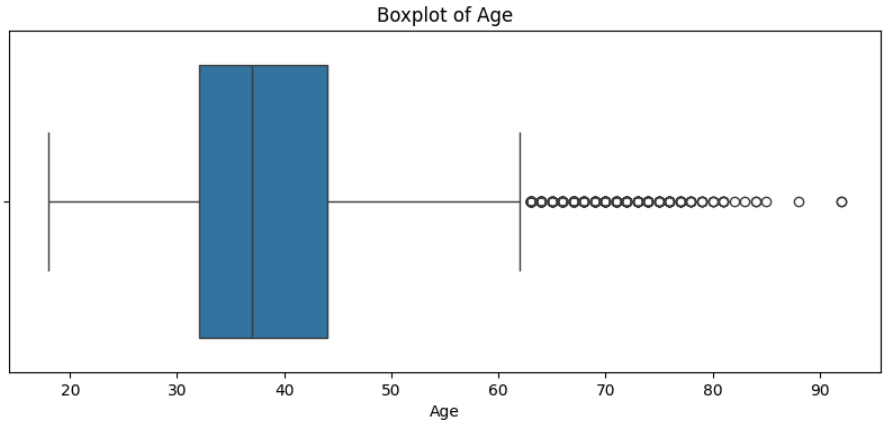
Uvod

Kao zadatak sam dobio da kreiram model za predviđanje odlaska klijenta iz banke na osnovu baze podataka “Churn_Modeling.csv”.

Priprema podataka

Kako su podaci već prikupljeni, neophodno ih je samo pripremiti za dalju obuku potencijalnih modela. Svi prosljeđeni podaci su ispravni - nema nedostataka. Što se tiče anomalija, one postoje u godinama klijenata, kreditnom skoru kao i u broju usluga svakog klijenta. Anomalije su vrijednosti koje odskaču od prosjeka podataka. Njihovo uklanjanje sam realizovao koristeći IQR metodu. Slika ispod ukazuje primjer prikaza anomalija po godinama klijenta.



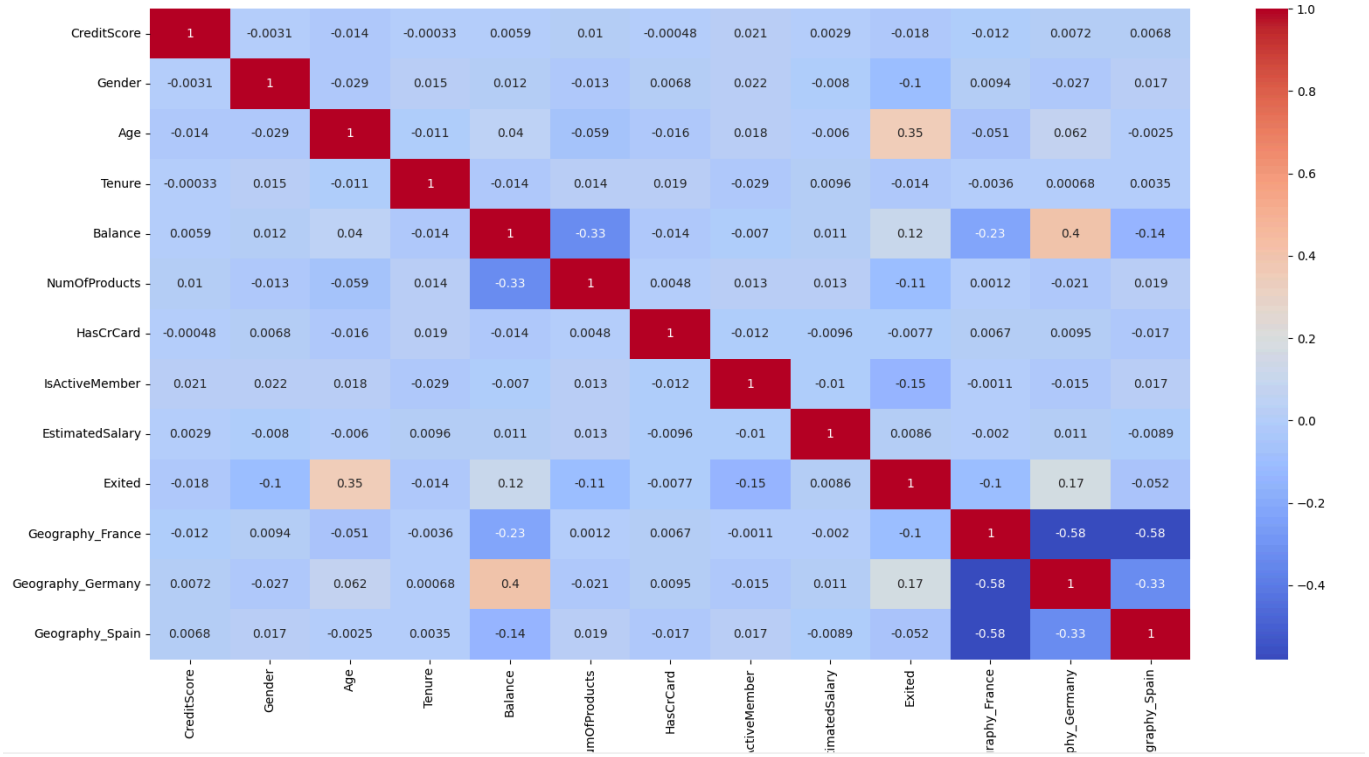
Anomalije po godinama starosti klijenta (anomalije su označene crnim tačkama)

Po kreditnom skoru postoji 15 anomalija, po godinama 359 a po broju proizvoda koje klijent ima u banci, postoji 60 anomalija. Sve anomalije su uklonjene.
Sasvim realna pretpostavka je da kolone *RowNumber, CustomerId* i *Surname* nikako ne utiču na to da li će klijent napustiti banku ili ne. Takođe sumnjam u kolone *Geography* i *Gender* ali njih ne bih uklanjao odmah na početku. U dijelu “Odabir najbitnihih atributa” će biti još riječi o tome koji atributi/kolone nisu potrebne tj. jedva utiču na izlaz modela.
Pošto postoje vrijednosti koje su tipa string, one se moraju pretvoriti u brojnu vrijednost. Istraživanjem sam zaključio da je za enkodovanje podataka najbolje koristiti metodu zvanu *One Hot Encoding* jer ona daje isti značaj svim podacima što nije u slučaju sa Label Encodingom. Ipak, Label Encoding sam koristio za kolonu *Gender* jer ona ima samo dvije vrijednosti te nema smisla za nju koristiti One Hot Encoding jer će se opet na kraju svesti na Label Encoding[1].
Nakon enkodovanja podataka slijedi podjela na X i y skupove kako bi započeo balansiranje i normalizaciju. Balansiranje je neophodno čime sam se uvjerio tako što sam ispisao koliko ima izlaza kada *Exited* ima vrijednost nula odnosno jedan i dobio da je odnos 7677:1891. Za balansiranje nisam koristio *RandomOver(Under)Sampler* jer se pokazao kao loš način balansiranja, već sam koristio metodu SMOTETomek koja kombinuje operacije under sampling i over sampling. Za ovu metodu balansiranja sam se odlučio jer smatram da je bitno da podaci koji su u manjini budu umnoženi u smislu da će biti generisano više sličnih podataka onim koji fale a da onih kojih ima više, odnosno više sličnih ili istih, će under samplingom da se smanje.[2]
Sada je na redu podjela na train i test skupove i normalizacija podataka. Našao sam jako puno metoda koje mogu izvšiti normaizaciju ali sam se ipak odlučio za *MinMaxScaler* jer on skalira na opseg [0,1] a takođe je preporučljiv kada je opseg podataka bitan.[3]

[1] https://www.analyticsvidhya.com/blog/2020/03/one-hot-encoding-vs-label-encoding-using-scikit-learn/
[2] https://machinelearningmastery.com/random-oversampling-and-undersampling-for-imbalanced-classification/
[3] https://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html

Eksplorativna analiza podataka

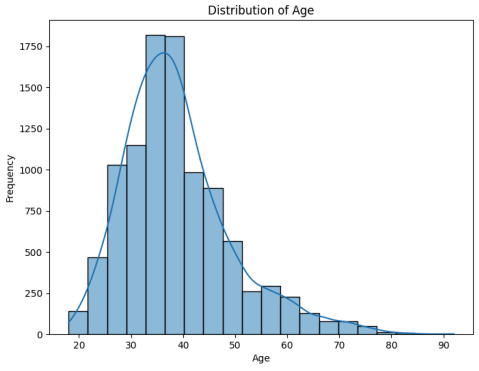
Eksplorativnu analizu započinjem korelacijom. Korelaciju sam još uradio i prije balansiranja ali pošto ona spada u eksplorativnu analizu, pišem je ovde. Korelaciona matrica izgleda ovako:

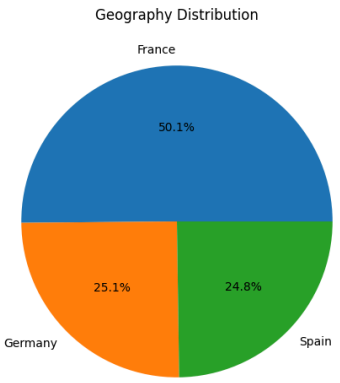


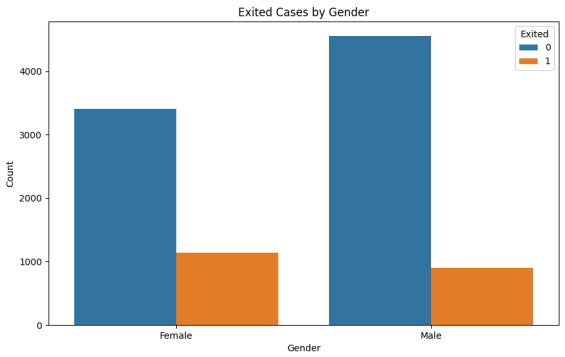
Korelaciona matrica

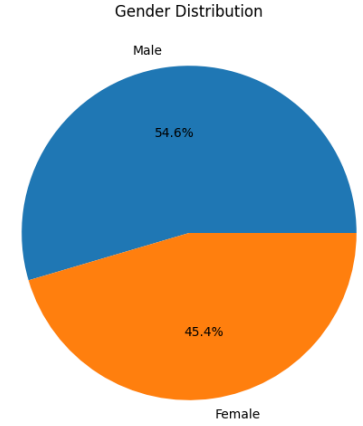
Najveća korelacija (izuzev dijagonale) je -0.58 što se negdje smatra i jakom korelacijom ali kako smo mi definisali da je jaka korelacija od 0.8 pa na dalje, neću ništa brisati.

Daljom analizom podataka sam utvrdio da je prosječna starost oko 39 godina, kreditni skor otprilike 650, prosječan iznos na računu oko 76485 a plata oko 100090… Sljedeći grafici daju više informacija o skupu podataka:









Modeli

Pored modela koje smo koristili na vježbama - *LinearRegression, KNeighborsClassifier* i *DecisionTreeClassifier*, dodao sam još nekoliko modela:

- GaussianNB
- Bagging (Bootstrap Aggregating) - RandomForestClassifier, ExtraTreesClassifier
- Boosting - GradientBoostingClassifier, AdaBoostClassifier
- Stacking - StackingClassifier

Naredna tabela pokazuje neke osnovne karakteristike svake grupe modela (bagging, boosting i stacking):

Osobina	Bagging	Boosting	Stacking
Treniranje modela	Paralelno	Sekvencijalno	Paralelno bazni modeli, zatim meta-model
Fokus	Smanjenje varijanse	Smanjenje pristrasnosti i varijanse	Kombinovanje različitih modela
Uzorkovanje	Sa vraćanjem	Ponderisano, sekvencijalno	Ne koristi uzorkovanje
Kombinacija	Prosečna vrednost ili većinsko glasanje	Ponderisano sabiranje	Meta-model
Primenjeni modeli	Isti model	Slabi učesnici (weak learners)	Različiti modeli

Tabela ukratko prikazuje osnovne karakteristike i međusobne razlike bagging, boosting i stacking modela[4]

Bagging (Bootstrap Aggregating)

Bagging je tehnika koja koristi više instanci istog modela treniranih na različitim podskupovima podataka. Cilj je smanjenje varijanse modela. Karakteristike:

- Uzorkovanje sa vraćanjem - Svaki model je treniran na različitim podskupovima trening podataka generisanim uz uzorkovanje sa vraćanjem (bootstrap samples).
- Paralelno treniranje - Svi modeli se treniraju nezavisno i paralelno.
- Kombinovanje predikcija - Predikcije se kombinuju kroz prosečnu vrednost (za regresiju) ili većinsko glasanje (za klasifikaciju).

Boosting

Boosting je sekvencijalna tehnika koja koristi više slabih modela (weak learners), obično jednostavne modele kao što su decision trees, gde svaki naredni model pokušava da ispravi greške prethodnih modela. Karakteristike:

- Sekvencijalno treniranje - Modeli se treniraju jedan za drugim, svaki pokušava da ispravi greške prethodnih.
- Ponderisanje uzoraka - Uzorci koji su pogrešno klasifikovani od strane prethodnih modela dobijaju veće težine.
- Kombinovanje predikcija - Kombinacija se vrši ponderisanjem predikcija svih modela.

Stacking

Stacking je tehnika koja kombinuje predikcije različitih modela (baznih učesnika) koristeći meta-model (stacker) koji donosi konačnu odluku na osnovu tih predikcija. Karakteristike:

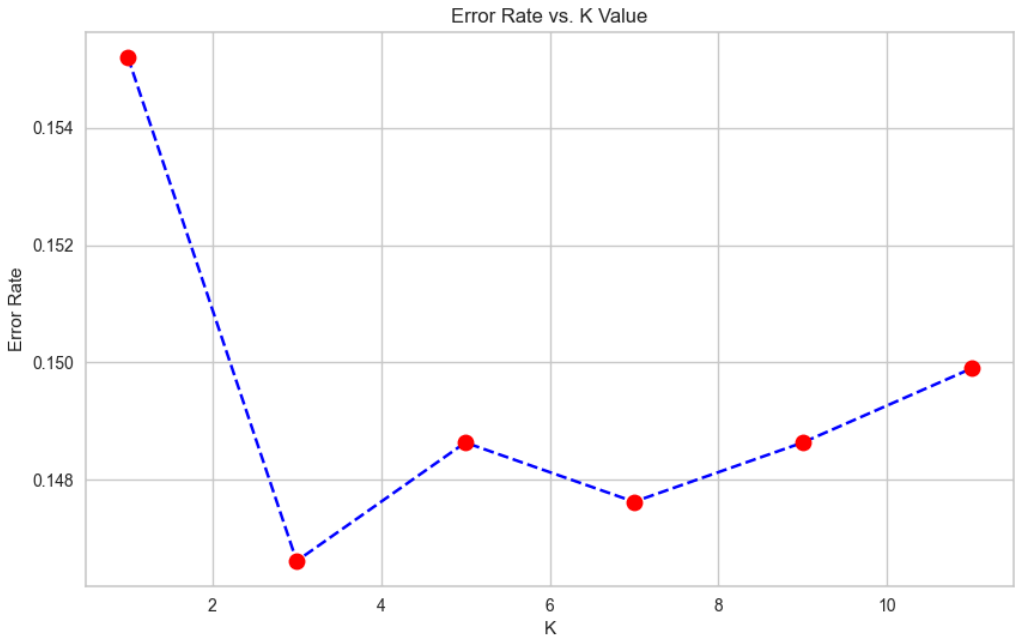
- Kombinacija različitih modela - Koriste se različiti tipovi modela (npr. logistic regression, SVM, decision tree).

[4] Tabela je dobijena sa ChatGPT-a

- Meta-model - Predikcije baznih modela se koriste kao ulazi za meta-model koji uči kako najbolje kombinovati ove predikcije.
- Dve faze treniranja - Prvo se treniraju bazni modeli, zatim se trenira meta-model na njihovim predikcijama.

KNeighborsClassifier i pronalaženje k

Poznato je da kada se govori o KNeigborsClassifier da je najveći problem odrediti k. Koristeći metodu latka (eng. Elbow method) u najvećem broju slučaja k bude tri.



Slika pokazuje vrijednsot greške kroz vrijednosti k - ovde je odabrana vrijednost 3 za k

Podešavanje hiperparametara korišćenih modela

S obzirom da skup nije baš mali a da treba obučiti više modela i da kod treba pokrenuti više puta, i uz sav istraživački i ostali rad, broj hiperparametara koje sam podešavao nije naročito velik jer bi se povećanjem broja parametara povećalo i vrijeme izvršavanja programa.
Hiperparametre sam podešavao koristeći GridSearchCV. Uzeo sam da podešavam sljedeće hiperparametre:

- LogisticRegressionCV - *Cs* i *cv*
- KNeighborsClassifier - *algorithm*
- DecisionTreeClasifier - *criterion*
- RandomForestClassifier - *n_estimators* i *max_depth*
- ExtraTreesClassifier - *n_estimators* i *max_features*
- StackingClassifier - *cv*
- GradientBoostingClassifier - *n_estimators* i max_depth
- AdaBoostClassifier - *n_estimators* i *learning_rate*

U dijelu “Analiza rezultata predikcije modela” sam pisao o rezultatima i nakon podešavanja hiperparametara.

Unakrsna validacija

Unakrsnu validaciju sam uradio za svaki model koristeći funkciju *cross_val_score* iz biblioteke *sklearn*.

Analiza rezultata predikcije modela

Analizu rezultata sam sproveo koristeći sledeće metrike:

- Tačnost (Accuracy score)
- Preciznost (Precision score)
- Odziv (Recall score)
- F1 skor (F1 score)
- Matrica konfuzije (Confussion matrix)
- Jaccard score[5] - je mjera sličnosti između dva skupa. Matematička formula je sljedeća:

  Jaccard score = (broj elemenata koji se nalaze i u jednom i u drugom skupu) / (broj elemata unije oba skupa)

  $$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

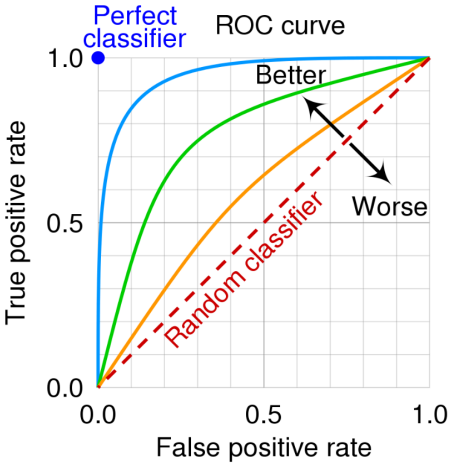
- Log loss score[6] - ukazuje na to koliko je vjerovatnoća predikcije bliska odgovarajućoj stvarnoj/tačnoj vrijednosti (0 ili 1 u slučaju binarne klasifikacije). Što se više predviđena vjerovatnoća razlikuje od stvarne vrijednosti, veća je vrijednost log-loss-a.

Tačnost modela varira od osamdeset do osamdeset i osam posto kao i preciznost, odziv i F1 skor. Iz matrice konfuzije se zaključuje da model otprilike omaši oko 500 uzoraka ukupno. Jaccardov skor se kreće oko 0.7 sto je zadovoljavajuće ali Log loss skor i nije baš u redu. Najbolja vrijednost je oko trojke ali i to nije za pohvalu s obzirom da je najbolje da Log loss skor bude blizu nule.
Od svih modela, kao najlošiji se pokazao GaussianNB i prije i poslije podešavanja hiperparametara a kao najbolji je izašao RandomForestClassifier uzimajući u obzir sve vrijednosti metrika i sa i bez podešavanja hiperparametara.
Podešavanjem hiperparametara se dobiju mali pomaci od oko jedan do dva posto na bolje po svim metrikama.
Rezultati dobijeni obučavanjem različitih modela su zadovoljavajući i model je upotrebljiv s obzirom da ne predviđamo da li neko boluje od neke bolesti ili ne pa je npr. FN u matrici konfuzije i u redu u ovom slučaju što ima vrijednost od oko dvije stotine na desetak hiljada podataka. Takođe, gledajući i ostale metrike, rezultati su u redu osim Log loss scora koji bi mogao biti bolji.

ROC krive

ROC kriva (Receiver Operating Characteristic curve)[7] je grafički prikaz performansi binarnog klasifikacionog modela, koji prikazuje odnos između stope tačnih pozitivnih (True Positive Rate - TPR) i stope lažnih pozitivnih (False Positive Rate - FPR) na različitim pragovima odluke. ROC kriva se koristi za procenu i upoređivanje performansi klasifikacionih modela. TPR i FPR se računaju prema sljedećim izrazima:

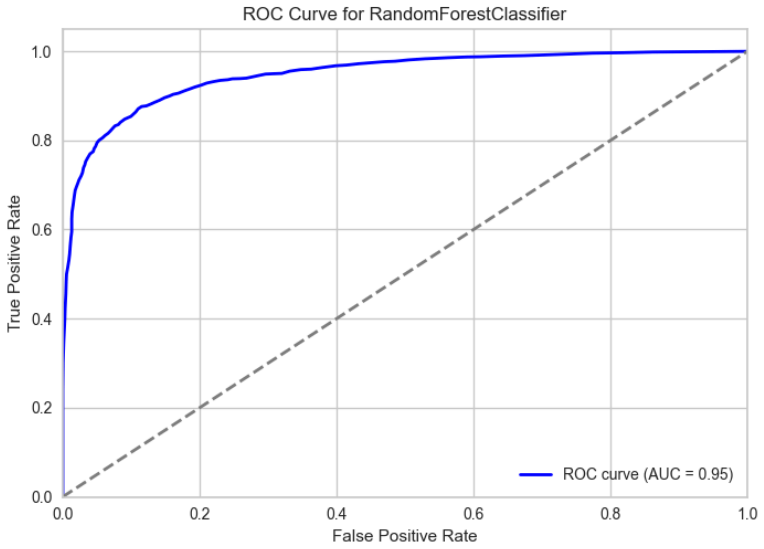
$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$



Slika pokazuje tačnost modela u zavisnosti od ROC krive

[5] https://www.statisticshowto.com/jaccard-index/
[6] https://towardsdatascience.com/intuition-behind-log-loss-score-4e0c9979680a
[7] https://en.wikipedia.org/wiki/Receiver_operating_characteristic

ROC krive, gledajući sve modele su poprilično dobre i veoma su blizu gornoj lijevoj ivici.



Primjer ROC krive za model RandomForestClassifier

Odabir najbitnijih atributa

Za odabir najbitnijih atributa koristio sam dva različita algoritma a to su *SelectKBest* i *RFE*. *SelectKBest* pri svakom pokretanju se odluči za *'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember', 'Geography_France', 'Geography_Germany' i 'Geography_Spain*' dok se *RFE* odluči za '*CreditScore', 'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'IsActiveMember', 'EstimatedSalary', 'Geography_France' i 'Geography_Spain'.*
Ova dva algoritma se razlikuju po svemu dva-tri atributa. Pri prolasku kroz modele, dobije se opet približno isti rezultati kao i kada se atributi ne smanjuju.
Pomenuo sam na početku da sam sumnjao u kolone *Gender* i *Geography* ali sada više ne sumnjam i smatram da su te kolone zapravo jako bitne s obzirom da ih algoritmi za odabir najbitnijih atributa nisu izbacili.
Razmišljao sam o tome da na kraju obučim model nad potpuno svim atributima ali to bi moglo dovesti do problema u obučavanju i lošijih rezultata, jer bi, na primjer, model smatrao da mu je kolona *RowNumber* bitna a sigurno nije jer je vrijednost jedinstvena za svakog čovjeka.

Zaključak

Modeli, s obzirom na namjenu, daju zadovoljavajuće rezultate i smatram ih upotrebljivim, naročito kada se odabere model sa najboljim rezultatima metrika. Atributi koji su uklonjeni na samom početku *RowNumber*, *Surname* i *CustomerId* definitivno trebaju ostati uklonjeni a dalje smanjivanje atributa je nepotrebno jer svakako daju malo lošije rezultate a njihovo prikupljanje svakako smatram jednostavnim jer ako će svi ostali atributi o nekom klijentu biti prikupljeni, biće i ti koji se odbacuju jer su bitne za banku da zna.