

**American University of Sharjah**

**College of Engineering**

**Department of Computer Science & Engineering**

**Spring 2021**

**CMP 466 – Machine Learning & Data Mining**

**Assignment 3**

**Farooq Mirza 80205**

**Hamad AlMaqoodi 76307**

**Omar Fayed 69702**

**Table of Contents**

[**Question 1**](#_jdphag8j5iqz)

[**Question 2**](#_66pay6cny5t9)

[**Question 3**](#_d0wuwkyrrd2q)

[**Question 4**](#_8qnemewpfe00)

[**Question 5**](#_9mdtp13u9tmv)

# **Question 1:**

We read the data from the .csv file as a pandas data frame. Then we got rid of all the unnecessary features and the labels of the dataset while storing it into X, more precisely we got rid of column 32 since it was an Unnamed feature full of Null values. We also convert the dataset type of X to float to ease NumPy’s underlying C calculations. Then we converted the categorical labels to integer ones, which were set as 0 for Malignant which was labeled as ‘M’ and 1 for Benign which was labeled as ‘B’.

**#-----------------------**

**# Reading the data**

**#-----------------------**

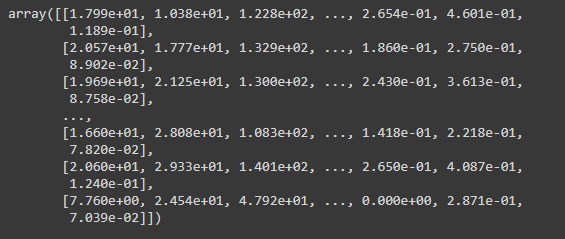
**import pandas as pd**

**import numpy as np**

**data = pd.read\_csv('Breast Cancer Wisconsin Data Set')**

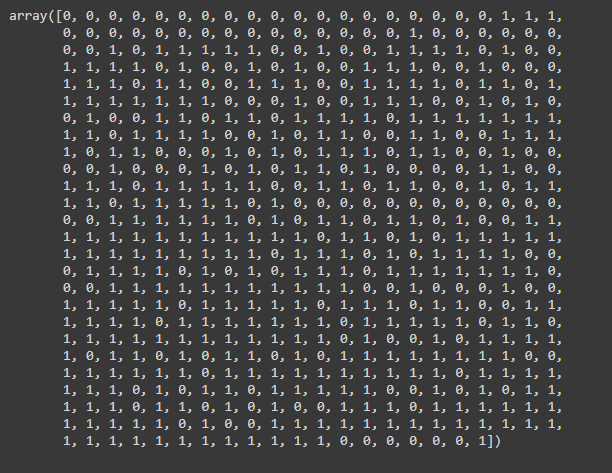
**X = data.drop(columns=['id', 'Unnamed: 32','diagnosis']).astype(float).to\_numpy()**

**print(X)**

****

**y = data.replace(['M','B'],[0, 1])['diagnosis'].astype(int).to\_numpy()**

**print(y)**

****

## **Question 2:**

In this part, we defined a function “print\_cross\_validate”, to print the cross validation results for n-folds which can be passed as parameters to the function. We then print the average training score and average testing score for the cross validation results.

**#-------------------------**

**# K-fold Cross Validation**

**#-------------------------**

**from sklearn.model\_selection import cross\_validate**

**def print\_cross\_validate(classifier, data, labels, folds):**

**cv\_results = cross\_validate(classifier, data, labels, cv=folds, return\_train\_score=True)**

**print("\tCross Validation Results (", folds, "- fold ):\n", cv\_results,**

**"\nAverage Train Score", np.average(cv\_results['train\_score']),**

**"\nAverage Test Score", np.average(cv\_results['test\_score']))**

### **Question 3:**

In this part, we applied the Linear Support Vector Machine (LinearSVC) by import svm from sklearn. We found the training and testing accuracy for 5-fold and 10-fold. Also, the average is calculated for the training and the testing accuracy.

**#-------------------------**

**# Applying Linear SVC**

**#-------------------------**

**from sklearn import svm**

**"""**

**when not specifying dual=False, the following warning is shown.**

**ConvergenceWarning: Liblinear failed to converge, increase the number of iterations.**

**"""**

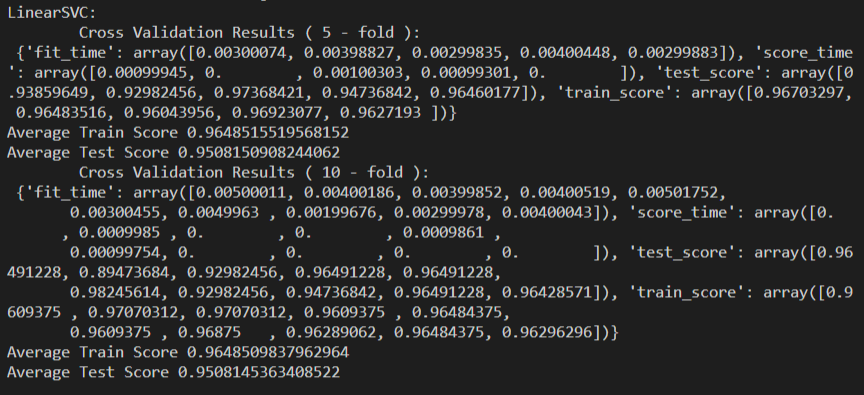
**clf = svm.LinearSVC(dual=False)**

**clf.fit(X, y)**

**print("LinearSVC:")**

**cv\_results = print\_cross\_validate(clf, X, y, 5)**

**cv\_results = print\_cross\_validate(clf, X, y, 10)**

****

#### **Question 4:**

In this part, we applied the SVC linear and non-linear kernels on our data using 5-fold and 10-fold cross validation. We applied the SVC Polynomial kernel for degrees 0 up to 9. We also applied the RBF kernel where we used the default hyperparameters, the maximum and minimum value for the C and gamma hyperparameters, and we tweaked the values for gamma and C. Finally, we applied the Sigmoid kernel on the data. Moreover, we calculated the training and testing accuracy for each kernel as well as calculating the averages.

**#-------------------------**

**# SVC Linear Kernel Classifier**

**#-------------------------**

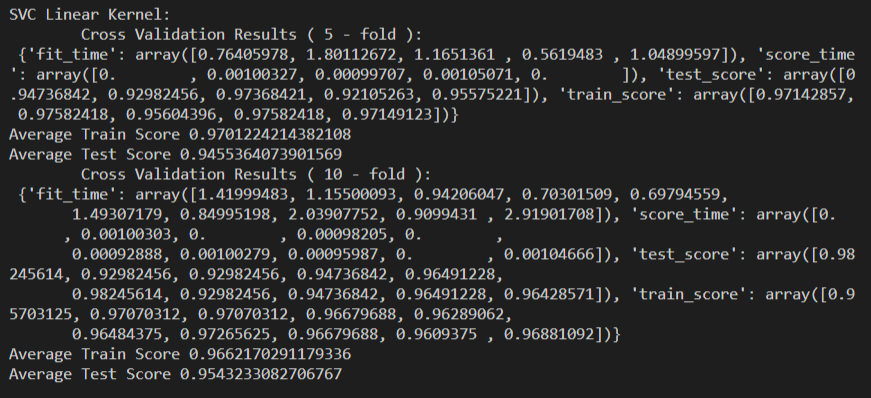
**clf = svm.SVC(kernel='linear')**

**clf.fit(X, y)**

**print("\nSVC Linear Kernel:")**

**cv\_results = print\_cross\_validate(clf, X, y, 5)**

**cv\_results = print\_cross\_validate(clf, X, y, 10)**

****

**#-------------------------**

**# SVC Polynomial Kernel Classifier**

**#-------------------------**

**def polynomial\_results(arg\_degree=3):**

**clf = svm.SVC(kernel='poly', degree=arg\_degree)**

**clf.fit(X, y)**

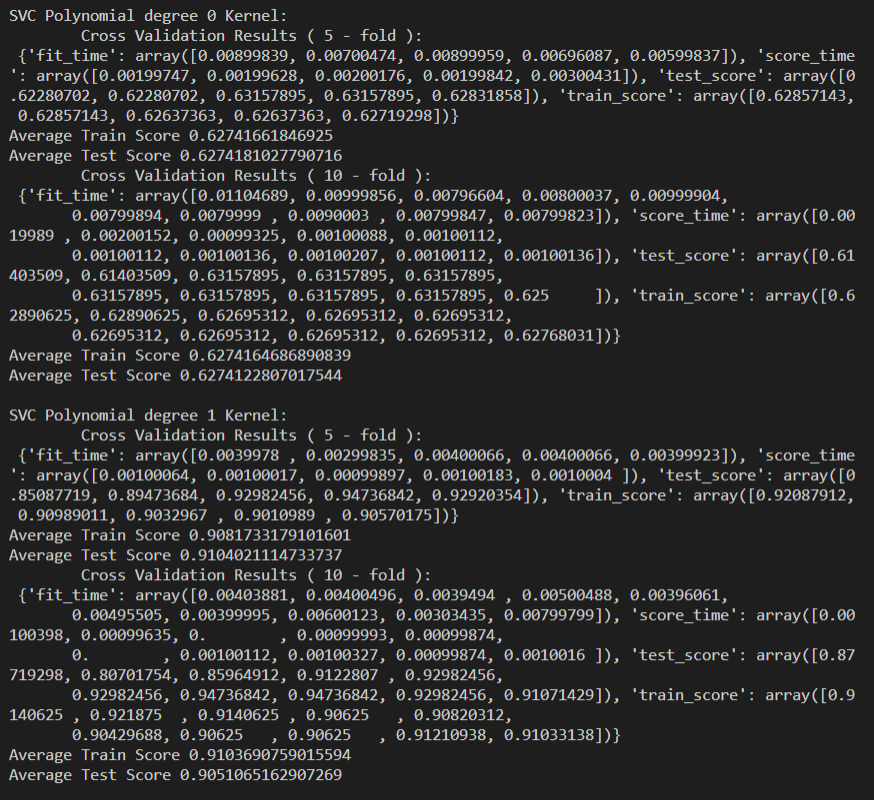
**print("\nSVC Polynomial degree", arg\_degree, "Kernel:")**

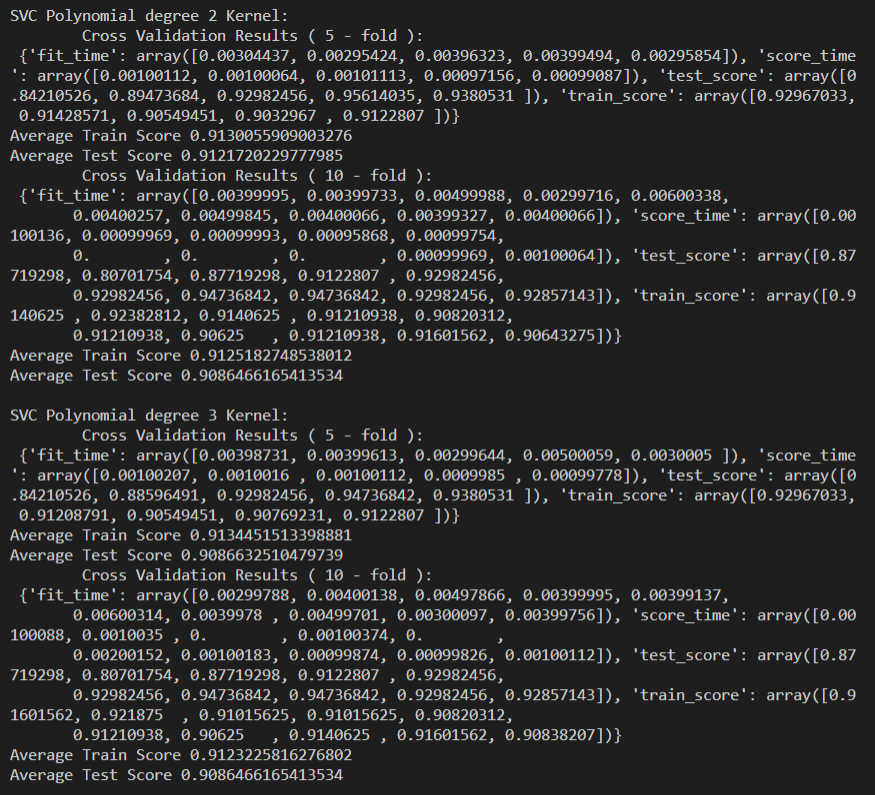
**cv\_results = print\_cross\_validate(clf, X, y, 5)**

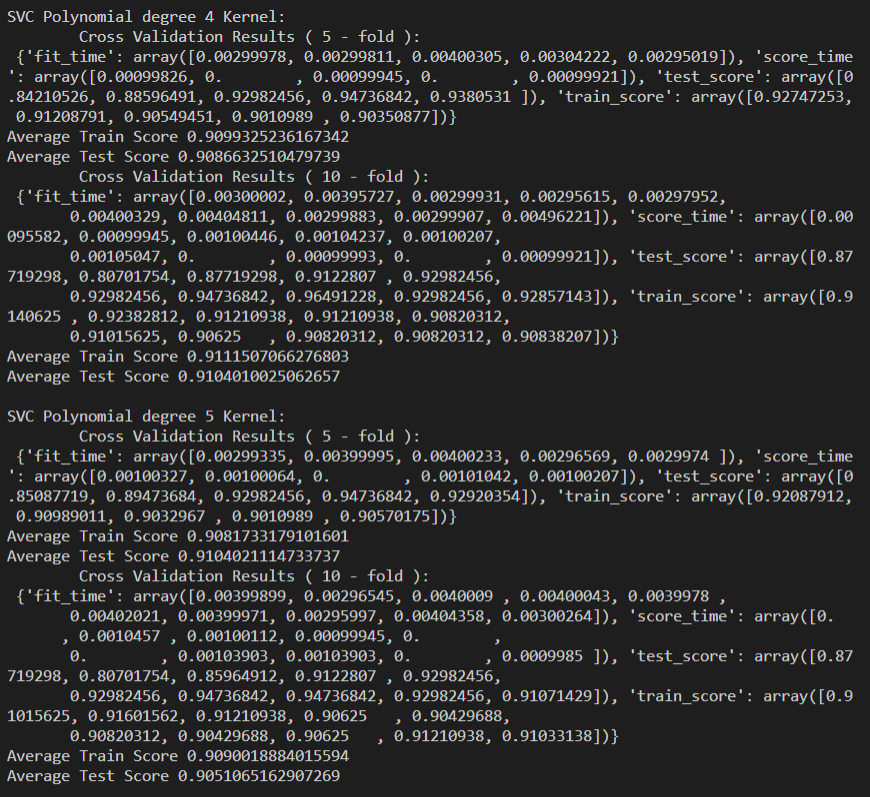
**cv\_results = print\_cross\_validate(clf, X, y, 10)**

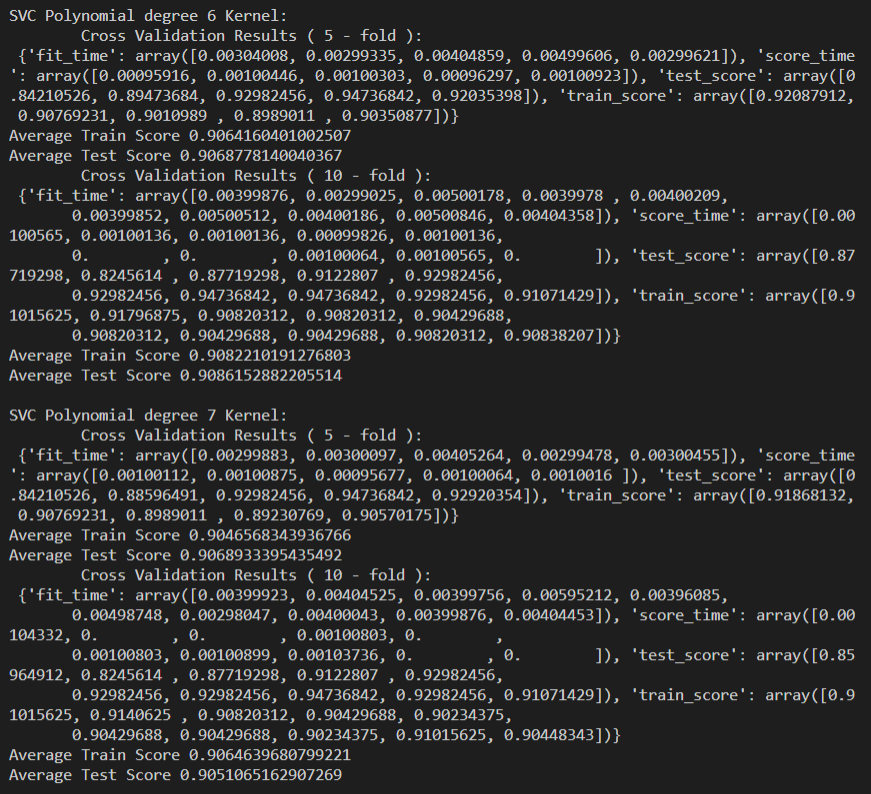
**for i in range (10):**

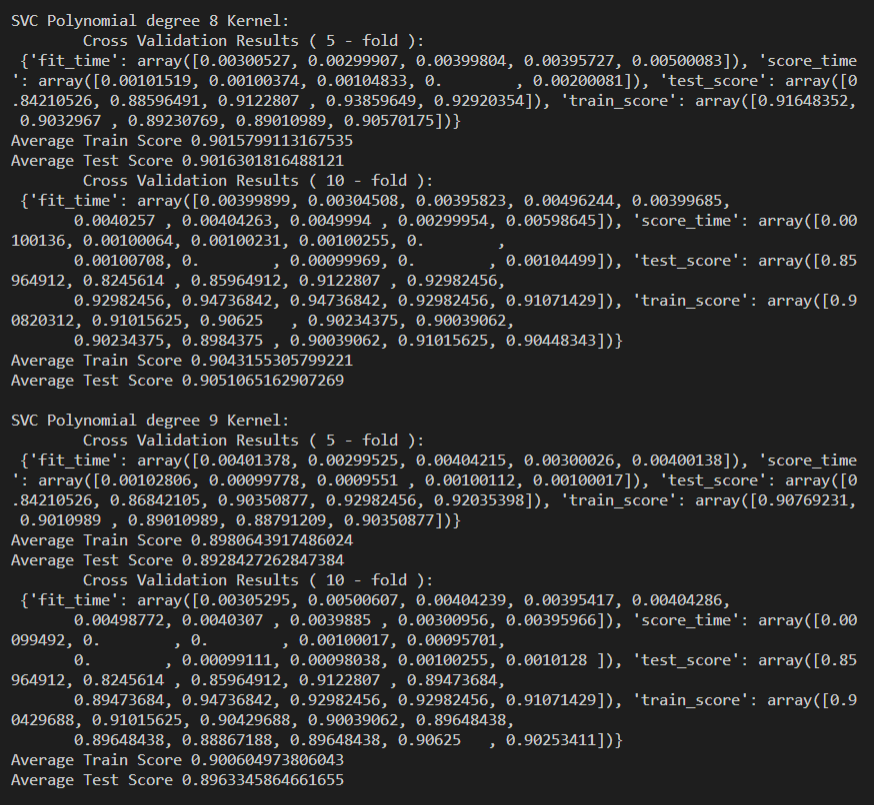
**polynomial\_results(i)**

****

****

****

****

****

**#-------------------------**

**# SVC RBF Kernel Classifier**

**#-------------------------**

**def rbf\_results(arg\_C=1.0, arg\_gamma='scale'):**

**clf = svm.SVC(kernel='rbf', C=arg\_C, gamma=arg\_gamma)**

**clf.fit(X, y)**

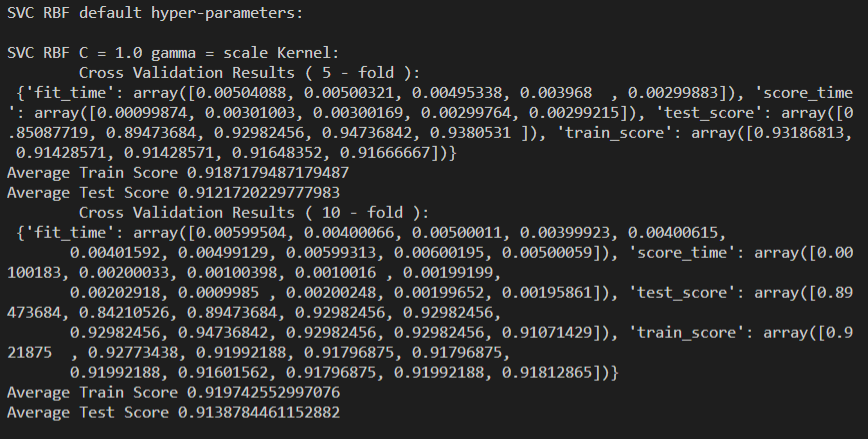
**print("\nSVC RBF C =", arg\_C, "gamma =", arg\_gamma, "Kernel:")**

**cv\_results = print\_cross\_validate(clf, X, y, 5)**

**cv\_results = print\_cross\_validate(clf, X, y, 10)**

**print("\nSVC RBF default hyper-parameters:")**

**rbf\_results()**

****

**print("\nTweaking C:\n")**

**for i in range(-1, 3):**

**if(i==-1):**

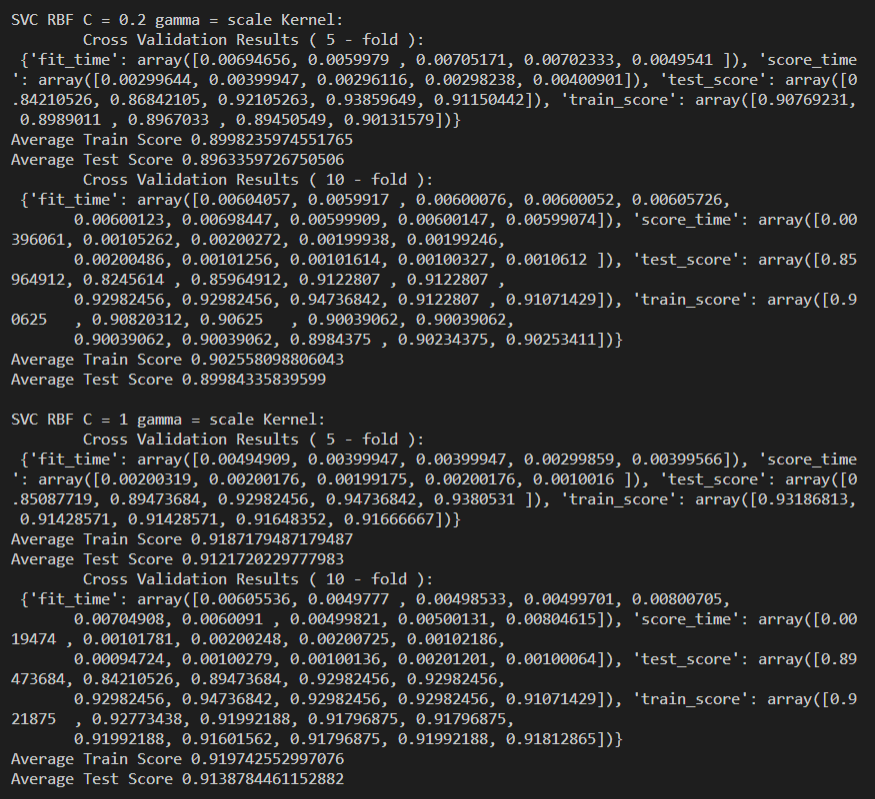
**rbf\_results(0.2)**

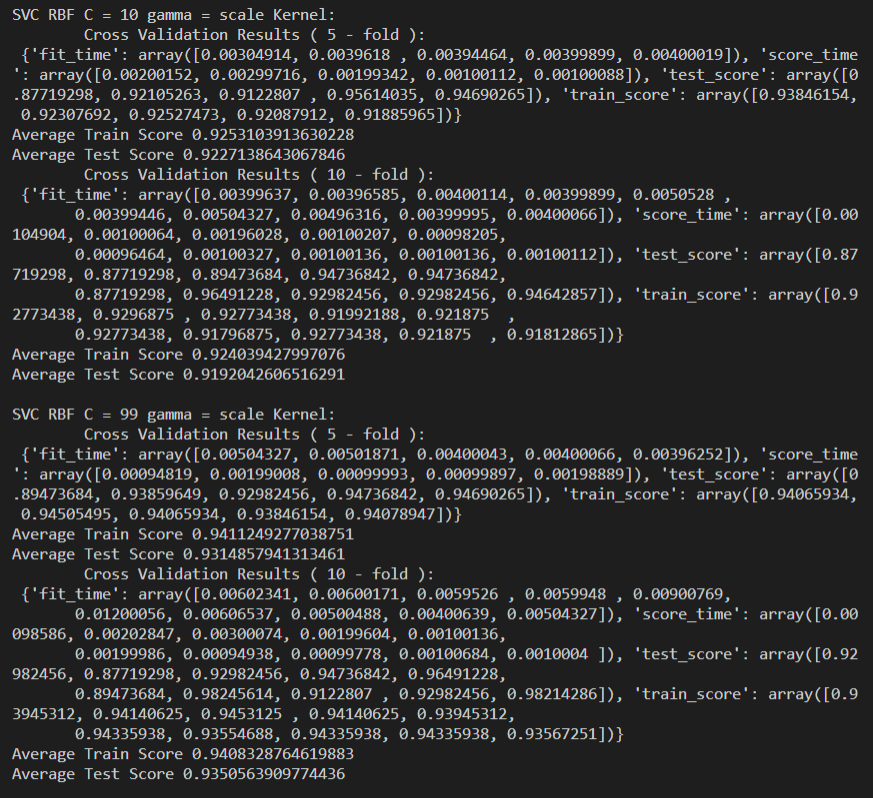
**elif(i==2):**

**rbf\_results(99)**

**else:**

**rbf\_results(10 \*\* i)**

****

****

**print("\nTweaking gamma:\n")**

**for i in range(-4, 2):**

**if(i==-4):**

**rbf\_results(0.0002)**

**elif(i==1):**

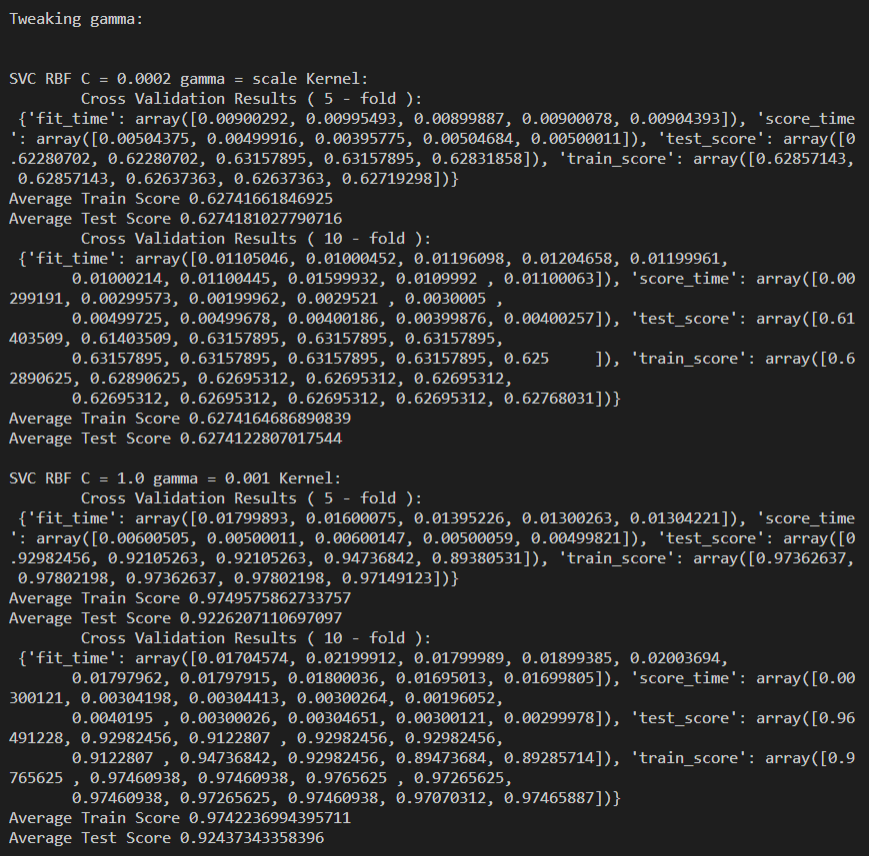
**rbf\_results(9)**

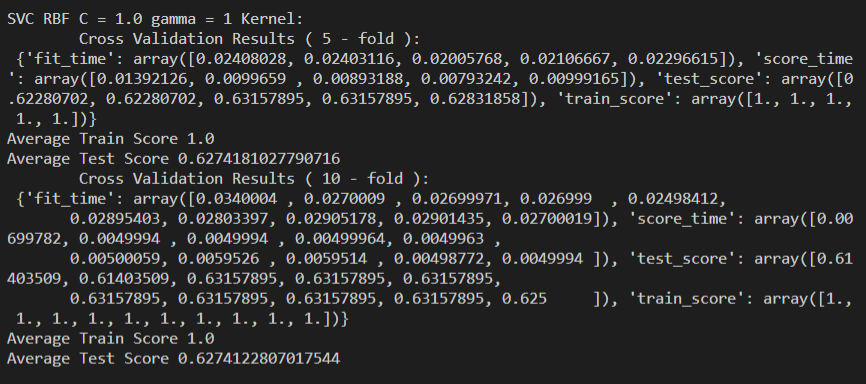
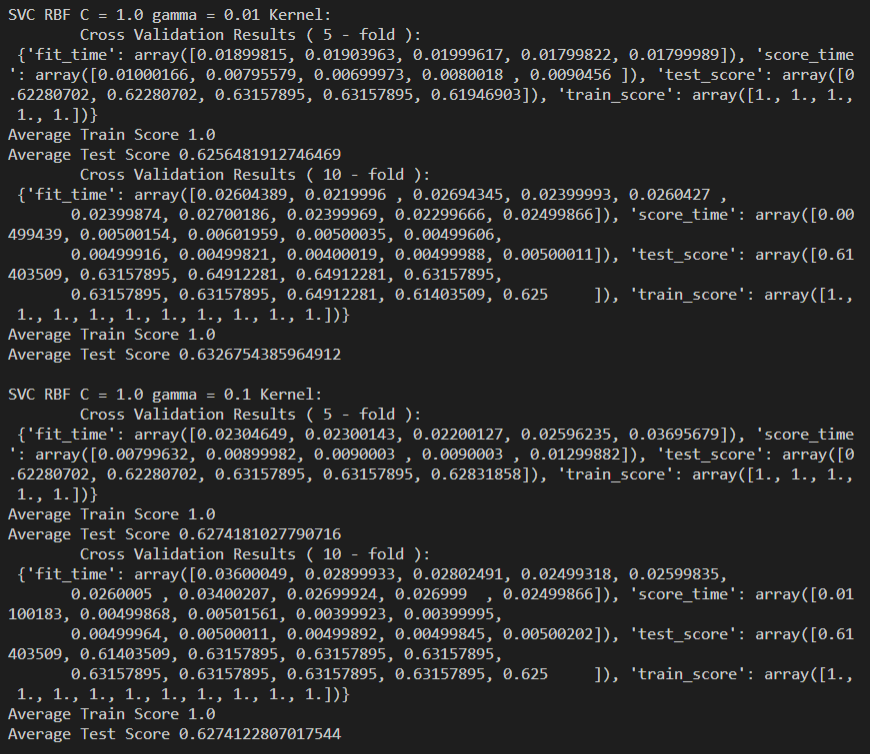
**else:**

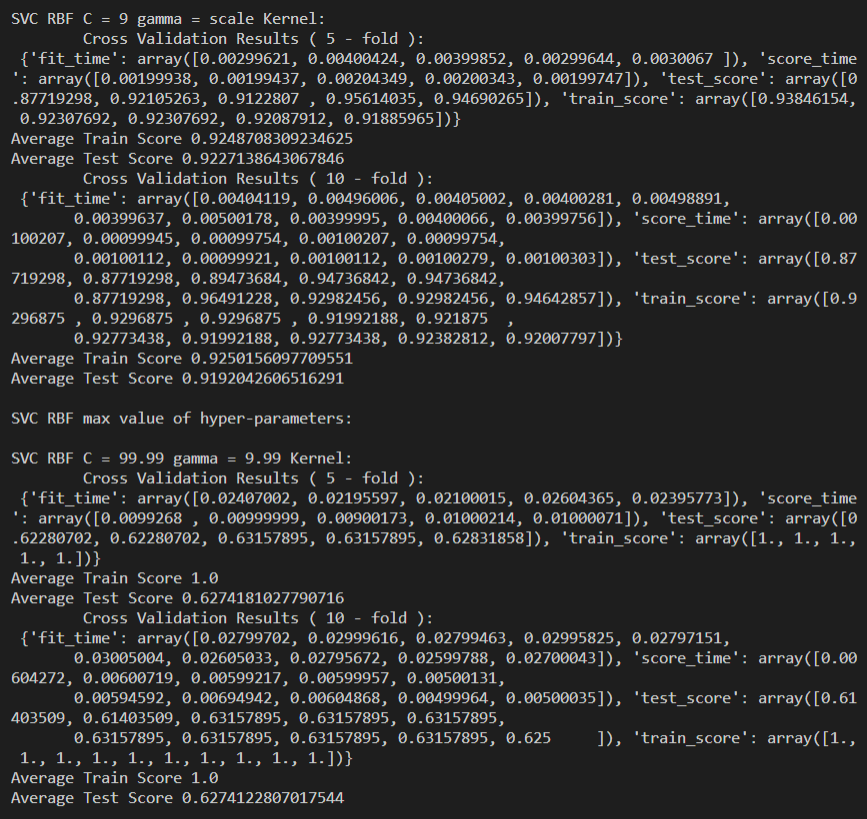
**rbf\_results(arg\_gamma = 10 \*\* i)**

**print("\nSVC RBF max value of hyper-parameters:")**

**rbf\_results(99.99, 9.99)**

****

****

****

**#-------------------------**

**# SVC Sigmoid Kernel Classifier**

**#-------------------------**

**def sigmoid\_results():**

**clf = svm.SVC(kernel='sigmoid')**

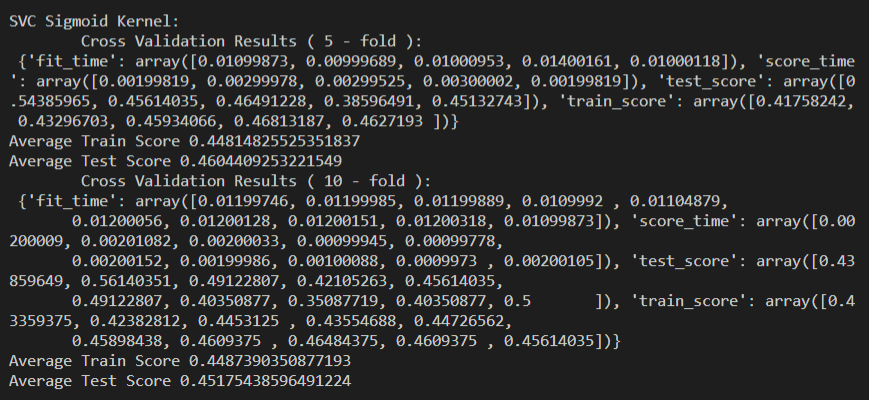
**clf.fit(X, y)**

**print("\nSVC Sigmoid Kernel:")**

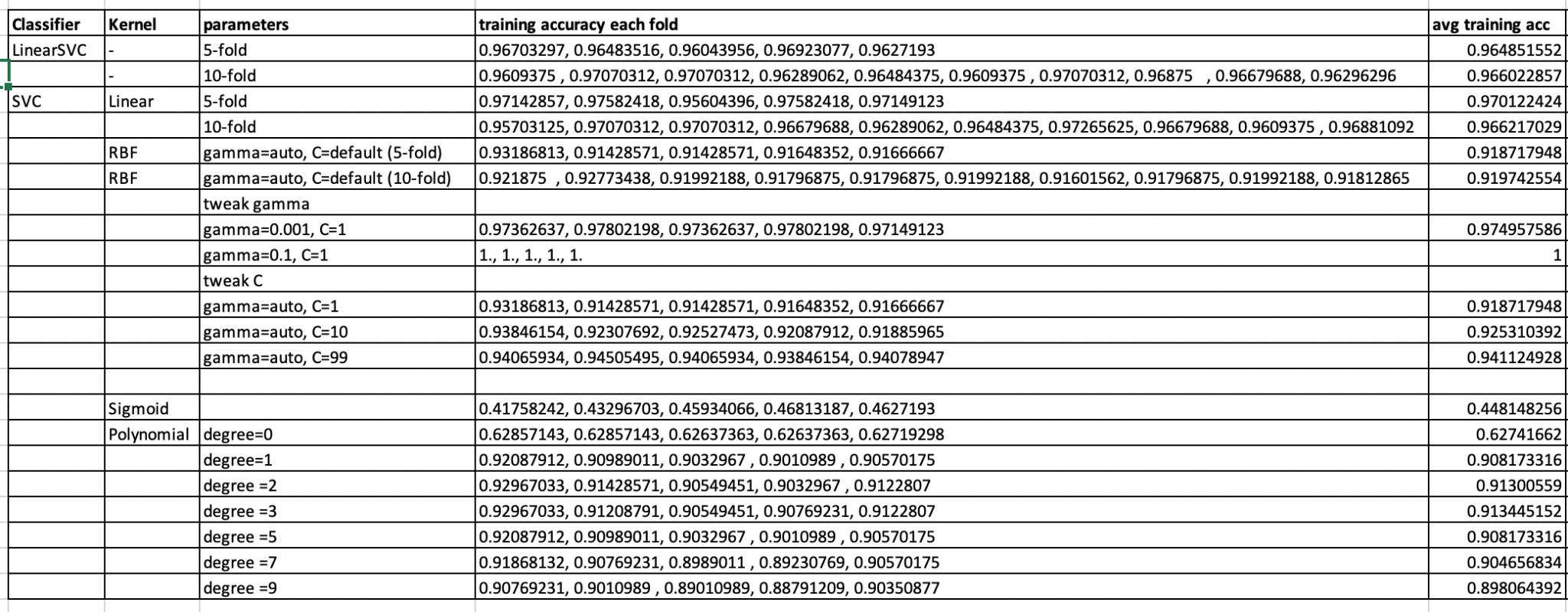
**cv\_results = print\_cross\_validate(clf, X, y, 5)**

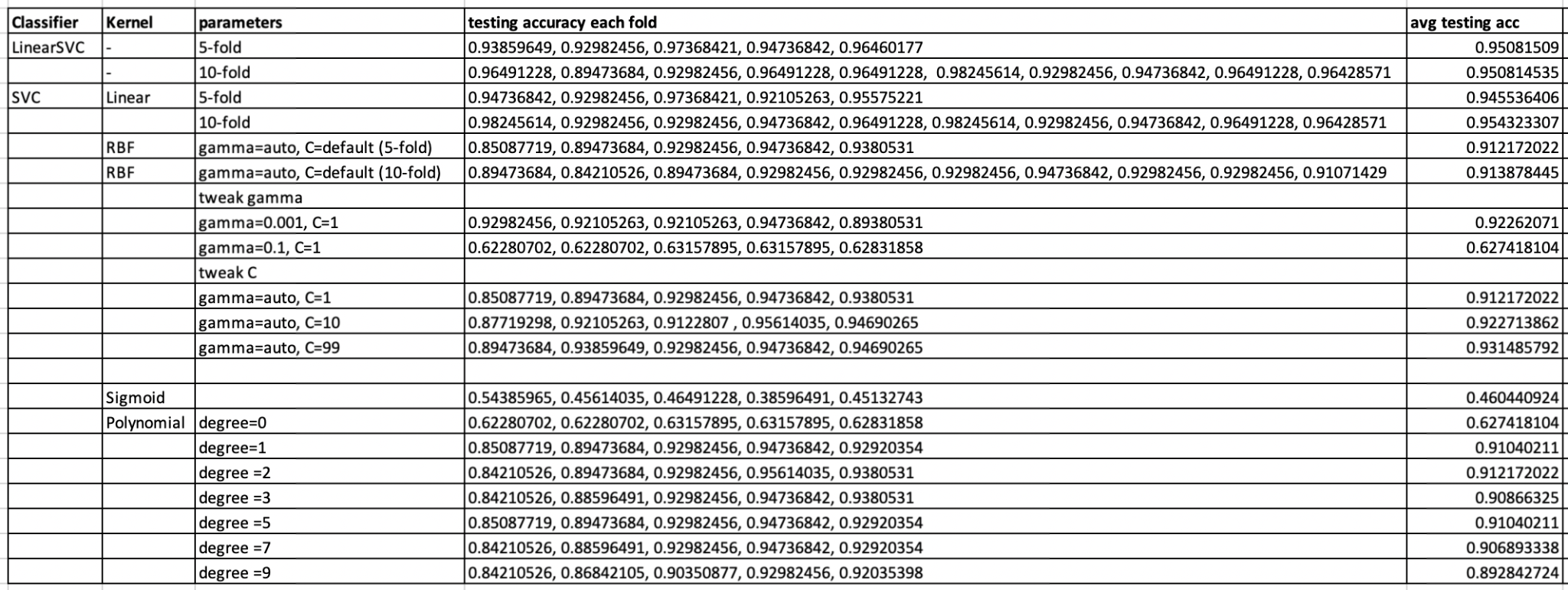
**cv\_results = print\_cross\_validate(clf, X, y, 10)**

**sigmoid\_results()**

****

##### **Question 5:**





The results used here in the table are for both 5-fold and 10-fold cross validation for LinearSVC, SVC linear kernel and RBF for the default parameters. However, for the rest of the kernels, RBF where gamma is tweaked, RBF where C is tweaked, Sigmoid, and Polynomial, only 5-fold cross validation results are used.

|  |  |  |  |
| --- | --- | --- | --- |
| The results show that the SVC with the Linear kernel and RBF with a low gamma value (0.001) have the highest training accuracy and highest testing accuracy from all the classifiers. | | | |
| Furthermore, the sigmoid kernel has the lowest training accuracy as well as the lowest testing accuracy. | | | |