Naive Bayes Classifier for Text

February 21, 2025

1 Problem description

Goal: Given a text, what is probability of the text been ham or spam? We will build this classifier based on Bayes' Theorem.

2 Machine learning preliminary

Naive Bayes classifiers are supervised machine learning algorithms used for classification tasks.

Supervised Learning: the machine learning algorithm uses labeled datasets to train algorithms predict new data.

Classification: the dataset labels are finite categorized. For example: ham and spam are two class classification labels. We also call it binary classification.

Training dataset: the subset of your dataset to train your algorithm. In this problem, it will be the dataset used to build the word probability table. Normally we take 80% of the total data.

Testing dataset: the subset of your dataset to test your algorithm. In this problem, it will be the dataset used to predict the probability of the text been ham or spam. Normally we take 20% of the total data. The train and test should not have any overlap.

3 Data Loader

Input of your Data Loader will be the whole dataset: $D = \{d_1, d_2, ..., d_i, ..., d_n\}$, where $d_i = \{\mathbf{x}, y\}$ is one line of text in test text file and n is number of data in the whole dataset; y is the label of this line of text (ham or spam) and $\mathbf{x}$ is the line of text (or a vector of words). After the first step of the process, the dataset become $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_i, ..., \mathbf{x}_n\}$ and $\mathbf{y} = \{y_1, y_2, ..., y_i, ..., y_n\}$, where $\mathbf{x}_i = \{x_i^1, x_i^2, ..., x_i^j, ..., x_i^m\}$ and x_i^j is j^{th} word (we also call it feature) in $\mathbf{x}_i$ and m is number of words in each line of text. Note that value of m may different from line to line. Therefore, after pre-process, you should have at least two list $X = [[x_1^1, x_1^2, ..., x_1^m], .., [x_n^1, x_n^2, ..., x_n^m]]$ and $\mathbf{y} = [y_1, ..., y_n]$. Then, we need to split the X and $\mathbf{y}$ into training (80%) and testing (20%). Let's name it $\{X_{train}, \mathbf{y}_{train}\}$ and $\{X_{test}, \mathbf{y}_{test}\}$. We are going to use $\{X_{train}, \mathbf{y}_{train}\}$ to train and obtain the probability table.

4 Naive Bayes Classifier

Let's start from Bayes' Theorem:
find the probability of an event given the probability of another event has already happened. Equation is:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)} \tag{1}$$

for our problem, we want to predict the probability of of text been ham or spam ($y_i = spam \mid y_i = ham$) given $\mathbf{x}_i$ by $max(P(y_i = spam|\mathbf{x}_i), P(y_i = ham|\mathbf{x}_i))$, it is computed as:

$$P(y = spam|\mathbf{x}_i) = \frac{P(\mathbf{x}_i|y = spam)P(y = spam)}{P(\mathbf{x}_i)} \tag{2}$$

$$P(y = ham|\mathbf{x}_i) = \frac{P(\mathbf{x}_i|y = ham)P(y = ham)}{P(\mathbf{x}_i)} \tag{3}$$

for those two equation, we find that $P(X)$ does not depend on y and value of X is given, it is constant. We can simply drop this term. Therefore, we have:

$$P(y|\mathbf{x}_i) = P(\mathbf{x}_i|y)P(y) \tag{4}$$

which is same as:

$$P(y|\mathbf{x}_i) = P(x_i^1, x_i^2, ..., x_i^m|y)P(y) \tag{5}$$

it is hard to compute. However, Naive Bayes Classifier has two assumptions, which will make this easier to work with. The first assumption is all features are conditionally independent. The second assumption is all features contribute equally to the outcome. Therefore, we can rewrite the equation as:

$$P(y|\mathbf{x}_i) = P(x_i^1|y) * P(x_i^2|y), ..., P(x_i^m|y)P(y) \tag{6}$$

same as:

$$P(y|\mathbf{x}_i) = P(y)\prod_{j=1}^{m} P(x_i^j|y) \tag{7}$$

for computer, it is easier to compute in log space to avoid underflow and increase speed. So we can rewrite the equation as:

$$P(y|\mathbf{x}_i) = logP(y) + \sum_{j=1}^{m} logP(x_i^j|y) \tag{8}$$

we can use word frequencies in the data to obtain the probabilities of $P(y)$ and $P(X|y)$. Therefore, the $P(y)$ computed as:

$$P(y = ham) = \frac{n_{ham}}{n} \tag{9}$$

where n_{ham} is number of lines in training data labeled with ham, and n is total number of lines in training data.

$$P(y = spam) = \frac{n_{spam}}{n} \tag{10}$$

where n_{spam} is number of lines in training data labeled with spam, and n is total number of lines in training data.

Now let's compute the $P(x_i^j|y)$:

$$\begin{aligned} P(x_i^j|y = ham) &= \frac{w_{ham}}{W_{ham}} \\ P(x_i^j|y = spam) &= \frac{w_{spam}}{W_{spam}} \end{aligned} \tag{11}$$

where w_{ham} is frequency of x_i^j in all ham text, w_{spam} is frequency of x_i^j in all spam text,W_{ham} is total number of words in all ham text and W_{spam} is total number of words in all spam text. You are going to use this equation to build your words probability table for spam and ham.

At prediction, you will predict your test dataset as:

$$P(y|\mathbf{x}_i) = \arg\max_{y \in Y} logP(y) + \sum_{j=1}^{m} logP(x_i^j|y) \tag{12}$$

5 Summary

Training: You will build spam and ham table.

```
# P(y)
logprior_ham = log(len(ham line)/ len(all lines))
logprior_spam = log(len(spam line)/ len(all lines))

ham_word_ct = {‘format’: 7, ‘expect’: 18, ‘delimiter’: 10,....}
spam_word_ct = {‘prints’: 6, ‘expect’: 10 ‘document’: 10,....}

# P(word|ham)
ham_word_proba = {‘class’: log(ham_word_ct['class']/len(ham_word_ct)),...}
# P(word|spam)
spam_word_proba = {‘class’: log(spam_word_ct['class']/len(spam_word_ct)),...}
```

Testing:

```
# predict probability of text been ham
for text in X_test:
    ham = logprior_ham
    spam = logprior_spam
    for word in text:
        if (word in ham_word_proba) and (word in spam_word_proba):
            ham += ham_word_proba[word]
            spam += spam_word_proba[word]

    if ham > spam:
        label the line of text as ham
    else:
        label the line of text as spam
```