Naive Bayes Classification using Scikit-learn

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.

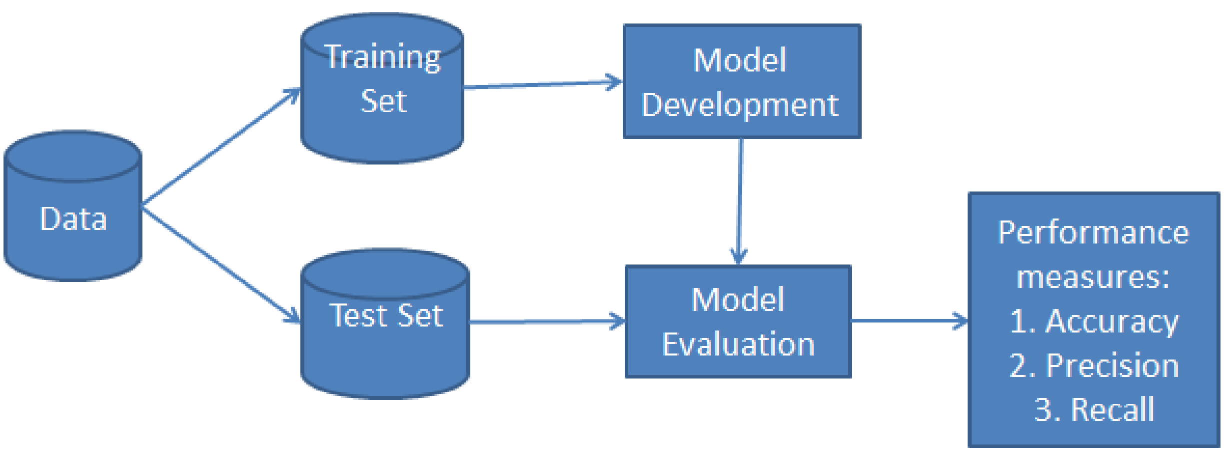
In this tutorial, you are going to learn about all of the following:

- Classification Workflow
- What is Naive Bayes classifier?
- How Naive Bayes classifier works?
- Classifier building in Scikit-learn
- Zero Probability Problem
- 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.

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- P(h): the probability of hypothesis h being true (regardless of the data). This is known as the prior probability of h.
- P(D): the probability of the data (regardless of the hypothesis). This is known as the prior probability.
- P(h|D): the probability of hypothesis h given the data D. This is known as posterior probability.
- P(D|h): the probability of data d given that the hypothesis h was true. This is known as posterior probability.

How Naive Bayes classifier works?

Let’s understand the working of Naive Bayes through an example. Given an example of weather conditions and playing sports. You need to calculate the probability of playing sports. Now, you need to classify whether players will play or not, based on the weather condition.

First Approach (In case of a single feature)

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 value 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.

For simplifying prior and posterior probability calculation you can use the two tables frequency and likelihood tables. Both of these tables will help you to calculate the prior and posterior probability. The Frequency table contains the occurrence of labels for all features. There are two likelihood tables. Likelihood Table 1 is showing prior probabilities of labels and Likelihood Table 2 is showing the posterior probability.

Whether	Play
Sunny	No
Sunny	No
Overcast	Yes
Rainy	Yes
Rainy	Yes
Rainy	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rainy	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table

Whether	No	Yes
Overcast		4
Sunny	2	3
Rainy	3	2
Total	5	9

Likelihood Table 1

Whether	No	Yes		
Overcast		4	=4/14	0.29
Sunny	2	3	=5/14	0.36
Rainy	3	2	=5/14	0.36
Total	5	9		
	=5/14	=9/14		
	0.36	0.64		

Likelihood Table 2

Whether	No	Yes	Posterior Probability for No	Posterior Probability for Yes
Overcast		4	0/5=0	4/9=0.44
Sunny	2	3	2/5=0.4	3/9=0.33
Rainy	3	2	3/5=0.6	2/9=0.22
Total	5	9		

Now suppose you want to calculate the probability of playing when the weather is overcast.

Probability of playing:

P(Yes | Overcast) = P(*Overcast* | Yes) P(Yes) / P (Overcast) ....(1)

1. Calculate Prior Probabilities:
   P(Overcast) = 4/14 = 0.29
   P(Yes)= 9/14 = 0.64
1. Calculate Posterior Probabilities:
   P(Overcast |Yes) = 4/9 = 0.44
1. Put Prior and Posterior probabilities in equation (1)
   P (Yes | Overcast) = 0.44 * 0.64 / 0.29 = 0.98(Higher)

Similarly, you can calculate the probability of not playing:

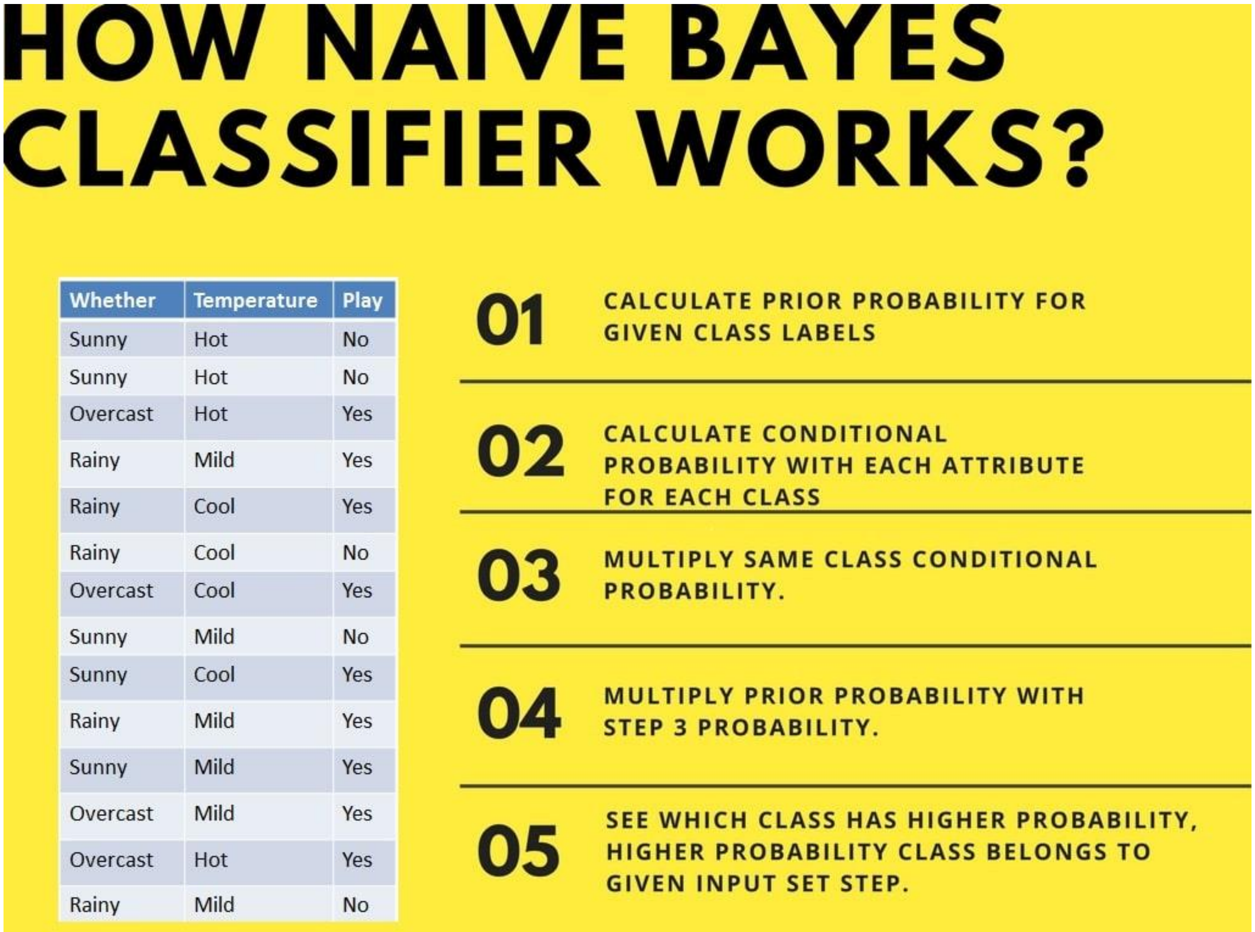
Probability of not playing:

P(No | Overcast) = P(*Overcast* | No) P(No) / P (Overcast) ....(2)

1. Calculate Prior Probabilities:
   P(Overcast) = 4/14 = 0.29
   P(No)= 5/14 = 0.36
1. Calculate Posterior Probabilities:
   P(Overcast |No) = 0/9 = 0
1. Put Prior and Posterior probabilities in equation (2)
   P (No | Overcast) = 0 * 0.36 / 0.29 = 0

The probability of a 'Yes' class is higher. So you can determine here if the weather is overcast than players will play the sport.

Second Approach (In case of multiple features)



Whether	Temperature	Play
Sunny	Hot	No
Sunny	Hot	No
Overcast	Hot	Yes
Rainy	Mild	Yes
Rainy	Cool	Yes
Rainy	Cool	No
Overcast	Cool	Yes
Sunny	Mild	No
Sunny	Cool	Yes
Rainy	Mild	Yes
Sunny	Mild	Yes
Overcast	Mild	Yes
Overcast	Hot	Yes
Rainy	Mild	No

Now suppose you want to calculate the probability of playing when the weather is overcast, and the temperature is mild.

Probability of playing:

P(Play= Yes | Weather=Overcast, Temp=Mild) = *P(Weather=Overcast, Temp=Mild | Play= Yes)*P(Play=Yes) ....(1)

P(Weather=Overcast, Temp=Mild | Play= Yes)= *P(Overcast* |Yes) P(Mild |Yes) ....(2)

1. Calculate Prior Probabilities: P(Yes)= 9/14 = 0.64
2. Calculate Posterior Probabilities: P(Overcast |Yes) = 4/9 = 0.44 P(Mild |Yes) = 4/9 = 0.44
3. Put Posterior probabilities in equation (2) P(Weather=Overcast, Temp=Mild | Play= Yes) = 0.44 * 0.44 = 0.1936(Higher)

4. Put Prior and Posterior probabilities in equation (1) P(Play= Yes | Weather=Overcast, Temp=Mild) = 0.1936*0.64 = 0.124

Similarly, you can calculate the probability of not playing:

Probability of not playing:

P(Play= No | *Weather=Overcast, Temp=Mild*) = P(*Weather=Overcast, Temp=Mild | Play*= No)P(Play=No) ....(3)

P(Weather=Overcast, Temp=Mild | Play= No)= *P(Weather=Overcast |Play*=No) P(Temp=Mild | Play=No) ....(4)

1. Calculate Prior Probabilities: P(No)= 5/14 = 0.36
2. Calculate Posterior Probabilities: P(Weather=Overcast |Play=No) = 0/9 = 0 P(Temp=Mild | Play=No)=2/5=0.4
3. Put posterior probabilities in equation (4) P(Weather=Overcast, Temp=Mild | Play= No) = 0 * 0.4= 0
4. Put prior and posterior probabilities in equation (3) P(Play= No | Weather=Overcast, Temp=Mild) = 0*0.36=0

The probability of a 'Yes' class is higher. So you can say here that if the weather is overcast than players will play the sport.