REPORT FOR ASSIGNMENT 1

1. Introduction

1.1. library

The libraries used in assignment 1:
NumPy, processing data in list and array
Pandas, reading data from the giving 'csv' document
Matplotlib, plotting figures of the dataset (basic library to load scikit-learn and seaborn)
Scikit-learn, training, testing data, generating the result
Seaborn, plotting histogram of dataset and heatmap of confusion matrix

1.2. Classification Methods and Hyperparameters

In this assignment, I choose the following three classification
models: Decision Tree, SVM, KNN.
For Decision Tree, I select *max_depth* from 1 to 10 and *min_samples_split* from 2 to 10 as the hyperparameters (both integers).
For SVM, I select *c* and *gamma* both from the list [0, 0.001, 0.01, 0.1, 1, 10] as the hyperparameters.
For KNN, I select *n_neighbors* from 1 to 10 (integers) and *weights* with 2 types: distance and uniform, as the hyperparameters.

1.3. Training and Testing Process

Firstly, randomly split the data into a training dataset and a testing dataset using the **train_test_split()** function of scikit-learn. In this case, 80% of the dataset is used as the train set, and the spare 20% of the dataset is used as the test set.
Secondly, I defined a function to generate all the output needed, including accuracy, precision, recall and F-1 score. It also could plot the confusion matrix by using **seaborn.heatmap()** and print the classification report.
Thirdly, generate each classification model and use the train set to train the model. And then, choose k = 5 for K-Fold Cross Validation. Next, use **GridSearchCV()** to find the best parameters. At last, invoke the function defined in step 2 to print the result.

```
Best parameters: {'max_depth': 3, 'min_samples_split': 6}
Best score: 0.9392857142857144
Accuracy of decision tree: 0.9428571428571428
Precision of decision tree: 0.9423076923076923
Recall of decision tree: 0.9074074074074074
F1-Score of decision tree: 0.9245283018867925
```

Decision Tree

This is the result of decision tree, it indicates the best parameters are *max_depth* = 3 and *min_asmples_split* = 6.

```
Best parameters: {'C': 0.1, 'gamma': 0.01}
Best score: 0.9678571428571429
Accuracy of SVM: 0.9714285714285714
Precision of SVM: 0.9310344827586207
Recall of SVM: 1.0
F1-Score of SVM: 0.9642857142857143
```

SVM

This is the result of svm, it indicates the best parameters are $c = 0.1$ and *gamma* = 0.01.

```
Best parameters: {'n_neighbors': 5, 'weights': 'distance'}
Best score: 0.9714285714285713
Accuracy of KNN: 0.9714285714285714
Precision of KNN: 0.9464285714285714
Recall of KNN: 0.9814814814814815
F1-Score of KNN: 0.9636363636363636
```

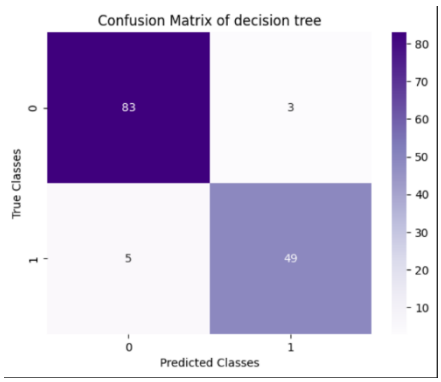
KNN

This is the result of knn, it indicates the best parameters are *n_neighbors* = 5 and *weights* is distance.

2. Evaluation

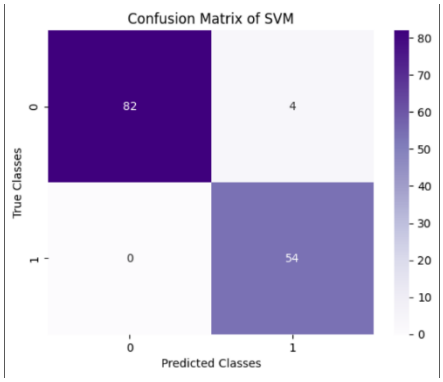
2.1. Confusion Matrix

Decision Tree:



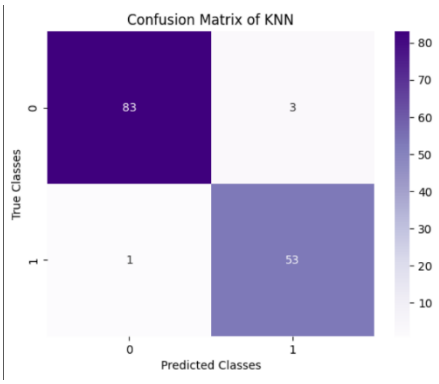
There are 83 samples whose class is class 0, and the prediction result of the model is also class 0.
There are 49 sample whose true class is class 1, and the prediction result of the model is also class 1.
There are 3 sample whose true class is class 0, but the prediction result of the model is class 1.
There are 5 sample whose true class is class 1, but the prediction result of the model is class 0.

SVM:



There are 82 samples whose class is class 0, and the prediction result of the model is also class 0.
There are 54 sample whose true class is class 1, and the prediction result of the model is also class 1.
There are 4 sample whose true class is class 0, but the prediction result of the model is class 1.
There are no sample whose true class is class 1, but the prediction result of the model is class 0.

KNN:



There are 83 samples whose class is class 0, and the prediction result of the model is also class 0.
There are 53 sample whose true class is class 1, and the prediction result of the model is also class 1.
There are 3 sample whose true class is class 0, but the prediction result of the model is class 1.
There are 1 sample whose true class is class 1, but the prediction result of the model is class 0.

2.2. Comparison

	Accuracy	Precision	Recall	F-1 Score
Decision Tree	**0.9428571428571428**	**0.9423076923076923**	**0.9074074074074074**	**0.9245283018867925**
SVM	**0.9714285714285714**	**0.9310344827586207**	**1.0**	**0.9642857142857143**
KNN	**0.9714285714285714**	**0.9464285714285714**	**0.9814814814814815**	**0.9636363636363636**

The table above comparing the Precision, Recall, F1-Score, and accuracy of all methods. We could clearly find that the SVM model achieves the highest F-1 score, which combines the output of precision and recall. It means SVM should be the best model for this case. The precision of SVM model is lower than KNN model, which means KNN model could classify class 0 more accurately. The recall of SVM model is higher than KNN model, which means SVM model could classify class 1 more accurately. The scores of various indicators of decision tree are not as good as svm and knn, which shows that it is not as suitable for this example as the first two models.

3. Conclusion

The data in this case has a total of 9 features, which could be considered as a high-dimensional dataset. As we know, SVM model works well in high-dimensional spaces. It may be the reason that this model achieves the highest F-1 score. On the other hand, Decision Trees can be prone to overfitting, especially when the tree is deep and complex. And it also can be sensitive to small variations in the data. I think that's why Decision Tree preforms the worst in this example.

In this assignment, I understand the use of many methods in seaborn and scikit-learn. But I still cannot fully understand the internal structure and skillfully use and advanced parameters of the methods in these libraries (such as *sklearn.pipeline*).