

Univerzitet u Kragujevcu
Fakultet inženjerskih nauka



Veštačka inteligencija

Projekat:
Klasifikacija tumora dojke

Student:
Staša Ristić 635/2018

Predmetni nastavnik:
Prof. Vesna Ranković

Kragujevac, 2022.

Contents

Opis zadatog problema.....	1
Istraživačka analiza podataka (Exploratory Data Analysis - EDA).....	2
Priprema baze	3
Analiza veza.....	4
Modeli i analiza performansi	7
Logistička regresija	7
Stablo odlučivanja	8
Slučajna šuma.....	9
Support Vector Machine.....	9
Naivni Bajes	10
ANN.....	12
Zaključak	14
Literatura	15

Opis zadatog problema

Tumor dojke je najčešća maligna bolest žena današnjice i u proseku 1 od 8 žena će u svom životu razviti rak dojke, gde oko dve trećine žena sa rakom dojke ima 55 ili više godina, dok se većina ostalih nalazi u starosnom dobu između 35 i 54 godina. Nakon raka pluća, ovaj oblik raka je drugi vodeći uzrok smrti kod žena i prvi najčešće dijagnostikovani. Bitno je napomenuti, da iako u malom procentu, i muškarci mogu oboleti od ove maligne bolesti (manje od 1% svih slučajeva tumora dojke).

Na sreću, rak dojke je veoma izlečiv ako se otkrije u ranoj fazi dok je još uvek lokalizovan (nije se proširio izvan dojke). Žena sa lokalizovanim rakom dojke ima oko 99% verovatnoću da će živeti najmanje 5 godina nakon dijagnoze. Ukoliko je karcinom počeo širenje (metastazu), proces lečenja biva kompleksniji, a procenat šanse preživljavanja se smanjuje. Sve ovo ukazuje na veliki značaj što ranije i što preciznije dijagnoze.

U ovom zadatku će se vršiti predviđanje dijagnoze tumora dojke. Neće biti razmatran tip tumora dojke kako je za to potrebno znatno više parametara. Cilj je razlučiti, na osnovu odstupanje od normalne formulacije tkiva i ćelija grudne mase, da li je tumor maligni ili benigni. Korišćena baza podataka je "Breast Cancer Wisconsin (Diagnostic) Data Set". Parametri su izračunati na osnovu digitalizovane slike biopsije grudne mase tankom iglom (FNA – fine needle biopsy). Oni opisuju karakteristike ćelijskog jezgra prisutnog na slici.

- Postoje dve atributske informacije:
- ID broj pacijenta
- Dijagnoza (gde M predstavlja Maligni tumor, a B se odnosi na Benigni)
- I deset parametara koji pripadaju sistemu realnih brojeva i izračunati su za svako jezgro ćelije:
- Poluprečnik (prava od centra jezgra do oboda ćelije)
- Tekstura (standardna devijacija vrednosti sive skale)
- Obim
- Površina
- Glatkoća (lokalna varijacija u dužinama radijusa)
- Kompaktnost ($\text{poluprečnik}^2 / (\text{površina} - 1,0)$)
- Konkavnost (oštrina konkavnih delova konture)
- Konkavne tačke (broj konkavnih delova konture)
- Simetrija
- Fraktalna dimenzija ("aproksimacija obale" - 1)

Srednja vrednost, standardna greška i "najgora" ili najveća (srednja vrednost od tri najveće vrednosti) vrednost ovih karakteristika su izračunate za svaku sliku što rezultira sa 30 karakteristika. Na primer, polje 3 je srednja vrednost radiusa, polje 13 je srednja greška poluprečnika, a polje 23 je najgori poluprečnik.

Na prvi pogled je uočljivo da se baza podataka sastoji od ulaza (parametara) koji su već označeni, i izlaza – dijagnoza, što ukazuje da model koji treba da se koristi spada u grupu modela nadgledanog učenja. Shodno tome algoritmi koji su korišćeni u ovom projektu su: logistička regresija, klasifikator stabla odlučivanja, klasifikator slučajnih šuma, naivni Bajes, mašina vektora podrške i neuronske mreže.

Istraživačka analiza podataka (Exploratory Data Analysis - EDA)

Istraživačka analiza podataka ili EDA je pristup analizi bazama podataka da bi se sumirale njihove glavne karakteristike, često koristeći se statističkim graphicima i drugim metodama vizualizacije. Primarno, EDA služi za uvid u ono što nam podaci mogu reći mimo formalnog zadatka modeliranja ili testiranja hipoteza. Istraživačku analizu podataka je promovisao Džon Tukey od 1970. Godine kako bi podstakao statističare da istraže podatke i eventualno formulišu hipoteze koje bi mogle da dovedu do novih prikupljanja podataka i eksperimenata. EDA takođe olakšava filtriranje viška i uklanjanje neispravnih podataka.

Pre početka same analize, neophodno je u projekat uneti neophodne biblioteke kao što su: seaborn, matplotlib, numpy, sklearn, keras, tensorflow.

Upotrebom pandas biblioteke, podaci su učitani iz csv fajla. Prvo je neophodno sagledati podatke u bazi u celosti i potražiti sva moguća odstupanja i nepravilnosti koje je potrebno očistiti, to je uradjeno pozivanjem na slici 1.1. prikazanih funkcija.

```
data = pd.read_csv("dataBC.csv", sep=",")
print(data.head())
print("-" * 75)

print("Data shape: ", data.shape)
print("-" * 75)

print("Dataset description:")
print(data.describe())
print("-" * 75)

print("Dataset Features:")
print(data.columns.values)
print("-" * 75)

print("Number of Dataset Features that are zeros:")
print(data.isnull().sum())
print("-" * 75)

print("Dataset Categorical Features")
print(data.describe(include=['O']))
print("-" * 75)

print("Dataset Value Count:")
print(data['diagnosis'].value_counts())
print("-" * 75)
```

Slika 1.1. Pozivi funkcija za prikaz detaljnih informacija o bazi

Nakon analize izlaza uočavamo da je oblik baze pre processing-a (569, 33), od kojih ukupno ima 32 parametra i jedan ciljani izlaz. Postoji ukupno 569 instanci (redova). Koriste se `isnull().sum()` funkcije da bi se proverile moguće rupe u bazi gde su vrednosti nulte ili nepostojeće. Takav je slučaj u poslednjoj "Bezimenoj" koloni.

Potom, upotrebom `describe()` funkcije, vrši se provera srednje vrednosti, standardne devijacije i IQR vrednosti.

```
In [61]: data.describe()
```

```
Out[61]:
```

	radius_mean	texture_mean	smoothness_mean	compactness_mean	symmetry_mean	fractal_dimension_mean	radius_se	texture_se	smoothness_se
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	0.096360	0.104341	0.181162	0.062798	0.405172	1.218853	0.007041
std	3.524049	4.301036	0.014084	0.052813	0.027414	0.007080	0.277313	0.551648	0.003003
min	6.981000	9.710000	0.052630	0.019380	0.108000	0.049980	0.111500	0.360200	0.001713
25%	11.700000	16.170000	0.086370	0.064920	0.161900	0.057700	0.232400	0.833900	0.005169
50%	13.370000	18.840000	0.095870	0.092630	0.179200	0.081540	0.324200	1.108000	0.006380
75%	15.780000	21.800000	0.105300	0.130400	0.195700	0.086120	0.478900	1.474000	0.008148
max	28.110000	39.280000	0.163400	0.345400	0.304000	0.097440	2.873000	4.885000	0.031130

Picture 1.2. Computed summary of statistics

Na slici 1.3. prikazana je raspodela izlaza. Odnos malignih i benignih tumora je oko 60-40 što znači da je baza relativno izbalansirana i nije potrebna dodatna preraspodela podataka.

```
In [62]: data.diagnosis.value_counts(normalize = True)
```

```
Out[62]: B    0.627417
         M    0.372583
         Name: diagnosis, dtype: float64
```

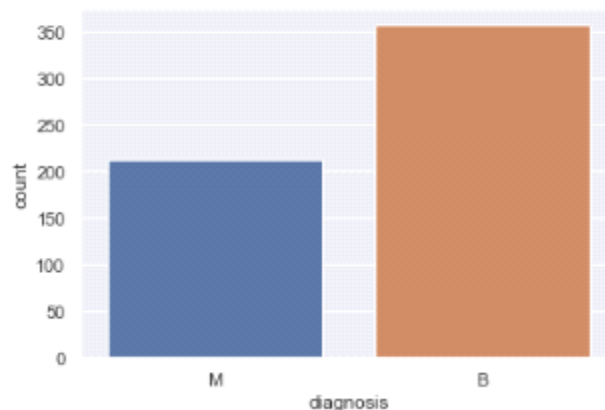
Slika 1.3. Odnos pozitivnih i negativnih izlaza

Priprema baze

Pre same pripreme baze i prikazan je grafik izlaza i odnosa izlaza. Postoji 357 instanci gde je dijagnoza "benigni tumor" (B), i 212 dijagnoza sa izlazom "maligni tumor" (M).

```
In [4]: sns.set()
        sns.countplot('diagnosis', data=data)
```

```
Out[4]: <AxesSubplot:xlabel='diagnosis', ylabel='count'>
```



Slika 2.1. Odnos dijagnoza vizualno reprezentovan

Prvi korak processing-a je čišćenje baze od informacija koje nemaju relacije sa krajnim izlazom odnosno dijagnozom. Kolona tog tipa, koju je neophodno ukloniti, je id kolona, kako informacije o pojedinačnoj osobi ne nose ništa relevantno za dijagnozu. Naredna kolona koja je još u analizi iskočila kao višak je "Bezimena" kolona čije vrednosti su nepostojeće.

Nakon čišćenja baze od nepotrebnih podataka, potrebno je sve tekstualne podatke prevesti u numeričke, kako bi modeli mašinskog učenja mogli da rade preračune sa njima. Kolona sa tekstualnim izlazom u ovom slučaju je "Dijagnoza". Vrednost "M" je zamenjena jedinicom, a vrednost izlaza "B" je zamenjena nulom.

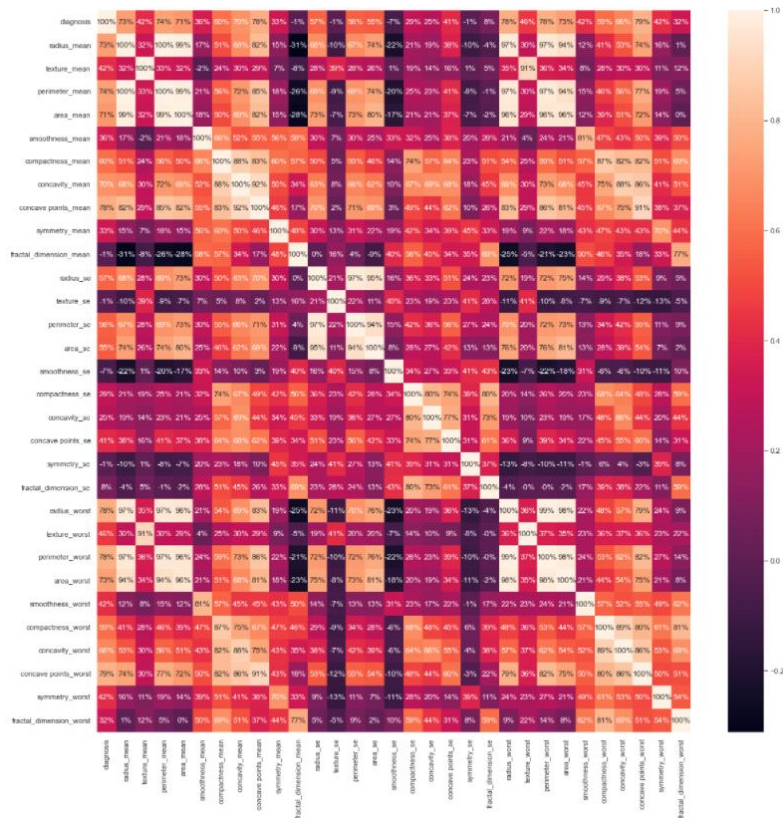
Upotrebom funkcije .nunique() vrši se provera mogućih ponavljanja unutar baze i uočava se da je takvih slučajeva jako malo, pa ta ponavljanja neće biti uklonjena.

```
In [64]: data.nunique()
Out[64]: diagnosis                2
         radius_mean             456
         texture_mean            479
         smoothness_mean         474
         compactness_mean        537
         symmetry_mean           432
         fractal_dimension_mean   499
         radius_se               540
         texture_se              519
         smoothness_se           547
         compactness_se          541
         symmetry_se             498
         fractal_dimension_se     545
         smoothness_worst        411
         compactness_worst       529
         concavity_worst         539
         concave_points_worst     492
         symmetry_worst          500
         fractal_dimension_worst  535
         dtype: int64
```

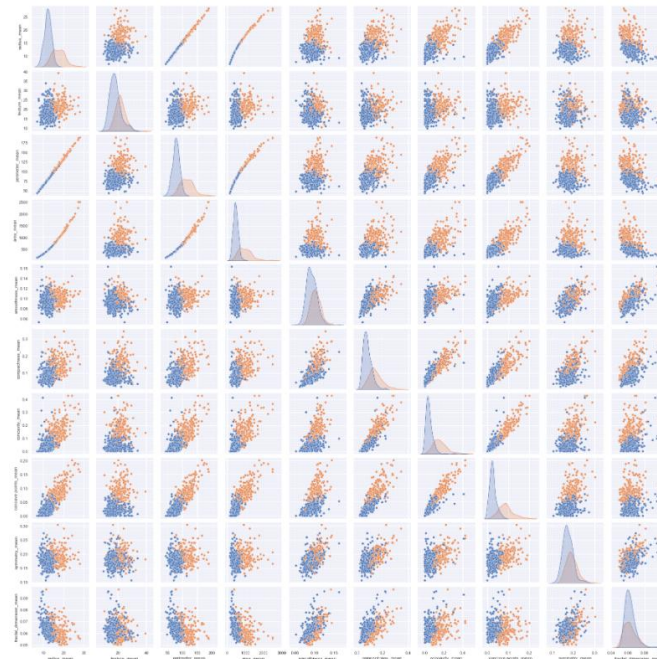
Slika 2.2. Prikaz neunikatnih vrednosti unutar kolona

Analiza veza

Prikazivanjem toplotne mape traže se moguće kolera između parametara. Na toplotnoj mapi uočljivi su veliki procenti, to definitivno ukazuje na korelaciju. Radi jasnije vidljivosti, iscrtava se graf uparenih parametara. Svi grafovi koji imaju gotovo linearni oblik su radius, perimeter i area atributi ukazuju na prisustvo multikolinearnosti. Multikolinearnost je pojava visokih interkorelacija između dve ili više nezavisnih promenljivih u modelu višestruke regresije. Uopšteno govoreći, multikolinearnost može dovesti do širih intervala poverenja koji mogu proizvesti manje pouzdane verovatnoće u smislu uticaja nezavisnih varijabli u modelu. Drugi skup varijabli koji može da implicira multikolinearnost su concavity, concave_points i compactness.



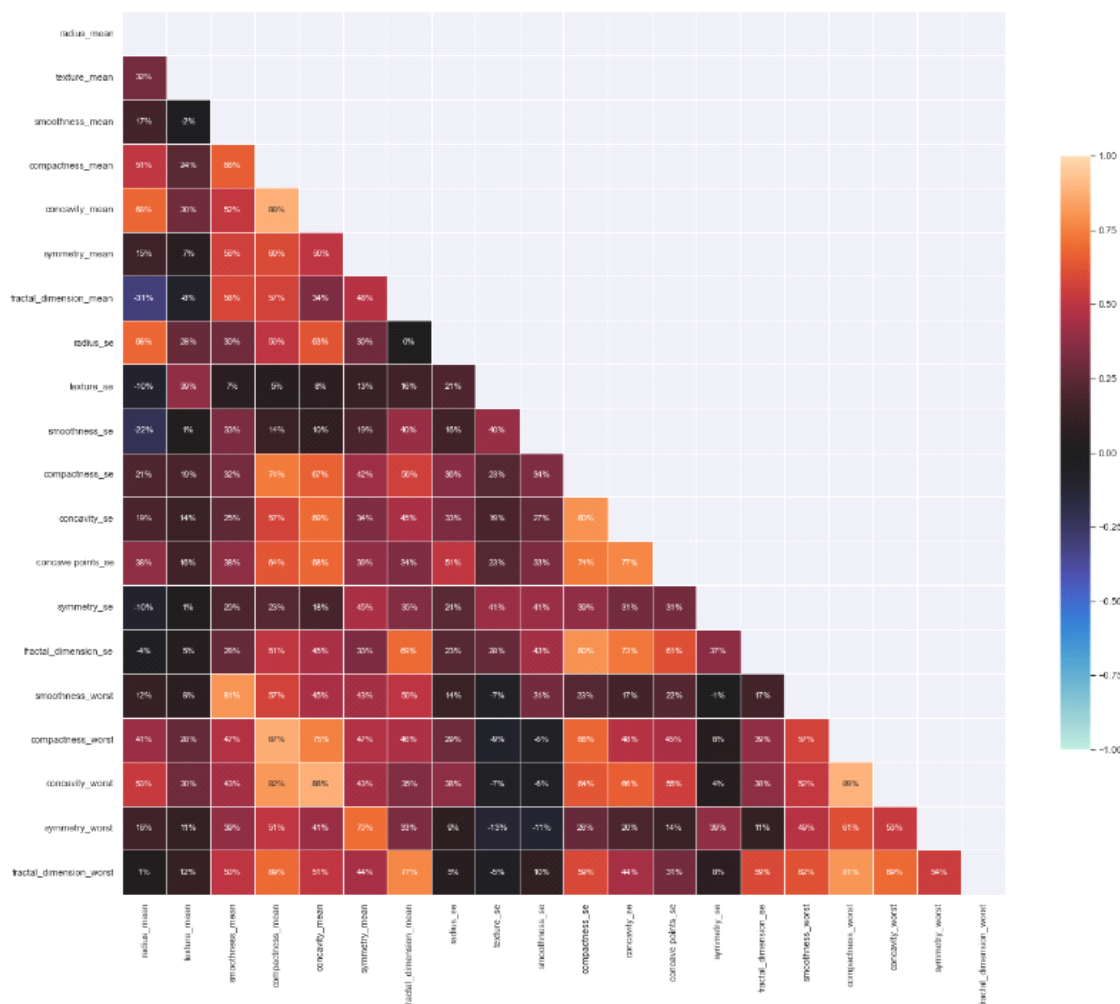
Slika 3.1. Korelaciona toplotna mapa



Slika 3.2. Upareni grafovi izmedju parametara

Generisanjem korelacione matrice i verifikovano je prisustvo multikolinearnosti izmedju nekih parametara. Na primer, kolona radius_mean ima korelaciju od 1 i 0.99 sa perimeter_mean i area_mean, respektivno. To znači da ove tri kolone u suštini sadrže iste informacije, a to je fizička veličina

posmatrane ćelije. Prvi pokušaj je bio uklanjanje svih parametara čija je koleracija na oko visoka, to je pogoršalo rezultate modela mašinskog učenja. Odlučeno je da se proces odabira korelacije automatizuje dodavanjem for loop-a koji hvata svaku korelaciju veću od 90%. Tehnički, najbolje bi bilo ukloniti sve korelacije veće od 70%, ali kako sama baza nema pozamašan broj parametara to bi samo ugrozilo predviđanje. Prikaz nove korelacione matrice ukazuje na bolju bazu.



There is no more big correlations that would slow down the calculating process.

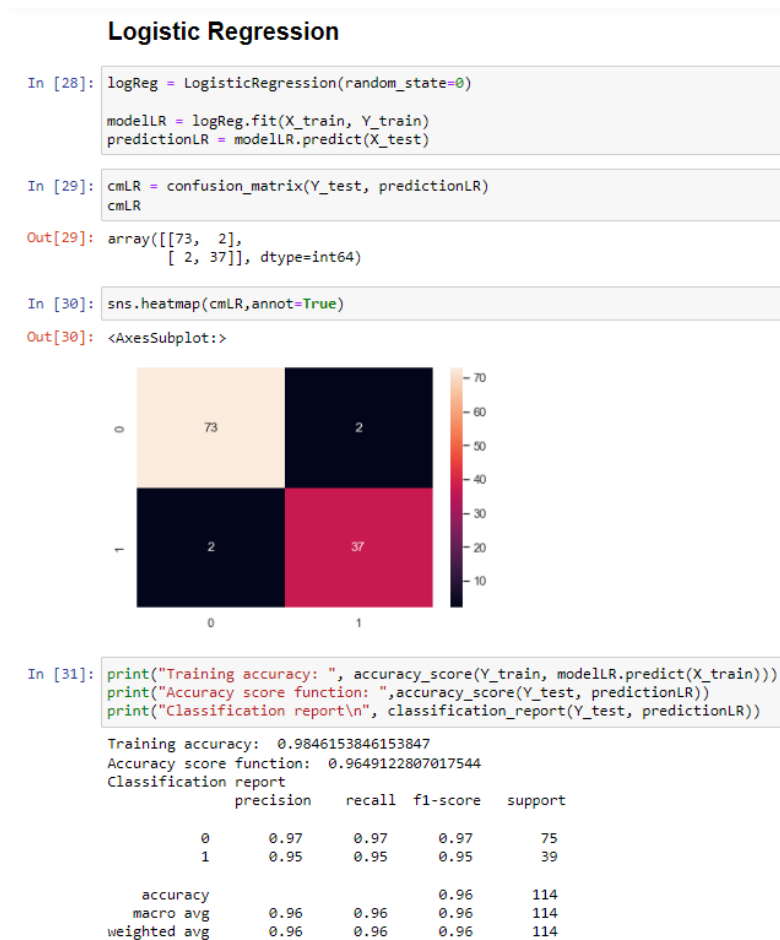
Slika 3.3. Korelaciona toplotna mapa nakon obrade podataka

Nakon završetka uklanjanja redundantnih podataka i čišćenja baze, prvobitno su podaci podeljeni u 2 grupe train i test, a kasnije primenom modela neuronskih mreža podaci su ponovo podeljeni u train, test i validate setove, i to po 80-20 i 80-10-10, respektivno. Ovim je obrada podataka gotova i upotrebnom funkcije `fit_transform()` na `X_train` transformisani su podaci. Način ka koji `StandardScaler` standardizuje parametre je upotrebom $\mu=0$ (mean), $\delta=1$ (standard deviation), koja je prethodno naučena iz podataka.

Modeli i analiza performansi

Logistička regresija

Logistička regresija je statističko model koji se koristi za izračunavanje verovatnoće nekog događaja ili određene klase kao što su, npr. maligni/benigni tumor. Matematički, binarni logistički model ima zavisnu promenljivu sa dve moguće vrednosti, kao što je benigno/maligno koje je predstavljeno indikatorskom promenljivom, gde su dve vrednosti označene sa „0” i „1”. Logistička regresija u svojoj osnovi koristi logističku funkciju koja se takođe naziva sigmoidna funkcija koja je matematička funkcija koja ima karakterističnu krivu u obliku slova „S”.



Analiziranje performansi je najlakše korišćenjem matrice konfuzije. Matrica konfuzije je tehnika sumiranja performansi klasifikacionog algoritma. Na [0][0] poziciji dobijamo tačno pozitivne rezultate – tumor je maligni i model mu daje pravu dijagnozu, [1][1] Tačno negativan – tumor je benigni i model ga je klasifikovao tako, [0][1] Lažno pozitivan – tumor je benigni, ali ga model ne dijagnostikuje ispravno, [1][0] Lažno negativan – tumor je maligni, ali ga model tako ne klasifikuje. Upoređujući tačnost treninga i testiranja, primećuje se da nema velike razlike između rezultata, što znači da nema overfit-ovanja i da logistička regresija ima zaista visoku tačnost od 97%.

Stablo odlučivanja

Stablo odlučivanja je klasifikator koji ima čvorove i grane u kojima se koristi da podeli podatke u različite kategorije. Konstruiše različite kategorije rekurzivnom evaluacijom podataka. Ova procedura se ponavlja ažuriranjem čvorova sve dok podaci ne budu ispravno razdvojeni.

Decision Tree

```
In [39]: decTree = DecisionTreeClassifier(criterion='entropy', random_state=0)

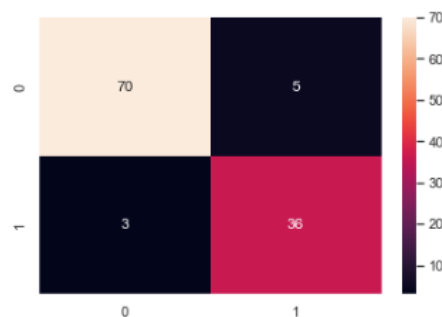
modelDT = decTree.fit(X_train, Y_train)
predictionDT = modelDT.predict(X_test)
```

```
In [40]: cmDT = confusion_matrix(Y_test, predictionDT)
cmDT
```

```
Out[40]: array([[70,  5],
               [ 3, 36]], dtype=int64)
```

```
In [41]: sns.heatmap(cmDT,annot=True)
```

```
Out[41]: <AxesSubplot:>
```



```
In [42]: print("Training accuracy: ", accuracy_score(Y_train, modelDT.predict(X_train)))
print("Accuracy score function: ", accuracy_score(Y_test, predictionDT))
print("Classification report\n", classification_report(Y_test, predictionDT))
```

```
Training accuracy: 1.0
Accuracy score function: 0.9298245614035088
Classification report
precision    recall  f1-score   support

     0       0.96    0.93    0.95        75
     1       0.88    0.92    0.90        39

 accuracy          0.93        114
  macro avg       0.92    0.93    0.92        114
 weighted avg     0.93    0.93    0.93        114
```

```
In [43]: # For decision tree I could probably use less training data since this
# Seems like a overfit
```

Kod stabla odlučivanja preciznost se ne može uzeti za validan rezultat kako ovi proračuni ukazuju na overfit-ovanje. Prevelika razlika između trening skupa i testing skupa ukazuje na prekomerno podešavanje hiperparametara trening skupu što kasnije model čini nespremim za bilo kakvo testiranje. Rešenje za ovaj problem je povećanje broja podataka ili regularizacija. Zbog velikog broja analiziranih modela u ovom seminarskom radu, ovaj model nije dalje štelovan.

Slučajna šuma

Slučajna šuma je, na najjednostavniji način objašnjena, grupa stabala odlučivanja. Svako stablo odlučivanja daje „glas“ u konačnoj odluci tako što samostalno donosi odluku o rezultatu. Odluka za koju se „glasa“ najviše puta dobija se rezultat.

Random Forest

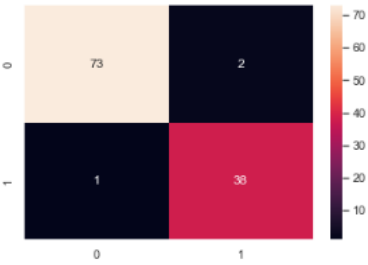
```
In [44]: randForest = RandomForestClassifier(criterion='entropy', random_state=0)
         modelRF = randForest.fit(X_train,Y_train)
         predictionRF = modelRF.predict(X_test)

In [45]: cmRF = confusion_matrix(Y_test, predictionRF)
         cmRF

Out[45]: array([[73,  2],
               [ 1, 38]], dtype=int64)

In [46]: sns.heatmap(cmRF,annot=True)

Out[46]: <AxesSubplot:>
```



```
In [47]: print("Training accuracy: ", accuracy_score(Y_train, modelRF.predict(X_train)))
         print("Accuracy score function: ",accuracy_score(Y_test, predictionRF))
         print("Classification report\n", classification_report(Y_test, predictionRF))

Training accuracy: 1.0
Accuracy score function: 0.9736842105263158
Classification report
      precision    recall  f1-score   support

      0       0.99      0.97      0.98         75
      1       0.95      0.97      0.96         39

   accuracy          0.97
  macro avg          0.97
 weighted avg          0.97
```

```
In [48]: # neophodna regularizacija!
```

Klasifikator slučajnih šuma ima veću tačnost od stabla odlučivanja za 5% i ovde je overfit i dalje prisutan, ali znatno manje.

Support Vector Machine

S obzirom da SVM klasifikatori obično nude dobru tačnost, rade sa klasifikacijom i regresijom i nisu pogodni za velike skupove podataka zbog dugog vremena obuke, ovaj model zvuči zaista obećavajuće. Glavni cilj SVM-a je da izdvoji dati skup podataka na najbolji mogući način. Udaljenost između bilo koje najbliže tačke je poznata kao margina. Ideja je da se izabere hiperravan sa maksimalnom mogućom marginom između vektora podrške u datom skupu podataka.

SVM

```
In [49]: from sklearn import svm

svm = svm.SVC(kernel='linear')

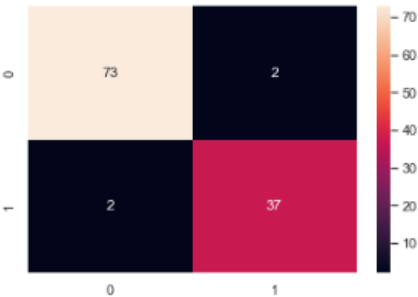
modelSVM = svm.fit(X_train, Y_train)
predictionSVM = modelSVM.predict(X_test)

In [50]: cmSVM = confusion_matrix(Y_test, predictionSVM)
cmSVM

Out[50]: array([[73,  2],
               [ 2, 37]], dtype=int64)

In [51]: sns.heatmap(cmSVM,annot=True)

Out[51]: <AxesSubplot:>
```



```
In [52]: print("Training accuracy: ", accuracy_score(Y_train, modelSVM.predict(X_train)))
print("Accuracy score function: ",accuracy_score(Y_test, predictionSVM))
print("Classification report\n", classification_report(Y_test, predictionSVM))

Training accuracy:  0.9824175824175824
Accuracy score function:  0.9649122807017544
Classification report
      precision    recall  f1-score   support

      0       0.97      0.97      0.97        75
      1       0.95      0.95      0.95        39

   accuracy          0.96          0.96          0.96       114
  macro avg          0.96          0.96          0.96       114
 weighted avg          0.96          0.96          0.96       114
```

Support Vector Machine ostvaruje preciznost od 96.5% i ostali parametri ne ukazuju na overfit ili undefit što ovaj model može definisati kao dobar i u ovom slučaju primenljiv.

Naivni Bajes

U statistici Naivni Bajes klasifikatori su porodica “klasifikatora verovatnoće” zasnovanih na primeni Bajesove teoreme sa jakim (naivnim) pretpostavkama nezavisnosti između karakteristika. Naivni Bajes klasifikatori su veoma skalabilni i zahtevaju niz parametara koji su linearni u broju varijabli (karakteristike/prediktori) u problemu učenja. Obuka maksimalne verovatnoće se može obaviti procenom izraza zatvorene forme, za koji je potrebno linearno vreme, umesto skupom iterativnom aproksimacijom koja se koristi za mnoge druge tipove klasifikatora.

Gaussian Naive Bayes

```
In [34]: from sklearn.naive_bayes import GaussianNB
```

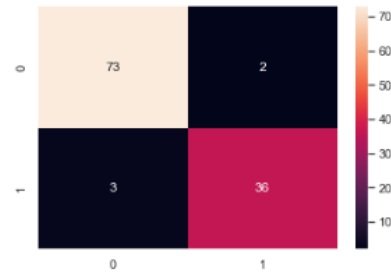
```
In [35]: gnb = GaussianNB()  
model_gnb = gnb.fit(X_train, Y_train)  
prediction_gnb = gnb.predict(X_test)
```

```
In [36]: cm_gnb = confusion_matrix(Y_test, prediction_gnb)  
cm_gnb
```

```
Out[36]: array([[73,  2],  
               [ 3, 36]], dtype=int64)
```

```
In [85]: sns.heatmap(cm_gnb,annot=True)
```

```
Out[85]: <AxesSubplot:>
```



```
In [37]: print("Training accuracy: ", accuracy_score(Y_train, model_gnb.predict(X_train)))  
print("Accuracy score function: ",accuracy_score(Y_test, prediction_gnb))  
print("Classification report\n", classification_report(Y_test, prediction_gnb))
```

```
Training accuracy: 0.9098901098901099  
Accuracy score function: 0.956140350877193  
Classification report
```

	precision	recall	f1-score	support
0	0.96	0.97	0.97	75
1	0.95	0.92	0.94	39
accuracy			0.96	114
macro avg	0.95	0.95	0.95	114
weighted avg	0.96	0.96	0.96	114

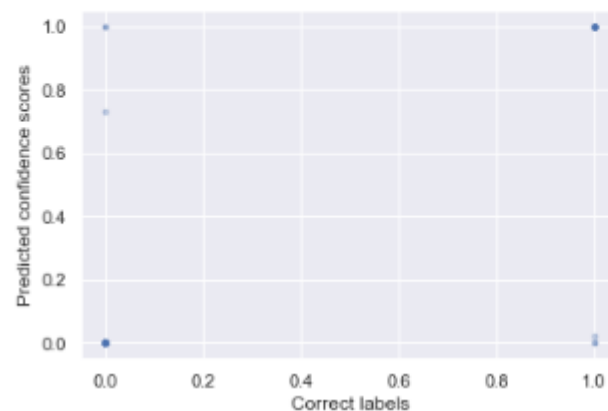
```
In [38]: # these results tell me that I could be using more of the training set,  
# but considering Naive Bayes rarely has better results than other algorithms  
# I won't change the train/test split
```

ANN

Prilikom kreiranja neuronske mreže radila se ponovna podela seta podataka. Ovog puta imamo 3 pod skupa, a to su: train, test, validate. Sva tri skupa su pri podeli prethodno izmešana, a potom su skupovi parametara normalizovani kako bi rad sa neuronskim mrežama bio lakši i precizniji. Nakon toga kreirane su 2 neuronske mreže. Prva sadrži 2 skrivena sloja sa relu aktivacionim funkcijama i po 50 neurona. Druga se sastoji od 3 skrivena sloja sa relu aktivacionim funkcijama koje redom imaju 32, 64, 128 neurona. Learning rate je svuda jednak 0.001 što je standardna vrednost u mašinskom učenju. Metrika koja se prati je accuracy.

Prva neuronska mreža vraća sledeće rezultate na validacionom skupu:

```
In [76]: plt.plot(Y_valid, prediction, '.', alpha=0.3)
plt.xlabel('Correct labels')
plt.ylabel('Predicted confidence scores')
plt.show()
```



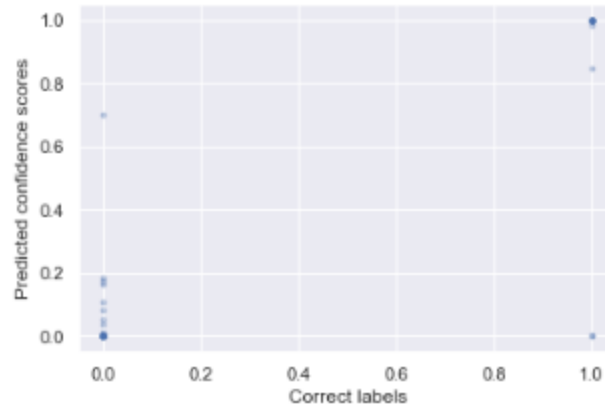
```
In [77]: from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

```
In [78]: accuracy= accuracy_score(Y_valid, prediction.round())
precision= precision_score(Y_valid, prediction.round())
recall= recall_score(Y_valid, prediction.round())
f1= f1_score(Y_valid, prediction.round())
print("Accuracy: %.2f%%" % (accuracy*100.0))
print("Precision: %.2f%%" % (precision*100.0))
print("Recall: %.2f%%" % (recall*100.0))
print("F1-score: %.2f%%" % (f1*100.0))
```

```
Accuracy: 89.47%
Precision: 84.21%
Recall: 84.21%
F1-score: 84.21%
```

Ova neuronska mreža očigledno može biti dodatno poboljšana dodatnim slojevima, pa je dobro preći na drugu neuronsku mrežu i sagledati njene rezultate.

```
In [76]: plt.plot(Y_valid, prediction, '.', alpha=0.3)
plt.xlabel('Correct labels')
plt.ylabel('Predicted confidence scores')
plt.show()
```



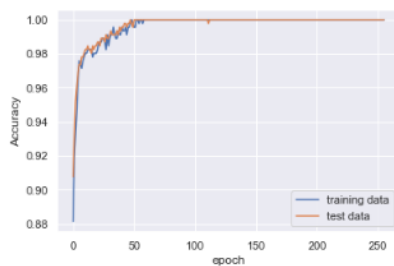
```
In [77]: from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

```
In [78]: accuracy= accuracy_score(Y_valid, prediction.round())
precision= precision_score(Y_valid, prediction.round())
recall= recall_score(Y_valid, prediction.round())
f1= f1_score(Y_valid, prediction.round())
print("Accuracy: %.2f%%" % (accuracy*100.0))
print("Precision: %.2f%%" % (precision*100.0))
print("Recall: %.2f%%" % (recall*100.0))
print("F1-score: %.2f%%" % (f1*100.0))
```

```
Accuracy: 94.74%
Precision: 94.44%
Recall: 89.47%
F1-score: 91.89%
```

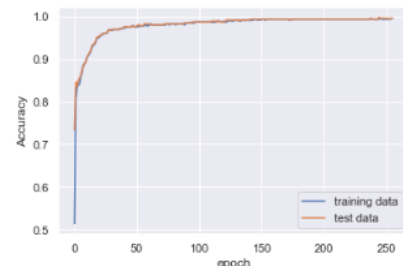
Produbljenje neuronske mreže zaista poboljšava rezultate. To je uočljivo i na poređenju rezultata training i test seta kod prve i druge neuronske mreže.

```
In [69]: plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.ylabel('Accuracy')
plt.xlabel('epoch')
plt.legend(['training data', 'test data'], loc='lower right')
plt.show()
```



Slika levo – prva neuronska mreža

```
In [79]: plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.ylabel('Accuracy')
plt.xlabel('epoch')
plt.legend(['training data', 'test data'], loc='lower right')
plt.show()
```



Slika desno – druga neuronska mreža

Zaključak

Nauka se godinama rapidno razvija na svim poljima, a ponajviše na tehnološkim, međutim ljudski strah i stigma koja okružuje sve što ne sadrži ljudski faktor usporava ulazak veštačke inteligencije u primenjenu medicinu. Terminalne bolesti postaju terminalne propuštenim vremenom i neispravnim dijagnozama. Statistika pokazuje da bi uvođenje još jednog stručnog mišljenja koje nije od “krvi i mesa” sigurno umanjilo propušteno vreme i šanse za pogrešne dijagnoze. Mašina, još uvek, ne bi donosila kranju odluku, mogla bi da u umu naučnika probudi pitanje “Da li...?” koje bi u gužvi pandemije bilo zanemareno, a u životu pojedinca presudno.

Literatura

- <https://www.kaggle.com/uciml/breast-cancer-wisconsin-data>
- <https://scikit-learn.org/stable/>
- http://theta.edu.pl/wp-content/uploads/2012/10/exploratorydataanalysis_tukey.pdf
- https://en.wikipedia.org/wiki/Logistic_regression
- https://en.wikipedia.org/wiki/Decision_tree
- https://en.wikipedia.org/wiki/Random_forest
- https://en.wikipedia.org/wiki/Naive_Bayes_classifier
- https://en.wikipedia.org/wiki/Artificial_neural_network