COMP9417 HomeWork 2

Ping GAO z5163482

March 29, 2020

1 Q1

1.1 Part A

```
                          DecisionTreeClassifier
-----------------------------------------------------------------------------------------------------------
   Dataset     |   5%    |  10%    |  15%    |  20%    |  25%    |  30%    |  35%    |  40%    |  45%    |  50%    |
-----------------------------------------------------------------------------------------------------------
australian     | 72.61% | 74.35% | 75.36% | 77.39% | 77.83% | 79.71% | 83.77% | 81.16% | 80.72% | 83.48% |
balance-scale  | 69.92% | 75.04% | 69.12% | 74.24% | 74.40% | 75.52% | 78.08% | 75.68% | 77.92% | 76.64% |
hypothyroid    | 94.94% | 96.31% | 97.77% | 99.18% | 99.20% | 99.42% | 99.42% | 99.52% | 99.34% | 99.20% |
```

```
                          BernoulliNB with priors
-----------------------------------------------------------------------------------------------------------
   Dataset     |   5%    |  10%    |  15%    |  20%    |  25%    |  30%    |  35%    |  40%    |  45%    |  50%    |
-----------------------------------------------------------------------------------------------------------
australian     | 73.48% | 79.86% | 81.45% | 80.43% | 79.71% | 79.86% | 79.86% | 81.16% | 82.17% | 81.88% |
balance-scale  | 46.08% | 46.08% | 46.08% | 46.08% | 46.24% | 46.08% | 46.08% | 46.24% | 46.24% | 46.08% |
hypothyroid    | 91.38% | 91.81% | 92.23% | 92.23% | 92.23% | 92.26% | 92.23% | 92.23% | 92.23% | 92.23% |
```

1.2 Part B

I think (3), (5) statements are true.

1.3 Part C

I choose (1).

2 Q2

2.1 Part A

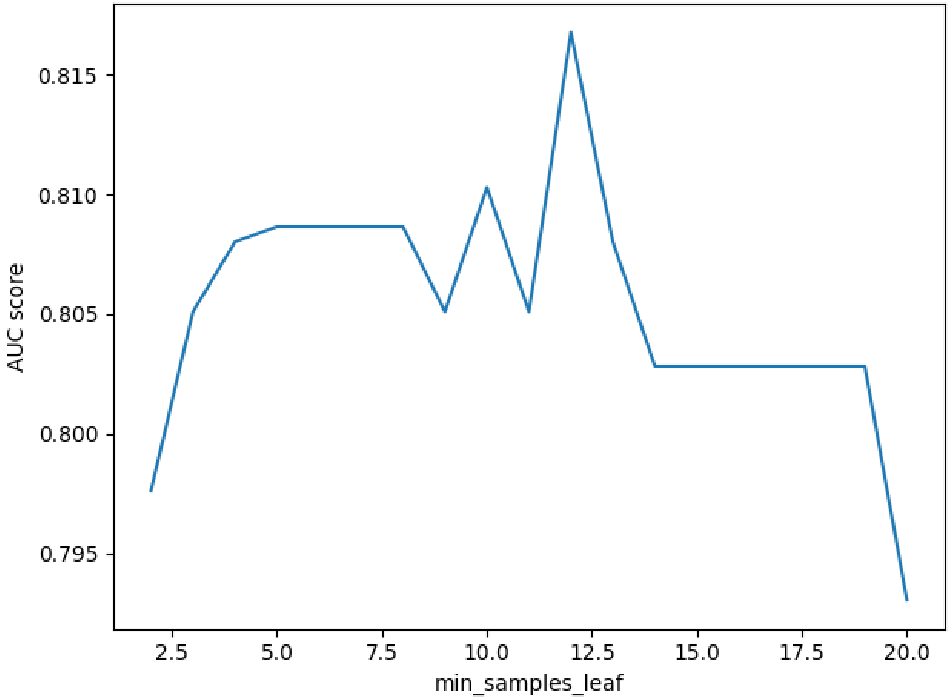
My accuracy score for the test dataset is 82.77%.
My accuracy score for the training dataset is 85.65%.

2.2 Part B

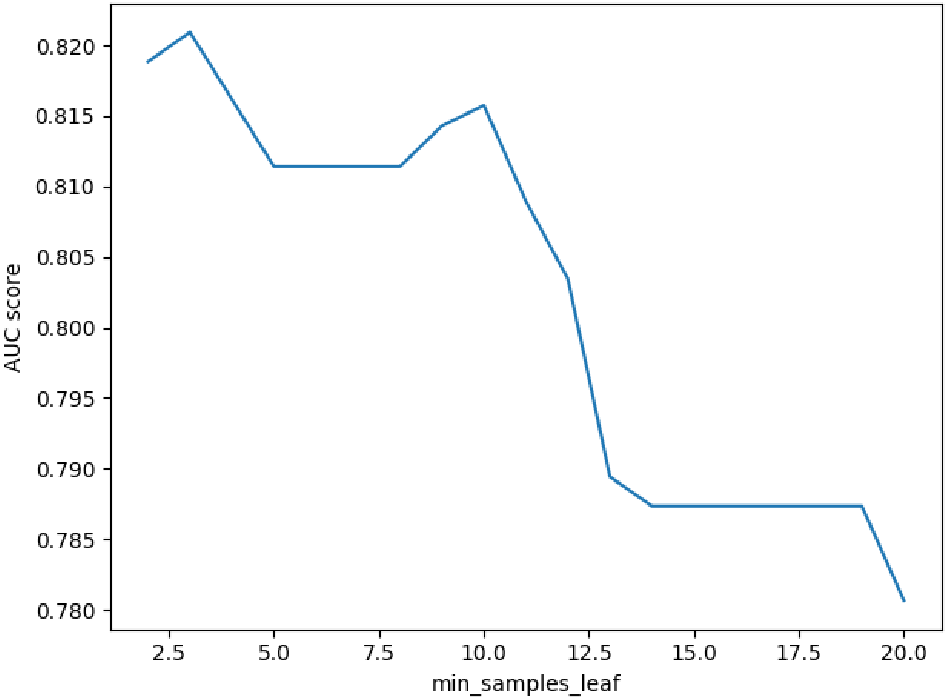
The min_samples_leaf number of 5 to 7 give the optimal result, this can be observed by compare the auc score in the part c's plots, pick the maximum score with the lowest varience.

2.3 Part C

Plot For Test Dataset



Plot For Train Dataset



2.4 Part D

We make the asumption that ’Sex’ and ’Pclass’ are independent feature. Thus, (S=true $|G = female, C = 1) = P(S = true|G = female) * P(S = true|C = 1)$

$P(S = true|G = female) = P(G = female) \cap P(S = true)/P(G = female)$

$= 109/573$

$P(S = true|C = 1) = 136/216$

$P(S = true|G = female, C = 1) = 11.98\%.$

2.5 My code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import accuracy_score, roc_curve, auc


# import numpy as np
# import seaborn as sns
# import graphviz
# import matplotlib.pyplot as plt


class TitanicSinkingModel:
def __init__(self, filename):
self.df = pd.read_csv(filename)
self.df_normalized = None
self.data_train = None
self.data_test = None
self.target_train = None
self.target_test = None
self.clf = None

def preprocessing(self):
# this can be done because the target value is either 0 or 1
# thus apply the normalization have no effect on them
scaler = MinMaxScaler()
self.df_normalized = pd.DataFrame(scaler.fit_transform(self.df.values),
columns=self.df.columns, index=self.df.index)
self.split_dataframe()

def split_dataframe(self):
data = self.df_normalized[
[col for col in self.df_normalized.columns if col != 'Survived']]
target = self.df_normalized['Survived']
# split the test and train data set
self.data_train, self.data_test, self.target_train, self.target_test =
    train_test_split(data, target,
test_size=0.3,
shuffle=False)

def fit_decision_tree(self):
self.clf = tree.DecisionTreeClassifier()
self.clf = self.clf.fit(self.data_train, self.target_train)

# dot_data = tree.export_graphviz(clf, out_file=None)
# graph = graphviz.Source(dot_data)
# graph.format = 'jpg'
# graph.render("decision_tree_plot_orignial")
labels_test = self.clf.predict(self.data_test)
acc = accuracy_score(labels_test, self.target_test)
print("acc for test set is : " + str(acc))
labels_test2 = self.clf.predict(self.data_train)
acc2 = accuracy_score(labels_test2, self.target_train)
print("acc for train set is : " + str(acc2))

def find_optimal_decision_tree(self):
# min_samples_leaf the minmum number of leaves a split
# can happen according to the value of entropy
auc_tain = {}
auc_test = {}
for i in range(2, 21):
descion_tree = tree.DecisionTreeClassifier(min_samples_leaf=i)
descion_tree.fit(self.data_train, self.target_train)
false_positive_rate, true_positive_rate, thresholds = roc_curve(self.
    target_train, descion_tree.predict(self.data_train))
```

```
auc_tain[i] = auc(false_positive_rate, true_positive_rate)
false_positive_rate, true_positive_rate, thresholds = roc_curve(self.
    target_test, descion_tree.predict(self.data_test))
auc_test[i] = auc(false_positive_rate, true_positive_rate)
# fig = plt.figure()
# d = {’min_samples_leaf’: np.array(list(auc_tain)), ’AUC score’: np.array(
    list(auc_tain.values()))}
# pd_plot = pd.DataFrame(d)
# sns.lineplot(x=’min_samples_leaf’, y=’AUC score’, data=pd_plot)
# plt.show()
# fig.savefig(’plot_train.png’)
self.clf = tree.DecisionTreeClassifier(min_samples_leaf=6)
self.clf.fit(self.data_train, self.target_train)
# dot_data = tree.export_graphviz(self.clf, out_file=None)
# graph = graphviz.Source(dot_data)
# graph.format = ’jpg’
# graph.render("decision_tree_plot_optimal")

def part_D_calculation(self):
print("female and survived: ")
print(self.df.query(’Sex == 1 & Survived ==1’))
print("all female: ")
print(self.df.query(’Sex == 1’))
print("first class and survived: ")
print(self.df.query(’Pclass == 1 & Survived ==1’))
print("all first class: ")
print(self.df.query(’Pclass == 1’))


def print_df_normalized(self):
print(self.df_normalized.head().to_string())


if __name__ == ’__main__’:
titianic_sinking_model = TitanicSinkingModel("titanic.csv")
titianic_sinking_model.preprocessing()
# titianic_sinking_model.print_df_normalized()
# print(titianic_sinking_model.data_train.to_string())
titianic_sinking_model.fit_decision_tree()
titianic_sinking_model.find_optimal_decision_tree()
titianic_sinking_model.part_D_calculation()
```