Name: Lingxiao Zhang

Student ID: 20475043

**Lab Report**

**Introduction**

The goal of this project is to understand and implement the basic operations of training models from given data and test the performance of the model using the test data.

**Lab Procedure**

This project can be separated into four steps, which are read and scale the input data, apply training models, compare performance, and make predictions and write the result to a csv file.

**Read and Scale the input data**

I used the Pandas to access and read the given input and test files, which are train data, train label and the test data. Then I scale the input data using the scale() function from Sklearn Preprocessing and check the shape of them by the way, the code is shown below:

# Data path for all files needed  
data\_path = '/Users/lingxiaozhang/Documents/6000b/project1/'  
input\_data = 'traindata.csv'  
input\_label = 'trainlabel.csv'  
test\_data = 'testdata.csv'  
  
# Read the input data  
train\_X = pd.read\_csv(data\_path + input\_data)  
train\_Y = pd.read\_csv(data\_path + input\_label)  
test\_X = pd.read\_csv(data\_path + test\_data)  
  
# Check the format  
# 3219, 57  
print train\_X.shape  
  
# 3219, 1  
print train\_Y.shape  
  
# avoid warning  
train\_Y = np.ravel(train\_Y)  
  
# 1379, 57  
print test\_X.shape  
train\_X = scale(train\_X)

**Train the model**

Although the specification requires us to apply a single model, I applied four different models to the input data, which are Logistic Regression, SVM, MLP and Adaboost. For most of them, I used the default settings on the parameters, except the MLP classifier since I got warnings that achieve the convergence later when fitting the data. And I set the max iterations to 1000 to ensure the convergence and it will take several seconds than before. The code is shown below:

# Apply the classifier  
#classifier1 = svm.SVC()  
#classifier2 = MLPClassifier()  
#classifier3 = AdaBoostClassifier()  
classifier = MLPClassifier(max\_iter=1000)

**Compare performance**

Since the label of the test data is not given, I used the cross validation to test and compare the performance among four classifiers mentioned above, and decide the classifier that I will use for the test data. It turns out that the MLP has the highest performance, which has an average accuracy of around 94.5% on the 5-fold cross validation. So I used the MLP classifier to predict the result of the test data. The code is shown below:

# Apply cross validation on the training set to compare performance  
score = cross\_val\_score(classifier, train\_X, train\_Y, cv=5)  
#score1 = cross\_val\_score(classifier1, train\_X, train\_Y, cv=5)  
#score2 = cross\_val\_score(classifier2, train\_X, train\_Y, cv=5)  
#score3 = cross\_val\_score(classifier3, train\_X, train\_Y, cv=5)

**Make predictions**

Finally, I fit the data into the MLP classifier, then I made predictions on the test data and write the result to a csv file. The code is shown below:

# Predictions on the test data and write to a csv file  
classifier.fit(train\_X, train\_Y)  
predict = classifier.predict(test\_X)  
print predict  
np.savetxt(data\_path + "project1\_20475043.csv", predict, fmt="%.1f", delimiter=",")

**Conclusion**

To sum up, I gained some experience on training a model and then test on a test set in this project. On the other hand, I gained some experience on using the Sklearn package.