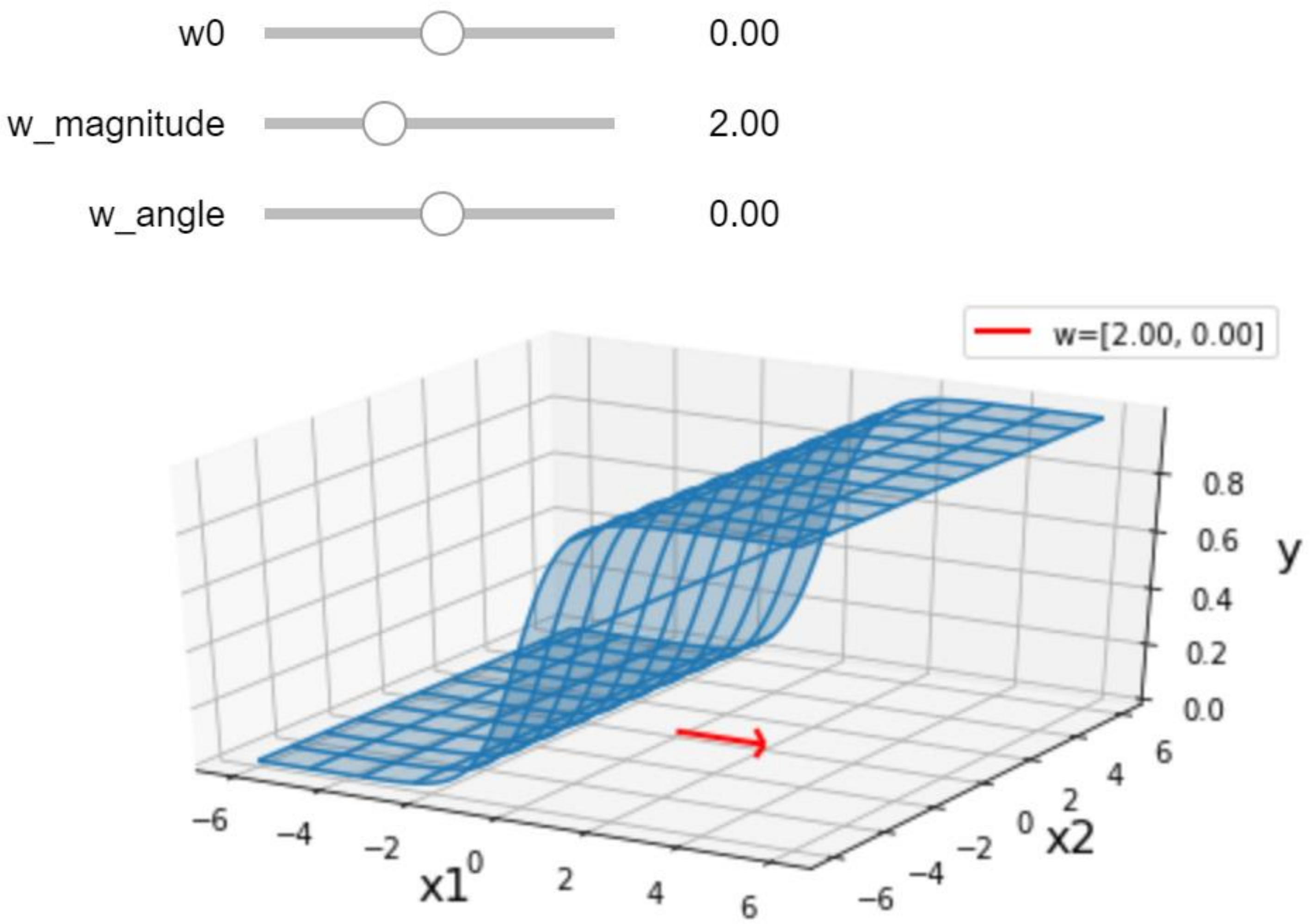
Warm-up as You Log In

Interact with the lec8.ipynb posted on the course website schedule



Announcements

Midterm 1

- Monday
- Lots of info on Piazza
- Stay tuned for one more post regarding day-of details

Plan

Last time

- Likelihood, MLE, conditional likelihood and M(C)LE

Today

- Logistic regression
  - Solving logistic regression
  - Decision boundaries
  - Multiclass logistic regression
- Feature engineering

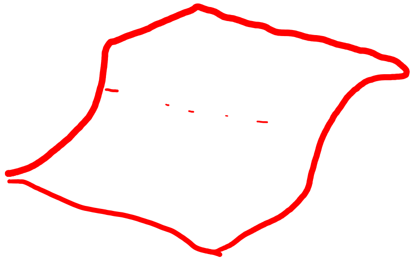
Introduction to Machine Learning

Logistic Regression and Feature Engineering

Instructor: Pat Virtue

BINARY LOGISTIC REGRESSION

Binary Logistic Regression



logistic

1) Model: $Y \sim Bern(\mu)$ $\quad \mu = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$ $\quad \sigma(z) = \frac{1}{1+e^{-z}}$

$$p(Y=y \mid \vec{x}, \vec{\theta}) = \begin{cases} \mu & \text{if } y=1 \\ 1-\mu & \text{if } y=0 \end{cases}$$

2) Objective function: negative log likelihood

$$\ell(\vec{\theta}) = \sum_{i=1}^{N} \log p(Y=y^{(i)} \mid \vec{x}, \vec{\theta})$$ ← log likelihood

$$J(\vec{\theta}) = -\frac{1}{N} \ell(\vec{\theta})$$

3) Solve for $\hat{\theta}$ SGD

Binary Logistic Regression

Gradient

Solve Logistic Regression

$Y \sim Bern(\mu) \qquad \mu = \sigma(\boldsymbol{\theta}^T \boldsymbol{x}) \qquad \sigma(z) = \frac{1}{1+e^{-z}}$

$$J(\boldsymbol{\theta}) = -\frac{1}{N}\sum_n \left(y^{(n)} \log \mu^{(n)} + \left(1 - y^{(n)}\right) \log\left(1 - \mu^{(n)}\right)\right)$$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = -\frac{1}{N}\sum_n \left(y^{(n)} - \mu^{(n)}\right) \boldsymbol{x}^{(n)}$$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = 0?$$

No closed form solution ☹

Back to iterative methods. Solve with (stochastic) gradient descent, Newton's method, or Iteratively Reweighted Least Squares (IRLS)

Piazza Poll 1

Which of the following is a correct description of SGD for Logistic Regression?

A. (1) compute the gradient of the log-likelihood for all examples (2) update all the parameters using the gradient

B. (1) compute the gradient of the log-likelihood for all examples (2) randomly pick an example (3) update only the parameters for that example

C. (1) randomly pick a parameter, (2) compute the partial derivative of the log-likelihood with respect to that parameter, (3) update that parameter for all examples

D. (1) randomly pick an example, (2) compute the gradient of the log-likelihood for that example, (3) update all the parameters using that gradient

E. (1) randomly pick a parameter and an example, (2) compute the gradient of the log-likelihood for that example with respect to that parameter, (3) update that parameter using that gradient

Slide credit: CMU MLD Matt Gormley

Piazza Poll 1

Which of the following is a correct description of SGD for Logistic Regression?

A. (1) compute the gradient of the log-likelihood for all examples (2) update all the parameters using the gradient

B. (1) compute the gradient of the log-likelihood for all examples (2) randomly pick an example (3) update only the parameters for that example

C. (1) randomly pick a parameter, (2) compute the partial derivative of the log-likelihood with respect to that parameter, (3) update that parameter for all examples

D. (1) randomly pick an example, (2) compute the gradient of the log-likelihood for that example, (3) update all the parameters using that gradient

E. (1) randomly pick a parameter and an example, (2) compute the gradient of the log-likelihood for that example with respect to that parameter, (3) update that parameter using that gradient

Slide credit: CMU MLD Matt Gormley

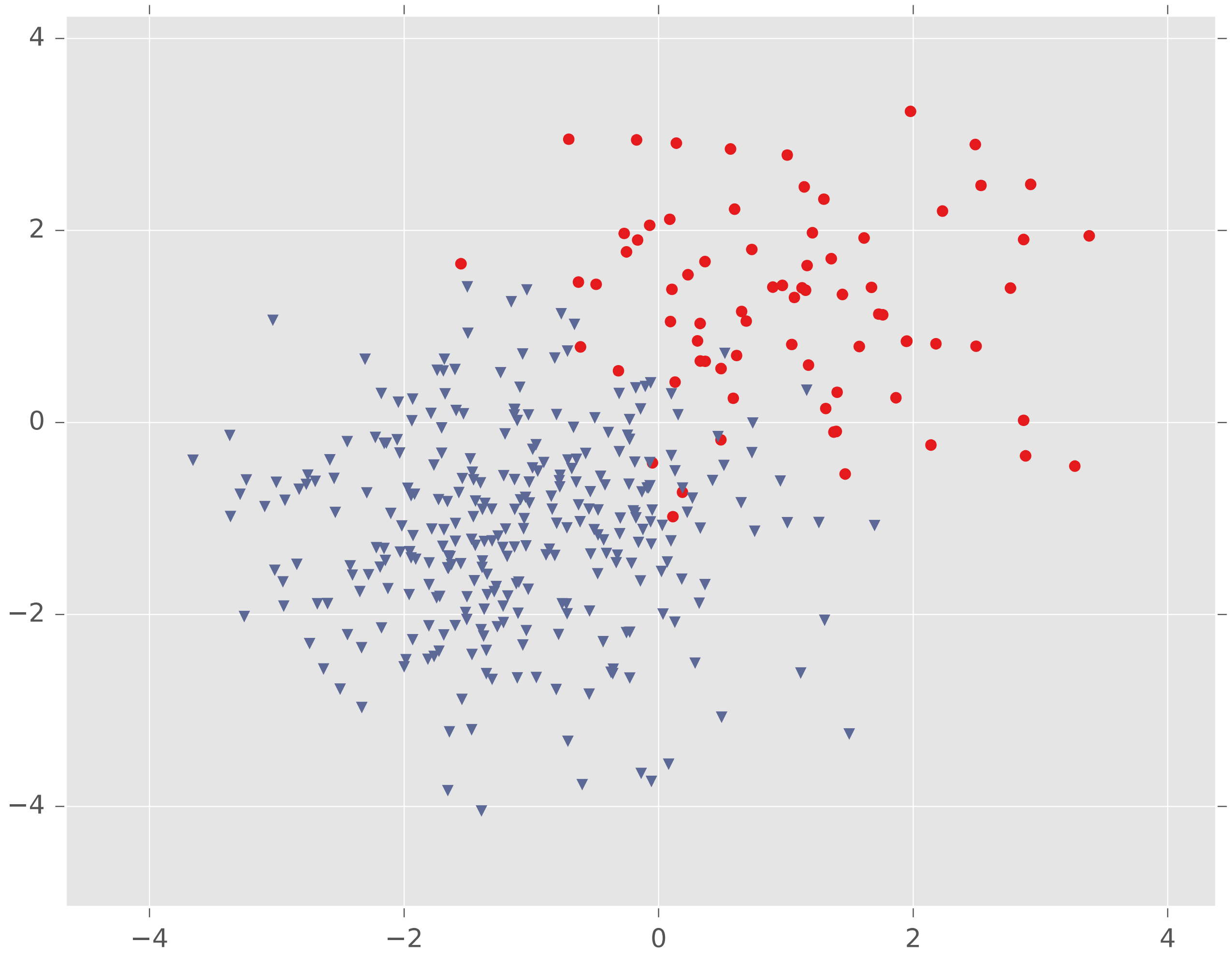
DECISION BOUNDARIES FOR LOGISTIC REGRESSION

Bayes Optimal Classifier

Given an oracle that perfectly knows everything, e.g. $p^*(Y = y \mid x, \theta)$,

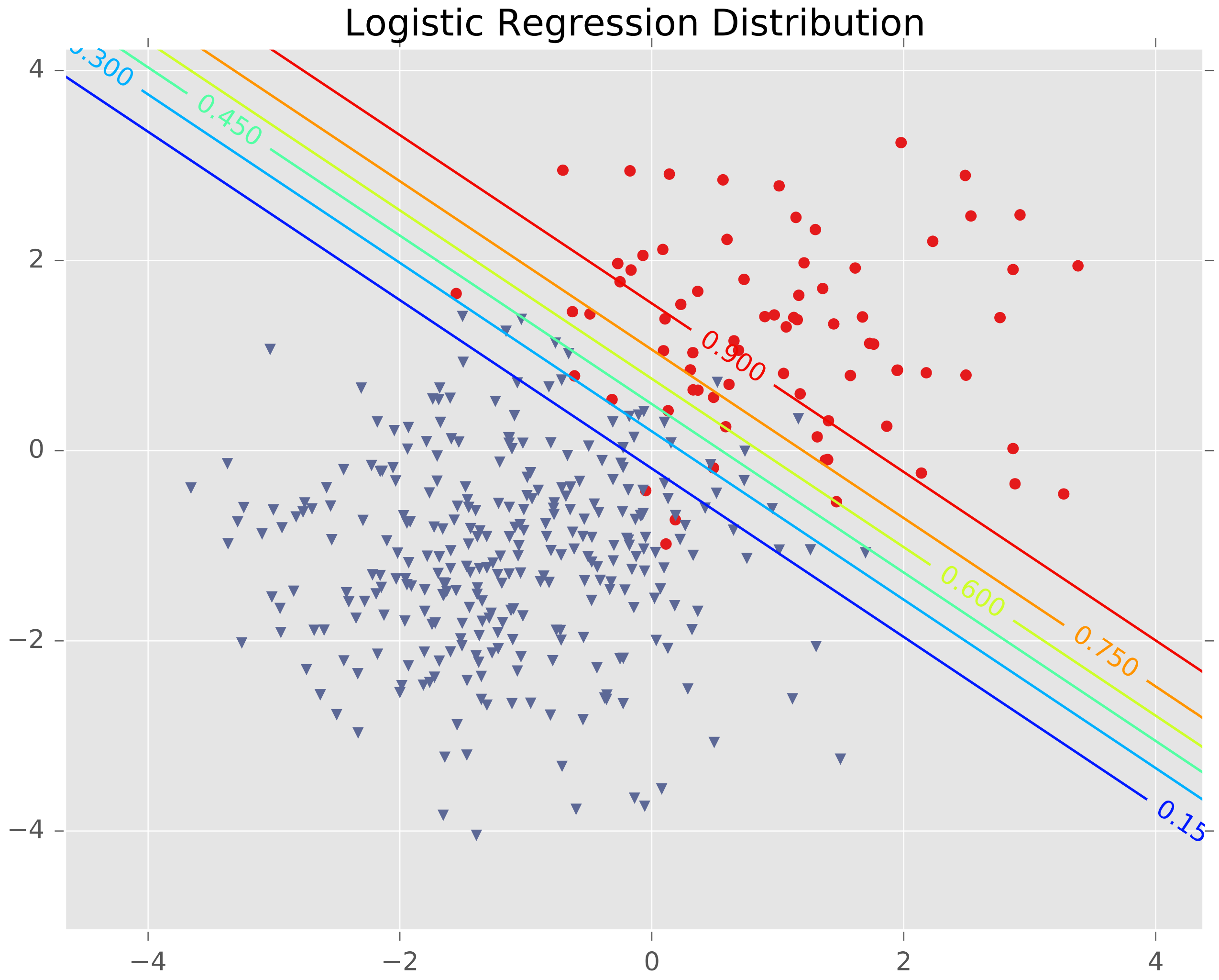
What is the optimal classifier in this setting?

Logistic Regression



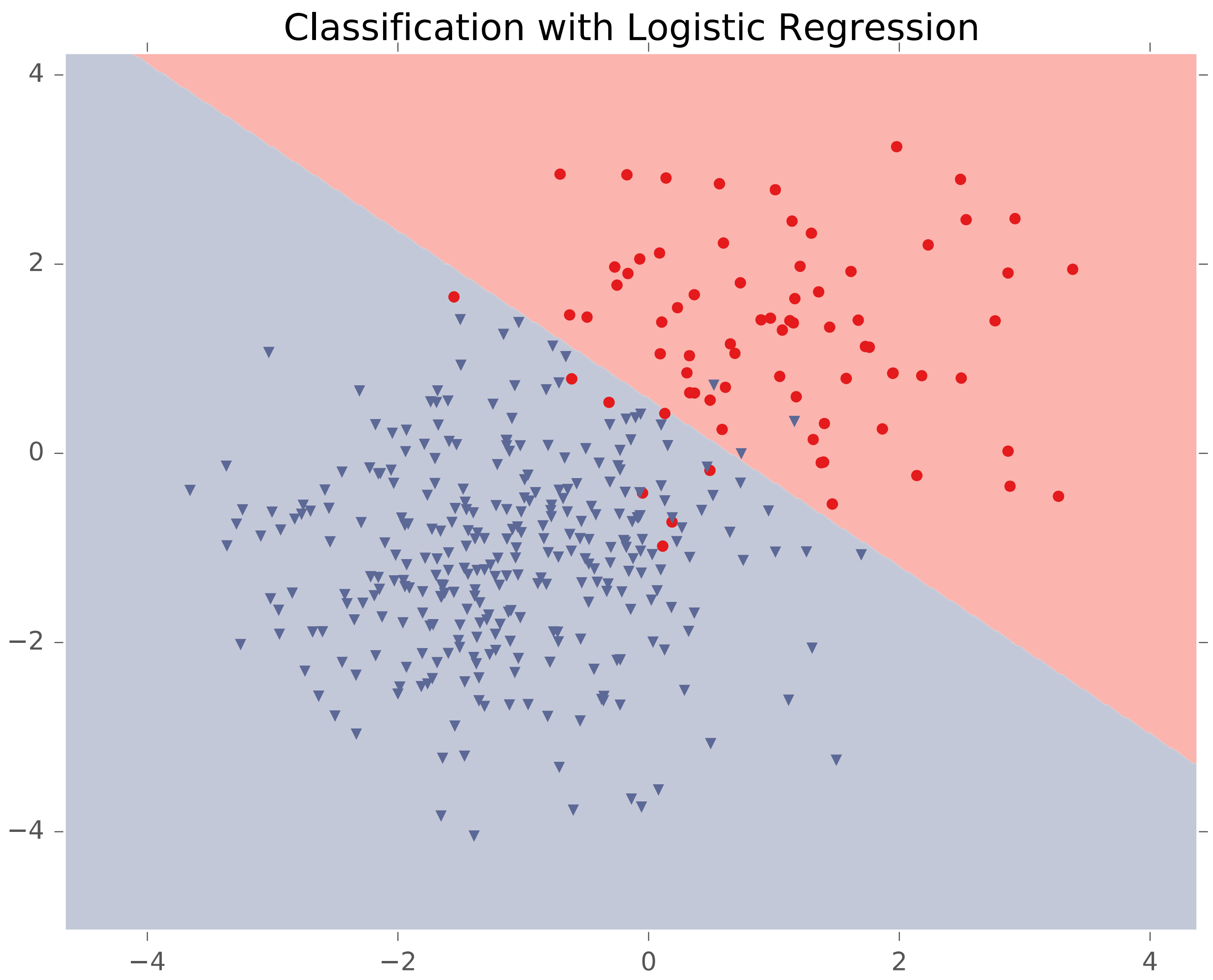
Slide credit: CMU MLD Matt Gormley

Logistic Regression



Slide credit: CMU MLD Matt Gormley

Logistic Regression



Linear in Higher Dimensions

1-D $y = wx + b$
2-D $y = w_1 x_1 + w_2 x_2 + b$

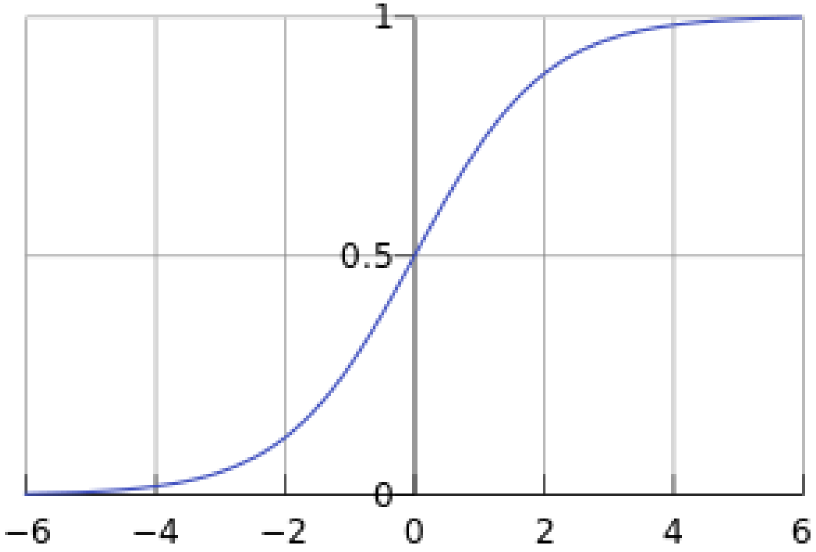
What are these linear shapes called for 1-D, 2-D, 3-D, M-D input?

	$\boldsymbol{x} \in \mathbb{R}$	$\boldsymbol{x} \in \mathbb{R}^2$	$\boldsymbol{x} \in \mathbb{R}^3$	$\boldsymbol{x} \in \mathbb{R}^M$
$y = \boldsymbol{w}^T \boldsymbol{x} + b$	line	plane	hyperplane	hyperplane
$\boldsymbol{w}^T \boldsymbol{x} + b = 0$	point	line	plane	hyperplane
$\boldsymbol{w}^T \boldsymbol{x} + b \geq 0$	halfline	halfplane	halfspace	halfspace

Piazza Poll 2

For a point $\boldsymbol{x}$ on the decision boundary of logistic regression, does $g(\mathbf{w}^T\mathbf{x} + b) = \mathbf{w}^T\mathbf{x} + b$?

$$g(z) = \frac{1}{1 + e^{-z}}$$



Logistic Regression

Data: Inputs are continuous vectors of length M. Outputs are discrete.

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N} \text{ where } \mathbf{x} \in \mathbb{R}^M \text{ and } y \in \{0, 1\}$$

Model: Logistic function applied to dot product of parameters with input vector.

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

Learning: finds the parameters that minimize some objective function. $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta})$

Prediction: Output is the most probable class.

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}}(y|\mathbf{x})$$

MULTI-CLASS LOGISTIC REGRESSION

Prep: Multi-class Logistic Regression

Logistic function

$$g(z) = \frac{e^z}{e^z+1}$$

$$p(\,Y = 1 \mid x, \theta\,) = g(\mu) = \frac{e^\mu}{e^\mu+1}$$

$$p(\,Y = 0 \mid x, \theta\,) = 1 - g(\mu) = 1 - \frac{e^\mu}{e^\mu+1}$$

Probability distribution sums to 1

$$\sum_y p(Y = y \mid x, \theta)$$

$$= p(\,Y = 0 \mid x, \theta\,) + p(\,Y = 1 \mid x, \theta\,)$$

$$= \; 1 - \frac{e^\mu}{e^\mu+1} + \frac{e^\mu}{e^\mu+1} = 1$$

Prep: Multi-class Logistic Regression

Bernoulli distribution:

$$Y \sim Bern(\phi)$$

$$p(y) = \begin{cases} \phi, & y = 1 \\ 1 - \phi, & y = 0 \end{cases}$$

$$L(\phi) = \prod_n p\left(y^{(n)}\right) = \prod_n \phi^{y^{(n)}} (1 - \phi)^{\left(1 - y^{(n)}\right)}$$

Categorical distribution:

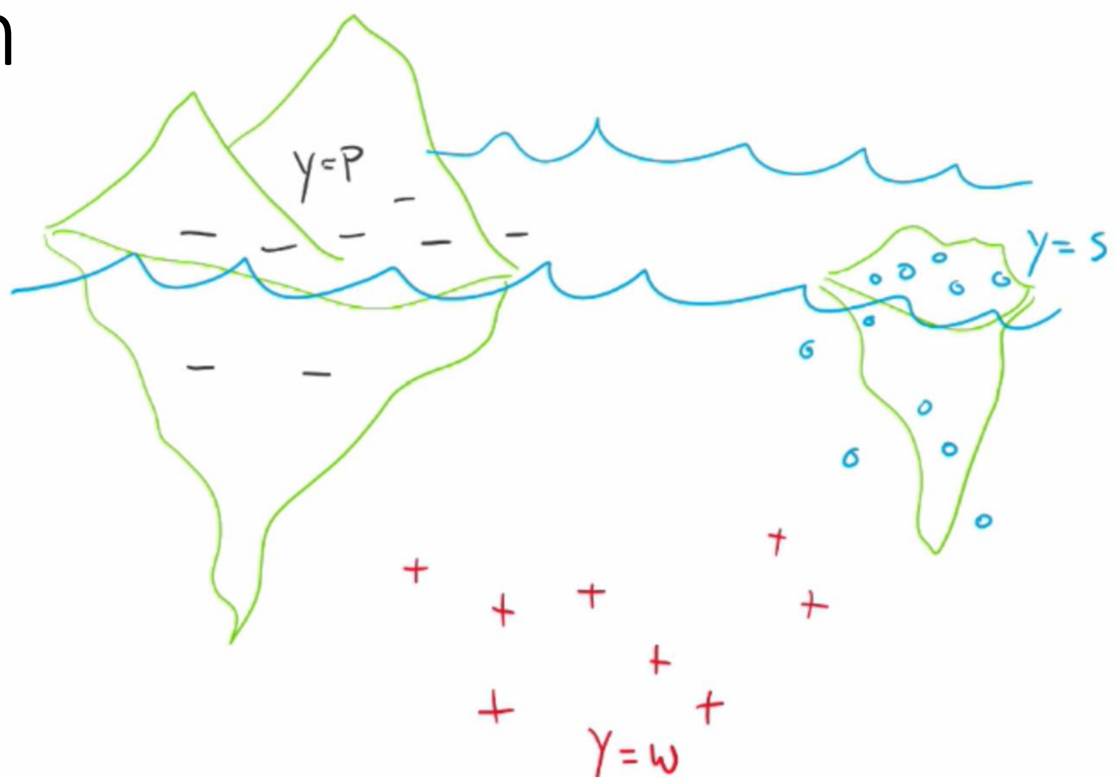
$$Y \sim Categorical(\phi_1, \phi_2, \ldots, \phi_K)$$

$$p(y) = \begin{cases} \phi_1, & y = 1 \\ & \vdots \end{cases}$$

$$L(\phi_1, \phi_2, \ldots, \phi_K) = \prod_n p\left(y^{(n)}\right) = \prod_n \prod_k \phi_k^{\mathbb{I}(y^{(n)} = k)}$$

Slide credit: CMU MLD Matt Gormley

Multi-class Logistic Regression



Slide credit: CMU MLD Matt Gormley

Multi-class Logistic Regression

Multi-class Logistic Regression

Gradient

Summary: Logistic Function

Logistic (sigmoid) function converts value from $(-\infty, \infty) \rightarrow (0, 1)$

$$g(z) = \frac{1}{1 + \mathrm{e}^{-z}} = \frac{e^z}{e^z + 1}$$

$g(z)$ and $1 - g(z)$ sum to one

Example 2 $\rightarrow g(2) = 0.88$, 1-$g(2) = 0.12$

Summary: Softmax Function

Softmax function convert each value in a vector of values from $(-\infty, \infty) \rightarrow (0, 1)$, such that they all sum to one.

$$g(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_K \end{bmatrix} \rightarrow \begin{bmatrix} e^{z_1} \\ e^{z_2} \\ \vdots \\ e^{z_K} \end{bmatrix} \cdot \frac{1}{\sum_{k=1}^{K} e^{z_k}}$$

Example $\begin{bmatrix} -1 \\ 4 \\ 1 \\ -2 \\ 3 \end{bmatrix} \rightarrow \begin{bmatrix} 0.0047 \\ 0.7008 \\ 0.0349 \\ 0.0017 \\ 0.2578 \end{bmatrix}$

Summary: Multiclass Predicted Probability

Multiclass logistic regression uses the parameters learned across all K classes to predict the discrete conditional probability distribution of the output Y given a specific input vector $\boldsymbol{x}$

$$\begin{bmatrix} p(Y = 1 \mid \boldsymbol{x}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3) \\ p(Y = 2 \mid \boldsymbol{x}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3) \\ p(Y = 3 \mid \boldsymbol{x}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3) \end{bmatrix} = \begin{bmatrix} e^{\boldsymbol{\theta}_1^T \boldsymbol{x}} \\ e^{\boldsymbol{\theta}_2^T \boldsymbol{x}} \\ e^{\boldsymbol{\theta}_3^T \boldsymbol{x}} \end{bmatrix} \cdot \frac{1}{\sum_{k=1}^{K} e^{\boldsymbol{\theta}_k^T \boldsymbol{x}}}$$

Debug that Program!

In-Class Exercise:

Debug the following program which is (incorrectly) attempting to run SGD for multinomial logistic regression

Buggy Program:

```
while not converged:
  for i in shuffle([1,…,N]):
    for k in [1,…,K]:
      theta[k] = theta[k] - gamma * grad(x[i], y[i], theta, k)
```

Assume: grad(x[i], y[i], theta, k) returns the gradient of the negative log-likelihood of the training example (x[i],y[i]) with respect to vector theta[k]. gamma is the learning rate. N = # of examples. K = # of output classes. M = # of features. theta is a K by M matrix.

Slide credit: CMU MLD Matt Gormley

FEATURE ENGINEERING

How Do We Deal with Real-world Problems

Politician Voting Classification

How Do We Deal with Real-world Problems

SPAM Classification