9417 Homework2

Q1(a).

Decision Tree Results

----------------------------------------------------------------------------------------------------------------

Dataset | Default | 0% | 25% | 50% | 75% |

----------------------------------------------------------------------------------------------------------------

australian | 56.52% ( 2) | 81.16% ( 7) | 86.96% ( 2) | 56.52% ( 2) | 20.77% ( 7) |

labor | 61.11% ( 2) | 94.44% ( 7) | 44.44% ( 7) | 61.11% (12) | 44.44% (12) |

diabetes | 66.23% ( 2) | 67.10% ( 7) | 64.07% (12) | 66.23% ( 2) | 35.50% (27) |

ionosphere | 66.04% ( 2) | 86.79% ( 7) | 82.08% (27) | 71.70% ( 7) | 18.87% (12) |

Q1(b). By increasing the value of the value of “max\_depth” parameter we can expect this to:

**Answer:**

**Choose (4), --- (4) increase overfitting by increasing max\_depth of the decision tree.**

**The deeper the tree, the more splits it has, and it captures more information about the data, so it will increase overfitting.**

Q1(c). Looking at your table, the performance result for datasets with 50% noise and “max\_depth” parameter. Does finding the best parameter in grid search helps the decision tree model to improve the test set accuracy compared to the default parameter settings? What is your answer?

**Answer:**

**Choose (2), --- (2) yes, for 1/4 of the datasets.**

**Only the accuracy of ionosphere increased, so chose 1/4.**

Q2.

Part A. Implement a kNN classifier for Australian credit risk prediction using sklearn library. You should set the n\_neighbors =2 for training the model. What is your accuracy score for training and test dataset?

**Answer:**

**Accuracy score for training dataset: 0.89694041868**

**Accuracy score for test dataset: 0.76811594203**

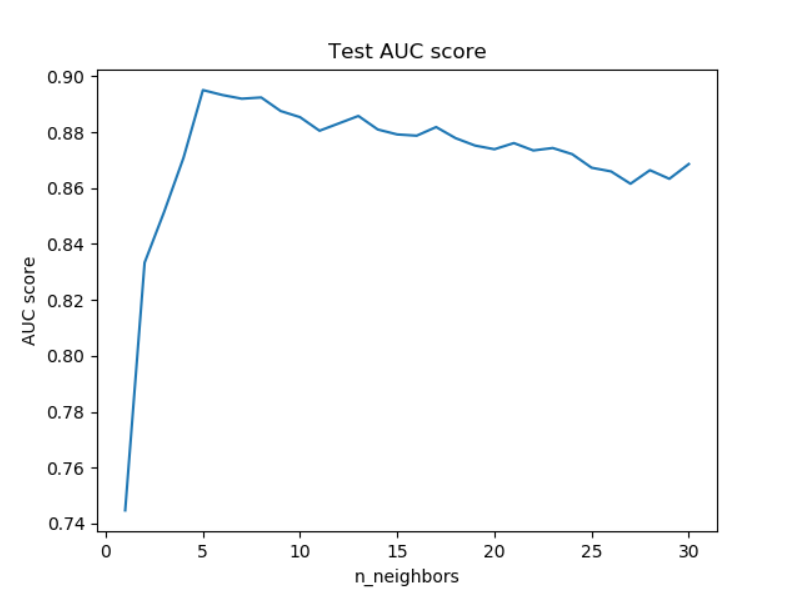
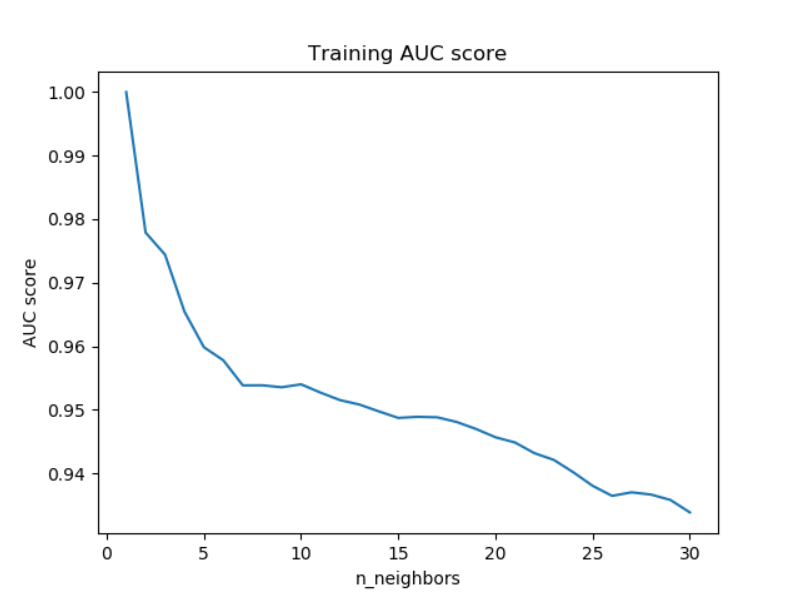
Part B. Find optimal number of neighbours by developing a search algorithm to find the optimal value of k. You should find the optimal number of k in a range between 1 to 30 and finding optimal value for number of k. please use AUC score to find the optimal number of neighbours.

**Answer:**

**K = 5**

Part C. Plot the AUC score for all iterations (k: 1,…,30) in training and test sets. (one plot for training, and one for test set).

**Answer:**



**Plot for training sets plot for test sets**

Part D. Compute precision and recall evaluation metrics for your kNN model with optimal number of neighbours and another model that you have built in part A. Compare these metrics for these two models.

**Answer:**

**when k = 2, the precision score is: 0.789473684211**

**when k = 2, the recall score is: 0.555555555556**

**when k = 5, the precision score is: 0.766666666667**

**when k = 5, the recall score is: 0.851851851852**

**Because it is about credit card, so it is better to find more people who have credit issues, so the larger recall is better and k = 5,the recall score is similar to k = 2. As a result, k = 5 is a better moedel. And also, we can compute the F1-score,according to recall and precision, F1-socre = 2\*precision\*recall/(precision+recal).**

**When k = 5, F1-score is 0.80701754386, when k = 2, F1-score is 0.652173913043, so k = 5 is also better model.**

Code:

import pandas as pd  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.metrics import roc\_auc\_score  
import matplotlib.pyplot as plt  
import numpy as np  
from sklearn.metrics import precision\_score  
from sklearn.metrics import recall\_score  
  
  
# from sklearn.model\_selection import GridSearchCV  
  
  
def minmax(x, min\_x, max\_x):  
 return (x - min\_x) / (max\_x - min\_x)  
  
  
data = pd.read\_csv('CreditCards.csv')  
X\_data = data.drop(['Y'], axis=1)  
  
X\_min = np.min(X\_data, axis=0)  
X\_max = np.max(X\_data, axis=0)  
  
X\_data = minmax(X\_data, X\_min, X\_max)  
  
X\_train = X\_data[:621]  
X\_test = X\_data[621:]  
  
y\_train = data.Y[:621]  
y\_test = data.Y[621:]  
knn\_2 = KNeighborsClassifier(n\_neighbors=2)  
knn\_2.fit(X\_train, y\_train)  
y\_predict = knn\_2.predict(X\_test)  
  
print("when k = 2, the accuracy score for test data is", knn\_2.score(X\_test, y\_test))  
print("when k = 2, the accuracy score for training data is", knn\_2.score(X\_train, y\_train))  
print("when k = 2, the precision score is: ", precision\_score(y\_test, y\_predict))  
print("when k = 2, the recall score is: ", recall\_score(y\_test, y\_predict))  
  
a = precision\_score(y\_test, y\_predict)  
b = recall\_score(y\_test, y\_predict)  
  
print("when k = 2, the f1-score is: ", 2 \* a \* b / (a + b))  
  
training\_auc\_score = []  
test\_auc\_score = []  
  
n\_neighbors = range(1, 31)  
best\_neighbor\_number, best\_score = None, None  
  
# GridSearchCV  
"""  
param\_grid = {'n\_neighbors': n\_neighbors}  
knn = KNeighborsClassifier()  
knn\_cv = GridSearchCV(knn, param\_grid, scoring='roc\_auc', cv=10)  
knn\_cv.fit(X\_data, data.Y)  
print(knn\_cv.best\_score\_)  
print(knn\_cv.best\_params\_)  
"""  
  
for neighbors in n\_neighbors:  
 knn = KNeighborsClassifier(neighbors)  
 knn.fit(X\_train, y\_train)  
 y\_train\_predict\_proba = knn.predict\_proba(X\_train)[:, 1]  
 y\_predict\_proba = knn.predict\_proba(X\_test)[:, 1]  
 training\_auc\_score.append(roc\_auc\_score(y\_train, y\_train\_predict\_proba))  
 test\_auc\_score.append(roc\_auc\_score(y\_test, y\_predict\_proba))  
 current\_score = roc\_auc\_score(y\_test, y\_predict\_proba)  
 if not best\_score or best\_score < current\_score:  
 best\_score = current\_score  
 best\_neighbor\_number = neighbors  
  
print("optimal value for number of k is", best\_neighbor\_number)  
  
plt.plot(n\_neighbors, training\_auc\_score)  
plt.title('Training AUC score')  
plt.ylabel('AUC score')  
plt.xlabel('n\_neighbors')  
plt.show()  
  
plt.plot(n\_neighbors, test\_auc\_score)  
plt.title('Test AUC score')  
plt.ylabel('AUC score')  
plt.xlabel('n\_neighbors')  
plt.show()  
  
knn\_5 = KNeighborsClassifier(n\_neighbors=5)  
knn\_5.fit(X\_train, y\_train)  
y\_predict = knn\_5.predict(X\_test)  
  
print("when k = 5, the accuracy score for test data is:", knn\_5.score(X\_test, y\_test))  
print("when k = 5, the accuracy score for training data is:", knn\_5.score(X\_train, y\_train))  
print("when k = 5, the precision score is: ", precision\_score(y\_test, y\_predict))  
print("when k = 5, the recall score is: ", recall\_score(y\_test, y\_predict))  
  
a = precision\_score(y\_test, y\_predict)  
b = recall\_score(y\_test, y\_predict)  
print("when k = 5, the f1-score is: ", 2 \* a \* b / (a + b))