

# COMP9727: Recommender Systems

## Lecture 2: Content-Based Recommender Systems

Wayne Wobcke

e-mail:w.wobcke@unsw.edu.au

### This Lecture

- Machine Learning Methodology
- Text Classification
  - ▶ Bernoulli Naive Bayes Classification
  - ▶ Multinomial Naive Bayes Classification
- User Profiles and Recommendation
  - ▶ Document similarity

### Content-Based Recommendation

Useful when items **can** easily be categorized

- News, Research Articles, Meals, Hotels, Restaurants, etc.
- Categorize items using a predefined(?) ontology?
- Build **user profile** of interests using the same ontology
- Recommend items based on “similarity” between item and profile

Data sources: Explicit user interests, user–system interactions

Many ways to define “similarity”

### Supervised Learning

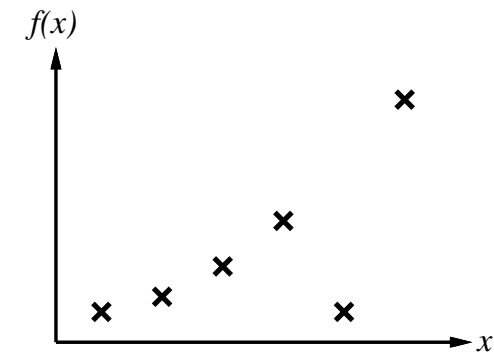
- Given a **training set** and a **test set**, each consisting of a set of items for each item in the training set, a set of features and a target output
- Learner must learn a **model** that can **predict** the target output for **any** given item (characterized by its set of features)
- Learner is given the input features and target output for each item in the training set
  - ▶ Items may be presented all at once (batch) or in sequence (online)
  - ▶ Items may be presented at random or in time order (stream)
  - ▶ Learner **cannot** use the test set **at all** in defining the model
- Model is evaluated by its performance on predicting the output for each item in the **test set**

## Methods vs Models

- Various learning **methods** can be used to generate models
  - ▶ Decision Trees
  - ▶ Support Vector Machines
  - ▶ Neural Networks/Deep Learning
- Evaluate methods by evaluating models on a variety of datasets
  - ▶ Problem with availability of standard benchmark datasets
  - ▶ Models depend on problem formulation and on parameters
  - ▶ End users may only care about a model, not a general method
  - ▶ Most machine learning research evaluates methods, not models

## Curve Fitting

Which curve gives the “best fit” to this data?

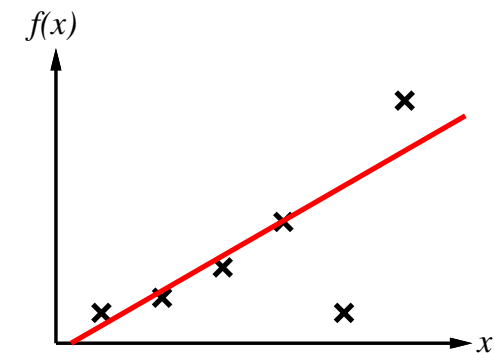


## Supervised Learning – Methodology

- Feature “engineering” – select relevant features
- Choose representation of input features and outputs
- Preprocessing method to extract features from raw data
- Choose learning method(s) to evaluate
- Choose training regime (including parameters)
- Evaluation
  - ▶ Choose **realistic** baseline for comparison
  - ▶ Choose type of internal **validation**, e.g. cross-validation
  - ▶ Sanity check results with human expertise, other benchmarks

## Curve Fitting

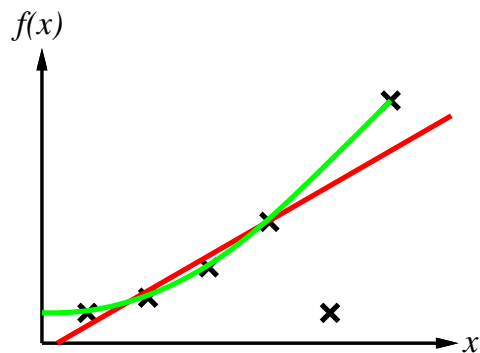
Which curve gives the “best fit” to this data?



Straight line?

## Curve Fitting

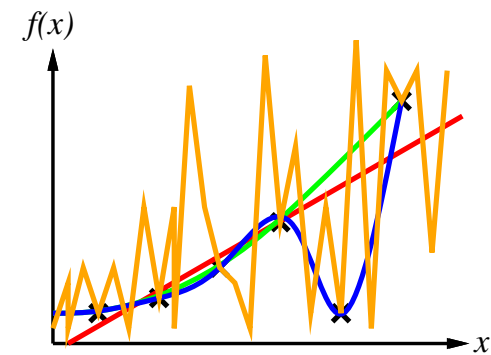
Which curve gives the “best fit” to this data?



Parabola?

## Curve Fitting

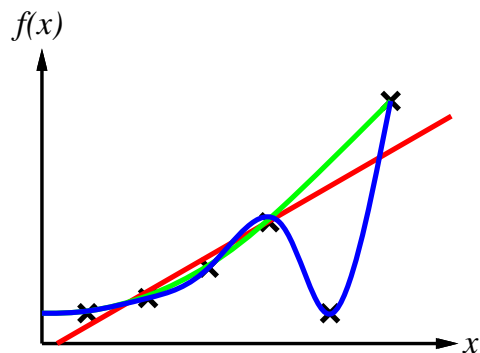
Which curve gives the “best fit” to this data?



Something else?

## Curve Fitting

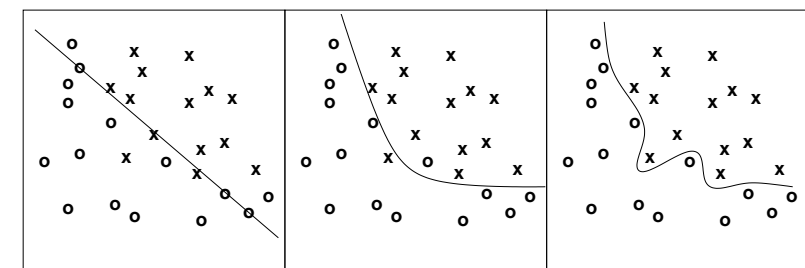
Which curve gives the “best fit” to this data?



4th order polynomial?

## Ockham's Razor

“The most likely hypothesis is the **simplest** one consistent with the data.”



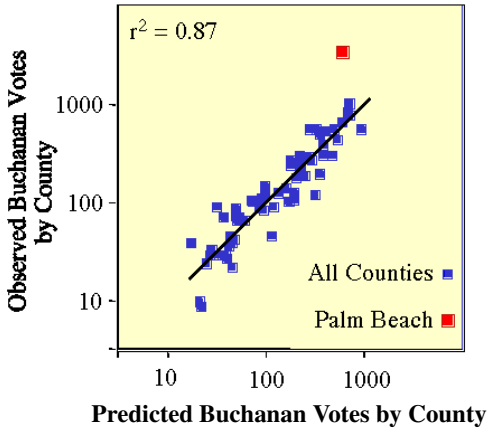
inadequate

good compromise

overfitting

Since there can be **noise** in the measurements, in practice need to make a **tradeoff** between simplicity of the hypothesis and how well it fits the data

Outliers

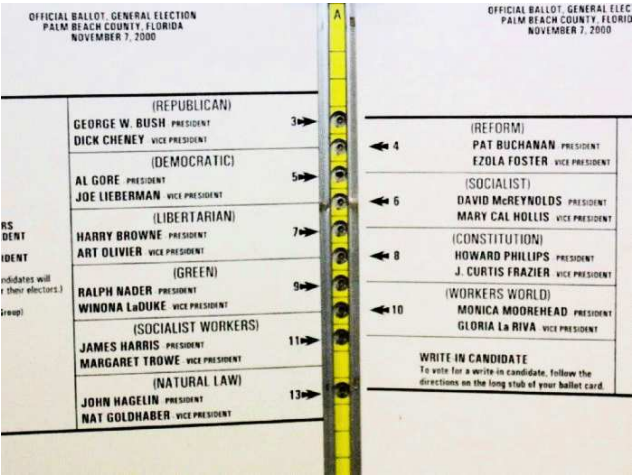


When is it OK to remove outliers?

Text Classification Applications

- Spam Detection
- E-Mail Classification
- News/Scientific Article Topic Classification
- Event Extraction (Event Type Classification)
- Employment Statistics from Job Advertisements
- Medical Treatment Categorization for Insurance Claims
- Sentiment Analysis from Reviews

Butterfly Ballot



Example Movie Reviews/Ratings

- ... unbelievably disappointing ...
- Full of zany characters and richly applied satire, and some great plot twists.
- The greatest screwball comedy ever filmed.
- It was pathetic. The worst part about it was the boxing scenes.

## Supervised Learning

- Input: A **document** (e-mail, news article, review, tweet)
- Output: One **class** (label) drawn from a **fixed set** of classes
  - ▶ So text classification is a **multi-class** classification problem
  - ▶ ... and sometimes a **multi-label** classification problem
- Learning Problem
  - ▶ Input: Training set of labelled documents  $\{(d_1, c_1), \dots\}$
  - ▶ Output: Learned classifier that maps  $d$  to predicted class  $c$

## Probability Theory

- **Simple event**: Atomic event/proposition/fact/belief/...
- **Event**: Set of simple events (meaning “any one of the events”)
- **Complex events**:  $A \wedge B = A \cap B$ ;  $A \vee B = A \cup B$ ;  $\neg A = \mathcal{U} - A$  where  $\mathcal{U}$  is the set of all simple events
- **Probability distribution**: Assignment from  $[0,1]$  to each event
  1.  $P(A) \geq 0$  for all  $A$
  2.  $P(\mathcal{U}) = 1$
  3.  $P(A \vee B) = P(A) + P(B)$  if  $A$  and  $B$  mutually exclusive
    - ▶ Hence  $P(\neg A) = 1 - P(A)$
    - ▶ Hence  $P(A \vee B) = P(A) + P(B) - P(A \wedge B)$

## Prior and Conditional Probabilities

How to **update** probabilities based on new information

- $P(A)$  is the **prior** or **unconditional** probability of  $A$  in the absence of any other information
- $P(A|B)$  is the **conditional** or **posterior** probability of  $A$  given  $B$ 
  - ▶ **Definition**:  $P(A|B) = \frac{P(A \wedge B)}{P(B)}$  provided  $P(B) > 0$
  - ▶ **Product Rule**:  $P(A \wedge B) = P(A|B).P(B)$

## Bayes' Rule

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

- Deriving Bayes' Rule:
  - $P(A \wedge B) = P(A|B)P(B)$  (Definition)
  - $P(B \wedge A) = P(B|A)P(A)$  (Definition)
  - So  $P(A|B)P(B) = P(B|A)P(A)$  since  $P(A \wedge B) = P(B \wedge A)$
  - Hence  $P(B|A) = \frac{P(A|B)P(B)}{P(A)}$  if  $P(A) \neq 0$
- **Note**: If  $P(A) = 0$ ,  $P(B|A)$  is undefined

## Conditional Independence

- $A$  is **conditionally independent** of  $B$  given background knowledge  $K$  if knowing  $B$  does not affect the conditional probability of  $A$  given  $K$ :

$$P(A|B, K) = P(A|K)$$

where  $P(A|B, K)$  means  $P(A|B \wedge K)$

- If  $A$  is conditionally independent of  $B$  given  $K$  then

$$P(A \wedge B|K) = P(A|K) \cdot P(B|K)$$

- ▶ Because by definition,  $P(A \wedge B|K) = P(A|B, K) \cdot P(B|K)$

- Typically make **assumptions** of conditional independence

## Feature Engineering

Example: SpamAssassin (Spam E-Mail)

- Mentions Generic Viagra
- Online Pharmacy
- Mentions millions of (dollar) ((dollar) NN,NNN,NNN.NN)
- Phrase: impress ... girl
- From: starts with many numbers
- Subject is all capitals
- HTML has a low ratio of text to image area
- One hundred percent guaranteed
- Claims you can be removed from the list

[http://spamassassin.apache.org/old/tests\\_3\\_3\\_x.html](http://spamassassin.apache.org/old/tests_3_3_x.html)

## Probabilistic Text Classification

- Events: Document has **feature**  $x_i$ , **class**  $c$
- Classify: Given document with features  $x_1, \dots, x_n$ , choose  $c$  so that  $P(c|x_1, \dots, x_n)$  is maximized
- Apply Bayes' Rule
  - ▶  $P(c|x_1, \dots, x_n) = \frac{P(x_1, \dots, x_n|c) \cdot P(c)}{P(x_1, \dots, x_n)}$
  - ▶ Therefore maximize  $P(x_1, \dots, x_n|c) \cdot P(c)$

## Data Cleansing

Typical preprocessing pipeline: see tutorial

1. Remove extraneous characters (\$, %, , , , , etc.)
2. Remove emojis and HTML tags or follow links??
3. Apply stemming (Porter, Lancaster, Snowball, ...)?
4. Remove stopwords: commonly occurring words (the, a, we, etc.)
5. Remove words of length 1 (a, I, etc.)
6. Convert upper to lower case?
7. Take N most frequent words, for some (what) N?
8. Features are frequencies or tf-idf values (see later)?

After this, each document is a set of words/word stems or a tf-idf vector

## Bernoulli Model

Maximize  $P(x_1, \dots, x_n | c) \cdot P(c)$

- Features are presence or absence of word  $w_i$  in document
- Apply independence assumptions
  - ▶  $P(x_1, \dots, x_n | c) = P(x_1 | c) \cdot \dots \cdot P(x_n | c)$
  - ▶ Probability of word  $w$  (not) in class  $c$  independent of context
- Estimate probabilities
  - ▶  $P(w | c) = \#(w \text{ in document in class } c) / \#(\text{documents in class } c)$
  - ▶  $P(\neg w | c) = 1 - P(w | c)$
  - ▶  $P(c) = \#(\text{documents in class } c) / \#(\text{documents})$

## Bag of Words Model

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!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1

## Bernoulli Naive Bayes Classification

$w_1$	$w_2$	$w_3$	$w_4$	Class
1	0	0	1	1
0	0	0	1	0
1	1	0	1	0
1	0	1	1	1
0	1	1	0	0
1	0	0	0	0
1	0	1	0	1
0	1	0	0	1
0	1	0	1	0
1	1	1	0	0

	Class = 1	Class = 0
$P(\text{Class})$	0.40	0.60
$P(w_1   \text{Class})$	0.75	0.50
$P(w_2   \text{Class})$	0.25	0.67
$P(w_3   \text{Class})$	0.50	0.33
$P(w_4   \text{Class})$	0.50	0.50

To classify document with  $w_2, w_3, w_4$

- $P(\text{Class} = 1 | \neg w_1, w_2, w_3, w_4)$   
 $\approx ((1 - 0.75) * 0.25 * 0.5 * 0.5) * 0.4$   
 $= 0.00625$
- $P(\text{Class} = 0 | \neg w_1, w_2, w_3, w_4)$   
 $\approx ((1 - 0.5) * 0.67 * 0.33 * 0.5) * 0.6$   
 $= 0.03333$

## Multinomial Naive Bayes Classification

Maximize  $P(x_1, \dots, x_n | c) \cdot P(c)$

- Features are occurrence of word in positions in document
- Apply independence assumptions
  - ▶  $P(w_1, \dots, w_n | c) = P(w_1 | c) \cdot \dots \cdot P(w_n | c)$
  - ▶ Position of word  $w$  in document doesn't matter
- Estimate probabilities
  - ▶ Let  $V$  be the vocabulary
  - ▶ Let “document”  $c$  = concatenation of documents in class  $c$
  - ▶  $P(w | c) = \#(w \text{ in document } c) / \sum_{w \in V} \#(w \text{ in document } c)$
  - ▶  $P(c) = \#(\text{documents in class } c) / \#(\text{documents})$

# Laplace Smoothing

- What if word in test document has not occurred in class  $c$  in training?
- Then  $P(w|c) = 0$  and so estimate for class  $c$  is 0
- Laplace smoothing
  - ▶ Assign small probability to unseen words
  - ▶  $P(w|c) = (\#(w \text{ in document } c)+1)/(\sum_{w \in V} \#(w \text{ in document } c)+|V|)$
  - ▶ Don't have to add 1, can be 0.05 or some parameter  $\alpha$

# Summary: Naive Bayes

- Very fast, low storage requirements
- Robust to irrelevant features
- Irrelevant features cancel each other without affecting results
- Very good in domains with many equally important features
- Good dependable baseline for text classification

# MNB Example

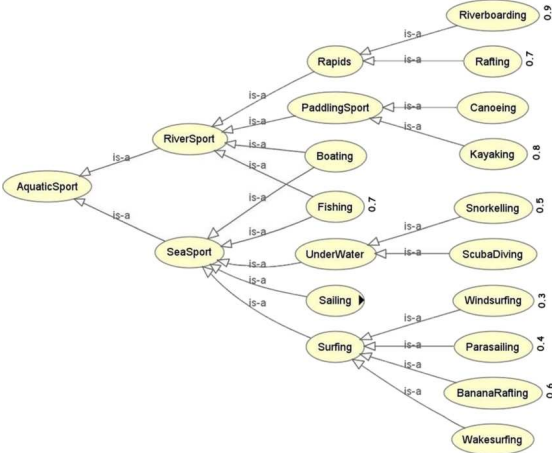
	Words	Class
$d_1$	Chinese Beijing Chinese	$c$
$d_2$	Chinese Chinese Shanghai	$c$
$d_3$	Chinese Macao	$c$
$d_4$	Tokyo Japan Chinese	$j$
$d_5$	Chinese Chinese Chinese Tokyo Japan	?

$P(\text{Chinese}|c) = (5+1)/(8+6) = 3/7$   
 $P(\text{Tokyo}|c) = (0+1)/(8+6) = 1/14$   
 $P(\text{Japan}|c) = (0+1)/(8+6) = 1/14$   
 $P(\text{Chinese}|j) = (1+1)/(3+6) = 2/9$   
 $P(\text{Tokyo}|j) = (1+1)/(3+6) = 2/9$   
 $P(\text{Japan}|j) = (1+1)/(3+6) = 2/9$

To classify document  $d_5$ 

- $P(c|d_5) \propto [(3/7)^3 \cdot 1/14 \cdot 1/14] \cdot 3/4 \approx 0.0003$
- $P(j|d_5) \propto [(2/9)^3 \cdot 2/9 \cdot 2/9] \cdot 1/4 \approx 0.0001$
- Choose Class  $c$

# User Profile: Aquatic Sports Ontology





## Issues with Ontologies

### Advantages

- ▶ User interests can be fine-grained and precise
- ▶ Transparency: users can see and sometimes edit profile
- ▶ Explanation: why article considered relevant

### Disadvantages

- ▶ Categories can be too few (coarse) or too many (unwieldy)
- ▶ Not clear to users what the numbers mean (in this example)
- ▶ Hard to classify if categories overlap, e.g. politics in sport
- ▶ Recommendations are very similar: users lose interest over time
- ▶ Polarization: articles may present only one side of an argument
- ▶ Limited scope for novelty, diversity

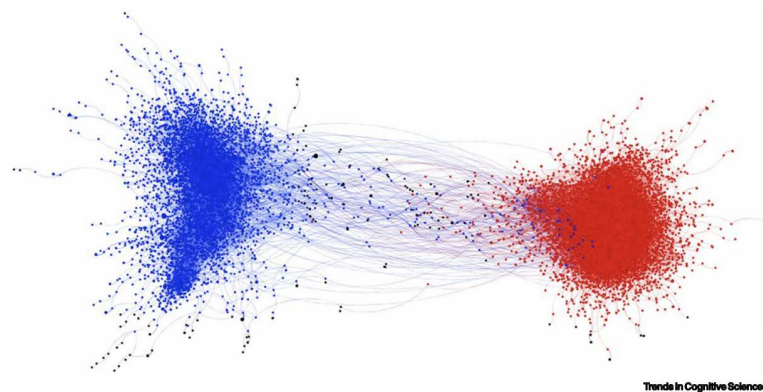
## User Profiles and Recommendation

Use information retrieval techniques:

- For each user, for each category, associate a “document” consisting of every article read by the user classified under that topic
  - ▶ Or a subset of those words if the whole document is too long?
- Recommend articles based on similarity to those “documents”
  - ▶ Or to the “average” vector of those documents?

Many ways to define document “similarity”

## Political Polarization on Twitter



Blue = Democrat; Red = Republican

## Ideas from Information Retrieval

- Document set  $D$ , each “document” a collection of **terms**
- Each document is represented by a vector, one element for each term
  - ▶ Term frequency  $\text{tf}(w, d)$ : number of occurrences of  $w$  in  $d$
  - ▶ Document frequency  $\text{df}(w)$ : number of documents containing  $w$
  - ▶ Inverse document frequency  $\text{idf}(w, d)$ :  $\log_2 \frac{|D|}{\text{df}(w)+1}$
  - ▶ tf-idf:  $\text{tf-idf}(w, d) = \text{tf}(w, d) \cdot \text{idf}(w, d)$

## Document Similarity

Each “document” is a set or bag of words (or word stems) or a vector of word counts or tf-idf values

- Jaccard similarity

- ▶  $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$

- Sørensen–Dice coefficient

- ▶  $DSC(A, B) = \frac{2 \cdot |A \cap B|}{|A| + |B|}$

- Cosine similarity: angle between  $\vec{A}$  and  $\vec{B}$

- ▶  $\cos(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \cdot \|\vec{B}\|}$  where  $\|\langle x_1, \dots, x_n \rangle\| = \sqrt{\sum_{i=1}^n x_i^2}$

- Euclidean distance: distance between  $\vec{A}$  and  $\vec{B}$

- ▶  $\|\vec{B} - \vec{A}\|$

## Other Text Classification Methods

- Support Vector Machines
- K-Nearest Neighbour
- Decision Trees
- Random Forests
- Multi-Layer Perceptrons
  - ▶ Using word2vec, GloVe, BERT, etc., representations
- Deep Learning
  - ▶ RNN, LSTM, HAN, CNN

## Assignment

Data from 2000 movies in 8 genres, 250 movies in each genre

- Topic (Genre) Classification

- ▶ Compare Naive Bayes and one other method for classification

- Recommendation Methods

- ▶ Train on combined first 150 movies in each genre, test on next 50
  - ▶ Use `TfidfVectorizer` to get tf-idf values on training set
  - ▶ Use vectorizer to define user profile: one vector for each genre
  - ▶ Recommend new movie to user if movie predicted to be in genre  $g$  and movie “similar” to user profile vector for genre  $g$

- User Study

- ▶ Choose one friendly user and simulate the recommender system

## Summary

- Useful when content easy to categorize
- Favours precision over recall?
- ... at potential cost of explicit user feedback
- Hence less diversity, novelty, serendipity
- Can handle cold start problem for items

Still lots of work to do for feature engineering