[Year]

[Document title]

Name: MD: Aynul Islam

Chinese Name: 叶子

Student Id: 4420190030

Subject: Big Data

Topic: Naive Bayes Algorithm

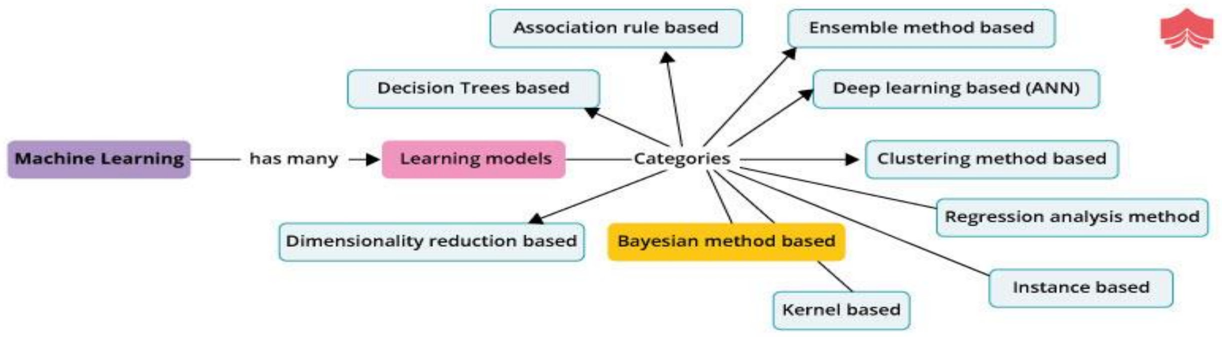
Table of Contents

Abstract

Naive Bayes algorithm is one of the most effective methods in the field of text classification, but only in the large training sample set can it get a more accurate result. The requirement of a large number of samples not only brings heavy work for previous manual classification, but also puts forward a higher request for storage and computing resources during the computer post-processing. This paper mainly studies Naïve Bayes classification algorithm based on Poisson distribution model, and the experimental results show that this method keeps high classification accuracy even in small sample set.

Introduction

Naive Bayes models are a group of extremely fast and simple classification algorithms that are often suitable for very high-dimensional datasets. Since they are so quick and have not many tunable boundaries, they wind up being extremely valuable as a no fuss pattern for an order issue. Naive Bayes is a basic however shockingly incredible probabilistic AI calculation utilized for prescient displaying and order assignments. “ Some regular uses of Naive Bayes are spam separating, conclusion forecast, characterization of reports, and so forth its anything but a famous calculation basically on the grounds that it tends to be effectively written in code and forecasts can be made genuine Quick which thus builds the adaptability of the arrangement.” [1] The Naive Bayes calculation is generally viewed as the calculation of decision for reasonable based applications for the most part in situations where prompt reactions are needed for client’s solicitations.



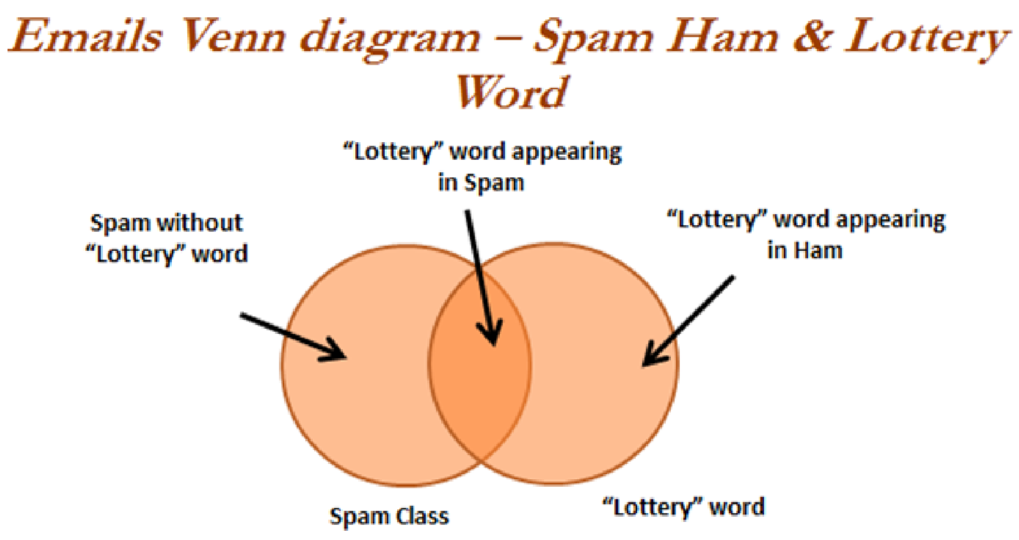
Bayesian learning is an administered learning method where the objective is to construct a model of the dispersion of class marks that have a substantial meaning of the objective trait. Innocent Bayes depends on applying Bayes' hypothesis with the credulous presumption of autonomy among every single pair of highlights.

Probability Fundamentals

Now we have at least an idea on how a Naive Bayes algorithm works. Likelihood of an occasion is characterized as the proportion of the quantity of trails of the event of an occasion to the complete number of trails. For instance, in the event that we arbitrarily pick 10 balls from a sack which contains both red and blue balls and 4 out of 10 are discovered to be red balls, then, at that point the likelihood of red balls is 4/10 or 0.4. Nonetheless, the absolute likelihood of the relative multitude of results should summarize to 100%.[2]

Joint Probability

Joint probability can be defined as a chance or likelihood that two events will happen at the same time although they remain independent on each other. let us assume 10 percent of all emails are spam and 5 percent are the emails that contain the word 'lottery'. Now if we have to calculate the joint probability of both that is, probability of the message being spam and the word 'lottery' occurring in a message, we can do the following. This also implies that the outcome of one event cannot influence the outcome of others.


Emails Venn diagram – Spam Ham & Lottery Word
"Lottery" word appearing in Spam
Spam without "Lottery" word
"Lottery" word appearing in Ham
Spam Class
"Lottery" word

p(spam ∩lottery) = p(spam) * p(lottery) = 0.1 * 0.05 = 0.005

Now we can say that of all the messages, 5 percent are spam having the word 'lottery'. This much is enough for now. There is a lot left.[3]

Conditional Probability

Conditional probability is actually what we are looking for. Naïve Bayes which works on Bayes theorem is totally based on conditional probability which is the probability of the outcome of an event given that another event has already occurred. If A and B are two events, then the conditional probability of both the events is given by

$$P(A\backslash B) = \frac{P(B\backslash A) * P(A)}{P(B)} = \frac{P(A \cap B)}{P(B)}$$

Now if we again consider the email classification example, considering prior probability of the message being a spam and the marginal likelihood of the word 'lottery' occurring in all messages, then the conditional probability can be calculated as

$$\underset{\text{Posterior Probability}}{P(spam \backslash lottery)} = \frac{\overset{\text{Likelihood}}{P(lottery \backslash spam)} * \overset{\text{Prior Probability}}{P(spam)}}{\underset{\text{Marginal Likelihood}}{P(lottery)}}$$

Naive Bayes only assumes one fact that one event in a class should be independent of another event belonging to the same class. The algorithm also assumes that the predictors have an equal effect on the outcomes or responses in the data.[4]

Probabilistic model

Uniquely, guileless Bayes is a restrictive likelihood model: given an issue case to be grouped, addressed by a vector {\displaystyle \mathbf {x} =(x_{1},\ldots ,x_{n})}{\displaystyle \mathbf {x} =(x_{1},\ldots ,x_{n})} addressing some n highlights (free factors), it allots to this occurrence probabilities

$$p(C_k \mid x_1, \ldots, x_n)$$

for each of K possible outcomes or classes {\displaystyle C_{k}}C_{k}

The issue with the above detailing is that if the quantity of highlights n is huge or assuming a component can take on an enormous number of qualities, putting together a particularly model with respect to likelihood tables is infeasible. The model should along these lines be reformulated to make it more manageable. Utilizing Bayes' hypothesis, the restrictive likelihood can be decayed as

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$$

In practice, there is interest only in the numerator of that fraction, because the denominator does not depend on {\displaystyle C}C and the values of the features {\displaystyle x_{i}}x_{i} are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model[3]

$$p(C_k, x_1, \ldots, x_n)$$

which can be rewritten as follows, using the chain rule for repeated applications of the definition of conditional probability:

$$\begin{aligned} p(C_k, x_1, \ldots, x_n) &= p(x_1, \ldots, x_n, C_k) \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k)\, p(x_2, \ldots, x_n, C_k) \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k)\, p(x_2 \mid x_3, \ldots, x_n, C_k)\, p(x_3, \ldots, x_n, C_k) \\ &= \cdots \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k)\, p(x_2 \mid x_3, \ldots, x_n, C_k) \cdots p(x_{n-1} \mid x_n, C_k)\, p(x_n \mid C_k)\, p(C_k) \end{aligned}$$

Now the "naïve" conditional independence assumptions come into play: assume that all features in conditional independenceare mutually independent, conditional on the category Under this assumption,

$$p(x_i \mid x_{i+1}, \ldots, x_n, C_k) = p(x_i \mid C_k)\,.$$

Thus, the joint model can be expressed as

$$\begin{aligned} p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) \\ &\propto p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\ &\propto p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,, \end{aligned}$$

This means that under the above independence assumptions, the conditional distribution over the class variable C is:

$$p(C_k \mid x_1, \ldots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$

where the evidence $Z = p(\mathbf{x}) = \sum_k p(C_k)\, p(\mathbf{x} \mid C_k)$ is a scaling factor dependent only on $x_1, \ldots, x_n$, that is, a constant if the values of the feature variables are known.

Parameter estimation and event models

A class's prior may be calculated by assuming equiprobable classes (i.e., p(C_{k})=1/K} p(C_{k})=1/K}), or by calculating an estimate for the class probability from the training set (i.e., <prior for a given class> = <number of samples in the class>/<total number of samples>). To estimate the parameters for a feature's distribution, one must assume a distribution or generate nonparametric models for the features from the training set.

Gaussian naïve Bayes

When dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. For example, suppose the training data contains a continuous attribute, X. The data is first segmented by the class, and then the mean and variance of X is computed in each class.

$$p(x = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}}$$

Multinomial naïve bayes

With an ever-growing amount of textual information stored in electronic form such as legal documents, policies, company strategies, etc., automatic text classification is becoming increasingly important. This requires a supervised learning technique that classifies every new document by assigning one or more class labels from a fixed or predefined class. It uses the bag of words approach, where the individual words in the document constitute its features, and the order of the words is ignored. This technique is different from the way we communicate with each other. It treats the language like it' s just a bag full of words and each message is a random handful of them. Large documents have a lot of words that are generally characterized by very high dimensionality feature space with thousands of features. Hence, the learning algorithm requires to tackle high dimensional problems, both in terms of classification performance and computational speed.[5]

$$Posterior\ Probability = \frac{Conditional\ Probability * Prior\ Probability}{Predictor\ Prior\ Probability}$$

$$P\left(\frac{A}{B}\right) = \left(\frac{P(A \cap B)}{P(B)}\right) = \frac{P(A) * P\left(\frac{B}{A}\right)}{P(B)}$$

Bernoulli Naive Bayes

This type of algorithm is useful in data having binary features. The features can be of value yes or not, granted or not granted, useful or useless, etc.

Bayes' Theorem

Bayes' Theorem assists you with analyzing the likelihood of an occasion dependent on the earlier information on any occasion that has correspondence to the previous occasion. Its uses are chiefly found in likelihood hypothesis and measurements. The term innocent is utilized as in the highlights given to the model are not subject to one another. In basic terms, on the off chance that you change the worth of one element in the calculation, it won't straightforwardly impact or change the worth of different highlights.

$$P(A|B) = [P(B|A)P(A)]/[P(B)]$$

$P(A|B)$: The conditional probability that event A occurs, given that B has occurred. This is termed as the posterior probability.

$P(A)$ and $P(B)$: The probability of A and B without any correspondence with each other.

$P(B|A)$: The conditional probability of the occurrence of event B, given that A has occurred.[6]

Naive Bayes Classifier

A classifier is an AI model which is utilized to arrange various articles dependent on certain conduct. Gullible Bayes classifiers in AI are a group of straightforward probabilistic AI models that depend on Bayes' Theorem. In straightforward words, it's anything but a characterization strategy with a presumption of freedom among indicators.

If given variables X, Y, and Z. X will be conditionally independent of Y given Z if and only if the probability distribution of X is independent of the value of Y given Z. This is the assumption of conditional dependence.

In other words, you can also say that X and Y are conditionally independent given Z if and only if, the knowledge of the occurrence of X provides no information on the likelihood of the occurrence of Y and vice versa, given that Z occurs. This assumption is the reason behind the term naive in Naive Bayes.[7]

$$P(X_1 \ldots X_n | Y) = \prod_{i=1}^{n} P(X_i | Y)$$

Working procedure of Naive Bayes Algorithm

Suppose, we have a training data set of 1200 fruits. The features in the data set are these: is the fruit yellow or not, is the fruit long or not, and is the fruit sweet or not. There are three three different classes: mango, banana, and others.

Step 1: Create a frequency table for all the features against the different classes.

Name Yellow Sweet Long Total Mango 350 450 0 650 Banana 400 300 350 400 Others 50 100 50 150 Total 800 850 400 1200

- Out of 1200 fruits, 650 are mangoes, 400 are bananas, and 150 are others.

- 350 of the total 650 mangoes are yellow and the rest are not and so on.

- 800 fruits are yellow, 850 are sweet and 400 are long from a total of 1200 fruits.

if we are given with a fruit which is yellow, sweet, and long and you have to check the class to which it belongs.

Step 2: Draw the likelihood table for the features against the classes.

Name Yellow Sweet Long Total Mango 350/800=P(Mango|Yellow) 450/850 0/400 650/1200=P(Mango) Banana 400/800 300/850 350/400 400/1200 Others 50/800 100/850 50/400 150/1200 Total 800=P(Yellow) 850 400 1200

Step 3: Calculate the conditional probabilities for all the classes, i.e., the following in our example:

$$
\begin{aligned}
P(\text{Mango}|\text{Yellow, Sweet, Long}) &= \frac{P(\text{Yellow}|\text{Mango}).P(\text{Sweet}|\text{Mango}).P(\text{Long}|\text{Mango}).P(\text{Mango})}{P(\text{Yellow, Sweet, Long})} \\
&= 0
\end{aligned}
$$

$$
\begin{aligned}
P(\text{Banana}|\text{Yellow, Sweet, Long}) &= \frac{P(\text{Yellow}|\text{Banana}).P(\text{Sweet}|\text{Banana}).P(\text{Long}|\text{Banana}).P(\text{Banana})}{P(\text{Yellow, Sweet, Long})} \\
&= \frac{400 * 300 * 350 * 400}{400 * 400 * 400 * 1200 * P(\text{Evidence})} \\
&= \frac{0.21875}{P(\text{Evidence})}
\end{aligned}
$$

$$
\begin{aligned}
P(\text{Others}|\text{Yellow, Sweet, Long}) &= \frac{P(\text{Yellow}|\text{Others}).P(\text{Sweet}|\text{Others}).P(\text{Long}|\text{Others}).P(\text{Others})}{P(\text{Yellow, Sweet, Long})} \\
&= \frac{50 * 100 * 50 * 150}{150 * 150 * 150 * 1200 * P(\text{Evidence})} \\
&= \frac{0.00926}{P(\text{Evidence})}
\end{aligned}
$$

Step 4: Calculate

the maximum probability is for the class banana, therefore, the fruit which is long, sweet and yellow is a banana by Naive Bayes Algorithm.[8]

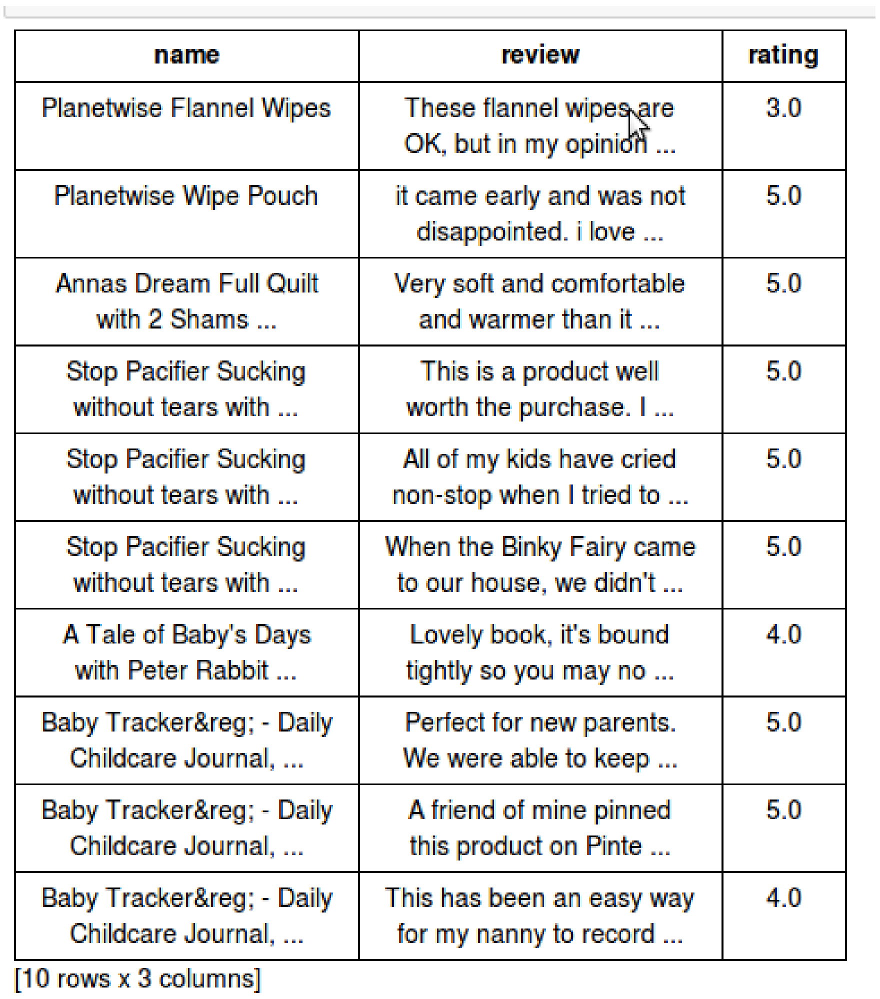
Applications

For social media marketing

In the social online media, a large number of text information is created each second and Naïve Bayes is extraordinary in grouping text information. Assume in the event that we need to distinguish a phony twitter client dependent on his tweets or age of the record or profile data, then, at that point Naïve Bayes can be useful. Likewise it can offer alleviation to many specialist coops by identifying fakes and spammers who are plundering clients by asserting their name.[9]

For media marketing

In the entertainment world, surveys and verbal assume a vital part for motion pictures and shows. Guileless Bayes model can foresee the decision dependent on them. Likewise, Naïve Bayes can be utilized in news coverage areas where a great many book information are created each moment.



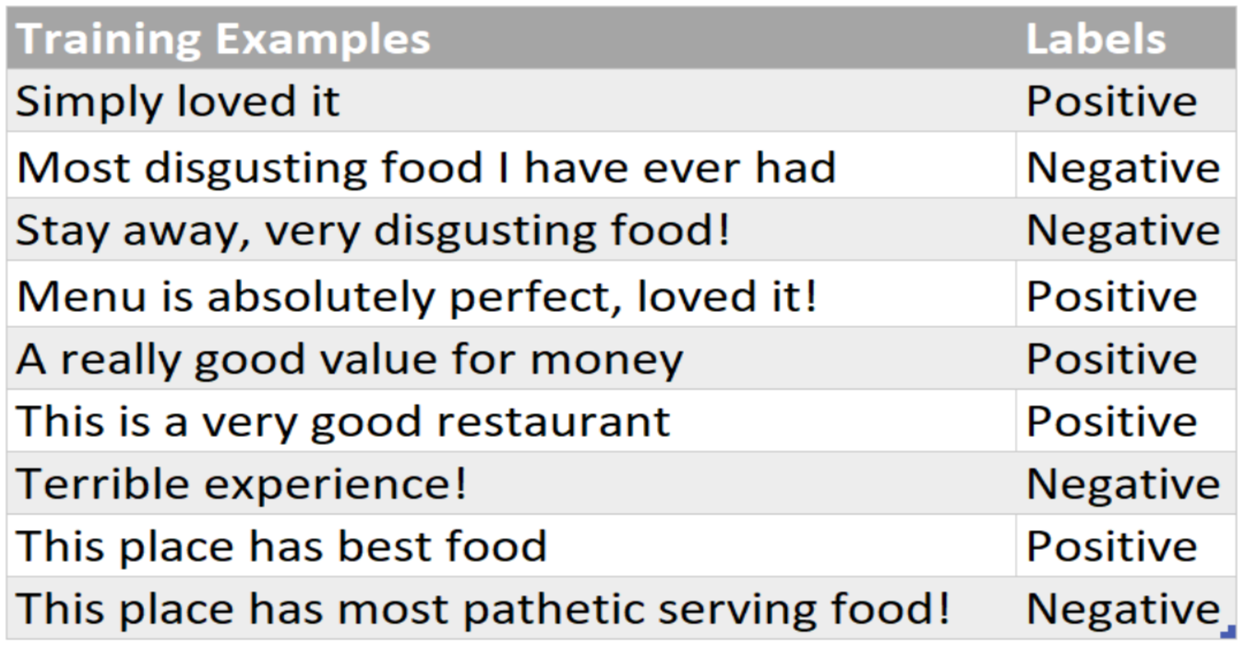
name	review	rating
Planetwise Flannel Wipes	These flannel wipes are OK, but in my opinion ...	3.0
Planetwise Wipe Pouch	it came early and was not disappointed. i love ...	5.0
Annas Dream Full Quilt with 2 Shams ...	Very soft and comfortable and warmer than it ...	5.0
Stop Pacifier Sucking without tears with ...	This is a product well worth the purchase. I ...	5.0
Stop Pacifier Sucking without tears with ...	All of my kids have cried non-stop when I tried to ...	5.0
Stop Pacifier Sucking without tears with ...	When the Binky Fairy came to our house, we didn't ...	5.0
A Tale of Baby's Days with Peter Rabbit ...	Lovely book, it's bound tightly so you may no ...	4.0
Baby Tracker&reg; - Daily Childcare Journal, ...	Perfect for new parents. We were able to keep ...	5.0
Baby Tracker&reg; - Daily Childcare Journal, ...	A friend of mine pinned this product on Pinte ...	5.0
Baby Tracker&reg; - Daily Childcare Journal, ...	This has been an easy way for my nanny to record ...	4.0

[10 rows x 3 columns]

The above picture discloses to us the crowd responses of films and their evaluations. The information is totally literary and for this situation, Naïve Bayes can foresee any concealed film rating dependent on the recurrence of the words utilized in responses.

For product management

Indeed, even item audits can be a worry to organizations and merchants where advertising includes a great deal of stakes. Guileless Bayes can group the nature of the item dependent on such standards. Additionally it tends to be a moderation measure and high murmur of help for such organization proprietors.



Training Examples	Labels
Simply loved it	Positive
Most disgusting food I have ever had	Negative
Stay away, very disgusting food!	Negative
Menu is absolutely perfect, loved it!	Positive
A really good value for money	Positive
This is a very good restaurant	Positive
Terrible experience!	Negative
This place has best food	Positive
This place has most pathetic serving food!	Negative

Labelled Training Dataset

The representation contains reviews of a restaurant where some people are loving it and some are not. For such cases, Naïve Bayes can be a choice for the owner.[10]

Text classification

It is utilized as a probabilistic learning strategy for text arrangement. The Naive Bayes classifier is quite possibly the best known calculations with regards to the order of text records, i.e., regardless of whether a book archive has a place with at least one classifications (classes).[10]

Spam filtration

It is an example of text classification. This has become a popular mechanism to distinguish spam email from legitimate email. Several modern email services implement Bayesian spam filtering.

Many server-side email filters, such as DSPAM, SpamBayes, SpamAssassin, Bogofilter, and ASSP, use this technique.

Sentiment Analysis

It very well may be utilized to examine the tone of tweets, remarks, and surveys—regardless of whether they are negative, positive or impartial.

Recommendation System

The Naive Bayes calculation in blend with community sifting is utilized to fabricate half and half suggestion frameworks which help in anticipating if a client might want a given asset or not.

Case Study in Python

This is where the "naive" in "naive Bayes" comes in: if we make very naive assumptions about the generative model for each label, we can find a rough approximation of the generative model for each class, and then proceed with the Bayesian classification. Different types of naive Bayes classifiers rest on different naive assumptions about the data, and we will examine a few of these in the following sections.[9]

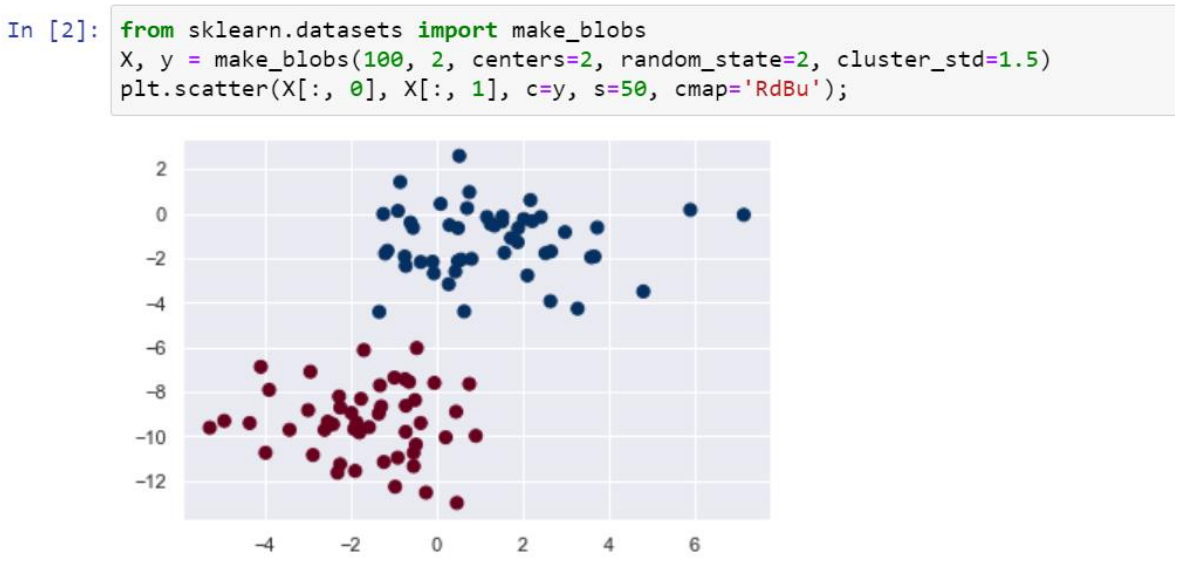
We begin with the standard imports:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
```

Gaussian Naive Bayes

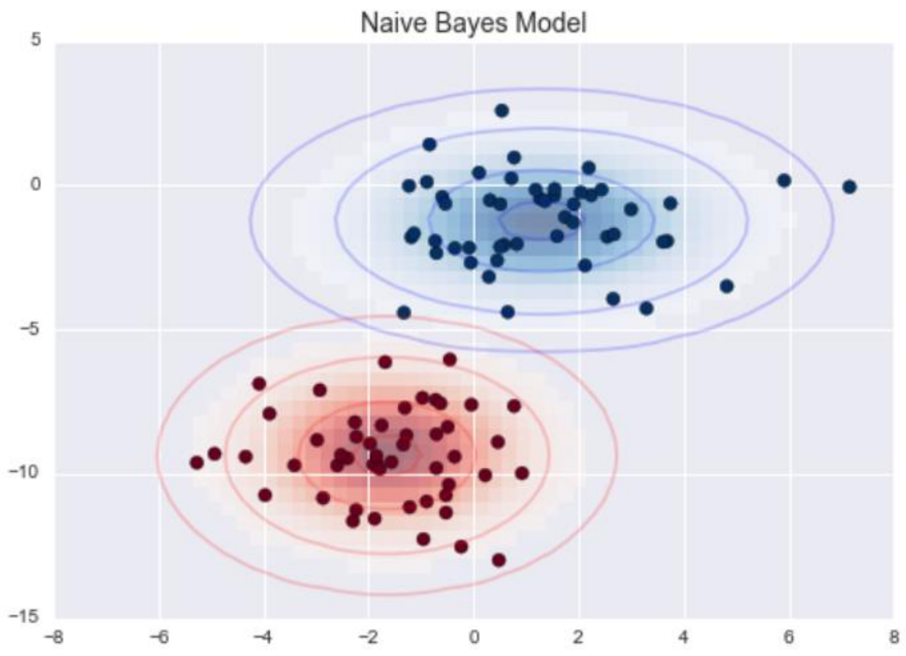
Perhaps the easiest naive Bayes classifier to understand is Gaussian naive Bayes. In this classifier, the assumption is that data from each label is drawn from a simple Gaussian distribution. Imagine that you have the following data:

```
from sklearn.datasets import make_blobs
X, y = make_blobs(100, 2, centers=2, random_state=2, cluster_std=1.5)
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu');
```



```
In [2]: from sklearn.datasets import make_blobs
        X, y = make_blobs(100, 2, centers=2, random_state=2, cluster_std=1.5)
        plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu');
```

One very quick approach to make a basic model is to expect that the information is portrayed by a Gaussian conveyance with no covariance between measurements. This model can be fit by just tracking down the mean and standard deviation of the focuses inside each mark, which is all you need to characterize such a dissemination. The aftereffect of this innocent Gaussian supposition that is displayed in the accompanying figure:



The ellipses here represent the Gaussian generative model for each label, with larger probability toward the center of the ellipses. With this generative model in place for each class, we have a simple recipe to compute the likelihood P(features | L1) for any data point, and thus we can quickly compute the posterior ratio and determine which label is the most probable for a given point.

This procedure is implemented in ScikitLearn's sklearn.naive_bayes.GaussianNB estimator:

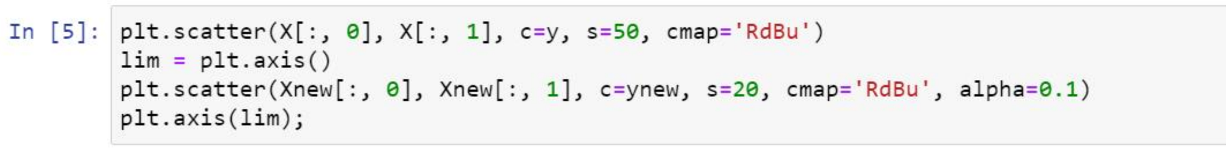
```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X, y);
```

generate some new data and predict the label:

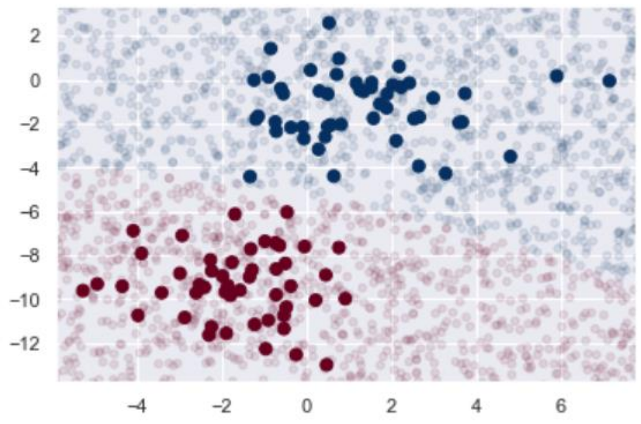
```
rng = np.random.RandomState(0)
Xnew = [-6, -14] + [14, 18] * rng.rand(2000, 2)
ynew = model.predict(Xnew)
```

Now we can plot this new data to get an idea of where the decision boundary is:

```
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu')
lim = plt.axis()
plt.scatter(Xnew[:, 0], Xnew[:, 1], c=ynew, s=20, cmap='RdBu', alpha=0.1)
plt.axis(lim);
```



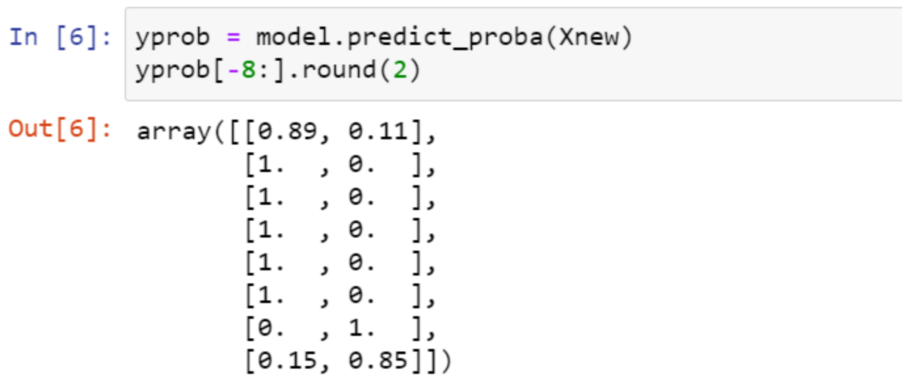
```
In [5]: plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu')
        lim = plt.axis()
        plt.scatter(Xnew[:, 0], Xnew[:, 1], c=ynew, s=20, cmap='RdBu', alpha=0.1)
        plt.axis(lim);
```



We see a slightly curved boundary in the classifications—in general, the boundary in Gaussian naive Bayes is quadratic.

A nice piece of this Bayesian formalism is that it naturally allows for probabilistic classification, which we can compute using the predict_proba method:

```
yprob = model.predict_proba(Xnew)
yprob[-8:].round(2)
```



```
In [6]: yprob = model.predict_proba(Xnew)
        yprob[-8:].round(2)

Out[6]: array([[0.89, 0.11],
               [1.  , 0.  ],
               [1.  , 0.  ],
               [1.  , 0.  ],
               [1.  , 0.  ],
               [1.  , 0.  ],
               [0.  , 1.  ],
               [0.15, 0.85]])
```

One very quick approach to make a basic model is to expect that the information is portrayed by a Gaussian conveyance with no covariance between measurements. This model can be fit by just tracking down the mean and standard deviation of the focuses inside each mark, which is all you need to characterize such a dissemination. The aftereffect of this innocent Gaussian supposition that is displayed in the accompanying figure:

The sections give the back probabilities of the first and second mark, separately. In the event that you are searching for assessments of vulnerability in your grouping, Bayesian methodologies like this can be a helpful methodology.

Multinomial Naive Bayes

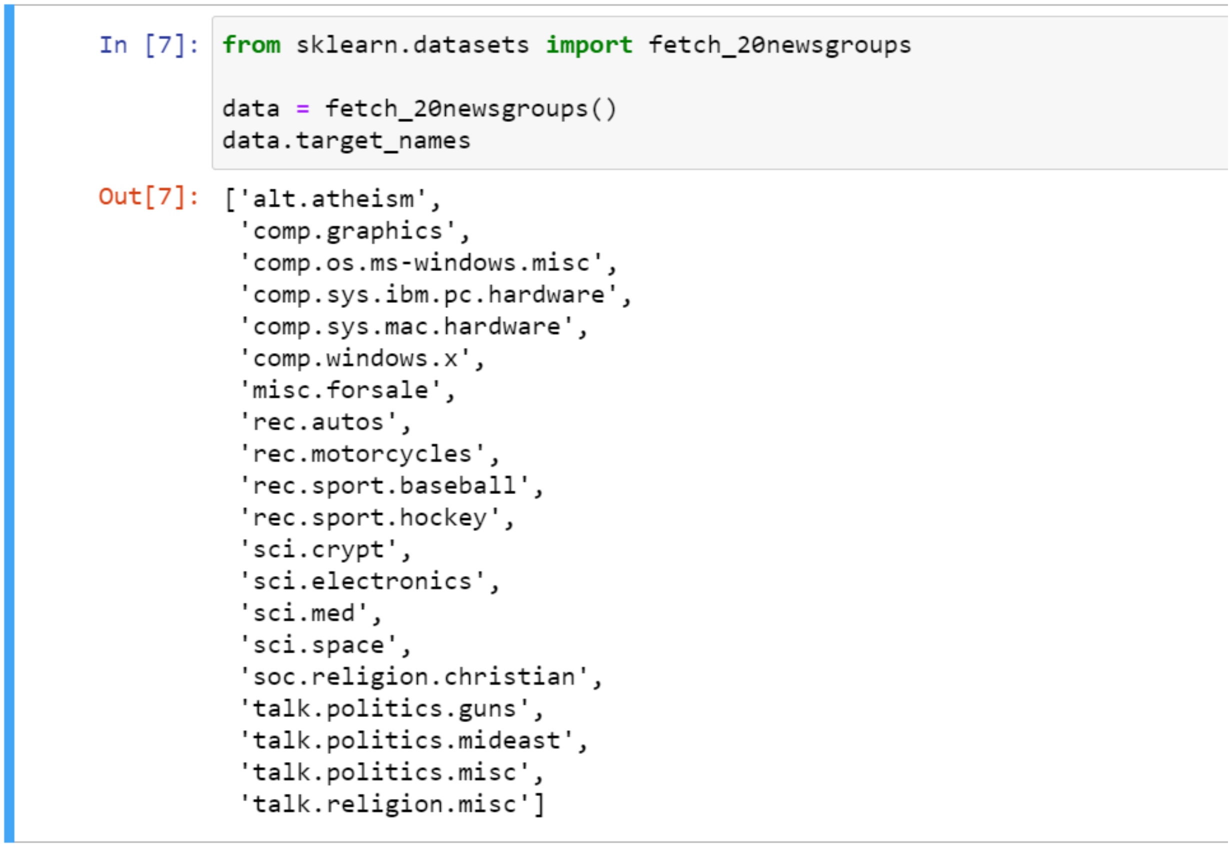
The Gaussian suspicion just portrayed is in no way, shape or form the lone basic presumption that could be utilized to indicate the generative appropriation for each name. Another helpful model is multinomial guileless Bayes, where the highlights are thought to be created from a straightforward multinomial circulation. The multinomial dissemination depicts the likelihood of noticing considers as a part of various classifications, and consequently multinomial gullible Bayes is generally fitting for highlights that address checks or tally rates.[11]

The thought is correctly equivalent to previously, then again, actually as opposed to displaying the information dispersion with the best-fit Gaussian, we model the information distribuiton with a best-fit multinomial dissemination.

Classifying Text

One spot where multinomial gullible Bayes is frequently utilized is in text characterization, where the highlights are identified with word tallies or frequencies inside the records to be arranged. We talked about the extraction of such highlights from text in Feature Engineering; here we will utilize the scanty word tally highlights from the 20 Newsgroups corpus to show how we may group these short reports into classifications[12]

```
from sklearn.datasets import fetch_20newsgroups

data = fetch_20newsgroups()
data.target_names
```



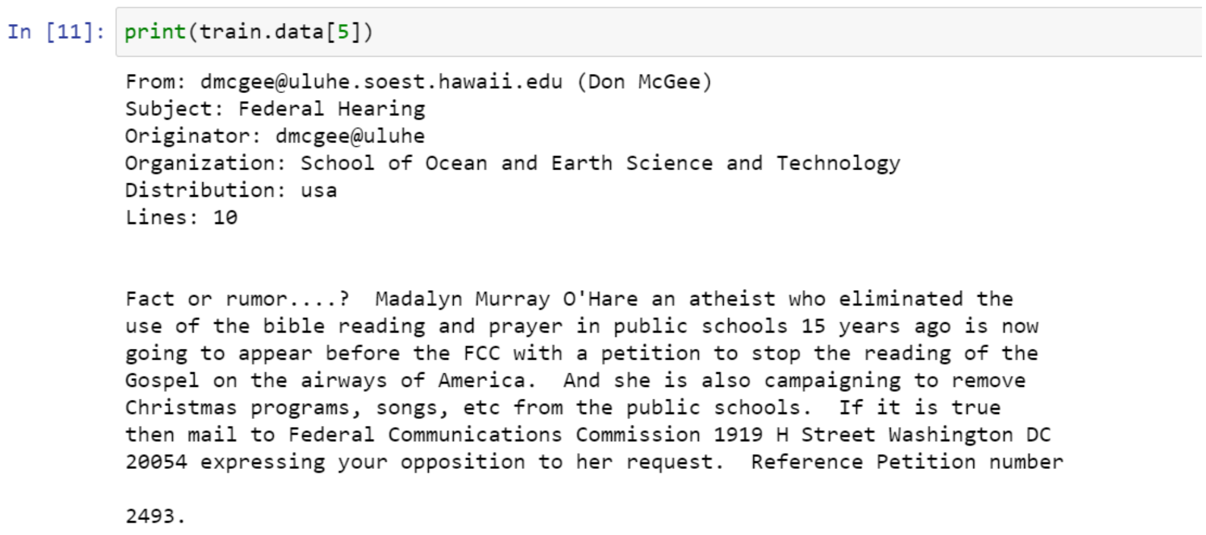
```
In [7]: from sklearn.datasets import fetch_20newsgroups

        data = fetch_20newsgroups()
        data.target_names

Out[7]: ['alt.atheism',
         'comp.graphics',
         'comp.os.ms-windows.misc',
         'comp.sys.ibm.pc.hardware',
         'comp.sys.mac.hardware',
         'comp.windows.x',
         'misc.forsale',
         'rec.autos',
         'rec.motorcycles',
         'rec.sport.baseball',
         'rec.sport.hockey',
         'sci.crypt',
         'sci.electronics',
         'sci.med',
         'sci.space',
         'soc.religion.christian',
         'talk.politics.guns',
         'talk.politics.mideast',
         'talk.politics.misc',
         'talk.religion.misc']
```

For simplicity here, we will select just a few of these categories, and download the training and testing set:

```
categories = ['talk.religion.misc', 'soc.religion.christian',
              'sci.space', 'comp.graphics']
train = fetch_20newsgroups(subset='train', categories=categories)
test = fetch_20newsgroups(subset='test', categories=categories)
```

Here is a representative entry from the data:

```
print(train.data[5])
```



```
In [11]: print(train.data[5])

         From: dmcgee@uluhe.soest.hawaii.edu (Don McGee)
         Subject: Federal Hearing
         Originator: dmcgee@uluhe
         Organization: School of Ocean and Earth Science and Technology
         Distribution: usa
         Lines: 10


         Fact or rumor....?  Madalyn Murray O'Hare an atheist who eliminated the
         use of the bible reading and prayer in public schools 15 years ago is now
         going to appear before the FCC with a petition to stop the reading of the
         Gospel on the airways of America.  And she is also campaigning to remove
         Christmas programs, songs, etc from the public schools.  If it is true
         then mail to Federal Communications Commission 1919 H Street Washington DC
         20054 expressing your opposition to her request.  Reference Petition number

         2493.
```

To utilize this information for AI, we should have the option to change over the substance of each string into a vector of numbers. For this we will utilize the TF-IDF vectorizer (examined in Feature Engineering), and make a pipeline that appends it's anything but a multinomial gullible Bayes classifier:

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline
```

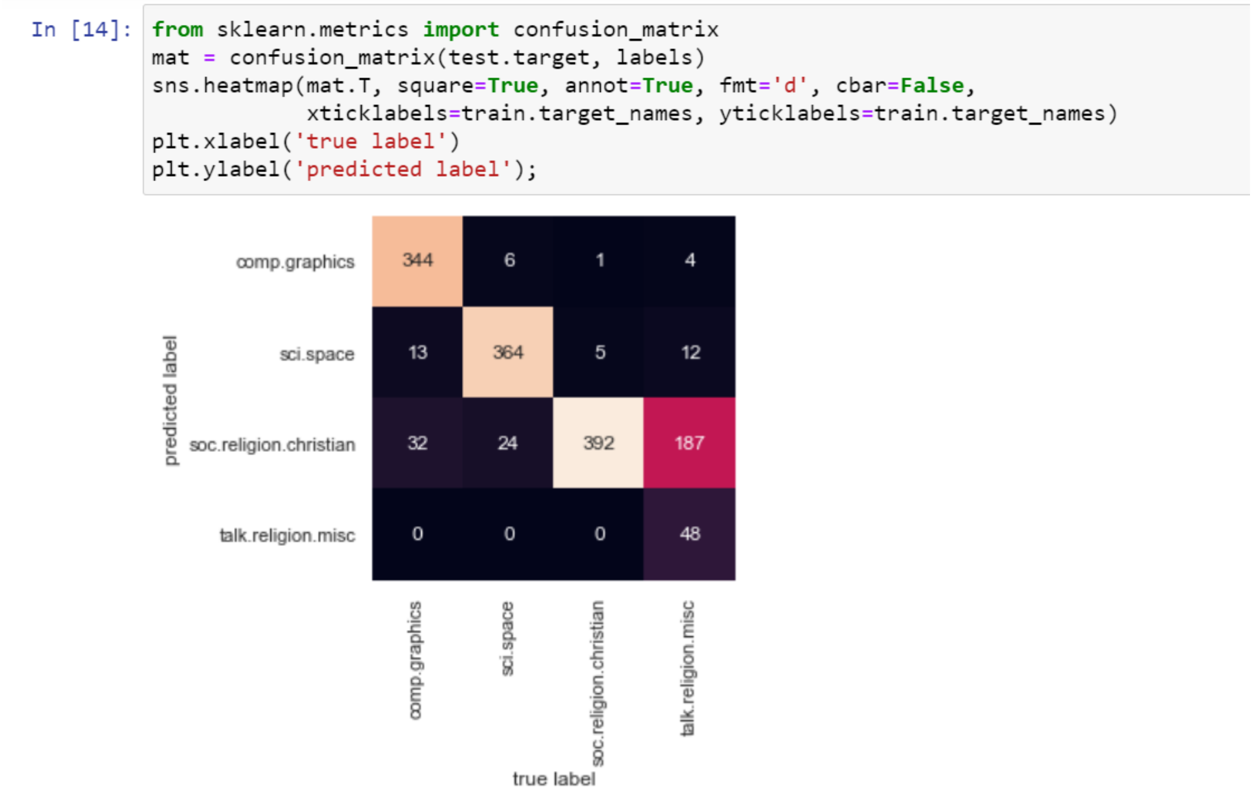
```
model = make_pipeline(TfidfVectorizer(), MultinomialNB())
```

With this pipeline, we can apply the model to the training data, and predict labels for the test data:

```
model.fit(train.data, train.target)
labels = model.predict(test.data)
```

Now that we have predicted the labels for the test data, we can evaluate them to learn about the performance of the estimator. For example, here is the confusion matrix between the true and predicted labels for the test data:

```
from sklearn.metrics import confusion_matrix
mat = confusion_matrix(test.target, labels)
sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False,
            xticklabels=train.target_names, yticklabels=train.target_names)
plt.xlabel('true label')
plt.ylabel('predicted label');
```



```
In [14]: from sklearn.metrics import confusion_matrix
         mat = confusion_matrix(test.target, labels)
         sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False,
                     xticklabels=train.target_names, yticklabels=train.target_names)
         plt.xlabel('true label')
         plt.ylabel('predicted label');
```

predicted label \ true label	comp.graphics	sci.space	soc.religion.christian	talk.religion.misc
comp.graphics	344	6	1	4
sci.space	13	364	5	12
soc.religion.christian	32	24	392	187
talk.religion.misc	0	0	0	48

Obviously, even this basic classifier can effectively isolate space talk from PC talk, however it gets confounded between talk about religion and talk about Christianity. This is maybe a normal space of disarray!

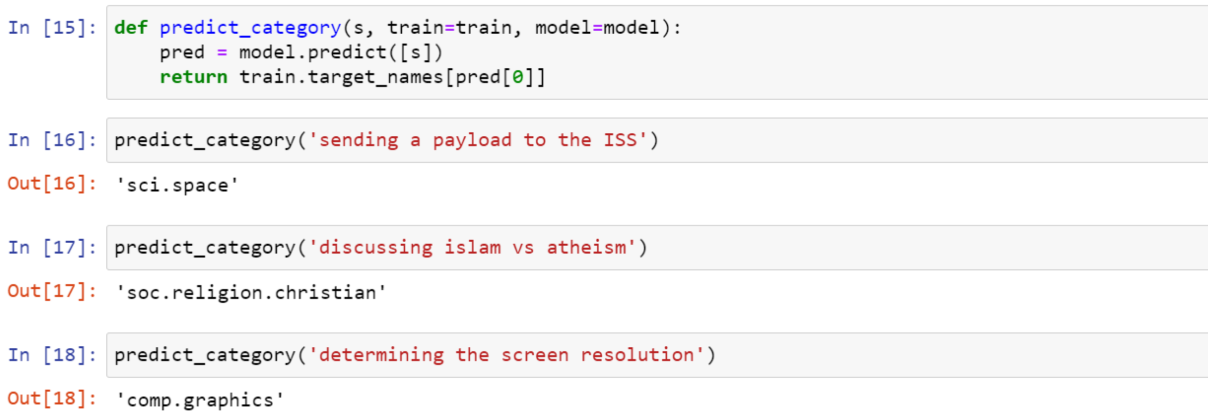
The very cool thing here is that we now have the tools to determine the category for *any* string, using the predict() method of this pipeline. Here's a quick utility function that will return the prediction for a single string:

```
def predict_category(s, train=train, model=model):
    pred = model.predict([s])
    return train.target_names[pred[0]]
```

```
predict_category('sending a payload to the ISS')
```

```
predict_category('discussing islam vs atheism')
```

```
predict_category('determining the screen resolution')
```



```
In [15]: def predict_category(s, train=train, model=model):
             pred = model.predict([s])
             return train.target_names[pred[0]]

In [16]: predict_category('sending a payload to the ISS')

Out[16]: 'sci.space'

In [17]: predict_category('discussing islam vs atheism')

Out[17]: 'soc.religion.christian'

In [18]: predict_category('determining the screen resolution')

Out[18]: 'comp.graphics'
```

Advantages and Disadvantages

Naïve Bayes calculation is not difficult to carry out with regards to message information.

1. It is really amicable with text mining
2. Intermingling is faster than models like strategic relapse that are discriminative in nature
3. It performs well in any event, when the information doesn't follow the suppositions it holds.

The lone burden of this calculation is that it neglects to discover a connection between highlights. "In any event, when the highlights have a solid relationship with one another, then, at that point unquestionably Naïve Bayes is an awful decision. Something else to remember is it allocates zero likelihood to the result when the indicator class is absent in the preparation information." [9]

Conclusion

Though Naive Bayes has a handful of limitations, it' s still a go-to algorithm to classify data, primarily due to its simplicity. It has worked particularly well for document classification and spam filtering. For a more hands-on understanding of Naive Bayes, Despite all the complicated math, the implementation of the Naive Bayes algorithm involves simply counting the number of objects with specific features and classes. Once these numbers are obtained, it is very simple to calculate probabilities and arrive at a conclusion.

Reference:

[1] F. Razaque *et al.*, “Using naïve bayes algorithm to students’ bachelor academic performances analysis,” *4th IEEE Int. Conf. Eng. Technol. Appl. Sci. ICETAS 2017*, vol. 2018-Janua, pp. 1–5, 2018, doi: 10.1109/ICETAS.2017.8277884.

[2] M. Hawari and B. Sinaga, “Naïve Bayes Algorithm Implementation To Predict Gum Production at PT. Sri Rahayu Court,” *J. Mantik*, vol. 3, no. 3, pp. 40–45, 2019.

[3] Wikipedia, “Naive Bayes classifier (Wikipedia),” *http://en.wikipedia.org/w/index.php?title=Naive_Bayes_classifier*, 2013. http://en.wikipedia.org/w/index.php?title=Naive_Bayes_classifier (accessed Jun. 29, 2021).

[4] A. Goel, J. Gautam, and S. Kumar, “Real time sentiment analysis of tweets using Naive Bayes,” *Proc. 2016 2nd Int. Conf. Next Gener. Comput. Technol. NGCT 2016*, pp. 257–261, 2017, doi: 10.1109/NGCT.2016.7877424.

[5] F. Harahap, A. Y. N. Harahap, E. Ekadiansyah, R. N. Sari, R. Adawiyah, and C. B. Harahap, “Implementation of Naïve Bayes Classification Method for Predicting Purchase,” *2018 6th Int. Conf. Cyber IT Serv. Manag. CITSM 2018*, 2019, doi: 10.1109/CITSM.2018.8674324.

[6] M. Minsky, *Steps toward Artificial Intelligence*. 1961.

[7] M. N. Murty and V. Susheela Devi, *Pattern Recognition: An Algorithmic Approach*. 2011.

[8] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell, “Text classification from labeled and unlabeled documents using EM,” *Mach. Learn.*, vol. 39, no. 2, pp. 103–134, 2000, doi: 10.1023/a:1007692713085.

[9] K. Keshari, “Naive Bayes Tutorial | Naive Bayes Classifier in Python | Edureka,” 2020. https://www.edureka.co/blog/naive-bayes-tutorial/ (accessed Jun. 29, 2021).

[10] M. Vadivukarassi, N. Puviarasan, and P. Aruna, “Sentimental Analysis of Tweets Using Naive Bayes Algorithm,” *World Appl. Sci. J.*, vol. 35, no. 1, pp. 54–59, 2017, doi: 10.5829/idosi.wasj.2017.54.59.

[11] N. Bayes, N. Bayes, and N. B. Algorithm, “Introduction to Naive Bayes,” 2017. https://www.mygreatlearning.com/blog/introduction-to-naive-bayes/ (accessed Jun. 29, 2021).

[12] HackerEarth, “An Introduction to the Naive Bayes Algorithm (with codes in Python and R).” https://medium.com/@hackerearth/an-introduction-to-the-naive-bayes-algorithm-with-codes-in-python-and-r-7c85cdb03490 (accessed Jun. 29, 2021).