**Naive Bayes Classification**

Learn how to build and evaluate a Naive Bayes Classifier using Python's Scikit-learn package.

Suppose you are a product manager, you want to classify customer reviews in positive and negative classes. Or As a loan manager, you want to identify which loan applicants are safe or risky? As a healthcare analyst, you want to predict which patients can suffer from diabetes disease. All the examples have the same kind of problem to classify reviews, loan applicants, and patients.

Naive Bayes is the most straightforward and fast classification algorithm, which is suitable for a large chunk of data. Naive Bayes classifier is successfully used in various applications such as spam filtering, text classification, sentiment analysis, and recommender systems. It uses Bayes theorem of probability for prediction of unknown class.

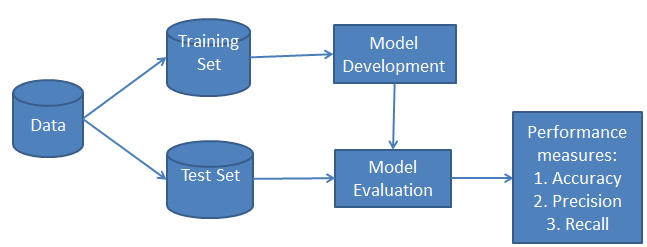
In this tutorial, you are going to learn about all of the following:

* Classification Workflow
* What is Naive Bayes classifier?
* How Naive Bayes classifier works?
* Classifier building in Scikit-learn
* It's advantages and disadvantages

**Classification Workflow**

Whenever you perform classification, the first step is to understand the problem and identify potential features and label. Features are those characteristics or attributes which affect the results of the label. For example, in the case of a loan distribution, bank manager's identify customer’s occupation, income, age, location, previous loan history, transaction history, and credit score. These characteristics are known as features which help the model classify customers.

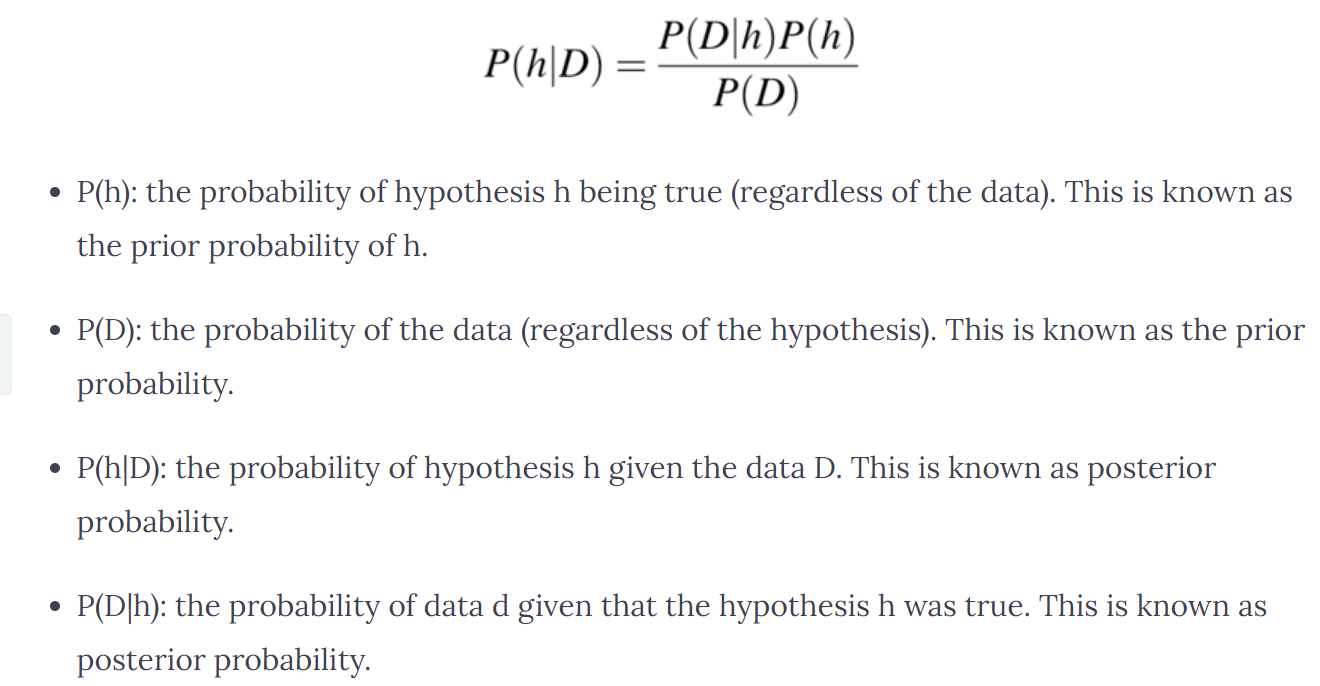
The classification has two phases, a learning phase, and the evaluation phase. In the learning phase, classifier trains its model on a given dataset and in the evaluation phase, it tests the classifier performance. Performance is evaluated on the basis of various parameters such as accuracy, error, precision, and recall.



**What is Naive Bayes Classifier?**

Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, accurate and reliable algorithm. Naive Bayes classifiers have high accuracy and speed on large datasets.

Naive Bayes classifier assumes that the effect of a particular feature in a class is independent of other features. For example, a loan applicant is desirable or not depending on his/her income, previous loan and transaction history, age, and location. Even if these features are interdependent, these features are still considered independently. This assumption simplifies computation, and that's why it is considered as naive. This assumption is called class conditional independence.



## **Your Task**

You are provided with a small dataset with different weather and temperature conditions. The task is for you to classify whether players will play or not, based on these conditions using the Naïve Bayes classifier.

Naive Bayes classifier calculates the probability of an event in the following steps:

Step 1: Calculate the prior probability for given class labels

Step 2: Find Likelihood probability with each attribute for each class

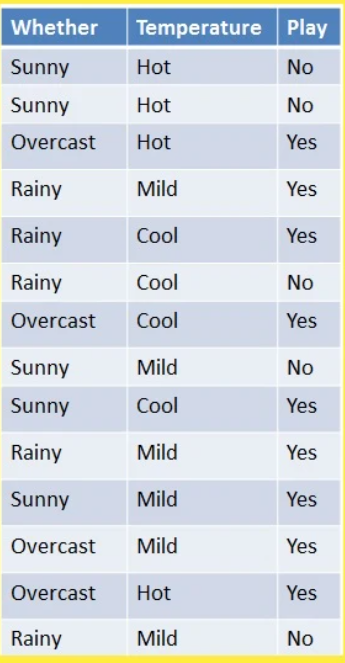
Step 3: Put these values in Bayes Formula and calculate posterior probability.

Step 4: See which class has a higher probability, given the input belongs to the higher probability class.

#### **Defining Dataset**

Let’s say you have gathered some information regarding 3 variables, Weather, Temperature and Play (whether a player will play outdoor sports or not on that day).

Weather Temperature Play



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. In other words, the independent variables will be weather, and temperature and the dependent variable is the Play column.

Open a new Jupyter Notebook and create a dummy dataset as shown by copying the codes below in your notebook and hit run:

***# Assigning features and label variables***

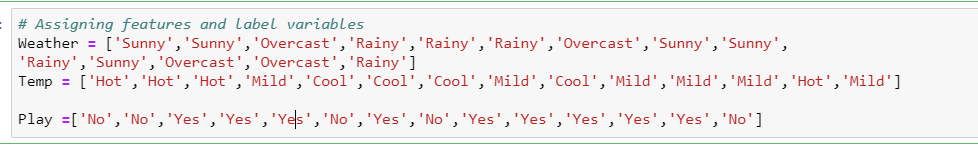
***weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny',***

***'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']***

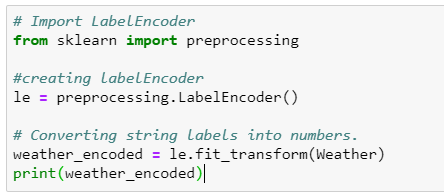
It should look like this:



#### **Encoding Features**

Next, you 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 LabelEncoder library for encoding labels with a value between 0 and one less than the number of discrete classes.

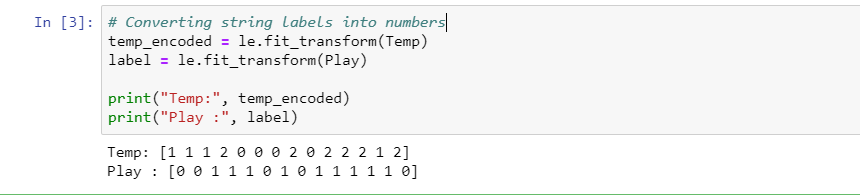
Type the following code to encode the weather column:



You will see an output like this:



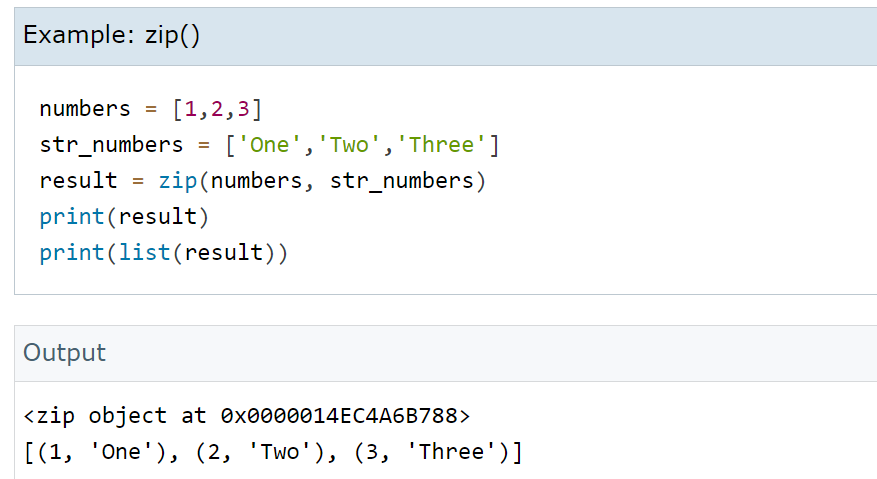
Similarly, encode Temperature and Play columns.



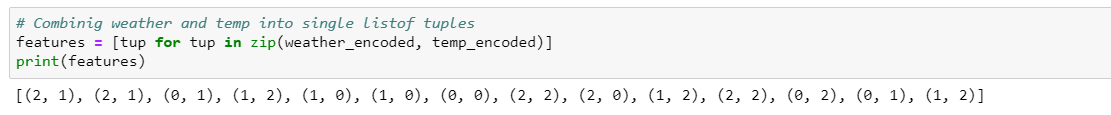
It will be easier to do a prediction if we have both values in a single variable.

Now combine both the features (weather and temp) in a single variable (list of tuples). A tuple is a collection of objects which is ordered. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets. You can use the zip function to combine the data. I have provided you a general example as well.

General example:



Use the code below to combine the encoded features into a single list of tuples.



#### **Generating Model**

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

* Create naive bayes classifier
* Fit the dataset on classifier
* Perform prediction with a new dataset which has [0,2] which is **Overcast and Mild.**

Use the following codes to build the model. We are using the current data to train the model. Then we ask the model to predict whether a player will play or not based on the new data.



You will see that you will get a predicted value of 1 which means players can play. All the calculations/probabilities have been done for you!

**Advantages**

* It is not only a simple approach but also a fast and accurate method for prediction.
* Naive Bayes has very low computation cost.
* It can efficiently work on a large dataset.
* It performs well in case of discrete response variable compared to the continuous variable.
* It can be used with multiple class prediction problems.
* It also performs well in the case of text analytics problems.
* When the assumption of independence holds, a Naive Bayes classifier performs better compared to other models like logistic regression.

**Conclusion**

In this tutorial, you learned about Naïve Bayes algorithm, it's working, Naive Bayes assumption, issues, implementation, advantages. Naive Bayes is the most straightforward and most potent algorithm. In spite of the significant advances of Machine Learning in the last couple of years, it has proved its worth. It has been successfully deployed in many applications from text analytics to recommendation engines.