Data Science 2 - Naive Bayes

February 22, 2022

Data Science 2 - Naive Bayes

Faculty of Mathematics and Physics

Naive Bayes
Introduction

- simple algorithm for binary or multi-class clasification
- suppose the target variable y and features $x_j, j = 1, \ldots, J$
- assume that the features x_j are conditionally independent given y:

$$\mathbb{P}[x_i|y] = \mathbb{P}\left[x_i|y, x_j\right], j \neq i$$

Now we want to model $\mathbb{P}[x_1, \ldots, x_J|y]$:

$$\begin{aligned}
\mathbb{P}[x_1, \ldots, x_J|y] &= \mathbb{P}[x_1|y]\,\mathbb{P}[x_2|y, x_1]\cdots\mathbb{P}[x_J|y, x_1, \ldots, x_{J-1}] \\
&= \mathbb{P}[x_1|y]\,\mathbb{P}[x_2|y]\cdots\mathbb{P}[x_J|y] \\
&= \prod_{j=1}^{J}\mathbb{P}\left[x_j|y\right]
\end{aligned} \tag{1}$$

Naive Bayes

Model

- we apply Bayes theorem:

$$\mathbb{P}\left[y = k | x_1, \ldots, x_J\right] = \frac{\mathbb{P}\left[x_1, \ldots, x_J | y = k\right] \mathbb{P}\left[y = k\right]}{\mathbb{P}\left[x_1, \ldots, x_J\right]}$$

- since $\mathbb{P}\left[x_1, \ldots, x_J\right]$ is constant, and does not depend on the class $y = k$, we model only the numerator:

$$\begin{aligned} \mathbb{P}\left[y = k | x_1, \ldots, x_J\right] &\propto \mathbb{P}\left[x_1, \ldots, x_J | y = k\right] \mathbb{P}\left[y = k\right] \\ &\propto \mathbb{P}\left[y = k\right] \prod_{j=1}^{J} \mathbb{P}\left[x_j | y = k\right] \end{aligned} \tag{2}$$

- to classify an obervation, we just take the class with highest probability:

$$\hat{y} = \underset{k=1,\ldots,K}{\arg\max}\, \mathbb{P}\left[y = k\right] \prod_{j=1}^{J} \mathbb{P}\left[x_j | y = k\right]$$

Naive Bayes

Application

- The classes prior $\mathbb{P}\left[y = k\right]$ is usually calculated as the share of each class in the training set or taken equiprobable $\mathbb{P}\left[y = k\right] = \frac{1}{K}$
- Feature models:
  - Continuous features: assume normal distribution or discretize them (grouping) -> Bernoulli model
  - Bernoulli model: features represent occurence of the word in a text
  - Multinomial model: features represent frequency, such as number of times the word occured in a text

Naive Bayes
Gaussian Naive Bayes

- Suppose we have N observations with y_i and $\boldsymbol{x}_i$ respectively
- For each of the classes $k = 1, \ldots, K$, calculate mean and variance of the feature x_j:

$$\begin{aligned} c_k &= \sum_{i:\, y_i = k} 1 \\ \mu_{kj} &= \frac{1}{c_k} \sum_{i:\, y_i = k} x_{ij} \\ \sigma_{kj}^2 &= \frac{1}{c_k - 1} \sum_{i:\, y_i = k} (x_{ij} - \mu_k)^2 \end{aligned} \tag{3}$$

- Now we have with normal distribution assumption:

$$\mathbb{P}\left[x_j = x | y = k\right] = \frac{1}{\sqrt{2\pi\sigma_{kj}^2}} \exp\left\{-\frac{(x - \mu_{kj})^2}{2\sigma_{kj}^2}\right\}$$

Naive Bayes

Bernoulli Naive Bayes

- For each class, we suppose that vector $(p_{k1}, \ldots, p_{kJ})$ represents the probability that event occurs for the features $x_1, \ldots, x_J$:

$$\mathbb{P}\left[x_1, \ldots, x_J | y = k\right] = \prod_{j=1}^{J} p_{kj}^{x^j} (1 - p_{kj})^{(1-x^j)}$$

- probabilities are estimated from the train set by observed events:

$$p_{kj} = \frac{1}{|i:\ y_i = k|} \sum_{i:\ y_i = k} x_{ij}$$

- If a given class and feature do not occur in the train set, the corresponding estimate would be zero, which would eliminate all other terms:
  - Laplacean smoothing: introduce at least one observation to each feature:

$$p_{kj} = \frac{1 + \sum_{i:\ y_i = k} x_{ij}}{J + |i:\ y_i = k|}$$

  - Lidstone smooting (mean target encoding)
  - Apply tf-idf, for instance in text classification

Naive Bayes
Multinomial Naive Bayes

- For each class, we suppose that multinomial vector $(p_{k1}, \ldots, p_{kJ})$ represents the probability of event happening in the features $x_1, \ldots, x_J$:

$$\mathbb{P}\left[x_1, \ldots, x_J | y = k\right] = \frac{(\sum_{j=1}^{J} x_j)!}{\prod_{j=1}^{J} x_j!} \prod_{j=1}^{J} p_{kj}^{x_j}$$

- probabilities are estimated from the train set by observed frequencies:

$$p_{kj} = \frac{\sum_{i:\, y_i=k} x_{ij}}{\sum_{i:\, y_i=k} \sum_{l=1}^{J} x_{il}}$$

- If a given class and feature do not occur in the train set, the corresponding estimate would be zero, which would eliminate all other terms:
  - Laplacean smoothing
  - Lidstone smooting (mean target encoding)
  - Apply tf-idf, for instance in text classification

Naive Bayes
Multinomial Naive Bayes

- In logarithm form this estimator becomes a linear model:

$$
\begin{aligned}
\log \mathbb{P}\left[y = k | x_1, \ldots, x_J\right] &\propto \mathbb{P}\left[y = k\right] \log \left\{ \frac{\left(\sum_{j=1}^{J} x_j\right)!}{\prod_{j=1}^{J} x_j!} \prod_{j=1}^{J} p_{kj}^{x^j} \right\} \\
&\propto \log \mathbb{P}\left[y = k\right] + \log \left\{ \prod_{j=1}^{J} p_{kj}^{x^j} \right\} \\
&\propto \log \mathbb{P}\left[y = k\right] + \sum_{j=1}^{J} x^j \log p_{kj} \\
&\propto \alpha_k + \sum_{j=1}^{J} x^j \beta_{kj}
\end{aligned} \tag{4}
$$

Naive Bayes
Summary

- Built on a strong assumption often not fulfilled in practive
- However, it provides good predictions even for general cases
- Quick and simple model for building the first prediction
- Suitable for use in text mining
- Highly scalable and efficient training of the model
- Suitable for train sets with low number of observations

Thank you!

TARAN

ADVISORY IN DATA & ANALYTICS