Predviđanje visine godišnjih prihoda  
na temelju popisnih podataka

Valerija Iva Banić   
Prirodoslovno-matemaatički fakultet  
Sveučilište u ZagrebuZagreb, Hrvatska  
[bavaler@student.math.hr](mailto:bavaler@student.math.hr) Petar Bratulić  
Prirodoslovno-matemaatički fakultet  
Sveučilište u ZagrebuZagreb, Hrvatska  
[brpetar@studnet.math.hr](mailto:brpetar@studnet.math.hr) Lea Bundalo  
Prirodoslovno-matemaatički fakultet  
Sveučilište u ZagrebuZagreb, Hrvatska  
[bulea@student.math.hr](mailto:bulea@student.math.hr)

**Sažetak: U ovom radu radu koristili smo Adult dataset. Primjenjivali smo logističku regresiju, Random Forest, SVM, kNN, Naive Bayes i neuronske mreže. Koristili smo Label encoding i OneHot encoding te StandardScaler, RobustScaler i MinMaxScaler.**

**Bottom of Form**

# Uvod

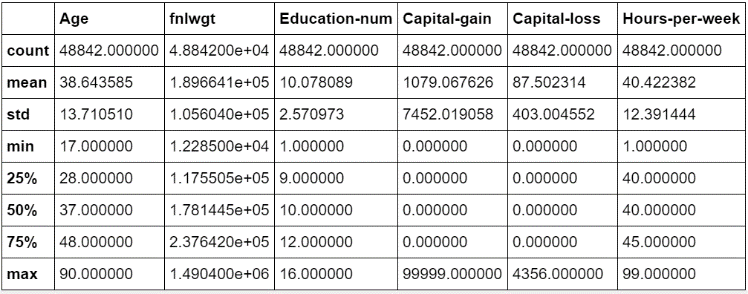
## Opis skupa podataka

Tema našeg projekta je predviđanje visine godišnjih prihoda na temelju popisnih podataka. Podaci koje ćemo koristiti dolaze iz tzv. Adult Dataset-a. Adult Dataset ili Income Census Dataset je skup podataka u kojem se uz demografske i ekonomske karakteristike pojedinca nalazi i podatak o prihodima. Ispitanici su podijeljeni s obzirom na visinu godišnjih prihoda u američkim dolarima. Prva skupina je ona koja ima godišnje prihode manje od 50 000 dolara, dok je druga ona čiji su prihodi veći od tog iznosa.

Podaci su prikupljeni 1994. godine iz popisne baze podataka (Census database) i postoji više istraživanja i radova iz strojnog učenja na tu temu. Naš tim pokušat će sa istim tim podacima koristeći razne algoritme i metode strojnog učenja što točnije predvidjeti koje osobe zarađuju više od 50 000 dolara godišnje, a koje manje od tog iznosa.

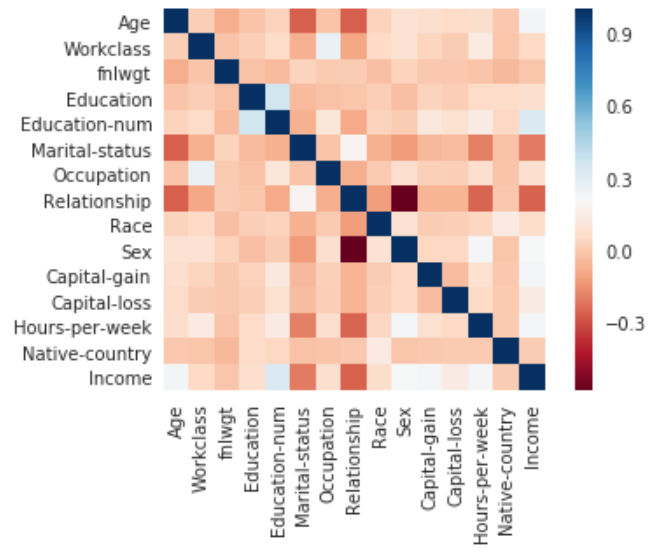
Skup podataka sastoji se od 48 842 instance koje predstavljaju različite ispitane osobe na popisu stanovništva te 14 demografskih i ekonomskih atributa. Atributi po kojima se razlikuju osobe su:

* Age: dob ispitanika, numerička varijabla
* Workclass: radni sektor, kategorijska varijabla (8 kategorija)



* fnlwgt: *final weight*, koristi se u anketama, ispitanici sa sličnim fnlwgt-om imaju slična demografska svojstva, više informacija može se naći na stranici Kaggle-a: <https://www.kaggle.com/uciml/adult-census-income>, numerička varijabla
* Education: najviši postignuti stupanj obrazovanja ispitanika, kategorijska varijabla (16 kategorija)
* Education-num: brojčana oznaka najvišeg postignutog stupnja obrazovanja ispitanika, numerička varijabla
* Marital-status: bračno stanje ispitanika, kategorijska varijabla (7 kategorija)
* Occupation: zanimanje ispitanika, kategorijska varijabla (14 kategorija)
* Relationship: položaj u obitelji ispitanika, kategorijska varijabla (6 kategorija)
* Race: rasa ispitanika, kategorijska varijabla (5 kategorija)
* Sex: spol ispitanika, kategorijska varijabla (2 kategorije)
* Capital-gain: prihod proizašao od investicija, numerička varijabla
* Capital-loss: gubitak priozašao od investicija, numerička varijabla
* Hours-per-week: broj radnih sati tjedno, numerička varijabla
* Native-country: država rođenja, kategorijska varijabla (41 kategorija)
* Income: podatak koji predivđamo (**Target**), zarađuje li ispitanik više ili manje od 50 000 američkih dolara tjedno, kategorijska varijabla (2 kategorije)

Kako bi još malo bolje razumjeli podatke u data setu, prikazujemo korelacijsku matricu svih svojstava:



Iz korelacijske matrice zaključujemo kako je većina svojstava slabo korelirana. Jaku korelaciju pronalazimo između svojstava Education i Education-num. Ta dva svojstva zapravo predstvaljaju istu stvar, Education-num je brojčana reprezentacija svojstva Education.

Također, postoji snažna negativna korelacija između spola i odnosa u obitelji, no to zapravo predstavlja povezanost muškarac-suprug i žena-supruga pa ta dva svojstva ostavljamo kakva jesu. Slično, postoji korelacija između dobi i bračnog stanja te dobi i odnosa u obitelji.

## Cilj istraživanja

Cilj našeg istraživanja bio je odrediti utječu li atributi iz skupa podataka na visinu godišnjih prihoda pojedinca. Obzirom da smo već u eksploratornoj alanizi utvrdili da neki od atributa utječu na tu ciljnu vrijednost, nadalje smo ustanovili koji atributi imaju više, a koji manje utjecaja te kakav im je utjecaj na prihode (koji ih povećavaju, koji smanjuju i što se događa kombinacijom atributa).

Nakon toga, implementirali smo neke od algoritama strojnog učenja kako bi za proizvoljnu osobu za koju znamo vrijednosti atributa mogli procijeniti godišnje prihode. Za kraj, usporedili smo različite metode i algoritme koje smo koristili te pronašli onu koja što bolje predviđa visinu godišnjih prihoda pojedinca. Uz to, usporedili smo naše rezultate s onima pronađenim u literaturi kako bismo vidjeli koliko su naši algoritmi dobri u odnosu na druge.

# Opis rješavanja problema

Problem smo pokušali rješiti sljedećim algoritmima:

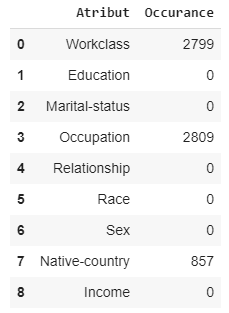
* Logistička regresija
* Random Forest
* SVM
* KNN algoritam
* Naive Bayes
* Neuronske mreže

Na nekoliko različitih načina preuredili smo naše podatke te usporedili uspješnost algoritama s obzirom na različite algoritme i uređivanje podataka. Prilikom razvoja algoritama koristili smo PCA (Analizu glavnih komponenata) kako bi vidjeli utječe li taj postupak na poboljšanje rezultata. Također, pri analizi rezultata promatrali smo razliku uspješnosti algoritama s obzirom na način rješavanja problema nedostajućih vrijednosti.

Mjeru uspješnosti mjerili smo pomoću roc-auc score-a te točnosti (accuracy), iako podaci nisu balansirani (75% instanci ima prihode <=50k dolara, dok ostatak ima >50k dolara) pa smo kao glavnu mjeru evaluacije uzeli roc-auc score, a točnost koristili za usporedbu s rezultatima iz [1] i [4]. Rezultate dobivene korištenjem metrika evaluacije usporedili smo s rezultatima iz literature.

## Label encoding

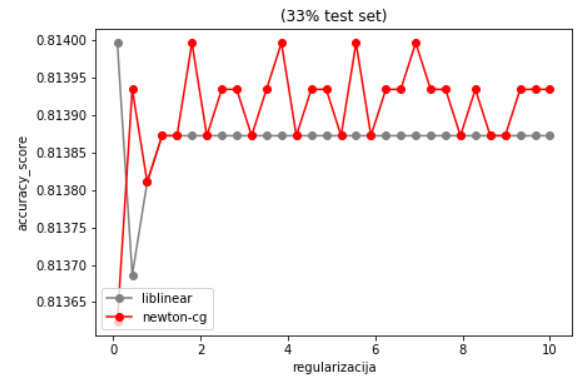
Sljedeća slika prikazuje broj nedostajućih vrijednosti (missing values) označenih s ?, za pojedini atribut u skupu podataka. U početnom razmatranju prvo ćemo nedostajuće vrijednosti smatrati novom kategorijom, a kasnije ćemo pokušati popraviti rezultate izbacivanjem redaka s nedostajućim vrijednostima.

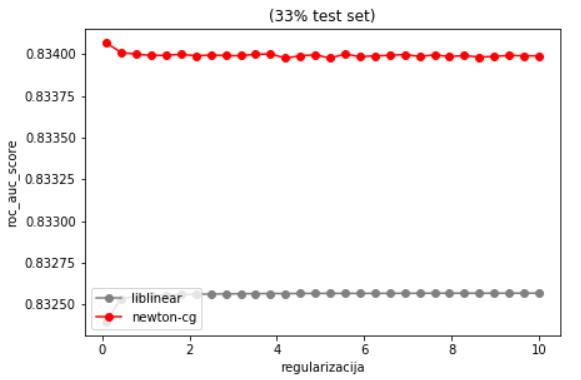


Za početak, zamijenili smo kategoričke vrijednosti podataka koje želimo predvidjeti numeričkima kako bi nam s njima bilo lakše raditi. Isto smo napravili i za ostale atribute koji imaju kategoričke vrijednosti. Novu numeričku vrijednost koja se ne pojavljuje u početnim podacima rezervirali smo za nedostajuće vrijednosti tako da one predstavljaju zasebnu kategoriju. Nakon toga, izbacili smo značajku Education koja nam nije potrebna jer u početnim podacima postoji značajka Education\_num koja predstavlja numeričke vrijednosti istih podataka. Također, svojstva Capital-gain i Capital-loss smo pretvorili u kategorijska s obzirom da su njihove vrijednosti za večinu podataka jednaka nuli.

### Logistička regresija

Proveli smo algoritam logističke regresije za više vrijednosti parametra (faktora regularizacije) c i uspoređujući rezultate zaključili da se točnost i roc-auc-score ne mijenjaju puno za različite vrijednosti parametra c. Također, testirali smo algoritam za različite vrijednosti prametra solver (algoritam optimizacije, liblinear i newton-cg) i ustanovili da su vrijednosti točnosti i roc-auc-score-a više za newton-cg algoritam optimizacije.





Iz konfuzijske matrice vidimo da na točnost i roc-auc-score algoritma najviše utječe velik broj false positive primjera:



### Random forest

Proveli smo random forest algoritam i dobili točnost 0.84 te roc-auc-score 0.886, što je više nego u prethodnom slučaju logističke regresije.

Također, ovdje smo pokušali smanjiti dimenzionalnost podataka. Sljedeći graf pokazuje svojstva poredana po važnosti u određivanju krajnje kategorije:

Nakon toga, proveli smo random forest algoritam sa prvih 9 najbitnijih svojstava i dobili malo bolje rezultate, točnost 0.844 i roc-auc-score 0.892.

### K Nearest Neighbours i Naive Bayes

Proveli smo KNN algoritam s različitim vrijednostima parametara weights i algorithm. Najbolji rezultati koje smo dobili su točnost 0.734 i roc-auc-score 0.635.

Kod Naive Bayes algoritma postigli smo nešto bolje rezultate: točnost 0.789 i roc-auc-score 0.806.

## OneHot encoding

U dosadašnjim algoritmima koristili smo Label encoding, tj. za sva svojstva smo pretpostavili da postoji ordinalnost među podacima, što većinom nije slučaj. Zbog toga nadalje koristimo OneHot encoder.

### Logistička regresija

Nadalje smo u logističkoj regresiji koristili liblinear vrijednost parametra solver jer nam je on davao bolje rezultate. Na ovaj način smo pomoću logističke regresije dobili bolje rezultate nego prije, točnost 0.855 i roc-auc-score 0.907.

Također, promatrali smo važnost značajki te ponovili izvođenje algoritma na podacima smanjene dimenzionalnosti, s brojem svojstava 80, ali tada nismo dobili poboljšanje u točnosti i roc-auc-score-u.

### Random forest

Kod random forest algoritma rezultati se pomoću OneHotEncodinga nisu puno promijenili u odnosu na one bez encodinga.

Također, promatrali smo važnost značajki te ponovili izvođenje algoritma na podacima smanjene dimenzionalnosti, s brojem svojstava 80, ali također nismo dobili poboljšanje u točnosti i roc-auc-score-u.

## Uklanjanje nedostajućih vrijednosti

U dosadašnjim algoritmima promatrali smo nedostajuće vrijednosti kao zasebnu kategoriju, no to nije najbolje rješenje pa smo u sljedećem koraku odlučili pokušati ukloniti sve podatke s nedostajućom vrijednosti nekog svojstva.

Također, bolje rezultate smo dobili uz korištenje OneHot encodera pa ćemo ga i u ovom dijelu nastaviti koristiti.

### Logistička regresija i random forest

Ovdje smo također proveli algoritam logističke regresije i dobili približno iste rezultate kao u prethodnom algoritmu logističke regresije.

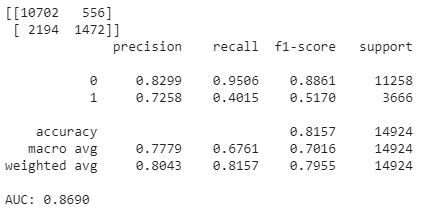
Također, proveli smo random forest algoritam i dobili približne nešto lošije rezultate.

### Neuronske mreže

Provođenjem algoritma neuronskih mreža, pretpostavili smo da će dobiveni rezultati imati lošije vrijednosti metrika zbog neskaliranosti podataka. U mnogo slučajeva, neuronske mreže predividjele su sve instance kao negativne pa smo takve slučajeve izbacili iz analize, tj. u nastavku smo prikazali samo one u kojima su neke instance predviđene kao pozitivne.

Neuronske mreže smo izvršavali za različite vrijednosti parametara hidden\_layer\_sizes (broj i veličina skrivenih slojeva, koristili smo neuronske mreže sa 3 i 4 skrivena sloja od po 13 neurona), activation (aktivacijska funkcija, najbolje rezultate dobili smo kada je postavljena na identity) te alpha i beta\_1. Također, koristili smo razne maksimalne brojeve iteracija algoritma kao način zaustavljanja, a isprobali smo i tol (tolerance for the optimization, algoritam se vrti dok god promijena score-a ne padne ispod postavljene vrijednosti u postavljenom broju koraka). Algoritam koji je dao najbolji roc-auc-score je onaj sa 3 skrivena sloja i 500 iteracija, gdje je točnost 0.728, a roc-auc-score 0.887.

Na slici je prikazan ispis rezultata jednog pokretanja algoritma neuronskih mreža (navedeni najbolji slučaj):

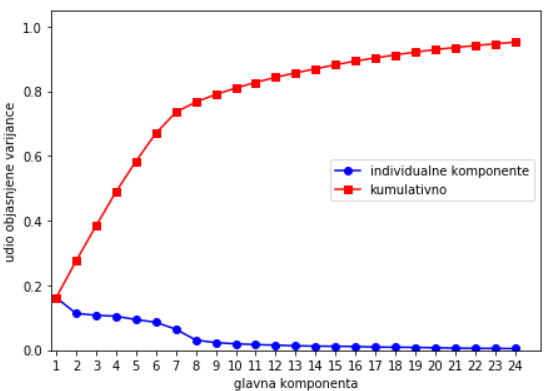


## StandardScaler

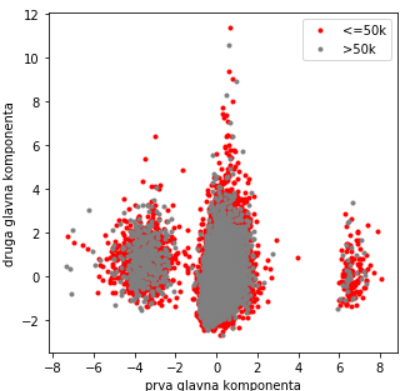
Izbacivanjem podataka u kojima se pojavljuju nedostajuće vrijednosti i smatranjem njih novom kategorijom, rezultati se mijenjaju tek u trećoj decimali pa zaključujemo da promjena nije značajna i nećemo se njome dalje baviti. Koristit ćemo podatke u kojima se ne pojavljuju nedostajuće vrijednosti. Također, i dalje ćemo koristiti OneHot encoder jer su nam se njegovim korištenjem rezultati popravili.

Numerička svojstva normaliziramo pomoću StandardScaler-a ugrađenog u python paketu sklearn, a kategorijske varijable pretvaramo u dummy varijable pomoću OneHot encodinga. Također, pomoću PCA (Principal component analysis) dobivamo da je približno 0.93 objašnjene varijance dobiveno u prvih 20 najbitnijih značajki. U algoritmima smo željeli koristiti broj značajki koji daje bar 0.95 objašnjene varijance i koristeći PCA dobili smo da je taj broj za većinu algoritama jednak 24. Nadalje, radimo s tako izmijenjenim podacima te smo svaki algoritam proveli koristeći PCA i bez korištenja PCA.

Sljedeći graf prikazuje razliku između udjela objašnjene varijance gledajući svaku komponentu (svojstvo) individualno i gledajući kumulativno za prve 24 komponente:



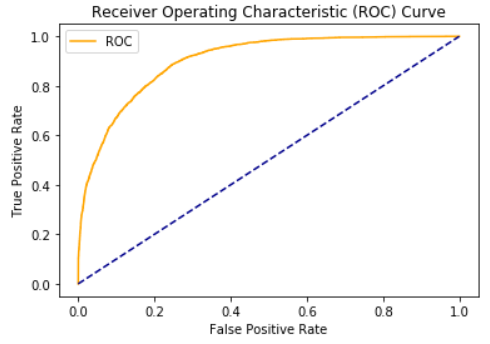
Također, napravili smo scatter-plotove između nekih od glavnih komponenata iz kojih možemo vidjeti kako glavne komponente nisu „lijepo“ podijeljene pa nam to daje objašnjenje zašto ne dobivamo veće vrijednosti točnosti i roc-auc-score-a:



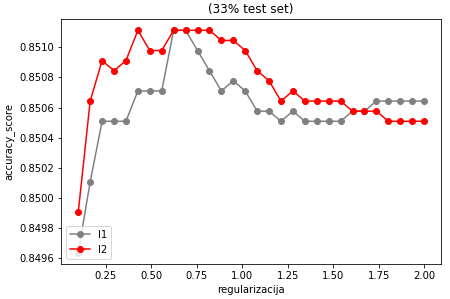
### Logistička regresija

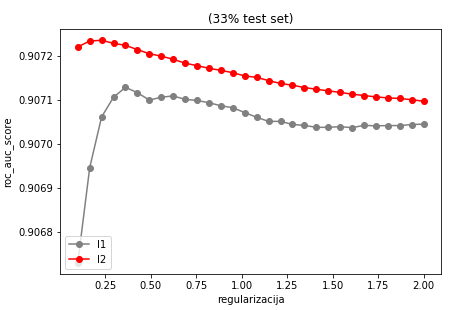
Kod logističke regresije korištenjem PCA dobili smo lošije rezultate točnosti, ali bolji roc-auc score nego kod prethodnih algoritama linearne regresije, kao i kod logističke regresije bez korištenja PCA.

Slika prikazuje ROC-krivulju za algoritam logističke regresije korištenjem PCA:



Ovdje smo također testirali algoritam za različite vrijednosti prametra solver (algoritam optimizacije, liblinear i newton-cg) i ustanovili da su vrijednosti točnosti i roc-auc-score-a više za newton-cg algoritam optimizacije.



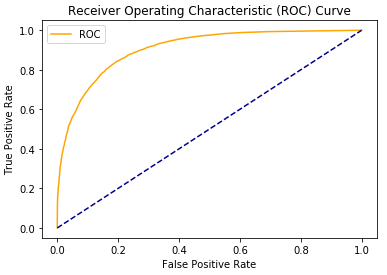


Iz matrice konfuzije zaključujemo da algoritam i dalje lošije klasificira pozitivne primjerke, tj. da je velika vrijednost false positive vrijednosti.

### Random forest

Kod random forest algoritma korištenjem PCA dobiveni su bolji rezultati nego ranije, ali kod istog algoritma bez PCA rezultati su još bolji pa ćemo njih navesti. Točnost najboljih rezultata ovog algoritma je 0.856, a roc-auc-score je 0.908, a to su i općenito najbolji rezultati do sada.

Slika prikazuje ROC-krivulju za random forest algoritam bez korištenja PCA:



### SVM

Proveli smo SVM algoritam bez PCA s parametrom kernel postavljenim na linear za više vrijednosti parametra c te s parametrom kernel postavljenim na rbf za više vrijednosti parametra gamma. Najbolje rezultate dobili smo za linear kernel i vrijednost parametra c=0.3. Tada je točnost jednaka 0.85, a roc-auc-score 0.905. Također, dobre rezultate dobili smo za kernel rbf te vrijednost parametra c=2.0. U tom slučaju točnost je opet 0.85, dok se roc-auc score približava 0.91.

### K Nearest Neighbours

Kod KNN algoritma bez PCA dobili smo bolje rezultate nego ranije s istim algoritmom, točnost je 0.836, a roc-auc-score je 0.867.

KNN algoritmom s korištenjem PCA nismo dobili bolje rezultate od prethodnih.

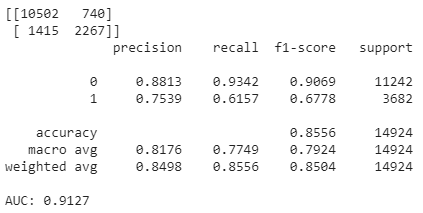
### Naive Bayes

Kod Naive Bayes algoritma također smo dobili bolje rezultate nego ranije, točnost je ovdje 0.792, a roc-auc-score 0.834. Za sada nam Naive Bayes daje lošije rezultate nego ostali algoritmi.

### Neuronske mreže

Za razliku od prethodnog rada s neuronskim mrežama, ovdje očekujemo bolje rezultate jer su podaci skalirani. Zaista, dobili smo bolje rezultate nego prethodno, u najboljem slučaju je točnost bila 0.856, a roc-auc-score 0.913. Ovo nam je do sada ukupno najbolji dobiveni rezultat.

Na slici je prikazan ispis rezultata pokretanja algoritma za ovaj najbolji pronađeni slučaj:



## RobustScaler i MinMaxScaler

Pokušali smo testirati iste algoritme koristeći drugačije metode skaliranja podataka - RobustScaller i MinMaxScaler. Većina rezultata je slična onima ranije dobivenim pa ćemo samo spomenuti one koji odstupaju od toga.

Kod KNN algoritma bez provedbe PCA kao i Naive Bayes-a bez PCA pomoću RobustScaler-a smo primijetili poboljšanje u odnosu na prethodno izvršavanje istog algoritma, ali rezultati nisu bolji od ukupno najboljih pronađenih.

Pomoću MinMaxScaler-a nakon izvođenja svih algoritama nismo dobili bitno poboljšanje niti u jednom algoritmu, stoga nećemo navoditi rezultate.

Nakon prethodnog koraka, pokušali smo još mijenjati dataset i izvoditi algoritme kako bi dobili bolje rezultate. Način kako smo to radili je bilo micanje nekih svojstava i promjena vrsta svojstava (numerička/kategorijska).Neki od tih pokušaja bili su pretvorba svih svojstava u kategorijske, izbacivanje određenih svojstava koji imaju veću korelaciju te micanje samo nekog određenog svojstva. Međutim, niti u jednom slučaju nismo dobili rezultate bolje od već navedenih pa se nećemo ovim dijelom baviti (što smo točno radili možete vidjeti u Pythom bilježnici priloženoj uz ovaj dokument).

# Usporedba rezultata

U uvodu smo spomenuli kako ćemo uspoređivati naše rezultate s rezultatima pronađenima u literaturi. Točnije, uspoređivat ćemo se s rezultatima iz izvora [1], [4] i [7].

Naši najbolji rezultati postignuti su za algoritme neuronske mreže (roc-auc-score 0.9127, točnost 0.8556), Logističku regresiju (roc-auc-score 0.9071, točnost 0.8552), Random Forest (roc-auc-score 0.9079, točnost 0.8571) te SVM (roc-auc-score 0.9047, točnost 0.8583).

U radu [7] najbolji rezultati postignuti su za KNN te NBtree algoritme s pripadnim vrijednostima metrika: KNN (roc-auc-score 0.889, točnost 0.8533) i NBTree (roc-auc-score 0.908, točnost 0.8593). Možemo primjetiti kako su naši rezultati na razini rezultata iz članka te su čak malo bolji kod određenih metoda. Također, naši rezultati KNN bile su u razini onih iz članka dok one Naive Bayesa bili su nešto lošiji od rezultata u članku.

U radu [4] te [1] ( u datoteci adult.names) točnost se u najboljem slučaju kretala oko iznosa od 0.8590 za NBTree algoritam što je slično našim najboljim rezultatima. Ostali algoritmi bili su nešto lošiji, točnost je iznosila oko 0.84 ili 0.83, prema čemu možemo zaključiti da smo dobili čak i nešto bolje rezultate od onih u članku.

# Zaključak

Najbolju točnost, iznosa 0.8583 dobili smo primjenom StandardScaler-a i SVM-a, a za roc-auc score najbolji rezultat, 0.9127, dobili smo primjenom StandardScaler-a i neuronskih mreža.

Najbolje rezultate daju logistička regresija, SVM, Random Forest i neuronske mreže. Primjena PCA u većini slučajeva nije poboljšala rezultate. Korištenjem scaler-a rezultati su se poboljšali, za neke algoritme najbolji rezultati su s MinMaxScalerom, a za neke sa Standard Scalerom.

Tablicu sa točnim vrijednostima za pojedini algoritam i usporedbom vrijednosti možete pronaći u Python bilježnici koja se nalazi uz ovaj dokument.

# Literatura

[1] UCI Machine Learning Repository http://archive.ics.uci.edu/ml/datasets/adult

[2] http://robotics.stanford.edu/~ronnyk/nbtree.pdf

[3] https://storage.googleapis.com/kaggle-forum-message-attachments/160002/5905/Paper%20on%20Machine%20Learning%20for%20Kaggle.pdf

[4] http://robotics.stanford.edu/~ronnyk/nbtree.pdf Ron Kohavi, "Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid", Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 1996

[5] https://towardsdatascience.com/normalization-vs-standardization-quantitative-analysis-a91e8a79cebf

[6] https://scikit-learn.org/stable/index.html

[7] http://www.dataminingmasters.com/uploads/studentProjects/Earning\_potential\_report.pdf

[8] https://github.com/pmf-strojnoucenje/Vjezbe

[9] https://scikit-learn.org/stable/modules/svm.html

[10] https://scikit-learn.org/stable/modules/neighbors.html

[11] https://scikit-learn.org/stable/modules/naive\_bayes.html