**MACHINE LEARNING PROJECT**

In this project, I have used SVM Regression Machine Learning model for the digits classification dataset from Kaggle website

**PROBLEM STATEMENT:**

**Automatic digit recognition is of popular interest today.**

**Deep Learning techniques makes it possible for object recognition in image data . This practice problem is meant to give you a kick start in deep learning. As usual, we will not only provide you with the challenge and a solution checker, but also a set of tutorials to get you off the ground!**

**The data set used for this problem is from the populat MNIST data set. Developed by Yann LeCun, Corina Cortes and Christopher Burger for evaluating machine learning model on the handwritten digit classification problem. It is a widely used data set in the machine learning community**

**ML METHODOLOGY:**

SVM a support vector machine(SVM)is machine learning algorithm that analyzes data for classification and regression analysis….SVMs are used in text categorization ,image classification ,handwriting recognition and in the science.

**DATASET DESCRIPTION:**

Some relevant columns in the dataset

* Label
* Image id

**PRE\_PROCESSING:**

Pre\_processing refers to the transformations applied to our data before feeding it to the algorithm.

%matplotlib inline

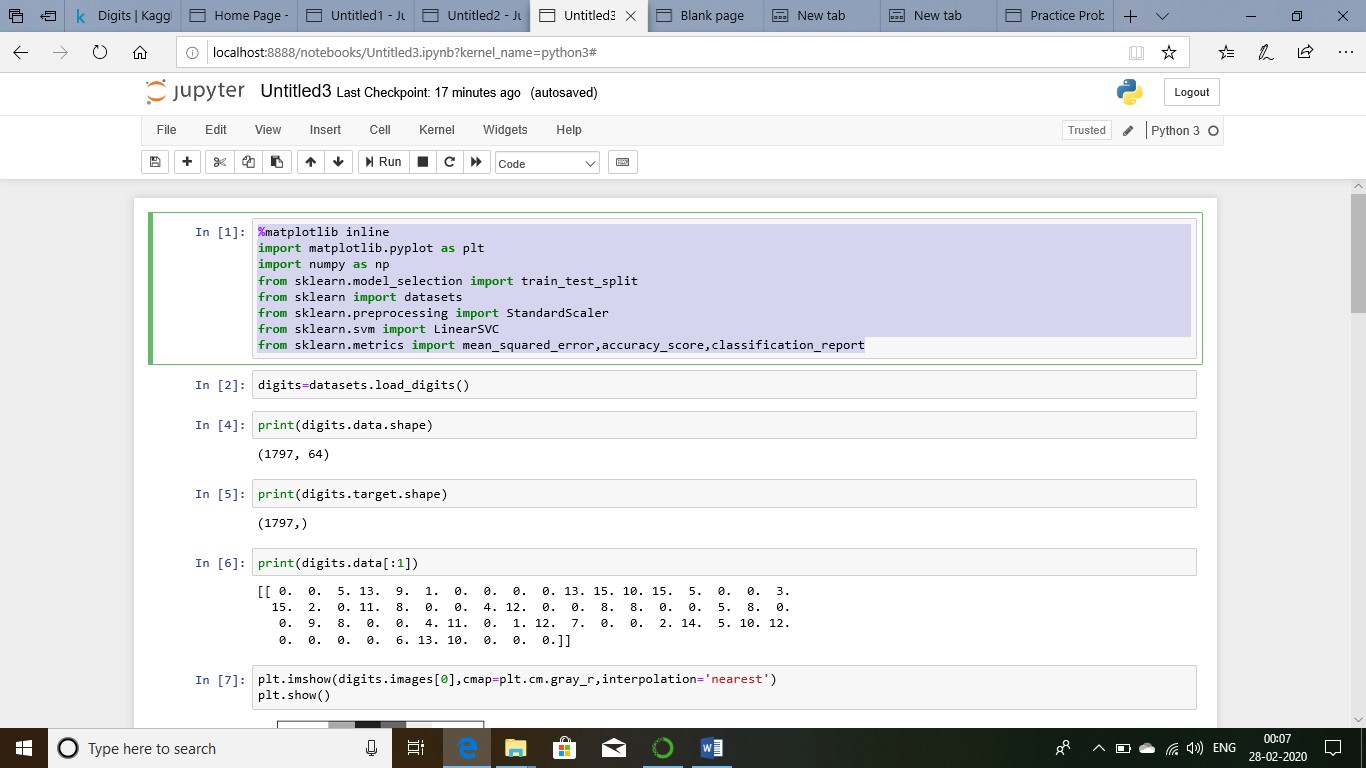
import matplotlib.pyplot as plt import numpy as np

from sklearn.model\_selection import train\_test\_split from sklearn import datasets

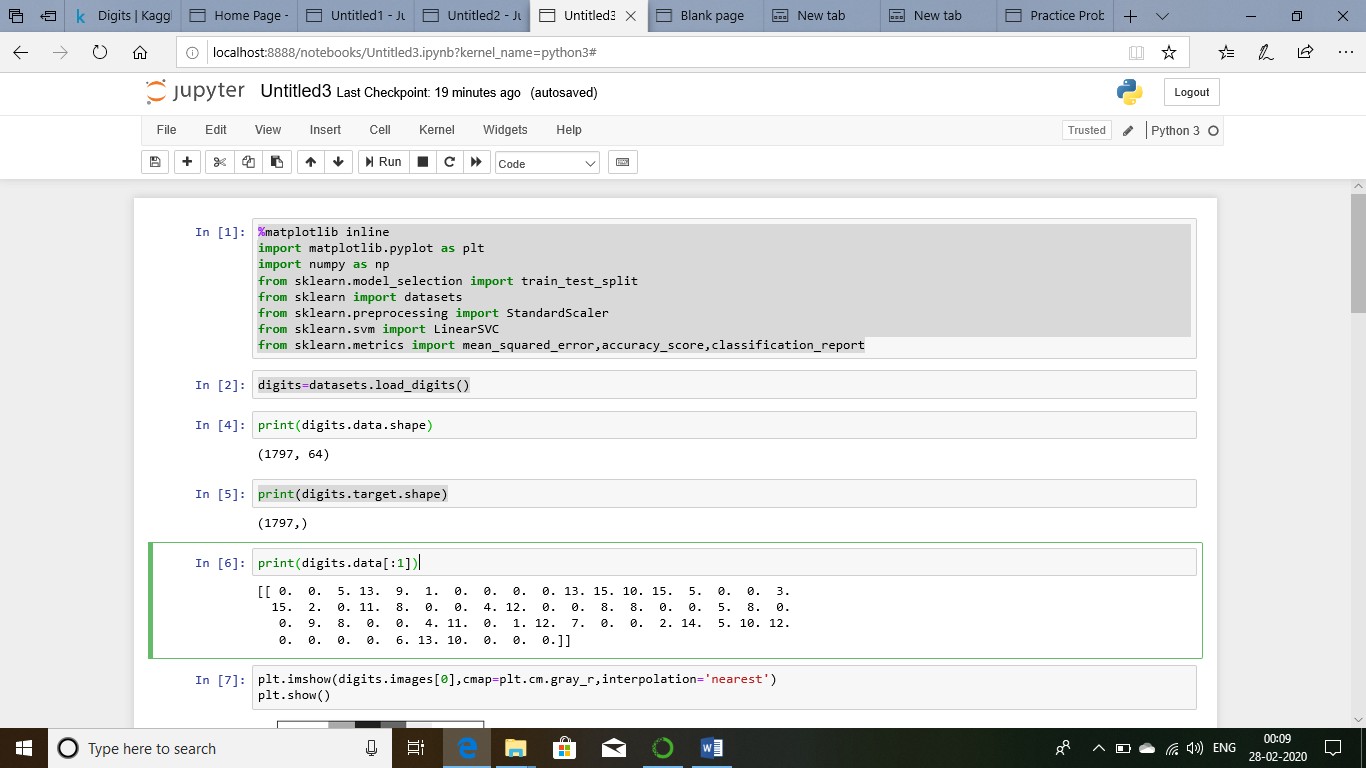
from sklearn.preprocessing import StandardScaler

from sklearn.svm import LinearSVC fromsklearn.metricsimport

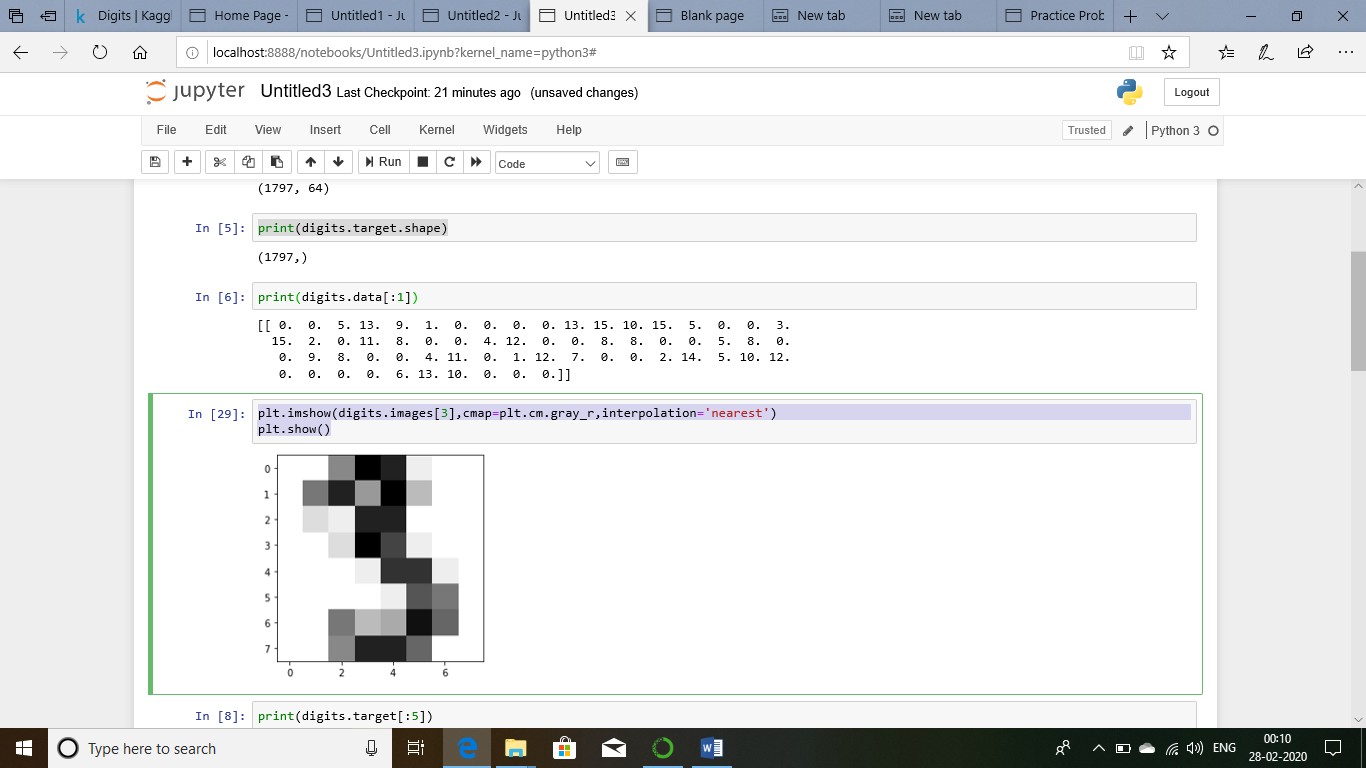
mean\_squared\_error,accuracy\_score,classification\_report



digits=datasets.load\_digits() print(digits.data.shape) print(digits.data.shape) print(digits.data[:1])



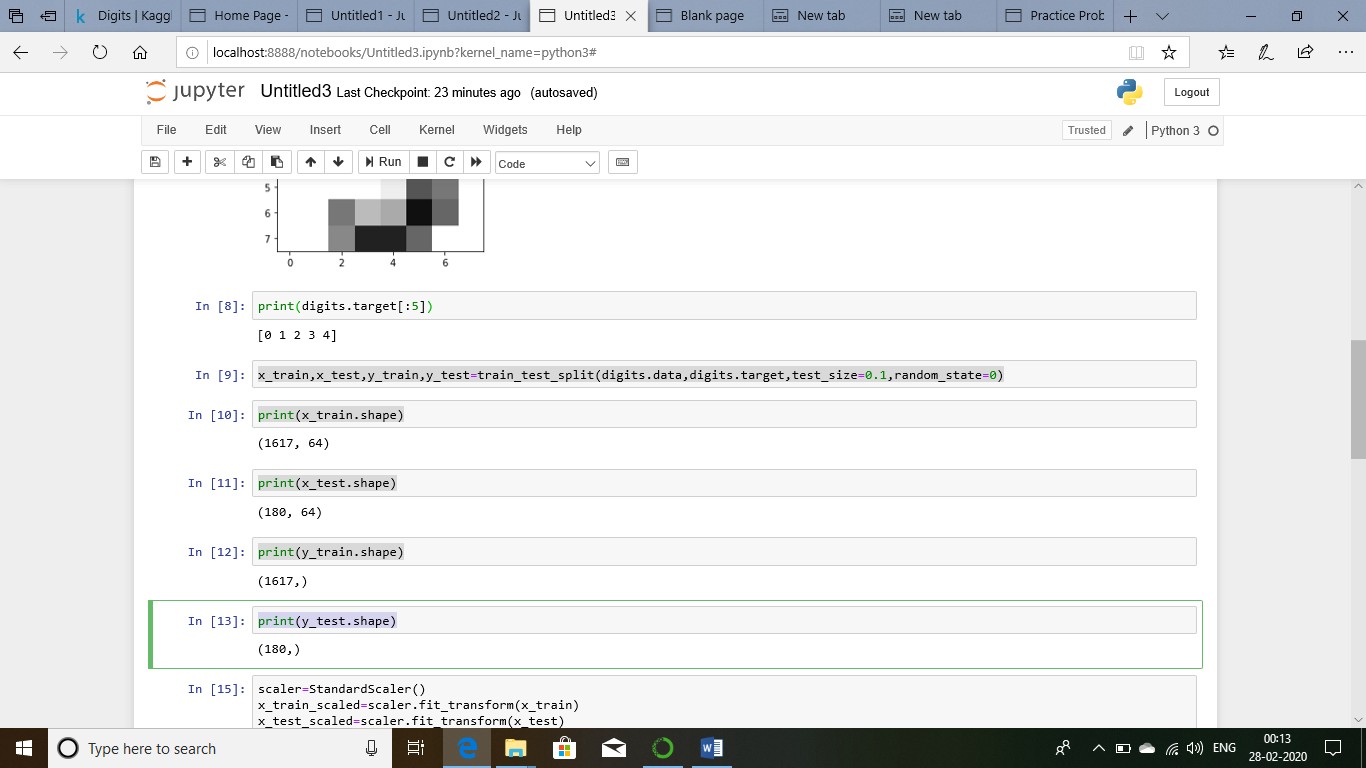
plt.imshow(digits.images[3],cmap=plt.cm.gray\_r,interpolation='nearest') plt.show()



print(digits.target[:5])

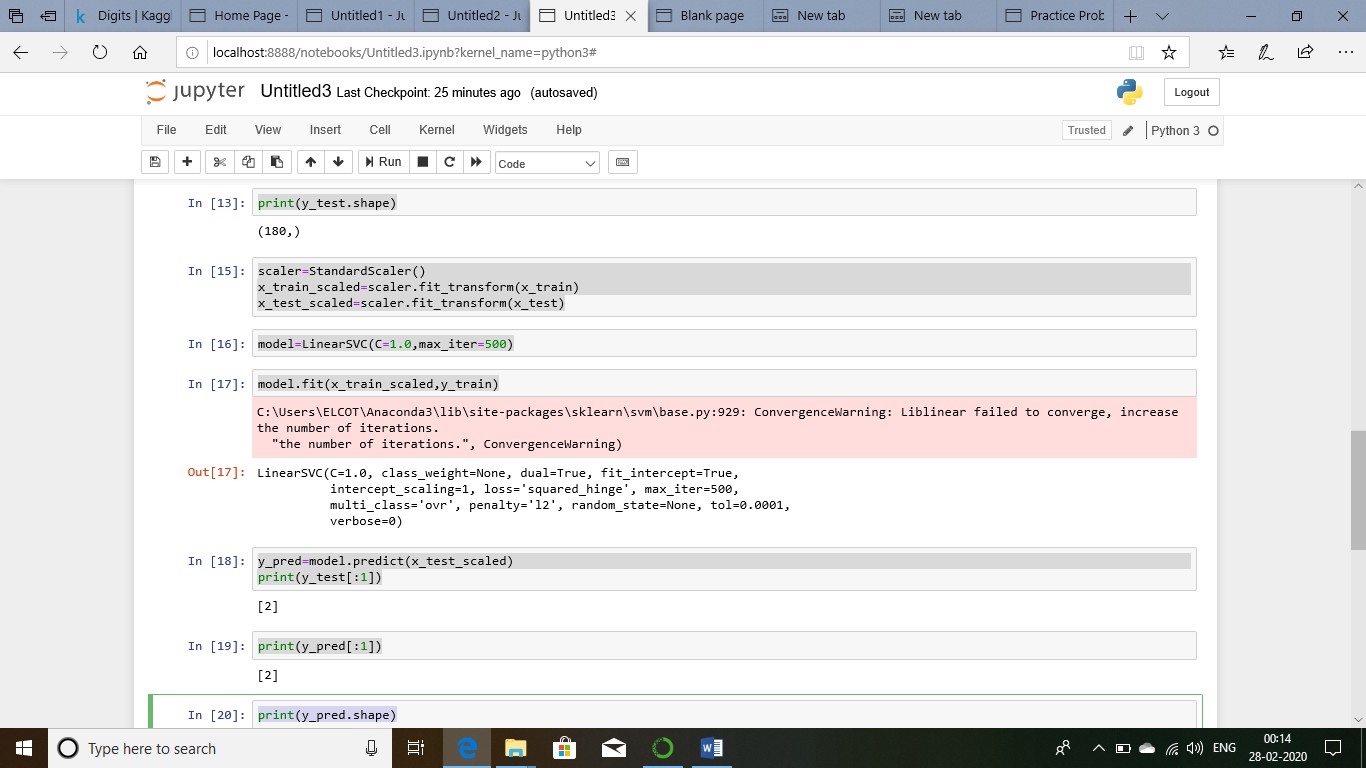
x\_train,x\_test,y\_train,y\_test=train\_test\_split(digits.data,digits.target,test\_siz e=0.1,random\_state=0)

print(x\_train.shape) print(x\_test.shape) print(y\_train.shape) print(y\_test.shape)



scaler=StandardScaler() x\_train\_scaled=scaler.fit\_transform(x\_train) x\_test\_scaled=scaler.fit\_transform(x\_test) model=LinearSVC(C=1.0,max\_iter=500) model.fit(x\_train\_scaled,y\_train) y\_pred=model.predict(x\_test\_scaled) print(y\_test[:1])

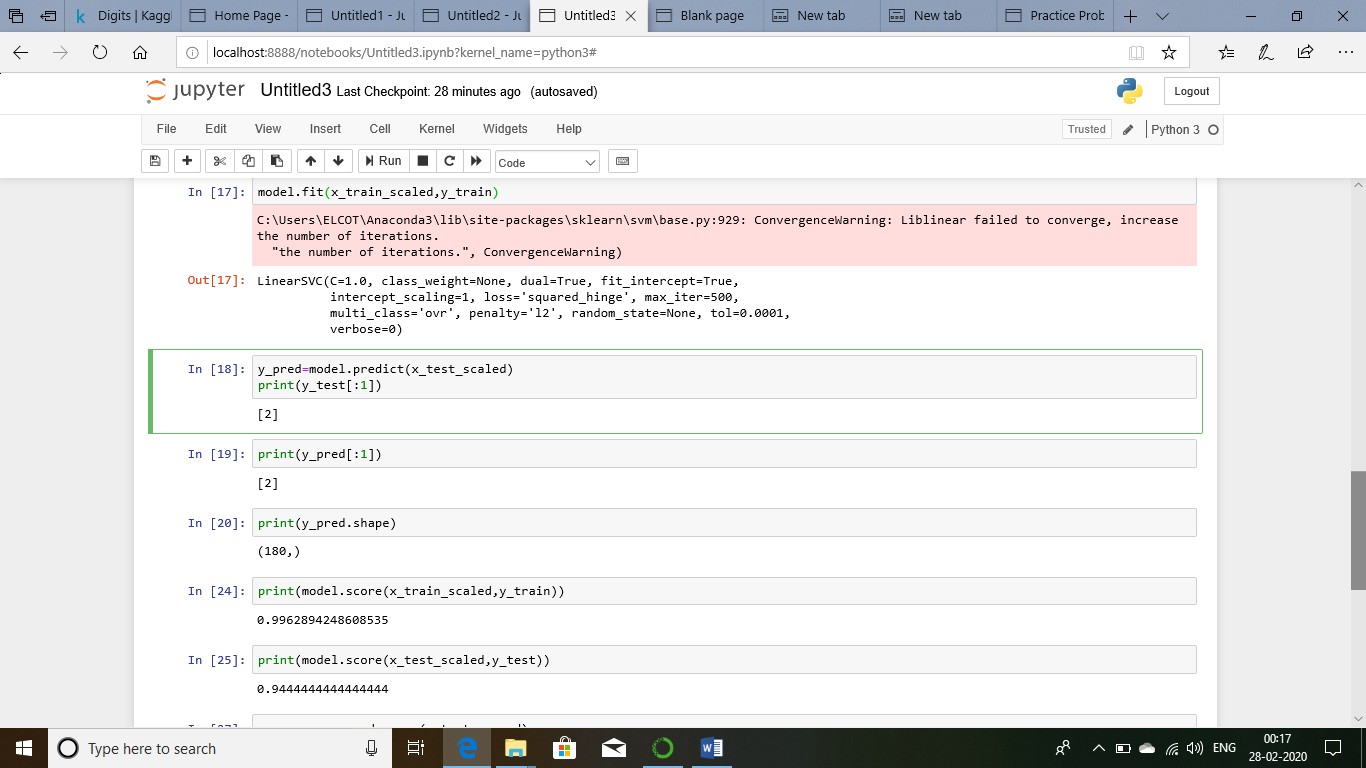
print(y\_pred[:1]) print(y\_pred.shape)

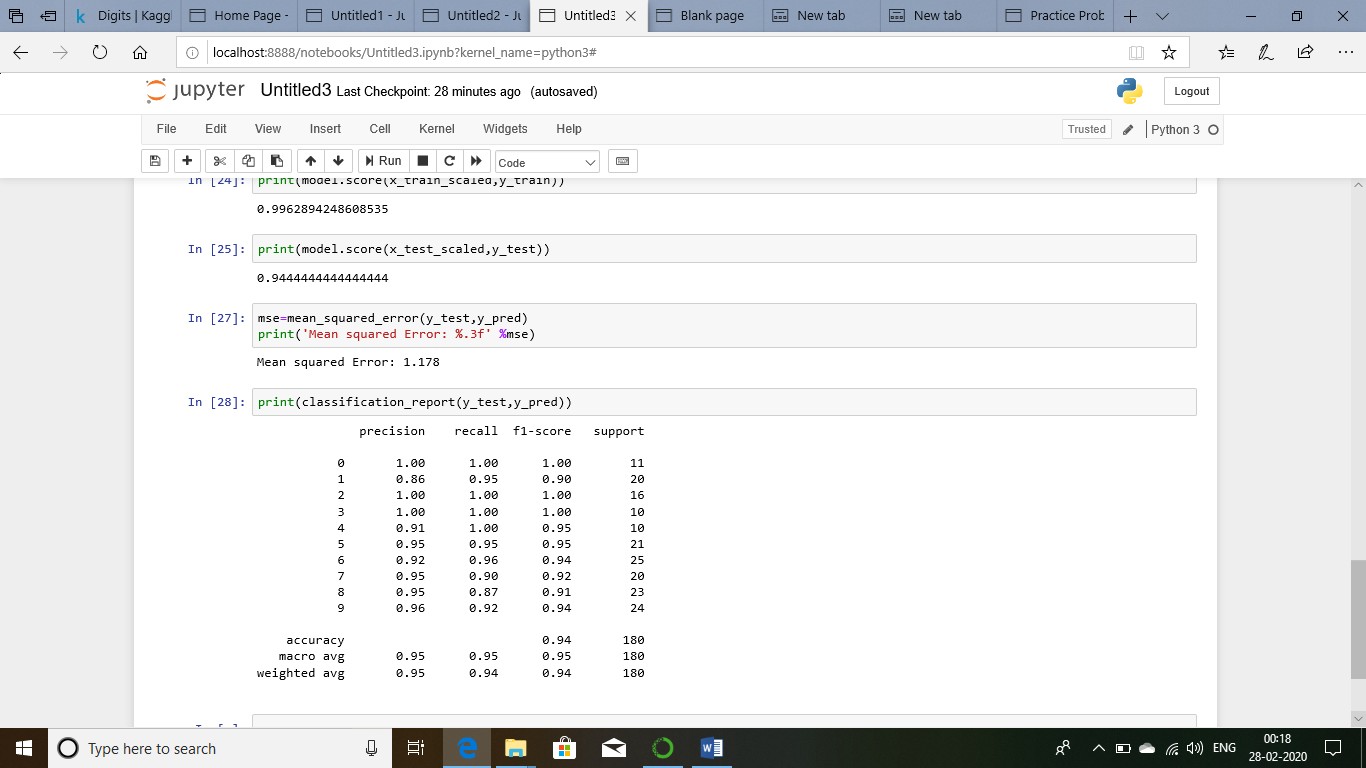


y\_pred=model.predict(x\_test\_scaled) print(y\_test[:1])

print(y\_pred[:1]) print(y\_pred.shape)

print(model.score(x\_train\_scaled,y\_train)) print(model.score(x\_test\_scaled,y\_test)) mse=mean\_squared\_error(y\_test,y\_pred) print('Mean squared Error: %.3f' %mse) print(classification\_report(y\_test,y\_pred))





Conclusion:

The digits classification using linear svm method

The data split into training test and testing test using linear svc classifier and values are calcluated by accuracy and mean squared method.