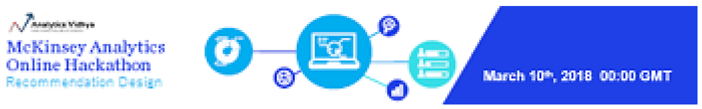

Analytics Vidhya
McKinsey Analytics Online Hackathon
Recommendation Design
March 10th, 2018 00:00 GMT

(https://datahack.analyticsvidhya.com/contest/mckinsey-analytics-online-hackathon-recommendation/?utm_source=AVhome_top)

6 Easy Steps to Learn Naive Bayes Algorithm (with codes in Python and R)

MACHINE LEARNING (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/MACHINE-LEARNING/) PYTHON (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/PYTHON-2/) R (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/R/)

ɔm/sharer.php?u=https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-
%20to%20Learn%20Naive%20Bayes%20Algorithm%20(with%20codes%20in%20Python%20and%20R)) (https://twitter.com/home?
%20Learn%20Naive%20Bayes%20Algorithm%20(with%20codes%
.google.com/share?url=https://www.analyticsvidhya.com/blog/201
n/blog/2017/09/naive-bayes-explained/&media=https://s3-ap-sou
ion=6%20Easy%20Steps%20to%20Learn%20Naive%20Bayes%20

Woohoo!

Analytics Vidhya Android App Is Coming Soon!

Leave your e-mail id to get early beta access

Enter your email id | SUBMIT

(https://acadgild.com/bi
aff_id=6014&source=AV&account=paid&utm_s

Note: This article was originally published on Sep 13th, 2015 and updated on Sept 11th, 2017

Introduction

Here's a situation you've got into:

You are working on a classification problem and you have generated your set of hypothesis, created features and discussed the importance of variables. Within an hour, stakeholders want to see the first cut of the model.

What will you do? You have hundreds of thousands of data points and quite a few variables in your training data set. In such situation, if I were at your place, I would have used '**Naive Bayes**', which can be extremely fast relative to other classification algorithms. It works on Bayes theorem of probability to predict the class of unknown data set.

In this article, I'll explain the basics of this algorithm, so that next time when you come across large data sets, you can bring this algorithm to action. In addition, if you are a newbie in Python or R (https://www.analyticsvidhya.com/learning-paths-data-science-business-analytics-business-intelligence-big-data/learning-path-data-science-python/), you should be overwhelmed by the presence of available codes in this article.

Analytics Vidhya
DataFest Closing Ceremony
Come & meet our Datavengers
Time: 9 PM to 10 PM (IST) (Webinar) | 10 May 2017
DATAFEST 2017
Register Now

(https://datahack.analyticsvidhya.com/contest/datafest-closing-ceremony/)

Table of Contents

Woohoo!

Analytics Vidhya Android App Is Coming Soon!

Leave your e-mail id to get early beta access

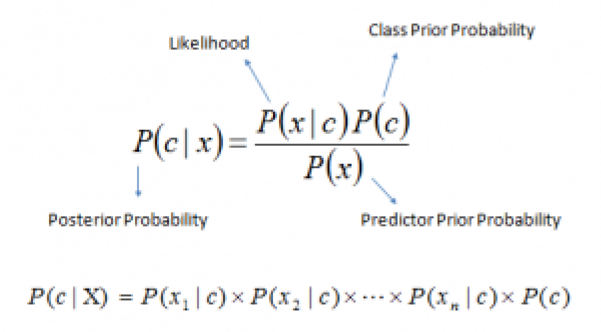
Enter your email id SUBMIT

What is Naive Bayes algorithm?

It is a classification technique based on Bayes' Theorem (https://en.wikipedia.org/wiki/Bayes%27_theorem) with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

(https://www.analyticsvidhya.com/wp-content/uploads/2015/09/Bayes_rule-300x172.png)Above,

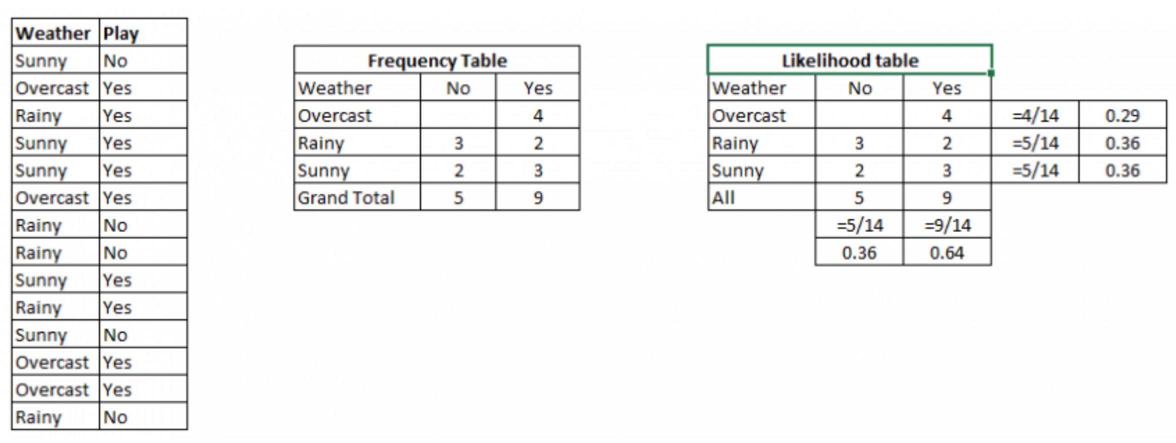
- *P*(*c|x*) is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- *P*(*c*) is the prior probability of *class*.
- *P*(*x|c*) is the likelihood which is the proba
- *P*(*x*) is the prior probability of *predictor*.

How Naive Bayes algorithm w

Let's understand it using an example. Belo
target variable 'Play' (suggesting possibilit
will play or not based on weather conditior

Step 1: Convert the data set into a frequenc

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



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

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

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

(https://www.analyticsvidhya.com/wp-content/uploads/2015/08/Bayes_41.png)

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

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

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

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

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

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

Naive Bayes uses a similar method to pred
attributes. This algorithm is mostly used in
classes.

What are the Pros and Cons c

Pros:

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

Cons:

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

4 Applications of Naive Bayes Algorithms

- **Real time Prediction:** Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
- **Multi class Prediction:** This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
- **Text classification/ Spam Filtering/ Sentiment Analysis:** Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)
- **Recommendation System:** Naiv (https://en.wikipedia.org/wiki/Collabora uses machine learning and data mining user would like a given resource or not

How to build a basic model us

Again, scikit learn (python library) will help here to build a Naive Bayes model in Python. There are three types of Naive Bayes model under scikit learn library:

- **Gaussian: (http://scikit-learn.org/stable/modules/naive_bayes.html)** It is used in classification and it assumes that features follow a normal distribution.

- **Multinomial (http://scikit-learn.org/stable/modules/naive_bayes.html):** It is used for discrete counts. For example, let's say, we have a text classification problem. Here we can consider bernoulli trials which is one step further and instead of "word occurring in the document", we have "count how often word occurs in the document", you can think of it as "number of times outcome number x_i is observed over the n trials".

- **Bernoulli (http://scikit-learn.org/stable/modules/naive_bayes.html):** The binomial model is useful if your feature vectors are binary (i.e. zeros and ones). One application would be text classification with 'bag of words' model where the 1s & 0s are "word occurs in the document" and "word does not occur in the document" respectively.

Based on your data set, you can choose any of above discussed model. Below is the example of Gaussian model.

Python Code

```
#Import Library of Gaussian Naive Bayes model
from sklearn.naive_bayes import GaussianNB
import numpy as np

#assigning predictor and target variables
x= np.array([[-3,7],[1,5], [1,2], [-2,0], [
7]])
Y = np.array([3, 3, 3, 3, 4, 3, 3, 4, 3, 4,
#Create a Gaussian Classifier
model = GaussianNB()

# Train the model using the training sets
model.fit(x, y)

#Predict Output
predicted= model.predict([[1,2],[3,4]])
print predicted

Output: ([3,4])
```

R Code:

```
require(e1071) #Holds the Naive Bayes Classifier
Train <- read.csv(file.choose())
Test <- read.csv(file.choose())

#Make sure the target variable is of a two-class classification problem only

levels(Train$Item_Fat_Content)

model <- naiveBayes(Item_Fat_Content~., data = Train)
class(model)
pred <- predict(model,Test)
table(pred)
```

Above, we looked at the basic Naive Bayes
tuning parameters and handle assumptic
performance of Naive Bayes Model.
(http://www.inf.ed.ac.uk/teaching/courses
details on Text classification using Naive Ba

Tips to improve the power of Naive Bayes Model

Here are some tips for improving power of Naive Bayes Model:

- If continuous features do not have normal distribution, we should use transformation or different methods to convert it in normal distribution.
- If test data set has zero frequency issue, apply smoothing techniques "Laplace Correction" to predict the class of test data set.
- Remove correlated features, as the highly correlated features are voted twice in the model and it can lead to over inflating importance.
- Naive Bayes classifiers has limited options for parameter tuning like alpha=1 for smoothing, fit_prior=[True|False] to learn class prior probabilities or not and some other options (look at detail here (http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html#sklearn.naive_bayes.Mul I would recommend to focus on your pre-processing of data and the feature selection.
- You might think to apply some *classifier combination technique like* ensembling, bagging and boosting but these methods would not help. Actually, "ensembling, boosting, bagging" won't help since their purpose is to reduce variance. Naive Bayes has no variance to minimize.

End Notes

In this article, we looked at one of the supervised machine learning algorithm "Naive Bayes" mainly used for classification. Congrats, if you've thoroughly & understood this article, you've already taken you first step to master this algorithm. From here, all you need is practice.

Further, I would suggest you to focus more on data pre-processing and feature selection prior to applying Naive Bayes algorithm.0 In future post, I will discuss about text and document classification using naive bayes in more det

Did you find this article helpful? Please s below.

Learn (https://www.analyticsvidhya.
(https://discuss.analyticsvidhya.com
(https://datahack.analyticsvidhya.co
(https://www.analyticsvidhya.com/jo