AIM: IMPLEMENTATION OF CLASSIFYING DATA USING SUPPORT VECTOR MACHINES (SVMS).

THEORY:

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well . Support Vectors are simply the co-ordinates of individual observation. The SVM classifier is a frontier which best segregates the two classes (hyperplane/ line) In the SVM classifier, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, the SVM algorithm has a technique called the kernel trick. The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts not separable problem to separable problem. It is mostly useful in nonlinear separation problem. Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you've defined. In Python, scikit-learn is a widely used library for implementing machine learning algorithms. SVM is also available in the scikit-learn library and we follow the same structure for using it(Import library, object creation, fitting model and prediction)

Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

SOURCE CODE:

1) **IMPORTING LIBRARIES:**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

2) **READING DATASET:**

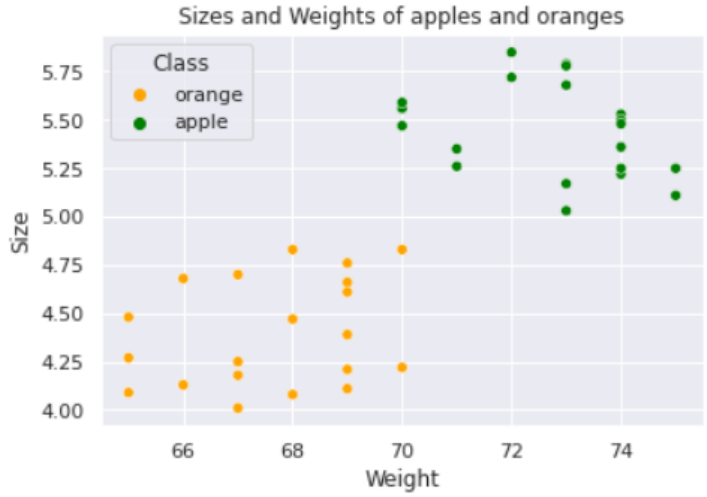
```
data=pd.read_csv('apples_and_oranges.csv')
data.head(5)
```

	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

3) PLOTTING GRAPH:

```
# create a dictionary to colour classes
color_dict = dict({'orange':'orange',
                   'apple':'green'})
# scatterplot
plt.title('Sizes and Weights of apples and oranges')
sns.scatterplot(data=data, x="Weight", y="Size", hue="Class", palette = color_dict)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa1132502d0>
```



4) CREATING SVM LINEAR MODEL:

```
# define input data
X = data[["Weight", "Size"]]
# define target
y = data.Class
# fitting the support vector machine using a linear kernel
from sklearn import svm
clf = svm.SVC(kernel = 'linear', C=10)
clf.fit(X, y)
```

```
SVC(C=10, kernel='linear')
```

5) OBTAINING HYPERPLANE:

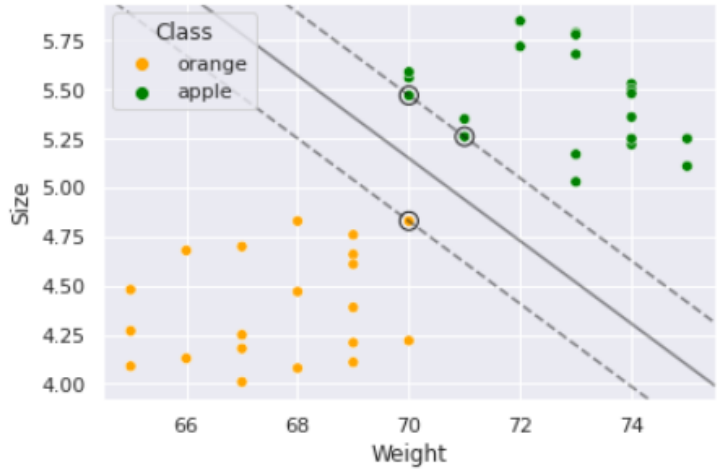
```
#Obtain hyperplane
b = clf.intercept_
w_1 = clf.coef_[0][0]
w_2 = clf.coef_[0][1]
b, w_1, w_2
```

```
(array([62.09725159]), -0.6575905440881797, -3.1196742877366077)
```

6) PLOTTING HYPERPLANE AND SVM:

```
# plotting the hyperplane and support vector lines
ax = plt.gca()
sns.scatterplot(data=data, x="Weight", y="Size", hue="Class", palette = color_dict)
xlim = ax.get_xlim()
ylim = ax.get_ylim()
xx = np.linspace(xlim[0], xlim[1], 30)
yy = np.linspace(ylim[0], ylim[1], 30)
YY, XX = np.meshgrid(yy, xx)
xy = np.vstack([XX.ravel(), YY.ravel()]).T
Z = clf.decision_function(xy).reshape(XX.shape)
ax.contour(XX, YY, Z, colors='k', levels=[-1, 0, 1], alpha=0.5,
           linestyles=['--', '-', '--'])
ax.scatter(clf.support_vectors_[:, 0], clf.support_vectors_[:, 1], s=100,
           linewidth=1, facecolors='none', edgecolors='k')
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not h
  "X does not have valid feature names, but"
```



7) OBTAINING SUPPORT VECTORS:

```
# obtain support vectors
clf.support_vectors_
```

```
array([[70.  ,  5.47],
       [71.  ,  5.26],
       [70.  ,  4.83]])
```

8) PREDICTING VALUES:

```
clf.predict([[70, 4.6]])
```

```
/usr/local/lib/python3.7/dist-pac
  "X does not have valid feature
array(['orange'], dtype=object)
```

```
clf.predict([[74, 5.3]])
```

```
/usr/local/lib/python3.7/dist-pac
  "X does not have valid feature
array(['apple'], dtype=object)
```

```
clf.predict([[62,3.0]])
```

```
/usr/local/lib/python3.7/dist-pac
  "X does not have valid feature
array(['orange'], dtype=object)
```

CONCLUSION:

From this practical, I have learned and implemented Linear Support Vector Machine(SVM) in python.