What is Text Classification

Text classification is automatically classifying a document into two or more *predefined* categories.

Examples of documents and their possible categories: Emails: spam or not spam Newswire articles: business, politics or sports Movie reviews: liked it or hated it

Supervised learning

What knowledge do we need to complete the task?

Classifying newswires into politics or sports: Read the newswires, find the characteristics which determine whether they are politics or sports, then make a judgement.

But what if you've never read a politics article before? How would you know what to look for?

Supervised learning

What knowledge do we need to complete the task?

Classifying newswires into politics or sports: Read the newswires, find the characteristics which determine whether they are politics or sports, then make a judgement.

But what if you've never read a politics article before? How would you know what to look for?

In *supervised learning*, the machine learns based on examples and their **already known** classifications (*training set*)

Based on the knowledge learnt from the examples, we can classify **unknown** examples (*test set*)

Today's problem

Lady Gaga vs. The Clash

Input to this problem: a bunch of text files containing song lyrics.

Preprocessing the input

Decompose (tokenise) each input file into a ‘bag of words’.

Bag of words: a list of words with corresponding frequency of that word occurring in the document (a weighted vector of words)

	$word_1$	$word_2$	$word_3$	$word_4$	$word_5$
doc_1					
doc_2					
doc_3					

Train the classifier

Multinomial Naive Bayes: a Generative learning algorithm

Generative learning algorithms create a model based on the training data, and apply that model on the test data.

Two categories: *gaga*, *clash*

We will build a model for each category.

Classification task: find the *most likely* category that the document belongs to.

Demo

How does it work?

Bayes' Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Where:

A and B are events

$P(B) \neq 0$ and is the probability of the data

$P(A|B)$: probability that A occurs given B

$P(B|A)$: probability that B occurs given A

Bayes' Theorem in Text Classification

Objective: find the likelihood that document d belongs in class c

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

Try each of the classes in turn i.e. $C = \{gaga, clash\}$

The most likely class is the one which returns the highest value of $P(c|d)$ which is defined as C_{MAP} (maximum a posteriori)

$$C_{MAP} = \underset{c \in C}{\arg\max}\ P(c|d)$$

Bayes' Theorem in Text Classification

Objective: find the likelihood that document d belongs in class c

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

Try each of the classes in turn i.e. $C = \{gaga, clash\}$

The most likely class is the one which returns the highest value of $P(c|d)$ which is defined as C_{MAP} (maximum a posteriori)

$$C_{MAP} = \underset{c \in C}{\arg\max}\ P(c|d)$$

(Apply Bayes' theorem again)

$$C_{MAP} = \underset{c \in C}{\arg\max}\ \frac{P(d|c)P(c)}{P(d)}$$

$$C_{MAP} = \underset{c \in C}{\arg\max}\ P(d|c)P(c)$$

Training

Recall that our document (d) is just a 'bag of words'. We can represent the words as the product of all of the word probabilties:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

Prior probability of the class:

$P(C) = \frac{N_c}{N}$

Where N_c is the number of docs in the class, and N is the total number of docs.

Relative frequency of term t occurring in a class c:

$$P(t|c) = \frac{T_{ct}}{\sum_{t \in V} T_{ct}}$$

where T_{ct} is the number of occurrences of t in training documents from class c. (V is the set of all terms)

The problem of zero

Since we need to iterate through all terms in our vocabulary for all classes, if a document in a class doesn't include a term existing in the other classes, $T_{ct} = 0$, which makes $P(c|d) = 0$ when included in the product of all terms.

For example, *gaga* documents include the word 'mascara' which appears in no *clash* documents, so $P(mascara|clash) = 0$.

We fix this by applying *add-one smoothing*:

$$P(t|c) = \frac{T_{ct}}{\sum_{t \in V} T_{ct}}$$

becomes

$$P(t|c) = \frac{T_{ct} + 1}{\sum_{t \in V}(T_{ct} + 1)} = \frac{T_{ct} + 1}{(\sum_{t \in V} T_{ct}) + |V|)}$$

where $|V|$ is the total number of terms in the vocabulary.

Example: training

Training set:

	N_c	animal	game	love	london	T_{ct}
gaga	2	66	1	21	0	*88*
clash	2	0	4	0	14	*18*

Class prior: $P(c) = \frac{N_c}{N} = \frac{2}{4}$ for both classes so we can dismiss it.

Term likelihood:

	animal	game	love	london
gaga	$P(animal\|gaga)$			
clash				

$P(t|c) = \frac{T_{ct}+1}{(\sum_{t \in V} T_{ct})+|V|)}$

$P(animal|gaga) = \frac{66+1}{88+106} = 0.345361$

Example: training

Training set:

	N_c	animal	game	love	london	T_{ct}
gaga	2	66	1	21	0	*88*
clash	2	0	4	0	14	*18*

Class prior: $P(c) = \frac{N_c}{N} = \frac{2}{4}$ for both classes so we can dismiss it.

Term likelihood:

	animal	game	love	london
gaga	0.345361			
clash	$P(animal\|clash)$			

$P(t|c) = \frac{T_{ct}+1}{(\sum_{t \in V} T_{ct})+|V|)}$

$P(animal|gaga) = \frac{66+1}{88+106} = 0.345361$

$P(animal|clash) = \frac{0+1}{18+106} = 0.008065$

Example: training

Training set:

	N_c	animal	game	love	london	T_{ct}
gaga	2	66	1	21	0	*88*
clash	2	0	4	0	14	*18*

Class prior: $P(c) = \frac{N_c}{N} = \frac{2}{4}$ for both classes so we can dismiss it.

Term likelihood:

	animal	game	love	london
gaga	0.345361			
clash	0.008065			

$P(t|c) = \frac{T_{ct}+1}{(\sum_{t \in V} T_{ct})+|V|)}$

$P(animal|gaga) = \frac{66+1}{88+106} = 0.345361$

$P(animal|clash) = \frac{0+1}{18+106} = 0.008065$

Example: training

Training set:

	N_c	animal	game	love	london	T_{ct}
gaga	2	66	1	21	0	*88*
clash	2	0	4	0	14	*18*

Class prior: $P(c) = \frac{N_c}{N} = \frac{2}{4}$ for both classes so we can dismiss it.

Term likelihood:

	animal	game	love	london
gaga	0.345361	0.010309	0.113402	0.005155
clash	0.008065	0.040323	0.008605	0.120968

$P(t|c) = \frac{T_{ct}+1}{(\sum_{t \in V} T_{ct})+|V|)}$

$P(animal|gaga) = \frac{66+1}{88+106} = 0.345361$

$P(animal|clash) = \frac{0+1}{18+106} = 0.008065$

Example: classifying

Our trained classifier:

	animal	game	love	london
gaga	0.345361	0.010309	0.113402	0.005155
clash	0.008065	0.040323	0.008605	0.120968

Test document:

	animal	game	love	london
???		1	2	1

Example: classifying

Our trained classifier:

	animal	game	love	london
gaga	0.345361	0.010309	0.113402	0.005155
clash	0.008065	0.040323	0.008605	0.120968

Test document:

	animal	game	love	london
???		1	2	1

Multiply the test document by the training set:

	animal	game	love	london
gaga		0.010309	0.226804	0.005155
clash		0.040323	0.016129	0.120968

Example: classifying

Our trained classifier:

	animal	game	love	london
gaga	0.345361	0.010309	0.113402	0.005155
clash	0.008065	0.040323	0.008605	0.120968

Test document:

	animal	game	love	london
???		1	2	1

Multiply the test document by the training set:

	animal	game	love	london
gaga		0.010309	0.226804	0.005155
clash		0.040323	0.016129	0.120968

This gives:

$$P(gaga) = 0.010309 * 0.226804 * 0.005155 = 6.02625e^{-6}$$

$$P(clash) = 0.040323 * 0.016129 * 0.120968 = 3.93365e^{-5}$$

Example: classifying

Our trained classifier:

	animal	game	love	london
gaga	0.345361	0.010309	0.113402	0.005155
clash	0.008065	0.040323	0.008605	0.120968

Test document:

	animal	game	love	london
???		1	2	1

Multiply the test document by the training set:

	animal	game	love	london
gaga		0.010309	0.226804	0.005155
clash		0.040323	0.016129	0.120968

This gives:

$$P(gaga) = 0.010309 * 0.226804 * 0.005155 = 6.02625e^{-6}$$

$$P(clash) = 0.040323 * 0.016129 * 0.120968 = 3.93365e^{-5} \boxed{\text{Wins}}$$

References

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schtze. (2008) *Introduction to Information Retrieval*, Cambridge University Press. https://nlp.stanford.edu/IR-book/