Ex.no: 9

Support Vector Machines

AIM:

Implement Support Vector Machines using dataset. Displaying it.

Algorithm:

Step 1:- Importing the dataset

Step 2: Splitting the dataset into training and test samples.

Step 3: Classifying the predictors and target

Step 4: Initializing Support Vector Mechines and fitting the training Data.

Step 5: Predicting the classes for test set.

Step 6: Attaching the predictions to test set for Comparing.

Step 7: Display the Accuracy of SVM of the Dataset and printing the Accuracy Metrics and Confusion Metrix.

Step 8: Visualizing the Dataset with the color of red and Orange for a given Dataset.

Program:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import svc.
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing import LabelEncoder
import numpy as np
```

```
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

pd.options.mode.chained_assignment = None

data = pd.read_csv("apples_and_oranges.csv")
print(data.head())
training_set, test_set = train_test_split(data, test_size = 0.2, random_state=1)
X_train = training_set.iloc[:, 0:2].values
y_train = training_set.iloc[:, 2].values
x_test = test_set.iloc[:, 0:2].values
y_test = test_set.iloc[:, 2].values

classifier = SVC(kernel ='rbf', random_state =5)
classifier.fit(X_train, y_train)

y_pred = classifier.predict(x_test)
test_set["Predictions"] = y_pred

cm = confusion_matrix(y_test, y_pred)
accuracy = float(cm.diagonal().sum())/len(y_test)
print("\nAccuracy of SVM for the Given Dataset :", accuracy)

le = LabelEncoder()
y_train = le.fit_transform(y_train)

classifier = SVC(kernel ='rbf', random_state =1)
classifier.fit(x_train, y_train)
print('Accuracy Metrics')
print('classification_report(y_test, y_pred)
```

```
print ('Confusion matrix')
print (confusion_matrix (y_test, y_pred))

print(
  plt.figure(figsize =(7,7))
  X_set, y_set =X_train, y_train)
X1, X2 = np.meshgrid(np.arrange(start = X_set[:,0].min() -1, stop = X_set[:,0]
  max()+1, step =0.01), np.arrange(start = X_set[:,1].min() -1, stop = X_set
plt.xlim (X1.min(), X1.max())
plt.ylim (X2.min(), X2.max())
for i, j in enumerate (np.unique (y_set)):
    plt.scatter (X_set[y_set == j, 0], X_set[y_set == j], c = ListedColormap
                                                    (('red', 'orange'))(i), label = j)
plt.title('Apple Vs Orange')
plt.xlabel('Weight In Grams')
plt.ylabel('Size in cm')
plt.legend()
plt.show()
```

Result:

Thus the Implementation of Support Vector Machines using Dataset is Verified and Succesfully executed.

ML/ exp9 - Jupyter Noteboc

localhost:8888/notebooks/ML/exp9.ipynb

jupyter **exp9** Last Checkpoint: 21 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

In [10]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix,classification_report
from sklearn.preprocessing import LabelEncoder
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
```

In [11]:
```
pd.options.mode.chained_assignment = None
```

In [12]:
```
data = pd.read_csv(r"C:\Users\Smart\anaconda3\dataset\apples_and_oranges.csv")
```

In [13]:
```
print(data.head())
training_set, test_set = train_test_split(data, test_size = 0.2, random_state = 1)
```

```
   Weight  Size   Class
0      69  4.39  orange
1      69  4.21  orange
2      65  4.09  orange
3      72  5.85   apple
4      67  4.70  orange
```

In [14]:
```
X_train = training_set.iloc[:,0:2].values
```

Type here to search

ENG IN 9:14 PM 9/18/2021

ML/ exp9 - Jupyter Noteboc

localhost:8888/notebooks/ML/exp9.ipynb

jupyter exp9 Last Checkpoint: 22 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

In [14]:
```
X_train = training_set.iloc[:,0:2].values
Y_train = training_set.iloc[:,2].values
X_test = test_set.iloc[:,0:2].values
Y_test = test_set.iloc[:,2].values
```

In [15]:
```
classifier = SVC(kernel='rbf', random_state = 5)
classifier.fit(X_train,Y_train)
```

Out[15]:
```
SVC(random_state=5)
```

In [16]:
```
Y_pred = classifier.predict(X_test)
test_set["Predictions"] = Y_pred
```

In [17]:
```
cm = confusion_matrix(Y_test,Y_pred)
accuracy = float(cm.diagonal().sum())/len(Y_test)
print("\nAccuracy Of SVM For The Given Dataset : ", accuracy)
```

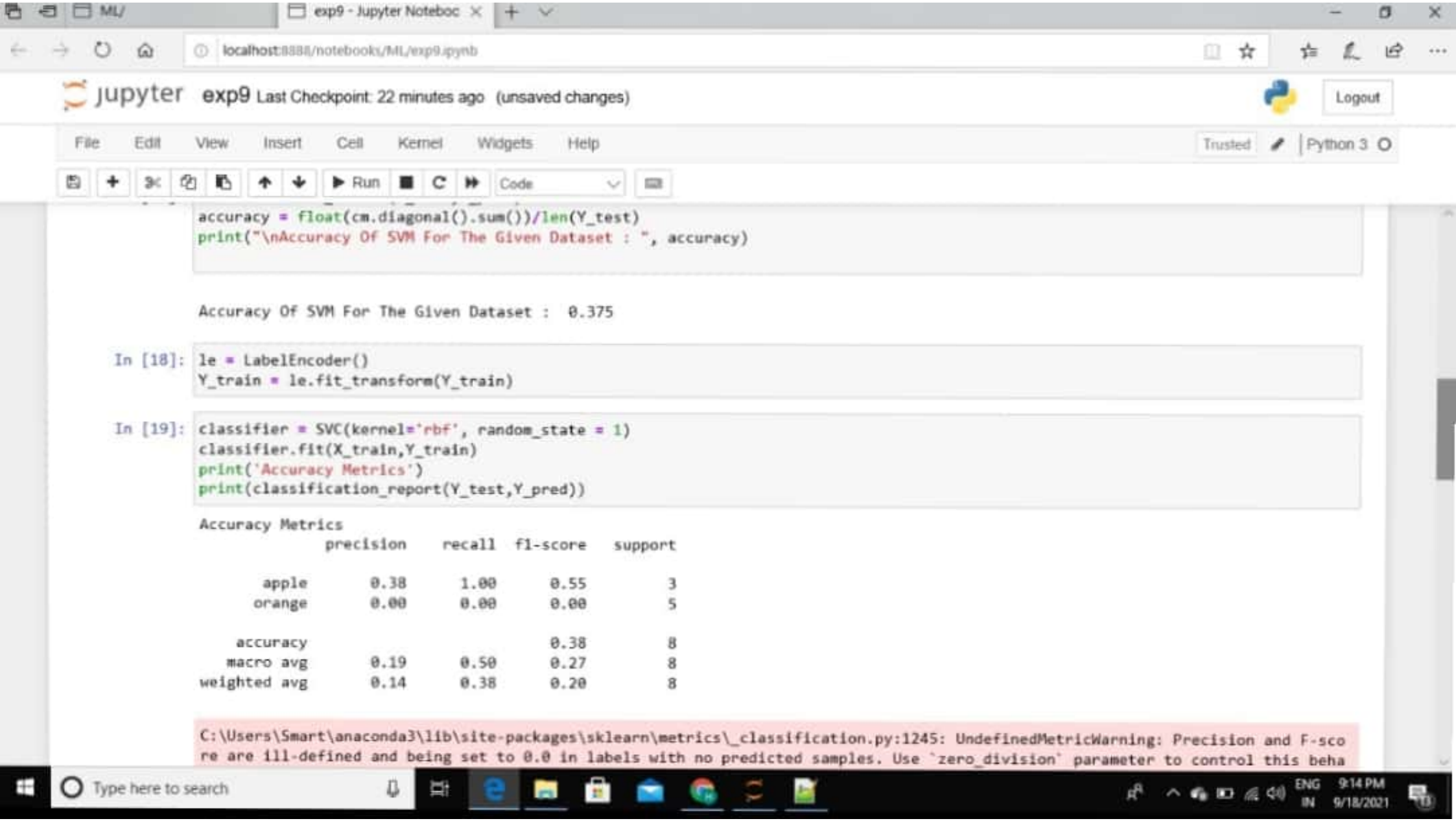
```
Accuracy Of SVM For The Given Dataset :  0.375
```

In [18]:
```
le = LabelEncoder()
Y_train = le.fit_transform(Y_train)
```

In [19]:
```
classifier = SVC(kernel='rbf', random_state = 1)
```

Type here to search

ENG IN 9:14 PM 9/18/2021



ML/ exp9 - Jupyter Noteboc

localhost:8888/notebooks/ML/exp9.ipynb

jupyter exp9 Last Checkpoint: 22 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

In [20]:
```
print('Confusion Matrix')
print(confusion_matrix(Y_test,Y_pred))
```

```
Confusion Matrix
[[3 0]
 [5 0]]
```

In [21]:
```
plt.figure(figsize = (7,7))
X_set, y_set = X_train, Y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01), np.arange(start = X_set
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape), alpha = 0.75, cmap = ListedColor
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1], c = ListedColormap(('red', 'orange'))(i), label = j)
plt.title('Apples Vs Oranges')
plt.xlabel('Weight In Grams')
plt.ylabel('Size in cm')
plt.legend()
plt.show()
```

```
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in
case its length matches with *x* & *y*.  Please use the *color* keyword-argument or provide a 2-D array with a single row if yo
u intend to specify the same RGB or RGBA value for all points.
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in
case its length matches with *x* & *y*.  Please use the *color* keyword-argument or provide a 2-D array with a single row if yo
```

Type here to search

ENG IN 9:14 PM 9/18/2021

ML/ exp9 - Jupyter Noteboc

localhost:8888/notebooks/ML/exp9.ipynb

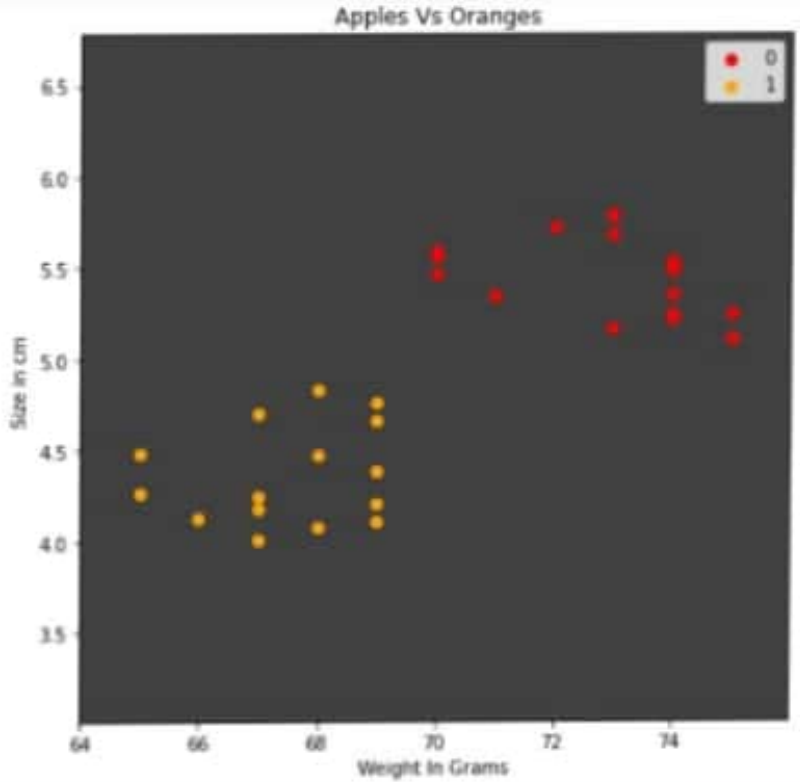
jupyter exp9 Last Checkpoint: 22 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code



In [ ]:

Type here to search

ENG IN 9:14 PM 9/18/2021