Need help with machine learning? Take the FREE Crash-Course.

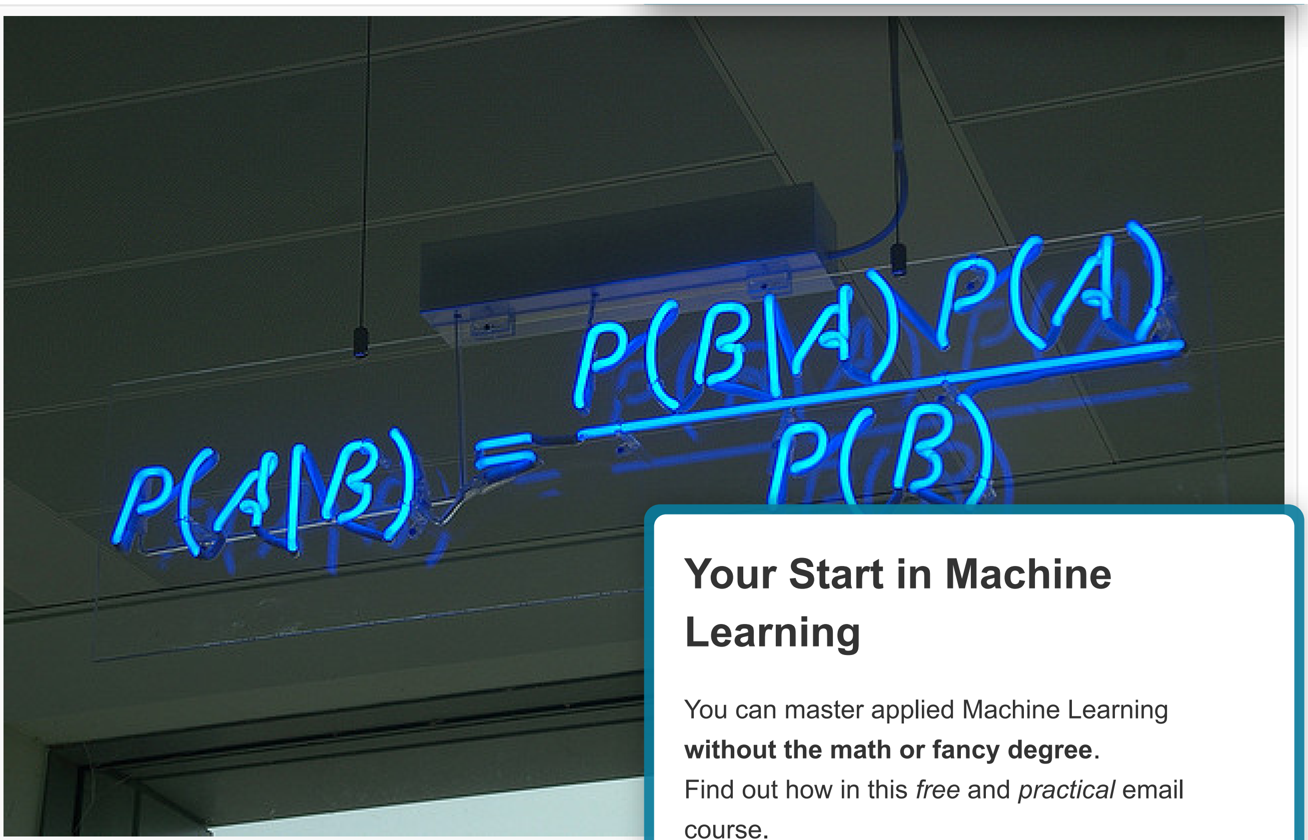
How To Implement Naive Bayes From Scratch in Python

by **Jason Brownlee** on December 8, 2014 in **Algorithms From Scratch**

The Naive Bayes algorithm is simple and effective and should be one of the first methods you try on a classification problem.

In this tutorial you are going to learn about the Naive Bayes algorithm including how it works and how to implement it from scratch in Python.

- **Update**: Check out the follow-up on tips for using the naive bayes algorithm titled: "Better Naive Bayes: 12 Tips To Get The Most From The Naive Bayes Algorithm".
- **Update March/2018**: Added alternate link to download the dataset as the original appears to have been taken down.



Naive Baye

Photo by Matt Buck, s

About Naive Bayes

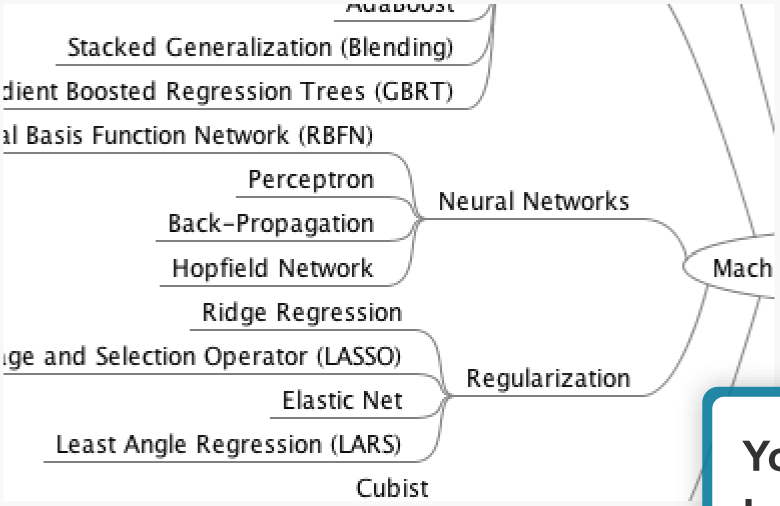
The Naive Bayes algorithm is an intuitive method that to each class to make a prediction. It is the supervised learning approach you would come up with if you wanted to model a predictive modeling problem probabilistically.

Naive bayes simplifies the calculation of probabilities by assuming that the probability of each attribute belonging to a given class value is independent of all other attributes. This is a strong assumption but results in a fast and effective method.

The probability of a class value given a value of an attribute is called the conditional probability. By multiplying the conditional probabilities together for each attribute for a given class value, we have a probability of a data instance belonging to that class.

To make a prediction we can calculate probabilities of the instance belonging to each class and select the class value with the highest probability.

Naive bases is often described using categorical data because it is easy to describe and calculate using ratios. A more useful version of the algorithm for our purposes supports numeric attributes and assumes the values of each numerical attribute are normally distributed (fall somewhere on a bell curve). Again, this is a strong assumption, but still gives robust results.


AdaBoost
Stacked Generalization (Blending)
dient Boosted Regression Trees (GBRT)
al Basis Function Network (RBFN)
Perceptron
Back-Propagation
Hopfield Network
Neural Networks
Mach
Ridge Regression
ge and Selection Operator (LASSO)
Elastic Net
Regularization
Least Angle Regression (LARS)
Cubist

Sample of the handy machine learning algorithms mind map

Predict the Onset of Diabetes

The test problem we will use in this tutorial is the Pima

This problem is comprised of 768 observations of med
describe instantaneous measurements taken from the
pregnant and blood workup. All patients are women ag ir
units vary from attribute to attribute.

Each record has a class value that indicates whether the patient suffered an onset of diabetes within 5 years of when the measurements were taken (1) or not (0).

This is a standard dataset that has been studied a lot in machine learning literature. A good prediction accuracy is 70%-76%.

Below is a sample from the *pima-indians.data.csv* file to get a sense of the data we will be working with (update: download from here).

```
6,148,72,35,0,33.6,0.627,50,1
1,85,66,29,0,26.6,0.351,31,0
8,183,64,0,0,23.3,0.672,32,1
1,89,66,23,94,28.1,0.167,21,0
0,137,40,35,168,43.1,2.288,33,1
```

Naive Bayes Algorithm Tutorial

This tutorial is broken down into the following steps:

1. **Handle Data**: Load the data from CSV file and split it into training and test datasets.

2. **Summarize Data**: summarize the properties in the training dataset so that we can calculate probabilities and make predictions.
3. **Make a Prediction**: Use the summaries of the dataset to generate a single prediction.
4. **Make Predictions**: Generate predictions given a test dataset and a summarized training dataset.
5. **Evaluate Accuracy**: Evaluate the accuracy of predictions made for a test dataset as the percentage correct out of all predictions made.
6. **Tie it Together**: Use all of the code elements to present a complete and standalone implementation of the Naive Bayes algorithm.

1. Handle Data

The first thing we need to do is load our data file. The data is in CSV format without a header line or any quotes. We can open the file with the open function an the csv module.

We also need to convert the attributes that were loade them. Below is the **loadCsv()** function for loading the

```
import csv
def loadCsv(filename):
    lines = csv.reader(open(filename, "rb"))
    dataset = list(lines)
    for i in range(len(dataset)):
        dataset[i] = [float(x) for x in dataset
    return dataset
```

We can test this function by loading the pima indians that were loaded.

```
filename = 'pima-indians-diabetes.data.csv'
dataset = loadCsv(filename)
print('Loaded data file {0} with {1} rows').format(filename, len(dataset))
```

Running this test, you should see something like:

```
Loaded data file pima-indians-diabetes.data.csv rows
```

Next we need to split the data into a training dataset that Naive Bayes can use to make predictions and a test dataset that we can use to evaluate the accuracy of the model. We need to split the data set randomly into train and datasets with a ratio of 67% train and 33% test (this is a common ratio for testing an algorithm on a dataset).

Below is the **splitDataset()** function that will split a given dataset into a given split ratio.

```
import random
def splitDataset(dataset, splitRatio):
    trainSize = int(len(dataset) * splitRatio)
    trainSet = []
    copy = list(dataset)
    while len(trainSet) < trainSize:
        index = random.randrange(len(copy))
        trainSet.append(copy.pop(index))
    return [trainSet, copy]
```

We can test this out by defining a mock dataset with 5 instances, split it into training and testing datasets and print them out to see which data instances ended up where.

```
dataset = [[1], [2], [3], [4], [5]]
splitRatio = 0.67
train, test = splitDataset(dataset, splitRatio)
print('Split {0} rows into train with {1} and test with {2}').format(len(dataset), train, test)
```

Running this test, you should see something like:

```
Split 5 rows into train with [[4], [3], [5]] and test with [[1], [2]]
```

2. Summarize Data

The naive bayes model is comprised of a summary of then used when making predictions.

The summary of the training data collected involves th ute, by class value. For example, if there are two class val ean and standard deviation for each attribute (7) and class summaries.

These are required when making predictions to calcul belonging to each class value.

We can break the preparation of this summary data d

1. Separate Data By Class
2. Calculate Mean
3. Calculate Standard Deviation
4. Summarize Dataset
5. Summarize Attributes By Class

Separate Data By Class

The first task is to separate the training dataset instances by class value so that we can calculate statistics for each class. We can do that by creating a map of each class value to a list of instances that belong to that class and sort the entire dataset of instances into the appropriate lists.

The **separateByClass()** function below does just this.

```
def separateByClass(dataset):
    separated = {}
    for i in range(len(dataset)):
        vector = dataset[i]
        if (vector[-1] not in separated):
            separated[vector[-1]] = []
        separated[vector[-1]].append(vector)
    return separated
```

You can see that the function assumes that the last attribute (-1) is the class value. The function returns a map of class values to lists of data instances.

We can test this function with some sample data, as follows:

```
dataset = [[1,20,1], [2,21,0], [3,22,1]]
separated = separateByClass(dataset)
print('Separated instances: {0}').format(separated)
```

Running this test, you should see something like:

```
Separated instances: {0: [[2, 21, 0]], 1: [[1, 20, 1], [3, 22, 1]]}
```

Calculate Mean

We need to calculate the mean of each attribute for a class value. The mean is the central middle or central tendency of the data, and we will use it as the middle of our gaussian distribution when calculating probabilities.

We also need to calculate the standard deviation of ea tion
describes the variation of spread of the data, and we v ach
attribute in our Gaussian distribution when calculating

The standard deviation is calculated as the square roo e
average of the squared differences for each attribute v
method, which subtracts 1 from the number of attribute

```
import math
def mean(numbers):
    return sum(numbers)/float(len(numbers))

def stdev(numbers):
    avg = mean(numbers)
    variance = sum([pow(x-avg,2) for x in numbers])/float(len(numbers)-1)
    return math.sqrt(variance)
```

We can test this by taking the mean of the numbers from 1 to 5.

```
numbers = [1,2,3,4,5]
print('Summary of {0}: mean={1}, stdev={2}').format(numbers, mean(numbers), stdev(numbers))
```

Running this test, you should see something like:

```
Summary of [1, 2, 3, 4, 5]: mean=3.0, stdev=1.58113883008
```

Summarize Dataset

Now we have the tools to summarize a dataset. For a given list of instances (for a class value) we can calculate the mean and the standard deviation for each attribute.

The zip function groups the values for each attribute across our data instances into their own lists so that we can compute the mean and standard deviation values for the attribute.

```
def summarize(dataset):
    summaries = [(mean(attribute), stdev(attribute)) for attribute in zip(*dataset)]
    del summaries[-1]
    return summaries
```

We can test this **summarize()** function with some test data that shows markedly different mean and standard deviation values for the first and second data attributes.

```
dataset = [[1,20,0], [2,21,1], [3,22,0]]
summary = summarize(dataset)
print('Attribute summaries: {0}').format(summary)
```

Running this test, you should see something like:

```
Attribute summaries: [(2.0, 1.0), (21.0, 1.0)]
```

Summarize Attributes By Class

We can pull it all together by first separating our training dataset into instances grouped by class. Then calculate the summaries for each attribute.

```
def summarizeByClass(dataset):
    separated = separateByClass(dataset)
    summaries = {}
    for classValue, instances in separated.ite
        summaries[classValue] = summarize(inst
    return summaries
```

We can test this **summarizeByClass()** function with a

```
dataset = [[1,20,1], [2,21,0], [3,22,1], [4,22
summary = summarizeByClass(dataset)
print('Summary by class value: {0}').format(su
```

Running this test, you should see something like:

```
Summary by class value:
{0: [(3.0, 1.4142135623730951), (21.5, 0.7071067811865476)],
1: [(2.0, 1.4142135623730951), (21.0, 1.4142135623730951)]}
```

3. Make Prediction

We are now ready to make predictions using the summaries prepared from our training data. Making predictions involves calculating the probability that a given data instance belongs to each class, then selecting the class with the largest probability as the prediction.

We can divide this part into the following tasks:

1. Calculate Gaussian Probability Density Function
2. Calculate Class Probabilities
3. Make a Prediction
4. Estimate Accuracy

Calculate Gaussian Probability Density Function

We can use a Gaussian function to estimate the probability of a given attribute value, given the known mean and standard deviation for the attribute estimated from the training data.

Given that the attribute summaries where prepared for each attribute and class value, the result is the conditional probability of a given attribute value given a class value.

See the references for the details of this equation for the Gaussian probability density function. In summary we are plugging our known details into the Gaussian (attribute value, mean and standard deviation) and reading off the likelihood that our attribute value belongs to the class.

In the **calculateProbability()** function we calculate the exponent first, then calculate the main division. This lets us fit the equation nicely on two lines.

```
import math
def calculateProbability(x, mean, stdev):
    exponent = math.exp(-(math.pow(x-mean,2)/(2*math.pow(stdev,2))))
    return (1 / (math.sqrt(2*math.pi) * stdev))
```

We can test this with some sample data, as follows.

```
x = 71.5
mean = 73
stdev = 6.2
probability = calculateProbability(x, mean, st
print('Probability of belonging to this class: 
```

Running this test, you should see something like:

```
Probability of belonging to this class: 0.0624
```

Calculate Class Probabilities

Now that we can calculate the probability of an attribu probabilities of all of the attribute values for a data instance and come up with a probability of the entire data instance belonging to the class.

We combine probabilities together by multiplying them. In the **calculateClassProbabilities()** below, the probability of a given data instance is calculated by multiplying together the attribute probabilities for each class. the result is a map of class values to probabilities.

```
def calculateClassProbabilities(summaries, inputVector):
    probabilities = {}
    for classValue, classSummaries in summaries.iteritems():
        probabilities[classValue] = 1
        for i in range(len(classSummaries)):
            mean, stdev = classSummaries[i]
            x = inputVector[i]
            probabilities[classValue] *= calculateProbability(x, mean, stdev)
    return probabilities
```

We can test the **calculateClassProbabilities()** function.

```
summaries = {0:[(1, 0.5)], 1:[(20, 5.0)]}
inputVector = [1.1, '?']
probabilities = calculateClassProbabilities(summaries, inputVector)
print('Probabilities for each class: {0}').format(probabilities)
```

Running this test, you should see something like:

```
Probabilities for each class: {0: 0.7820853879509118, 1: 6.298736258150442e-05}
```

Make a Prediction

Now that can calculate the probability of a data instance belonging to each class value, we can look for the largest probability and return the associated class.

The **predict()** function belong does just that.

```
def predict(summaries, inputVector):
    probabilities = calculateClassProbabilities(summaries, inputVector)
    bestLabel, bestProb = None, -1
    for classValue, probability in probabilities.iteritems():
        if bestLabel is None or probability > bestProb:
            bestProb = probability
            bestLabel = classValue
    return bestLabel
```

We can test the **predict()** function as follows:

```
summaries = {'A':[(1, 0.5)], 'B':[(20, 5.0)]}
inputVector = [1.1, '?']
result = predict(summaries, inputVector)
print('Prediction: {0}').format(result)
```

Running this test, you should see something like:

```
Prediction: A
```

4. Make Predictions

Finally, we can estimate the accuracy of the model by making predictions for each data instance in our test dataset. The **getPredictions()** will do this and return a list of predictions for each test instance.

```
def getPredictions(summaries, testSet):
    predictions = []
    for i in range(len(testSet)):
        result = predict(summaries, testSet[i])
        predictions.append(result)
    return predictions
```

We can test the **getPredictions()** function.

```
summaries = {'A':[(1, 0.5)], 'B':[(20, 5.0)]}
testSet = [[1.1, '?'], [19.1, '?']]
predictions = getPredictions(summaries, testSet)
print('Predictions: {0}').format(predictions)
```

Running this test, you should see something like:

```
Predictions: ['A', 'B']
```

5. Get Accuracy

The predictions can be compared to the class values in the test dataset and a classification accuracy can be calculated as an accuracy ratio between 0& and 100%. The getAccuracy() will calculate this accuracy ratio.

```
def getAccuracy(testSet, predictions):
    correct = 0
    for x in range(len(testSet)):
        if testSet[x][-1] == predictions[x]:
            correct += 1
    return (correct/float(len(testSet))) * 100.0
```

We can test the **getAccuracy()** function using the sample code below.

```
testSet = [[1,1,1,'a'], [2,2,2,'a'], [3,3,3,'b']]
predictions = ['a', 'a', 'a']
accuracy = getAccuracy(testSet, predictions)
print('Accuracy: {0}').format(accuracy)
```

Running this test, you should see something like:

```
Accuracy: 66.6666666667
```

6. Tie it Together

Finally, we need to tie it all together.

Below provides the full code listing for Naive Bayes im

```
# Example of Naive Bayes implemented from Sc
import csv
import random
import math

def loadCsv(filename):
    lines = csv.reader(open(filename, "rb"))
    dataset = list(lines)
    for i in range(len(dataset)):
        dataset[i] = [float(x) for x in dataset[i]]
    return dataset

def splitDataset(dataset, splitRatio):
    trainSize = int(len(dataset) * splitRatio)
    trainSet = []
    copy = list(dataset)
    while len(trainSet) < trainSize:
        index = random.randrange(len(copy))
        trainSet.append(copy.pop(index))
    return [trainSet, copy]

def separateByClass(dataset):
    separated = {}
    for i in range(len(dataset)):
        vector = dataset[i]
        if (vector[-1] not in separated):
            separated[vector[-1]] = []
        separated[vector[-1]].append(vector)
    return separated

def mean(numbers):
    return sum(numbers)/float(len(numbers))

def stdev(numbers):
    avg = mean(numbers)
    variance = sum([pow(x-avg,2) for x in numbers])/float(len(numbers)-1)
    return math.sqrt(variance)

def summarize(dataset):
    summaries = [(mean(attribute), stdev(attribute)) for attribute in zip(*dataset)]
    del summaries[-1]
    return summaries

def summarizeByClass(dataset):
    separated = separateByClass(dataset)
    summaries = {}
    for classValue, instances in separated.iteritems():
        summaries[classValue] = summarize(instances)
    return summaries

def calculateProbability(x, mean, stdev):
    exponent = math.exp(-(math.pow(x-mean,2)/(2*math.pow(stdev,2))))
    return (1 / (math.sqrt(2*math.pi) * stde

def calculateClassProbabilities(summaries, i
    probabilities = {}
    for classValue, classSummaries in summari
        probabilities[classValue] = 1
        for i in range(len(classSummaries)):
            mean, stdev = classSummaries[i]
            x = inputVector[i]
            probabilities[classValue] *= cal
    return probabilities

def predict(summaries, inputVector):
    probabilities = calculateClassProbabilit
    bestLabel, bestProb = None, -1
    for classValue, probability in probabilit
        if bestLabel is None or probability >
            bestProb = probability
            bestLabel = classValue
    return bestLabel

def getPredictions(summaries, testSet):
    predictions = []
    for i in range(len(testSet)):
        result = predict(summaries, testSet[i])
        predictions.append(result)
    return predictions

def getAccuracy(testSet, predictions):
    correct = 0
    for i in range(len(testSet)):
        if testSet[i][-1] == predictions[i]:
            correct += 1
    return (correct/float(len(testSet))) * 100.0

def main():
    filename = 'pima-indians-diabetes.data.csv'
    splitRatio = 0.67
    dataset = loadCsv(filename)
    trainingSet, testSet = splitDataset(dataset, splitRatio)
    print('Split {0} rows into train={1} and test={2} rows').format(len(dataset), len(training
    # prepare model
    summaries = summarizeByClass(trainingSet)
    # test model
    predictions = getPredictions(summaries, testSet)
    accuracy = getAccuracy(testSet, predictions)
    print('Accuracy: {0}%').format(accuracy)

```

```
main()
```

Running the example provides output like the following:

```
Split 768 rows into train=514 and test=254 rows
Accuracy: 76.3779527559%
```

Implementation Extensions

This section provides you with ideas for extensions that you could apply and investigate with the Python code you have implemented as part of this tutorial.

You have implemented your own version of Gaussian Naive Bayes in python from scratch.

You can extend the implementation further.

- **Calculate Class Probabilities**: Update the exam ce belonging to each class as a ratio. This can be ca belonging to one class, divided by the sum of the ch class. For example an instance had the probabilit likelihood of the instance belonging to class A is (
- **Log Probabilities**: The conditional probabilities fo When they are multiplied together they result in ve underflow (numbers too small to represent in Pyth f the probabilities together. Research and impleme
- **Nominal Attributes**: Update the implementation t and the summary information you can collect for each attribute is the ratio of category values for each class. Dive into the references for more information.
- **Different Density Function** (*bernoulli* or *multinomial*): We have looked at Gaussian Naive Bayes, but you can also look at other distributions. Implement a different distribution such as multinomial, bernoulli or kernel naive bayes that make different assumptions about the distribution of attribute values and/or their relationship with the class value.

Resources and Further Reading

This section will provide some resources that you can use to learn more about the Naive Bayes algorithm in terms of both theory of how and why it works and practical concerns for implementing it in code.

Problem

More resources for learning about the problem of predicting the onset of diabetes.

- Pima Indians Diabetes Data Set: This page provides access to the dataset files, describes the attributes and lists papers that use the dataset.
- Dataset File: The dataset file.
- Dataset Summary: Description of the dataset attributes.
- Diabetes Dataset Results: The accuracy of many standard algorithms on this dataset.

Code

This section links to open source implementations of Naive Bayes in popular machine learning libraries. Review these if you are considering implementing your own version of the method for operational use.

- Naive Bayes in Scikit-Learn: Implementation of naive bayes in the scikit-learn library.
- Naive Bayes documentation: Scikit-Learn documentation and sample code for Naive Bayes
- Simple Naive Bayes in Weka: Weka implementation of naive bayes

Books

You may have one or more books on applied machine learning. This section highlights the sections or chapters in common applied books on machine learning that refer to Naive Bayes.

- Applied Predictive Modeling, page 353
- Data Mining: Practical Machine Learning Tools an
- Machine Learning for Hackers, page 78
- An Introduction to Statistical Learning: with Applic
- Machine Learning: An Algorithmic Perspective, pa
- Machine Learning in Action, page 61 (Chapter 4)
- Machine Learning, page 177 (chapter 6)

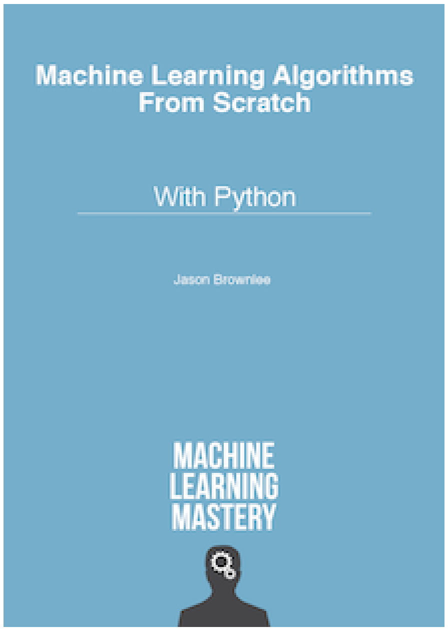
Next Step

Take action.

Follow the tutorial and implement Naive Bayes from scratch. Adapt the example to another problem. Follow the extensions and improve upon the implementation.

Leave a comment and share your experiences.

Update: Check out the follow-up on tips for using the naive bayes algorithm titled: “Better Naive Bayes: 12 Tips To Get The Most From The Naive Bayes Algorithm”



About Jason Brownlee

Jason Brownlee, Ph.D. is a machine learni
with modern machine learning methods via
View all posts by Jason Brownlee →

< Lessons Learned from Building Machine Learning Systems

Better Naive Bayes: 12 Tips To Get The Most From The Naive Bayes Algorithm >

165 Responses to *How To Implement Naive Bayes From Scratch in Python*

david jensen December 12, 2014 at 3:28 am # REPLY

Statistical methods should be developed from scratch because of misunderstandings. Thank you.

Anurag December 14, 2014 at 1:11 pm # REPLY

This is a wonderful article. Your blog is one of those blogs that I visit everyday. Thanks for sharing this stuff. I had a question about the programming language that should be used for building these algorithms from scratch. I know that Python is widely used because it's easy to write code by importing useful libraries that are already available. Nevertheless, I am a C++ guy. Although I am a beginner in practical ML, I have tried to write efficient codes before I started learning and implementing ML. Now I am aware of the complexities involved in coding if you're using C++: more coding is to be done than what is required in Python. Considering that, what language is your preference and under what situations? I know that it's lame to ask about preferences of programming language as it is essentially a personal choice. But still I'd like you to share your take on this. Also try to share the trade-offs while choosing these programming languages.

Thank you.

Jason Brownlee December 15, 2014 at 7:53 a

Thanks Anurag

SHRUTI April 17, 2018 at 12:33 am #

I am getting error while I try to implem
for classValue, classSummaries in summaries.i

AttributeError: 'list' object has no attribute 'iteri
When I try to run it,with your csv file,it says
ataset = list(lines)

Error: iterator should return strings, not bytes (did you open the file in text mode?)
What to do?

Jason Brownlee April 17, 2018 at 6:02 am #

REPLY ↰

The code was written for Python 2.7. Perhaps you're using Python 3?

fatih_conqueror April 29, 2018 at 6:25 am #

REPLY ↰

use items instead of iteritems. when you write erorro message to google, you can find lots of resolve like stackoverflow.

Alcides Schulz January 15, 2015 at 12:32 am #

REPLY ↰

Hi Jason, found your website and read it in one day. Thank you, it really helped me to understand ML and what to do.
I did the 2 examples here and I think I will take a look at scikit-learn now.
I have a personal project that I want to use ML, and I’ll keep you posted on the progress.

One small note on this post, is on the “1. Handle data” you refer to iris.data from previous post.

Thank you so much, example is really good to show how to do it. Please keep it coming.

Jason Brownlee January 15, 2015 at 7:43 am #

REPLY ↰

Thanks for the kind words Alcides.

Fixed the reference to the iris dataset.

vivek December 5, 2017 at 9:45 pm #

hi jason im not able to run the code a
‘pima-indians-diabetes.data.csv’”

Jason Brownlee December 6, 2017

Ensure you are running the code from the command line and that the data file and code file are in the same directory.

toolate January 22, 2015 at 2:16 am #

REPLY ↰

Hi Jason, still one more note on your post, is on the “1. Handle data” the flower measures that you refer to iris.data.

Jason Brownlee January 22, 2015 at 5:43 am #

REPLY ↰

Thanks, fixed!

Tamilselvan February 4, 2015 at 11:37 pm #

REPLY ↰

Great Article. Learned a Lot. Thanks. Thanks.

Abhinav kumar February 23, 2015 at 8:13 pm #

REPLY

thank u

Jason Brownlee February 24, 2015 at 7:40 am #

REPLY

You're welcome!

Roy March 7, 2015 at 2:53 pm #

Thanks for your nice article. I really appreciate

malini March 17, 2015 at 7:19 pm #

hello sir, plz tell me how to compare the data s

Isha March 21, 2015 at 5:40 pm #

Why does the accuracy change every time you run this code?
when i tried running this code every time it gave me different accuracy percentage in the range from 70-78%
Why is it so?
Why is it not giving a constant accuracy percent?

Harry April 9, 2015 at 8:37 am #

REPLY

As Splitting of dataset into testdata and traindata is done using a random function accuracy varies.

Nitin Ramesh October 27, 2017 at 4:00 pm #

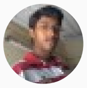
REPLY

I'm extremely new to this concept so please help me with this query.

The algorithm splits the dataset into the same top 67% and bottom 33% every single time.
The test-set is same on every single run.

So even though we use a random function on the top 67%(training set) to randomly index them.
A calculation like
((4+2)/6) and ((2+4)/6) will yield same result every-time.How is this yielding different result?

Is this something to do with the order of calculation in the Gaussian probability density function?



Abdul Salam December 22, 2017 at 10:43 pm #

REPLY

well.. the math calculations would come under to the view if you had the same sample taken again and again..

but the point here is.. we are taking the random data itself.. like.. the variables will not be same
in every row right.. so if u change the rows
taken that into consideration.. you can take
“UN-fluctuated accuracy” just in case if you

hope it helps..

Sheepsy90 March 25, 2015 at 8:12 pm #

Hey nice article – one question – why do you

Jason Brownlee March 26, 2015 at 5:33 am #

REPLY

Thanks!

N-1 is the sample (corrected or unbiased) standard deviation. See this wiki page.

Vaishali April 8, 2015 at 6:01 pm #

REPLY

Hey! Thanks a ton! This was very useful.
It would be great if you give an idea on how other metrics like precision and recall can be calculated.

Thanks!

Ashwin Perti April 24, 2015 at 5:28 pm #

REPLY

Sir

When I am running the same code in IDLE (python 2.7) – the code is working fine, but when I run the same code in eclipse. the error coming is:

1) warning – unused variable dataset
2) undefined variable dataset in for loop

Why this difference.

Melvin Tjon Akon May 21, 2015 at 1:46 am #

REPLY

Great post, Jason.
For a MBA/LLM, it makes naive bayes very easy to understand and to implement in legal coding. Looking forward to read more. Best, Melvin

Igor Balabine June 10, 2015 at 11:44 am #

Jason,

Great example. Thanks! One nit: “calculateProbability”
calculates Gaussian probability density – pdf value ma

Cheers,

-Igor

- Ruud - November 26, 2016 at 2:23 am #

Good point, thanks!

Alex Ubot July 2, 2015 at 10:06 pm #

REPLY

Hi Jason,

Fantastic post. I really learnt a lot. However I do have a question? Why don’ t you use the P(y) value in your calculateClassProbabilities() ?
If I understood the model correctly, everything is based on the bayes theorem :
P(y|x1….xn) = P(x1…..xn|y) * P(y) / P(x1……xn)
P(x1……xn) will be a constant so we can get rid of it.
Your post explain very well how to calculate P(x1……xn|y) (assumption made that x1…..xn are all independent we then have
P(x1……xn|y) = P(x1|y) * …. P(xn|y))
How about p(y) ? I assume that we should calculate the frequency of the observation y in the training set and then multiply it to probabilities[classValue] so that we have :
P(y|x1…..xn) = frequency(classValue) * probabilities[classValue]

Otherwise let' s assume that in a training set of 500 lines, we have two class 0 and 1 but observed 100 times 0 et 400 times 1. If we do not compute the frequency, then the probability may be biased, right ? Did I misunderstand something ? Hopefully my post is clear. I really hope that you will reply because I am a bit confused.

Thanks
Alex

Babu February 28, 2016 at 7:43 am #

REPLY

I have the same question – why is multiplying by p(y) is omitted?

Babu March 10, 2016 at 2:09 pm #

No Answer yet – no one on internet h

Just don't want to accept answers without unde

frong April 3, 2016 at 3:15 pm #

yeah，I have the same question t

is not so low when P(y) is missing? is it proving that bayes model is powerful.

gd April 7, 2016 at 2:27 am #

hi,

I believe this is because P(y) = 1 as classes are already segregated before calculating P(x1…xn|Y).

Can experts comment on this please?

Babu May 23, 2016 at 7:32 am #

There is huge bug in this implementation;

First of all the implementation using GaussianNB gives totally a different answer.
Why is no one is replying even after 2 months of this.

My concern is, there are so many more bad bayesians in a wrong concept.
My lead read this article and now he thinks I am wrong

At least the parameters are correct – something wrong with calculating probs.

```
def SplitXy(Xy):
Xy10=Xy[0:8]
Xy10 = Xy;
#print Xy10
#print "========"
zXy10=list(zip(*Xy10))
y= zXy10[-1]
del zXy10[-1]
z1=zip(*zXy10)
X=[list(t) for t in z1]
return X,y

from sklearn.naive_bayes import Gaus
X,y = SplitXy(trainingSet)
Xt,yt = SplitXy(testSet)

model = GaussianNB()
model.fit(X, y)

### Compare the models built by Pyth

print ("Class: 0")
for i,j in enumerate(model.theta_[0]):
print ("({:8.2f} {:9.2f} {:7.2f} )".format(j, m                                              )
print ("==> ", summaries[0][i])

print ("Class: 1")
for i,j in enumerate(model.theta_[1]):
print ("({:8.2f} {:9.2f} {:7.2f} )".format(j, model.sigma_[1][i], sqrt(model.sigma_[1][i])) , end="")
print ("==> ", summaries[1][i])

'''
Class: 0
( 3.18 9.06 3.01 )==> (3.1766467065868262, 3.0147673799630748)
( 109.12 699.16 26.44 )==> (109.11976047904191, 26.481293163857107)
( 68.71 286.46 16.93 )==> (68.712574850299404, 16.950414098038465)
( 19.74 228.74 15.12 )==> (19.742514970059879, 15.146913806453629)
( 68.64 10763.69 103.75 )==> (68.640718562874255, 103.90387227315443)
( 30.71 58.05 7.62 )==> (30.710778443113771, 7.630215185470916)
( 0.42 0.09 0.29 )==> (0.42285928143712581, 0.29409299864249266)
( 30.66 118.36 10.88 )==> (30.658682634730539, 10.895778423248444)
Class: 1
( 4.76 12.44 3.53 )==> (4.7611111111111111, 3.5365037952376928)
( 139.17 1064.54 32.63 )==> (139.17222222222222, 32.71833930500929)
( 69.27 525.24 22.92 )==> (69.272222222222226, 22.98209907114023)
( 22.64 309.59 17.60 )==> (22.638888888888889, 17.644143437447358)
( 101.13 20409.91 142.86 )==> (101.12777777777778, 143.2617649699204)
( 34.99 57.18 7.56 )==> (34.99388888
( 0.54 0.14 0.37 )==> (0.5354444444444439, 0.3702077209795522)
( 36.73 112.86 10.62 )==> (36.727777777777774, 10.653417924304598)
’’’
```

EL YAMANI May 22, 2016 at 8:57 am #

REPLY

Hello,

Thanks for this article , it is very helpful . I just have a remark about the probabilty that you are calculating which is P(x|Ck) and then you make predictions, the result will be biased since you don’t multiply by P(Ck) , P(x) can be omitted since it’s only a normalisation constant.

Anand July 20, 2015 at 9:12 pm #

Thanks a lot for this tutorial, Jason.

I have a quick question if you can help.

In the `separateByClass()` definition, I could not unde `vector` is an `int` type object.

If I try the same commands one by one outside the fun throws a `TypeError: 'int' object has no attrib`

Then how is it working inside the function?

I am sorry for my ignorance. I am new to python. Thank you.

Sarah August 26, 2015 at 5:50 pm #

REPLY

Hello Jason! I just wanted to leave a message to say thank you for the website. I am preparing for a job in this field and it has helped me so much. Keep up the amazing work!! 🙂

Jason Brownlee August 26, 2015 at 6:56 pm #

REPLY

You’re welcome! Thanks for leaving such a kind comment, you really made my day 🙂

Jaime Lopez September 7, 2015 at 8:52 am #

REPLY

Hi Jason,

Very easy to follow your classifier. I try it and works well on your data, but is important to note that it works just on numerical databases, so maybe one have to transform your data from categorical to numerical format.

Another thing, when I transformed one database, sometimes the algorithm find division by zero error, although I avoided to use that number on features and classes.

Any suggestion Jason?

Thanks, Jaime

syed belgam April 11, 2016 at 2:05 pm #

REPLY ↰

Thanks

eduardo September 28, 2015 at 1:32 pm #

It is by far the best material I've found , please

Jason Brownlee September 29, 2015 at 5:25

Thanks eduardo!

Thibroll September 29, 2015 at 9:11 pm #

REPLY ↰

Hello.

This is all well explained, and depicts well the steps of machine learning. But the way you calculate your P(y|X) here is false, and may lead to unwanted error.

Here, in theory, using the Bayes law, we know that : P(y|X) = P(y).P(X|y)/P(X). As we want to maximize P(y|X) with a given X, we can ignore P(X) and pick the result for the maximized value of P(y).P(X|y)

2 points remain inconsistent :
– First, you pick a gaussian distribution to estimate P(X|y). But here, you calculateProbability calculates the DENSITY of the function to the specific points X, y, with associated mean and deviation, and not the actual probability.
– The second point is that you don't take into consideration the calculation of P(y) to estimate P(y|X). Your model (with the correct probability calculation) may work only if all samples have same amount in every value of y (considering y is discret), or if you are lucky enough.

Anyway, despite those mathematical issue, this is a good work, and a god introduction to machine learning

mondet October 6, 2015 at 10:08 am #

REPLY

Thanks Jason for all this great material. One thing that i adore from you is the intellectual honesty, the spirit of collaboration and the parsimony.

In my opinion you are one of the best didactics exponents in the ML.

Thanks to Thibroll too. But i would like to have a real example of the problem in R, python or any other language.

Regards,

Emmanuel:.

Erika October 15, 2015 at 10:03 am #

Hi Jason,
I have trying to get started with machine learning and y
towards that. Thank you for your efforts! 🙂

Swagath November 9, 2015 at 5:35 pm #

i need this code in java.. please help me//

Sarah November 16, 2015 at 11:54 pm #

REPLY

I am working with this code – tweaking it here or there – have found it very helpful as I implement a NB from scratch. I am trying to take the next step and add in categorical data. Any suggestions on where I can head to get ideas for how to add this? Or any particular functions/methods in Python you can recommend? I've brought in all the attributes and split them into two datasets for continuous vs. categorical so that I can work on them separately before bringing their probabilities back together. I've got the categorical in the same dictionary where the key is the class and the values are lists of attributes for each instance. I'm not sure how to go through the values to count frequencies and then how to store this back up so that I have the attribute values along with their frequencies/probabilities. A dictionary within a dictionary? Should I be going in another direction and not using a similar format?

Emmanuel Nuakoh November 19, 2015 at 6:36 am #

REPLY

Thank you Jason, this tutorial is helping me with my implementation of NB algorithm for my PhD Dissertation. Very elaborate.

Anna January 14, 2016 at 2:32 am #

REPLY

Hi! thank you! Have you tried to do the same for the textual datasets, for example 20Newsgroups http://qwone.com/~jason/20Newsgroups/ ? Would appreciate some hints or ideas)

Randy January 16, 2016 at 4:15 pm #

REPLY

Great article, but as others pointed out there are some mathematical mistakes like using the probability density function for single value probabilities.

Meghna February 7, 2016 at 7:45 pm #

Thank you for this amazing article!! I impleme

these tutorials helped me so much!! 🙂

David February 7, 2016 at 11:17 pm #

I got an error with the first print statement, bec

(which returns None) before you're calling format, so in

print('Split {0} rows into train with {1} and test with {2}').format(len(dataset), train, test)

it should be

print('Split {0} rows into train with {1} and test with {2}'.format(len(dataset), train, test))

Anyway, thanks for this tutorial, it was really useful, cheers!

Kumar Ramanathan February 12, 2016 at 12:20 pm #

REPLY

Sincere gratitude for this most excellent site. Yes, I never learn until I write code for the algorithm. It is such an important exercise, to get concepts embedded into one's brain. Brilliant effort, truly !

Syed February 18, 2016 at 8:15 am #

REPLY

Just to test the algorithm, i change the class of few of the data to something else i.e 3 or 4, (last digit in a line) and i get divide by zero error while calculating the variance. I am not sure why. does it mean that this particular program works only for 2 classess? cant see anything which restricts it to that.

Takuma Udagawa March 20, 2016 at 1:19 pm #

REPLY

Hi, I'm a student in Japan.
It seems to me that you are calculating p(X1|Ck)*p(X2|Ck)*…*p(Xm|Ck) and choosing Ck such that this value would be maximum.
However, when I looked in the Wikipedia, you are supposed to calculate p(X1|Ck)*p(X2|Ck)*…*p(Xm|Ck)*p(Ck).
I don't understand when you calculated p(Ck).
Would you tell me about it?

jessie November 24, 2016 at 1:21 am #

Had the same thought, where's the prior c

Jorge January 9, 2018 at 12:00 pm #

Its seems it is a bug of this implement

I looked for the implementation of scikitlearn a

reference:
https://github.com/scikit-learn/scikit-learn/blob/e41c4d5e3944083328fd69aeacb590cbb78484da/sklearn/naive_bayes.py#L432

Jason Brownlee January 9, 2018 at 3:19 pm #

REPLY

You can add the prior easily. I left it out because it was a constant for this dataset.

Babu May 23, 2016 at 7:36 am #

REPLY

This is the same question as Alex Ubot above.

Calculating the parameters are correct.
but prediction implementation is incorrect.

Unfortunately this article comes up high and everyone is learning incorrect way of doing things I think

Swapnil June 10, 2016 at 1:21 am #

Really nice tutorial. Can you post a detailed implementation of RandomForest as well ? It will be very helpful for us if you do so.

Thanks!!

Jason Brownlee June 14, 2016 at 8:20 am #

REPLY

Great idea Swapnil.

You may find this post useful:
http://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/

sourena maroofi July 22, 2016 at 12:24 am #

thanks Jason…very nice tutorial.

Gary July 27, 2016 at 5:44 pm #

Hi,

I was interested in this Naive Bayes example and dow

However, when I try to run it in Pycharm IDE using Python 3.5 I get no end of run-time errors.

Has anyone else run the code successfully? And if so, what IDE/environment did they use?

Thanks

Gary

Sudarshan August 10, 2016 at 5:05 pm #

REPLY

Hi Gary,

You might want to run it using Python 2.7.

Sudarshan August 10, 2016 at 5:02 pm #

REPLY

Hi,

Thanks for the excellent tutorial. I've attempted to implement the same in Go.

Here is a link for anyone that's interested interested.

https://github.com/sudsred/gBay

Atlas August 13, 2016 at 6:40 am #

REPLY

This is AWESOME!!! Thank you Jason.

Where can I find more of this?

Alex August 20, 2016 at 4:34 pm #

That can be implemented in any language be

SAFA August 28, 2016 at 1:39 am #

Hello,
there is some errors in "def splitDataset"
in machine learning algorithm , split a dataset into train
(duplication) , so the index = random.randrange(len(co
for example " index = 0 192 1 2 0 14 34 56 1 ………
the spliting method must be done without duplication o

Krati Jain September 12, 2016 at 2:35 pm #

REPLY

This is a highly informative and detailed explained article. Although I think that this is suitable for Python 2.x versions for 3.x, we don't have 'iteritems' function in a dict object, we currently have 'items' in dict object. Secondly, format function is called on lot of print functions, which should have been on string in the print funciton but it has been somehow called on print function, which throws an error, can you please look into it.

upen September 16, 2016 at 5:01 pm #

REPLY

hey Jason
thanks for such a great tutorial im newbie to the concept and want to try naive bayes approach on movie-review on the review of a single movie that i have collected in a text file
can you please provide some hint on the topic how to load my file and perform positve or negative review on it

Jason Brownlee September 17, 2016 at 9:28 am #

REPLY

You might find some benefit in this tutorial as a template:
http://machinelearningmastery.com/machine-learning-in-python-step-by-step/

Abhis September 20, 2016 at 3:00 am #

REPLY

Would you please help me how i can implement naive bayes to predict student performance using their marks and feedback

Jason Brownlee September 20, 2016 at 8:35

I'm sorry, I am happy to answer your ques
don't have the capacity.

Vinay October 13, 2016 at 2:18 pm #

Hey Jason,

Thanks a lot for such a nice article, helped a lot in unde

i have a problem while running the script.
I get the below error

"

if (vector[-1] not in separated):
IndexError: list index out of range
"

can you please help me in getting it right?

Jason Brownlee October 14, 2016 at 8:58 am #

REPLY

Thanks Vinay.

Check that the data was loaded successfully. Perhaps there are empty lines or columns in your loaded data?

Viji October 20, 2016 at 8:57 pm #

Hi Jason,

Thank you for the wonderful article. U have used the '?'(testSet = [[1.1, '?'], [19.1, '?']]) in the test set. can u please tell me what it specifies

jeni November 15, 2016 at 9:11 pm #

REPLY ↰

please send me a code in text classification using naive bayes classifier in python . the data set classifies +ve,-ve or neutral

Jason Brownlee November 16, 2016 at 9:28 a

Hi jeni, sorry I don't have such an exampl

MLNewbie November 28, 2016 at 1:21 pm #

I am a newbie to ML and I found your website
available on the Internet. I bookmarked it and thanks fo
everyday going forward.

Jason Brownlee November 29, 2016 at 8:47 am #

REPLY ↰

Thanks, I'm glad you like it.

Anne January 7, 2017 at 6:58 pm #

REPLY ↰

```
def predict(summaries, inputVector):
probabilities = calculateClassProbabilities(summaries, inputVector)
bestLabel, bestProb = None, -1
for classValue, probability in probabilities.iteritems():
if bestLabel is None or probability > bestProb:
bestProb = probability
bestLabel = classValue
return bestLabel
```

why is the prediction different for these
summaries = {'A' : [(1, 0.5)], 'B': [(20, 5.0)]} –predicts A
summaries = {'0' : [(1, 0.5)], '1': [(20, 5.0)]} — predicts 0
summaries = {0 : [(1, 0.5)], 1: [(20, 5.0)]} — predicts 1

ML704 January 18, 2017 at 6:16 pm #

REPLY

Hi, can someone please explain the code snippet below:

```
def separateByClass(dataset):
separated = {}
for i in range(len(dataset)):
vector = dataset[i]
if (vector[-1] not in separated):
separated[vector[-1]] = []
separated[vector[-1]].append(vector)
return separated
```

What do curly brackets mean in separated = {}?
vector[-1] ?

Massive thanks!

Jason Brownlee January 19, 2017 at 7:30 am

Curly brackets are a set or dictionary in P
https://docs.python.org/3/tutorial/datastructures.htm

S February 27, 2017 at 8:47 am #

REPLY

I am trying to create an Android app which works as follows:

1) On opening the App, the user types a data in a textbox & clicks on search

2) The app then searches about the entered data via internet and returns some answer (Using machine learning algorithms)

I have a dataset of around 17000 things.

Can you suggest the approach? Python/Java/etc…? Which technology to use for implementing machine learning algorithm & for connecting to dataset? How to include the dataset so that android app size is not increased?

Basically, I am trying to implement an app described in a research paper.

I can implement machine learning(ML) algorithms in Python on my laptop for simple ML examples. But, I want to develop an Android app in which the data entered by user is checked from a web site and then from a "data set (using ML)" and result is displayed in app based on both the comparisons. The problem is that the data is of 40 MB & how to reflect the ML results from laptop to android app?? By the way, the dataset is also available online. Shall I need a server? Or, can I use

Which python server should I use? I would also need to check the data entered from a live website. Can I connect my Android app to live server and localhost simultaneously? Is such a scenario obvious for my app? What do you suggest? Is Anaconda software sufficient?

Jason Brownlee February 28, 2017 at 8:08 am #

REPLY

Sorry I cannot make any good suggestions, I think you need to talk to some app engineers, not ML people.

Roy March 1, 2017 at 4:20 am #

Hello Jason,

lines = csv.reader(open(filename, “rb”))
IOError: [Errno 2] No such file or directory: ‘pima-indiar

I have the csv file downloaded and its in the same folde

What should I do about this?

Jason Brownlee March 1, 2017 at 8:44 am #

Hi Roy,

Confirm that the file name in your directory exactly matches the expectation of the script.

Confirm you are running from the command line and both the script and the data file are in the same directory and you are running from that directory.

If using Python 3, consider changing the ‘rb’ to ‘rt’ (text instead of binary).

Does that help?

Jordan March 1, 2017 at 8:40 pm #

REPLY

Hi Jason. Great Tutorial! But, why did you leave the P(Y) in calculateClassProbability() ? The prediction produces in my machine is fine… But some people up there have mentioned it too that what you actually calculate is probability density function. And you don’t even answer their question.

Ali March 7, 2017 at 6:21 am #

REPLY

Hi Jason,
Can you please help me fixing below error, The split is working but accuracy giving error

```
Split 769 rows into train=515 and test=254 rows
Traceback (most recent call last):
File “indian.py”, line 100, in
main()
File “indian.py”, line 95, in main
summaries = summarizeByClass(trainingSet)
File “indian.py”, line 45, in summarizeByClass
separated = separateByClass(dataset)
File “indian.py”, line 26, in separateByClass
if (vector[-1] not in separated):
IndexError: list index out of range
```

shankru Guggari March 13, 2017 at 9:52 pm #

Class wise selection of training and testing da
For Example
In Iris Dataset : Species Column we have classes calle

I want to select 80% of data from each class values.

Advance thanks
Shankru
shankar286@gmail.Com

Jason Brownlee March 14, 2017 at 8:17 am #

REPLY ↰

You can take a random sample or use a stratified sample to ensure the same mixture of classes in train and test sets.

Namrata March 19, 2017 at 5:33 pm #

REPLY ↰

error in Naive Bayes code
IndexError:list index out of range

velu March 28, 2017 at 4:25 pm #

REPLY ↰

hai guys
i am velmurugan iam studying annauniversity tindivanam

i have a code for summarization for english description in java

Kamal March 29, 2017 at 10:24 am # REPLY

Hi Jason,

This example works. really good for Naive Bayes, but I was wondering what the approach would be like for joint probability distributions. Given a dataset, how to construct a bayesian network in Python or R?

Joelon johnson March 30, 2017 at 7:20 pm #

Hello Jason,

Joelon here. I am new to python and machine learning

above script. Is it possible I send you screenshots of th

Jason Brownlee March 31, 2017 at 5:53 am

What problem are you getting exactly?

Bill April 2, 2017 at 10:52 pm # REPLY

Hello Jason !

Thank you for this tutorial.
I have a question: what if our x to predict is a vector? How can we calculate the probability to be in a class (in the function calculateProbability for example) ?

Thank you

Jason Brownlee April 4, 2017 at 9:07 am # REPLY

Not sure I understand Bill. Perhaps you can give an example?

Asmita Singh April 9, 2017 at 12:33 pm # REPLY

HI Jason,
Thanks for such a wonderful article. Your efforts are pr

single value probabilities. Which part of code requires any smoothening.

Jason Brownlee April 9, 2017 at 3:01 pm #

REPLY

Sorry, I'm not sure I understand your question. Perhaps you can restate it?

Mohammed Ehteramuddin April 11, 2017 at 12:51 am #

REPLY

Hello Jason,
First of all I thank you very much for such a nice tutoria

I have a quick question for you if you could find some c

Question: Why is that summarizeByClass(dataset) wor e
dataset in your example and does not work with the dif
[9,7,3], [12,9,0],[29,0,0]]

I guess it has to work for all the different possible datas

Thanks,
Mohammed Ehteramuddin

Jason Brownlee April 11, 2017 at 9:33 am #

It should work with any dataset as long as the last variable is the class variable.

Mohammed Ehteramuddin April 12, 2017 at 7:49 pm #

REPLY

Oh! you mean the last variable of the dataset (input) can not be other that the two values that we desire to classify the data into, in our case it should either be 0 or 1.

Thanks you very much.

Jason Brownlee April 13, 2017 at 9:58 am #

REPLY

Yes.

Salihins Gund April 19, 2017 at 6:36 am #

REPLY

What is `correct` variable refers in getAccuracy function? Can you elaborate it more?

Salihins Gund April 19, 2017 at 7:04 am #

REPLY

Sorry that was wrong question.

Edit:
Ideally the Gaussian Naive Bayes has lambda (threshold) value to set boundary. I was wondering which part of this code include the threshold?

Jason Brownlee April 19, 2017 at 7:56 a

I do not include threshold.

way win April 21, 2017 at 3:00 am #

Can you provide an extension to the data. I ca

Jason Brownlee April 21, 2017 at 8:41 am #

Here is a direct link to the data file:
https://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/pima-indians-diabetes.data

Sometimes the UCI ML repo will go down for a few hours.

way win April 21, 2017 at 10:40 am #

REPLY

I trying running the program but it keeps saying
File “nb.py”, line 102, in
main()
File “nb.py”, line 92, in main
dataset = loadCsv(filename)
File “nb.py”, line 11, in loadCsv
dataset[i] = [float(x) for x in dataset[i]]
ValueError: invalid literal for float(): 1,85,66,29,0,26.6,0.351,31,0

Mian Saeed Akbar May 9, 2017 at 10:55 am #

REPLY ↩

Hi Jason…!
Thank You so much for coding the problem in a very clear way. As I am new in machine learning and also I have not used Python before therefore I feel some difficulties in modifying it. I want to include the serial number of in data set and then show which testing example (e.g. example # 110) got how much probability form class 0 and for 1.

Jason Brownlee May 10, 2017 at 8:38 am #

REPLY ↩

You might be better off using Weka:
http://machinelearningmastery.com/start-here/#we

Blazej May 27, 2017 at 9:33 pm #

Hi Jason,

I encountered a problem at the beginning. After loading

```
filename = ‘pima-indians-diabetes.data.csv’
dataset = loadCsv(filename)
print(‘Loaded data file {0} with {1} rows’).format(filenam
```

I get the Error: “iterator should return strings, not bytes (did you open the file in text mode?)”

btw i am using python 3.6

thank you for the help

Jason Brownlee June 2, 2017 at 12:02 pm #

REPLY ↩

Change the loading of the file from binary to text (e.g. ‘rt’)

Marcus June 10, 2017 at 9:13 am #

REPLY ↩

This code is not working with Python 3.16 :S

Marcus June 10, 2017 at 9:14 am #

REPLY ↩

3.6

Jason Brownlee June 11, 2017 at 8:19 am # REPLY

Thanks Marcus.

Marcus June 13, 2017 at 10:28 am # REPLY

Is there a way you can try to fix and make it work with 3.6 maybe Jason?

Marcus June 15, 2017 at 2:23 am #

Jason can you explain what this c

Jason Brownlee June 15, 201

Yes, see here:
http://machinelearningmastery.com/na

Darmawan Utomo December 13, 2017 at 9:16 pm # REPLY

I run the code in python 3.6.3 and here are the corrections:

————————————————————————–

1. change the “rb” to “rt”

2. print(‘Split {0} rows into train={1} and test={2} rows’.format(len(dataset), len(trainingSet), len(testSet)))

3. change for each .iteritems() to .items()

4. print(‘Accuracy: {0}%’.format(accuracy))

Here are some of the results:

Split 768 rows into train=514 and test=254 rows
Accuracy: 71.25984251968504%

Split 768 rows into train=514 and test=254 rows
Accuracy: 76.77165354330708%

Jason Brownlee December 14, 2017 at 5:36 am #

Thanks for the tips.

Guy Person June 14, 2017 at 4:34 am #

REPLY

The code in this tutorial is riddled with error after error… The string output formatting isn't even done right for gods sakes!

Person Guy June 15, 2017 at 7:56 am #

This was written in 2014, the python docu

have been updated

J Wang June 15, 2017 at 8:02 am #

Hey Jason, I really enjoyed the read as it was

like myself understandable overall. However, like Marc t

out how to edit the parts that have new syntax in pytho

Also, this version utilized the Gaussian probability density function, how would we use the other ones, would the math be different or the code?

Giselle July 12, 2017 at 10:53 am #

REPLY

Hi Jason, thank you for this post it's super informative, I just started college and this is really easy to follow! I was wondering how this could be done with a mixture of binary and categorical data. For example, if I wanted to create a model to determine whether or not a car would break down and one category had a list of names of 10 car parts while another category simply asked if the car overheated (yes or no). Thanks again!

Jason Brownlee July 13, 2017 at 9:46 am #

REPLY

Absolutely.

Giselle July 14, 2017 at 3:28 am #

REPLY ↰

Would the categorical data have to be preprocessed?

Jason Brownlee July 14, 2017 at 8:34 am #

REPLY ↰

No. Naive Bayes is much simpler on categorical data:
http://machinelearningmastery.com/naive-bayes-tutorial-for-machine-learning/

Thomas July 14, 2017 at 7:53 am #

Hi! Thanks for this helpful article. I had a quick
should the denominator be multiplied by the variance i

i.e. should

```
return (1 / (math.sqrt(2 * math.pi) * stdev)
```

instead be:

```
return (1 / (math.sqrt(2 * math.pi) * math.p
```

Rezaul Karim August 6, 2017 at 5:20 pm #

REPLY ↰

Hi Jason. I visited ur site several times. This is really helpful. I look for SVM implementation in python from scratch like the way you implemented Naive Bayes here. Can u provide me SVM code??

Jason Brownlee August 7, 2017 at 8:40 am #

REPLY ↰

Yes, I provide it in my book here:
https://machinelearningmastery.com/machine-learning-algorithms-from-scratch/

Charmaine Ponay August 17, 2017 at 10:06 pm #

REPLY ↰

Thank you so much Mr. Brownlee. I would like to ask your permission if I can show my students your implementations? They are very easy to understand and follow. Thank you very much again 🙂

Jason Brownlee August 18, 2017 at 6:20 am # REPLY

No problem as long as you credit the source and link back to my website.

Charmaine Ponay August 22, 2017 at 3:19 pm # REPLY

thank you very much 🙂

THAMILSELVAM B September 9, 2017 at 2:30 am

Very good basic tutorial. Thank you.

Jason Brownlee September 9, 2017 at 11:59

Thanks.

Chandar September 25, 2017 at 10:48 pm #

Hi Jason,
I would have exported my model using joblib and as I have converted the categorical data to numeric in the training data-set to develop the model and now I have no clue on how to convert the a new categorical data to predict using the trained model.

Jason Brownlee September 26, 2017 at 5:38 am # REPLY

New data must be prepared using the same methods as data used to train the model.

Sometimes this might mean keeping objects/coefficients used to scale or encode input data along with the model.

Narendra December 4, 2017 at 3:46 am # REPLY

where/what is learning in this code. I think it is just naive bayes classification. Please specify the learning.

Jason Brownlee December 4, 2017 at 7:59 am #

REPLY

Good question, the probabilities are "learned" from data.

Christian December 18, 2017 at 1:01 am #

REPLY

Great example. You are doing a great work thanks. Please am working on this example but i am confused on how to determine attribute relevance analysis. That is how do i determine which attribute is (will) be relevant for my model.

Jason Brownlee December 18, 2017 at 5:26 a

Perhaps you could look at the independen

Christian December 18, 2017 at 3:00 pm #

thanks very much. Grateful

SUIMEE December 25, 2017 at 10:05 pm #

Can you please send me the code for pedestrian detection using HOG and NAIVE BAYES?

Jason Brownlee December 26, 2017 at 5:15 am #

REPLY

Sorry, I cannot.

Jasper January 25, 2018 at 1:14 pm #

REPLY

what does this block of code do

```
while len(trainSet) < trainSize:
index = random.randrange(len(copy))
trainSet.append(copy.pop(index))
return [trainSet, copy]
```

Jason Brownlee January 26, 2018 at 5:37 am # REPLY

Selects random rows from the dataset copy and adds them to the training set.

Scott January 27, 2018 at 9:54 am # REPLY

Jason:

I am very happy with this implementation! I used it as inspiration for an R counterpart. I am unclear about one thing. I understand the training set mean and sd are parameters used to evaluate the test set, but I don't know why that works lol.

How does evaluating the GPDF with the training set da
model? I may be confusing myself by interpreting "train

I think of train as repetitiously doing something multiple
these iterations ultimately produce some catalyst for hi
iteration of defining the training set's mean and sd. Not
that is the case.

Any help is truly, genuinely appreciated!

Scott Bishop

Jason Brownlee January 28, 2018 at 8:20 am

Here, the training data provides the basis for estimating the probabilities used by the model.

som February 5, 2018 at 4:14 am # REPLY

Hi Jason,
Sometimes I am getting the "ZeroDivisionError: float division by zero" when I am running the program

```
# -*- coding: utf-8 -*-
"""
Created on Sun Feb  4 07:30:57 2018

@author: SONAM
"""

# Example of Naive Bayes implemented from Scratch in Python
from __future__ import division
import csv
import random
import math


def loadCsv(filename):
    lines = csv.reader(open(filename, "rt"))
    dataset = list(lines)
    for i in range(len(dataset)):
        dataset[i] = [float(x) for x in dataset[i]]
    return dataset

def splitDataset(dataset, splitRatio):
    trainSize = int(len(dataset) * splitRatio)
    trainSet = []
    copy = list(dataset)
    while len(trainSet)  bestProb:
            bestProb = probability
            bestLabel = classValue
    return bestLabel

def getPredictions(summaries, testSet):
    predictions = []
    for i in range(len(testSet)):
        result = predict(summaries, testSet
        predictions.append(result)
    return predictions

def getAccuracy(testSet, predictions):
    correct = 0
    for i in range(len(testSet)):
        if testSet[i][-1] == predictions[i
            correct += 1
    return (correct/float(len(testSet))) *

def main():
    filename = 'Data.csv'
    splitRatio = 0.67
    dataset = loadCsv(filename)
    trainingSet, testSet = splitDataset(dat
    print('Split {0} rows into train={1} ar
    # prepare model
    summaries = summarizeByClass(trainingSet)
    # test model
    predictions = getPredictions(summaries, testSet)
    accuracy = getAccuracy(testSet, predictions)
    print('Accuracy: {0}%'.format(accuracy))


main()
```

shadhana February 23, 2018 at 4:12 am # REPLY

is there a way to get the elements that fall under each class after the classification is done

Jason Brownlee February 23, 2018 at 12:03 pm # REPLY

Do you mean a confusion matrix:
https://machinelearningmastery.com/confusion-matrix-machine-learning/

Christian Post March 3, 2018 at 12:42 am #

REPLY

I added this function:

```
def confusionMatrix(testSet, predictions):
    #TP = true positive, FP = false positive
    #TN = true negative, FN = false negative
    TP, FP, TN, FN = 0,0,0,0
    for i in range(len(testSet)):
        if testSet[i][-1] == 1:
            if testSet[i][-1] == predictions[i]:
                TP +=1
            else:
                FP +=1
        else:
            if testSet[i][-1] == predictions[i]:
                TN +=1
            else:
                FN +=1
    return[TP, FP, TN, FN]
```

There is probably a more elegant way to write this

The returning array lets you calculate all sorts of
likelihood ratio etc.

Christian Post March 9, 2018 at 2:01 am

Whoops I realized I mixed something
the outer if-clause checks the true condition. I
Anyways, looks like the confusion matrix lives up to its name.

Christian Post March 3, 2018 at 12:48 am #

REPLY

Hello Jason,

first of all, thanks for this blog. I learned a lot both on python (which I am pretty new to) and also this specific classifier.

I tested this algorithm on a sample database with cholesterol, age and heart desease, and got better results than with a logistic regression.
However, since age is clearly not normally distributed, I am not sure if this result is even legit.
Could you explain how I can change the calculateProbability function to a different distribution?

Oh, and also: How can I add code tags to my replies so that it becomes more readable?

Jason Brownlee March 3, 2018 at 8:17 am #

REPLY

This code is for learning purposes, I would recommend using the built-in functions in scikit-learn for real projects:
https://machinelearningmastery.com/start-here/#python

Nil March 7, 2018 at 8:10 pm #

REPLY

Hi DR. Jason,

It is a very good post.
I did not see the K Fold Cross Validation in this post like I saw in your post of Neural Network from scratch.
Does it mean that Naive Bayes does not need K Fold Cross Validation? Or does not work with K Fold CV?
It is because I am trying to use K Fold CV with Naive B
to split data by class to make some calculations, we fin
classification dataset (but we have one K Fold CV func
I am facing serious difficulties to understand K Fold CV
depends on the classifier we are using.

If you have some answer or tips to this approach (valic
) please let me know.

Regards.

Jason Brownlee March 8, 2018 at 6:22 am #

It is a good idea to use CV to evaluate algorithms including naive bayes.

y April 23, 2018 at 11:34 pm #

REPLY

If the variance is equal to 0, how to deal with？

Jason Brownlee April 24, 2018 at 6:34 am #

REPLY

If the variance is 0, then all data in your sample has the same value.

y April 25, 2018 at 5:43 pm #

REPLY

```
def calculateProbability(x, mean, stdev):
    exponent = math.exp(-(math.pow(x-mean,2)/(2*math.pow(stdev,2))))
    return (1 / (math.sqrt(2*math.pi) * stdev)) * exponent
```

if 'stdev' is 0 , how to deal with it?

Jason Brownlee April 26, 2018 at 6:23 am #

REPLY

If stdev is 0 it means all values are the same and that feature should be removed.

kk May 29, 2018 at 4:40 pm #

REPLY

How can I retrain my model, without training it from scratch again?
Like if I got some new label for instances, how to appro

Jason Brownlee May 30, 2018 at 6:33 am #

Many models can be updated with new da

With naive bayes, you can keep track of the freque
PDF/mass function for continuous values and upda

Leave a Reply

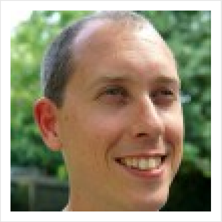
Name (required)

Email (will not be published) (required)

Website

SUBMIT COMMENT

Welcome to Machine Learning Mastery



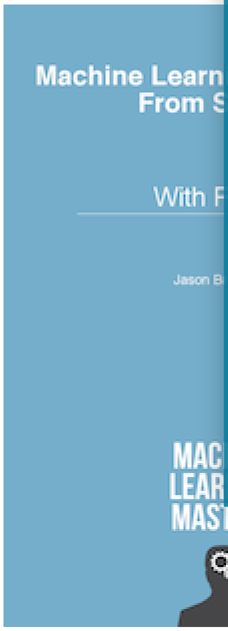
Hi, I'm Jason Brownlee, Ph.D.
My goal is to make developers like *YOU* awesome at applied machine learning.

Read More

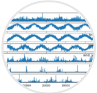
Code Algorithms From Scratch in Python

Discover how to code top machine learning algorithms from first principles with Python.

Code Machine Learning Algo



POPULAR



Multivariate Time Series Forecasting with LSTMs in Keras
AUGUST 14, 2017

How to Develop a Deep Learning Photo Caption Generator from Scratch
NOVEMBER 27, 2017

How to Use Word Embedding Layers for Deep Learning with Keras
OCTOBER 4, 2017

How to Define an Encoder-Decoder Sequence-to-Sequence Model for Neural Machine Translation in Keras
OCTOBER 26, 2017

How to Develop a Neural Machine Translation System from Scra

JANUARY 10, 2018

How to Develop an Encoder-Decoder Model with Attention for Sequence-to-Sequence Prediction in Keras
OCTOBER 17, 2017

Why One-Hot Encode Data in Machine Learning?
JULY 28, 2017

How to Reshape Input Data for Long Short-Term Memory Networks in Keras
AUGUST 30, 2017

How to Develop an Encoder-Decoder Model for Seq
NOVEMBER 2, 2017

What is the Difference Between Test and Validation
JULY 14, 2017

You might also like…

- Your First Machine Learning Project in Python
- Your First Neural Network in Python
- How to Setup a Python for Machine Learning
- Your First Classifier in Weka
- Your First Model for Time Series Forecasting

Privacy | Disclaimer | Terms of Service | Search | Newsletter | Contact | About