Document Classification: Part II

Statistical Analysis and Document Mining

Spring 2021

Vasilii Feofanov

Université Grenoble Alpes

vasilii.feofanov@univ-grenoble-alpes.fr

Outline

1 Introduction
1.1 Framework
1.2 Bayes Classifier

2 Naive Bayes Classifier
2.1 Principle
2.2 Bernoulli Distribution
2.3 Multinomial Distribution

3 Other Word Representations
3.1 N-grams
3.2 Neural Network Approaches

Document Classification Scheme

- Last time, we learned how to pre-process the text data and represent it in a vectorial space using the bag-of-words and various weighting rules.
- The next step is to learn a classification model using vectorized documents.

Document Classification Task

- Each observation is a document $\mathbf{d} = (w_{j,d})_{i=1}^{V}$, where
  - V is a vocabulary size;
  - $w_{j,d}$ is a value (importance) corresponding to the term t_j.

Document Classification Task

- Each observation is a document $\mathbf{d} = (w_{j,d})_{i=1}^{V}$, where
  - V is a vocabulary size;
  - $w_{j,d}$ is a value (importance) corresponding to the term t_j.
- Training set: $S = \{\mathbf{d}_i, y_i\}_{i=1}^{n}$, where
  - $\mathbf{d}_i \in \mathbb{R}^V$;
  - $y_i \in \{1, \ldots, K\}$.

- Each observation is a document $\mathbf{d} = (w_{j,d})_{i=1}^{V}$, where
  - V is a vocabulary size;
  - $w_{j,d}$ is a value (importance) corresponding to the term t_j.
- Training set: $S = \{\mathbf{d}_i, y_i\}_{i=1}^{n}$, where
  - $\mathbf{d}_i \in \mathbb{R}^V$;
  - $y_i \in \{1, \dots, K\}$.
- *Target*: minimise the misclassification error:

$$P(h(\mathbf{D}) \neq Y) = \sum_{c \in \{1,\dots,K\}} P(Y = c) P(h(\mathbf{D}) \neq c | Y = c).$$

Reminder: Bayes' Rule

$$\underbrace{P(Y|\mathbf{D})}_{\text{Posterior}} = \frac{\overbrace{P(\mathbf{D}|Y)}^{\text{Likelihood}}\,\overbrace{P(Y)}^{\text{Class Prior}}}{\underbrace{P(\mathbf{D})}_{\text{Evidence}}}$$

Bayes Classifier

- The Bayes classifier predicts a class with the highest posterior probability:

$$h_B(\mathbf{d}) := \operatorname*{argmax}_{y \in \mathcal{Y}} P(Y = y | \mathbf{D} = \mathbf{d}).$$

- This is equivalent to:

$$\begin{aligned} h_B(\mathbf{d}) &\propto \operatorname*{argmax}_{y \in \mathcal{Y}} P(\mathbf{D} = \mathbf{d} | Y = y) P(Y = y) \\ &\propto \operatorname*{argmax}_{y \in \mathcal{Y}} \log P(\mathbf{D} = \mathbf{d} | Y = y) + \log P(Y = y). \end{aligned}$$

Outline

1 Introduction
1.1 Framework
1.2 Bayes Classifier

2 Naive Bayes Classifier
2.1 Principle
2.2 Bernoulli Distribution
2.3 Multinomial Distribution

3 Other Word Representations
3.1 N-grams
3.2 Neural Network Approaches

Naive Bayes Classifier

- *Problem:* Since V is usually large, it is hard to estimate $P(\mathbf{D}|Y)$.

- *Problem:* Since V is usually large, it is hard to estimate $P(\mathbf{D}|Y)$.
- *Idea:* We assume *naively* that terms are conditionally *independent* given the class.

- *Problem:* Since V is usually large, it is hard to estimate $P(\mathbf{D}|Y)$.
- *Idea:* We assume *naively* that terms are conditionally *independent* given the class.
- Then, denoting $\mathbf{D} = (W_1, \ldots, W_V)$, $\mathbf{d} = (w_1, \ldots, w_V)$, we obtain that:

$$P(\mathbf{D} = \mathbf{d}|Y = c) = \prod_{j=1}^{V} P(W_{j,D} = w_{j,d}|Y = c).$$

Naive Bayes Classifier

- *Problem:* Since V is usually large, it is hard to estimate $P(\mathbf{D}|Y)$.
- *Idea:* We assume *naively* that terms are conditionally *independent* given the class.
- Then, denoting $\mathbf{D} = (W_1, \ldots, W_V)$, $\mathbf{d} = (w_1, \ldots, w_V)$, we obtain that:

$$P(\mathbf{D} = \mathbf{d}|Y = c) = \prod_{j=1}^{V} P(W_{j,D} = w_{j,d}|Y = c).$$

- Thus, the *naive Bayes classifier* is defined in the following way:

$$h_B(\mathbf{d}) := \operatorname*{argmax}_{c\in\mathcal{Y}} P(Y = c) \prod_{j=1}^{V} P(W_{j,D} = w_{j,d}|Y = c)$$

$$\propto \operatorname*{argmax}_{c\in\mathcal{Y}} \log P(Y = c) + \sum_{j=1}^{V} \log P(W_{j,D} = w_{j,d}|Y = c).$$

Gaussian Naive Bayes Classifier

- We assume that the j-th feature of observations from the class $c \in \mathcal{Y}$ is normally distributed[1]:

$$[W_{j,D} | Y = c] \sim \mathcal{N}(\mu_{j,c}, \sigma_{j,c}).$$

[1] It is appropriate rather for the tf-idf rule than the tf and the binary ones.

- We assume that the j-th feature of observations from the class $c \in \mathcal{Y}$ is normally distributed[1]:

$$[W_{j,D}|Y = c] \sim \mathcal{N}(\mu_{j,c}, \sigma_{j,c}).$$

- Then, the distribution of the class $c \in \mathcal{Y}$ is defined as:

$$P(\mathbf{D} = \mathbf{d}|Y = c) = \prod_{j=1}^{V} \frac{1}{\sqrt{2\pi}\sigma_{j,c}} e^{-\frac{(w_{j,d}-\mu_{j,c})^2}{2\sigma_{j,c}^2}}.$$

[1] It is appropriate rather for the tf-idf rule than the tf and the binary ones.

- We assume that the j-th feature of observations from the class $c \in \mathcal{Y}$ is normally distributed[1]:

$$[W_{j,D}|Y = c] \sim \mathcal{N}(\mu_{j,c}, \sigma_{j,c}).$$

- Then, the distribution of the class $c \in \mathcal{Y}$ is defined as:

$$P(\mathbf{D} = \mathbf{d}|Y = c) = \prod_{j=1}^{V} \frac{1}{\sqrt{2\pi}\sigma_{j,c}} e^{-\frac{(w_{j,d}-\mu_{j,c})^2}{2\sigma_{j,c}^2}}.$$

- Finally, the Gaussian naive Bayes classifier is defined as:

$$h_B(\mathbf{d}) := \operatorname*{argmax}_{c\in\mathcal{Y}} \left[\ln P(Y = c) - \sum_{j=1}^{V} \ln \sigma_{j,c} - \sum_{j=1}^{V} \frac{(w_{j,d} - \mu_{j,c})^2}{2\sigma_{j,c}^2}\right].$$

[1] It is appropriate rather for the tf-idf rule than the tf and the binary ones.

- What is the time complexity of the training phase?

Gaussian Naive Bayes: Discussion

- What is the time complexity of the training phase?
- Compare the algorithm with the linear discriminant analysis in terms of:
    - the computation speed;

Gaussian Naive Bayes: Discussion

- What is the time complexity of the training phase?
- Compare the algorithm with the linear discriminant analysis in terms of:
  - the computation speed;
  - the number of algorithm's parameters to estimate;

Gaussian Naive Bayes: Discussion

- What is the time complexity of the training phase?
- Compare the algorithm with the linear discriminant analysis in terms of:
  - the computation speed;
  - the number of algorithm's parameters to estimate;
  - the behaviour on data sets of large dimension (vocabulary);

Gaussian Naive Bayes: Discussion

- What is the time complexity of the training phase?
- Compare the algorithm with the linear discriminant analysis in terms of:
  - the computation speed;
  - the number of algorithm's parameters to estimate;
  - the behaviour on data sets of large dimension (vocabulary);
  - the realism of the naive assumption.

Bernoulli Naive Bayer Classifier

- The *binary weighting rule*: If term t_j is present in the document d_i, then $w_{j,d_i} = 1$. Otherwise, $w_{j,d_i} = 0$.

Bernoulli Naive Bayer Classifier

- The *binary weighting rule*: If term t_j is present in the document d_i, then $w_{j,d_i} = 1$. Otherwise, $w_{j,d_i} = 0$.
- With the conditional independence assumption, how terms are distributed in this case?

- The *binary weighting rule*: If term t_j is present in the document d_i, then $w_{j,d_i} = 1$. Otherwise, $w_{j,d_i} = 0$.
- With the conditional independence assumption, how terms are distributed in this case?
  - We obtain that $W_{j,D}|Y = c$ is distributed acc. to the *Bernoulli* law with the success and the failure probabilities defined as:

$$P(W_{j,D} = 1|Y = c) = p_{t_j|c};$$
$$P(W_{j,D} = 0|Y = c) = 1 - p_{t_j|c}.$$

Bernoulli Naive Bayer Classifier

- The *binary weighting rule*: If term t_j is present in the document d_i, then $w_{j,d_i} = 1$. Otherwise, $w_{j,d_i} = 0$.
- With the conditional independence assumption, how terms are distributed in this case?
  - We obtain that $W_{j,D}|Y = c$ is distributed acc. to the *Bernoulli* law with the success and the failure probabilities defined as:

$$P(W_{j,D} = 1|Y = c) = p_{t_j|c};$$
$$P(W_{j,D} = 0|Y = c) = 1 - p_{t_j|c}.$$

- Then, the Bernoulli naive Bayes classifier is defined as:

$$h_{NB}(\mathbf{d}) := \operatorname*{argmax}_{c \in \mathcal{Y}} P(Y = c) \prod_{j=1}^{V} P(W_{j,D} = w_{j,d}|Y = c)$$
$$= \operatorname*{argmax}_{c \in \mathcal{Y}} P(Y = c) \prod_{j=1}^{V} p_{t_j|c}^{w_{j,d}} (1 - p_{t_j|c})^{1 - w_{j,d}}.$$

Estimation of Bernoulli NB Parameters

- The prior estimation: # of docs from the class divided by the total # of docs,

$$P(Y = c) \leftarrow \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(y_i = c) =: \frac{n_c}{n}.$$

Estimation of Bernoulli NB Parameters

- The prior estimation: # of docs from the class divided by the total # of docs,

$$P(Y = c) \leftarrow \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(y_i = c) =: \frac{n_c}{n}.$$

- Estimation of $p_{t_j|c}$: document frequency of the term t_j among docs from the class c divided by n_c,

$$\hat{p}_{t_j|c} = \frac{\sum_{i=1}^{n}\mathbb{I}(w_{j,d_i} = 1 \wedge y_i = c)}{\sum_{i=1}^{n}\mathbb{I}(y_i = c)} =: \frac{\mathrm{df}_{t_j}(c)}{n_c}.$$

Estimation of Bernoulli NB Parameters

- The prior estimation: # of docs from the class divided by the total # of docs,

$$P(Y=c) \leftarrow \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(y_i = c) =: \frac{n_c}{n}.$$

- Estimation of $p_{t_j|c}$: document frequency of the term t_j among docs from the class c divided by n_c,

$$\hat{p}_{t_j|c} = \frac{\sum_{i=1}^{n} \mathbb{I}(w_{j,d_i} = 1 \wedge y_i = c)}{\sum_{i=1}^{n} \mathbb{I}(y_i = c)} =: \frac{\mathrm{df}_{t_j}(c)}{n_c}.$$

- What happens when $\hat{p}_{t_j|c} = 0$?

Bernoulli NB with Laplace Smoothing

- *Laplace smoothing:* to avoid situations when $\hat{p}_{t_j|c} = 0$, we estimate $\hat{p}_{t_j|c}$ by:

$$\hat{p}_{t_j|c} = \frac{\mathrm{df}_{t_j}(c) + 1}{n_c + 2}.$$

Bernoulli NB with Laplace Smoothing

- *Laplace smoothing:* to avoid situations when $\hat{p}_{t_j|c} = 0$, we estimate $\hat{p}_{t_j|c}$ by:

$$\hat{p}_{t_j|c} = \frac{\mathrm{df}_{t_j}(c) + 1}{n_c + 2}.$$

- Considering the log scale, we predict the label for new $\mathbf{d}$ by the following rule:

$$h_{NB}(\mathbf{d}) := \operatorname*{argmax}_{c \in \mathcal{Y}} [\ln \frac{n_c}{n} + \sum_{\substack{j=\{1,\dots,V\} \\ t_j \in d}} \ln \hat{p}_{t_j|c} + \sum_{\substack{j=\{1,\dots,V\} \\ t_j \notin d}} \ln(1 - \hat{p}_{t_j|c})].$$

Term Frequency Distribution

- *Term frequency rule:* importance of the term t_j is defined by its number of occurrences in the document:

$$w_{j,d} = \mathrm{tf}_{t_j,d}, \quad \sum_{j=1}^{V} \mathrm{tf}_{t_j,d} = N_d.$$

- *Term frequency rule:* importance of the term t_j is defined by its number of occurrences in the document:

$$w_{j,d} = \mathrm{tf}_{t_j,d}, \quad \sum_{j=1}^{V} \mathrm{tf}_{t_j,d} = N_d.$$

- *Naive assumption*: all N_d words of $\mathbf{d}$ are *i.i.d.* trials with the output $\in \{t_1, \dots, t_V\}$ and the following distribution:

$$P(\text{word} = t_j) = p_{t_j|c},\ j = \{1, \dots, V\},\ \sum_{j=1}^{V} p_{t_j|c} = 1.$$

- *Term frequency rule:* importance of the term t_j is defined by its number of occurrences in the document:

$$w_{j,d} = \mathrm{tf}_{t_j,d}, \quad \sum_{j=1}^{V} \mathrm{tf}_{t_j,d} = N_d.$$

- *Naive assumption*: all N_d words of $\mathbf{d}$ are *i.i.d.* trials with the output $\in \{t_1, \ldots, t_V\}$ and the following distribution:

$$P(\text{word} = t_j) = p_{t_j|c}, \; j = \{1, \ldots, V\}, \; \sum_{j=1}^{V} p_{t_j|c} = 1.$$

- Then, the r.v. $\mathbf{D}|Y = c$ is distributed acc. to the multinomial distribution with parameters $(p_{t_j|c})_{j=1}^{V}$:

$$P(\mathbf{D} = \mathbf{d}|Y = c) = \frac{N_d!}{w_{1,d}! \cdots w_{V,d}!} p_{t_1|c}^{w_{1,d}} \cdots p_{t_V|c}^{w_{V,d}}.$$

Multinomial Naive Bayes Classifier

- Then, we write:

$$P(\mathbf{D} = \mathbf{d} | Y = c) \propto \prod_{j=1}^{V} p_{t_j|c}^{w_{j,d}}.$$

Multinomial Naive Bayes Classifier

- Then, we write:

$$P(\mathbf{D} = \mathbf{d} | Y = c) \propto \prod_{j=1}^{V} p_{t_j|c}^{w_{j,d}}.$$

- We obtain the Multinomial naive Bayes decision rule:

$$\begin{aligned} h_{NB}(\mathbf{d}) &:= \operatorname*{argmax}_{c \in \mathcal{Y}} [P(Y = c) \prod_{j=1}^{V} p_{t_j|c}^{w_{j,d}}] \\ &\propto \operatorname*{argmax}_{c \in \mathcal{Y}} [\ln P(Y = c) + \sum_{j=1}^{V} w_{j,d} \ln p_{t_j|c}]. \end{aligned}$$

Estimation of Multinomial NB Parameters

- The prior estimation: # of docs from the class divided by the total # of docs,

$$P(Y = c) \leftarrow \frac{n_c}{n}.$$

Estimation of Multinomial NB Parameters

- The prior estimation: # of docs from the class divided by the total # of docs,

$$P(Y = c) \leftarrow \frac{n_c}{n}.$$

- Estimation of $\hat{p}_{t_j|c}$: the sum of term frequencies of the term t_j over the docs from the class c divided by the total # of words (*not terms*) in all documents from the class c,

$$\hat{p}_{t_j|c} = \frac{\sum_{i=1}^{n} \mathrm{tf}_{t_j,d_i} \mathbb{I}(y_i = c)}{\sum_{j'=1}^{V} \sum_{i=1}^{n} \mathrm{tf}_{t_{j'},d_i} \mathbb{I}(y_i = c)}.$$

Multinomial NB with Laplace Smoothing

- *Laplace smoothing:* we estimate $\hat{p}_{t_j|c}$ by:

$$\hat{p}_{t_j|c} = \frac{\sum_{i=1}^{n} \mathrm{tf}_{t_j,d_i} \mathbb{I}(y_i = c) + 1}{\sum_{j'=1}^{V} \sum_{i=1}^{n} \mathrm{tf}_{t_{j'},d_i} \mathbb{I}(y_i = c) + V}.$$

- *Laplace smoothing:* we estimate $\hat{p}_{t_j|c}$ by:

$$\hat{p}_{t_j|c} = \frac{\sum_{i=1}^{n} \mathrm{tf}_{t_j,d_i} \mathbb{I}(y_i = c) + 1}{\sum_{j'=1}^{V} \sum_{i=1}^{n} \mathrm{tf}_{t_{j'},d_i} \mathbb{I}(y_i = c) + V}.$$

- We classify new $\mathbf{d}$ by the following rule:

$$h_{NB}(\mathbf{d}) := \operatorname*{argmax}_{c \in \mathcal{Y}} [\ln \frac{n_c}{n} + \sum_{j=1}^{V} w_{j,d} \ln \hat{p}_{t_j|c}].$$

Outline

1 Introduction
1.1 Framework
1.2 Bayes Classifier

2 Naive Bayes Classifier
2.1 Principle
2.2 Bernoulli Distribution
2.3 Multinomial Distribution

3 Other Word Representations
3.1 N-grams
3.2 Neural Network Approaches

Drawback of the BoW

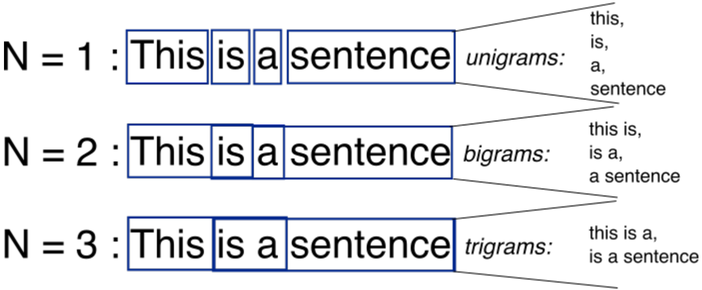
By using the bag-of-words, we lose some important information from text including:

- The word order:

  Trump supports Americans. $\neq$ *Americans support Trump.*

- Words similarities (sometimes, we want to have similar words close to each other in the vectorial space);
- The context, particularly, one word may have different meanings:
  - *The weather is <u>cold</u>.*
  - *Blue is a <u>cold</u> color.*
  - *I got a <u>cold</u> yesterday!*

- Several words could be more important when they appear together.
- When $N = 1$, we have the bag-of-words.
- The approach is very costly when V is large.



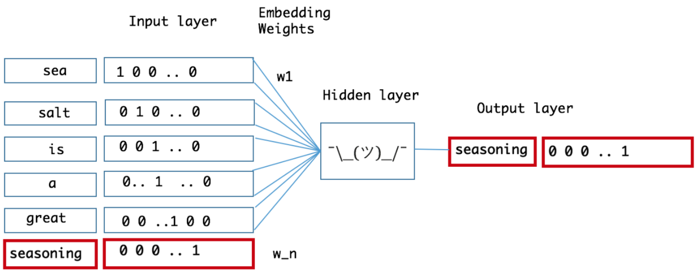
- The approach is often used in bioinformatics to extract features from nucleotide sequences.

. . . ATTACACGGTGACCAACCTATT. . .

Gram	Frequency
ATTA	4
GACC	3
CGGT	5
. . .	. . .

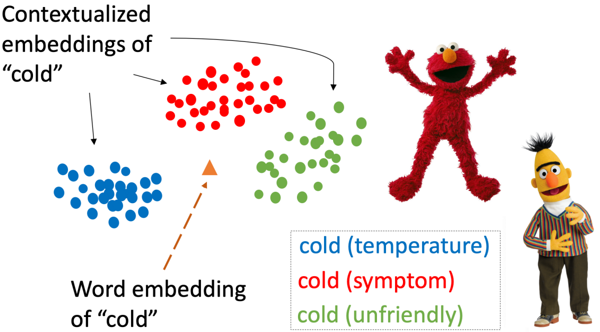
Word2vec (Mikolov et al., 2013)

- Continuous BoW: we learn a neural network s.t. given a word's context (some window of neighbours words) outputs this word. The weight matrix of the hidden layer will correspond to the representation of words.
- Due to the task complexity, in practice, one uses a model already pre-trained on a large text corpus.



Deep Contextualized Representations

- Nowadays: a token's representation is a function of the input sentence (in Word2vec, it is averaged across all sentences).
- By doing this, models like ELMO[2] and BERT[3] can capture aspects of word meaning that are context-dependent.



[2] (Peters et al., 2018) Deep contextualized word representations.

[3] (Devlin et al., 2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.