In [1]:
```
import numpy as np
```

In [2]:
```
X = np.array([["Red", "Sports", "Domestic"],
              ["Red", "Sports", "Domestic"],
              ["Red", "Sports", "Domestic"],
              ["Yellow", "Sports", "Domestic"],
              ["Yellow", "Sports", "Imported"],
              ["Yellow", "SUV", "Imported"],
              ["Yellow", "SUV", "Imported"],
              ["Yellow", "SUV", "Domestic"],
              ["Red", "SUV", "Imported"],
              ["Red", "Sports", "Imported"]])
Y = np.array(["Yes", "No", "Yes", "No", "Yes", "No", "Yes", "No", "No", "Yes"])
```

In [3]:
```
rows = int(X.size/3)
print(rows)

for i in range(rows):
    for j in range(3):
        if X[i][j] == "Red":
            X[i][j] = 0
        if X[i][j] == "Sports":
            X[i][j] = 0
        if X[i][j] == "Domestic":
             X[i][j] = 0
        if X[i][j] == "Yellow":
            X[i][j] = 1
        if X[i][j] == "SUV":
            X[i][j] = 1
        if X[i][j] == "Imported":
            X[i][j] = 1
X = X.astype(np.float)
X
```

```
10
```

Out[3]:
```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.],
       [1., 0., 0.],
       [1., 0., 1.],
       [1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 0.],
       [0., 1., 1.],
       [0., 0., 1.]])
```

In [4]:
```
type(X)
```

Out[4]:
```
numpy.ndarray
```

In [5]:

```
for i in range(Y.size):
    if Y[i] == "Yes":
        Y[i] = int(1)
    else:
        Y[i] = int(0)

Y = Y.astype(np.float)
Y
```

Out[5]:

```
array([1., 0., 1., 0., 1., 0., 1., 0., 0., 1.])
```

In [6]:

```
#Create a Gaussian Classifier
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()

# Train the model using the training sets
model.fit(X, Y)

#Predict Output
predicted= model.predict([[0, 1, 0]]) # Red, SUV, Domestic
predicted
```

Out[6]:

```
array([0.])
```

In [7]:

```
# Sử dụng BernoulliNB
from sklearn.naive_bayes import BernoulliNB
clf = BernoulliNB()
clf.fit(X, Y)
```

Out[7]:

```
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
```

In [8]:

```
#Predict Output
predicted= clf.predict([[0, 1, 0]]) # Red, SUV, Domestic
predicted
```

Out[8]:

```
array([0.])
```