In [1]:
```
from sklearn.datasets import load_iris
```

In [2]:
```
iris=load_iris()
```

In [3]:
```
X=iris.data
y=iris.target
```

In [4]:
```
from sklearn.svm import SVC
```

In [5]:
```
import pandas as pd
df=pd.read_csv("f:/dataset/classification/fruits.csv")
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
```

In [6]:
```
#with linear kernel and binary classification
model=SVC(kernel='linear')
model.fit(X,y)
```

Out[6]:
```
▾ SVC
SVC(kernel='linear')
```

In [7]:
```
df
```

Out[7]:

	diameter	weight	FruitName
0	3.0	30	Banana
1	6.0	100	Apple
2	6.1	95	Apple
3	3.2	35	Banana
4	5.5	80	Apple
5	7.1	120	Banana
6	2.5	60	Banana
7	2.3	100	Banana
8	4.8	70	Apple
9	4.8	79	Apple
10	5.8	120	Apple
11	2.6	85	Banana
12	6.0	110	Apple
13	6.3	95	Apple
14	3.0	40	Banana
15	3.5	25	Banana
16	5.5	100	Apple
17	7.5	120	Apple
18	2.5	50	Banana
19	2.7	40	Banana

20	4.8	90	Apple
21	5.8	90	Apple

In [8]:
```
model.coef_
```

Out[8]:
```
array([[-0.87956593, -0.01319065]])
```

In [9]:
```
model.intercept_
```

Out[9]:
```
array([4.40849056])
```

In [12]:
```
sample=[[3,70],[5,90]]
model.decision_function(sample)
```

Out[12]:
```
array([ 0.84644742, -1.1764974 ])
```

In [11]:
```
model.predict([sample])
```

Out[11]:
```
array(['Banana'], dtype=object)
```

In [14]:
```
df.FruitName=df.FruitName.map({'Apple':-1,'Banana':1})
```

In [15]:
```
df
```

Out[15]:

	diameter	weight	FruitName
0	3.0	30	1
1	6.0	100	-1
2	6.1	95	-1
3	3.2	35	1
4	5.5	80	-1
5	7.1	120	1
6	2.5	60	1
7	2.3	100	1
8	4.8	70	-1
9	4.8	79	-1
10	5.8	120	-1
11	2.6	85	1
12	6.0	110	-1
13	6.3	95	-1
14	3.0	40	1
15	3.5	25	1
16	5.5	100	-1
17	7.5	120	-1
18	2.5	50	1
19	2.7	40	1

20	4.8	90	-1
21	5.8	90	-1

In [16]:
```
model.coef_
```

Out[16]:
```
array([[-0.87956593, -0.01319065]])
```

In [17]:
```
model.intercept_
```

Out[17]:
```
array([4.40849056])
```

In [18]:
```
sample=[[3,70],[5,90]]
d=-0.87956593*3+-0.01319065*70+4.40849056
print(d)
```

```
0.8464472699999996
```

In [19]:
```
d=-0.87956593*5+-0.01319065*90+4.40849056
print(d)
```

```
-1.1764975900000003
```

In [35]:
```
gm=.1
model=SVC(kernel='rbf',gamma=gm)
model.fit(X,y)
```

Out[35]:
```
▾ SVC
SVC(gamma=0.1)
```

In [21]:
```
model.coef_
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[21], line 1
----> 1 model.coef_

File ~\anaconda3\Lib\site-packages\sklearn\svm\_base.py:658, in BaseLibSVM.coef_(self)
    651 """Weights assigned to the features when `kernel="linear"`.
    652 
    653 Returns
    654 -------
    655 ndarray of shape (n_features, n_classes)
    656 """
    657 if self.kernel != "linear":
--> 658     raise AttributeError("coef_ is only available when using a linear kernel")
    660 coef = self._get_coef()
    662 # coef_ being a read-only property, it's better to mark the value as
    663 # immutable to avoid hiding potential bugs for the unsuspecting user.

AttributeError: coef_ is only available when using a linear kernel
```

In [ ]:
```
sample=[[3,70],[5,90]]
```

In [36]:
```
model.n_support_
```

Out[36]:
```
array([11, 10])
```

In [37]:
```
model.support_vectors_
```

Out[37]:
```
array([[  6. , 100. ],
           [  5.5,  80. ],
           [  4.8,  70. ],
           [  4.8,  79. ],
           [  5.8, 120. ],
           [  6. , 110. ],
           [  6.3,  95. ],
           [  5.5, 100. ],
           [  7.5, 120. ],
           [  4.8,  90. ],
           [  5.8,  90. ],
           [  3. ,  30. ],
           [  3.2,  35. ],
           [  7.1, 120. ],
           [  2.5,  60. ],
           [  2.3, 100. ],
           [  2.6,  85. ],
           [  3. ,  40. ],
           [  3.5,  25. ],
           [  2.5,  50. ],
           [  2.7,  40. ]])
```

In [43]:

```
import numpy as np
sample=[3,70]
dc=model.dual_coef_
sqr_eucl_dist=np.square(model.support_vectors_-sample).sum(axis=1)
#print(sqr_eucl_dist)
kernal=np.exp(-gm*sqr_eucl_dist)
#print(kernal)
kernal_dc=dc*kernal
d=kernal_dc.sum()+model.intercept_
print(d)
```

```
[-0.68220774]
```

In [44]:

```
X=iris.data
y=iris.target
```

In [45]:

```
model=SVC(kernel='linear',decision_function_shape='ovr')
model.fit(X,y)
```

Out[45]:

```
▾      SVC
SVC(kernel='linear')
```

In [46]:

```
model.coef_
```

Out[46]:

```
array([[-0.04625854,  0.5211828 , -1.00304462, -0.46412978],
       [-0.00722313,  0.17894121, -0.53836459, -0.29239263],
       [ 0.59549776,  0.9739003 , -2.03099958, -2.00630267]])
```

In [47]:

```
model.intercept_
```

Out[47]:

```
array([1.4528445 , 1.50771313, 6.78097119])
```

In [48]:

```
sample=[3.6,1.3,5.7,.7]
model.decision_function([sample])
```

Out[48]:

```
array([[-0.28311297,  1.18765319,  2.27101983]])
```

In [49]:

```
model.predict([sample])
```

Out[49]:

```
array([2])
```

In [ ]: