DATA130030.01, Natural Language Processing (*15-25 minutes to read*) 03/13/2024

Lecture-03: Text classification

Lecturer: Baojian Zhou *The School of Data Science, Fudan University*

In this note, we will introduce the two linear models for text classification: Naive Bayes (NB) and Logistic Regression (LR).

3.1 Text classification

A text object (e.g., an article, email, passage) is presented as a feature vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_d]^\top \in \mathbb{R}^d$ where $\boldsymbol{x}$ has to capture all the information about the text item that the classifier needs. This text item $\boldsymbol{x}$ is associated with a label $y \in \mathcal{Y} = \{1, 2, \ldots, k\}$. Our goal is to learn a function (a classifier) f so that the input text items $\boldsymbol{x} \in \mathcal{X}$ can be mapped into labels $y \in \mathcal{Y}$ (e.g., $\mathcal{Y} = \{\text{spam}, \text{not spam}\}$ for spam detection). Let us assume the training dataset is $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$ and the learning algorithm returns a model $\boldsymbol{\theta}$.

3.2 Naive Bayes (NB) classifier

The NB classifier is a probabilistic classifier that provides posterior probability given the text item $\boldsymbol{x}$, i.e., $p(y; \boldsymbol{x})$. Specifically, the probability that the class label is y when the input feature vector is $\boldsymbol{x}$, we want to find the most likely classifier y given the input feature vector $\boldsymbol{x}$:

$$\hat{y} \in \underset{y \in \mathcal{Y}}{\arg\max} \quad P(Y = y \mid X = \boldsymbol{x}). \tag{3.1}$$

The key to the NB classifier is to build a model based on the Bayes Rule. Recall that given two random variable X and Y, by noticing $P(Y \mid X) = P(Y, X)/P(X)$, the Bayes rule can be written as

$$\begin{aligned}\underbrace{P(Y \mid X)}_{\text{posterior}} &= \frac{P(Y, X)}{P(X)} = \frac{\overbrace{P(X \mid Y)}^{\text{likelihood}} \cdot \overbrace{P(Y)}^{\text{prior}}}{\underbrace{P(Y)}_{\text{evidence}}} \\ &\propto P(X \mid Y) \cdot P(Y),\end{aligned} \tag{3.2}$$

where it basic says the posterior $P(Y \mid X)$ is proportional to the prior $P(Y)$ times the likelihood $P(X \mid Y)$. The notion $\propto$ means that $P(X)$ is a constant in term of maximizing $P(Y = y \mid X = \boldsymbol{x})$ in (3.1). The word "naive" comes from its assumption of independence between feature variables within the model. This is often considered *naive* because the features are usually not completely independent of each other in real-world data. Specifically, in a typical naive Bayes classifier, there is a bag-of-words assumption that the word's position in a document does not matter and a conditional independence assumption that all words in a document are independent conditional on the class label. That is,

$$P\left(X = \boldsymbol{x}_i \mid y_i\right) \approx \prod_{w_j \in w_{1:t}} P\left(w_j \mid y_i\right), \tag{3.3}$$

where $w_{1:t}$ is the list of words in document with $w_i \in \mathcal{V}$. Substituting this assumption into (3.2), we have

$$\hat{y}_i \approx \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \; P(y) \cdot \prod_{w_j \in w_{1:t}} P\left(w_j \mid y\right).$$

By adding log as we did in previous lecture, we refer the following as the NB classifier

$$\hat{y}_i = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \left\{ \log P(y) + \sum_{w_j \in w_{1:t}} \log P\left(w_j \mid y\right) \right\}. \tag{3.4}$$

3.2.1 Estimation based on training data $\mathcal{D}$.

Clearly, we need to estimate $P(y)$ and $\log P\left(v_j \mid y\right)$ for $y \in \mathcal{Y}$ and $v_j \in \mathcal{V}$ based on $\mathcal{D}$. $|\mathcal{Y}| + |\mathcal{Y}| \times |\mathcal{V}|$ parameters to estimate. Now, we will calculate these estimates one by one using MLE. Denote an estimate with a hat, e.g., $\hat{P}(\cdot)$. The estimate of the prior is[1]

$$\hat{P}(y) = \frac{\sum_{i=1,2,\ldots,n} \mathbf{1}\left(y_i = y\right)}{n}$$

where $\mathbf{1}$ is the indicator function and $n = |\mathcal{D}|$. Similarly, for any word type v_j, we calculate the estimates across all documents

$$\hat{P}\left(v_i \mid y\right) = \frac{\operatorname{Count}\left(v_i, y\right)}{\sum_{v_j \in \mathcal{V}} \operatorname{Count}\left(v_j, y\right)},$$

where $\operatorname{Count}\left(v_i, y\right)$ is the number of tokens for which the word type v_i co-occurs with class label y among all documents. To avoid $\hat{P}\left(v_i \mid y\right) = 0$, we use smoothing method such as Laplace smoothing

$$\hat{P}\left(v_i \mid y\right) = \frac{\operatorname{Count}\left(v_i, y\right) + 1}{\sum_{v_j \in \mathcal{V}} \operatorname{Count}\left(v_j, y\right) + |\mathcal{V}|}.$$

3.3 Logistic Regression (LR)

The LR classifier is derived from the logistic function, which is defined as

$$\sigma(z) = \frac{1}{1 + e^{-z}}. \tag{3.5}$$

In text classification, the parameter z is the linear combination of the input features $\boldsymbol{x}$ of a document d and the model weights (parameters) $\boldsymbol{w}$, plus a bias term b. So, z can be written as $z = \boldsymbol{x}^\top \boldsymbol{w} + b$. Logistic regression (LR) uses this logistic function to predict the probability (P) that a given input $\boldsymbol{x}$ belongs to the default category (often labeled as 1):

$$P(y = 1 \mid \boldsymbol{x}) = \frac{1}{1 + e^{-(\boldsymbol{x}^\top \boldsymbol{w} + b)}}, \tag{3.6}$$

where the output of the LR model is thus the probability that the input $\boldsymbol{x}$ belongs to category 1. The name "logistic regression" is somewhat historical and can be misleading, as it is used for classification rather than regression in the traditional sense. The term "logistic" comes from the logistic function used to model the probability of binary outcomes, and "regression" refers to the method's origin in statistical methods used to predict a continuous outcome.

Logistic regression can be extended to multi-label cases. Let there be K classes. Each class has its own parameter $\boldsymbol{\theta}_k$, which maps $\boldsymbol{x}$ to $\boldsymbol{\theta}_k^\top \boldsymbol{x} + b_k$ where b_k is the bias for class k. The probability is defined via the softmax function

$$P(y = k \mid \boldsymbol{x}) = \frac{\exp\left(\boldsymbol{\theta}_k^\top \boldsymbol{x} + b_k\right)}{\sum_{i=1}^{K} \exp\left(\boldsymbol{\theta}_i^\top \boldsymbol{x} + b_k\right)}, \quad k = 1, 2, \ldots, K. \tag{3.7}$$

[1] Recall our previous lecture notes, we show that MLE for these probability estimations can be calculated by their relative frequencies.

3.3.1 Estimation of parameters

To find $\boldsymbol{\theta}$ and b for binary case[2], we can maximize the conditional log likelihood of training data $\mathcal{D} := \{(\boldsymbol{x}_i, y_i) : i = 1, 2, \ldots, n\}$:

$$\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log P\left(y_i \mid \boldsymbol{x}_i; \boldsymbol{\theta}\right).$$

However, two bad things happen when the training data is linearly separable: 1. $\|\boldsymbol{\theta}\|$ goes to infinity; 2. There are an infinite number of MLEs. To see this, note any step function (sigmoid with $\|\boldsymbol{\theta}\| = \infty$) that is in the gap between the two classes is an MLE. One way to avoid this is to incorporate a prior on $\boldsymbol{\theta}$ in the form of a zero-mean Gaussian with covariance $\frac{1}{2\lambda}\boldsymbol{I}$,

$$\boldsymbol{\theta} \sim \mathcal{N}\left(0, \frac{1}{2\lambda}\boldsymbol{I}\right).$$

Hence, we can seek the MAP estimate. This is smoothing since large $\boldsymbol{\theta}$ values will be penalized more. That is, (the following case is when $y_i \in \{\pm 1\}$)

$$\begin{aligned}
&\max_{\theta} \log P\left(\boldsymbol{\theta}, y_{1:n} \mid \boldsymbol{x}_{1:n}\right) \\
=& \log P(\boldsymbol{\theta}) + \sum_{i=1}^{n} \log P\left(y_i \mid x_i, \boldsymbol{\theta}\right) \\
=& -\lambda \|\boldsymbol{\theta}\|^2 + \sum_{i=1}^{n} \log p\left(y_i \mid x_i, \boldsymbol{\theta}\right) \\
=& -\lambda \|\boldsymbol{\theta}\|^2 - \sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i \boldsymbol{\theta}^\top x_i\right)\right).
\end{aligned}$$

Equivalently, one minimizes the ℓ_2-regularized negative log likelihood loss

$$\min_{\boldsymbol{\theta}} \lambda \|\boldsymbol{\theta}\|^2 + \sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i \boldsymbol{\theta}^\top x_i\right)\right)$$

3.4 References

- 1. `https://pages.cs.wisc.edu/~jerryzhu/cs838/LR.pdf`
- 2. `https://www.cs.williams.edu/~kkeith/teaching/s23/cs375/attach/2-naive-bayes.pdf`

[2]We will redefine $\boldsymbol{\theta} = [\boldsymbol{\theta}, b]$ so that b can be written into $\boldsymbol{\theta}$.