UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET
ODSJEK ZA RAČUNARSTVO I INFORMATIKU

Detekcija prevara sa kreditnim karticama

PROJEKTNI ZADATAK

- DRUGI CIKLUS STUDIJA -

Profesor:

Prof. dr. Amila Akagić

Studenti:

Bečirević Lejla
Nedžibović Amila
Topalović Senad

Sarajevo,
januar 2023.

Sadržaj

Uvod

Danas se većina modernih rješenja za detekciju prevara kreditnih kartica oslanja na metode vještačke inteligencije i mašinskog učenja. U ovom radu je predložen model duboke neuronske mreže za detekciju prevara sa kreditnim karticama. Korišteni skup podataka sadrži podatke o transakcijama iz septembra 2013. godine, prikupljenih u Europi u toku 2 dana. Ukupan broj transakcija iznosi 284807, od kojih su 492 transakcije klasifikovane kao lažne. S obzirom da je skup podataka nebalansiran, za balansiranje trening skupa korištena je SMOTE tehnika *oversampling*-a, te hibridna metoda: SMOTE i *Random undersampling*. Predloženi model duboke neuronske mreže se sastoji od ulaznog sloja, 6 skrivenih slojeva, 4 dropout sloja i izlaznog sloja sa sigmoid aktivacijskom funkcijom. Za pronalazak optimalnih hiperparametara, ovaj rad koristi KerasTuner biblioteku i Hyperband tuner. Pored predloženog modela duboke neuronske mreže, urađena je i klasifikacija validnih i lažnih transakcija sa modelom logističke regresije i random forest klasifikatorom, te je data usporedba performansi sva tri modela.

1. Opis problema

Kreditna kartica se odnosi na karticu dodijeljenu korisniku, koja mu omogućava kupovinu robe i usluga unutar limita kreditne kartice. Jedan od čestih problema koji se dešavaju sa kreditnim karticama jeste neovlašteni pristup, odnosno prevare na kreditnim karticama. Prevara je jedna od najznačajnijih briga u smislu novčanih gubitaka, ne samo za trgovce već i za pojedinca, stoga je od velikog značaja pravovremena identifikacija transakcijskih prevara.

1.1 Credit card fraud

Credit card fraud je termin koji se koristi za neovlašteni pristup kreditnim ili debitnim karticama, odnosno neovlašteno uzimanje podataka o kreditnoj kartici drugog lica u svrhu naplate kupovina ili skidanja sredstava sa kartice. Kada dođe do neovlaštenog pristupa platnoj kartici pojedinca, posljedice snose svi koji su uključeni u taj proces - od pojedinca čiji su podaci ukradeni do preduzeća koje je izdalo karticu, te trgovca koji izvršava transakciju. Zbog navedenog je od velikog značaja identifikovati prevare platnih kartica na samom početku.
Prevare na kreditnim karticama mogu se podijeliti na:

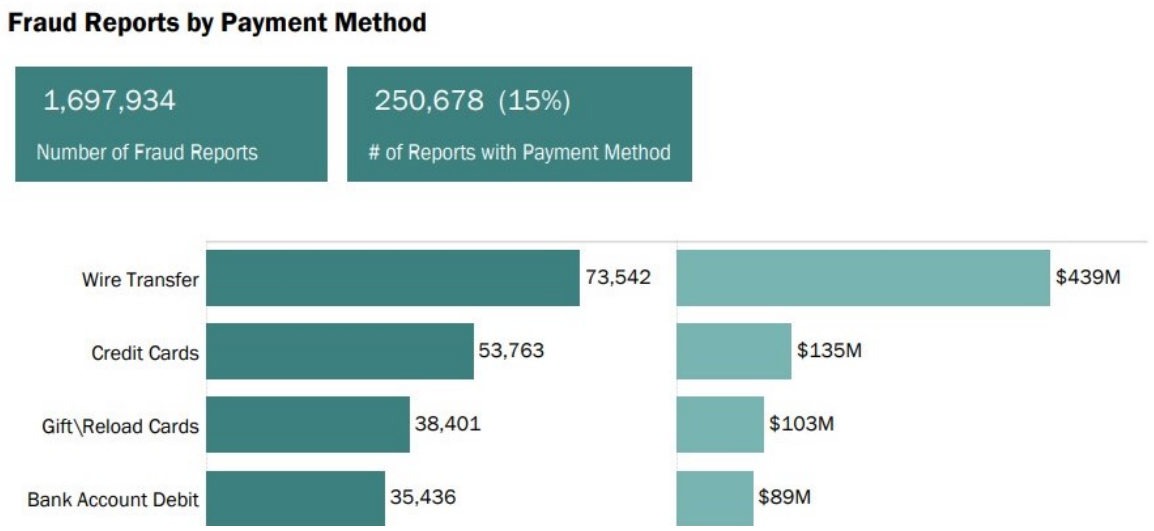
- Card present prevare - neovlaštenim, fizičkim korištenjem kartice ili pomoću ilegalno instaliranih uređaja na mjestima za plaćanje (*skimmers*) koji uzimaju podatke sa kartice.
- Card-not-present prevare - neovlaštenim korištenjem podataka sa kartice: broj kartice, ime vlasnika kartice, CVV.

1.1.1 Sigurnost kreditnih kartica

Od 2015. godine, card-not-present prevare su postale zastupljenije, a card-present prevare su se gotovo prestale dešavati uvođenjem EMV (Europay, MasterCard i Visa) tehnologije čipova. EMV tehnologija je uvela korištenje mikročipova koje je moguće očitati ubacivanjem kartica u čitač kartica ili skeniranjem prilikom beskontaktnog plaćanja. EMV metoda je uvela velike promjene koje se tiču sigurnosti, za razliku od magnetne trake koja se koristila ranije, te je onemogućila čitanje podataka sa kartice pomoću skimmera.
Međutim, istovremeno je došlo do pojave crnog tržišta za kompromitovane podatke o kreditnim karticama. Dostupnost kompromitovanih podataka o karticama je značajno povećalo card-not-present transakcijske prevare, posebno sa porastom online kupovine.

Tehnološko rješenje koje bi smanjilo card-not-present prevare, slično kao EMV, trenutno ne postoji, te je potrebno da izdavaoci kreditnih kartica primjene što bolje mjere sigurnosti, kako bi omogućili detekciju lažnih transakcija i onemogućili iste.

Na grafiku u nastavku prikazane su najzastupljenije prevare koje su prijavljene FTC-u (Federalnoj trgovinskoj komisiji SAD-a) u 2019. godini, kao i odgovarajuće iznose gubitaka u dolarima. Možemo zaključiti da su prevare sa kreditnim karticama druga najčešće prijavljivana vrsta prevare, što ukazuje na to da je potrebno uložiti velike napore kako bi se spriječile ove vrste prevara.



Slika 1.1: Broj prevara s kreditnim karticama u SAD-u u 2019. godini [1]

Danas se većina modernih rješenja za detekciju prevara kreditnih kartica oslanja na metode vještačke inteligencije i mašinskog učenja.

1.1.2 Detekcija prevara metodama vještačke inteligencije

Otkrivanje prevare s kreditnim karticama ima rastuću istragu uz pomoć tehnika vještačke inteligencije. Tehnike vještačke inteligencije omogućavaju procjenu transakcija na osnovu postojećih podataka o neovlaštenim transakcijama, te se na taj način zaustavljaju korisnici koji pokušavaju neovlašteno koristiti kreditnu karticu drugog lica. Proces automatskog razlikovanja neovlaštenih i ovlaštenih korisnika poznat je pod pojmom *credit card fraud detection* - otkrivanje prevara kreditnih kartica.

U smislu snažnog funkcionalnog sistema, glavni cilj je da se otkrije najveći mogući broj lažnih transakcija koristeći konačan skup podataka, u našem slučaju tretiran PCA metodom za anonimizaciju korisnika i minimiziranje/maksimiziranje korelacije podataka. Budući da se prevare dešavaju rjeđe nego obične transakcije, baze podataka su uvijek neuravnotežene. Potrebno je pronaći efikasnu duboku neuronsku arhitekturu zasnovanu na tačnosti.

Neuronske mreže

Vještačke neuronske mreže pokazale su se kao učinkovite u pronalasku prevare kreditnih kartica. Vještačke neuronske mreže *(eng. Artificial Neural Network - ANN)* su dizajnirane kao matematička generalizacija komponenti ljudskog mozga, posebno mreže neurona koji primaju podatke kako bi naučili karakteristike i poduzeli radnje u skladu s ciljem vještačkih neuronskih mreža [2].
Modeli zasnovani na rekurentnim neuronskim mrežama pokazuju obećavajuće rezultate[3] koristeći sekvencijalna ponašanja korisnika koja mogu pravovremeno odražavati njihove dinamične i evoluirajuće namjere, tj. uzimaju u obzir sekvence transakcija koje su se dogodile u prošlosti, kako bi se utvrdilo da li je nova transakcija legitimna ili lažna.

Detekcija prevara metodama vještačke inteligencije je obradjena u mnogim radovima koji će biti spomenuti u nastavku dokumenta, što je pokazatelj da je to često korišten alat za ovu temu.

2. Pregled stanja u oblasti

U ovom poglavlju je analizirano trenutno stanje u oblasti posmatranog problema kroz referentne radove. S obzirom da će u ovom radu biti upoređene različite metode iz područja dubokog učenja i oblasti mašinskog učenja, u ovom poglavlju su urađene analize iz obje oblasti, te je izvršena usporedba rezultata svake od metoda i navedene su njihove prednosti i nedostaci.

2.1 Analiza radova

Rad: Deep Learning Detecting Fraud in Credit Card Transactions [4]

Roy, Sun, Mahoney, Alonzi, Adam i Beling u radu *Deep Learning Detecting Fraud in Credit Card Transactions* [4] su analizirani 4 različite topologije dubokog učenja: ANN, RNNs, LSTMs, GRUs kako bi utvrdili koji model će najbolje razlikovati legalne od nelegalnih transakcija. Modeli su trenirani na nebalansiranom skupu podataka, koji sadrži samo 0.14% nelegalnih transakcija. Svaka od topologija ima različitu strukturu i parametre koji utiču na performanse učenja. Svi modeli, osim ANN-a, su dali tačnosti preko 90%, pri čemu je GRU arhitektura, zasnovana na 10-fold unakrsnoj validaciji dala najbolje rezultate, a tačnost modela iznosi 0.916. Za treniranje ovog modela, korišten je *learning rate* od 0.05, *tanh* aktivacijska funkcija i srednja apsolutna greška kao funkcija greške. Svaki model je dao najbolje performanse nakon treniranja na većim neuronskim mrežama, u ovom slučaju, sa 6 skrivenih slojeva i 150 čvorova u svakom sloju.

Rad: Credit Card Fraud Detection using Classification, Unsupervised, Neural Networks Models [5]

Bhavya, Sasidhar, Anjali i Karishma u radu *Credit Card Fraud Detection using Classification, Unsupervised, Neural Networks Models* [5] su primijenili metode mašinskog i dubokog učenja za detekciju prevara sa kreditnim karticama. Primjenje metode mašinskog učenja su Logistička regresija i K-Means, a za duboko učenje je korištena CNN sa 2 i sa 5 skrivenih slojeva. Za treniranje i testiranje modela korišten je nebalasnirani dataset, koji sadrži samo 0.172% lažnih transakcija. 75% dataset-a je korišteno za treniranje, a 25% za testiranje. Najbolje performanse je dala logistička regresija, gdje su korištene težine klasa (*class weight*) kako bi se promijenila granica odlučivanja. Logistička regresija je dala stopu tačnosti od 99.88%, a logistička regresija sa uravnoteženim težinama klasa 97.5%. Model konvolucione neuronske mreže ima gotovo jednako dobre rezultate kao i logistička regresija, gdje stopa tačnosti za model sa 5 skrivenih slojeva iznosi 99.61%. K-means ima veoma loše performanse za posmatrani dataset, s obzirom da legalne i nelegalne transkacije izgledaju veoma slično, a K-means vrši grupisanje na osnovu sličnosti i razlika u karakteristikama skupa podataka.

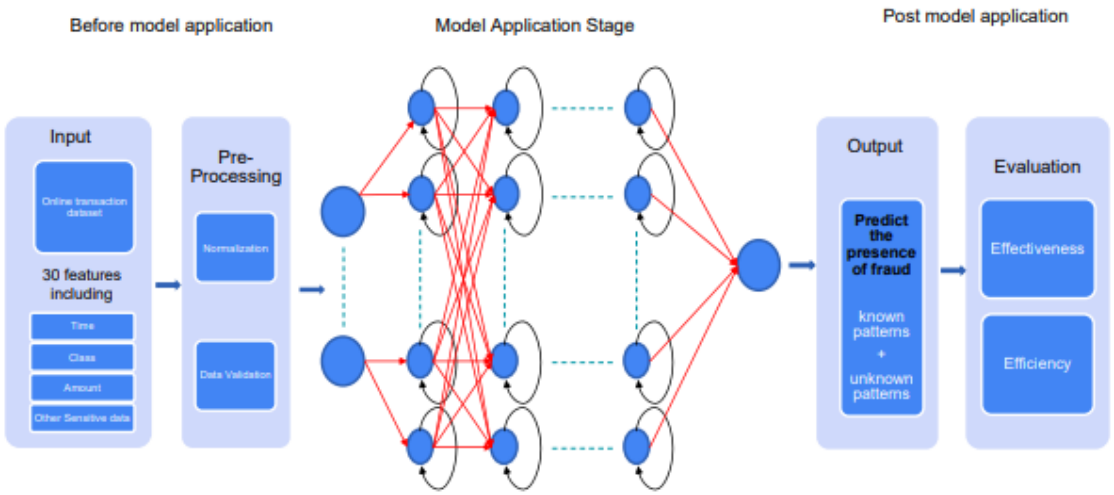
Rad: Credit card fraud detection using Bayesian and neural networks

Maes, Tuyls, Vanschoenwinkel, Manderick u radu *Credit Card Fraud Detection Using Bayesian and Neural Networks* su primjenili dvije tehnike za detekciju prevara sa kreditnim karticama: Bayesian Belief Networks i Artifical Neural Netowrks. Prilikom obrađivanja seta podataka

koristeći vještačke neuralne mreze, zaključili su da je veoma bitno preprocesiranje podataka. Nakon izvršene analize seta od 10 stavki, uočili su da je jedna stavka u strogoj korelaciji sa ostalim. Uklanjanjem ove stavke dovelo je do znatnih poboljšanja rezultata. ANN je pokazao da za 70% true positive transakcija (one koje se klasificirane kao ispravne) imamo samo 15% false positives (transakcije koje su prevara ali su označene kao ispravne). Što se tiče Bayesian Belief Networks, koristili su dataset koji ima 4 stavke i labelu "prevara". Kada se primjenio STAGE algoritam na ovu mrežu, 68% neispravnih transakcija je ispravno označeno dok je samo 10% ispravnih transakcija označeno kao prevara. Pokazali su da BBN ima bolje performanse nego ANN kada se primjeni na detekciju prevare sa transakcijama. U nekim slucajevima BNN detektuje i do 8% vise neispravnih transakcija. Također su primjetili da treniranje mreže za ANN traje i nekoliko sati dok za BNN to trajanje je znatno manje i traje do 20 minuta.

Rad: Deep Convolution Neural Network Model for Credit-Card Fraud Detection and Alert

Joy Iong-Zong Chen, Kong-Long Lai u radu *Deep Convolution Neural Network Model for Credit-Card Fraud Detection and Alert* su predstavili model koji je podjeljen u tri kategorije: prije primjene modela, tokom i nakon primjene.



Slika 2.1: Predloženi model

Data set koji je korišten sadrži 5 miliona transakcija koje su izvršene tokom 24 sata od kojih je 6223 kategorizovano kao neispravna. Set je veoma nebalansiran jer samo 0.12% seta čine neispravne transakcije. Set posjeduje preko 30 stavki. Nakon što se preprocesiranje završi, neuronska mreža dubokog učenja bazirana na pamćenju, Convolution Neural Network se primjenjuje u svrhu detekcije prevare transakcija. Prethodno obrađeni podaci se unose u model koristeći DCNN slojeve memorijskih ćelija. Rezultati su poboljšani hiperparametrima koje koristi ovaj model. Prilikom obrade, izlaz i njegova evaluacija se dobijaju tokom završne faze primjene modela. Algoritam donosi odluku da li je transakcija lažna ili ne. Također, razni parametri se testiraju kako bi se potvrdila autentičnost izlaza. Ostvarena je preciznost od 99% za 583 sekunde uz stopu gubitka od 0.32%. Za poređenje se koriste RMSprop, Adagrad i Adam optimizatori evaluacija različitih iteracija i slojeva. Sa 3 sloja kodiranja i dekodiranja i iteracijama od 500 i 1000, najbolje se pokazao Adam optimizator. Također se pokazalo da je računarski efikasan, zahtjeva manje memorijskog prostora i može da obrađuje veliku kolicinu podataka.

Što se tiče logičke regresije, u trajanju od 20 sekundi postignuta je tačnost od 99% . Kada se uporedi prema RNN, LR je postigao manju preciznost uprkos smanjenju vremenskog trajanja.

Rad: Evaluation of Deep Neural Networks for Reduction of Credit Card Fraud Alerts [6]

Autori *Rafael San Miguel Carrasco i Miguel-Ángel Sicilia-Urbán* su napisali rad na temu pronalaska prevare korištenjem kreditnih kartica upotrebom dubokog učenja putem dubokih neuronskih mreža. Cilj je bio da procijene mogu li duboke neuronske mreže precizno razlikovati dobro poznate lažne pozitivne rezultate, čime se smanjuje broj upozorenja koje analitičari moraju istražiti, i, ako je tako, u kojoj mjeri i po kojoj stopi greške. Spominje se da se u jednoj nedavnoj studiji zaključuje da su neuronske mreže najefikasnija metoda za otkrivanje prevare s kreditnim karticama, upoređujući njihove performanse s drugim tradicionalnim tehnikama mašinskog učenja, uključujući: pravila asocijacije, Support Vector Machine (SVM), K-nearest Neighbor (KNN), logistička regresija itd.
Dalje se spominju tehnike nagledanog (*Supervised*) i nenagledanog (Unsupervised) učenja, gdje su zaključili da nenadgledano učenje nadmašuje nadizirne metode kada se uzme u obzir *skewness*. Međutim, metode prevelikog uzorkovanja kao što je SMOTE dokazale su da poboljšavaju preciznost u nadgledanim modelima, kao i da kombinovanje više rezultata može negativno uticati na tačnost.
Koristio se dataset sa 446.076 stvarnih upozorenja vezanih za sumnjive transakcije kreditnim karticama iz spanske organizacije za obradu plaćanja, koji sadrži kategoričke i numeričke vrijednosti. Korištene su dvije metode enkodiranja kategoričkih podataka: Binary i One Hot Encoding.
Za nadgledano učenje korišteni su Multi-Layer Perceptron (MLP) i Convolutional Neural Network (CNN) arhitekture neuronskih mreža, dok je za nenadgledano učenje korišten Deep Autoencoder (DAE). Rezultati ovih arhitektura zajedno sa različitim metodama enkodiranja dataseta pokazali su da je najbolja MLP arhitektura sa manjom dubinom (odnosno, manje skrivenim slojeva) sa One Hot enkodiranjem. Za optimalnu konfiguraciju (MLP2OH128H918) postigla se stopa smanjenja upozorenja (threshold = 0,1) od 35,16% pri hvatanju 91,79% slučajeva prevare (stopa pogrešne klasifikacije 8,21%).

Rad: Credit Card Fraud Detection using Deep Learning based on Auto-Encoder and Restricted Boltzmann Machine[7]

Autori *Apapan Pumsirirat, Liu Yan* su koristili tri dataseta za prikazivanje eksperimenta, od kojih je jedan i dataset koji će biti obrađen u ovom dokumentu. U ovom radu je fokus na nenadgledano duboko učenje sa Auto-Encoder (AE) algoritmom. Za implementaciju AE algoritma korištena je hiperbolična tangens funkcija *tanh* za dekodiranje i enkodiranje. Takodjer je korištena i backpropagacija.
Korištena su dva programa za implementaciju AE algoritma: Keras i H20. Za Keras metod je dizajnirano 6 skrivenih slojeva sa po 3 enkodera i dekodera, a svaki sloj je korištena *tanh* aktivacijska funkcija. Dalje su te metode korištene na sva tri dataseta radi evaluacije pogodnosti za svaki. Prva dva, Njemački i Australijski dataset su malo manjeg obima od trećeg, Evropskog, dataseta. Za Njemački i Australijski se dobio niži AUC, dok je za Evropski dobiven AUC od 0.9603 što pokazuje da je AE mnogo bolji za veće datasetove, jer postoji mnogo više podataka za treniranje.

2.2 Prednosti i mane analiziranih radova

Prednosti određenih analiziranih radova se ogledaju u tome što koriste isti dataset nad kojim će biti urađeno treniranje i testiranje modela u ovom radu. Također, analizirani radovi obrađuju metode dubokog učenja, ali i metode mašinskog učenja, što će biti korišteno i u ovom radu. Pored toga, korištene su i neke druge metode koje mogu pomoći u daljoj evaluaciji modela, a koje možemo iskoristiti i prilikom treniranja našeg modela. Jedan od nedostataka analiziranih radova je u nedostatku informacija vezanim za implementaciju modela, kao i za podešavanje parametara koji su doprinijeli povećanju stope tačnosti.

3. Skup podataka

U ovom poglavlju je dat pregled dataset-ova koji su vezani za detekciju prevara kreditnih kartica. Urađena je detaljna analiza dataset-a koji će biti korišten u ovom radu, opisane su metode predprocesiranja podataka, te je urađeno i samo predprocesiranje.

3.1 Pregled postojećih dataset-ova

S obzirom da skupovi podataka, vezani za problem koji se rješava, sadrže privatne podatke, na internetu je javno dostupan samo jedan dataset. Nekoliko referentnih radova, opisanih u poglavlju 2, koristili su neke druge, privatne datasetove, koji nisu javno dostupni, te će opis tih dataset-ova biti dat na osnovu opisa u radovima.

3.1.1 Dataset obezbijeđen od institucije za maloprodajno bankarstvo

Skup podataka korišten u analiziranom radu 2.1 *Deep Learning Detecting Fraud in Credit Card Transactions* [4] koristi skup podataka koji je obezbijedila finansijska institucija za maloprodajno bankarstvo, a sadrži gotovo 80 miliona transakcija prikupljenih u toku 8 mjeseci. Skup podataka je labeliran, gdje su lažne transakcije označene sa 1, a ispravne transakcije sa 0. Skup sadrži informacije o transakcijama i o računu korisnika, kao što su iznos transakcije, tip trgovca kod kojeg je napravljena transakcija, datum otvaranja računa i slično.
Dataset sadrži 99.86% legitimnih i 0.14% lažnih transakcija, što ovaj skup čini nebalansiranim skupom.

Metode korištene za predprocesiranje podataka

S obzirom da je u datasetu bilo dosta NA vrijednosti (vrijednosti koje nedostaju), za popunjavanje nedostajućih vrijednosti kategoričkih varijabli, korišten je *mode*. Umjesto nedostajućih vrijednosti, dodane su i *dummy* vrijednosti.
Kako bi se riješio problem nebalansiranosti, urađen je *undersampling* legitimnih transakcija na osnovu broja računa korisnika. Dataset je podijeljen na dva podskupa: podskup sa lažnim transakcijama i podskup sa legitimnim transakcijama, te je izvršeno izdvajanje jedinstvenih brojeva računa iz podskupa koji sadrži lažne transakcije. Nakon toga je urađeno nasumično uzorkovanje transakcija u podskupu legitimnih transakcija na osnovu izdvojenih brojeva računa korisnika. Omjer uzorkovanja iznosi 10:1 (legitimne transakcije:lažne transakcije).

3.1.2 Dataset obezbijeđen iz Španjolske organizacije za obradu plaćanja

Skup podataka korišten u analiziranom radu 2.1 *Evaluation of Deep Neural Networks for Reduction of Credit Card Fraud Alerts* [6] koristi skup podataka koji je obezbijedila Španjolska organizacija za obradu plaćanja. Dataset sadrži 446076 stvarnih upozorenja vezanih uz sumnjive transakcije kreditnim karticama. Upozorenja obuhvaćaju razdoblje od šest mjeseci. Skup podataka sadrži i numeričke i kategoričke podatke.

Dataset sadrži Label atribut koji sadrži labele gdje vrijednost 1 označava sve potvrđene slučajeve (klasa prevare), a 0 odbačene slučajeve (lažno pozitivna klasa).

Metode korištene za predprocesiranje podataka

Korištene su različite metode kodiranja:

- *Binarno kodiranje*
  Korišteno za kategoričke podatke. Svaka vrijednost indeksirana je pozitivnim cijelim brojem. Rezultirajući indeks je pretvoren u binarnu vrijednost. Dužina vektora je odabrana u odnosu nabroj bita koji je bio potreban da predstavi najveću binarnu vrijednost. Na slici 3.1 mozemo vidjeti primjer izvođenja binarnog kodiranja:

Label	# Records	Percentage (%)
B_1	0	000
B_2	1	001
B_3	2	010
B_4	3	011
B_5	4	100

Slika 3.1: Primjer binarnog kodiranja

- *One-Hot kodiranje*
  Korišteno za kategoričke podatke. Svaka vrijednost indeksirana je pozitivnim cijelim brojem. Dužina vektora jednaka je broju jedinstvenih vrijednosti atributa. Svi biti vektora postavljeni su na 0, osim za bite na poziciji koja predstavlja indeks, koji su postavljeni na 1. Na slici 3.2 mozemo vidjeti primjer izvođenja onehot kodiranja:

Feature value	Index	Encoded value
O_1	0	00001
O_2	1	00010
O_3	2	00100
O_4	3	01000
O_5	4	10000

Slika 3.2: Primjer one-hot kodiranja

- *Razvrstavanje (Binning)*
  Korišteno za numeričke podatke. Svaka vrijednost je dodijeljena u spremnik na temelju unaprijed definisanih pragova spremnika definisani iz raspona vrijednosti atributa. Na slici 3.3 mozemo vidjeti primjer izvođenja binning kodiranja:

Feature name	Bin thresholds
Score	[-∞, 10, 20, 30, 40, 50, 60, 70, 80, 90, ∞]
Day of month	[-∞, 10, 20, ∞]
Hour	[-∞, 4, 8, 12, 16, 20, ∞]
Amount	[-∞, 10, 100, 1000, 10000, 100000, ∞]

Slika 3.3: Primjer binning kodiranja

3.2 Analiza izabranog dataset-a

U ovom radu će biti korišten skup podataka dostupan na *Kaggle-u*: `https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud`. Skup sadrži podatke o transakcijama iz septembra 2013. godine, prikupljenih u Europi u toku 2 dana. Ukupna količina podataka iznosi 69MB, a ukupan broj transakcija iznosi 284807, od kojih su 492 transakcije klasifikovane kao lažne.

Zbog povjerljivosti informacija i zaštite identiteta korisnika, skup ne sadrži originalne karakteristike podataka. Na originalne, numeričke podatke primijenjena je PCA (*Principal Component Analysis*). PCA transformacija se koristi za smanjenje dimenzionalnosti podataka. Ona transformiše podatke velikih dimenzija u niže dimenzije zadržavajući što je moguće više informacija.

PCA transformacijom dobijeno je 28 glavnih komponenti, koje su označene sa V1, V2, ... V28. Pored ovih varijabli, skup sadrži varijable Vrijeme, Iznos i labelu klase (slika 3.4).

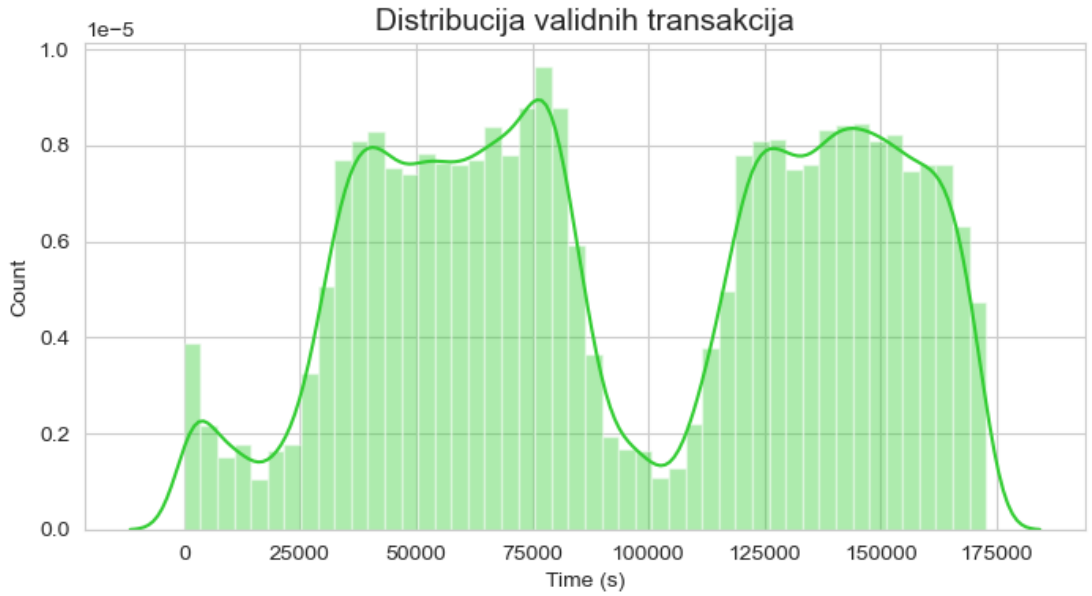
	Time	V1	V2	V3	V4	...	V27	V28	Amount	Class
0	0.0	-1.359807	-0.072781	2.536347	1.378155	...	0.133558	-0.021053	149.62	0
1	0.0	1.191857	0.266151	0.166480	0.448154	...	-0.008983	0.014724	2.69	0
2	1.0	-1.358354	-1.340163	1.773209	0.379780	...	-0.055353	-0.059752	378.66	0
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	...	0.062723	0.061458	123.50	0
4	2.0	-1.158233	0.877737	1.548718	0.403034	...	0.219422	0.215153	69.99	0

Slika 3.4: Varijable u dataset-u

Varijabla Class predstavlja labelu klase, koja ima vrijednost 1 u slučaju da je transakcija lažna, a vrijednost 0 u suprotnom.

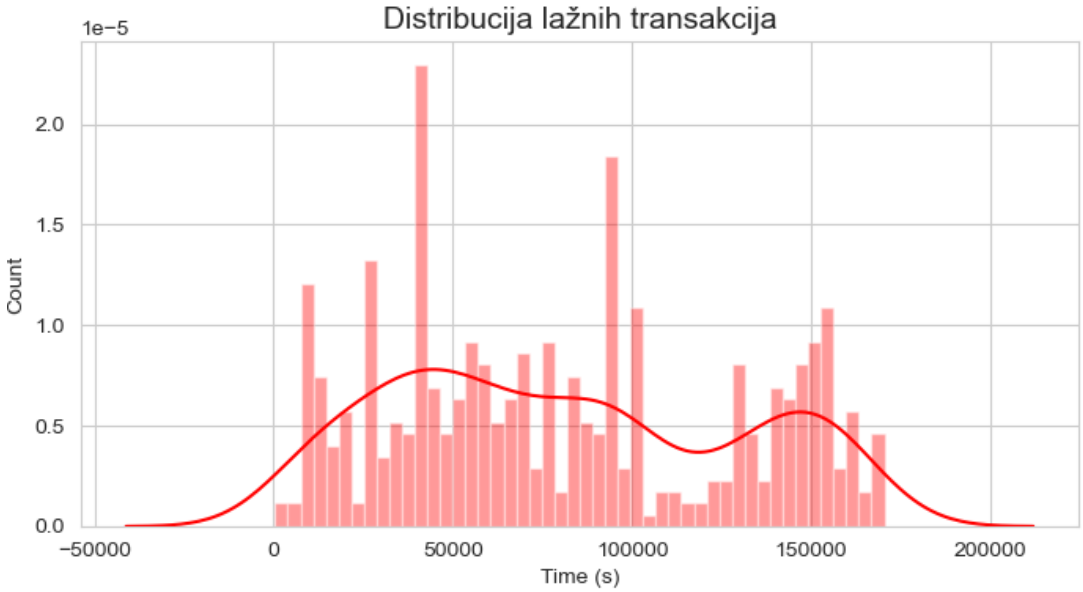
Varijabla Amount označava iznos transkacije. Uvid u to koje transakcije imaju veći iznos, možemo dobiti izračunavanjem srednje vrijednosti iznosa za lažne i za validne transakcije. Nakon izračunavanja prosječnih vrijednosti, zaključeno je da, u prosjeku, veći iznos imaju lažne transakcije.

Varijabla Vrijeme sadrži sekunde protekle između prve transakcije i svake transakcije. Na slici 3.5 prikazana je distribucija validnih transakcija u odnosu na varijablu vrijeme. S obzirom da su podaci prikupljani tokom 2 dana, odnosno 48 sati, maksimalna vrijednost varijable vrijeme na grafiku iznosi 172792 u sekundama, tj. približno 48 u satima, a minimalna vrijednost iznosi 0.



Slika 3.5: Raspodjela validnih transakcija po vremenu

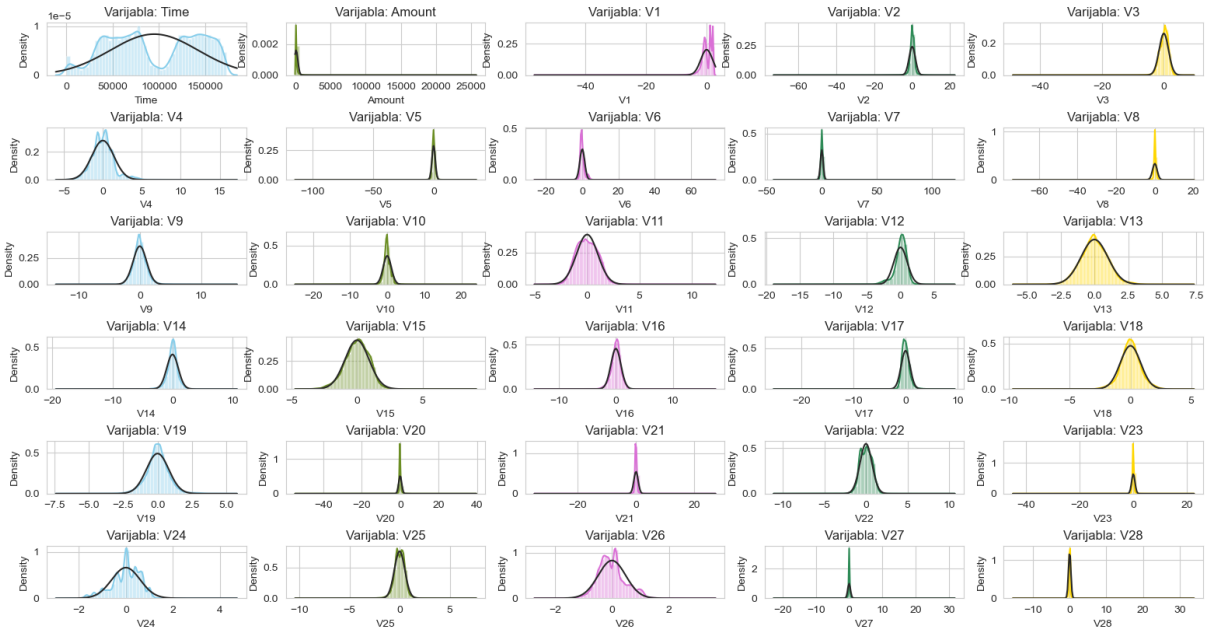
Sa grafika možemo pročitati da je manji broj validnih transakcija zabilježen u ranim jutarnjim i kasnim večernjim satima (na primjer: 100000s odgovara 3h ujutru), a da je najveći broj transakcija zabilježen tokom dana.
Najveći broj lažnih transakcija zabilježen je tokom kasnih večernjih sati, a dana u kojima su se desile lažne transakcije ima znatno manje (slika 3.6).



Slika 3.6: Raspodjela validnih transakcija po vremenu

3.2.1 Vizualizacija podataka

Prije početka predprocesiranja, urađena je vizualizacija podataka, čime je omogućeno bolje razumijevanje karakteristika varijabli. Na osnovu vizualizacije je zaključeno da karakteristike pojedinih varijabli nisu pravilno raspoređene, odnosno da ne prate Gaussovu raspodjelu, te da neke varijable imaju distribuciju nagnutu u jednu stranu (slika 3.7). Primjer varijable koja nema dobru raspodjelu karakteristika je Time. Varijable Amount i V1 imaju raspodjelu koja je nagnuta u jednu stranu (*skewed distribution)*, a za ostale varijable se može reći da imaju dobru raspodjelu karakteristika.

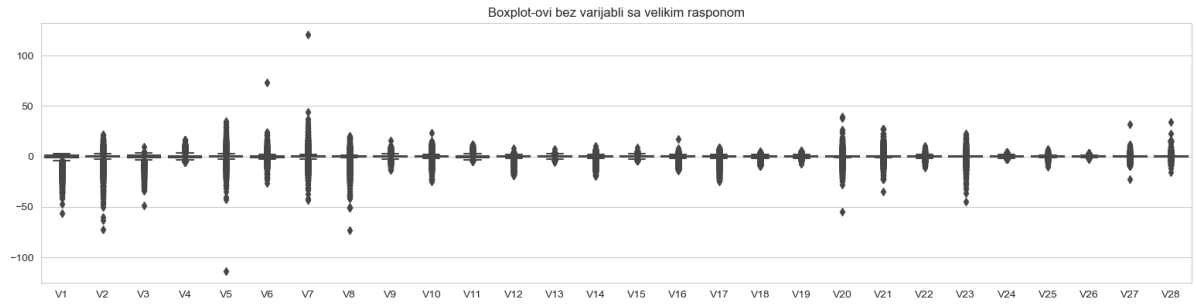


Slika 3.7: Raspodjele karakteristika varijabli u dataset-u

Osim raspodjele, još jedan bitan aspekt prilikom proučavanja karakteristika varijabli jeste raspon vrijednosti. Vrijednosti varijabli bi trebale imati što približniji opseg. Sa slike 3.8 je moguće zaključiti da varijable Time i Amount imaju dosta veće raspone vrijednosti i da su suzbile boxplot-ove ostalih varijabli.

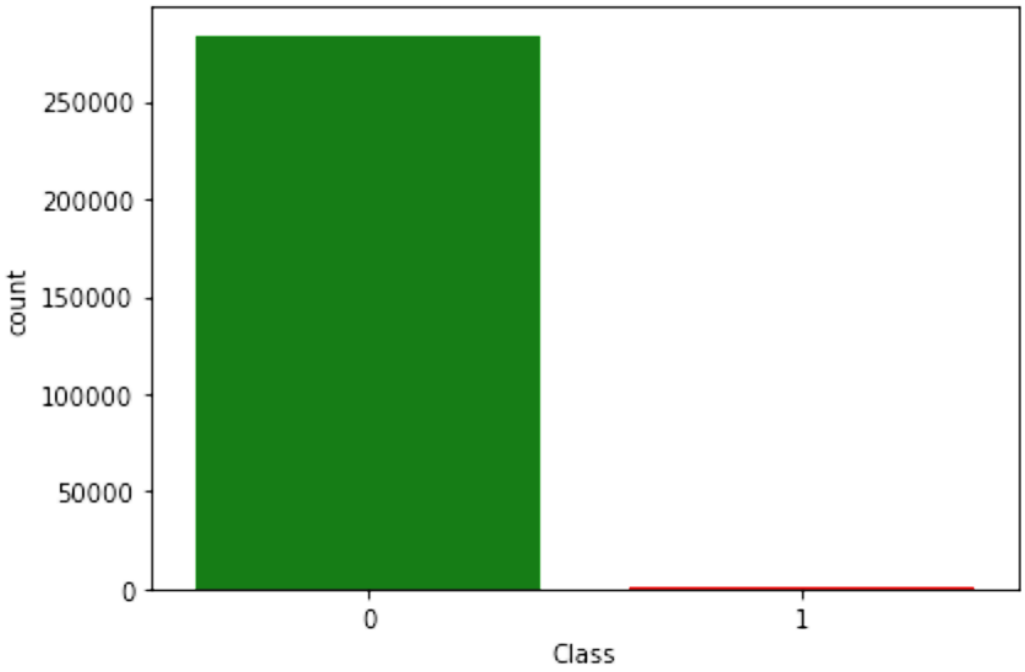
Slika 3.8: Prikaz boxplot-ova za sve varijable iz dataset-a

U svrhe pregleda vrijednosti ostalih varijabli, prikazani su i boxplot-ovi varijabli bez varijabli Time, Amount i Class, a rezultati su prikazani na slici 3.9. Sa prikazanih boxplot-ova moguće je vidjeti i da varijable V5, V6, V7 i Amount imaju nekoliko ekstremnih outlier-a.



Slika 3.9: Prikaz boxplot-ova bez varijabli sa velikim rasponom

Još jedan faktor koji igra bitnu ulogu prilikom treniranja modela jeste balansiranost skupa podataka. Kao što je spomenuto, podaci su u velikom mjeri neuravnoteženi, odnosno dataset nije balansiran, kao što je vidljivo na slici 3.10.



Slika 3.10: Prikaz nebalansiranosti dataset-a

Učenje iz neuravnoteženih skupova podataka je težak zadatak jer većina algoritama učenja nije dizajnirana da se nosi sa velikom razlikom između broja slučajeva koji pripadaju različitim klasama. Model je veoma teško trenirati nad nebalansiranim skupovima podataka jer ima tendenciju da se preoptereti za većinsku klasu, a nedovoljno za klasu manjina.

Sve navedene probleme, koji su uočeni prilikom vizualizacije podataka, potrebno je pokušati ukloniti kako bi modeli, koji će biti korišteni, dali što bolje predikcije za problem koji se rješava. Podatke je potrebno predprocesirati.

3.2.2 Podjela skupa podataka

Podaci će biti podijeljeni na skupinu za treniranje, validaciju i testiranje.
Skup za treniranje se koristi kako bi model naučio skrivene karakteristike ili obrasce u podacima. Ovaj skup treba sadržavati podatke sa različitim karakteristikama kako bi model mogao naučiti što više uzoraka koji se mogu pojaviti u podacima, kasnije. U ovom radu će skup za treniranje sadržavati 60% od ukupne količine podataka.

Skup podataka za validaciju, kako mu i samo ima kaže, koristi se za validiranje performansi modela u toku procesa treniranja. Informacije koje se dobijaju u toku procesa validacije omogućavaju podešavanje hiperparametara. Skup podataka za validaciju će sadržavati 15% podataka od ukupne količine podataka.

Nakon treniranja, podaci će biti testirani na testnom skupu podataka. Nakon testiranja, dobija se uvid u to koliko dobro model radi. Skup podataka za testiranje će, u ovom radu, sadržavati 25% podataka od ukupne količine podataka.

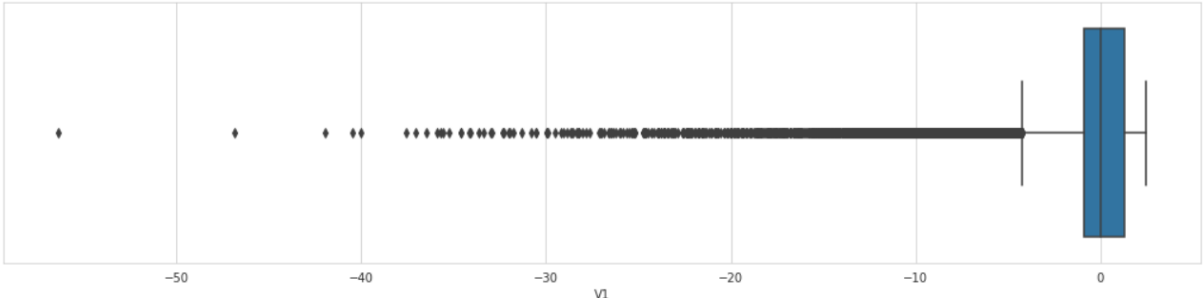
Dakle, omjer podjele iznosi: 60%:15%:25%.

3.2.3 Predprocesiranje podataka

Predprocesiranje podataka se, u ovom radu, sastoji od nekoliko koraka: detekcija i uklanjanje outlier-a, balansiranje skupa podataka i normalizacija (skaliranje) varijabli.

Detekcija i uklanjanje outlier-a

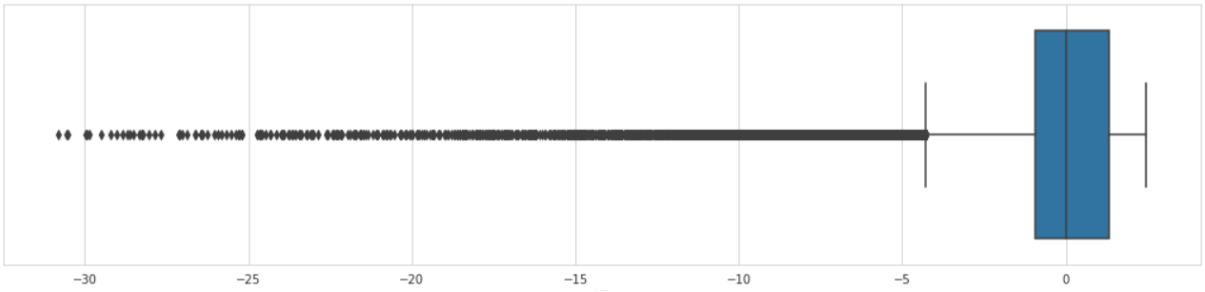
Postojanje outlier-a može dovesti do izobličenja tačnosti modela. Međutim, potrebno je izbjeći pretjerano gubljenje informacija, kako ne bi došlo do underfitting-a. Uklanjanje outlier-a je izvršeno nad čitavom skupu podataka, prije podjele podataka.
Prilikom vizualizacije podataka, uočeno je prisustvo ekstremnih outlier-a kod nekoliko varijabli: Amount, V5, V6 i V7, te ih je potrebno ukloniti. Kako ne bi došlo do uklanjanja karakteristika koje su označene kao lažne, izvršeno je iscrtavanje boxplot-ova i analiza dataset-a, na osnovu čega su uklonjene pojedine karakteristike.
U ovom odjeljku ce biti prikazan način detekcije i uklanjanja outlier-a samo za jednu varijablu iz dataset-a (V1), a isti postupak je primijenjen i na ostale varijable. Boxplot za varijablu V1 prikazan je na slici ispod.



Slika 3.11: Boxplot za varijablu V1

Na slici vidimo da postoji mnogo outlier-a. Uklonjeni su outlier-i atributa koji su manji od -31, a ustanovljeno je da među tim atributima nema onih koji pripadaju klasi označenoj kao 1 -

lažna. Nakon uklanjana outlier-a, dobijen je sljedeći boxplot.



Slika 3.12: Boxplot za varijablu V1 nakon uklanjanja outlier-a

Analogna analiza je urađena i za ostale varijable.
Nakon analize i uklanjanja outlier-a u preostalim varijabla, broj karakteristika klasificiranih kao validne iznosi 283842, a broj lažnih karakteristika je ostao isti.

Skaliranje podataka

Za skaliranje (standardizaciju) podataka, korištene su dvije metode: StandardScaler i PowerTransformer iz *sklearn.preprocessing* biblioteke.
PowerTransformer je korišten nad varijablama Amount i V1, s obzirom da je njihova distribucija nagnuta u jednu stranu, a PowerTransformer koristi logaritamske funkcije koje pomažu u minimiziranju asimetrije i mapiranje distribucije na normalnu, što je moguće bliže.

Nakon primjene PowerTransformer-a na Amount i V1, izvršeno je skaliranje podataka korištenjem StandardScaler-a.

Skaliranje podataka je izvršeno nakon podjele skupa podataka. Skaliranje trening skupa podataka, korištena je *fit_transform* metoda koja vrši prilagođavanje scaler-a podacima za trening, a nakon toga ih transformiše. Scaler se prilagođava samo podacima za trening, a ne testnim i validacijskim podacima, kako ne bi došlo do overfitting-a.
Nakon standardizacije i prilagođavanja scaler-a podacima za treniranje, scaler je iskorišten za transformaciju testnih i validacijskih podataka pomoću *transform* metode.

Balansiranje skupa podataka

Tehnike za učenje iz nebalansiranih podaka moguće je podijeliti na metode osjetljive na troškove i metode ponovnog uzorkovanja (*resampling*). U ovom radu će biti korištene tehnike ponovnog uzorkovanja. Ove tehnike se koriste kako bi se skup podataka balansirao prije treniranja modela. Predložene su brojne metode za ponovno uzorkovanje neuravnoteženih skupova podataka, a mogu se kategorizirati u tri glavne strategije: prekomjerno uzorkovanje (*oversampling*), nedovoljno uzorkovanje (*undersampling*) i hibridne strategije.[8]. Oversampling tehnike balansiraju skup podataka kreiranjem novih (sintetičkih) podataka za klasu manjina, a undersampling tehnike smanjuju broj uzoraka većinske klase, što poboljšava performanse modela u smislu brzine treniranja, jer se reducira veličina skupa podataka. Kombinacijom ove dvije tehnike, odnosno korištenjem hibridnih metoda, u mnogim slučajevima je dalo najbolje performanse modela. [8]

U ovom radu će biti korištena SMOTE metoda oversampling-a i hibridna metoda koja je kombinacija je SMOTE-a i *random undersampling*-a (RUC). SMOTE i SMOTE + RUC su u ovom radu implementirane pomoću *imblearn* Python biblioteke.

Navedene tehnike balansiranja će biti korištene samo za balansiranje skupa podataka za treniranje, s obzirom da uzorkovanje čitavog skupa podataka (prije podjele na test, train i val), dovodi do overfitting-a.

Implementacija **SMOTE**-a prikazana je u isječku koda ispod. Kako ne bi došlo do overfitting-a, parametar *sampling_strategy* je stavljen na vrijednost 0.6. Ovaj parametar označava proporciju uzorkovanja. Nakon izvršavanja prikazanog koda, broj uzoraka u klasi označenoj kao lažna iznosi 102185, dok je broj validnih uzoraka ostao isti (170308).

```
smote = imblearn.over_sampling.SMOTE(sampling_strategy=0.6, random_state=0)
train_smote_data, train_smote_data['Class'] = smote.fit_resample(train_data
    , train_data['Class'])
```

Implementacija **SMOTE + RUC** prikazana je u isječku koda ispod. Parametar *sampling_strategy = 0.5* govori da je omjer neuravnoteženosti 0.5, što znači da će oversampling kreirati uzorke sve dok broj uzoraka iz klase manjina ne dostigne vrijednost 170308/2 = 85154. Omjer neuravnoteženosti kod RUC je stavljen na 1, što znači da će se uzorci većinske klase uklanjati sve dok njihov broj ne dostigne broj uzoraka koji se nalazi u smote_ruc_dataset-u nakon primjene oversampling-a. Dakle, nakon pokretanja prikazanog koda, broj validnih uzoraka jednak je broju lažnih i broj iznosi 85154.

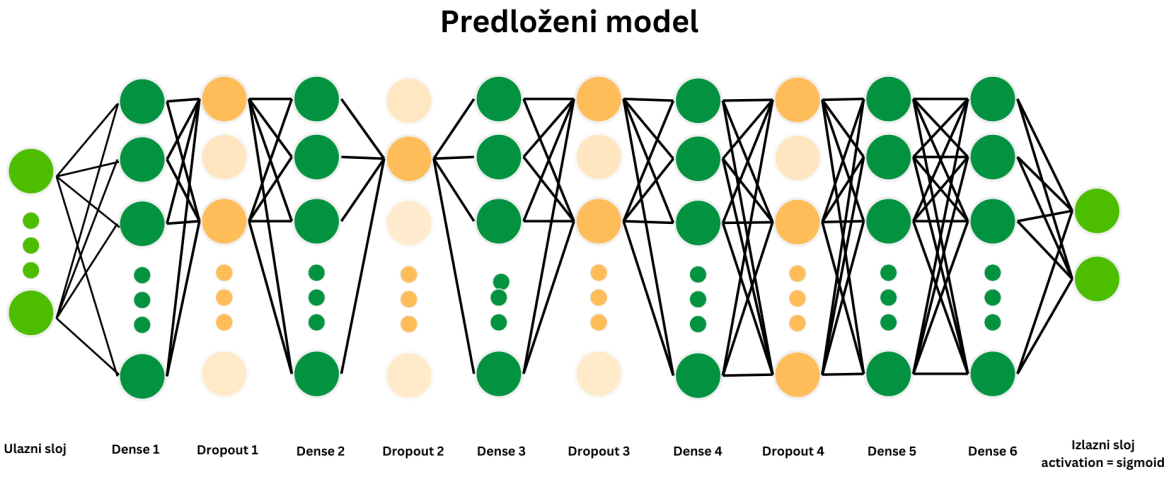
```
smote2 = imblearn.over_sampling.SMOTE(sampling_strategy=0.5, random_state
    =0)
smote_ruc_dataset, smote_ruc_dataset['Class'] = smote2.fit_resample(
    train_data, train_data['Class'])

ruc = imblearn.under_sampling.RandomUnderSampler(sampling_strategy=1.0,
    random_state=0)
train_smote_ruc_data, train_smote_ruc_data['Class'] = ruc.fit_resample(
    smote_ruc_dataset, smote_ruc_dataset['Class'])
```

4. Treniranje modela

U ovom poglavlju su opisane tri metode koje su korištene za treniranje modela i dat je detaljan uvid u svaku od metoda i korištenih parametara.

4.1 Duboka neuronska mreža

Predloženi model duboke neuronske mreže (slika 4.1) sadrži ulazni sloj, šest skrivenih *Dense* slojeva, izlazni sloj i četiri*Dropout* sloja. Dropout slojevi se nalaze između pojedinih Dense slojeva i nasumično postavljaju dio aktivacija prethodnog sloja na nulu tokom treniranja modela, u zavisnosti od vrijednosti *dropout rate*-a. Dropout slojevi se koriste kako ne bi došlo do overfitting-a.



Slika 4.1: Predloženi model duboke neuronske mreže

Kako bi opisana mreža dala što bolje rezultate, urađeno je tuniranje hiperparametara pomoću KerasTuner biblioteke. Hiperparametri su varijable koje se ne uče u toku procesa treniranja i oni su u tom procesu konstanti. Hiperparametri su veoma važni jer mogu značajno uticati na performanse modela. Na primjer, postavljanje previsoke stope učenja može dovesti do odstupanja modela i neuspjeha u učenju, dok postavljanje preniske stope može usporiti proces učenja. Slično tome, dodavanje previše ili premalo skrivenih jedinica ili slojeva također može utjecati na performanse modela.

Tuniranje hiperparmetara urađeno je na malom skupu podataka.
U predloženom modelu, urađen je tuning sljedećih hiperparametara: broj jedinica (*units*) u skrivenim Dense slojevima, aktivacijske funkcije u skrivenim slojevima, dropout rate u Dropout slojevima, stope učenja i optimizatora.

U isječku koda ispod, prikazana je implementacija modela sa tuniranjem hiperparametara. Za

tuniranje je korišten Hyperband. [1].

```
def build_model(hp):
  model = keras.Sequential()
  model.add(layers.Input(shape=(30,)))

  model.add(layers.Dense(
                units=hp.Int("units1", min_value=32, max_value=512, step
    =32),
                activation=hp.Choice('dense_activation1', values=['relu', '
    tanh', 'elu', 'selu'], default='relu')))
  model.add(layers.Dropout(hp.Float("dropout1", 0, 0.8, step=0.1, default
    =0.5)))

  model.add(layers.Dense(
                units=hp.Int("units2", min_value=32, max_value=512, step
    =32),
                activation=hp.Choice('dense_activation2', values=['relu', '
    tanh', 'elu', 'selu'], default='relu')))
  model.add(layers.Dropout(hp.Float("dropout2", 0, 0.8, step=0.1, default
    =0.5)))

  model.add(layers.Dense(
                units=hp.Int("units3", min_value=32, max_value=512, step
    =32),
                activation=hp.Choice('dense_activation3', values=['relu', '
    tanh', 'elu', 'selu'], default='relu')))
  model.add(layers.Dropout(hp.Float("dropout3", 0, 0.8, step=0.1, default
    =0.5)))


  model.add(layers.Dense(
                units=hp.Int("units4", min_value=32, max_value=512, step
    =32),
                activation=hp.Choice('dense_activation4', values=['relu', '
    tanh', 'elu', 'selu'], default='relu')))
  model.add(layers.Dropout(hp.Float("dropout4", 0, 0.8, step=0.1, default
    =0.5)))

  model.add(layers.Dense(
                units=hp.Int("units5", min_value=32, max_value=512, step
    =32),
                activation=hp.Choice('dense_activation5', values=['relu', '
    tanh', 'elu', 'selu'], default='relu')))

  model.add(layers.Dense(
                units=hp.Int("units6", min_value=32, max_value=512, step
    =32),
                activation=hp.Choice('dense_activation6', values=['relu', '
    tanh', 'elu', 'selu'], default='relu')))

  model.add(layers.Dense(1, activation="sigmoid"))

  hp_learning_rate = hp.Choice('learning_rate', values=[1e-2, 1e-3, 1e-4, 1
    e-5, 1e-6])
```

[1]Čitav kod projekta je dostupan na: Implementacija modela

```
37    optimizer = hp.Choice("optimizer", ["adam", "adamax", "rmsprop", "
       adadelta", "sgd"])
38
39    if optimizer == 'rmsprop':
40        optimizer = keras.optimizers.RMSprop(learning_rate=hp_learning_rate)
41    elif optimizer == 'adam':
42        optimizer = keras.optimizers.Adam(learning_rate=hp_learning_rate)
43    elif optimizer == 'adamax':
44        optimizer = keras.optimizers.Adamax(learning_rate=hp_learning_rate)
45    elif optimizer == 'adadelta':
46        optimizer = keras.optimizers.Adamax(learning_rate=hp_learning_rate)
47    elif optimizer == 'sgd':
48        optimizer = keras.optimizers.Adamax(learning_rate=hp_learning_rate)
49    else:
50        raise
51    model.compile(optimizer=optimizer, loss=keras.losses.BinaryCrossentropy()
       , metrics=METRICS)
52
53    return model
54
55 build_model(keras_tuner.HyperParameters())
```

Program 4.1: Implementacija modela sa tuniranjem hiperparametara

Kreirana su dva modela prikazane, duboke neuronske mreže: *smote* i *smote + ruc* model. Modeli su istrenirani nad 100 epoha, a vrijednost *batch_size*-a iznosi 128. Prilikom treniranja je korištena ***early stopping*** callback funkcija. Funkcija je definisana tako da zaustavi treniranje modela ukoliko ne dođe do smanjivanja validacijskog gubitka nakon 20 epoha, kako ne bi došlo do overfitting-a.

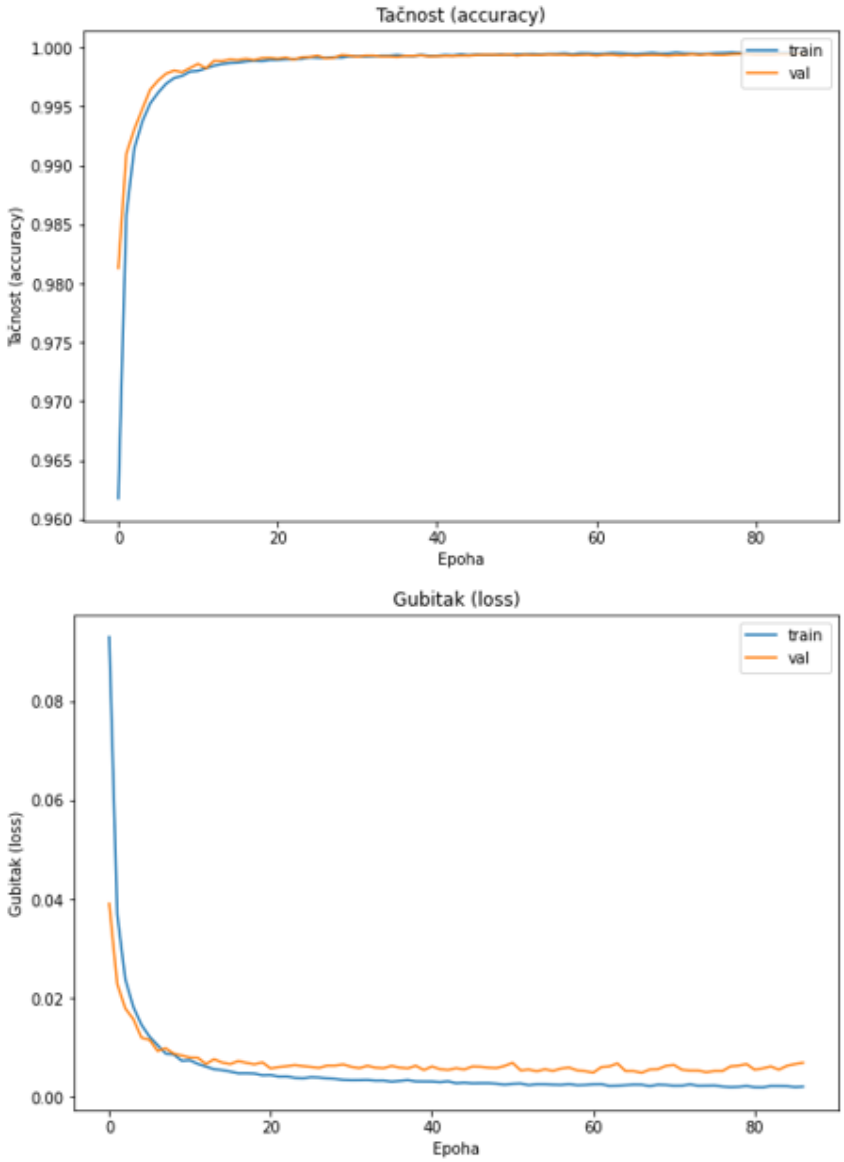
Za pregled rezultata treniranja korištene su metrike: loss i accuray. U nastavku su dati rezultati spomenuta dva modela DNN-a.

4.1.1 Rezultati treniranja SMOTE modela

Nakon tuniranja, model je istreniran nad skupom za treniranje. Tuniranje i treniranje modela nad malim skupom podataka sa predloženim hiperparametrima urađeno je nekoliko puta, a ona kombinacija parametara koja je dala najbolje rezultate, korištena za treniranje i testiranje modela. Hiperparametri koji su se pokazali najboljim su:

- Aktivacijske funkcije: relu, tahn i elu
- Droput: 0.4, 0.5 i 0.2
- Stopa učenja: 0.0001
- Optimizator: Adam

Grafici za *accuracy* i *loss* nastali kao rezultat treniranja modela sa iznad prikazanom kombinacijom parametara vidljivi su na slici 4.2. Možemo zaključiti da su rezultati za *accuracy* zadovoljavajući, s obzirom da krive na grafiku rastu do tačke stabilnosti i prate jedna drugu. Rezultati za *loss* su veoma dobri s obzirom da je vrijednost gubitka veoma mala kako za trening, tako i za validacijski skup podataka. Međutim, možemo vidjeti da na samom kraju grafika za gubitak, validacijski gubitak počinje polahko rasti, što ukazuje na to da sa povećanjem broja epoha, vjerovatno ne bismo dobili bolje rezultate, jer je moguća pojava overfitting-a, kao i to da smo ovaj model mogli trenirati i na malo manjem broju epoha.



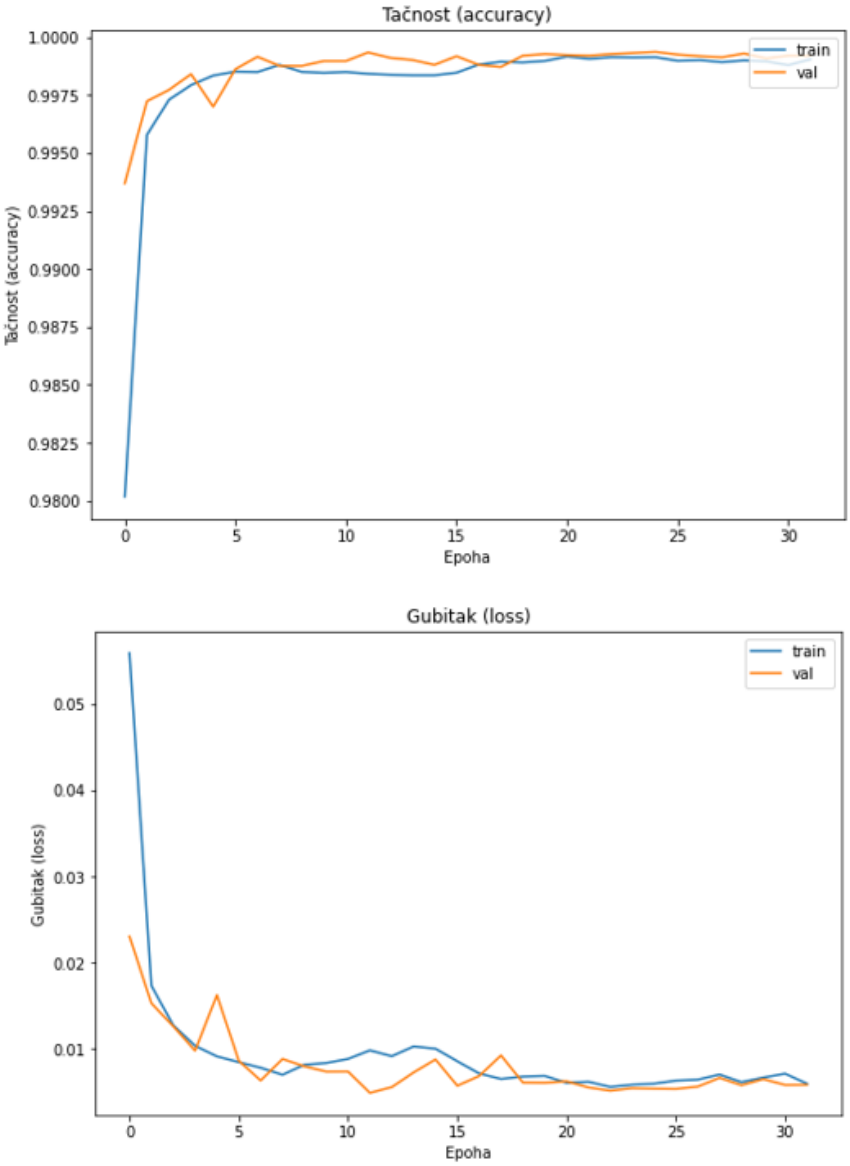
Slika 4.2: Grafici tačnosti i gubitka kroz epohe za SMOTE model

4.1.2 Rezultati treniranja SMOTE + RUC modela

Kombinacije parametara koje su dale najbolje rezultate za skup podataka nad kojim je primjenjen smote i random undersampling su:

- Aktivacijske funkcije: relu i tahn
- Droput: 0.6, 0.5 i 0.2
- Stopa učenja: 0.001
- Optimizator: Rmsprop

Rezultati treniranja modela nad SMOTE + RUC skupom prikazani su na slici 4.3.
Za razliku od prethodnog modela, ovaj model bi mogao dati bolje rezultate ukoliko bismo ga trenirali duži broj epoha. Međutim i ovi grafici su sasvim zadovoljavajući, a vidimo da je, kao i u prethodnom slučaju, postignuta dobra tačnost i mali gubitak.



Slika 4.3: Grafici tačnosti i gubitka kroz epohe za SMOTE + RUC model

Na osnovu navedenih rezultata treniranja, možemo očekivati dobre rezultate testiranja, ali pošto je u pitanju klasifikacija nad nebalansiranim skupom podataka, rješenja ipak mogu varirati. Posebnu pažnju je potrebno obratiti na broj klasa koje su ispravno klasifikovane kao lažne, s obzirom da one predstavljaju klasu manjina.

4.2 Random Forest Klasifikator

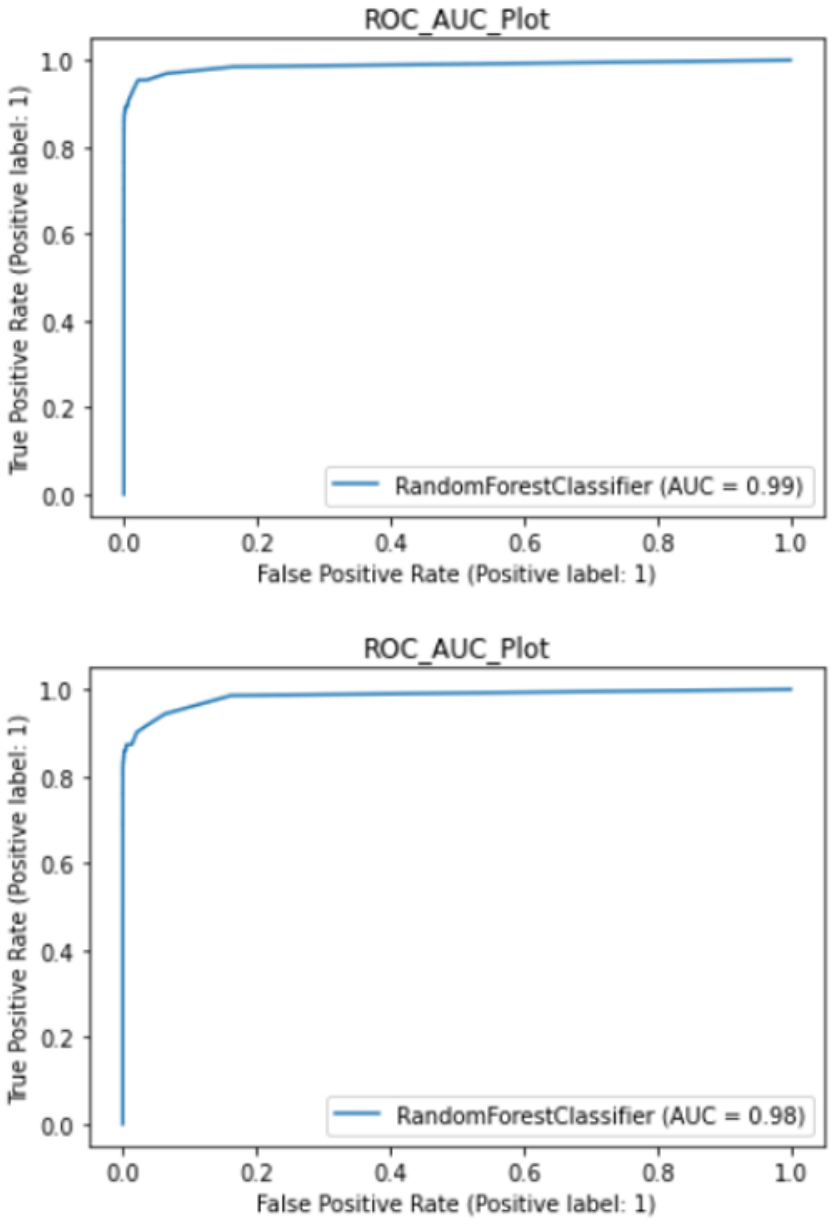
Random Forest klasifikator je vrsta metode učenja po grupi, koja kombinira više odlučujućih stabala da bi dala preciznije i otpornije predviđanje. Radi tako što trenira više odlučujućih stabala na različitim podskupovima podataka, a zatim uzima prosjek predviđanja koje svako stablo daje. Ovo pomaže da se smanji pretjerano prilagođavanje i poveća ukupna tačnost modela. Random Forest se često koristi u različitim primjenama kao što su klasifikacija slika, obrada prirodnog jezika i bioinformatičke nauke.

4.2.1 Rezultati treniranja SMOTE modela i predikcija modela

U isječku koda ispod prikazana je definicija modela i predikcija istog:

```
 randomForestModel = RandomForestClassifier(n_jobs=NO_JOBS,
                                random_state=RANDOM_STATE,
                                criterion=RFC_METRIC,
                                n_estimators=NUM_ESTIMATORS,
                                verbose=False)
rf_fit1 = randomForestModel.fit(x_train_smote,y_train_smote.values)
rfp1 = rf_fit1.predict(x_val)
rfp2 = rf_fit1.predict(x_test)

```

Rezultati treniranja modela nad SMOTE skupom prikazani su na slici 4.4.

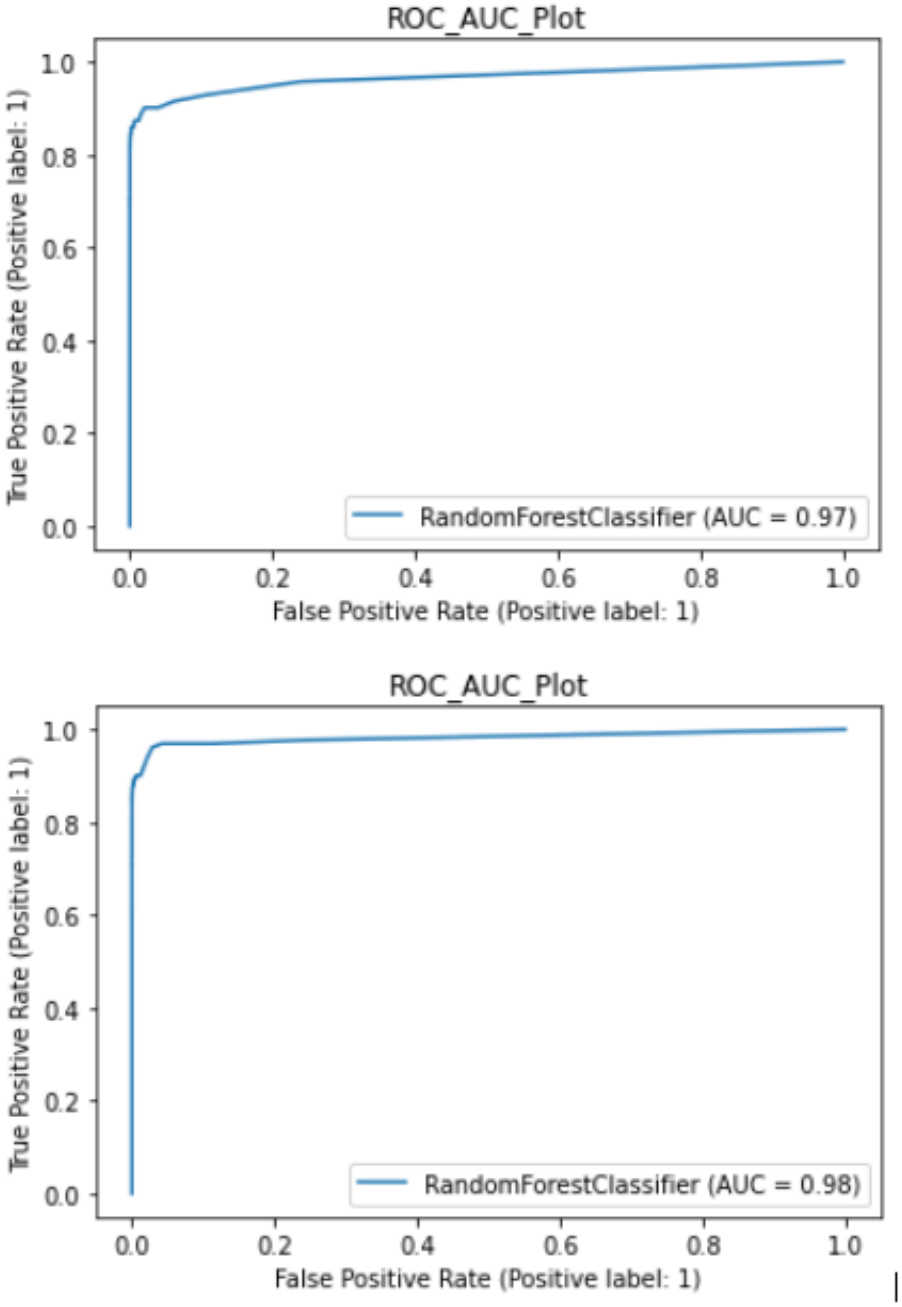


Slika 4.4: Grafici tačnosti za testni i validacijski skup

Za validacijski skup dobiven je AUC 0.8802 dok za testni skup AUC iznosi 0.9007.

4.2.2 Rezultati treniranja SMOTE + RUC modela i predikcija modela

Za potrebe ovog modela je korišten isti kod kao u prethodnom poglavlju samo se koriste drugi dijelovi dataseta.



Slika 4.5: Grafici tačnosti za validacijski i testni skup

Koristeći ovaj model dobiven je malo bolji AUC za validacijski i testni skup i oni iznose 0.8943 i 0.9236 respektivno.

4.3 Logistička Regresija

Logistička regresija je jedan od široko korištenih klasifikacijskih algoritama u mašinskom učenju. Logistička regresija je statistička metoda za kategorizaciju podataka u klase. Kao i drugi regresijski modeli (linearna regresija), koristi se u prediktivnoj analizi. Medjutim, logistička regresija je naprednija od linearne regresije. Razlog za to je što linearna regresija ne može klasificirati podatke koji su široko rasprostranjeni u datom prostoru.

Što se tiče hiperparametara, korišteni su sljedeći za model logističke regresije:

- Jačina regularizacije (C): Ovaj parametar kontroliše inverznu jačinu regularizacije. Manja vrijednost C primjenjuje više regularizacije, što može spriječiti prenaučenost, ali također može uzrokovati podnaučenost. Za model smo koristili vrijednost **10**.
- Kazna (penalty): Ovaj parametar određuje vrstu regularizacije koja će se primijeniti. L1 regularizacija, poznata i kao Lasso, dodaje kazneni izraz u funkciju troška koji je proporcionalan apsolutnoj vrijednosti koeficijenata. L2 regularizacija, poznata i kao Ridge, dodaje kazneni izraz u funkciju troška koji je proporcionalan kvadratu koeficijenata. Za ovaj model korištena je **L2 regularizacija**.
- Random state: Koristi se prilikom inicijalizacije modela. To je generator pseudoslučajnih brojeva koji se koristi za slučajno uzorkovanje. Ovdje je odabrana vrijednost **0**, da bi dobili iste rezultate na svakom pokretanju koda.

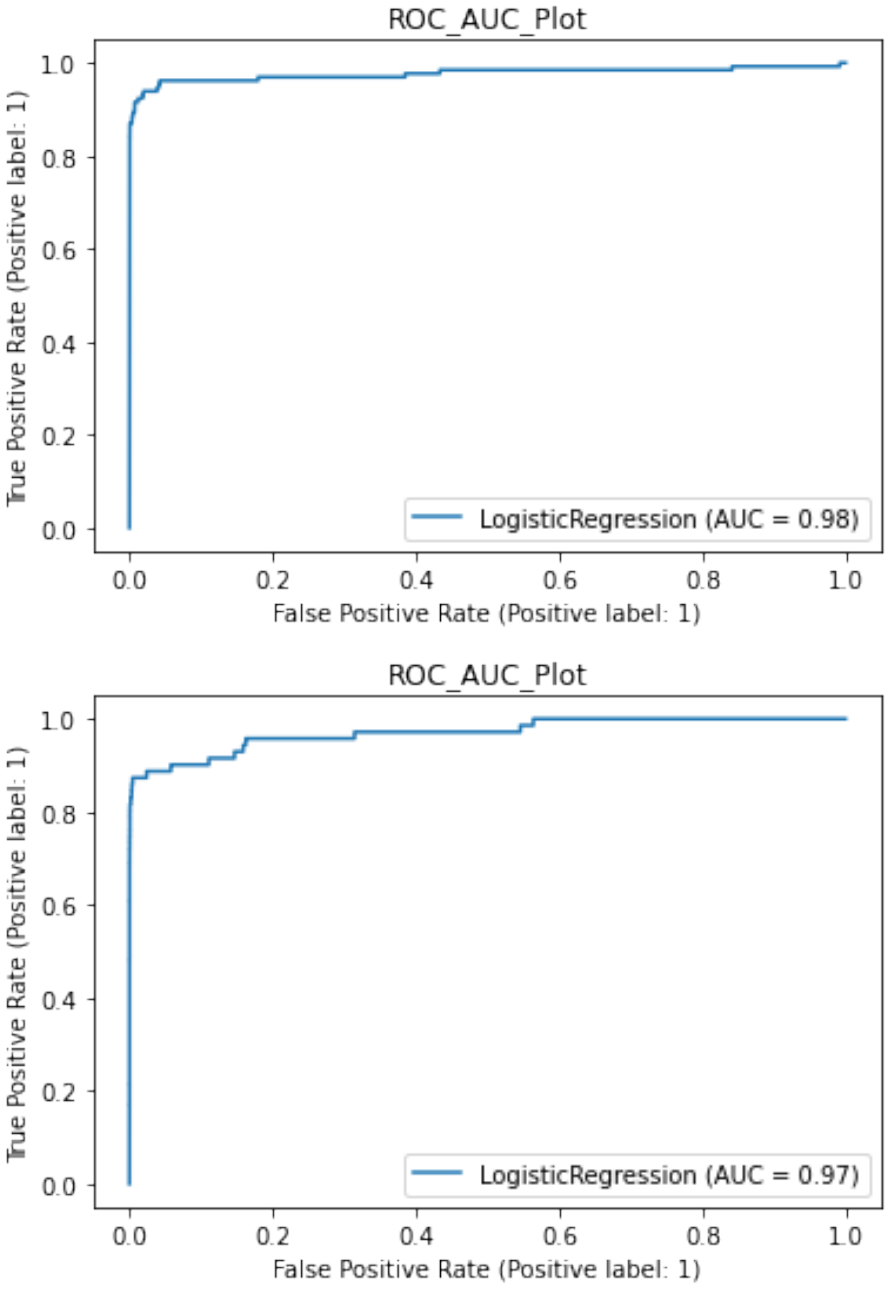
4.3.1 Rezultati treniranja SMOTE modela

U isječku koda ispod je prikazano kreiranje klasifikatora i funkcija za treniranje modela, čime su kreirane dvije ROC krive, jedna za testni a jedna za validacijski skup podataka:

```
1
2 classifier_lr = LogisticRegression(random_state = 0,C=10,penalty= 'l2')
3
4 def model(classifier,x_train,y_train,x_test,y_test):
5
6     classifier.fit(x_train,y_train)
7     prediction = classifier.predict(x_test)
8     cv = RepeatedStratifiedKFold(n_splits = 10,n_repeats = 3,random_state =
      1)
9     print("Cross Validation Score : ",'{0:.2%}'.format(cross_val_score(
     classifier,x_train,y_train,cv = cv,scoring = 'roc_auc').mean()))
10    print("ROC_AUC Score : ",'{0:.2%}'.format(roc_auc_score(y_test,
     prediction)))
11    plot_roc_curve(classifier, x_test,y_test)
12    plt.title('ROC_AUC_Plot')
13    plt.show()
14
15 model(classifier_lr,x_train_smote,y_train_smote,x_test,y_test)
16
17 model(classifier_lr,x_train_smote,y_train_smote,x_val,y_val)
```

Program 4.2: Treniranje modela logističke regresije

Nakon pokretanja metode model za testni i validacijski skup dobili smo sljedeće ROC krive:



Slika 4.6: Grafici tačnosti za validacijski i testni skup

Za testni skup smo dobili malo bolji rezultat, gdje je AUC vrijednost 0.9533, dok smo za validacijski skup dobili AUC vrijednost 0.9283.

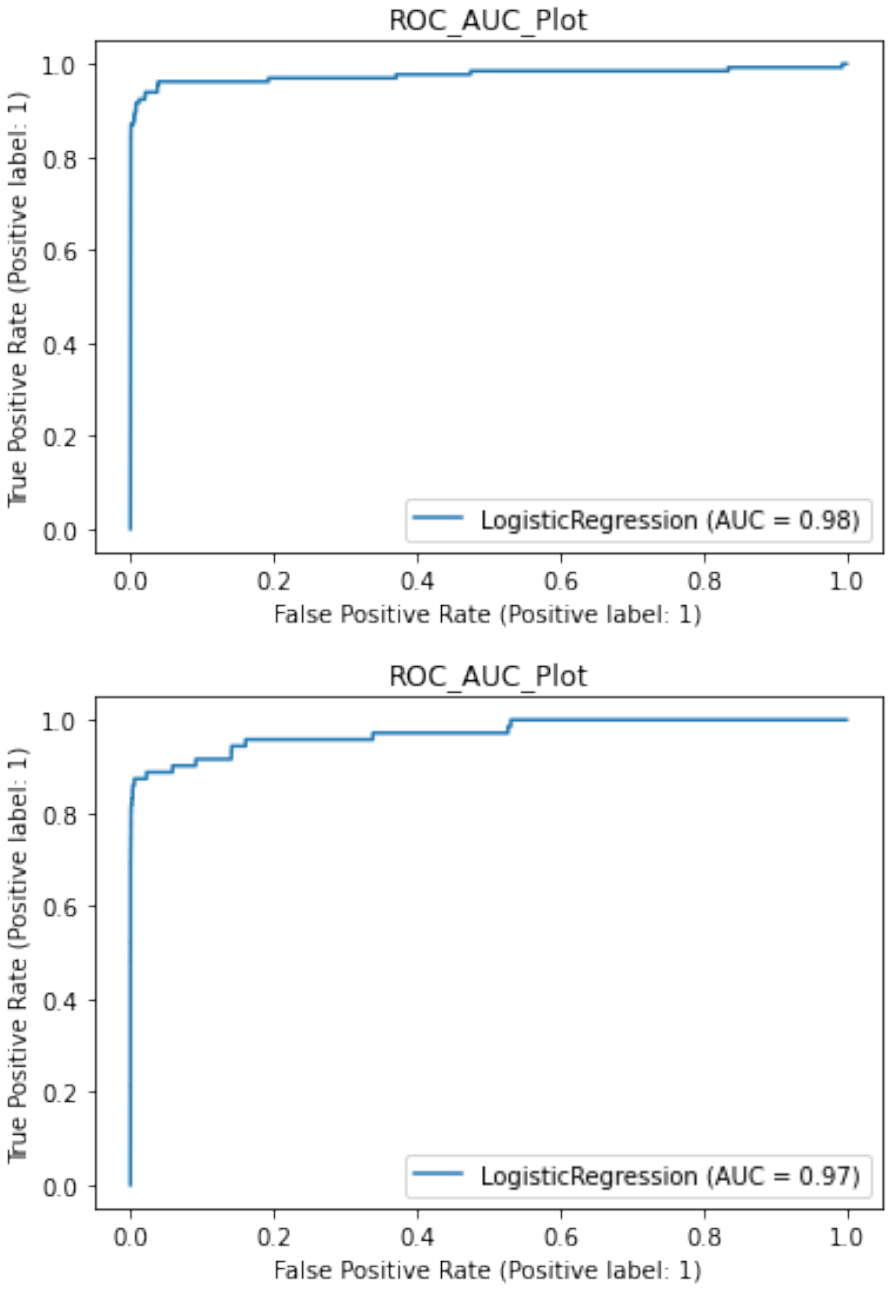
4.3.2 Rezultati treniranja SMOTE + RUC modela

Za SMOTE + RUC model se pozivala metoda model iz prethodnog potpoglavlja, ali sa datasetovima koji odgovaraju ovom modelu:

```
1
2 model(classifier_lr,x_train_smote_ruc,y_train_smote_ruc,x_test,y_test)
3
4 model(classifier_lr,x_train_smote_ruc,y_train_smote_ruc,x_val,y_val)
```

Program 4.3: Treniranje SMOTE + RUC modela logističke regresije

Nakon pokretanja metode model za testni i validacijski skup dobili smo sljedeće ROC krive:



Slika 4.7: Grafici tačnosti za validacijski i testni skup

Kod ovog modela smo za testni skup dobili AUC vrijednost 0.9548, dok smo za validacijski skup dobili AUC vrijednost 0.9293.

5. Testiranje modela

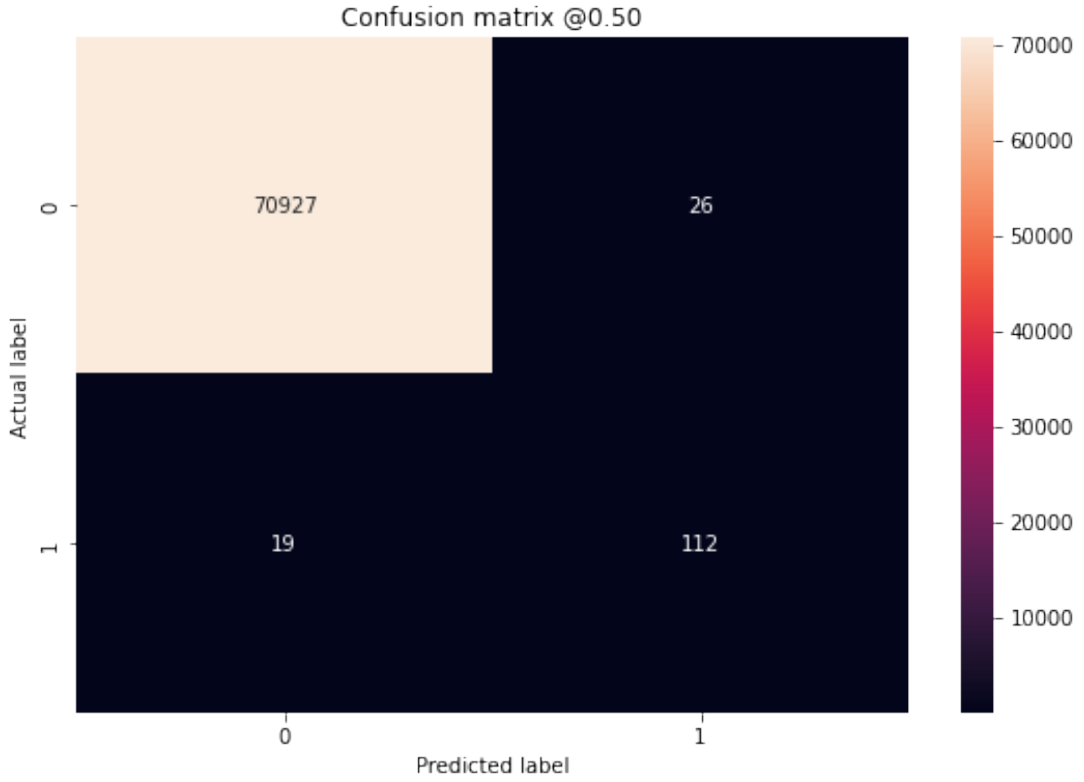
U ovom poglavlju su opisani rezultati treniranja modela i dat je opis metrika korištenih za evaluaciju modela.

5.1 Testiranja modela duboke neuronske mreže

Modeli su istrenirani nad do sada neviđenom skupu podataka, a koji sadrži 70953 podataka koji su označeni kao validni i 131 podatak označen kao lažan.
Rezultate testiranja modela je najlakše predstaviti preko pripadajućih konfuzijskih matrica. Konfuzijska matrica je tabela koja sadrži 4 različite kombinacije predviđenih i stvarnih vrijednosti: True Positive, True Negative, False Positive, False Negative.

5.1.1 Rezultati testiranja SMOTE modela

SMOTE model je dao tačnost od 99.93% nad testnim skupom podataka, a gubitak iznosi 0.0045. Ovo su odlični rezultati, ali s obzirom da postoji velika nebalansiranost u podacima, bolji uvid u rezultate treniranja možemo dobiti pomoću konfuzijske matrice (slika 5.1).



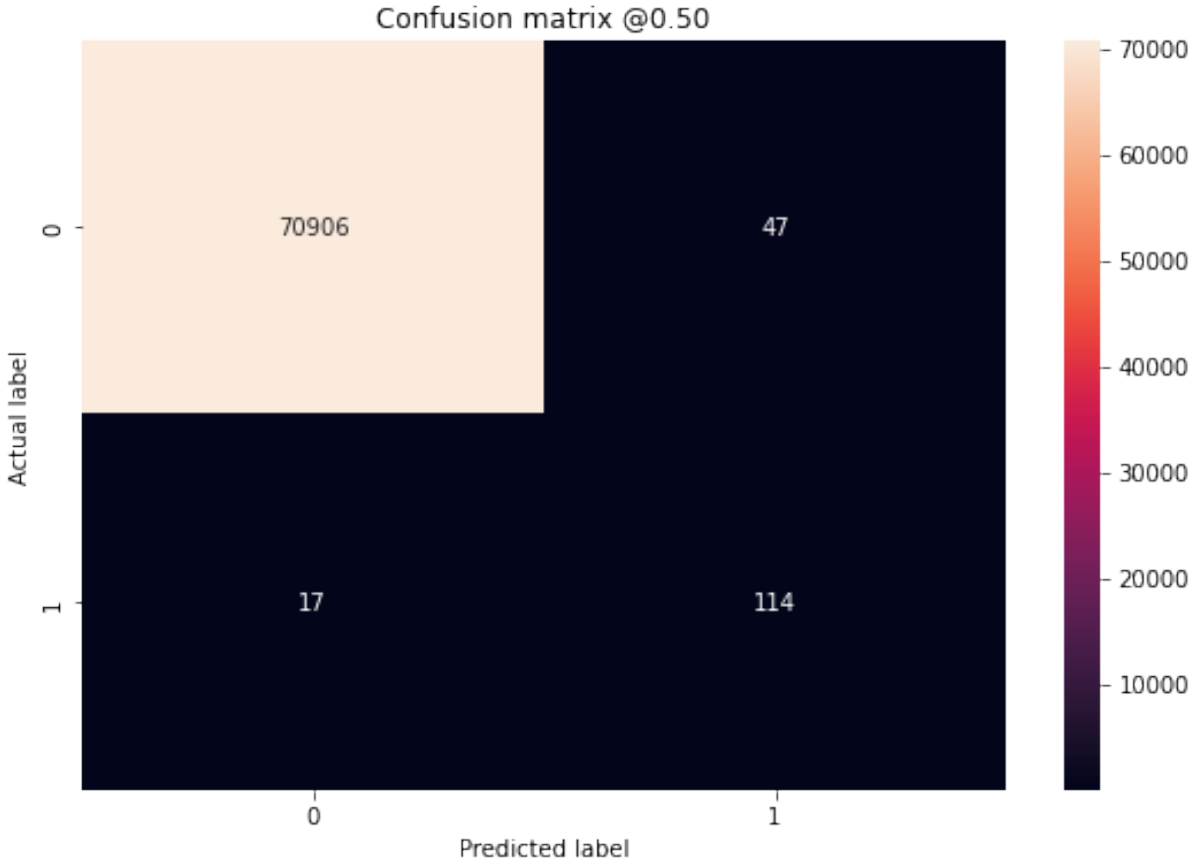
Slika 5.1: Rezultati treniranja model DNN-a nad SMOTE skupom podataka

Sa konfuzijske matrice možemo pročitati da je model ispravno predvidio gotovo sve validne transakcije, a od 131 lažnu transakciju, ispravno je predvidio njih 112, što je otprilike 85% od

ukupnog broja lažnih transakcija. 19 preostalih lažnih transakcija su predviđene kao validne i to su False Positive (fp) vrijednosti.
Dakle, iako su postignu odlični rezultati za tačnost i gubitak ovog modela, 19 lažnih transakcija je klasifikovano kao validne, što znači da su potrebna poboljšanja ovog modela, kako bi se broj smanjio na minimum.

5.1.2 Rezultati testiranja SMOTE + RUC modela DNN-a

Rezultati testiranja drugog modela prikazani su na konfuzijskoj matrici na slici 5.2. Ovaj model je dao napredak kada je u pitanju klasifikacija lažnih transakcija, gdje je ispravno klasifikovao 114, a neispravno 17 lažnih transakcija. Iako je ovo sasvim mala promjena kada su u pitanje FP vrijednosti, za razmatrani problem je i ova mala promjena od velikog značaja.
Za razliku od smote modela, ovaj model je napravio više grešaka prilikom klasifikacije validnih transakcija, s obzirom da je korištena random undersampling metoda.



Slika 5.2: Rezultati testiranja SMOTE + RUC modela DNN-a

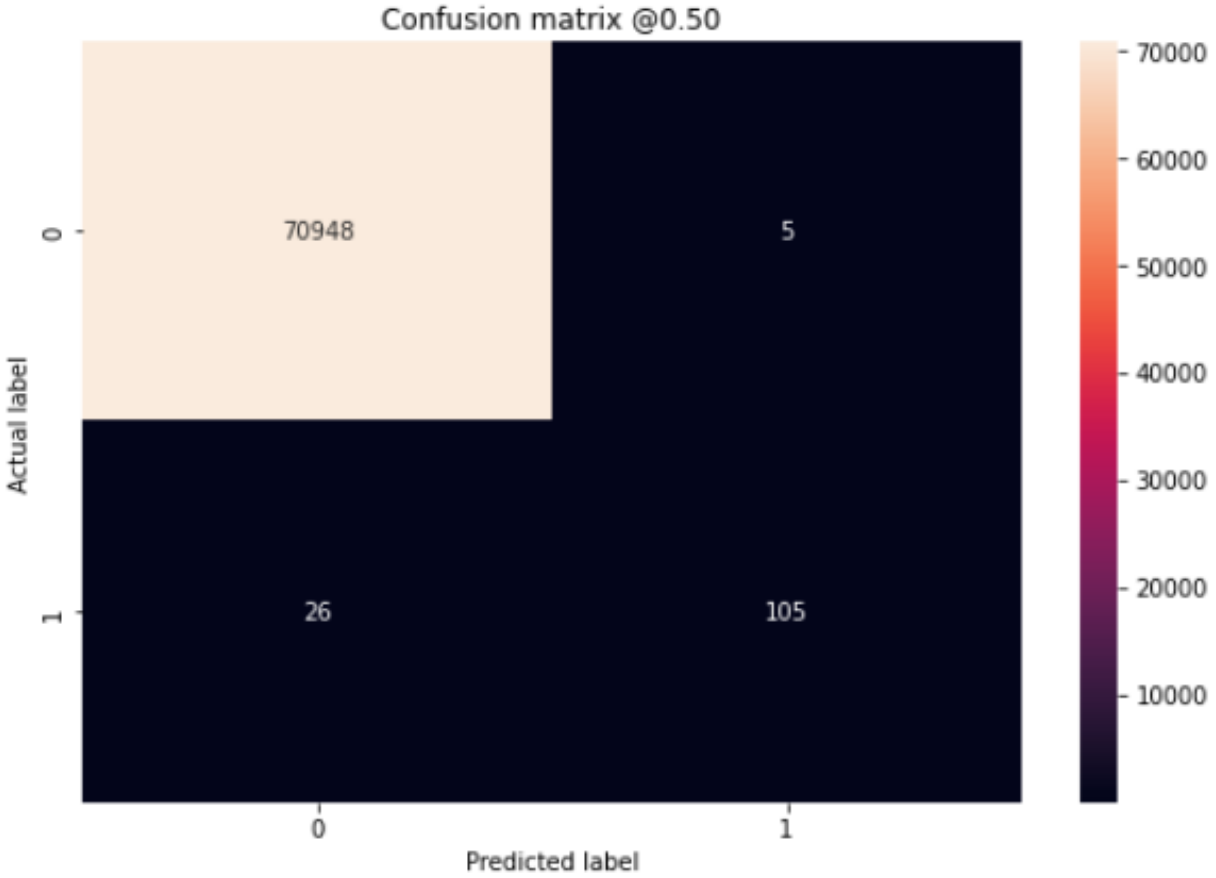
Kao i za smote model i smote + ruc model zahtjeva dodatna poboljšanja kako bi dao što bolje rezultate.

5.2 Testiranje Random Forest klasifikatora

Kao i u poglavlju 5.1 model je treniran nad datasetom koji sadrži 70953 podatka koji su označeni kao validni i 131 podatak označen kao lažan.

5.2.1 Rezultati testiranja SMOTE modela

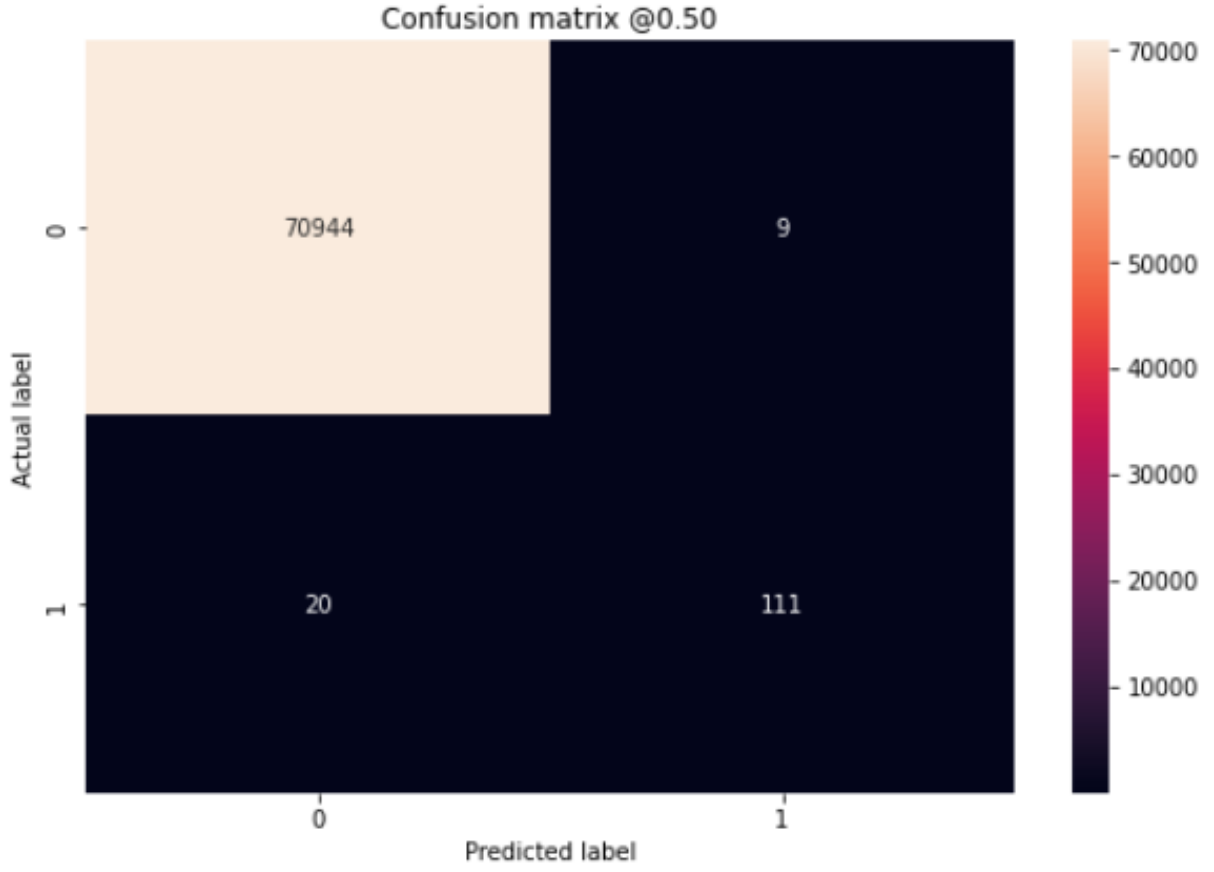
Na konfuzijskog matrici vidimo da je model ispravno kategorisao 105 transakcija od ukupno 131 lažnih što je 80.15%. Ostalih 26 je kategorisao kao false positive vrijednosti.



Slika 5.3: Rezultati testiranja SMOTE Random Forest klasifikatora

5.2.2 Rezultati testiranja SMOTE + RUC modela

Ovdje vidimo poboljšane rezultate modela, kategorisao je uspješno 111 od 131 lažnih transakcija dok je preostalih 20 kategorisao kao false positive. Uočavamo da su potrebna poboljšanja ovog modela uprkos dobrim rezultatima.



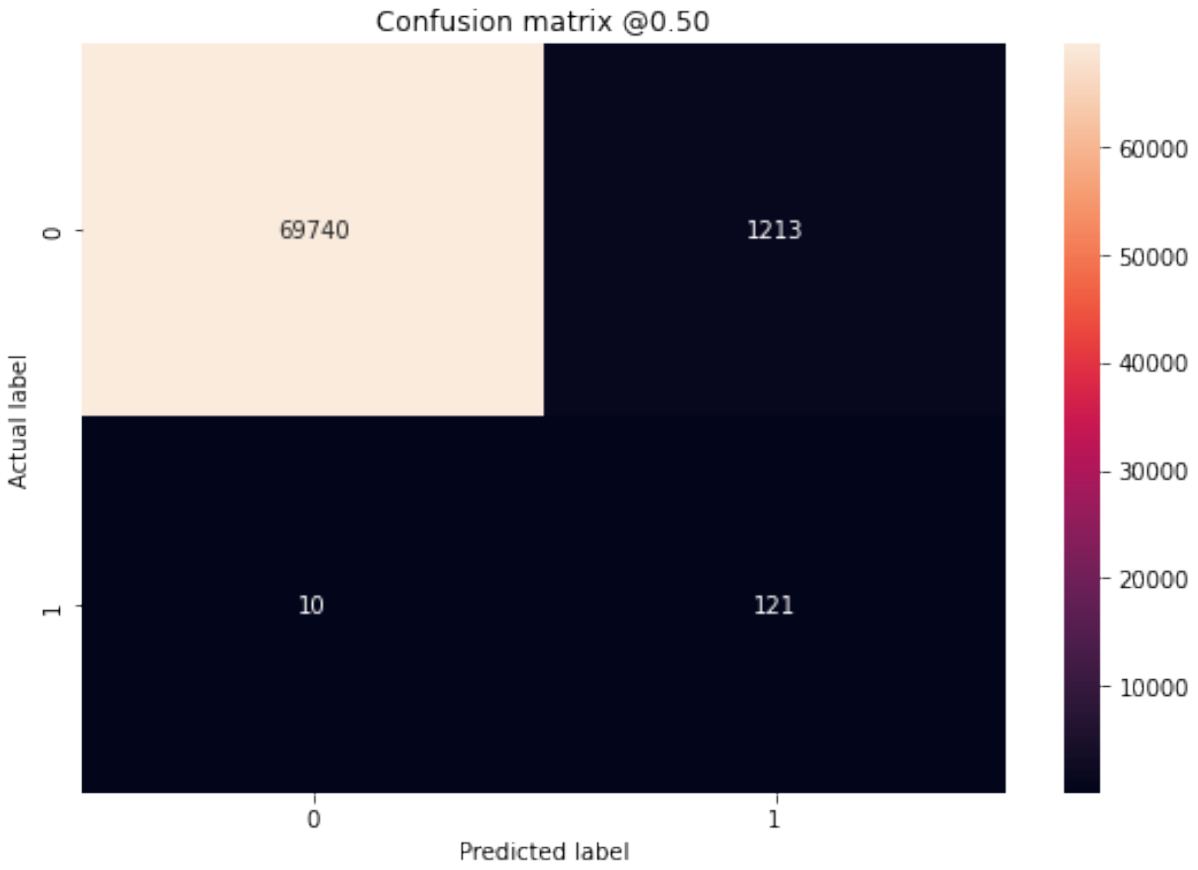
Slika 5.4: Rezultati testiranja SMOTE + RUC Random Forest klasifikatora

5.3 Testiranje modela logističke regresije

Rezultate testiranja modela logističke regresije takodjer će se prikazati putem konfuzijske matrice, koja sadrži 4 različite kombinacije predvidjenih i stvarnih vrijednosti: True Positive, True Negative, False Positive i False Negative.

5.3.1 Rezultati testiranja SMOTE modela

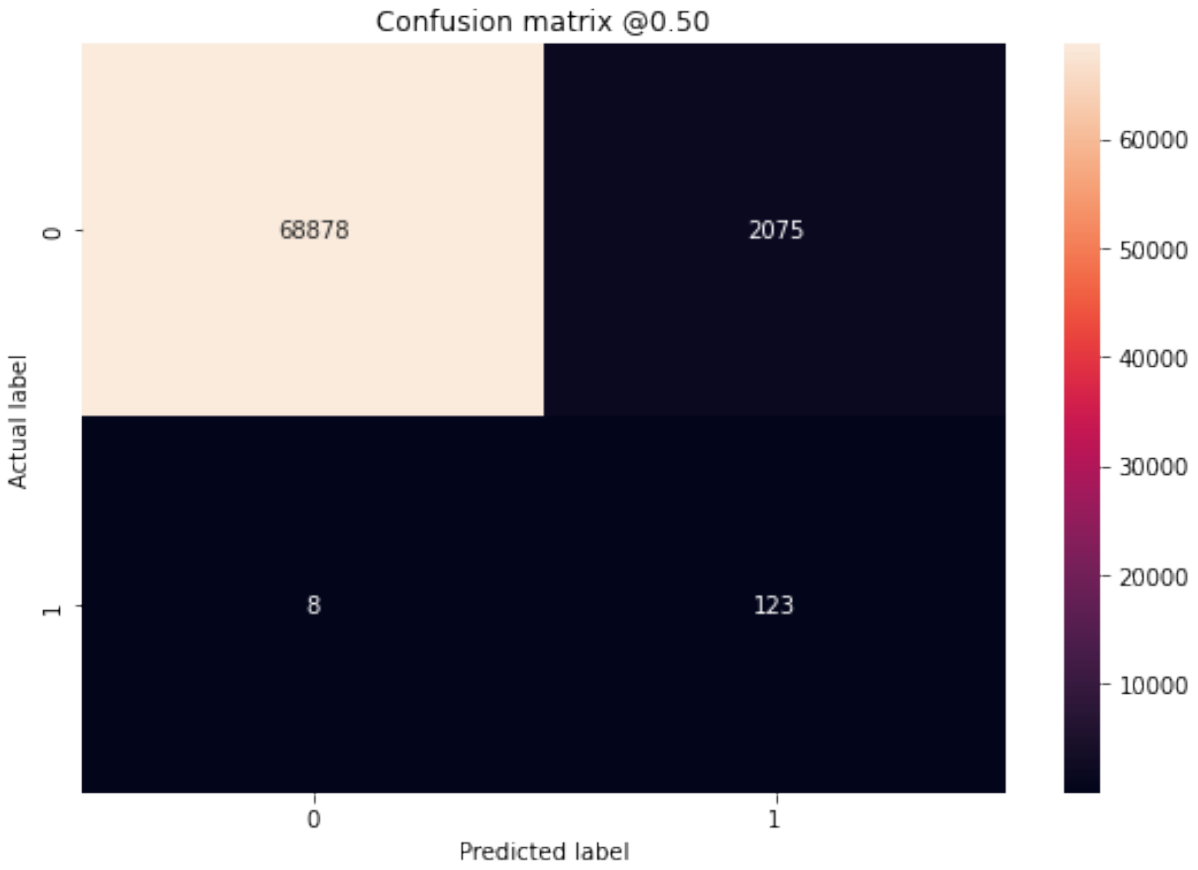
SMOTE model logističke regresije je dao tačnost 98.85% nad testnim skupom podataka. Kada gledamo konfuzijsku matricu za ovaj model, vidimo da nad testnim skupom podataka imamo 121 True Negative ili ispravno predvidjenih vrijednosti od ukupne 131 lažne transakcije, dok je preostalih 10 pogrešno predvidjeno kao validno, tj. False Positive. Ovo je u odnosu na SMOTE model za duboku neuronsku mrežu bolji rezultat, iako je tačnost manja nego u tom modelu. Medjutim, za validne transakcije imamo znatno više pogrešno klasificiranih instanci, gdje je 1213 validnih vrijednosti klasificirano kao lažno (False Negative).



Slika 5.5: Konfuzijska matrica za SMOTE model logističke regresije

5.3.2 Rezultati testiranja SMOTE + RUC modela

SMOTE + RUC model logističke regresije je dao tačnost 98.85% nad testnim skupom podataka, kao i za SMOTE model obradjen u prethodnom potpoglavlju. Konfuzijska matrica za ovaj model nam pokazuje da nad testnim skupom podataka imamo 123 True Negative ili ispravno predvidjenih vrijednosti od ukupne 131 lažne transakcije, dok je preostalih 8 pogrešno predvidjeno kao validno, tj. False Positive. Ovdje imamo bolji rezultat nego u SMOTE + RUC modelu za duboku neuronsku mrežu što se tiče klasifikacije lažnih transakcija, ali znatno lošiji rezultat za validne transakcije, gdje je 2075 validnih transakcija klasifikovano kao lažno (False Negative).



Slika 5.6: Konfuzijska matrica za SMOTE + RUC model logističke regresije

6. Osvrt na rezultate

6.1 Osvrt na rezultate predloženog modela DNN

Iako su postignuti zadovoljavajući rezultati u ovom radu, ostalo je još prostora za poboljšanja modela duboke neuronske mreže, a koja autori rada nisu mogli uraditi zbog nedostatka vremenskih i kompjuterskih resursa. Predloženi model duboke neuronske mreže je dao dobre rezultate kada su u pitanju tačnost i gubitak, a dodatna poboljšanja bi zahtijevala smanjenje broja lažnih transakcija koje su klasifikovane kao validne (FP). Iako je određeni broj validnih transakcija klasifikovano kao lažne, ovaj broj je zanemariv ukoliko uporedimo sam broj validnih transakcija u dataset-u.

Dodatna poboljšanja

Dodatna poboljšanja mogu uključivati sljedeće:

- Drugačiji pristup predprocesiranju podataka - uklanjanje preostalih outlier-a ili uklanjanje manjeg broja outlier-a
- Korištenje drugog scaler-a za skaliranje podataka
- Specificiranje težina klasa (*class weights*) umjesto balansiranja skupa podataka
- Korištenje modela autoenkodera za generisanje klasa manjina
- Dodatna i detaljnija eksperimentisanja sa hiperparametrima, brojem epoha za treniranje, brojem slojeva duboke neuronske mreže ili batch size-om

Poređenje rezultata sa SOTA

Ukoliko predloženi model uporedimo sa modelima iz referentnih radova - poglavlje 2, sa stanovišta tačnosti, predloženi model je dao mnogo bolje rezultate. Ono što je zajedničko sa predloženim modelom i modelima iz referentnih radova, koji su dali uvid u hiperparametre, jeste upravo izbor hiperparametara i to da je tačnost povećana, a gubitak smanjen sa povećanjem broja epoha i slojeva neuronske mreže.
Analizom rezultata iz tri analizirana rada (2.1) u poglavlju 2 koja koriste isti dataset kao i ovaj rad, a u kojim je urađen uvid u rezultate putem konfuzijske matrice, možemo zaključiti da je predloženi model neuronske mreže iz ovog rada dao bolje rezultate kada je u pitanju broj lažno pozitivnih (FP) i lažno negativnih (FN).

Odlučujući faktori koji su mogli dovesti do boljih rezultata u ovom radu u odnosu na SOTA su sljedeći:

- Detaljna analiza skupa podataka i njihovo predprocesiranje
- Tuniranje gotovo svih hiperparametara sa Keras Tunerom
- Korištenje dublje neuronske mreže sa 6 skrivenih slojeva zajedno sa Dropout slojevima

Na osnovu prethodno napisanog, iako su uvijek moguća dodatna poboljšanja i eksperimentisanja, možemo zaključiti da su rezultati ovog rada veoma zadovoljavajući i da su itekako uporedivi sa rezultatima iz analiziranih radova.

6.2 Osvrt na rezultate Random Forest klasifikatora

Iako je Random Forest klasifikator dao dobre rezultate (84.7% ispravno kategorisao neispravne transakcije), predloženi model DNN se pokazao boljim gdje tačnost iznosi 99.93% i ispravno je kategorisao 87.02% neispravnih transakcija.

6.3 Osvrt na rezultate modela logističke regresije

U odnosu na analizirani rad [5], gdje je za logističku regresiju dobivena stopa tačnosti od 99.88% za ovaj model imamo malo slabiju stopu tačnosti od 98.85%. Bitno je naglasiti da se i u referentnom radu koristio isti dataset kao i za trenutni rad.
Međutim, ako gledamo konfuzijsku matricu za referentni rad i poredimo rezultate, vidljivo je da su rezultati za trenutni model logističke regresije bolji za pojedine vrijednosti u odnosu na model iz referentnog rada. Rezultati konfuzijske matrice za referentni rad su prikazani na slici 6.1:

Confusion matrix:

Predicted	False	True	__all__
Actual			
False	99477	30	99507
True	76	100	176
__all__	99553	130	99683

Slika 6.1: Konfuzijska matrica za model logističke regresije referentnog rada

Ukupan broj lažnih transakcija u testnom skupu podataka u modelu iz spomenutog rada iznosi 176. Vidimo znatno lošiju klasifikaciju za lažne transakcije i to da 76 od 176 lažnih transakcija je klasificirano kao validno (False Positive). Za trenutni model logističke regresije imamo znatno manji procenat false positive vrijednosti u odnosu na ukupni broj lažnih transakcija. Međutim, u trenutnom modelu logističke regresije imamo lošije predikcije za validne transakcije u odnosu na prikazani referentni model logističke regresije, gdje su skoro sve validne transakcije ispravno klasificirane.

U odnosu na predloženi model DNN, tačnost je manja, ali je klasifikacija lažnih transakcija bolja kod modela logističke regresije. Međutim, klasifikacija validnih transakcija je dosta bolja kod predloženog modela DNN.

7. Zaključak

U tabeli na slici 7.1 je dat sažetak rezultata za False Positives i False Negatives vrijednosti. Kada su u pitanju False Positives vrijednosti (neispravna klasifikacija lažnih transakcija), model logističke regresije je dao najbolje rezultate. Međutim, ovaj model je neispravno klasifikovao dosta validnih transakcija. Random Forest klasifikator se pokazao najboljim za klasifikaciju validnih transakcija. Predloženi model DNN je, gledajući zajedno i False Positives i False Negatives vrijednosti, dao najbolje rezultate.

	False positives			**False Negatives**		
	Predloženi model DNN	Random forest	Logistička regresija	Predloženi model DNN	Random forest	Logistička regresija
SMOTE	19	26	**10**	26	**5**	1213
SMOTE + RUC	17	20	**8**	47	**9**	2075

Slika 7.1: Sažetak rezultata

Reference

[1] TJ, H. Credit Card Fraud: It's Still a Thing (and as Big as Ever), dostupno na: https://www.fico.com/blogs/credit-card-fraud-its-still-thing-and-big-ever

[2] Kriesel, D., "A brief introduction to neural networks, 2007", URL http://www. dkriesel. com, 2015.

[3] Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P.-E., He-Guelton, L., Caelen, O., "Sequence classification for credit-card fraud detection", Expert Systems with Applications, Vol. 100, 2018, str. 234–245.

[4] Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., Beling, P., "Deep learning detecting fraud in credit card transactions", in 2018 Systems and Information Engineering Design Symposium (SIEDS). IEEE, 2018, str. 129–134.

[5] Bhavya, L., Sasidhar Reddy, V., Anjali Mohan, U., Karishma, S., "Credit card fraud detection using classification, unsupervised, neural networks models", International Journal of Engineering Research, 2020.

[6] Carrasco, R. S. M., Sicilia-Urbán, M.-Á., "Evaluation of deep neural networks for reduction of credit card fraud alerts", IEEE Access, Vol. 8, 2020, str. 186 421–186 432.

[7] Pumsirirat, A., Liu, Y., "Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine", International Journal of advanced computer science and applications, Vol. 9, No. 1, 2018.

[8] Le Borgne, Y.-A., Siblini, W., Lebichot, B., Bontempi, G., Reproducible Machine Learning for Credit Card Fraud Detection - Practical Handbook. Université Libre de Bruxelles, 2022, dostupno na: https://github.com/Fraud-Detection-Handbook/fraud-detection-handbook

[9] Dornadula, V. N., Geetha, S., "Credit card fraud detection using machine learning algorithms", Procedia computer science, Vol. 165, 2019, str. 631–641.

[10] Inscribe. Credit Card Fraud Detection: Everything You Need to Know, dostupno na: https://www.inscribe.ai/fraud-detection/credit-fraud-detection

[11] BEXGBoost. How to Differentiate Between Scaling, Normalization, and Log Transformations, dostupno na: https://towardsdatascience.com/how-to-differentiate-between-scaling-normalization-and-log-transformations-69873d365a94

[12] ProjectPro, "Credit Card Fraud Detection Project using Machine Learning", 2022, dostupno na: https://www.projectpro.io/article/credit-card-fraud-detection-project-with-source-code-in-python/568

[13] Mahesh, K. P., Afrouz, S. A., Areeckal, A. S., “Detection of fraudulent credit card transactions: A comparative analysis of data sampling and classification techniques”, in Journal of Physics: Conference Series, Vol. 2161, No. 1. IOP Publishing, 2022, str. 012072.