10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

# Midterm Exam Review

# +

# Binary Logistic Regression

Matt Gormley
Lecture 10
Sep. 25, 2019

# Reminders

- **Homework 3: KNN, Perceptron, Lin.Reg.**
  - **Out: Wed, Sep. 18**
  - **Due: Wed, Sep. 25 at 11:59pm**
- **Midterm Exam 1**
  - **Thu, Oct. 03, 6:30pm – 8:00pm**
- **Homework 4: Logistic Regression**
  - **Out: Wed, Sep. 25**
  - **Due: Fri, Oct. 11 at 11:59pm**
- **Today's In-Class Poll**
  - **http://p10.mlcourse.org**
- *Reading on Probabilistic Learning is reused later in the course for MLE/MAP*

# MIDTERM EXAM LOGISTICS

# Midterm Exam

- **Time / Location**
  - **Time:** Evening Exam
    **Thu, Oct. 03 at 6:30pm – 8:00pm**
  - **Room:** We will contact each student individually with **your room assignment.** The rooms are **not** based on section.
  - **Seats:** There will be **assigned seats.** Please arrive early.
  - Please watch Piazza carefully for announcements regarding room / seat assignments.
- **Logistics**
  - Covered material: Lecture 1 – Lecture 9
  - Format of questions:
    - Multiple choice
    - True / False (with justification)
    - Derivations
    - Short answers
    - Interpreting figures
    - Implementing algorithms on paper
  - No electronic devices
  - You are allowed to **bring** one 8½ x 11 sheet of notes (front and back)

# Midterm Exam

- **How to Prepare**
  - Attend the midterm review lecture (right now!)
  - Review prior year’s exam and solutions (we’ll post them)
  - Review this year’s homework problems
  - Consider whether you have achieved the “learning objectives” for each lecture / section

# Midterm Exam

- **Advice (for during the exam)**
  - Solve the easy problems first
    (e.g. multiple choice before derivations)
    - if a problem seems extremely complicated you're likely missing something
  - Don't leave any answer blank!
  - If you make an assumption, write it down
  - If you look at a question and don't know the answer:
    - we probably haven't told you the answer
    - but we've told you enough to work it out
    - imagine arguing for some answer and see if you like it

# Topics for Midterm 1

- Foundations
  - Probability, Linear Algebra, Geometry, Calculus
  - Optimization
- Important Concepts
  - Overfitting
  - Experimental Design
- Classification
  - Decision Tree
  - KNN
  - Perceptron
- Regression
  - Linear Regression

# SAMPLE QUESTIONS

# Sample Questions

## 1.4 Probability

Assume we have a sample space $\Omega$. Answer each question with **T** or **F**.

(a) [1 pts.] **T or F:** If events $A$, $B$, and $C$ are disjoint then they are independent.

(b) [1 pts.] **T or F:** $P(A|B) \propto \frac{P(A)P(B|A)}{P(A|B)}$. (The sign '$\propto$' means 'is proportional to')

# Sample Questions

## 5.2 Constructing decision trees

Consider the problem of predicting whether the university will be closed on a particular day. We will assume that the factors which decide this are whether there is a snowstorm, whether it is a weekend or an official holiday. Suppose we have the training examples described in the Table 5.2.

| Snowstorm | Holiday | Weekend | Closed |
|---|---|---|---|
| T | T | F | F |
| T | T | F | T |
| F | T | F | F |
| T | T | F | F |
| F | F | F | F |
| F | F | F | T |
| T | F | F | T |
| F | F | F | T |

Table 1: Training examples for decision tree

- [**2 points**] What would be the effect of the Weekend attribute on the decision tree if it were made the root? Explain in terms of information gain.

- [**8 points**] If we cannot make Weekend the root node, which attribute should be made the root node of the decision tree? Explain your reasoning and show your calculations. (You may use $\log_2 0.75 = -0.4$ and $\log_2 0.25 = -2$)

# Sample Questions

## 4 K-NN [12 pts]

Now we will apply K-Nearest Neighbors using Euclidean distance to a binary classification task. We assign the class of the test point to be the class of the majority of the $k$ nearest neighbors. A point can be its own neighbor.

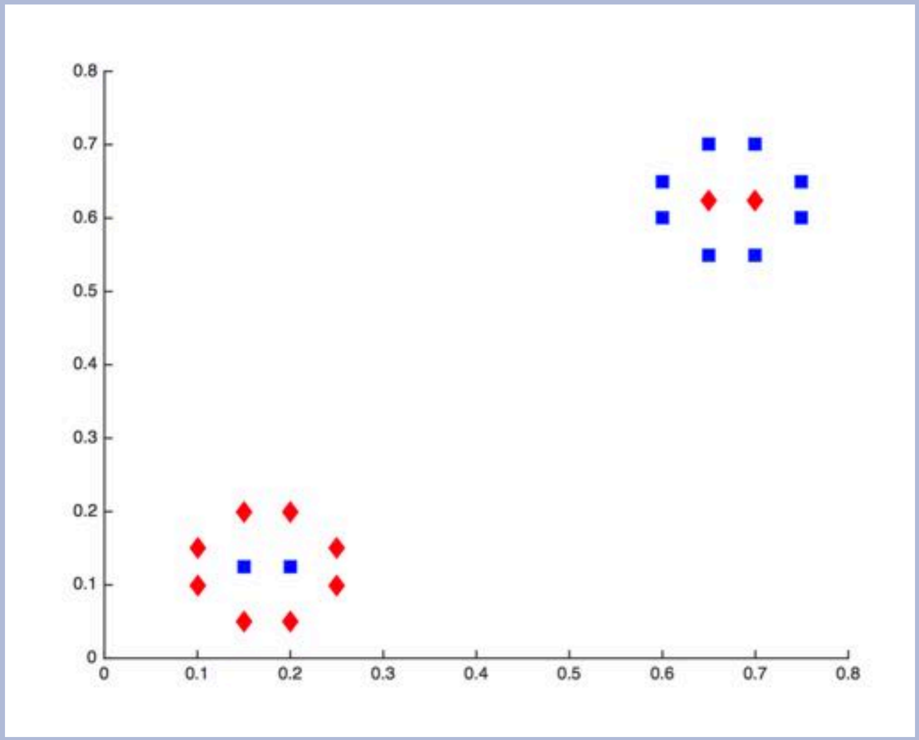



Figure 5

3. **[2 pts]** What value of $k$ minimizes leave-one-out cross-validation error for the dataset shown in Figure 5? What is the resulting error?

# Sample Questions

## 4.1 True or False

Answer each of the following questions with **T** or **F** and **provide a one line justification**.

(a) [2 pts.] Consider two datasets $D^{(1)}$ and $D^{(2)}$ where $D^{(1)} = \{(x_1^{(1)}, y_1^{(1)}), ..., (x_n^{(1)}, y_n^{(1)})\}$ and $D^{(2)} = \{(x_1^{(2)}, y_1^{(2)}), ..., (x_m^{(2)}, y_m^{(2)})\}$ such that $x_i^{(1)} \in \mathbb{R}^{d_1}$, $x_i^{(2)} \in \mathbb{R}^{d_2}$. Suppose $d_1 > d_2$ and $n > m$. Then the maximum number of mistakes a perceptron algorithm will make is higher on dataset $D^{(1)}$ than on dataset $D^{(2)}$.

# Sample Questions

## 3.1 Linear regression

Consider the dataset $S$ plotted in Fig. 1 along with its associated regression line. For each of the altered data sets $S^{\text{new}}$ plotted in Fig. 3, indicate which regression line (relative to the original one) in Fig. 2 corresponds to the regression line for the new data set. Write your answers in the table below.

| Dataset | (a) | (b) | (c) | (d) | (e) |
|---|---|---|---|---|---|
| Regression line | | | | | |

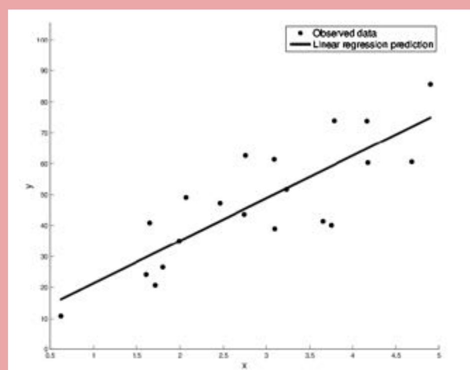
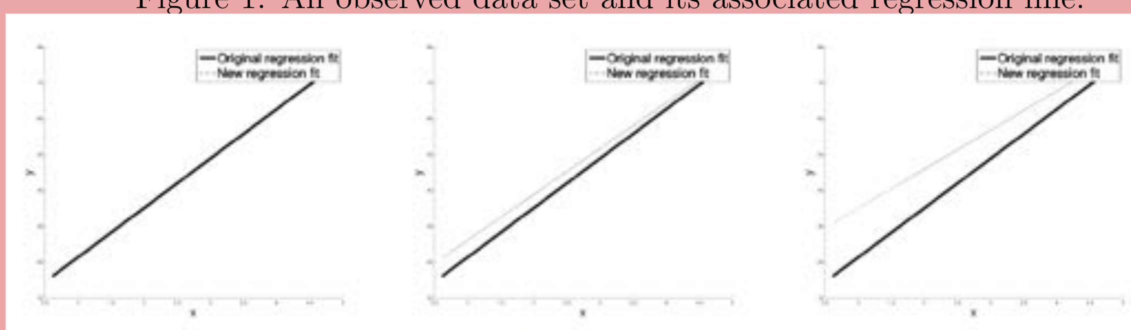


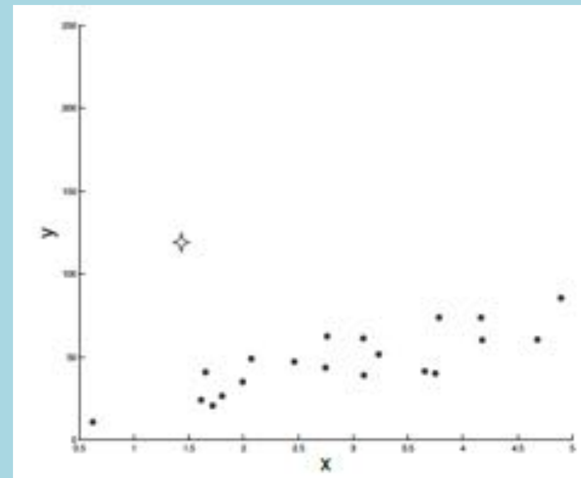

Figure 1: An observed data set and its associated regression line.




(a) Old and new regression lines. (b) Old and new regression lines. (c) Old and new regression lines.

Figure 2: New regression lines for altered data sets $S^{\text{new}}$.

## Dataset




(a) Adding one outlier to the original data set.

# Sample Questions

### 3.1 Linear regression

Consider the dataset $S$ plotted in Fig. 1 along with its associated regression line. For each of the altered data sets $S^{\text{new}}$ plotted in Fig. 3, indicate which regression line (relative to the original one) in Fig. 2 corresponds to the regression line for the new data set. Write your answers in the table below.

| Dataset | (a) | (b) | (c) | (d) | (e) |
|---|---|---|---|---|---|
| Regression line | | | | | |

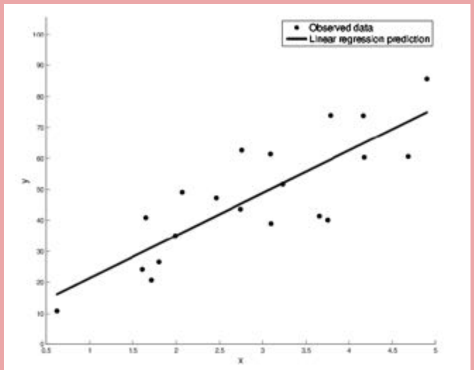
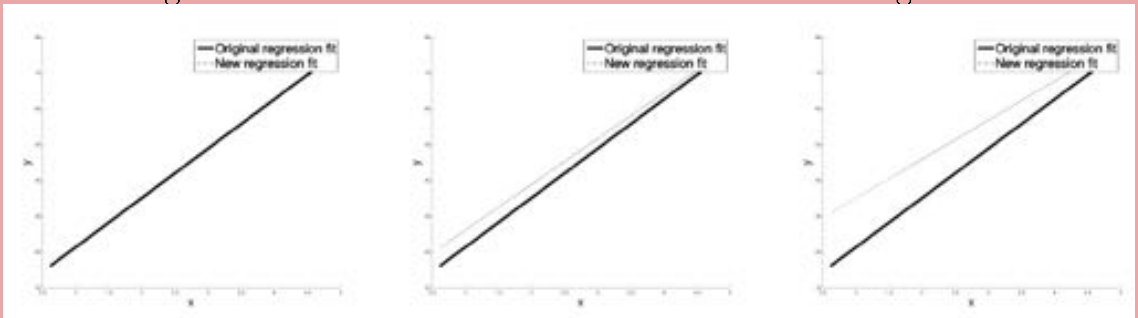


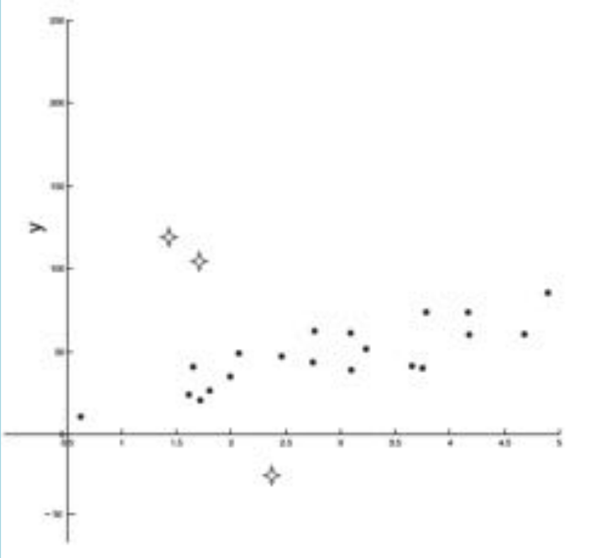

Figure 1: An observed data set and its associated regression line.




(a) Old and new regression lines. (b) Old and new regression lines. (c) Old and new regression lines.

Figure 2: New regression lines for altered data sets $S^{\text{new}}$.

Dataset



(c) Adding three outliers to the original data set. Two on one side and one on the other side.

# Sample Questions

### 3.1 Linear regression

Consider the dataset $S$ plotted in Fig. 1 along with its associated regression line. For each of the altered data sets $S^{\text{new}}$ plotted in Fig. 3, indicate which regression line (relative to the original one) in Fig. 2 corresponds to the regression line for the new data set. Write your answers in the table below.

| Dataset | (a) | (b) | (c) | (d) | (e) |
|---|---|---|---|---|---|
| Regression line | | | | | |

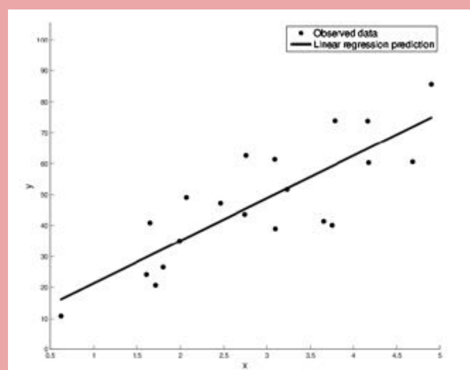
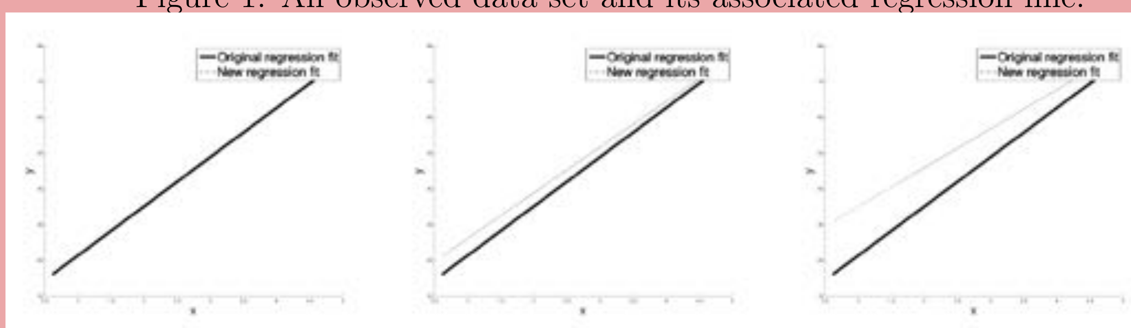


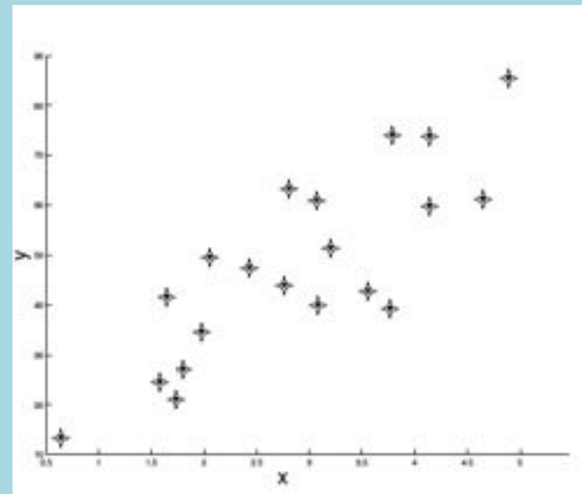

Figure 1: An observed data set and its associated regression line.




(a) Old and new regression lines. (b) Old and new regression lines. (c) Old and new regression lines.

Figure 2: New regression lines for altered data sets $S^{\text{new}}$.

## Dataset



(d) Duplicating the original data set.

# Sample Questions

### 3.1 Linear regression

Consider the dataset $S$ plotted in Fig. 1 along with its associated regression line. For each of the altered data sets $S^{new}$ plotted in Fig. 3, indicate which regression line (relative to the original one) in Fig. 2 corresponds to the regression line for the new data set. Write your answers in the table below.

| Dataset | (a) | (b) | (c) | (d) | (e) |
|---|---|---|---|---|---|
| Regression line | | | | | |

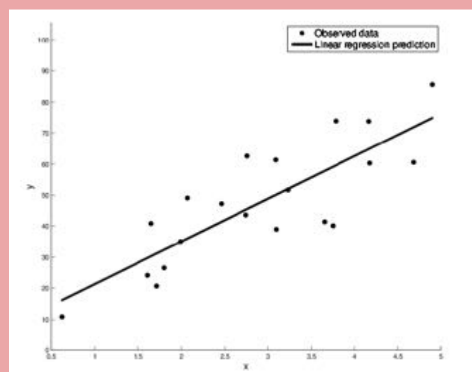
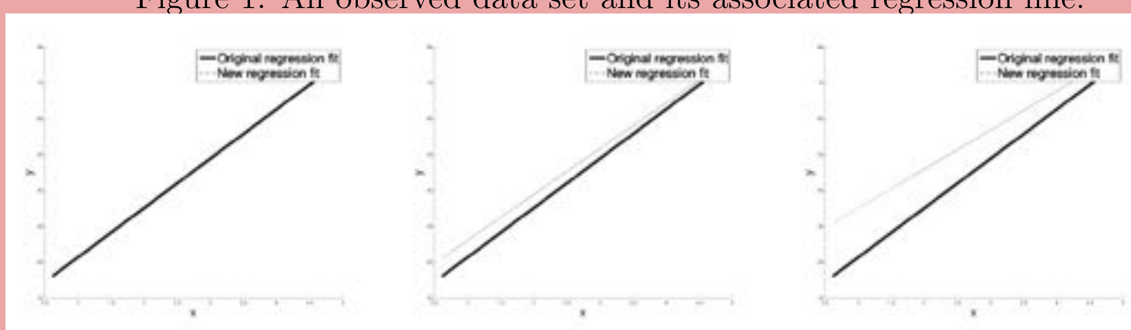


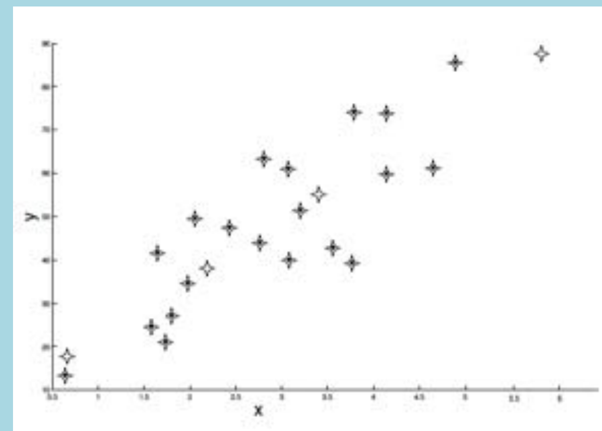

Figure 1: An observed data set and its associated regression line.




(a) Old and new regression lines. (b) Old and new regression lines. (c) Old and new regression lines.

Figure 2: New regression lines for altered data sets $S^{new}$.

## Dataset




(e) Duplicating the original data set and adding four points that lie on the trajectory of the original regression line.

# Matching Game

**Goal:** Match the Algorithm to its Update Rule

| | |
|---|---|
| **1. SGD for Logistic Regression** $h_{\boldsymbol{\theta}}(\mathbf{x}) = p(y\|x)$ | **4.** $\theta_k \leftarrow \theta_k + (h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})$ |
| **2. Least Mean Squares** $h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x}$ | **5.** $\theta_k \leftarrow \theta_k + \frac{1}{1 + \exp \lambda(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})}$ |
| **3. Perceptron** $h_{\boldsymbol{\theta}}(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$ | **6.** $\theta_k \leftarrow \theta_k + \lambda(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})x_k^{(i)}$ |

A. 1=5, 2=4, 3=6
B. 1=5, 2=6, 3=4
C. 1=6, 2=4, 3=4
D. 1=5, 2=6, 3=6
E. 1=6, 2=6, 3=6
F. 1=6, 2=5, 3=5
G. 1=5, 2=5, 3=5
H. 1=4, 2=5, 3=6

# Q&A

# PROBABILISTIC LEARNING

# Maximum Likelihood Estimation

The principle of Maximum likelihood estimator (MLE):

Choose parameters that make the data "most likely".

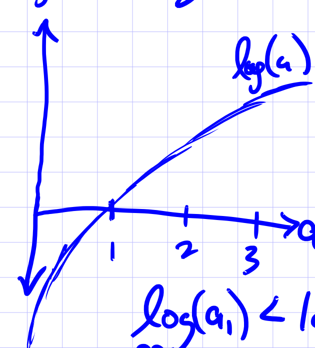
Assumptions: Data generated iid from distribution $p^*(x \mid \vec{\theta}^*)$ and comes from a family of distn parameterized $\theta \in \Theta$ ← set of possible parameters

Formally:

$$\theta_{MLE} = \underset{\theta \in \Theta}{\operatorname{argmax}}\; p(D \mid \theta)$$

$$= \underset{\theta \in \Theta}{\operatorname{argmax}}\; \log p(D \mid \theta)$$

since log is monotonic




$\log(a_1) < \log(a_2)$ iff $a_1 < a_2$

$\Rightarrow \log(f(a_1)) < \log(f(a_2))$ iff $f(a_1) < f(a_2)$

$$= \underset{\theta \in \Theta}{\operatorname{argmax}}\; \ell(\theta)$$

usually a continuous optimization

where $\ell(\theta) \triangleq \log p(D \mid \theta)$

"log-likelihood"

treat as function of $\theta$ where $D$ is constant

# Learning from Data (Frequentist)

*Whiteboard*

- Principle of Maximum Likelihood Estimation (MLE)
- Strawmen:
  - Example: Bernoulli
  - Example: Gaussian
  - Example: Conditional #1
    (Bernoulli conditioned on Gaussian)
  - Example: Conditional #2
    (Gaussians conditioned on Bernoulli)

# LOGISTIC REGRESSION

# Logistic Regression

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

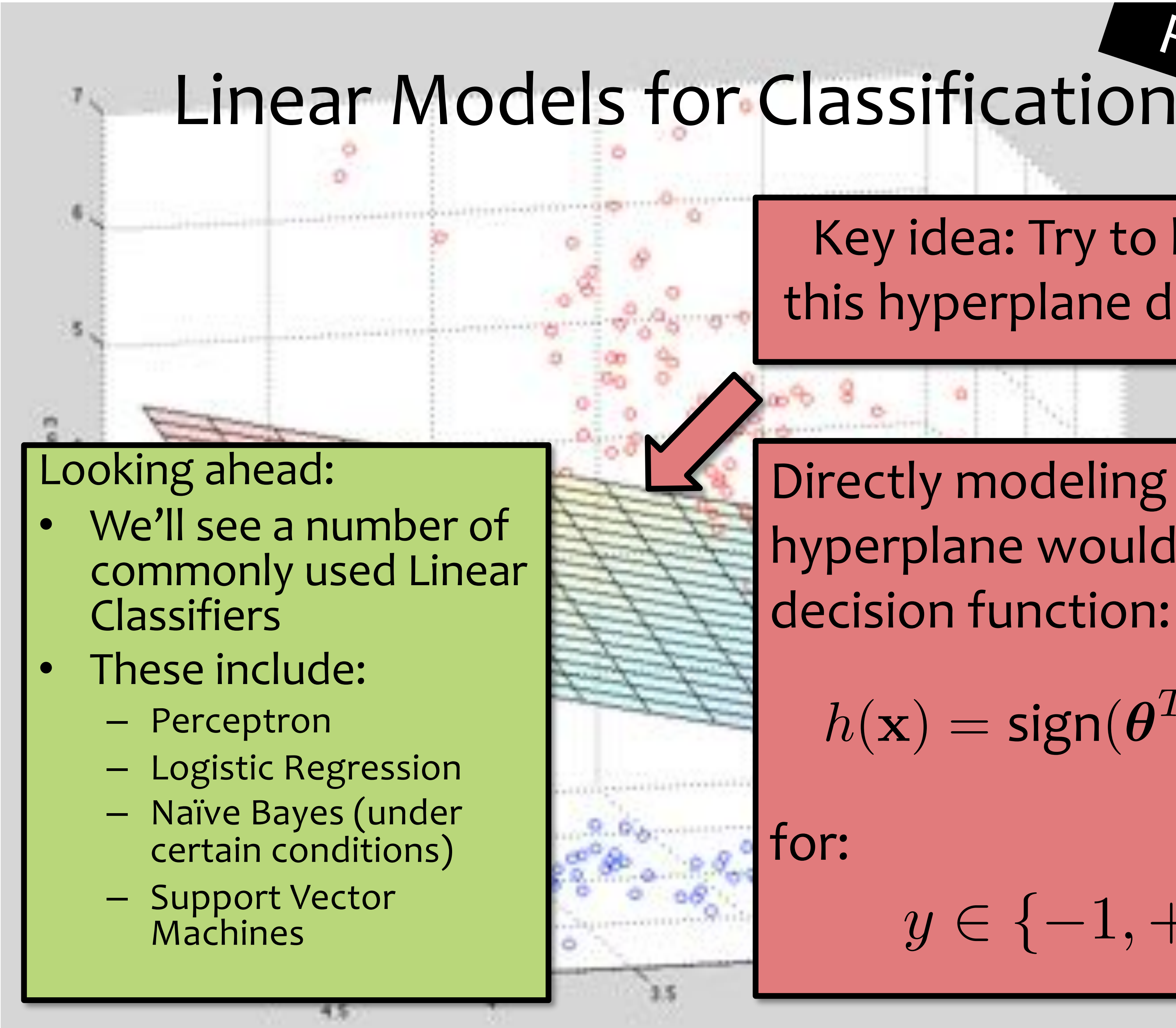
$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N} \text{ where } \mathbf{x} \in \mathbb{R}^M \text{ and } y \in \{0, 1\}$$

We are back to classification.

Despite the name logistic **regression.**

# Linear Models for Classification

Recall…



Key idea: Try to learn this hyperplane directly

Looking ahead:

- We'll see a number of commonly used Linear Classifiers
- These include:
  - Perceptron
  - Logistic Regression
  - Naïve Bayes (under certain conditions)
  - Support Vector Machines

Directly modeling the hyperplane would use a decision function:

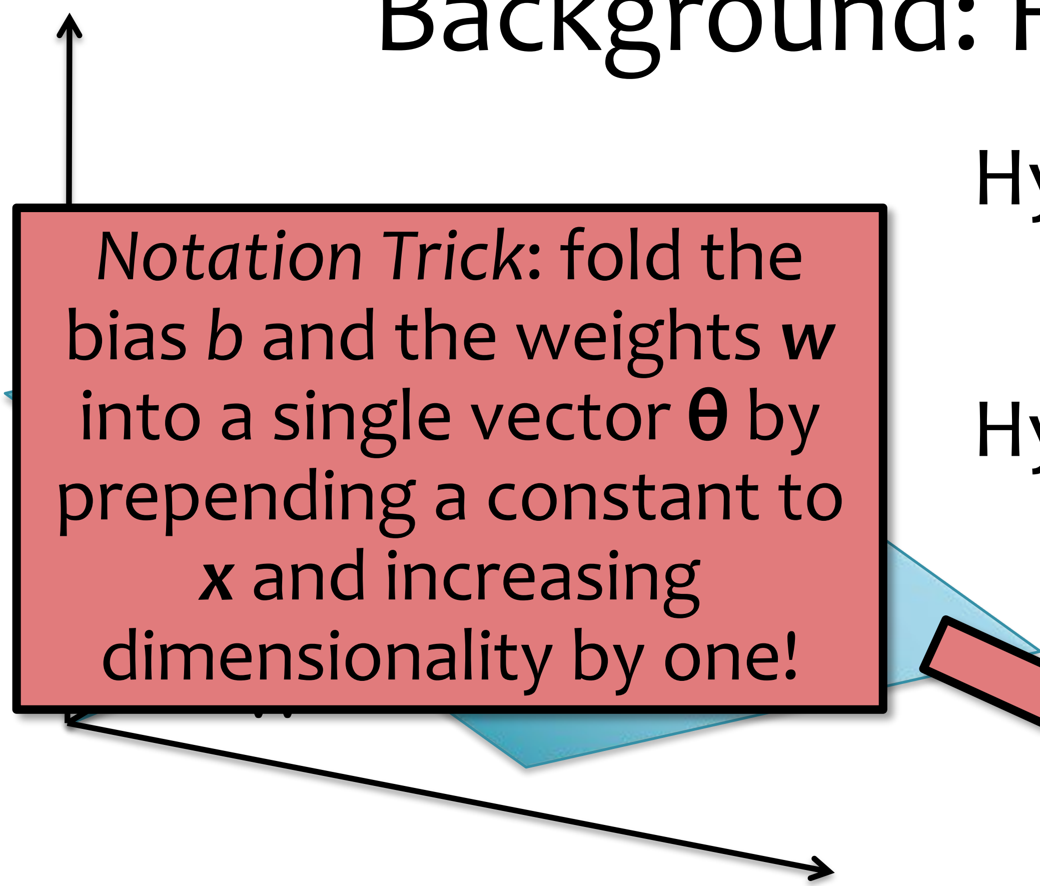
$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$

for:

$$y \in \{-1, +1\}$$

Recall...

# Background: Hyperplanes



*Notation Trick*: fold the bias *b* and the weights ***w*** into a single vector **θ** by prepending a constant to ***x*** and increasing dimensionality by one!

Hyperplane (Definition 1):

$$\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T \mathbf{x} = b\}$$

Hyperplane (Definition 2):

$$\mathcal{H} = \{\mathbf{x} : \boldsymbol{\theta}^T \mathbf{x} = 0 \text{ and } x_0 = 1\}$$

$$\boldsymbol{\theta} = [b, w_1, \ldots, w_M]^T$$

Half-spaces:

$$\mathcal{H}^+ = \{\mathbf{x} : \boldsymbol{\theta}^T \mathbf{x} > 0 \text{ and } x_0 = 1\}$$

$$\mathcal{H}^- = \{\mathbf{x} : \boldsymbol{\theta}^T \mathbf{x} < 0 \text{ and } x_0 = 1\}$$

# Using gradient ascent for linear classifiers

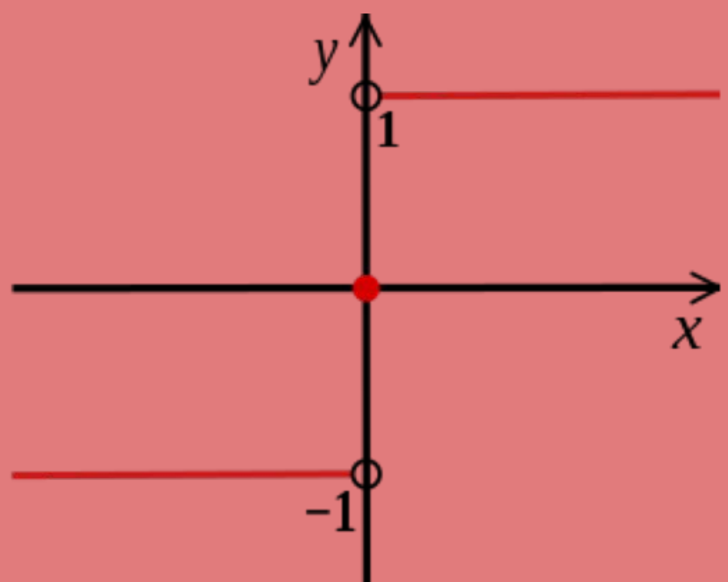
Key idea behind today's lecture:

1. Define a linear classifier (logistic regression)
2. Define an objective function (likelihood)
3. Optimize it with gradient descent to learn parameters
4. Predict the class with highest probability under the model

# Using gradient ascent for linear classifiers

This decision function isn't differentiable:

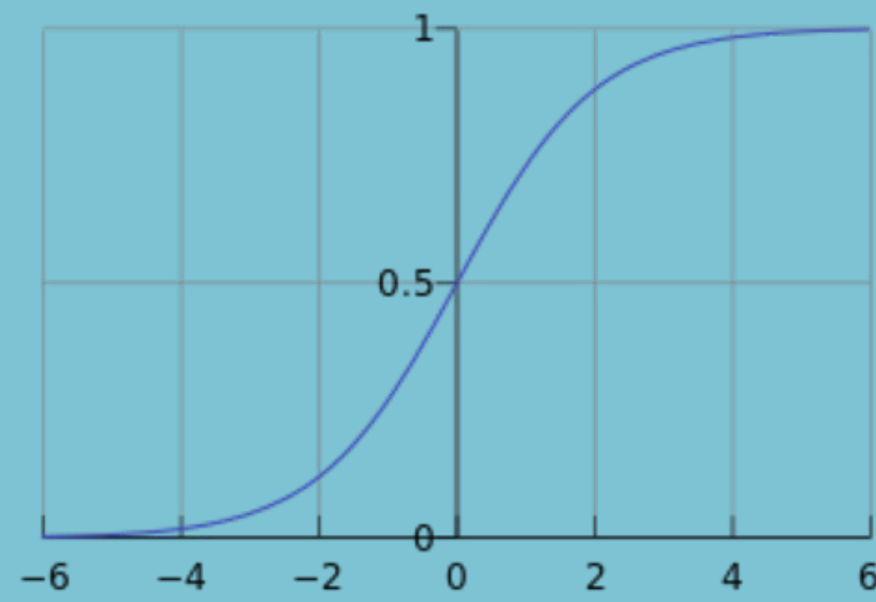
$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$




sign(x)

Use a differentiable function instead:

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

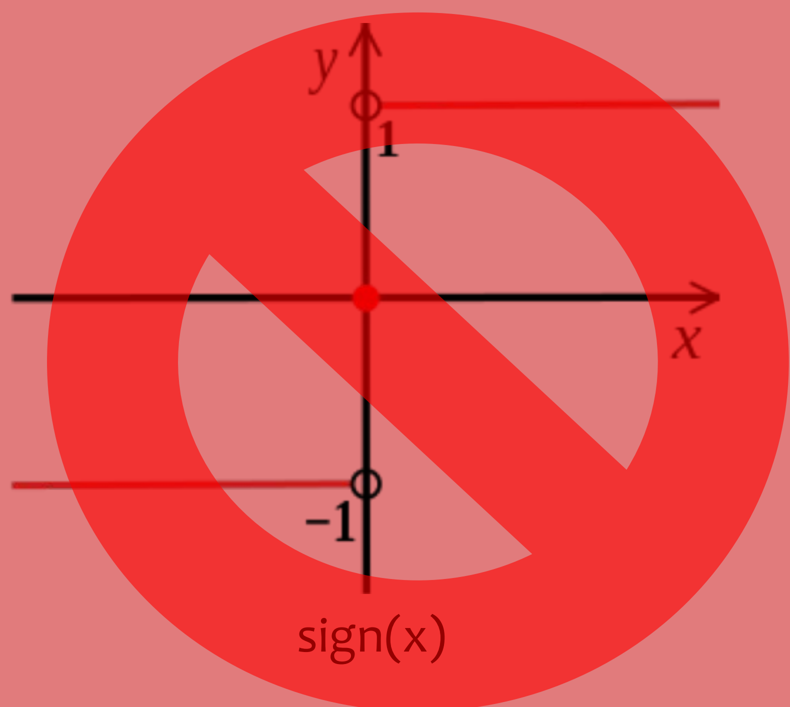



$$\text{logistic}(u) \equiv \frac{1}{1 + e^{-u}}$$

# Using gradient ascent for linear classifiers

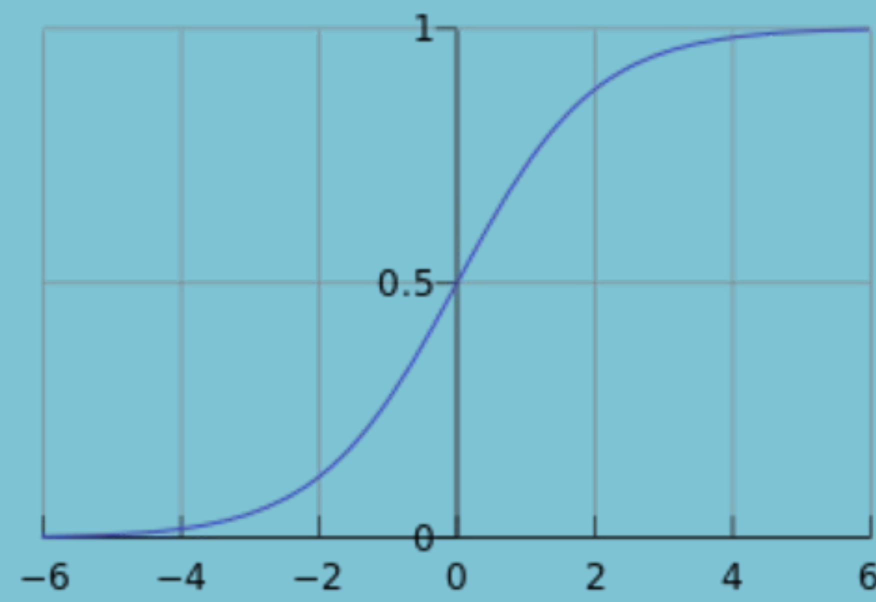
This decision function isn't differentiable:

$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$




Use a differentiable function instead:

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




$$\text{logistic}(u) \equiv \frac{1}{1+e^{-u}}$$

# Logistic Regression

*Whiteboard*

- Logistic Regression Model
- Learning for Logistic Regression
  - Partial derivative for Logistic Regression
  - Gradient for Logistic Regression

# 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.

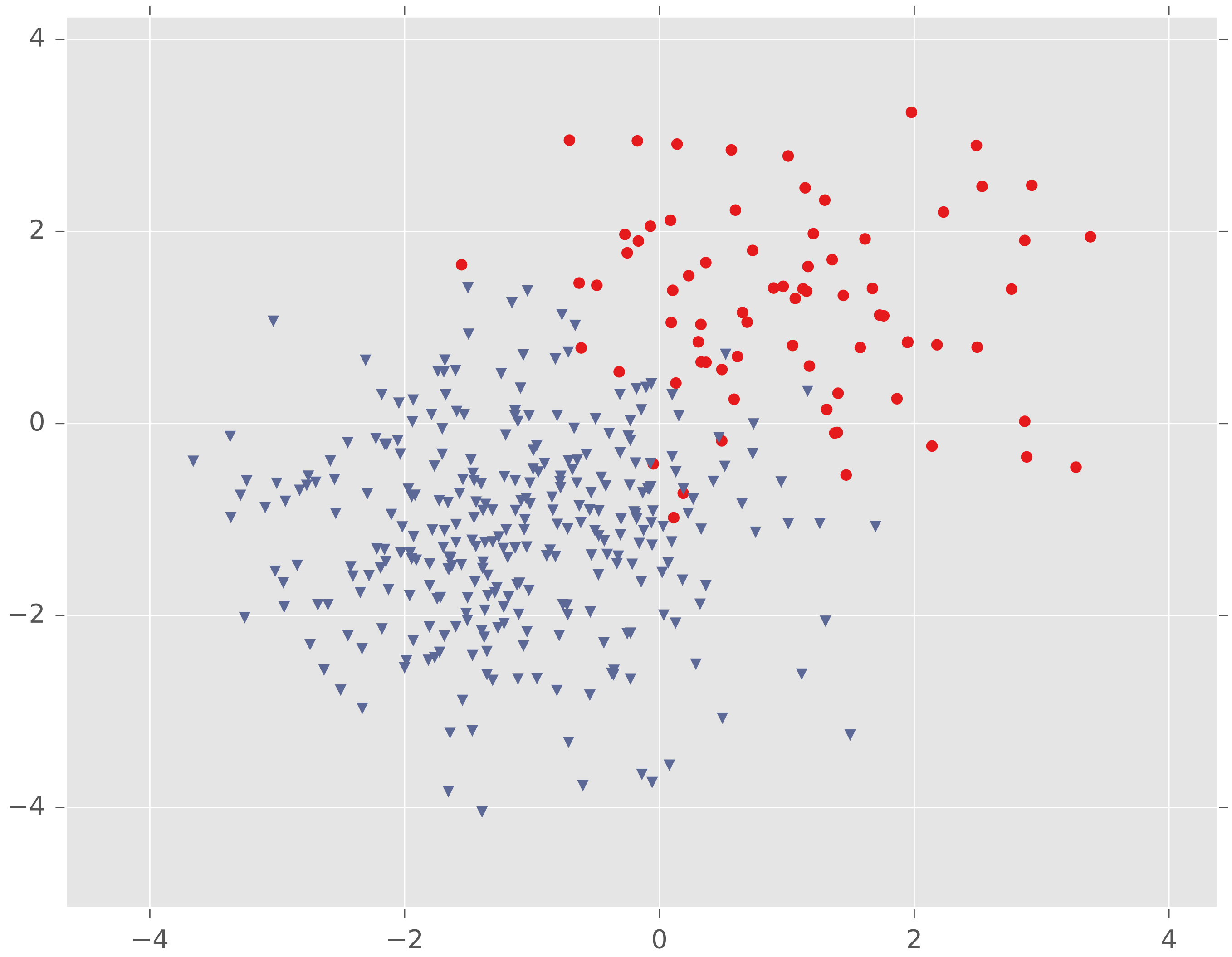
$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

**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.

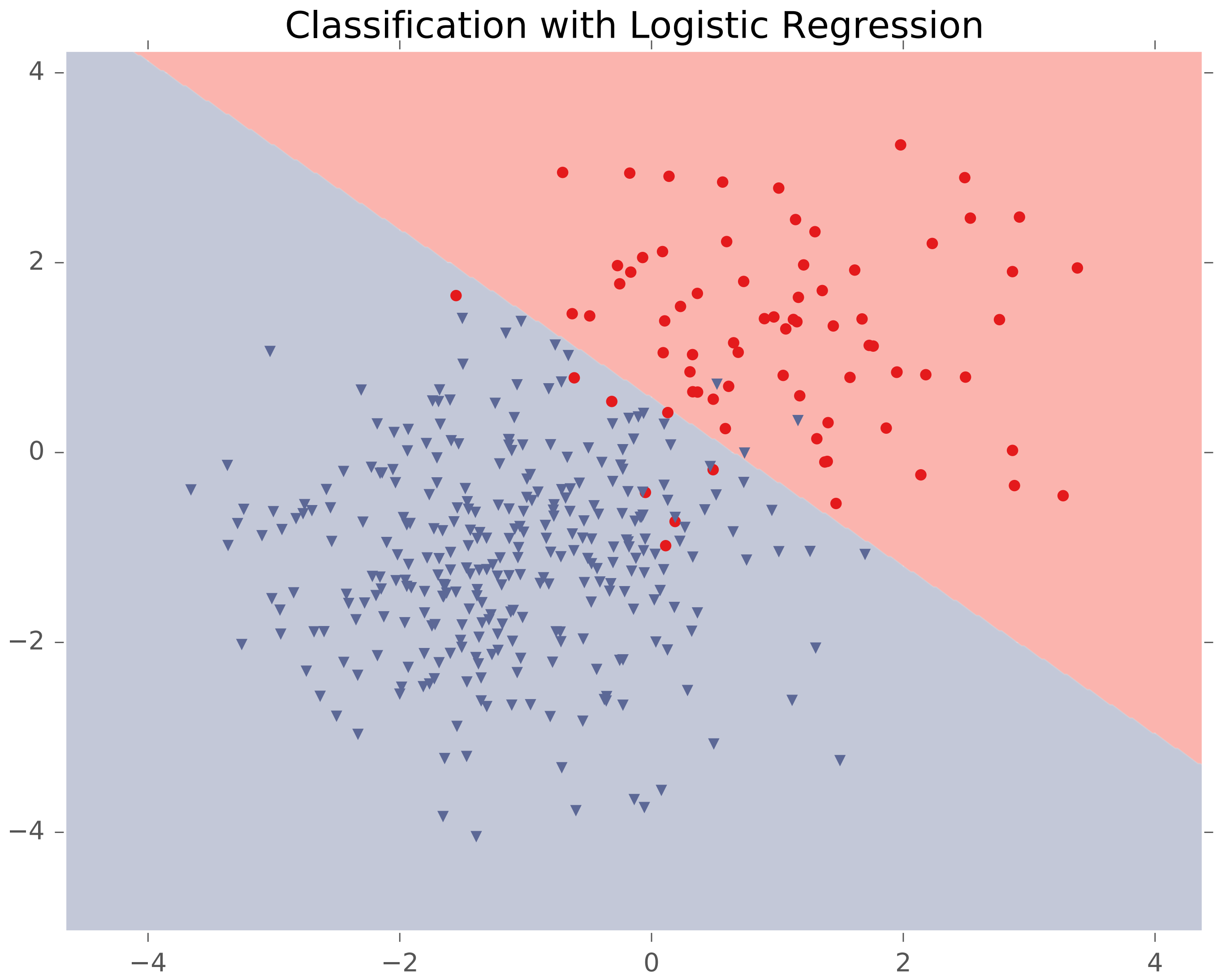
$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}}(y|\mathbf{x})$$

# Logistic Regression



# Logistic Regression

# Logistic Regression



# LEARNING LOGISTIC REGRESSION

# Maximum **Conditional** Likelihood Estimation

**Learning:** finds the parameters that minimize some objective function.

$$\boldsymbol{\theta}^* = \operatorname*{argmin}_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

We minimize the *negative* log conditional likelihood:

$$J(\boldsymbol{\theta}) = -\log \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(y^{(i)} | \mathbf{x}^{(i)})$$

Why?

1. We can't maximize likelihood (as in Naïve Bayes) because we don't have a joint model p(x,y)
2. It worked well for Linear Regression (least squares is MCLE)

# Maximum **Conditional** Likelihood Estimation

**Learning:** Four approaches to solving $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\text{argmin}}\, J(\boldsymbol{\theta})$

**Approach 1:** Gradient Descent
(take larger – more certain – steps opposite the gradient)

**Approach 2:** Stochastic Gradient Descent (SGD)
(take many small steps opposite the gradient)

**Approach 3:** Newton's Method
(use second derivatives to better follow curvature)

**Approach 4:** Closed Form???
(set derivatives equal to zero and solve for parameters)

# Maximum **Conditional** Likelihood Estimation

**Learning:** Four approaches to solving $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\text{argmin}}\, J(\boldsymbol{\theta})$

**Approach 1:** Gradient Descent
(take larger – more certain – steps opposite the gradient)

**Approach 2:** Stochastic Gradient Descent (SGD)
(take many small steps opposite the gradient)

**Approach 3:** Newton's Method
(use second derivatives to better follow curvature)

~~**Approach 4:** Closed Form???~~
~~(set derivatives equal to zero and solve for parameters)~~

Logistic Regression does not have a closed form solution for MLE parameters.

# SGD for Logistic Regression

**Question:**

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

**Answer:**

At each step (i.e. iteration) of SGD for Logistic Regression we…

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) ask Matt for a description of SGD for Logistic Regression, (2) write it down, (3) report that answer

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

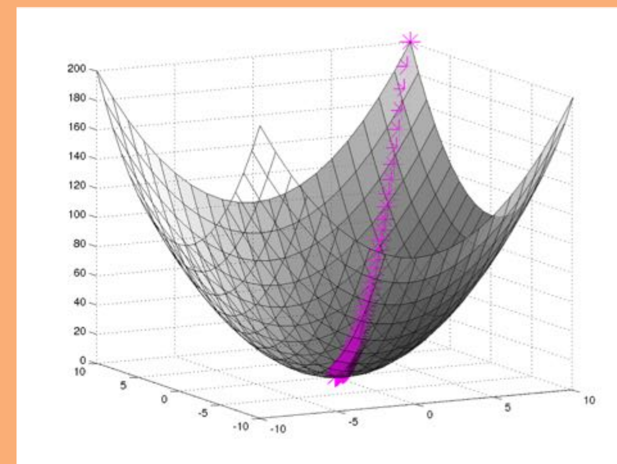
F. (1) randomly pick a parameter and an example, (2) compute the gradient of the log-likelihood for that example with respect to that parameter, (3) update that parameter using that gradient

Recall...

# Gradient Descent

**Algorithm 1** Gradient Descent

$$
\begin{array}{l}
\textbf{procedure } \mathrm{GD}(\mathcal{D}, \boldsymbol{\theta}^{(0)}) \\
\quad \boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)} \\
\quad \textbf{while } \text{not converged } \textbf{do} \\
\quad\quad \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \\
\quad \textbf{return } \boldsymbol{\theta}
\end{array}
$$



In order to apply GD to Logistic Regression all we need is the **gradient** of the objective function (i.e. vector of partial derivatives).

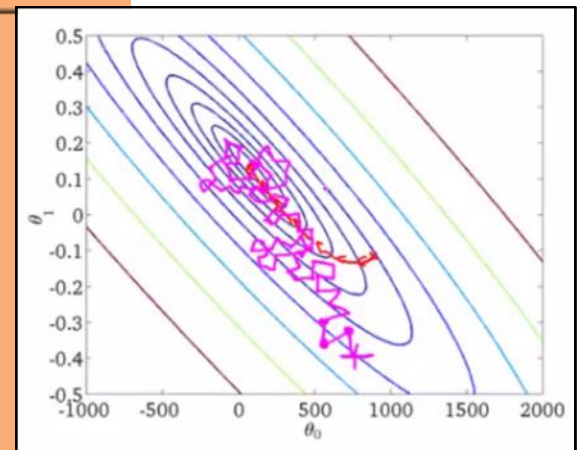
$$
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \begin{bmatrix} \frac{d}{d\theta_1} J(\boldsymbol{\theta}) \\ \frac{d}{d\theta_2} J(\boldsymbol{\theta}) \\ \vdots \\ \frac{d}{d\theta_M} J(\boldsymbol{\theta}) \end{bmatrix}
$$

Recall...

# Stochastic Gradient Descent (SGD)

**Algorithm 1** Stochastic Gradient Descent (SGD)

**procedure** SGD($\mathcal{D}, \boldsymbol{\theta}^{(0)}$)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
    **while** not converged **do**
        **for** $i \in$ shuffle($\{1, 2, \ldots, N\}$) **do**
            $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda \nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta})$
    **return** $\theta$




We can also apply SGD to solve the MCLE problem for Logistic Regression.

We need a per-example objective:

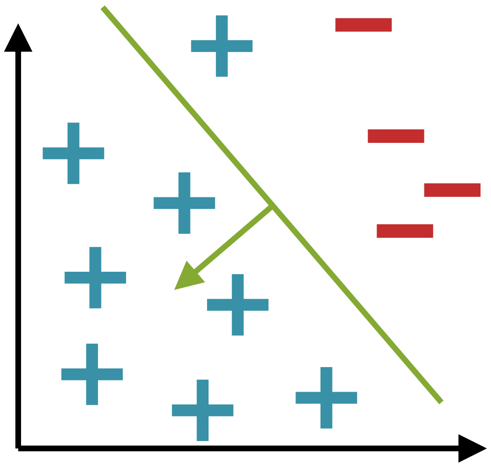
$$\text{Let } J(\boldsymbol{\theta}) = \sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$
$$\text{where } J^{(i)}(\boldsymbol{\theta}) = -\log p_{\boldsymbol{\theta}}(y^i | \mathbf{x}^i).$$

# Logistic Regression vs. Perceptron

## Question:

**True or False:** Just like Perceptron, **one step** (i.e. iteration) **of SGD for Logistic Regression** will result in a change to the parameters **only** if the current example is **incorrectly** classified.

## Answer:



# Summary

1. Discriminative classifiers directly model the **conditional**, p(y|x)
2. Logistic regression is a **simple linear classifier**, that retains a **probabilistic** semantics
3. Parameters in LR are learned by **iterative optimization** (e.g. SGD)

# Logistic Regression Objectives

*You should be able to…*

- Apply the principle of maximum likelihood estimation (MLE) to learn the parameters of a probabilistic model
- Given a discriminative probabilistic model, derive the conditional log-likelihood, its gradient, and the corresponding Bayes Classifier
- Explain the practical reasons why we work with the **log** of the likelihood
- Implement logistic regression for binary or multiclass classification
- Prove that the decision boundary of binary logistic regression is linear
- For linear regression, show that the parameters which minimize squared error are equivalent to those that maximize conditional likelihood