A Gentle Introduction to Bayes Theorem for Machine Learning

by **Jason Brownlee** on October 4, 2019 in **Probability**

Last Updated on December 4, 2019

Bayes Theorem provides a principled way for calculating a conditional probability.

It is a deceptively simple calculation, although it can be used to easily calculate the conditional probability of events where intuition often fails.

Although it is a powerful tool in the field of probability, Bayes Theorem is also widely used in the field of machine learning. Including its use in a probability framework for fitting a model to a training dataset, referred to as maximum a posteriori or MAP for short, and in developing models for classification predictive modeling problems such as the Bayes Optimal Classifier and Naive Bayes.

In this post, you will discover Bayes Theorem for calculating conditional probabilities and how it is used in machine learning.

After reading this post, you will know:

- What Bayes Theorem is and how to work through the calculation on a real scenario.
- What the terms in the Bayes theorem calculation mean and the intuitions behind them.
- Examples of how Bayes theorem is used in classifiers, optimization and causal models.

Kick-start your project with my new book Probability for Machine Learning, including *step-by-step tutorials* and the *Python source code* files for all examples.

Let's get started.

- **Update Oct/2019**: Join the discussion about this tutorial on HackerNews.
- **Update Oct/2019**: Expanded to add more examples and uses of Bayes Theorem.

A Gentle Introduction to Bayes Theorem for Machine Learning
Photo by Marco Verch, some rights reserved.

Overview

This tutorial is divided into six parts; they are:

1. Bayes Theorem of Conditional Probability
2. Naming the Terms in the Theorem
3. Worked Example for Calculating Bayes Theorem
    1. Diagnostic Test Scenario
    2. Manual Calculation
    3. Python Code Calculation
    4. Binary Classifier Terminology
4. Bayes Theorem for Modeling Hypotheses
5. Bayes Theorem for Classification
    1. Naive Bayes Classifier
    2. Bayes Optimal Classifier
6. More Uses of Bayes Theorem in Machine Learning
    1. Bayesian Optimization
    2. Bayesian Belief Networks

Bayes Theorem of Conditional Probability

Before we dive into Bayes theorem, let's review marginal, joint, and conditional probability.

Recall that marginal probability is the probability of an event, irrespective of other random variables. If the random variable is independent, then it is the probability of the event directly, otherwise, if the variable is dependent upon other variables, then the marginal probability is the probability of the event summed over all outcomes for the dependent variables, called the sum rule.

- **Marginal Probability**: The probability of an event irrespective of the outcomes of other random variables, e.g. P(A).

The joint probability is the probability of two (or more) simultaneous events, often described in terms of events A and B from two dependent random variables, e.g. X and Y. The joint probability is often summarized as just the outcomes, e.g. A and B.

- **Joint Probability**: Probability of two (or more) simultaneous events, e.g. P(A and B) or P(A, B).

The conditional probability is the probability of one event given the occurrence of another event, often described in terms of events A and B from two dependent random variables e.g. X and Y.

- **Conditional Probability**: Probability of one (or more) event given the occurrence of another event, e.g. P(A given B) or P(A | B).

The joint probability can be calculated using the conditional probability; for example:

- P(A, B) = P(A | B) * P(B)

This is called the product rule. Importantly, the joint probability is symmetrical, meaning that:

- P(A, B) = P(B, A)

The conditional probability can be calculated using the joint probability; for example:

- P(A | B) != P(B | A)

We are now up to speed with marginal, joint and conditional probability. If you would like more background on these fundamentals, see the tutorial:

- A Gentle Introduction to Joint, Marginal, and Conditional Probability

An Alternate Way To Calculate Conditional Probability

Now, there is another way to calculate the conditional probability.

Specifically, one conditional probability can be calculated using the other conditional probability; for example:

- P(A|B) = P(B|A) * P(A) / P(B)

The reverse is also true; for example:

- P(B|A) = P(A|B) * P(B) / P(A)

This alternate approach of calculating the conditional probability is useful either when the joint probability is challenging to calculate (which is most of the time), or when the reverse conditional probability is available or easy to calculate.

This alternate calculation of the conditional probability is referred to as Bayes Rule or Bayes Theorem, named for Reverend Thomas Bayes, who is credited with first describing it. It is grammatically correct to refer to it as Bayes' Theorem (with the apostrophe), but it is common to omit the apostrophe for simplicity.

- **Bayes Theorem**: Principled way of calculating a conditional probability without the joint probability.

It is often the case that we do not have access to the denominator directly, e.g. P(B).

We can calculate it an alternative way; for example:

- P(A|B) = P(B|A) * P(A) / P(B|A) * P(A) + P(B|not A) * P(not A)

Or with brackets around the denominator for clarity:

- P(A|B) = P(B|A) * P(A) / (P(B|A) * P(A) + P(B|not A) * P(not A))

Note: the denominator is simply the expansion we gave above.

As such, if we have P(A), then we can calculate P(not A) as its complement; for example:

- P(not A) = 1 – P(A)

Additionally, if we have P(not B|not A), then we can calculate P(B|not A) as its complement; for example:

- P(B|not A) = 1 – P(not B|not A)

Now that we are familiar with the calculation of Bayes Theorem, let's take a closer look at the meaning of the terms in the equation.

Naming the Terms in the Theorem

It can be helpful to think about the calculation from these different perspectives and help to map your problem onto the equation.

Firstly, in general, the result P(A|B) is referred to as the **posterior probability** and P(A) is referred to as the **prior probability**.

- P(A|B): Posterior probability.
- P(A): Prior probability.

Sometimes P(B|A) is referred to as the **likelihood** and P(B) is referred to as the **evidence**.

- P(B|A): Likelihood.
- P(B): Evidence.

This allows Bayes Theorem to be restated as:

- Posterior = Likelihood * Prior / Evidence

We can make this clear with a smoke and fire case.

What is the probability that there is fire given that there is smoke?

Where P(Fire) is the Prior, P(Smoke|Fire) is the Likelihood, and P(Smoke) is the evidence:

- P(Fire|Smoke) = P(Smoke|Fire) * P(Fire) / P(Smoke)

You can imagine the same situation with rain and clouds.

Now that we are familiar with Bayes Theorem and the meaning of the terms, let's look at a scenario where we can calculate it.

Worked Example for Calculating Bayes Theorem

First we will define a scenario then work through a manual calculation, a calculation in Python, and a calculation using the terms that may be familiar to you from the field of binary classification.

1. Diagnostic Test Scenario
2. Manual Calculation
3. Python Code Calculation
4. Binary Classifier Terminology

Let's go.

Diagnostic Test Scenario

An excellent and widely used example of the benefit of Bayes Theorem is in the analysis of a medical diagnostic test.

Scenario: Consider a human population that may or may not have cancer (Cancer is True or False) and a medical test that returns positive or negative for detecting cancer (Test is Positive or Negative), e.g. like a mammogram for detecting breast cancer.

> ***Problem***: *If a randomly selected patient has the test and it comes back positive, what is the probability that the patient has cancer?*

Manual Calculation

Medical diagnostic tests are not perfect; they have error.

Sometimes a patient will have cancer, but the test will not detect it. This capability of the test to detect cancer is referred to as the **sensitivity**, or the true positive rate.

In this case, we will contrive a sensitivity value for the test. The test is good, but not great, with a true positive rate or sensitivity of 85%. That is, of all the people who have cancer and are tested, 85% of them will get a positive result from the test.

Given this information, our intuition would suggest that there is an 85% probability that the patient has cancer.

Our intuitions of probability are wrong.

This type of error in interpreting probabilities is so common that it has its own name; it is referred to as the base rate fallacy.

It has this name because the error in estimating the probability of an event is caused by ignoring the base rate. That is, it ignores the probability of a randomly selected person having cancer, regardless of the results of a diagnostic test.

In this case, we can assume the probability of breast cancer is low, and use a contrived base rate value of one person in 5,000, or (0.0002) 0.02%.

- P(Cancer=True) = 0.02%.

We can correctly calculate the probability of a patient having cancer given a positive test result using Bayes Theorem.

Let's map our scenario onto the equation:

- P(A|B) = P(B|A) * P(A) / P(B)
- P(Cancer=True | Test=Positive) = P(Test=Positive|Cancer=True) * P(Cancer=True) / P(Test=Positive)

We know the probability of the test being positive given that the patient has cancer is 85%, and we know the base rate or the prior probability of a given patient having cancer is 0.02%; we can plug these values in:

- P(Cancer=True | Test=Positive) = 0.85 * 0.0002 / P(Test=Positive)

We don't know P(Test=Positive), it's not given directly.

Instead, we can estimate it using:

- P(B) = P(B|A) * P(A) + P(B|not A) * P(not A)
- P(Test=Positive) = P(Test=Positive|Cancer=True) * P(Cancer=True) + P(Test=Positive|Cancer=False) * P(Cancer=False)

Firstly, we can calculate P(Cancer=False) as the complement of P(Cancer=True), which we already know

- P(Cancer=False) = 1 – P(Cancer=True)
- = 1 – 0.0002
- = 0.9998

Let's plugin what we have:

We can plug in our known values as follows:

This requires additional information.

Specifically, we need to know how good the test is at correctly identifying people that do not have cancer. That is, testing negative result (Test=Negative) when the patient does not have cancer (Cancer=False), called the true negative rate or the **specificity**.

We will use a contrived specificity value of 95%.

- P(Test=Negative | Cancer=False) = 0.95

With this final piece of information, we can calculate the false positive or false alarm rate as the complement of the true negative rate.

- P(Test=Positive|Cancer=False) = 1 – P(Test=Negative | Cancer=False)
- = 1 – 0.95
- = 0.05

We can plug this false alarm rate into our calculation of P(Test=Positive) as follows:

- P(Test=Positive) = 0.85 * 0.0002 + 0.05 * 0.9998
- P(Test=Positive) = 0.00017 + 0.04999
- P(Test=Positive) = 0.05016

Excellent, so the probability of the test returning a positive result, regardless of whether the person has cancer or not is about 5%.

We now have enough information to calculate Bayes Theorem and estimate the probability of a randomly selected person having cancer if they get a positive test result.

- P(Cancer=True | Test=Positive) = P(Test=Positive|Cancer=True) * P(Cancer=True) / P(Test=Positive)
- P(Cancer=True | Test=Positive) = 0.85 * 0.0002 / 0.05016
- P(Cancer=True | Test=Positive) = 0.00017 / 0.05016
- P(Cancer=True | Test=Positive) = 0.003389154704944

The calculation suggests that if the patient is informed they have cancer with this test, then there is only 0.33% chance that they have cancer.

It is a terrible diagnostic test!

The example also shows that the calculation of the conditional probability requires *enough* information.

For example, if we have the values used in Bayes Theorem already, we can use them directly.

This is rarely the case, and we typically have to calculate the bits we need and plug them in, as we did in this case. In our scenario we were given 3 pieces of information, the the **base rate**, the **sensitivity** (or true positive rate), and the **specificity** (or true negative rate).

- **Sensitivity**: 85% of people with cancer will get a positive test result.

We might imagine that Bayes Theorem allows us to be even more precise about a given scenario. For example, if we had more information about the patient (e.g. their age) and about the domain (e.g. cancer rates for age ranges), and in turn we could offer an even more accurate probability estimate.

That was a lot of work.

Let's look at how we can calculate this exact scenario using a few lines of Python code.

Python Code Calculation

To make this example concrete, we can perform the calculation in Python.

The example below performs the same calculation in vanilla Python (no libraries), allowing you to play with the parameters and test different scenarios.

```
# calculate the probability of cancer patient and diagnostic test

# calculate P(A|B) given P(A), P(B|A), P(B|not A)
def bayes_theorem(p_a, p_b_given_a, p_b_given_not_a):
 # calculate P(not A)
 not_a = 1 - p_a
 # calculate P(B)
 p_b = p_b_given_a * p_a + p_b_given_not_a * not_a
 # calculate P(A|B)
 p_a_given_b = (p_b_given_a * p_a) / p_b
 return p_a_given_b

# P(A)
p_a = 0.0002
# P(B|A)
p_b_given_a = 0.85
# P(B|not A)
p_b_given_not_a = 0.05
# calculate P(A|B)
result = bayes_theorem(p_a, p_b_given_a, p_b_given_not_a)
# summarize
print('P(A|B) = %.3f%%' % (result * 100))
```

This is a helpful little script that you may want to adapt to new scenarios.

Now, it is common to describe the calculation of Bayes Theorem for a scenario using the terms from binary classification. It provides a very intuitive way for thinking about a problem. In the next section we will review these terms and see how they map onto the probabilities in the theorem and how they relate to our scenario.

Binary Classifier Terminology

It may be helpful to think about the cancer test example in terms of the common terms from binary (two-class) classification, i.e. where notions of specificity and sensitivity come from.

Personally, I find these terms help everything to make sense.

Firstly, let's define a confusion matrix:

```
                     | Positive Class      | Negative Class
Positive Prediction | True Positive (TP)  | False Positive (FP)
Negative Prediction | False Negative (FN) | True Negative (TN)
```

We can then define some rates from the confusion matrix:

- True Positive Rate (TPR) = TP / (TP + FN)
- False Positive Rate (FPR) = FP / (FP + TN)
- True Negative Rate (TNR) = TN / (TN + FP)
- False Negative Rate (FNR) = FN / (FN + TP)

These terms are called rates, but they can also be interpreted as probabilities.

Also, it might help to notice:

- TPR + FNR = 1.0, or:
  - FNR = 1.0 – TPR

Recall that in a previous section that we calculated the false positive rate given the complement of true negative rate, or FPR = 1.0 – TNR.

Some of these rates have special names, for example:

- Sensitivity = TPR
- Specificity = TNR

We can map these rates onto familiar terms from Bayes Theorem:

- **P(B|A)**: True Positive Rate (TPR).
- **P(not B|not A)**: True Negative Rate (TNR).
- **P(B|not A)**: False Positive Rate (FPR).
- **P(not B|A)**: False Negative Rate (FNR).

We can also map the base rates for the condition (class) and the treatment (prediction) on familiar terms from Bayes Theorem:

- **P(A)**: Probability of a Positive Class (PC).
- **P(not A)**: Probability of a Negative Class (NC).
- **P(B)**: Probability of a Positive Prediction (PP).
- **P(not B)**: Probability of a Negative Prediction (NP).

Now, let's consider Bayes Theorem using these terms:

- P(A|B) = P(B|A) * P(A) / P(B)
- P(A|B) = (TPR * PC) / PP

Where we often cannot calculate P(B), so we use an alternative:

- P(B) = P(B|A) * P(A) + P(B|not A) * P(not A)
- P(B) = TPR * PC + FPR * NC

Now, let's look at our scenario of cancer and a cancer detection test.

The class or condition would be "*Cancer*" and the treatment or prediction would the "*Test*".

First, let's review all of the rates:

- True Positive Rate (TPR): 85%
- False Positive Rate (FPR): 5%
- True Negative Rate (TNR): 95%
- False Negative Rate (FNR): 15%

Let's also review what we know about base rates:

- Positive Class (PC): 0.02%
- Negative Class (NC): 99.98%

Plugging things in, we can calculate the probability of a positive test result (a positive prediction) as the probability of a positive test result given cancer (the true positive rate) multiplied by the base rate for having cancer (the positive class), plus the probability if a positive test result given no cancer (the false positive rate) plus the probability of not having cancer (the negative class).

The calculation with these terms is as follows:

- P(B) = P(B|A) * P(A) + P(B|not A) * P(not A)
- P(B) = TPR * PC + FPR * NC
- P(B) = 85% * 0.02% + 5% * 99.98%
- P(B) = 5.016%

We can then calculate Bayes Theorem for the scenario, namely the probability of cancer given a positive test result (the posterior) is the probability of a positive test result given cancer (the true positive rate) multiplied by the probability of having cancer (the positive class rate), divided by the probability of a positive test result (a positive prediction).

The calculation with these terms is as follows:

- P(A|B) = P(B|A) * P(A) / P(B)
- P(A|B) = TPR * PC / PP
- P(A|B) = 85% * 0.02% / 5.016%
- P(A|B) = 0.339%

It turns out that in this case, the posterior probability that we are calculating with the Bayes theorem is equivalent to the precision, also called the Positive Predictive Value (PPV) of the confusion matrix:

- PPV = TP / (TP + FP)

Or, stated in our classifier terms:

- P(A|B) = PPV
- PPV = TPR * PC / PP

So why do we go to all of the trouble of calculating the posterior probability?

Because we don't have the confusion matrix for a population of people both with and without cancer that have been tested and have been not tested. Instead, all we have is some priors and probabilities about our population and our test.

This highlights when we might choose to use the calculation in practice.

Specifically, when we have beliefs about the events involved, but we cannot perform the calculation by counting examples in the real world.

Bayes Theorem for Modeling Hypotheses

Bayes Theorem is a useful tool in applied machine learning.

It provides a way of thinking about the relationship between data and a model.

A machine learning algorithm or model is a specific way of thinking about the structured relationships in the data. In this way, a model can be thought of as a hypothesis about the relationships in the data, such as the relationship between input (*X*) and output (*y*). The practice of applied machine learning is the testing and analysis of different hypotheses (models) on a given dataset.

If this idea of thinking of a model as a hypothesis is new to you, see this tutorial on the topic:

- What is a Hypothesis in Machine Learning?

Bayes Theorem provides a probabilistic model to describe the relationship between data (*D*) and a hypothesis (h); for example:

- P(h|D) = P(D|h) * P(h) / P(D)

Breaking this down, it says that the probability of a given hypothesis holding or being true given some observed data can be calculated as the probability of observing the data given the hypothesis multiplied by the probability of the hypothesis being true regardless of the data, divided by the probability of observing the data regardless of the hypothesis.

> *Bayes theorem provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself.*

— Page 156, Machine Learning, 1997.

Under this framework, each piece of the calculation has a specific name; for example:

This gives a useful framework for thinking about and modeling a machine learning problem.

If we have some prior domain knowledge about the hypothesis, this is captured in the prior probability. If we don't, then all hypotheses may have the same prior probability.

If the probability of observing the data P(D) increases, then the probability of the hypothesis holding given the data P(h|D) decreases. Conversely, if the probability of the hypothesis P(h) and the probability of observing the data given hypothesis increases, the probability of the hypothesis holding given the data P(h|D) increases.

The notion of testing different models on a dataset in applied machine learning can be thought of as estimating the probability of each hypothesis (h1, h2, h3, … in H) being true given the observed data.

The optimization or seeking the hypothesis with the maximum posterior probability in modeling is called maximum a posteriori or MAP for short.

> *Any such maximally probable hypothesis is called a maximum a posteriori (MAP) hypothesis. We can determine the MAP hypotheses by using Bayes theorem to calculate the posterior probability of each candidate hypothesis.*

— Page 157, Machine Learning, 1997.

Under this framework, the probability of the data (D) is constant as it is used in the assessment of each hypothesis. Therefore, it can be removed from the calculation to give the simplified unnormalized estimate as follows:

- max h in H P(h|D) = P(D|h) * P(h)

If we do not have any prior information about the hypothesis being tested, they can be assigned a uniform probability, and this term too will be a constant and can be removed from the calculation to give the following:

- max h in H P(h|D) = P(D|h)

That is, the goal is to locate a hypothesis that best explains the observed data.

Fitting models like linear regression for predicting a numerical value, and logistic regression for binary classification can be framed and solved under the MAP probabilistic framework. This provides an alternative to the more common maximum likelihood estimation (MLE) framework.

Bayes Theorem for Classification

The problem of classification predictive modeling can be framed as calculating the conditional probability of a class label given a data sample, for example:

- P(class|data) = (P(data|class) * P(class)) / P(data)

Where P(class|data) is the probability of class given the provided data.

This calculation can be performed for each class in the problem and the class that is assigned the largest probability can be selected and assigned to the input data.

In practice, it is very challenging to calculate full Bayes Theorem for classification.

The priors for the class and the data are easy to estimate from a training dataset, if the dataset is suitability representative of the broader problem.

The conditional probability of the observation based on the class P(data|class) is not feasible unless the number of examples is extraordinarily large, e.g. large enough to effectively estimate the probability distribution for all different possible combinations of values. This is almost never the case, we will not have sufficient coverage of the domain.

As such, the direct application of Bayes Theorem also becomes intractable, especially as the number of variables or features (n) increases.

Naive Bayes Classifier

The solution to using Bayes Theorem for a conditional probability classification model is to simplify the calculation.

The Bayes Theorem assumes that each input variable is dependent upon all other variables. This is a cause of complexity in the calculation. We can remove this assumption and consider each input variable as being independent from each other.

This changes the model from a dependent conditional probability model to an independent conditional probability model and dramatically simplifies the calculation.

This means that we calculate P(data|class) for each input variable separately and multiple the results together, for example:

- P(class | X1, X2, …, Xn) = P(X1|class) * P(X2|class) * … * P(Xn|class) * P(class) / P(data)

We can also drop the probability of observing the data as it is a constant for all calculations, for example:

- P(class | X1, X2, …, Xn) = P(X1|class) * P(X2|class) * … * P(Xn|class) * P(class)

The word "*naive*" is French and typically has a diaeresis (umlaut) over the "i", which is commonly left out for simplicity, and "Bayes" is capitalized as it is named for Reverend Thomas Bayes.

For tutorials on how to implement Naive Bayes from scratch in Python see:

- How to Develop a Naive Bayes Classifier from Scratch in Python
- Naive Bayes Classifier From Scratch in Python

Bayes Optimal Classifier

The Bayes optimal classifier is a probabilistic model that makes the most likley prediction for a new example, given the training dataset.

This model is also referred to as the Bayes optimal learner, the Bayes classifier, Bayes optimal decision boundary, or the Bayes optimal discriminant function.

- **Bayes Classifier**: Probabilistic model that makes the most probable prediction for new examples.

Specifically, the Bayes optimal classifier answers the question:

> *What is the most probable classification of the new instance given the training data?*

This is different from the MAP framework that seeks the most probable hypothesis (model). Instead, we are interested in making a specific prediction.

The equation below demonstrates how to calculate the conditional probability for a new instance (*vi*) given the training data (*D*), given a space of hypotheses (*H*).

- P(vj | D) = sum {h in H} P(vj | hi) * P(hi | D)

Where *vj* is a new instance to be classified, *H* is the set of hypotheses for classifying the instance, *hi* is a given hypothesis, *P(vj | hi)* is the posterior probability for *vi* given hypothesis *hi*, and *P(hi | D)* is the posterior probability of the hypothesis *hi* given the data *D*.

Selecting the outcome with the maximum probability is an example of a Bayes optimal classification.

Any model that classifies examples using this equation is a Bayes optimal classifier and no other model can outperform this technique, on average.

We have to let that sink in. It is a big deal.

Because the Bayes classifier is optimal, the Bayes error is the minimum possible error that can be made.

The Naive Bayes classifier is an example of a classifier that adds some simplifying assumptions and attempts to approximate the Bayes Optimal Classifier.

For more on the Bayesian optimal classifier, see the tutorial:

- A Gentle Introduction to the Bayes Optimal Classifier

More Uses of Bayes Theorem in Machine Learning

Developing classifier models may be the most common application on Bayes Theorem in machine learning.

Nevertheless, there are many other applications. Two important examples are optimization and causal models.

Bayesian Optimization

Global optimization is a challenging problem of finding an input that results in the minimum or maximum cost of a given objective function.

Typically, the form of the objective function is complex and intractable to analyze and is often non-convex, nonlinear, high dimension, noisy, and computationally expensive to evaluate.

Bayesian Optimization provides a principled technique based on Bayes Theorem to direct a search of a global optimization problem that is efficient and effective. It works by building a probabilistic model of the objective function, called the surrogate function, that is then searched efficiently with an acquisition function before candidate samples are chosen for evaluation on the real objective function.

Bayesian Optimization is often used in applied machine learning to tune the hyperparameters of a given well-performing model on a validation dataset.

For more on Bayesian Optimization including how to implement it from scratch, see the tutorial:

- How to Implement Bayesian Optimization from Scratch in Python

Bayesian Belief Networks

Probabilistic models can define relationships between variables and be used to calculate probabilities.

An alternative is to develop a model that preserves known conditional dependence between random variables and conditional independence in all other cases. Bayesian networks are a probabilistic graphical model that explicitly capture the known conditional dependence with directed edges in a graph model. All missing connections define the conditional independencies in the model.

As such Bayesian Networks provide a useful tool to visualize the probabilistic model for a domain, review all of the relationships between the random variables, and reason about causal probabilities for scenarios given available evidence.

The networks are not exactly Bayesian by definition, although given that both the probability distributions for the random variables (nodes) and the relationships between the random variables (edges) are specified subjectively, the model can be thought to capture the "belief" about a complex domain.

For more on Bayesian Belief Networks, see the tutorial:

- A Gentle Introduction to Bayesian Belief Networks

Further Reading

This section provides more resources on the topic if you are looking to go deeper.

Related Tutorials

- A Gentle Introduction to Joint, Marginal, and Conditional Probability
- What is a Hypothesis in Machine Learning?
- How to Develop a Naive Bayes Classifier from Scratch in Python
- Naive Bayes Classifier From Scratch in Python
- How to Implement Bayesian Optimization from Scratch in Python
- A Gentle Introduction to Bayesian Belief Networks

Books

- Pattern Recognition and Machine Learning, 2006.
- Machine Learning, 1997.
- Pattern Classification, 2nd Edition, 2001.
- Machine Learning: A Probabilistic Perspective, 2012.

- Maximum a posteriori estimation, Wikipedia.
- False positives and false negatives, Wikipedia.
- Base rate fallacy, Wikipedia.
- Sensitivity and specificity, Wikipedia.
- Taking the Confusion out of the Confusion Matrix, 2016.

Summary

In this post, you discovered Bayes Theorem for calculating conditional probabilities and how it is used in machine learning.

Specifically, you learned:

- What Bayes Theorem is and how to work through the calculation on a real scenario.
- What the terms in the Bayes theorem calculation mean and the intuitions behind them.
- Examples of how Bayes theorem is used in classifiers, optimization and causal models.

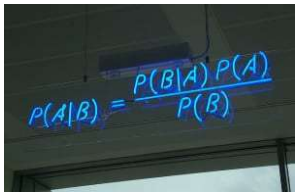
Do you have any questions?
Ask your questions in the comments below and I will do my best to answer.

More On This Topic

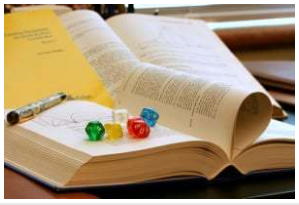
A Gentle Introduction to the Bayes Optimal Classifier

How to Develop a Naive Bayes Classifier from Scratch…

Develop an Intuition for Bayes Theorem With Worked Examples



Naive Bayes Classifier From Scratch in Python



Naive Bayes for Machine Learning

A Gentle Introduction to the Central Limit Theorem…

About Jason Brownlee

Jason Brownlee, PhD is a machine learning specialist who teaches developers how to get results with modern machine learning methods via hands-on tutorials.

View all posts by Jason Brownlee →

‹ Probability for Machine Learning (7-Day Mini-Course)

How to Develop a Naive Bayes Classifier from Scratch in Python ›

44 Responses to *A Gentle Introduction to Bayes Theorem for Machine Learning*

Ante October 4, 2019 at 11:15 am #

REPLY

P(Cancer=True | Test=Positive) = 0.85 * 0.0002 / [0.85 * 0.0002 + P(Test=Positive|Cancer=False) * 0.9998]

Jason Brownlee October 6, 2019 at 8:06 am #

REPLY

Great suggestion, thanks.

Bryan Weast October 5, 2019 at 7:30 am #

REPLY

So are we all gonna die or what

Jason Brownlee October 6, 2019 at 8:13 am #

REPLY

Eventually. But not from bayes theorem.

Ahmed October 5, 2019 at 4:25 pm #

REPLY

Very informative!

Thank you for clearing the misconception in the last part!!

Jason Brownlee October 6, 2019 at 8:15 am #

REPLY

I'm happy the tutorial was helpful.

Jeremy Tt October 12, 2019 at 5:12 am #

REPLY

"Given this information, our intuition would suggest that there is an 85% probability that the patient has cancer.

And again, our intuitions of probability are wrong."

You lost me here. I'll blame my ignorance. Please explain how if the probability of the test returning positive for a real positive for a given outcome makes us incorrect to say that there is an 85% chance that "the" patient has cancer. Should the statement say that it would suggest … "a" patient has cancer where someone extrapolates the base across the population, not the specific outcome?

Jason Brownlee October 12, 2019 at 7:08 am #

REPLY

Perhaps continue working through the example to see what is going on?

Jeremy Tt October 12, 2019 at 7:58 am #

REPLY

I believe I do, but I really just want to be sure.

What we are trying to say is instead of “The correct calculation suggests that if the patient is informed they have cancer with this test, then there is only 0.33% chance that they have cancer. It is a terrible diagnostic test!”, we are saying that a diagnostic test that has an 85% chance of correctly identifying cancer in a patient as well as a 95% rate of correctly identifying no cancer would have a rate of 0.33% of false positives for the presence of cancer in a sample set of 0.02% of the set actually having cancer and a 0.51% (?) of false negatives in the same population. Therefore 3 out of 1000 people who took the test would be incorrectly told they have cancer. (Hopefully take it two more times… to virtually eliminate the probalistic error.).

Is that correct?

Jeremy Tt October 12, 2019 at 8:13 am #

REPLY

Sorry, I know it is a contrived sample, but articles like these are truly helpful to me to understand what I have not studied. My last question would be, why would the h be the 85% as I assume that was measured during the clinical trial period and the 0.2% is the unknown that is hypothesized to be the incident rate of existing cancer cases?

Thank you!

Jason Brownlee October 12, 2019 at 8:16 am #

REPLY

Not quite.

Recall the beginning of the section, the root question is: if someone takes the test and it reports they have cancer, what is the probability they actually have cancer?

Note that 95% refers to the test detecting cancer IF the person has cancer. And 85% of no cancer IF the person has no cancer.

The conditions on each claim are critical to unraveling the scenario.

The final result answer the question. If a person who may or may not have cancer takes the test and is told they have cancer, what is the probability they have cancer, and the answer is 0.33%. As in, “extremely unlikely”.

Jeremy Tt October 12, 2019 at 8:34 am #

Right. Using the theory is useful to answer the question, but as you said, the claims

more correct, but over a larger percentage of the same population gives a higher total percentage.

“The test is good, but not great, with a true positive rate or sensitivity of 85%. That is, of all the people who have cancer and are tested, 85% of them will get a positive result from the test.

P(Test=Positive | Cancer=True) = 0.85”

…

“P(Test=Negative | Cancer=False) = 0.95”

“We can plug this false alarm rate into our Bayes Theorem as follows:

P(Cancer=True | Test=Positive) = 0.85 * 0.0002 / 0.85 * 0.0002 + 0.05 * 0.9998
= 0.00017 / 0.00017 + 0.04999
= 0.00017 / 0.05016
= 0.003389154704944”

Jason Brownlee October 13, 2019 at 9:53 am #

I think I get where you’re coming from.

Firstly, let’s define a confusion matrix:

```
1                    | Positive Class      | Negative Class
2 Positive Prediction | True Positive (TP)  | False Positive (FP)
3 Negative Prediction | False Negative (FN) | True Negative (TN)
```

We can then define some rates:

– True Positive Rate (TPR) = TP / (TP + FN)
– False Positive Rate (FPR) = FP / (FP + TN)
– True Negative Rate (TNR) = TN / (TN + FP)
– False Negative Rate (FNR) = FN / (FN + TP)

Also:

– Sensitivity = TPR
– Specificity = TNR

We can then map these rates onto Bayes Theorem:

– P(B|A): True Positive Rate (TPR).
– P(not B|not A): True Negative Rate (TNR).
– P(B|not A): True Negative Rate (TNR).
– P(not B|A): False Positive Rate (FPR).

And the base rates:

– P(A): Positive Class (PC)
– P(not A): Negative Class (NC)
– P(B): Positive Prediction (PP)

P(A|B) = P(B|A) * P(A) / P(B)
P(A|B) = (TPR * PC) / PP

Where we like to use:

P(B) = P(B|A) * P(A) + P(B|not A) * P(not A)
P(B) = TPR * PC + FPR * NC

Now, let's look at our scenario.

The "Class" would be "Cancer" and the "Prediction" would the "Test". We know the rates:

– TPR: 85%
– FPR: 5%
– TNR: 95%
– FNR: 15%

Let's review what we know about base rates:

– PC: 0.02%
– NC: 99.98%
– PP: 5.016%
– NP: 94.984%

Plugging things in, we get:

P(A|B) = P(B|A) * P(A) / P(B)
P(A|B) = (TPR * PC) / PP
P(A|B) = (85% * 0.02%) / 5.016%

Or, when we calculated P(B):

P(B) = P(B|A) * P(A) + P(B|not A) * P(not A)
P(B) = TPR * PC + FPR * NC
P(B) = 85% * 0.02% + 5% * 99.98%

Does that help?

I might update the post to add this discussion.

JG March 30, 2020 at 10:05 pm #

REPLY

Hi Jason,

I have read this Bayes ML tutorial and, in my case it is summarized pretty well all the concepts and math notation around Bayes probability approach vs. frequency.

One more time I want to say you are great person with a lot of generosity by teaching to all of us these marvelous ML technologics procedures… Thank you.

As a matter of facts I am working right now on Covid19 Image diagnosis by x-rays chest and, these ideas are the keys for binary classification.

regards,
JG

Thanks, I'm happy it helped.

Good luck with your project!

David Ackerman June 28, 2020 at 5:03 am #

REPLY

Hi,

Great job in explaining the formula and using it in an example.

I think there is one error. In computing P(Test=Positive|Cancer=False), after the equal sign you use: 1-P(Test=Negative)|Cancer=False). I do not think the right side equals what is on the left side of the equal sign. The complement needed to match the left side would be: 1-P(Test=Positive|Cancer=False) — and that information is not given in the facts.

David

David

Jason Brownlee June 28, 2020 at 5:59 am #

REPLY

P(Test=Positive|Cancer=False) = 1 – P(Test=Negative | Cancer=False)

We can restate the same thing more generally, taken from the definition of terms above:

P(B|not A) = 1 – P(not B|not A)

David Ackerman June 28, 2020 at 5:35 am #

REPLY

I just realized I was wrong in what I said above. 😕

I think I should have said:
a. the expression on the right does not provide the information needed on the right side.
b. the facts given do not allow one to compute the left side.

David

David Ackerman June 28, 2020 at 5:39 am #

REPLY

Oops. I made another error. (Not my day.)

In (a) in the prior comment, the last two words should have been: *left* side.

David

Not sure I agree. Perhaps you can elaborate how exactly?

David Ackerman June 28, 2020 at 10:17 pm #

REPLY

Jason,

I think this is the problem with your hypothetical:

1. You correctly compute the numerator (as .0002 * .85).

2. You then say, “We still do not know the probability of a positive test result given no cancer.” We do not know that quite yet. In order to have that last piece of information we need to add to the denominator P(B|notA *P(1-A). In your example this is (Test=Positive|Cancer=False * Cancer=False). We already have that in the facts: it is .15 * .9998. (‘.15’ is the complement of Test=Positive|Cancer=True and you’ve already computed Cancer=False as .9998.)

If you’d send me your email address, I’d like to offer a private comment which is probably not of general interest to others.

David

Jason Brownlee June 29, 2020 at 6:35 am #

REPLY

You can contact me any time here:
https://machinelearningmastery.com/contact/

Sivakumar B February 26, 2021 at 11:19 pm #

REPLY

Excellent Explanation with examples. For people in my generation who haven’t learnt ML in the colleges and want to understand the concept this is the best tutorial. Thanks for putting this in a nice manner and also make it open for people. Great work. God Bless you.

Jason Brownlee February 27, 2021 at 6:04 am #

REPLY

Thanks!

Amir March 3, 2021 at 2:38 pm #

REPLY

Thanks for your useful post as always. You mentioned that Bayes inference can be intractable that I think it can be related to marginalization that cannot be done analytically in most cases.
I think this was the emerging point of approximation methods like MCMC and drop out in deep learning.

section 2.3 MAXIMUM A POSTERIORI ESTIMATION OF THE PARAMETER VECTOR in reference Simon O. Haykin – Neural Networks and Learning Machines-Prentice Hall (2008)

Any comments on this?
Thanks!

Jason Brownlee March 4, 2021 at 5:46 am #

REPLY

Not familiar with that paper sorry.

I think we're talking about different things. Generally when it comes to bayes, we cannot calculate the joint probabilities – we don't have all the data, so we approximate.

K.S Lam April 17, 2021 at 12:23 am #

REPLY

The probability that a patient has cancer given the test returns a positive result is 33.9%. This probability is called positive predictive value (PPV). The false positive probability is 66.1%.

Whereas the probability that a patient has no cancer given the test returns a negative result is 100%. This probability is called negative predictive value (NPV). The false negative probability is 0%.

Ma May 17, 2021 at 12:17 am #

REPLY

Hi, in bayesian network how can calcute kullback leibler divergence with R software?

Jason Brownlee May 17, 2021 at 5:39 am #

REPLY

I don't know, sorry.

alireza July 13, 2021 at 3:22 am #

REPLY

Hi and thank you for your grate post

If you have anything about Bayesian Latent Transition Analysis please let me know.

alirezamomajed@gmail.com

Jason Brownlee July 13, 2021 at 5:19 am #

REPLY

Not at this stage.

Hi Jason,
You're saying:
We can calculate it an alternative way; for example:
P(B) = P(B|A) * P(A) + P(B|not A) * P(not A)

What would be the formula, if we want to estimate P(A) in alternative way? It doesn't seem straightforward, not just a swap of A and B.

Jason Brownlee August 11, 2021 at 7:46 am #

REPLY

If you have P(B) and not P(A), perhaps just reverse your terms.

Jan October 10, 2021 at 3:32 am #

REPLY

You know what makes things confusing and you know how to explain them well. I have been reading different sources regarding Bayes but I am very much comfortable with how you drill things down in great detail!

The posterior, prior, likelihood and evidence and not to mention the confusion matrix. Thank you!

Adrian Tam October 13, 2021 at 5:58 am #

REPLY

Thank you. Hope you like other posts here as well.

Saeid December 1, 2021 at 1:41 am #

REPLY

Dear Jason.

Ooooh my God.

This is a fucking nice post. Thank you.

"Fitting models like linear regression for predicting a numerical value, and logistic regression for binary classification can be framed and solved under the MAP probabilistic framework. This provides an alternative to the more common maximum likelihood estimation (MLE) framework."

I think prior probablity is the important challenge in Bayes approach.

Regression and Classification are supervise learning and we have labels.

Clustering or Co-clustering are unsupervise learning and we have no labels.

When we have no labels, we don't approximate prior probability and we can't use Bayes theorem, unless we assumption prior equal uniform.

Is this true?

Adrian Tam December 2, 2021 at 2:05 am #

REPLY

All Bayesian methods need a prior. That's the characteristic of it.

cp December 3, 2021 at 7:08 am #

REPLY

I am trying to solidify my understanding of recall and naive bayes with ML train/test sets.

In the article, it was mentioned as follows:

P(B|A): True Positive Rate (TPR) (which is basically recall).

According to the example, P(B|A) means P(Test=Positive | Cancer=True). So, if I am doing k-fold sampling, I train things as usual, but for test sets, if I modify them so that they only contain (Cancer=true) data points, feed them into an ML algorithm (for example, logistic regression) and then count how many data points that the algorithm predicts as positive, would the result be equivalent to P(Test=Positive | Cancer=True) which should be again the same as recall?

Adrian Tam December 8, 2021 at 6:45 am #

REPLY

It seems you don't realize P(B|A) is not a precise notation as we don't know how this probability is computed. Recall is exactly TP/(TP+FN). Here TP should be both Cancer=True and Test=True while FN is Cancer=False and Test=True

cp December 9, 2021 at 4:45 pm #

REPLY

Hmm, thanks so much for the explanation, but I have to admit that I still feel a bit unclear.

This post offered a good explanation that

P(Test=Positive | Cancer=True) = 0.85 means "all the people who have cancer and are tested, 85% of them will get a positive result from the test."

So, if we consider logistic regression as a testing mechanism, and feed all the data of people who have cancer, then the number of people that the regression predicts as positive is basically: P(Test=Positive | Cancer=True). Am I understanding correctly?

Adrian Tam December 10, 2021 at 4:19 am #

REPLY

Yes. But you don't test in this way. You need samples from Cancer=False as well.

Understood now. Thanks!

Shekar June 6, 2022 at 12:52 am #

REPLY

Excellent Overview of Bayes theorem.. Thank you

James Carmichael June 6, 2022 at 8:57 am #

REPLY

Thank you for the feedback Shekar!

Leave a Reply

Name (required)

Email (will not be published) (required)

SUBMIT COMMENT

Welcome!
I'm *Jason Brownlee* PhD
and I **help developers** get results with **machine learning**.
Read more

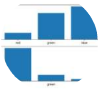
Never miss a tutorial:

Picked for you:


How to Use ROC Curves and Precision-Recall Curves for Classification in Python


How and When to Use a Calibrated Classification Model with scikit-learn


How to Implement Bayesian Optimization from Scratch in Python


How to Calculate the KL Divergence for Machine Learning


A Gentle Introduction to Cross-Entropy for Machine Learning