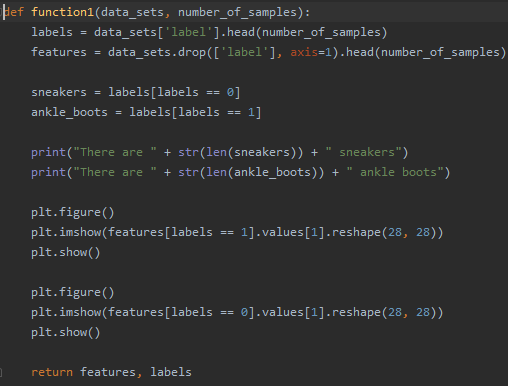
Assignment2R00164774

# Task 1



function1 takes in two parameters, (data\_sets, number\_of\_sampes)

* data\_sets 🡪 represents the actual pandas dataframe
* number\_of\_samples 🡪 represents the number of samples that will be used for this function

labels 🡪 this represents the prediction class which determines if the image is an ankle boot or a sneaker

features 🡪 this represents the list of numbers which will be converted to an image

* 0 🡪 sneaker
* 1 🡪 ankle boot

Below represents the number of sneakers and ankle boots in the sample dataset. In this case, we are only using the top 1000 rows in the data as represented by the following values:





It will then output an image of the ankle boot

To display the list of integers as an image, the feature needs to be reshaped in a 28x28 matrix.



Image of the ankle boot

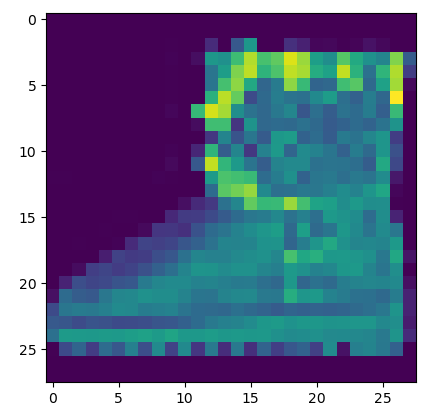
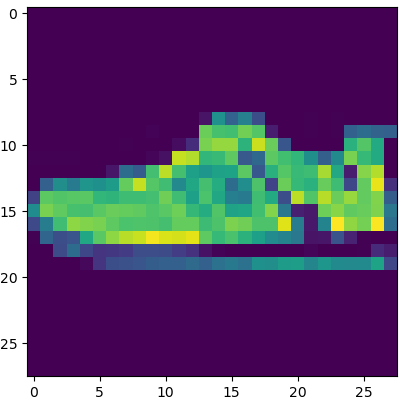


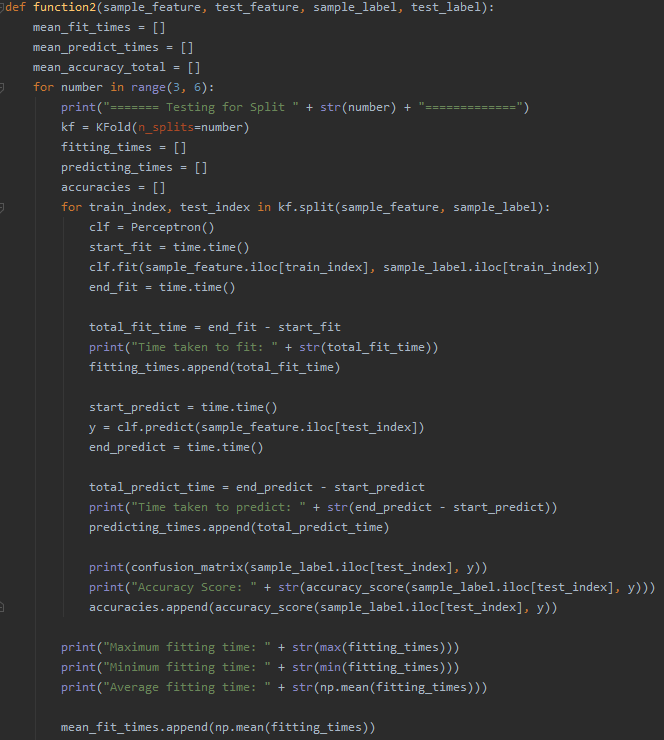
Image of the sneaker

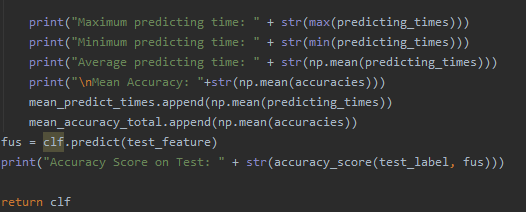




I then split my data into the training and testing data with the train\_test\_split

# Question 2





This function uses a perceptron classifier to train and predict the data in a K-fold cross validation. The parameters include

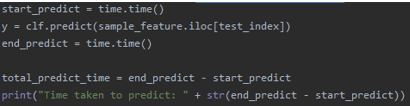
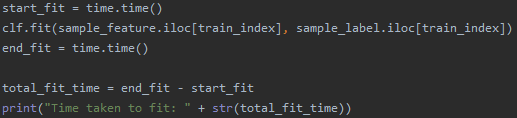
* sample\_feature 🡪 Represents the train features extracted in train\_test\_split of function1
* test\_feature 🡪 represents the test features extracted in train\_test\_split of function1
* sample\_label 🡪 represents the train labels extracted in train\_test\_split of function1
* test\_label 🡪 represents the test labels extracted in train\_test\_split of function1

three arrays are created:

* mean\_fit\_times 🡪 it holds all of the average fitting times calculated in each KFold split
* mean\_predict\_times 🡪 it holds all of the average predicting times calculated in each KFold split
* mean\_accuracy\_total 🡪 it holds all the accuracy scores predicted on each split

The function uses several values for KFold(n\_splits=) (3-5)

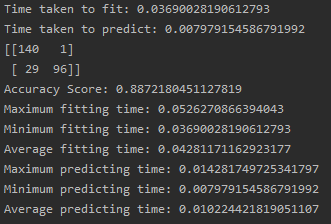
* splits the data into the train and test data
* instantiates a Perceptron() classifier
* fits the training data into the classifier
* predicts the test labels
* outputs a confusion matrix and the accuracy\_score(y\_true, y\_pred)
* it also calculates the amount of time it takes to fit the data or predict the labels as described below



It will also print out the accuracy score on the Test features after predicting the labels with the test features

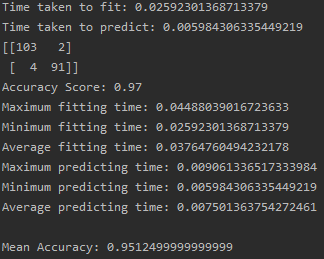
The function returns the classifier that has been trained

Sample output for KFold(n\_splits=3)

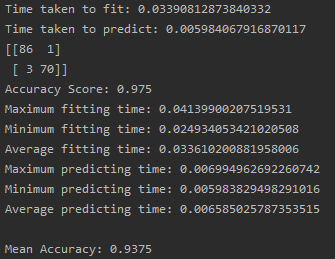




Sample output for KFold(n\_splits=4)



Sample output for KFold(n\_splits=5)



Increasing the amount of n\_splits will slightly decrease the average fitting time and the predicting time. However, the maximum fitting time will be a bit larger and the minimum time will be slightly smaller if the n\_splits increase. There is a slight variation between the minimum predicting times as the number of K\_FOLD increases however, the maximum predicting time will decrease the higher the n\_splits value.

KFold(n\_splits=4) has outputted the highest accuracy

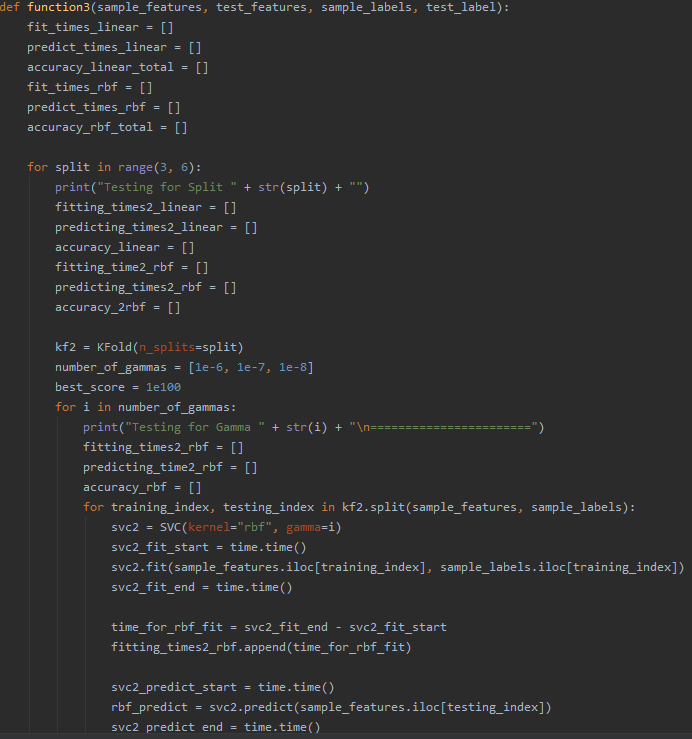
It will also print out the accuracy score on the Test features after predicting the labels with the test features

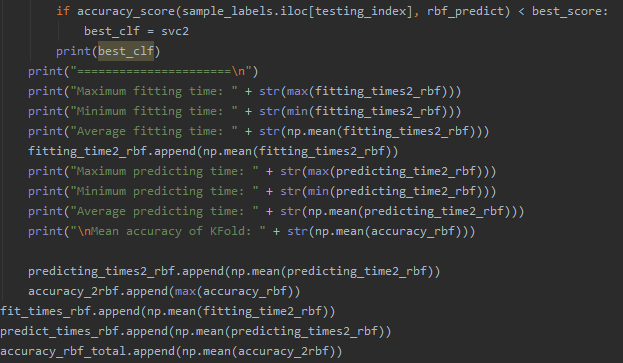


# Question 3

This function will use SVM with two types of kernels, RBF and Linear.

The RBF SVM has an extra parameter known as gamma and defines the distance the influence of one training example can reach. The lower the value, the further the influence can go and vice versa



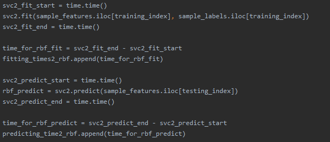


First, it will test the RBF SVM classifier for three values for the gamma parameter of the SVC with the kernel=”rbf”,

It tested for gamma 🡪 1e-6, 1e-7, 1e-8.

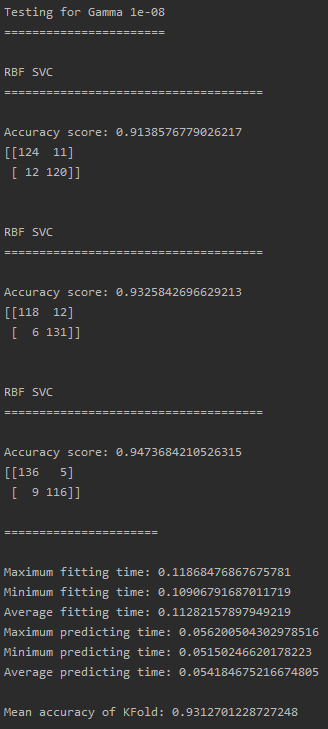
The function then splits the data into KFolds according to the value on the n\_splits parameter and will go through all the tests for n\_splits (3-5)

Once again, the training and predicting times are calculated by the following lines

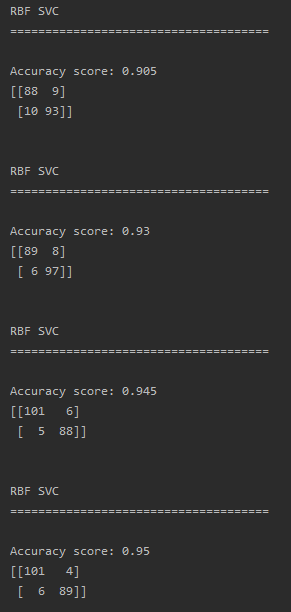


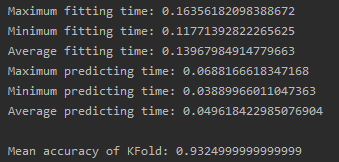
It will then output the accuracy score and confusion matrix for each of the splits and gamma values.

Sample output for KFold(n\_splits=3) and SVC(kernel=”rbf”, gamma=1e-08)

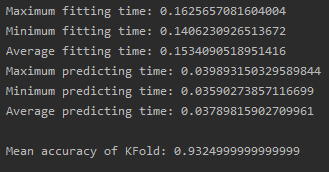


Sample output for KFold(n\_splits=4) and SVC(kernel=”rbf”, gamma=1e-08)





Sample results for KFold(n\_splits=5) and SVC(kernel=”rbf”, gamma=1e-08)

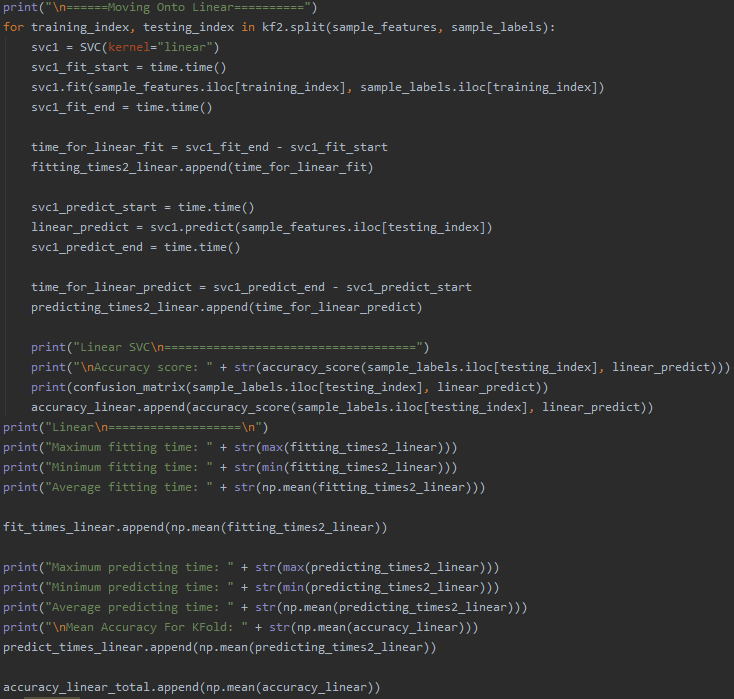


Increasing the amount of KFold splits will either decrease or increase the maximum fitting time. The mean accuracy will vary depending on the KFold split, so it will either increase or decrease if the number of n\_splits are increased.

It will then pick out the highest accuracy out of all of the gamma values tested.



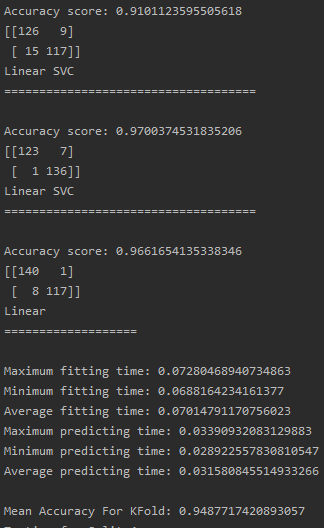
It then does the same for the Linear SVC which is simpler than the RBF SVC in that it doesn’t use a gamma parameter



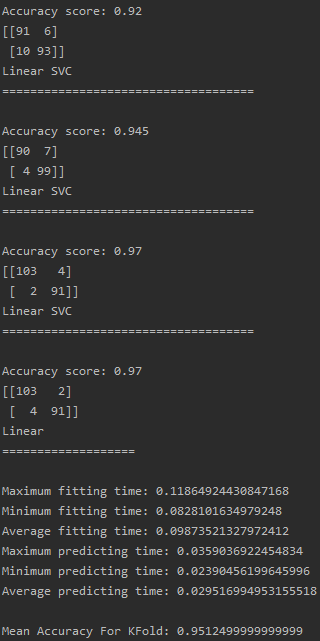
Once again it follows the same pattern of

* instantiating the classifier
* fitting the data
* predicting the labels
* calculating the time it took to fit the data and to predict the labels

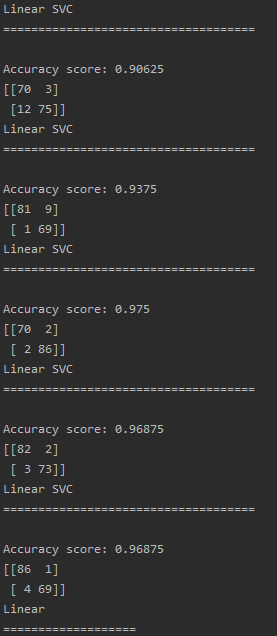
Sample output for KFold n\_splits=3

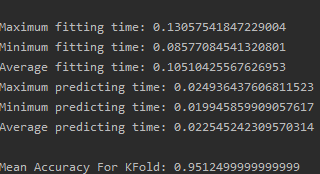


Sample output for the linear SVC model for KFold(n\_splits=4)



Sample output for the linear SVC model for KFold(n\_splits=5)



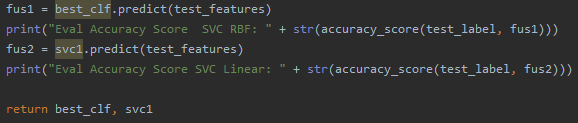


For each KFold split, the accuracy score and the confusion matrix is outputted

After the KFold cross validation process is finished, the programm will print the maximum, minimum and average times for fitting the model and predicting the labels

It will also print out the mean accuracy for the total KFold process.

Once again, it will return the two classifiers used for this function, in this case the RBF SVC and the Linear SVC.



The final accuracy predicted on the test features for both the RBF SVC and the Linear SVC has been printed out

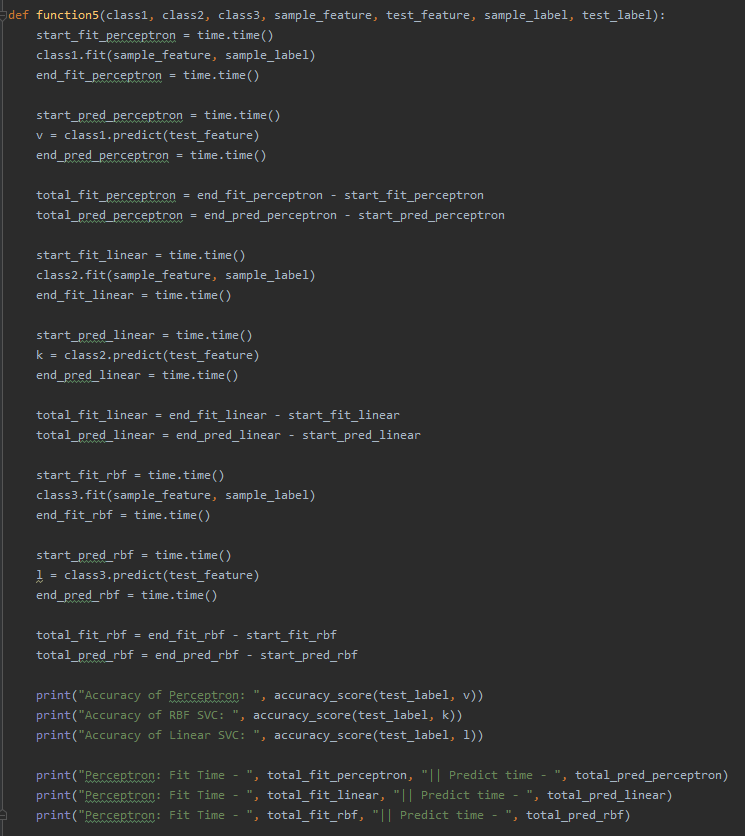


The ideal amount of KFold split is n\_splits=4 because it produces the highest accuracy. Overall, increasing the amount of splits will increase the overall fitting time and decreases the overall predicting time

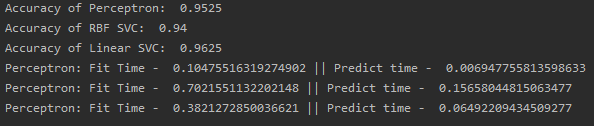
# Question 4

The function takes in the three classifiers returned from function2 and function3 and will once again repeat the procedure of fitting the training data and the training labels and predicting the actual test labels based on the testing features.

It will also calculate the amount of time it takes to fit and predict respectively



It will then display the fit time, prediction time, accuracy of the Perceptron, RBF SVC and Linear SVC respectively



According to the results, Perceptron is the fastest classifier when it comes to fitting and predicting the data.

However, the RBF SVC with the best gamma parameter will output the highest accuracy.

In conclusion, I will choose the RBF SVC for accuracy but I will choose Perceptron for speed. RBF and linear SVCs fitting speed tends to be severely affected by the size of the training dataset.