LAB2

August 25, 2019

[46]:
```
from sklearn import preprocessing
from sklearn.naive_bayes import GaussianNB
import numpy as np
```

[2]:
```
weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny','Rainy'
temp=['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','
play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']
```

[4]:
```
le = preprocessing.LabelEncoder()
weather_encoded=le.fit_transform(weather)
print(weather_encoded)
temp_encoded=le.fit_transform(temp)
label=le.fit_transform(play)
print("Temp:",temp_encoded)
print("Play:",label)
```

```
[2 2 0 1 1 1 0 2 2 1 2 0 0 1]
Temp: [1 1 1 2 0 0 0 2 0 2 2 2 1 2]
Play: [0 0 1 1 1 0 1 0 1 1 1 1 1 0]
```

[5]:
```
features=tuple(zip(weather_encoded,temp_encoded))
print(features)
```

```
((2, 1), (2, 1), (0, 1), (1, 2), (1, 0), (1, 0), (0, 0), (2, 2), (2, 0), (1, 2),
(2, 2), (0, 2), (0, 1), (1, 2))
```

[6]:
```
model = GaussianNB()
model.fit(features,label)
```

[6]:
```
GaussianNB(priors=None, var_smoothing=1e-09)
```

[7]:
```
predicted= model.predict([[2,2]])
print("Predicted Value:", predicted)
```

```
Predicted Value: [0]
```

[8]:
```
outlook = ['Rainy', 'Rainy', 'Overcast', 'Sunny', 'Sunny', 'Sunny', 'Overcast',␣
 ↪'Rainy', 'Rainy', 'Sunny', 'Rainy','Overcast', 'Overcast', 'Sunny']
temperature = ['Hot', 'Hot', 'Hot', 'Mild', 'Cool', 'Cool', 'Cool', 'Mild',␣
 ↪'Cool', 'Mild', 'Mild', 'Mild', 'Hot', 'Mild']
humidity = ['High', 'High', 'High', 'High', 'Normal', 'Normal', 'Normal',␣
 ↪'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'High']
wind = ['False', 'True', 'False', 'False', 'False', 'True', 'True', 'False',␣
 ↪'False', 'False', 'True', 'True', 'False', 'True']
play = [ 'No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes',␣
 ↪'Yes', 'Yes', 'Yes', 'No']
```

[16]:
```
le = preprocessing.LabelEncoder()
outlook_enc = le.fit_transform(outlook)
print("Outlo:",outlook_enc)
temperature_enc = le.fit_transform(temperature)
print("Temp:",temperature_enc)
humidity_enc = le.fit_transform(humidity)
print("Humidity",humidity_enc)
wind_enc = le.fit_transform(wind)
print("Wind",wind_enc)
label=le.fit_transform(play)
print("Play",label)
```

```
Outlo: [1 1 0 2 2 2 0 1 1 2 1 0 0 2]
Temp: [1 1 1 2 0 0 0 2 0 2 2 2 1 2]
Humidity [0 0 0 0 1 1 1 0 1 1 1 0 1 0]
Wind [0 1 0 0 0 1 1 0 0 0 1 1 0 1]
Play [0 0 1 1 1 0 1 0 1 1 1 1 1 0]
```

[17]:
```
features = tuple(zip(outlook_enc,temperature_enc,humidity_enc,wind_enc))
print(features)
```

```
((1, 1, 0, 0), (1, 1, 0, 1), (0, 1, 0, 0), (2, 2, 0, 0), (2, 0, 1, 0), (2, 0, 1,
1), (0, 0, 1, 1), (1, 2, 0, 0), (1, 0, 1, 0), (2, 2, 1, 0), (1, 2, 1, 1), (0, 2,
0, 1), (0, 1, 1, 0), (2, 2, 0, 1))
```

[18]:
```
model = GaussianNB()
model.fit(features,label)
```

[18]:
```
GaussianNB(priors=None, var_smoothing=1e-09)
```

[19]:
```
'''
1.3.2
(1)
    Outlook: Rainy
    Temperature: Mild
    Humidity: Normal
    Wind: False
```

```
'''
predicted = model.predict([[1,2,1,0]])
predicted
```

[19]: array([1])

[21]:
```
'''
1.3.2
(2)
    Outlook: Sunny
    Temperature: Cool
    Humidity: High
    Wind: True
'''
predicted = model.predict([[2,0,0,1]])
predicted
```

[21]: array([0])

[30]:
```
'''
Exercise 1.4 Precedure (Wine Dataset)
'''
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_wine
from sklearn import metrics
wine = load_wine()
```

[25]:
```
print("Features: ", wine.feature_names)
print("Labels: ", wine.target_names)
wine.data.shape
```

```
Features:  ['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium',
'total_phenols', 'flavanoids', 'nonflavanoid_phenols', 'proanthocyanins',
'color_intensity', 'hue', 'od280/od315_of_diluted_wines', 'proline']
Labels:  ['class_0' 'class_1' 'class_2']
```

[25]: (178, 13)

[26]:
```
data_train, data_test, target_train, target_test = train_test_split(wine.
↪data,wine.target, test_size = 0.30, random_state = 10)
```

[27]:
```
gnb = GaussianNB()
gnb.fit(data_train, target_train)
target_pred = gnb.predict(data_test)
```

[29]:
```
print("Accuracy:",metrics.accuracy_score(target_test, target_pred))
```

```
Accuracy: 0.8888888888888888
```

[31]:
```
confusion_matrix(target_test, target_pred)
```

[31]:
```
array([[14,  1,  0],
       [ 2, 22,  3],
       [ 0,  0, 12]])
```

[32]:
```
'''
Exercise 1.5
(1) Three categories
'''

from sklearn.datasets import load_iris
iris = load_iris()
```

[34]:
```
print("Featurea: ",iris.feature_names)
print("Lables :",iris.target_names)
iris.data.shape
```

```
Featurea:  ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal
width (cm)']
Lables : ['setosa' 'versicolor' 'virginica']
```

[34]:
```
(150, 4)
```

[35]:
```
data_train, data_test, target_train, target_test = train_test_split(iris.
↪data,iris.target, test_size = 0.30, random_state = 10)
```

[36]:
```
gnb = GaussianNB()
gnb.fit(data_train, target_train)
target_pred = gnb.predict(data_test)
```

[37]:
```
print("Accuracy:",metrics.accuracy_score(target_test, target_pred))
```

```
Accuracy: 1.0
```

[38]:
```
confusion_matrix(target_test, target_pred)
```

[38]:
```
array([[14,  0,  0],
       [ 0, 17,  0],
       [ 0,  0, 14]])
```

[69]:
```
'''
(2) Two categories
'''

t1 = iris.target
t = t1[t1>0]
d = iris.data[t1>0]
```

[71]:
```
data_train, data_test, target_train, target_test = train_test_split(d,t,␣
↪test_size = 0.30, random_state = 10)
```

[72]:
```
gnb = GaussianNB()
gnb.fit(data_train, target_train)
target_pred = gnb.predict(data_test)
```

[73]:
```
print("Accuracy:",metrics.accuracy_score(target_test, target_pred))
```

```
Accuracy: 0.9666666666666667
```

[74]:
```
confusion_matrix(target_test, target_pred)
```

[74]:
```
array([[13,  1],
       [ 0, 16]])
```

[ ]:
```

```