# Naive Bayes for Sentiment Analysis

Yannet Interian

Jan 10th 2018

# Agenda

- Review of Naive Bayes and Sentiment Analysis
- Coding Naive Bayes in Spark

# Positive or negative movie review?

- unbelievably disappointing

- full of zany characters and richly applied satire, and some great plot twists

- this is the greatest screwball comedy ever filmed

- it was pathetic. The worst part about it was the boxing scenes.

Important commercial application

# Naive Bayes Intuition

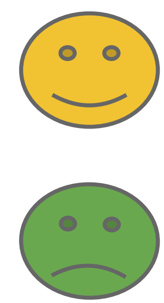
- Simple (“naive”) classification method based on Bayes rule
- Relies on very simple representation of document
  - Bag of words / unigram language model

# The bag of words representation

F(

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun… It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet.

)=c



# The bag of words representation

F(

I **love** this movie! It's **sweet**, but with **satirical** humor. The dialogue is **great** and the adventure scenes are **fun**… It manages to be **whimsical** and **romantic** while **laughing** at the conventions of the fairy tale genre. I would **recommend** it to just about anyone. I've seen it **several** times, and I'm always **happy** to see it **again** whenever I have a friend who hasn't seen it yet.

)=c

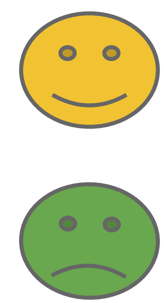
# The bag of words representation:
## using a subset of words

# The bag of words representation

F(

| | |
|---|---|
| great | 2 |
| love | 2 |
| recommend | 1 |
| laugh | 1 |
| happy | 1 |
| ... | ... |

)=c



# Posterior class probability

For a document *d* and a class *c*

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

**P(c | d)** depends on the **training data** and the choice of **modeling technique**

# Bayes’ Rule Applied to Documents and Classes

For a document *d* and a class *c*

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

# Naive Bayes Classifier (I)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}}\, P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}}\, P(d \mid c)P(c)$$

Dropping the denominator

likelihood

prior

# Naïve Bayes Classifier (II)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

Document d represented as features x1..xn

# Multinomial Naïve Bayes Independence Assumptions

- Bag of Words assumption: Assume position doesn't matter
- Example for $x_i$ = { occurrence of word *like* }
- Conditional Independence: Assume the features $x_i|c$ are independent for every class $c$.

$$P(x_1,\ldots,x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

# Multinomial Naïve Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \prod_{x \in X} P(x \mid c)$$

P(x|c) "how much evidence *x* contributes that *c* the correct class"

# Naïve Bayes Learning

# Learning the Multinomial Naïve Bayes Model

First attempt: simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

What is the problem with this attempt?

# Laplace (add-1) smoothing for Multinomial Naïve Bayes

- What if we have seen no training documents with the word "fantastic" and classified in the topic positive?

$$\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{count(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} count(w, \text{positive})} = 0$$

- Add-1 Smoothing

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

# Multinomial Naïve Bayes Learning: summary

For every class

- compute priors

$$N = \text{Number of documents}$$
$$N_c = \text{Number of documents in class } c$$
$$\widehat{P}(c) = \frac{N_c}{N}$$

For every word w and every class c

$$V = \text{Set of unique words}$$
$$count(w, c) = \text{Frequency of } w \text{ in } c$$
$$count(c) = \text{Number of words in } c$$
$$\widehat{P}(w|c) = \frac{count(w, c) + 1}{count(c) + |V|}$$

Find c such that: $argmax \widehat{P}(c)\widehat{P}(d|c)$

What is the time complexity of this algorithm?

# Naïve Bayes: unknown words

- If your training set is expected to have unknown words: add a new word $w_u$ to the vocabulary.

$$\hat{P}(w_u \mid c) = \frac{count(w_u, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V + 1|}$$

$$= \frac{1}{\left(\sum_{w \in V} count(w, c)\right) + |V + 1|}$$

# Summary

`count(w, pos)`

`count(w, neg)`

`count(pos)` and `count(neg)`

`V` number of unique words in the training set

`P(w| c) = (count(w, c) + 1) /( count(c) + V + 1)`

Given that how do we compute the class of a document?

`P(c|d)`

# Summary

`count(w, pos)`

`count(w, neg)`

`count(pos)` and `count(neg)`

`V` number of unique words in the training set

`P(w| c) = [count(w, c) + 1] /( count(c) + V + 1)`

Given that how do we compute the class of a document?

**log** `P(c|d) ~ sum_i` **log** `P(w_i|c) P(c)`

# Example

| | docID | words in document | in $c = China$? |
|---|---|---|---|
| training set | 1 | Chinese Beijing Chinese | yes |
| | 2 | Chinese Chinese Shanghai | yes |
| | 3 | Chinese Macao | yes |
| | 4 | Tokyo Japan Chinese | no |
| test set | 5 | Chinese Chinese Chinese Tokyo Japan | ? |

# Acknowledgment

**Some of these slides are adapted from the NLP class from coursera.org taught by the Stanford professors:** Dan Jurafsky and Chris Manning.
https://class.coursera.org/nlp/