Slovenská technická univerzita v Bratislave
Fakulta elektrotechniky a informatiky

Neurónové siete

Zadanie č.1

2021 **Jakub Jančík**

Obsah

Zoznam obrázkov

1 Načitanie a spracovanie dát

Prvým krokom je načítanie údajov pomocou knižnice **pandas** do nášho DataFrame-u. Škálovanie dát sme uskutočnili pomocou knižnice MinMaxScaler.

Priemerna hodnota pred normalizaciou - water train.

```
ph                    7.041691
Hardness            196.310660
Solids            21756.102873
Chloramines           7.163238
Sulfate             335.505887
Conductivity        424.118360
Organic_carbon       14.260028
Trihalomethanes      66.115472
Turbidity             4.027384
Potability            0.390244
```

Št. odchýlka pred normalizaciou - water train.

```
ph                    1.595465
Hardness             34.804883
Solids             8796.578151
Chloramines           1.621486
Sulfate              41.991894
Conductivity         80.942866
Organic_carbon        3.257755
Trihalomethanes      17.125366
Turbidity             0.794728
Potability            0.488177
```

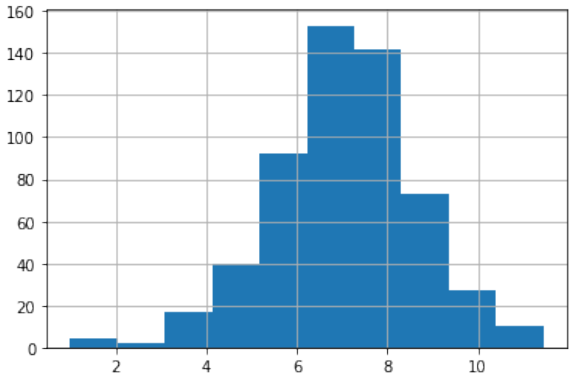
Priemerna hodnota PO normalizacií - water train.

```
ph                  0.578573
Hardness            0.474704
Solids              0.351937
Chloramines         0.441357
Sulfate             0.514041
Conductivity        0.435472
Organic_carbon      0.467340
Trihalomethanes     0.548045
Turbidity           0.468048
Potability          0.390244
```

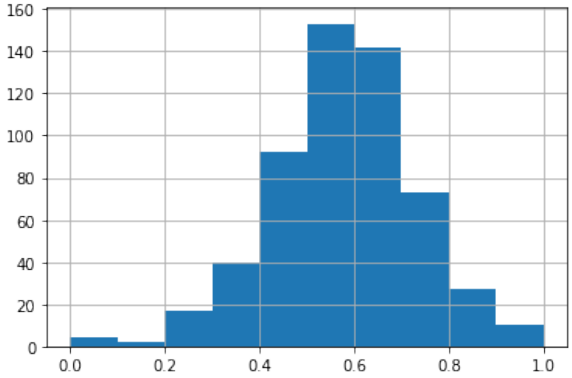
Št. odchýlka PO normalizacií - water train.

```
ph                  0.152533
Hardness            0.144171
Solids              0.144428
Chloramines         0.157565
Sulfate             0.145519
Conductivity        0.200047
Organic_carbon      0.165337
Trihalomethanes     0.143558
Turbidity           0.155906
Potability          0.488177
```

Obr. 1: Stlpec ph pred škalovaním.

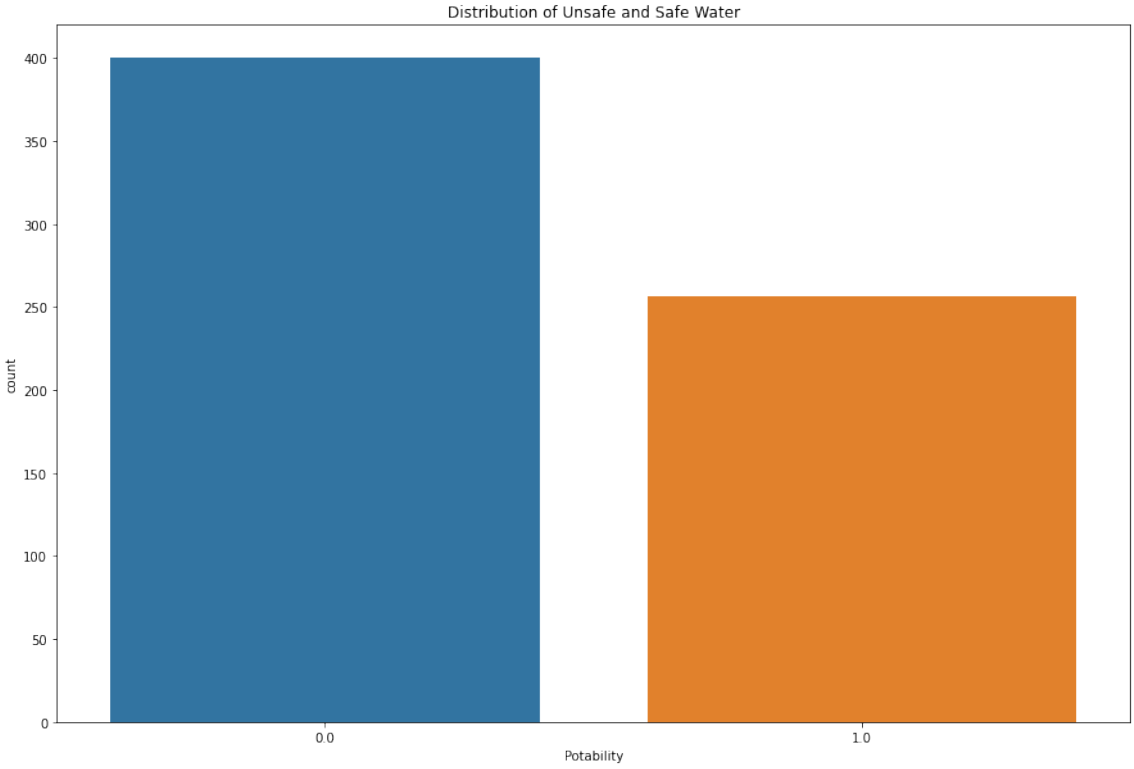


Obr. 2: Stlpec ph po škálovaní.



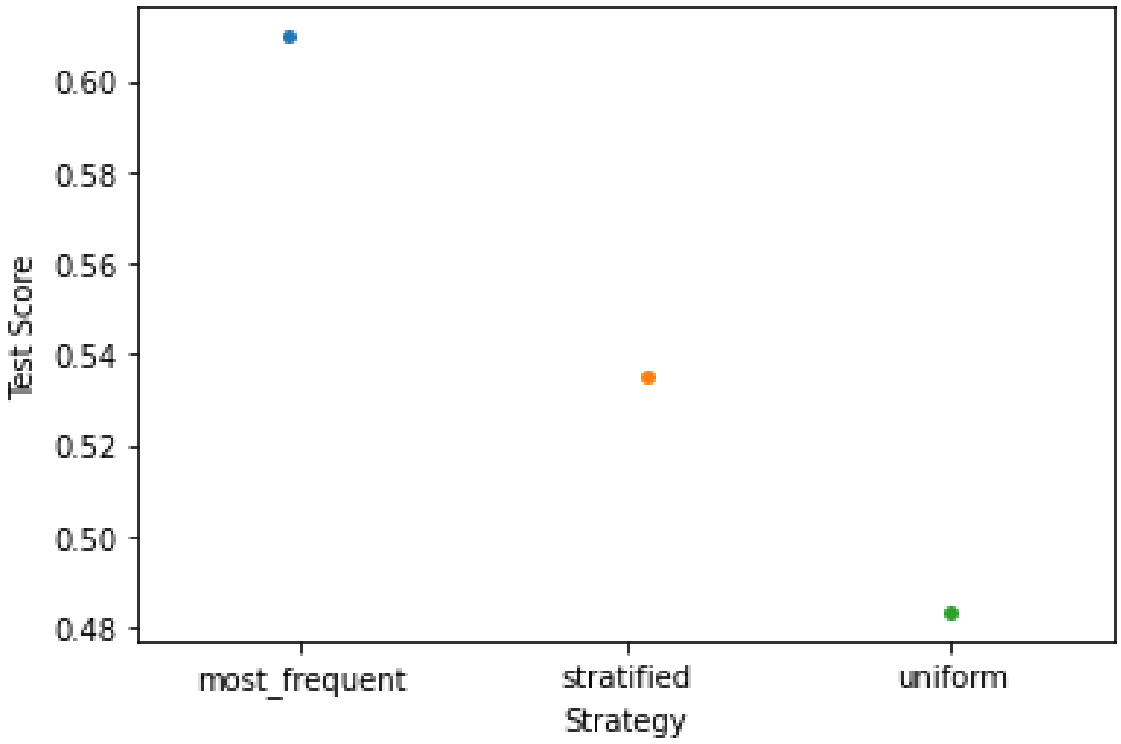
Výsledky máme rovnake avšak, práve po normalizácií (od 0 po 1) dát nám umožňuje modelu konvergovať k lepším hmotnostiam a následne vedie k presnejšiemu modelu.

Obr. 3: Zobrazenie pitnosť vody v našom datasete.



Podľa tohto grafu vieme určit, že náš dataset obsahuje väčšinu nepitnej vody (označené číslom 0).

Obr. 4: DummyClassifier strategy.

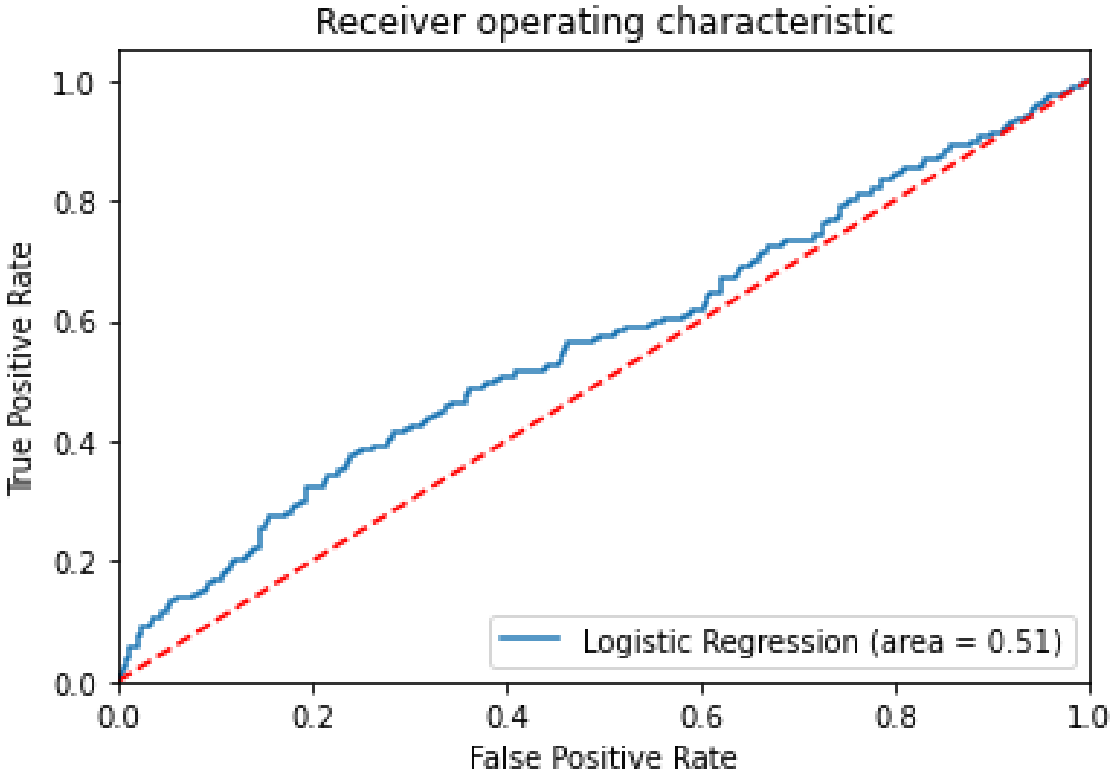


2 Zhodnotenie náročnosti problému

Pri strategií najfrekventovanejši klasifikátor vždy predpovedá najčastejšie označenie triedy v trénovacích údajoch to nám vyšlo 60.1 percent.Pri strategií stratifikovaný nám vyšla presnosť približne 53.5 percent, ktorý generuje náhodné predpovede rešpektovaním distribúcie tried tréningovej množiny. Pri stratégií uniform nám generuje predpovede rovnomerne náhodne kde nám vyšlo 48.3 percent.

Klasifikácia pomocou logistickej regresie na testovacej množine nám presnosť vyšla 62 percent.

Obr. 5: Klasifikácia pomocou log. regiresie.



Bodkovaná čiara predstavuje ROC krivku čisto náhodného klasifikátora; dobrý klasifikátor zostáva čo najďalej od tejto čiary (smerom k ľavému hornému rohu).

3 Trénovanie, vyhodnotenie siete a nájdenie dobrých parametrov

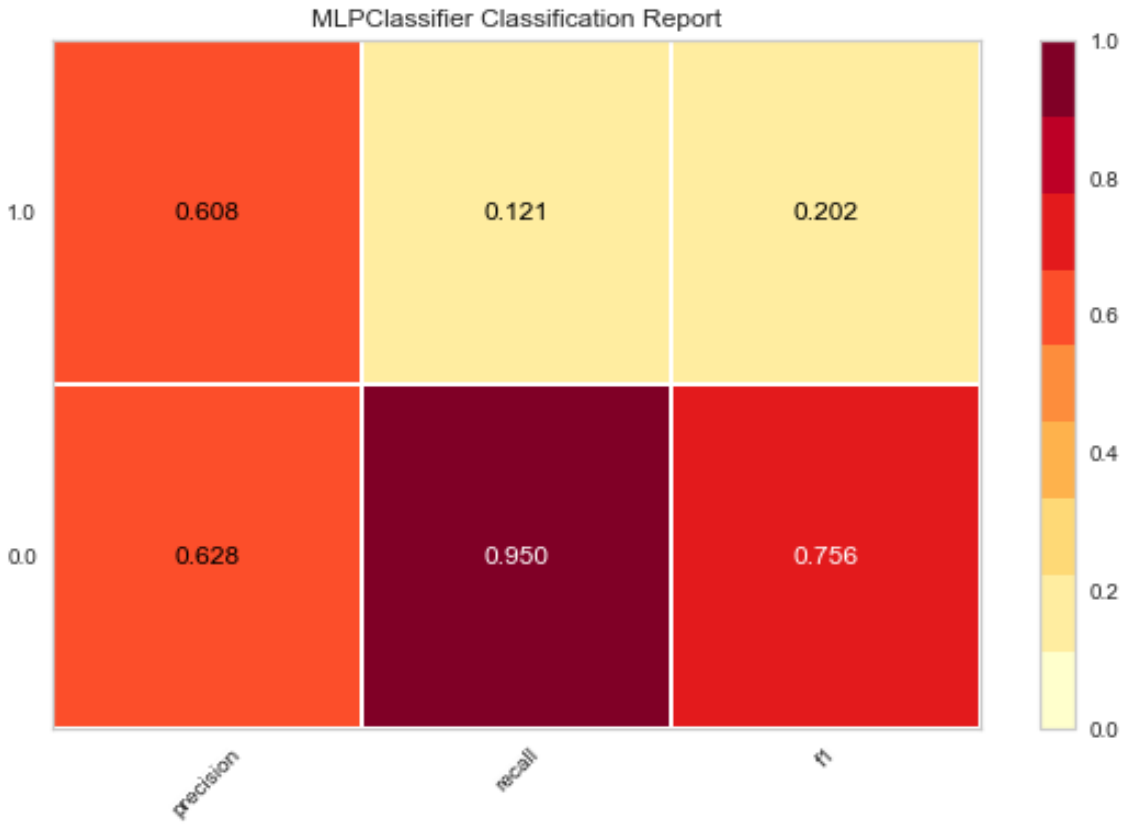
```
x_train, x_val, y_train, y_val = train_test_split(
```

```
    x_train, y_train, test_size=0.2, random_state=1)

mlpc = MLPClassifier(activation='tanh', solver='adam', alpha=0.0001,
   learning_rate_init= 0.01, max_iter=50,  hidden_layer_sizes=(12, 6)
   , random_state=0)
mlpc = mlpc.fit(x_val,y_val)
result = mlpc.predict(x_test)
```

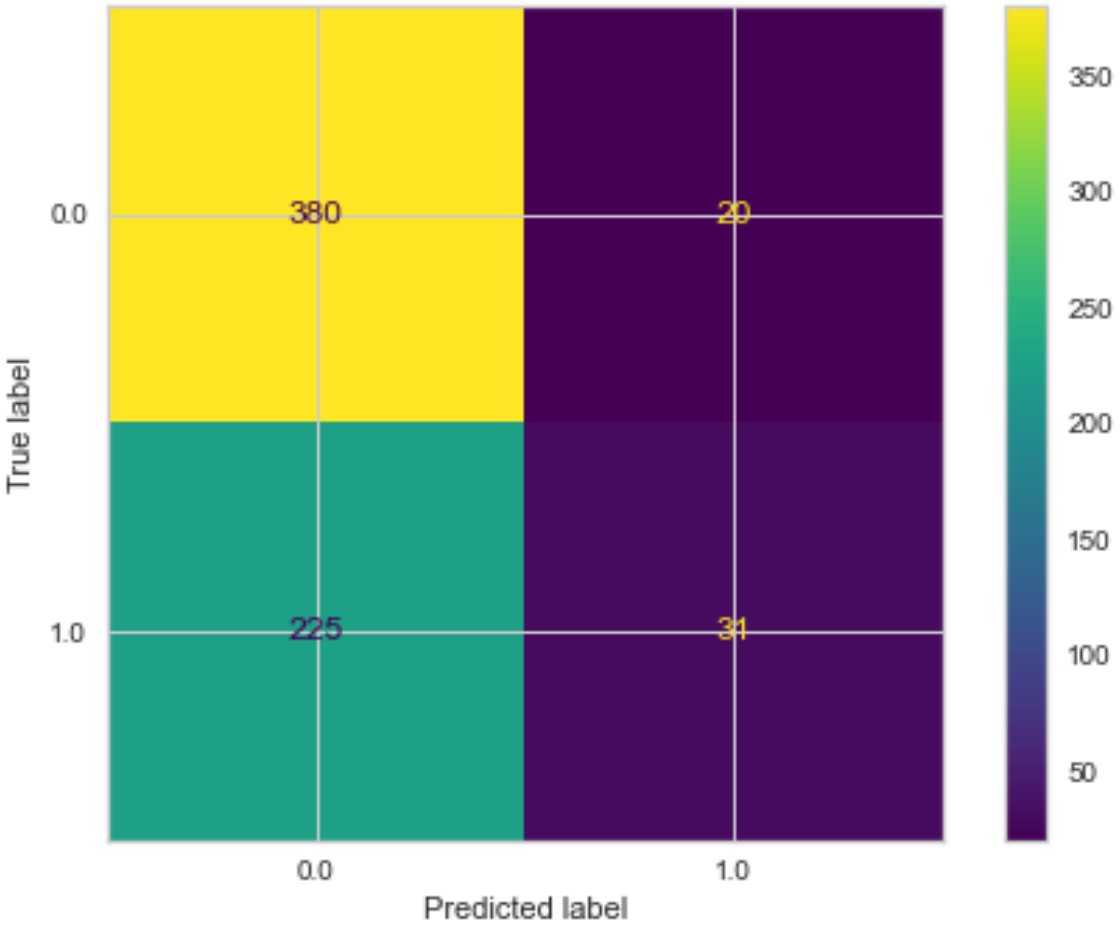
Na trénovanie sme použili tiedu MLPClassifier s aktivačnou funkciou - the hyperbolic tan function, solver je použítý defaultne adam. Skryté vrstvy s počtom 12 a 6. Počet iterácií sme nastavili na 50. Validačné dáta sme rozdelili z trénovacej množniny v pomere 80:20.

Obr. 6: MLPClassifier klasifikačný report.

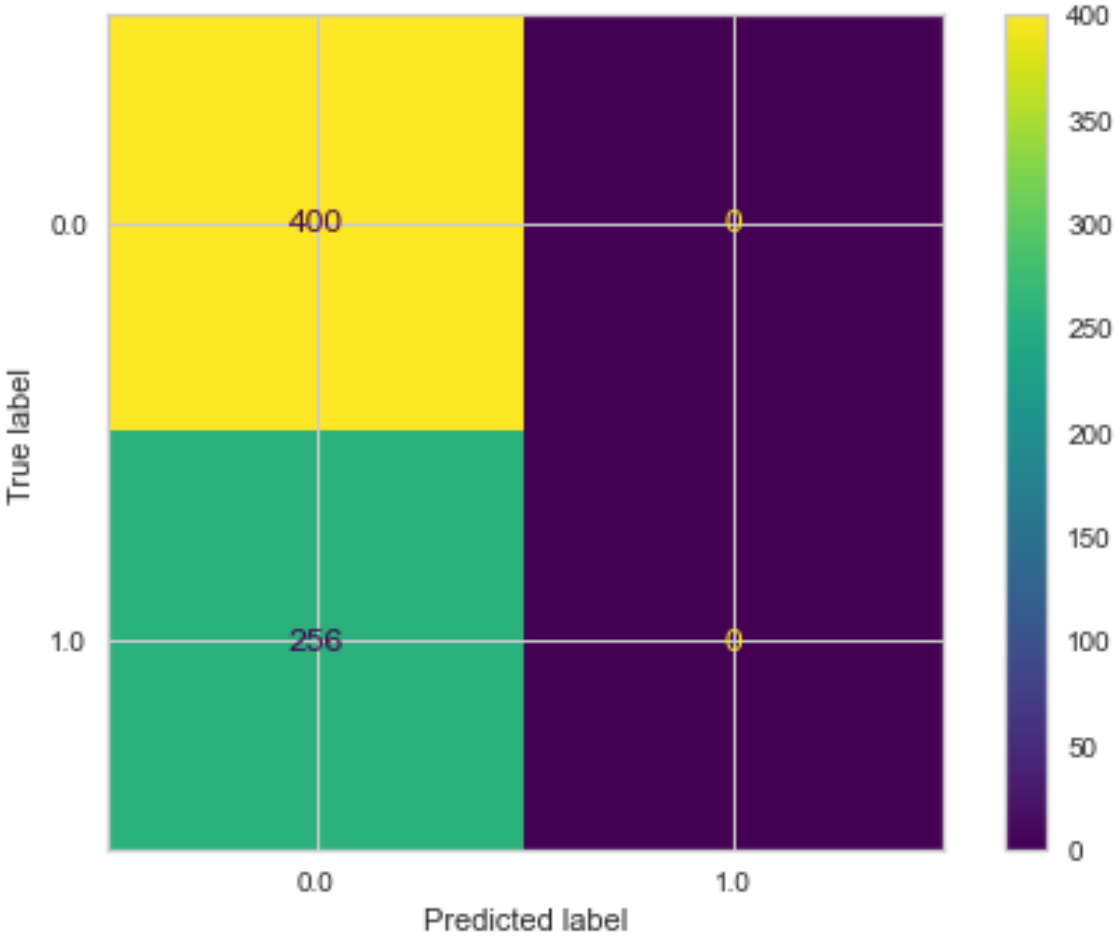


Presnosť pri tejto klasifikácií nám vyšla 62 percent.

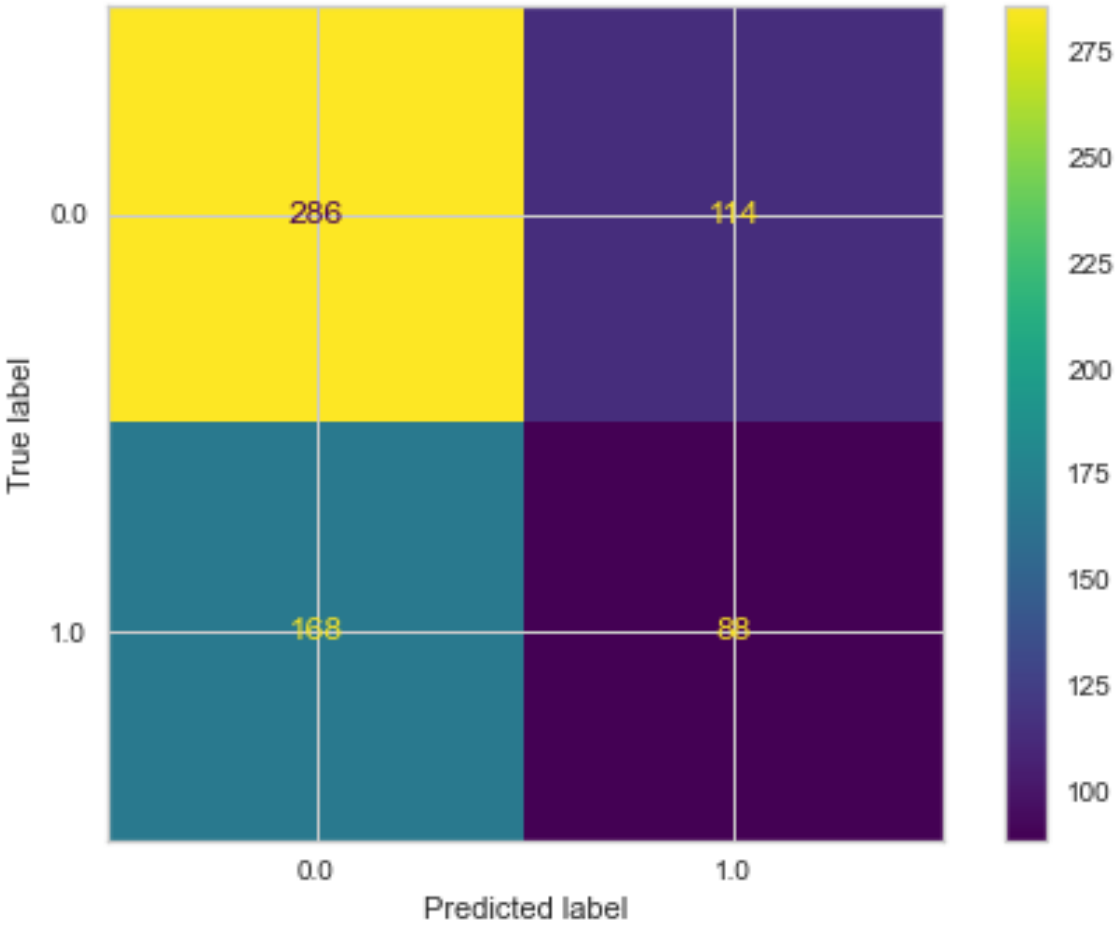
Obr. 7: Konfuzna matica pre treningovy set pri solveri adam.



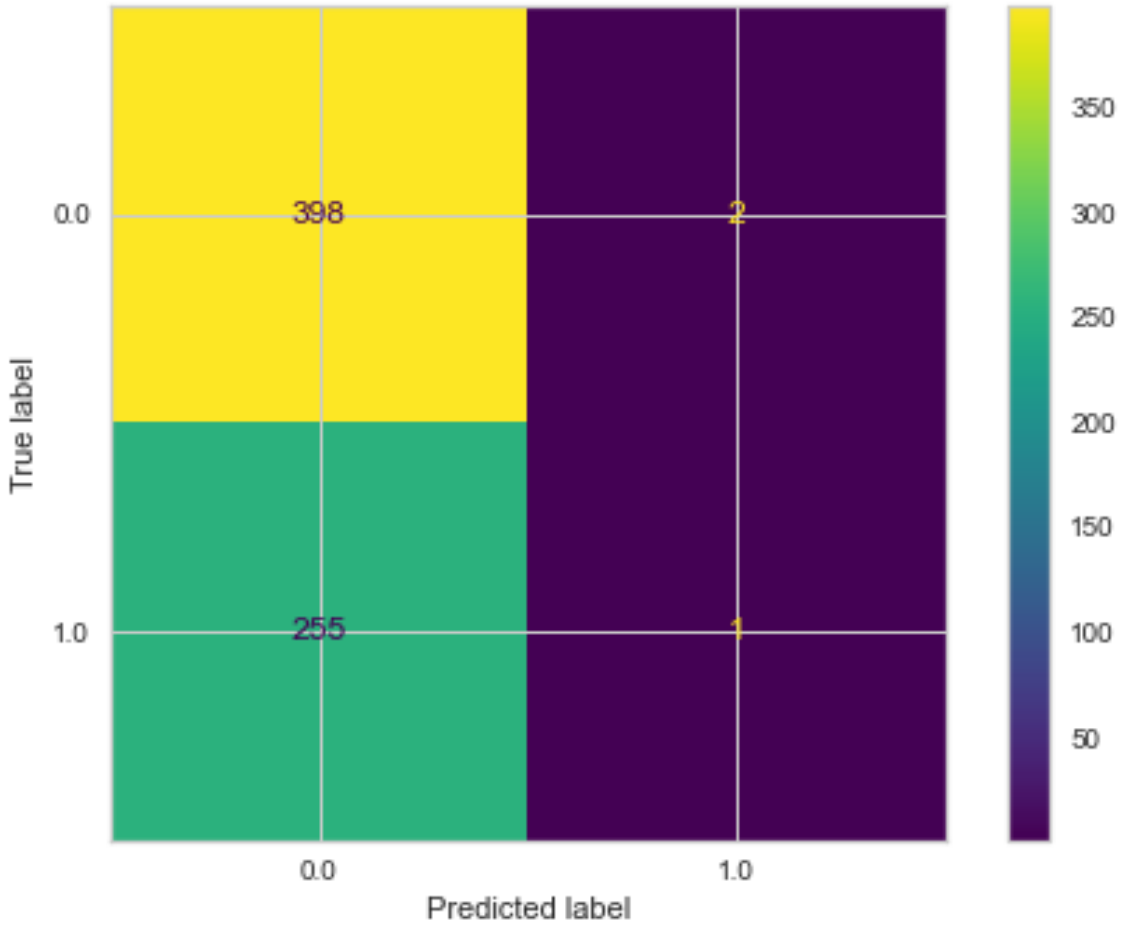
Obr. 8: Konfuzna matica pre treningovy set pri solveri SGD.



Obr. 9: Konfuzna matica pre validačný set pri solveri adam.



Obr. 10: Konfuzna matica pre validačný set pri solveri SGD.



Použili sme solver SGD.Max iter. 50 a skrytými vrstavami 18,12 8. Pri validačnej množine nám vyšla presnosť 54 percent a pri tréningovej množine nám vyšla presnosť siete 62.59 percent.

Zdá sa, že lepšie výsledky sme dosiahli pre trénongovú množinu aj pomocou solvera SGD. Môžeme to vidieť na obr. číslo 10 pri konfúznej matici a pri celkovej úspešnosti 62.6 percent.

```
a = cross_val_score(mlpc, x_train, y_train, cv=5,
                    scoring='accuracy').mean()*100
print("%f " % a)
```

Skúšali sme meniť rôzne nastavenia ako sú solvery, počet skrytých vrstiev, počet iterácií či rýchlosť učenia. Lepšie výsledky sme ziskali keď sme nastavili solver SGD. Pri nastavení rýchlosti učenia, aktivačnej funkcí sme získali najväššiu presnosti a to 71.6 percent.

4 Grid search

Teraz sa pozrime, ako použiť GridSearchCV na zlepšenie presnosti nášho modelu.

```
param_grid = {'C': [0.1, 1, 10, 100],
              'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
              'gamma':['scale', 'auto'],
              'kernel': ['linear']}
grid = GridSearchCV(SVC(), param_grid, refit = True, verbose = 3,
   n_jobs=-1)
# fitting the model for grid search
grid.fit(x_train, y_train)

# print best parameter after tuning
print(grid.best_params_)
grid_predictions = grid.predict(x_test)

# print classification report
print(classification_report(y_test, grid_predictions))
```

```
{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
              precision    recall  f1-score   support

         0.0       0.61      1.00      0.76       400
         1.0       0.00      0.00      0.00       256

    accuracy                           0.61       656
   macro avg       0.30      0.50      0.38       656
weighted avg       0.37      0.61      0.46       656
```

Môže sa zdať, že ‘C‘: 100, ‚gamma‘: ‚scale‘, ‚kernel‘: ‚linear‘ sú najlepšie hodnoty pre hyperparametre pre model SVM. No nemusí to tak byť. Ale pre akýkoľvek iný súbor údajov môže mať model SVM rôzne optimálne hodnoty pre hyperparametre, ktoré môžu zlepšiť jeho výkon.

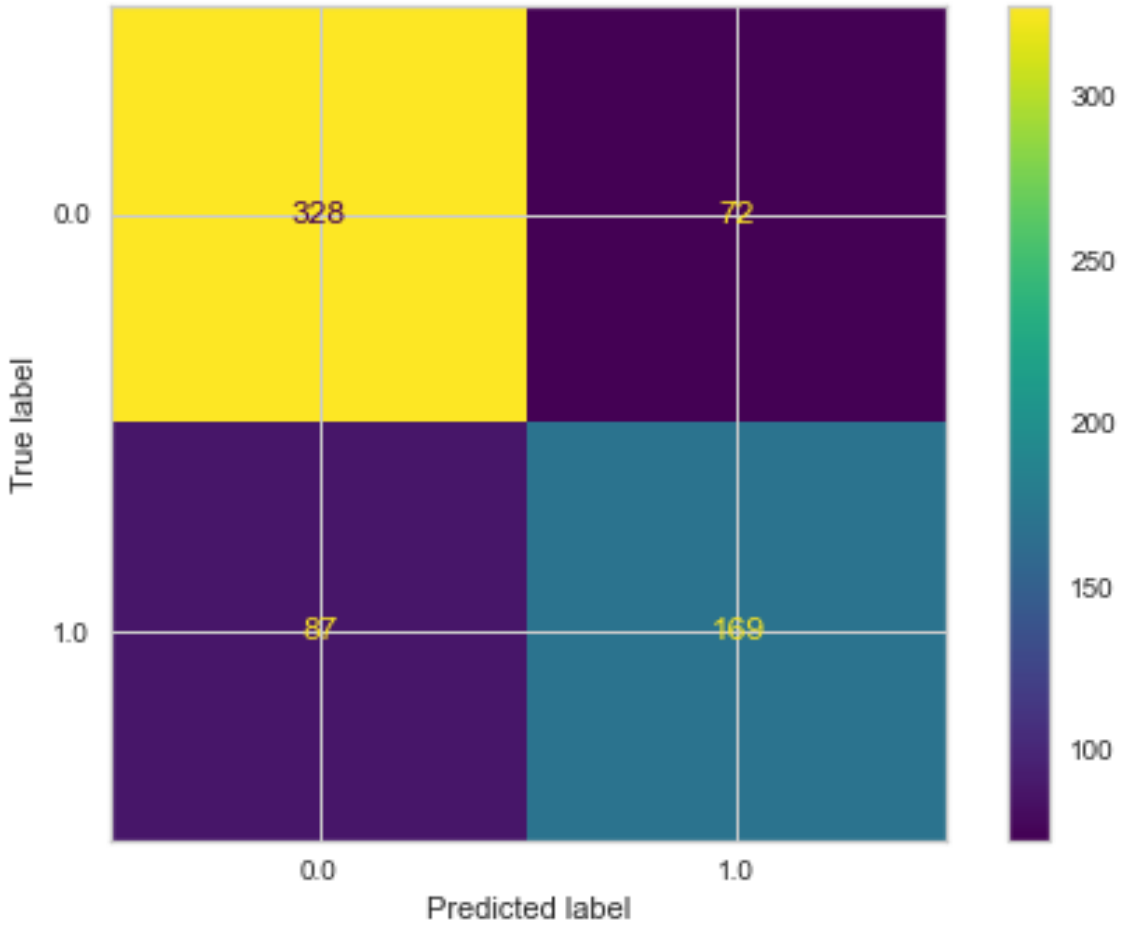
5 Tabulka experimentov

Obr. 11: Tabulka experimentov.

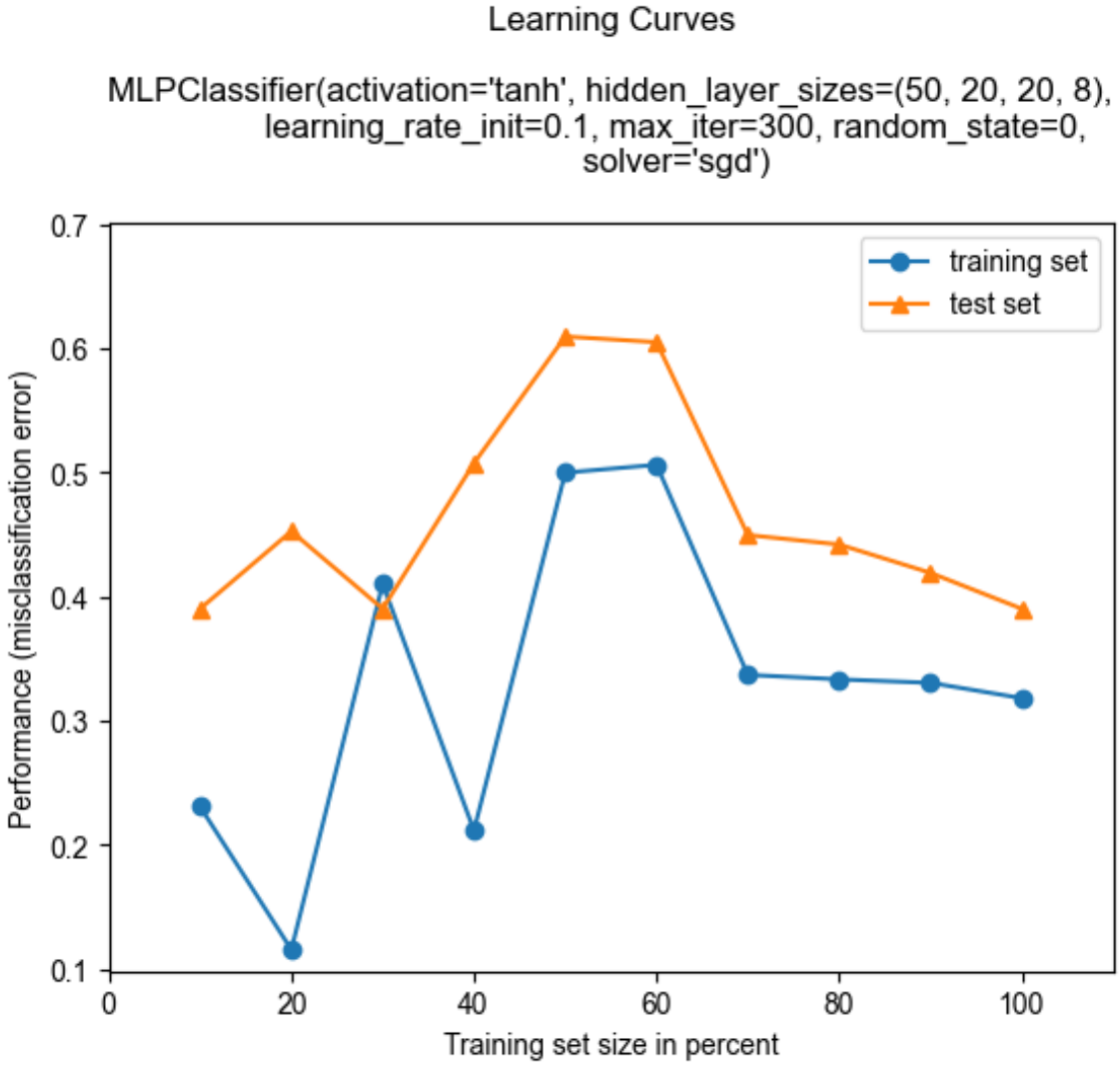
set	ativation	solver	learning rate	max iter	hidden layers	accuracy %
train	tanh	sgd	0.01	50	18,12,8	62.5
train	relu	sgd	0.1	100	50,30,8	66
train	relu	sgd	0.1	150	50,30,8	71.6
train	relu	sgd	0.1	200	50,30,8	73
train	relu	sgd	0.1	300	50,30,8	72.8
train	relu	sgd	0.1	300	50,20,20, 8	69.8
train	tanh	sgd	0.1	300	50,20,20, 8	75.7
train	tanh	adam	0.1	300	50,20,20, 8	60

Skúšali sme rôzne nastavenia parametrov a najlepší výsledok trénovania pomocou MLPClassifier sme získali 75.7 percent.

Obr. 12: Konfúzna matica pre náš najlepší výsledok.



Obr. 13: Priebeh trénovania pre náš najlepší výsledok.



Literatúra

https://scikit-learn.org/stable/modules/generated/sklearn.neural$_n etwork.MLPClassifier.html$