CS 4375
Midterm Review: Part II

Rishabh Iyer
University of Texas at Dallas

Topics for the Midterm Exam

- Linear Regression
- Perceptron
- Support Vector Machines
- Nearest Neighbor Methods
- Decision Trees
- Bayesian Methods and Parameter Estimation
- Naïve Bayes
- Logistic Regression

Topics for the Midterm Exam

- Linear Regression
- Perceptron
- Support Vector Machines
- Nearest Neighbor Methods
- Decision Trees
- **Bayesian Methods and Parameter Estimation**
- Naïve Bayes
- Logistic Regression

Maximum Likelihood Estimation (MLE)

- **Data:** Observed set of α_H heads and α_T tails
- **Hypothesis:** Coin flips follow a Bernoulli distribution
- **Learning:** Find the "best" θ
- **MLE:** Choose θ to maximize probability of D given θ

$$
\begin{aligned}
\widehat{\theta} &= \arg\max_{\theta} \; P(\mathcal{D} \mid \theta) \\
&= \arg\max_{\theta} \; \ln P(\mathcal{D} \mid \theta)
\end{aligned}
$$

Coin Flipping – Binomial Distribution

- $P(Heads) = \theta,\ P(Tails) = 1 - \theta$
- Flips are i.i.d.
  - Independent events
  - Identically distributed according to Binomial distribution
- Our training data consists of α_H heads and α_T tails

$$p(D|\theta) = \theta^{\alpha_H} \cdot (1-\theta)^{\alpha_T}$$

First Parameter Learning Algorithm

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

Set derivative to zero, and solve!

$$\begin{aligned}\frac{d}{d\theta} \ln P(\mathcal{D} \mid \theta) &= \frac{d}{d\theta}\left[\ln \theta^{\alpha_H}(1-\theta)^{\alpha_T}\right] \\ &= \frac{d}{d\theta}\left[\alpha_H \ln \theta + \alpha_T \ln(1-\theta)\right] \\ &= \alpha_H \frac{d}{d\theta} \ln \theta + \alpha_T \frac{d}{d\theta} \ln(1-\theta) \\ &= \frac{\alpha_H}{\theta} - \frac{\alpha_T}{1-\theta} \quad = 0\end{aligned}$$

First Parameter Learning Algorithm

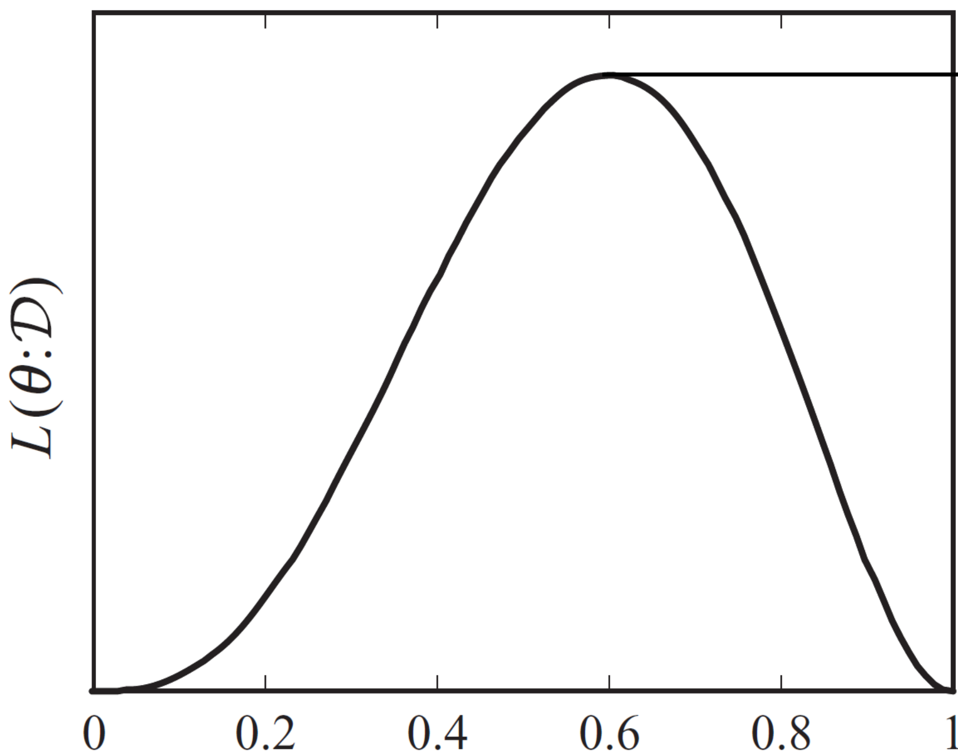
$$\begin{aligned}\widehat{\theta} &= \arg\max_{\theta} \ \ln P(\mathcal{D} \mid \theta) \\ &= \arg\max_{\theta} \ \ln \theta^{\alpha_H}(1-\theta)^{\alpha_T}\end{aligned}$$

Set derivative to zero, and solve!

$$\begin{aligned}\frac{d}{d\theta} \ln P(\mathcal{D} \mid \theta) &= \frac{d}{d\theta}\left[\ln \theta^{\alpha_H}(1-\theta)^{\alpha_T}\right] \\ &= \frac{d}{d\theta}\left[\alpha_H \ln \theta + \alpha_T \ln(1-\theta)\right] \\ &= \alpha_H \frac{d}{d\theta} \ln \theta + \alpha_T \frac{d}{d\theta} \ln(1-\theta) \\ &= \frac{\alpha_H}{\theta} - \frac{\alpha_T}{1-\theta} \quad = 0\end{aligned}$$

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

Coin Flip MLE



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

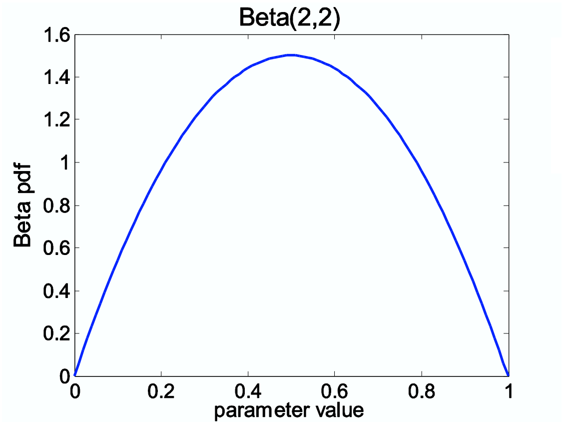
Priors

- Priors are a Bayesian mechanism that allow us to take into account "prior" knowledge about our belief in the outcome

- Rather than estimating a single θ, consider a distribution over possible values of θ given the data

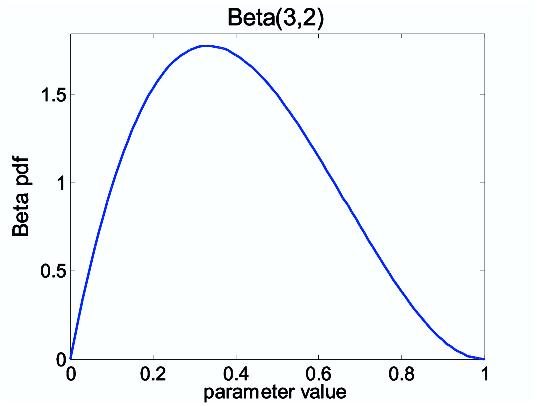
  - Update our prior after seeing data

Our best guess in the absence of any data



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

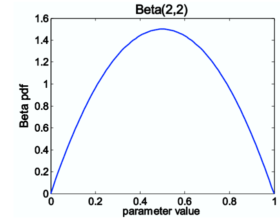
Our estimate after we see some data



Bayesian Learning

Apply Bayes rule:

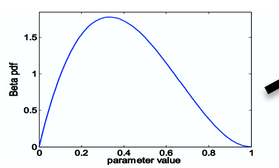
Prior



Data Likelihood

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}$$

Posterior



Normalization

- Or equivalently: $p(\theta|D) \propto p(D|\theta)p(\theta)$
- For uniform priors this reduces to the MLE objective

$$p(\theta) \propto 1 \qquad \Rightarrow \qquad p(\theta|D) \propto p(D|\theta)$$

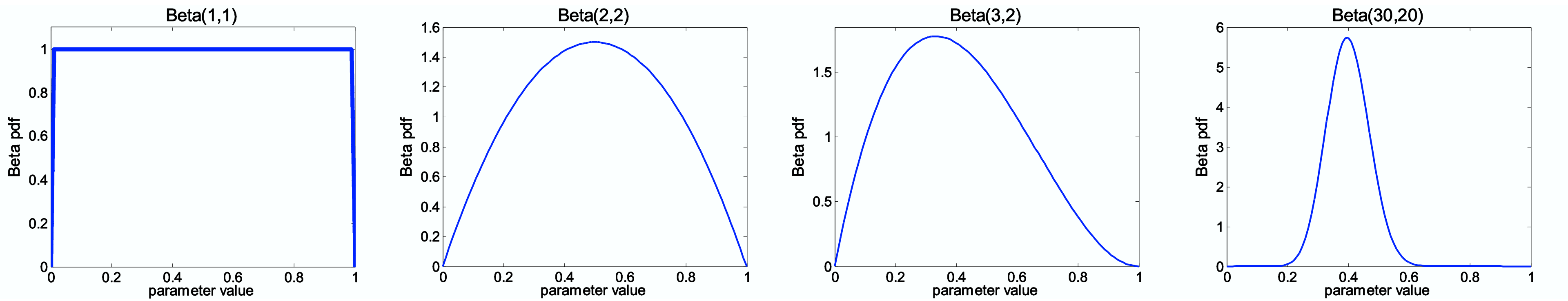
Coin Flips with Beta Distribution

Likelihood function:

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

Prior:

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



$$P(\theta \mid \mathcal{D}) \propto \theta^{\alpha_H}(1-\theta)^{\alpha_T} \; \theta^{\beta_H - 1}(1-\theta)^{\beta_T - 1}$$

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

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

MAP Estimation

- Choosing θ to maximize the posterior distribution is called maximum a posteriori (MAP) estimation

$$\theta_{MAP} = \arg\max_{\theta} p(\theta|D)$$

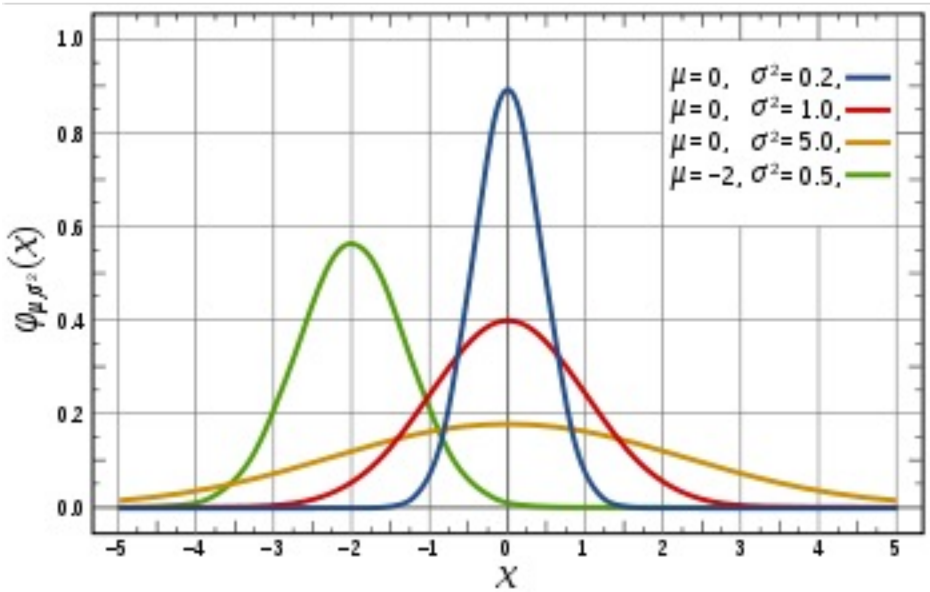
- The only difference between θ_{MLE} and θ_{MAP} is that one assumes a uniform prior (MLE) and the other allows an arbitrary prior

MAP for the Coin Flip Model

- Suppose we have 5 coin flips all of which are heads
  - MLE would give $\theta_{MLE} = 1$
  - MLE with a $Beta(2{,}2)$ prior gives $\theta_{MAP} = \frac{6}{7} \approx .857$
  - As we see more data, the effect of the prior diminishes
    - $\theta_{MAP} = \frac{\alpha_H + \beta_H - 1}{\alpha_H + \beta_H + \alpha_T + \beta_T - 2} \approx \frac{\alpha_H}{\alpha_H + \alpha_T}$ for large # of observations

MLE for Gaussian Distributions

- Two parameter distribution characterized by a mean and a variance



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

Learning a Gaussian

- Collect data
  - Hopefully, i.i.d. samples
  - e.g., exam scores
- Learn parameters
  - Mean: μ
  - Variance: σ

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:

- Probability of N i.i.d. samples $D = x^{(1)}, \dots, x^{(N)}$

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

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

- Log-likelihood of the data

$$\ln p(D|\mu,\sigma) = -\frac{N}{2}\ln 2\pi\sigma^2 - \sum_{i=1}^{N} \frac{\left(x^{(i)}-\mu\right)^2}{2\sigma^2}$$

MLE for the Mean of a Gaussian

$$\frac{\partial}{\partial\mu}\ln p(D|\mu,\sigma) = \frac{\partial}{\partial\mu}\left[-\frac{N}{2}\ln 2\pi\sigma^2 - \sum_{i=1}^{N}\frac{\left(x^{(i)}-\mu\right)^2}{2\sigma^2}\right]$$

$$= \frac{\partial}{\partial\mu}\left[-\sum_{i=1}^{N}\frac{\left(x^{(i)}-\mu\right)^2}{2\sigma^2}\right]$$

$$= \sum_{i=1}^{N}\frac{\left(x^{(i)}-\mu\right)}{\sigma^2}$$

$$= \frac{\left[N\mu - \sum_{i=1}^{N} x^{(i)}\right]}{\sigma^2} = 0$$

$$\boxed{\mu_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x^{(i)}}$$

MLE for Variance

$$\frac{\partial}{\partial \sigma} \ln p(D|\mu,\sigma) = \frac{\partial}{\partial \sigma}\left[-\frac{N}{2}\ln 2\pi\sigma^2 - \sum_{i=1}^{N}\frac{\left(x^{(i)}-\mu\right)^2}{2\sigma^2}\right]$$

$$= -\frac{N}{\sigma} + \frac{\partial}{\partial \sigma}\left[-\sum_{i=1}^{N}\frac{\left(x^{(i)}-\mu\right)^2}{2\sigma^2}\right]$$

$$= -\frac{N}{\sigma} + \sum_{i=1}^{N}\frac{\left(x^{(i)}-\mu\right)^2}{\sigma^3} = 0$$

$$\boxed{\sigma_{MLE}^2 = \frac{1}{N}\sum_{i=1}^{N}\left(x^{(i)}-\mu_{MLE}\right)^2}$$

Topics for the Midterm Exam

- Linear Regression
- Perceptron
- Support Vector Machines
- Nearest Neighbor Methods
- Decision Trees
- Bayesian Methods and Parameter Estimation
- **Naïve Bayes**
- Logistic Regression

Bayesian Categorization/Classification

- Given features $x = (x_1, \ldots, x_m)$ predict a label y
- If we had a joint distribution over x and y, given x we could find the label using MAP inference

$$\arg\max_y p(y|x_1, \ldots, x_m)$$

- Can compute this in exactly the same way that we did before using Bayes rule:

$$p(y|x_1, \ldots, x_m) = \frac{p(x_1, \ldots, x_m|y)p(y)}{p(x_1, \ldots, x_m)}$$

Bag of Words

aardvark	0
about	2
all	2
Africa	1
apple	0
anxious	0
...	
gas	1
...	
oil	1
…	
Zaire	0

Naïve Bayes

- Naïve Bayes assumption
  - Features are independent given class label

$$p(x_1, x_2 | y) = p(x_1|y)\, p(x_2|y)$$

  - More generally

$$p(x_1, \ldots, x_m | y) = \prod_{i=1}^{m} p(x_i | y)$$

- How many parameters now?
  - Suppose x is composed of d binary features

Naïve Bayes

- Naïve Bayes assumption
  - Features are independent given class label

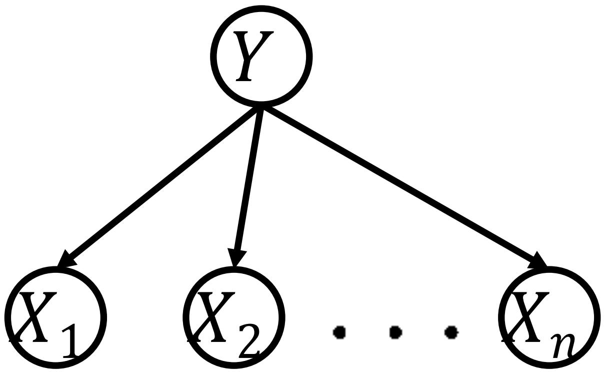
$$p(x_1, x_2 | y) = p(x_1 | y)\, p(x_2 | y)$$

  - More generally

$$p(x_1, \ldots, x_m | y) = \prod_{i=1}^{m} p(x_i | y)$$

- How many parameters now?
  - Suppose x composed of d binary features $\Rightarrow O(d \cdot L)$ where L is the number of class labels

The Naïve Bayes Classifier



- **Given**
  - Prior $p(y)$
  - m conditionally independent features X given the class Y
  - For each X_i, we have likelihood $P(X_i|Y)$
- Classify via

$$y^* = h_{NB}(x) = \arg\max_y p(y) p(x_1, \dots, x_m|y)$$

$$= \arg\max_y p(y) \prod_i^m p(x_i|y)$$

MLE for the Parameters of NB

- Given dataset, count occurrences for all pairs
  - $Count(X_i = x_i, Y = y)$ is the number of samples in which $X_i = x_i$ and $Y = y$
- MLE for discrete NB

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

$$p(X_i = x_i | Y = y) = \frac{Count(X_i = x_i, Y = y)}{\sum_{x_i'} Count\,(X_i = x_i', Y = y)}$$

See this link for more insights: http://www.datasciencecourse.org/notes/mle/

NB and MAP: Laplace Smoothing

- To fix this, use a prior!
  - Already saw how to do this in the coin-flipping example using the Beta distribution
  - For NB over discrete spaces, can use the Dirichlet prior
  - The Dirichlet distribution is a distribution over $z_1, \ldots, z_k \in (0,1)$ such that $z_1 + \cdots + z_k = 1$ characterized by k parameters $\alpha_1, \ldots, \alpha_k$

$$f(z_1, \ldots, z_k; \alpha_1, \ldots, \alpha_k) \propto \prod_{i=1}^{k} z_i^{\alpha_i - 1}$$

    - Called smoothing, what are the MLE estimates under these kinds of priors?

Continuous Naïve Bayes

- Continuous Naïve Bayes, also known as Guassian Naïve Bayes is where the features are continuous
- The distribution $p(X_i = x_i \mid Y = y) = N\left(x_i, \mu_y, \sigma_y^2\right)$
- In other words, the conditional distribution of each feature given the class is a Guassian distribution with mean μ_y and variance σ_y^2
- We can use the Naïve Bayes assumption and assume:

$$p(x_1, \ldots, x_m|y) = \prod_{i=1}^{m} p(x_i|y)$$

- The distribution of labels is the same as the multinomial case

Parameter Estimation of Cont. NB

- The parameter estimation can similarly be obtained using the Maximum Likelihood Estimation
- The mean and variance can be estimated as the standard Gaussian distribution except that we restrict to each label

$$\mu_y = \frac{\sum_{j=1}^{m} x_i^{(j)} 1\{y^{(j)} = y\}}{\sum_{j=1}^{m} 1\{y^{(j)} = y\}},$$

$$\sigma_y^2 = \frac{\sum_{j=1}^{m} (x_i^{(j)} - \mu_y)^2 1\{y^{(j)} = y\}}{\sum_{j=1}^{m} 1\{y^{(j)} = y\}}$$

Parameter Estimation of Cont. NB

- Finally, we need to estimate $p(y)$
- This is like the discrete Naïve Bayes case:

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

- We can classify a test example in a similar way to discrete NB:

$$y^* = h_{NB}(x) = \arg\max_y p(y) p(x_1, \ldots, x_m | y)$$
$$= \arg\max_y p(y) \prod_i^m p(x_i | y)$$

- Here $p(x_i \, | y) = \mathrm{N}\left(\mathrm{x_i}, \mu_y, \sigma_y^2\right)$

Summary of Naïve Bayes Models

- Two kinds of Naïve Bayes: Discrete and Continuous
- Learning is often very simple
  - Using counts (discrete NB) or mean/variance (cont. NB), obtain estimates for $p(x_i \mid y)$
  - Using counts, obtain estimates for $p(y)$
- At inference time, we classify based on:

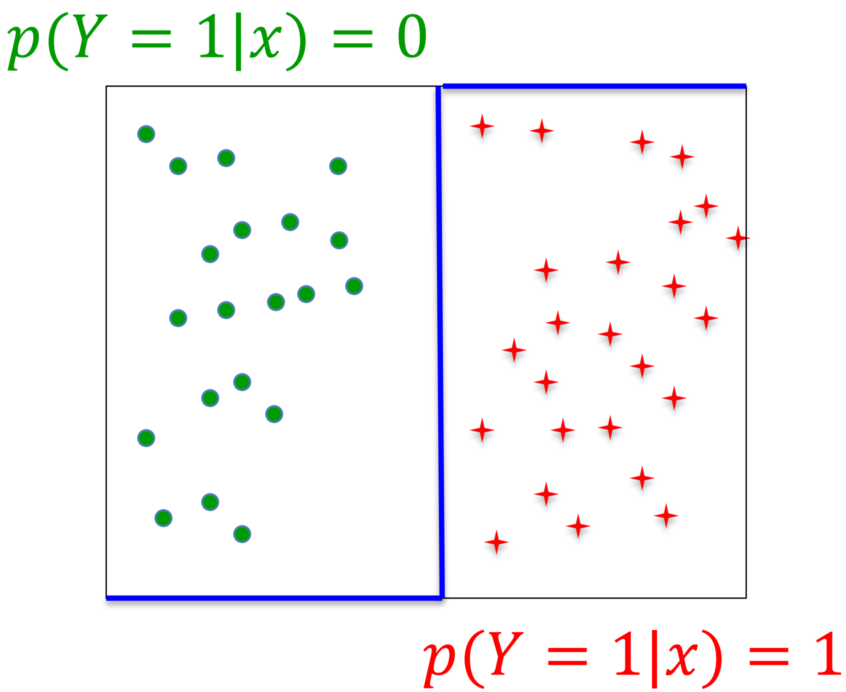
$$
\begin{aligned}
y^* = h_{NB}(x) = &\ \arg\max_y p(y) p(x_1, \ldots, x_m \mid y) \\
= &\ \arg\max_y p(y) \prod_i^m p(x_i \mid y)
\end{aligned}
$$

Topics for the Midterm Exam

- Linear Regression
- Perceptron
- Support Vector Machines
- Nearest Neighbor Methods
- Decision Trees
- Bayesian Methods and Parameter Estimation
- Naïve Bayes
- **Logistic Regression**

Ideal 0/1 Probability

- Learn $p(Y|X)$ directly from the data
  - Assume a particular functional form, e.g., a linear classifier $p(Y = 1|x) = 1$ on one side and 0 on the other
  - Not differentiable…
    - Makes it difficult to learn
    - Can't handle noisy labels

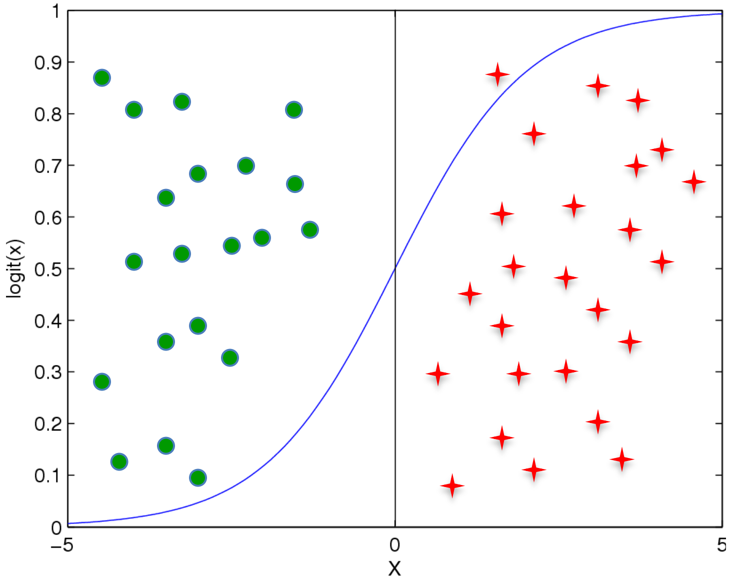


Logistic Regression

- Learn $p(y|x)$ directly from the data
  - Assume a particular functional form

$$p(Y=-1|x) = \frac{1}{1+\exp(w^T x + b)}$$

$$p(Y=1|x) = \frac{\exp(w^T x + b)}{1+\exp(w^T x + b)}$$



Functional Form: Two classes

- Given some w and b, we can classify a new point x by assigning the label 1 if $p(Y = 1|x) > p(Y = -1|x)$ and -1 otherwise
  - This leads to a linear classification rule:
    - Classify as a 1 if $w^T x + b > 0$
    - Classify as a -1 if $w^T x + b < 0$

Learning the Weights

- To learn the weights, we maximize the conditional likelihood

$$(w^*, b^*) = \arg\max_{w,b} \prod_{i=1}^{N} p(y^{(i)}|x^{(i)}, w, b)$$

- This is the not the same strategy that we used in the case of naive Bayes
  - For naive Bayes, we maximized the log-likelihood

Learning the Weights

$$
\begin{aligned}
\ell(w,b) &= \ln \prod_{i=1}^{N} p(y^{(i)} | x^{(i)}, w, b) \\
&= \sum_{i=1}^{N} \ln p(y^{(i)} | x^{(i)}, w, b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \ln p(Y=1 | x^{(i)}, w, b) + \left(1 - \frac{y^{(i)}+1}{2}\right) \ln p(Y=-1 | x^{(i)}, w, b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \ln \frac{p\left(Y=1 \middle| x^{(i)}, w, b\right)}{p\left(Y=-1 \middle| x^{(i)}, w, b\right)} + \ln p(Y=-1 | x^{(i)}, w, b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \left(w^T x^{(i)} + b\right) - \ln\left(1 + \exp\left(w^T x^{(i)} + b\right)\right)
\end{aligned}
$$

Learning the Weights

$$
\begin{aligned}
\ell(w,b) &= \ln \prod_{i=1}^{N} p(y^{(i)}|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \ln p(y^{(i)}|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \ln p(Y=1|x^{(i)},w,b) + \left(1-\frac{y^{(i)}+1}{2}\right) \ln p(Y=-1|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \ln \frac{p\left(Y=1\middle|x^{(i)},w,b\right)}{p\left(Y=-1\middle|x^{(i)},w,b\right)} + \ln p(Y=-1|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2}\left(w^T x^{(i)}+b\right) - \ln\left(1+\exp\left(w^T x^{(i)}+b\right)\right)
\end{aligned}
$$

This is concave in w and b: take derivatives and solve!

Learning the Weights

$$
\begin{aligned}
\ell(w,b) \qquad &= \ln \prod_{i=1}^{N} p(y^{(i)}|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \ln p(y^{(i)}|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \ln p(Y=1|x^{(i)},w,b) + \left(1 - \frac{y^{(i)}+1}{2}\right) \ln p(Y=-1|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \ln \frac{p\left(Y=1\middle|x^{(i)},w,b\right)}{p\left(Y=-1\middle|x^{(i)},w,b\right)} + \ln p(Y=-1|x^{(i)},w,b) \\
&= \sum_{i=1}^{N} \frac{y^{(i)}+1}{2} \left(w^T x^{(i)} + b\right) - \ln\left(1 + \exp\left(w^T x^{(i)} + b\right)\right)
\end{aligned}
$$

No closed form solution ☹

Learning the Weights

- Can apply gradient ascent to maximize the conditional likelihood

$$\frac{\partial \ell}{\partial b} = \sum_{i=1}^{N} \left[\frac{y^{(i)} + 1}{2} - p(Y = 1 | x^{(i)}, w, b) \right]$$

$$\frac{\partial \ell}{\partial w_j} = \sum_{i=1}^{N} x_j^{(i)} \left[\frac{y^{(i)} + 1}{2} - p(Y = 1 | x^{(i)}, w, b) \right]$$

Priors

- Can define priors on the weights to prevent overfitting
  - Normal distribution, zero mean, identity covariance

$$p(w) = \prod_j \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{w_j^2}{2\sigma^2}\right)$$

  - “Pushes” parameters towards zero
- Regularization
  - Helps avoid very large weights and overfitting

Priors as Regularization

- The log-MAP objective with this Gaussian prior is then

$$\ln \prod_{i=1}^{N} p\left(y^{(i)} \middle| x^{(i)}, w, b\right) p(w) p(b) = \left[\sum_{i}^{N} \ln p\left(y^{(i)} \middle| x^{(i)}, w, b\right)\right] - \frac{\lambda}{2} \|w\|_2^2$$

  - Quadratic penalty: drives weights towards zero
  - Adds a negative linear term to the gradients
  - Different priors can produce different kinds of regularization

Somtimes called an ℓ_2 regularizer

Generative vs. Discriminative Classifiers

Generative classifier:
(e.g., Naïve Bayes)

- Assume some **functional form** for $p(x|y), p(y)$
- Estimate parameters of $p(x|y)$, $p(y)$ directly from training data
- Use Bayes rule to calculate $p(y|x)$
- This is a **generative model**
  - **Indirect** computation of $p(Y|X)$ through Bayes rule
  - As a result, **can also generate a sample of the data,** $p(x) = \sum_y p(y)p(x|y)$

Discriminative classifiers:
(e.g., Logistic Regression)

- Assume some **functional form for** $p(y|x)$
- Estimate parameters of $p(y|x)$ directly from training data
- This is a discriminative model
  - Directly learn $p(y|x)$
  - But **cannot obtain a sample of the data** as $p(x)$ is not available
  - Useful for discriminating labels