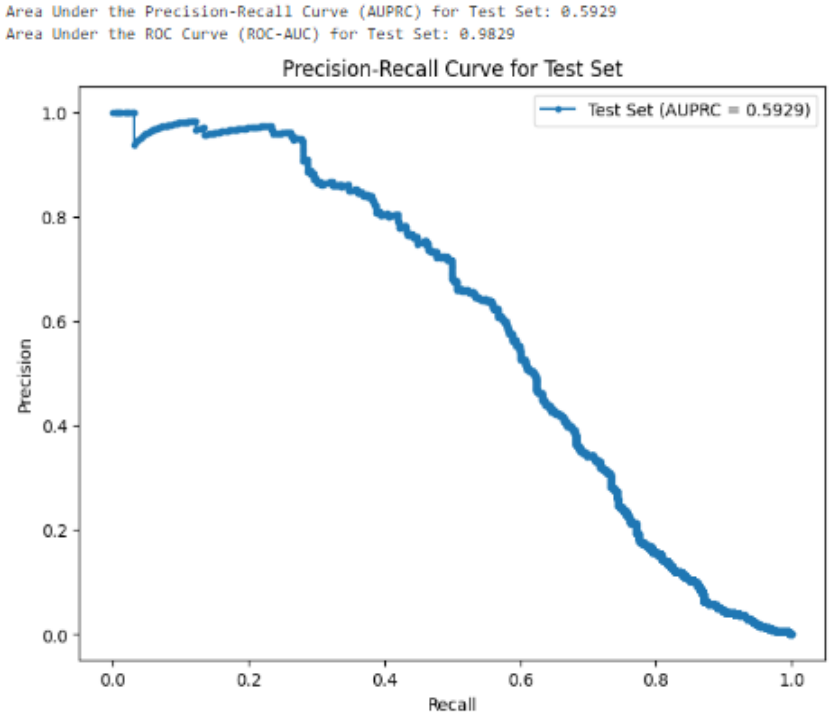



```
Classification Report for Test Set
              precision    recall  f1-score   support

   Non-fraud       1.00      1.00      1.00    284323
       Fraud       0.72      0.49      0.58       484

    accuracy                           1.00    284807
   macro avg       0.86      0.75      0.79    284807
weighted avg       1.00      1.00      1.00    284807
```

Slika 3.34: XGB model treniran na početnom datasetu



Slika 3.35: Precision-Recall kriva XGB modela treniranog na početnom datasetu

Kao što se može vidjeti iz Tabele 3.6, XGB model treniran na početnom datasetu postiže sljedeće rezultate na testnom skupu sa omjerom 3:1:

- **Preciznost za validne transakcije**: 0.86
- **Odziv za validne transakcije**: 1.00
- **F1-skor za validne transakcije**: 0.92
- **Preciznost za prevarne transakcije**: 1.00
- **Odziv za prevarne transakcije**: 0.49
- **F1-skor za prevarne transakcije**: 0.66
- **Ukupna tačnost**: 0.87
- **Makro prosjek F1-skora**: 0.79

AUPRC i ROC-AUC vrijednosti prikazane su u Tabeli 3.7, gdje AUPRC iznosi 0.9639, a ROC-AUC 0.9820.

Klasa	**Preciznost**	**Odziv**	**F1-skor**	**Podrška**
Validne transakcije	0.86	1.00	0.92	1452
Prevarne transakcije	1.00	0.49	0.66	484
Ukupno	**0.87**	**0.87**	**0.86**	**1936**

Tabela 3.6: Izvještaj o klasifikaciji za testni skup (3:1 omjer)

Metrika	**Vrijednost**
AUPRC	0.9639
ROC-AUC	0.9820

Tabela 3.7: AUPRC i ROC-AUC za testni skup (3:1 omjer)

Iz ove uporedbe jasno je da XGB model treniran na kombinovanom datasetu 1 postiže bolje rezultate na testnom skupu u odnosu na model treniran na početnom datasetu. Preciznost, odziv i F1-skor su viši za obje klase, što ukazuje na superiornu sposobnost modela da prepozna i validne i prevarne transakcije. AUPRC i ROC-AUC vrijednosti također su više za model treniran na kombinovanom datasetu 1, što potvrđuje njegovu bolju ukupnu performansu. Ovi rezultati sugerišu da proširenje i balansiranje datasetova može značajno poboljšati performanse modela u detekciji prevara.

Zaključak

U ovom radu je predstavljen značaj sintetičkih podataka, te metode i alati koji se koriste za generisanje istih. Značaj sintetičkih podataka je praktično prikazan na osnovu problema detekcije prevarnih transakcija, u okviru kojeg su se koristili različiti modeli mašinskog učenja kako bi se poboljšala njihova detekcija.

Ostvareni ciljevi završnog rada

Istraživanje je obuhvatilo generisanje sintetičkih podataka korištenjem različitih alata i tehnika, te njihovu primjenu u detekciji prevarnih transakcija. Rezultati su prikazani kroz nekoliko faza, uključujući treniranje modela na originalnim i sintetičkim datasetovima, te evaluaciju njihovih performansi. Korištenjem alata kao što su SDV, Faker, i Gretel Synthetics, generisani su datasetovi sa različitim omjerima prevarnih i neprevarnih transakcija. Najbolji rezultati postignuti su korištenjem CTGAN modela unutar SDV-a, koji je generisao podatke visoke kvalitete zadržavajući ključne karakteristike originalnog skupa podataka. Modeli mašinskog učenja, uključujući Logistic Regression, Random Forest, XGBoost, neuronske mreže i LightGBM, trenirani su na kombinovanim datasetovima. Evaluacija je pokazala da su ansambl metode poput XGBoost-a i Random Forest-a postigle najbolje rezultate, posebno u smislu AUPRC metrike.

Dobijeni rezultati upoređeni su sa nalazima iz literature navedenim u uvodnom poglavlju. Radovi kao što su [15] i [12], koji predstavljaju osnovne principe GAN-ova i detekciju prevara, pružili su osnovu za metodologije korištene u ovom radu. Rezultati dobijeni korištenjem CTGAN modela u generisanju sintetičkih podataka potvrđuju nalaze [15], koji ističu efikasnost GAN-ova u generisanju realističnih podataka. Također, evaluacija modela mašinskog učenja na neuravnoteženim datasetovima potvrdila je nalaze [12], koji naglašavaju važnost korištenja balansiranih datasetova za poboljšanje performansi modela u detekciji prevara.

U određenim aspektima, dobijeni rezultati su se razlikovali od onih prikazanih u literaturi. Na primjer, performanse neuronskih mreža nisu bile na očekivanom nivou, što se može pripisati specifičnostima korištenog dataset-a i potrebom za dodatnim prilagođavanjem hiperparametara. Ove razlike naglašavaju potrebu za daljnjim istraživanjem i prilagođavanjem modela specifičnim karakteristikama podataka.

Dobijeni rezultati ukazuju na značaj sintetičkih podataka u treniranju modela mašinskog učenja, posebno u kontekstu detekcije prevarnih transakcija. Sintetički podaci omogućavaju prevazilaženje ograničenja povezanih s dostupnošću i kvalitetom stvarnih podataka, te pružaju efikasno rješenje za treniranje modela na balansiranim datasetovima.

Prilozi

Prilog A

Početni dataset

Uključuje početni dataset ‘creditcard.csv‘ koji je korišten za treniranje i evaluaciju modela mašinskog učenja. Dataset je moguće preuzeti sa sljedeće adrese: [Credit Card Fraud Detection Dataset](https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud/data).

Prilog B

Programski kod

B.1 Učitavanje i vizualizacija podataka

```
%pip install sdv
import pandas as pd
import numpy as np
import sdv

from google.colab import drive
drive.mount('/content/drive')

df = pd.read_csv("/content/drive/MyDrive/creditcard.csv")
```

Program B.1: Učitavanje biblioteka i početnog dataseta

```
import matplotlib.pyplot as plt
import seaborn as sns

print(df.head())

plt.figure(figsize=(8, 6))
sns.countplot(x='Class', data=df)
plt.title('Raspodjela klasa u pocetnom datasetu')
plt.xlabel('Klasa')
plt.ylabel('Broj transakcija')
plt.show()

plt.figure(figsize=(14, 6))
sns.histplot(df[df['Class'] == 0]['Amount'], bins=50, color='blue',
    kde=True, label='Nije prevara')
sns.histplot(df[df['Class'] == 1]['Amount'], bins=50, color='red',
    kde=True, label='Prevara')
plt.legend()
plt.title('Distribucija iznosa transakcija')
plt.xlabel('Iznos')
plt.ylabel('Frekvencija')
plt.show()
```

Program B.2: Vizualizacija podataka

B.2 Generisanje sintetičkog modela i podataka

```
from sdv.metadata import SingleTableMetadata
metadata = SingleTableMetadata()
metadata.detect_from_dataframe(df)

from sdv.single_table import CTGANSynthesizer

synthesizer = CTGANSynthesizer(
    metadata,
    enforce_rounding=False,
    epochs=100,
    verbose=True
)

synthesizer.fit(df)
```

Program B.3: Kreiranje sintetičkog modela

```
from sdv.sampling import Condition
total_rows = len(df)
num_fraud = df['Class'].sum()
num_non_fraud = total_rows - num_fraud
target_fraud = total_rows // 2
target_non_fraud = total_rows - target_fraud


fraud_condition = Condition(num_rows=target_fraud, column_values={'
    Class': 1})
non_fraud_condition = Condition(num_rows=target_non_fraud, column_
    values={'Class': 0})

synthetic_fraud_data = synthesizer.sample_from_conditions([fraud_
    condition])
synthetic_non_fraud_data = synthesizer.sample_from_conditions([non_
    fraud_condition])

synthetic_data = pd.concat([synthetic_fraud_data, synthetic_non_
    fraud_data]).reset_index(drop=True)
synthetic_data = synthetic_data.sample(frac=1).reset_index(drop=True
    )
synthetic_data.to_csv("/content/drive/MyDrive/synthetic_datasetv2",
    index=False)
```

Program B.4: Kreiranje sintetičkog dataseta sa balansovanim podacima

```
total_rows = len(df)
num_fraud = df['Class'].sum()
num_non_fraud = total_rows - num_fraud
target_fraud = round(total_rows * 0.0017)
target_non_fraud = total_rows - target_fraud

fraud_condition = Condition(num_rows=target_fraud, column_values={'
    Class': 1})
non_fraud_condition = Condition(num_rows=target_non_fraud, column_
    values={'Class': 0})

```

```
synthetic_fraud_data = synthesizer.sample_from_conditions([fraud_
    condition])
synthetic_non_fraud_data = synthesizer.sample_from_conditions([non_
    fraud_condition])

synthetic_data = pd.concat([synthetic_fraud_data,synthetic_non_fraud
    _data]).reset_index(drop=True)
synthetic_data = synthetic_data.sample(frac=1).reset_index(drop=True
    )
synthetic_data.to_csv("/content/drive/MyDrive/synthetic_datasetv4",
    index=False)
```

Program B.5: Kreiranje sintetičkog dataseta sličnog početnom

```
total_transactions = 284807
num_fraud_samples = total_transactions // 11
num_non_fraud_samples = num_fraud_samples * 10

fraud_condition = Condition(num_rows=num_fraud_samples, column_
    values={'Class': 1})
non_fraud_condition = Condition(num_rows=num_non_fraud_samples,
    column_values={'Class': 0})

synthetic_fraud_data = synthesizer.sample_from_conditions([fraud_
    condition])
synthetic_non_fraud_data = synthesizer.sample_from_conditions([non_
    fraud_condition])

synthetic_data = pd.concat([synthetic_fraud_data, synthetic_non_
    fraud_data]).reset_index(drop=True)
synthetic_data = synthetic_data.sample(frac=1).reset_index(drop=True
    )
synthetic_data.to_csv('/content/drive/MyDrive/synthetic_data_10_1_
    ratio.csv', index=False)

class_counts = synthetic_data['Class'].value_counts()
print(class_counts)
```

Program B.6: Kreiranje sintetičkog dataseta sa omjerom 10:1

```

from sdv.sampling import Condition
total_rows = len(df)
num_fraud = df['Class'].sum()
num_non_fraud = total_rows - num_fraud
target_fraud = round(total_rows * 0.0017)  # Ciljani broj prijevara
target_non_fraud = total_rows - target_fraud

fraud_condition = Condition(num_rows=target_fraud, column_values={'
    Class': 1})
non_fraud_condition = Condition(num_rows=target_non_fraud, column_
    values={'Class': 0})

synthetic_fraud_data = synthesizer.sample_from_conditions([fraud_
    condition])
synthetic_non_fraud_data = synthesizer.sample_from_conditions([non_
    fraud_condition])

```

```
synthetic_data = pd.concat([synthetic_fraud_data, synthetic_non_
    fraud_data]).reset_index(drop=True)
synthetic_data = synthetic_data.sample(frac=1).reset_index(drop=True
    )
synthetic_data.to_csv("/content/drive/MyDrive/synthetic_datasetv4",
    index=False)


dfv1 =  pd.read_csv("/content/drive/MyDrive/synthetic_datasetv1")
dfv2 =  pd.read_csv("/content/drive/MyDrive/synthetic_datasetv2")
dfv3 = pd.read_csv("/content/drive/MyDrive/synthetic_data_10_1_ratio
    .csv")
dfv4 = pd.read_csv("/content/drive/MyDrive/synthetic_datasetv4")


def plot_distribution(data, title, filename):
    plt.figure(figsize=(10, 6))
    sns.countplot(x='Class', data=data)
    plt.title(title)
    plt.xlabel('Class')
    plt.ylabel('Count')
    plt.xticks([0, 1], ['Non-Fraud', 'Fraud'])
    plt.savefig(filename)
    plt.show()

plot_distribution(dfv4, 'Distribucija␣transakcija␣u␣sintetickom␣
    datasetu␣4', 'dfv4.png')

class_counts=dfv1['Class'].value_counts()
print(class_counts)

class_counts=dfv2['Class'].value_counts()
print(class_counts)

class_counts=dfv3['Class'].value_counts()
print(class_counts)

class_counts=dfv4['Class'].value_counts()
print(class_counts)
```

Program B.7: Kreiranje još jednog sintetičkog dataseta sličnog početnom

B.3 Evaluacija kvalitete sintetičkih datasetova

```
from sdv.evaluation.single_table import evaluate_quality

quality_report = evaluate_quality(
    real_data=df,
    synthetic_data=dfv1,
    metadata=metadata)
```

Program B.8: Evaluacija kvaliteta za dataset 1

```
from sdv.evaluation.single_table import evaluate_quality

quality_report = evaluate_quality(
```

```
    real_data=df,
    synthetic_data=dfv2,
    metadata=metadata)
```

Program B.9: Evaluacija kvaliteta za dataset 2

```
from sdv.evaluation.single_table import evaluate_quality

quality_report = evaluate_quality(
    real_data=df,
    synthetic_data=dfv3,
    metadata=metadata)
```

Program B.10: Evaluacija kvaliteta za dataset 3

```
from sdv.evaluation.single_table import evaluate_quality

quality_report = evaluate_quality(
    real_data=df,
    synthetic_data=dfv4,
    metadata=metadata)
```

Program B.11: Evaluacija kvaliteta za dataset 4

```
from sdv.evaluation.single_table import get_column_plot
import matplotlib.pyplot as plt

fig = get_column_plot(
    real_data=df,
    synthetic_data=dfv2,
    metadata=metadata,
    column_name='Class'
)
fig.show()
```

Program B.12: Vizualizacija kolona za dataset 2

```
from sdv.evaluation.single_table import get_column_plot
import matplotlib.pyplot as plt

fig = get_column_plot(
    real_data=df,
    synthetic_data=dfv3,
    metadata=metadata,
    column_name='Class'
)
fig.show()
```

Program B.13: Vizualizacija kolona za dataset 3

B.4 Kombinovanje pocetnog i sintetičkih datasetova

```
merged_data  = pd.concat([df, dfv2], ignore_index=True)
merged_data = merged_data.sample(frac=1).reset_index(drop=True)
merged_data2  = pd.concat([df, dfv1], ignore_index=True)
merged_data2 = merged_data2.sample(frac=1).reset_index(drop=True)
```

```
merged_data3 = pd.concat([df, dfv3], ignore_index=True)
merged_data3 = merged_data3.sample(frac=1).reset_index(drop=True)

class_counts=merged_data['Class'].value_counts()
print(class_counts)
class_counts=merged_data2['Class'].value_counts()
print(class_counts)
class_counts=merged_data3['Class'].value_counts()
print(class_counts)
```

Program B.14: Mergeovanje podataka

B.5 Treniranje i testiranje modela: Distribucija podataka

```
import matplotlib.pyplot as plt
import seaborn as sns

plot_distribution(merged_data, 'Distribucija transakcija u Kombinovanom datasetu 1', 'Merged_data_distribution.png')
plot_distribution(merged_data2, 'Distribucija transakcija u Kombinovanom datasetu 2', 'Merged_data2_distribution.png')
plot_distribution(merged_data3, 'Distribucija transakcija u Kombinovanom datasetu 3', 'Merged_data3_distribution.png')
```

Program B.15: Distribucija podataka

B.6 Evaluacija i plot funkcija

```
import matplotlib.pyplot as plt
from sklearn.metrics import precision_recall_curve, roc_curve, auc, roc_auc_score

def evaluate_and_plot_model(y_val, y_probs, model_name):
    precision, recall, _ = precision_recall_curve(y_val, y_probs)
    auprc = auc(recall, precision)
    print(f'Area Under the Precision-Recall Curve (AUPRC) for {model_name}: {auprc:.4f}')

    fpr, tpr, _ = roc_curve(y_val, y_probs)
    roc_auc = roc_auc_score(y_val, y_probs)
    print(f'Area Under the ROC Curve (ROC-AUC) for {model_name}: {roc_auc:.4f}')

    plt.figure(figsize=(8, 6))
    plt.plot(recall, precision, marker='.', label=f'{model_name} (AUPRC = {auprc:.4f})')
    plt.xlabel('Recall')
    plt.ylabel('Precision')
    plt.title(f'Precision-Recall Curve for {model_name}')
    plt.legend()
    plt.show()

    plt.figure(figsize=(8, 6))
```

```
    plt.plot(fpr, tpr, marker='.', label=f'{model_name} (ROC-AUC = {
        roc_auc:.4f})')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title(f'ROC Curve for {model_name}')
    plt.legend()
    plt.show()
```

Program B.16: Evaluacija i plot funkcija

B.7 Logistička regresija: Realni podaci

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np

x = df.drop(columns=['Class']).values
y = df['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

lr_model = LogisticRegression()
lr_model.fit(x_train, y_train.ravel())

joblib_file = "lr_realmodel.pkl"
joblib.dump(lr_model, joblib_file)

y_pred_lr = lr_model.predict(x_val)
cnf_matrix_lr = confusion_matrix(y_val, y_pred_lr)

sns.heatmap(pd.DataFrame(cnf_matrix_lr), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Actual Label')
plt.xlabel('Predicted Label')
plt.title('Confusion Matrix for LogisticRegression on Validation Set
    ')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lr,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tClassification Report for Validation Set")
print(indented_classification_report)

y_probs_lr = lr_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lr, 'Logistic Regression')
```

Program B.17: Logistička regresija: Realni podaci

B.8 Logistička regresija: Merged Data

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report,
    roc_curve, auc, roc_auc_score
import numpy as np

x = merged_data.drop(columns=['Class']).values
y = merged_data['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
   =0.2, random_state=0, stratify=y)

lr_model = LogisticRegression()
lr_model.fit(x_train, y_train)

joblib_file = "lr_model.pkl"
joblib.dump(lr_model, joblib_file)

y_pred_lr = lr_model.predict(x_val)
cnf_matrix_lr = confusion_matrix(y_val, y_pred_lr)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_lr), annot=True, cmap="YlGnBu",
   fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za LogisticRegression na
   Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lr,
   target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
   classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

y_probs_lr = lr_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lr, 'Logistička Regresija')
```

Program B.18: Logistička regresija: Merged Data

B.9 Logistička regresija: Merged Data 2

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report,
    roc_curve, auc, roc_auc_score
```

```
import numpy as np

x = merged_data2.drop(columns=['Class']).values
y = merged_data2['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

lr_model = LogisticRegression()
lr_model.fit(x_train, y_train)

joblib_file = "lr_model2.pkl"
joblib.dump(lr_model, joblib_file)

y_pred_lr = lr_model.predict(x_val)
cnf_matrix_lr = confusion_matrix(y_val, y_pred_lr)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_lr), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za LogisticRegression na
    Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lr,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

y_probs_lr = lr_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lr, 'Logistička Regresija')
```

Program B.19: Logistička regresija: Merged Data 2

B.10 Logistička regresija: Merged Data 3

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report,
    roc_curve, auc, roc_auc_score
import numpy as np

x = merged_data3.drop(columns=['Class']).values
y = merged_data3['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

lr_model = LogisticRegression()
```

```
lr_model.fit(x_train, y_train)

joblib_file = "lr_model3.pkl"
joblib.dump(lr_model, joblib_file)

y_pred_lr = lr_model.predict(x_val)
cnf_matrix_lr = confusion_matrix(y_val, y_pred_lr)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_lr), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predvidena Labela')
plt.title('Konfuzijska Matrica za LogisticRegression na
    Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lr,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tIzvjestaj Klasifikacije za Validacijski Set")

y_probs_lr = lr_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lr, 'Logisticka Regresija')
```

Program B.20: Logistička regresija: Merged Data 3

B.11 Random Forest Classifier: Realni podaci

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np

x = df.drop(columns=['Class']).values
y = df['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

rfc_model = RandomForestClassifier()
rfc_model.fit(x_train, y_train.ravel())

joblib_file = "rfc_realmodel.pkl"
joblib.dump(rfc_model, joblib_file)

y_pred_rfc = rfc_model.predict(x_val)
cnf_matrix_rfc = confusion_matrix(y_val, y_pred_rfc)

```

```
sns.heatmap(pd.DataFrame(cnf_matrix_rfc), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska matrica za RandomForestClassifier na Validacijskom Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_rfc, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_rfc = rfc_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_rfc, 'RandomForestClassifier')
```

Program B.21: Random Forest Classifier: Realni podaci

B.12 Random Forest Classifier: Merged Data

```
from sklearn.ensemble import RandomForestClassifier

x = merged_data.drop(columns=['Class']).values
y = merged_data['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size=0.2, random_state=0, stratify=y)

rfc_model = RandomForestClassifier()
rfc_model.fit(x_train, y_train)

joblib_file = "rfc_model.pkl"
joblib.dump(rfc_model, joblib_file)

y_pred_rfc = rfc_model.predict(x_val)
cnf_matrix_rfc = confusion_matrix(y_val, y_pred_rfc)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_rfc), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za RandomForestClassifier na Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_rfc, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

```

```
y_probs_rfc = rfc_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_rfc, 'Random Forest Classifier')
```

Program B.22: Random Forest Classifier: Merged Data

B.13 Random Forest Classifier: Merged Data 2

```
from sklearn.ensemble import RandomForestClassifier

x = merged_data2.drop(columns=['Class']).values
y = merged_data2['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size=0.2, random_state=0, stratify=y)

rfc_model = RandomForestClassifier()
rfc_model.fit(x_train, y_train)

joblib_file = "rfc_model2.pkl"
joblib.dump(rfc_model, joblib_file)

y_pred_rfc = rfc_model.predict(x_val)
cnf_matrix_rfc = confusion_matrix(y_val, y_pred_rfc)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_rfc), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za RandomForestClassifier na Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_rfc, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

y_probs_rfc = rfc_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_rfc, 'Random Forest Classifier')
```

Program B.23: Random Forest Classifier: Merged Data 2

B.14 Random Forest Classifier: Merged Data 3

```
from sklearn.ensemble import RandomForestClassifier

x = merged_data3.drop(columns=['Class']).values
y = merged_data3['Class'].values
```

```
5
6  x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
       =0.2, random_state=0, stratify=y)
7
8  rfc_model = RandomForestClassifier()
9  rfc_model.fit(x_train, y_train)
10
11 joblib_file = "rfc_model3.pkl"
12 joblib.dump(rfc_model, joblib_file)
13
14 y_pred_rfc = rfc_model.predict(x_val)
15 cnf_matrix_rfc = confusion_matrix(y_val, y_pred_rfc)
16
17 plt.figure(figsize=(8, 6))
18 sns.heatmap(pd.DataFrame(cnf_matrix_rfc), annot=True, cmap="YlGnBu",
       fmt='g')
19 plt.ylabel('Stvarna␣Labela')
20 plt.xlabel('Predviđena␣Labela')
21 plt.title('Konfuzijska␣Matrica␣za␣RandomForestClassifier␣na␣
       Validacijskom␣Setu')
22 plt.show()
23
24 labels = ['Non-fraud', 'Fraud']
25 classification_report_str = classification_report(y_val, y_pred_rfc,
       target_names=labels)
26 indented_classification_report = '\n'.join(['\t' + line for line in
       classification_report_str.split('\n')])
27 print("\tIzvještaj␣Klasifikacije␣za␣Validacijski␣Set")
28
29 y_probs_rfc = rfc_model.predict_proba(x_val)[:, 1]
30 evaluate_and_plot_model(y_val, y_probs_rfc, 'Random␣Forest␣
       Classifier')
```

Program B.24: Random Forest Classifier: Merged Data 3

B.15 XGBoost: Realni podaci

```
1  import xgboost as xgb
2
3  x = df.drop(columns=['Class']).values
4  y = df['Class'].values
5
6  x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
       =0.2, random_state=0, stratify=y)
7
8  xgb_model = xgb.XGBClassifier()
9  xgb_model.fit(x_train, y_train)
10
11 joblib_file = "xgb_modelreal.pkl"
12 joblib.dump(xgb_model, joblib_file)
13
14 y_pred_xgb = xgb_model.predict(x_val)
15 cnf_matrix_xgb = confusion_matrix(y_val, y_pred_xgb)
16
17 plt.figure(figsize=(8, 6))
```

```
sns.heatmap(pd.DataFrame(cnf_matrix_xgb), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za XGBoost na Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_xgb, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

y_probs_xgb = xgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_xgb, 'XGBoost')
```

Program B.25: XGBoost: Realni podaci

B.16 XGBoost: Merged Data

```
import xgboost as xgb

x = merged_data.drop(columns=['Class']).values
y = merged_data['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size=0.2, random_state=0, stratify=y)

xgb_model = xgb.XGBClassifier()
xgb_model.fit(x_train, y_train)

joblib_file = "xgb_model.pkl"
joblib.dump(xgb_model, joblib_file)

y_pred_xgb = xgb_model.predict(x_val)
cnf_matrix_xgb = confusion_matrix(y_val, y_pred_xgb)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_xgb), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za XGBoost na Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_xgb, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

y_probs_xgb = xgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_xgb, 'XGBoost')
```

Program B.26: XGBoost: Merged Data

B.17 XGBoost: Merged Data 2

```
import xgboost as xgb

x = merged_data2.drop(columns=['Class']).values
y = merged_data2['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

xgb_model = xgb.XGBClassifier()
xgb_model.fit(x_train, y_train)

joblib_file = "xgb_model2.pkl"
joblib.dump(xgb_model, joblib_file)

y_pred_xgb = xgb_model.predict(x_val)
cnf_matrix_xgb = confusion_matrix(y_val, y_pred_xgb)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_xgb), annot=True, cmap="YlGnBu",
     fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za XGBoost na Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_xgb,
     target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

y_probs_xgb = xgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_xgb, 'XGBoost')
```

Program B.27: XGBoost: Merged Data 2

B.18 XGBoost: Merged Data 3

```
import xgboost as xgb

x = merged_data3.drop(columns=['Class']).values
y = merged_data3['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

xgb_model = xgb.XGBClassifier()
```

```
xgb_model.fit(x_train, y_train)

joblib_file = "xgb_model3.pkl"
joblib.dump(xgb_model, joblib_file)

y_pred_xgb = xgb_model.predict(x_val)
cnf_matrix_xgb = confusion_matrix(y_val, y_pred_xgb)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_xgb), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za XGBoost na Validacijskom Setu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_xgb, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Set")

y_probs_xgb = xgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_xgb, 'XGBoost')
```

Program B.28: XGBoost: Merged Data 3

B.19 XGBoost: Test na dfv4

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report, precision_recall_curve, roc_curve, auc, roc_auc_score
from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
import joblib

x_real = dfv4.drop(columns=['Class']).values
y_real = dfv4['Class'].values

x_train, x_test, y_train, y_test = train_test_split(x_real, y_real, test_size=0.2, random_state=42, stratify=y_real)

model = joblib.load("xgb_model.pkl")

y_pred = model.predict(x_test)
y_probs = model.predict_proba(x_test)[:, 1]

cnf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
```

```
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska matrica za XGBoost na Testnom Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_test, y_pred,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za validacijski Skup")

precision, recall, _ = precision_recall_curve(y_test, y_probs)
auprc = auc(recall, precision)
print(f'Površina ispod Precision-Recall krivulje (AUPRC) za
    validacijski Skup: {auprc:.4f}')

fpr, tpr, _ = roc_curve(y_test, y_probs)
roc_auc = roc_auc_score(y_test, y_probs)
print(f'Površina ispod ROC krivulje (ROC-AUC) za validacijski Skup:
    {roc_auc:.4f}')

evaluate_and_plot_model(y_test, y_probs, 'XGBoost na Testnom Skupu')
```

Program B.29: XGBoost: Test na dfv4

B.20 XGBoost: Test na dfv4 (3:1)

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
import joblib

x_real = dfv4.drop(columns=['Class']).values
y_real = dfv4['Class'].values

fraud_indices = np.where(y_real == 1)[0]
non_fraud_indices = np.where(y_real == 0)[0]

num_fraud_samples = len(fraud_indices)
num_non_fraud_samples = min(3 * num_fraud_samples, len(non_fraud_
    indices))

selected_fraud_indices = fraud_indices
selected_non_fraud_indices = np.random.choice(non_fraud_indices, num
    _non_fraud_samples, replace=False)

selected_indices = np.concatenate([selected_fraud_indices, selected_
    non_fraud_indices])
x_test_3to1 = x_real[selected_indices]
y_test_3to1 = y_real[selected_indices]

```

```
model = joblib.load("xgb_model.pkl")

y_pred_3to1 = model.predict(x_test_3to1)
cnf_matrix_3to1 = confusion_matrix(y_test_3to1, y_pred_3to1)

plt.figure(figsize=(8, 6))
sns.heatmap(pd.DataFrame(cnf_matrix_3to1), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za 3:1 validacijski Skup')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_test_3to1, y_pred_3to1, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za 3:1 validacijski Skup")

y_probs_3to1 = model.predict_proba(x_test_3to1)[:, 1]
evaluate_and_plot_model(y_test_3to1, y_probs_3to1, 'XGBoost na 3:1 Testnom Skupu')
```

Program B.30: XGBoost: Test na dfv4 (3:1)

B.21 XGBoost: Realni podaci test na dfv4 (3:1)

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report, precision_recall_curve, roc_curve, auc, roc_auc_score
from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
import joblib

x_real = dfv4.drop(columns=['Class']).values
y_real = dfv4['Class'].values

fraud_indices = np.where(y_real == 1)[0]
non_fraud_indices = np.where(y_real == 0)[0]

num_fraud_samples = len(fraud_indices)
num_non_fraud_samples = min(3 * num_fraud_samples, len(non_fraud_indices))

selected_fraud_indices = fraud_indices
selected_non_fraud_indices = np.random.choice(non_fraud_indices, num_non_fraud_samples, replace=False)

selected_indices = np.concatenate([selected_fraud_indices, selected_non_fraud_indices])
x_test_3to1 = x_real[selected_indices]
y_test_3to1 = y_real[selected_indices]
```

```
25
26 model = joblib.load("xgb_realmodel.pkl")
27
28 y_pred_3to1 = model.predict(x_test_3to1)
29 cnf_matrix_3to1 = confusion_matrix(y_test_3to1, y_pred_3to1)
30
31 plt.figure(figsize=(8, 6))
32 sns.heatmap(pd.DataFrame(cnf_matrix_3to1), annot=True, cmap="YlGnBu"
       , fmt='g')
33 plt.ylabel('Stvarna␣Labela')
34 plt.xlabel('Predviđena␣Labela')
35 plt.title('Konfuzijska␣Matrica␣za␣3:1␣validacijski␣Skup')
36 plt.show()
37
38 labels = ['Non-fraud', 'Fraud']
39 classification_report_str = classification_report(y_test_3to1, y_
       pred_3to1, target_names=labels)
40 indented_classification_report = '\n'.join(['\t' + line for line in
       classification_report_str.split('\n')])
41 print("\tIzvještaj␣Klasifikacije␣za␣3:1␣validacijski␣Skup")
42
43 y_probs_3to1 = model.predict_proba(x_test_3to1)[:, 1]
44 evaluate_and_plot_model(y_test_3to1, y_probs_3to1, 'XGBoost␣na␣3:1␣
       Testnom␣Skupu')
```

Program B.31: XGBoost: Realni podaci test na dfv4 (3:1)

B.22 LightGBM: Realni podaci

```
1  import matplotlib.pyplot as plt
2  import seaborn as sns
3  import joblib
4  from sklearn.model_selection import train_test_split
5  from sklearn.metrics import confusion_matrix, classification_report,
       precision_recall_curve, roc_curve, auc, roc_auc_score
6  import numpy as np
7  import lightgbm as lgb
8
9  x = df.drop(columns=['Class']).values
10 y = df['Class'].values
11
12 x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
       =0.2, random_state=0, stratify=y)
13
14 lgb_model = lgb.LGBMClassifier()
15 lgb_model.fit(x_train, y_train)
16
17 joblib_file = "lgb_realmodel.pkl"
18 joblib.dump(lgb_model, joblib_file)
19
20 y_pred_lgb = lgb_model.predict(x_val)
21 cnf_matrix_lgb = confusion_matrix(y_val, y_pred_lgb)
22
23 sns.heatmap(pd.DataFrame(cnf_matrix_lgb), annot=True, cmap="YlGnBu",
       fmt='g')
24 plt.ylabel('Stvarna␣Labela')
```

```
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za LightGBM na Validacijskom Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lgb,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
   classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_lgb = lgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lgb, 'LightGBM')
```

Program B.32: LightGBM: Realni podaci

B.23 LightGBM: Merged Data

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np
import lightgbm as lgb

x = merged_data.drop(columns=['Class']).values
y = merged_data['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
   =0.2, random_state=0, stratify=y)

lgb_model = lgb.LGBMClassifier()
lgb_model.fit(x_train, y_train)

joblib_file = "lgb_model.pkl"
joblib.dump(lgb_model, joblib_file)

y_pred_lgb = lgb_model.predict(x_val)
cnf_matrix_lgb = confusion_matrix(y_val, y_pred_lgb)

sns.heatmap(pd.DataFrame(cnf_matrix_lgb), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za LightGBM na Validacijskom Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lgb,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
   classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

```

```
y_probs_lgb = lgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lgb, 'LightGBM')
```

Program B.33: LightGBM: Merged Data

B.24 LightGBM: Merged Data 2

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np
import lightgbm as lgb

x = merged_data2.drop(columns=['Class']).values
y = merged_data2['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
   =0.2, random_state=0, stratify=y)

lgb_model = lgb.LGBMClassifier()
lgb_model.fit(x_train, y_train)

joblib_file = "lgb_model2.pkl"
joblib.dump(lgb_model, joblib_file)

y_pred_lgb = lgb_model.predict(x_val)
cnf_matrix_lgb = confusion_matrix(y_val, y_pred_lgb)

sns.heatmap(pd.DataFrame(cnf_matrix_lgb), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za LightGBM na Validacijskom Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lgb,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
   classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_lgb = lgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lgb, 'LightGBM')
```

Program B.34: LightGBM: Merged Data 2

B.25 LightGBM: Merged Data 3

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np
import lightgbm as lgb

x = merged_data3.drop(columns=['Class']).values
y = merged_data3['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
   =0.2, random_state=0, stratify=y)

lgb_model = lgb.LGBMClassifier()
lgb_model.fit(x_train, y_train)

joblib_file = "lgb_model3.pkl"
joblib.dump(lgb_model, joblib_file)

y_pred_lgb = lgb_model.predict(x_val)
cnf_matrix_lgb = confusion_matrix(y_val, y_pred_lgb)

sns.heatmap(pd.DataFrame(cnf_matrix_lgb), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska Matrica za LightGBM na Validacijskom Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_lgb,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
   classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_lgb = lgb_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_lgb, 'LightGBM')
```

Program B.35: LightGBM: Merged Data 3

B.26 Neural Networks: Realni podaci

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np

x = df.drop(columns=['Class']).values
y = df['Class'].values

```

```
x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

nn_model = MLPClassifier()
nn_model.fit(x_train, y_train.ravel())

joblib_file = "nn_realmodel.pkl"
joblib.dump(nn_model, joblib_file)

y_pred_nn = nn_model.predict(x_val)
cnf_matrix_nn = confusion_matrix(y_val, y_pred_nn)

sns.heatmap(pd.DataFrame(cnf_matrix_nn), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska matrica za Neuronsku Mrežu na Validacijskom
    Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_nn,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_nn = nn_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_nn, 'Neuronska Mreža')
```

Program B.36: Neural Networks: Realni podaci

B.27 Neural Networks: Merged Data

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, classification_report,
    precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np

x = merged_data.drop(columns=['Class']).values
y = merged_data['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

nn_model = MLPClassifier()
nn_model.fit(x_train, y_train.ravel())

joblib_file = "nn_model.pkl"
joblib.dump(nn_model, joblib_file)

y_pred_nn = nn_model.predict(x_val)
```

```
cnf_matrix_nn = confusion_matrix(y_val, y_pred_nn)

sns.heatmap(pd.DataFrame(cnf_matrix_nn), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska matrica za Neuronsku Mrežu na Validacijskom Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_nn, target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_nn = nn_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_nn, 'Neuronska Mreža')
```

Program B.37: Neural Networks: Merged Data

B.28 Neural Networks: Merged Data 2

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, classification_report, precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np

x = merged_data2.drop(columns=['Class']).values
y = merged_data2['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size=0.2, random_state=0, stratify=y)

nn_model = MLPClassifier()
nn_model.fit(x_train, y_train.ravel())

joblib_file = "nn_model2.pkl"
joblib.dump(nn_model, joblib_file)

y_pred_nn = nn_model.predict(x_val)
cnf_matrix_nn = confusion_matrix(y_val, y_pred_nn)

sns.heatmap(pd.DataFrame(cnf_matrix_nn), annot=True, cmap="YlGnBu", fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska matrica za Neuronsku Mrežu na Validacijskom Skupu')
plt.show()

```

```
labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_nn,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_nn = nn_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_nn, 'Neuronska Mreža')
```

Program B.38: Neural Networks: Merged Data 2

B.29 Neural Networks: Merged Data 3

```
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, classification_report,
     precision_recall_curve, roc_curve, auc, roc_auc_score
import numpy as np

x = merged_data3.drop(columns=['Class']).values
y = merged_data3['Class'].values

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size
    =0.2, random_state=0, stratify=y)

nn_model = MLPClassifier()
nn_model.fit(x_train, y_train.ravel())

joblib_file = "nn_model3.pkl"
joblib.dump(nn_model, joblib_file)

y_pred_nn = nn_model.predict(x_val)
cnf_matrix_nn = confusion_matrix(y_val, y_pred_nn)

sns.heatmap(pd.DataFrame(cnf_matrix_nn), annot=True, cmap="YlGnBu",
    fmt='g')
plt.ylabel('Stvarna Labela')
plt.xlabel('Predviđena Labela')
plt.title('Konfuzijska matrica za Neuronsku Mrežu na Validacijskom
    Skupu')
plt.show()

labels = ['Non-fraud', 'Fraud']
classification_report_str = classification_report(y_val, y_pred_nn,
    target_names=labels)
indented_classification_report = '\n'.join(['\t' + line for line in
    classification_report_str.split('\n')])
print("\tIzvještaj Klasifikacije za Validacijski Skup")

y_probs_nn = nn_model.predict_proba(x_val)[:, 1]
evaluate_and_plot_model(y_val, y_probs_nn, 'Neuronska Mreža')
```

Program B.39: Neural Networks: Merged Data 3

Literatura

[1] Joshi, A., Soni, S., Jain, V., "An experimental study using unsupervised machine learning techniques for credit card fraud detection", ResearchGate, 2021, dostupno na: https://www.researchgate.net/publication/353143969_An_Experimental_Study_using_Unsupervised_Machine_Learning_Techniques_for_Credit_Card_Fraud_Detection

[2] Li, H., Xiong, L., Jiang, X., "Differentially private synthesization of multi-dimensional data using copula functions", ResearchGate, 2014, dostupno na: https://www.researchgate.net/figure/Synthetic-data-generation_fig1_268793256

[3] Lad, S., "Introduction to generative adversarial networks - part i", dostupno na: https://www.linkedin.com/pulse/introduction-generative-adversarial-networks-part-i-sagar-lad/ Accessed: 2024-07-15. 2020.

[4] Alizadeh, E., "Sdv library for modeling datasets", Ebrahim Alizadeh Blog, 2021, dostupno na: https://ealizadeh.com/blog/sdv-library-for-modeling-datasets/

[5] Patki, N., Wedge, R., Veeramachaneni, K., "The synthetic data vault", in 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), 2016, str. 399-410.

[6] Xu, L., Veeramachaneni, K., "Synthesizing tabular data using generative adversarial networks", arXiv preprint arXiv:1907.00503, 2019, dostupno na: https://arxiv.org/abs/1907.00503

[7] Goodfellow, I., Bengio, Y., Courville, A., Deep Learning. MIT Press, 2016.

[8] Radford, A., Metz, L., Chintala, S., "Unsupervised representation learning with deep convolutional generative adversarial networks", arXiv preprint arXiv:1511.06434, 2015.

[9] Kroese, D. P., Taimre, T., Botev, Z. I., Handbook of Monte Carlo Methods. Hoboken, NJ: John Wiley Sons, 2011.

[10] Metropolis, N., Ulam, S., "The monte carlo method", Journal of the American Statistical Association, Vol. 44, No. 247, 1949, str. 335–341.

[11] Hastie, T., Tibshirani, R., Friedman, J., The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York, NY: Springer, 2001.

[12] Pozzolo, A. D., Caelen, O., Boracchi, G., Alippi, C., Bontempi, G., "Credit card fraud detection: a realistic modeling and a novel learning strategy", IEEE transactions on neural networks and learning systems, Vol. 29, No. 8, 2018, str. 3784-3797.

[13] Carcillo, F., Pozzolo, A. D., Borgne, Y.-A. L., Caelen, O., Bontempi, G., "Scarff: a scalable framework for streaming credit card fraud detection with spark", Information Fusion, Vol. 41, 2018, str. 182-194.

[14] Borgne, Y.-A. L., Bontempi, G., "Reproducible machine learning for credit card fraud detection - practical handbook", in International Conference on Data Science and Advanced Analytics (DSAA). IEEE, 2019.

[15] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., "Generative adversarial networks", arXiv preprint arXiv:1406.2661, 2014.

[16] Mirza, M., Osindero, S., "Conditional generative adversarial nets", in arXiv preprint arXiv:1411.1784, 2014.

[17] Xu, L., Skoularidou, M., Cuesta-Infante, A., Veeramachaneni, K., "Modeling tabular data using conditional gan", arXiv preprint arXiv:1907.00503, 2019.

[18] Hochreiter, S., Schmidhuber, J., "Long short-term memory", Neural Computation, Vol. 9, No. 8, 1997, str. 1735-1780.

[19] Kingma, D. P., Welling, M., "Auto-encoding variational bayes", arXiv preprint arXiv:1312.6114, 2013.

[20] (SDV), S. D. V., "Synthetic data vault (sdv) documentation", dostupno na: https://sdv.dev/ 2021.

[21] Carcillo, F., Borgne, Y.-A. L., Caelen, O., Bontempi, G., "Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization", International Journal of Data Science and Analytics, Vol. 5, No. 4, 2018, str. 285–300.

[22] Patki, N., Wedge, R., Veeramachaneni, K., "The synthetic data vault", in IEEE International Conference on Data Science and Advanced Analytics (DSAA), 2016, str. 399-410.

[23] Reynolds, D. A., Chellappa, R., "Gaussian mixture models", Encyclopedia of Biometrics, 2009, str. 659-663.

[24] Dempster, A. P., Laird, N. M., Rubin, D. B., "Maximum likelihood from incomplete data via the em algorithm", Journal of the Royal Statistical Society: Series B (Methodological), Vol. 39, No. 1, 1977, str. 1-22.

[25] Koller, D., Friedman, N., Probabilistic Graphical Models: Principles and Techniques. Cambridge, MA: MIT Press, 2009.

[26] Faker, "Faker documentation", dostupno na: https://faker.readthedocs.io/ 2021.

[27] Synthetics, G., "Gretel synthetics documentation", dostupno na: https://docs.gretel.ai/ 2021.

[28] Pozzolo, A. D., Caelen, O., Johnson, R. A., Bontempi, G., "Calibrating probability with undersampling for unbalanced classification", in Symposium on Computational Intelligence and Data Mining (CIDM). IEEE, 2015.

[29] Pozzolo, A. D., Caelen, O., Borgne, Y.-A. L., Waterschoot, S., Bontempi, G., "Learned lessons in credit card fraud detection from a practitioner perspective", Expert systems with applications, Vol. 41, No. 10, 2014, str. 4915–4928.

[30] Pozzolo, A. D., Boracchi, G., Caelen, O., Alippi, C., Bontempi, G., "Credit card fraud detection: a realistic modeling and a novel learning strategy", IEEE transactions on neural networks and learning systems, Vol. 29, No. 8, 2018, str. 3784–3797.

[31] Pozzolo, A. D., "Adaptive machine learning for credit card fraud detection", Doktorski rad, ULB MLG, 2015, phD thesis supervised by G. Bontempi.

[32] Carcillo, F., Pozzolo, A. D., Borgne, Y.-A. L., Caelen, O., Mazzer, Y., Bontempi, G., "Scarff: a scalable framework for streaming credit card fraud detection with spark", Information fusion, Vol. 41, 2018, str. 182–194.

[33] Jolliffe, I. T., Principal Component Analysis, 2nd ed. New York, NY: Springer, 2002.

[34] Wold, S., Esbensen, K., Geladi, P., "Principal component analysis", Chemometrics and Intelligent Laboratory Systems, Vol. 2, 1987, str. 37-52.

[35] Powers, D. M. W., Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation. New York, NY: Springer, 2011.

[36] Fawcett, T., "An introduction to roc analysis", Pattern Recognition Letters, Vol. 27, No. 8, 2006, str. 861-874.

[37] Van Rijsbergen, C. J., "Information retrieval", Butterworth-Heinemann, 1979.

[38] Saito, T., Rehmsmeier, M., "The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets", PLOS ONE, Vol. 10, No. 3, 2015, str. 1-21.

[39] Hanley, J. A., McNeil, B. J., "The meaning and use of the area under a receiver operating characteristic (roc) curve", Radiology, Vol. 143, No. 1, 1982, str. 29-36.

[40] Hasanin, e. a., "Data sampling approaches with severely imbalanced big data for medicare fraud detection", Journal of Big Data, 2018.

[41] Neptune.ai, "F1 score vs roc auc vs accuracy vs pr auc: Which evaluation metric should you choose?", Neptune Blog, 2020, dostupno na: https://neptune.ai/blog/f1-score-accuracy-roc-auc-pr-auc

[42] Bishop, C. M., Pattern Recognition and Machine Learning. New York, NY: Springer, 2006.

[43] Breiman, L., "Random forests", Machine Learning, Vol. 45, No. 1, 2001, str. 5-32.

[44] Chen, T., Guestrin, C., "Xgboost: A scalable tree boosting system", in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, 2016, str. 785-794.

[45] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.-Y., "Lightgbm: A highly efficient gradient boosting decision tree", in Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, 2017, str. 3149-3157.

[46] Breiman, L., “Random forests”, Machine Learning, Vol. 45, No. 1, 2001, str. 5-32.

[47] Chen, C., Liaw, A., Breiman, L., “Using random forest to learn imbalanced data”, University of California, Berkeley, Tech. Rep., 2004.

[48] Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P., “Smote: Synthetic minority over-sampling technique”, in Journal of Artificial Intelligence Research, 2002, str. 321–357.

[49] Bellinger, C., Drummond, C., Japkowicz, N., “Synthetic oversampling for advanced data mining applications”, Knowledge and Information Systems, Vol. 62, No. 3, 2020, str. 849-878.

[50] Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P., “Smote: Synthetic minority over-sampling technique”, Journal of Artificial Intelligence Research, Vol. 16, 2002, str. 321-357.

[51] Bellinger, C., Drummond, C., Japkowicz, N., “Synthetic oversampling for advanced data mining applications”, Knowledge and Information Systems, Vol. 62, No. 3, 2020, str. 849-878.