**Experiment 4**

**Student Name:** Sahil Kaundal **UID:** 21BCS8197

**Branch:** BE CSE (Lateral Entry) **Section/Group:** 616/A

**Semester:** 5th **Date of Performance:** 05/10/2022

**Subject Name:** ML Lab **Subject Code:** 20CSP-317

1. **Aim/Overview of the practical:**

Classifying data using Support Vector Machines (SVMs) in Python.

1. **Task To Be Done:**

Classifying data using Support Vector Machines (SVMs) in Python.

**3. Apparatus / Simulator Used:**

1. Windows 7 or above.
2. Google Collab.

**Introduction to SVMs:** In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyse data used for classification and regression analysis. A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

**What is Support Vector Machine?**

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification, implicitly mapping their inputs into high-dimensional feature spaces.

**Steps:**

Step 1: Importing Python libraries.

Step 2: Creating the dataset.

Step 3: Opening the dataset.

Step 4: Uploading the dataset.

Step 5: Feature Scaling and Normalization.

Step 6: Add a column of ones to the X vector.

Step 7: Plotting the dataset

Step 8: The Hypothesis (Linear Regression Model)

Step 9: Calculating the Cost Function

Step 10: Gradient Descent

Step 11: Predictions

Step 12: Intuitions concerning the learning constant α

Step 13: Contour plot of J and θ

Step 14: How to adapt the code for Multiple Variables

**4. Program / Commands:**

#Sahil Kaundal

#21BCS8197

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn import svm

X=np.array([1,5,1.5,8,1,9,7,8.7,2.3,5.5,7.7,6.1])

y=np.array([2,8,1.8,8,0.6,11,10,9.4,4,3,8.8,7.5])

plt.scatter(X,y)

plt.show()

#shaping data for training the model

training\_X = np.vstack((X,y)).T

training\_y = [0,1,0,1,0,1,1,1,0,0,1,1]

#define the model

clf = svm.SVC(kernel='linear', C=1.0)

#train the model

clf.fit(training\_X, training\_y)

#get the weight values for the linear equation from the trained SVM mode:

w = clf.coef\_[0]

#get the y-offset fot the linear equation

a = -w[0] / w[1]

#make the x-axis space for the data points

XX = np.linspace(0,13)

#get the y-value to plot the decision boundary

yy = a\*XX - clf.intercept\_[0]/w[1]

#plot the decision boundary

plt.plot(XX, yy, 'k-')

#show the plot visually

plt.scatter(training\_X[:, 0], training\_X[:, 1], c=training\_y)

plt.legend()

plt.show()

import matplotlib.pyplot as plt

import numpy as np

from sklearn import datasets

from sklearn import svm

#non-linear data

circle\_X,circle\_y = datasets.make\_circles(n\_samples=300, noise=0.05)

#show raw non-linear data

plt.scatter(circle\_X[:, 0], circle\_X[:, 1], c=circle\_y, marker='.')

plt.show()

#make non-linear algorithm for model

nonlinear\_clf = svm. SVC(kernel='rbf', C=1.0)

nonlinear\_clf.fit(circle\_X, circle\_y)

#plot the decision boundary for non linear SVM problem

def plot\_decision\_boundary(model, ax=None):

  if ax is None:

    ax = plt.gca()

  xlim = ax.get\_xlim()

  ylim = ax.get\_ylim()

  # create grid to evaluate model

  x=np.linspace(xlim[0],xlim[1],30)

  y=np.linspace(ylim[0],ylim[1],30)

  Y,X=np.meshgrid(y,x)

  #shape Data

  xy=np.vstack([X.ravel(),Y.ravel()]).T

  #get the decision boundary

  P=model.decision\_function(xy).reshape(X.shape)

#plot decision boundary

  ax.contour(X,Y,P,

             levels=[0],alpha=0.5,

             linestyles=['-'])

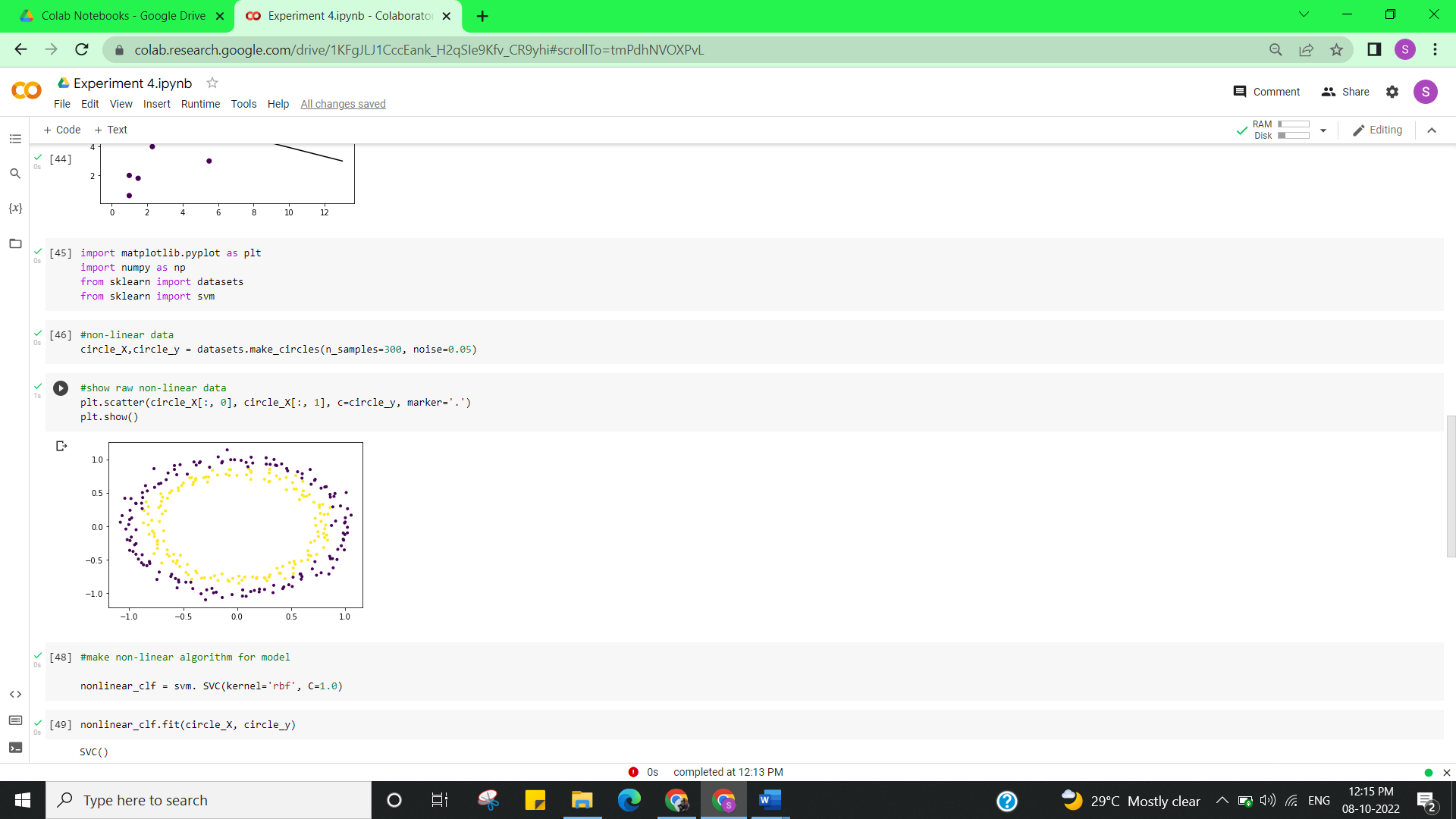
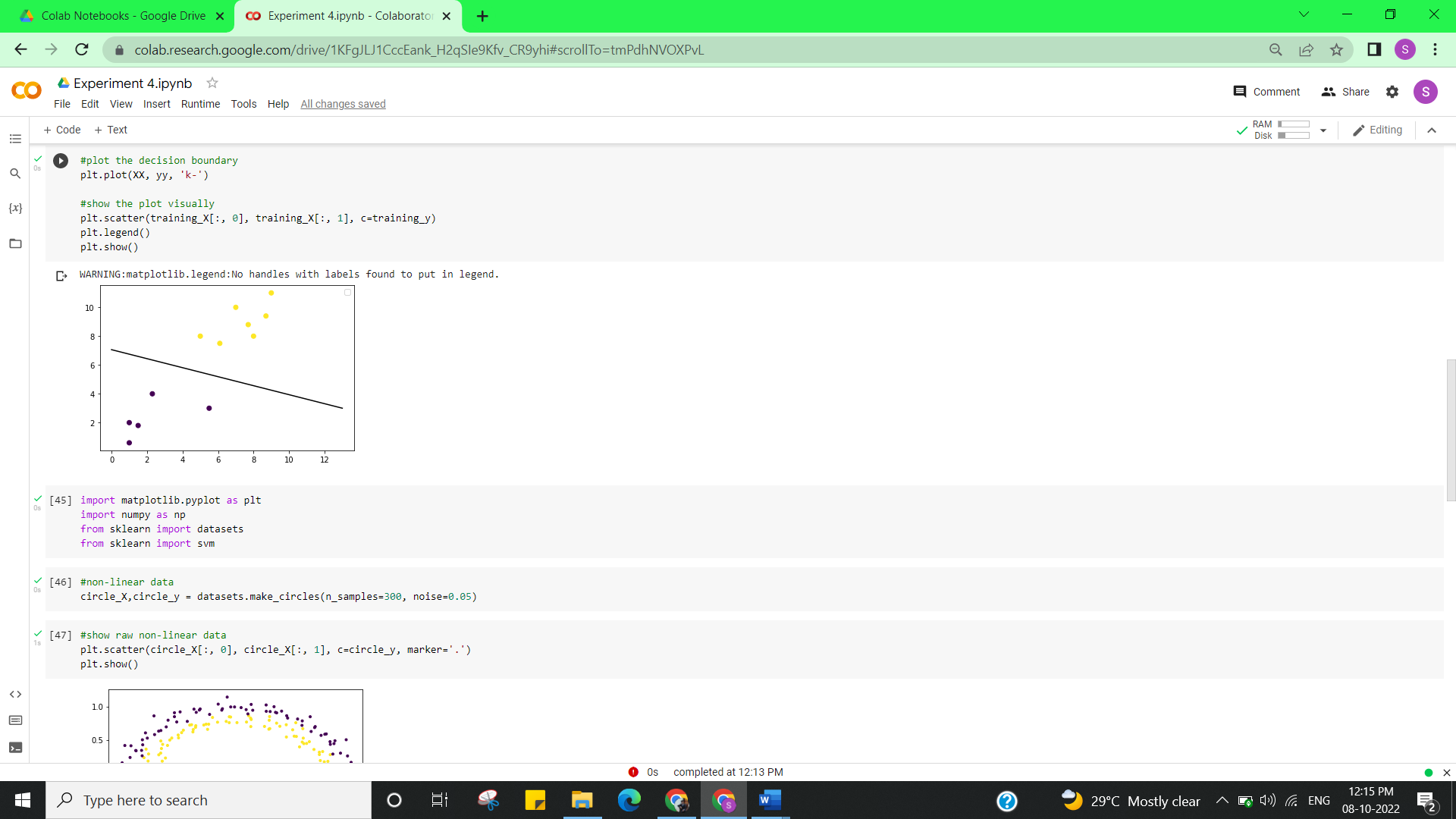
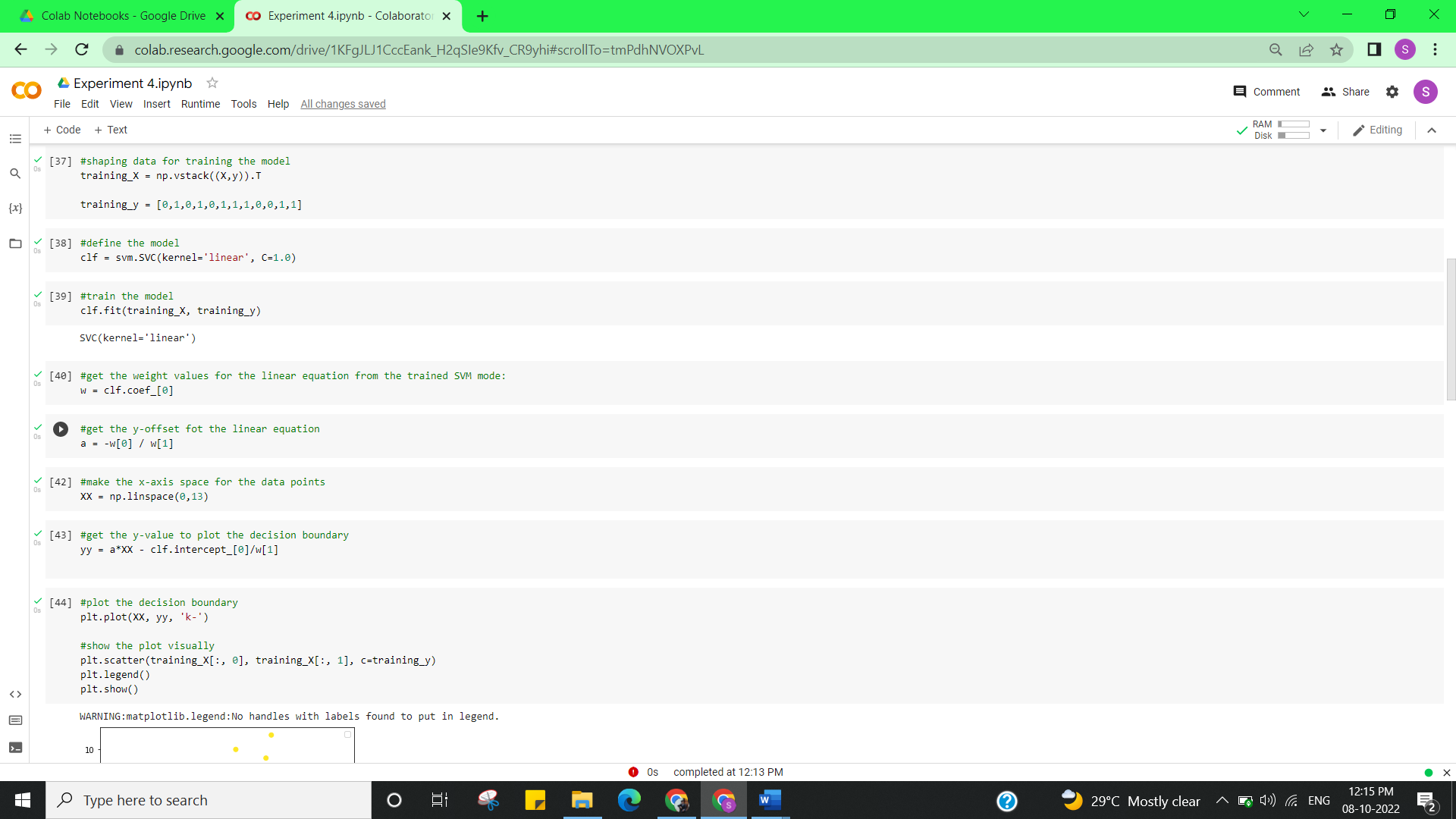
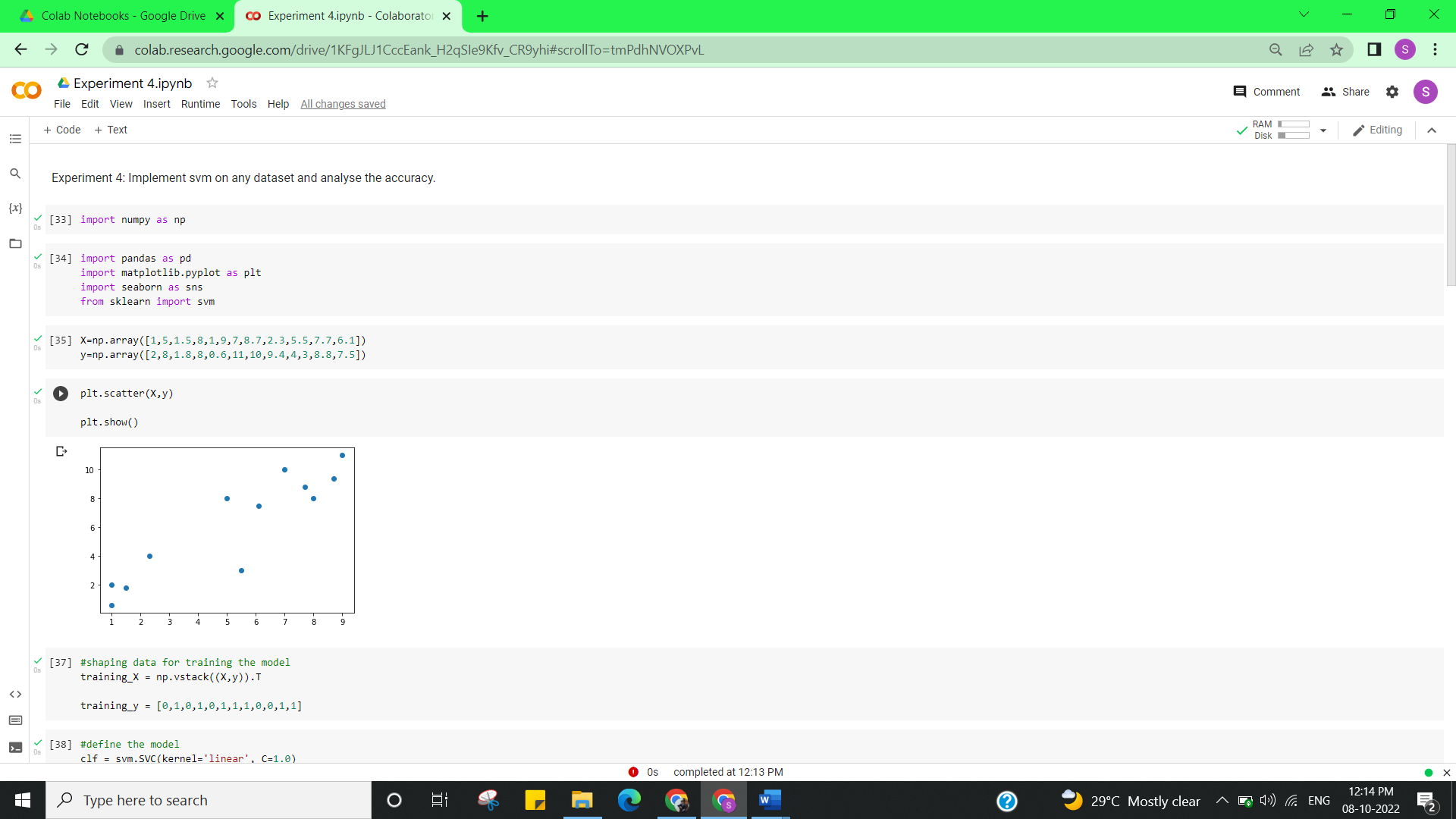
plt.scatter(circle\_x[:,0],circle\_x[:,1],c=circle\_y,s=50)

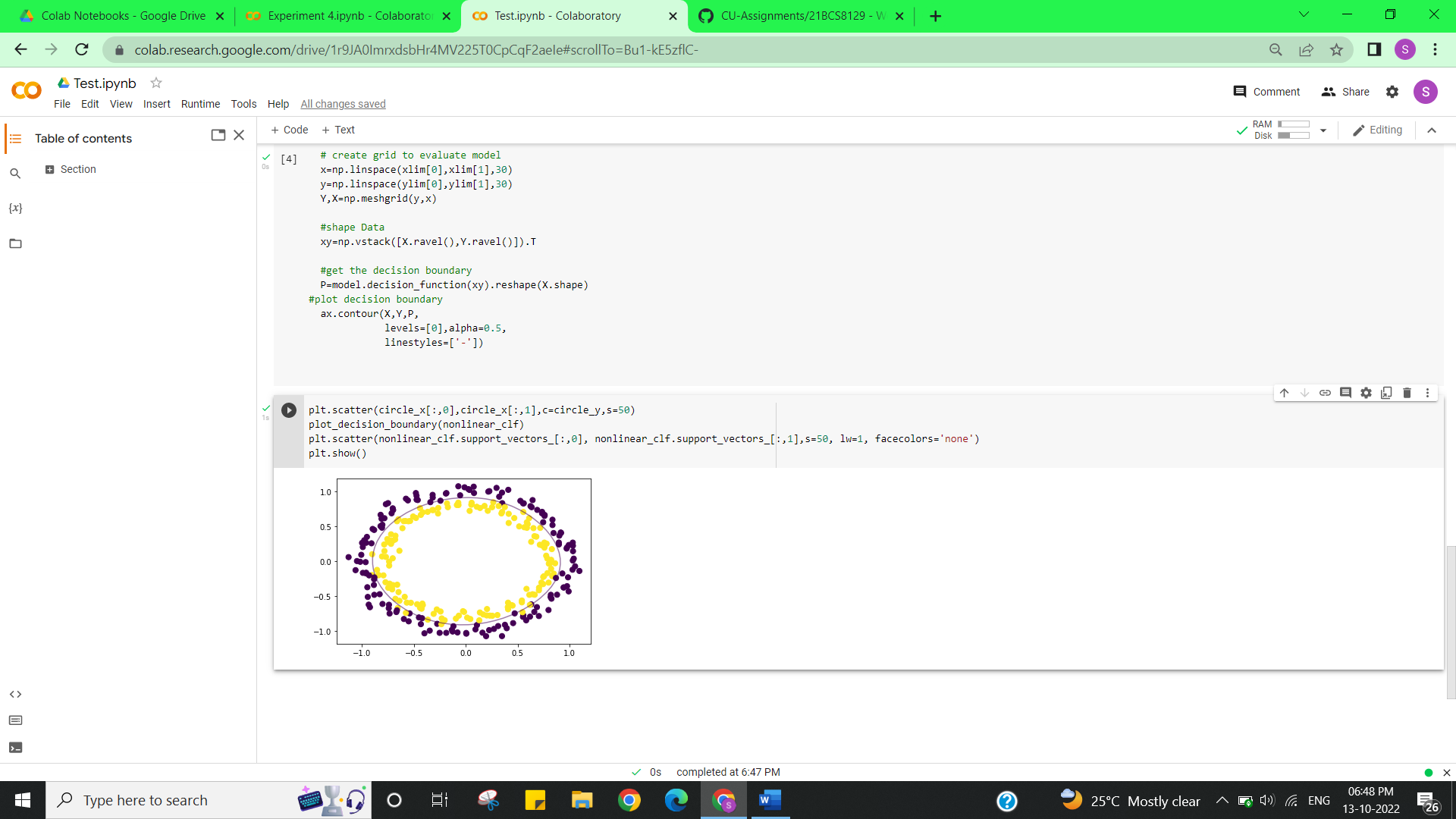
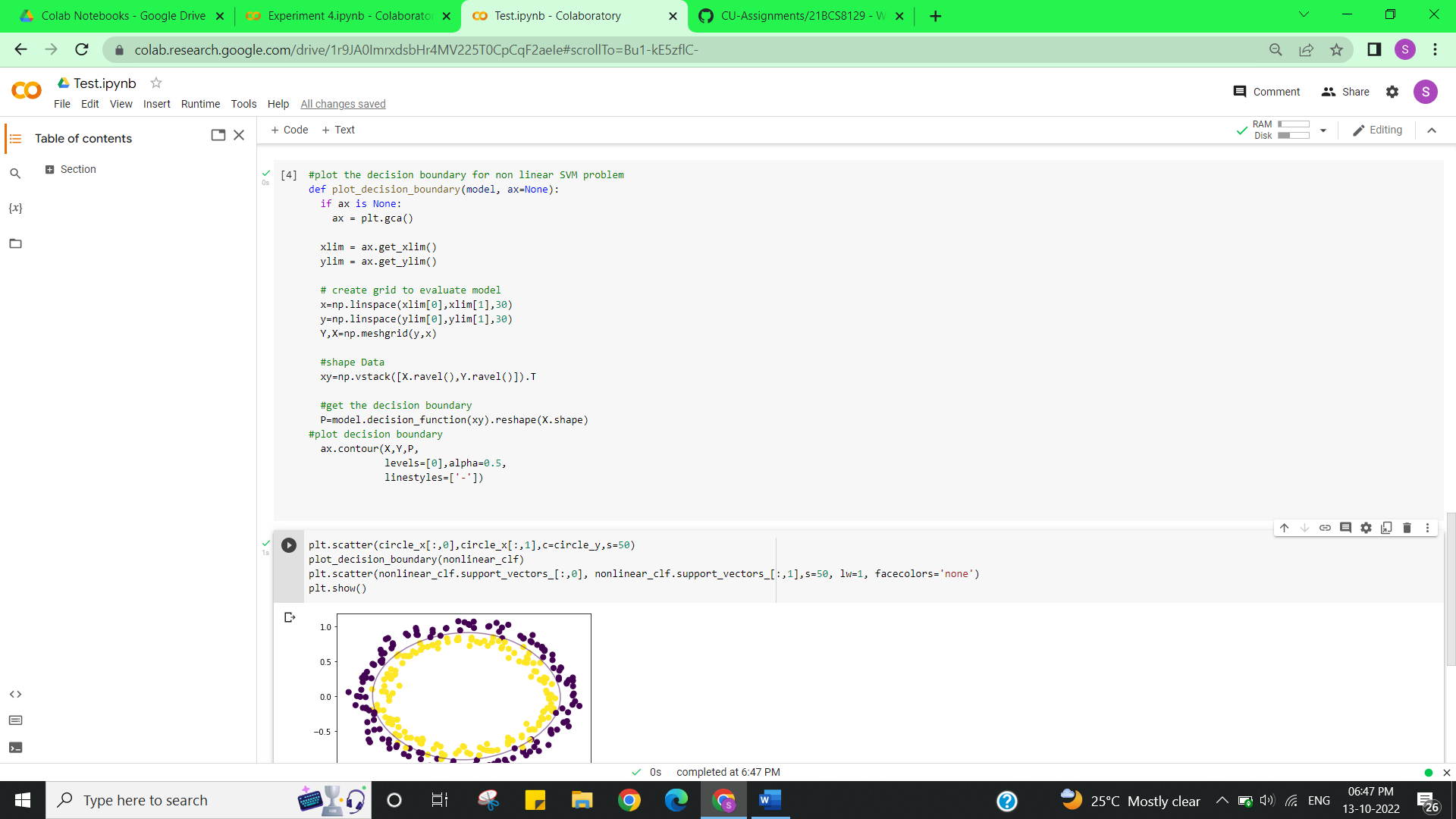
plot\_decision\_boundary(nonlinear\_clf)

plt.scatter(nonlinear\_clf.support\_vectors\_[:,0], nonlinear\_clf.support\_vectors\_[:,1],s=50, lw=1, facecolors='none')

plt.show()

1. **Result/Output/Writing Summary:**





**Learning outcomes (What I have learnt):**

* Learned about Classifying data using Support Vector Machines (SVMs) in Python.
* Learned about Support Vector Machines (SVMs) in Python.
* Plotting the Hyperplane and Decision Boundary.

**Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Parameters | Marks Obtained | Maximum Marks |
| 1. |  |  |  |
| 2. |  |  |  |
| 3. |  |  |  |
|  |  |  |  |