UNIVERZITET SINGIDUNUM

TEHNIČKI FAKULTET

SISTEM ZA PREDIKCIJU KREDITNOG RIZIKA PREDUZEĆA ZASNOVAN NA METODU GRADIJENTNOG PODSTICANJA MAŠINSKOG UČENJA

- diplomski rad -

Mentor:
prof. dr ***Milan Milosavljević***

Kandidat:
Toma Joksimović

Beograd, ***2020*****.**

Sadržaj

1. Uvod

Od početka 21. veka došlo je došlo do velike ekspanzije u količini prikupljenih podataka razvojem veba, kao i sve većem broju kompanija koje klijentima nude usluge, a za uzvrat od klijenata dobijaju veliku količinu podataka koje samim kompanijama mogu pomoći tako što bi izvlačenjem korisnih informacija iz tih podataka mogli videti korisne šablone koje mogu iskoristiti za unapređenje poslovanja.
Jedna od grani privrede koja prikuplja ogromnu količinu podataka od strane svojih klijenata su distributivne kompanije koje dostavljaju robu svojim klijentima bilo da su fizička ili pravna lica. Klijenti često imaju potrebu da traže kredite od njihovih distributera u određenoj novčanoj vrednosti. Zbog prethodnih iskustava, ovo predstavlja veliki izazov i rizik za distributere, zato što se može desiti da klijent u određenom trenutku nije više u mogućnosti da vrati ostatak kredita. Iz ovog problema javila se velika potreba za upotrebom Veštačke Inteligencije kako bi se što više umanjila iscrpna analiza velike količine podataka od strane analitičara i pritom umanjila greška koju često prouzrokuje ljudski faktor u obradi velike količine podataka velike dimenzionalnosti, a gde su se računari adekvatnim algoritmima i modelima pokazali izuzetno pouzdanim.
Aktuelna oblast Veštačke Inteligencije koja rešava ovaj problem je Mašinsko učenje. Ova oblast je doživela izuzetan razvoj poslednje dve decenije, a poslednjih godina taj razvoj se sve više čini da je eksponencijalne prirode zbog intezivnog napretka podoblasti Mašinskog učenja zvanoj Duboko učenje.
Mašinsko učenje je grubo rečeno oblast koja automatizuje klasično programiranje.
Deli se na Nadgledano učenje, Polu-nadgledano učenje, Nenadgledano učenje i Učenje sa potkrepljivanjem.
Postoji način i da se određenim matematičkim formulama jedan tip Mašinskog učenja prevede u drugi. Kada dobijemo kao ulaz podatke sa obeležjima bez izlazne vrednosti, kao što je to slučaj sa podacima koji su korišćeni u ovom radu, može se upotrebiti takozvana „Altman Z Score“ formula koja u zavisnosti od toga da li je klijent fizičko ili pravno lice dodeljuje različite koeficijente određenim parametrima (koji su u ovom slučaju bilansi stanja i bilansi uspeha klijenata).
S obzirom da nam „Altman Z Score“ formula kao rezultat daje četiri intervala koja ćemo prevesti u kategorije rizika: *Bez rizika, Nizak rizik, Srednji rizik i Visok rizik*.
Ova formula se u privredi najčešće koristi za predviđanje bankrota određenog preduzeća.
Da bi se što efikasnije i preciznije dobila kategorija rizika, a kako bi se sačuvala transparentnost i uvid u model Mašinskog učenja korišćena je posebna metoda klasifikacije šumom stabala odlučivanja.
Izuzetno korisna tehnika u predikciji tj. skorovanju kreditnog rizika u mreži distributera je takozvana „Podsticanje“ (engl. „*Boosting“*) tehnika koja spada u ansambl metode koja generiše stabla odlučivanja Gradijentnog Pojačavanja koja će za ulazne podatke o klijentu dati kao izlaz kategoriju rizika. Ova metoda se pokazala kao jedna od najmoćnijih za građenje prediktivnih modela.

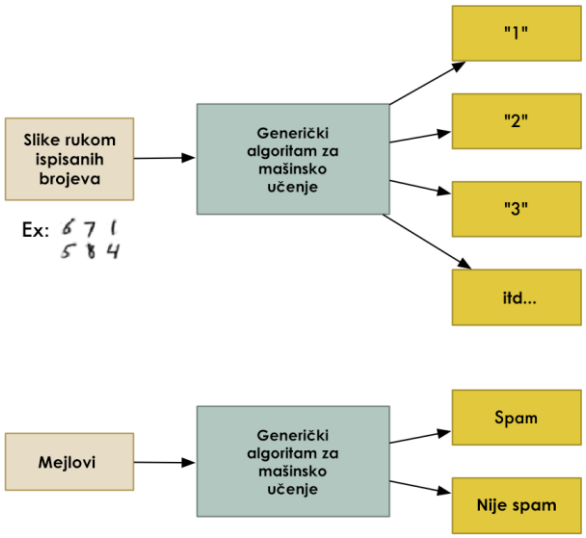
Originalna ideja ove metode potiče od razmatranja da li model koji slabo uči može sebe unaprediti da nekako postane bolji učenik. Slab učenik tj. model je onaj čije su performanse predikcije za nijansu bolje od šanse dobre klasifikacije slučajnim izborom. Ideja je da se slabo naučeni delovi modela uče više puta kako bi za one „teže“ slučajeve za koje se nisu uspešno obučili, narednih par puta bolje obučili. Prvi metod koji je korišćen je Adaptivno Podsticanje koji predstavlja stabla odlučivanja sa jednim rascepom i koja se zbog svoje male dubine nazivaju panjevima odlučivanja. Breiman je AdaBoost i srodne algoritme prepravio u statistički okvir, nazivajući ih ARCing algoritmima. Ovaj okvir je dalje razvio Friedman i nazvao „Gradient Boosting Machines“. Kasnije nazvan samo „Gradijentno Podsticanje“ (engl. „*Gradient Boosting*“).
Statističkog okvir odbacuje „Podsticanje“ kao numerički problem optimizacije gde je cilj minimizirati gubitak modela dodavanjem slabih učenika koristeći postupak nalik na gradijentni spust. Ova klasa algoritama opisana je kao fazni aditivni model. To je zato što se istovremeno dodaje jedan novi slabi učenik, a postojeći slabi učenici u modelu se zamrzavaju i ostaju nepromenjeni. Generalizacija je dozvolila upotrebu proizvoljnih diferencijabilnih funkcija gubitaka, proširujući tehniku izvan problema binarne klasifikacije kako bi podržala regresiju, klasifikaciju u više klasa (kao u što se predstavlja u ovom radu) i još mnogo toga.
Upotrebom metode „Principal Component Analysis“ (PCA) u transformaciji podataka dobili bi smo nova obeležja koja su visoko korelisana sa izlaznom vrednošću.
U ovom modelu se koristi naivna sumarizacija (proseci) radi lakših memorijskih zahteva. Za precizniju sumarizaciju se mogu koristiti Rekurentne Neuronske Mreže, konkretno „Long Short Term Memory“ (LSTM) Neuronska mreža. Međutim u tom slučaju ne bi smo i imali transparentan uvid u postupak odabira kategorije rizika.

2. Mašinsko učenje

Prema Arturu Samuelu, algoritmi Mašinskog učenja omogućavaju računarima da uče iz podataka, pa čak i da se poboljšavaju, bez dodatnog programiranja. Mašinsko učenje je kategorija algoritma koji omogućava softverskim aplikacijama da postanu tačnije u predviđanju ishoda bez izričitog programiranja.Osnovna premisa mašinskog učenja je izgradnja algoritama koji mogu da primaju ulazne podatke i koriste statističku analizu za predviđanje rezultata, dok ažuriraju izlaze kako novi podaci postaju dostupni.
Zahvaljujući statistici, mašinsko učenje je postalo veoma poznato 1990-ih. Ukrštanje računarstva i statistike rodilo je probabilistički pristup u Veštačkoj Inteligenciji. Imajući na raspolaganju velike podatke, naučnici su počeli da grade inteligentne sisteme koji su mogli da analiziraju i uče na velikim količinama podataka. Razvijene su različite tehnike učenja za izvršavanje različitih zadataka.
Tipovi algoritama mašinskog učenja razlikuju se u svom pristupu, tipu ulaznih i izlaznih podataka i samom problemu koji teže da reše. Najpoznatija podela je na:

- Nadgledano učenje (engl. *Supervised Learning*) - podaci sadrže i rešenje
- Nenadgledano učenje (engl. *Unsupervised Learning*) - podaci nisu označeni
- Polu-nadgledano učenje (engl. *Semi-supervised Learning*) - podaci za trening su delimično označeni
- Učenje potkrepljivanjem (engl. *Reinforcement learning*)

Mašinsko učenje se zasniva na ideji da postoje generički algoritmi koji mogu reći nešto interesantno o skupu podataka, a da pritom ne mora da se napiše poseban kod za taj problem. Umesto da se piše kod, ubace se podaci u generički algoritam, a on napravi svoju logiku na osnovu podataka.
Dosada najbolje shvaćen od svih navedenih zadataka je odlučivanje preko jednog pokušaja (engl. one-shot learning). Računaru je dat opis jednog objekta (događaja ili situacije) i od njega se očekuje da kao rezultat izbaci klasifikaciju tog objekta. Na primer, program za prepoznavanje alfanumeričkih znakova kao ulaznu vriednost ima digitalizovanu sliku nekog alfanumeričkog znaka i kao rezultat treba da izbaci njegovo ime. On može da smesti podatke u različite grupe. Isti klasifikacioni algoritam koji se koristi da prepozna rukom napisane brojeve mogao bi da se koristi za klasifikaciju mejlova u “spam” i “nije spam”, bez promene i jedne jedine linije koda. To je isti algoritam, ali su u njega uneti različiti trening podaci, pa on smišlja različitu logiku za klasifikaciju.



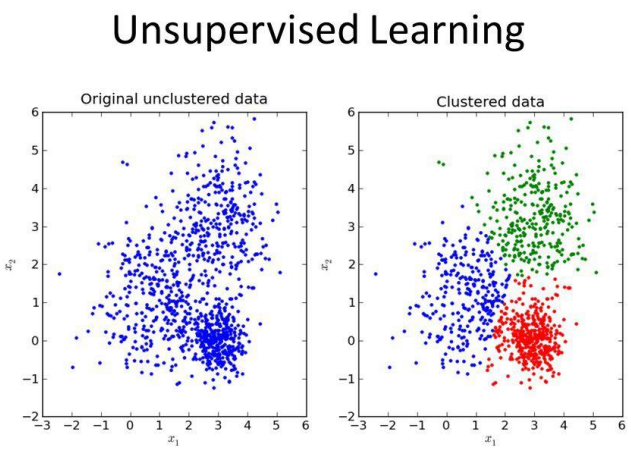
Slika 2.1 - Klasifikacija rukom pisanih znakova i spam mejlova

Ovaj algoritam za Mašinsko učenje je crna kutija — smišlja sopstvenu logiku na osnovu podataka. Mašinsko učenje je krovni termin koji pokriva mnogo ovakvih vrsta klasifikacionih algoritama. Dobijeno znanje koristi se kako bi se dali odgovori na pitanja za entitete/pojave koji nisu ranije viđeni.

2.1. Nenadgledano učenje

Nenadgledano učenje je oblast Mašinskog učenja gde algoritam uči iz jednostavnih primera bez ikakvog povezanog odgovora, prepuštajući algoritmu da sam utvrđuje obrasce podataka. Ova vrsta algoritma teži prestrukturiranju podataka u nešto drugo, poput novih karakteristika koje mogu predstavljati klasu ili novu seriju neusklađenih vrednosti. Oni su vrlo korisni u pružanju uvida ljudima u značenje podataka i novih korisnih unosa u nadgledane algoritme mašinskog učenja.
Kao vrsta učenja, podseća na metode koje ljudi koriste da bi utvrdili da su određeni predmeti ili događaji iz iste klase, na primer posmatranjem stepena sličnosti između predmeta. Neki sistemi preporuka koji se na mreži mogu pronaći u obliku marketinške automatizacije zasnovani su na ovoj vrsti učenja.
Kako su neobeleženi podaci sve češći i prusitniji od obeleženih, posebno su dragocene metode mašinskog učenja koje olakšavaju učenje bez učitelja (obeleženih podataka).
Cilj učenja bez učitelja može biti podjednako jednostavan kao otkrivanje skrivenih šablona unutar skupa podataka, ali takođe može imati za cilj i učenje karakteristika, što omogućava računarskoj mašini da automatski otkrije prikaze potrebne za klasifikaciju sirovih podataka.
Učenje bez nadzora se često koristi za transakcione podatke. Posedovanjem velikog skupa kupaca i njihovih kupovina ljudsko biće nije u stanju da lako shvati koji slični atributi mogu da se izvuku iz profila kupaca i njihovih vrsta kupovina. Uz pomoć ovih podataka u algoritmu učenja bez učitelja, može se utvrditi da su žene određenog uzrasta koje kupuju sapune bez mirisa verovatno trudne, pa stoga marketinška kampanja vezana za trudnoću i proizvode za bebe može biti usmerena na ovu publiku kako bi da povećaju njihov broj kupovina.
Bez da im se kaže „tačan“ odgovor, metode učenja bez učitelja mogu da gledaju na složenije podatke koji su obimniji i naizgled nepovezani kako bi ih organizovali na potencijalno značajne načine. Učenje bez učitelja često se koristi za otkrivanje anomalija, uključujući i za nepoštene kupovine kreditnim karticama i sisteme preporučivača koji preporučuju koje proizvode treba sledeće kupiti. U učenju bez učitelja, neobeležene fotografije pasa mogu se koristiti kao ulazni podaci za algoritam za pronalaženje sličnosti i zajedničko klasifikovanje fotografija pasa.
Tipovi Nenadgledanog učenja su:

- Klasterovanje (engl. *Clustering*) - metod u kome se teži otkrivanju svojstvene grupe podataka, kao što je grupisanje kupaca prema ponašanju kupovine.
- Asocijacija (engl. *Association*) - učenja pravila udruživanja je metod gde se teži otkrivanju pravila koja opisuju velike delove podataka, kao što su ljudi koji kupuju X takođe teže da kupe Y.



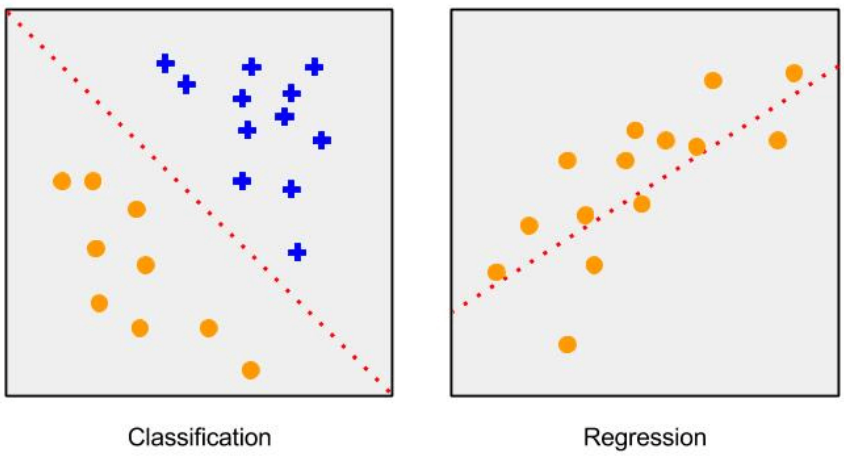
Slika 2.2 – Nenadgledano učenje klasterovanjem

Na slici iznad je prikaz Nenadgledanog učenja metodom klasterovanja nazvanom „K-srednjih“ (engl. *K-means*). Ovaj algoritam inicijalno postavlja centriode u prostoru obeležja i iterativno ažurira položaje centrioda tako što za svaku centriodu računa srednje Euklidsko rastojanje svih tačaka koje su pripadale tom klasteru, uz eventualno dodavanje novih tačaka koje do promene položaja nisu pripadale klasteru, kao i dodeljivanje dotadašnjih tačaka drugom klasteru čijoj centriodi su bliže. Podaci se klasteruju u nekoliko klastera po međusobnoj sličnosti u unutar klastera ,a sami klasteri međusobno po što većoj različitosti.

2.2. Nadgledano učenje

Nadgledano učenje bila je prva vrsta učenja koja se istraživala na polju Veštačke Inteligencije. Od njegove koncepcije, bezbrojni algoritmi - koji se razlikuju u složenosti od skromne Logističke Regresije do masivne Neuronske Mreže - istraženi su kako bi se poboljšala tačnost i snaga predviđanja.
Algoritam Nadgledanog učenja uči na primerima podataka i povezanim ciljnim odgovorima koji se mogu sastojati od numeričkih vrednosti ili labeli stringova, kao što su klase ili oznake, kako bi kasnije predvideo što tačniji odgovor kada se dobije nov podatak. Ovaj pristup je zaista sličan ljudskom učenju pod nadzorom učitelja. Učitelj pruža dobre primere koje učenik može da pamti, a učenik zatim iz ovih konkretnih primera izvodi opšta pravila.
Učenje sa učiteljem koristi trening skup za podučavanje modela koji daju željene rezultate. Ovaj skup podataka obuke uključuje ulaze i tačne izlaze, koji omogućavaju modelu da vremenom uči. Algoritam meri svoju tačnost kroz funkciju gubitka (engl. *Loss Function*), prilagođavajući se dok greška ne bude dovoljno smanjena. Zamislite učitelja koji će, znajući tačan odgovor, ili dodeliti ocene učeniku ili uzeti ocene od učenika na osnovu ispravnosti odgovora na pitanje.
Kao i svi algoritmi mašinskog učenja, i Nadgledano učenje se zasniva na treningu. Tokom faze treninga sistem se dopunjava ogromnim količinama podataka koji upućuju model koji izlaz treba dobiti iz svake određene ulazne vrednosti. Obučeni model se zatim prikazuje test podacima kako bi se verifikovao rezultat treninga i izmerila tačnost.
U algoritmima neuronske mreže, nadgledani proces učenja se poboljšava stalnim merenjem rezultujućeg rezultata modela i finim podešavanjem sistema kako bi se približio ciljnoj tačnosti. Dostupni nivo tačnosti zavisi od dve stvari: dostupnih podataka i algoritma koji se koristi.
Visoka tačnost nije nužno dobar pokazatelj. To bi takođe moglo značiti da model pati od preoubučenosti (engl. *Overfitting*) tj. preterano je prilagođen svom određenom skupu podataka na treninigu. Takav skup podataka mogao bi se dobro izvoditi u testnim scenarijima, ali neuspešno bi radio kada mu se predoče novi podaci. Da bi se izbegla preobučenost, važno je da se podaci na testu razlikuju od podataka na trennigu kako bi se osiguralo da model ne koristi odgovore iz svog prethodnog iskustva, već da se zaključak modela generalizuje.
Tipovi Nadgledanog učenja su:

- Klasifikacija (engl. *Classification*) - metod koristi algoritam za tačno dodeljivanje podataka o testovima u određene kategorije. Prepoznaje određene entitete unutar skupa podataka i pokušava izvući neke zaključke o tome kako bi ti entiteti trebali biti označeni ili definisani. Uobičajeni algoritmi klasifikacije su Linearni Klasifikatori, Mašine Nosećih Vektora (SVM), Stabla Odlučivanja, K-najbližih Sused i Slučajna Šuma.
- Regresija (engl. Regression) – koristi se za razumevanje odnosa između zavisnih i nezavisnih promenljivih. Obično se koristi za izradu projekcija, na primer za prihod od prodaje za određeno preduzeće. Linearna Regresija, Logistička Regresija i Polinomna Regresija popularni su algoritmi.



Slika 2.3 – Nadgledano učenje (Klasifikacija i Regresija)

2.3. Klasifikacija

Klasifikacija je postupak predviđanja klase zadatih podataka. Klase se ponekad nazivaju kategorijama / ciljnim atributima ili labelama. Klasifikaciono prediktivno modeliranje je zadatak približavanja funkcije mapiranja (f) od ulaznih promenljivih (X) do diskretnih izlaznih promenljivih (Y).
Na primer, otkrivanje spma mejlova kod provajdera usluga e-pošte može se identifikovati kao problem klasifikacije. Ovo je Binarna klasifikacija, jer postoje samo 2 klase spam mejlovi i mejlovi koji nisu spam. Klasifikator koristi neke podatke na treningu da bi razumeo kako se date ulazne promenljive odnose na klasu. U ovom slučaju, poznati podaci o spam i pošti koja nije spam se moraju koristiti kao podaci na treningu. Kada se klasifikator pravilno obuči, može se koristiti za otkrivanje nepoznate e-pošte.
Klasifikacija spada u kategoriju Nadgledanog učenja gde su ciljevi takođe dati sa ulaznim podacima. Velika je primena u klasifikacije u mnogobrojnim domenima, poput odobrenja kredita (koja je glavna tema ovog rada), medicinske dijagnoze, ciljnog marketinga itd.
Postoje dve vrste učenika:

- Lenji učenici (engl. *Lazy Learners*) - jednostavno čuvaju podatke sa treninga i čekaju dok se ne pojave podaci sa testiranja. Kada se to dogodi, klasifikacija se vrši na osnovu podataka koji su najviše povezani sa usklađenim podacima na treningu. U poređenju sa željnim učenicima, lenji učenici imaju manje vremena za obuku, ali više vremena za predviđanje.
- Željni učenici (engl. *Eager Learners*) - konstruišu model klasifikacije na osnovu datih podataka sa treninga pre nego što dobiju podatke za klasifikaciju. Moraju biti u stanju da se posvete jednoj hipotezi koja pokriva čitav prostor instance. Zbog konstrukcije modela, željnim učenicima treba puno vremena za obuku i manje vremena za predviđanje.

Sada je dostupno puno algoritama za klasifikaciju, ali nije moguće zaključiti koji je superiorniji od drugog. Zavisi od primene i prirode dostupnog skupa podataka. Na primer, ako su klase linearno odvojive, linearni klasifikatori poput Logistička regresija, Fisherov linearni diskriminant mogu nadmašiti sofisticirane modele i obrnuto.
Postoje četiri vrste klasifikacije:

- Binarna klasifikacija (engl. *Binary Classification*)
- Višeklasna klasifikacija (engl. *Multi-Class Classification*)
- Višeznačna klasifikacija (engl. *Multi-Label Classification*)
- Nebalansirana klasifikacija (engl. *Imbalanced Classification*)

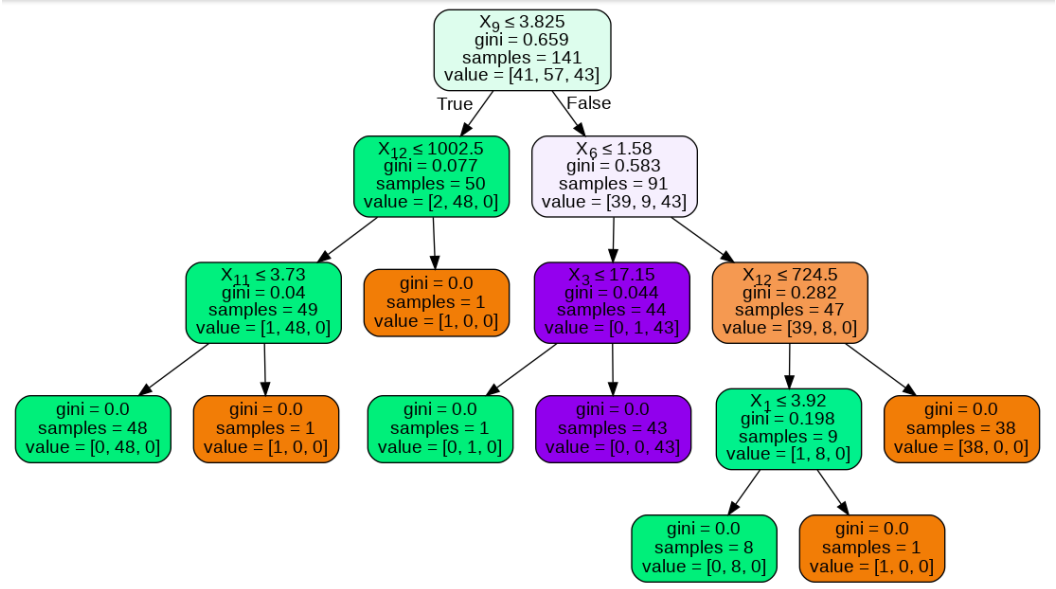
Projekat koji je predstavljen u ovom radu koristi Višeklasnu klasifikaciju s obzirom da imamo 4 klase izlazne vrednosti (Nema rizika, Nizak rizik, Srednji rizik, Visok rizik). Višeklasna Klasifikacija odnosi se na one zadatke klasifikacije koji imaju više od dve oznake klase kako i sam naziv govori.
Za razliku od Binarne klasifikacije, Višeklasna klasifikacija nema pojam normalnih i abnormalnih ishoda. Umesto toga, primeri su klasifikovani kao da pripadaju jednoj od poznatih klasa.
Broj klasa može biti vrlo velik kod nekih problema. Na primer, model može predvideti da fotografija pripada jednom od hiljada ili desetina hiljada lica u sistemu za prepoznavanje lica.
Problemi koji uključuju predviđanje niza reči, kao što su modeli prevođenja teksta, takođe se mogu smatrati posebnom vrstom Višeklasne klasifikacije. Svaka reč u nizu reči koju treba predvideti uključuje Višeklasnu klasifikaciju gde veličina rečnika definiše broj mogućih klasa koje se mogu predvideti i može biti velika na desetine ili stotine hiljada reči.
Popularni algoritmi koji se koriste u Višeklasnoj klasifikaciji su:

- K-Najbližih Suseda (engl. *K-Nearest Neighbors*)
- Stabla odlučivanja (engl. *Decison Trees*)
- Naivni Bajes (engl. *Naive Bayes*)
- Slučajne šume (engl. *Random Forest*)
- Gradijentno Podsticanje (engl. *Gradient Boosting*)

Algoritmi dizajnirani za Binarnu klasifikaciju mogu se prilagoditi za upotrebu u višeklasnim problemima. U klasifikaciji vezanoj za kreditni rizik se najčešće koriste stabla odlučivanja ili ansambl stabala.

Stablo odlučivanja gradi modele klasifikacije ili regresije u obliku strukture stabla. Koristi skup pravila „ako-onda“ (engl. „*if-then*“) koji se međusobno isključuju i iscrpno klasifikuju. Pravila se uče uzastopno koristeći podatke sa treninga jedan po jedan. Svaki put kada se pravilo nauči, torke obuhvaćene pravilima se uklanjaju. Ovaj proces se nastavlja u obuci do ispunjenja uslova za prekid.
Stablo je konstruisano na način da se rekurzivno „podeli-osvoji“ (engl. „*Divide-and-Conquer*“) „odozgo-nadole“ (engl. „*top-down*“) pristupom. Svi atributi treba da budu kategorični. U suprotnom, trebalo bi ih unapred diskretizovati. Atributi na vrhu stabla imaju veći uticaj na klasifikaciju i oni se identifikuju pomoću koncepta „informacionog dobitka“ (engl. „*Information Gain*“).
Stablo odluke može se lako prekomerno preobučiti, generišući previše grana i može odražavati anomalije zbog šuma ili odstupanja određenih podataka. Prekomerno obučen model ima vrlo loše performanse na neviđenim podacima, iako daje impresivne performanse na podacima sa treninga. To se može izbeći kresanjem koje rano zaustavlja izgradnju stabla ili kasnijim kresanjem koje uklanja grane sa potpuno poraslog stabla.
Stablo odluke daje prednost jednostavnosti za razumevanje i vizualizaciju, zahteva i vrlo malo pripreme podataka. Nedostatak koji sledi sa stablom odluke je taj što može stvoriti složena stabla koja se ne mogu efikasno kategorizovati. Stabla odluke mogu biti prilično nestabilna jer čak i pojednostavljena promena podataka može poremetiti celu strukturu stabla odlučivanja.
Najčešće upotrebe stabla odlučivanja su:

- Istraživanje podataka (engl. *Data exploration*)
- Prepoznavanje šablona (engl. *Pattern Recognition*)
- Opciono određivanje cena u finansijama (engl. *Option pricing in finances*)
- Utvrđivanje bolesti i pretnji od rizika (engl. *Identifying disease and risk threats*)



Slika 2.4 – Primer Stabla Odlučivanja

Stablo odlučivanja se gradi prateći sledeće korake:

1. Izaberite prediktorsku promenljivu) koja najbolje klasifikuje skup podataka u željene klase i dodeliti je korenskom čvoru.
2. Preći preko korenskog čvora, istovremeno donoseći relevantne odluke na svakom unutrašnjem čvoru tako da svaki unutrašnji čvor najbolje klasifikuje podatke.
3. Vratiti se do koraka 1 i ponavljati korake dok se ulaznim podacima ne dodeli klasa.

U sistemu za predikciju kreditnog rizika koji je opisan dalje u ovom radu je model Mašinskog učenja zasnovan na šumi tj. ansamblu stabla odlučivanja i oni zajedno sarađuju primenom metode „pojačavanja“ (engl. „*Boosting*“). Pojedinačna stabla odlučivanja nisu na kompleksnijim problemima toliko moćna koliko ansambli stabala.

2.3.1. Ansambl tehnike

Učenje Ansambl tehnikom pomaže u poboljšanju rezultata Mašinskog učenja kombinovanjem nekoliko modela. Ovaj pristup omogućava proizvodnju boljih prediktivnih performansi u poređenju sa jednim modelom. Zbog toga su ansambl metode na prvom mestu mnogih prestižnih takmičenja u Mašinskom učenju.

Za razliku od statističkog ansambla u statističkoj mehanici, koja je obično beskonačna, ansambl mašinskog učenja se sastoji samo od konkretnog konačnog skupa alternativnih modela, ali obično omogućava da među tim alternativama postoji mnogo fleksibilnija struktura.

Učenje u ansamblima može se smatrati načinom nadoknađivanja loših algoritama učenja izvođenjem puno dodatnih proračuna. S druge strane, alternativa je mnogo više učenja na jednom ne-ansambl sistemu. Ansambl sistem može biti efikasniji u poboljšanju ukupne tačnosti za isto povećanje računarskih, memorijskih ili komunikacionih resursa korišćenjem tog povećanja na dve ili više metoda, nego što bi bio poboljšan povećanom upotrebom resursa za jednu metodu.

Ansambl je sam po sebi algoritam Nadgledanog učenja, jer se može obučiti, a zatim koristiti za predviđanje. Obučeni ansambl, dakle, predstavlja jednu hipotezu. Ova hipoteza, međutim, nije nužno sadržana u prostoru hipoteza modela iz kojih je izgrađena. Tako se ansamblima može pokazati da imaju veću fleksibilnost u funkcijama koje mogu da predstavljaju. Ova fleksibilnost im, u teoriji, može omogućiti da previše obuče (engl. *over-fit*) podatke na treningu više nego što bi to učinio jedan model, ali u praksi neke ansambl tehnike (posebno „Upakivanje“) imaju tendenciju da smanje probleme povezane sa prekomernim obučavanjem podataka na treningu.

Empirijski, ansambli imaju tendenciju da daju bolje rezultate kada postoji velika raznolikost među modelima. Stoga mnoge metode ansambla nastoje da istaknu različitost među modelima koje kombinuju. Iako se možda neintuitivno, više slučajnih algoritama (poput slučajnih stabala odlučivanja) može koristiti za stvaranje jačeg ansambla od vrlo konkretnih algoritama (poput stabala odluka koje smanjuju entropiju). Međutim, pokazalo se da je upotreba različitih jačih algoritama učenja efikasnija od korišćenja tehnika koje pokušavaju da „zaglupe“ modele kako bi se promovisala različitost.

Iako broj klasifikatora ansambla ima veliki uticaj na tačnost predviđanja, postoji ograničen broj studija koje se bave ovim problemom. A priori određivanje veličine ansambla, obima i brzine velikih tokova podataka čine ovo još presudnijim za dostupne klasifikatore ansambala. Za određivanje tačnog broja komponenata korišćeni su uglavnom statistički testovi. U novije vreme, teorijski okvir sugeriše da postoji idealan broj klasifikatora komponenti za celinu takav da bi posedovanje više ili manje od ovog broja klasifikatora pogoršalo tačnost. Zove se „zakon opadajućeg prinosa u konstrukciji ansambla“ (engl. „*the law of diminishing returns in ensemble construction*“). Njihov teorijski okvir pokazuje da upotreba istog broja nezavisnih klasifikatora komponenata kao oznaka klasa daje najveću tačnost.

Metode ansambla su meta-algoritmi koji kombinuju nekoliko tehnika Mašinskog učenja u jedan prediktivni model kako bi se smanjila varijansa („Upakivanje“), šum („Podsticanje“) ili poboljšala predviđanja („Naslaganje“).

Tehnike ansambla se mogu podeliti u dve grupe:

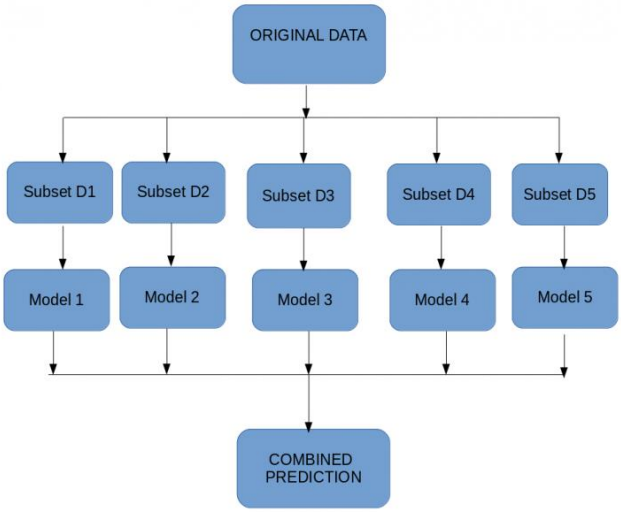
- Sekvencijalne tehnike ansambla (engl. „*Sequental Ensemble Learning*“) gde se osnovni učenici generišu sekvencijalno (npr. „AdaBoost“, „Stochastic Gradient Boosting“).
  Osnovna motivacija sekvencijalnih tehnika je iskorišćavanje zavisnosti između osnovnih učenika. Ukupne performanse se mogu poboljšati vaganjem prethodno pogrešno označenih primera sa većom težinom.
- Paralelne tehnike ansambla (engl. „ *Parallel Ensemble Learning*“) gde se osnovni učenici generišu paralelno (npr. „Random Forest“, „Bagged Decision Trees“, „Extra Trees“). Osnovna motivacija paralelnih tehnika je iskorišćavanje nezavisnosti među osnovnim učenicima, jer se greška može dramatično smanjiti usrednjavanjem.

Većina metoda u ansamblima koristi algoritam jednostrukog učenja za stvaranje homogenih bazičnih učenika, tj. učenika istog tipa, što dovodi do „homogenih ansambala“.

2.3.1.1. Upakivanje

Ideja koja stoji iza Upkaivanja je kombinovanje rezultata više modela (na primer, svih stabala odlučivanja) da bi se dobio generalizovani rezultat.
Upakivanje (engl. *Bagging*) podrazumeva da svaki model u ansamblu glasa sa jednakom težinom. Da bi se promovisala varijansa modela, Upakivanje trenira svaki model u ansamblu koristeći nasumično izvučeni podskup skupa za obuku. Kao primer, algoritam slučajnih šuma kombinuje stabla slučajnih odluka sa upakivanjem da bi postigao vrlo visoku tačnost klasifikacije.
Upakivanjem se uzorci generišu na takav način da se uzorci međusobno razlikuju, međutim zamena je dozvoljena. Zamena znači da se instanca može pojaviti u više uzoraka više puta ili se uopšte ne može pojaviti u nekim uzorcima. Ovi uzorci se zatim daju višestrukim učenicima, a zatim se rezultati svakog učenika kombinuju u obliku glasanja.
Veličina podskupova je ista kao i veličina originalnog skupa. Tehnika Upakivanja koristi ove podskupove („pakovanja“) da bi stekla poštenu predstavu o distribuciji (kompletan skup). Veličina podskupova generisanih za upakivanje može biti manja od originalnog skupa.
Jednom kada su svi prediktori obučeni, ansambl može da predvidi novi primer agregiranjem predviđenih vrednosti svih obučenih prediktora. Iako svaki pojedinačni prediktor ima veći šum nego da je obučen na originalnom skupu podataka, agregacija omogućava smanjenje šuma i varijanse.
Upakivanje unosi više raznolikosti u svaki podskup, stoga se može očekivati veći šum, što znači da su prediktori u manjoj korelaciji, pa je varijansa ansambla smanjena.
Proces tehnike Upakivanja je sažet sledećim koracima:

1. Iz originalnog skupa podataka kreira se više podskupova, birajući zapažanja sa zamenom.
2. Nad svakim od ovih podskupova kreira se osnovni model (slab model).
3. Modeli se obučavaju paralelno i nezavisni su jedan od drugog.
4. Konačna predviđanja se određuju kombinovanjem rezultata predviđanja iz svih modela.



Slika 2.5 – Proces tehnike Upakivanja modela

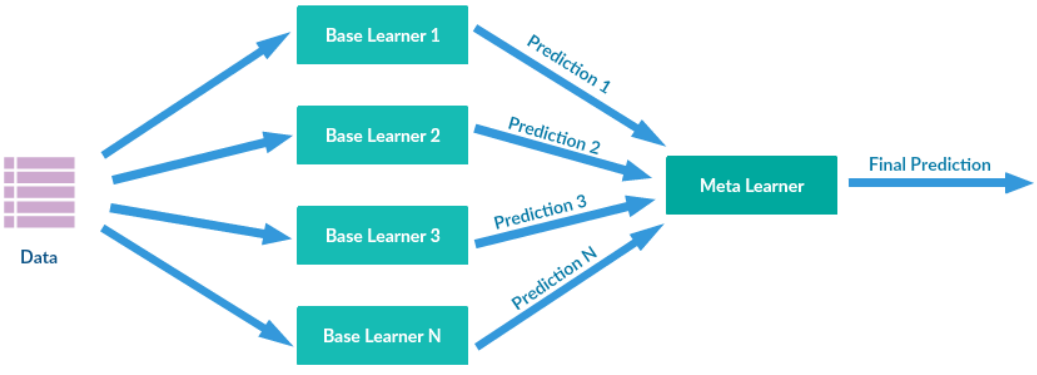
Na primer, možemo obučiti M različita stabla na različitim podskupovima podataka (izabranih slučajno sa zamenom) i izračunati ansambl datom formulom:

$$f(x) = 1/\mathrm{M} \sum_{m=1}^{M} f_m(x)$$

Za objedinjavanje rezultata osnovnih učenika, Upakivanje koristi glasanje za klasifikaciju i usrednjavanje za regresiju. Kombinovanje stabilnih učenika je manje povoljno jer ansambl neće pomoći u poboljšanju performansi generalizacije. Klasa algoritama koji se najčešće koriste su šume nasumičnih stabala (engl. *randomized trees*).

2.3.1.2. Naslaganje

Naslaganje (ponekad se naziva i složena generalizacija) uključuje obuku algoritma učenja za kombinovanje predviđanja nekoliko drugih algoritama učenja. Prvo se svi ostali algoritmi obučavaju koristeći raspoložive podatke, zatim se algoritam kombinovanja obučava za pravljenje konačnog predviđanja koristeći sva predviđanja ostalih algoritama kao dodatne ulaze. Ako se koristi proizvoljan algoritam kombinovanja, tada naslaganje teoretski može predstavljati bilo koju od ansambl tehnika opisanih u ovom radu, iako se u praksi često koristi model Logističke regresije kao kombinator.
Slaganje obično daje performanse bolje od bilo kog od obučenih modela. Uspešno se koristi i za zadatke učenja pod sa učiteljem (Regresija, Klasifikacija i učenje na daljinu) i za učenje bez učitelja (procena gustine). Takođe se koristi za procenu stope grešaka u Upakivanju. Izvešteno je da je premašio Bajesovo usrednjavanje. Dvoje najuspešnijih na Netflikskovom takmičenju su koristili kombinovanje, što se može smatrati oblikom Naslaganja. Ovaj model se koristi za predviđanje na test skupu.
Takođe poznata kao složena generalizacija, ova tehnika se zasniva na ideji da se obuči model koji bi izveo uobičajeno agregiranje koje je ranije viđeno. Imamo N prediktora, svaki daje svoja predviđanja i vraća konačnu vrednost. Kasnije će Meta Učenik (engl. *Meta Learner*) ili Blender uzeti ova predviđanja kao ulaz i dati konačno predviđanje.
Uobičajeni pristup za obuku Meta učenika je skup za zadržavanje (engl. „*Hold-out*“). Prvo, trening skup podeliti u dva podskupa. Prvi podskup podataka koristi se za obuku prediktora.
Kasnije, isti prediktori koji su trenirali na prvom podskupu koriste se za predviđanje drugog (zadržanog) skupa. Time se osigurava čistoća predviđanja jer ti algoritmi nikada nisu videli podatke.
Pomoću ovih novih predviđanja, može se generisati novi trening skup koji će se koristiti kao ulazna karakteristika (čineći novi trening skup trodimenzionalnim) i zadržavajući ciljne vrednosti.
Zbog toga se na kraju Meta Učenik obučava za ovaj novi skup podataka i predviđa ciljne vrednosti uzimajući u obzir vrednosti date u prvim predviđanjima.
Štaviše, može se obučiti nekoliko Meta Učenika koristeći ovu metodologiju (npr. Jedan koji koristi Linearnu regresiju, drugi koji koristi Regresiju Slučajne Šume, itd.). Da biste koristili više od jednog Meta Učenika, trening skup mora se podeliti na tri ili više podskupova: prvi za obuku prvog sloja prediktora, drugi koji se koristi za predviđanje nepoznatih podataka i stvaranje novog skupa podataka, a treći za obučavanje Meta Učenika i napraviti predviđanja za ciljne vrednosti.



Slika 2.6 – Proces tehnike Naslaganja modela

Osnovni nivo se često sastoji od različitih algoritama učenja, pa su stoga ansambli za Naslaganje često heterogeni. Naslaganje je često korišćena tehnika za pobedu na Kaggle takmičenju u Nauci o Podacima. Na primer, prvo mesto za izazov Otto Group Classification Product osvojio je ansambl za Naslaganje od preko 30 modela čiji su izlazi korišćeni kao karakteristike za tri meta-klasifikatora: XGBoost, Neuronska mreža i AdaBoost.

2.3.1.3. Podsticanje

Ako je tačka podataka netačno predvidena od strane prvog modela kao i sledećeg (verovatno svih modela), da li će kombinacija njihovih rezultata pružiti bolje predviđanje? Ovde Podsticanje stupa na snagu. Podsticanje uključuje postupno razvijanje ansambla treniranjem svake nove instance modela kako bi se naglasili primeri obuke koje su prethodni modeli pogrešno klasifikovali. U nekim slučajevima se pokazalo da Podsticanje daje bolju tačnost od Upakivanja, ali je takođe verovatno da će prekomerno uklopiti podatke na treningu. Daleko najčešća primena Podsticanja je AdaBoost, iako se navodi da neki noviji algoritmi postižu bolje rezultate.

U Podsticanju, jednaka težina (ujednačena raspodela verovatnoće) daje se uzorku podataka na treningu (recimo D1) u samoj početnoj fazi. Ovi podaci (D1) se zatim daju osnovnom učeniku (recimo L1). Pogrešno klasifikovanim primercima L1 dodeljuje se težina veća od tačno klasifikovanih primera, ali imajući na umu da će ukupna raspodela verovatnoće biti jednaka 1. Ovi podsticani podaci (recimo D2) se zatim daju drugom osnovnom učeniku (recimo L2) i tako dalje. Rezultati se zatim kombinuju u obliku glasanja.

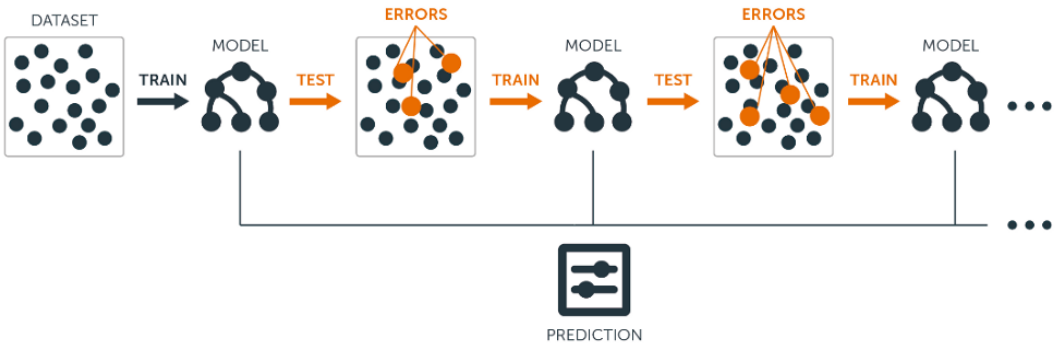
Podsticanje je sekvencijalni proces, gde svaki sledeći model pokušava da ispravi greške prethodnog modela. Sledeći modeli zavise od prethodnog modela. Takođe poznat kao podsticanje hipoteze, odnosi se na bilo koju metodu ansambla koja može kombinovati slabe učenike u jače. To je sekvencijalni proces gde svaki sledeći model pokušava da ispravi greške svog prethodnika.

Svaki model ne bi imao dobre performanse na čitavom skupu podataka. Međutim, oni imaju odlične performanse u delovima skupa podataka. Otuda, od Podsticanja možemo očekivati da će svaki model zapravo doprineti poboljšanju performansi celokupnog ansambla.

U sledećim koracima je objašnjeno kako tehnika Podsticanja funkcioniše:

1. Podskup se kreira iz originalnog skupa podataka.
2. U početku su sve tačke podataka jednake težine.
3. Na ovom podskupu kreira se osnovni model.
4. Ovaj model se koristi za predviđanje celog skupa podataka.
5. Greške se izračunavaju pomoću stvarnih vrednosti i predviđenih vrednosti.
6. Posmatranja koja su netačno predviđena, poprimaju veću težinu.
7. Izrađuje se drugi model i prave se predviđanja na skupu podataka.
   (Ovaj model pokušava da ispravi greške iz prethodnog modela)
8. Slično tome, kreira se više modela, svaki ispravljajući greške prethodnog modela.
9. Konačni model (jak učenik) je težinska sredina svih modela (slabih učenika).

Dakle, algoritam Podsticanja kombinuje niz slabih učenika koji formiraju jakog učenika.



Slika 2.7 – Proces tehnike Podsticanja modela

Neki od poznatih algoritama Podsticanja su:

- AdaBoost
- Gradijentno Podsticanje (engl. *Gradient Boosting*)
- XGBoost (Ekstremno Gradijentno Podsticanje)
- Lako Gradijentno Podsticanje (engl. Light Gradient Boosting)
- CatBoost

2.3.1.3.1. Gradijentno Podsticanje

Gradijento Podsticanje je još jedan algoritam Mašinskog učenja od ostalih ansambl tehnika koji radi i za probleme regresije i klasifikacije. Gradijento Podsticanje koristi tehniku podsticanja, kombinujući niz slabih učenika da bi formirao jakog učenika. Regresiona stabla koja se koriste kao osnovni učenici, svako naredno stablo u nizu gradi se na greškama koje je izračunalo prethodno stablo.
Stabla Gradijentnog Podsticanja su generalizacija pojačanja na proizvoljne diferencijabilne funkcije gubitaka. Gradijento Podsticanje gradi model na sekvencijalni način.

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

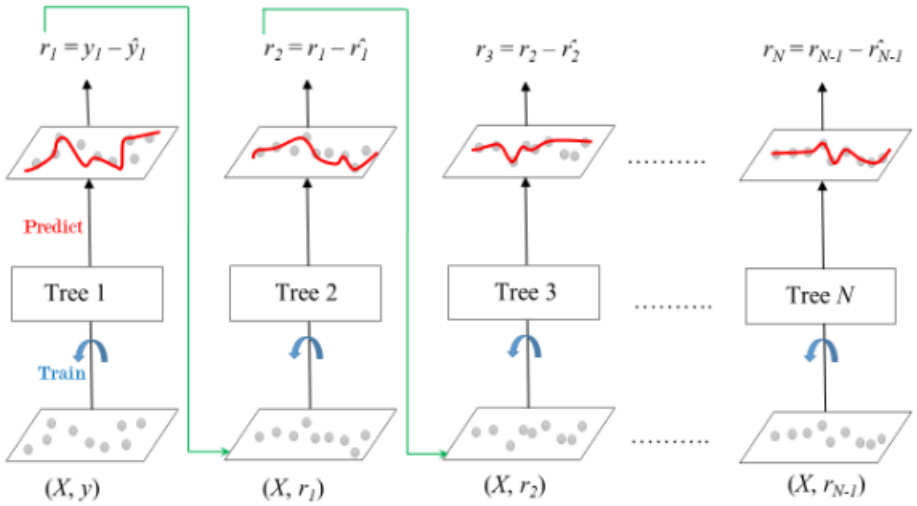
U svakoj fazi se bira stablo odluke hm (x) da se minimizira funkcija gubitka L s obzirom na trenutni model Fm-1 (x).

$$F_m(x) = F_{m-1}(x) + argmin \sum_{i=1}^{n} L\big(y_i, F_{m-1}(x_i) + h(x_i)\big)$$

Algoritmi za regresiju i klasifikaciju razlikuju se po tipu funkcije koja se koristi. Zapamtiti da je ključ Podsticanja modela učenje na prethodnim greškama. Gradijento Podsticanje uči na grešci - rezidualnoj grešci direktno, umesto da ažurira težine tačaka podataka i pravi novo predviđanje jednostavnim sabiranjem predviđanja svih stabala.
Postupak obučavanja modela metodom Gradijetnog Podsticanja se može opisati sledećim koracima:

1. **Izračunati prosek ciljne promenljive.** Kada se bavimo problemima regresije, započinjemo sa listom stabla koji je prosečna vrednost promenljive koju želimo da predvidimo. Ovaj list će se koristiti kao polazna osnova za približavanje tačnom rešenju u sledećim koracima.
2. **Izračunati ostatke.** Za svaki uzorak izračunavamo ostatak prema sledećoj formuli:
   ostatak = stvarna vrednost - predviđena vrednost
   Predviđena vrednost jednaka je srednjoj vrednosti izračunatoj u prethodnom koraku, a stvarna vrednost se može naći u koloni ciljne promenljive svakog uzorka.
3. **Konstruisati stablo odlučivanja.** Zatim gradimo stablo s ciljem predviđanja ostatka. Drugim rečima, svaki list će sadržati predviđanje vrednosti ostatka (a ne željene vrednosti). U slučaju da ima više ostataka nego lsitova, neki ostaci će završiti unutar istog lista. Kada se to dogodi, izračunavamo njihov prosek i smeštamo ih u list.
4. **Predvideti ciljnu vrednost svim stablima koje se nalaze u ansamblu.** Svaki uzorak prolazi kroz čvorove odlučivanja novoformiranog stabla dok ne dostigne zadati cilj. Ostatak u navedenom listu koristi se za predviđanje ciljne promenljive. Eksperimentisanjem se pokazalo da se malim inkrementalnim koracima ka rešenju postiže uporediv šum sa nižom ukupnom varijansom (manja varijansa dovodi do bolje tačnosti na uzorcima izvan trening podataka). Stoga, da bismo sprečili prekomerno obučavanje, uvodimo hiperparametar koji se naziva „stopa učenja“ (engl. „*learning rate*“). Kada predviđamo, svaki ostatak se množi sa stopom učenja. To nas primorava da koristimo više stabala odlučivanja, pri čemu svako stablo čini mali korak ka konačnom rešenju.
5. **Izračunati nove ostatke.** Izračunavamo novi skup ostataka oduzimajući stvarne vrednosti ciljne promenljive od predviđanja datih u prethodnom koraku. Ostaci će se zatim koristiti za listove sledećeg stabla odlučivanja kako je opisano u koraku 3.
6. **Ponavljati korake 3 do 5 dok se broj iteracija ne podudara sa brojem navedenim hiperparametrom (tj. brojem estimatora).**
7. **Jednom obučeni, koristiti sva stabla u ansamblu da se napravi konačno predviđanje vrednosti ciljne promenljive.** Konačno predviđanje biće jednako srednjoj vrednosti koja je izračunata u prvom koraku, plus svi ostaci koji smo dobili predviđanjem stabala koje čine šumu i množenjem tog predviđenog ostatka sa stopom učenja.

Tehnika Stabla Gradijentnog Podsticanja (engl. *Gradient Boosted Trees*) koristi kao osnovnog bazičnog učenika CART (Klasifikaciona i Regresiona stabla).
Ansambl se sastoji od *N* stabala. Stablo „Tree1“ se obučava koristeći matricu karakteristika *X* i oznake *y*. Predviđanja označena sa *y1* (^) koriste se za određivanje grešaka ostataka *r1* na skupu treninga. Stablo „Tree2“ se zatim obučava koristeći matricu karakteristika *X* i preostale greške *r1* Stabla „Tree1“ kao oznake. Tada se predviđeni rezultati *r1* (^) koriste za određivanje ostataka *r2*. Postupak se ponavlja dok se sva *N* stabla koja čine ansambl ne obuče.



Slika 2.8 – Regresija Stabla Gradijentnog Podsticanja

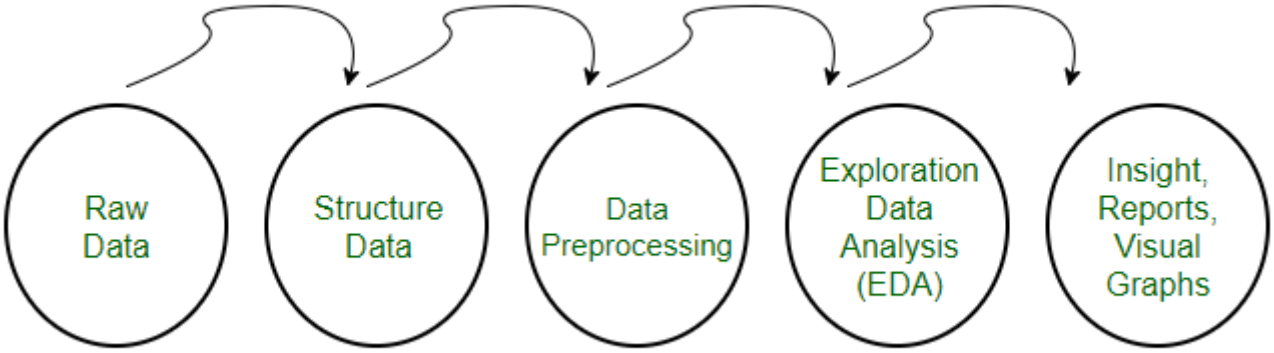
U ovoj tehnici se koristi važan parametar poznat kao „Skupljanje“ (engl. „*Shrinkage*“).
Skupljanje se odnosi na činjenicu da se predviđanje svakog stabla u ansamblu smanjuje nakon što se pomnoži sa stopom učenja (eta) koja se kreće od 0 do 1. Postoji kompromis između eta i broja estimatora, smanjivanje stope učenja treba nadoknaditi povećanjem estimatora kako bi se postigle određene performanse modela. Nakon što su sva stabla obučena, mogu se praviti predviđanja.
Uopštavanjem se dolazi do toga da algoritam Gradijentnog Podsticanja uključuje tri elementa:

1. Da se optimizuje funkcija gubitka.
2. Da slabi učenici formiraju predikcije.
3. Da aditivni model nadoda slabe učenike kako bi se minimizirala funkcija gubitka.

Funkcija gubitka koja će se koristiti zavisi od problema koji se rešava. Mora se razlikovati, ali podržane su mnoge standardne funkcije gubitka i mogu se samostalno definisati sopstvene funkcije gubitka.
Na primer, regresija može da koristi kvadratnu grešku, a klasifikacija logaritamski gubitak.
Prednost okvira Gradijentno Podsticanje je u tome što novi algoritam za podsticanje ne mora biti izveden za svaku funkciju gubitka koja bi možda želela da se koristi, već je generički dovoljan okvir da se može koristiti bilo koja raznolika funkcija gubitka.
Stablo je građeno na „pohlepan“ (engl. „*Greedy*“) način, birajući najbolje tačke razdvajanja na osnovu rezultata čistoće poput Ginija (engl. *Gini*) ili umesto toga da se gubitak svede na najmanju vrednost.
Uobičajeno je ograničiti slabe učenike na određene načine, kao što je maksimalan broj slojeva, čvorova, razdvajanja ili lsitova. Se želi osigurati da učenici ostanu slabi, ali i dalje mogu biti konstruirani na pohlepan način. Stabla se dodaju jedno po jedno, a postojeća stabla u modelu se ne menjaju.
Postupak gradijentnog spuštanja (engl. *Gradient Descent*) koristi se kako bi se gubici pri dodavanju stabla sveli na minimum. Umesto parametara, imamo slabe podmodele učenika ili tačnije stabla odlučivanja.
Nakon izračunavanja gubitka, da bi se izvršila procedura gradijentnog spuštanja, na model se mora dodati stablo koje smanjuje gubitak (tj. pratiti gradijent). To se radi parametrizacijom stabla, a zatim se modifikuju parametri stabla i krećući se u pravom smeru (smanjenjem ostalih gubitaka). Dodaje se fiksni broj stabala ili se trening zaustavlja kada gubici dostignu prihvatljivi nivo ili se više ne poboljšaju na spoljnom validacionom skupu podataka.

2.4. Pretprocesiranje podataka

Mašinsko učenje je 80% pretrocesiranje i 20% izrada modela. U realnom projektu Nauke o Podacima, pretprocesiranje podataka je jedna od najvažnijih stvari i jedan je od zajedničkih faktora uspeha modela, tj. ako postoji dobro pretprocesiranje podataka i fičer inženjering, taj model je verovatniji da će postići primetno bolje rezultate u poređenju sa modelom za koji podaci nisu dobro unapred obrađeni.
Pretprocesiranje se odnosi na transformacije primenjene na naše podatke pre nego što se unesu u algoritam. Prethodna obrada podataka je tehnika koja se koristi za pretvaranje sirovih podataka u čisti skup podataka. Drugim rečima, kad god se podaci prikupljaju iz različitih izvora, oni se prikupljaju u sirovom formatu, što nije adekvatno za analizu podataka.
Da bi se postigli bolji rezultati primenjenog modela u projektima mašinskog učenja, format podataka mora biti na odgovarajući način prilagođen. Nekom navedenom modelu mašinskog učenja potrebne su informacije u određenom formatu, na primer, algoritam Slučajnih šuma ne podržava nulte (eng. *null*) vrednosti, stoga se za izvršavanje algoritma Slučajnih šuma nultim vrednostima mora upravljati iz originalnog skupa sirovih podataka.
Sledeći aspekt je da skup podataka treba formatirati na takav način da se u jednom skupu podataka izvrši više algoritama za mašinsko učenje i duboko učenje i odabere se najbolji od njih.



Slika 2.9 – Proces obrade podataka

Kada se podaci sastoje od atributa koji različito skaliraju, mnogi algoritmi mašinskog učenja mogu imati koristi od ponovnog skaliranja atributa tako da svi imaju isto skaliranje. Ovo je korisno za optimizacione algoritme koji se koriste u jezgru algoritama mašinskog učenja poput gradijentnog spuštanja. Takođe je korisno za algoritme koji sadrže težinski ulaz kao što su Regresija, Neuronske mreže i algoritmi koji koriste mere daljine poput K-Najbližig suseda. Nakon ponovnog skaliranja, može se primetiti da su sve vrednosti u opsegu između 0 i 1. Ako je skup podataka pun NaN-ova (engl. *Not a Number*) i neupotrebljivih vrednosti, onda će sigurno i model izvoditi neupotrebljive vrednosti. Stoga je važno voditi računa o takvim vrednostima koje nedostaju. Jedan od načina popunjavanja nedostajućih vrednosti je popunjavanje srednjim vrednostima te kolone. Ako je broj redova koji imaju vrednosti koje nedostaju manji ili su podaci takvi da se ne savetuje popunjavanjevrednosti koje nedostaju, onda možemo da odbacimo redove koji nedostaju.
Za kategoričke fičere možemo se pobrinuti pretvaranjem u celobrojne. Postoje 2 uobičajena načina za to: **Labelarno enkodovanje** i „**One hot**“ **enkodovanje**. U Labelarnom enkodovanju, kategorije vrednosti možemo pretvoriti u numeričke oznake. U „One Hot“ enkodovanju pravimo novu kolonu za svaku jedinstvenu kategoričku vrednost, a vrednost je 1 za tu kolonu, ako je u stvarnom okviru podataka (engl. *dataframe*) ta vrednost tu, u suprotnom je 0. Standardizacijom možemo uzeti atribute sa Gausovskom raspodelom i različitim aritmetičkim sredinama i standardnim devijacijama i transformisati ih u standardnu Gausovskuu raspodelu sa sredinom 0 i standardnom devijacijom 1. Poslednji deo pretrpocesiranja podataka je normalizacija skupa podataka. Iz određenih eksperimenata je dokazano da modeli mašinskog učenja i dubokog učenja imaju mnogo bolji učinak na normalizovanom skupu podataka u poređenju sa skupom podataka koji nije normalizovan.

2.4.1. Transformacije podataka

Transformacija podataka je presudan deo mašinskog učenja, jer nestruktuirani podaci utiču na performanse algoritma mašinskog učenja (modela). Akcenat je stavljen na značenje transformacije. Ova napomena predstavlja statistička svojstva netransformiranih podataka sa pristupom transformacije.
Efikasno maksimizovanje performansi algoritma mašinskog učenja zahteva dobro transformisane podatke, otuda i potrebu za transformisanjem nebalansiranog skupa podataka.
Transformacija podataka čini podatke korisnim. Transformacija podataka je proces u kojem se preuzimaju podaci iz njihovog sirovog, izdvojenog i normalizovanog izvora i transformišu se u podatke koji su objedinjeni, dimenzionalno modelirani, denormalizovani i spremni za analizu. Bez odgovarajućeg tehnološkog steka, transformacija podataka može biti dugotrajna, skupa i iscrpna. Ipak, transformacijom podataka obezbediće se maksimalan kvalitet podataka koji je neophodan za sticanje tačne analize, što će dovesti do dragocenih uvida koji će na kraju podržati odluke zasnovane na podacima.
Izgradnja i obuka modela za obradu podataka je sjajan koncept, a više preduzeća je usvojilo ili planira da primeni mašinsko učenje za rukovanje mnogim praktičnim aplikacijama. Ali da bi modeli mogli da uče iz podataka kako bi dali dragocena predviđanja, sami podaci moraju biti organizovani tako da njihova analiza daje dragocene uvide. Na slici ispod su prikazane metode koje obuhvata transformacija podataka.

Slika 2.10 – Metode Transformacije podataka

Skaliranje je neophodan korak koji prethodi modeliranju. To je u osnovi zato što se podaci iz stvarnog sveta uvek pojavljuju u različitim razmerama. Na primer, skup podataka o stanovanju sadrži karakteristike poput veličine sobe (u kvadratnim metrima), cene (u USD) ili starosti kuće (u godinama). Skaliranje skupa podataka pomaže u smanjenju širenja ili razlike između ovih karakteristika.
Standardizacija je korisna za poređenje promenljivih izraženih u različitim jedinicama. Standardizovane vrednosti su često između 0 i 1 i bez jedinice mere.

$$z = \frac{x - \bar{x}}{\sigma}$$

Normalizacija je takođe koristan postupak skaliranja karakteristika, preusmeravanjem opsega karakteristika na skalu [0,1] ili [-1, 1].

$$z = \frac{x - min\ (x)}{max\ (x) - min\ (x)}$$

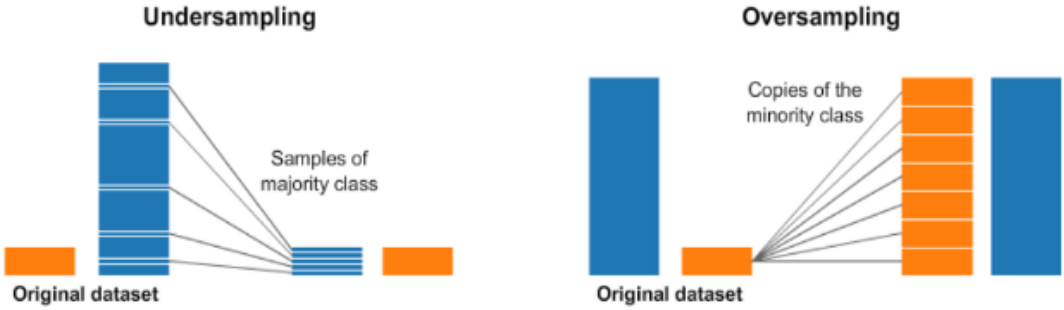
Binarizacijom podatke možemo transformisati pomoću binarnog praga. Sve vrednosti iznad praga označene su sa 1, a sve jednake ili niže označene su sa 0. Može biti korisno kada se od verovatnoća žele napraviti diskretne vrednosti. Takođe je korisno kada se radi fičer inženjering dodaju se novi fičeri koji ukazuju na nešto značajno. Uklanjanje srednje vrednosti sa radi za svaki fičer tako da je centriran na nulu. Srednje uklanjanje pomaže u uklanjanju bilo kakvih šumova iz atributa.

2.4.2. Uzorkovanje

Neuravnoteženost podataka u mašinskom učenju odnosi se na nejednaku distribuciju klasa unutar skupa podataka. Ovaj slučaj se sreće uglavnom u zadacima klasifikacije u kojima distribucija klasa ili oznaka u datom skupu podataka nije ujednačena. Jednostavna metoda za rešavanje ovog problema je metoda ponovnog uzorkovanja dodavanjem instanci manjinskoj klasi ili uklanjanjem instanci iz većinske klase. Ustanovljene su dve široko usvojene tehnike uzorkovanja: **Preuzorkovanje** (engl. *oversampling*) dodavanjem novih instanci i **Poduzorkovanje** (engl. *undersampling*) uklanjanjem postojećih instanci. Jedno od ključnih otkrića je primećivanje da preuzorkovanje ima bolje rezultate od poduzorkovanja za različite klasifikatore i postiže veće ocene u različitim metrikama vrednovanja.
Uzorkovanje se vrši da bi se iz uzoraka izvukli zaključci o populacijama i omogućava nam da utvrdimo karakteristike populacije direktnim posmatranjem samo dela (uzorka) populacije.
Tri glavne prednosti uzorkovanja su:

1. Izbor uzorka zahteva manje vremena nego odabir svake instance u populaciji.
2. Izbor uzorka je isplativa i efikasna metoda.
3. Analiza uzorka je manje glomazna i praktičnija od analize celokupne populacije.

I Preuzorkovanje i Poduzorkovanje uključuju uvođenje šuma radi odabira više uzoraka iz jedne klase nego iz druge, kako bi se nadoknadila neravnoteža koja je ili već prisutna u podacima, ili bi se mogla razviti ako je uzet potpuno slučajan uzorak. Krajnji rezultat Preuzorkovanja i Poduzorkovanja je stvaranje balansiranog skupa podataka. Mnoge tehnike mašinskog učenja, poput neuronskih mreža, daju pouzdanija predviđanja od obuke sa uravnoteženim podacima. Međutim, određene analitičke metode, posebno linearna regresija i logistička regresija, nemaju koristi od balansiranog pristupa.
Preuzorkovanje se obično koristi češće od Poduzorkovanja, posebno kada detaljni podaci tek treba da se prikupe anketom, intervjuom ili na neki drugi način. Poduzorkovanje se primenjuje mnogo ređe.
Prekomerno mnoštvo već prikupljenih podataka postalo je problem samo u eri „Velikih Podataka“, a razlozi za korišćenje Poduzorkovanja uglavnom su praktični i povezani sa troškovima resursa. Konkretno, iako je za izvođenje valjanih statističkih zaključaka potrebna odgovarajuća velika količina uzorka, podaci se moraju očistiti pre nego što se mogu koristiti.



Slika 2.11 – Razlika između Preuzorkovanja i Poduzorkovanja

U klasifikacionim problemima Preuzorkovanje se deli na sledeće tehnike:

- Slučajno Preuzorkovanje
- SMOTE (engl. *Synthetic Minority Oversampling Technique*)
- ADASYN (engl. *The Adaptive Synthetic Sampling Approach*)

U klasifikacionim problemima Poduzorkovanje se deli na sledeće tehnike:

- Slučajno Poduzorkovanje
- Klaster uzorkovanje
- Tomek veze (engl. *Tomek links*)
- Poduzorkovanje učenjem ansambla

Moguće je kombinirati tehnike preuzorkovanja i poduzorkovanja u hibridnu strategiju. Uobičajeni primeri uključuju veze SMOTE i Tomek ili SMOTE i ENN (engl. *Edited Nearest Neighbors*).

2.4.3. Izbor atributa

Sa nedavnim razvojem velikih podataka, dobio se veći pristup visokodimenzionalnim podacima. Shodno tome, performanse modela mašinskog učenja su se znatno poboljšale. S druge strane, postoje značajno beskorisni atributi koji se često prikupljaju ili generišu od različitih senzora i metoda. Ovi nepotrebni atributi ne samo da utiču na tačnost modela, već mogu zahtevati i mnogo računarskih resursa.
Stoga je izbor karakteristika kritičan proces u bilo kom cevovodu za mašinsko učenje, dizajniran da ukloni nebitne, suvišne i bučne karakteristike i sačuva mali podskup karakteristika iz primarnog prostora karakteristika.
Tehnike izbora atributa se koriste iz sledećih razloga:

- Pojednostavljenje modela kako bi se lakše interpretirali od strane istraživača/korisnika,
- Kraće vreme obučavanja modela,
- Da bi se izbegli problemi sa visokom dimenzionalnošću
- Pojačava se generalizacija smanjenjem preobučenosti (formalno, smanjenje varijanse).

U mašinskom učenju i nauci podataka uopšte, odabir atributa (poznat i kao izbor promenljivih, izbor atributa ili izbor podskupa) je postupak kojim naučnik podataka automatski (engl. *Data Scientist*) ili ručno bira podskup relevantnih karakteristika za upotrebu u izgradnji modela mašinskog učenja.
U stvari, to je jedan od osnovnih koncepata mašinskog učenja koji ima ogroman uticaj na performanse modela, jer je ključ za stvaranje pouzdanih modela mašinskog učenja.
S obzirom na skup funkcija, proces će odabrati najbolji podskup atributa koji su najvažniji i koji imaju veliki doprinos u vreme predviđanja.
Algoritam odabira atributa može se videti kao kombinacija tehnike pretraživanja za predlaganje novih podskupova obeležja, zajedno sa merom evaluacije koja ocenjuje različite podskupove atributa.
Najjednostavniji algoritam je testiranje svakog mogućeg podskupa atributa pronalaženjem onog koji minimizira stopu grešaka. Ovo je iscrpna pretraga prostora i računski je nerešiva za sve skupove atributa, osim za najmanji. Izbor metrike evaluacije u velikoj meri utiče na algoritam i upravo te metrike evaluacije razlikuju tri glavne kategorije algoritama za izbor karakteristika:

1. **Metode koje koriste omotače** (engl. *Wrapper methods*). Metode omotača koriste prediktivni model za ocenjivanje podskupova atributa. Svaki novi podskup koristi se za obuku modela, koji se testira na „Hold-out“ skupu. Brojanje grešaka napravljenih na tom „Hold-out“ skupu (stopa grešaka modela) daje rezultat za taj podskup. Kako metode omotača obučavaju novi model za svaki podskup, one su vrlo računski intenzivne, ali obično pružaju najbolje karakteristike za taj tip modela ili tipični problem.
2. **Metode koje korsite filtere** (engl. *Filter methods*). Metode filtriranja koriste proksi meru umesto stope grešaka za ocenjivanje podskupa atributa. Ova mera je izabrana za brzo izračunavanje, a istovremeno beleži korisnost skupa atributa. Uobičajene mere uključuju međusobnu informaciju, tačkaste međusobne informacije, Pearsonov proizvod-momenat koeficijent korelacije, algoritmi bazirani na olakšicama, udaljenost među / unutar klase ili ocene testova značajnosti za svaku klasu / atribut kombinaciju. Filteri su obično računski manje intenzivni od omotača, ali proizvode skup karakteristika koji nije prilagođen određenoj vrsti prediktivnog modela.
3. **Ugrađene metode** (engl. *Embedded methods*). Ugrađene metode su sveobuhvatna grupa tehnika koje izvršavaju izbor atributa kao deo procesa konstrukcije modela. Primer ovog pristupa je LASSO metoda za konstruisanje linearnog modela, koja regresione koeficijente kažnjava kaznom L1, smanjujući mnoge od njih na nulu. Sve atribute koji imaju koeficijente regresije koji nisu nulti su 'izabrani' pomoću algoritma LASSO.

U tradicionalnoj regresionoj analizi, najpopularniji oblik izbora atributa je stepenasta regresija, koja je jedna od metoda omotača. To je pohlepni algoritam koji dodaje najbolji atribut (ili briše najgori atribut) u svakoj rundi. Glavno kontrolno pitanje je odlučivanje kada zaustaviti algoritam. U mašinskom učenju, to se obično radi unakrsnom validacijom. U statistici su neki kriterijumi optimizovani. To dovodi do nezaobilaznog problema ugnježdavanja.

2.5. Mere performansi klasifikacionog sistema

Ako se ne procene performanse određenog modela mašinskog učenja, nikada se neće znati da li je model koristan ili ne, u surpotnom se model može nadgledati ceo dan bez konkretnog efekta i zaključka o evaluaciji.
Ideja o izgradnji modela mašinskog učenja deluje na principu konstruktivne povratne sprege. Izrađuje se model, dobijaju se povratne informacije od pokazatelja, unosite poboljšanja i nastavljate dok ne postignete željenu tačnost. Metrike evaluacije objašnjavaju performanse modela. Važan aspekt metrike evaluacije je njihova sposobnost da razlikuju rezultate modela.
Može se videti puno analitičara i ambicioznih naučnika za podatke koji se čak ni ne trude da provere koliko je njihov model robustan. Kada završe sa izradom modela, užurbano mapiraju predviđene vrednosti na neviđenim podacima. Ovo je pogrešan pristup.
Jednostavna izgradnja prediktivnog modela ne bi trebala biti motiv nijednog profesionalnog naučnika za podatke. Radi se o stvaranju i odabiru modela koji daje visoku tačnost podataka iz uzorka. Stoga je ključno proveriti tačnost modela mašinskog učenja pre izračunavanja predviđenih vrednosti.
U ovoj industriji se razmatraju različite vrste metrika kako bi se ocenili modeli. Izbor metrike u potpunosti zavisi od vrste modela i plana implementacije modela. Međutim, bilo koji dati model ima nekoliko ograničenja u zavisnosti od distribucije podataka. Nijedan od njih ne može biti potpuno tačan, jer su to samo procene (čak i ako su na „steroidima“). Ova ograničenja su popularno poznata pod nazivom šum i varijansa. Model sa velikim šumom će pojednostaviti ne obraćajući previše pažnje na tačke treninga (npr .: u Linearnoj regresiji, bez obzira na distribuciju podataka, model će uvek polagati linearnu vezu).
Model sa velikom varijansom ograničiće se na podatke na treningu ne generališući test tačke koje ranije nije video (npr .: Slučajna šuma sa *max_depth* = None).
Problem se javlja kada su ograničenja suptilna, kao kada se mora birati između algoritma slučajnih šuma i algoritma gradijentnog podsticanja ili između dve varijacije istog algoritma stabla odlučivanja. Oboje će imati veliku varijansu i nizak šum. Ovde dolazi do izražaja izbor i ocena performansi modela.
Modeli se mogu proceniti pomoću više metrika. Međutim, pravi izbor evaluacione metrike je presudan i često zavisi od problema koji se rešava. Jasno razumevanje širokog spektra metrika može pomoći ocenjivaču da se većom verovatnoćom, brže i efikasnije pronađe odgovarajuće poklapanje problema i metrike. Za mere performansi klasifikacionih problema se korsite sledeće metrike:

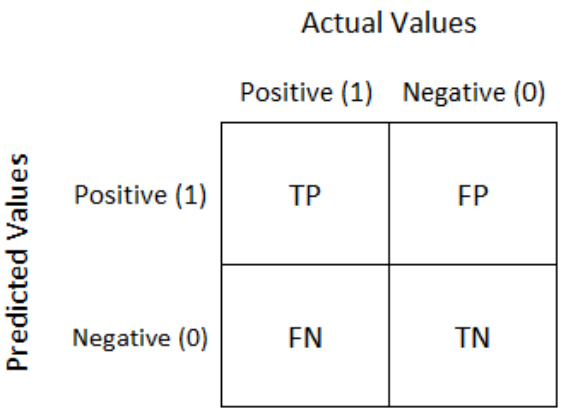
- Matrica konfuzije (engl. *Confusion Matrix*),
- ROC kriva,
- Grafikoni dobitka i postizanja (engl. *Gain and Lift Charts*),
- Tačnost (engl. *Accuracy*),
- Preciznost (engl. *Precision*),
- Odziv (engl. *Recall*) ili kompletnost – odnosi se na stopu tačno pozitivinh,
- Specifičnosti (engl. *Specificity*) – odnosi se na stopu tačno negativnih,
- F1 skor,
- Log gubitak (engl. *Log Loss*),
- Kolmogorov-Smirnov (K-S) grafikon

U višeklasnim problemima za razliku od procesa binarnih problema klasifikacije, ne mora da se odabere prag ocene verovatnoće da bi se predviđalo. Predviđeni odgovor je klasa (tj. Oznaka) sa najvećim predviđenim rezultatom. U nekim slučajevima će se možda želeti korišćenje predviđenih odgovora samo ako je predviđen sa visokim rezultatom. U ovom slučaju može se odabrati prag na predviđenim rezultatima na osnovu kojeg će se prihvatiti predviđeni odgovor ili ne. Tipične metrike koje se koriste u višeklasnim problemima su iste kao i metrike korišćene u slučaju binarne klasifikacije. Metrika se izračunava za svaku klasu tretirajući je kao binarni problem klasifikacije nakon grupisanja svih ostalih klasa kao da se radi o drugoj klasi u binarnoj klasifikaciji. Tada se binarna metrika usrednjava za sve klase da bi se dobio ili makro prosek (tretiraju svaku klasu jednako) ili težinska prosečna (težinska učestalost klase) metrika.

2.5.1. Matrica konfuzije

Na polju mašinskog učenja i posebno problema statističke klasifikacije, matrica konfuzije, poznata i kao matrica grešaka, je specifičan raspored tabele koji omogućava vizuelizaciju performansi algoritma, obično Nadgledanog učenja (u učenje bez nadzora obično se naziva matrica podudaranosti). Svaki red matrice predstavlja instance u predviđenoj klasi, dok svaka kolona predstavlja instance u stvarnoj klasi (ili obrnuto). Naziv potiče iz činjenice da olakšava uvid u to da li sistem meša dve klase (tj. Često pogrešno označava jednu kao drugu). To je posebna vrsta nepredviđenih tabela, sa dve dimenzije („stvarna“ i „predviđena“) i identičnim skupovima „klasa“ u obe dimenzije (svaka kombinacija dimenzije i klase je promenljiva u tabeli nepredviđenih događaja).
Matrica konfuzije je izuzetno korisna za merenje Odziva, Preciznosti, Specifičnosti, Tačnosti i najvažnije AUC-ROC krive.
U prediktivnoj analitici, Matrica konfuzije je tabela sa dva reda i dve kolone koja izveštava o broju lažno pozitivnih, lažno negativnih, istinskih pozitivnih i istinskih negativnih vrednosti klasa. Ovo omogućava detaljniju analizu od pukog udela tačnih klasifikacija (tačnost). Tačnost će dovesti do obmanjujućih rezultata ako je skup podataka neuravnotežen; odnosno kada se brojevi posmatranja u različitim klasama veoma razlikuju. Na primer, ako je u podacima bilo 95 mačaka i samo 5 pasa, određeni klasifikator mogao bi sva zapažanja klasifikovati kao mačke. Ukupna tačnost bi bila 95%, ali detaljnije klasifikator bi imao stopu prepoznavanja (osetljivost) od 100% za klasu mačaka, ali stopu prepoznavanja od 0% za klasu pasa. Rezultat F1 je u tim slučajevima još nepouzdaniji i ovde bi doneo preko 97,4%, dok informisanost uklanja tako veliki šum i daje 0 kao verovatnoću informisane odluke za bilo koji oblik pogađanja (ovde uvek mačka koja pogađa).
Navedenim koracima ispod je prikazan proces računanja Matrice konfuzije:

1. Potreban vam je testni skup podataka ili skup podataka za validaciju sa očekivanim vrednostima ishoda.
2. Napravite predviđanje za svaki red u vašem skupu podataka za testiranje.
3. Od očekivanih ishoda i predviđanja računa se:
   3.1 Broj tačnih predviđanja za svaki razred.
   3.2 Broj netačnih predviđanja za svaki razred, koje je organizovao razred koji je predviđen.



Ilustracija 1Slika 2.12 – Matrica konfuzije

Ova matrica sadrži četiri polja koja imaju sledeća značenja:

1. Instinski pozitivni (TP) su one vrednosti koje su predviđene pozitivno i zaista su pozitivne.
2. Lažno pozitivni (FP) su one vrednosti koje su predviđene pozitivno, ali su zapravo negativne.
3. Instinski negativni (TN) su one vrednosti koje su predviđene negativno i zaista su negativne.
4. Lažno negativni (FN) su one vrednosti koje su predviđene negativno, ali su zapravo pozitivne.

Matrica konfuzije se takođe osim binarne klasifikacije može formirati i za višeklasnu klasifikaciju. U tom slučaju umesto 2 x 2 dimenzije bi morala biti N x N dimenzija ukoliko imamo N klasa za predikciju.

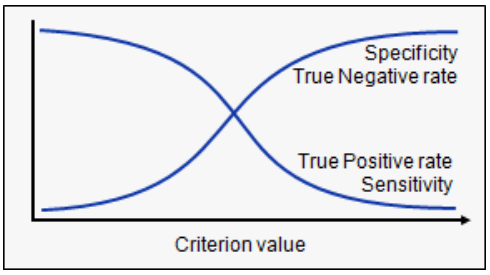
2.5.2. ROC kriva

Ovo je ponovo jedna od popularnih metrika koja se koristi u industriji. Najveća prednost korišćenja ROC krive je u tome što je ona nezavisna o promeni proporcije ispitanika. Ova izjava će biti jasnija u sledećim odeljcima. Pokušajmo prvo da razumemo šta je ROC (operativna karakteristika prijemnika) kriva. Ako pogledamo donju matricu konfuzije, primećujemo da za probabilistički model dobijamo različitu vrednost za svaku metriku.

Confusion Matrix		Target			
		Positive	Negative		
Model	Positive	a	b	*Positive Predictive Value*	a/(a+b)
	Negative	c	d	*Negative Predictive Value*	d/(c+d)
		Sensitivity	*Specificity*	**Accuracy** = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		

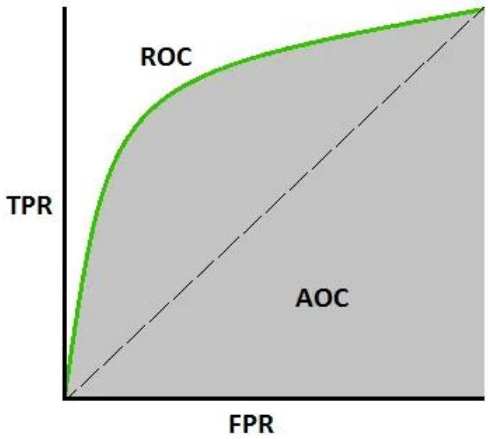
Slika 2.13 – Matrica konfuzije sa formulama

Dakle, za svaku senzitivnost dobijamo različitu specifičnost. Te dve se razlikuju na sledeći način:



Slika 2.14 – Odnos Senzitivnosti i Specifičnosti

ROC krive pokazuju kompromis između istinske pozitivne stope (TPR ili odziv) i lažno pozitivne stope (FPR). Utvrđeno je iz gornjih formula, TPR i FPR nisu ništa drugo nego zavisnost između senzitivnosti i (1 - specifičnost), tako da se na njih takođe može gledati kao na kompromis između senzitivnosti (istinska pozitivna stopa) i specifičnosti (lažna pozitivna stopa). Odnos između tačno pozitivne stope i lažno pozitivne stope poznata je kao ROC kriva. Kao što možete videti iz donjeg uzorka grafikona, za veće vrednosti TPR-a, takođe će se imati veće vrednosti FPR-a, što možda nije dobro. Dakle, sve je u pronalaženju ravnoteže između ove dve metrike. Dobra ROC kriva je ona koja dodiruje gornji levi ugao grafikona; pa što je veća površina ispod krive (AUC) od ROC krive, to je bolji vaš model.



Slika 2.15 – AUC – ROC kriva

AUC (Area under Curve) je druga vrsta metrike. Ona meri sposobnost modela da predvidi veći rezultat za pozitivne primere u poređenju sa negativnim primerima. Nezavisna je od izabranog praga, iz AUC metrike može se steći osećaj performansi predviđanja modela bez odabira praga

2.5.3. Grafikoni dobitka i podizanja

Grafikoni dobitka i podizanja su alati koji procenjuju performanse modela baš kao i matrica konfuzije i ROC kriva, ali sa suptilnom, ali značajnom razlikom. Matrica konfuzije određuje performanse modela na celoj populaciji ili na celom skupu testova, dok grafikoni dobitka i podizanja procenjuju model na delovima cele populacije. Zbog toga imamo skor (osa y) za svaki procenat populacije (osa x).
Grafikoni podizanja mere poboljšanje koje model donosi u poređenju sa slučajnim predviđanjima. Poboljšanje se naziva „Podizanje“ (engl. „*lift*“).
Grafikon dobitka i podizanja uglavnom se bavi proverom redosleda verovatnoće. Evo koraka za izgradnju grafikona za dobitak i podizanje:

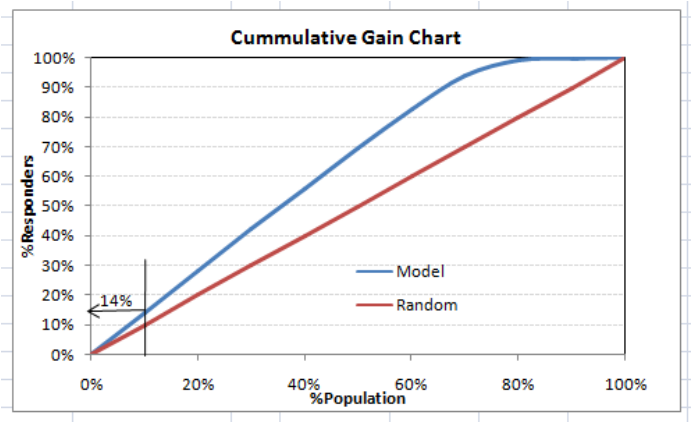
1. Izračunati verovatnoću za svako posmatranje.
2. Poređati ove verovatnoće u opadajući redosled.
3. Graditi decile sa svakom grupom koja ima skoro 10% posmatranja.
4. Izračunajte stopu odgovora na svakom decilu za dobar, loš i ukupan (odziv).

Nakon ova četiri koraka iznad se dobija sledeća tabela iz koje treba da se naprave grafikoni dobitka i podizanja:

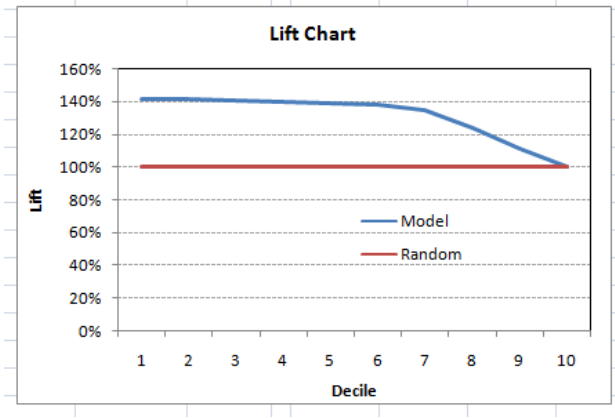
Lift/Gain	Column Labels			%Rights	%Wrongs	%Population	Cum %Right	Cum %Pop	Lift @decile	Total Lift
Row Labels	0	1	Grand Total	0%	0%	0%	0%	0%		
1		543	543	14%	0%	10%	14%	10%	141%	141%
2	2	542	544	14%	0%	10%	28%	20%	141%	141%
3	7	537	544	14%	0%	10%	42%	30%	139%	141%
4	15	529	544	14%	1%	10%	56%	40%	137%	140%
5	20	524	544	14%	1%	10%	69%	50%	136%	139%
6	42	502	544	13%	3%	10%	83%	60%	130%	138%
7	104	440	544	11%	7%	10%	94%	70%	114%	134%
8	345	199	544	5%	22%	10%	99%	80%	52%	124%
9	515	29	544	1%	32%	10%	100%	90%	8%	111%
10	540	5	545	0%	34%	10%	100%	100%	1%	100%
Grand Total	1590	3850	5440							

Slika 2.16 – Tabela za grafikon dobitka i podizanja

Ovo je vrlo informativna tabela. Grafikon kumulativnog dobitka je grafikon između kumulativnog procenta dobrih odziva i kumulativnog procenta populacije. Kriva podizanja je grafikon između ukupnog dizanja i procenta populacije. Treba imati na umu da za slučajni model ovo uvek ostaje na nivou od 100%. Na slici ispod je prikaz grafikona kumulativnog dobitka (levo) i grafikon podizanja (desno) za ovaj slučaj:



Slika 2.17 – Grafikon kumualtivnog dobitka



Slika 2.18 – Grafikon podizanja

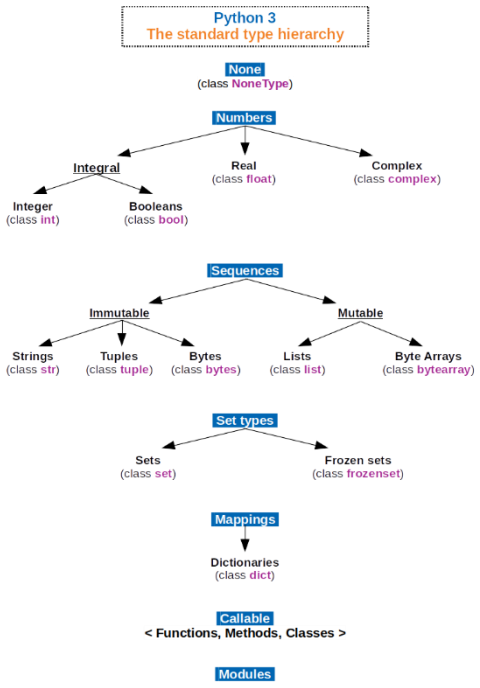
Grafikon kumulativnog dobitka govori koliko dobro model razdvaja dobre odzive od loših odziva. Na primer, prvi decil, koji međutim ima 10% stanovništva, ima 14% ispitanika dobrog odziva. To znači da je prisutno povećanje od 140% na 1. decilu. Grafikon podizanja kaže da model dobro radi do 7. decila. Bilo koji model sa podizanjem decila iznad 100% do minimalno 3. decila i maksimuma 7. decila je dobar model.

3. Implementacija

Primena Mašinskog učenja u oblasti predikcije ili skorovanja kredinog rizika ili baknrotstva određene kompanje, preduzetnika ili fizičkog lica dobija na sve većem značaju. S obzirom da je sve veća količina podataka, sve je više bamaka ili distributivnih kompanija koji odobravaju kredit svojim klijentima.
Sve je više istraživanja koja se bave ovom oblašću. U zavisnosti od podataka koji su prikupljeni za obučavanje modela i tipa predikcija, različiti algoritmi su među favoritima. Dok sa jedne strane preovladavaju algoritmi ansambla stabala odlučivanja, a posebno algoritimi Podsticanja, sa druge strane su se izuzetno dobro pokazale Duboke neuronske mreže, što i ne čudi s obzirom da su ove mreže u sve više oblasti dominantnije od ostalih tipova modela po pitanju performansi. Pretpostavlja se da je razlog tako velikog prodora Dubokih neuronski mreža taj što su dizajnirane po analogiji ljudskog mozga (konkretno Vizuelnog korteksa u mozgu čoveka).
U ovom radu je primenjen model Gradijentnog Podsticanja ansambla stabala odlučivanja. S obzirom da je to metoda koja se koristi u klasifikaciji nad označenim podacima (Nadgledano učenje), a podaci koji su prikupljeni nisu označeni ciljnim atributom, bilo je poželjno prevesti Nenadgledano učenje u Nadgledano nekom formulom, heuristikom. To se postiglo upotrebom popularne formule u finansijama za predviđanje bankrotstva takozvanom „Altman Z skor“ (engl. „*Altman Z score*“) formulom. Ova formula kao parametre uzima određene bilanse stanja i uspeha koje množeći sa određenim koeficijentima i međusobnim ostalim računskim operacijama daje decimalnu vrednost ovog Z skora. Vrednosti Altman Z skora se kreću u određenom ospegu. Taj opseg se morao podeliti na 4 jednaka decimalna intervala s obzirom da za ovaj problem kreditnog rizika nam trebaju 4 klase za određeni iznos uzet u vidu kredita za robu: *Nema rizika, Nizak rizik, Srednji rizik* i *Visok rizik*. Pomoću ove formule se mogu svakom kupcu dodeliti vrednosti Altman Z skora za četiri iznosa u vidu kredita (500 000 din, 2 000 000 din, 4 000 000 din i 6 000 000 din). Neophodno će biti da se za sve 4 navedene sume obuče 4 različita modela, jer će svaki Z skor biti drugačiji za određenu sumu.
Evaluacija modela se može obaviti matricom konfuzije, ukupnom tačnošću modela na validacionom skupu, kao i ROC krivom. Performanse se mogu unaprediti raznim načinima, mada ne značajno s obzirom da početna tačnost postiže već zadovoljavajući rezultat.
Ono što predstavlja veliki problem u obučavanju su nedostajuće vrednosti, kojih je ovom skupu podataka mnogo. Sa nedostajućim vrednostima sa bilansima stanja i uspeha postoji način da se izborimo a da sačuvamo podatke, međutim problem predstavlja činjenica da je ovih nedostajućih vrednosti ogroman broj. Na nešto malo iznad 8 000 instanci postoji čak preko 5 000 instacni koji sadrže nedostajuće vrednosti za bilanse stanja i bilanse uspeha. Te instance se mogu odbaciti ili da se dodaju srednje vrednosti za te atribute kojima nedostaju vrednosti. U ovom radu su sve nedostajuće vrednosti zamenjene srednjom vrednošću te kolone. Obučno su nedostajuće vrednosti zastupljenje za celu instancu tamo gde su prisutne za jednu kolonu što je olakšavajuća okolnost.
Nakon što su nedostajuće vrednosti zamenjene, urađena je transformacija ključnih parametara za Altman Z skor formulu (definisanih bilansa stanja i bilansa uspeha). Svaka kolona za tri godine koje se nalaze u skupu podataka (2016, 2017, 2018) je zamenjena srednjom vrednošću za ove tri godine kako bi se u formuli upotrebio parameter koji bi se odnosi na ukupno poslovanje preduzeća u prethodne tri godine. Takođe je potrebno naglasiti da postoje tri verzije formule u zavisnosti od Pravne forme datog preduzeća. Nakon što su izračunate vrednosti Altman Z skora sledi računanje verovatnoće bankrota preduzeća koji je još značajnija vrednost od samog Altman Z skora.
Za svaku originalnu vrednost verovatnoće bankrota su izračunate dodatne četiri vrednosti za četiri iznosa koja se mogu uzeti kao krediti za kupovinu robe. Za svaku od ovih vrednosti mora se transoformisati u jednu od 4 kategorije rizika tako što će se verovatnoća bankrota preduzeća podeliti na ćeitir ustaljena intervala vrednosti gde će se vrednosti koje pripadaju određenom intervalu mapirati u istu kategoriju. Ovim problem se problem transformiše u multiklasifikacioni s obzirom da je i menadžmentu rizika lakše da tumači vrednosti kroz četiri zadane kategorije. Obučavajući model je XGBoost (Ekstremno Gradijentno Podsticanje) koje se obučava distribuirano i daje bolje performance od bazičnog Gradijentnog Podsticanja.

3.1. Python programski jezik

Python je interpretirani programski jezik opšte namene visokog nivoa. Python filozofija dizajna naglašava čitljivost koda svojom zapaženom upotrebom značajnog razmaka. Njegove jezičke konstrukcije i objektno orijentisani pristup imaju za cilj da pomognu programerima da napišu jasan, logičan kod za male i velike projekte.
Python se dinamički kuca i prikuplja smeće. Podržava više paradigmi programiranja, uključujući strukturirano (posebno proceduralno), objektno orijentisano i funkcionalno programiranje. Python se često opisuje kao jezik „sa baterijama“ zbog sveobuhvatne standardne biblioteke.
Python je stvoren krajem 1980-ih, a prvi put objavljen 1991. godine, Guido van Rossum kao naslednik programskog jezika ABC. Python 2.0, objavljen 2000. godine, predstavio je nove funkcije, poput razumevanja spiska i sistema za sakupljanje smeća sa brojanjem referenci, a ukinut je sa verzijom 2.7 u 2020. godini. Python 3.0, objavljen 2008. godine, bio je glavna revizija jezika koji nije u potpunosti kompatibilan sa unazad i mnogi Python 2 kodovi ne rade nepromenjeni na Python 3. Sa Python 2-inim krajem života, samo Python 3.6.x i novije verzije su podržane, a starije verzije i dalje podržavaju npr Windows 7 (i stari instalacioni programi koji nisu ograničeni na 64-bitni Windows).
Python interpreteri su podržani za uobičajene operativne sisteme i dostupni su za još nekoliko (a u prošlosti i za mnoge druge). Globalna zajednica programera razvija i održava CPython, besplatnu implementaciju referenci otvorenog koda. Neprofitna organizacija, Python Softvare Foundation, upravlja i usmerava resurse za razvoj Pythona i CPython-a. Trenutno je povezan sa Javom i Javaskriptom kao jednim od najpopularnijih programskih jezika na svetu.



Slika 3.1 – Podela tipova i struktura u Python programskom jeziku

Python je programski jezik sa više paradigmi. Objektno orijentisano programiranje i strukturirano programiranje su u potpunosti podržani, a mnoge njegove funkcije podržavaju funkcionalno programiranje i aspektno orijentisano programiranje (uključujući metaprogramiranje i metaobjekte (magijske metode)). Mnoge druge paradigme podržane su ekstenzijama, uključujući dizajn po ugovoru i logičko programiranje.
Python je zamišljen kao lako čitljiv jezik. Njegovo formatiranje je vizuelno neuredno i često koristi ključne reči na engleskom tamo gde drugi jezici koriste interpunkciju. Za razliku od mnogih drugih jezika, on ne koristi vitičaste zagrade za razdvajanje blokova, a tačka i zarez nakon izraza nisu obavezni. Ima manje sintaksičkih izuzetaka i posebnih slučajeva od C ili Pascal-a. Python koristi „duck“ tipiziranje i ima tipizirane objekte, ali netipizirana imena promenljivih. Ograničenja tipa se ne proveravaju u vreme kompajliranja; radije, operacije na objektu mogu propasti, što znači da dati objekat nije odgovarajućeg tipa.

3.1.1. scikit-learn biblioteka

Scikit-learn (ranije scikits.learn i poznat i kao sklearn) je besplatna softverska biblioteka za mašinsko učenje za programski jezik Python. Sadrži različite algoritme za klasifikaciju, regresiju i klasterovanje, uključujući mašine nosećih vektora, slučajne šume, pojačavanje gradijenta, k-srednjih i DBSCAN, a dizajniran je za interakciju sa Python numeričkim i naučnim bibliotekama NumPy i SciPy.
Projekat scikit-learn započet je kao scikits.learn, projekat „Google Summer of Code“, autora Davida Cournapeau-a. Njegovo ime potiče od ideje da je to „SciKit“ (SciPy Toolkit), posebno razvijeno i distribuirano nezavisno proširenje za SciPy. Prvobitnu bazu koda kasnije su prepisali drugi programeri. U 2010. godini Fabian Pedregosa, Gael Varokuauk, Alekandre Gramfort i Vincent Michel, svi iz Francuskog instituta za istraživanje računarskih nauka i automatizacije u Rockuencourt-u, Francuska, preuzeli su vođstvo nad projektom i objavili prvo javno izdanje 1. februara 2010. Od različitih scikit-ova, scikit-learn kao i scikit-slika opisani su kao „dobro održavani i popularni“ u novembru 2012. Scikit-learn je jedna od najpopularnijih biblioteka mašinskog učenja na GitHub-u.
Scikit-learn je u velikoj meri napisan u Pythonu i koristi numpy u velikoj meri za linearnu algebru i niz operacija visokih performansi. Dalje, neki osnovni algoritmi su napisani na Cython-u da bi poboljšali performanse. Mašine nosećih vektora su implementirane od Cython omotača oko LIBSVM; logistička regresija i linearne vektorske mašine za podršku sličnim omotom oko LIBLINEAR. U takvim slučajevima proširivanje ovih metoda pomoću Pythona možda neće biti moguće.
Scikit-learn se dobro integriše sa mnogim drugim Python bibliotekama, kao što su matplotlib i plotly za crtanje, numpy za vektorizaciju niza, pandas okviri za podatke, scipy i još mnogo toga.
Scikit-learn pruža niz nadgledanih i nenadgledanih algoritama učenja putem doslednog interfejsa u Python-u. Vizija biblioteke je nivo robusnosti i podrške potreban za upotrebu u proizvodnim sistemima. To znači duboku usredsređenost na probleme kao što su jednostavnost upotrebe, kvalitet koda, saradnja, dokumentacija i performanse. Iako je interfejs Python, c-biblioteke su podloga za performanse kao što su numpy za nizove i operacije matrice, LAPACK, LibSVM i pažljiva upotreba cython-a.
Biblioteka je usredsređena na modeliranje podataka. Nije fokusirana na učitavanje, manipulisanje i rezimiranje podataka. Za ove funkcije pogledati NumPy i Pandas.
Neke popularne grupe modela koje nudi scikit-learn uključuju:

- **Grupisanje** (engl. *Clustering*) : za grupisanje neobeleženih podataka kao što je KMeans.
- **Unakrsna validacija** (engl. *Cross Validation*): za procenu učinka nadgledanih modela na neviđenim podacima.
- **Skupovi podataka** (engl. *Datasets*): za testne skupove podataka i za generisanje skupova podataka sa specifičnim svojstvima za ispitivanje ponašanja modela.
- **Smanjenje dimenzionalnosti** (engl. *Dimensionality reduction*): za smanjenje broja atributa u podacima za sažimanje, vizuelizaciju i izbor karakteristika kao što je analiza glavnih komponenata.
- **Metode ansambla** (engl. *Ensemble methods*): za kombinovanje predviđanja višestrukih nadgledanih modela.
- **Izdvajanje atributa** (engl. *Feature extraction*): za definisanje atributa u slikovnim i tekstualnim podacima.
- **Izbor atributa** (engl. *Feature selection*): za prepoznavanje značajnih atributa od kojih će se stvoriti nadgledani modeli.
- **Podešavanje parametara** (engl. *Parameter Tuning*): za dobijanje maksimuma od nadgledanih modela.
- **Mnogostruko učenje** (engl. *Manifold learning*): Za sažimanje i prikazivanje složenih višedimenzionalnih podataka.
- **Nadgledani modeli** (engl. *Supervised models*): široka lepeza koja nije ograničena na generalizovane Linearne modele, Diskriminatorsku analizu, Naivni Bajes, Lenje metode, Neuronske mreže, Mašine Nosećih Vektora i Stabla odlučivanja.

3.1.2. xgboost biblioteka

XGBoost je softverska biblioteka otvorenog koda koja pruža okvir za pojačavanje gradijenta za C ++, Java, Python, R, Julia, Perl i Scala. Radi na Linuksu, Windows-u i macOS-u. Iz opisa projekta, cilj joj je pružiti „Biblioteku koja se može skalirati, prenositi i distribuirati pojačavanjem gradijenta (GBM, GBRT, GBDT)". Radi na jednoj mašini, kao i na distribuiranim okvirima za obradu Apache Hadoop, Apache Spark i Apache Flink. U poslednje vreme stekala je veliku popularnost i pažnju kao algoritam izbora za mnoge pobedničke timove takmičenja u mašinskom učenju.
XGBoost je u početku započeo kao istraživački projekat Tianki Chen-a kao deo grupe „Distributed (Deep) Machine Learning Community" (DMLC). U početku je počeo kao terminalna aplikacija koja se mogla konfigurisati pomoću datoteke za konfiguraciju libsvm. Postao je dobro poznat u takmičarskim krugovima ML-a nakon njegove upotrebe u pobedničkom rešenju Higgs-ovog izazova za mašinsko učenje. Ubrzo nakon toga, Python i R paketi su izgrađeni, a XGBoost sada ima implementacije paketa za Java, Scala, Julia, Perl i druge jezike. Ovo je biblioteku dovelo do više programera i doprinelo njenoj popularnosti među zajednicom Kaggle, gde se koristi za veliki broj takmičenja.
Ubrzo je integrisan sa nizom drugih paketa koji olakšavaju upotrebu u njihovim zajednicama. Sada je integrisan sa scikit-learn za korisnike Pythona i sa paketom caret za R korisnike. Takođe se može integrisati u okvire toka podataka kao što su Apache Spark, Apache Hadoop i Apache Flink koristeći apstrahovani Rabit i XGBoost4J. XGBoost je takođe dostupan na OpenCL-u za FPGA. Efikasnu, skalabilnu primenu XGBoost-a objavili su Tianki Chen i Carlos Guestrin.
Istaknute karakteristike XGBoost-a koje ga čine drugačijim od ostalih algoritama za pojačavanje gradijenta uključuju:

- Pametna kazna stabala (engl. *Clever penalization of trees*),
- Proporcionalno skupljanje listova (engl. *A proportional shrinking of leaf nodes*),
- Njutn podsticanje (engl. *Newton Boosting*),
- Dodatni parametar randomizacije (engl. *Extra randomization parameter*),
- Implementacija na pojedinačnim, distribuiranim sistemima i van jezgra računanja

XGBoost (Ektremno Gradijent Podsticanje) pripada porodici podsticajnih algoritama i u osnovi koristi okvir podsticanja gradijenta (GBM). To je optimizirana distribuirana biblioteka za pojačavanje gradijenta. XGBoost je jedan od najpopularnijih algoritama mašinskog učenja danas. Bez obzira na vrstu predviđenog zadatka; regresija ili klasifikacija. Poznato je da XGBoost pruža bolja rešenja od ostalih algoritama mašinskog učenja. Zapravo, od svog osnivanja, postao je „najsavremeniji" algoritam mašinskog učenja za bavljenje strukturiranim podacima.
Ono što čini XGBoost toliko popularnim je sledeće:

- **Brzina i perforamnse:** Izvorno napisan na jeziku C ++, relativno je brži od ostalih klasifikatora ansambala.
- **Osnovni algoritam je paralelizabilan:** Budući da je osnovni XGBoost algoritam paralelan, može iskoristiti snagu računara sa više jezgara. Takođe je paralelan na GPU-u i na različitim mrežama računara, što omogućava izvođenje obuke i na vrlo velikim skupovima podataka.
- **Dosledno nadmašuje druge metode algoritma:** Pokazao je bolje performanse na raznim skupovima podataka referentnih podataka o mašinskom učenju.
- **Širok izbor parametara za podešavanje:** XGBoost interno ima parametre za unakrsnu validaciju, regularizaciju, korisnički definisane funkcije cilja, vrednosti koje nedostaju, parametre stabla, API kompatibilan sa scikit-learn itd.

Ako se planira korišćenje XBoost na skupu podataka koji ima kategorijalne karakteristike, možda bi bilo dobro razmisliti o primeni nekog kodiranja (kao što je „One Hot" kodiranje) na takve funkcije pre nego što se obuči model. Takođe, ako u skupu podataka postoje neke vrednosti koje nedostaju, kao što je NaN, može se uraditi zaseban tretman za njih, ali i ne mora jer XGBoost može interno da obrađuje nedostajuće vrednosti. Na vlastiti sistem se može instalirati biblioteke xgboost koristeći pip install xgboost u konzoli.

3.2. Korišćeni podaci

Podaci koji se koriste u ovom radu su preuzeti nakon hakaktona gde je rađen slučaj za kompaniju Nelt koja je inače najveća distributivna kompanija u regionu. Tri skupa podataka predstavljaju kolekciju klijenata, faktura i faktura po proizvodima koje su klijenti poručivali od kompanije Nelt u vidu uzimanja kredita za robu. Četvrta tabela predstavlja finansijsku i vlasničku strkturu samih klijenata. Svi skupovi su .csv formata (engl. *comma separated values*). Detaljno se navode četiri skupa podataka i opisi njihovih atributa:

1. **Fakture i Proizvodi** je skup podataka koji predstavlja koja grupa proizvoda i u kojoj količini se kupovala po fakturi. Postoji više instanca jedne fakture ako se uzimalo više grupa proizvoda. Svi atributi su numerički osim grupe proizvoda koja je kategorički i prisutno je 1 506 147 instanci i sledećih 6 atributa:
    - **FakturaID** – jedinstveni identifikator svake facture,
    - **GrupaProizvoda –** šifra kategorije određene grupe proizvoda,
    - **Kolicina** – broj uzete količine proizvoda u pakovanjima,
    - **IznosNeto –** cena u dinarima bez PDV,
    - **PDV –** porez na potrošnju,
    - **IznossaPDV –** cena u dinarima sa uračunatim PDV.

2. **Fakture** je skup podataka koji daje uvid u podatke samih faktura i njihovog statusa sa klijentom koji je otvorio određenu fakturu. Jedan klijent može imati više faktura. Svi atributi su numeriči uključujući i datume osim Statusa Fakture koji je kategorički atribut. U skupu je prisutno 955 457 i sledećih 9 atributa:
    - **FakturaID –** jedinstveni identifikator svake facture,
    - **KupacID –** jedinstveni identifikator svakog kupca,
    - **Datum –** datum podnošenja zahteva fakture,
    - **Iznos –** ukupan iznos fakture u dinarama sa uračunatim PDV,
    - **ValutaDana -** Sa brojem dana valute plaćanja se definiše posle kojeg perioda (u danima) će biti zadnji rok za plaćanje neke robe ili usluge,
    - **DatumDospeca –** datum kada je klijent primio kredit fakture,
    - **PlaceniIznos –** plaćeni iznos u dinarima u trenutku zatvaranja,
    - **DatumZatvaranja –** datum zatvaranja fakture kao zahteva za kredit klijenta, uglavnom isti kao DatumDospeca, ili može kasniti nekoliko dana,
    - **StatusFakture –** status fakture može imati vrednosti “Plaćena”, “Delimično plaćena” ili “Otvorena” u zavisnosti od plaćenog iznosa fakture.

3. **Kupci** je skup podataka koji prikazuje najopštije podatke o klijentima koji su podizali fakture. Prisutna su 2 numerička atributa, jedan kategorički i jedan karakterni što je svega 4 atributa sa 2300 instanci klijenata. Atributi su sledeći:
    - **KupacID –** jedinstveni identifikator svakog kupca,
    - **KupacNaziv –** naziv klijenta (nije pravi naziv klijenta),
    - **Opstina –** opština na kojoj je klijent registrovan na teritoriji Srbije,
    - **BrojObjekata –** broj registrovanih objekata koje klijent poseduje.

4. **Finansijska i vlasnička struktura** je zaseban skup podataka koji sadrži i klijente koji su do sada podnosili zahtev za kredit i oni koji uopšte nisu i pritom ne postoji većina podataka za njih što predstavlja dodatan problem. Ceo skup u suštini sadrži proširene podatke o vlasništvu klijenata uz AOP šifre bilansa stanja i uspeha za 2016., 2017., i 2018. godinu u dinarskoj vrednosti. Prisutno je 8082 istance uz 644 atributa koji su sledeći:
    - **KupacID –** jedinstveni identifikator svakog kupca,

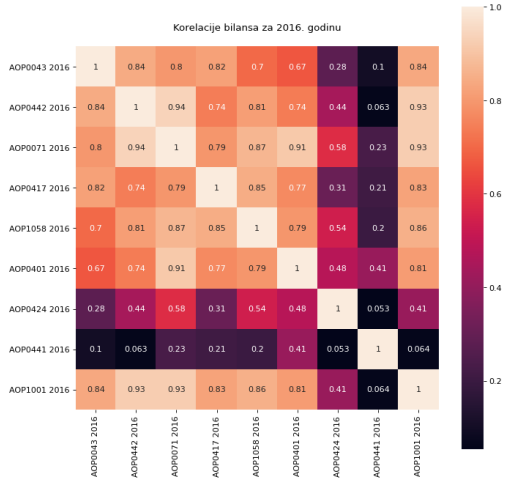
- **KupacNaziv –** naziv klijenta (nije pravi naziv klijenta),
- **Status kompanije –** status aktivnosti kompanije sa vrednostima: „Aktivno“,
- **Šifra delatnosti –** broj koji identifikuje delatnost kojom se klijent bavi,
- **Naziv delatnosti –** naziv delatnosti kojom se klijent bavi,
- **Pravna forma –** definiše kog je tipa društvo klijenta i može sadržati vrednosti: „Akcionarsko društvo“, „Društvo sa ograničenom odgovornošću“, „Ortačko društvo“, „Zadruga“ ili „Preduzetnik“.
- **Opština** - opština na kojoj je klijent registrovan na teritoriji Srbije,
- **Mesto -** mesto u kojem je klijent registrovan na teritoriji Srbije,
- **Kreditna_ocena –** ocena kreditne sposobnosti klijenta,
- **status_racuna –** da li je racun na koji se zahteva kredit, aktivan, blokiran ili je stanje nepoznato,
- **Trenutni broj dana u blokadi –** broj dana koliko je račun preduzeća u blokadi,
- **Trenutni iznos blokade** - iznos za koji su računi firme blokirani u trenutku provere i pokazuje aktuelno dugovanje firme po izvršnim rešenjima i nalozima,
- **Broj dana u blokadi u proteklih 24 meseca –** ukupan broj dana koliko je račun preduzeća proveo u blokadi u poslednjih godinu dana,
- **Ulazak u blokadu –** datum kada je klijent ušao u blokadu,
- **Godina bilansa 2018** – Godina objavljenog stanja u trenutku 2018. godine,
- **Set bilansa 2018 –** predstavlja vrstu skupa bilansa objavljenih u 2018. godini. U ovom skupu postoje širi i uži set bilansa. Uži skup obuhvata bilanse stanja, bilanse uspeha i statističke izveštaje. Širi skup obuhvata bilanse stanja, bilanse uspeha, statističke izveštaje, izveštaj o ostalom rezultatu, izveštaj o promenama na kapitalu i izveštaje o tokovima gotovine. Od toga da li preduzeće treba da ima širi ili uži skup bilansa zavisi koja je njegova Pravna forma, odnosno kog je tipa društvo preduzeća.
- **AOP0001 2016 … AOP1071 2016 –** dinarska vrednost šifre AOP za sve bilanse u 2016. godini,
- **AOP0001 2017 … AOP1071 2017** – dinarska vrednost šifre AOP za sve bilanse u 2017. godini,
- **AOP0001 2018 … AOP1071 2018 –** dinarska vrednost šifre AOP za sve bilanse u 2018. godini,

AOP atributi su ustanovljeni kao jedni od najbitnijih za predikciju kreditnog rizika s obzirom da oni direktno govore o stanju i prometu kapitala određenog preduzeća. Takođe je bitno napomenuti da se određeni bilanse izvode iz drugih preko formula, pa takođe i ti izvedeni izlazni bilansi mogu predstavljati ulazni atribut za izvođenje drugog i potencijalno još bitnijeg bilansa. Tako se dobija kaskadna veza među bilansa. Bilo bi bitno napomenuti da postoje efikasne formule za predikciju kreditnog rizika tojest baknrota preduzeća. Najpopularnije formule su **Altman Z skor formula** i Olsonova formula. U ovom radu se koristi Altman Z skor formula koja ima nekoliko verzija u zavisnosti od pravne forme preduzeća. Altman Z skor formula je detaljnije objašnjena u posebnoj sekciji u nastavku rada. Za sada je važno napomenuti da su uz pomoć te formule podaci od neoznačenih simulirano transformisani u označene tako što su se uz pomoć Altman Z skor formule generisale vrednosti za ciljnu promenljivu.

Od četiri skupa podataka navedena u ovom radu, model mašinskog učenja najviše značaja i informacija dobija iz poslednjeg skupa podataka vezanog za finansijsku i vlasničku strukturu. Iako je to činjenica, ne znači da ostala tri skupa podataka trebaju u potpunosti biti izostavljena. Iz njih su uzeti samo najbitniji atributi poput ukupno uzetog kredita i otplaćenog dela kredita, kao i broja objekata koje klijent poseduje. Neki atributi su izbačeni koje ne doprinose ili čak smanjuju performanse modela, dok su neki dodati kao značajni.

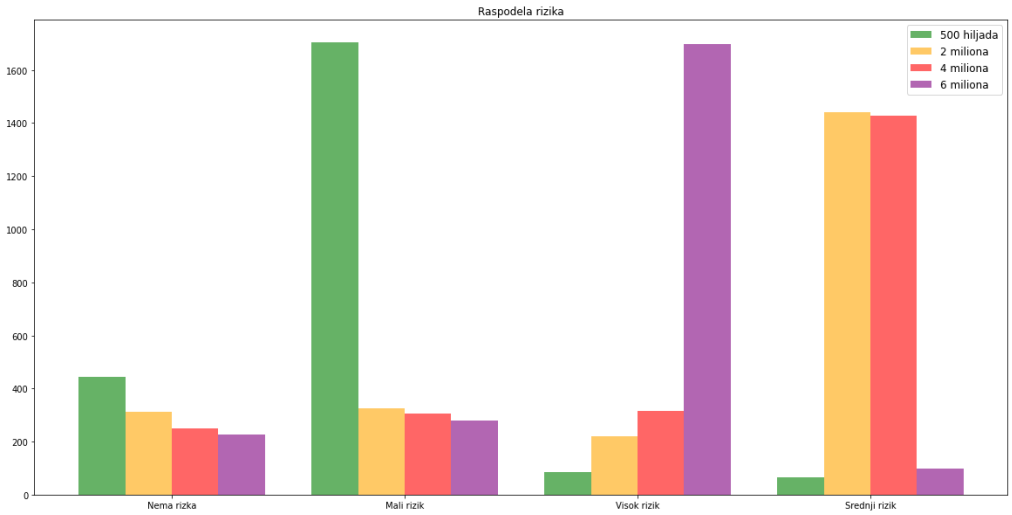
3.3. Eksplorativna analiza atributa

Eksplorativna analiza podataka je pristup analizi skupova podataka radi sumiranja njihovih glavnih karakteristika, često vizuelnim metodama. Statistički model se može koristiti ili ne, ali prvenstveno EDA (engl. Exploratory Data Analysis) služi za utvrđivanje šta podaci mogu reći izvan formalnog zadataka modeliranja ili testiranja hipoteza. John Tukei je promovisao Eksplorativnu analizu podataka kako bi podstakao statističare da istražuju podatke i verovatno formulisali hipoteze koje bi mogle dovesti do novog prikupljanja podataka i eksperimenata. EDA se razlikuje od početne analize podataka (IDA), koja se usko fokusira na proveru pretpostavki potrebnih za uklapanje modela i testiranje hipoteza, i rukovanje nedostajućim vrednostima i izradu transformacija promenljivih po potrebi. Prva analiza koja je urađena dok se još podaci nisu preveli u označene je analiza korelacija između bitnijih bilansa zasebno za sve tri godine. Za vizuelizaciju međusobnih korelacija korišćena je Heat mapa. Na slici ispod je prikazana Heat mapa korelacija između 9 bilansa koji su posle korišćeni u Altman Z skor fomruli. Korelacije su na ovoj slici analizirane za 2016. godinu. Analizirane su za svaku godinu posebno na isti način.



Slika 3.2 – Korelacije bilansa za 2016. godinu

Svetliji kvadrati pokazuju veću korelaciju između atributa. Ono što je bitno zaključiti je da atribute koji su u jakoj međusobnoj korelacji ne treba uključivati sve u model kao ulazne atrbiute, već bi bilo dobro pojednostaviti model, ali ovi atributi će se zadržati s obzirom da su oni baš promenljive za Z skor formulu. Nakon označavanja podataka uz pomoć formule, sledeća jako bitna analiza je raspodela klasa izlaznih promenljivih kojih u ovom slučaju ima 4 s obzirom da se predikcije vrše za 4 različita iznosa kredita.



Slika 3.3 – Raspodela rizika za 4 iznosa kredita

Na slici iznad je za svaki bin prikazan broj klijenata (y osa) za određenu klasu rizika (x osa) i bojom obeležen za određenu sumu (legenda u gornjem desnom uglu). Klase po iznosima nisu baš ravnomereno raspoređeni, osim kod izonsa od 2 i 4 miliona dinara gde su sve klase osim Srednjeg rizika sličnih razmera, što će dovesti do malo boljih rezultata obučavanja za ta dva iznosa.

3.3.1. Transformacija atributa

Transformacije atributa se mogu primeniti tamo gde tip i vrednosti atributa to dzvoljavaju. U skupu podataka za finansijsku i vlasničku strukturu se mogu primetiti podaci sa ''-'' vrednošću što označava da ne postoji podatak za određenog klijenta u datoj koloni što se može poistovetiti sa nedostajućom ''NaN'' vrednosti.
Takođe treba naglasiti da postoje dosta kategoričkih vrednosti koje bi se morale labelirano kodovati u cele brojeve za obučavanje modela gradijentnog podsticanja. Prva transformacija je izvršena sa nedostajućim vrednostima u skupu za finansijsku i vlasničku strukturu tako što su nedostajuće vrednosti uklonjene iz razloga što se radi o klijentima koji nisu nikad sarađivali sa Neltom i nemaju nikakav podatak osim sopstvenog naziva. Nakon uklanjanja nedostajućih redova iz skupa podataka, urađene su sledeće transformacije atributa:

- Iako su uklonjene nedostajuće vrednosti iz skupa podataka za finansijsku i vlasničku strukturu, postoje i dalje ponegde vrednosti koje nemaju eksplicitno nedostajuću vrednost ''NaN'', ali sadrže vrednosti ''-'' koje se mogu isto kotirati kao nedostajuće. Ove vrednosti su zamenjene srednjom vrednosti svake kolone bilansa. Ova transformacija se odnosi na sve atribute **AOP0001 2016 – AOP1071 2018**.
- Nakon toga su sve vrednosti bilansa (**AOP0001 2016 – AOP1071 2018)** koje su prethodno bile karakternog tipa iako su u pitanju numeričke vrednosti, pretvorene se u celobrojni int tip kako bi se mogle iskoristiti kao parametri za Altman Z skor formulu.
- Zatim je izvršeno usrednjavanje vrednosti bilansa (**AOP0043, AOP0442, AOP0071, AOP0417, AOP1058, AOP0401, AOP0424, AOP0441, AOP1001**) koji su neophodni za Altman Z skor formulu tako što je umesto tri vrednosti koje su bile za jedan bilans tokom 3 godine, izračunata srednja vrednost za ove tri godine i dodeljena u jedinstven atribut za sve ove bilanse tokom sve tri godine kako bi se olakšalo računanje Altman Z skor formule korišćenjem jedne umesto tri vrednosti za bilanse koji su potrebni za formulu.
- Nakon što je izračunata vrednost Altman Z skora, ta vrednost se kao ulazni parametar prosledila računanju verovatnoće bankrotstva kompanije, koja se sudeći po istraživanju pokazala kao verodostojnija i robustnija mera od Altman Z skor formule. Formula za verovatnoću bankrota preduzeća glasi: $P(z) = 1 - \frac{e^z}{1+e^z}$
- Nakon što su od ove formule formirana 4 atributa za kreditne iznose, oni su diskretizovani u 4 kategorije rizika u zavisnosti kom intervalu pripadaju. Intervali su sledeći: *Nema rizika*: P(z) < 0.03, *Mali rizik*: 0.03 <= P(z) < 0.6**,** *Srednji rizik*: 0.6 <= P(z) <= 1.324 i na kraju *Visok rizik*: P(z) > 1.324.
- Na kraju su svi ostali karakterni kategorički atributi kodovani u numeričke kategoričke vrednosti kako bi se mogle kao ulazni atributi uneti u obučavajući model Gradijentnog Podsticanja.

Ulazni skup numeričkih kontinualnih podataka *X* ima sledeće vrednosti za statističke parametre.

	Trenutni broj dana u blokadi	Trenutni iznos blokade	Broj dana u blokadi u proteklih 24 meseca	BrojObjekata	PlaceniIznos	IznossaPDV	AOP0043	AOP0442	AOP0071	AOP0417	AOP1058	AOP0401	AOP0424	AOP0441	AOP1001
count	2299.000000	2.299000e+03	2299.000000	2299.000000	2.299000e+03	2.299000e+03	2.299000e+03	2.299000e+03	2.299000e+03	2.299000e+03	2299.000000	2.299000e+03	2.299000e+03	2299.000000	2.299000e+03
mean	0.797738	8.959459e+04	2.155285	2.829056	1.781911e+07	1.833752e+07	5.508423e+04	4.971801e+04	1.059240e+05	3.280116e+04	7206.956068	4.739710e+04	8.751060e+03	212.988256	2.680739e+05
std	11.959624	2.629144e+06	16.903885	17.766984	1.011178e+08	1.047247e+08	1.592631e+05	1.647361e+05	3.003377e+05	8.484826e+04	21288.871913	1.391586e+05	3.204927e+04	2143.007883	7.193875e+05
min	0.000000	0.000000e+00	0.000000	1.000000	7.844260e+03	7.844260e+03	2.230000e+02	5.500000e+01	2.230000e+02	0.000000e+00	0.000000	0.000000e+00	0.000000e+00	0.000000	6.530000e+02
25%	0.000000	0.000000e+00	0.000000	1.000000	2.657190e+06	2.717178e+06	1.231900e+04	8.370000e+03	1.470300e+04	3.552000e+03	680.000000	4.501000e+03	0.000000e+00	0.000000	5.341550e+04
50%	0.000000	0.000000e+00	0.000000	1.000000	5.515341e+06	5.668100e+06	5.508400e+04	4.971800e+04	1.059240e+05	3.280100e+04	7207.000000	4.739700e+04	8.751000e+03	213.000000	2.680740e+05
75%	0.000000	0.000000e+00	0.000000	1.000000	1.147409e+07	1.181152e+07	5.508400e+04	4.971800e+04	1.059240e+05	3.280100e+04	7207.000000	4.739700e+04	8.751000e+03	213.000000	2.680740e+05
max	256.000000	1.095305e+08	447.000000	781.000000	4.159145e+09	4.320462e+09	6.072811e+06	4.540662e+06	7.821253e+06	2.057192e+06	504212.000000	3.270104e+06	1.113892e+06	99956.000000	1.846768e+07

Slika 3.4 – Statistički parametri za numeričke kontinualne atribute

3.3.2. Altman Z skor formula

Formulu Z-skora za predviđanje bankrota objavio je 1968. Edvard I. Altman, koji je u to vreme bio docent za finansije na Univerzitetu u Njujorku. Formula se može koristiti za predviđanje verovatnoće da će firma bankrotirati u roku od dve godine. Z-rezultati se koriste za predviđanje zadatih vrednosti preduzeća i jednostavno za izračunavanje meru kontrole za status finansijske stabilnosti kompanija na akademskim studijama. Z-skor koristi višestruke korporativne prihode i bilansne vrednosti za merenje finansijskog zdravlja preduzeća.
Z-skor je linearna kombinacija četiri ili pet uobičajenih poslovnih varijabli tojest bilansa, ponderisanih koeficijentima. Koeficijenti su procenjeni identifikovanjem niza firmi koje su proglasile bankrot, a zatim je prikupljen odgovarajući uzorak firmi koje su preživele, uz podudaranje prema delatnosti i približnoj veličini (imovini). Altman je primenio statističku metodu diskriminantne analize na skupu podataka javno držanih proizvođača. Procena se prvobitno zasnivala na podacima javno držanih proizvođača, ali je od tada ponovo procenjena na osnovu drugih skupova podataka za privatne proizvodne, neproizvodne i uslužne kompanije. Originalni uzorak podataka sastojao se od 66 firmi, od kojih je polovina podnela stečaj. Tridesetih i dvadesetih godina, Mervin i drugi su prikupljali odgovarajuće uzorke i došli do zaključka da su različiti računovodstveni pokazatelji dragoceni u predviđanju bankrota.
Altmanova Z skor formula koristi sledećih 9 bilansa:

- Obrtna imovina (Kratkoročna sredstva) – atribut **AOP0043**,
- Kratkoročne obaveze – atribut **AOP0442**,
- Ukupna aktiva (Poslovna imovina) – atribut **AOP0071**,
- Neraspoređeni dobitak – atribut **AOP0417**,
- Dobitak pre oporezivanja (EBIT) – atribut **AOP1058**,
- Kapital – atribut **AOP0401**,
- Dugoročna rezervisanja i obaveze – atribut **AOP0424**,
- Odložene poreske obaveze – atribut **AOP0441**,
- Poslovni prihodi – atribut **AOP1001**

Zatim se sa gore navedenim bilansima formira pet ulaznih promenljivih za Altman Z skor formulu:

- T1 = (Obrtna imovina - Kratkoročne obaveze) / Ukupna aktiva,
- T2 = Neraspoređeni dobitak / Ukupna aktiva,
- T3 = Dobitak pre oporezivanja / Ukupna aktiva,
- T4 = Kapital / (Dugoročna rezervisanja i obaveze + Odložene poreske obaveze + Kratkoročne obaveze),
- T5 = Poslovni prihodi / Ukupna aktiva

Nakon definisanih ulaznih promenljivih se formiraju tri verzije formule u zavisnosti od Pravne forme preduzeća. Ispod su navedene tri različite verzije Altman Z skor formule i Pravna forma za koju se ta verzija formule primenjuje:

- $z = 1.2\,T_1 + \ 1.4\,T_2 + \ 3.3\,T_3 + 0.6\,T_4 + 0.999\,T_5$ (za **Akcionarsko društvo**),
- $z = 6.56\,T_1 + \ 3.26\,T_2 + \ 6.72\,T_3 + 1.05\,T_4$ (za **Društvo sa ograničenom odgovornošću**),
- $z = 0.717\,T_1 + \ 0.847\,T_2 + \ 3.107\,T_3 + 0.420\,T_4 + 0.998\,T_5$ (za sva ostala društva u skupu podataka: **Ortačko društvo, Zadruga** i **Preduzetnik**)

Za sve ove verzije formula se nakon izračunavanja istih primenjuje ista formula za verovatnoću bankrota, koja glasi: $P(z) = 1 - \frac{e^z}{1+ e^z}$
Nakon što se izračuna verovatnoća bankrota, 4 kategorije rizika se dele po sledećim intervalima verovatnoće:

- Nema rizika: *P(z)* < 0.03,
- Mali rizik: 0.03 <= *P(z)* < 0.6**,**
- Srednji rizik: 0.6 <= *P(z)* <= 1.324,
- Visok rizik: *P(z)* > 1.324

3.3.3. Kreiranje novih atributa

Posebno bitan deo inžinjeringa fičera je kreiranje novih atributa iz postojećih kako bi se dobili atributi koji su više korelisani, informativniji za ciljni atribut. Kreiranje novih atributa je izvršeno u skoro svakom skupu podataka. Ispod su navedeni svi kreirani aributi u ovom radu:

- Za svaku fakturu je sumiran iznos sa PDV svih poručenih proizvoda i taj novokreirani atribut je dodat u *fakture* skup podataka pod nazivom **IznossaPDV** kako bi se imala evidencija koliki je ukupan iznos svake fakture koju je klijent poručio.
- Takođe je u istom skupu dodat novi atribut pod nazivom **PreostaoIznos**, koji je razlika novog atribtua IznossaPDV i PlaceniIznos.
- U skupu podataka *kupci* su dodata tri nova atributa: **PlaceniIznos**, **IznossaPDV** i **PreostaoIznos,** koji predstavljaju sumu vrednosti za sve fakture po određenom klijentu.
- Iz skupa podataka vezanog za finansijsku i vlasničku strukturu su kreirani novih devet atributa usrednjavanjem za sve tri godine devet ključnih bilansa za računanje Altman Z skor formule, gde se umesto tri atributa dobija jedan koji je aritmetička sredina za 2016., 2017. i 2018. godinu. Novi atributi su definisani pod sličnim nazivom samo bez godine, to su AOP atributi: **AOP0043, AOP0442, AOP0071, AOP0417, AOP1058, AOP0401, AOP0424, AOP0441** i **AOP1001**.
- Za ulazne parametre Altman Z skor formule je bilo potrebno napraviti pet novih atributa koji dobijeni iz gore navedenih kalkulacija bilansa. Ti atributi su definisani kao: **T1, T2, T3, T4** i **T5**.
- Nakon računanja Altman Z skora i verovatnoće bankrota gde se koristi Altmanov Z skor, generisana su četiri nova atributa koja sadrže vrednosti verovatnoća bankrota ako klijent uzme jednu od navedene 4 sume. Atributi su definisani pod nazivima: "**P_z 500k**" (verovatnoća bankrota za kredit u iznosu od 500 hiljada dinara), "**P_z 2M**" (verovatnoća bankrota za kredit u iznosu od 2 miliona dinara), "**P_z 4M**" (verovatnoća bankrota za kredit u iznosu od 4 miliona dinara) i "**P_z 6M**" (verovatnoća bankrota za kredit u iznosu od 6 miliona dinara).
- Radi veće transparentnosti su za svaku od gore navedenih četiri atributa kreirana nova četiri atributa gde su dodeljene kategorije rizika (*Nema rizika, Mali rizik, Srednji rizik i Visok rizik*) u zavisnosti kom intervalu prirpada verovatnoća bankrota. Ti novi atribut su dodati pod nazivima: "**P_z 500k Rizik**" (kategorija rizika za kredit u iznosu od 500 hiljada dinara), "**P_z 2M Rizik**" (kategorija rizika za kredit u iznosu od 2 miliona dinara), "**P_z 4M Rizik**" (kategorija rizika za kredit u iznosu od 4 miliona dinara) i "**P_z 6M Rizik**" (kategorija rizika za kredit u iznosu od 6 miliona dinara). I kreiranjem ovih atributa je izvršen transfer neoznačenih podataka u označene, s obzirom da su baš ova četiri nova atributa zapravo ciljne promenljive.

Ovako označene ciljne promenljive kao karakterno kategoričke će se koristiti kao Y promenljiva koja će simulirati prave izlazne vrednosti. Prikaz ova četiri atributa je dat na slici ispod.

	KupacID	P_z 500k Rizik	P_z 2M Rizik	P_z 4M Rizik	P_z 6M Rizik
0	10001	Nema rizika	Mali rizik	Mali rizik	Mali rizik
1	10002	Nema rizika	Nema rizika	Nema rizika	Nema rizika
2	10003	Nema rizika	Nema rizika	Nema rizika	Nema rizika
3	10004	Nema rizika	Nema rizika	Nema rizika	Nema rizika
4	10005	Nema rizika	Nema rizika	Mali rizik	Mali rizik
...	...	...	...	...	...
2294	12295	Mali rizik	Srednji rizik	Srednji rizik	Visok rizik
2295	12296	Mali rizik	Srednji rizik	Srednji rizik	Visok rizik
2296	12297	Visok rizik	Visok rizik	Visok rizik	Visok rizik
2297	12298	Visok rizik	Visok rizik	Visok rizik	Visok rizik
2298	12299	Mali rizik	Srednji rizik	Srednji rizik	Visok rizik

Slika 3.5 – Kreirana 4 ciljna atributa

3.4. Treniranje modela klasifikacije

Nakon što su podaci istraženi, očišćeni i kreirani novi atributi, može se pribeći organizaciji podataka za obučavanje modela Ekstremnog Gradijentnog podsticanja XGBoost. Kreiranje navedenog obučavajućeg modela prati linearni tok programa. Celokupan proces se sasotji od 6 koraka:

1. Korak predstavlja pravljenje „recepta“ odnosno modifikacije, uklanjanja ili dodavanja atributa za ulazni skup atributa u obučavajući model kako bi sam model postigao što bolje performanse i dobio što manji šum od određenih atributa. Pravljenje „recepta“ za obučavajući model se može podeliti u tri faze:
    - Dodati su atributi za agregirane vrednosti podataka koji bi potencijalno doprineli performansama modela. Ti atributi su: **BrojObjekata**, **PlaceniIznos** i **IznossaPDV**.
    - Nakon što su gore navedeni agregirani atributu dodati, mogu se ukloniti atributi KupacID i KupacNaziv s obzirom za ovi atributi sadrže jedinstvene vrednosti i mogu samo uneti sa sobom šum u obučavajući model, jer algoritam modela neće biti u mogućnosti da uhvati šablon učenja toliko dobro, kao bez ova dva atributa.
    - Kodovanje svih karakterno kategoričkih atributa u numeričke kako bi kao brojčane vrednosti mogli da se proslede modelu XGBoost koji inače radi sa numeričkim vrednostima što se tiče ulaznih podataka.
    - Na kraju su dodati svi najbitniji bilansi koji su bili iskoršćeni za Altman Z skor formulu.

```
X = df_ml.loc[:, "KupacID":"Set bilansa 2018"]
X["KupacID"] = df_ml["KupacID"].astype(int)

# Dodat je broj objekata za svakog klijenta.
X["BrojObjekata"] = X["KupacID"].map(df_kupci.set_index('KupacID1')['BrojObjekata'])
X["PlaceniIznos"] = X["KupacID"].map(df_kupci.set_index('KupacID1')['PlaceniIznos'])
X["IznossaPDV"] = X["KupacID"].map(df_kupci.set_index('KupacID1')['IznossaPDV'])

# Odbacujemo atribute koji prouzrokuju šum
X = X.drop(columns=['KupacID'])
X = X.drop(columns=['KupacNaziv'])

# Enkodovanje karakternih kategoričkih atributa u numeričke.
for column in X:
  if X[column].dtype == "object":
    X[column] = pd.Categorical(X[column])
    X[column] = X[column].cat.codes

X = pd.concat([X, df_AOP_Z_formula_means], axis=1)
```

Slika 3.6 – Pravljenje „recepta“ ulaznih atributa za model

2. Naredni korak je podela skupa podataka na trening i validacioni skup. Treningom će se model naučiti, a na validacionom skupu će se testirati performanse ovog klasifikacionog modela. Podele se vrše za četiri ciljna atributa što znači da će biti četiri obučavajuća i četiri validaciona skupa. Za trening skup je uzeto 80% uzoraka, dok je za validacioni skup uzeto 20% od ukupnog broja uzoraka.

```
# Podeli podatke na trening i validacioni skup.
from sklearn.model_selection import train_test_split

X_train_500k, X_validation_500k, y_train_500k, y_validation_500k = train_test_split(X, y_500k, train_size = 0.81, random_state=234)
X_train_2M, X_validation_2M, y_train_2M, y_validation_2M = train_test_split(X, y_2M, train_size = 0.81, random_state=234)
X_train_4M, X_validation_4M, y_train_4M, y_validation_4M = train_test_split(X, y_4M, train_size = 0.81, random_state=234)
X_train_6M, X_validation_6M, y_train_6M, y_validation_6M = train_test_split(X, y_6M, train_size = 0.81, random_state=234)
```

Slika 3.7 – Podela podataka na trening i validacioni skup

3. Sledeći korak je kreiranje podrazumevanog obučavajućeg modela i optimizacija hiperparametara iscrpnom pretragom po rešetci (engl. *GridSearch*). Pretraga po rešetci je računarski izuzetno zahtevna metoda koja će isprobava sve kombinacije po rešetci hiperparametara, ali će zato naći najoptimalnije vrednosti hiperparametara.

```
# Optimizacija hiperparametara Grid Search metodom
from sklearn.model_selection import GridSearchCV
xgb_model = XGBClassifier()
param_tuning = {
        'learning_rate': [0.02, 0.03, 0.04, 0.05, 0.06, 0.07],
        'max_features': [2, 3, 4],
        'max_depth': [5, 6, 7, 8, 9],
        'min_child_weight': [6, 8, 10, 12, 14],
        'n_estimators' : [150, 200, 250, 300],
        "random_state": [234],
        'objective': ['multi:softprob']
    }
gsearch = GridSearchCV(estimator = xgb_model, param_grid = param_tuning, cv = 5, n_jobs = -1, verbose = 1)
gsearch.fit(X_train_500k, y_train_500k)
gsearch.best_params_
```

Slika 3.8 – Optimizacija hiperparametara GridSearch metodom

4. Kad metoda pretrage po rešetci izbaci najoptimalnije hiperparametre, formira se obučavajući model sa dobijenim optimalnim hiperparametrima. Ovde je naveden samo jedan od modela koji se obučava nad ciljnim atributom koji klasifikuje rizik za zahteve kredita u iznosu od 500 hiljada dinara što predstavlja najmanji od četiri iznosa.

```
xgb = XGBClassifier(n_estimators = 300, learning_rate = 0.03, max_features = 2, max_depth = 8, num_class = 4,
                    min_child_weight = 12, objective = "multi:softprob", random_state = 234)
xgb.fit(X_train_500k, y_train_500k)
```

Slika 3.9 – Obučavanje XGBoost modela sa optimalnim hiperparametrima

5. Nakon obučavanja se model tesitira nad validacionim podacima koji se testiraju takođe za iznos od 500 hiljada dinara kako bi se ocenile određene performanse modela. Ovde se model testira nad do sada neviđenim podacima, prema tome treba istaknuti da su performanse modela na testiranju skoro uvek lošije od onih na treningu, ako je model dobar.

```
predictions = xgb.predict(X_validation_500k)
```

Slika 3.10 – Testiranje XGBoost modela

6. U ovom koraku se izvršava ponavljajuća unakrsna validacija 10. reda što znači da će se skup podeliti na 10 jednakih delova. Odabrana je ponavljajuća unakrsna validacija s obzirom da je ona robusnija od bazične unakrsne validacije K. Reda. Takođe je promenjena stopa učenja sa 0.03 na 0.04 s obzirom da se pokazalo u unakrsnoj validaciji da to daje veću prosečnu tačnost zato što je za razliku od gore primenjenog modela ovde više puta obučen model nad različitom podelom podataka između trening i test skupa podataka.

```
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)
xgb_cv = XGBClassifier(n_estimators = 300, learning_rate = 0.04, max_features = 2, max_depth = 8, num_class = 4,
                    min_child_weight = 12, objective = "multi:softprob", random_state = 234)
scores = cross_val_score(xgb_cv, X, y_500k, scoring='accuracy', cv=cv, n_jobs=-1)
```

Slika 3.11 – Primena ponavljajuće unakrsne validacije 10.reda

3.5. Evaluacija modela klasifikacije

Posle procesa obučavanja modela počinje proces evaluacije samog modela nad neviđenim podacima iz validacionog skupa podataka. S obzirom da se radi o nebalansiranom skupu podataka, pored tačnosti kao važne metrike performansi modela, analizira se i preciznost, odziv i F1 skor.
Tačnost meri koliki je procenat tačno klasifikovanih instanci. Preciznost je odnos pravih pozitivnih sa predviđenim pozitivnim. Odziv meri koliko je pravih pozitivnih predviđeno kao pozitivne vrednosti, takođe je i poznata kao Senzitivnost ili stopa istinitih pozitivnih (engl. True Positive Rate – TPR). F1 skor je sredina između Preciznosti i Odziva i tu se uspostavlja balans između te dve metrike upotrebom Preciznosti i Tačnosti.
Za objektnu funkciju se koristi posebna verzija *Softmaks* funkcije nazvana *Softprob* koja zapravo predstavlja modifikovanu verziju Softmaksa, koja kao izlaz daje vektor N podataka * K klasa, što znači da za svaki podatak će izračunati verovatnoću da pripada određenoj klasi. Softmaks objektna funkcija je jako pogodna za višekalsne probleme u Mašinskom učenju, a Softprob kao posebna verzija Softmaksa je pogodnija za crtanje ROC krive nakon obučavanja modela zbog numeričkih vrednosti kao rezultata. Doduše treba napomenuti da je ova Softprod objektna funkcija računarski zahtevnija od Softmaks s obzirom da mora za svaku klasu da računa procenat verovatnoće da pripada određenoj klasi.
F1 skor je takođe značajan za merenje značaja atributa i njihove informativnosti, kako bi se potencijalno redukovao broj atributa prema nekoj metrici. Ovim skorom se došlo do očekivanog zaključka da su atributi koji imaju najveći uticaj zapravo bilansi stanja i uspeha, a najviše Kratkoročne obaveze uz Poslovne prihodi, za koje se i intuitivno očekuje da od bilansa budu među najbitnijim atributima. Pored bilansa, kreditna ocena se pokazuje kao izuzetno važan atribut, što takođe samim nazivom atributa govori koliko je klijent ocenjen kao pogodan za vraćanje kredita.
Eksperimentisanjem podeljenosti skupa podataka na trening i validacioni se došlo do toga da je najbolje za performanse modela podeliti skup tako da 81% se uzme za trening podatke, dok će se 19% uzeti za validacioni skup podataka.
Kao najbitnija performansa modela tačnost se konstantnim unapređenjem, eksperimentisanjem i modifikacijom modela dovela na odličnih 89.016% što se ispotavilo kao odličan rezultat s obzirom na neuređenost početnog skupa podataka i broja kvalitetnih instanci za obučavajući model.
Ispod je prikaz jednostavnog klasifikacionog izveštaja modela koji obuhvata prikaz preciznosti, odziva, F1 skora po svim klasama modela. Takođe je prikaza tačnost modela na dnu.

Classification Report

	precision	recall	f1-score	support
Mali rizik	0.92	0.96	0.94	334
Nema rizka	0.79	0.81	0.80	75
Srednji rizik	0.00	0.00	0.00	14
Visok rizik	0.73	0.57	0.64	14
accuracy			0.89	437
macro avg	0.61	0.59	0.60	437
weighted avg	0.86	0.89	0.88	437

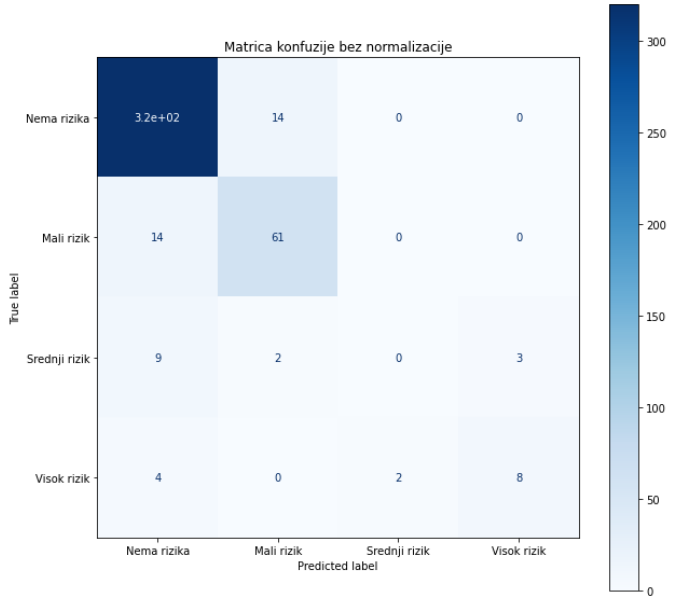
Accuracy: 89.016 %

Slika 3.12 – Klasifikacioni izveštaj

Na slici se može primetiti da su performanse modela odlične za predviđanje klasa: Mali rizik i Nema rizika. Relativno prihvatljive su performanse za klasu Visok rizik i jako loše za predviđanje instanci da je Srednjeg rizika, verovatno iz razloga što je instanci koje se zaista Srednji rizik bilo jako malo i model nije uspeo tačno da ih klasifikuje.

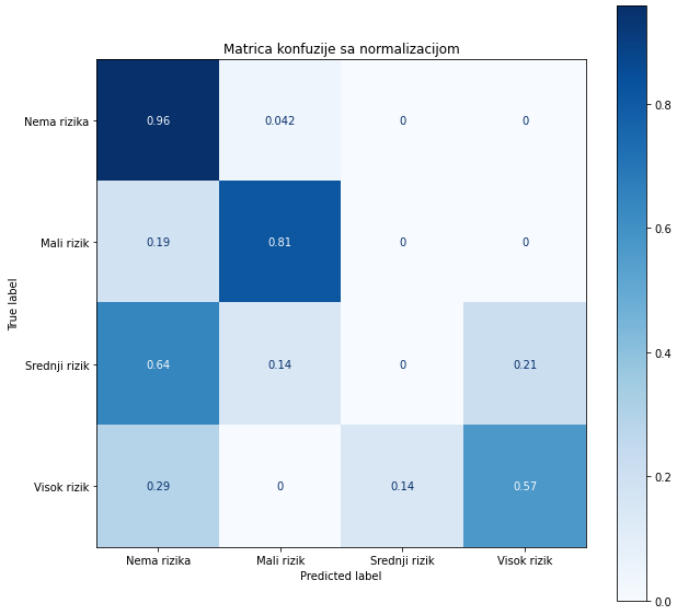
3.5.1. Matrica konfuzije

Jedna od najbitnijih i podrazumevanih metrika koja se koristi za evaluaciju svakog klasifikacionog modela je Matrica konfuzije. Njome se jasno može istaći efektivnost modela, bilo da se radi o binarnom ili višeklasnom problemu. Često se među iskusnim naučnicima kaže da dobro upravljanje i tumačenje rezultatima matrice konfuzije razlikuje dobrog od lošeg naučnika podataka.
Posmatra se koliko je od svake klase tačno klasifikovano instanci, a koliko pogrešno, u odnosu na istinite vrednosti. Ispod je prikazana matrica konfuzije za klasifikaciju rizika ako je iznos kredita 500 hiljada dinara.



Slika 3.13 – Matrica konfuzije bez normalizacije

Radi lakšeg poređenja broja klasifikovanih instanci ubacuje se normalizacija rezultata matrice konfuzije koja je prikzana na slici ispod.

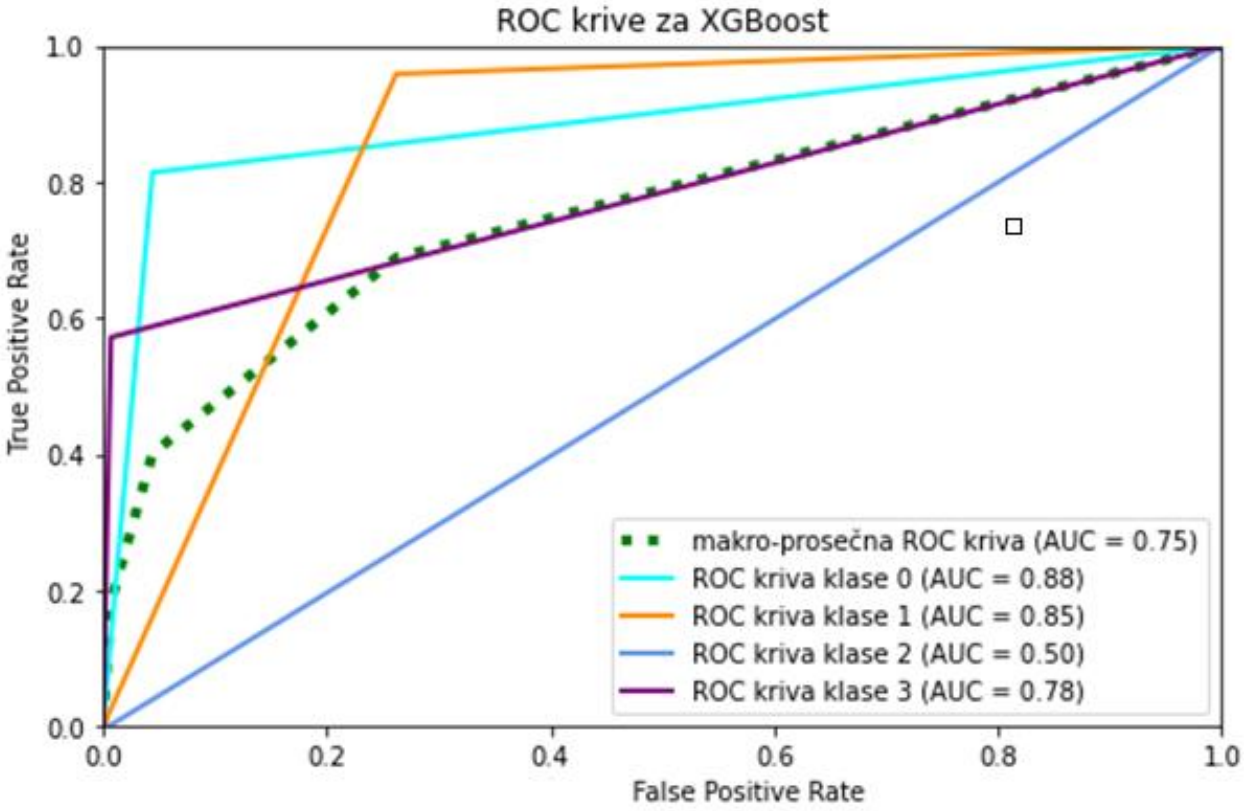


Slika 3.14 – Matrica konfuzije sa normalizacijom

Na slici se sada mogu videti normalizovane vrednosti između 0 i 1 što predstavlja mnogo jasniiji uvid u odnosu na vrednosti koje nisu normalizovane. Prema matrici konfuzije se da zaključiti da je najefikasnija predikcija bila za vrednosti koje su istinski obeležene klasom „Nema rizika“ sa čak 96% tačno predviđenih, dok je za klasu „Mali rizik“ previđeno tačno 81% što je nešto manji procenat, ali i dalje zadovoljavajući. Problem kod ostale dve klase je što je ukupno zastupljen mali broj instanci, pa je i klasifikacija lošija.

3.5.2. ROC kriva

ROC (engl. *Receiver Operating Characteristic*) kriva je još jedna od jako bitnih metrika za evaluaciju modela klasifikacije. Ova metrika evaluira kvalitet izlaznih klasifikacija modela klasifikacije. Grafik ROC krive na Y osi prikazuje odziv tojest stopu istinito pozitivnih (engl. *True Positive Rate*), dok se na X osi prikazuje stopa lažno pozitivnih (engl. *False Positive Rate*) vrednosti.
To znači da je gornji levi ugao grafika „idealna“ tačka - lažno pozitivna stopa je nula i istinski pozitivna stopa je jedan. Ovo nije baš realno da se desi, ali znači da je veća površina ispod krive (AUC) obično bolja. „Strmina“ ROC krivih je takođe važna, jer je idealno maksimizirati istinsku pozitivnu stopu, a lažno pozitivnu stopu minimizirati.
ROC krive se obično koriste u binarnoj klasifikaciji za proučavanje rezultata klasifikatora. Da bi se ROC kriva i ROC područje proširili na klasifikaciju sa više oznaka, potrebno je binarizirati izlaz. Jedna ROC kriva može se nacrtati po oznaci, ali takođe se može nacrtati ROC kriva uzimajući u obzir svaki element matrice indikatora oznake kao binarno predviđanje (mikro-usrednjavanje). Još jedna mera ocenjivanja za klasifikaciju sa više klasa je makro-usrednjavanje, koja daje jednaku težinu klasifikaciji svake klase, inače se za evaluaciju modela u ovom radi koristi makro-usrednjavanje. Tako da će se za svaku izlaznu klasu prikazati po jedna ROC kriva i još jedna kao makro-prosečna za sve ROC krive zajedno.

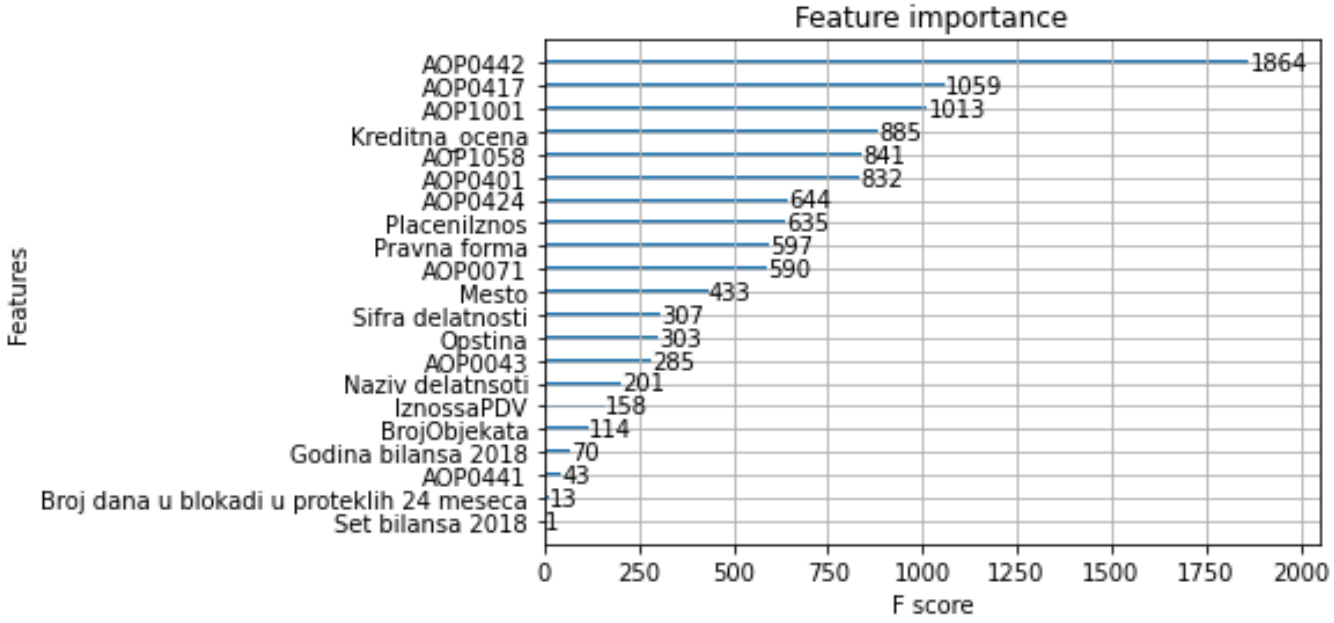


Slika 3.15 – ROC kriva

Na slici iznad se može primetiti da najveću vrednost za AUC (engl. *Area Under Curve*) tojest oblast ispod ROC krive zauzima klasa 0 – odnosno klasa „Nema rizika“ što se intuitivno i moglo pretpostaviti nakon analize matrice konfuzije. Malo ispod je AUC vrednost za klasu 1, odnosno klasu „Mali rizik“, kao što se i može primetiti na grafiku. Nešto manja je površina pokrivena klasom 3, odnosno klasom „Visok rizik“ koje je i istniskim vrednostima manji broj, pa je teže klasifikovati, međutim klasa 2 koja je „Srednji rizik“ ima najgori kvalitet učenja nad istinskim vrednostima od 0.50 što praktično znači da je jednaka nasumičnom pogađanju, koje bi čovek pogađao nasumično. Za ostale tri klase koje imaju vrednosti između 0.5 i 1.0 se mogu interpretirati tako da postoji velika šansa da će klasifikator moći da razlikuje pozitivne vrednosti klase od negativnih vrednosti klase. To je zato što je onda klasifikator u stanju da otkrije veći broj istinitih pozitivnih i istinitih negativnih nego lažnih pozitivnih i lažnih negativnih vrednosti. Kod ROC krive, veća vrednost na X osi ukazuje na veći broj lažno pozitivnih u odnosu na istinito negativne vrednosti.

3.5.3. Izbor atributa

Kada postoji veliki broj atributa polaznog skupa podataka, jako je korisno i zahvalno za model izvršiti dobru selekciju atributa koji će se koristiti u klasifikacionom modelu. Postoje razni načini za odabir i redukciju dimenzionalnosti skupa atributa, od manje kompleksnih ka više kompleksnim. U ovom radu se za potrebe analize atributa i njihovog izbora koristi značaj atributa na osnovu F skora (engl. *Feature Score*). Ispod sledi prikaz grafika sa atributa i njihovim značajem nakon obučavanja modela, kako bi se potencijalno mogao smanjiti broj atributa u ponovnom optimalnijem obučavanju. Rezultati se mogu razlikovati s obzirom na stohastičku prirodu algoritma ili postupka procene ili razlike u numeričkoj preciznosti.

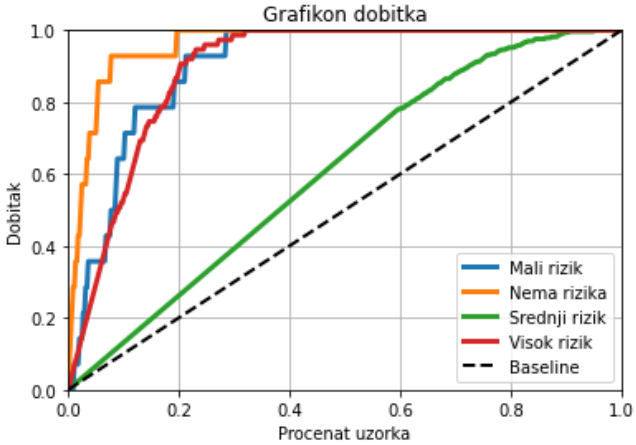


Slika 3.16 – Grafik značaja atributa

Kao što se može primetiti na grafiku iznad najznačajniji atribut je **AOP0442** koji predstavlja bilans kraktoročnih obaveza. Ovaj bilans sudeći po grafiku ubedljivo najviše utiče na klasifikaciju instanci. Intuitivno se to može zaključiti na osnovu činjenice da se ovaj bilans pojavljuje eksplicitno na dva jako važna dela u formuli za računanje Altman Z skora. Takođe se i indirektno pojavljuje u ostalim bilansima s obzirom da je već navedeno da se neki bilansi dobijaju računanjem drugih.
Odmah iza sledi atribut **AOP0417** koji predstavlja bilans neraspoređenog dobitka. Taj atribut se pojavljuje samo na jednom mestu u Altman Z skor formuli, ali bez obzira na to ima značajan uticaj na donošenje odluka u klasifikaciji klijenata, mada i dalje znatno manje od bilansa kratkoročnih obaveza koji ima F skor vrednost 1864, bilans neraspoređenog dobitka ima vrednost F skora 1059.
Neznatno manju F skor vrednost ima atribut **AOP1001** koji predstavlja bilans poslovnih prihoda. Poslovni prihodi se intuitivno mogu bez gafika predpostaviti da imaju veliki značaj u donošenju odluka klasifikacije, s obzirom da što su veći prihodi, rizik će biti sve manji, što se slično može zaključiti i za bilans neraspoređenog dobitka.
Pored svih bilansa koji se koriste u Altman Z skor formuli i koji su među najvažnijim atributima, na četvrtom mestu je atribut **Kreditna ocena**. Ovaj atribut je takođe intuitivno već lako zaključiti koliko je bitan s obzirom da bi i analitičar ili menadžer rizika koji donosi odluke o riziku takođe izuzetno obratio pažnju na kreditnoj oceni klijenta. Kreditna ocena je ocena kreditne sposobnosti klijenta, odnosno u ovom slučaju preduzeća. Ona govori o tome koliko je klijent ocenjen kao sposoban da vrati kredit na osnovu već psotojećih kredita koje taj klijent otplaćuje, koristi dozvoljeni minus, na osnovu visine mesečnih obaveza (već pomenutih bilansa koji se koriste u formuli kod prve i četvrte promenljive (da bi se odredilo da li može još da se zaduži) i da li i koliko ispunjava te mesečne obaveze.

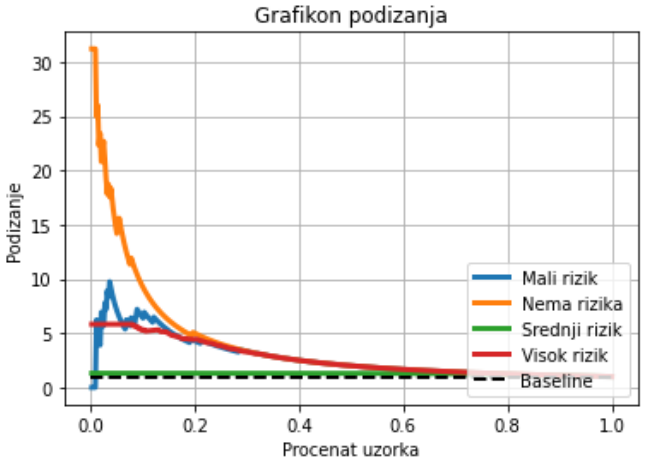
3.5.4. Grafikoni dobitka i podizanja

Poslednje 2 metrike za evaluaciju performansi klasifikacionog modela, ali ne i manje značajna su grafikon dobitka i grafikon podizanja.
Grafikon kumulativnih dobitaka je sjajna univerzalna metrika u nauci o podacima. Prenosi zaista korisne informacije sa stvarnim poslovnim uvidima na osnovu kojih se mogu donositi informisane odluke o smanjenju troškova, a istovremeno maksimizirati korisnost prediktivnog modela. Grafikon kumulativnih dobitaka prikazuje procenat ukupnog broja slučajeva u datoj kategoriji „stečenih“ ciljanjem procenta od ukupnog broja slučajeva. Pogledati kako se to odnosi na izabrani skup podataka u ovom radu na slici ispod.



Slika 3.17 – Grafikon dobitka

Može se primetiti da najveći dobitak ima klasa „Nema rizika“ koja je imala i najveću AUC površinu takođe. Klase „Mali rizik“ i „Visok rizik“ su jako slične po rastu senzitivnosti tojest istinite pozitivne stope, dok je rast krive za klasu „Srednji rizik“ jako usporen. To znači da je već oko 35% uzorka dovoljno da se dostignu dobre performanse za prve tri navedene klase osim za klasu „Srednji rizik“ gde bi sa 35% uzoraka bilo svega malo preko 40% predviđanja za ovu klasu.
Grafikon podizanja koristi slične parametre kao i grafikon dobitka. Kriva podizanja je poslednji grafikon koji dovršava ovu analizu performansi višeklasnog modela klasifikacije. Nije toliko važan niti pronicljiv kao ostali, ali ga ipak vredi napomenuti.



Slika 3.18 – Grafikon podizanja

Na Grafikonu podizanja se može primetiti da je prilično izveden iz grafikona dobitka. Na Y osi je Podizanje koje predstavlja odnos dobitka modela i slučajnog modela, što pokazuje koliko bolje obučavani klasifikacioni model predviđa bolje od nasumičnog modela. Slučajnom „Baseline“ modelu su skoro identična predviđanja vezana za klasu „Srednji rizik“ što je posledica jako malog broja instanci ove klase. Najbolji odnos i dalje ima klasa „Nema rizika“, dok kod klase „Mali rizik“ kriva u početku menja monotonost što govori o nestabilnosti podataka za ovu klasu, dok klasa „Visok rizik“ ima blag nagib nadole.

3.6. Primena modela

Na kraju nakon što se klasifikacioni model obuči, evaluira i eventulano dodatno modifikuje, na redu dolazi primena modela u produkcionom okruženju. Kao najznačajnija metrika za performanse se koristi tačnost. Obučavaju se sva četiri modela u zavisonsti od kreditnog iznosa i na kraju se postojeći klijent može pretražiti na osnovu imena preduzeća ili ID i dobiće se prikaz njegovih podataka X i svih četiri predviđenih vrednosti y'. Kao ulaz u funkciju za pretragu klijenta se dovode svih četiri predviđenih vrednosti i X ulazni skup podataka sa dodatnim kolanama **KupacID** i **KupacNaziv**. Najpre se se prikazuje funkcija za objedinnjavanje svih kolona u jedan okvir podataka, ta funkcija je prikazana na slici ispod.

```
def merge_into_dataframe(kupacID, kupacNaziv, X, predictions1, predictions2, predictions3, predictions4):
  result_df = pd.DataFrame()
  result_df["KupacID"] = kupacID
  result_df["KupacNaziv"] = kupacNaziv
  result_df = pd.concat([result_df, X],  axis = 1)
  result_df["y_500k"] = pd.Series(predictions1)
  result_df["y_2M"] = pd.Series(predictions2)
  result_df["y_4M"] = pd.Series(predictions3)
  result_df["y_6M"] = pd.Series(predictions4)
  return result_df

dataframe = merge_into_dataframe(df_ml["KupacID"], df_ml["KupacNaziv"], X, predictions1, predictions2, predictions3, predictions4)
```

Slika 3.19 – Objedinjavanje kolona za primenu modela

Na slici iznad je prikazana funkcija koja prikuplja sve kolone iz X skupa, kolone koje čine jedinstvene identifikatore za kupca i sve četiri predikcije za kreditni rizik klijenta u zavisnosti od X atributa. Najpre se kreirao prazan okvir podataka (engl. *Dataframe*) i onda su dodate kolone sa jedinstvenim vrednostima (*KupacID* i *KupacNaziv*), zatim su metodom *pd.concat()* spojeni dosadašnji okvir podataka i X ulazni skup podataka po y osi. Na kraju su ostale četiri izlazne kolone dodate u okvir podataka nakon konvertovanja *numpy* nizova u vremenske serije i na kraju je samo vraćen celokupan okvir podataka sa svim objedinjenim atributima. Ovim će se pojednostaviti metoda pretraživanja određenog klijenta u skupu podataka i pored predviđenih izlaznih atributa imao uvid i u ostale atribute. Funkcija za pronalaženje određenog klijenta će jednostavnim načinom pronaći traženog klijenta. Ova funkcija je prikazana na slici ispod.
Navedena funkcija prihvata kao atribute okvir podataka svih klijenata koji je formiram prethodnom funkcijom i identifikatora koji predstavlja int vresdnost KupacID atributa ili string vrednost KupacNaziv

```
def find_client(dataframe, identifier):
  client = ''
  if type(identifier) == int:
    client = dataframe.loc[dataframe['KupacID'] == identifier]
  else:
    client = dataframe.loc[dataframe['KupacNaziv'] == identifier]
  if client.empty:
    client = 'Klijent sa takvinm identifikatorom ne postoji u skupu podataka.'
  return client
```

Slika 3.20 – Pretraga klijenta na osnovu indetifikatora

atributa. Vrši se pretraga klijenta u okviru podataka, na osnovu atributa određenog tipa i na kraju se vraća red kao rezultat instance tog klijenta ili poruka da takav klijent ne postoji u skupu podataka ako nije pronađen na osnovu ulaznog identifikatora. Na kraju se unosi atribut i indetifikator na osnovu koijh će se pronaći određeni klijent kao što je prikazano na slici ispod.

```
atribut = input('Da li zeltie da nadjete kliijenta na osnovu KupacID ili KupacNaziv atributa? ')
if atribut == "KupacID":
  id = int(input("Unesite id klijenta: "))
  client = find_client(dataframe, id)
elif atribut == "KupacNaziv":
  naziv = input("Unesite naziv klijenta: ")
  client = find_client(dataframe, naziv)
else:
  print("Klijent se moze traziti samo na osnovu KupacID i KupacNaziv atributa")
client
```

Slika 3.21 – Unošenje atributa i indetifikatora za pronalaženje klijenta

4. Zaključak

Ovaj rad opisuje jedan od danas najpopularnijih metoda za predikciju kreditnog rizika kako za preduzeća, tako i za sama fizička lica kao samostalne klijente. Bitno je naglasiti da se izvršeno obeležavanje podataka izlaznim atributima, s obzirom da prvobitni podaci nisu bili obeleženi. Pored toga što podaci nisu bili obeleženi, bilo je dosta nedostajućih vrednosti, u velikom broju slučajeva skoro čitavi redovi podataka nemaju uopšte vrednosti, a određen broj atributa u nekim redovima ima nedefinisane vrednosti poput "-" koje treba zameniti vrednostima koje će doprineti što boljim performansama modela.
Transfer iz neoznačenih u označene podatke je realizovan simuliranim ishodima uz pomoć Altman Z skor formule, koja predviđa kreditni rizik na osnovu parametara različitih bilansa stanja i uspeha, kao i njihovih koeficijenata. Nakon izračunavanja ove formule je potrebno kao ulaznu vrednost Z skor proslediti izračunavanju verovatnoće bankrotstva preduzeća koja je još preciznija i robustnija vrednost od same Altman Z skor vrednosti.
Ove vrednosti se moraju razvrstati u četiri klase tako što će se podeliti u četiri intervala u koje će se dobijene vrenosti iz formule za verovatnoću bankrota razvrstati. Zatim se ove čeitir izlazne promenljive koriste za obučavanje modela nakon što se podaci dodatno transformišu i podele na trening i test skupove.
Mogu se koristiti i ostale metode ansambla, konkretno Podsticanja, ali se pokazalo u većini slučajeva da Gradijentno Podsticanje daje rezultate, odnosno performanse modela koji spadaju među najbolje, a u nekim slučajevima spadaju i u najbolje. Takođe jako velika prednost je što modeli Gradijentnog Podsticanja redukuju i varijansu i pomeraj. Varijansu redukuju zato što se koristi više naslaganih modela, dok se pomeraj redukuje u procesu treninga tako što se narednim modelima ukaže na greške koje su napravili prethodni modeli.
Suština samih model ai obučavanja Gradijetnog Podsticanja je učenje tako što se koriste razlike između predviđanih i stvarnih vrednosti iz prethodnih modela, pa se tako i nova izlazna razlika kao ulaz prosleđuje narednom modelu.
Ovi modeli su posebno robustni kada su u pitanju podaci lošijeg kvaliteta, sa nebalansiranim skupom podataka. Među modelima Gradijetnog Podsticanja, posebno se u ovom radu izdvaja i koristi model Ekstremnog Gradijentnog Podsticanja. Najpre treba napomenuti da ovaj model ima mnogo bolje vreme izvršavanja s obzirom da se ansambl stabala može distribuirano obučavati i time uštedeti vremenske resurse, što je jako značajno kada se radi sa jako velikim skupovima podataka, gde je potrebno samim tim i dosta vremena za obučavanje. Ekstremno Gradijentno Podsticanje ili takozvani **XGBoost** podržava uklapanje različitih vrsta objektnih funkcija, uključujući regresiju, klasifikaciju i rangiranje. XGBoost nudi povećanu fleksibilnost, s obzirom da se optimizacija izvršavana nad proširenim skupom hiperparametara, dok u potpunosti podržava *online* metod treniranja.
S obzirom da je potrebno izvršiti predikciju za četiri kategorije rizika: Nema rizika, Mali rizik, Srednji rizik i Visok rizik, stoga se obučavaju i evaluiraju zasebno četiri modela od kojih svaki ima malo drugačije performanse, s obzirom da su i podaci koji se odnose na date kategorije različito raspoređeni, izbalansirani i različite količine.
Bitno je napomenuti koliko je značajna optimizacija hiperparametara pre početka obučavanja modela, pogotovo kod složenijih modela kod kojih je veliki broj hiperparametara i kod kojih se ne može lako prema intuiciji i malom broju eksperimanata utvrditi koje su optimalne vrednosti hiperparametara. Postoje različite metode optimizacije hiperparametara. U ovom radu se koristi metoda intezivne pretrage po rešetki. Gde su u zamišljenoj rešetki isprobavaju sve kombinacije svih unetih vrednosti hiperparametara. Ova pretraga će posle nekog vremena najverovatnije naći najoptimalnije vrednosti hiperparametara, ali je problem što sa velikim brojem hiperparametara i njihovih vrednosti može doći do eksponencijalne vremenske složenosti s obzirom da se isprobavaju sve moguće kombinacije vrednosti. Neke vremenski optimalnije metode optimizacije hiperparametara su Genetski algoritmi, Bajesovska optimizacija, kao i razni algoritmi inteligencije rojeva gde se koriste određene metaheuristike. Takođe se nakon obučavanja modela izvršava unakrsna validacija 10. reda kako bi se videla prosečna tačnost modela i ostale metrike kada se skup podeli na 10 delova i istrenira se svaki put tako da je drugačiji deo skupa podataka testni skup za trenutni model. Ovde se dobija prosečna tačnost od 89.5%.

Prilozi

Lista slika

Literatura

1. Milosavljević M., *Veštačka Inteligencija*, 2015.

2. Jason Brownlee, *A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning*, 2020. Dostupno na: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/

3. Wikipedia, *Mašinsko učenje,* 2014. Dostupno na: https://sh.wikipedia.org/wiki/Mašinsko_učenje

4. Startit, *Mašinsko učenje je zabavno,* 2016. Dostupno na: https://startit.rs/masinsko-ucenje-je-zabavno/

5. Siddharth Pandey, *An introduction to Machine Learning*, 2020. Dostupno na: https://www.geeksforgeeks.org/introduction-machine-learning/

6. Ayush Pant, *Introduction to Machine Learning for Beginners*, 2019. Dostupno na: https://towardsdatascience.com/introduction-to-machine-learning-for-beginners-eed6024fdb08

7. Lisa Tagliaferri, *An Introduction to Machine Learning*, 2017. Dostupno na: https://www.digitalocean.com/community/tutorials/an-introduction-to-machine-learning#

8. IBM Cloud Education, *Supervised Learning*, 2020. Dostupno na: https://www.ibm.com/cloud/learn/supervised-learning

9. Margaret Rouse, *Supervised Learning*, 2020. Dostupno na: https://searchenterpriseai.techtarget.com/definition/supervised-learning

10. Sidath Asiri, *Machine Learning Classifiers*, 2018. Dostupno na: https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623

11. Jason Brownlee, *4 Types of Classification Tasks in Machine Learning*, 2020. Dostupno na: https://machinelearningmastery.com/types-of-classification-in-machine-learning/

12. Mohammad Waseem, *How To Implement Classification In Machine Learning?*, 2020. Dostupno na: https://www.edureka.co/blog/classification-in-machine-learning/

13. Wikipedia, *Ensemble learning*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Ensemble_learning

14. Vadim Smolyakov, *Ensemble Learning to Improve Machine Learning Results*, 2017. Dostupno na: https://blog.statsbot.co/ensemble-learning-d1dcd548e936

15. Aishwarya Singh, *A Comprehensive Guide to Ensemble Learning*, 2018. Dostupno na: https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/

16. Diogo Menezes Borges, *Ensemble Learning: 5 Main Approaches*, 2019. Dostupno na: https://www.kdnuggets.com/2019/01/ensemble-learning-5-main-approaches.html

17. Lujing Chen, *Basic Ensemble Learning (Random Forest, AdaBoost, Gradient Boosting)- Step by Step Explained*, 2019. Dostupno na: https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

18. Cory Maklin, *Gradient Boosting Decision Tree Algorithm Explained*, 2019. Dostpuno na: https://towardsdatascience.com/machine-learning-part-18-boosting-algorithms-gradient-boosting-in-python-ef5ae6965be4

19. nikki2398, *ML – Gradient Boosting*, 2020. Dostupno na: https://www.geeksforgeeks.org/ml-gradient-boosting/

20. Jason Brownlee, *A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning*, 2016. Dostupno na: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/

21. Ahmad Anis, *Easy Guide To Data Preprocessing In Python*, 2020. Dostupno na: https://www.kdnuggets.com/2020/07/easy-guide-data-preprocessing-python.html

22. Abhishek Sharma, *Data Preprocessing for Machine learning in Python*, 2018. Dostpuno na: https://www.geeksforgeeks.org/data-preprocessing-machine-learning-python/

23. Ezukwoke K.I, *Data Transformation for Machine Learning*, 2020. Dostpuno na: https://www.academia.edu/40436475/Data_Transformation_for_Machine_Learning

24. Damian Chan, *Why You Need Data Transformation in Machine Learning*, 2019. Dostupno na: https://www.datanami.com/2019/11/08/why-you-need-data-transformation-in-machine-learning/

25. Roweida Mohammed, Jumanah Rawashdeh and Malak Abdullah, *Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results*, 2020. Dostupno na: https://www.researchgate.net/publication/340978368_Machine_Learning_with_Oversampling_and_Undersampling_Techniques_Overview_Study_and_Experimental_Results

26. Ronak Gangwal, *A Data Scientist's Guide to 8 Types of Sampling Techniques*, 2019. Dostupno na: https://www.analyticsvidhya.com/blog/2019/09/data-scientists-guide-8-types-of-sampling-techniques/

27. Wikipedia, *Oversampling and undersampling in data analysis*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Oversampling_and_undersampling_in_data_analysis

28. Younes Charfaoui, *Hands-on with Feature Selection Techniques: An Introduction*, 2020. Dostupno na: https://heartbeat.fritz.ai/hands-on-with-feature-selection-techniques-an-introduction-1d8dc6d86c16

29. Wikipedia, *Feature selection*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Feature_selection

30. Saurabh Raj, How to Evaluate the Performance of Your Machine Learning Model, 2020. Dostupno na: https://www.kdnuggets.com/2020/09/performance-machine-learning-model.html

31. Tavish Srivastava, *11 Important Model Evaluation Metrics for Machine Learning Everyone should know*, 2019. Dostupno na: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/

32. Samadrita Ghosh, *The Ultimate Guide to Evaluation and Selection of Models in Machine Learning*, 2020. Dostupno na: https://neptune.ai/blog/the-ultimate-guide-to-evaluation-and-selection-of-models-in-machine-learning

33. Niwratti Kasture, *10 essential ways to evaluate Machine learning model performance*, 2020. Dostupno na: https://medium.com/analytics-vidhya/10-essential-ways-to-evaluate-machine-learning-model-performance-6bf6e11f9502

34. Wikipedia, *Confusion Matrix*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Confusion_matrix

35. Sarang Narkhede, *Understanding Confusion Matrix*, 2018. Dostupno na: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

36. Jason Brownlee, *What is a Confusion Matrix in Machine Learning*, 2020. Dostupno na: https://machinelearningmastery.com/confusion-matrix-machine-learning/

37. Wikipedia, *Python (programming language)*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Python_(programming_language)

38. Wikipedia, *scikit-learn*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Scikit-learn

39. Jason Brownlee, *A Gentle Introduction to Scikit-Learn: A Python Machine Learning Library*, 2020. Dostupno na: https://machinelearningmastery.com/a-gentle-introduction-to-scikit-learn-a-python-machine-learning-library/

40. Wikipedia, *XGBoost*, 2020. Dostupno na: https://en.wikipedia.org/wiki/XGBoost

41. Manish Pathak, *Using XGBoost in Python*, 2019. Dostupno na: https://www.datacamp.com/community/tutorials/xgboost-in-python

42. Wikipedia, *Exploratory data analysis*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Exploratory_data_analysis

43. Wikipedia, *Altman Z-score*, 2020. Dostupno na: https://en.wikipedia.org/wiki/Altman_Z-score

44. Goranka Knežević, Nemanja Stanišić, Vule Mizdravković, *Analiza Finansijskih Izveštaja*, 2017. Dostupno na: https://singipedia.singidunum.ac.rs/preuzmi/42555-analiza-finansijskih-izvestaja-drugo-izmenjeno-i-dopunjeno-izdanje/2826

45. Jason Brownlee, *Repeated k-Fold Cross-Validation for Model Evaluation in Python*, 2020. Dostupno na: https://machinelearningmastery.com/repeated-k-fold-cross-validation-with-python/

46. Georgios Sarantitis, AUC-ROC, *Gains Chart and Lift Curve explained with business implications*, 2020. Dostupno na: https://gsarantitis.wordpress.com/2020/04/29/auc-roc-gains-chart-and-lift-curve-explained-with-business-implications/

47. IBM Knowledge Center, *Cumulative Gains and Lift Charts*, 2020. Dostupno na: https://www.ibm.com/support/knowledgecenter/sr/SSLVMB_24.0.0/spss/tutorials/mlp_bankloan_outputtype_02.html

48. Anastasios Petropoulos, Vasilis Siakoulis, Evaggelos Stavroulakis, Aristotelis Klamargias, Bank of Greece, *A robust machine learning approach for credit risk analysis of large loan level datasets using deep learning and extreme gradient boosting*, 2018.