**Navie Bayes Algorithm**

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes’ Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

**Defining Dataset**

In this example, you can use the dummy dataset with three columns: weather, temperature, and play. The first two are features (weather, temperature) and the other is the label.

# Assigning features and label variables

weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny','Rainy','Sunny','Overcast','Overcast','Rainy']

temp=['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mild']

play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']

#### Encoding Features

First, we need to convert these string labels into numbers. For example: 'Overcast', 'Rainy', 'Sunny' as 0, 1, 2. This is known as label encoding. Scikit-learn provides Label Encoder library for encoding labels with a value between 0 and one less than the number of discrete classes.

# Import LabelEncoder

from sklearn import preprocessing

#creating labelEncoder

le = preprocessing.LabelEncoder()

# Converting string labels into numbers.

wheather\_encoded=le.fit\_transform(weather)

print wheather\_encoded

Similarly, you can also encode temp and play columns.

# Converting string labels into numbers

temp\_encoded=le.fit\_transform(temp)

label=le.fit\_transform(play)

print "Temp:",temp\_encoded

print "Play:",label

Now combine both the features (weather and temp) in a single variable (list of tuples).

#Combinig weather and temp into single listof tuples

features=zip(weather\_encoded,temp\_encoded)

print features

**Generating Model**

Generate a model using naive bayes classifier in the following steps:

* Create naive bayes classifier
* Fit the dataset on classifier
* Perform prediction

#Import Gaussian Naive Bayes model

from sklearn.naive\_bayes import GaussianNB

#Create a Gaussian Classifier

model = GaussianNB()

# Train the model using the training sets

model.fit(features,label)

#Predict Output

predicted= model.predict([[0,2]]) # 0:Overcast, 2:Mild

print "Predicted Value:", predicted

Here, 1 indicates that players can 'play'.