Zadanie 2

Načítanie dát

Dáta sa nachádzali v csv súbore. Načítal som ich pomocou knižnice pandas do takzvaného dataframu.

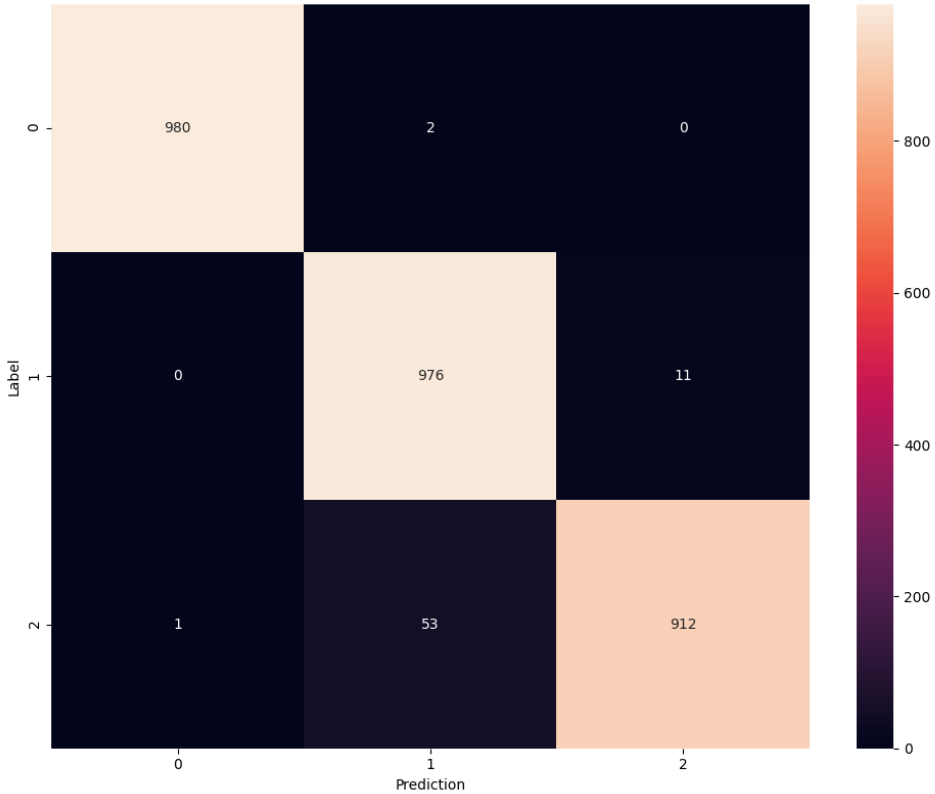
Predspracovanie dát

- Nahradenie názvov žánru, číslami, a to nasledovne:
  - START - 0
  - GALAXY – 1
  - QSO – 2
- Odstránenie nepotrebných stĺpcov z dát
  - Konkrétne som odstránil nasledujúce stĺpce
    - *Objid, specobjid, run, mjd, fiberid*

Súborový klasifikátor

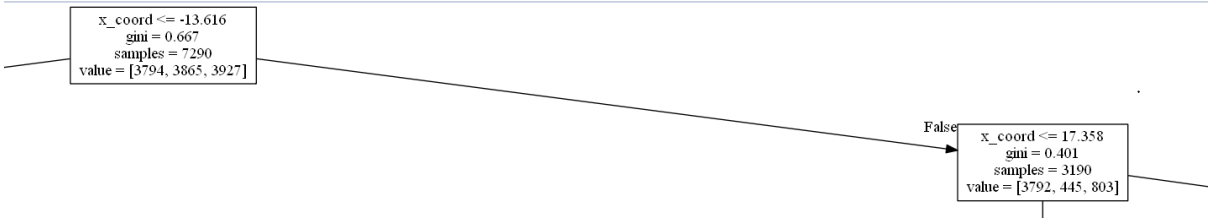
Vybral som takzvaný Bagging typ súborového klasifikátora. Kedy sa pre každý strom v súbore klasifikátorov vyberá náhodná množina dát.

Použil som RandomForestClassifier od sklearn. Maximálnu hĺbku stromu som mal nastavenú na 10, a počet stromov 10. Počet stromov však nehral príliš dôležitú úlohu. Pri takmer akomkoľvek nastavaní dosahoval klasifikátor výborné výsledky. To môžeme vidieť aj v nasledujúcom confusion matrix-e.



Analýza jedného slabého klasifikátora

Z nášho lesa stromov som vybral jeden náhodný strom a môžeme sa naň pozrieť. Koreňom stromu je súradnica x. Tam vidíme že nám to výrazne rozdelí našu množinu dát. Takmer všetky objekty ktoré patria do prvej kategórie majú súradnicu x väčšiu ako -13.616.



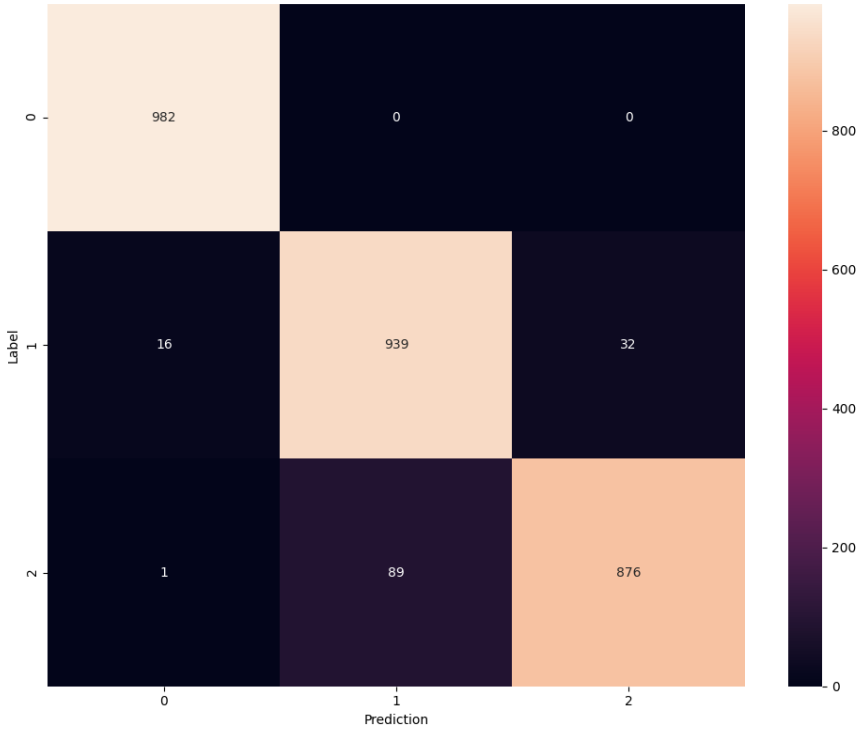
Maximálnu hĺbku stromu som nastavil na 10, čiže aj tento strom má hĺbku 10. Celý strom je možné pozrieť si aj v priloženom súbore: tree.png

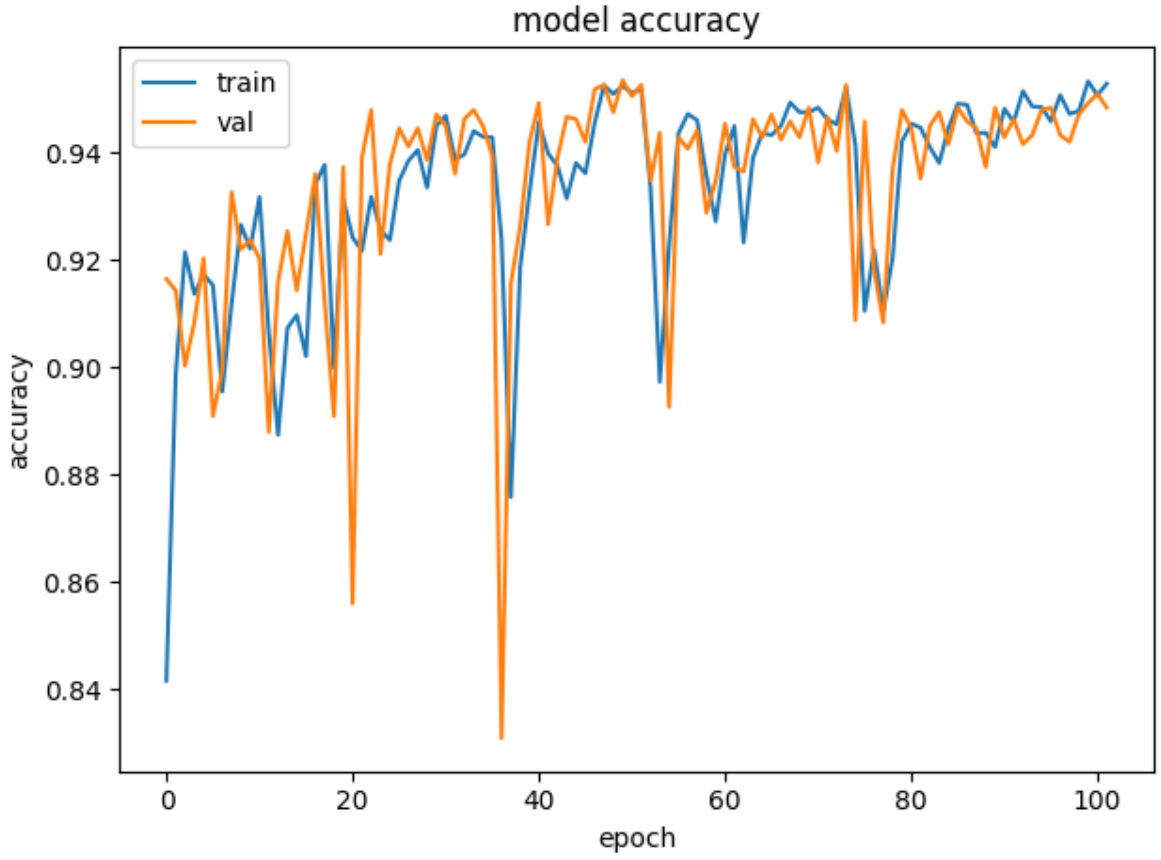
Neurónová sieť na klasifikovanie objektov

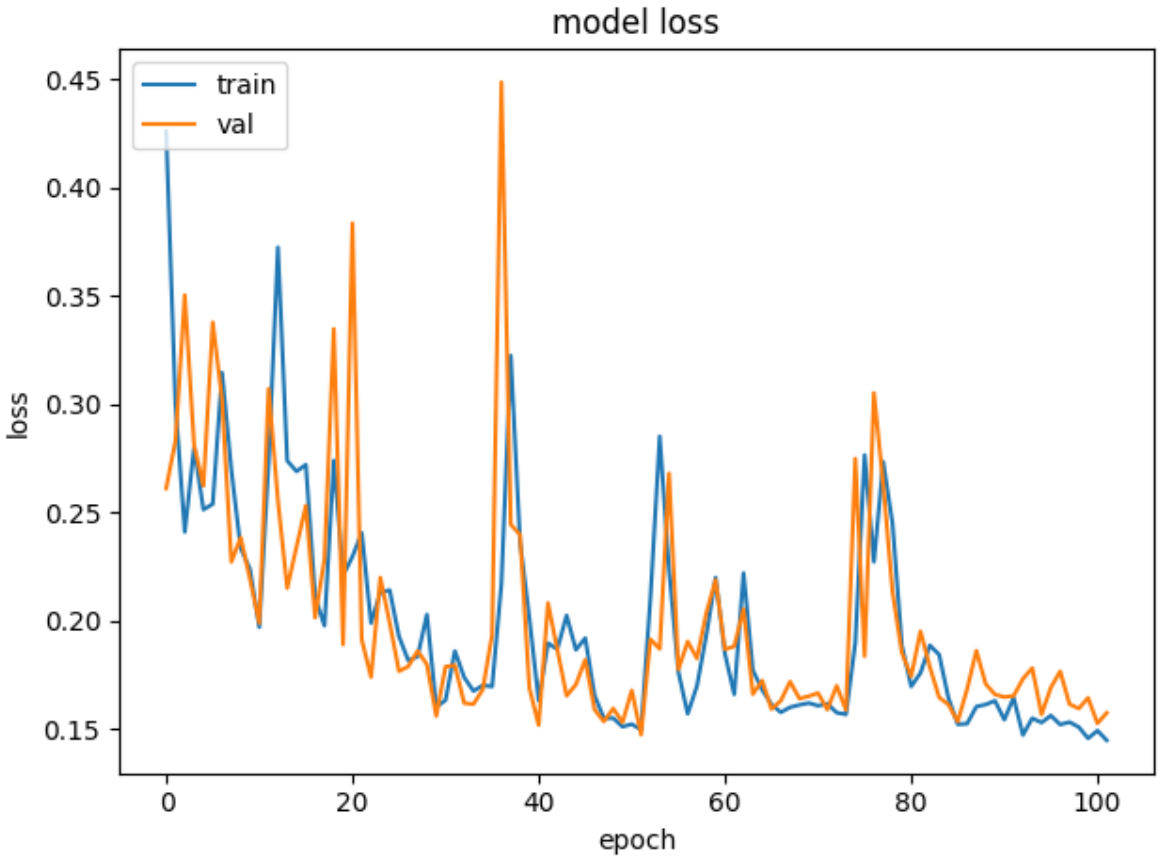
Ako ďalší klasifikátor som si vybral neurónovú sieť. Použil som tie isté dáta ako pri súborovom klasifikátore. Neurónová sieť mala nasledujúce hyperparametre:

```
model = kr.Sequential()
model.add(kr.layers.Dense(100, input_dim=11, activation="sigmoid"))
model.add(kr.layers.Dense(20, activation="sigmoid"))
model.add(kr.layers.Dense(3, activation="sigmoid"))

model.summary()
optimizer = kr.optimizers.Adam(0.01)

model.compile(loss="categorical_crossentropy", optimizer=optimizer, metrics=['accuracy'])
early_stopping = kr.callbacks.EarlyStopping(monitor='val_loss', patience=50)

training = model.fit(train_x, train_y, epochs=1000, validation_data=(val_x, val_y), callbacks=[early_stopping])
```

Aj nerónová sieť dosahovala výborné výsledky, na validačných dátach okolo 95%. Jej výsledky si môžeme pozrieť na nasledujúcich obrázkoch:


982
0
0
16
939
32
1
89
876
Label
Prediction
0
1
2
800
600
400
200
0


model accuracy
train
val
accuracy
epoch
0.94
0.92
0.90
0.88
0.86
0.84
0
20
40
60
80
100



Môžeme pozorovať že obidva modely mali veľmi podobnú úspešnosť. Taktiež boli veľmi podobné aj v chybovosti. Avšak, trénovanie súborového klasifikátora prebehlo neporovnateľne rýchlejšie oproti trénovaniu neurónovej siete. Pri ich takmer identickej úspešnosti by som teda zhodnotil že viac sa v tomto prípade oplatí použiť *RandomForestClassifier.*

Súborový regresor

Ďalšou úlohou bolo natrénovanie súborového regresora pre určenie karteziánskych súradníc jednotlivých objektov. Na trénovanie takého modelu som použil *MultiOutputRegressor* a ako estimator som použil *RandomForestRegressor. MultiOutputRegressor* použije daný estimator pre každú odhadovanú hodnotu. Takýto model dosahoval najlepšie výsledky pri nasledujúcich parametroch:

```
regr_multirf = MultiOutputRegressor(RandomForestRegressor(n_estimators=30,
                                                          max_depth=30))
regr_multirf.fit(train_x, train_y)
```

Pri testovaní skúšaní rôznych parametrov sa R2 pohybovalo v rozmedzí od 0.55 až po 0.75. R2 s hodnotou okolo 0.75 som dosahoval práve pri použití 30 stromov s maximálnou hĺbkou 30.

MLP Regresor

Rovnaký princíp som použil aj v neurónovej sieti. Ako estimator som do *MultioutputRegressora* dal *MLPRegressor* s nasledujúcimi parametrami:

```
regr = MLPRegressor(random_state=1,hidden_layer_sizes=[100,],learning_rate_init=0.01, max_iter=500)
```

Takýto regresor dosahoval výsledky R2 cca 0.22 . Ak ho porovnáme so súborovým regresorom môťme jasne vidieť že súborový regresor je úspešnejší.