# Practical\_7

# Write a program for support vector machine using jupyter notebook

In [2]:

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

In [3]:

df = pd.read\_csv('data(1).csv')

df.head(10)

Out[3]:

|  | **grade1** | **grade2** | **label** |
| --- | --- | --- | --- |
| 0 | 34.623660 | 78.024693 | 0 |
| 1 | 30.286711 | 43.894998 | 0 |
| 2 | 35.847409 | 72.902198 | 0 |
| 3 | 60.182599 | 86.308552 | 1 |
| 4 | 79.032736 | 75.344376 | 1 |
| 5 | 45.083277 | 56.316372 | 0 |
| 6 | 61.106665 | 96.511426 | 1 |
| 7 | 75.024746 | 46.554014 | 1 |
| 8 | 76.098787 | 87.420570 | 1 |
| 9 | 84.432820 | 43.533393 | 1 |

In [4]:

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

x=df[["grade1","grade2"]]

y=df["label"]

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.35,random\_state=4)

In [5]:

print(x\_train.shape)

print(y\_train.shape)

x\_train.head()

(65, 2)

(65,)

Out[5]:

|  | **grade1** | **grade2** |
| --- | --- | --- |
| 78 | 60.455556 | 42.508409 |
| 18 | 70.661510 | 92.927138 |
| 88 | 78.635424 | 96.647427 |
| 7 | 75.024746 | 46.554014 |
| 39 | 34.183640 | 75.237720 |

In [6]:

y\_train.head()

Out[6]:

78 0

18 1

88 1

7 1

39 0

Name: label, dtype: int64

In [7]:

print(x\_test.shape)

print(y\_test.shape)

x\_test.head()

(35, 2)

(35,)

Out[7]:

|  | **grade1** | **grade2** |
| --- | --- | --- |
| 20 | 67.372028 | 42.838438 |
| 10 | 95.861555 | 38.225278 |
| 96 | 42.261701 | 87.103851 |
| 16 | 69.070144 | 52.740470 |
| 63 | 30.058822 | 49.592974 |

In [8]:

y\_test.head()

Out[8]:

20 0

10 0

96 1

16 1

63 0

Name: label, dtype: int64

In [10]:

from sklearn import svm

In [11]:

model = svm.SVC(gamma='scale')

In [12]:

model.fit(x\_train,y\_train)

Out[12]:

SVC(C=1.0, cache\_size=200, class\_weight=None, coef0=0.0,

decision\_function\_shape='ovr', degree=3, gamma='scale', kernel='rbf',

max\_iter=-1, probability=False, random\_state=None, shrinking=True,

tol=0.001, verbose=False)

In [14]:

score=model.score(x\_test,y\_test)

print("prediction accuracy : ",score,"%")

prediction accuracy : 0.8571428571428571 %

In [15]:

f=np.array([60.6,60.9]).reshape(1,-1)

print(f)

res=model.predict(f)

res

[[60.6 60.9]]

Out[15]:

array([0], dtype=int64)

In [16]:

yp = model.predict(x\_test)

In [17]:

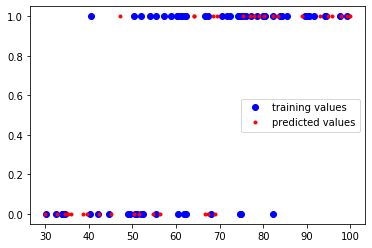
plt.plot(x\_train['grade1'],y\_train,'o',color='blue')

plt.plot(x\_test['grade1'],yp,'.',color='r')

plt.legend(['training values','predicted values'])

Out[17]:

<matplotlib.legend.Legend at 0x28c8b583a88>



**Github link:**