Text Mining and Language Analytics

Lecture 7

Naïve Bayes and Sentiment Classification

Dr Stamos Katsigiannis

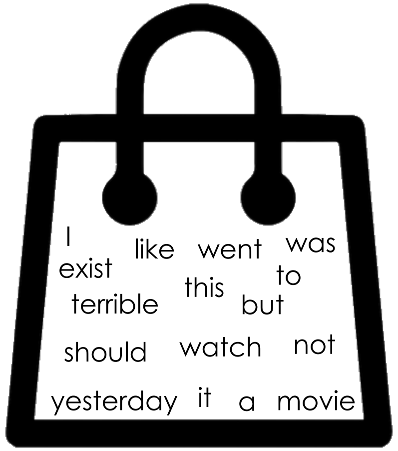
2023-24

Durham University

Naïve Bayes (NB) classification

- Simple ("naïve") classification method based on Bayes rule
- Probabilistic classification approach
- Based on the probabilities of a document belonging to a class
- Relies on the Bag of Words representation of a document

I went to watch a movie yesterday but it was terrible. A terrible movie like this should not exist.



Naïve Bayes classifier

- For a document d and a class c out of the possible classes C

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

- Based on the Bayes Rule, a document should be assigned to the class:

$$\hat{c} = \arg\max_{c \in C} P(c|d) = \arg\max_{c \in C} \frac{P(d|c)P(c)}{P(d)} = \arg\max_{c \in C} (P(d|c)P(c))$$

- $P(c|d)$ is computed for all $c \in C$ and the denominator $P(d)$ is dropped because it does not change across classes

Durham University

Multinomial Naïve Bayes classifier (I)

- $P(c) \rightarrow$ Prior probability of class c

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

- $P(d|c) = P(w_1, w_2, \dots, w_n|c) \rightarrow$ Likelihood
  - **Bag of Words assumption:** Assume that word position does not matter. w_i will represent word identity and not position
  - **Conditional Independence assumption:** Assume that word probabilities $P(w_i|c)$ are independent given the class c
  - **Both assumptions are simplistic and incorrect!** Hence the "Naïve" …

$$P(d|c) = P\left(w_1, w_2, \dots, w_{|V|}\middle|c\right) = P(w_1|c) \cdot P(w_2|c) \cdot \ \dots \cdot P(w_{|V|}|c)$$

Durham University

Multinomial Naïve Bayes classifier (II)

- Consequently

$$c_{NB} = \arg\max_{c \in C} \left(P\left(w_1, w_2, \ldots, w_{|V|} \middle| c\right) P(c)\right) = \arg\max_{c \in C} \left(P(c) \prod_{w_i \in V} P(w_i|c)\right)$$

- Typically computed in log space to increase speed and avoid underflow:

$$c_{NB} = \arg\max_{c \in C} \left(\log P(c) + \sum_{i=1}^{|V|} \log P(w_i|c)\right)$$

Durham University

Multinomial Naïve Bayes classifier (III)

- **How to compute $P(w_i|c)$?**

- Use Maximum Likelihood Estimation

$$P(w_i|c) = \frac{count(w_i, c)}{\sum_{w \in V} count(w, c)} = \frac{\text{Count of word } w_i \text{ in documents of class } c}{\text{Count of words in documents of class } c}$$

- What if a word does not appear in documents of class c?
  - Then $P(w_i|c) = 0$
  - Use Laplace Smoothing (or Add-λ smoothing)

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

Durham University

NB example: Movie review sentiment (I)

Document	Class	Set
just plain boring	-	Training
entirely predictable and lacks energy	-	
no surprises and very few laughs	-	
very powerful	+	
the most fun film of the summer	+	
predictable with no fun	**?**	**Test**

$$P(-) = \frac{N_-}{N_{doc}} = \frac{3}{5}$$

$$P(+) = \frac{N_+}{N_{doc}} = \frac{2}{5}$$

$$|V| = 20$$

$$count(\text{words}, -) = 14$$

$$count(\text{words}, +) = 9$$

Word	Count All	Count -	Count +
and	2	2	0
boring	1	1	0
energy	1	1	0
entirely	1	1	0
few	1	1	0
film	1	0	1
fun	1	0	1
just	1	1	0
lacks	1	1	0
laughs	1	1	0
most	1	0	1
no	1	1	0
of	1	0	1
plain	1	1	0
powerful	1	0	1
predictable	1	1	0
summer	1	0	1
surprises	1	1	0
the	2	0	2
very	2	1	1

"with" does not exist in training set → **Ignore!**

Durham University

NB example: Movie review sentiment (II)

Some probabilities are 0 → **Laplace smoothing**

$$P(\text{"predictable"} \mid -) = \frac{count(\text{"predictable"}, -) + 1}{count(\text{words}, -) + |V|} = \frac{1+1}{14+20} = \frac{2}{34}$$

$$P(\text{"no"} \mid -) = \frac{count(\text{"no"}, -) + 1}{count(\text{words}, -) + |V|} = \frac{1+1}{14+20} = \frac{2}{34}$$

$$P(\text{"fun"} \mid -) = \frac{count(\text{"fun"}, -) + 1}{count(\text{words}, -) + |V|} = \frac{0+1}{14+20} = \frac{1}{34}$$

$$P(\text{"predictable"} \mid +) = \frac{count(\text{"predictable"}, +) + 1}{count(\text{words}, +) + |V|} = \frac{0+1}{9+20} = \frac{1}{29}$$

$$P(\text{"no"} \mid +) = \frac{count(\text{"no"}, +) + 1}{count(\text{words}, +) + |V|} = \frac{0+1}{9+20} = \frac{1}{29}$$

$$P(\text{"fun"} \mid +) = \frac{count(\text{"fun"}, +) + 1}{count(\text{words}, +) + |V|} = \frac{1+1}{9+20} = \frac{2}{29}$$

Word	Count All	Count -	Count +
and	2	2	0
boring	1	1	0
energy	1	1	0
entirely	1	1	0
few	1	1	0
film	1	0	1
fun	1	0	1
just	1	1	0
lacks	1	1	0
laughs	1	1	0
most	1	0	1
no	1	1	0
of	1	0	1
plain	1	1	0
powerful	1	0	1
predictable	1	1	0
summer	1	0	1
surprises	1	1	0
the	2	0	2
very	2	1	1

Durham University

Example: Movie review sentiment (III)

Document	Class	Set
just plain boring	-	Training
entirely predictable and lacks energy	-	
no surprises and very few laughs	-	
very powerful	+	
the most fun film of the summer	+	
predictable with no fun	**?**	**Test**

$$P(\text{"predictable"}|-) = \frac{2}{34}$$

$$P(\text{"no"}|-) = \frac{2}{34}$$

$$P(\text{"fun"}|-) = \frac{1}{34}$$

$$P(\text{"predictable"}|+) = \frac{1}{29}$$

$$P(\text{"no"}|+) = \frac{1}{29}$$

$$P(\text{"fun"}|+) = \frac{2}{29}$$

$$P(-) = \frac{3}{5}$$

$$P(+) = \frac{2}{5}$$

$$P(-)P(S|-) = P(-)P(\text{"predictable"}|-)P(\text{"no"}|-)P(\text{"fun"}|-) = \frac{3}{5} \cdot \frac{2}{34} \cdot \frac{2}{34} \cdot \frac{1}{34} \approx 6.1 \cdot 10^{-5}$$

$$P(+)P(S|+) = P(+)P(\text{"predictable"}|+)P(\text{"no"}|+)P(\text{"fun"}|+) = \frac{2}{5} \cdot \frac{1}{29} \cdot \frac{1}{29} \cdot \frac{2}{29} \approx 3.2 \cdot 10^{-5}$$

$$P(-)P(S|-) > P(+)P(S|+)$$

Review classified as negative

Durham University

NB optimisations: Binary NB (I)

- **Occurrence of word matters more than frequency** for sentiment classification and some other text classification tasks

- Improve performance by clipping word counts per document to 1

- **Binary multinomial Naïve Bayes (Binary NB)**

- Apply similar to regular multinomial Naïve Bayes (NB)

Durham University

NB optimisations: Binary NB (II)

Four original documents:

– it was pathetic the worst part was the boxing scenes
– no plot twists or great scenes
+ and satire and great plot twists
+ great scenes great film

After per-document binarization:

– it was pathetic the worst part boxing scenes
– no plot twists or great scenes
+ and satire great plot twists
+ great scenes film

	NB Counts +	NB Counts −	Binary Counts +	Binary Counts −
and	2	0	1	0
boxing	0	1	0	1
film	1	0	1	0
great	3	1	2	1
it	0	1	0	1
no	0	1	0	1
or	0	1	0	1
part	0	1	0	1
pathetic	0	1	0	1
plot	1	1	1	1
satire	1	0	1	0
scenes	1	2	1	2
the	0	2	0	1
twists	1	1	1	1
was	0	2	0	1
worst	0	1	0	1

Daniel Jurafsky, James H. Martin, "Speech and Language Processing", 3rd edition

$|V| = 20$

NB counts:

$count(\text{words}, -) = 16$

$count(\text{words}, +) = 10$

e.g. $P(\text{"great"}|+) = \frac{3}{10}$

Binary NB counts:

$count(\text{words}', -) = 14$

$count(\text{words}', +) = 8$

e.g. $P'(\text{"great"}|+) = \frac{2}{8}$

NB optimisations: Negation (I)

- Negation plays an important role in text classification

- Consider the following sentences
  - **Example 1:**
    - I really like this film
    - I didn't like this film
  - **Example 2:**
    - Dismiss this film
    - Don't dismiss this film

- Negation expressed by "didn't" and "Don't" completely alters the inferences drawn by "like" and "dismiss"
  - **Example 1:** Positive → Negative
  - **Example 2:** Negative → Positive

Durham University

NB optimisations: Negation (II)

- Prepend prefix **NOT_** to every word after a token of logical negation (-n't, not, no, never,…) until the next punctuation mark

- **Simple baseline for text classification for sentiment analysis**

- Example:

  … didn't like this film, but I … → … didn't **NOT_like NOT_this NOT_film**, but I …

- Words of positive sentiment with the prefix NOT_ will occur more often in negative documents and act as cues for negative sentiment
  - e.g. NOT_like, NOT_recommend

- Words of negative sentiment with the prefix NOT_ will acquire positive associations
  - e.g. NOT_bored, NOT_dismiss

Durham University

NB optimisations: Unknown words (I)

- **What about words in the test set that don't exist in the training set?**

- Option 1 → Ignore them
  - What about test documents with many unknown words?

- Option 2 → Add an extra word w_u to the vocabulary
  - w_u: the "unknown" word
  - Count unknown words as occurrences of w_u
  - $V' = \{V, w_u\}, |V'| = |V| + 1$
  - $P(w_u|c) = \frac{count(w_u,c)+1}{\sum_{w \in V'}(count(w,c)+1)} = \frac{0+1}{(\sum_{w \in V'} count(w,c))+|V'|} = \frac{1}{(\sum_{w \in V} count(w,c))+|V|+1}$

$count(w_u, c) = 0$
w_u does not occur in any class

$\sum_{w \in V'} count(w,c) = \sum_{w \in V} count(w,c)$
w_u does not occur in any class

Durham University

NB optimisations: Unknown words (II)

Document	Class	Set
just plain boring	-	Training
entirely predictable and lacks energy	-	
no surprises and very few laughs	-	
very powerful	+	
the most fun film of the summer	+	
predictable with no fun	**?**	**Test**

$$|V| = 20$$

$$\sum_{w \in V} count(w, -) = 14$$

$$\sum_{w \in V} count(w, +) = 9$$

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

$$P(\text{"with"}, -) = P(w_u, -) = \frac{1}{14 + 20 + 1} = \frac{1}{35} \approx 0.0285$$

$$P(\text{"with"}, +) = P(w_u, +) = \frac{1}{9 + 20 + 1} = \frac{1}{30} \approx 0.0333$$

Durham University

NB optimisations: Lexicons (I)

- **What if we have insufficient labelled training data for sentiment analysis?**

- **We can use sentiment lexicons!**
  - **General Inquirer**(Stone et al., 1966)
  - **LIWC**(Pennebaker et al., 2007)
  - **Hu and Liu lexicon**(Hu and Liu, 2004)
  - **MPQA subjectivity lexicon**(Wilson et al., 2005)
  - …

- Sentiment lexicon → List of words annotated with positive or negative sentiment

- Example from MPQA:
  - **+** : admirable, beautiful, confident, dazzling, ecstatic, favour, glee, great
  - **-** : awful, bad, bias, catastrophe, cheat, deny, envious, foul, harsh, hate

- Stone, P., Dunphry, D., Smith, M., and Ogilvie, D. (1966). The General Inquirer: A Computer Approach to Content Analysis. Cambridge, MA: MIT Press.
- Pennebaker, J. W., Booth, R. J., and Francis, M. E. (2007). Linguistic Inquiry and Word Count: LIWC 2007. Austin, TX.
- Hu, M. and Liu, B. (2004a). Mining and summarizing customer reviews. In KDD, 168–177.
- Wilson, T.,Wiebe, J., and Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. In HLT-EMNLP-05, 347–354.

NB optimisations: Lexicons (II)

- **How can we use the sentiment lexicons for classification with Naïve Bayes?**

- Consider a lexicon with C sentiment classes and n_c words in class c
  - Total words in lexicon $\rightarrow N = \sum_{c \in C} n_c$
  - Vocabulary size $|V| = N$
  - **Prior probabilities of each class** $\rightarrow P(c) = \frac{n_c}{N}$
  - Number of words in class $c \rightarrow \sum_{w \in V} count(w, c) = n_c$
  - $count(w_i, c) = \begin{cases} 1, & \text{if } w_i \text{ in } c \\ 0, & \text{if } w_i \text{ not in } c \end{cases}$
  - $P(w_i|c) = \frac{count(w_i,c)+1}{(\sum_{w \in V} count(w,c))+|V|} = \frac{count(w_i,c)+1}{n_c+N}$

Assuming that a word can belong only to one class and there are no duplicate words

Durham University

NB optimisations: Lexicons (III)

Example

- Consider the following sentiment lexicon and the sentence

I had a great, almost amazing, night but the movie was terrible.

Lexicon

Words list	Class
great, amazing, excellent, impressive	+
worst, disaster, bad, terrible, boring	-

- **Let's ignore the unknown words:**
  - I, had, a, almost, night, but, the, movie, was
- $P(S|c) = P(c)P(\text{"great"}|c)P(\text{"amazing"}|c)P(\text{"terrible"}|c)$

$$P(-) = \frac{5}{9}$$

$$P(+) = \frac{4}{9}$$

$$|V| = 9$$

$$\sum_{w \in V} count(w, -) = 5$$

$$\sum_{w \in V} count(w, +) = 4$$

$$P(\text{great}|-) = \frac{count(\text{"great"}, -) + 1}{(\sum_{w \in V} count\,(w, -)) + |V|} = \frac{0+1}{5+9} = \frac{1}{14}$$

$$P(\text{great}|+) = \frac{count(\text{"great"}, +) + 1}{(\sum_{w \in V} count\,(w, +)) + |V|} = \frac{1+1}{4+9} = \frac{2}{13}$$

$$P(\text{"amazing"}|-) = \frac{0+1}{5+9} = \frac{1}{14} \qquad P(\text{"amazing"}|+) = \frac{1+1}{4+9} = \frac{2}{13}$$

$$P(\text{"terrible"}|-) = \frac{1+1}{5+9} = \frac{2}{14} \qquad P(\text{"terrible"}|+) = \frac{0+1}{4+9} = \frac{1}{13}$$

$$P(-) \cdot P(S|-) = \frac{5}{9} \cdot \frac{1}{14} \cdot \frac{1}{14} \cdot \frac{2}{14} \approx 4 \cdot 10^{-4}$$

$$P(+) \cdot P(S|+) = \frac{4}{9} \cdot \frac{2}{13} \cdot \frac{2}{13} \cdot \frac{1}{13} \approx 8 \cdot 10^{-4}$$

Durham University

Measuring classification performance

- **How to evaluate the performance of a machine learning model?**
- **Use the model on a labelled test set!**
- The labels of the test set are referred as *gold labels* or *ground truth*
- Compare the model's output with the ground truth
- Always keep the training and test data separate!

Example:

Sample	Actual	Prediction	
1	+	-	Wrong
2	+	+	Correct
3	-	-	Correct
4	-	-	Correct

Durham University

The confusion matrix (I)

2-class (binary) problem

		Actual class	
		P	**N**
Predicted class	**P**	TP	FP
	N	FN	TN

TP: True Positive
TN: True Negative
FP: False Positive
FN: False Negative

Multi-class problem

		Actual class				
		Class 1	**Class 2**	**Class 3**	**…**	**Class N**
Predicted class	**Class 1**	C(1,1)	C(1,2)	C(1,3)		C(1,N)
	Class 2	C(2,1)	C(2,2)	C(2,3)	…	C(2,N)
	Class 3	C(3,1)	C(3,2)	C(3,3)		C(3,N)
	…	…			C(i,i)	…
	Class N	C(N,1)	C(N,2)	C(N,3)	…	C(N,N)

C(x,y): Count of samples of class **y** that were predicted as class **x**

Durham University

The confusion matrix (II)

Example:

Sample	Actual	Prediction
1	+	-
2	+	+
3	-	-
4	-	-

		Actual class	
		+	-
Predicted class	+	1	0
	-	1	2

Durham University

Classification performance metrics (I)

Accuracy

- Proportion of correct predictions

Precision

- Proportion of correct positive identifications

Recall / Sensitivity / True positive Rate

- Proportion of actual positives that were identified correctly

Specificity / True Negative Rate

- Proportion of actual negatives that were identified correctly

F1-score

- Harmonic mean of precision and recall: $F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$

Classification performance metrics (II)

		Actual class	
		P	**N**
Predicted class	**P**	TP	FP
	N	FN	TN

TP: True Positive
TN: True Negative
FP: False Positive
FN: False Negative

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Sensitivity(Recall) = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Durham University

Classification performance metrics (III)

Example:

		Actual class	
		+	-
Predicted class	+	1	0
	-	1	2

Actual positives = TP + FN
Actual negatives = TN + FP

$$Accuracy = \frac{1+2}{1+2+0+1} = 75\%$$

$$\text{Precision} = \frac{1}{1+0} = 100\%$$

$$Recall = \frac{1}{1+1} = 50\%$$

$$Specificity = \frac{2}{2+0} = 100\%$$

$$F1 = 2 \cdot \frac{1 \cdot 0.5}{1+0.5} = \frac{1}{1.5} \approx 66.66\%$$

Durham University

Classification performance metrics (IV)

- The former definitions apply to binary classification!
- **What about multi-class classification?**

$$Accuracy = \frac{Sum\ of\ diagonal\ of\ confusion\ matrix}{Sum\ of\ all\ elements\ of\ confusion\ matrix}$$

- **For the other metrics:**
  - Create multiple confusion matrices, each considering one class as positive and all the others as negative
  - Compute metrics for each class
  - Compute the mean metrics across classes

Durham University

Questions?

Durham University