In [5]:

```
import pandas as pd
data=pd.read_csv('/content/golfboll.csv')
print(data)

y=list(map(lambda v: 1 if v=='Yes'  else 0 ,data['PlayGolf'].values ))
X = data[['Outlook', 'Temperature', 'Humidity', 'Windy','PlayGolf']].values
print(X)
y_train = y[:10]
y_val = y[10:]

X_train = X[:10]
X_val = X[10:]

class NaiveBayesClassifier:

    def __init__(self, X, y):
        self.X, self.y = X, y
        self.N = len(self.X) # Length of the training set
        self.dim = len(self.X[0]) # Dimension of the vector of features
        self.attrs = [[] for _ in range(self.dim)] # Here we'll store the columns of the
training set
        self.output_dom = {} # Output classes with the number of ocurrences in the train
ing set. In this case we have only 2 classes
        self.data = [] # To store every row [Xi, yi]


        for i in range(len(self.X)):
            for j in range(self.dim):
                # if we have never seen this value for this attr before,
                # then we add it to the attrs array in the corresponding position
                if not self.X[i][j] in self.attrs[j]:
                    self.attrs[j].append(self.X[i][j])

            # if we have never seen this output class before,
            # then we add it to the output_dom and count one occurrence for now
            if not self.y[i] in self.output_dom.keys():
                self.output_dom[self.y[i]] = 1
            # otherwise, we increment the occurrence of this output in the training set b
y 1
            else:
                self.output_dom[self.y[i]] += 1
            self.data.append([self.X[i], self.y[i]])
        print(self.data)



    def classify(self, entry):

        solve = None # Final result
        max_arg = -1 # partial maximum

        for y in self.output_dom.keys():

            prob = self.output_dom[y]/self.N # P(y)

            for i in range(self.dim):
                cases = [x for x in self.data if x[0][i] == entry[i] and x[1] == y] # a
ll rows with Xi = xi
                n = len(cases)
                prob *= n/self.N # P *= P(Xi = xi)

            # if we have a greater prob for this output than the partial maximum...
            if prob > max_arg:
                max_arg = prob
                solve = y

        return solve
```

```
nbc = NaiveBayesClassifier(X_train, y_train)
total_cases = len(y_val)
good = 0
bad = 0
for i in range(total_cases):
    predict = nbc.classify(X_val[i])
    if y_val[i] == predict:
        good += 1
    else:
        bad += 1

print('TOTAL EXAMPLES:', total_cases)
print('RIGHT:', good)
print('WRONG:', bad)
print('ACCURACY:', good/total_cases)
```

```
    index   Outlook Temperature Humidity  Windy PlayGolf
0       0     Rainy         Hot     High  False       No
1       1     Rainy         Hot     High   True       No
2       2  Overcast         Hot     High  False      Yes
3       3     Sunny        Mild     High  False      Yes
4       4     Sunny        Cool   Normal  False      Yes
5       5     Sunny        Cool   Normal   True       No
6       6  Overcast        Cool   Normal   True      Yes
7       7     Rainy        Mild     High  False       No
8       8     Rainy        Cool   Normal  False      Yes
9       9     Sunny        Mild   Normal  False      Yes
10     10     Rainy        Mild   Normal   True      Yes
11     11  Overcast        Mild     High   True      Yes
12     12  Overcast         Hot   Normal  False      Yes
13     13     Sunny        Mild     High   True       No
[['Rainy' 'Hot' 'High' False 'No']
 ['Rainy' 'Hot' 'High' True 'No']
 ['Overcast' 'Hot' 'High' False 'Yes']
 ['Sunny' 'Mild' 'High' False 'Yes']
 ['Sunny' 'Cool' 'Normal' False 'Yes']
 ['Sunny' 'Cool' 'Normal' True 'No']
 ['Overcast' 'Cool' 'Normal' True 'Yes']
 ['Rainy' 'Mild' 'High' False 'No']
 ['Rainy' 'Cool' 'Normal' False 'Yes']
 ['Sunny' 'Mild' 'Normal' False 'Yes']
 ['Rainy' 'Mild' 'Normal' True 'Yes']
 ['Overcast' 'Mild' 'High' True 'Yes']
 ['Overcast' 'Hot' 'Normal' False 'Yes']
 ['Sunny' 'Mild' 'High' True 'No']]
[[array(['Rainy', 'Hot', 'High', False, 'No'], dtype=object), 0], [array(['Rainy', 'Hot',
'High', True, 'No'], dtype=object), 0], [array(['Overcast', 'Hot', 'High', False, 'Yes'],
dtype=object), 1], [array(['Sunny', 'Mild', 'High', False, 'Yes'], dtype=object), 1], [ar
ray(['Sunny', 'Cool', 'Normal', False, 'Yes'], dtype=object), 1], [array(['Sunny', 'Cool'
, 'Normal', True, 'No'], dtype=object), 0], [array(['Overcast', 'Cool', 'Normal', True, '
Yes'], dtype=object), 1], [array(['Rainy', 'Mild', 'High', False, 'No'], dtype=object), 0
], [array(['Rainy', 'Cool', 'Normal', False, 'Yes'], dtype=object), 1], [array(['Sunny',
'Mild', 'Normal', False, 'Yes'], dtype=object), 1]]
TOTAL EXAMPLES: 4
RIGHT: 4
WRONG: 0
ACCURACY: 1.0
```