UVA CS 6316 : Machine Learning

Lecture 17c: Naïve Bayes Classifier for Text Classification

Dr. Yanjun Qi

University of Virginia
Department of Computer Science

Where are we ? → Three major sections for classification

- We can divide the large variety of classification approaches into roughly three major types

1. ~~Discriminative~~
   ~~- directly estimate a decision rule/boundary~~
   ~~- e.g., SVM, NN~~, decision tree

2. Generative:
   - build a generative statistical model
   - e.g., naïve bayes classifier, Bayesian networks

3. ~~Instance based classifiers~~
   ~~- Use observation directly (no models)~~
   ~~- e.g. K nearest neighbors~~

Review: Naïve Bayes Classifier

$= p(x_1|c_j)\, p(x_2|c_j) \ldots p(x_p|c_j)$

- Bayes classification

$$\underset{c_j \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_p | c_j) P(c_j)$$

Difficulty: learning the joint probability

- Naïve Bayes classification
  - Assumption that all input attributes are conditionally independent!

given C variable

Today : Naïve Bayes Classifier for Text

- ✓ Dictionary based Vector space representation of text article
- ✓ Multivariate Bernoulli vs. Multinomial
- ✓ Multivariate Bernoulli naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
- ✓ Multinomial naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models (Extra)

Text document classification, e.g. spam email filtering

- Input: document D
- Output: the predicted class C, c is from $\{c_1,\ldots,c_L\}$

- E.g.,
  - Spam filtering Task: Classify email as *'Spam'*, *'Other'*.

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 !

Change your life NOW by taking a simple course!
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm

→ P (C=spam | D)

Naive Bayes is Not So Naive

- Naive Bayes won 1st and 2nd place in KDD-CUP 97 competition out of 16 systems

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

- A good dependable baseline for text classification (but not the best)!
- For most text categorization tasks, there are many relevant features and many irrelevant ones

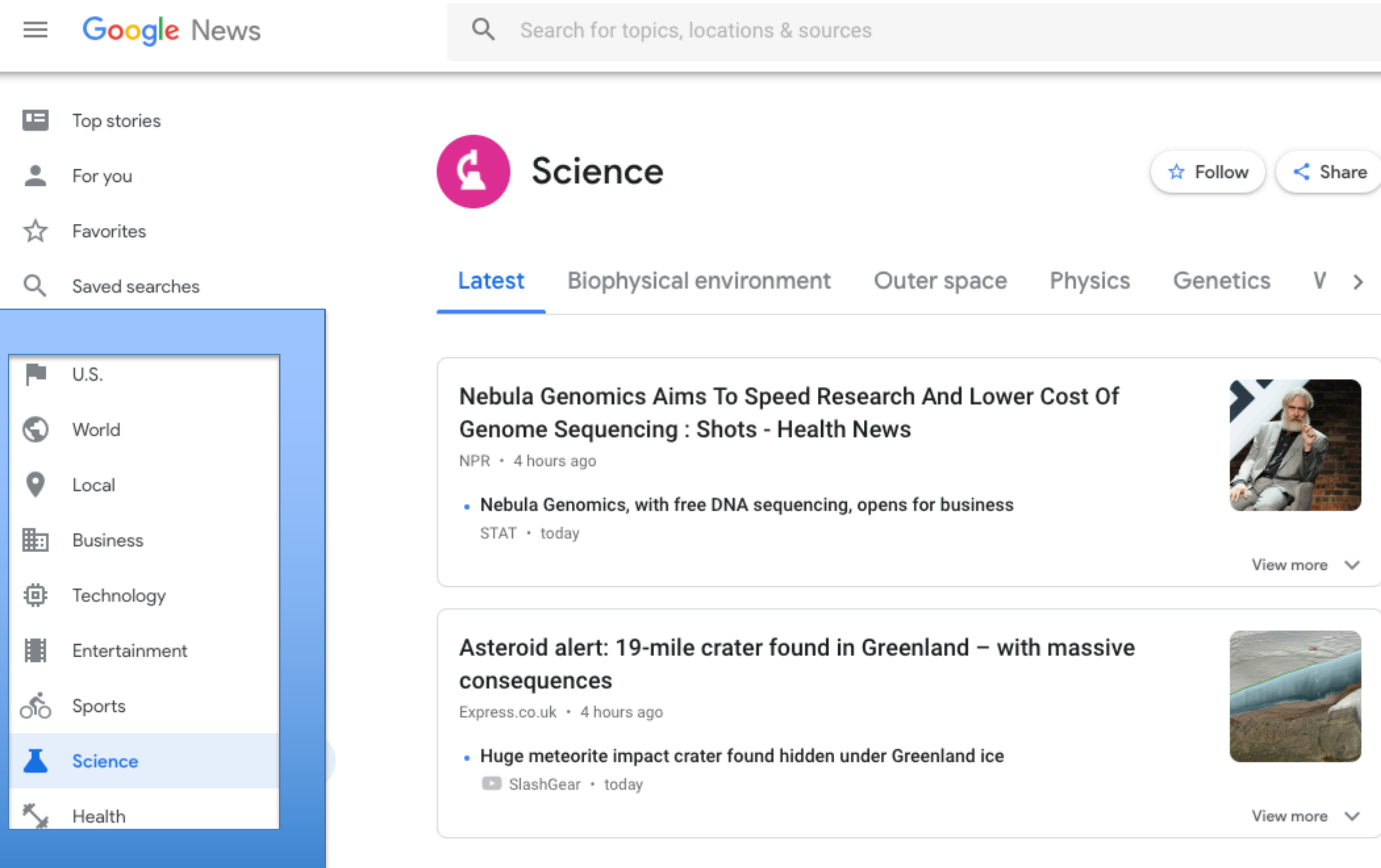
Text classification Tasks

- Input: document D
- Output: the predicted class C, c is from {c_1,...,c_L }

Text classification examples:

- Classify email as *'Spam', 'Other'.*
- Classify web pages as *'Student', 'Faculty', 'Other'*
- Classify news stories into topics *'Sports', 'Politics'..* → Google News
- Classify business names by industry.
- Classify movie reviews as *'Favorable', 'Unfavorable', 'Neutral'*
- *... and many more.*

Netflix


Google News
Search for topics, locations & sources
Top stories
For you
Favorites
Saved searches
U.S.
World
Local
Business
Technology
Entertainment
Sports
Science
Health
Science
Follow
Share
Latest
Biophysical environment
Outer space
Physics
Genetics
Nebula Genomics Aims To Speed Research And Lower Cost Of Genome Sequencing : Shots - Health News
NPR • 4 hours ago
Nebula Genomics, with free DNA sequencing, opens for business
STAT • today
View more
Asteroid alert: 19-mile crater found in Greenland – with massive consequences
Express.co.uk • 4 hours ago
Huge meteorite impact crater found hidden under Greenland ice
SlashGear • today
View more

Text Categorization/Classification

- Given:
  - A representation of a text document *d*
    - Issue: how to represent text documents.
    - Usually some type of high-dimensional space – bag of words
  - A fixed set of output classes:

    $C = \{c_1, c_2, \ldots, c_J\}$

The bag of words representation

f(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

$$f\left(\begin{array}{|l|l|}\hline \texttt{great} & \texttt{2} \\ \hline \texttt{love} & \texttt{2} \\ \hline \texttt{recommend} & \texttt{1} \\ \hline \texttt{laugh} & \texttt{1} \\ \hline \texttt{happy} & \texttt{1} \\ \hline \texttt{...} & \texttt{...} \\ \hline \end{array}\right) = c$$

Representing text: a list of words → Dictionary

D

> 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.

word	*frequency*
great	2
love	2
recommend	1
laugh	1
happy	1
. . .	.

1 2 3

Common refinements: [remove stopwords], [stemming], [collapsing multiple occurrences of words into one….]

→ [NLTK]

love
loves
loving
→ love

'Bag of words' representation of text

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.

word	*frequency*
great	2
love	2
recommend	1
laugh	1
happy	1
...	.

Bag of word representation:

Represent text as a vector of word *frequencies*.

$$D = (W_1, W_2, \cdots, W_K)$$

Another "Bag of words" representation of text ➔ Each dictionary word as Boolean

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.

word	*Boolean*
great	Yes
love	Yes
recommend	Yes
laugh	Yes
happy	Yes
hate	No
...	.

Bag of word representation:

Represent text as a vector of Boolean representing if a word *Exists or NOT.*

$D = (W_1, W_2, \cdots, W_K)$

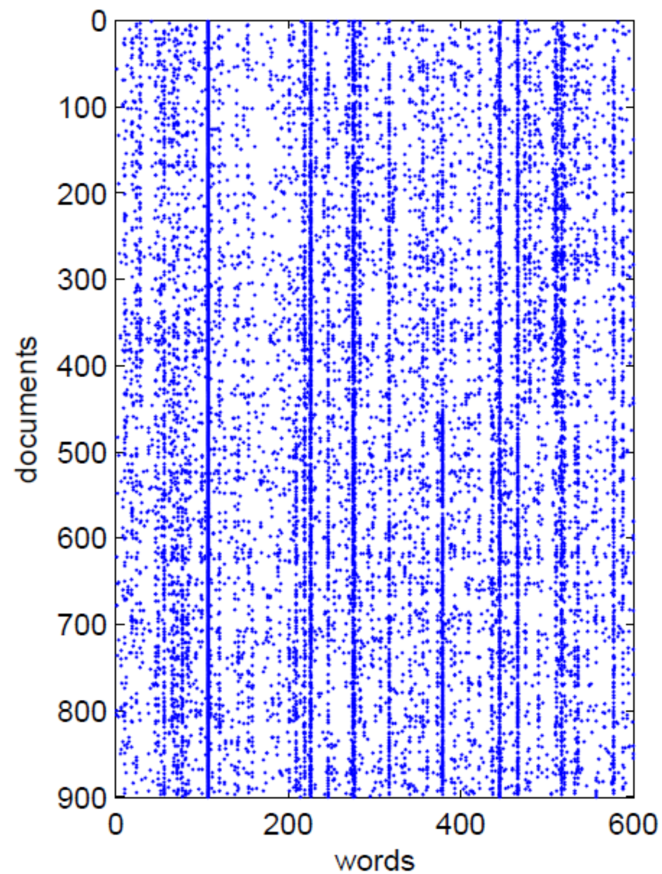
Bag of words

- What simplifying assumption are we taking?

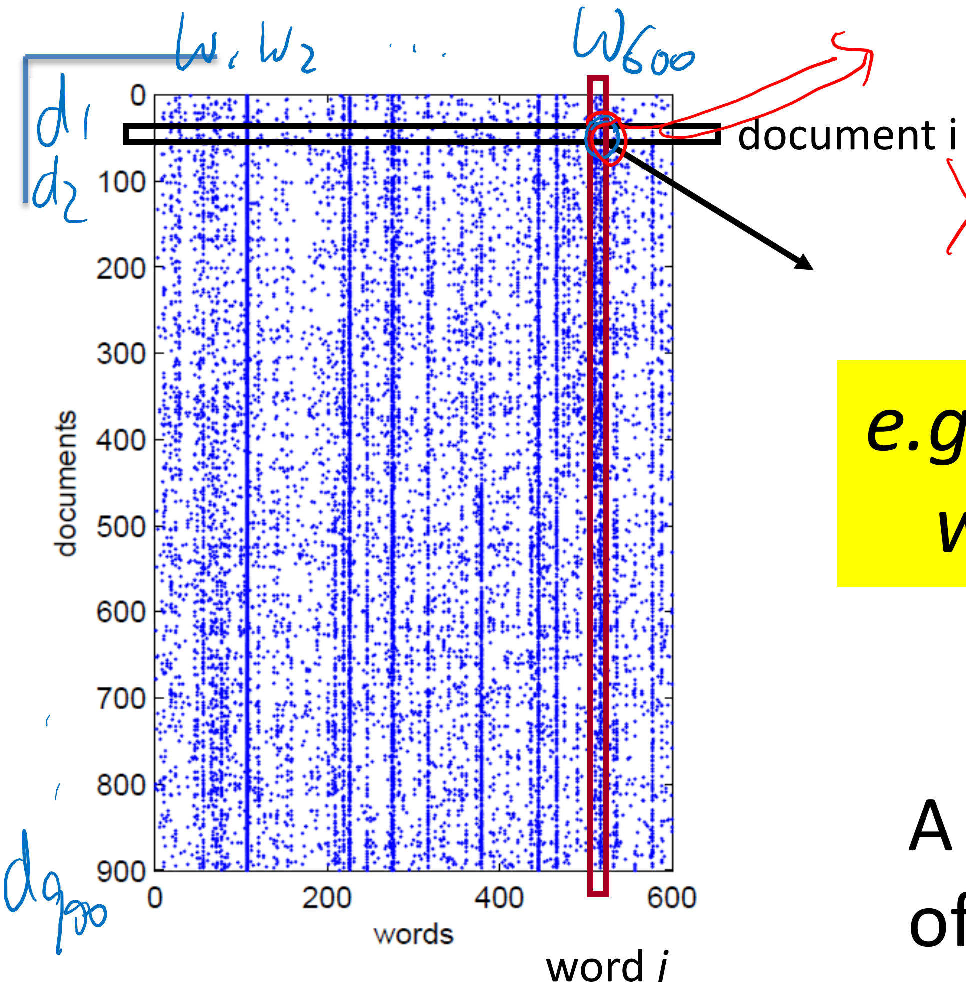
We assumed *word order* is not important.

$$D = (w_1, w_2, \cdots, w_K)$$

many other choices, e.g. BM25 / TF-IDF / ..



Bag of words representation



$X(d_i, w_j)$

e.g., X (i,j) = Frequency of word j* in document *i

A collection of documents

	X_1	X_2	X_3	C
S_1				
S_2				
S_3				
S_4				
S_5				
S_6				

Unknown Words

- How to handle words in the test corpus that did not occur in the training data, i.e. ***out of vocabulary*** (OOV) words?
- Train a model that includes an explicit symbol for an unknown word (<UNK>).
  - Choose a vocabulary in advance and replace other (i.e. not in vocabulary) words in the corpus with <UNK>.
  - Very often, <UNK> also used to replace rare words

Today : Naïve Bayes Classifier for Text

- ✓ Dictionary based Vector space representation of text article
- ✓ Multivariate Bernoulli vs. Multinomial
- ✓ Multivariate Bernoulli naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
- ✓ Multinomial naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models (Extra)

‘Bag of words’ → what probability 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.

word	
great	.
love	.
recommend	.
laugh	.
happy	.
...	.

$$\underset{i \in \{1,2,..,L\}}{\arg\max}\, P(C_i) \boxed{P(d \mid C_i)}$$

$$\Pr(D = d \mid C = c_i)$$

?

'Bag of words' ➔ what probability model?

$$\Pr(D \mid C = c) = \ ?$$

Two Previous models

$$\Pr(W_1 = true, W_2 = false \ldots, W_k = true \mid C = c)$$

$$\Pr(W_1 = n_1, W_2 = n_2, \ldots, W_k = n_k \mid C = c)$$

$D = (W_1, W_2, \ldots, W_K)$

Naïve Probabilistic Models of text documents

$w_1 \ldots w_K \in$ Vocabulary

d_1
$\vdots$
d_n

$$\Pr(D \mid C = c) =$$

$D = (w_1, w_2, \ldots, w_K)$

Two Previous models

$$\Pr(W_1 = true, W_2 = false \ldots, W_k = true \mid C = c)$$

Multivariate Bernoulli Distribution

$$\Pr(W_1 = n_1, W_2 = n_2, \ldots, W_k = n_k \mid C = c)$$

Multinomial Distribution

Text Classification with Naïve Bayes Classifier

- Multinomial vs Multivariate Bernoulli?
- Multinomial model is almost always more effective in text applications!

Adapt From Manning' textCat tutorial

Experiment: Multinomial vs multivariate Bernoulli

- M&N (1998) did some experiments to see which is better
- Determine if a university web page is {student, faculty, other_stuff}
- Train on ~5,000 hand-labeled web pages
  - Cornell, Washington, U.Texas, Wisconsin

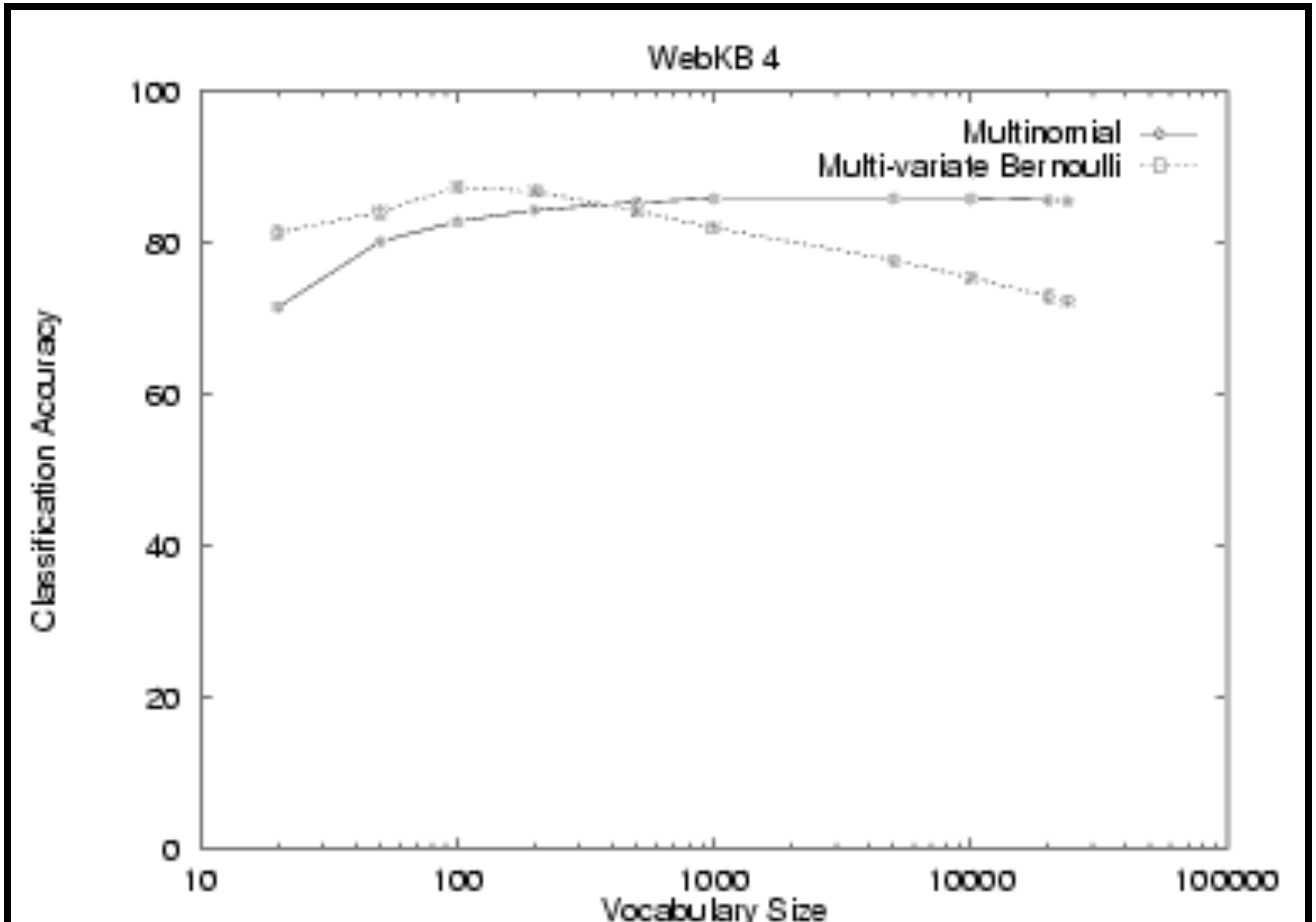
train

- Crawl and classify a new site (CMU)

test

Adapt From Manning' textCat tutorial

Multinomial vs. multivariate Bernoulli



Today : Naïve Bayes Classifier for Text

- ✓ Dictionary based Vector space representation of text article
- ✓ Multivariate Bernoulli vs. Multinomial
- ✓ Multivariate Bernoulli
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
- ✓ Multinomial naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models (Extra)

Model 1: Multivariate Bernoulli

- *Model 1*: Multivariate Bernoulli
  - For each word in a dictionary, feature X_w
  - X_w = true in document d if w appears in d

OR not ⇒ Binary

word	*Boolean*
great	Yes
love	Yes
recommend	Yes
laugh	Yes
happy	Yes
hate	No

```
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.
```

Model 1: Multivariate Bernoulli

- *Model 1*: Multivariate Bernoulli
  - One feature X_w for each word in dictionary
  - X_w = true in document *d* if [*w* appears in *d*]

    OR not ⇒ Binary

  - Naive Bayes assumption:
    - Given the document's class label, C_j
      appearance of one word in the document tells us nothing about chances that another word appears

$$\Pr(W_1 = true, W_2 = false \ldots, W_k = true \mid C = c)$$

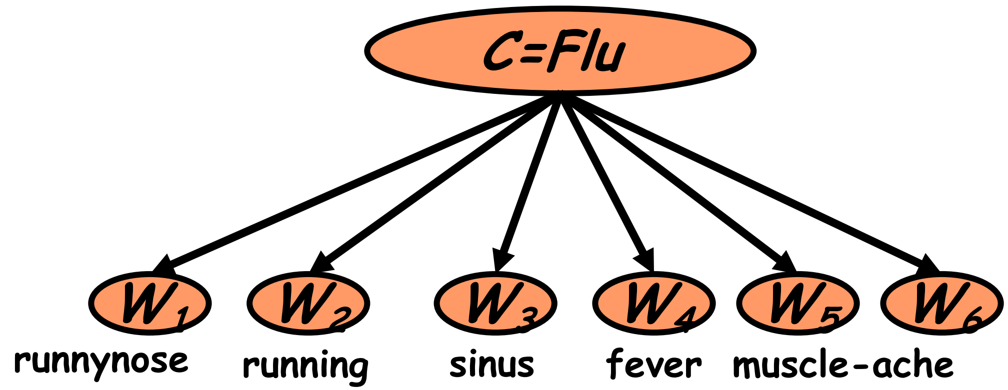
Model 1: Multivariate Bernoulli Naïve Bayes Classifier

$$P(W_1, W_2, \ldots, W_k | C) = P(W_1|C)\,P(W_2|C) \ldots P(W_k|C)$$

word	*True/false*
great	Yes
love	Yes
recommend	Yes
laugh	Yes
happy	Yes
hate	No
...	.

- **Conditional Independence Assumption:** Features (word presence) are *independent* of each other given the class variable:
- Multivariate Bernoulli model is appropriate for binary feature variables

Model 1: Multivariate Bernoulli



this is naïve

$$\Pr(W_1 = true, W_2 = false, \ldots, W_k = true \mid C = c)$$

$$= P(W_1 = true \mid C) \bullet P(W_2 = false \mid C) \bullet \cdots \bullet P(W_k = true \mid C)$$

Bernoulli parameter

Review: Bernoulli Distribution e.g. Coin Flips

- You flip a coin
  - Head with probability *p*
  - Binary random variable
  - Bernoulli trial with success probability *p*

$$\Pr(W_i = true \mid C = c_j) = p_{w_i, j}$$

p estimated from data

$$\begin{array}{c|c}
 & c_1 \quad c_2 \quad \cdots \quad c_L \\
\hline
\begin{array}{c} w_1 \\ w_2 \\ \vdots \\ w_K \end{array} & \left[\begin{array}{c} \\ p_{w_k, c_j} \\ \\ \end{array} \right]
\end{array}$$

p_{w_i, c_j}

Review: Deriving the Maximum Likelihood Estimate for Bernoulli

$$-l(p) = -\log(L(p)) = -\log\left[p^x(1-p)^{n-x}\right]$$

Minimize the negative log-likelihood

→ MLE parameter estimation

$$= -\log(p^x) - \log((1-p)^{n-x})$$

$$= -x\log(p) - (n-x)\log(1-p)$$

$$\hat{p} = \frac{x}{n}$$

i.e. Relative frequency of a binary event

Parameter estimation

- Multivariate Bernoulli model:

$$\hat{P}(w_i = true \mid c_j) = \text{fraction of documents of label } c_j \text{ in which word } w\_i \text{ appears}$$

- Smoothing to Avoid Overfitting

Adapt From Manning' textCat tutorial

Testing Stage: (Look Up Operations)

$$d_{ts} = \{ w_1 = true, \; w_2 = false, \; w_3 = true \}$$

$$p(d_{ts} | c_j) = p_{w1,j} (1 - p_{w2,j}) \, p_{w3,j} \, .$$

$$p(c_j)$$

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.

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

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

Adapt From Manning' textCat tutorial

Today : Naïve Bayes Classifier for Text

- ✓ Dictionary based Vector space representation of text article
- ✓ Multivariate Bernoulli vs. Multinomial
- ✓ Multivariate Bernoulli
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
- ✓ Multinomial naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models (Extra)

Model 2: Multinomial Naïve Bayes

- 'Bag of words' representation of text

word	*frequency*
great	2
love	2
recommend	1
laugh	1
happy	1
...	.

$$\Pr(W_1 = n_1, \ldots, W_k = n_k \mid C = c)$$

Can be represented as a multinomial distribution.

Model 2: Multinomial Naïve Bayes - 'Bag of words' representation of text

word	*frequency*
great	2
love	2
recommend	1
laugh	1
happy	1
...	.

$$\Pr(W_1 = n_1, \ldots, W_k = n_k \mid C = c)$$

Can be represented as a multinomial distribution.

Words = like colored balls, there are *K* possible type of them (i.e. from a dictionary of K words)

Model 2: Multinomial Naïve Bayes - 'Bag of words' representation of text

word	*frequency*
great	2
love	2
recommend	1
laugh	1
happy	1
...	.

$$\Pr(W_1 = n_1, \ldots, W_k = n_k \mid C = c)$$

Can be represented as a multinomial distribution.

Words = like colored balls, there are *K* possible type of them (i.e. from a dictionary of K words)

A Document = contains N words, each word occurs n_i times (like a bag of N colored balls)

Multinomial distribution

- The multinomial distribution is a generalization of the binomial distribution.

- The binomial distribution counts successes of an event (for example, heads in coin tosses).
- The parameters:
  - – N (number of trials)
  - – p (the probability of success of the event)

$K=2$

flip N times of the same coin ⇒

e.g. $N_{Head} + N_{Tail} = N$

$p_{head} = p$

$p_{tail} = 1 - p$

A binomial distribution is the multinomial distribution with k=2 and $\theta_1 = p$

$$\theta_2 = 1 - \theta_1$$

Multinomial distribution

- The multinomial distribution is a generalization of the binomial distribution.

- The binomial distribution counts successes of an event (for example, heads in coin tosses).

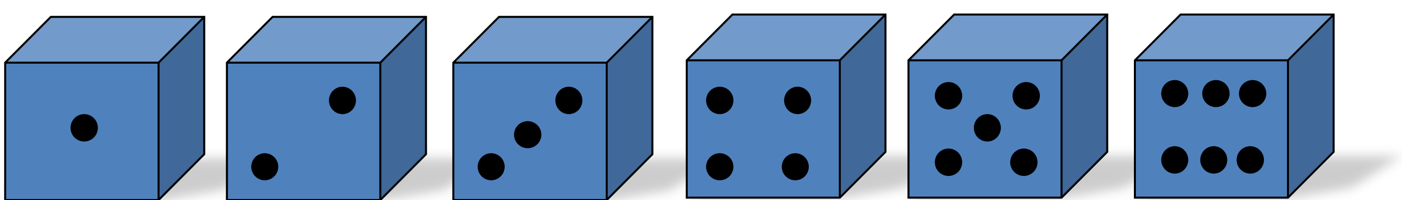
$K=2$

- The parameters:
  - N (number of trials)
  - p (the probability of success of the event)

- The multinomial counts the number of a set of events (for example, how many times each side of a die comes up in a set of rolls).
  - The parameters:
  - N (number of trials)
  - $\theta_1 .. \theta_k$ (the probability of success for each category)

$N_1 + N_2 + \dots + N_k = N$

$K=6$



Multinomial Distribution for Text Classification

- $W_1, W_2, .. W_k$ are variables

Number of possible orderings of N balls

$$P(W_1 = n_1, ..., W_k = n_k \mid c, N, \theta_{1,c}, .., \theta_{k,c}) = \frac{N!}{n_1! n_2! .. n_k!} \theta_{1,c}{}^{n_1} \theta_{2,c}{}^{n_2} .. \theta_{k,c}{}^{n_k}$$

Multinomial Distribution for Text Classification

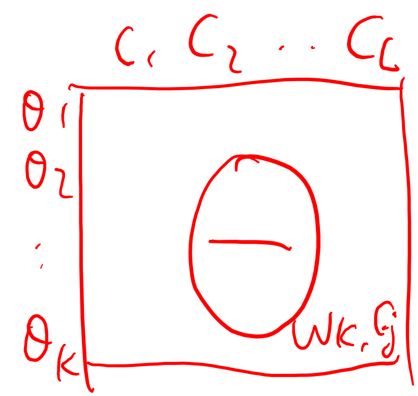
- $W_1, W_2, .. W_k$ are variables

Number of possible orderings of N balls

$$P(W_1 = n_1, \ldots, W_k = n_k \mid c, N, \theta_{1,c}, \ldots, \theta_{k,c}) = \cancel{\frac{N!}{n_1! n_2! .. n_k!}} \theta_{1,c}^{\ n_1} \theta_{2,c}^{\ n_2} .. \theta_{k,c}^{\ n_k}$$

Label invariant

$$\sum_{i=1}^{k} n_i = N \qquad \sum_{i=1}^{k} \theta_{i,c} = 1$$



Model 2: Multinomial Naïve Bayes - 'Bag of words' – TESTING Stage

word	*frequency*
great	2
love	2
recommend	1
laugh	1
happy	1
...	.

$$\arg\max_{c} P(W_1 = n_1, \ldots, W_k = n_k, c)$$

$$= \arg\max_{c} \{ p(c) * \theta_{1,c}^{\ n_1} \theta_{2,c}^{\ n_2} .. \theta_{k,c}^{\ n_k} \}$$

Today : Naïve Bayes Classifier for Text

- ✓ Dictionary based Vector space representation of text article
- ✓ Multivariate Bernoulli vs. Multinomial
- ✓ Multivariate Bernoulli
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
- ✓ Multinomial naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models (Extra)

estimate $\Theta_{K \times L}$ from training data

$$
\begin{array}{c|cccc}
 & C_1 & C_2 & \cdots & C_L \\
\hline
\theta_1 \to w_1 & & & & \\
\theta_2 \to w_2 & & & & \\
\vdots & & \Theta_{w_i, c_j} & & \\
\theta_K \to w_K & & & &
\end{array}
$$

Deriving the Maximum Likelihood Estimate for multinomial distribution

LIKELIHOOD:

$$\arg\max_{\theta_1,..,\theta_k} P(d_1,\ldots,d_T \mid \theta_1,..,\theta_k)$$

function of θ

θ vector

$$= \arg\max_{\theta_1,..,\theta_k} \prod_{t=1}^{T} P(d_t \mid \theta_1,..,\theta_k)$$

$$= \arg\max_{\theta_1,..,\theta_k} \prod_{t=1}^{T} \cancel{\frac{N_{d_t}!}{n_{1,d_t}!\,n_{2,d_t}!\ldots n_{k,d_t}!}} \theta_1^{n_{1,d_t}} \theta_2^{n_{2,d_t}} ..\theta_k^{n_{k,d_t}}$$

$$= \arg\max_{\theta_1,..,\theta_k} \prod_{t=1}^{T} \theta_1^{n_{1,d_t}} \theta_2^{n_{2,d_t}} ..\theta_k^{n_{k,d_t}}$$

$$s.t. \sum_{i=1}^{k} \theta_i = 1$$

Deriving the Maximum Likelihood Estimate for multinomial distribution

$$\arg\max_{\theta_1,..,\theta_k} \log(L(\theta))$$

Constrained optimization

$$s.t. \sum_{i=1}^{k} \theta_i = 1$$

$$= \arg\max_{\theta_1,..,\theta_k} \log(\prod_{t=1}^{T} \theta_1^{n_{1,d_t}} \theta_2^{n_{2,d_t}} .. \theta_k^{n_{k,d_t}})$$

Deriving the Maximum Likelihood Estimate for multinomial distribution

$$\arg\max_{\theta_1,\ldots,\theta_k} \log(L(\theta))$$

Constrained optimization

$$s.t. \sum_{i=1}^{k} \theta_i = 1$$

$$= \arg\max_{\theta_1,\ldots,\theta_k} \log\left(\prod_{t=1}^{T} \theta_1^{n_{1,d_t}} \theta_2^{n_{2,d_t}} .. \theta_k^{n_{k,d_t}}\right)$$

$$= \arg\max_{\theta_1,\ldots,\theta_k} \sum_{t=1,\ldots T} n_{1,d_t} \log(\theta_1) + \sum_{t=1,\ldots T} n_{2,d_t} \log(\theta_2) + \ldots + \sum_{t=1,\ldots T} n_{k,d_t} \log(\theta_k)$$

Deriving the Maximum Likelihood Estimate for multinomial distribution

$$\arg\max_{\theta_1,\ldots,\theta_k} \log(L(\theta))$$

Constrained optimization

$$s.t. \sum_{i=1}^{k} \theta_i = 1$$

$$= \arg\max_{\theta_1,\ldots,\theta_k} \log\left(\prod_{t=1}^{T} \theta_1^{n_{1,d_t}} \theta_2^{n_{2,d_t}} \ldots \theta_k^{n_{k,d_t}}\right)$$

$$= \arg\max_{\theta_1,\ldots,\theta_k} \sum_{t=1,\ldots T} n_{1,d_t} \log(\theta_1) + \sum_{t=1,\ldots T} n_{2,d_t} \log(\theta_2) + \ldots + \sum_{t=1,\ldots T} n_{k,d_t} \log(\theta_k)$$

Constrained optimization MLE estimator

$$\theta_i = \frac{\sum_{t=1,\ldots T} n_{i,d_t}}{\sum_{t=1,\ldots T} n_{1,d_t} + \sum_{t=1,\ldots T} n_{2,d_t} + \ldots + \sum_{t=1,\ldots T} n_{k,d_t}} = \frac{\sum_{t=1,\ldots T} n_{i,d_t}}{\sum_{t=1,\ldots T} N_{d_t}}$$

Deriving the Maximum Likelihood Estimate for multinomial distribution

$$\arg\max_{\theta_1,\ldots,\theta_k} \log(L(\theta))$$

Constrained optimization

$$s.t. \sum_{i=1}^{k} \theta_i = 1$$

$$= \arg\max_{\theta_1,\ldots,\theta_k} \log\left(\prod_{t=1}^{T} \theta_1^{n_{1,d_t}} \theta_2^{n_{2,d_t}} \ldots \theta_k^{n_{k,d_t}}\right)$$

$$= \arg\max_{\theta_1,\ldots,\theta_k} \sum_{t=1,\ldots T} n_{1,d_t} \log(\theta_1) + \sum_{t=1,\ldots T} n_{2,d_t} \log(\theta_2) + \ldots + \sum_{t=1,\ldots T} n_{k,d_t} \log(\theta_k)$$

Constrained optimization MLE estimator

$$\theta_i = \frac{\sum_{t=1,\ldots T} n_{i,d_t}}{\sum_{t=1,\ldots T} n_{1,d_t} + \sum_{t=1,\ldots T} n_{2,d_t} + \ldots + \sum_{t=1,\ldots T} n_{k,d_t}} = \frac{\sum_{t=1,\ldots T} n_{i,d_t}}{\sum_{t=1,\ldots T} N_{d_t}}$$

➔ i.e. We can create a mega-document by concatenating all documents d_1 to d_T

➔Use relative frequency of w_i in mega-document

Deriving the Maximum Likelihood Estimate for multinomial distribution

Constrained optimization MLE estimator

$$\theta_i = \frac{\sum_{t=1,\dots T} n_{i,d_t}}{\sum_{t=1,\dots T} n_{1,d_t} + \sum_{t=1,\dots T} n_{2,d_t} + \dots + \sum_{t=1,\dots T} n_{k,d_t}} = \frac{\sum_{t=1,\dots T} n_{i,d_t}}{\sum_{t=1,\dots T} N_{d_t}}$$

➔i.e. We can create a mega-document by concatenating all documents d_1 to d_T

➔Use relative frequency of *w* in mega-document

Deriving the Maximum Likelihood Estimate for multinomial Bayes Classifier

train T documents

LIKELIHOOD: $$\underset{\theta_1,\ldots,\theta_k}{\arg\max}\, P(d_1,\ldots,d_T \mid \theta_1,\ldots,\theta_k), (C=C_j)$$

$w_1, \ldots, w_K$

estimate $\Theta_{K \times L}$ from training data

$$\begin{array}{c|ccccc|}
 & C_1 & C_2 & \cdots & & C_L \\
\hline
w_1 & & & & & \\
w_2 & & & & & \\
\vdots & & & \Theta_{w_i, C_j} & & \\
w_K & & & & & \\
\hline
\end{array}$$

Parameter estimation

Multinomial model:

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

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

Adapt From Manning' textCat tutorial

Multinomial : Learning Algorithm for parameter estimation with MLE

- From training corpus, extract *Vocabulary*
- Calculate required $P(c_j)$ and $P(w_k \mid c_j)$ terms
  - For each c_j in C do
    - $docs_j$ ← subset of documents for which the target class is c_j

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

Multinomial : Learning Algorithm for parameter estimation with MLE

- From training corpus, extract *Vocabulary*
- Calculate required $P(c_j)$ and $P(w_k \mid c_j)$ terms
  - For each c_j in C do
    - $docs_j \leftarrow$ subset of documents for which the target class is c_j

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

- $Text_j \leftarrow$ is length n_j and is a single document containing all $docs_j$ for class c_j
- for each word w_k in *Vocabulary*
  - $n_{k,j} \leftarrow$ number of occurrences of w_k in $Text_j$; n_j *is length of* $Text_j$
  - $$P(w_k \mid c_j) \leftarrow \frac{n_{k,j} + \alpha}{n_j + \alpha |Vocabulary|} \qquad e.g., \alpha = 1$$ (Smoothing)

Relative frequency of word *w_k* appears across all documents of class c_j

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

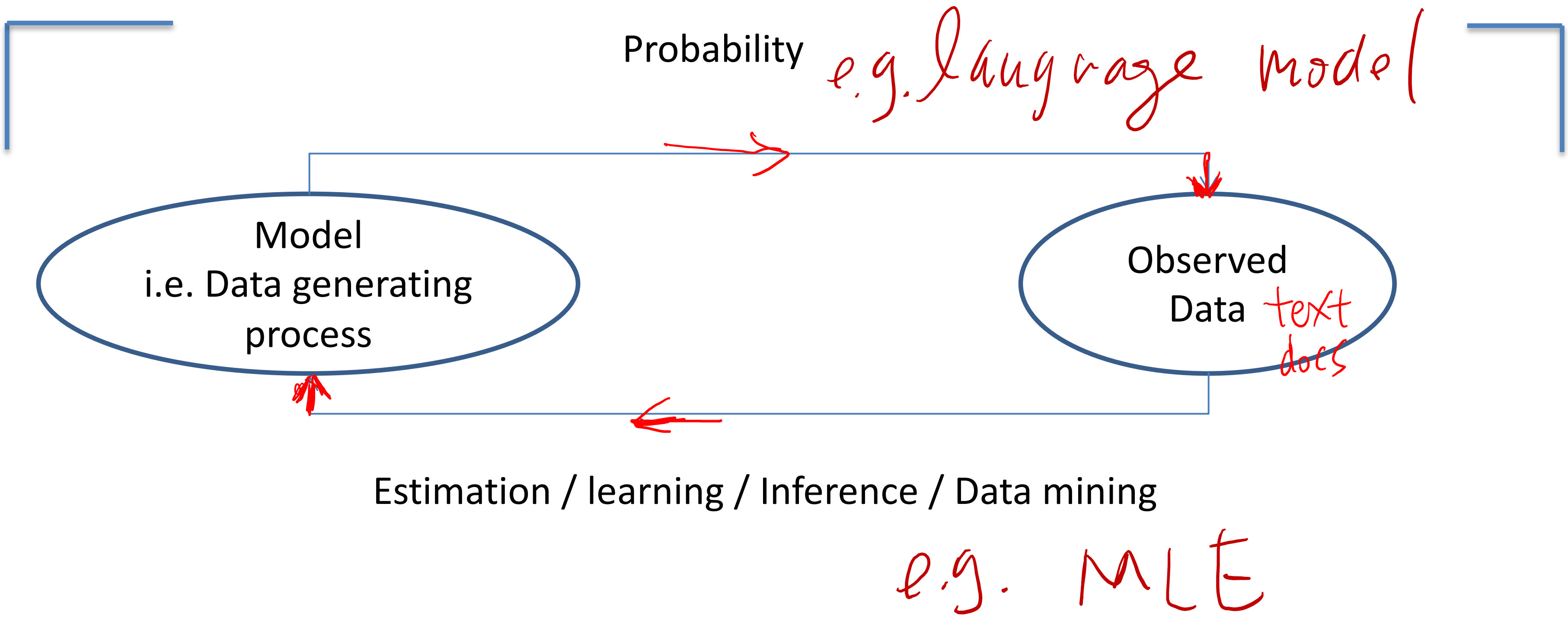
References

- Prof. Andrew Moore's review tutorial
- Prof. Ke Chen NB slides
- Prof. Carlos Guestrin recitation slides
- Prof. Raymond J. Mooney and Jimmy Lin's slides about language model
- Prof. Manning' textCat tutorial

Today : Naïve Bayes Classifier for Text

- ✓ Dictionary based Vector space representation of text article
- ✓ Multivariate Bernoulli vs. Multinomial
- ✓ Multivariate Bernoulli
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
- ✓ Multinomial naïve Bayes classifier
  - Testing
  - Training With Maximum Likelihood Estimation for estimating parameters
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models (Extra)

The Big Picture



But how to specify a model?

Build a *generative model* that approximates how data is produced.

Model 2: Multinomial Naïve Bayes

- 'Bag of words' representation of text

word	*frequency*
grain(s)	3
oilseed(s)	2
total	3
wheat	1
maize	1
soybean	1
tonnes	1
...	...

$$\Pr(W_1 = n_1, \ldots, W_k = n_k \mid C = c)$$

Can be represented as a multinomial distribution.

Words = like colored balls, there are *K* possible type of them (i.e. from a dictionary of K words)

Document = contains N words, each word occurs n_i times (like a bag of N colored balls)

WHY is this naïve ???

multinomial coefficient, normally can leave out in practical calculations.

why naive ???

$$P(W_1 = n_1, \ldots, W_k = n_k \mid c, N, \theta_1, \ldots, \theta_k) = \boxed{\frac{N!}{n_1! n_2! \ldots n_k!}} \theta_1^{n_1} \theta_2^{n_2} \ldots \theta_k^{n_k}$$

Main Question:

WHY MULTINOMIAL ON TEXT IS NAÏVE PROB. MODELING ?

Multinomial Naïve Bayes as ➔ a *generative model* that approximates how a text string is produced

- **Stochastic Language Models:**
  - Model *probability* of generating strings (each word in turn following the sequential ordering in the string) in the language (commonly all strings over dictionary ∑).
  - E.g., unigram model

Model C_1

0.2	the	→ θ_1
0.1	a	→ θ_2
0.01	boy	→ θ_3
0.01	dog	
0.03	said	
0.02	likes	
…		→ θ_K

Σ_α

Adapt From Manning' textCat tutorial

Multinomial Naïve Bayes as ➔ a *generative model* that approximates how a text string is produced

- **Stochastic Language Models:**
  - Model *probability* of generating strings (each word in turn following the sequential ordering in the string) in the language (commonly all strings over dictionary ∑).
  - E.g., unigram model

$P(d|C_1) = P(\text{the boy like the dog} | C_1)$

Model C_1

0.2	the
0.1	a
0.01	boy
0.01	dog
0.03	said
0.02	likes
…	

the	boy	likes	the	dog
0.2	0.01	0.02	0.2	0.01

Multiply all five terms

P(d | C_1) = 0.00000008

Adapt From Manning' textCat tutorial

Multinomial Naïve Bayes as Conditional Stochastic Language Models

- Model conditional *probability* of generating any string from two possible models

Model C1	
0.2	the
0.01	boy
0.0001	said
0.0001	likes
0.0001	black
0.0005	dog
0.01	garden

Model C2	
0.2	the
0.0001	boy
0.03	said
0.02	likes
0.1	black
0.01	dog
0.0001	garden

P(C2) = P(C1) ⇒ [from training]

	the	boy	likes	black	dog
C1	0.2	0.01	0.0001	0.0001	0.0005
C2	0.2	0.0001	0.02	0.1	0.01

P(d|C2) P(C2) > P(d|C1) P(C1)

[testing]

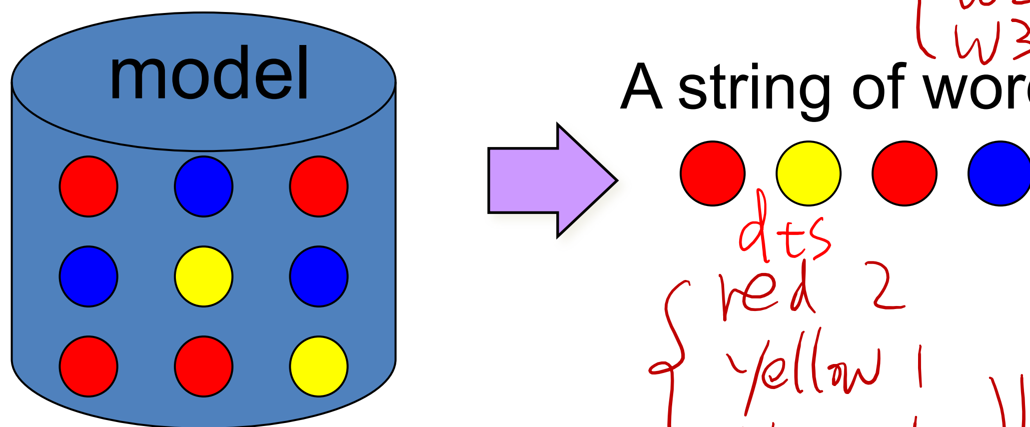
➔ d is more likely to be from class C2

A Physical Metaphor

- Colored balls are randomly drawn from (with replacement)

$K=3$

$\left\{ \begin{array}{lll} w_1 & \text{red} & \theta_1 \\ w_2 & \text{blue} & \theta_2 \\ w_3 & \text{yellow} & \theta_3 \end{array} \right.$



dts

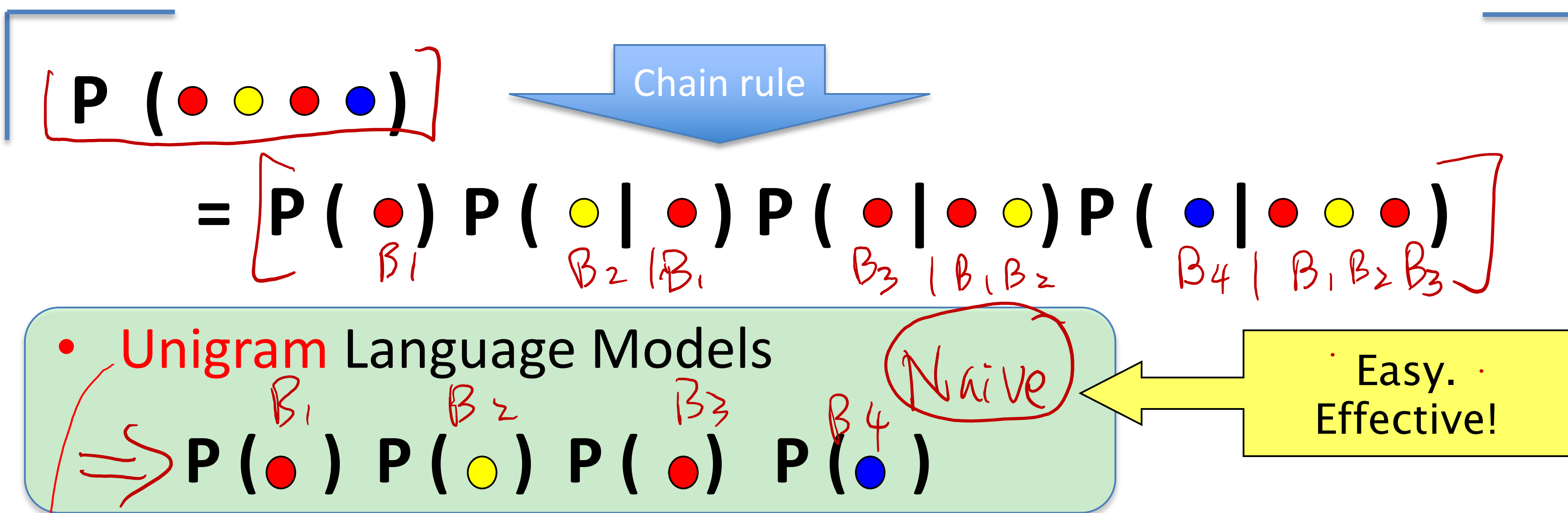
$\left\{ \begin{array}{ll} \text{red} & 2 \\ \text{yellow} & 1 \\ \text{blue} & 1 \end{array} \right.$

$\left. \begin{array}{l} \theta_1 \\ \theta_2 \\ \theta_3 \end{array} \right\}$ model paras

$= \theta_1^2 \theta_2^1 \theta_3^1$

[multinomial Distri]

Unigram language model ➔ More general:
Generating language string from a probabilistic model



NAÏVE : conditional independent on each position of the string

Unigram model : each position is independent from other positions in the text

Adapt From Manning' textCat tutorial

Unigram language model ➔ More general:

Generating language string from a probabilistic model

$$P\,({\color{red}\bullet}\,{\color{gold}\bullet}\,{\color{red}\bullet}\,{\color{blue}\bullet})$$

Chain rule

$$= \Big[\underbrace{P({\color{red}\bullet})}_{B_1}\ \underbrace{P({\color{gold}\bullet}\mid{\color{red}\bullet})}_{B_2 \mid B_1}\ \underbrace{P({\color{red}\bullet}\mid{\color{red}\bullet}\,{\color{gold}\bullet})}_{B_3 \mid B_1, B_2}\ \underbrace{P({\color{blue}\bullet}\mid{\color{red}\bullet}\,{\color{gold}\bullet}\,{\color{red}\bullet})}_{B_4 \mid B_1, B_2, B_3}\Big]$$

- Unigram Language Models   Naive

$$\Rightarrow \overset{B_1}{P({\color{red}\bullet})}\ \overset{B_2}{P({\color{gold}\bullet})}\ \overset{B_3}{P({\color{red}\bullet})}\ \overset{B_4}{P({\color{blue}\bullet})}$$

Easy. Effective!

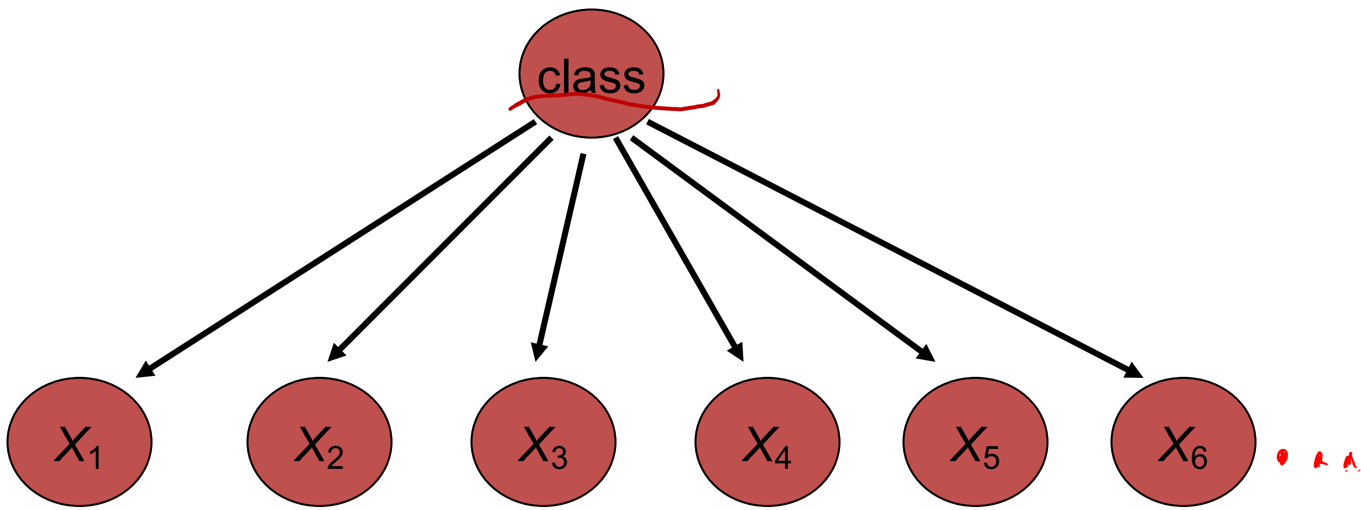
NAÏVE : conditional independent on each position of the string

- Also could be bigram (or generally, *n*-gram) Language Models

$$\underset{\text{univ.}}{\overset{B_1}{P({\color{red}\bullet})}}\ \underset{\text{of} \mid \text{univ.}}{\overset{B_2 \mid B_1}{P({\color{gold}\bullet}\mid{\color{red}\bullet})}}\ \underset{\text{virg.} \mid \text{of}}{\overset{B_3 \mid B_2}{P({\color{red}\bullet}\mid{\color{gold}\bullet})}}\ \overset{B_4 \mid B_3}{P({\color{blue}\bullet}\mid{\color{red}\bullet})}\qquad B_j \mid B_{j-1}$$

Adapt From Manning' textCat tutorial

Multinomial Naïve Bayes classifier = a class conditional unigram language model



- Think of X_i as the word on the i^{th} position in the document string
- Effectively, the probability of each class is done as a class-specific unigram language model

Adapt From Manning' textCat tutorial

	the	boy	likes	the	dog
C_j	0.2	0.01	0.02	0.2	0.01
position	1	2	3	4	5

$$\begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_K \end{bmatrix}_1 \quad \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_K \end{bmatrix}_2 \quad \cdots \quad \Rightarrow$$

approximate with the same vector across all positions

$$\begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_K \end{bmatrix}$$

Using Multinomial Naive Bayes Classifiers to Classify Text: Basic method

- Attributes are text positions, values are words.

$\Rightarrow \operatorname{argmax} P(C_j | X)$

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

the boy like the dog

$$= \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \underbrace{P(x_1 = \text{"the"} | c_j)}_{?} \cdots \underbrace{P(x_n = \text{"the"} | c_j)}_{?}$$

- Still too many possibilities
  - Use same parameters for a word across positions
  - Result is bag of words model (over word tokens)

Multinomial Naïve Bayes: Classifying Step

testing

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

Easy to implement, no need to construct bag-of-words vector explicitly !!!

P(wk|Cj) at position i

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

the	boy	likes	black	dog
0.2	0.01	0.0001	0.0001	0.0005
0.2	0.0001	0.02	0.1	0.01

P(s|C2) P(C2) > P(s|C1) P(C1)

Adapt From Manning' textCat tutorial

Multinomial Naïve Bayes: Classifying Step

- Positions ← all word positions in current document which contain tokens found in *Vocabulary*

- Return c_{NB}, where

Easy to implement, no need to construct bag-of-words vector explicitly !!!

P(wk|Cj)

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

Equal to, (leaving out of multinomial coefficient)

$$\Pr(W_1 = n_1, \ldots, W_k = n_k \mid C = c_j)$$

the	boy	likes	black	dog
0.2	0.01	0.0001	0.0001	0.0005
0.2	0.0001	0.02	0.1	0.01

P(s|C2) P(C2) > P(s|C1) P(C1)

Adapt From Manning' textCat tutorial

Multinomial Bayes: Time Complexity

T: num. doc
|V| dict size
|C| class size

- Training Time: $O(T^*L_d + |C||V|)$
  where L_d is the average length of a document in D.
  - Assumes V and all D_i , n_i, and $n_{k,j}$ pre-computed in $O(T^*L_d)$ time during one pass through all of the data.
  - |C| |V| = Complexity of computing all probability values (loop over words and classes)
  - Generally just $O(T^*L_d)$ since usually $|C||V| < T^*L_d$
- 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 words.
  - Plus, robust in practice

Adapt From Manning' textCat tutorial

References

- Prof. Andrew Moore's review tutorial
- Prof. Ke Chen NB slides
- Prof. Carlos Guestrin recitation slides