Applied Statistics in R

Classification

Elena Parilina

Master's Program
Game Theory and Operations Research

Saint Petersburg State University

2019

Agenda

1. Naive Bayes Classifier
   - Types of Naive Bayes Algorithm
2. Naive Bayes classifier in R

Naive Bayes Classifier

Bayes Theorem

Bayes theorem named after Thomas Bayes. It works on conditional probability. Conditional probability is the probability that something will happen, given that something else has already occurred. Using the conditional probability, we can calculate the probability of an event using its prior knowledge.
Below is the formula for calculating the conditional probability:

$$P(H/E) = \frac{P(E/H)P(H)}{P(E)},$$

- $P(H)$ is the probability of hypothesis H being true. This is known as the prior probability.
- $P(E)$ is the probability of the evidence(regardless of the hypothesis).
- $P(E/H)$ is the probability of the evidence given that hypothesis is true.
- $P(H/E)$ is the probability of the hypothesis given that the evidence is there.

Example

Problem

A Path Lab is performing a Test of disease say "D" with two results "Positive" & "Negative." They guarantee that their test result is 99% accurate: if you have the disease, they will give test positive 99% of the time. If you don't have the disease, they will test negative 99% of the time. If 3% of all the people have this disease and test gives "positive" result, what is the probability that you actually have the disease?

For solving the above problem, we will have to use conditional probability:

- Probability of people suffering from Disease D: $P(D) = 0.03 = 3\%$.
- Probability that test gives "positive" result and patient have the disease, $P(Pos/D) = 0.99 = 99\%$.
- Probability of people not suffering from Disease D, $P(\bar{D}) = 0.97 = 97\%$.
- Probability that test gives "positive" result and patient does have the disease, $P(Pos/\bar{D}) = 0.01 = 1\%$.

Example (cntd)

For calculating the probability that the patient actually have the disease i.e, $P(D/Pos)$ we will use Bayes theorem:

$$P(D/Pos) = \frac{P(Pos/D)P(D)}{P(Pos)},$$

where the probability $P(Pos)$ is

$$\begin{aligned} P(Pos) &= P(D \cap Pos) + P(\bar{D} \cap Pos) \\ &= P(Pos/D)P(D) + P(Pos/\bar{D})P(\bar{D}) \\ &= 0.99 \cdot 0.03 + 0.01 \cdot 0.97 = 0.0394. \end{aligned}$$

Let calculate probability

$$P(D/Pos) = \frac{P(Pos/D)P(D)}{P(Pos)} = \frac{0.99 \cdot 0.03}{0.0394} = 0.753807107.$$

So, there are approximately 75% chances that the patient is actually suffering from disease.

Maximum A Posteriori (MAP)

Idea

Naive Bayes is a kind of classifier which uses the Bayes Theorem. It predicts membership probabilities for each class such as the probability that given record or data point belongs to a particular class. The class with the highest probability is considered as the most likely class. This is also known as Maximum A Posteriori (MAP).

Assume that there are several hypotheses, i.e., the set of hypotheses is $H = \{H_1, \ldots, H_l\}$. The MAP for a hypothesis is:

$$MAP(H^*) = \max_{H_i \in H} P(H_i/E) = \max_{H_i \in H} \frac{P(E/H_i)P(H_i)}{P(E)}$$
$$= \frac{\max_{H_i \in H} [P(E/H_i)P(H_i)]}{P(E)},$$

where $P(E)$ is evidence probability, and it is used to normalize the result.

Assumption 1

Naive Bayes classifier assumes that all the features are unrelated to each other. Presence or absence of a feature does not influence the presence or absence of any other feature.

Wikipedia example for explaining the logic of Assumption 1

A fruit may be considered to be an apple if it is red, round, and about 4" in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple.

In real datasets, we test a hypothesis given multiple evidence (features). So, calculations become complicated. To simplify the work, the feature independence approach is used to uncouple multiple evidence and treat each as an independent one:

$$P(H_i/E_1 \cap \ldots \cap E_n) = \frac{P(E_1/H_i)P(E_2/H_i) \cdot \ldots \cdot P(E_n/H_i)P(H_i)}{P(E_1 \cap \ldots \cap E_n)}$$

Example of Naive Bayes Classifier

Consider a training dataset with 1500 records and 3 classes. We presume that there are no missing values in our data. We have 3 classes associated with Animal Types:

1. Parrot,
2. Dog,
3. Fish.

The Predictor features set consists of 4 features as

1. Swim,
2. Wings,
3. Green Color,
4. Dangerous Teeth.

All the features are categorical variables with either of the 2 values: T (True) or F (False).

Frequency table

Swim	Wings	Green Color	Dangerous Teeth	Animal Type
50/500	500/500	400/500	0	Parrot
450/500	0	0	500/500	Dog
500/500	0	100/500	50/500	Fish

The above table shows a frequency table of our data. In our training data:

- Parrots have 50 out of 500 (10%) value for Swim, i.e., 10% parrot can swim according to our data, 500 out of 500 (100%) parrots have wings, 400 out of 500 (80%) parrots are Green and 0 (0%) parrots have Dangerous Teeth.
- Classes with Animal type Dogs shows that 450 out of 500 (90%) can swim, 0 (0%) dogs have wings, 0 (0%) dogs are of Green color and 500 out of 500 (100%) dogs have Dangerous Teeth.
- Classes with Animal type Fishes shows that 500 out of 500 (100%) can swim, 0 (0%) fishes have wings, 100 (20%) fishes are of Green color and 50 out of 500 (10%) dogs have Dangerous Teeth.

Now, it time to work on predict classes using the Naive Bayes model. We have taken 2 records that have values in their feature set, but the target variable needs to predicted:

	Swim	Wings	Green Color	Dangerous Teeth
1	True	False	True	False
2	True	False	True	True

Task

We have to predict animal type using the feature values. We have to predict whether the animal is a Dog, a Parrot or a Fish.

Consider the first record. The Evidence here is "Swim & Green". The Hypothesis can be an animal type to be Dog (H_1), Parrot (H_2), Fish (H_3).

Testing H_1

$$P(Dog/Swim, Green) = \frac{P(Swim/Dog)P(Green/Dog)P(Dog)}{P(Swim, Green)}$$
$$= \frac{0.9 \cdot 0 \cdot 0.333}{P(Swim, Green)} = 0.$$

Testing H_2

$$P(Parrot/Swim, Green)$$
$$= \frac{P(Swim/Parrot)P(Green/Parrot)P(Parrot)}{P(Swim, Green)}$$
$$= \frac{0.1 \cdot 0.80 \cdot 0.333}{P(Swim, Green)} = \frac{0.0264}{P(Swim, Green)}.$$

Testing H_3

$$P(Fish/Swim, Green) = \frac{P(Swim/Fish)P(Green/Fish)P(Fish)}{P(Swim, Green)}$$
$$= \frac{1 \cdot 0.2 \cdot 0.333}{P(Swim, Green)} = \frac{0.0666}{P(Swim, Green)}.$$

The denominator of all the above calculations is same, i.e, $P(Swim, Green)$. The value of $P(Fish/Swim, Green)$ is the maximal one.

Result

Using Naive Bayes, we can predict that the class of this record is Fish.

Exercise

Classify the 2nd record using Naive Bayes Classifier.

Gaussian Naive Bayes

When attribute values are continuous, an assumption is made that the values associated with each class are distributed according to Gaussian i.e., Normal Distribution.
If in our data, an attribute x contains continuous data. We first segment the data by the class and then compute mean μ_y and Variance ${\sigma_y}^2$ of each class y. We use formula of conditional probability:

$$P(x_i/y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right).$$

MultiNomial Naive Bayes

MultiNomial Naive Bayes is preferred to use on data that is multinomially distributed. It is one of the standard classic algorithms. Which is used in text categorization (classification). Each event in text classification represents the occurrence of a word in a document.

Bernoulli Naive Bayes

Bernoulli Naive Bayes is used on the data that is distributed according to multivariate Bernoulli distributions.i.e., multiple features can be there, but each one is assumed to be a binary-valued (Bernoulli, boolean) variable. So, it requires features to be binary valued.

Advantages of Naive Bayes classifier:

1. Naive Bayes Algorithm is a fast, highly scalable and simple algorithm.
2. Naive Bayes can be use for Binary and Multiclass classification. It provides different types of Naive Bayes Algorithms like GaussianNB, MultinomialNB, BernoulliNB.
3. Great choice for Text Classification problems. It's a popular choice for spam email classification.
4. It can be easily trained on small dataset.

Disadvantages of Naive Bayes classifier:

1. It considers all the features to be unrelated, so it cannot learn the relationship between features. E.g., Let's say Remo is going to a part. While cloth selection for the party, Remo is looking at his cupboard. Remo likes to wear a white color shirt. In Jeans, he likes to wear a brown Jeans, but Remo doesn't like wearing a white shirt with Brown Jeans. Naive Bayes can learn individual features importance but can't determine the relationship among features.

Naive Bayes classifier in R

Package "e1071"

Package

library(e1071)
library(klaR)

naiveBayes(x, y, type="raw", ...)

Arguments:

- `x`: A numeric matrix, or a data frame of categorical and/or numeric variables.
- `y`: Class vector.
- `formula`: A formula of the form $class \sim x1 + x2 + \ldots$
- `type`: If "raw", the conditional a-posterior probabilities for each class are returned, and the class with maximal probability else.

Package "klaR"

NaiveBayes(x, grouping, prior, usekernel = FALSE, fL = 0, ...)

Arguments:

- `x`: a numeric matrix, or a data frame of categorical and/or numeric variables.
- `y`: class vector (a factor).
- `formula`: a formula of the form $class \sim x1 + x2 + \ldots$
- `prior`: the prior probabilities of class membership. If unspecified, the class proportions for the training set are used. If present, the probabilities should be specified in the order of the factor levels.
- `usekernel`: if TRUE a kernel density estimate (density) is used for density estimation. If FALSE a normal density is estimated.
- `fL`: Factor for Laplace correction, default factor is 0, i.e. no correction.

Example (about spam)

Useful link: `https://www.youtube.com/watch?v=lB50qUQ8COw`

```
# Load Libraries
library(kernlab) # for spam data
library(e1071)
library(klaR)
# Load Data
data(spam)
```

```
# e1071 naiveBayes
set.seed(3456)
fit1 <- naiveBayes(spam, spam$type, type="raw")
pred1 <- predict(fit1, spam, type="class")
confusionMatrix(pred1, spam$type)
```

```
# klaR NaiveBayes
set.seed(3456)
fit2 <- NaiveBayes(spam, spam$type, usekernal = FALSE, fL = 0)
pred2 <- predict(fit2, spam)
# Warnings that probability is 0 for some cases
confusionMatrix(pred2$class, spam$type)
```

Output # e1071 naiveBayes

```
Confusion Matrix and Statistics
    Reference
```

Prediction	nonspam	spam
nonspam	1621	91
spam	1167	1722

```
Accuracy :  0.7266
95% CI : (0.7135, 0.7394)
No Information Rate :  0.606
P-Value [Acc > NIR] : < 2.2e-16
Kappa :  0.4813
Mcnemar's Test P-Value :  < 2.2e-16
```

Output # e1071 naiveBayes

```
Sensitivity :  0.5814
 Specificity :  0.9498
 Pos Pred Value :  0.9468
 Neg Pred Value :  0.5961
 Prevalence :  0.6060
 Detection Rate :  0.3523
 Detection Prevalence :  0.3721
 Balanced Accuracy :  0.7656
```

Output # klaR NaiveBayes

The same as if you use # e1071 naiveBayes.