

Introduction to Bayesian methods

Lecture 10

David Sontag
New York University

Slides adapted from Luke Zettlemoyer, Carlos Guestrin, Dan Klein,
and Vibhav Gogate

Bayesian learning

- Bayesian learning uses **probability** to *model* data and *quantify uncertainty* of predictions
 - Facilitates incorporation of prior knowledge
 - Gives optimal predictions
 - Allows for decision-theoretic reasoning

Your first consulting job

- A billionaire from the suburbs of Manhattan asks you a question:
 - He says: I have thumbtack, if I flip it, what's the probability it will fall with the nail up?
 - You say: Please flip it a few times:



- You say: The probability is:
 - $P(\text{heads}) = 3/5$
- He says: **Why???**
- You say: Because...

Outline of lecture

- Review of probability
- Maximum likelihood estimation

2 examples of Bayesian classifiers:

- Naïve Bayes
- Logistic regression

Random Variables

- A random variable is some aspect of the world about which we (may) have uncertainty
 - R = Is it raining?
 - D = How long will it take to drive to work?
 - L = Where am I?
- We denote random variables with capital letters
- Random variables have domains
 - R in $\{\text{true}, \text{false}\}$ (sometimes write as $\{+r, \neg r\}$)
 - D in $[0, \infty)$
 - L in possible locations, maybe $\{(0,0), (0,1), \dots\}$

Probability Distributions

- Discrete random variables have distributions

T	P
warm	0.5
cold	0.5

W	P
sun	0.6
rain	0.1
fog	0.3
meteor	0.0

- A discrete distribution is a TABLE of probabilities of values
- The probability of a state (lower case) is a single number

$$P(W = \text{rain}) = 0.1$$

$$P(\text{rain}) = 0.1$$

- Must have:

$$\forall x P(x) \geq 0$$

$$\sum_x P(x) = 1$$

Joint Distributions

- A *joint distribution* over a set of random variables: X_1, X_2, \dots, X_n specifies a real number for each assignment:

$$P(X_1 = x_1, X_2 = x_2, \dots, X_n = x_n)$$

$$P(x_1, x_2, \dots, x_n)$$

- How many assignments if n variables with domain sizes d ?

- Must obey:

$$P(x_1, x_2, \dots, x_n) \geq 0$$

$$\sum_{(x_1, x_2, \dots, x_n)} P(x_1, x_2, \dots, x_n) = 1$$

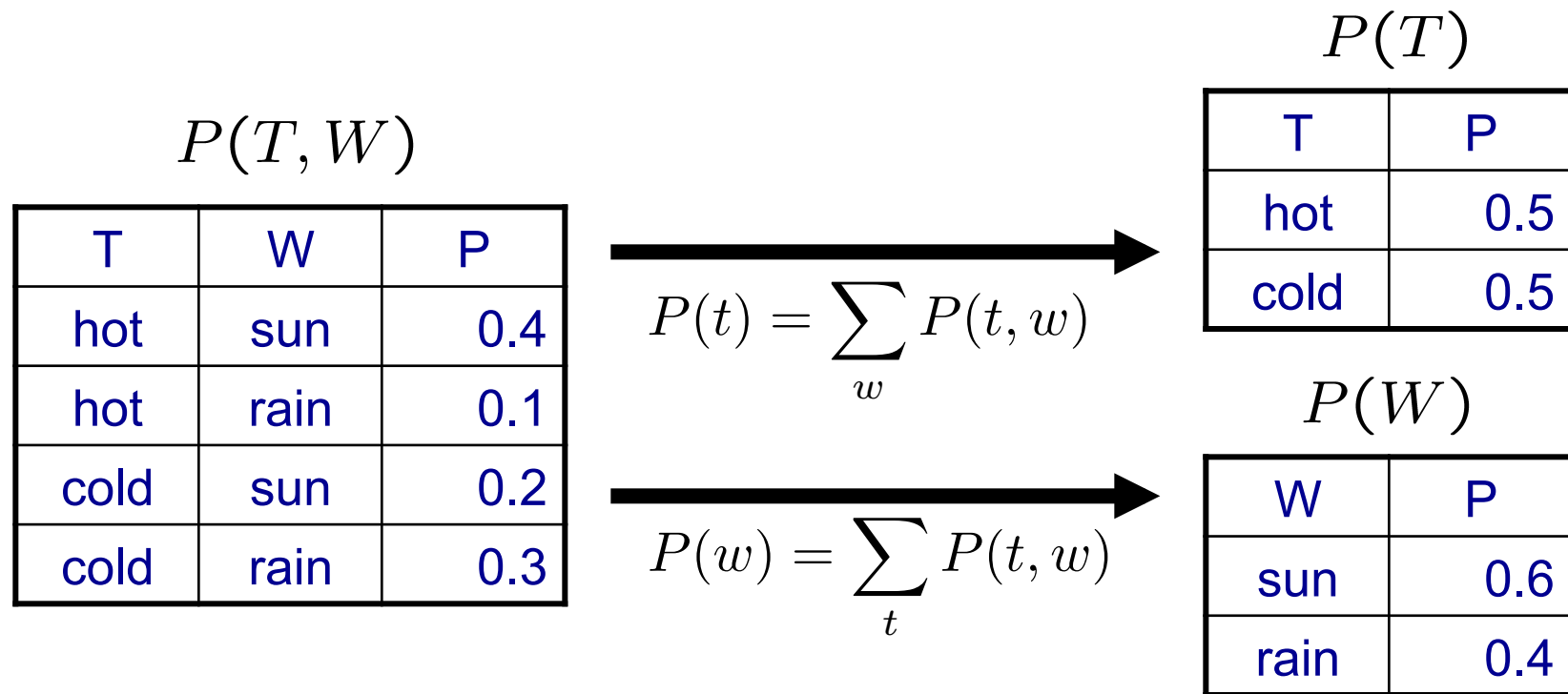
$$P(T, W)$$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

- For all but the smallest distributions, impractical to write out or estimate
 - Instead, we make additional assumptions about the distribution

Marginal Distributions

- Marginal distributions are sub-tables which eliminate variables
- Marginalization (summing out): Combine collapsed rows by adding

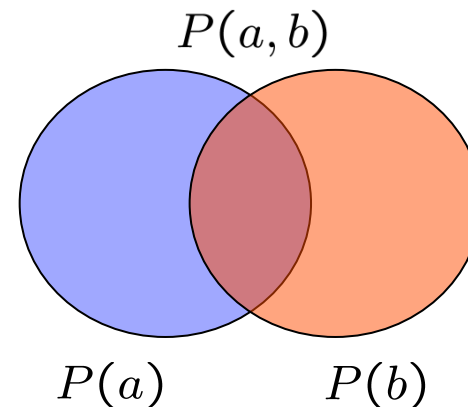


$$P(X_1 = x_1) = \sum_{x_2} P(X_1 = x_1, X_2 = x_2)$$

Conditional Probabilities

- A simple relation between joint and conditional probabilities
 - In fact, this is taken as the *definition* of a conditional probability

$$P(a|b) = \frac{P(a, b)}{P(b)}$$



$P(T, W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

$$P(W = r|T = c) = ???$$

Conditional Distributions

- Conditional distributions are probability distributions over some variables given fixed values of others

Conditional Distributions

$P(W|T)$

$P(W T = hot)$	
W	P
sun	0.8
rain	0.2

$P(W T = cold)$	
W	P
sun	0.4
rain	0.6

Joint Distribution

$P(T, W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

The Product Rule

- Sometimes have conditional distributions but want the joint

$$P(x|y) = \frac{P(x, y)}{P(y)} \quad \longleftrightarrow \quad P(x, y) = P(x|y)P(y)$$

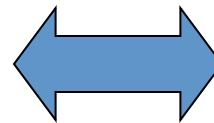
- Example:

$P(W)$

W	P
sun	0.8
rain	0.2

$P(D|W)$

D	W	P
wet	sun	0.1
dry	sun	0.9
wet	rain	0.7
dry	rain	0.3



$P(D, W)$

D	W	P
wet	sun	0.08
dry	sun	0.72
wet	rain	0.14
dry	rain	0.06

Bayes' Rule

- Two ways to factor a joint distribution over two variables:

$$P(x, y) = P(x|y)P(y) = P(y|x)P(x)$$

- Dividing, we get:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$



- Why is this at all helpful?
 - Let's us build one conditional from its reverse
 - Often one conditional is tricky but the other one is simple
 - Foundation of many practical systems (e.g. ASR, MT)
- In the running for most important ML equation!

Returning to thumbtack example...

- $P(\text{Heads}) = \theta$, $P(\text{Tails}) = 1-\theta$



- Flips are *i.i.d.*: $D = \{x_i | i=1 \dots n\}$, $P(D | \theta) = \prod_i P(x_i | \theta)$
 - Independent events
 - Identically distributed according to Bernoulli distribution
- Sequence D of α_H Heads and α_T Tails

$$P(\mathcal{D} | \theta) = \theta^{\alpha_H} (1 - \theta)^{\alpha_T}$$

Called the “likelihood” of the data under the model

Maximum Likelihood Estimation

- **Data:** Observed set D of α_H Heads and α_T Tails
- **Hypothesis:** Bernoulli distribution
- **Learning:** finding θ is an optimization problem
 - What's the objective function?

$$P(\mathcal{D} | \theta) = \theta^{\alpha_H} (1 - \theta)^{\alpha_T}$$

- **MLE:** Choose θ to maximize probability of D

$$\begin{aligned}\hat{\theta} &= \arg \max_{\theta} P(\mathcal{D} | \theta) \\ &= \arg \max_{\theta} \ln P(\mathcal{D} | \theta)\end{aligned}$$

Your first parameter learning algorithm

$$\begin{aligned}\hat{\theta} &= \arg \max_{\theta} \ln P(\mathcal{D} | \theta) \\ &= \arg \max_{\theta} \ln \theta^{\alpha_H} (1 - \theta)^{\alpha_T}\end{aligned}$$

- Set derivative to zero, and solve!

$$\frac{d}{d\theta} \ln P(\mathcal{D} | \theta) = \frac{d}{d\theta} [\ln \theta^{\alpha_H} (1 - \theta)^{\alpha_T}]$$

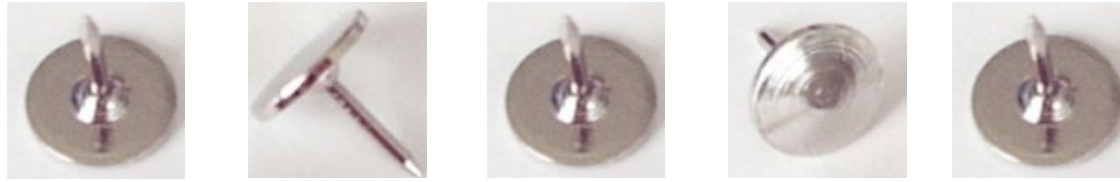
$$= \frac{d}{d\theta} [\alpha_H \ln \theta + \alpha_T \ln(1 - \theta)]$$

$$= \alpha_H \frac{d}{d\theta} \ln \theta + \alpha_T \frac{d}{d\theta} \ln(1 - \theta)$$

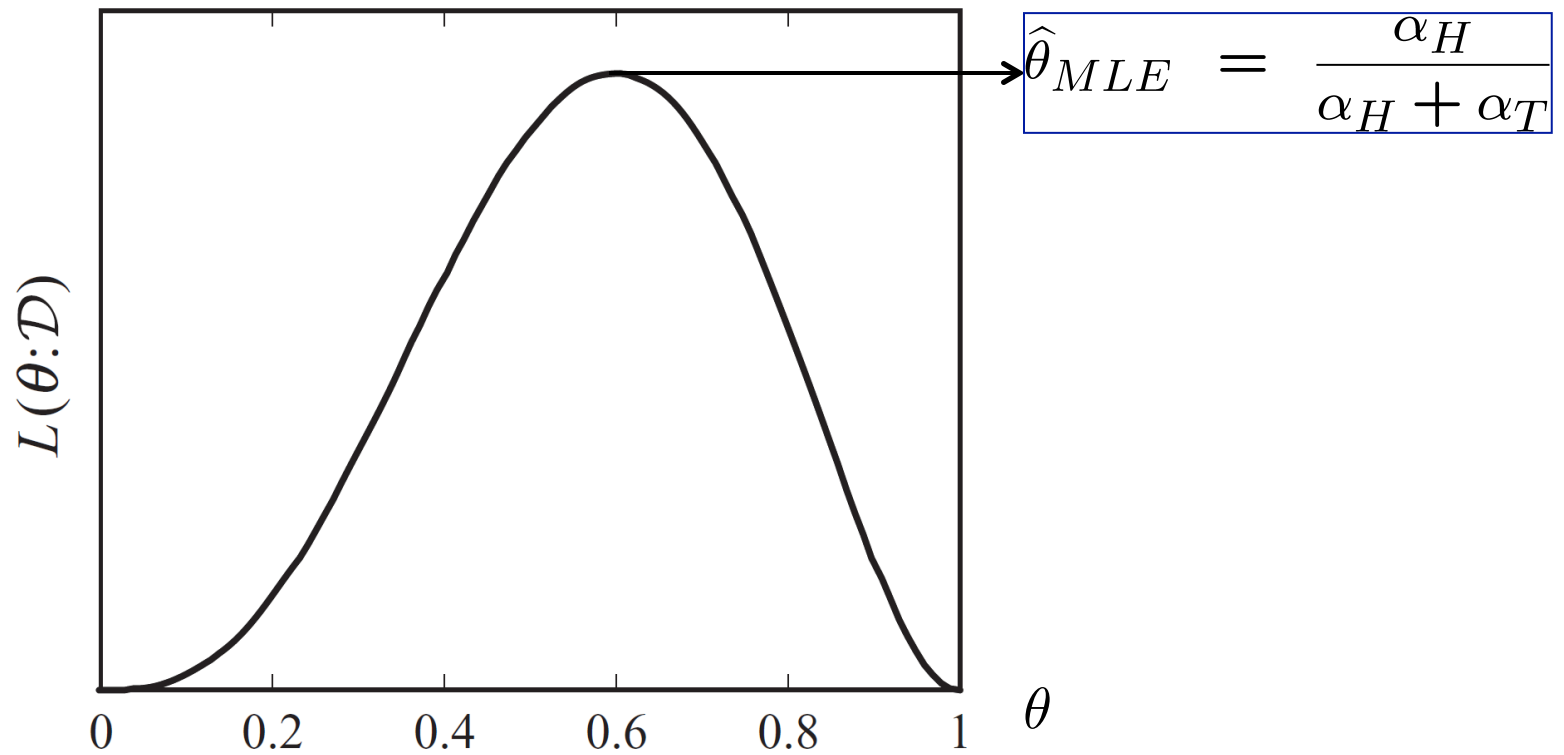
$$= \frac{\alpha_H}{\theta} - \frac{\alpha_T}{1 - \theta} = 0$$

$$\boxed{\hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}}$$

Data



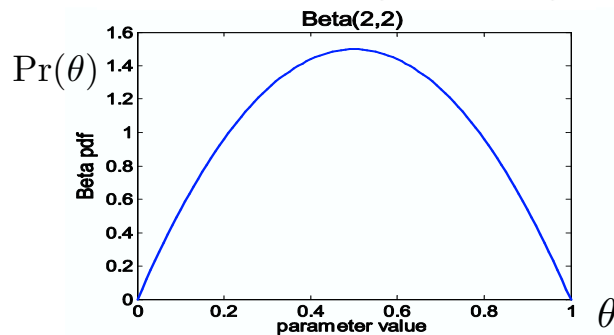
$$L(\theta; \mathcal{D}) = \ln P(\mathcal{D}|\theta)$$



What if I have prior beliefs?

- Billionaire says: Wait, I know that the thumbtack is “close” to 50-50. What can you do for me now?
- **You say: I can learn it the Bayesian way...**
- Rather than estimating a single θ , we obtain a distribution over possible values of θ

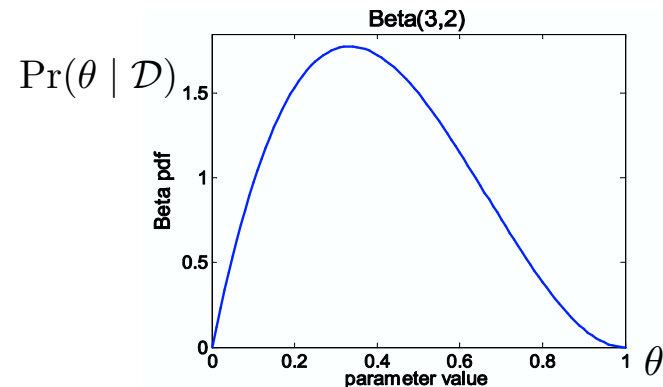
In the beginning



Observe flips
e.g.: {tails, tails}



After observations

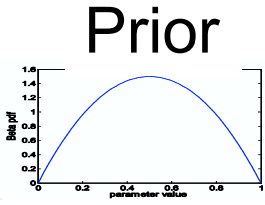


Bayesian Learning

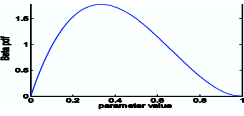
- Use Bayes' rule!

Posterior $P(\theta | \mathcal{D}) = \frac{P(\mathcal{D} | \theta)P(\theta)}{P(\mathcal{D})}$

Data Likelihood \downarrow

Prior 

Normalization \swarrow



- Or equivalently: $P(\theta | \mathcal{D}) \propto P(\mathcal{D} | \theta)P(\theta)$
- For *uniform* priors, this reduces to maximum likelihood estimation!

$$P(\theta) \propto 1 \quad P(\theta | \mathcal{D}) \propto P(\mathcal{D} | \theta)$$

Bayesian Learning for Thumbtacks

$$P(\theta | \mathcal{D}) \propto P(\mathcal{D} | \theta)P(\theta)$$

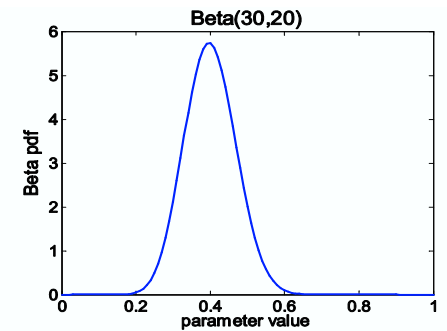
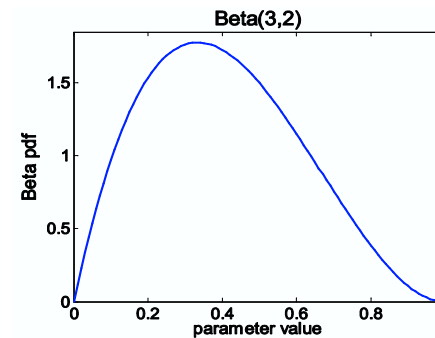
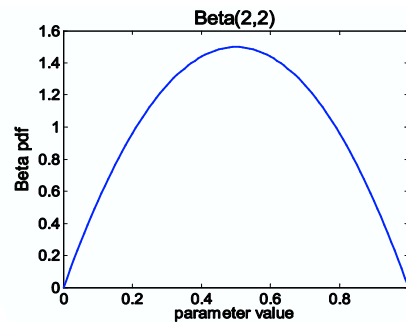
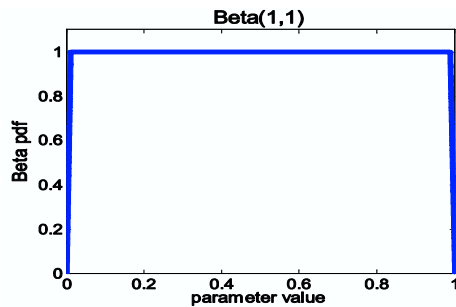
Likelihood: $P(\mathcal{D} | \theta) = \theta^{\alpha_H} (1 - \theta)^{\alpha_T}$

- What should the prior be?
 - Represent expert knowledge
 - Simple posterior form
- For binary variables, commonly used prior is the Beta distribution:

$$P(\theta) = \frac{\theta^{\beta_H-1} (1 - \theta)^{\beta_T-1}}{B(\beta_H, \beta_T)} \sim \text{Beta}(\beta_H, \beta_T)$$

Beta prior distribution – $P(\theta)$

$$P(\theta) = \frac{\theta^{\beta_H-1}(1-\theta)^{\beta_T-1}}{B(\beta_H, \beta_T)} \sim \text{Beta}(\beta_H, \beta_T)$$



- Since the Beta distribution is *conjugate* to the Bernoulli distribution, the posterior distribution has a particularly simple form:

$$P(\theta | \mathcal{D}) \propto P(\mathcal{D} | \theta)P(\theta)$$

$$\propto \theta^{\alpha_H} (1 - \theta)^{\alpha_T} \theta^{\beta_H-1} (1 - \theta)^{\beta_T-1}$$

$$= \theta^{\alpha_H + \beta_H - 1} (1 - \theta)^{\alpha_T + \beta_T - 1}$$

$$= \text{Beta}(\alpha_H + \beta_H, \alpha_T + \beta_T)$$

Using Bayesian inference for prediction

- We now have a **distribution** over parameters
- For any specific f , a function of interest, compute the expected value of f :

$$E[f(\theta)] = \int_0^1 f(\theta) P(\theta | \mathcal{D}) d\theta$$

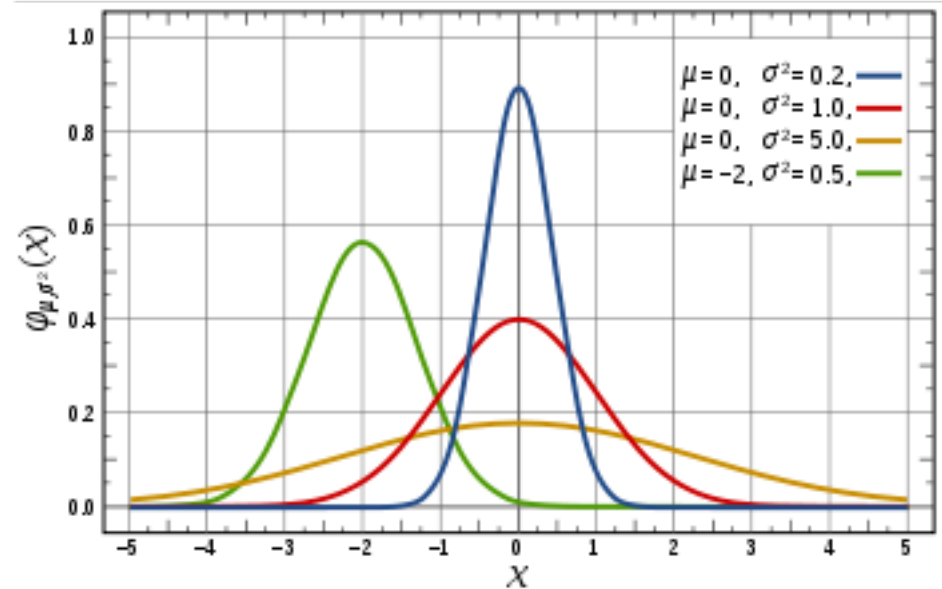
- Integral is often hard to compute
- *As more data is observed, prior is more concentrated*
- **MAP (Maximum a posteriori approximation)**: use most likely parameter to approximate the expectation

$$\hat{\theta} = \arg \max_{\theta} P(\theta | \mathcal{D})$$

$$E[f(\theta)] \approx f(\hat{\theta})$$

What about continuous variables?

- Billionaire says: If I am measuring a continuous variable, what can you do for me?
- You say: Let me tell you about Gaussians...



$$P(x \mid \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Some properties of Gaussians

- Affine transformation (multiplying by scalar and adding a constant) are Gaussian

- $X \sim N(\mu, \sigma^2)$

- $Y = aX + b \rightarrow Y \sim N(a\mu + b, a^2\sigma^2)$

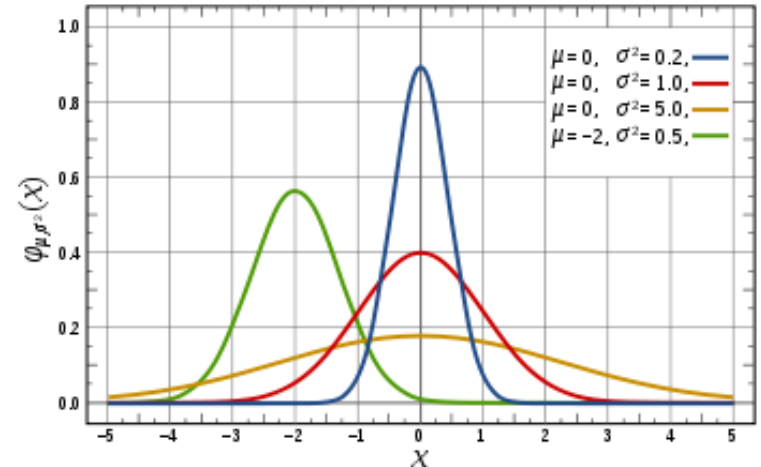
- Sum of Gaussians is Gaussian

- $X \sim N(\mu_X, \sigma_X^2)$

- $Y \sim N(\mu_Y, \sigma_Y^2)$

- $Z = X + Y \rightarrow Z \sim N(\mu_X + \mu_Y, \sigma_X^2 + \sigma_Y^2)$

- Easy to differentiate, as we will see soon!



Learning a Gaussian

- Collect a bunch of data
 - Hopefully, i.i.d. samples
 - e.g., exam scores
- Learn parameters
 - μ (“mean”)
 - σ (“variance”)

x_i $i =$	Exam Score
0	85
1	95
2	100
3	12
...	...
99	89

$$P(x \mid \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

MLE for Gaussian: $P(x | \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

- Prob. of i.i.d. samples $D=\{x_1, \dots, x_N\}$:

$$\mu_{MLE}, \sigma_{MLE} = \arg \max_{\mu, \sigma} P(\mathcal{D} | \mu, \sigma)$$

$$P(\mathcal{D} | \mu, \sigma) = \left(\frac{1}{\sigma\sqrt{2\pi}} \right)^N \prod_{i=1}^N e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- Log-likelihood of data:

$$\begin{aligned} \ln P(\mathcal{D} | \mu, \sigma) &= \ln \left[\left(\frac{1}{\sigma\sqrt{2\pi}} \right)^N \prod_{i=1}^N e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} \right] \\ &= -N \ln \sigma\sqrt{2\pi} - \sum_{i=1}^N \frac{(x_i - \mu)^2}{2\sigma^2} \end{aligned}$$

Your second learning algorithm: MLE for mean of a Gaussian

- What's MLE for mean?

$$\begin{aligned}\frac{d}{d\mu} \ln P(\mathcal{D} \mid \mu, \sigma) &= \frac{d}{d\mu} \left[-N \ln \sigma \sqrt{2\pi} - \sum_{i=1}^N \frac{(x_i - \mu)^2}{2\sigma^2} \right] \\ &= \frac{d}{d\mu} \left[-N \ln \sigma \sqrt{2\pi} \right] - \sum_{i=1}^N \frac{d}{d\mu} \left[\frac{(x_i - \mu)^2}{2\sigma^2} \right] \\ &= \sum_{i=1}^N \frac{(x_i - \mu)}{\sigma^2} = 0 \\ &= \sum_{i=1}^N x_i - N\mu = 0\end{aligned}$$

$$\hat{\mu}_{MLE} = \frac{1}{N} \sum_{i=1}^N x_i$$

MLE for variance

- Again, set derivative to zero:

$$\begin{aligned}\frac{d}{d\sigma} \ln P(\mathcal{D} \mid \mu, \sigma) &= \frac{d}{d\sigma} \left[-N \ln \sigma \sqrt{2\pi} - \sum_{i=1}^N \frac{(x_i - \mu)^2}{2\sigma^2} \right] \\ &= \frac{d}{d\sigma} \left[-N \ln \sigma \sqrt{2\pi} \right] - \sum_{i=1}^N \frac{d}{d\sigma} \left[\frac{(x_i - \mu)^2}{2\sigma^2} \right] \\ &= -\frac{N}{\sigma} + \sum_{i=1}^N \frac{(x_i - \mu)^2}{\sigma^3} = 0\end{aligned}$$

$$\hat{\sigma}_{MLE}^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \hat{\mu})^2$$

Learning Gaussian parameters

- MLE:

$$\hat{\mu}_{MLE} = \frac{1}{N} \sum_{i=1}^N x_i$$

$$\hat{\sigma}_{MLE}^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \hat{\mu})^2$$

- MLE for the variance of a Gaussian is **biased**
 - Expected result of estimation is **not** true parameter!
 - Unbiased variance estimator:

$$\hat{\sigma}_{unbiased}^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \hat{\mu})^2$$

Bayesian learning of Gaussian parameters

- Conjugate priors
 - Mean: Gaussian prior
 - Variance: Wishart Distribution

- Prior for mean:

$$P(\mu | \eta, \lambda) = \frac{1}{\lambda\sqrt{2\pi}} e^{-\frac{(\mu-\eta)^2}{2\lambda^2}}$$

Outline of lecture

- Review of probability
- Maximum likelihood estimation

2 examples of Bayesian classifiers:

- **Naïve Bayes**
- Logistic regression

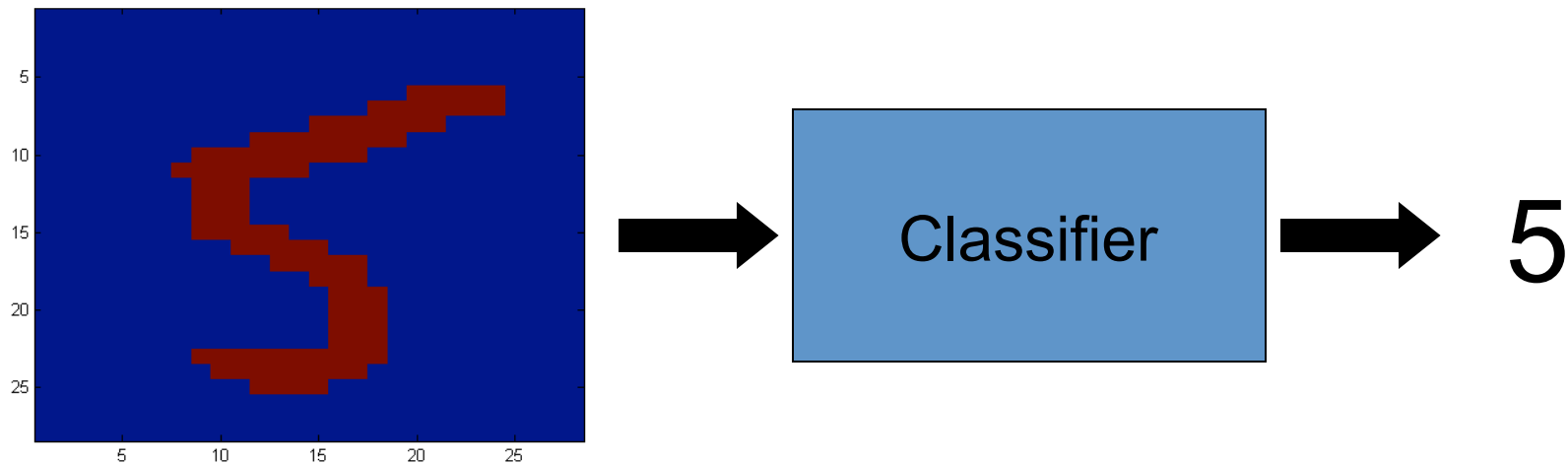
Bayesian Classification

- Problem statement:
 - Given features X_1, X_2, \dots, X_n
 - Predict a label Y

[Next several slides adapted from:
Vibhav Gogate, Jonathan Huang, Luke Zettlemoyer, Carlos
Guestrin, and Dan Weld]

Example Application

- **Digit Recognition**



- $X_1, \dots, X_n \in \{0,1\}$ (Black vs. White pixels)
- $Y \in \{0,1,2,3,4,5,6,7,8,9\}$

The Bayes Classifier

- If we had the joint distribution on $\mathbf{X}_1, \dots, \mathbf{X}_n$ and \mathbf{Y} , could predict using:

$$\arg \max_Y P(Y | X_1, \dots, X_n)$$

- (for example: what is the probability that the image represents a 5 given its pixels?)
-
- So ... How do we compute that?

The Bayes Classifier

- Use Bayes Rule!

$$P(Y|X_1, \dots, X_n) = \frac{\overset{\text{Likelihood}}{P(X_1, \dots, X_n|Y)} \overset{\text{Prior}}{P(Y)}}{\underset{\text{Normalization Constant}}{P(X_1, \dots, X_n)}}$$

- Why did this help? Well, we think that we might be able to specify how features are “generated” by the class label

The Bayes Classifier

- Let's expand this for our digit recognition task:

$$P(Y = 5|X_1, \dots, X_n) = \frac{P(X_1, \dots, X_n|Y = 5)P(Y = 5)}{P(X_1, \dots, X_n|Y = 5)P(Y = 5) + P(X_1, \dots, X_n|Y = 6)P(Y = 6)}$$
$$P(Y = 6|X_1, \dots, X_n) = \frac{P(X_1, \dots, X_n|Y = 6)P(Y = 6)}{P(X_1, \dots, X_n|Y = 5)P(Y = 5) + P(X_1, \dots, X_n|Y = 6)P(Y = 6)}$$

- To classify, we'll simply compute these probabilities, one per class, and predict based on which one is largest

Model Parameters

- How many parameters are required to specify the likelihood, $P(X_1, \dots, X_n | Y)$?
 - (Supposing that each image is 30x30 pixels)
- The problem with explicitly modeling $P(X_1, \dots, X_n | Y)$ is that there are usually way too many parameters:
 - We'll run out of space
 - We'll run out of time
 - And we'll need tons of training data (which is usually not available)

Naïve Bayes

- Naïve Bayes assumption:
 - Features are independent given class:

$$\begin{aligned}P(X_1, X_2|Y) &= P(X_1|X_2, Y)P(X_2|Y) \\ &= P(X_1|Y)P(X_2|Y)\end{aligned}$$

- More generally:

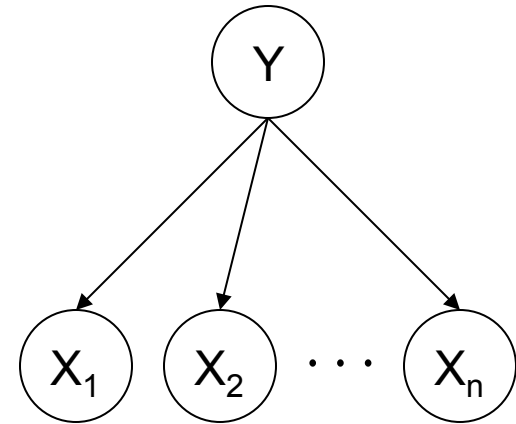
$$P(X_1 \dots X_n | Y) = \prod_i P(X_i | Y)$$

- How many parameters now?
 - Suppose \mathbf{X} is composed of n binary features

The Naïve Bayes Classifier

- Given:

- Prior $P(Y)$
- n conditionally independent features \mathbf{X} given the class Y
- For each X_i , we have likelihood $P(X_i | Y)$



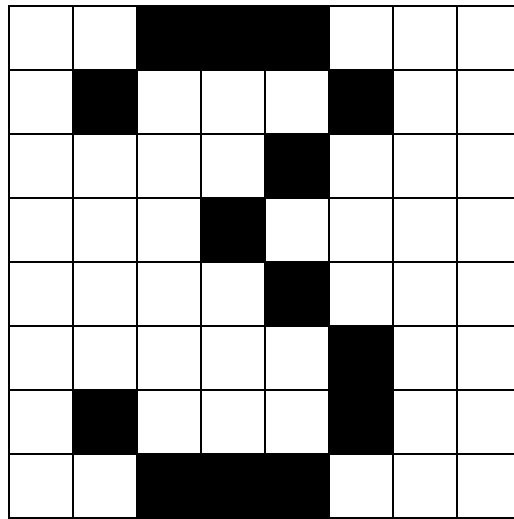
- Decision rule:

$$\begin{aligned} y^* = h_{NB}(\mathbf{x}) &= \arg \max_y P(y) P(x_1, \dots, x_n | y) \\ &= \arg \max_y P(y) \prod_i P(x_i | y) \end{aligned}$$

If certain assumption holds, NB is optimal classifier!
(they typically don't)

A Digit Recognizer

- Input: pixel grids



- Output: a digit 0-9

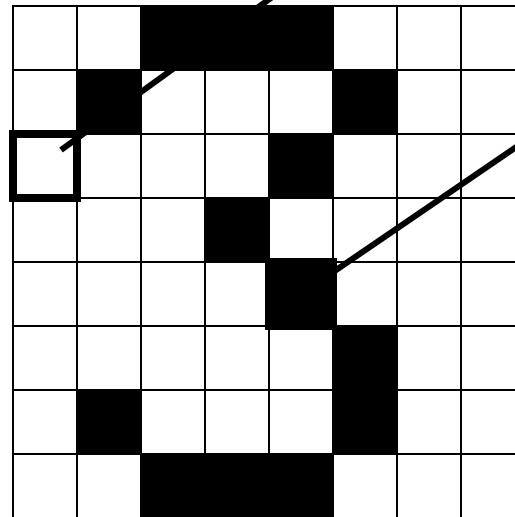
Are the naïve Bayes assumptions realistic here?



What has to be learned?

$P(Y)$

1	0.1
2	0.1
3	0.1
4	0.1
5	0.1
6	0.1
7	0.1
8	0.1
9	0.1
0	0.1



$P(F_{3,1} = on|Y)$ $P(F_{5,5} = on|Y)$

1	0.01
2	0.05
3	0.05
4	0.30
5	0.80
6	0.90
7	0.05
8	0.60
9	0.50
0	0.80

1	0.05
2	0.01
3	0.90
4	0.80
5	0.90
6	0.90
7	0.25
8	0.85
9	0.60
0	0.80

MLE for the parameters of NB

- Given dataset
 - $\text{Count}(A=a, B=b) \leftarrow$ number of examples where $A=a$ and $B=b$
- MLE for discrete NB, simply:
 - Prior:

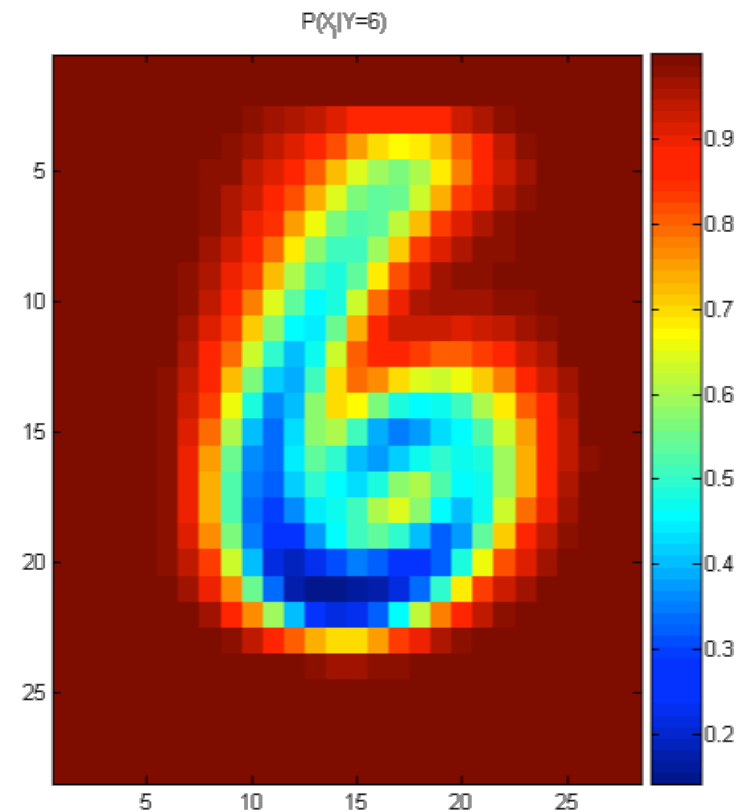
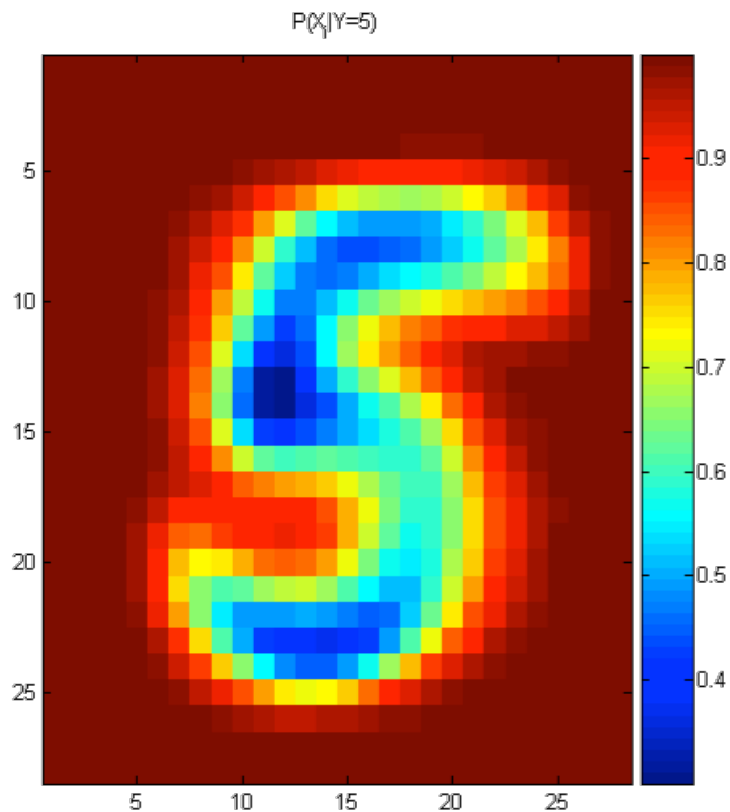
$$P(Y = y) = \frac{\text{Count}(Y = y)}{\sum_{y'} \text{Count}(Y = y')}$$

- Observation distribution:

$$P(X_i = x | Y = y) = \frac{\text{Count}(X_i = x, Y = y)}{\sum_{x'} \text{Count}(X_i = x', Y = y)}$$

MLE for the parameters of NB

- Training amounts to, for each of the classes, averaging all of the examples together:



MAP estimation for NB

- Given dataset
 - $\text{Count}(A=a, B=b) \leftarrow$ number of examples where $A=a$ and $B=b$
- MAP estimation for discrete NB, simply:
 - Prior:

$$P(Y = y) = \frac{\text{Count}(Y = y)}{\sum_{y'} \text{Count}(Y = y')}$$

- Observation distribution:

$$P(X_i = x | Y = y) = \frac{\text{Count}(X_i = x, Y = y) + \mathbf{a}}{\sum_{x'} \text{Count}(X_i = x', Y = y) + |\mathbf{X}_i| * \mathbf{a}}$$

- Called “smoothing”. Corresponds to Dirichlet prior!

What about if there is missing data?

- One of the key strengths of Bayesian approaches is that they can naturally handle missing data
- Suppose don't have value for some attribute X_i
 - applicant's credit history unknown
 - some medical test not performed on patient
 - how to compute $P(X_1=x_1 \dots X_j=? \dots X_d=x_d | y)$

- Easy with Naïve Bayes

- ignore attribute in instance where its value is missing
- compute likelihood based on observed attributes
- no need to “fill in” or explicitly model missing values
- based on conditional independence between attributes

$$P(x_1 \dots X_j \dots x_d | y) = \prod_{i \neq j}^d P(x_i | y)$$

Naive Bayes = Linear Classifier

- **Theorem:** assume that $x_i \in \{0, 1\}$ for all $i \in [1, N]$.
Then, the Naive Bayes classifier is defined by

$$\mathbf{x} \mapsto \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b),$$

Outline of lecture

- Review of probability
- Maximum likelihood estimation

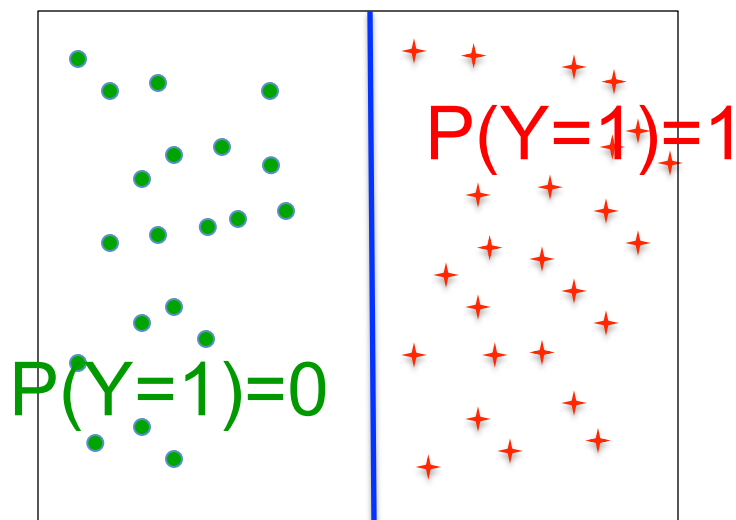
2 examples of Bayesian classifiers:

- Naïve Bayes
- **Logistic regression**

[Next several slides adapted from:
Vibhav Gogate, Luke Zettlemoyer, Carlos Guestrin, and Dan Weld]

Logistic Regression

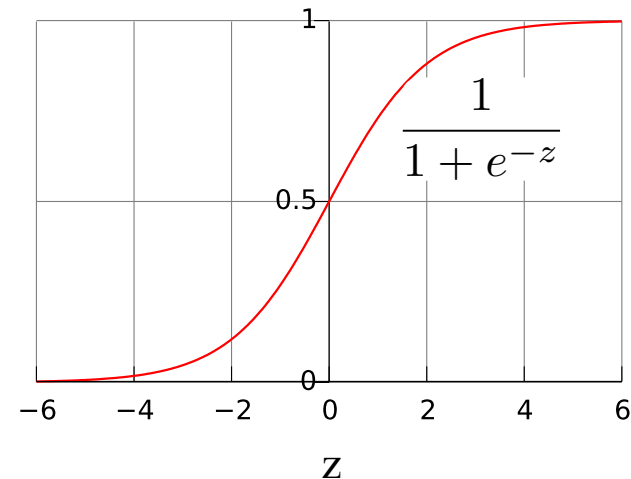
- Learn $P(Y|\mathbf{X})$ directly!
 - Assume a particular functional form
 - ★ Linear classifier? On one side we say $P(Y=1|X)=1$, and on the other $P(Y=1|X)=0$
 - ★ ***But, this is not differentiable (hard to learn)... doesn't allow for label noise...***



Logistic Regression

Logistic function (Sigmoid):

- Learn $P(Y|X)$ directly!
 - Assume a particular functional form
 - Sigmoid applied to a linear function of the data:



$$P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

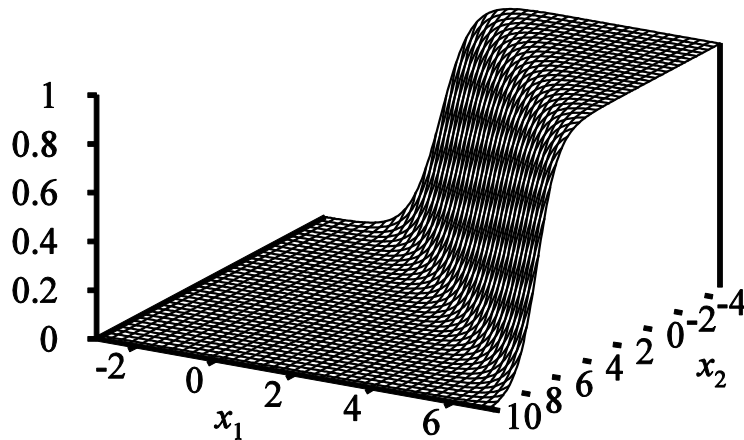
$$P(Y = 0|X) = \frac{\exp(w_0 + \sum_{i=1}^n w_i X_i)}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

Features can be discrete or continuous!

Logistic Function in n Dimensions

$$P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

Sigmoid applied to a linear function of the data:



Features can be discrete or continuous!

Logistic Regression: decision boundary

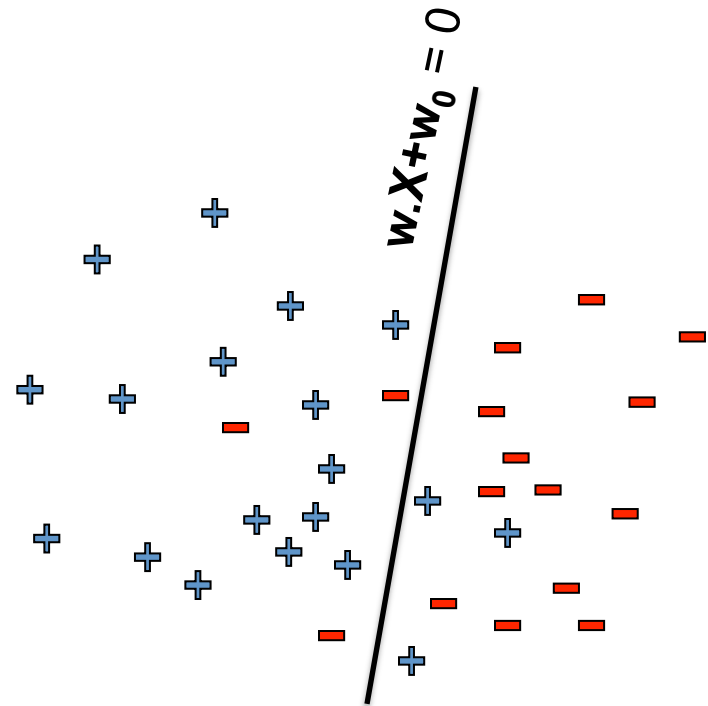
$$P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i)} \quad P(Y = 0|X) = \frac{\exp(w_0 + \sum_{i=1}^n w_i X_i)}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

- **Prediction:** Output the Y with highest $P(Y|X)$
 - For binary Y, output $Y=0$ if

$$1 < \frac{P(Y = 0|X)}{P(Y = 1|X)}$$

$$1 < \exp(w_0 + \sum_{i=1}^n w_i X_i)$$

$$0 < w_0 + \sum_{i=1}^n w_i X_i$$



A Linear Classifier!

Likelihood vs. Conditional Likelihood

Generative (Naïve Bayes) maximizes **Data likelihood**

$$\begin{aligned}\ln P(\mathcal{D} | \mathbf{w}) &= \sum_{j=1}^N \ln P(\mathbf{x}^j, y^j | \mathbf{w}) \\ &= \sum_{j=1}^N \ln P(y^j | \mathbf{x}^j, \mathbf{w}) + \sum_{j=1}^N \ln P(\mathbf{x}^j | \mathbf{w})\end{aligned}$$

Discriminative (Logistic Regr.) maximizes **Conditional Data Likelihood**

$$\ln P(\mathcal{D}_Y | \mathcal{D}_X, \mathbf{w}) = \sum_{j=1}^N \ln P(y^j | \mathbf{x}^j, \mathbf{w})$$

Focuses only on learning $P(Y | \mathbf{X})$ - all that matters for classification

Maximizing Conditional Log Likelihood

$$l(\mathbf{w}) \equiv \ln \prod_j P(y^j | \mathbf{x}^j, \mathbf{w})$$

$$= \sum_j y^j (w_0 + \sum_i w_i x_i^j) - \ln(1 + \exp(w_0 + \sum_i w_i x_i^j))$$

0 or 1!

$$P(Y = 0 | X, W) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

$$P(Y = 1 | X, W) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

Bad news: no closed-form solution to maximize $l(\mathbf{w})$

Good news: $l(\mathbf{w})$ is concave function of $\mathbf{w} \rightarrow$

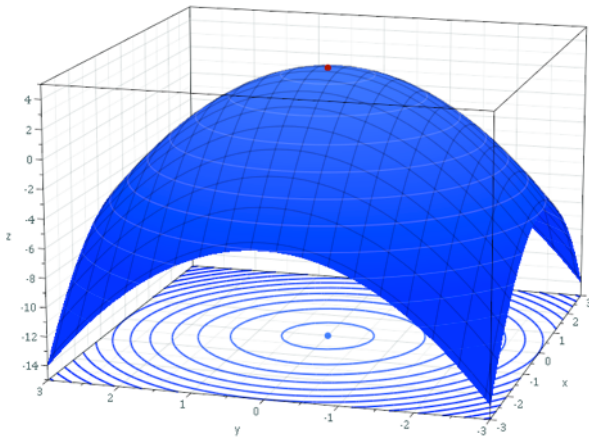
No local maxima

Concave functions easy to optimize

Optimizing concave function – Gradient ascent

- Conditional likelihood for Logistic Regression is concave →

Gradient: $\nabla_{\mathbf{w}} l(\mathbf{w}) = \left[\frac{\partial l(\mathbf{w})}{\partial w_0}, \dots, \frac{\partial l(\mathbf{w})}{\partial w_n} \right]'$



Learning rate, $\eta > 0$

Update rule: $\Delta \mathbf{w} = \eta \nabla_{\mathbf{w}} l(\mathbf{w})$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \frac{\partial l(\mathbf{w})}{\partial w_i}$$

Maximize Conditional Log Likelihood: Gradient ascent

$$P(Y = 1|X, W) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

$$l(\mathbf{w}) = \sum_j y^j (w_0 + \sum_i w_i x_i^j) - \ln(1 + \exp(w_0 + \sum_i w_i x_i^j))$$

$$\frac{\partial l(w)}{\partial w_i} = \sum_j \left[\frac{\partial}{\partial w_i} y^j (w_0 + \sum_i w_i x_i^j) - \frac{\partial}{\partial w_i} \ln \left(1 + \exp(w_0 + \sum_i w_i x_i^j) \right) \right]$$

$$= \sum_j \left[y^j x_i^j - \frac{x_i^j \exp(w_0 + \sum_i w_i x_i^j)}{1 + \exp(w_0 + \sum_i w_i x_i^j)} \right]$$

$$= \sum_j x_i^j \left[y^j - \frac{\exp(w_0 + \sum_i w_i x_i^j)}{1 + \exp(w_0 + \sum_i w_i x_i^j)} \right]$$

$$\frac{\partial l(w)}{\partial w_i} = \sum_j x_i^j (y^j - P(Y^j = 1|x^j, w))$$

Gradient Ascent for LR

Gradient ascent algorithm: (learning rate $\eta > 0$)

do:

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

For $i=1$ to n : (iterate over features)

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

until “change” $< \epsilon$

Loop over training examples!



Naïve Bayes vs. Logistic Regression

Learning: $h: X \mapsto Y$

X – features

Y – target classes

Generative

- Assume functional form for
 - $P(X|Y)$ **assume cond indep**
 - $P(Y)$
 - Est. params from train data
- Gaussian NB for cont. features
- Bayes rule to calc. $P(Y|X=x)$:
 - $P(Y | \mathbf{X}) \propto P(\mathbf{X} | Y) P(Y)$
- **Indirect** computation
 - Can generate a sample of the data
 - **Can easily handle missing data**

Discriminative

- Assume functional form for
 - $P(Y|X)$ **no assumptions**
 - Est params from training data
- **Handles discrete & cont features**
- **Directly calculate $P(Y|X=x)$**
 - Can't generate data sample

Naïve Bayes vs. Logistic Regression

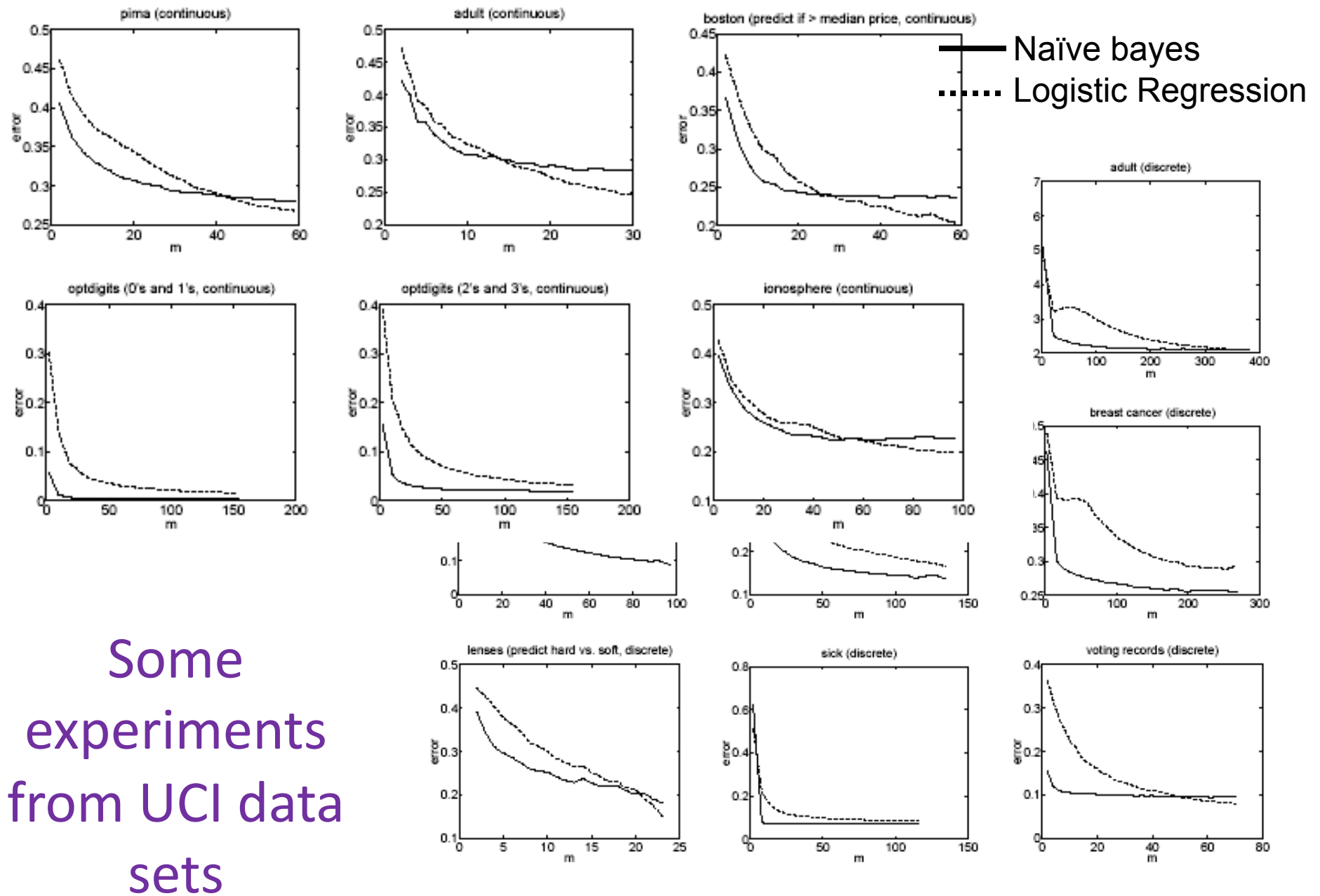
[Ng & Jordan, 2002]

- Generative vs. Discriminative classifiers
- Asymptotic comparison
(# training examples \rightarrow infinity)
 - when model correct
 - NB, Linear Discriminant Analysis (with class independent variances), and Logistic Regression produce identical classifiers
 - when model incorrect
 - LR is less biased – does not assume conditional independence
 - **therefore LR expected to outperform NB**

Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

- Generative vs. Discriminative classifiers
- Non-asymptotic analysis
 - convergence rate of parameter estimates,
(**n = # of attributes in X**)
 - Size of training data to get close to infinite data solution
 - Naïve Bayes needs $O(\log n)$ samples
 - Logistic Regression needs $O(n)$ samples
 - Naïve Bayes converges more quickly to its (*perhaps less helpful*) asymptotic estimates



Some
 experiments
 from UCI data
 sets

Figure 1: Results of 15 experiments on datasets from the UCI Machine Learning repository. Plots are of generalization error vs. m (averaged over 1000 random train/test splits). Dashed line is logistic regression; solid line is naive Bayes.