Naive Bayes Classifier

- Sentiment Analysis Task
- Experiment on IMDB, movie review dataset

Jihyuk Kim

Recap: Naive Bayes Classifier (NBC)

- Generative Classifier

$$p(y \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y)p(y)}{p(\mathbf{x})}$$

$$p(y \mid \mathbf{x}) \propto p(\mathbf{x} \mid y) \times p(y)$$

posterior

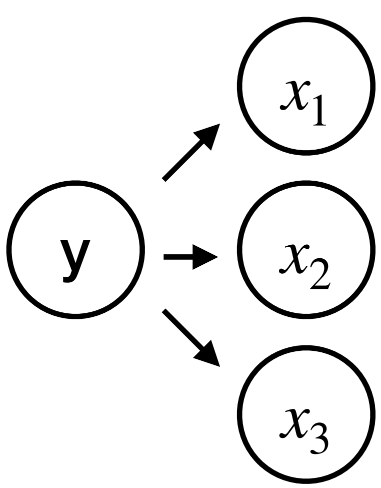
likelihood

prior

- Conditional Independence Assumption

  - Suppose we want to predict posterior $p(y \mid \mathbf{x})$ using three features, $\mathbf{x} = [x_1, x_2, x_3]$, i.e., we will estimate $p(x_1, x_2, x_3 \mid y) \times p(y)$.

  - Naive bayes classifiers assume conditional independence between features given a class.



$$p(x_1, x_2, x_3 \mid y) = p(x_1 \mid y) \times p(x_2 \mid y) \times p(x_3 \mid y)$$

Sentiment Analysis using NBC

- Sentiment Analysis (SA)
  - The model predict a rating for the given movie review
- SA using NBC
  - y: rating, x: review, a sequence of words, $\{w_i\}_{i=1}^{L}$, with length L
  - Conditional independence assumption
    - We assume that words are conditionally independent with each other, where the condition is the given rating, such that:

$$p(w_1, \ldots, w_L \mid y) = \prod_{i=1}^{L} p(w_i \mid L)$$

$$p(y \mid \{w_i\}_{i=1}^{L}) \propto p(y) \times \prod_{i=1}^{L} p(w_i \mid y)$$

$$\log p(y \mid \{w_i\}_{i=1}^{L}) \propto \log p(y) + \sum_{i=1}^{L} \log p(w_i \mid y)$$

Sentiment Analysis using NBC

- For example,
  - Conditional Independence Assumption

$$p(\mathbf{x} = \text{"I love it"} \,|\, y = +) = p(\text{"I"} \,|\, y = +) \times p(\text{"love"} \,|\, y = +) \times p(\text{"it"} \,|\, y = +)$$

  - Posterior Estimation

$$\log p(y = + \,|\, \mathbf{x} = \text{"I love it"}) \propto \underbrace{\log p(y = +)}_{\text{prior}} + \underbrace{\log p(\text{"I"} \,|\, y = +) + \log p(\text{"love"} \,|\, y = +) + \log p(\text{"it"} \,|\, y = +)}_{\text{likelihood}}$$

Procedure: 1. preprocessing

- Converting ratings into binary classes
  - 0 (negative) for $y = \{1,2,3,4,5\}$
  - 1 (positive) for $y = \{6,7,8,9,10\}$
- Constructing vocabulary
  - Converting words into lowercase
- Converting words into word index
  -

Words	good	interesting	movie	story
Index	0	1	2	3

  - ,e.g., review=[“interesting”, “movie”] → [1, 2]

Procedure: 2. estimating prior

- Notation
  - $y \in \{\text{positive, negative}\}$, or equivalently $y \in \{1,0\}$
- Procedure of estimating prior, p(y), as follows:
  - 1. Count the number of documents for each class
  - 2. Normalize counts

Sentiment	negative (0)	positive (1)
Count	N= 14938	P= 52488
p(y)	N/(P+N)=0.22	P/(P+N)=0.78

P+N= 67426

Procedure: 3. estimating likelihood

- Recall: conditional independence assumption
  - We want to estimate likelihood, $p(\mathbf{x} \mid y) = p((w_1, \ldots, w_L) \mid y)$.
    - $\mathbf{x}$ stands for sequence of words (w_i) with length L
    - y stands for a sentiment label
  - Naive bayes classifiers assume w_i is conditionally independent with $w_j, j \neq i$ given y, i.e.,

$$p(\mathbf{x} \mid y) = p((w_1, \ldots, w_L) \mid y) = \prod_{i=1}^{L} p(w_i \mid y)$$

  - Therefore, we only need to estimate likelihood for each word, $p(w_i \mid y)$.

Procedure: 3. estimating likelihood

- Count words and normalize the counts

$$p(w \mid y) = \frac{count(w, y)}{\sum_{w' \in V} count(w', y)}$$

- ex) Suppose we have training data,

$$y_1 = 1 \text{ (positive)}, \mathbf{x}_1\text{=“wonderful story”}$$
$$y_2 = 1 \text{ (positive)}, \mathbf{x}_2\text{=“wonderful movie”}$$
$$y_3 = 0 \text{ (negative)}, \mathbf{x}_3\text{=“unfunny movie”}$$

- The number of occurrence (counts) of each word

	wonderful	unfunny	movie	story	Total
negative (0)	0	1	1	1	3
positive (1)	2	0	1	0	3

- Likelihood for each word

	wonderful	unfunny	movie	story
negative (0)	0	1/3	1/3	1/3
positive (1)	2/3	0	1/3	0

Procedure: 3. estimating likelihood

- Challenge: unknown words

$$p(\text{"I recommenddd you this wonderful movie"} \mid y = +)$$
$$= \cdots \times p(\text{"recommenddd"} \mid y = +) \cdots \times p(\text{"wonderful"} \mid y = +) \cdots$$

  - The likelihood is expected to be high because of the word, "wonderful".
  - However, since "recommenddd" (typo) is never observed in train dataset, we have "0" likelihood, and thus "0" posterior probability.

- Solution: Laplace smoothing
  - We add smoothing constant k for counts of all words

$$\hat{p}(w_i \mid y) = \frac{count(w_i, y) + k}{\sum_{w \in V} [count(w, y) + k]}$$
$$= \frac{count(w_i, y) + k}{k \times |V| + \sum_{w \in V} count(w, y)}$$

, where V is vocabulary

Procedure: 4. evaluation

- Estimating posterior using conditional independence assumption

Estimating posterior

- Using training dataset, we estimated
  - prior distribution, $p(y)$,
  - and likelihood of each word, $p(w \mid y)$.
- Given any reviews, we can estimate posterior distribution as follows:

$$p(y \mid \{w_i\}_{i=1}^{L}) = \frac{p(y) \times p(\{w_i\}_{i=1}^{L} \mid y)}{p(\{w_i\}_{i=1}^{L})} \qquad (\because \text{bayes rule})$$

$$\propto p(y) \times p(\{w_i\}_{i=1}^{L} \mid y)$$

$$= p(y) \times \prod_{i=1}^{L} p(w_i \mid y) \qquad (\because \text{assumption})$$

$$\log p(y \mid \{w_i\}_{i=1}^{L}) \propto \log p(y) + \sum_{i=1}^{L} \log p(w_i \mid y)$$

$$y^* = \arg\max_{y} \log p(y \mid \{w_i\}_{i=1}^{L})$$

Hands-on practice

- Github code
  - https://github.com/zizi1532/NaiveBayesClassifier/blob/master/imdb_jupyter.ipynb

Discussion

- Advantages of NBC
  - It is easy to implement
  - It is fast to train and evaluate the model
  - It can be used as a simple baseline
- Disadvantage of NBC
  - Conditional independence assumption is too strong to correctly estimate complex distribution
    - ex)

$$
\begin{aligned}
& p(\text{"not good"} \mid y = -) \\
& = p(\text{"not"} \mid y = -) \times p(\text{"good"} \mid \text{"not"}, y = -) \\
& \neq p(\text{"not"} \mid y = -) \times p(\text{"good"} \mid y = -)
\end{aligned}
$$

  - Limitation of lexical representation
    - We cannot consider semantic similarity between words.
    - We have long-tail distributions over words, i.e., Zipfs' law.

Thanks