Pawlak_Dominik_HW3

April 13, 2021

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
import seaborn as sns
warnings.filterwarnings('ignore')
np.random.seed = 44
```

[2]:
```
weather_df = pd.read_csv('australia.csv')
weather_df.head()
```

[2]:
```
   MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine  WindGustSpeed  \
0     17.9     35.2       0.0         12.0      12.3           48.0
1     18.4     28.9       0.0         14.8      13.0           37.0
2     19.4     37.6       0.0         10.8      10.6           46.0
3     21.9     38.4       0.0         11.4      12.2           31.0
4     24.2     41.0       0.0         11.2       8.4           35.0

   WindSpeed9am  WindSpeed3pm  Humidity9am  Humidity3pm  Pressure9am  \
0           6.0          20.0         20.0         13.0       1006.3
1          19.0          19.0         30.0          8.0       1012.9
2          30.0          15.0         42.0         22.0       1012.3
3           6.0           6.0         37.0         22.0       1012.7
4          17.0          13.0         19.0         15.0       1010.7

   Pressure3pm  Cloud9am  Cloud3pm  Temp9am  Temp3pm  RainToday  RainTomorrow
0       1004.4       2.0       5.0     26.6     33.4          0             0
1       1012.1       1.0       1.0     20.3     27.0          0             0
2       1009.2       1.0       6.0     28.7     34.9          0             0
3       1009.1       1.0       5.0     29.1     35.6          0             0
4       1007.4       1.0       6.0     33.6     37.6          0             0
```

[3]:
```
X = weather_df.drop(['RainTomorrow'], axis=1)
y = np.array(weather_df['RainTomorrow'])
```

[4]:
```
weather_df.describe()
```

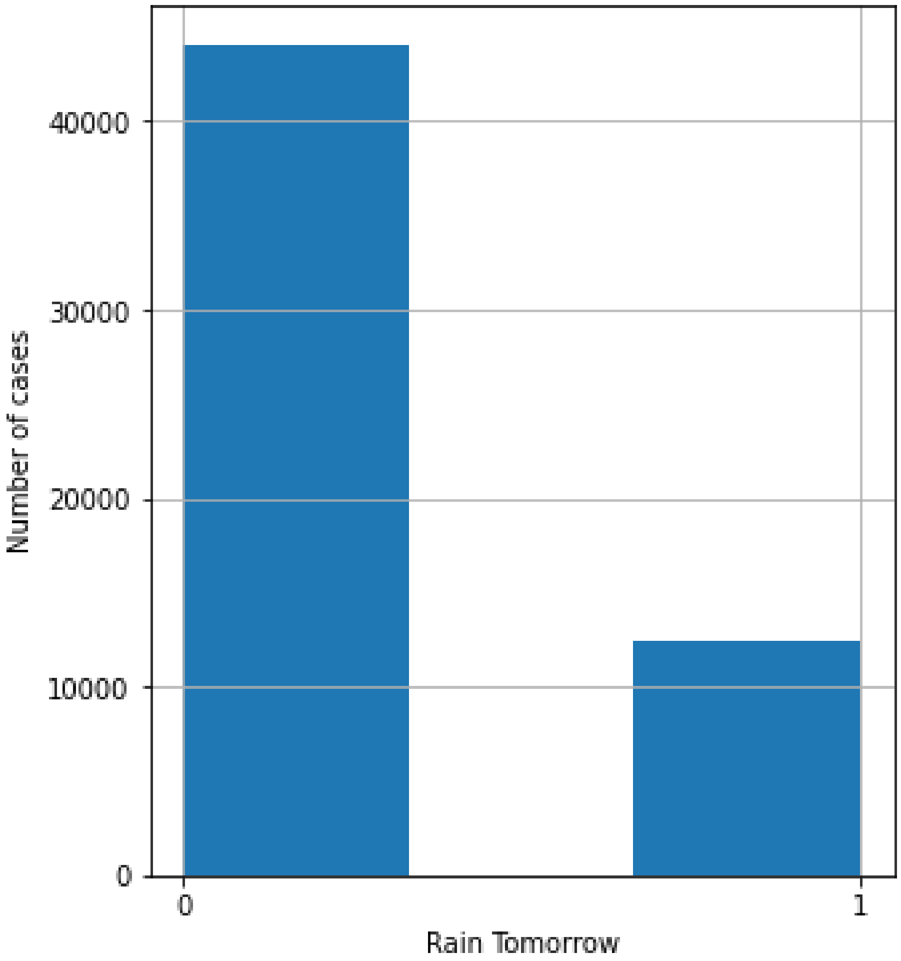
[4]:
```
            MinTemp       MaxTemp      Rainfall   Evaporation      Sunshine  \
count  56420.000000  56420.000000  56420.000000  56420.000000  56420.000000
mean      13.464770     24.219206      2.130397      5.503135      7.735626
std        6.416689      6.970676      7.014822      3.696282      3.758153
min       -6.700000      4.100000      0.000000      0.000000      0.000000
25%        8.600000     18.700000      0.000000      2.800000      5.000000
50%       13.200000     23.900000      0.000000      5.000000      8.600000
75%       18.400000     29.700000      0.600000      7.400000     10.700000
max       31.400000     48.100000    206.200000     81.200000     14.500000

       WindGustSpeed  WindSpeed9am  WindSpeed3pm   Humidity9am   Humidity3pm  \
count   56420.000000  56420.000000  56420.000000  56420.000000  56420.000000
mean       40.877366     15.667228     19.786778     65.874123     49.601985
std        13.335232      8.317005      8.510180     18.513289     20.197040
min         9.000000      2.000000      2.000000      0.000000      0.000000
25%        31.000000      9.000000     13.000000     55.000000     35.000000
50%        39.000000     15.000000     19.000000     67.000000     50.000000
75%        48.000000     20.000000     26.000000     79.000000     63.000000
max       124.000000     67.000000     76.000000    100.000000    100.000000

        Pressure9am   Pressure3pm      Cloud9am      Cloud3pm       Temp9am  \
count  56420.000000  56420.000000  56420.000000  56420.000000  56420.000000
mean    1017.239505   1014.795580      4.241705      4.326515     18.204961
std        6.909357      6.870892      2.797162      2.647251      6.567991
min      980.500000    977.100000      0.000000      0.000000     -0.700000
25%     1012.700000   1010.100000      1.000000      2.000000     13.100000
50%     1017.200000   1014.700000      5.000000      5.000000     17.800000
75%     1021.800000   1019.400000      7.000000      7.000000     23.300000
max     1040.400000   1038.900000      8.000000      9.000000     39.400000

            Temp3pm     RainToday  RainTomorrow
count  56420.000000  56420.000000  56420.000000
mean      22.710333      0.220879      0.220259
std        6.836543      0.414843      0.414425
min        3.700000      0.000000      0.000000
25%       17.400000      0.000000      0.000000
50%       22.400000      0.000000      0.000000
75%       27.900000      0.000000      0.000000
max       46.100000      1.000000      1.000000
```

[5]:
```
weather_df['RainTomorrow'].hist(bins = 3, figsize = (5, 6))
plt.ylabel('Number of cases')
plt.xlabel('Rain Tomorrow')
plt.xticks([0, 1])
plt.show()
```



Zatem klasa “brak deszczu” występuje dużo więcej razy.

```
[6]: from sklearn.model_selection import train_test_split
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
      ↪stratify=y)
```

```
[7]: print(X_train.shape, X_test.shape)
```

```
(45136, 17) (11284, 17)
```

1 Logistic Regression

```
[8]: from sklearn.metrics import accuracy_score
     from sklearn.model_selection import cross_val_score
     from sklearn.metrics import roc_auc_score
     from sklearn.metrics import roc_curve
     from sklearn.linear_model import LogisticRegression
     from sklearn import tree
     from sklearn.svm import SVC

     modelLR = LogisticRegression(penalty = 'none', random_state=35, max_iter=100000)
     modelLR.fit(X_train, y_train)
     y_hat = modelLR.predict(X_test)

     pred_prob1 = modelLR.predict_proba(X_test)
     fpr1, tpr1, _ = roc_curve(y_test, pred_prob1[:,1], pos_label=1)


     print(f'Accuracy: {round(accuracy_score(y_test, y_hat), 4)}')
     print(f'Cross-Validation mean score: {round(np.mean(cross_val_score(modelLR,␣
      ↪X_test, y_test)), 4)}')
     print(f'ROC-AUC Score: {round(roc_auc_score(y_test, pred_prob1[:,1]), 4)}')
```

```
Accuracy: 0.8535
Cross-Validation mean score: 0.8526
ROC-AUC Score: 0.8816
```

2 Decision tree

[9]:

```
modelDT = tree.DecisionTreeClassifier(random_state = 35, max_depth = 10)
modelDT.fit(X_train, y_train)
y_hat = modelDT.predict(X_test)

pred_prob2 = modelDT.predict_proba(X_test)
fpr2, tpr2, _ = roc_curve(y_test, pred_prob2[:, 1], pos_label=1)

print(f'Accuracy: {round(accuracy_score(y_test, y_hat), 4)}')
print(f'Cross-Validation mean score: {round(np.mean(cross_val_score(modelDT,␣
 ↪X_test, y_test)), 4)}')
print(f'ROC-AUC Score: {round(roc_auc_score(y_test, pred_prob2[:, 1]), 4)}')
```

```
Accuracy: 0.8379
Cross-Validation mean score: 0.8292
ROC-AUC Score: 0.8449
```

3 SVC

[10]:

```
modelSVC = SVC(gamma='scale', degree = 5, kernel = 'poly', max_iter = 100000,␣
 ↪random_state = 35)
modelSVC.fit(X_train, y_train)
y_hat = modelSVC.predict(X_test)

fpr3, tpr3, _ = roc_curve(y_test, y_hat, pos_label=1)
```

```
print(f'Accuracy: {round(accuracy_score(y_test, y_hat), 4)}')
print(f'Cross-Validation mean score: {round(np.mean(cross_val_score(modelDT,␣
 ↪X_test, y_test)), 4)}')
print(f'ROC-AUC Score: {round(roc_auc_score(y_test, y_hat), 4)}')
```

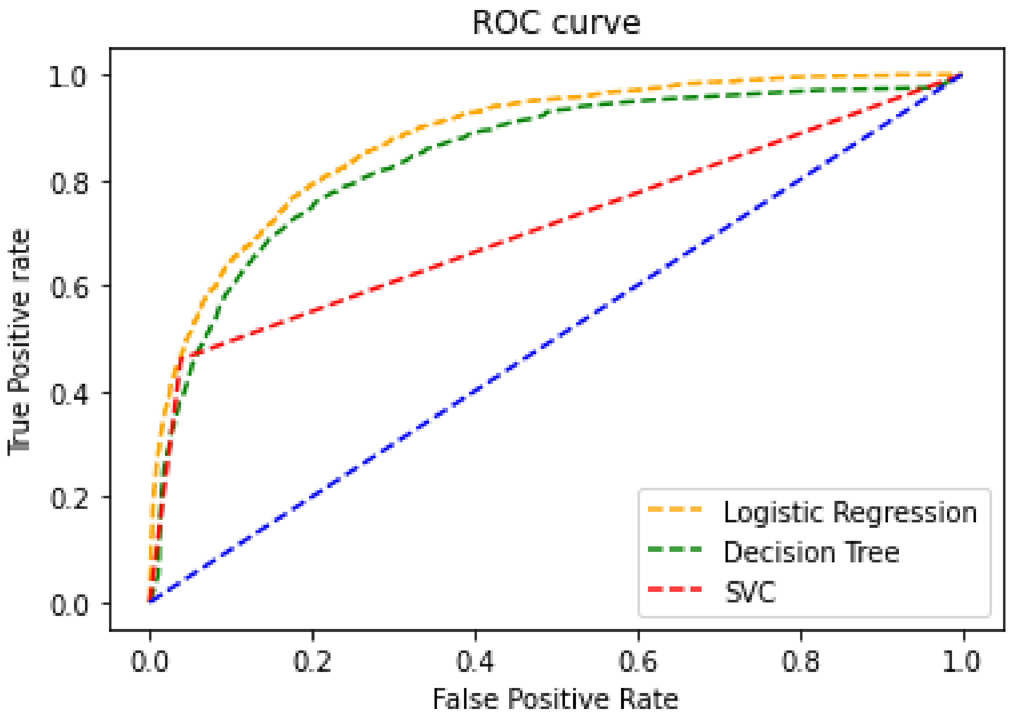
```
Accuracy: 0.8513
Cross-Validation mean score: 0.8292
ROC-AUC Score: 0.711
```

[11]:

```
random_probs = [0 for i in range(len(y_test))]
p_fpr, p_tpr, _ = roc_curve(y_test, random_probs, pos_label=1)

plt.plot(fpr1, tpr1, linestyle='--',color='orange', label='Logistic Regression')
plt.plot(fpr2, tpr2, linestyle='--',color='green', label='Decision Tree')
plt.plot(fpr3, tpr3, linestyle='--',color='red', label='SVC')
plt.plot(p_fpr, p_tpr, linestyle='--', color='blue')

plt.title('ROC curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive rate')
plt.legend(loc='best')
plt.show();
```



Po wykresie krzywej ROC oraz przedstawionych wynikach innych miar jakości klasyfikatora, widzimy, że z wybranych przeze mnie modeli najlepiej radzi sobie model regresji logistycznej.