UVA CS 6316/4501 – Fall 2016 Machine Learning

Lecture 13: Naïve Bayes Classifier for Text Classification

Dr. Yanjun Qi

University of Virginia

Department of Computer Science

Where are we ? → Five major sections of this course

- ❑ Regression (supervised)
- ❑ Classification (supervised)
- ❑ Unsupervised models
- ❑ Learning theory
- ❑ Graphical models

Where are we ? → Three major sections for classification

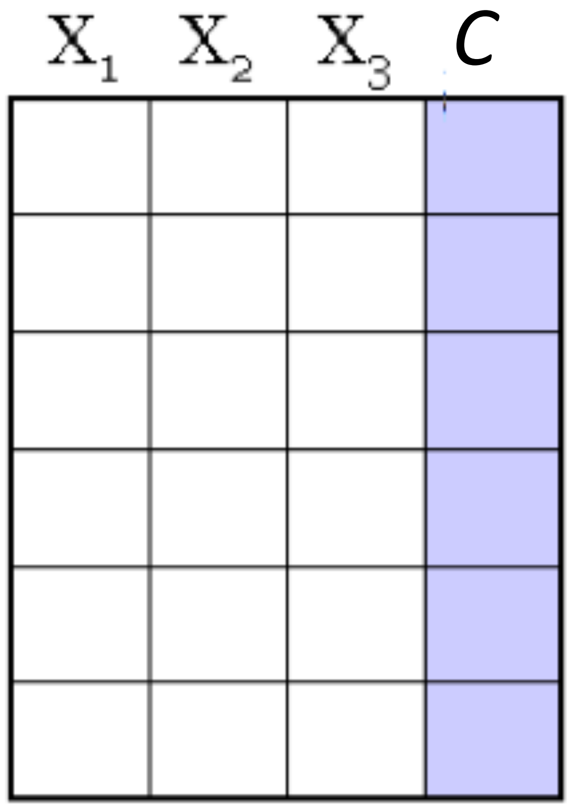
- We can divide the large variety of classification approaches into roughly three major types

1. Discriminative
   - directly estimate a decision rule/boundary
   - e.g., support vector machine, 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

A Dataset for classification



$$f : \boxed{X} \longrightarrow \boxed{C}$$

Output as Discrete Class Label
$C_1, C_2, \ldots, C_L$

$$\boxed{P(C \mid \mathbf{X})}$$

$(x_1, x_2, \ldots, x_p)$

- **Data**/*points/instances/examples/samples/records*: [rows]
- **Features**/*attributes/dimensions/independent variables/covariates/predictors/regressors*: [columns, except the last]
- **Target**/*outcome/response/label/dependent variable*: special column to be predicted [last column]

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
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - Training With Maximum Likelihood Estimation for estimating parameters

Text document classification, e.g. spam email filtering

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

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

TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

99 MILLION EMAIL ADDRESSES FOR ONLY $99

$\rightarrow$ P (C=spam | D)

Text classification Tasks

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

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

Text Classification: Examples

- Classify shipment articles into one 93 categories. $\{C_1, C_2, \cdots, C_{93}\}$
- An example category 'wheat'

ARGENTINE 1986/87 GRAIN/OILSEED REGISTRATIONS

BUENOS AIRES, Feb 26

Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:

Bread wheat prev 1,655.8, Feb 872.0, March 164.6, total 2,692.4 (4,161.0).

Maize Mar 48.0, total 48.0 (nil).

Sorghum nil (nil)

Oilseed export registrations were:

Sunflowerseed total 15.0 (7.9)

Soybean May 20.0, total 20.0 (nil)

The board also detailed export registrations for subproducts, as follows....

Representing text:
a list of words → Dictionary

D

argentine, 1986, 1987, grain, oilseed, registrations, buenos, aires, feb, 26, argentine, grain, board, figures, show, crop, registrations, of, grains, oilseeds, and, their, products, to, february, ~~11~~, in, …

→ C

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

ARGENTINE 1986/87 GRAIN/OILSEED REGISTRATIONS
BUENOS AIRES, Feb 26
Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:
Bread wheat prev 1,655.8, Feb 872.0, March 164.6, total 2,692.4 (4,161.0).
Maize Mar 48.0, total 48.0 (nil).
Sorghum nil (nil)
Oilseed export registrations were:
Sunflowerseed total 15.0 (7.9)
Soybean May 20.0, total 20.0 (nil)
The board also detailed export registrations for sub-products, as follows....

→

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

ARGENTINE 1986/87 GRAIN/OILSEED REGISTRATIONS
BUENOS AIRES, Feb 26
Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:
Bread wheat prev 1,655.8, Feb 872.0, March 164.6, total 2,692.4 (4,161.0).
Maize Mar 48.0, total 48.0 (nil).
Sorghum nil (nil)
Oilseed export registrations were:
Sunflowerseed total 15.0 (7.9)
Soybean May 20.0, total 20.0 (nil)
The board also detailed export registrations for sub-products, as follows....

➔

word	*Boolean*
grain(s)	Yes
oilseed(s)	Yes
total	Yes
wheat	Yes
maize	Yes
Panda	No
Tiger	No
...	...

Bag of word representation:

Represent text as a vector of word *frequencies*.

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

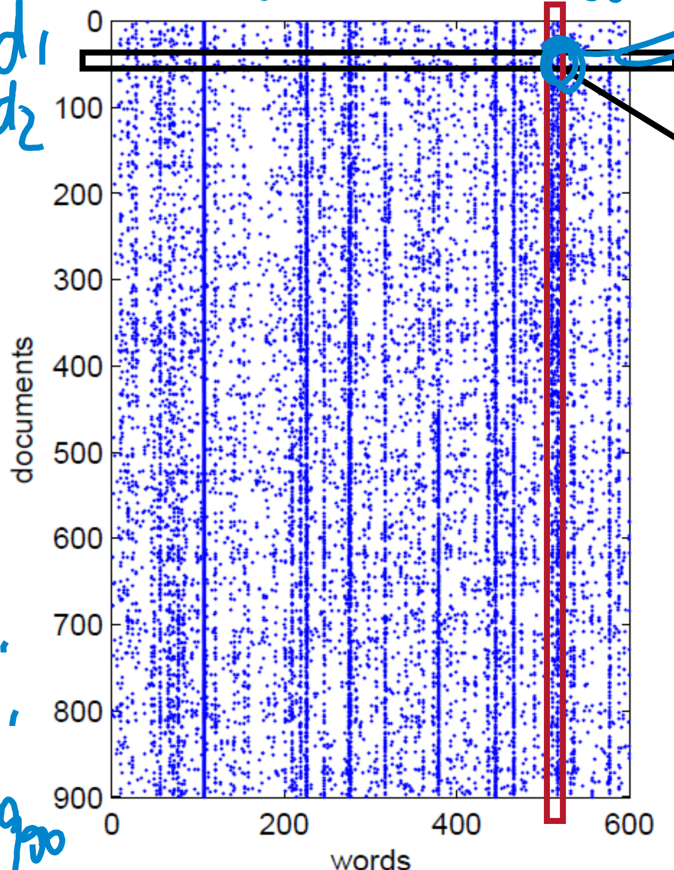
BoW _ many Variations

Bag of words representation

[p = #voc]

$w_1, w_2 \; \ldots \; w_{600}$

d_1, d_2, ..., d_{900}



word *j*

document i

$X(d_i, w_j)$

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

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

A collection of documents

	X_1	X_2	X_3	C
s_1				
s_2				
s_3				
s_4				
s_5				
s_6				

Bag of words

- What simplifying assumption are we taking?

We assumed *word order* is not important.

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

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 training corpus with <UNK>.

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
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - Training With Maximum Likelihood Estimation for estimating parameters

'Bag of words' representation of text

ARGENTINE 1986/87 GRAIN/OILSEED REGISTRATIONS
BUENOS AIRES, Feb 26
Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:
Bread wheat prev 1,655.8, Feb 872.0, March 164.6, total 2,692.4 (4,161.0).
Maize Mar 48.0, total 48.0 (nil).
Sorghum nil (nil)
Oilseed export registrations were:
Sunflowerseed total 15.0 (7.9)
Soybean May 20.0, total 20.0 (nil)
The board also detailed export registrations for sub-products, as follows....

→

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

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

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

?

$$\underset{i \in \{1,2,\ldots L\}}{\operatorname{argmax}} \; p(C_i) \boxed{p(d \mid C_i)}$$

'Bag of words' representation of text

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

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

Two Previous models

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

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

Probabilistic Models of text documents

$W_1 \ldots W_K \in$ Vocabulary

d_1 ⋮ d_n

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

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

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!

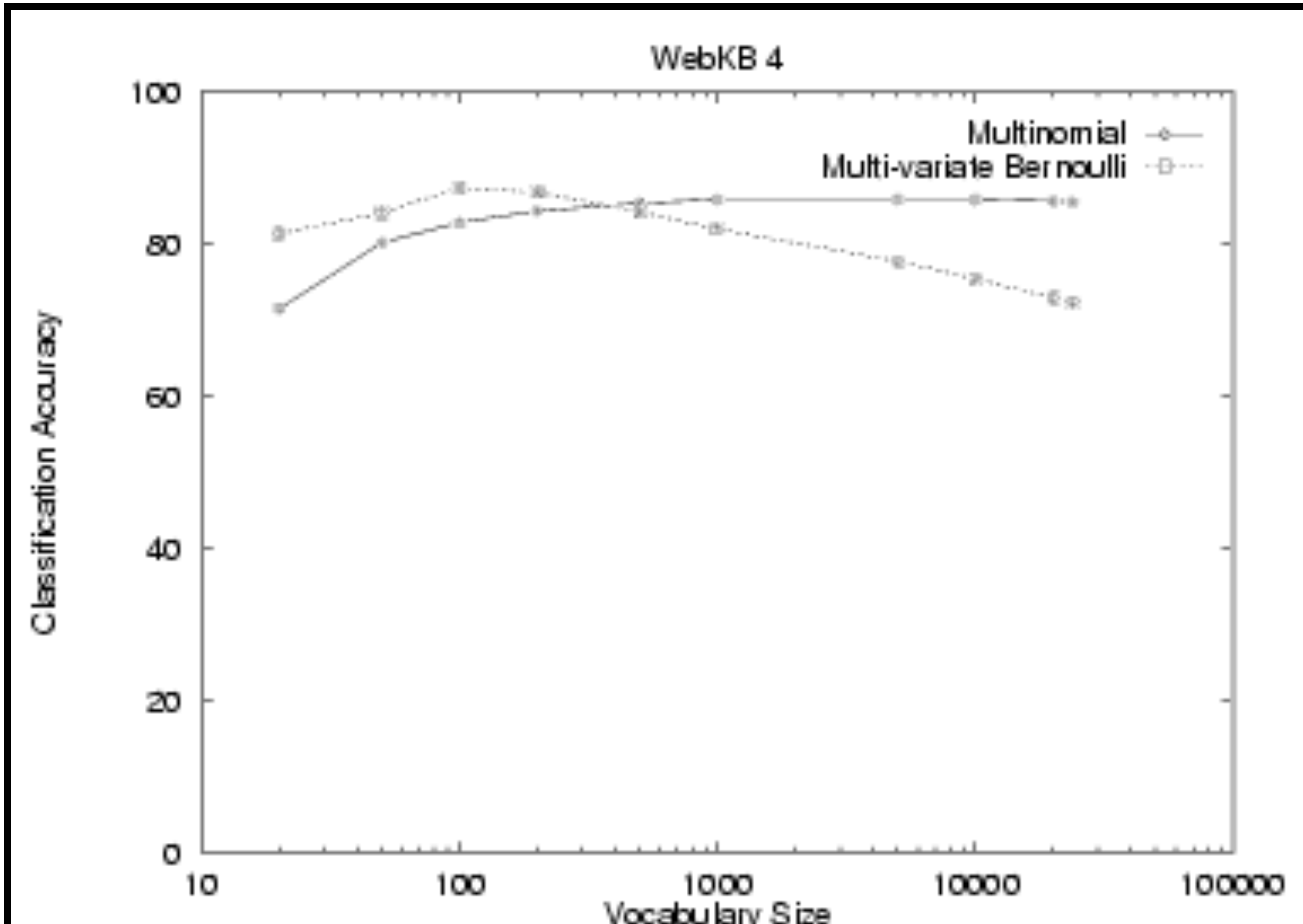
Adapt From Manning' textCat tutorial

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 train
  - Cornell, Washington, U.Texas, Wisconsin
- 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
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - Training With Maximum Likelihood Estimation for estimating parameters

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 topic class label, C_j appearance of one word in the document tells us nothing about chances that another word appears

$p(d|c)$

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

dictionary of words

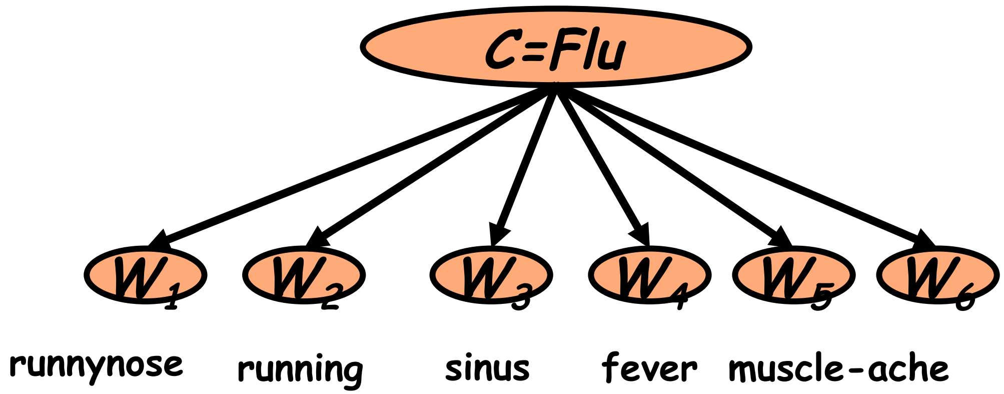
Model 1: Multivariate Bernoulli Naïve Bayes Classifier

$$\underbrace{P(w_1, w_2, \ldots, w_k \mid c)}_{P(d \mid C_j)} = P(w_1 \mid c)\, P(w_2 \mid c) \ldots P(w_k \mid c)$$

word	*True/false*
grain(s)	True
oilseed(s)	True
total	True
wheat	True
chemical	False
...	...

- **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



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

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

a generation process

- Multivariate Bernoulli model is appropriate for binary feature variables

Review: Bernoulli Distribution e.g. Coin Flips

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

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

Review: Bernoulli Distribution e.g. Coin Flips

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

$d_{ts} = \{W_1 = \text{true},\ W_2 = \text{false},\ W_3 = \text{true}\}$

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

$p(d_{ts} \mid c_j) = P_{w1,j}(1 - P_{w2,j})\, P_{w3,j}$

$p(c_j)$

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
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - Training With Maximum Likelihood Estimation for estimating parameters

p estimated from data

$$\begin{array}{c|ccccc|}
 & c_1 & c_2 & \cdots & & c_C \\
\hline
w_1 & & & & & \\
w_2 & & & p_{w_k,c_j} & & \\
\vdots & & & & & \\
w_K & & & & & \\
\hline
\end{array}$$

p_{w_i,c_j}

Maximum Likelihood Estimation

A general Statement

Consider a sample set T=(X_1...X_n) which is drawn from a probability distribution P(X|\theta) where \theta are parameters.

If the Xs are independent with probability density function P(X_i|\theta), the joint probability of the whole set is

$$P(X_1 \ldots X_n|\theta) = \prod_{i=1}^{n} P(X_i|\theta)$$

this may be maximised with respect to \theta to give the maximum likelihood estimates.

The idea is to

- ✓ assume a particular model with unknown parameters, θ
- ✓ we can then define the probability of observing a given event conditional on a particular set of parameters. $P(X_i \mid \theta)$
- ✓ We have observed a set of outcomes in the real world. $x_1, x_2, \ldots, x_n$
- ✓ It is then possible to choose a set of parameters which are most likely to have produced the observed results.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \left[P(X_1 \ldots X_n \mid \theta) \right] \rightarrow \text{likehood}$$

This is maximum likelihood. In most cases it is both consistent and efficient. It provides a standard to compare other estimation techniques.

$$\log(L(\theta)) = \sum_{i=1}^{n} \left[\log(P(X_i \mid \theta)) \right] \Rightarrow \text{log likehood}$$

It is often convenient to work with the Log of the likelihood function.

Review: Bernoulli Distribution e.g. Coin Flips

- You flip *n* coins
  - How many heads would you expect
  - Head with probability *p*
  - Number of heads X out of n trial
  - Each Trial following Bernoulli distribution with parameters *p*

N trails, e.g. $\{H\ H\ T\ H\ H\ T\ H\ T \ldots H\}$

$x_1\ x_2\ x_3\ x_4 \ldots x_n$

Defining Likelihood for Bernoulli

Bernoulli

$x_i = \{0, 1\}$ or $\{H, T\}$

$p(x_1, x_2 \ldots, x_n | p)$

- Likelihood = p(data | parameter) $= \prod p(x_i | p)$

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

→ e.g., for n independent tosses of coins, with unknown p

function of x_i

PMF: $$f(x_i | p) = p^{x_i}(1-p)^{1-x_i}$$

$$x = \sum_{i=1}^{n} x_i$$

LIKELIHOOD:

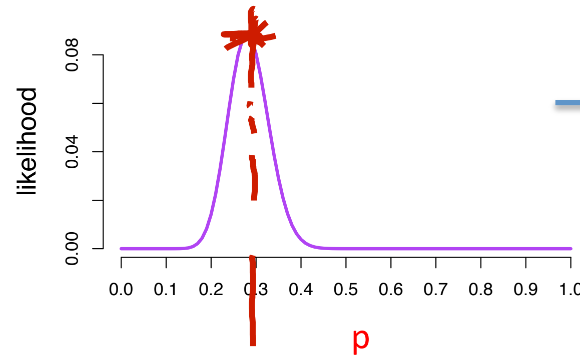
$$L(p) = \prod_{i=1}^{n} p^{x_i}(1-p)^{1-x_i} = p^x(1-p)^{n-x}$$

function of p

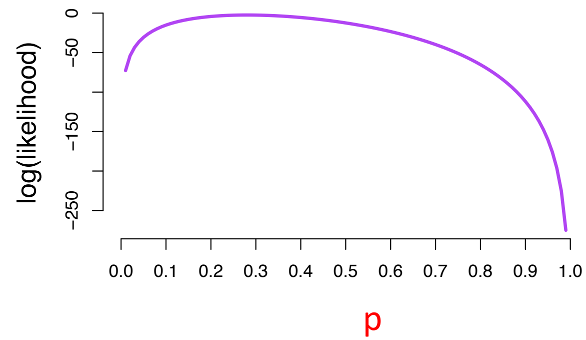
Observed data → x heads-up from n trials

Deriving the Maximum Likelihood Estimate for Bernoulli

$$\underbrace{L(p)}_{\text{maximize}} = p^x (1-p)^{n-x}$$

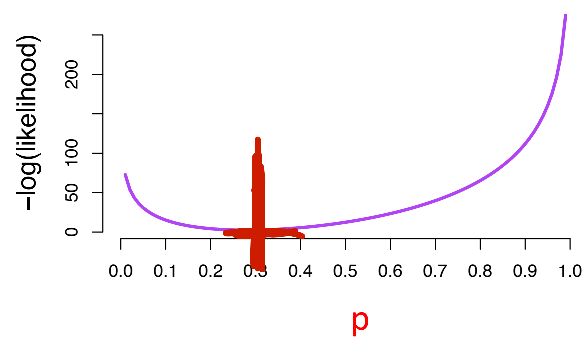


$$\underbrace{\log(L(p))}_{\text{maximize}} = \log\left[p^x (1-p)^{n-x}\right]$$



Minimize the negative log-likelihood

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



Deriving the Maximum Likelihood Estimate for Bernoulli

Minimize the negative log-likelihood

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

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

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

Deriving the Maximum Likelihood Estimate for Bernoulli

$$\arg_p\{-l(p)\} = \arg\min_p \{-x\log(p)-(n-x)\log(1-p)\}$$

$$\frac{dl(p)}{dp} = -\frac{x}{p} - \frac{-(n-x)}{1-p} = 0$$

$$0 = -\frac{x}{\hat{p}} + \frac{n-x}{1-\hat{p}}$$

$$0 = \frac{-x(1-\hat{p}) + \hat{p}(n-x)}{\hat{p}(1-\hat{p})}$$

$$0 = -x + \hat{p}x + \hat{p}n - \hat{p}x$$

$$0 = -x + \hat{p}n$$

Minimize the negative log-likelihood

➔ MLE parameter estimation

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

i.e. Relative frequency of a binary event

Parameter estimation

- Multivariate Bernoulli model:

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

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
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - Training With Maximum Likelihood Estimation for estimating parameters

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.

In a document class of 'wheat', "grain" is more likely. where as in a "hard drive" shipment class the parameter for 'grain' is going to be smaller.

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)

In a document class of 'wheat', "grain" is more likely. where as in a "hard drive" shipment class the parameter for 'grain' is going to be smaller.

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)

In a document class of 'wheat', "grain" is more likely. where as in a "hard drive" shipment class the parameter for 'grain' is going to be smaller.

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

$P_{head} = p$

$P_{tail} = 1 - p$

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).

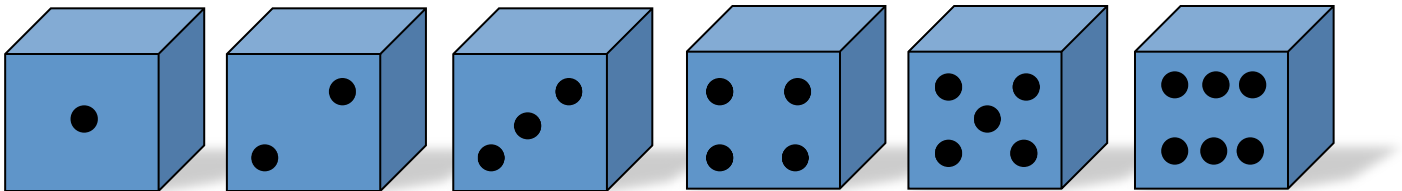
$K=2$

- 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)

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

$K=6$



Multinomial Distribution

$P(D|C_i) = P(W_1, W_2 .. W_k | C)$

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

Number of possible orderings of N balls

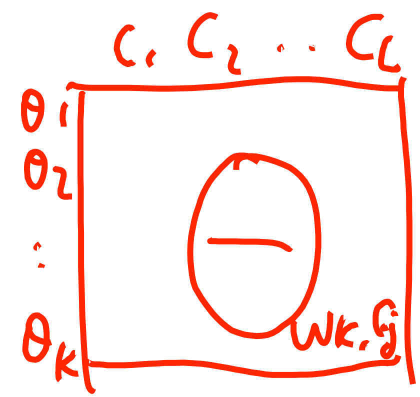
$$P(W_1 = n_1, ..., W_k = n_k \mid c_i, N, \theta_1, ..., \theta_k) = \cancel{\frac{N!}{n_1! n_2! .. n_k!}} \theta_1^{n_1} \theta_2^{n_2} .. \theta_k^{n_k} \Big| C = C_i$$

$p(d|c_i) =$

Note events are independent

order invariant selections

$$\sum_{i=1}^{k} n_i = N \qquad \sum_{i=1}^{k} \theta_i = 1$$



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

$\theta_2 = 1 - \theta_1$

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)

multinomial coefficient, normally can leave out in practical calculations.

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

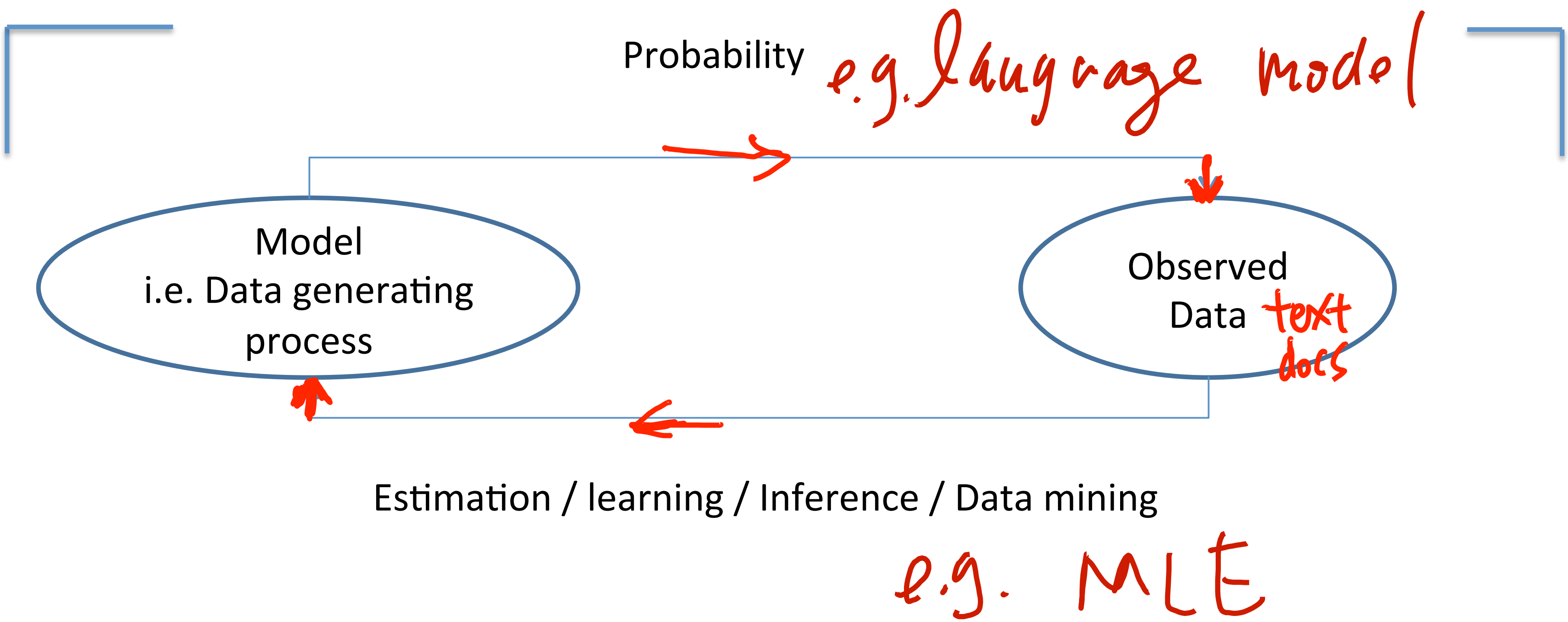
estimate $\Theta_{K \times L}$ from training data

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

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
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - Training With Maximum Likelihood Estimation for estimating parameters

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)

multinomial coefficient, normally can leave out in practical calculations.

WHY is this naïve ???

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

why naive ???

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

Σ

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

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

$$P(d \mid C\_1) = 0.00000008$$

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]

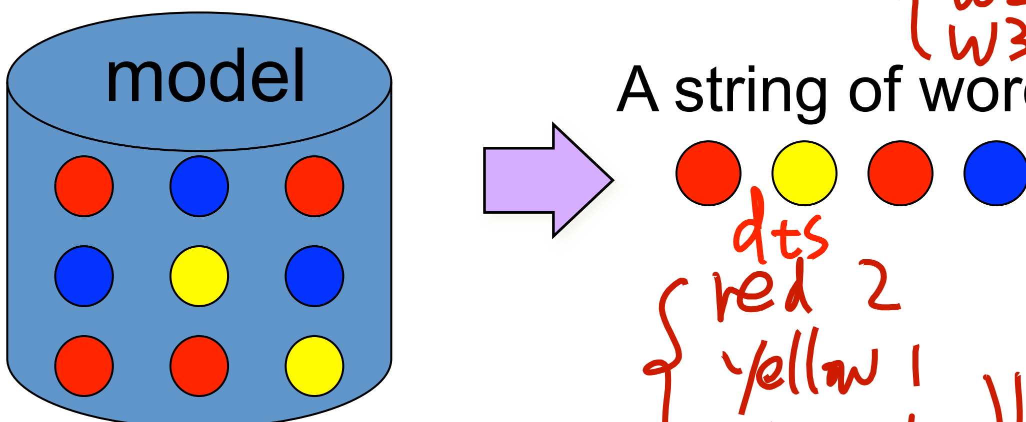
➔ d is more likely to be from class C2

A Physical Metaphor

- Colored balls are randomly drawn from (with replacement)

K=3

$\begin{cases} w_1 & \text{red} & \theta_1 \\ w_2 & \text{blue} & \theta_2 \\ w_3 & \text{yellow} & \theta_3 \end{cases}$



dts

$\begin{cases} \text{red} & 2 \\ \text{yellow} & 1 \\ \text{blue} & 1 \end{cases}$ $\Downarrow$

$\left.\begin{matrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{matrix}\right\}$ model params

$$P(\underbrace{\overset{dts}{\bullet \; \bullet \; \bullet \; \bullet}}) = P(\overset{w_1}{\bullet})\; P(\overset{w_3}{\bullet})\; P(\overset{w_1}{\bullet})\; P(\overset{w_2}{\bullet})$$

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

[multinomial Distri]

Unigram language model ➔ More general:

Generating language string from a probabilistic model

P (🔴 🟡 🔴 🔵)

Chain rule

= P (🔴) P (🟡 | 🔴) P (🔴 | 🔴 🟡) P (🔵 | 🔴 🟡 🔴)

B_1 $B_2 | B_1$ $B_3 | B_1, B_2$ $B_4 | B_1, B_2, B_3$

- Unigram Language Models

Naive

B_1 B_2 B_3 B_4

⇒ P (🔴) P (🟡) P (🔴) P (🔵)

Easy.
Effective!

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 (🔴 🟡 🔴 🔵)

Chain rule

= P (🔴) P (🟡 | 🔴) P (🔴 | 🔴 🟡) P (🔵 | 🔴 🟡 🔴)

B_1 $\quad$ $B_2 | B_1$ $\quad$ $B_3 | B_1, B_2$ $\quad$ $B_4 | B_1, B_2, B_3$

- Unigram Language Models

Naive

B_1 $\quad$ B_2 $\quad$ B_3 $\quad$ B_4

⇒ P (🔴) P (🟡) P (🔴) P (🔵)

Easy. Effective!

NAÏVE : conditional independent on each position of the string

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

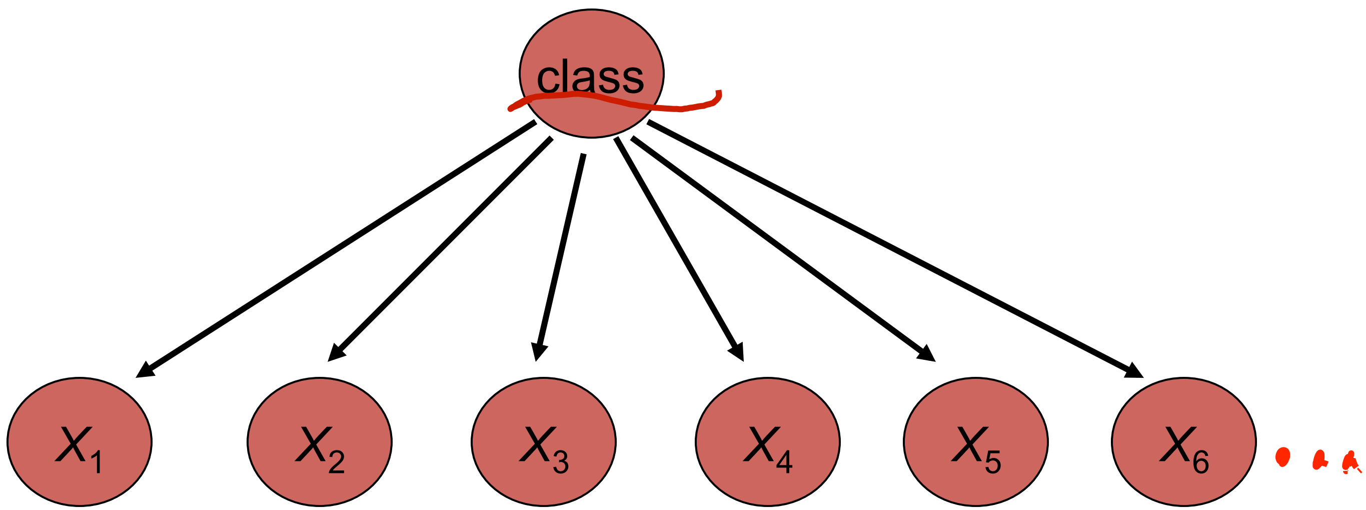
B_1 $\quad$ $B_2 | B_1$ $\quad$ $B_3 | B_2$ $\quad$ $B_4 | B_3$ $\quad$ $B_j | B_{j-1}$

P (🔴) P (🟡 | 🔴) P (🔴 | 🟡) P (🔵 | 🔴)

univ. $\quad$ of | univ. $\quad$ virg. | of

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

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

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.

$\Rightarrow \operatorname{argmax} P(c_j \mid X)$

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

the boy like the dog

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

- 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 !!!

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

P(wk|cj) at position i

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

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…

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
  - Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - Training With Maximum Likelihood Estimation for estimating parameters

The Big Picture

Probability e.g. language model

Estimation / learning / Inference / Data mining

e.g. MLE

But how to specify a model?

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

Generative Model & MLE

- Language model can be seen as a probabilistic automata for generating text strings

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

- Relative frequency estimates can be proven to be ***maximum likelihood estimates*** (MLE) since they maximize the probability that the model M will generate the training corpus T.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, P(Train \mid M(\theta))$$

Deriving the Maximum Likelihood Estimate for multinomial distribution

train T documents

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

$w_1, \ldots, w_K$

function of θ

θ vector

$$= \underset{\theta_1,\ldots,\theta_k}{\arg\max} \boxed{\prod_{t=1}^{T}} P(d_t \mid \theta_1,\ldots,\theta_k)$$

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

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

$$= \underset{\theta_1,\ldots,\theta_k}{\arg\max} \prod_{t=1}^{T} \theta_1^{\,n_{1,d_t}} \theta_2^{\,n_{2,d_t}} \ldots \theta_k^{\,n_{k,d_t}}$$

data likihood

Deriving the Maximum Likelihood Estimate for multinomial distribution

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

Constrained optimization

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

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

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

Constrained optimization MLE estimator

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

for C_j

How optimize ? See Handout - EXTRA

➔ 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

Naïve 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 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* $Text_j$
  - $$P(w_k \mid c_j) \leftarrow \frac{n_{k,j} + \alpha}{n_j + \alpha |Vocabulary|} \quad e.g., \alpha = 1$$ (Smoothing)

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

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

Parameter estimation

Multinomial model:

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

Adapt From Manning' textCat tutorial

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