Matematički fakutet, Univerzitet u Beogradu

Računarska inteligencija, Septembar 2023

Detekcija prevare korišćenjem autoenkodera

Autori: Marija Marković 60/2019
Dragana Zdravković 309/2019

Profesor: Aleksandar Kartelj

Asistenti: Stefan Kapunac
Denis Aličić

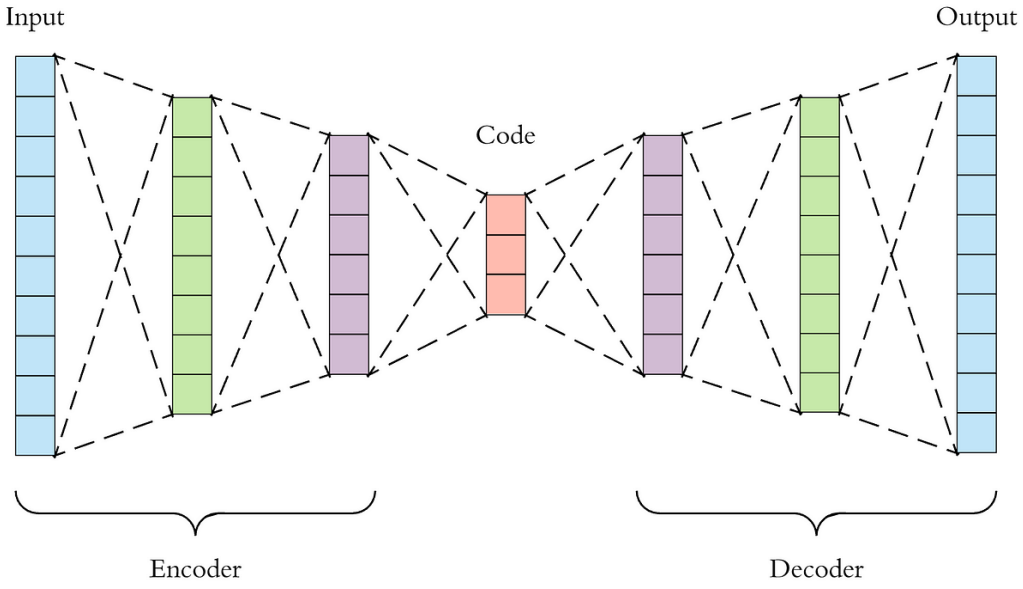
Sadržaj

1 Uvod

U ovom radu cilj je predstaviti mogućnosti korišćenja autoenkodera za potrebe detekcije prevare. U ovom slučaju, biće predstavljeno otkrivanje zloupotrebe kreditnih kartica od strane trećih lica, a u svrhu zaštite korisnika banke. Korišćen je skup podataka koji se može pronaći na linku.

2 O metodi

Teorijska osnova algoritma otkrivanja prevare koji ćemo ovde izvesti leži u otkrivanju anomalija. Detekcija anomalija podrazumeva uočavanje šablona koji nisu u skladu sa očekivanim ponašanjem. Upravo anomalije, odnosno neobične karakteristike transakcija prevare biće iskorišćene za razlikovanje zloupotreba od regularnih transakcija. Naša neuronska mreža (autoenkoder) biće trenirana isključivo na regularnim transakcijama, sa namerom da se pri testiranju uoče neobično velike greške u predviđanju kod instanci zloupotrebe.



Slika 1: Autoenkoder

Autoenkoder je neuronska mreža koja se trenira da dobijeni ulaz približno kopira na izlaz. Ovaj metod pripada oblasti nenadgledanog mašinskog učenja i može biti koristan u kompresiji podataka i smanjenju njihove dimenzionalnosti, zato što omogućava rekonstrukciju ulaza na osnovu kompresovane verzije. Uprkos navedenim karakteristikama koje ih izdvajaju od „klasičnih" neuronskih mreža, autoenkoderi se mogu posmatrati kao bilo koja neuronska mreža sa propagacijom unapred.
U našem slučaju, nenagledano učenje nam odgovara zbog male količine podataka o zloupotrebama, koji bi nam predstavljali problem u slučaju nadgledanog učenja.

3 Pregled radova na ovu temu

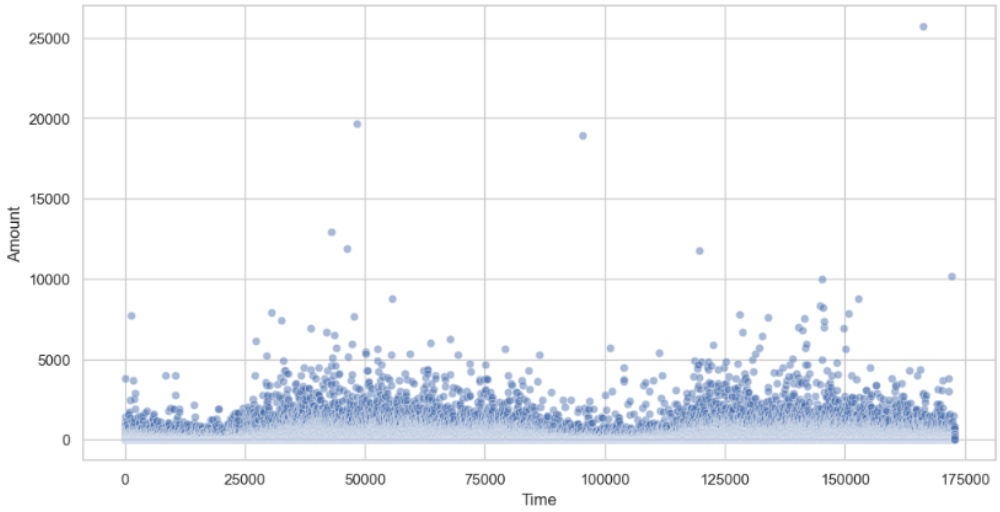
U narednom radu možemo videti da je pristup rešavanju problema sličan našem. Model se takođe trenira isključivo na regularnim transakcijama, a tek pri testiranju dobija i drugu klasu. Suštinska razlika je u arhitekturi samog autoenkodera, koja je nešto jednostavnija. Takođe, za granicu greške (eng. *threshold*) izabrana je vrednost 4.5, koja je znatno veća od naše koja je 2.9. Poređenje rezultata možete pogledati ovde.

Analiziraćemo još jedan pristup problemu, potpuno drugačiji od našeg, on se može naći na linku. Korišćena je metoda slušajnih šuma (*eng. Random Forest*), čija je ideja kombinacija koncepta ansambla i stabla odlučivanja. Konstruiše se veliki broj stabala, pri čemu svako stablo u ansamblu daje nezavisnu predikciju. Konačna predikcija se dobija agregiranjem predikcija svih stabala. Model je treniran na skupu koji sadrži transakcije obe klase, atribut *Time* nije odbačen, a nije izvršeno ni skaliranje atributa *Amount*. Poređenje rezultata može pogledati ovde.

4 Analiza skupa podataka

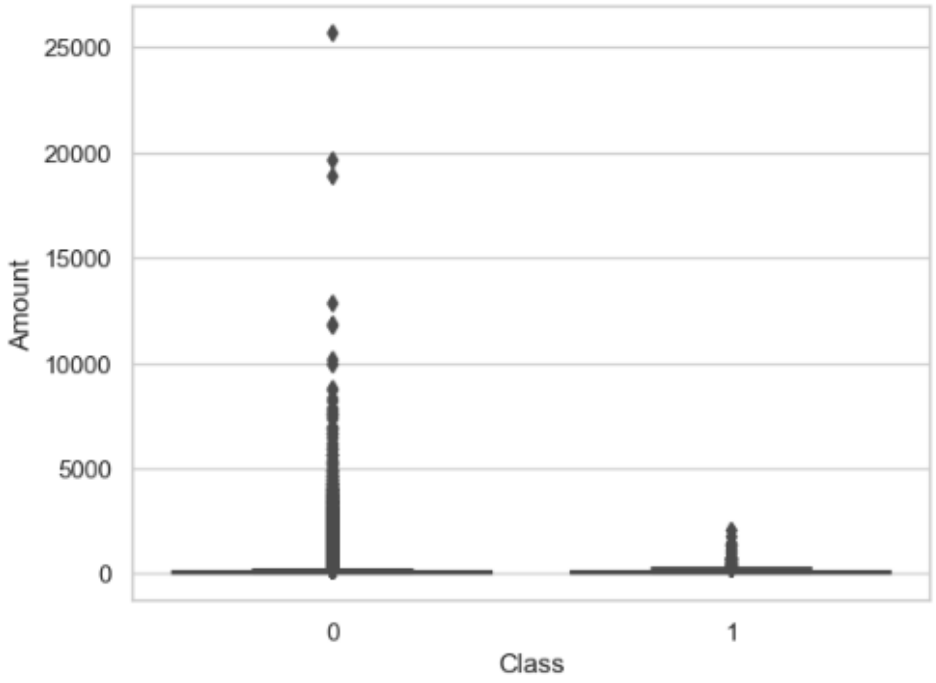
Skup podataka sa kojim radimo sadrži podatke o transakcijama kreditnih kartica u Evropi, septembra 2013. godine. Transakcije se odvijaju u toku dva dana, pri čemu su 492 od 284807 transakcija zabeležene kao prevara.
Kao što vidimo, skup podataka je neuravnotežen, jer zloupotrebe predstavljaju tek 0.172% ukupnog broja transakcija.
Neuravnoteženost skupa podataka može uzrokovati tendenciju modela da favorizuje većinsku klasu, smanjenje sposobnosti modela da identifikuje manjinsku klasu, ali i potencijalno nerealno visoke vrednosti tačnosti klasifikacije, što može negativno uticati na uspešnost procene kvaliteta modela. To nas navodi na ideju izvođenja *oversampling*-a ili *undersampling*-a. Međutim, kako se za trening našeg modela koriste samo regularne transakcije, nebalansiranost skupa podataka neće predstavljati nikakav problem i bilo kakve tehnike koje obezbeđuju balansiranost skupa bile bi u ovom slučaju bez ikakvog efekta.

Što se tiče atributa, od ukupno 31 atributa, poznati su nam *Time* (vreme proteklo od prve transakcije u skupu, izraženo u sekundama), *Amount* (iznos transakcije) i *Class* (1 - prevara, 0 - regularna transakcija), dok su ostali atributi (*V1-V28*) kodirani radi zaštite privatnosti korisnika i oni su dobijeni PCA transformacijom.
Dalje, primećujemo da postoje velike razlike u iznosima transakcija (atribut *Amount* - pogledati na slici 2), pa izvršavamo skaliranje korišćenjem funkcije StandardScaler iz biblioteke scikit-learn.
Analizom modela sa i bez atributa *Time* zaključeno je da on ne doprinosi tačnosti procene, pa je izbačen iz skupa (više o tome pogledati *ovde*).



Slika 2: Iznosi transakcija

Takođe možemo proveriti da li se iznos transakcije značajno menja u zavisnosti od klase kojoj transakcija pripada:



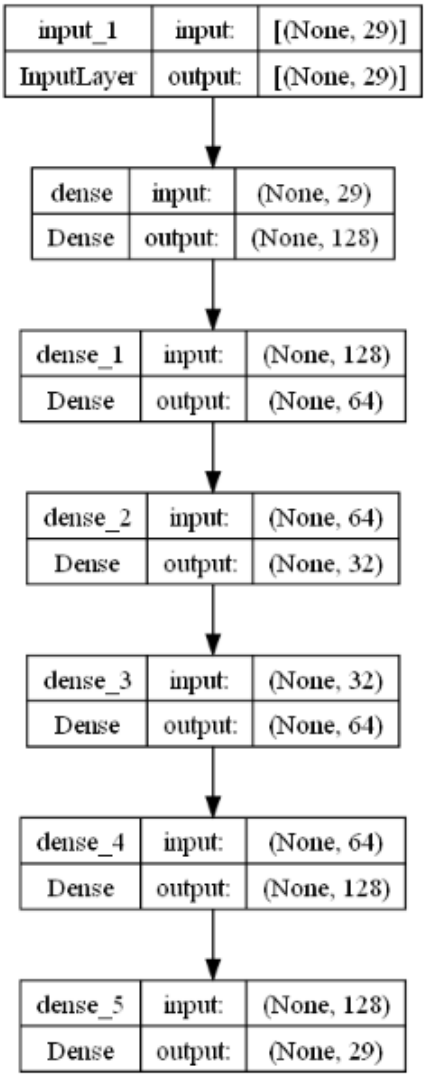
Slika 3: Iznosi transakcija u odnosu na klase

Pokazuje se da izvesna razlika postoji i da je atribut *Amount* vredan razmatranja u našem modelu.

5 Podela na trening i test skup

Ideja je da naš model treniramo isključivo na regularnim transakcijama, a da se testiranje vrši i na regularnim i na prevarama. Kako naš model tokom treniranja „ne zna" za sumnjive transakcije, ovime se postiže uočavanje neregularnosti ukoliko on pravi veliku grešku u proceni prilikom testiranja (grešku iznad unapred zadate granice koja se određuje empirijski).
Veličina test skupa predstavlja 20% veličine dela skupa u kome nije došlo do prevare uz dodatak 492 transakcije u kojima je do prevare došlo.

6 Pravljenje i treniranje modela



Slika 4: Arhitektura modela

Za implementaciju autoenkodera korišćene su mogućnosti koje nudi biblioteka keras i tako je dobijen model koji se može videti na slici 4.

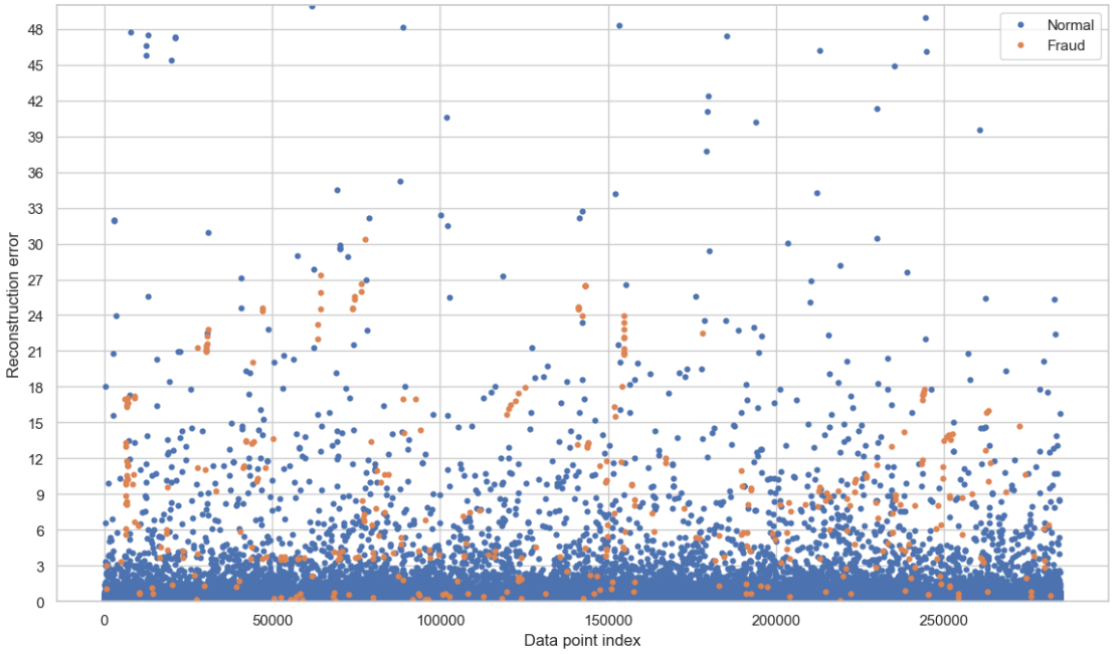
Za aktivacionu funkciju unutrašnjih slojeva odabrana je ispravljena linearna funkcija ($ReLu$), dok je aktivaciona funkcija izlaznog slo-

ja dekodera sigmoidna. Sledi kompajliranje u okviru koga vršimo podešavanje parametara pre početka treniranja. Kao optimizator koristili smo Adam iz biblioteke keras, a za meru greške modela odabrana je srednje kvadratna greška (*eng. mean squared error*).
Kada smo završili sa kompajliranjem, može se preći na treniranje modela. Trening se vrši u 50 epoha, uz implementirano rano zaustavljanje ukoliko u 5 uzastopnih epoha nema nikakvog napretka. Rano zaustavljanje je implementirano korišćenjem funkcije EarlyStopping iz biblioteke keras i tu pratimo vrednosti gubitka na validacionom skupu (validacioni skup je izdvojen kao 20% trening skupa).
Na kraju treninga pretpostavljamo da je naš model naučio karakteristike regularnih transakcija, pa ćemo moći da uočimo neobične greške koje pravi kada na ulazu dobije transakciju koja je prevara.
Kako bismo imali uvid u to kolike greške se prave u odnosu na klasu instance koja je na ulazu modela, pravimo skup podataka koji u sebi sadrži grešku za svaku instancu test skupa kao i klasu kojoj ta instanca pripada. Posmatrajmo deo ovog skupa:

	Reconstruction_error	True_class
138028	0.150737	0
63099	0.074191	0
73411	0.485470	0
164247	0.240192	0
148999	0.748361	0
...	...	...
279863	6.131059	1
280143	3.407078	1
280149	2.723834	1
281144	6.361973	1
281674	0.493450	1

Slika 5: Error dataframe

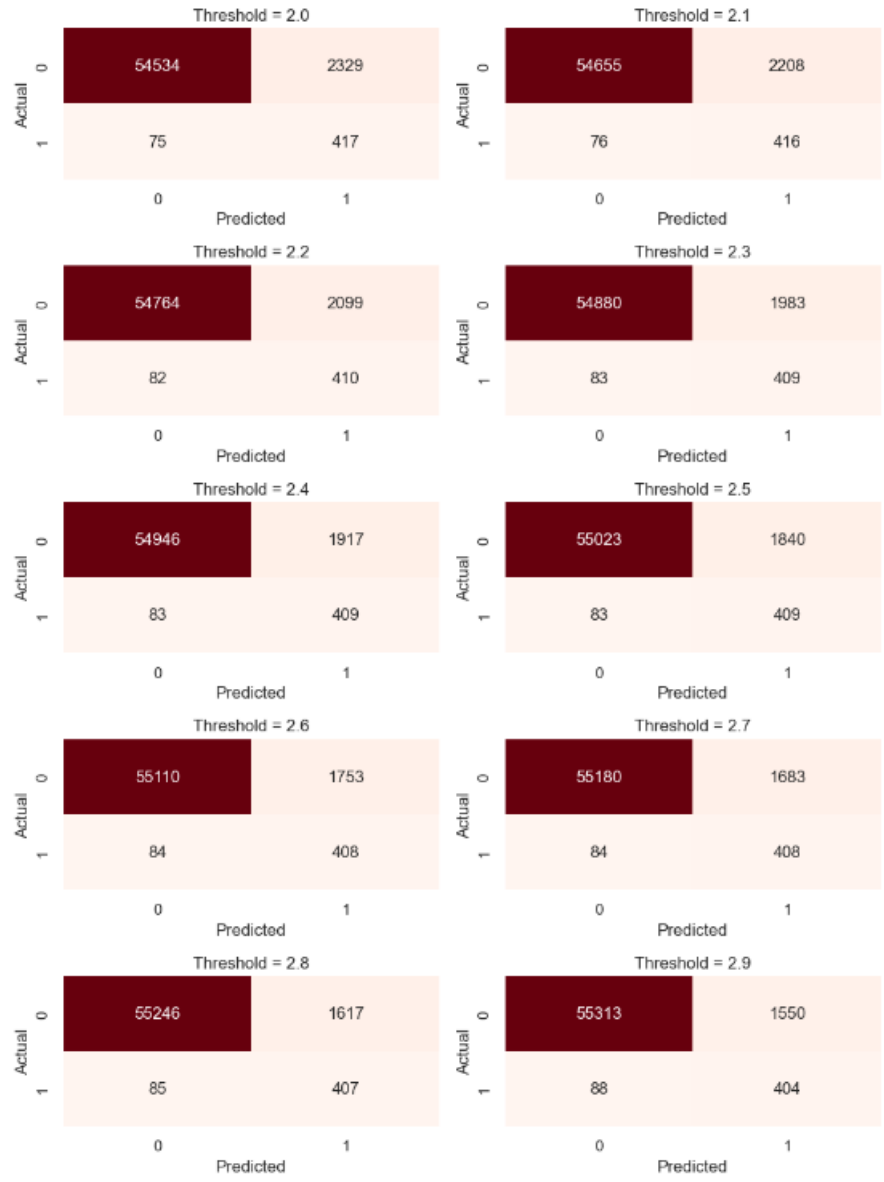
Možemo primetiti da u ovom malom uzorku model uglavnom pravi znatno veće greške kada je u pitanju klasa 1 (što nam je i bila namera), ali to ne mora biti pravilo, kao što možemo videti kod poslednje prikazane instance. Zato je dobro pogledati i vizuelni prikaz grešaka u odnosu na klase instanci.



Slika 6: Vizuelizacija raspodele grešaka u odnosu na klase

Zašto nam je uopšte potreban ovaj vizuelni prikaz? Naime, potrebno je odrediti granicu „dozvoljene greške", odnosno granicu iznad koje ćemo sve instance tretirati kao zloupotrebe. Kao što na slici možemo videti, ne postoji dovoljno dobra granica koja će regularne transakcije u potpunosti odvojiti od prevara, ali potrebno je napraviti balans tako da se prepozna što više zloupotreba uz što manji broj pogrešno klasifikovanih regularnih transakcija.
Možemo primetiti da su na ovoj slici regularne transakcije gusto raspoređene u opsegu vrednosti od 0 do 3, a zatim se proređuju i mešaju sa instancama druge klase. Mali opseg vrednosti manjih ili većih od 3 može biti pogodan upravo za postavljanje granice između dve klase. Izbor

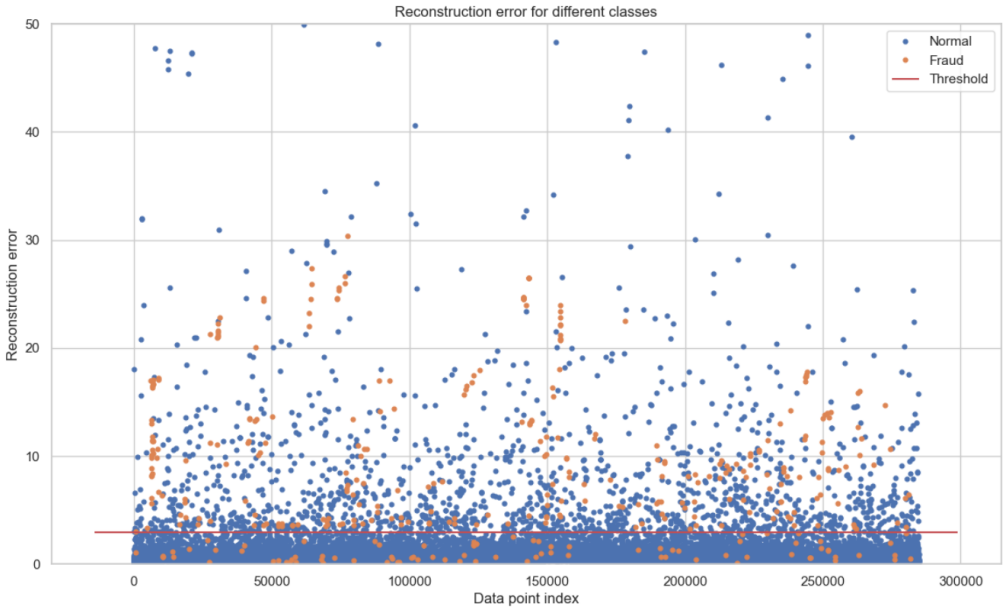
vrednosti za granicu najbolje se može napraviti uz posmatranje performansi modela u zavisnosti od zadate granice, i upravo to ćemo i uraditi - posmatramo matrice konfuzije.



Slika 7: Matrice konfuzije za različite vrednosti granice greške

Kao što na slici možemo videti, što je manja vrednost odabrana za granicu, to se više zloupotreba ispravno klasifikuje. Međutim, ma-

la promena u broju ispravno klasifikovanih zloupotreba povlači veći broj pogrešno klasifikovanih regularnih transakcija i tu treba napraviti balans. Na osnovu analize drugih naučnih radova koji se bave ovom temom i vrednosti ostalih metrika,

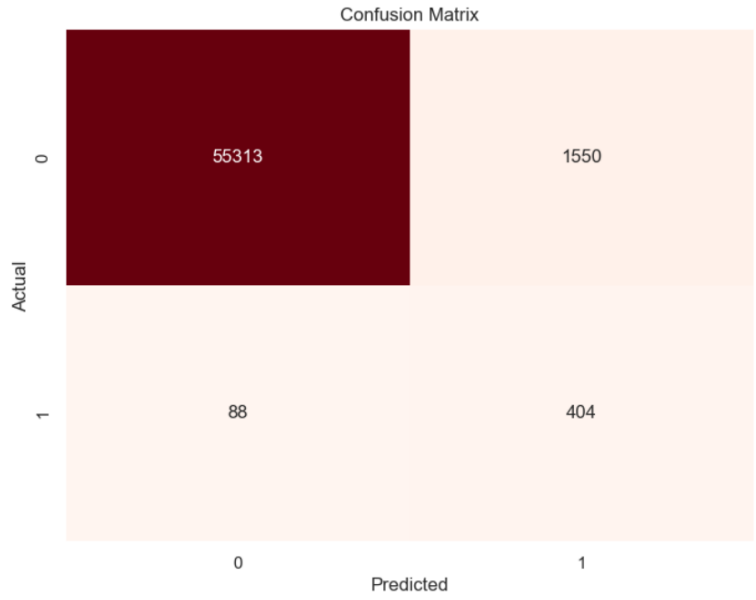


Slika 8: Postavljena granica greške

za granicu je odabrana vrednost **2.9**.
Ovime završavamo podešavanje parametara našeg modela i možemo preći na analizu dobijenih rezultata.

7 Analiza dobijenih rezultata

Analizu rezultata započinjemo matricom konfuzije.



Slika 9: Matrica konfuzije

Naš model ispravno klasifikuje 82.11% lažnih i 97.27% regularnih transakcija. Ako gledamo isključivo brojeve, videćemo da broj regularnih transakcija koje su klasifikovane kao zloupotrebe nije tako mali, ali se može reći da je u praksi dopustiv, jer i inače u oblasti finansija korisnici često prilikom plaćanja prolaze kroz razne sigurnosne provere (odnosno provere da li je sumnjiva transakcija zapravo regularna).

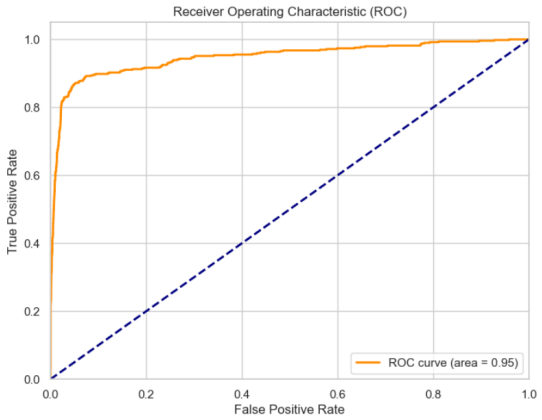
Pogledajmo sada i ostale parametre učinka našeg modela.

	precision	recall	f1-score	support
0	1.00	0.97	0.99	56863
1	0.21	0.82	0.33	492
accuracy			0.97	57355
macro avg	0.60	0.90	0.66	57355
weighted avg	0.99	0.97	0.98	57355

Slika 10: Parametri učinka

Primećujemo da su sve metrike klase regularnih transakcija odlične,

dok su preciznost i F1 mera kod druge klase relativno loše. Ovo naravno ima i svoj razlog. Preciznost meri koliko je zapravo pozitivnih instanci od svih onih koje je naš model označio kao pozitivne. U neuravnoteženim skupovima podataka, ako je jedna klasa znatno manje zastupljena od druge, može doći do problema jer će preciznost u manjinskoj klasi često biti mala, za razliku od preciznosti zastupljenije klase gde može biti mnogo veća. Kako F1 mera kombinuje preciznost i odziv, isti je razlog i za njenu malu vrednost. Zato je dobro uzeti u obzir i druge mere, kao što je ROC (*Receiver Operating Characteristic*) kriva.

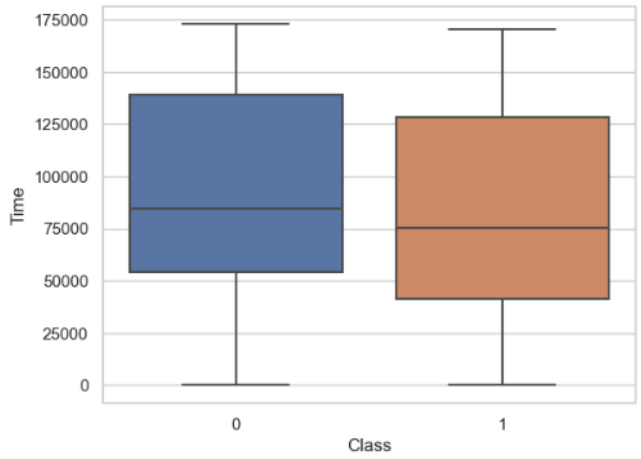


Slika 11: ROC kriva

Kao što vidimo, ona daje odlične rezultate, sa površinom ispod krive (*AUC - Area Under the Curve*) od **0.95**.

7.1 Atribut Time

Kao što smo spomenuli, atribut *Time* ne utiče na kvalitet modela. Možemo pogledati i vizuelni prikaz vrednosti ovog atributa u odnosu na klasu kojoj pripada transakcija, gde se jasno vidi da je razlika neznatna.



Slika 12: Vreme transakcije u odnosu na klasu kojoj pripada

Ovaj atribut obično ne nosi značajne obrasce ili informacije koje bi bile korisne za detekciju prevara. Iz ovih razloga, uobičajena praksa je izostaviti ga.

Umesto toga, bolje je fokusirati se na relevantne osobine transakcija koje su verovatnije da će otkriti nepravilnosti i prevaru. U nastavku možemo pogledati i poređenje rezultata modela sa i bez ovog atributa, iz čega i sledi odluka da se on odbaci.

	precision	recall	f1-score	support
0	1.00	0.97	0.99	56863
1	0.21	0.82	0.34	492
accuracy			0.97	57355
macro avg	0.61	0.90	0.66	57355
weighted avg	0.99	0.97	0.98	57355

Slika 13: Sa atributom

	precision	recall	f1-score	support
0	1.00	0.97	0.99	56863
1	0.21	0.82	0.34	492
accuracy			0.97	57355
macro avg	0.61	0.90	0.66	57355
weighted avg	0.99	0.97	0.98	57355

Slika 14: Bez atributa

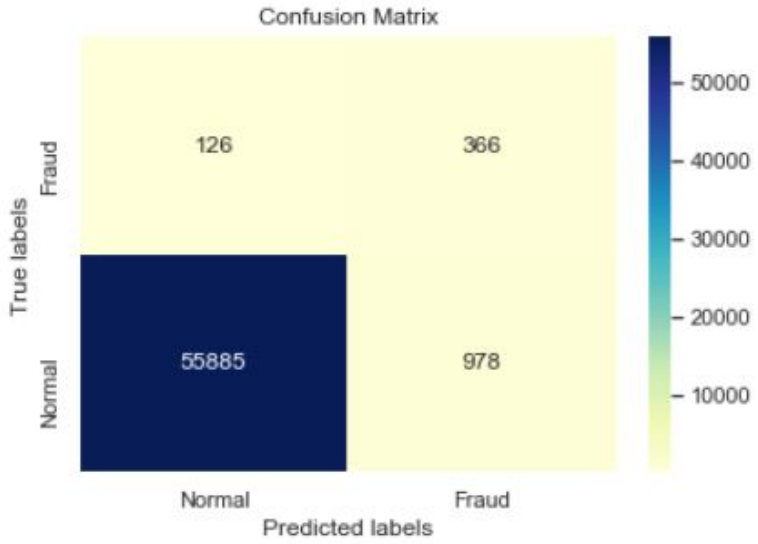
7.2 Poređenje sa rezultatima drugih radova

U ovoj sekciji uporedićemo ponašanje našeg modela, sa modelima drugih autora koji su koristili sličan ili potpuno drugačiji pristup od našeg. Ovo poređenje će nam pomoći da razumemo relativnu efikasnost našeg pristupa u odnosu na postojeće metode, doprinoseći tako širem

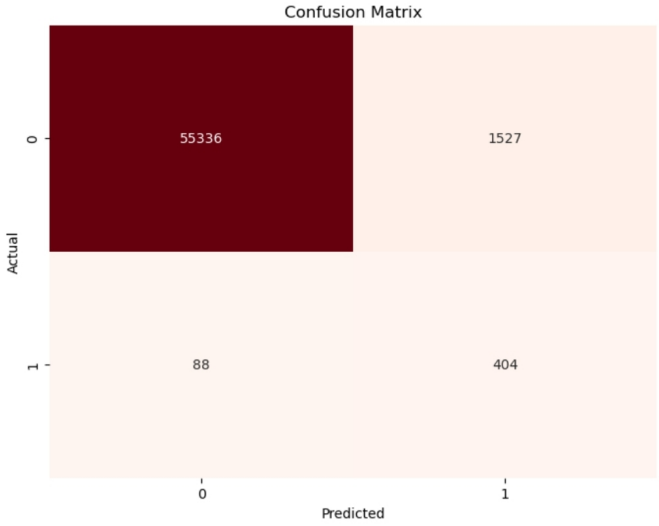
razumevanju efikasnosti različitih tehnika i pristupa u rešavanju kako ovog problema, tako i problema sličnih njemu.

7.2.1 Prvi rad

Prvi pristup koji posmatramo veoma je sličan pristupu koji smo mi primenili, pa je interesantno videti da li postoje razlike u rezultatima i kako su one izražene. Za početak upoređujemo matrice konfuzija.



Slika 15: Matrica konfuzije drugog rada



Slika 16: Matrica konfuzije našeg rada

Jasno se vidi da je kod našeg modela fokus na tačnom prepoznavanju što više neregularnih transakcija, dok se u drugom modelu koji posmatramo težilo balansu između tačnog prepoznavanja regularnih i neregularnih transakcija. Ovi rezultati direktno zavise od unapred zadate granice greške koja je prethodno spomenuta (u našem modelu 2.9, a u drugom 4.0). Što je ta granica veća, veća je i preciznost, dok odziv opada.
Stoga, ne možemo da kažemo da je jedan model bolji od drugog, već da su različiti modeli bolji u različitim okolnostima, odnosno izbor modela zavisi od zahteva i prioriteta u upotrebi modela.

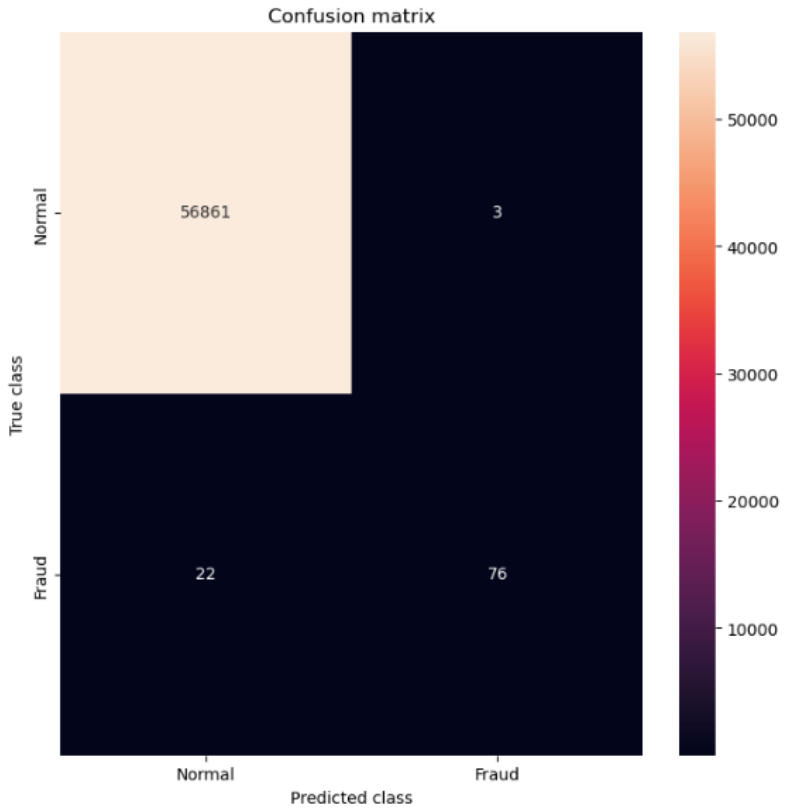
7.2.2 Drugi rad

Za razliku od prvog rada koji smo analizirali, drugi pristup je potpuno drugačiji od našeg, pa je dobro proveriti da li postoji značajna razlika u rezultatima koji se dobijaju. Dobijeni su sledeći rezultati:

- tačnost: 0.99 (naš model 0.97)
- preciznost: 0.98 (naš model 0.21)
- odziv: 0.77 (naš model 0.82)

- F1-score: 0.85 (naš model 0.34)

Vidimo da su preciznost i *F1-score* značajno bolji nego u našem modelu, dok je odziv nešto lošiji. Pre donošenja bilo kakvih zaključaka uporedićemo i matrice konfuzije.



Slika 17: Matrica konfuzije drugog rada

Možemo primetiti da je bez obzira na sve ostale parametre koji su kod ovog modela bolji u odnosu na naš model, procenat neregularnih transakcija koje ispravno klasifikuje ipak manji nego kod našeg modela. Zato ne treba prerano donositi zaključke o tome koji model je bolji, već proveriti sve aspekte, i kao što je ranije navedeno, odabrati pravi model koji će najbolje odgovarati željenoj upotrebi.

8 Zaključak

U ovom radu, istraživali smo primenu autoenkodera kao moćnog alata za detekciju prevara u finansijskim transakcijama. Autoenkoderi su se pokazali kao koristan instrument za identifikaciju nepravilnih i potencijalno lažnih transakcija zato što uče rekonstrukciju normalnih transakcija i identifikuju one koje značajno odstupaju od očekivanog.
Iako autoenkoderi nude efikasan način za detekciju prevara, važno je imati na umu da nijedna tehnika nije savršena. Za postizanje najboljih rezultata, treba imati adekvatan broj primeraka prevara u skupu podataka, pažljivo optimizovati i validirati modele.
Posmatrajući već dostupna rešenja ovog problema, uvideli smo da različite tehnike daju različite mogućnosti, pa čak i minimalne promene određenih parametara mogu uticati na veliku razliku u performansama modela. Zato je bitno na početku definisati ciljeve koje želimo da ostvarimo putem našeg modela, i u skladu sa tim odabrati tehniku koja je za to najpogodnija, ne postoji univerzalno rešenje.
U budućim istraživanjima, treba se fokusirati na razvoj rešenja korišćenjem drugih tehnika, ali i na isprobavanje različitih arhitektura autoenkodera.

9 Literatura

1. Materijali sa kursa Računarska inteligencija, dostupno na: https://github.com/MATF-RI/Materijali-sa-vezbi/tree/master/2022_2023/live

2. Kaggle skup podataka: Credit Card Fraud Detection, dostupno na: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

3. Fraud detection using Autoencoders, dostupno na: https://gsarantitis.wordpress.com/2020/05/14/fraud-detection-using-autoencoders/

4. ML — Credit Card Fraud Detection, dostupno na: https://www.geeksforgeeks.org/ml-credit-card-fraud-detection/