Student ID: z5195715
Name: Junyu Ren

COMP 9417 Homework 2

Question 1

Part A
To answer this question, you should run the python code in the notebook 'comp9417_hw2.ipynb'. copy and paste the table in your report as your answer for "Question Part A".

```
D:\PyCharmProjects\venv\Scripts\python.exe D:/PyCharmProjects/9417Homework2_Part1.py
                                        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           |  66.67% ( 2)    |  94.44% ( 7)    |  44.44% ( 7)    |  66.67% ( 7)    |  50.00% (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)    |
```

Part B (I marked my answer in yellow background, same for Part C)
By increasing the value of "max_depth" parameter we can expect this to:
--- (1) over-fitting not changed by decreasing max_depth of the decision tree
--- (2) decrease over-fitting by increasing max_depth of the decision tree
--- (3) increase over-fitting by decreasing max_depth of the decision tree
--- (4) increase over-fitting by increasing max_depth of the decision tree

Explanation: Increasing max_depth parameter makes the decision tree have more branches and fit the current data-set very well but it may not performs well on other test sets.This is over-fitting.

Part C
Looking at your table, the performance result for data-sets 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?
--- (1) no
--- (2) yes, for 1/4 of the data-sets
--- (3) yes, for 2/4 of the data-sets
--- (4) yes, for 3/4 of the data-sets
--- (5) yes, for 4/4 of the data-sets

Explanation: Compare default column to 50% noise column, only for ionosphere, the test-set accuracy increases(from 66.04% to 71.07%). The others remain unchanged.So, 1/4 of the

data-sets improve test-set accuracy.

Question 2

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 data-set?

Training accuracy score is roughly 0.897

Test accuracy score is roughly 0.768

Part B

Find optimal number of neighbors 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 neighbors.

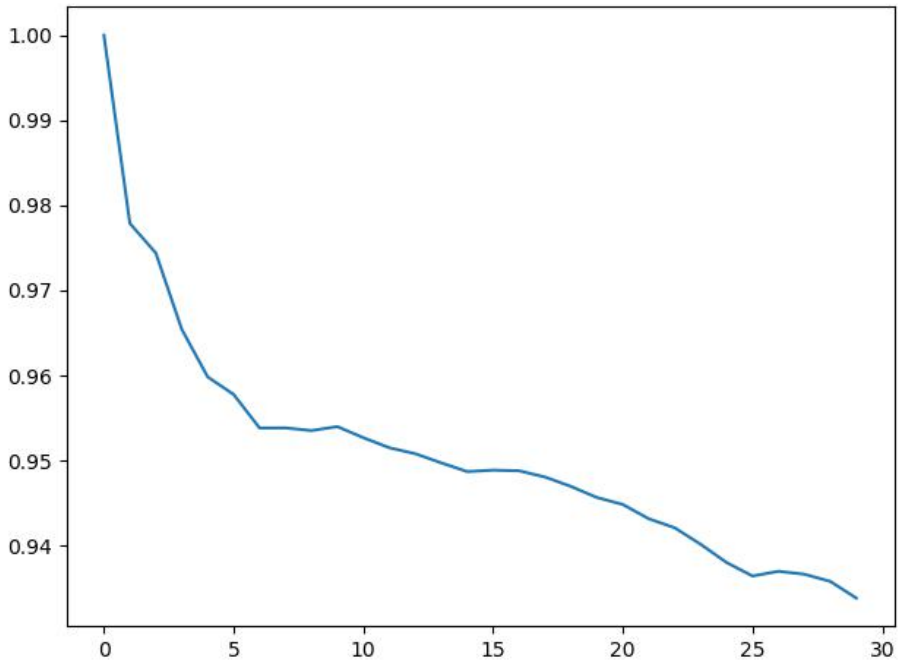
Optimal k is 5

AUC score for 5NN is approximately 0.895

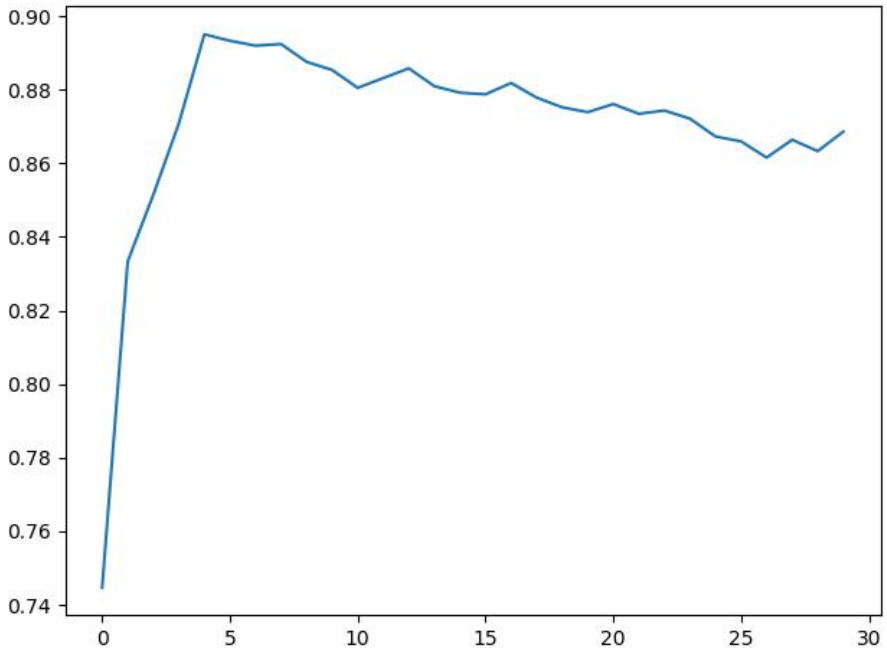
Part C

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

Plot for training set:



Plot for test set:



Part D

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

2NN precision is roughly 0.7895

2NN recall is roughly 0.5556

5NN precision is roughly 0.7667

5NN recall is roughly 0.8519

Compare:

Precision = TP / (TP + FP).It means the ratio of actually positive samples among all samples that are predicted as positive.Compare the 2 models, their precisions don't have big differences.

Recall = TP / (TP + FN).It means the ratio of samples that are correctly predicted among all actually positive samples.Compare the 2 models, 5NN recall is much greater than that of 2NN.

Code:

```
import sys
import matplotlib.pyplot as plt
import numpy as np
import csv
from sklearn import preprocessing
from sklearn import neighbors
from sklearn import metrics
from sklearn.metrics import accuracy_score


def read_csv(filename):
	data = []
	with open(filename) as file:
		csv_file = csv.reader(file)
		for row in csv_file:
			data.append(row)
	return np.array(data[1:]).astype(float)


raw_data = read_csv('CreditCards.csv')
# split x and y
raw_x = raw_data[:, :14]
raw_y = raw_data[:, 14]
min_max_scaler = preprocessing.MinMaxScaler()
processed_x = min_max_scaler.fit_transform(raw_x)
# split train and test
train_x = processed_x[:621, :]
test_x = processed_x[621:, :]
train_y = raw_y[:621]
test_y = raw_y[621:]

# Part A
clf1 = neighbors.KNeighborsClassifier(2)
clf1.fit(train_x, train_y)
prediction_train_2NN = clf1.predict(train_x)
acc_train = accuracy_score(train_y, prediction_train_2NN)
print("Train accuracy: "+str(acc_train))

prediction_test_2NN = clf1.predict(test_x)
acc_test = accuracy_score(test_y, prediction_test_2NN)
print("Test accuracy: "+str(acc_test))

# Part B
```

```
train_auc = []
test_auc = []
for k in range(1, 31):
    clf2 = neighbors.KNeighborsClassifier(k)
    clf2.fit(train_x, train_y)
    prediction_train = clf2.predict_proba(train_x)
    prediction_test = clf2.predict_proba(test_x)
    train_auc.append(metrics.roc_auc_score(train_y, prediction_train[:, 1]))
    test_auc.append(metrics.roc_auc_score(test_y, prediction_test[:, 1]))
print("Best NN: "+str(test_auc.index(max(test_auc))+1))
print("AUC value for best NN: "+str(max(test_auc)))

# Part C
plt.plot(train_auc)
plt.show()
plt.plot(test_auc)
plt.show()

# Part D
recall_2NN = metrics.recall_score(test_y, prediction_test_2NN)
print("Recall 2NN: "+str(recall_2NN))
precision_2NN = metrics.precision_score(test_y, prediction_test_2NN)
print("Precision 2NN: "+str(precision_2NN))

clf3 = neighbors.KNeighborsClassifier(5)
clf3.fit(train_x, train_y)
prediction_test_5NN = clf3.predict(test_x)
# prediction_train_5NN = clf3.predict(train_x)
recall_5NN = metrics.recall_score(test_y, prediction_test_5NN)
precision_5NN = metrics.precision_score(test_y, prediction_test_5NN)
print("Recall 5NN: "+str(recall_5NN))
print("Precision 5NN: "+str(precision_5NN))
```

Running screenshot of the above code:

```
D:\PyCharmProjects\venv\Scripts\python.exe D:/PyCharmProjects/9417Homework2_Part2.py
Train accuracy: 0.8969404186795491
Test accuracy: 0.7681159420289855
Best NN: 5
AUC value for best NN: 0.8950617283950617
Recall 2NN: 0.5555555555555556
Precision 2NN: 0.7894736842105263
Recall 5NN: 0.8518518518518519
Precision 5NN: 0.7666666666666667
```