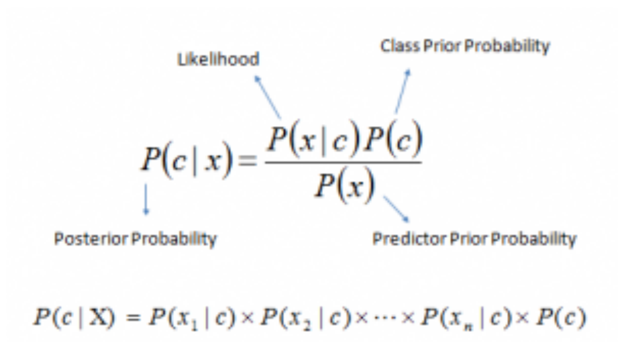
Naïve Bayes Classifier

Abstract

Text classification is one of the most important and typical task in supervised machine learning. Tasks such as assigning categories to documents like web pages, library books, media and news article, gallery etc. Naïve Bayes is one of such text classifier which uses probability technique based on Bayes theorem and it is widely used in applications like Spam filtering, email routing, sentiment analysis etc. I will explore this classification on Newsgroup dataset.

Concept

Naïve Bayes Theorem:



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

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

source: https://www.analyticsvidhya.com/wp-content/uploads/2015/09/Bayes_rule-300x172.png

- The way of reading Bayes Theorem is called “diachronic” interpretation
- It describes how the Hypothesis gets updated over time in light of new evidence
- In this context:
    - **P(c)**: Prior Probability which means probability of Class C before we see any Evidence (document) or Class prior probability
    - **P(c|x)**: Posterior Probability is the probability of Class C after we have seen the Evidence or document **x**
    - **P(x|c)**: Likelihood is the probability of seeing the document (or each token) assuming its true for Class C
    - **P(x):** Normalizing Content probability of seeing the document (token) under any circumstances whether its Class C is true or not

Naïve Bayes Classifier:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \hat{P}(c|d) = \arg\max_{c \in \mathbb{C}} \ \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)$$

- **arg max** is going through all the probability of each Class and picking the one with maximum possibility.
- **P(c):** Probability of each class
- **P(t_k|c):** Probability each token within a document from Sample Class C

- Probabilities $P(t_k|c)$ are multiplied throughout n_d where it is considered as the length of the document
- It is the measure of how much evidence token k contributes that C is the correct class
- The goal is to find the “best” class, which is the most likely Maximum posterior (MAP) class C_{map}
- Since multiplying lots of small probabilities can result in floating point underflow problem
- Since log(xy) = log(x) + log(y) we can sum log probabilities instead of multiplying
- So in practice we use:

$$c_{map} = \arg\max_{c\in\mathbb{C}} \left[\log \hat{P}(c) + \sum_{1\leq k\leq n_d} \log \hat{P}(t_k|c)\right]$$

- $\hat{P}(C) = \frac{Nc}{N}$ Nc: Number of docs in Class C , N: Number of total docs
- $\hat{P}(t \mid C) = \frac{T_{ct}}{\sum_{t\in V} T_{ct}}$ T_{ct}: Number of tokens of t in training doc from class C (multiple occurance is included)

OBERVATION & DATA INTERPRETATION

The data we are using here is **Newsgroup dataset** which consist of documents separated across 20 Class types

The dataset is in the form

- alt.athesim
    - text_file 1
    - text_file 2
    - …
    - text_file n
- comp.graphics
    - text_file 1 for comp.graphics
    - etc.
- etc….

The training and testing dataset are separated into two folders.

- Let’s consider text_file 1 under alt.athesim
- We retrieve the Tokens or each word from the file eg. I, am, not, looking, to ,buy, the, bibal, that, you, may, be, selling, I ,am, also, looking , for, another, version, of, it.
- We make use of removing the ‘Stop Words’ which are some of the most common words in the English vocabulary and we remove such words at the time of text preprocessing

- It does not negatively impact the performance of the system.
- And then we use the formula to calculate the probabilities of each word according to corresponding class

NOTE

- Not removing stop words yields around **79% to 80%** accuracy.
- Removing stop words gives us around **81.15% accuracy** which is a good improvement.
- We can yield a better result if I had used Laplace Smoothening which is basically considering the words or vocabulary used in the training sample and also counting the probability
- With more precise preprocessing techniques, I am sure we can yield a better result but as with most algorithms Naïve Bayes has its limitations too.
- It might give equal or more weightage to one or more word that is shared between two class documents and the algorithm might classify one over the other.
- The English dictionary has around 171,000 words and using all the words for every document is computationally expensive hence the limitation.
- To check the result of probabilities for the words from all docs from each Class or Category check the folder for csv file for detailed information.