27/12/25
Exp No: 6

A python Rlogram to implement SVM Classifier Model

Aim :

To implement a SVM classifier model using python and determine the accuracy.

Algorithm :

Step 1:- Import Necessary Libraries

1. Import numpy as np
2. Import pandas as pd
3. Import SVM from Sklearn
4. Import matplotlib.pyplot as plt.
5. Import seaborn as sns
6. Set the front_scale attribute to 1.2 in seaborn

Step 2:- Load and Display Dataset

1. Read the dataset (muffins.csv) using pd. read_csv().
2. Display the first five instances using the head() function.

Step 3:. Plot Initial Data.

1. Use the sns.Implot()' function.
2. Set the x and y axes to "Sugar" and "Flour",

3. Assign "Type' to the hue parameter
4. Set the platte to "set 1'.
5) Set fit_reg to false
6) Plot the Graph

Step 4: Prepare Data for SVM

1. Extract "sugar" and "butter" column from the reciepes dataset and assign to variables" sugar_butter.

2. Create a new variable 'type_label'.

3. For each value in the 'Type' column assign 0 if it's "Muffin" and 1 otherwise

Step 5: Train SVM Model

1. Import the SVC model from the svm library

2. Create the svc model with kernel type set to linear

3. Fit the model using 'sugar_butter' and 'type_label' as the parameters

Step 6: Calculate support vector boundaries.

1. Assign the first support vector

Step 11: Split Dataset

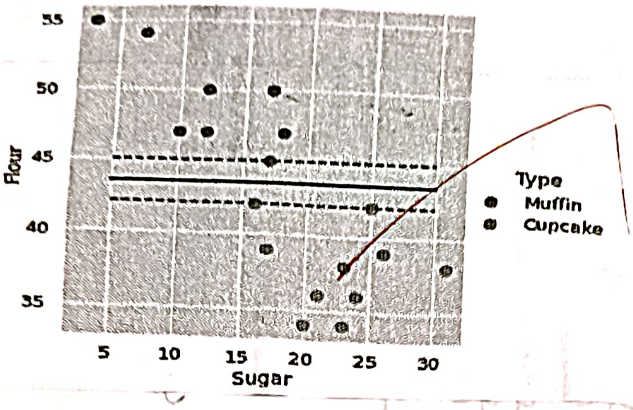
Step 12: Train new model

Step 13: Make prediction

Step 14: Evaluate Model

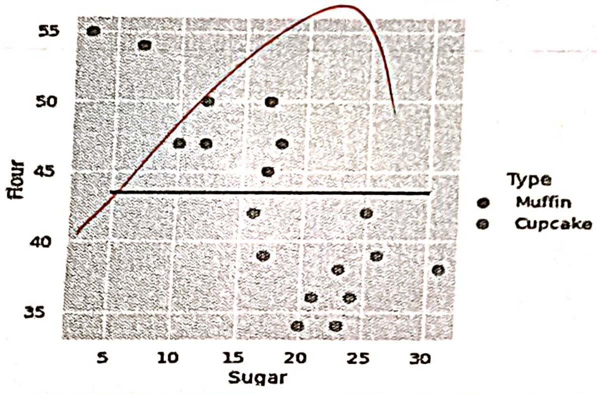
Program:

```
import numpy as np
import pandas as pd
from sklearn import svm
import matplotlib.pyplot as plt
import seaborn as sns; sns.set(font-scale=10)
recipes = pd.read.csv('.....csv').
recipes.head();
recipes: shape.
sns lmplot ('Sugar'. Flour', data = Recipes
                       hue = 'Type')
palette = 'Set1'. fit.reg = False, scatter
                       kws = {'s": 70}}
Sugar_button = recipe[["Sugar", 'Flour']].
                                      values
type-label = np.where (recipes['type']=='Muffin'.o)
model = svm.SVC(kernel = 'linear')
model.fit( sugar_button.type_label)
SVC(kernel = 'linear')
w = model_coef_[0]
hyperplane a = -w[0] / w[1]
xx = np.linspace (5, 30)
yy = a*xx - (model.intercept_[0]/w[1])
b = model.support_vector[0].
yy.down = a*xx + (b[1] - a*b[0])
b = model.support_vectors_[2]
```



Flour
55
50
45
40
35
5
10
15
20
25
30
Sugar
Type
● Muffin
● Cupcake

```
[<matplotlib.lines.Line2D at 0x7fca4a98ba50>]
```



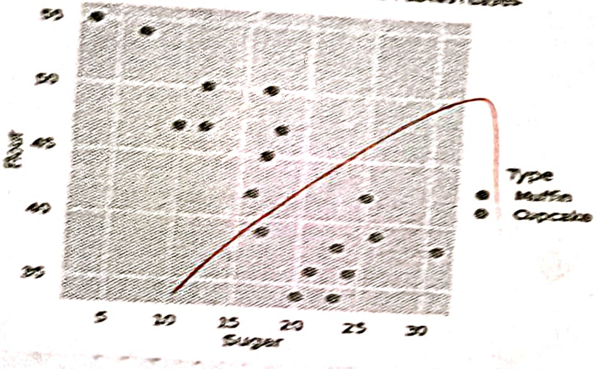
Flour
55
50
45
40
35
5
10
15
20
25
30
Sugar
Type
● Muffin
● Cupcake

```
yy_down = a*xx + b[1] - a*b[0])
sns.lmplot('Sugar','Flour', data= recipes,
hue='Type', palette='Set1', fit_reg=False,
scatter_kws ={"s":70})
plt.plot(xx, yy, linewidth=2, color=
            'black')
scatter_kws ={"s", 70})
plt.plot(xx, yy, linewidth=2, color ='black')
scatter_kws ={"s", 70})
plt.plot(xx, yy, linewidth=2, color='black')
plt.plot(xx, yy_down, 'k--')
plt.plot(xx, yy_up, 'k--')
plt.scatter(model.support_vectors_[:,0], model.
                                    support_
vectors_[:,1], s=80, facecolor='none')
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_
split
from sklearn.metrics important classification_
                                    report
x_train, x_test, y_train, y_test
model.fit(x_train, y_train)
pred = modell.predict(x_test)
print(pred)
print(confusion_matrix(y_test, pred))
print(classification_report(y_test, pred)).
```

Result:-

Thus the python program to implement svm has been executed successfully

```
<seaborn.axisgrid.FacetGrid at 0x7fcaba1f8d0>
```


Flour
55
50
45
40
35
5
10
15
20
25
30
Sugar
Type
● Muffin
● Cupcake