Naïve Bayes

Stephanie Okosa[1]

Contents

Abstract: Naïve Bayes is a simple, agile supervised learning algorithm used in machine learning, for the purpose of classifying very often high-dimensional data set. Its is a long used approach in information retrieval. As a result of its fast and few tunable parameters, it is being used mostly as a baseline for classification problems. In this paper, naïve bayes is described operating under the principle of the Bayesian method. This paper gives an introduction into the Bayes theorem, explaining the founding principle of naïve bayes learning, some use case implementations, as well as naive bayes assumptions, violations and successes.

[1] stephanie-chinenye.okosa@stud.hshl.de

1 Motivation

When computers are instructed on how to learn in order to solve problems with given data inputs, they have the ability to progress beyond what they are told through diverse data models. Data models include descriptors in a table form which are; the attribute in the column of the data table called features, and the examples specified in the rows of the data table termed instances. Numerous learning systems which are applied in today's world work with various learning algorithms such as, image recognition systems in self driving cars; business learning systems that enable optimization of business processes on different sites; Medical learning systems for example, in triage; and many other sectors using observed data. The process of building data models with tunable parameters that can conform to different observed data is known as Machine learning [Va22]. There are two main classification of machine learning: unsupervised learning and supervised learning. Unsupervised learning entails modelling the features of a dataset without making any reference to the examples of the data model. These type of learning can be applied in pattern recognition.
Supervised learning includes modelling the relationship between observed features of data set and some examples associated within the data. This modelling category is used in Naive bayes. Naive Bayes are easy-to-train classifiers that use bayes theorem to ascertain the probability of an outcome with a given set of conditions. It is referred to as 'naive' based on the fundamental assumption of independence between each and every pair of features and all features contribute equally to the outcome. Naive bayes algorithms have excellent performance where the probability of a class is determined by the probability of some causal factors. In the following sections of this paper, a foundation to bayes theorem is given as well as explicit use case applications.

2 Foundation of Naïve Bayes

Naïve Bayes is known as the probabilistic classifier as it operates on the Bayesian learning using the Bayes's rule. Bayesian learning which is the Bayes Theorem is a statistical-based method that depicts a detailed equation of the conditional probabilities of statistical quantities [IB]. Bayes's rule states that the probability of x given y is the probability that they happen together relative to the probability that y happens at all. It permits the transpose of conditional probability, which represents the probability of an event given that some other event has occurred as shown below.

$$p(x|y) = p(x) \; or \; p(y/x) = p(y) \tag{1}$$

$$p(x|y) = (p(y|x))/(p(y)) \tag{2}$$

$$p(y) = p(y|x) * p(x) + p(y|x^c) * p(x^c) \tag{3}$$

where $x^c = x_2 \cup x_3 \cup \ldots x_n$, and redefining $x = x_1$ [Be18]. The conditional probability p(x|y) is the joint probability of x and y divided by the probability of y as shown in equation 4. Assuming that $x \neq 0$ and $y \neq 0$,

$$p(x|y) = p(x \cap y)/p(y) \tag{4}$$

From equation 4, it is obvious that,

$$p(x \cap y) = p(x|y) * p(y) = p(y|x) * p(x) \tag{5}$$

Therefore,

$$p(x|y) = \frac{p(y|x) * p(x)}{p(y)} \tag{6}$$

which is the Bayes theorem. For a sample space, can be divided into finite number of mutually exclusive events $x_i = x_1, x_2, \ldots, x_n$. If y an event with p(y) > 0 is a subset of the union of all events x_i, then Bayes Formula would be:

$$p(x_i|y) = \frac{p(y|x_i) * p(x_i)}{\sum_{i=1}^{n} p(y/x_i) * p(x)} \tag{7}$$

Employing Bayes theorem stemming from equation [7] and [3], The probabilities are denoted as prior and posterior probability. Therefore, the posterior probability of a hypothesis, x given observed data, y is

$$p(x|y) = \frac{p(x|y) * p(y)}{p(y|x) * p(x) + p(y|x^c) * p(x^c)} \tag{8}$$

where $p(x|y)$ represents the posterior probability which is the probability that an event occurs when more information or knowledge of any assumption is provided for a problem context; $p(x)$ is the prior probability that no prior information or knowledge of assumption is provided for the problem context; $p(y|x)$ is the likelihood of the data given the hypothesis; $p(y)$ is the probability of the data regardless of the given hypothesis i.e. $p(y)$ is usually constant [Go16]. Recall: $p(y) = p(y|x) * p(x) + p(y|x^c) * p(x^c)$ [Go16].
An applied example of this theorem would be:- Assume there are two bowls of nuts. The first bowl contains 30 cashew nuts, 10 pistachios and the second bowl contains 20 of each. What is the probability that a cashew nut is selected from the first bowl? [Go16]
Solution: Using [6],

$$p(Bowl1|Cashew) = \frac{p(Bowl1) * p(Cashew|Bowl1)}{p(Cashew)}$$

Where: $p(Cashew|Bowl1) = 30/40 = 3/4$, $p(Bowl1) = 1/2$, and $p(Cashew) = 50/80 = 5/8$.
Therefore,

$$p(Bowl1|Cashew) = \frac{3/8}{5/8} = 3/5$$

3 Naïve Bayes

Naïve Bayes classifiers operate under the following assumptions [IB]:

- Each event are conditionally independent in a model
- All features contribute equally to the outcome in a model

Observing from section 1, data models have tunable parameter. However, one of the characteristics of naive Bayes models is that they have few tunable parameters which enables them to be extremely fast and are used as a baseline for classification problems.

3.1 Naïve Bayes Classification

There are different types of naive Bayes classifiers. This classifiers vary depending on their feature value distributions and follow the crisp classification rule i.e. Bayes Theorem [8], which assigns each instance to exactly one class. The word "$naive$" is prescribed from the simplified assumption of the generative model used in specifying the hypothetical random process that generates the data of a model [Va22]. The diverse forms of naive bayes classifiers have different naive assumption about the data of their model which would be discussed in the following subsections.

3.1.1 Gaussian Naive Bayes

The Gaussian Naïve Bayes as the name implies is applied in Gaussian distributions i.e. continuous variables and normal distributions by, finding the mean and standard deviation of each class [IB]. The assumption is that data from each instance is acquired from a simple Gaussian distribution. A classical example can be seen in spam classifications shown in 1. Here, spam filtering uses the Gaussian distribution classification method in order to filter spam emails from random data inputs with predefined labels [Va22]. The interest is to find the probability of having a spam given some messages in an email i.e., $P(Spam|Messages)$ refer to [8]. Having this generative model, the $P(Spam|Messages)$ can be computed as follows in [1] using python with the Scikit-Learn library.

Fig. 1: Fig 1:Spam Classification

3.1.2 Multinomial Naive Bayes

Multinomial Naïve Bayes employs discrete data to describe the probability of observing counts of different categories. Here features are assumed to be produced from a simple multinomial distribution [Va22]. An application can be seen in sentiment analysis for text classification as seen in figure [2].

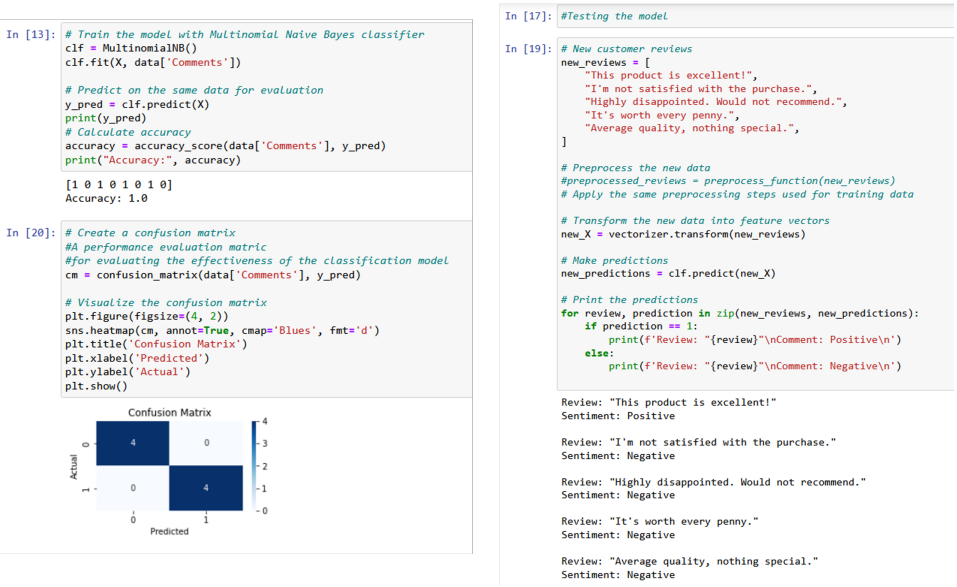

Fig. 2: Fig 2:Sentiment Analysis

Multinomial Naïve Bayes classifier is most appropriate for features depicting counts or count rates such as frequency counts. In order to to make use of the data for machine learning, the data content of each string is transformed into vectors of number. In figures [1] and [2], Evaluation of the performance of the predicted data for test labels can be realized and visualized in a confusion matrix.

3.1.3 Bernoulli Naive Bayes

Another variant of the Naïve Bayes classifier is the Bernoulli Naïve Bayes classifier which is used with Boolean variables i.e. 1's, 0's or true,false. It is effective when features are either present or absent. If X is a Bernoulli-distributed random variable, the probability is [Bo18];

$$p(X) = \begin{cases} p \text{ if } X = 1, & \text{where } q = 1 - p \text{ and } 0 \leq p \leq 1 \\ q \text{ if } X = 0 \end{cases} \tag{9}$$

If n samples exist, the probability becomes p_i for the number of times $N_x(i)$ for which the i^{th} feature is 1.

$$p_i = \frac{N_x(i) = 1}{n} \tag{10}$$

This can be applied in weather predictions as seen in figure [3]. Bernoulli naive bayes employs binary feature vectors and makes use of a *binarize* parameter that enables an internal threshold value used in transforming the features to be specified [Bo18].

```
In [227]: # Split the data into features (X) and labels (y)
X = new_data[['Temperature', 'Wind', 'Precipitation']]
y = new_data['Summary']

# Convert categorical features to binary using one-hot encoding
X = pd.get_dummies(X, drop_first=True)

# Split into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the Bernoulli Naive Bayes classifier
#classifier = BernoulliNB(binarize=True)
#classifier.fit(X, y)

# Transform text data into numerical features
vectorizer = CountVectorizer(binary=True)
#X_train = vectorizer.fit_transform(X_train)
#X_test = vectorizer.transform(X_test)

# Train the Bernoulli Naive Bayes model
model = BernoulliNB()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)
#print(y_pred)
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)

Accuracy: 0.8293515358361775
```

```
In [233]: #test prediction
data = {
    'Precipitation': [0.2, 0.0, 0.5, 0.8, 0.0],
    'Temperature': [25, 30, 22, 28, 26],
    'Wind': [10, 5, 15, 8, 12],
}

# Create a pandas DataFrame from the dictionary
test_example = pd.DataFrame(data)
test_example.head()


# Convert new categorical features to binary using one-hot encoding
new_X = pd.get_dummies(test_example, drop_first=True)
new_X = new_X.reindex(columns=X_train.columns, fill_value=0)
# Make predictions
new_summary = model.predict(new_X)
print("Predicted labels:", new_summary)
# Print the predictions

Predicted labels: ['rain' 'sun' 'rain' 'rain' 'sun']
```

Fig. 3: Fig 3: Weather Prediction

The features of values with the Naïve Bayes classifier are easy and fast to estimate. Thus, Naïve Bayes is presented as a simple classifier that provides straightforward and interpretable probabilistic predictions [Va22] which are applied in both in data science and machine learning. However, the most known downside of this method is that it the conditional independence assumption does not always hold, leading to false classification [IB].

4 Conclusion

Naive bayes classifiers are very fast, straightforward, and easily interpretable for training and predicting outcomes of models because of its few tunable parameters. However, it is most more suitable as a classification baseline as it has higher accuracy for well-separated categories of data, high dimensional data with less model complexity, and mostly when the naive assumptions match the data which is rare[Va22].The independence assumptions in naive bayes seldom holds true for natural data sets. This has resulted in further research being carried out to produce better modifiers by easing the independent assumptions, modify feature sets in order to validate the independence assumptions, and to undertake explaining why the independence assumption is inessential [Le05].

5 Declaration of Originality

I, Stephanie Okosa, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

Date & Place - Stephanie Okosa

Bibliography

[Be18] Berrar, Daniel: Bayes' Theorem and Naive Bayes Classifier. January, 2018. Gives an elaborate explanation of bayes theorem. Deriving the probability equation and stating the meaning for each variable of the equation e.g prior probability.

[Bo18] Bonaccorso, Giuseppe: Machine Learning Algorithms - Second Edition. August, 2018. explains the naive bayes classifications.

[Go16] Gollapudi, Sunila: Practical Machine Learning. January, 2016. Gives the foundation of bayes theorem and example.

[IB] IBM: What are Naïve Bayes classifiers? Talks about naive bayes classifiers, advantages and disadvantages and how Naïve Bayes classifiers uses principles of probability to perform classification tasks.

[Le05] Lewis, David D.: Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. 1 January 2005. Gives an introduction into bayes theorem, explanation on the naive bayes classifiers with some violated assumptions and success of naive bayes, amongst other subsections.

[Va22] VanderPlas, Jake: Python Data Science Handbook, 2nd Edition. December, 2022. Gives an introductory explanation to Machine Learning and the naive classification methods.Also got my conclusion text from here.