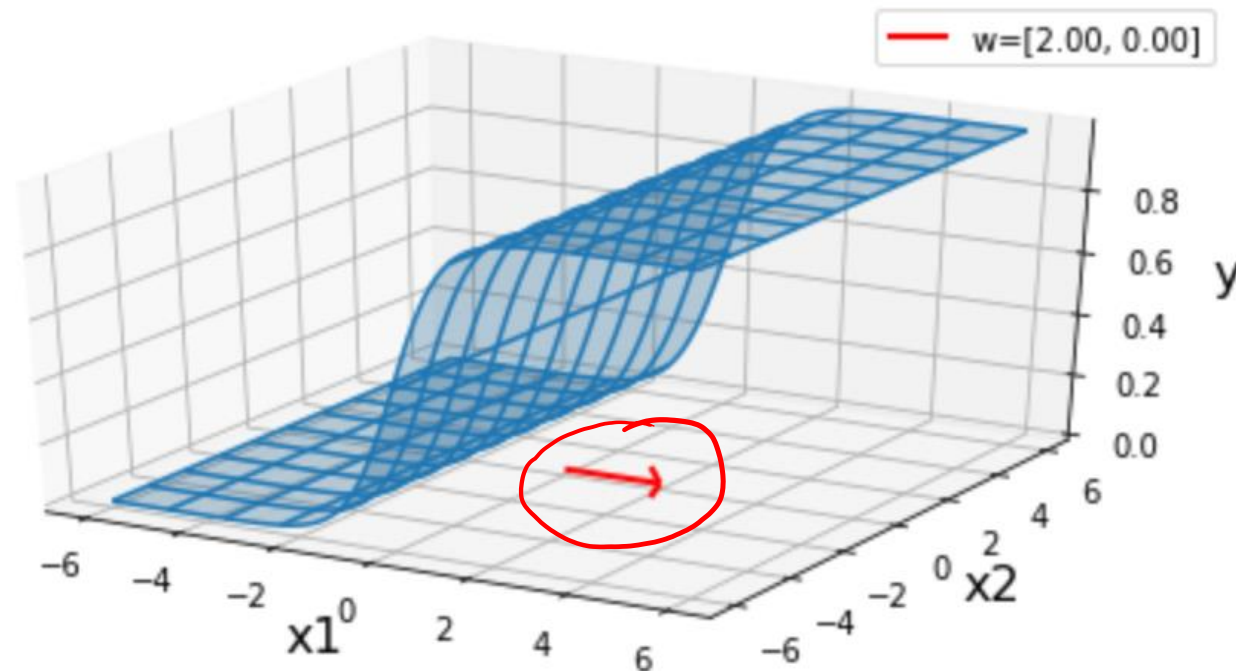
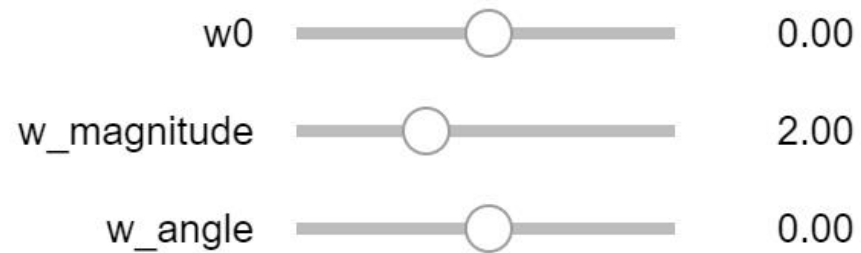


Warm-up as You Log In

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

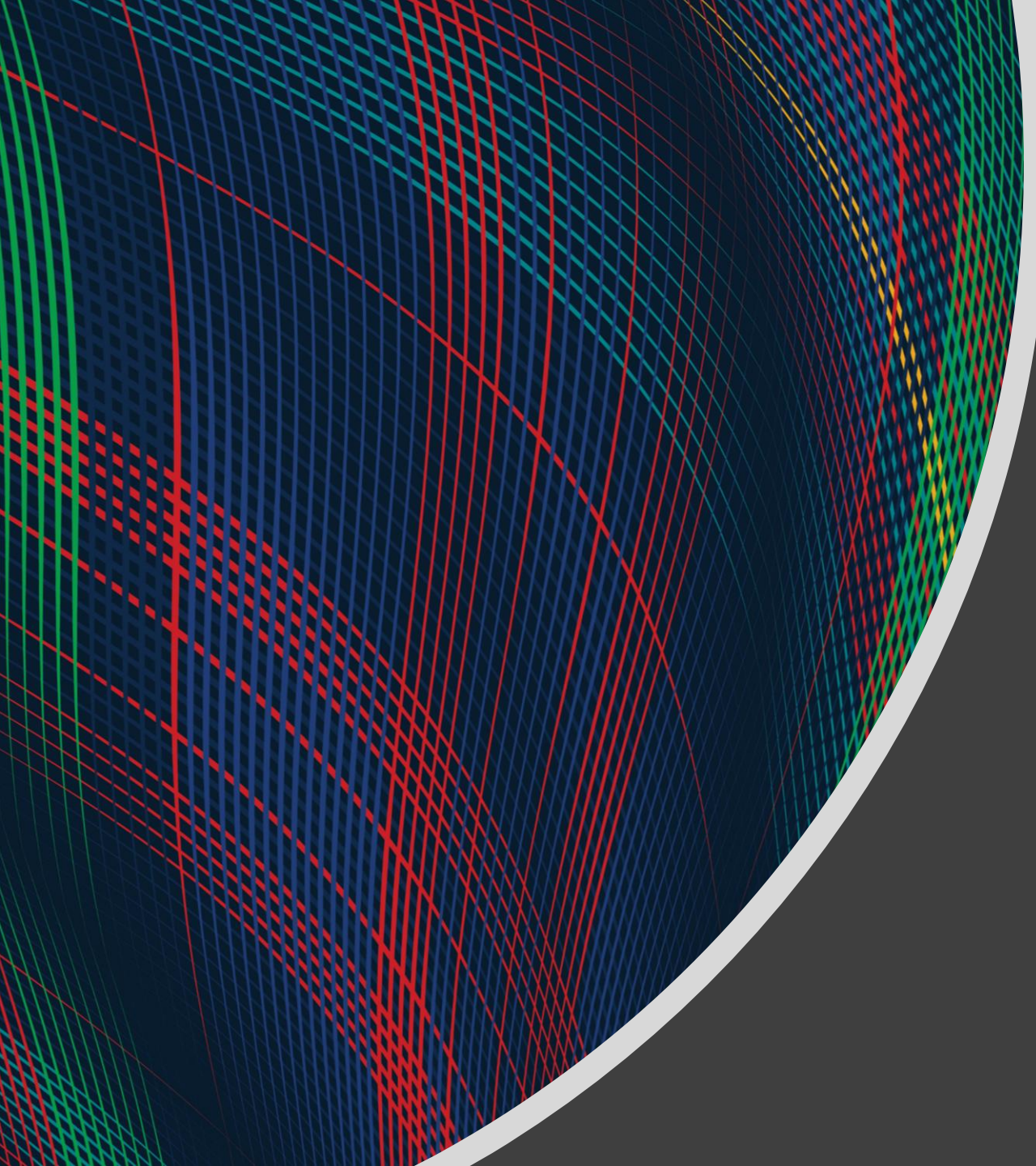
Plan

Last time

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

Today

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

An abstract graphic on the left side of the slide, featuring a sphere-like shape composed of a dense grid of intersecting red, green, and blue lines. The lines are curved and follow the contour of the sphere, creating a complex, woven pattern. The sphere is set against a dark gray background.

Introduction to Machine Learning

Logistic Regression and Feature Engineering

Instructor: Pat Virtue

BINARY LOGISTIC REGRESSION

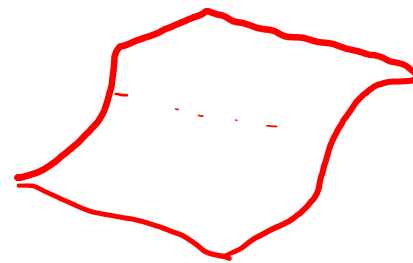
Binary Logistic Regression

1) Model: $Y \sim \text{Bern}(\mu)$

$$\mu = \sigma(\theta^T x)$$

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

logistic



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

$$\sigma(\theta^T x) \leftarrow$$
$$1 - \sigma(\theta^T x) \leftarrow$$

2) Objective function: negative log likelihood

$$l(\vec{\theta}) = \sum_{i=1}^N \log p(Y=y^{(i)} | \vec{x}, \vec{\theta}) \leftarrow \text{log likelihood}$$

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

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

Binary Logistic Regression

$$\vec{x} \in \mathbb{R}^m \quad \vec{\theta} \in \mathbb{R}^m$$

Gradient

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N \underbrace{\log p(y^{(i)} | \vec{x}^{(i)}, \theta)}_{J^{(i)}(\theta)}$$

$$\nabla_{\theta} J(\theta) : \mathbb{R}^m \rightarrow \underline{\mathbb{R}^m}$$

$$\nabla_{\theta} J^{(i)}(\theta) = \begin{bmatrix} \frac{\partial J^{(i)}}{\partial \theta_1} \\ \vdots \\ \frac{\partial J^{(i)}}{\partial \theta_m} \\ \vdots \end{bmatrix} \rightarrow \frac{\partial J^{(i)}}{\partial \theta_m} = - (y^{(i)} - \sigma(\vec{\theta}^T \vec{x})) \underline{x_m^{(i)}}$$

$$\nabla_{\theta} J^{(i)} = - (y^{(i)} - \sigma(\vec{\theta}^T \vec{x})) \vec{x}$$

SGD

Solve Logistic Regression

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

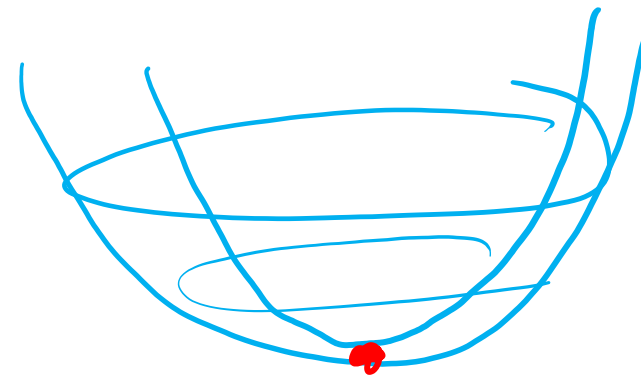
$$J(\boldsymbol{\theta}) = -\frac{1}{N} \sum_n (y^{(n)} \log \mu^{(n)} + (1 - y^{(n)}) \log(1 - \mu^{(n)}))$$

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

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

No closed form solution ☹️

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



← CONVEX

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 42% → 65%
- 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

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

DECISION BOUNDARIES FOR LOGISTIC REGRESSION

Bayes Optimal Classifier

$$\begin{aligned} p(Y=1 | x, \theta) &\leftarrow \\ p(Y=0 | x, \theta) &\leftarrow \end{aligned}$$

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

What is the optimal classifier in this setting?

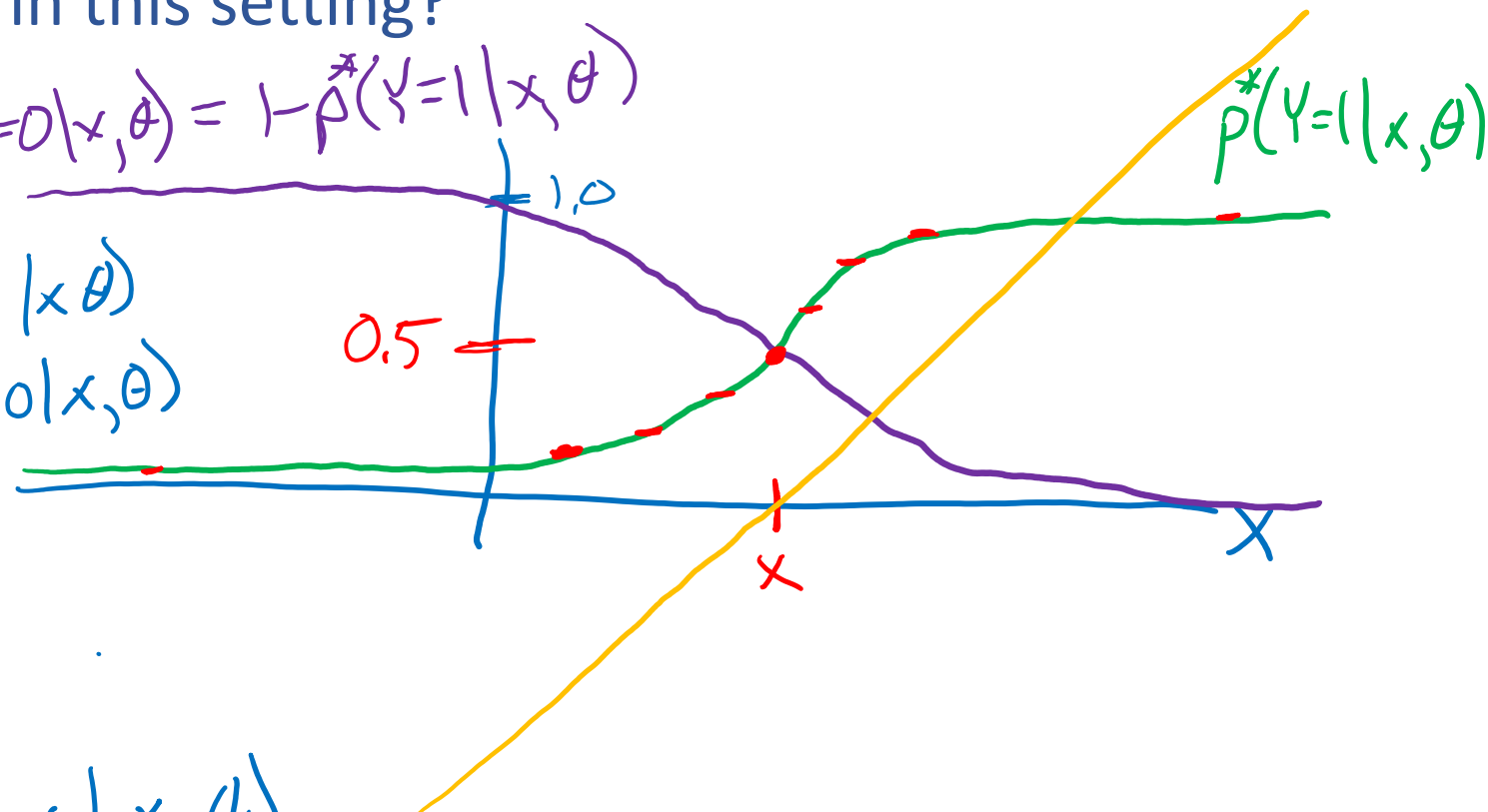
$$\hat{y} = h(\vec{x})$$

$$= \begin{cases} 1 \\ 0 \end{cases}$$

$$\text{if } p(Y=1 | x, \theta) \geq p(Y=0 | x, \theta)$$

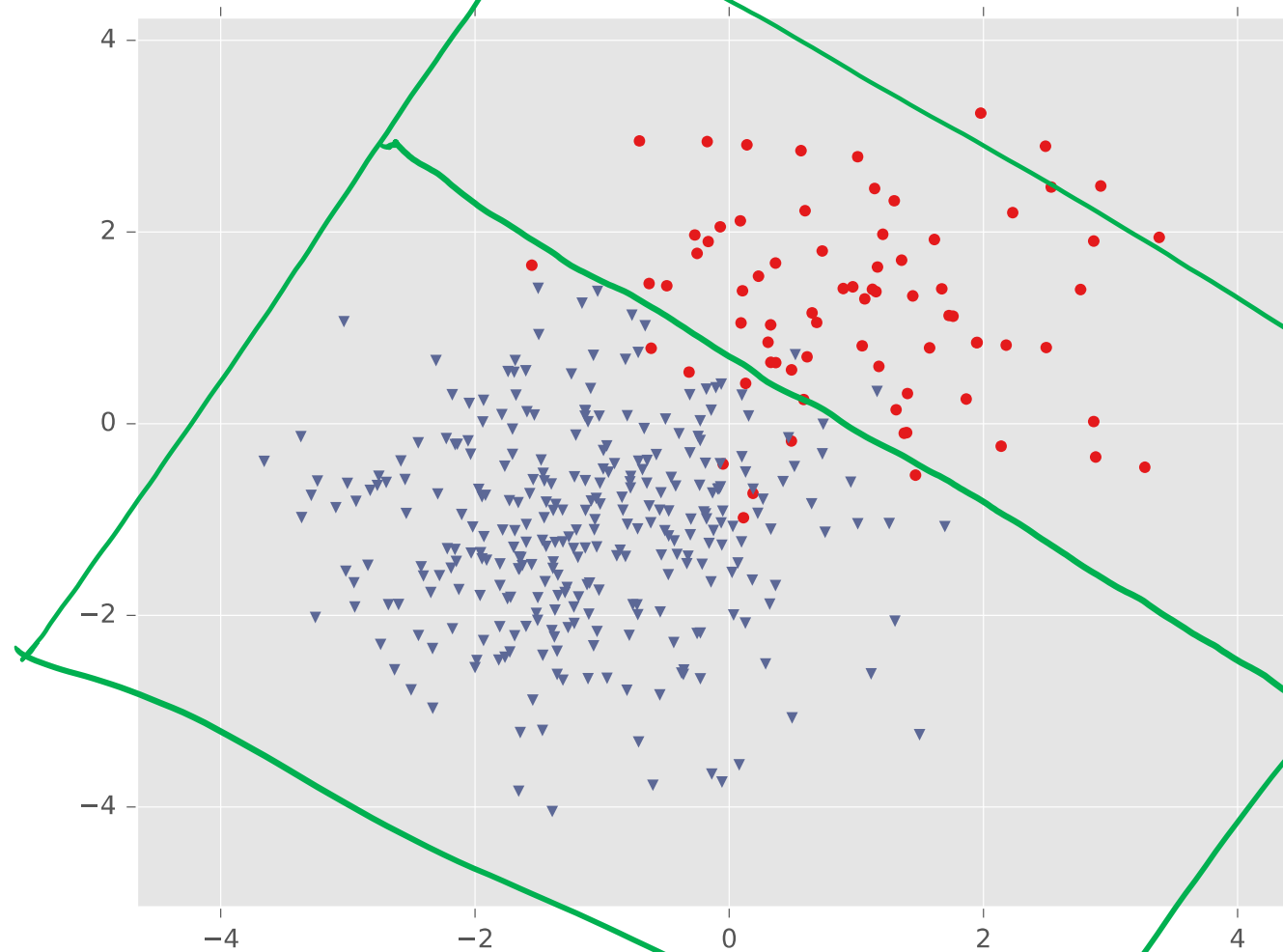
o.w.

$$p^*(Y=0 | x, \theta) = 1 - p^*(Y=1 | x, \theta)$$



$$\hat{y} = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} p(Y=y | x, \theta)$$

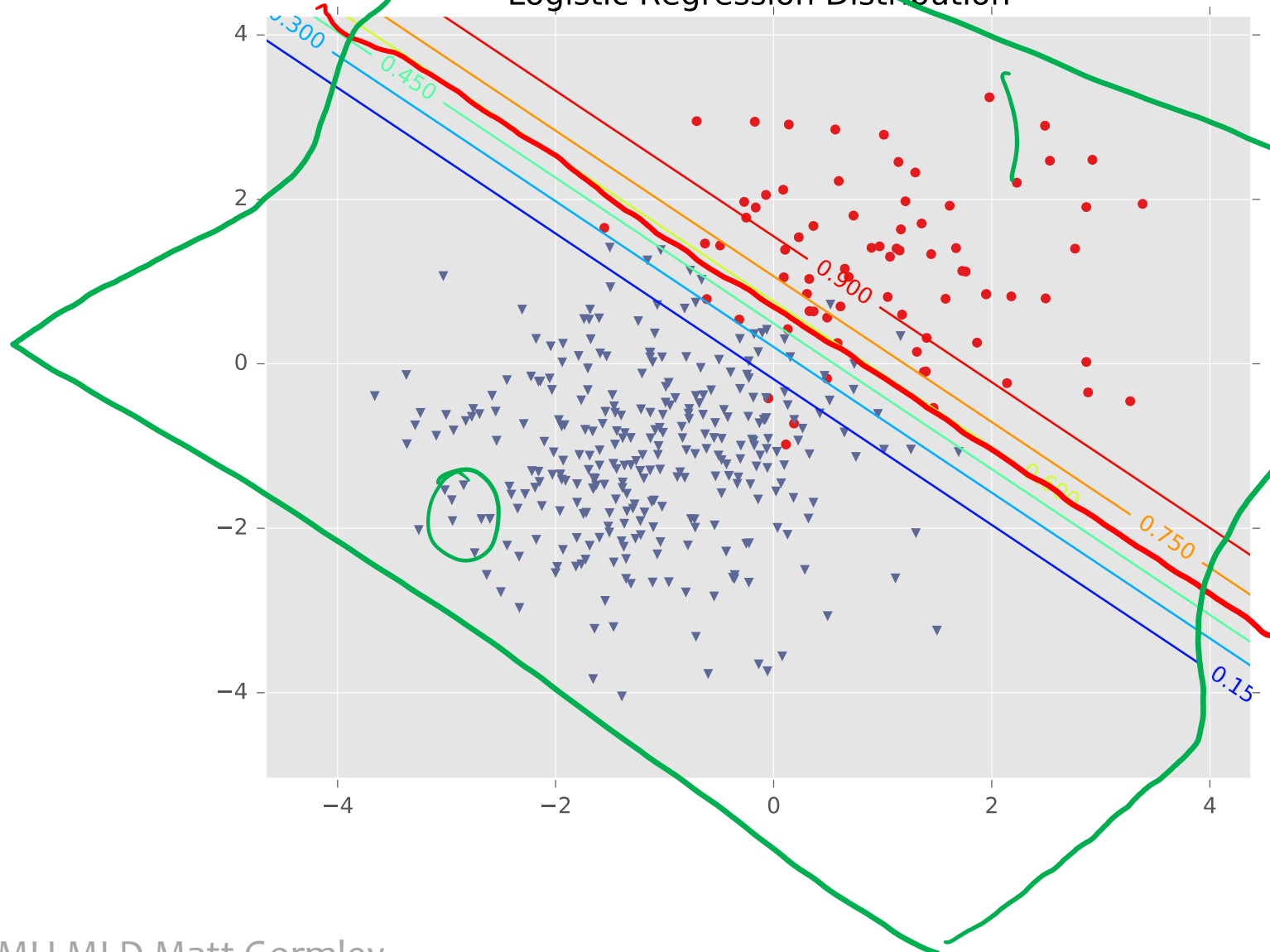
Logistic Regression



$$y = w^T x + b = 0$$

Logistic Regression

Logistic Regression Distribution



$$\begin{aligned} p(Y=1 | x, \theta) \\ &= p(Y=0 | x, \theta) \\ &= 0.5 \end{aligned}$$

Logistic Regression



$$p(Y=1 | \vec{x}, \theta)$$

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?

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

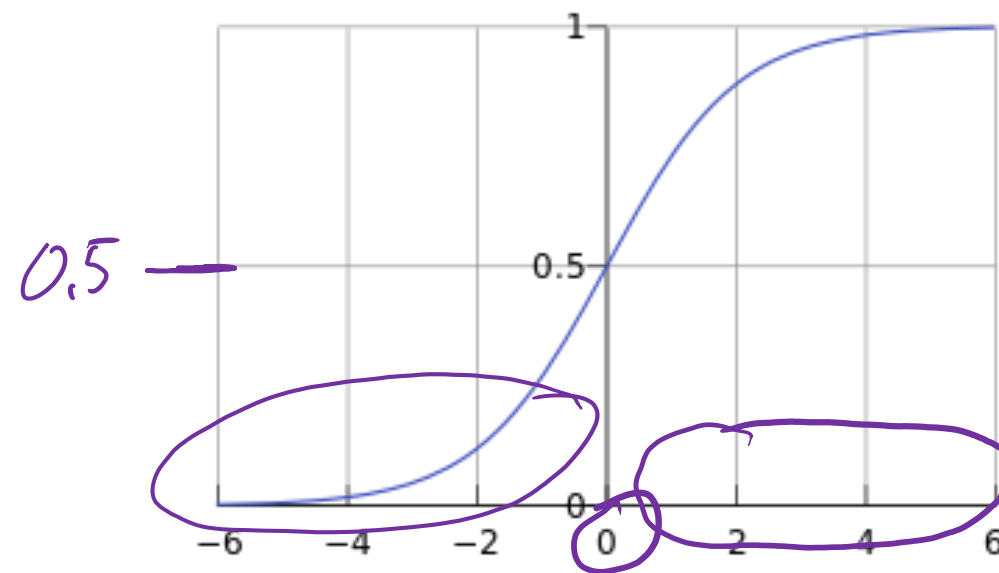
Piazza Poll 2

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

$$\begin{aligned} &g(\mathbf{w}^T \mathbf{x} + b) && 0 = \mathbf{w}^T \mathbf{x} + b \\ &= p(Y=1 | \vec{x}, \vec{\theta}) \\ &= 0.5 \end{aligned}$$

Yes
No 50% → 75%

$$0.5 \leftarrow g(z) = \frac{1}{1 + e^{-z}}$$



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.

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

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

Prediction: Output is the most probable class.

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

MULTI-CLASS LOGISTIC REGRESSION

Prep: Multi-class Logistic Regression

$$\mu = \theta^T x$$

Logistic function

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

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

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

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

Probability distribution sums to 1

$$\begin{aligned} & \left[\sum_y p(Y = y \mid x, \theta) \right. \\ &= \underline{p(Y = 0 \mid x, \theta)} + \underline{p(Y = 1 \mid x, \theta)} \\ &= 1 - \frac{e^\mu}{e^\mu + 1} + \frac{e^\mu}{e^\mu + 1} = \underline{1} \end{aligned}$$

Prep: Multi-class Logistic Regression

Bernoulli distribution:

$$Y \sim \text{Bern}(\phi) \quad \phi_H \quad \phi_T = 1 - \phi_H$$

× Binomial

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

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

Categorical distribution:

$$Y \sim \text{Categorical}(\phi_1, \phi_2, \dots, \phi_K)$$

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

× Multinomial

[2, 4, 3]

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

$\phi_2 \cdot \phi_4 \cdot \phi_3$



Slide credit: CMU MLD Matt Gormley

Multi-class Logistic Regression

$$w_p^T x + b_p = 0 \quad \theta_p^T x = 0$$

$$w_w^T x + b_w = 0 \quad \theta_w^T x = 0$$

$$w_s^T x + b_s = 0 \quad \theta_s^T x = 0$$

$$p(Y=p|x, \theta) = \exp(\theta_p^T x) / z$$

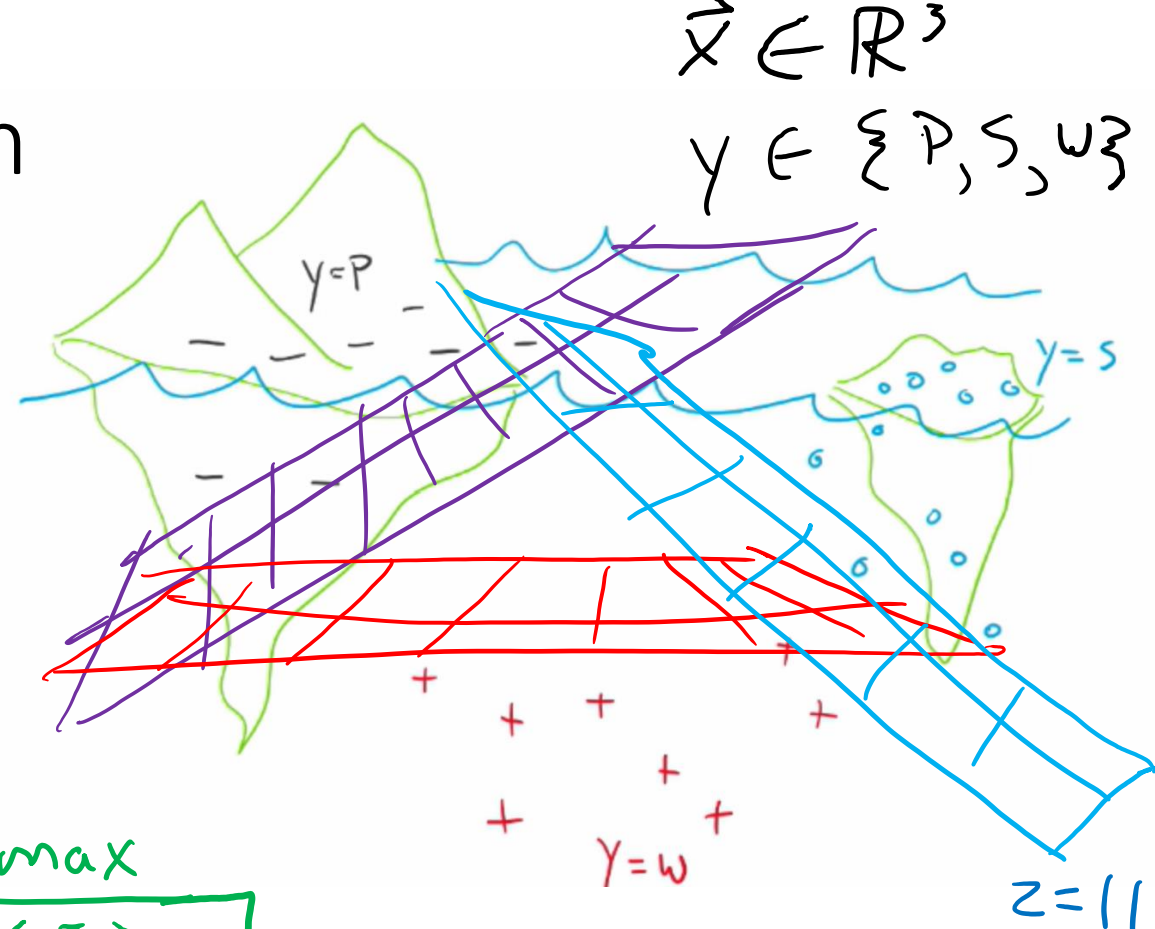
$$p(Y=w|x, \theta) = \exp(\theta_w^T x) / z$$

$$p(Y=s|x, \theta) = \exp(\theta_s^T x) / z$$

$$z = \sum_{k=1}^K \exp(\theta_k^T x)$$

Softmax

$$\frac{\exp(\theta_j^T x)}{\sum_{k=1}^K \exp(\theta_k^T x)}$$



$$p(Y=p|\vec{x}, \hat{\theta}_p, \hat{\theta}_w, \hat{\theta}_s) \approx 1/11$$

$$p(Y=w|\vec{x}, \hat{\theta}) \approx 3/11$$

$$p(Y=s|\vec{x}, \hat{\theta}) \approx 7/11$$

Multi-class Logistic Regression

1) Model $Y \sim \text{Categorical}(\vec{\mu}) \quad \vec{\mu} = [\mu_1, \mu_2, \dots, \mu_K]^T \quad \sum_{k=1}^K \mu_k = 1$

$$\left[\mu_j = \frac{\exp(\theta_j^T x)}{\sum_{k=1}^K \exp(\theta_k^T x)} \right] \quad \Theta \in \mathbb{R}^{K \times M}$$

2) Objective neg. (conditional) log likelihood $\ell(\Theta) = \sum \log p(y^i | x^i, \Theta)$

$$\Theta = \begin{bmatrix} - & \vec{\theta}_1^T & - \\ & \vdots & \\ - & \theta_k^T & - \end{bmatrix}$$

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

3) Solve $\hat{\Theta}_{MLE}$ SGD

Multi-class Logistic Regression

$$\Theta \in \mathbb{R}^{k \times m}$$

Gradient $J(\Theta) = \frac{1}{N} \sum -\log p(y|x, \Theta)$

$$\frac{\partial J^{(i)}(\Theta)}{\partial \Theta_{k,m}} = - \left(\mathbb{I}(y^{(i)} = k) - p(y^{(i)} | x^{(i)}, \Theta) \right) x_m^{(i)}$$

$$\frac{\partial J^{(i)}(\Theta)}{\partial \vec{\Theta}_k} = - \left(\mathbb{I}(y^{(i)} = k) - p(y^{(i)} | x^{(i)}, \Theta) \right) \vec{x}^{(i)} \in \mathbb{R}^m$$

$$\boxed{\frac{\partial J^{(i)}(\Theta)}{\partial \Theta} \in \mathbb{R}^{k \times m}}$$

Summary: Logistic Function

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

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

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

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

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 $\mathbf{x}$

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

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

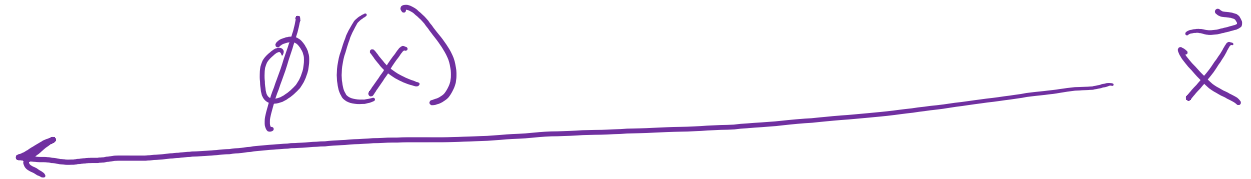
FEATURE ENGINEERING

How Do We Deal with Real-world Problems

Politician Voting Classification

Y	$\phi_1(x)$	$\phi_2(x)$
1	0	0
0	1	0
0	1	1

Party	Vote ₁	Vote ₂
Y	x_1	x_2
Dem	no	no
Rep	yes	no
Rep	yes	yes



How Do We Deal with Real-world Problems

SPAM Classification

$$x = \{ \text{email body, subject, from} \}$$



$$\Phi(x) \in \mathbb{R}^M$$

$$\begin{bmatrix} \phi_1(x) \\ \phi_2(x) \\ \vdots \\ \phi_m(x) \end{bmatrix} \begin{array}{l} = 0 \text{ or } 1 \text{ free appear} \\ = \# \text{ words in all CAPS} \end{array}$$