Experiment – 4

Student Name: Vivek Kumar **UID: 21BCS8129**
Branch: BE-CSE(LEET) **Section/Group: WM-20BCS-616/A**
Semester: 5th **Date of Performance: 28/09/2022**
Subject Name: Machine Learning Lab **Subject Code: 20CSP-317**

1. Aim/Overview of the practical:

Implement Support Vector Machine on any data set and analyze the accuracy with Logistic regression

2. Task to be done/ Which logistics used:

Implement SVM on any data set using sklearn.

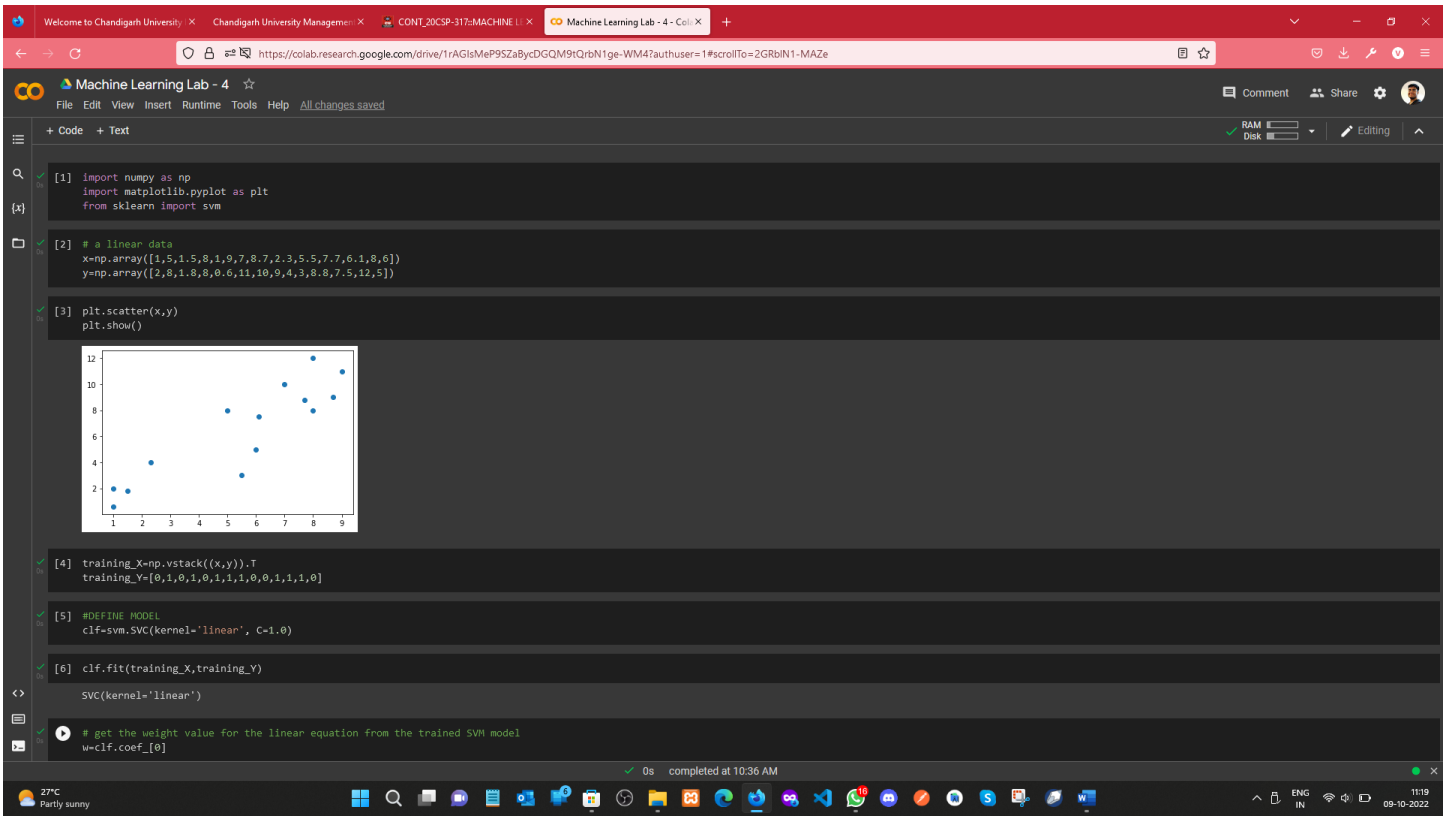
3. Steps for experiment/practical/Code:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm

# a linear data
x=np.array([1,5,1.5,8,1,9,7,8.7,2.3,5.5,7.7,6.1,8,6])
y=np.array([2,8,1.8,8,0.6,11,10,9,4,3,8.8,7.5,12,5])

plt.scatter(x,y)
plt.show()

training_X=np.vstack((x,y)).T
training_Y=[0,1,0,1,0,1,1,1,0,0,1,1,1,0]

#DEFINE MODEL
clf=svm.SVC(kernel='linear', C=1.0)

clf.fit(training_X,training_Y)

# get the weight value for the linear equation from the trained SVM model
w=clf.coef_[0]

# get the y-offset for the liner equation
a=-w[0]/w[1]

#make the x-axis space fo the data points
xx=np.linspace(0,10)

#get the y-values 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

circle_x,circle_y=datasets.make_circles(n_samples=300,noise=0.05)

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)

#training non-linear model
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()
```

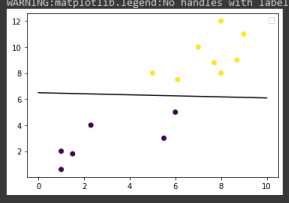
4. Result/Output/Writing Summary:



Machine Learning Lab - 4

```
[7] # get the weight value for the linear equation from the trained SVM model
    w=clf.coef_[0]

    # get the y-offset for the liner equation
    a=-w[0]/w[1]

    #make the x-axis space fo the data points
    xx=np.linspace(0,10)

    #get the y-values 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()
```

```
WARNING:matplotlib.legend:No handles with labels found to put in legend.
```

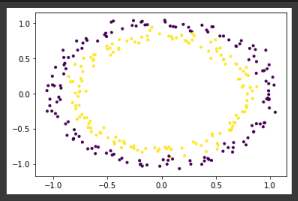


```
[8] import matplotlib.pyplot as plt
    import numpy as np
    from sklearn import datasets
    from sklearn import svm

[9] circle_x,circle_y=datasets.make_circles(n_samples=300,noise=0.05)
```

0s completed at 10:36 AM

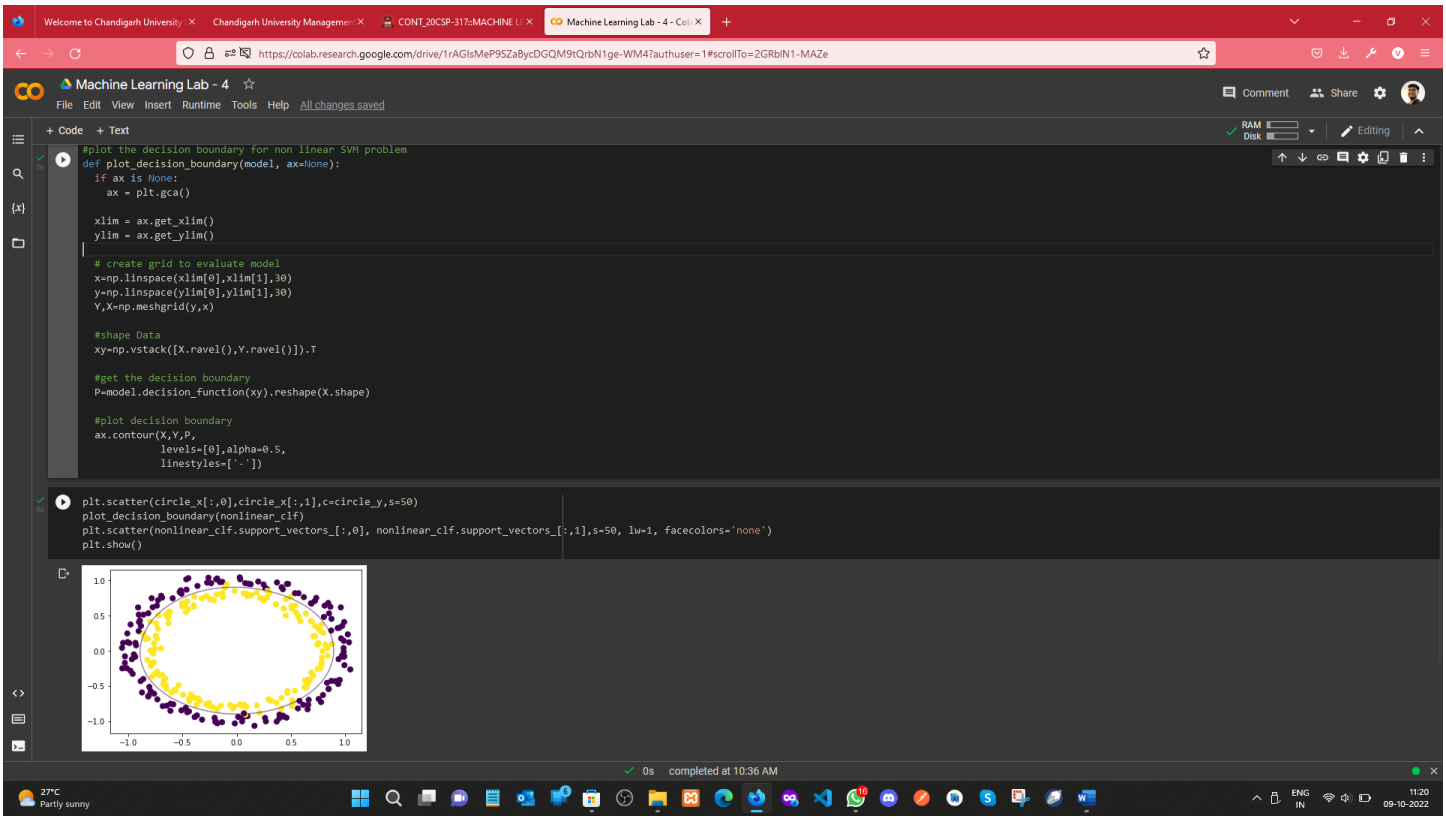
Machine Learning Lab - 4

```
[10] plt.scatter(circle_x[:,0],circle_x[:,1],c=circle_y,marker='.')
     plt.show()
```



```
[11] #make non-linear algorithm for model
     nonlinear_clf = svm.SVC(kernel='rbf',C=1.0)

[12] #training non-linear model
     nonlinear_clf.fit(circle_x,circle_y)

     SVC()

[13] #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
```

0s completed at 10:36 AM



Learning outcomes (What I have learnt):

1. Understood the concept of SVM
2. Learnt how to find the Hyperplane and Decision boundary.
3. 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.			