Data Mining and MapReduce

Adapted from Lectures by
Prabhakar Raghavan (Yahoo and Stanford) and Christopher Manning (Stanford)

Overview

- Text Classification
- K-Means Classification
- The Naïve Bayes algorithm

Relevance feedback revisited

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant, that can be used to *improve* search results.
- Suppose we just tried to learn a *filter* for nonrelevant documents
- This is an instance of a text classification problem:
  - Two "classes": **relevant**, **nonrelevant**
  - For each document, decide whether it is relevant or nonrelevant
- The notion of classification is very general and has many applications within and beyond IR.

Standing queries

- The path from information retrieval to text classification:
  - You have an information need, say:
    - Unrest in the Niger delta region
  - You want to rerun an appropriate query periodically to find *new* news items on this topic
    - I.e., it's classification not ranking
- Such queries are called ***standing queries***
  - Long used by "information professionals"
  - A modern mass instantiation is **Google Alerts**

Spam filtering: Another text classification task

From: "" <takworlld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay

Anyone can buy real estate with no money down

Stop paying rent TODAY !

There is no need to spend hundreds or even thousands for similar courses

I am 22 years old and I have already purchased 6 properties using the methods outlined in this truly INCREDIBLE ebook.

Change your life NOW !

==========

Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm

==========

Categorization/Classification

- Given:
  - A description of an instance, $x \in X$, where X is the *instance language* or *instance space*.
    - *Issue*: how to represent text documents.
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$
- Determine:
  - The category of x: $c(x) \in C$, where $c(x)$ is a *classification function* whose domain is X and whose range is C.
    - We want to know how to build classification functions ("classifiers").

More Text Classification Examples:

Many search engine functionalities use classification

Assign labels to each document or web-page:

- Labels are most often topics such as Yahoo-categories
  e.g., "finance," "sports," "news>world>asia>business"
- Labels may be genres
  e.g., "editorials" "movie-reviews" "news"
- Labels may be opinion on a person/product
  e.g., "like", "hate", "neutral"
- Labels may be domain-specific
  e.g., "interesting-to-me" : "not-interesting-to-me"
  e.g., "contains adult language" : "doesn't"
  e.g., language identification: English, French, Chinese, ...
  e.g., search vertical: about Linux versus not
  e.g., "link spam" : "not link spam"

Classification Methods (1)

- Manual classification
  - Used by Yahoo! (originally; now downplayed), Looksmart, about.com, ODP, PubMed
  - Very *accurate* when job is done by experts
  - *Consistent* when the problem size and team is small
  - *Difficult and expensive to scale*
    - Means we need automatic classification methods for big problems

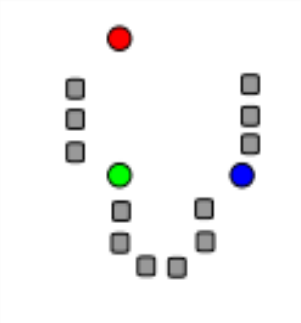
Classification Methods (2)

- Supervised learning of a document-label assignment function
  - Many systems partly rely on machine learning (Autonomy, MSN, Verity, Enkata, Yahoo!, ...)
    - k-Nearest Neighbors (simple, powerful)
    - Naive Bayes (simple, common method)
    - Support-vector machines (new, more powerful)
    - ... plus many other methods
    - No free lunch: requires hand-classified training data
    - But data can be built up (and refined) by amateurs
- Note that many commercial systems use a mixture of methods

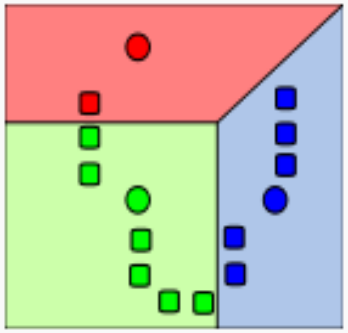
Clustering

- k-means
  - Basic and fundamental
  - Original Algorithm
    1. Pick k initial center points
    2. Iterate until converge
       1. Assign each point with the nearest center
       2. Calculate new centers
  - Easy to parallellize

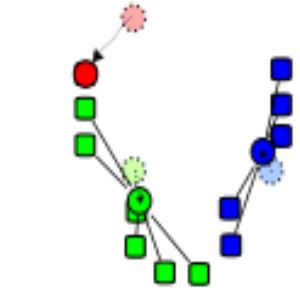
K-Means Clustering



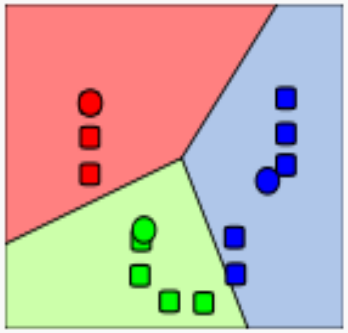
1) *k* initial "means" (in this case *k*=3) are randomly selected from the data set (shown in color).



2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

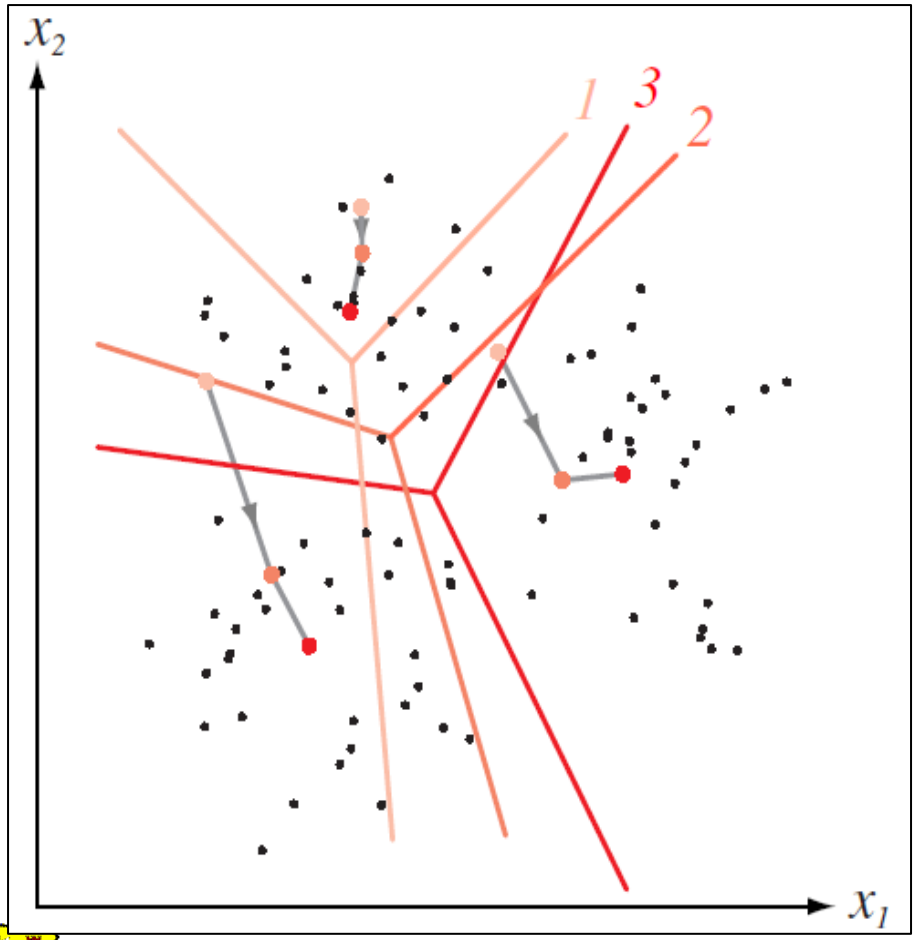


3) The centroid of each of the *k* clusters becomes the new means.



4) Steps 2 and 3 are repeated until convergence has been reached.

How Does K-means partition?



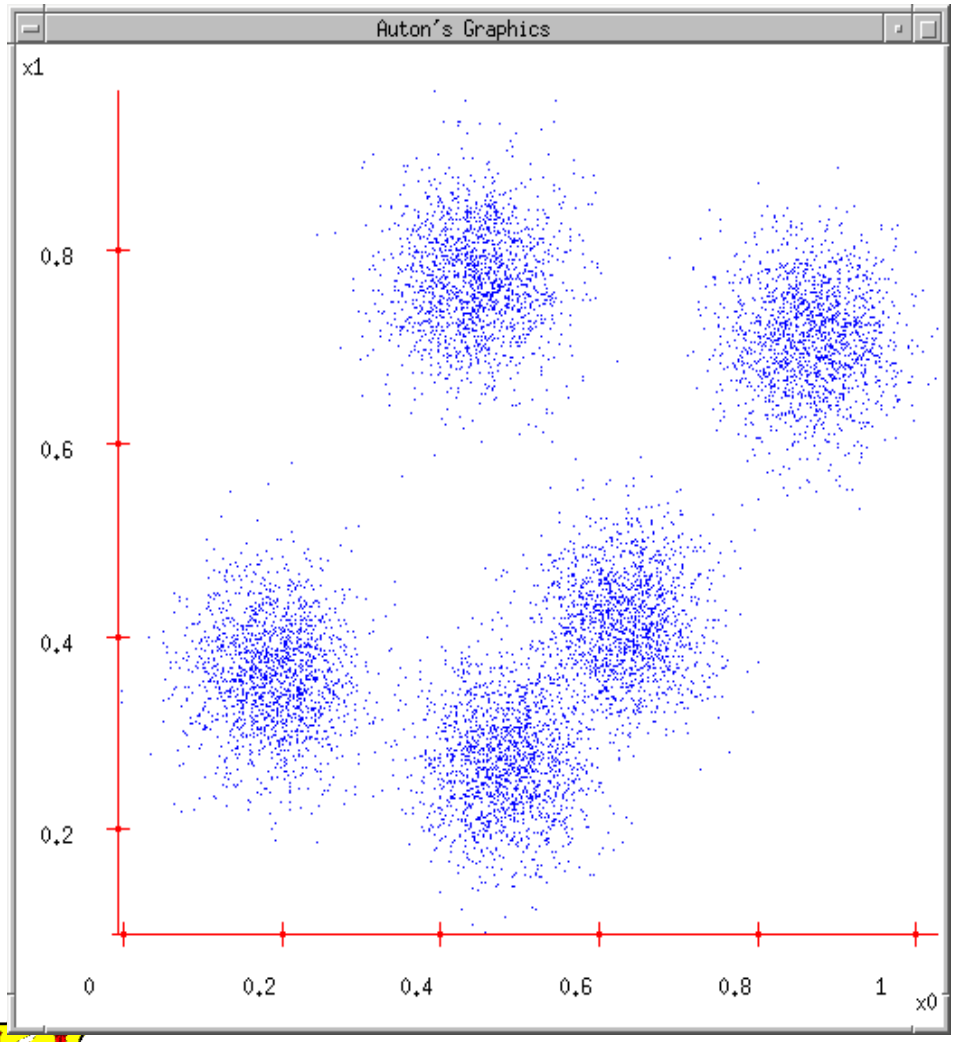
When *K* centroids are set/fixed, they partition the whole data space into *K* mutually exclusive subspaces to form a partition.

A partition amounts to a

Voronoi Diagram

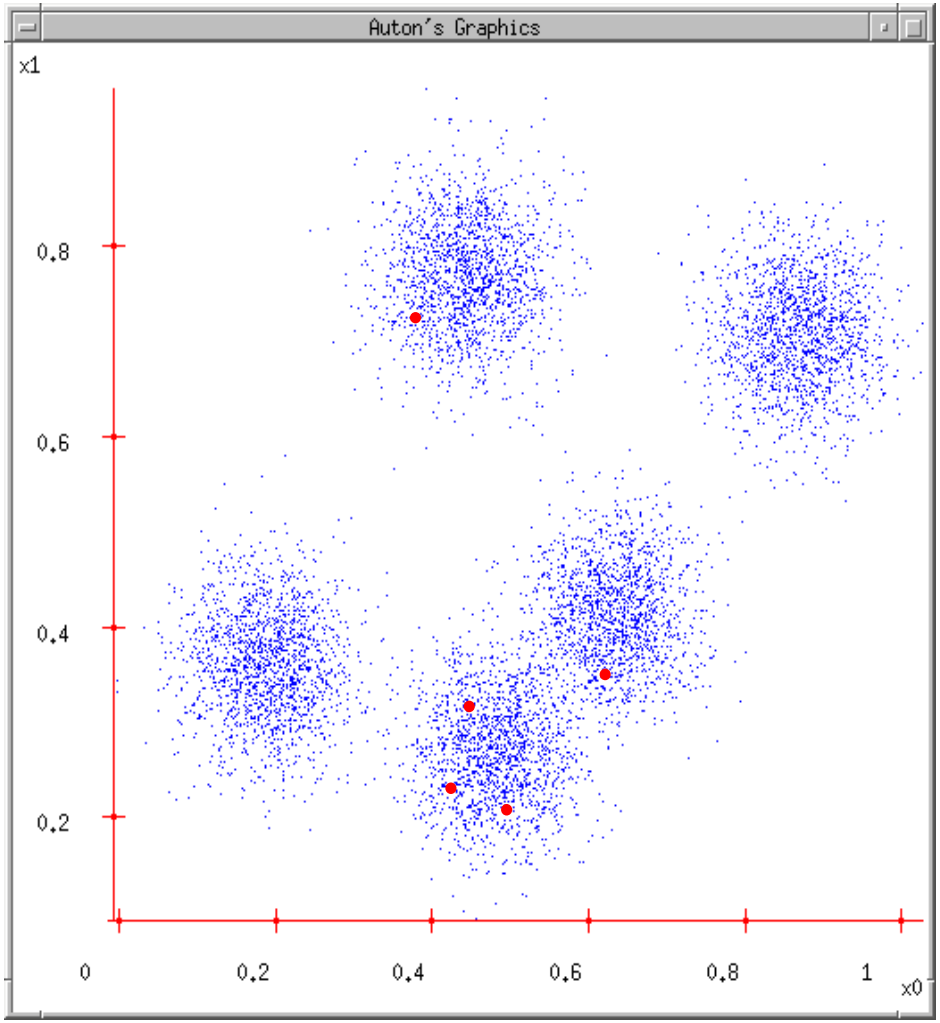
Changing positions of centroids leads to a new partitioning.

K-means Demo



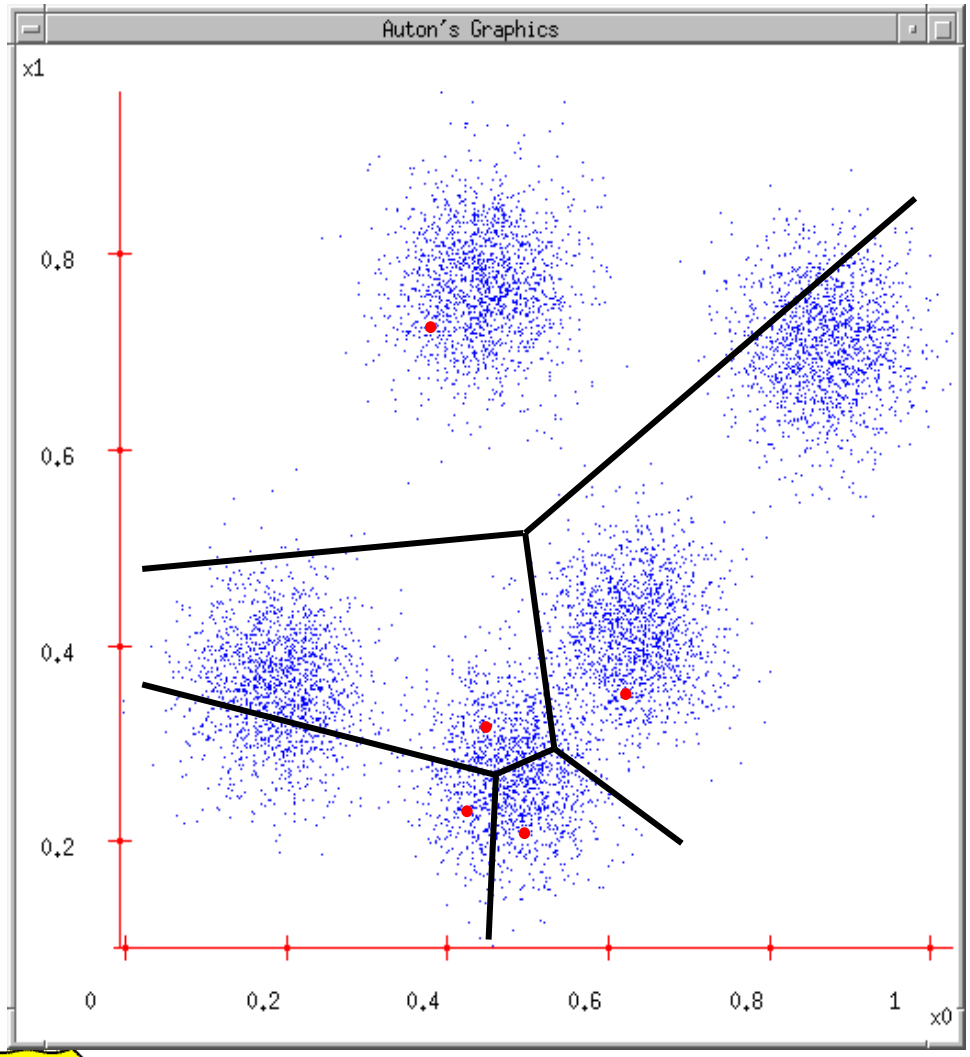
1. User set up the number of clusters they'd like. *(e.g. k=5)*

K-means Demo



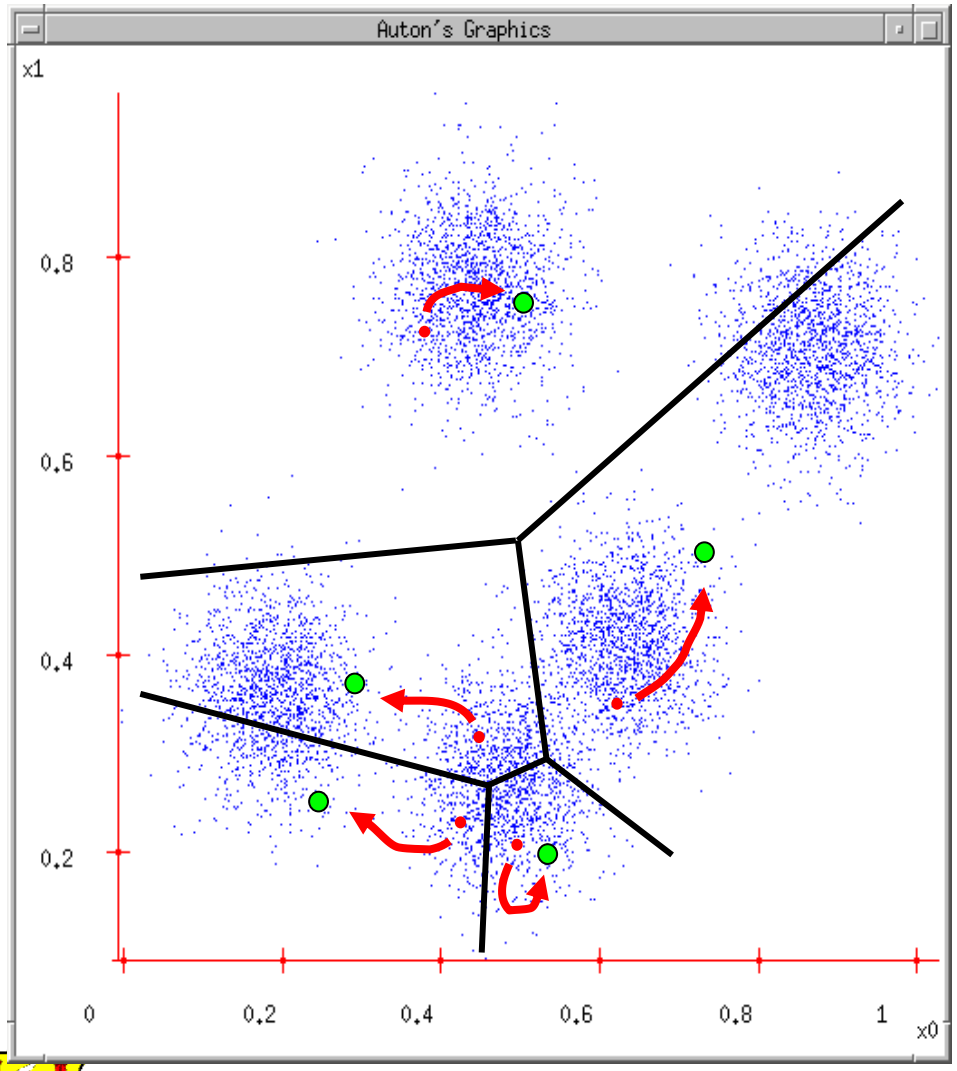
1. User set up the number of clusters they'd like. *(e.g. K=5)*
2. Randomly guess K cluster Center locations

K-means Demo



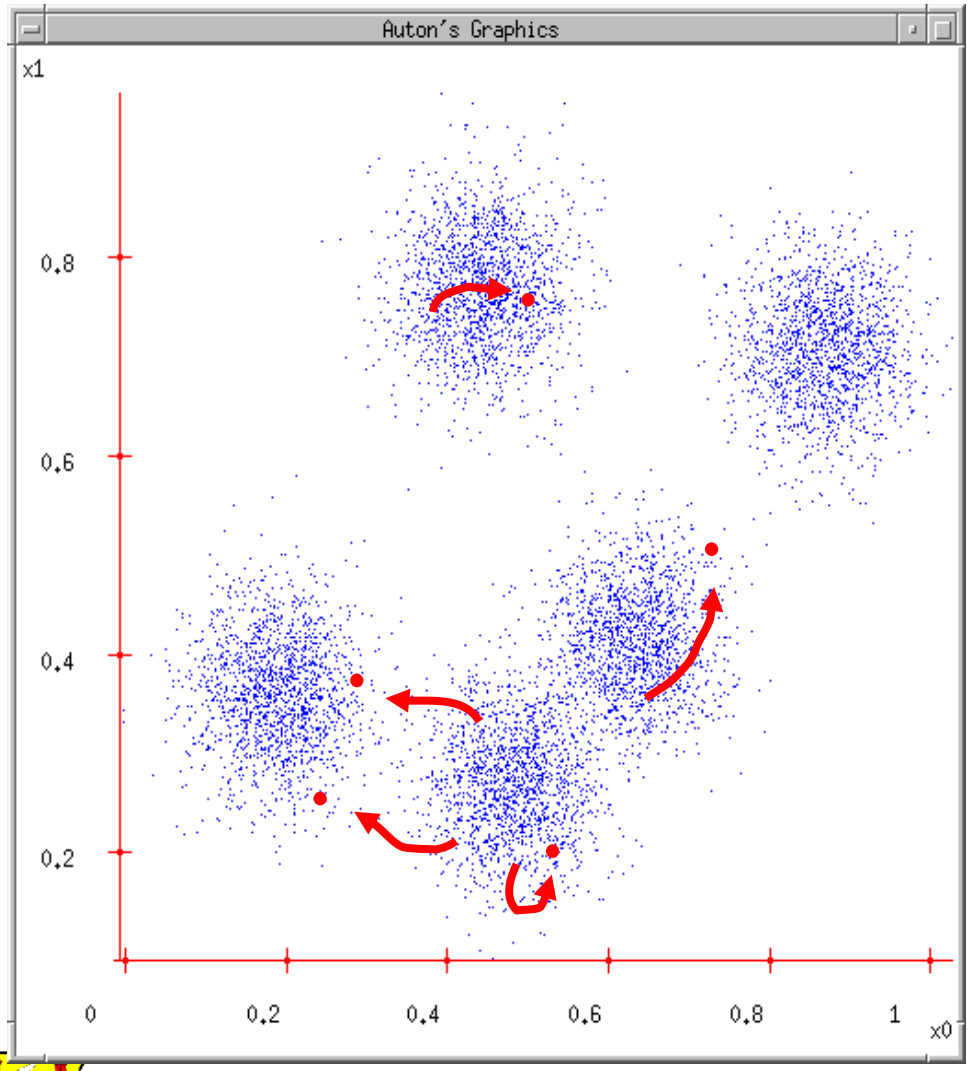
1. User set up the number of clusters they'd like. *(e.g. K=5)*
2. Randomly guess *K* cluster Center locations
3. Each data point finds out which Center it's closest to. (Thus each Center "owns" a set of data points)

K-means Demo



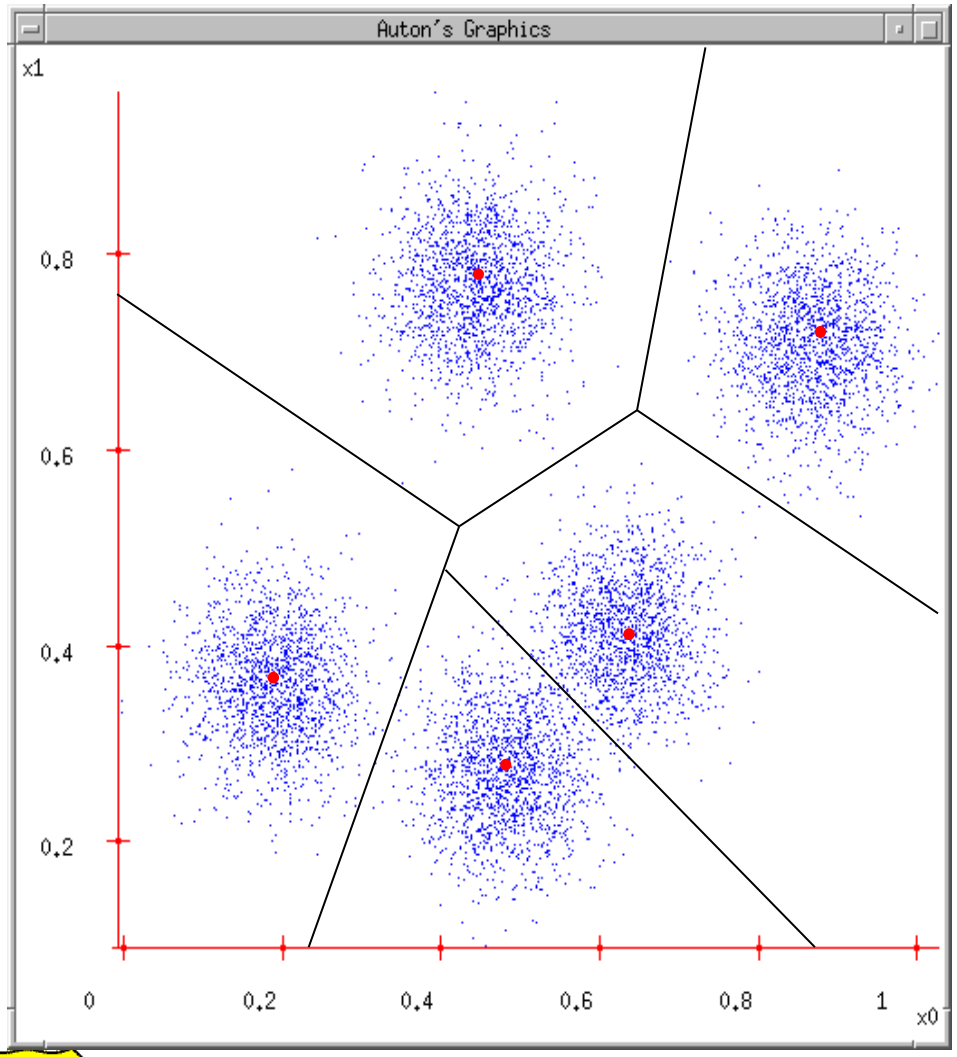
1. User set up the number of clusters they'd like. *(e.g. K=5)*
2. Randomly guess *K* cluster centre locations
3. Each data point finds out which centre it's closest to. (Thus each Center "owns" a set of data points)
4. Each centre finds the centroid of the points it owns

K-means Demo



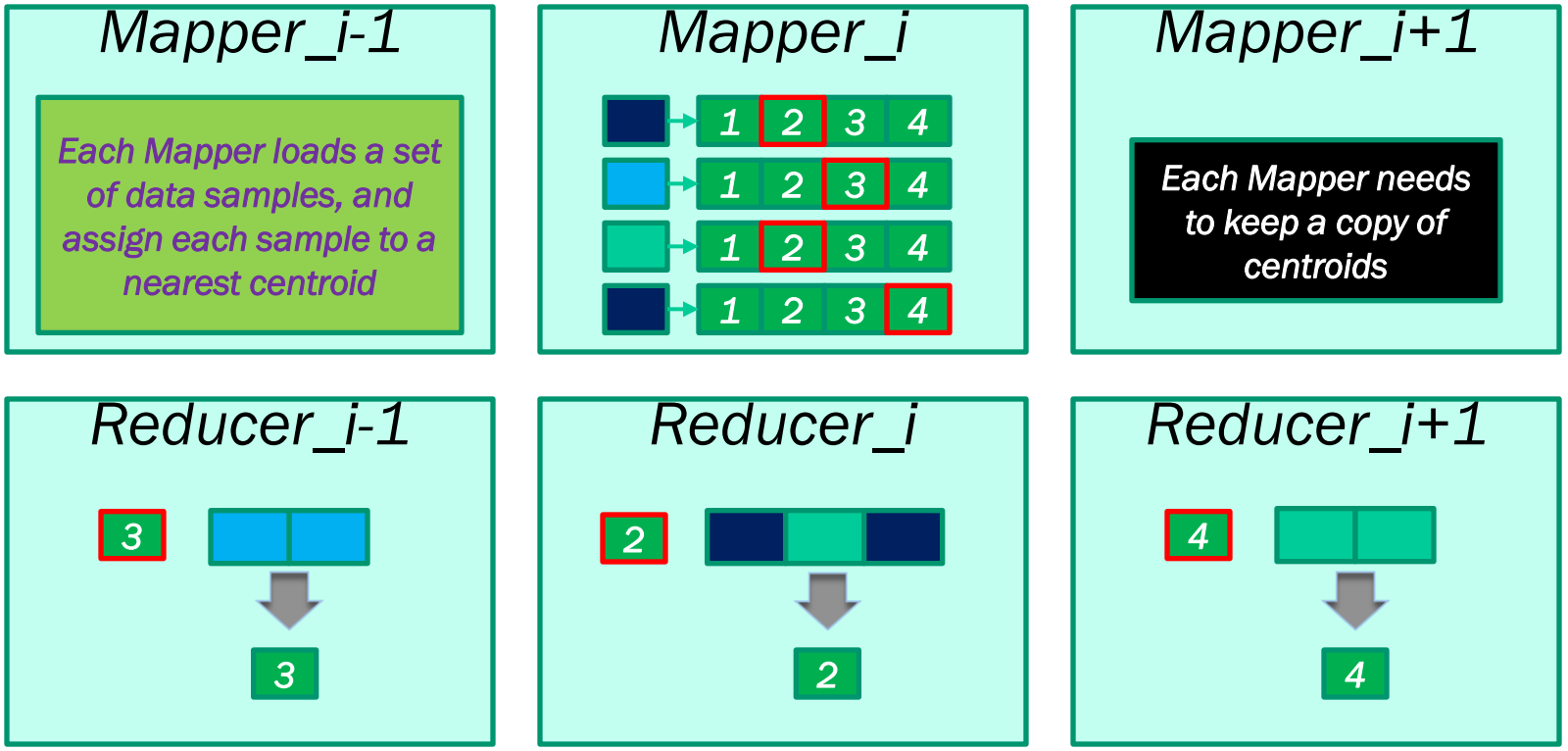
1. User set up the number of clusters they'd like. (e.g. *K=5*)
2. Randomly guess *K* cluster centre locations
3. Each data point finds out which centre it's closest to. (Thus each centre "owns" a set of data points)
4. Each centre finds the centroid of the points it owns
5. …and jumps there

K-means Demo



1. User set up the number of clusters they'd like. (e.g. *K=5*)
2. Randomly guess *K* cluster centre locations
3. Each data point finds out which centre it's closest to. (Thus each centre "owns" a set of data points)
4. Each centre finds the centroid of the points it owns
5. …and jumps there
6. …Repeat until terminated!

K-Means Clustering with MapReduce



How to set the initial centroids is very important!
Usually we set the centroids using Canopy Clustering.

[McCallum, Nigam and Ungar: "Efficient Clustering of High Dimensional Data Sets with Application to Reference Matching", SIGKDD 2000]

Clustering

- k-means
  - a shared file contains centroid points
  - `map`
    1. for each point, find the nearest center
    2. generate `<key,value>` pair
       - `key`: center id
       - `value`: current point's coordinate
  - `reduce`
    1. collect all points belonging to the same cluster (they have the same `key` value)
    2. calculate the average → new center
  - iterate

Probabilistic relevance feedback

- Rather than re-weighting in a vector space…
- If user has told us some relevant and some irrelevant documents, then we can proceed to build a probabilistic classifier, such as the Naive Bayes model:
  - $P(t_k|R) = |D_{rk}| / |D_r|$
  - $P(t_k|NR) = |D_{nrk}| / |D_{nr}|$
    - t_k is a term; D_r is the set of known relevant documents; D_{rk} is the subset that contain t_k; D_{nr} is the set of known irrelevant documents; D_{nrk} is the subset that contain t_k.

Recall a few probability basics

- For events *a* and *b:*
- Bayes' Rule

$$p(a,b) = p(a \cap b) = p(a \mid b)\,p(b) = p(b \mid a)\,p(a)$$

$$p(\bar{a} \mid b)\,p(b) = p(b \mid \bar{a})\,p(\bar{a})$$

$$p(a \mid b) = \frac{p(b \mid a)\,p(a)}{p(b)} = \frac{p(b \mid a)\,p(a)}{\sum_{x=a,\bar{a}} p(b \mid x)\,p(x)}$$

Prior

Posterior

- Odds:

$$O(a) = \frac{p(a)}{p(\bar{a})} = \frac{p(a)}{1 - p(a)}$$

Bayesian Methods

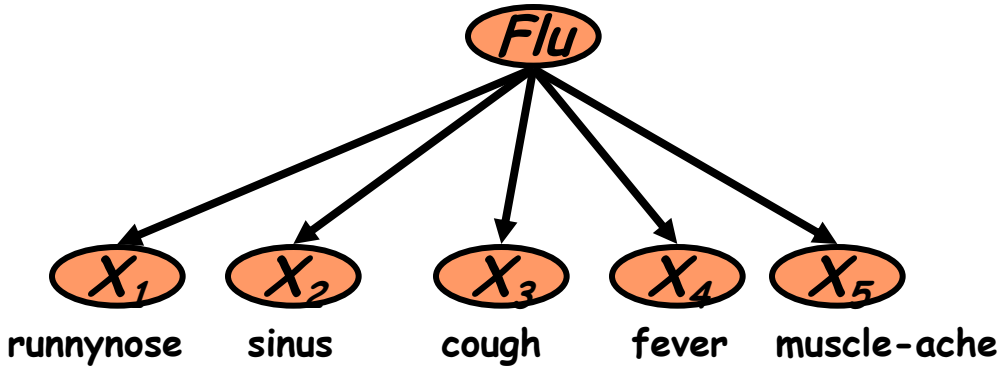
- Learning and classification methods based on probability theory.
  - Bayes theorem plays a critical role in probabilistic learning and classification.
- Build a *generative model* that approximates how data is produced.
- Uses *prior* probability of each category given no information about an item.
- Categorization produces a *posterior* probability distribution over the possible categories given a description of an item (and prior probabilities).

Bayes' Rule

$$P(C,D) = P(C \mid D)P(D) = P(D \mid C)P(C)$$

$$P(C \mid D) = \frac{P(D \mid C)P(C)}{P(D)}$$

The Naïve Bayes Classifier



- **Conditional Independence Assumption:** Features (term presence) are *independent* of each other given the class:

$$P(X_1,\ldots,X_5 \mid C) = P(X_1 \mid C) \bullet P(X_2 \mid C) \bullet \cdots \bullet P(X_5 \mid C)$$

- This model is appropriate for binary variables
  - Multivariate Bernoulli model

Naive Bayes Classifiers

Task: Classify a new instance *D* based on a tuple of attribute values $D = \langle x_1, x_2, \ldots, x_n \rangle$ into one of the classes $c_j \in C$

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

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

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

MAP = Maximum Aposteriori Probability

Naïve Bayes Classifier: Naïve Bayes Assumption

- $P(c_j)$
  - Can be estimated from the frequency of classes in the training examples.
- $P(x_1,x_2,\ldots,x_n|c_j)$
  - $O(|X|^n \bullet |C|)$ parameters
  - Could only be estimated if a very, very large number of training examples was available.

Naïve Bayes Conditional Independence Assumption:

- Assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(x_i|c_j)$.

Example

- Example: Play Tennis

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Example

- Learning Phase

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=No
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

$P(\text{Play}=Yes) = 9/14$ $P(\text{Play}=No) = 5/14$

Example

- Test Phase
  - Given a new instance, predict its label

    $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
  - Look up tables achieved in the learning phrase

P(Outlook=*Sunny*|Play=*Yes*) = 2/9
P(Temperature=*Cool*|Play=*Yes*) = 3/9
P(Huminity=*High*|Play=*Yes*) = 3/9
P(Wind=*Strong*|Play=*Yes*) = 3/9
P(Play=*Yes*) = 9/14

P(Outlook=S*unny*|Play=*No*) = 3/5
P(Temperature=*Cool*|Play==*No*) = 1/5
P(Huminity=*High*|Play=*No*) = 4/5
P(Wind=*Strong*|Play=*No*) = 3/5
P(Play=*No*) = 5/14

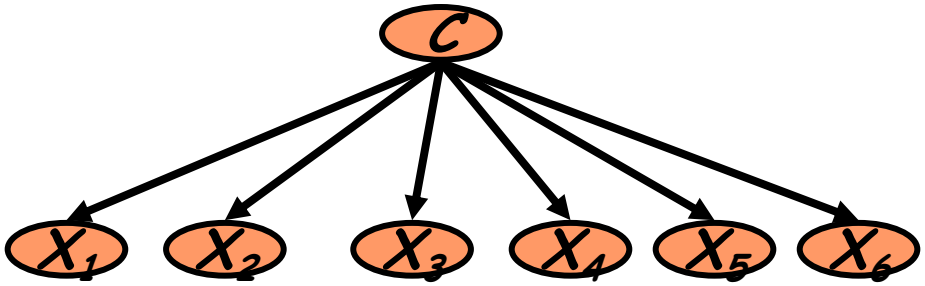
  - Decision making with the MAP rule

P(*Yes*|$\mathbf{x}'$): [P(*Sunny*|Yes)P(*Cool*|*Yes*)P(*High*|Yes)P(*Strong*|*Yes*)]P(Play=*Yes*) = 0.0053

P(*No*|$\mathbf{x}'$): [P(*Sunny*|No) P(*Cool*|No)P(*High*|*No*)P(*Strong*|*No*)]P(Play=*No*) = 0.0206

Given the fact P(*Yes*|$\mathbf{x}'$) < P(*No*|$\mathbf{x}'$), we label $\mathbf{x}'$ to be "*No*".

Learning the Model

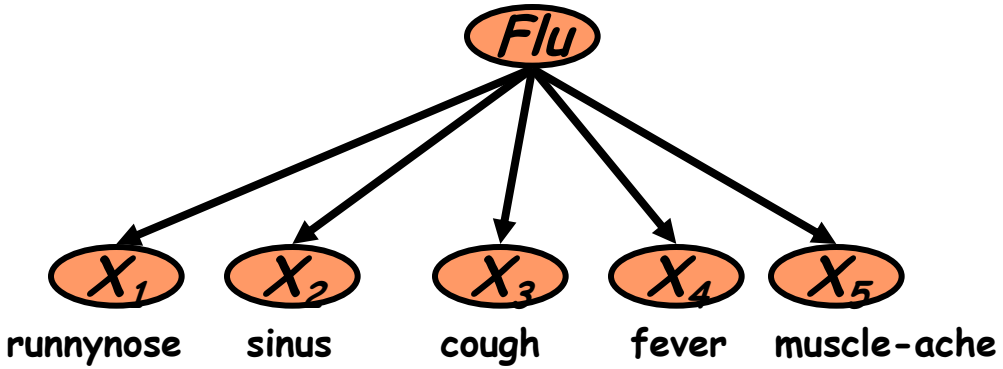


- *First attempt*: maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{N(C = c_j)}{N}$$

$$\hat{P}(x_i \mid c_j) = \frac{N(X_i = x_i, C = c_j)}{N(C = c_j)}$$

Problem with Max Likelihood



$$P(X_1,\ldots,X_5 \mid C) = P(X_1 \mid C) \bullet P(X_2 \mid C) \bullet \cdots \bullet P(X_5 \mid C)$$

- What if we have seen no training cases where patient had no flu and muscle aches?

$$\hat{P}(X_5 = t \mid C = nf) = \frac{N(X_5 = t, C = nf)}{N(C = nf)} = 0$$

- Zero probabilities cannot be conditioned away, no matter the other evidence!

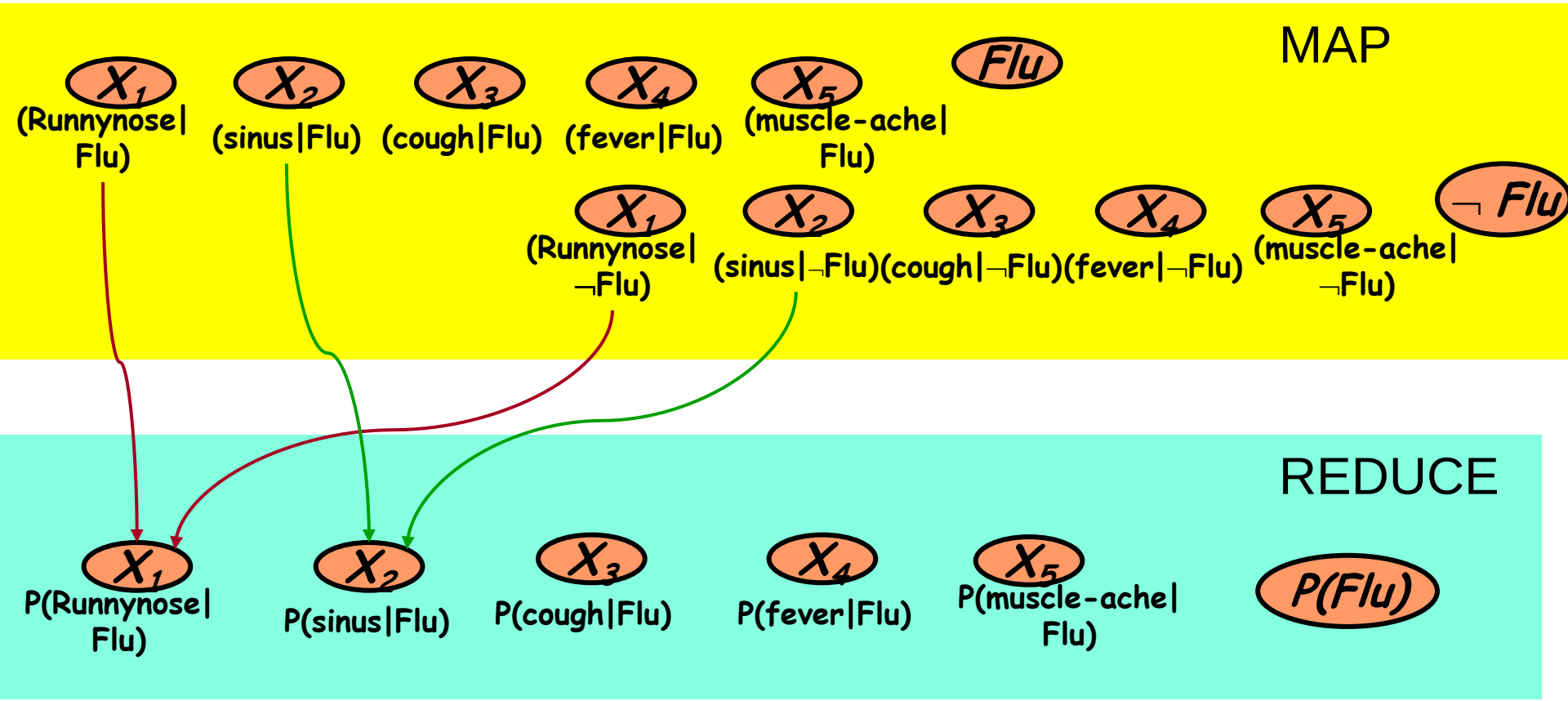
$$\ell = \arg\max_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

MapReduce Approach:

Simple Method

I_1: (Runnynose,sinus,cough,fever,muscle-ache,Flu)

I_m: (¬Runnynose,sinus,cough, ¬fever, ¬muscle-ache, ¬Flu)



Machine Learning Algorithm Transformation

- Naïve Bayes Classification (more complex)

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid a_1 \ldots a_n)$$

$$v_{MAP} = \arg\max_{v_j \in V} [P(a_1 \ldots a_n \mid v_j) P(v_j) \mid P(a_1 \ldots a_n)]$$

$$v_{MAP} = \arg\max_{v_j \in V} [P(a_1 \ldots a_n \mid v_j) P(v_j)]$$

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod P(a_i \mid v_j)$$

reduce

map

Machine Learning Algorithm Transformation to MapReduce

- Solution
  - Find statistics calculation part
  - Distribute calculations on data using `map`
  - Gather and refine all statistics in `reduce`

Exercise

	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	?

- **Estimate parameters of Naive Bayes classifier**
- **Classify test document**

Example: Parameter estimates

Priors: $\hat{P}(c) = 3/4$ and $\hat{P}(\overline{c}) = 1/4$ Conditional probabilities:

$$\begin{aligned}
\hat{P}(\textsc{Chinese}|c) &= (5+1)/(8+6) = 6/14 = 3/7 \\
\hat{P}(\textsc{Tokyo}|c) = \hat{P}(\textsc{Japan}|c) &= (0+1)/(8+6) = 1/14 \\
\hat{P}(\textsc{Chinese}|\overline{c}) &= (1+1)/(3+6) = 2/9 \\
\hat{P}(\textsc{Tokyo}|\overline{c}) = \hat{P}(\textsc{Japan}|\overline{c}) &= (1+1)/(3+6) = 2/9
\end{aligned}$$

The denominators are (8 + 6) and (3 + 6) because the lengths of $text_c$ and $text_{\overline{c}}$ are 8 and 3, respectively, and because the constant

***B* is 6 as the vocabulary consists of six terms.**

Example: Classification

$$\hat{P}(c|d_5) \propto 3/4 \cdot (3/7)^3 \cdot 1/14 \cdot 1/14 \approx 0.0003$$
$$\hat{P}(\overline{c}|d_5) \propto 1/4 \cdot (2/9)^3 \cdot 2/9 \cdot 2/9 \approx 0.0001$$

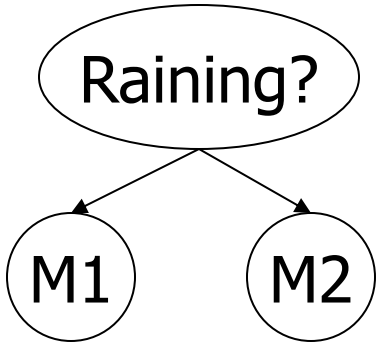
Thus, the classifier assigns the test document to c = *China*. The reason for this classification decision is that the three occurrences of the positive indicator `CHINESE` **in d_5 outweigh the occurrences of the two negative indicators** `JAPAN` **and** `TOKYO`**.**

Example: Sensors

Reality

P(+,+,r) = 3/8 P(-,-,r) = 1/8 P(+,+,s) = 1/8 P(-,-,s) = 3/8

NB Model



NB FACTORS:

- P(s) = 1/2
- P(+|s) = 1/4
- P(+|r) = 3/4

PREDICTIONS:

- P(r,+,+) = (½)(¾)(¾)
- P(s,+,+) = (½)(¼)(¼)
- P(r|+,+) = 9/10
- P(s|+,+) = 1/10

Smoothing to Avoid Overfitting

$$\hat{P}(x_i \mid c_j) = \frac{N(X_i = x_i, C = c_j) + 1}{N(C = c_j) + k}$$

# of values of X_i

Smoothing to Avoid Overfitting

$$\hat{P}(x_i \mid c_j) = \frac{N(X_i = x_i, C = c_j) + 1}{N(C = c_j) + k}$$

# of values of X_i

- Somewhat more subtle version

overall fraction in data where $X_i = x_{i,k}$

$$\hat{P}(x_{i,k} \mid c_j) = \frac{N(X_i = x_{i,k}, C = c_j) + mp_{i,k}}{N(C = c_j) + m}$$

extent of "smoothing"

Using Multinomial Naive Bayes Classifiers to Classify Text: Basic method

- Attributes are text positions, values are words.

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

$$= \underset{c_j \in C}{\operatorname{argmax}}\ P(c_j)P(x_1 = \text{"our"} \mid c_j)\cdots P(x_n = \text{"text"} \mid c_j)$$

- Still too many possibilities
- Assume that classification is *independent* of the positions of the words
  - Use same parameters for each position
  - Result is bag of words model (over tokens not types)

Naïve Bayes: Learning Algorithm

- From training corpus, extract *Vocabulary*
- Calculate required $P(c_j)$ and $P(x_k \mid c_j)$ terms
  - For each c_j in C do
    - $docs_j \leftarrow$ subset of documents for which the target class is c_j
    - $$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$
    - $Text_j \leftarrow$ single document containing all $docs_j$
    - for each word x_k in *Vocabulary*
      - $n_k \leftarrow$ number of occurrences of x_k in $Text_j$
      - $$P(x_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Naïve Bayes: Classifying

- positions ← all word positions in current document which contain tokens found in *Vocabulary*
- Return c_{NB}, where

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

Naive Bayes: Time Complexity

- Training Time: $O(|D|L_d + |C||V|))$
  where L_d is the average length of a document in D.
  - Assumes V and all D_i, n_i, and n_{ij} pre-computed in $O(|D|L_d)$ time during one pass through all of the data.
  - Generally just $O(|D|L_d)$ since usually $|C||V| < |D|L_d$

Why?

- Test Time: $O(|C| L_t)$
  where L_t is the average length of a test document.
  - Very efficient overall, linearly proportional to the time needed to just read in all the data.
  - Plus, robust in practice

Underflow Prevention: log space

- Multiplying lots of probabilities, which are between 0 and 1, can result in floating-point underflow.
- Since log(*xy*) = log(*x*) + log(*y*), it is better to perform all computations *by summing logs of probabilities rather than multiplying probabilities*.
- Class with highest final un-normalized log probability score is still the most probable.

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

- Note that model is now just max of sum of weights…

Note: Two Models

- *Model 1*: Multivariate Bernoulli
  - One feature X_w for each word in dictionary
  - X_w = true in document d if w appears in d
  - Naive Bayes assumption:
    - Given the document's topic, appearance of one word in the document tells us nothing about chances that another word appears
- This is the model used in the binary independence model in classic probabilistic relevance feedback in hand-classified data

Two Models

- *Model 2*: Multinomial = Class conditional unigram
  - One feature X_i for each word pos in document
    - feature's values are all words in dictionary
  - Value of X_i is the word in position i
  - Naïve Bayes assumption:
    - Given the document's topic, word in one position in the document tells us nothing about words in other positions
  - Second assumption:
    - Word appearance does not depend on position

$$P(X_i = w \mid c) = P(X_j = w \mid c)$$

for all positions *i,j*, word *w*, and class *c*

Prasad

Parameter estimation

- Multivariate Bernoulli model:

$$\hat{P}(X_w = t \mid c_j) = \text{fraction of documents of topic } c_j \text{ in which word } w \text{ appears}$$

- Multinomial model:

$$\hat{P}(X_i = w \mid c_j) = \frac{\text{fraction of times in which word } w \text{ appears across all documents of topic } c_j}{}$$

  - Can create a mega-document for topic j by concatenating all documents on this topic
  - Use frequency of w in mega-document

Classification

- Multinomial vs Multivariate Bernoulli?
- Multinomial model is almost always more effective in text applications!
- See *IIR* sections 13.2 and 13.3 for worked examples with each model

Feature Selection: Why?

- Text collections have a large number of features
  - 10,000 – 1,000,000 unique words … and more
- Feature Selection
  - Makes using a particular classifier feasible
    - Some classifiers can't deal with 100,000 of features
  - Reduces training time
    - Training time for some methods is quadratic or worse in the number of features
  - Can improve generalization (performance)
    - Eliminates noise features
    - Avoids overfitting

Feature selection: how?

- Two ideas:
  - Hypothesis testing statistics:
    - Are we confident that the value of one categorical variable is associated with the value of another
    - Chi-square test (χ^2)
  - Information theory:
    - How much information does the value of one categorical variable give you about the value of another
    - Mutual information (MI)
- They're similar, but χ^2 measures confidence in association, (based on available statistics), while MI measures extent of association (assuming perfect knowledge of probabilities)

χ^2 statistic (CHI)

- $\chi 2$ is interested in $(f_o - f_e)^2/f_e$ summed over all table entries: is the observed number what you'd expect given the marginals?

- The null hypothesis is rejected with confidence .999,
- since 12.9 > 10.83 (the value for .999 confidence).

	Term = jaguar	*Term ≠ jaguar*
Class = auto	2 *(0.25)*	500 *(502)*
Class ≠ auto	3 *(4.75)*	9500 *(9498)*

expected: f_e

observed: f_o

χ^2 statistic (CHI)

There is a simpler formula for 2x2 χ^2:

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

A = #(t,c)	C = #(¬t,c)
B = #(t,¬c)	D = #(¬t, ¬c)

Yields 1.29?

$$N = A + B + C + D$$

Value for complete independence of term and category?

Feature selection via Mutual Information

- In training set, choose *k* words which best discriminate (give most info. on) the categories.
- The Mutual Information between a word, class is:

$$I(w,c) = \sum_{e_w \in \{0,1\}} \sum_{e_c \in \{0,1\}} p(e_w, e_c) \log \frac{p(e_w, e_c)}{p(e_w)p(e_c)}$$

  - For each word *w* and each category *c*

Feature selection via MI (contd.)

- For each category we build a list of *k* most discriminating terms.
- For example (on 20 Newsgroups):
  - ***sci.electronics:*** circuit, voltage, amp, ground, copy, battery, electronics, cooling, …
  - ***rec.autos:*** car, cars, engine, ford, dealer, mustang, oil, collision, autos, tires, toyota, …
- *Greedy*: does not account for correlations between terms

Feature Selection

- Mutual Information
  - Clear information-theoretic interpretation
  - May select rare uninformative terms
- Chi-square
  - Statistical foundation
  - May select very slightly informative frequent terms that are not very useful for classification
- Just use the commonest terms?
  - No particular foundation
  - In practice, this is often 90% as good

Feature selection for NB

- In general, feature selection is *necessary* for multivariate Bernoulli NB.
- Otherwise, you suffer from noise, multi-counting

- "Feature selection" really means something different for multinomial NB. It means dictionary truncation.
  - This "feature selection" normally isn't needed for multinomial NB, but may help a fraction with quantities that are badly estimated

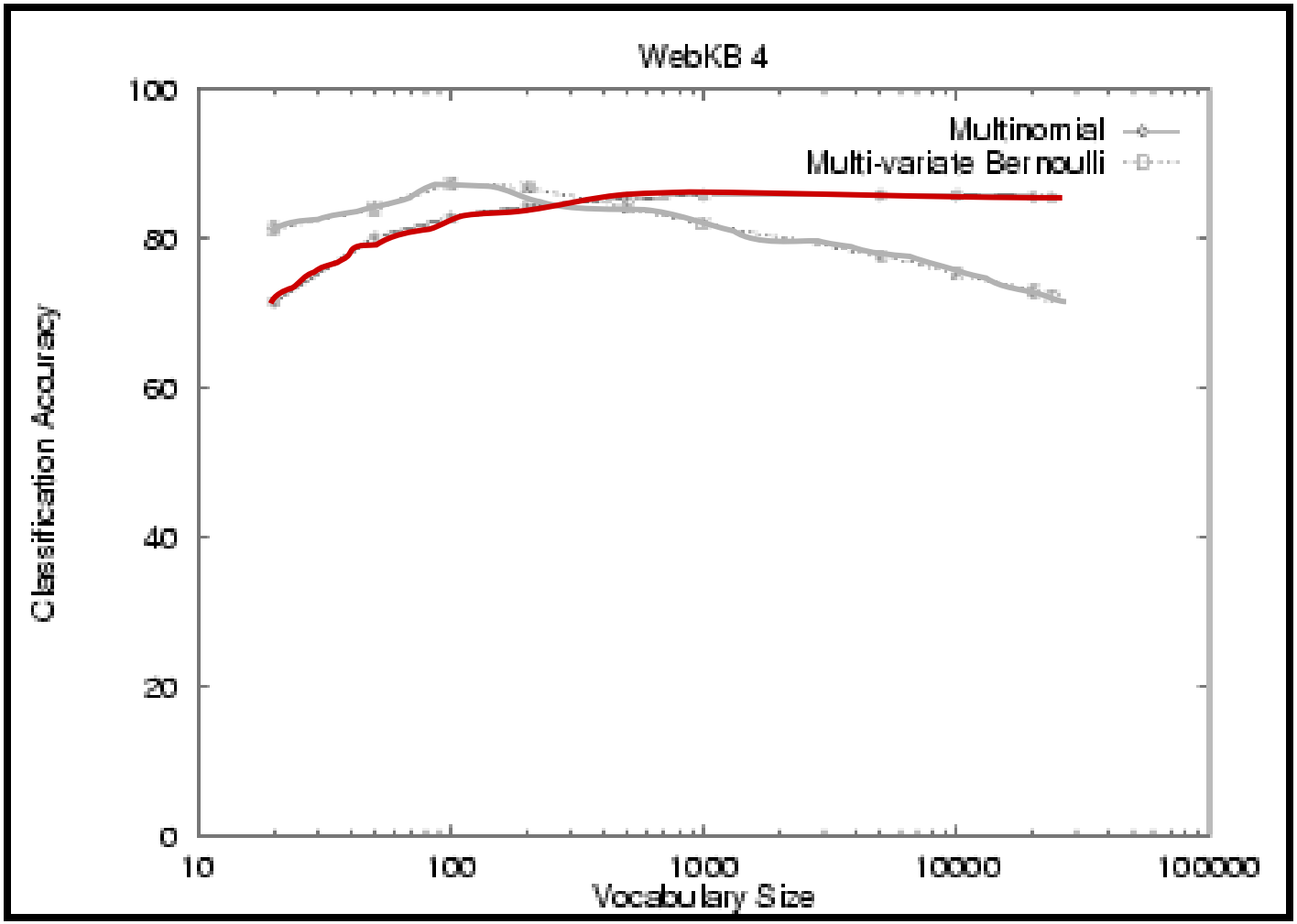
WebKB Experiment (1998)

- Classify webpages from CS departments into:
  - student, faculty, course,project
- Train on ~5,000 hand-labeled web pages
  - Cornell, Washington, U.Texas, Wisconsin
- Crawl and classify a new site (CMU)

- Results:

	Student	Faculty	Person	Project	Course	Departmt
Extracted	180	66	246	99	28	1
Correct	130	28	194	72	25	1
Accuracy:	72%	42%	79%	73%	89%	100%

NB Model Comparison: WebKB



Prasad

Faculty

associate	0.00417
chair	0.00303
member	0.00288
ph	0.00287
director	0.00282
fax	0.00279
journal	0.00271
recent	0.00260
received	0.00258
award	0.00250

Students

resume	0.00516
advisor	0.00456
student	0.00387
working	0.00361
stuff	0.00359
links	0.00355
homepage	0.00345
interests	0.00332
personal	0.00332
favorite	0.00310

Courses

homework	0.00413
syllabus	0.00399
assignments	0.00388
exam	0.00385
grading	0.00381
midterm	0.00374
pm	0.00371
instructor	0.00370
due	0.00364
final	0.00355

Departments

departmental	0.01246
colloquia	0.01076
epartment	0.01045
seminars	0.00997
schedules	0.00879
webmaster	0.00879
events	0.00826
facilities	0.00807
eople	0.00772
postgraduate	0.00764

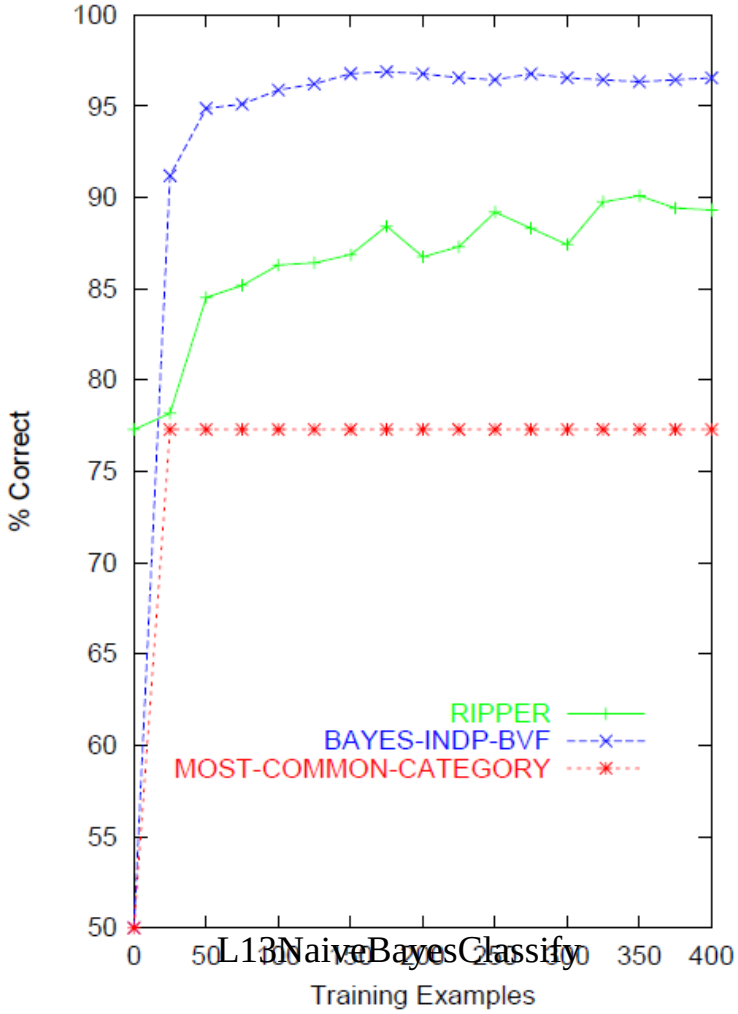
Research Projects

investigators	0.00256
group	0.00250
members	0.00242
researchers	0.00241
laboratory	0.00238
develop	0.00201
related	0.00200
arpa	0.00187
affiliated	0.00184
project	0.00183

Others

type	0.00164
jan	0.00148
enter	0.00145
random	0.00142
program	0.00136
net	0.00128
time	0.00128
format	0.00124
access	0.00117
begin	0.00116

Naïve Bayes on spam email



Violation of NB Assumptions

- Conditional independence
- “Positional independence”
- Examples?

Naive Bayes is Not So Naive

- Naïve Bayes: First and Second place in KDD-CUP 97 competition, among 16 (then) state of the art algorithms

  Goal: Financial services industry direct mail response prediction model: Predict if the recipient of mail will actually respond to the advertisement – 750,000 records.

- Robust to Irrelevant Features

  Irrelevant Features cancel each other without affecting results

  Instead Decision Trees can heavily suffer from this.

- Very good in domains with many equally important features

  Decision Trees suffer from *fragmentation* in such cases – especially if little data

- A good dependable baseline for text classification (but not the best)!
- Optimal if the Independence Assumptions hold: If assumed independence is correct, then it is the Bayes Optimal Classifier for problem
- Very Fast: Learning with one pass of counting over the data; testing linear in the number of attributes, and document collection size
- Low Storage requirements