# UVA CS 4501 - 001 / 6501 – 007 Introduction to Machine Learning and Data Mining

## Lecture 15: Generative Bayes Classifier, MLE, & Discriminative Model

Yanjun Qi / Jane, , PhD

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

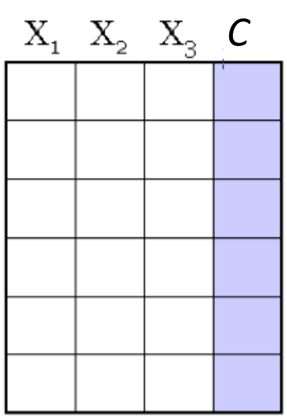
# Where are we ? ➔ Three major sections for classification

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

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

$$f : X \longrightarrow C$$

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

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

- **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 ]

## Last Lecture Recap:



### Naïve Bayes Classifier

- ✓ Why Bayes Classification – MAP Rule?
  - ▪ Empirical Prediction Error, 0-1 Loss function for Bayes Classifier
- ✓ Naïve Bayes Classifier for Text document categorization
  - ✓ Bag of words representation
  - ✓ Multinomial vs. multivariate Bernoulli
  - ✓ Multinomial naïve Bayes classifier





### Review : Probability

- Joint
- Conditional
- Marginal
- Independence and Conditional independence

$$P\left(X = x \cap Y = y \middle| Z = z\right) = P\left(X = x \middle| Z = z\right) P\left(Y = y \middle| Z = z\right)$$

- Bayes' Rule

$$P(C \mid X) = \frac{P(X,C)}{P(X)} = \frac{P(X \mid C)P(C)}{P(X)}$$

# Bayes Classifiers – MAP Rule

*Task*: Classify a new instance $X$ based on a tuple of attribute values $X = \langle X_1, X_2, \ldots, X_p \rangle$ into one of the classes $c_j \in C$

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

WHY ?

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

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

MAP = Maximum Aposteriori Probability



Adapt From Carols' prob tutorial





# Expected prediction error (EPE)

- Expected prediction error (EPE), with expectation taken w.r.t. the joint distribution Pr(C,X),
  - $\Pr(C,X) = \Pr(C \mid X)\Pr(X)$

$$\text{EPE}(f) = E_{X,C}(L(C, f(X)) = \text{E}_X \sum_{k=1}^{K} L[C_k, f(X)] \Pr(C_k, X)$$

Consider sample population distribution

➔ 0-1 loss function used: $L(k, \ell) = 1 - \delta_{kl}$

- Pointwise minimization suffices

$$\hat{G}(X) = \text{argmin}_{g \in C} \sum_{k=1}^{K} L(C_k, g) \Pr(C_k \mid X = x)$$

- ➔ simply

$$\hat{G}(X) = C_k \text{ if}$$
$$\Pr(C_k \mid X = x) = \max_{g \in C} \Pr(g \mid X = x)$$

Bayes Classifier

# Naïve Bayes Classifier for 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"*





# '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 |
| ... | ... |

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

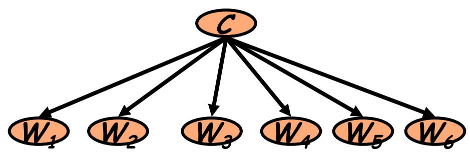
→ C*

Two models

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

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

# *Model 1*: Multivariate Bernoulli




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

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

- Multivariate Bernoulli model is appropriate for binary feature variables



Adapt From Manning' textCat tutorial





# *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)

The multinomial distribution of words is going to be different for different document class.

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.

## **Experiment: Multinomial vs** multivariate Bernoulli

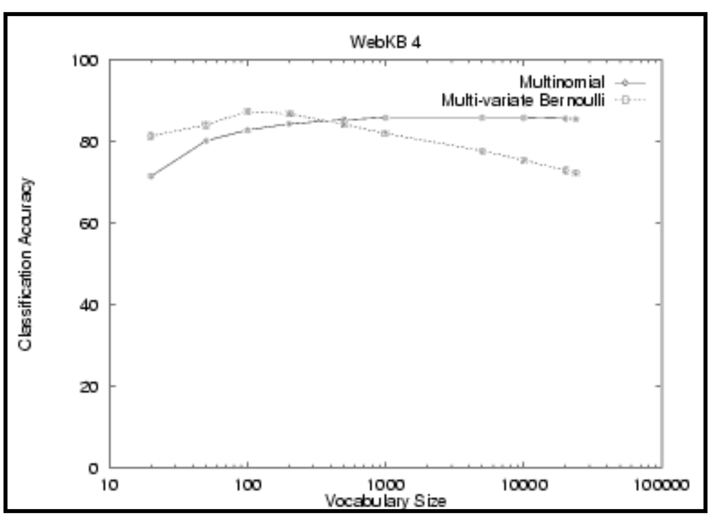
- M&N (1998) did some experiments to see which is better
- Determine if a university web page is {student, faculty, other_stuff}
- Train on ~5,000 hand-labeled web pages
  - Cornell, Washington, U.Texas, Wisconsin
- Crawl and classify a new site (CMU)



Adapt From Manning' textCat tutorial



## **Multinomial vs.** multivariate Bernoulli






Adapt From Manning' textCat tutorial

## **Today :** Generative vs. Discriminative

- ✓ Multinomial naïve Bayes classifier as Conditional Stochastic Language Models
  - ✓ a unigram Language model approximates how a text document is produced.

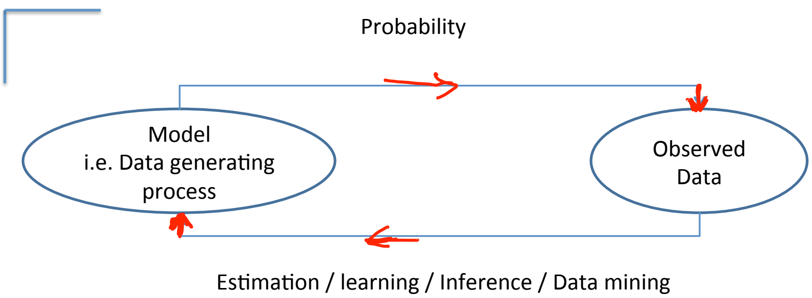
$$\Pr(W_1 = n_1, \ldots, W_k = n_k \mid C = c)$$

- ✓ Maximum Likelihood Estimation of parameters
- ✓ A discriminative model: logistic regression





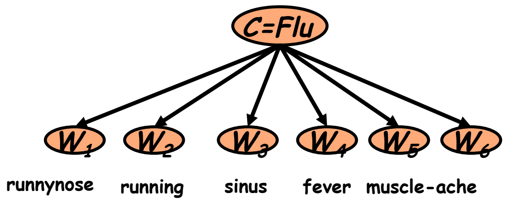
## The Big Picture




But how to specify a model?

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

# *Model 1*: Multivariate Bernoulli




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

this is naïve

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

- Multivariate Bernoulli model is appropriate for binary feature variables



Adapt From Manning' textCat tutorial





# *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.

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

## 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

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

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





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 |

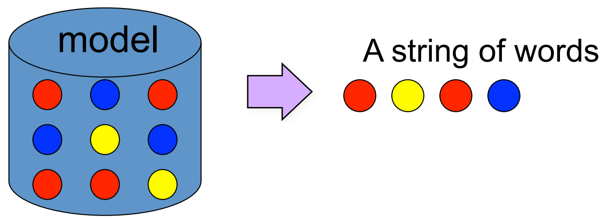
| the | boy | likes | black | dog |
|---|---|---|---|---|
| 0.2 | 0.01 | 0.0001 | 0.0001 | 0.0005 |
| 0.2 | 0.0001 | 0.02 | 0.1 | 0.01 |

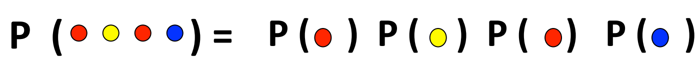
$P(s|C2)\,P(C2) > P(s|C1)\,P(C1)$

→ S is more likely to be from class C2

# A Physical Metaphor

- Colored balls are randomly drawn from (with replacement)

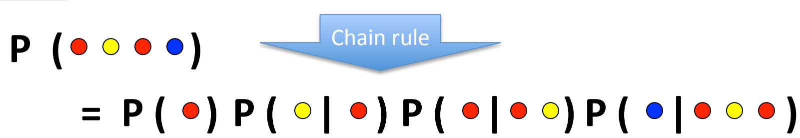


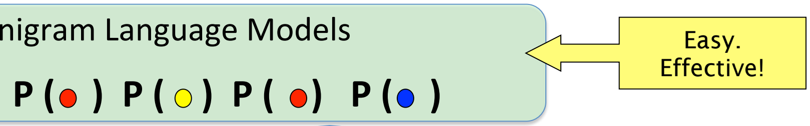





# Unigram language model ➔ Generating language string from a probabilistic model

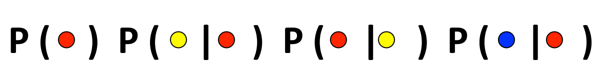



- Unigram Language Models




NAÏVE : conditional independent on each position of the string

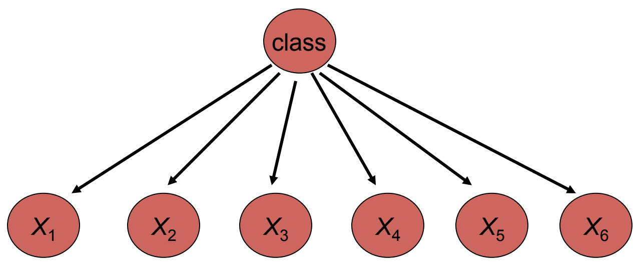
- Also could be bigram (or generally, *n*-gram) Language Models





Adapt From Manning' textCat tutorial

# Multinomial Naïve Bayes = 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



# Using Multinomial Naive Bayes Classifiers to Classify Text: Basic method

- Attributes are text positions, values are words.

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

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

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

# 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 !!!

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

Equal to, (with 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





# 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>.
  - Replace the first occurrence of each word in the training data with <UNK>.

# 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 :** Generative vs. Discriminative

- ✓ Multinomial naïve Bayes classifier as conditional Stochastic Language Models
  - ✓ a unigram Language model approximates how a text document is produced.

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

- ✓ Maximum Likelihood Estimation of parameters
- ✓ A discriminative model: logistic regression
  - ✓ Generative vs. discriminative

# Parameter estimation

- Multivariate Bernoulli model:

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

- Multinomial model:

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

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





Adapt From Manning' textCat tutorial



# Generative Model & MLE

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

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

- 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*.

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

# Maximum Likelihood Estimation

A general Statement

Consider a sample set T=($X_1$...$X_n$) which is drawn from a probability distribution P(X|A) where A are parameters. If the Xs are independent with probability density function P($X_i$|A), the joint probability of the whole set is

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

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

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, P(Train \mid M(\theta)) = \underset{\theta}{\operatorname{argmax}}\, P(X_1 \ldots X_n \mid \theta)$$





The idea is to

- ✓ assume a particular model with unknown parameters, $\theta$
- ✓ 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.
- ✓ It is then possible to choose a set of parameters which are most likely to have produced the observed results.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, P(X_1 \ldots X_n \mid \theta)$$

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

$$\log(L(\theta)) = \sum_{i=1}^{n} \log(P(X_i \mid \theta))$$

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

## Defining Likelihood

- Likelihood = p(data | parameter)

➔ e.g., for a binomial distribution with known n, but unknown p

function of x

PDF: $f(x \mid p) = \binom{n}{x} p^x (1-p)^{n-x}$

LIKELIHOOD: $L(p) = \binom{n}{x} p^x (1-p)^{n-x}$

function of p

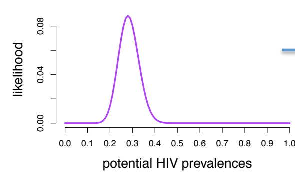




## Deriving the Maximum Likelihood Estimate

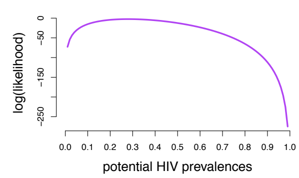
maximize

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



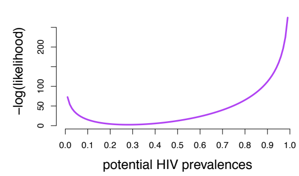

maximize

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




Minimize the negative log-likelihood

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



# Deriving the Maximum Likelihood Estimate

Minimize the negative log-likelihood

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

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

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





# Deriving the Maximum Likelihood Estimate

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

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

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

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

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

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

Minimize the negative log-likelihood

➔ MLE parameter estimation

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

i.e. Relative frequency of a binary event

# Deriving the Maximum Likelihood Estimate for multinomial distribution

LIKELIHOOD:

function of θ

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

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

$$= \arg\max_{\theta_1,\ldots,\theta_k} \prod_{t=1}^{T} \frac{N!}{n_1!n_2!..n_k!} \theta_1^{n_{1,d_t}} \theta_2^{n_{2,d_t}} ..\theta_k^{n_{k,d_t}}$$

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

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

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







# Deriving the Maximum Likelihood Estimate for multinomial distribution

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

Constrained optimization

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

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

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

Constrained optimization MLE estimator

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

How optimize ? See Handout

➔ 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 Bayes: 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 each word $w_k$ in *Vocabulary*
  - $n_k \leftarrow$ number of occurrences of $w_k$ in $Text_j$; $n$ *is length of* $Text_j$
  - $$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|} \qquad e.g., \alpha = 1$$

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





# **Today :** Generative vs. Discriminative

- ✓ Multinomial naïve Bayes classifier as Stochastic Language Models
  - ✓ a unigram Language model approximates how a text document is produced.

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

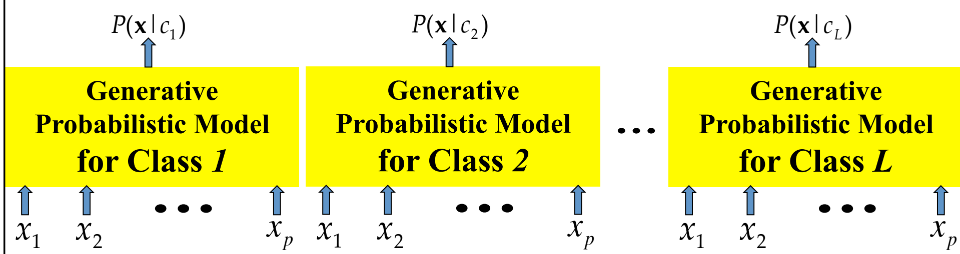
- ✓ Maximum Likelihood Estimation of parameters
- ✓ A discriminative model: logistic regression

## Establishing a probabilistic model for classification (cont.)



– **(1) Generative model**

$$\arg\max_C P(C \mid X) = \arg\max_C P(X, C)$$
$$= \arg\max_C P(X \mid C) P(C)$$




$$\mathbf{x} = (x_1, x_2, \cdots, x_p)$$



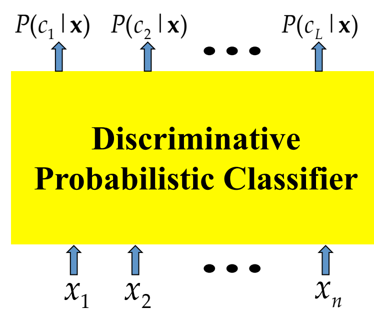
Adapt from Prof. Ke Chen NB slides

## Establishing a probabilistic model for classification



– **(2) Discriminative model**

$$P(C \mid \mathbf{X}) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$$




$$\mathbf{x} = (x_1, x_2, \cdots, x_n)$$



Adapt from Prof. Ke Chen NB slides

# Discriminative vs. Generative

Generative approach

- Model the joint distribution p(X, C) using p(X | C = $c_k$) and p(C = $c_k$)

Class prior

Discriminative approach

- Model the conditional distribution p(y| X) directly

e.g., $\frac{1}{1+e^{-(\beta_0+\beta_1 * X))}}$



# RECAP: Multivariate linear regression

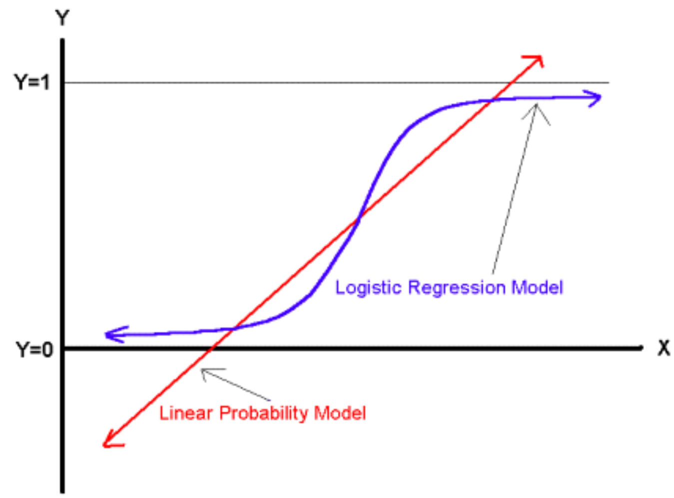
$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_i x_i$$

| | |
|---|---|
| Dependent | Independent variables |
| Predicted | Predictor variables |
| Response variable | Explanatory variables |
| Outcome variable | Covariables |

Logistic regression

$$\ln\left[\frac{P(y|x)}{1-P(y|x)}\right] = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

## Logistic regression – Binary outcome target variable Y








## Logistic Regression—when?

Logistic regression models are appropriate for target variable coded 0/1.

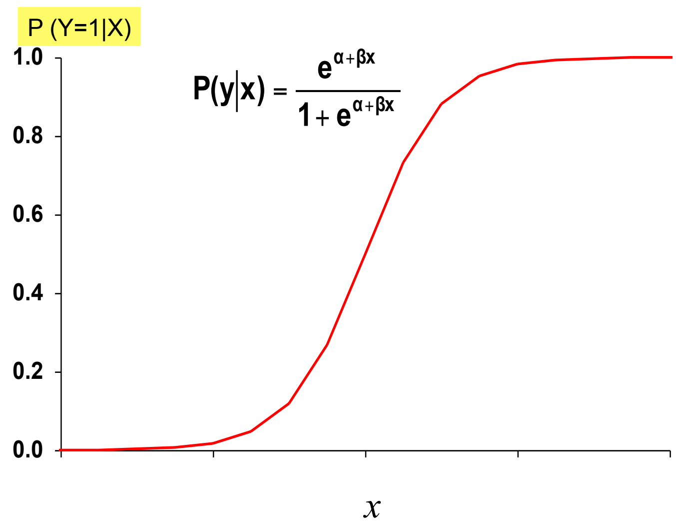
We only observe "0" and "1" for the target variable—but we think of the target variable conceptually as a probability that "1" will occur.

This means we use Bernoulli distribution to model the target variable with its Bernoulli parameter p=p(y=1|x) predefined.

The main interest ➔ predicting the probability that an event occurs (i.e., the probability that p(y=1|x) ).

# The logistic function (1)

e.g.
Probability of disease








# The logistic function (2)

$$P(y|x) = \frac{e^{\alpha+\beta x}}{1+e^{\alpha+\beta x}}$$

$$\ln\left[\frac{P(y|x)}{1-P(y|x)}\right] = \alpha + \beta x$$

logit of $P(y|x)$

From probability to logit, i.e. log odds (and back again)

$$z = \log\left(\frac{p}{1-p}\right) \qquad \text{logit function}$$

$$\frac{p}{1-p} = e^z$$

$$p = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}} \qquad \text{logistic function}$$





# The logistic function (3)

- Advantages of the logit
  - Simple transformation of P(y|x)
  - Linear relationship with x
  - Can be continuous (Logit between - ∞ to + ∞)
  - Directly related to the notion of log odds of target event

$$\ln\left(\frac{P}{1-P}\right) = \alpha + \beta x \qquad \frac{P}{1-P} = e^{\alpha+\beta x}$$

# Logistic Regression Assumptions

- Linearity in the logit – the regression equation should have a linear relationship with the logit form of the target variable

- There is no assumption about the feature variables / predictors being linearly related to each other.





# Parameter Estimation for LR → MLE from the data

- RECAP: Linear regression → Least squares

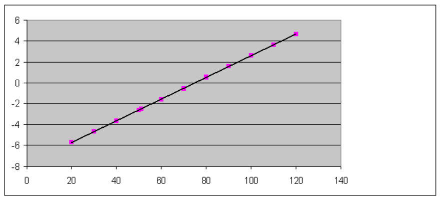
- Logistic regression: → Maximum likelihood estimation

# Binary Logistic Regression

In summary that the logistic regression tells us two things at once.

- Transformed, the “log odds” are linear.

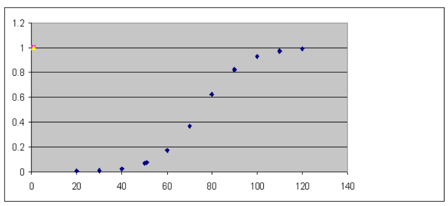
ln[p/(1-p)]




x

- Logistic Distribution

P (Y=1)




x



Yanjun Qi / UVA CS 4501-01-6501-07

# **Next :** Generative vs. Discriminative

- ✓ Multinomial naïve Bayes classifier as Stochastic Language Models
  - ✓ a unigram Language model approximates how a text document is produced.

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

- ✓ Maximum Likelihood Estimation of parameters
- ✓ A discriminative model: logistic regression
  - ✓ Discriminative vs. Generative models

# References

- Prof. Tom Mitchell's tutorials
- Prof. Raymond J. Mooney and Jimmy Lin's slides about language model