COMP30810
Intro to Text Analytics

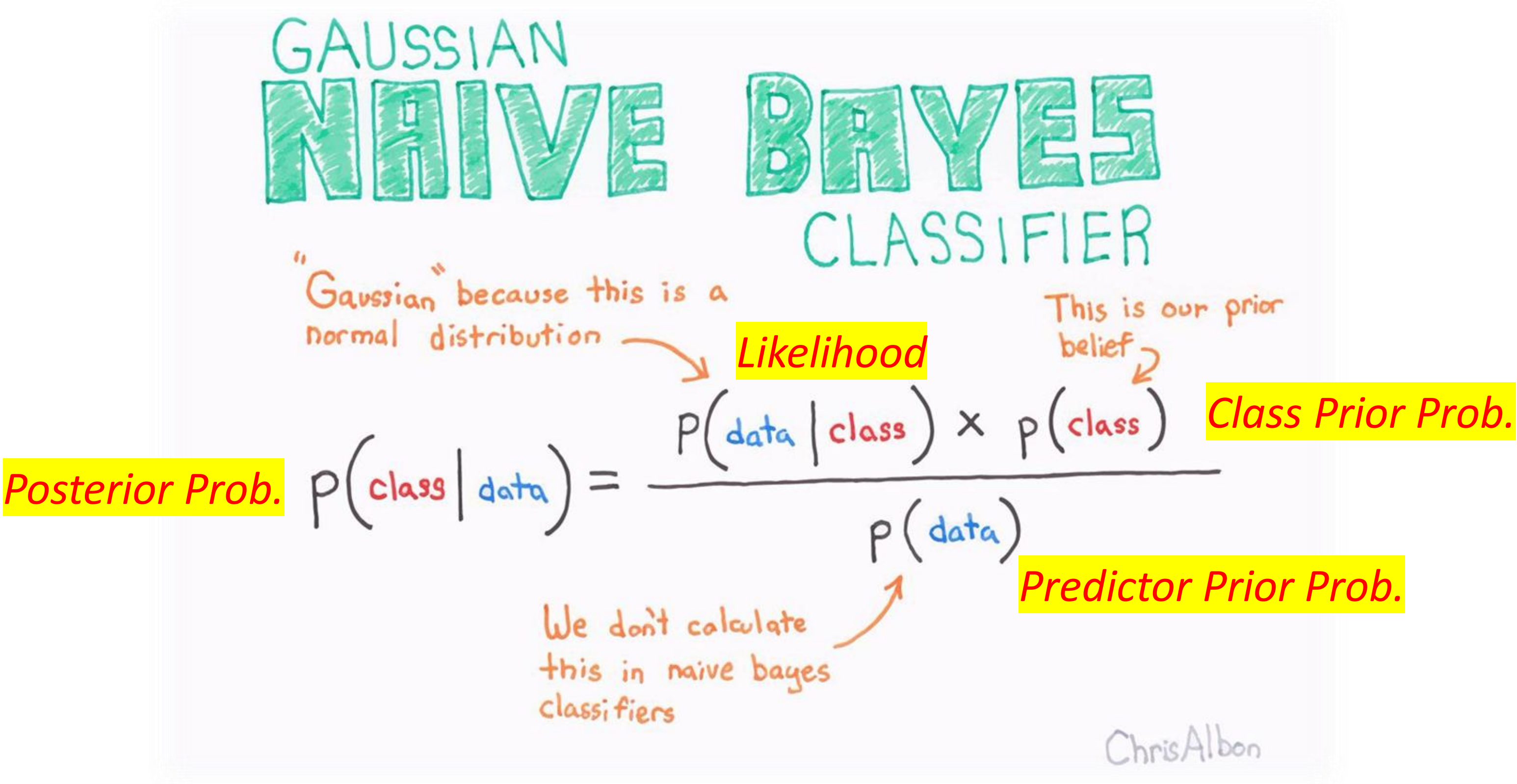
Dr. Binh Thanh Le
thanhbinh.le@ucd.ie
Insight Centre for Data Analytics
School of Computer Science
University College Dublin

Today Goals:

- Understanding the main concept of Naive Bayes classifier
- Applying the classifiers to Text Mining

What is Naive Bayes theorem?

In probability theory and statistics, **Bayes' theorem** (alternatively **Bayes' law** or **Bayes' rule**) describes the probability of an event, based on prior knowledge of conditions that might be related to the event.



Example

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

We have a training data set of weather and corresponding target variable ‘Play’ (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition.

Step 1: Convert the data set into a frequency table

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Example

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

We have a training data set of weather and corresponding target variable ‘Play’ (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition.

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Likelihood table

Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Example

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

We have a training data set of weather and corresponding target variable ‘Play’ (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Players will play if weather is sunny. Is this statement is correct?

P(Yes | Sunny) = P(Sunny | Yes) * P(Yes) / P (Sunny)

$$= \left(\frac{3}{9} * \frac{9}{14}\right) / \left(\frac{5}{14}\right) \quad = \mathbf{0.60}$$

P(No | Sunny) = P(Sunny | No) * P(No) / P (Sunny)

$$= \left(\frac{2}{5} * \frac{5}{14}\right) / \left(\frac{5}{14}\right) \quad = \mathbf{0.40}$$

Classification Methods

Supervised Machine Learning

- Input:
  - a document ***d***
  - a fixed set of classes $\mathbf{C = \{c_1, c_2, \ldots, c_J\}}$
  - A training set of m hand-labeled documents $\mathbf{(d_1,c_1),\ldots,(d_m,c_m)}$
- Output:
  - a learned classifier $\gamma\mathbf{:d \rightarrow c}$

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. 

Classification Methods
Supervised Machine Learning

- Any kind of classifier
  - Naïve Bayes
  - k-Nearest Neighbors
  - Logistic regression
  - Support-vector machines
  - …

The bag of words representation

$\gamma($

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

$\gamma($

x **love** xxxxxxxxxxxxxxxx **sweet**
xxxxxxx **satirical** xxxxxxxxxx
xxxxxxxxxxx **great** xxxxxxx
xxxxxxxxxxxxxxxxxxx **fun** xxxx
xxxxxxxxxxxxx **whimsical** xxxx
romantic xxxx **laughing**
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxx **recommend** xxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx **several** xxxxxxxxxxxxxxxxx
xxxxx **happy** xxxxxxxxx **again**
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxx

$) = c$

The bag of words representation

$\gamma($

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

$) = c$

Bayes' Rule Applied to Documents and Classes

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

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

Output

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

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

Bayes' Rule Applied to Documents and Classes

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

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

Document d represented as features x1..xn

Independence Assumptions

- **Bag of Words assumption**: Assume position doesn't matter
- **Conditional Independence**: Assume the feature probabilities ***$P(x_i | c_j)$*** are independent given the class c.

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

Bayes' Rule Applied to Documents and Classes

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

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

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

positions ← all word positions in test document

Bayes' Rule Applied to Documents and Classes

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

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

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

fraction of times word w_i appears among all words in documents of topic c_j

Algorithm

- From training corpus, extract *Vocabulary (V)*
- Calculate $P(c_j)$ terms
  - For each $c_j \in C$ do

$$docs_j \leftarrow \text{all docs with class} = c_j$$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

- Calculate $P(w_k|c_j)$ terms
  - $Text_j \leftarrow$ single doc containing all $docs_j$
  - For each word w_k in ***V***

$$n_k \leftarrow \text{\# of occurrences of } w_k \text{ in } Text_j$$

$$P(w_k|c_j) \leftarrow \frac{(n_k) + 1}{\sum_{w \in V} count(w|c_j) + |V|}$$

$P(c_1) * \prod P(w_k\|c_1)$	**A**
$P(c_2) * \prod P(w_k\|c_2)$	B

If (A>B) → Class 1
Else→ Class 2

Example

Train


That is a **good** film, the content is really **great**, especially **great** ending.


Great movie! **Excellent** ending, and **great** actions.


Great film ever! But the ending is so **sad**.


The ending is **boring**. The action is so **poor**. But the main actress's dress is so **great**!

Test

The film's ending is **great**. Main actor is super **great**, I love him.
Main actress is also **great**, but her action is a bit **boring** and **poor**.

	Doc	Words	Class
Train	1	Good Great Great	Positive
	2	Great Excellent Great	Positive
	3	Great Sad	Positive
	4	Boring Poor Great	Negative
Test	5	Great Great Great Boring Poor	?

Example

	Doc	Words	Class
Train	1	Good Great Great	Positive
	2	Great Excellent Great	Positive
	3	Great Sad	Positive
	4	Boring Poor Great	Negative
Test	5	Great Great Great Boring Poor	?

	Great	Good	Excellent	Sad	Boring	Poor	Class
Doc1	2	1	0	0	0	0	1
Doc2	2	0	1	0	0	0	1
Doc3	1	0	0	1	0	0	1
Doc4	1	0	0	0	1	1	-1

Prior Prob.

$$P\{+1\} = \frac{3}{4}$$

$$P\{-1\} = \frac{1}{4}$$

Example

	Great	Good	Excellent	Sad	Boring	Poor	Class
Doc1	2	1	0	0	0	0	1
Doc2	2	0	1	0	0	0	1
Doc3	1	0	0	1	0	0	1
Doc4	1	0	0	0	1	1	-1

Prior Probabilities:

$$P\{+1\} = \frac{3}{4}$$

$$P\{-1\} = \frac{1}{4}$$

Conditional Probabilities:

P(Great|1) = (5+1) / (8+6) = 6/14
P(Good|1) = (1+1) / (8+6) = 2/14
P(Excellent|1) = (1+1) / (8+6) = 2/14
P(Sad |1) = (1+1) / (8+6) = 2/14
P(Boring |1) = (0+1) / (8+6) = 1/14
P(Poor|1) = (0+1) / (8+6) = 1/14

P(Great|-1) = (1+1) / (3+6) = 2/9
P(Good|-1) = (0+1) / (3+6) = 1/9
P(Excellent|-1) = (0+1) / (3+6) = 1/9
P(Sad|-1) = (0+1) / (3+6) = 1/9
P(Boring|-1) = (1+1) / (3+6) = 2/9
P(Poor|-1) = (1+1) / (3+6) = 2/9

Example

Conditional Probabilities:

P(Great|1) = (5+1) / (8+6) = 6/14
P(Good|1) = (1+1) / (8+6) = 2/14
P(Excellent|1) = (1+1) / (8+6) = 2/14
P(Sad |1) = (1+1) / (8+6) = 2/14
P(Boring |1) = (0+1) / (8+6) = 1/14
P(Poor|1) = (0+1) / (8+6) = 1/14

P(Great|-1) = (1+1) / (3+6) = 2/9
P(Good|-1) = (0+1) / (3+6) = 1/9
P(Excellent|-1) = (0+1) / (3+6) = 1/9
P(Sad|-1) = (0+1) / (3+6) = 1/9
P(Boring|-1) = (1+1) / (3+6) = 2/9
P(Poor|-1) = (1+1) / (3+6) = 2/9

Prior Probabilities:

$$P\{+1\} = \frac{3}{4}$$

$$P\{-1\} = \frac{1}{4}$$

$$P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

Test	5	Great Great Great Boring Poor	?

Choosing a class:

P(+1|d5) = (3/4) * (6/14) * (6/14) * (6/14) * (1/14) * (1/14)
≈ 0.0003

P(+1|d5) = (1/4) * (2/9) * (2/9) * (2/9) * (2/9) * (2/9)
≈ 0.0001

➔ d5 is Positive

Summary

- Naive Bayes is
  - Very Fast, low storage requirements
  - Robust to Irrelevant Features
  - Very good in domains with many equally important features
  - Optimal if the independence assumptions hold
  - A good dependable baseline for text classifica1on