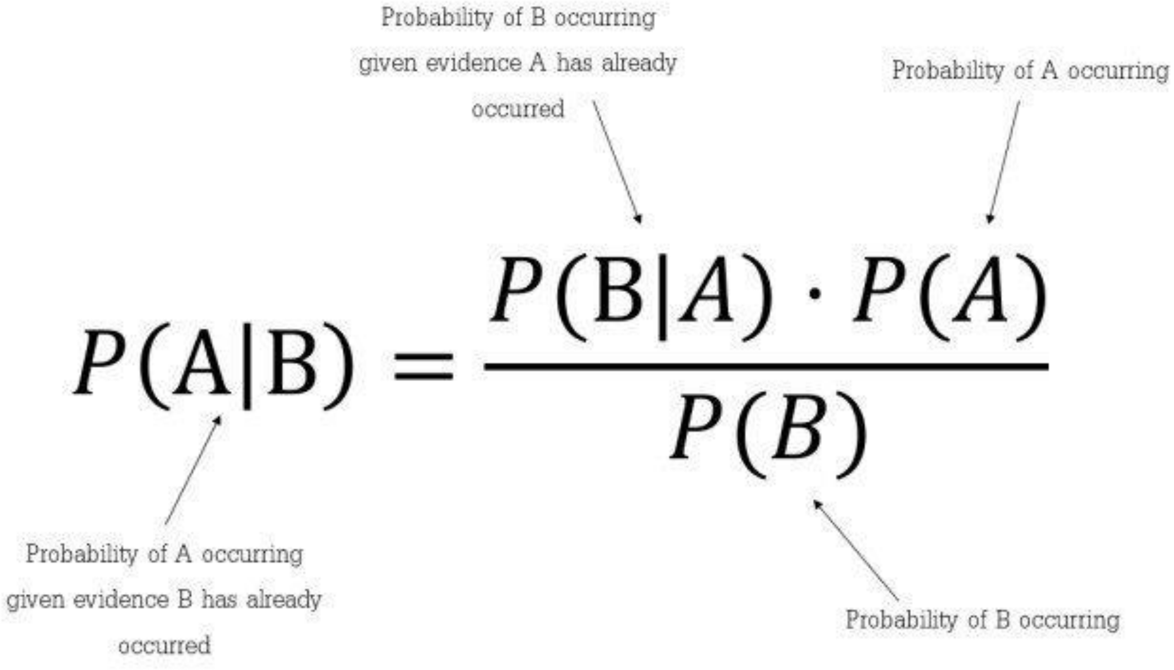
Naïve Bayes (Cameron Looney)

Bayes Theorem

Bayes’ Theorem is a simple mathematical formula used for calculating conditional probabilities.

Conditional probability is a measure of the probability of an event occurring given that another event has (by assumption, presumption, assertion, or evidence) occurred.

The formula is: —



$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{B}|A) \cdot P(A)}{P(B)}$$

Probability of B occurring given evidence A has already occurred
Probability of A occurring
Probability of A occurring given evidence B has already occurred
Probability of B occurring

Which tells us: how often A happens *given that B happens*, written **P(A|B)** also called posterior probability, When we know: how often B happens *given that A happens*, written **P(B|A)** and how likely A is on its own, written **P(A)** and how likely B is on its own, written **P(B).**

In simpler terms, Bayes' Theorem is a way of finding a probability when we know certain other probabilities.

Assumptions Made by Naïve Bayes

The fundamental Naïve Bayes assumption is that each feature makes an:

- independent
- equal

contribution to the outcome.

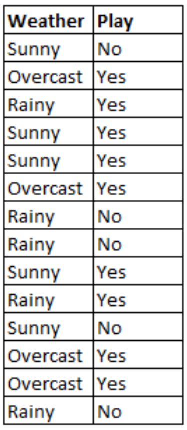
Let us take an example to get some better intuition. Consider the car theft problem with attributes Color, Type, Origin, and the target, Stolen can be either Yes or No.

How Naive Bayes algorithm works?

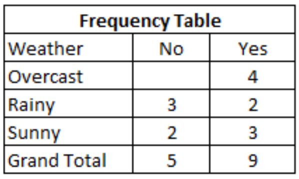
Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play' (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.

Step 1: Convert the data set into a frequency table

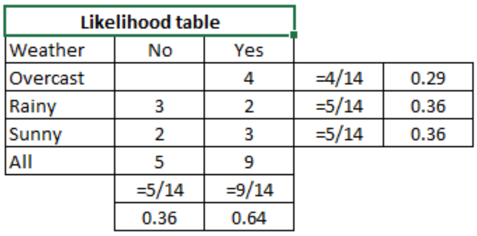
Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.



Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No



Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9



Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Problem: Players will play if weather is sunny. Is this statement is correct?

We can solve it using above discussed method of posterior probability.

P(Yes | Sunny) = P(Sunny | Yes) * P(Yes) / P (Sunny)

Here we have P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64

Now, P (Yes | Sunny) = 0.33 * 0.64 / 0.36 = 0.60, which has higher probability.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

What are the Pros and Cons of Naive Bayes?

Pros:

- It is easy and fast to predict class of test data set. It also perform well in multi class prediction

- When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
- It perform well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons:

- If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.
- On the other side naive Bayes is also known as a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.
- Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

Types of Naïve Bayes Classifiers

1. Multinomial Naïve Bayes Classifier

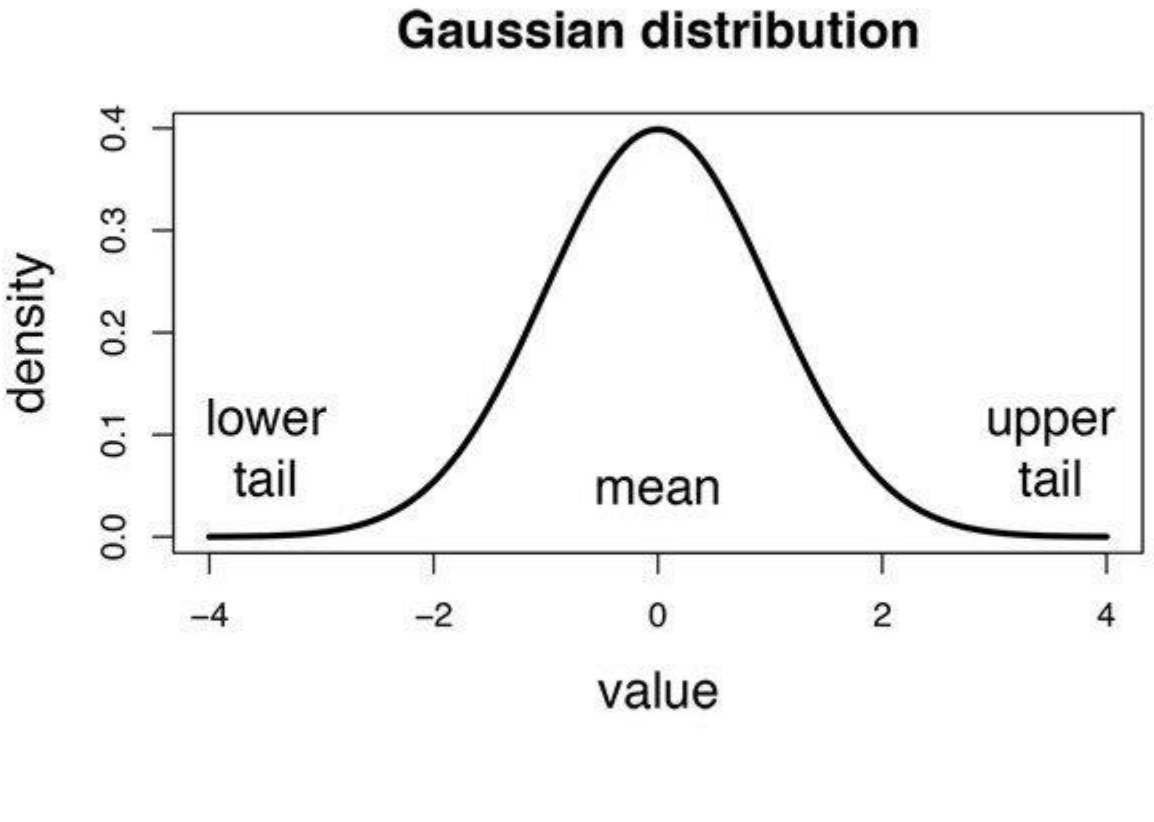
Feature vectors represent the frequencies with which certain events have been generated by a **multinomial distribution**. This is the event model typically used for document classification.

2. Bernoulli Naïve Bayes Classifier:

In the multivariate Bernoulli event model, features are independent booleans (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification tasks, where binary term occurrence (i.e. a word occurs in a document or not) features are used rather than term frequencies (i.e. frequency of a word in the document).

3. Gaussian Naïve Bayes Classifier:

In Gaussian Naïve Bayes, continuous values associated with each feature are assumed to be distributed according to a **Gaussian distribution (**Normal distribution**)**. When plotted, it gives a bell-shaped curve which is symmetric about the mean of the feature values as shown below:



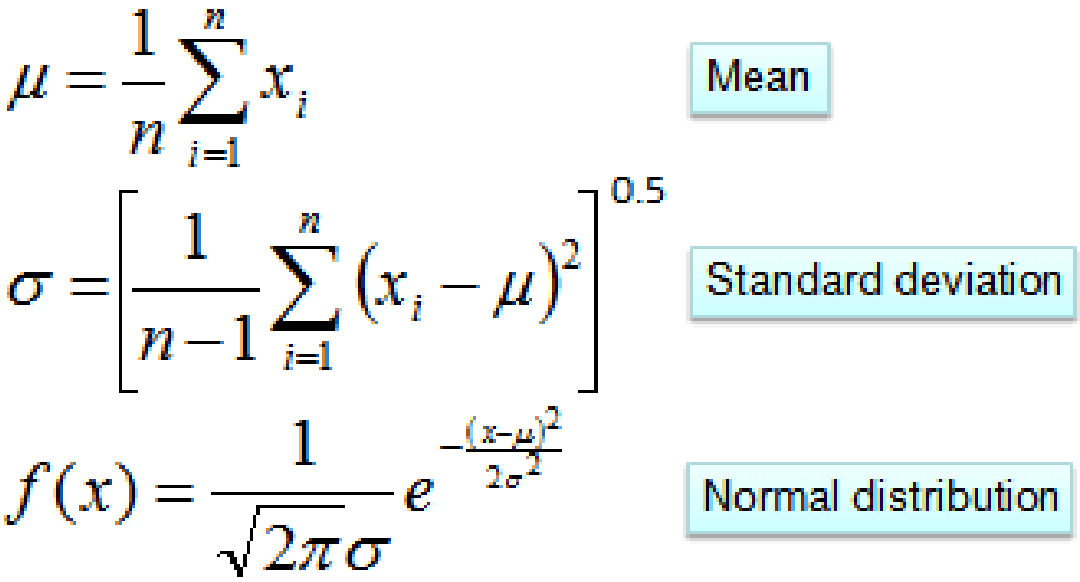
The likelihood of the features is assumed to be Gaussian, hence, conditional probability is given by:

$$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Now, what if any feature contains numerical values instead of categories i.e. Gaussian distribution.

One option is to transform the numerical values to their categorical counterparts before creating their frequency tables. The other option, as shown above, could be using the distribution of the numerical variable to have a good guess of the frequency. For example, one common method is to assume normal or gaussian distributions for numerical variables.

The probability density function for the normal distribution is defined by two parameters (mean and standard deviation).



$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Mean

$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu)^2\right]^{0.5}$$

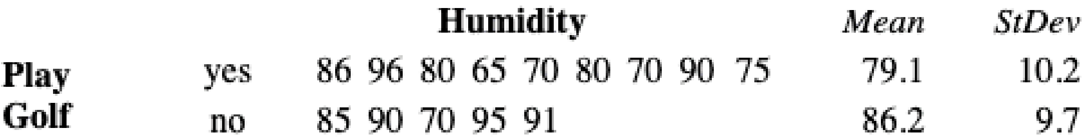
Standard deviation

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

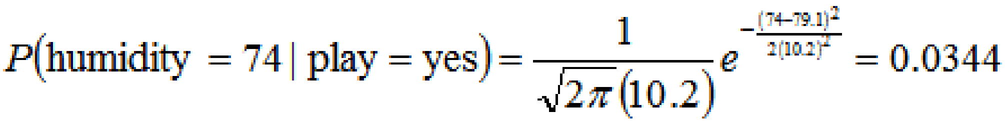
Normal distribution

Image credit

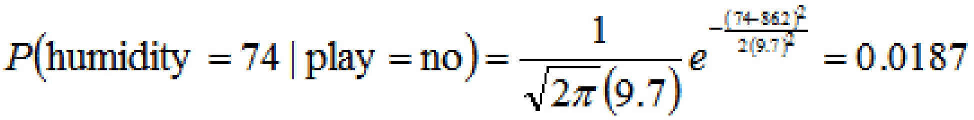
Consider the problem of playing golf, here the only predictor is `Humidity` and `Play Golf?` is the target. Using the above formula we can calculate posterior probability if we know the mean and standard deviation.



		Humidity	*Mean*	*StDev*
Play Golf	yes	86 96 80 65 70 80 70 90 75	79.1	10.2
	no	85 90 70 95 91	86.2	9.7



$$P(\text{humidity} = 74 \mid \text{play} = \text{yes}) = \frac{1}{\sqrt{2\pi}(10.2)} e^{-\frac{(74-79.1)^2}{2(10.2)^2}} = 0.0344$$



$$P(\text{humidity} = 74 \mid \text{play} = \text{no}) = \frac{1}{\sqrt{2\pi}(9.7)} e^{-\frac{(74-86.2)^2}{2(9.7)^2}} = 0.0187$$

Laplace smoothing

It is introduced to solve the problem of zero probability i.e. **when a query point contains a new observation, which is not yet seen in training data while calculating probabilities**.

The idea behind Laplace Smoothing: To ensure that our posterior probabilities are never zero, we add 1 to the numerator, and we add k to the denominator. So, in the case that we don't have a particular ingredient in our training set, the posterior probability comes out to 1 / N + k instead of zero. Plugging this value into the product doesn't kill our ability to make a prediction as plugging in a zero does.