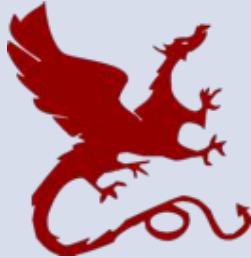


10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Logistic Regression

Matt Gormley
Lecture 8
Feb. 12, 2018



10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Logistic Regression

Probabilistic Learning

Matt Gormley
Lecture 8
Feb. 12, 2018

Reminders

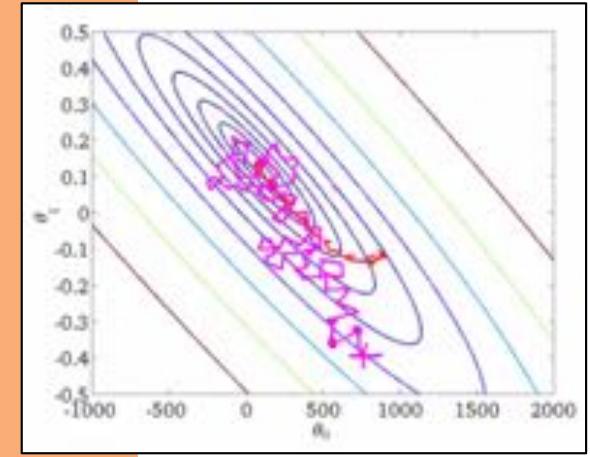
- Homework 3: KNN, Perceptron, Lin.Reg.
 - Out: Wed, Feb 7
 - Due: Wed, Feb 14 at 11:59pm

STOCHASTIC GRADIENT DESCENT

Stochastic Gradient Descent (SGD)

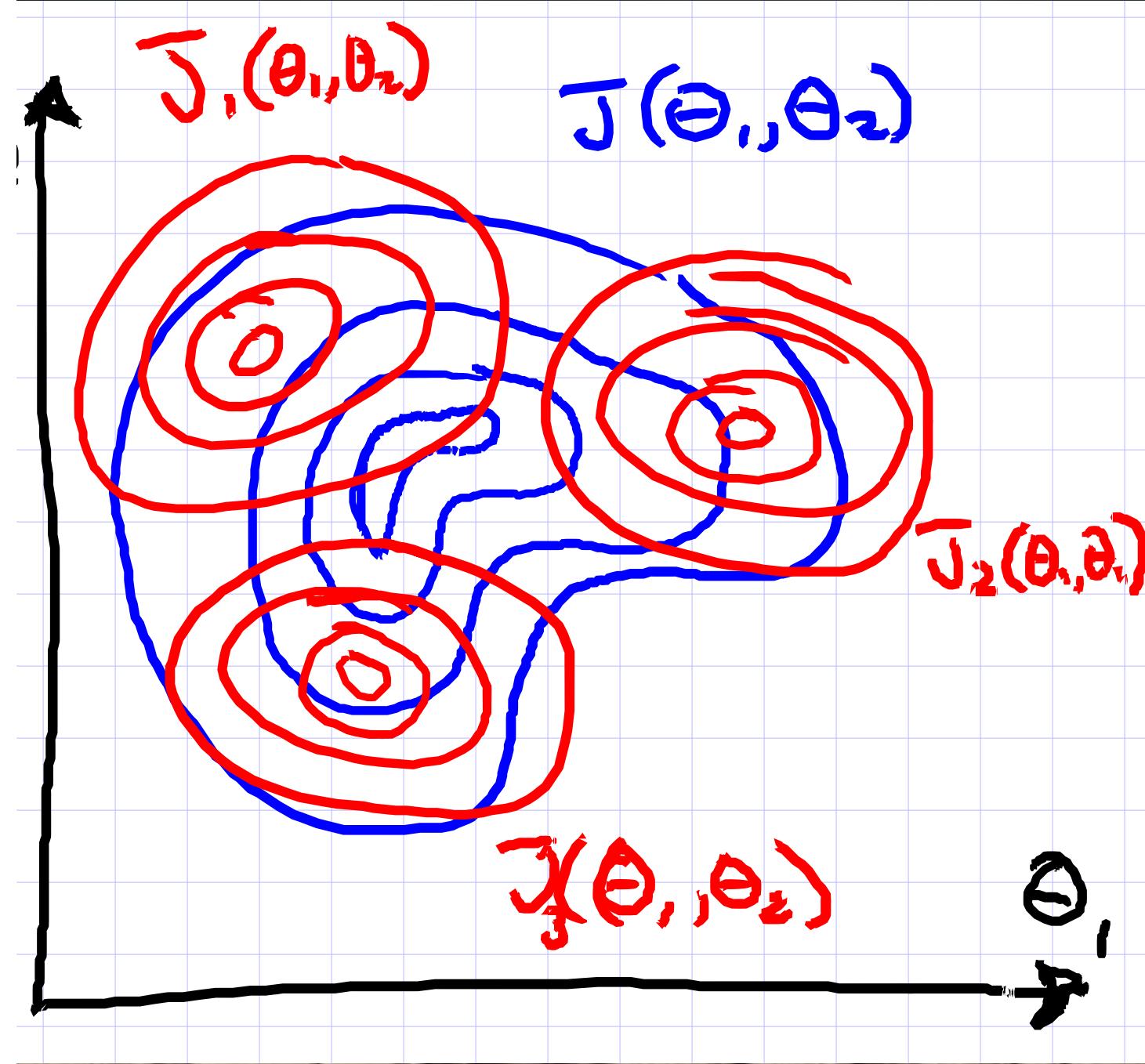
Algorithm 2 Stochastic Gradient Descent (SGD)

```
1: procedure SGD( $\mathcal{D}$ ,  $\theta^{(0)}$ )
2:    $\theta \leftarrow \theta^{(0)}$ 
3:   while not converged do
4:     for  $i \in \text{shuffle}(\{1, 2, \dots, N\})$  do
5:        $\theta \leftarrow \theta - \lambda \nabla_{\theta} J^{(i)}(\theta)$ 
6:   return  $\theta$ 
```



We need a per-example objective:

Let $J(\theta) = \sum_{i=1}^N J^{(i)}(\theta)$



Expectations of Gradients

$$\frac{\partial J(\vec{\theta})}{\partial \theta_j} = \frac{1}{N} \sum_{i=1}^N \frac{\partial}{\partial \theta_j} (J_i(\vec{\theta}))$$

$$\nabla J(\vec{\theta}) = \begin{bmatrix} \cdot \\ \vdots \\ \cdot \\ \cdot \end{bmatrix} = \frac{1}{N} \sum_{i=1}^N \nabla J_i(\vec{\theta})$$

Recall : for any discrete r.v. X

$$E_X[f(x)] \triangleq \sum_x P(X=x) f(x)$$

Q: What is the expected value of a randomly chosen $\nabla J_i(\vec{\theta})$?

$$\text{Let } I \sim \text{Uniform}\{1, \dots, N\}$$

$$\Rightarrow P(I=i) = \frac{1}{N} \text{ if } i \in \{1, \dots, N\}$$

$$\begin{aligned} E_I[\nabla J_I(\vec{\theta})] &= \sum_{i=1}^N P(I=i) \nabla J_i(\vec{\theta}) \\ &\Rightarrow \frac{1}{N} \sum_{i=1}^N \nabla J_i(\vec{\theta}) \\ &= \nabla J(\vec{\theta}) \end{aligned}$$

Convergence of Optimizers

Convergence Analysis:

Def: convergence is when $J(\vec{\theta}) - J(\vec{\theta}^*) < \epsilon$

→ true unknown min

Methods	Steps to Converge	Computation per iteration
Newton's Method	$O(\ln \ln \frac{1}{\epsilon})$	$\nabla J(\theta) \quad \nabla^2 J(\theta) \leftarrow O(NM^2)$
GD	$O(\ln \frac{1}{\epsilon})$	$\nabla J(\theta) \leftarrow O(NM)$
SGD	$O(\frac{1}{\epsilon})$	$\nabla J_i(\theta) \leftarrow O(M)$

"almost sure" convergence
lots of caveats
and conditions

way less computation

| Takeaway: SGD has much slower asymptotic convergence.
but is often faster in practice.

Optimization Objectives

You should be able to...

- Apply gradient descent to optimize a function
- Apply stochastic gradient descent (SGD) to optimize a function
- Apply knowledge of zero derivatives to identify a closed-form solution (if one exists) to an optimization problem
- Distinguish between convex, concave, and nonconvex functions
- Obtain the gradient (and Hessian) of a (twice) differentiable function

Linear Regression Objectives

You should be able to...

- Design k-NN Regression and Decision Tree Regression
- Implement learning for Linear Regression using three optimization techniques: (1) closed form, (2) gradient descent, (3) stochastic gradient descent
- Choose a Linear Regression optimization technique that is appropriate for a particular dataset by analyzing the tradeoff of computational complexity vs. convergence speed
- Distinguish the three sources of error identified by the bias-variance decomposition: bias, variance, and irreducible error.

PROBABILISTIC LEARNING

Probabilistic Learning

Function Approximation

Previously, we assumed that our output was generated using a **deterministic target function**:

$$\mathbf{x}^{(i)} \sim p^*(\cdot)$$

$$y^{(i)} = c^*(\mathbf{x}^{(i)})$$

Our goal was to learn a hypothesis $h(\mathbf{x})$ that best approximates $c^*(\mathbf{x})$

Probabilistic Learning

Today, we assume that our output is **sampled** from a **conditional probability distribution**:

$$\mathbf{x}^{(i)} \sim p^*(\cdot)$$

$$y^{(i)} \sim p^*(\cdot | \mathbf{x}^{(i)})$$

Our goal is to learn a probability distribution $p(y|\mathbf{x})$ that best approximates $p^*(y|\mathbf{x})$

Robotic Farming

	Deterministic	Probabilistic
Classification (binary output)	Is this a picture of a wheat kernel?	Is this plant drought resistant?
Regression (continuous output)	How many wheat kernels are in this picture?	What will the yield of this plant be?



Maximum Likelihood Estimation

The principle of Maximum likelihood estimator (MLE):

Choose parameters that make the data "most likely".

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

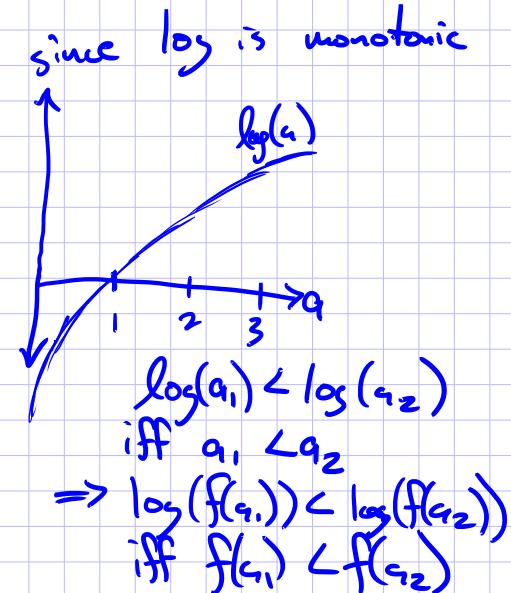
Formally:

$$\begin{aligned}\hat{\theta}_{MLE} &= \underset{\theta \in \mathbb{H}}{\operatorname{argmax}} p(D|\theta) \\ &= \underset{\theta \in \mathbb{H}}{\operatorname{argmax}} \log p(D|\theta)\end{aligned}$$

$$= \underset{\theta \in \mathbb{H}}{\operatorname{argmax}} l(\theta)$$

where $l(\theta) \triangleq \log p(D|\theta)$
↳ "log-likelihood"

usually
continuous
optimization



↑ treat as function of θ
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)

Outline

- **Motivation:**
 - Choosing the right classifier
 - Example: Image Classification
- **Logistic Regression**
 - Background: Hyperplanes
 - Data, Model, Learning, Prediction
 - Log-odds
 - Bernoulli interpretation
 - Maximum Conditional Likelihood Estimation
- **Gradient descent for Logistic Regression**
 - Stochastic Gradient Descent (SGD)
 - Computing the gradient
 - Details (learning rate, finite differences)

MOTIVATION: LOGISTIC REGRESSION

Example: Image Classification

- ImageNet LSVRC-2010 contest:
 - **Dataset:** 1.2 million labeled images, 1000 classes
 - **Task:** Given a new image, label it with the correct class
 - **Multiclass** classification problem
- Examples from <http://image-net.org/>

Bird

Warm-blooded egg-laying vertebrates characterized by feathers and forelimbs modified as wings

2126 pictures
92.85% Popularity Percentile



- marine animal, marine creature, sea animal, sea creature (1)
- scavenger (1)
- biped (0)
- predator, predatory animal (1)
- larva (49)
- acrodont (0)
- feeder (0)
- stunt (0)
- chordate (3087)
 - tunicate, urochordate, urochord (6)
 - cephalochordate (1)
 - vertebrate, craniate (3077)
 - mammal, mammalian (1169)
 - bird (871)
 - dickeybird, dickey-bird, dickybird, dicky-bird (0)
 - cock (1)
 - hen (0)
 - nester (0)
 - night bird (1)
 - bird of passage (0)
 - protoavis (0)
 - archaeopteryx, archeopteryx, Archaeopteryx lithographica, Sinornis (0)
 - ibero-mesornis (0)
 - archaeornis (0)
 - ratite, ratite bird, flightless bird (10)
 - carinate, carinate bird, flying bird (0)
 - passerine, passeriform bird (279)
 - nonpasserine bird (0)
 - bird of prey, raptor, raptorial bird (80)
 - gallinaceous bird, gallinacean (114)



German iris, Iris kochii

Iris of northern Italy having deep blue-purple flowers; similar to but smaller than Iris germanica

469
pictures49.6%
Popularity
Percentile

- halophyte (0)
- succulent (39)
- cultivar (0)
- cultivated plant (0)
- weed (54)
- evergreen, evergreen plant (0)
- deciduous plant (0)
- vine (272)
- creeper (0)
- woody plant, ligneous plant (1868)
- geophyte (0)
- desert plant, xerophyte, xerophytic plant, xerophile, xerophytic mesophyte, mesophysic plant (0)
- aquatic plant, water plant, hydrophyte, hydrophytic plant (11)
- tuberous plant (0)
- bulbous plant (179)
 - ↳ iridaceous plant (27)
 - ↳ iris, flag, fleur-de-lis, sword lily (19)
 - ↳ bearded iris (4)
 - Florentine iris, orris, Iris germanica florentina, Iris
 - German iris, Iris germanica (0)
 - German iris, Iris kochii (0)
 - Dalmatian iris, Iris pallida (0)
 - beardless iris (4)
 - bulbous iris (0)
 - dwarf iris, Iris cristata (0)
 - stinking iris, gladdon, gladdon iris, stinking gladwyn, Persian iris, Iris persica (0)
 - yellow iris, yellow flag, yellow water flag, Iris pseudacorus (0)
 - dwarf iris, vernal iris, Iris verna (0)
 - blue flag, Iris versicolor (0)



Court, courtyard

An area wholly or partly surrounded by walls or buildings; "the house was built around an inner court"

165 pictures
92.61% Popularity Percentile



Numbers in brackets: (the number of synsets in the subtree).

- ImageNet 2011 Fall Release (32326)

- plant, flora, plant life (4486)
- geological formation, formation (175)
- natural object (1112)
- sport, athletics (176)
- artifact, artefact (10504)
 - instrumentality, instrumentation (5494)
 - structure, construction (1405)
 - airdock, hangar, repair shed (0)
 - altar (1)
 - arcade, colonnade (1)
 - arch (31)
 - area (344)
 - aisle (0)
 - auditorium (1)
 - baggage claim (0)
 - box (1)
 - breakfast area, breakfast nook (0)
 - bullpen (0)
 - chancel, sanctuary, bema (0)
 - choir (0)
 - corner, nook (2)
 - court, courtyard (6)
 - atrium (0)
 - bailey (0)
 - cloister (0)
 - food court (0)
 - forecourt (0)
 - narvis (0)

Treemap Visualization



Images of the Synset

Downloads

Example: Image Classification

CNN for Image Classification

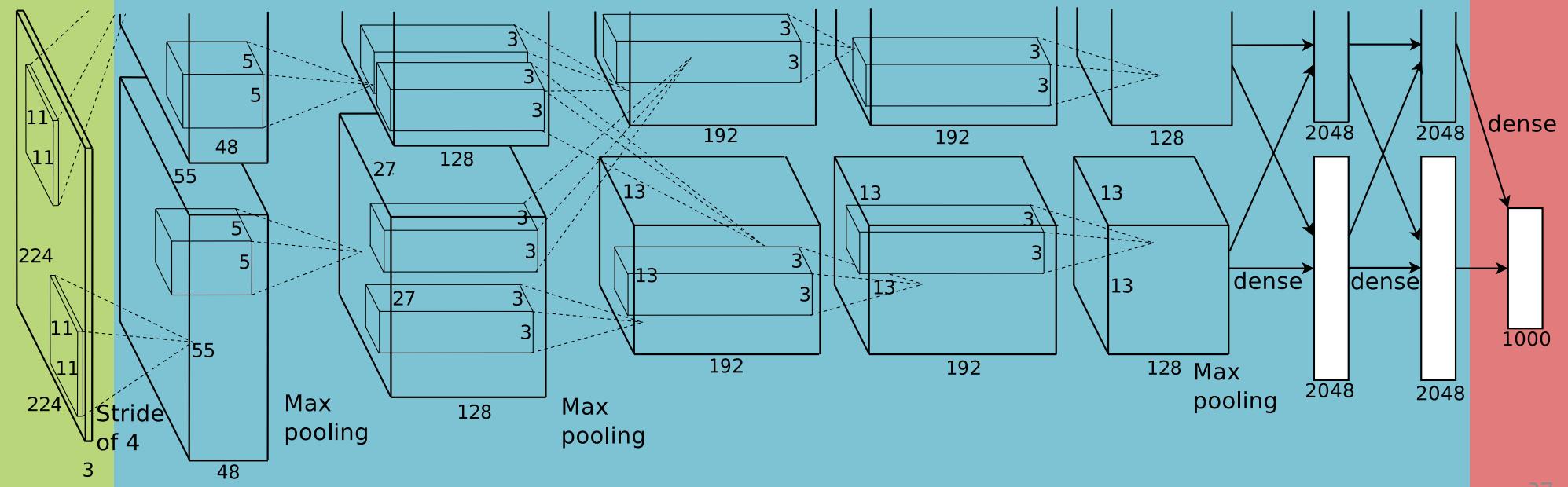
(Krizhevsky, Sutskever & Hinton, 2011)

17.5% error on ImageNet LSVRC-2010 contest

Input
image
(pixels)

- Five convolutional layers (w/max-pooling)
- Three fully connected layers

1000-way
softmax



Example: Image Classification

CNN for Image Classification

(Krizhevsky, Sutskever & Hinton, 2011)

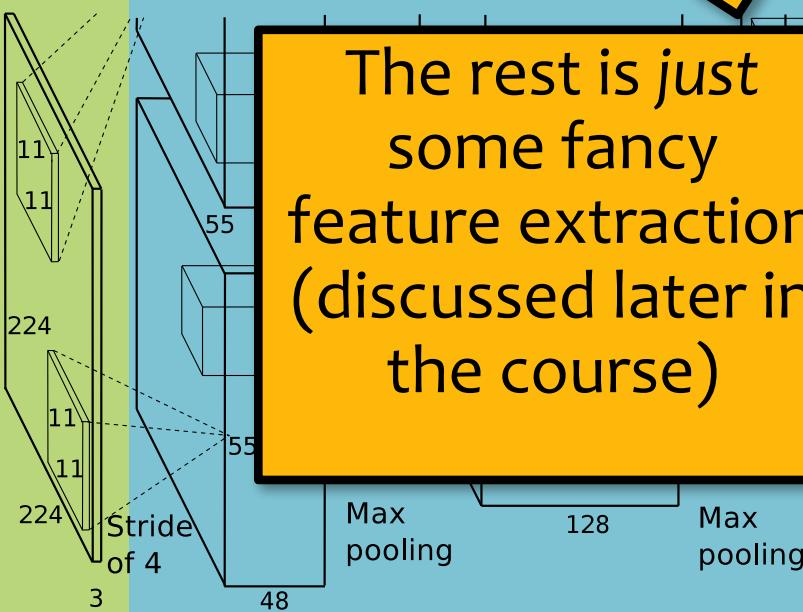
17.5% error on ImageNet LSVRC-2010 contest

Input
image
(pixels)

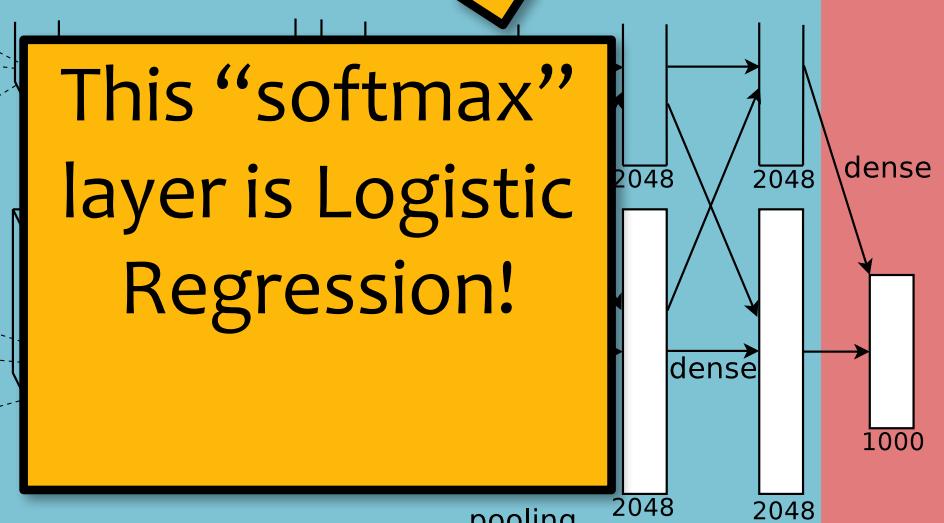
- Five convolutional layers (w/max-pooling)
- Three fully connected layers

1000-way
softmax

The rest is just
some fancy
feature extraction
(discussed later in
the course)



This “softmax”
layer is Logistic
Regression!



LOGISTIC REGRESSION

Logistic Regression

Data: Inputs are continuous vectors of length K. 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

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

Key idea: Try to learn this hyperplane directly



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):

$$\mathcal{H} = \{\mathbf{x} : \boldsymbol{\theta}^T \mathbf{x} = 0$$

$$\text{and } x_0 = 1\}$$

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

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

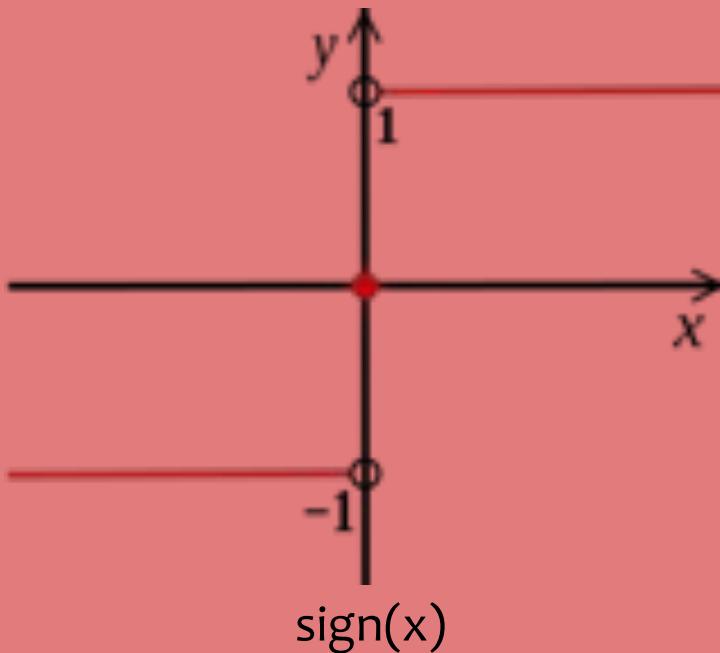
Key idea behind today's lecture:

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:

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



Use a differentiable function instead:

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

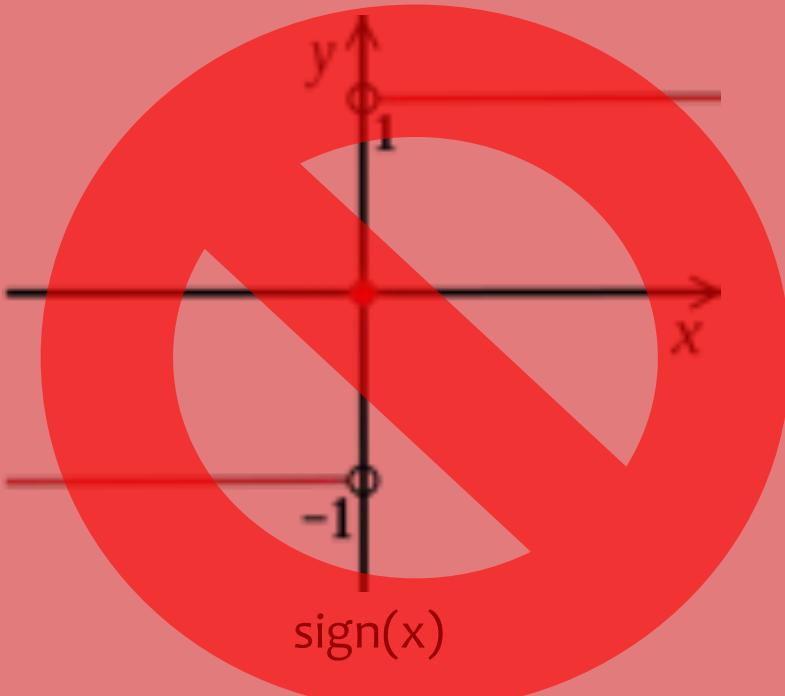


$$\text{logistic}(u) \equiv \frac{1}{1 + e^{-u}}$$

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

Data: Inputs are continuous vectors of length K. 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. $\boldsymbol{\theta}^* = \operatorname{argmin}_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

Prediction: Output is the most probable class.

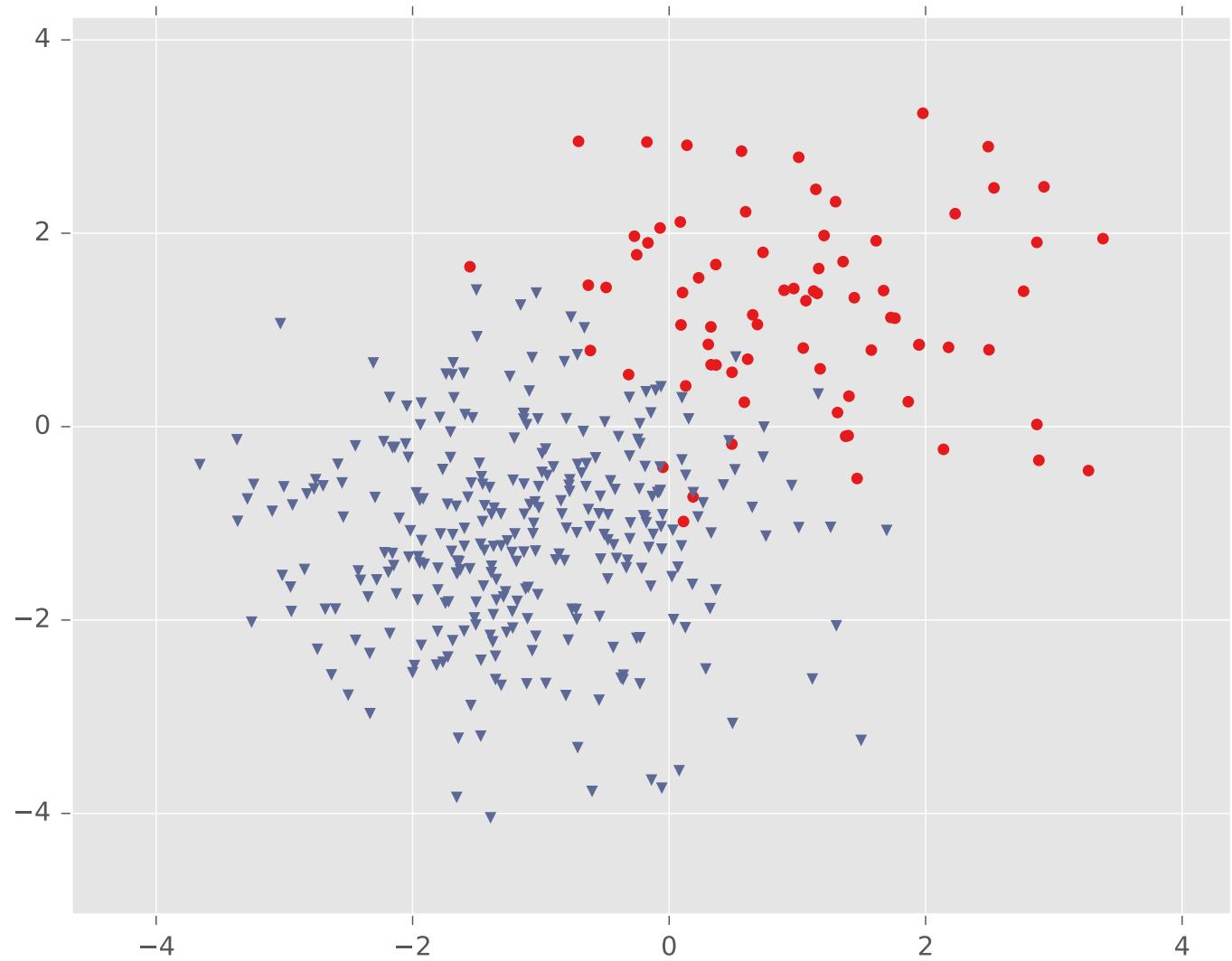
$$\hat{y} = \operatorname{argmax}_{y \in \{0, 1\}} p_{\boldsymbol{\theta}}(y | \mathbf{x})$$

Logistic Regression

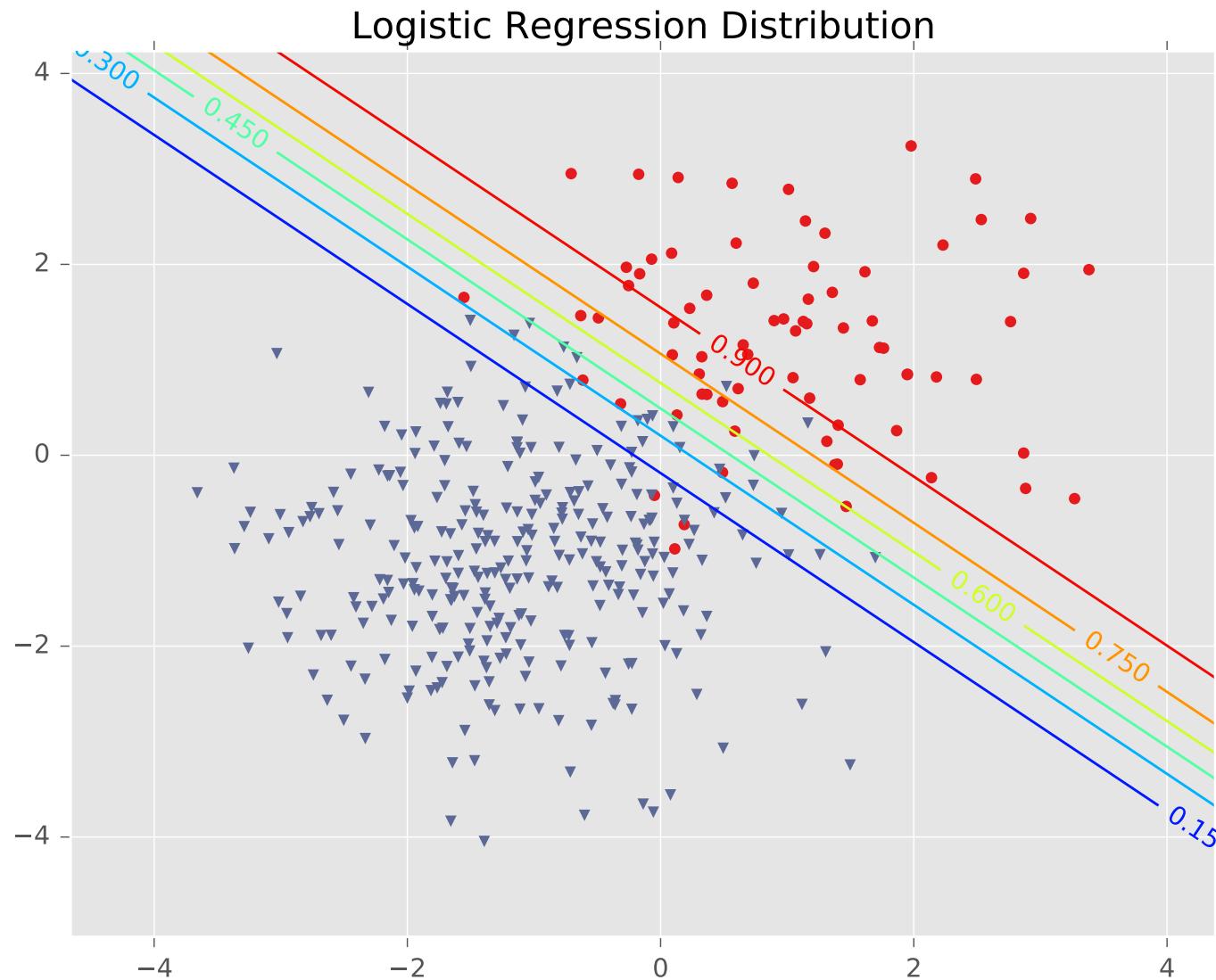
Whiteboard

- Bernoulli interpretation
- Logistic Regression Model
- Decision boundary

Logistic Regression

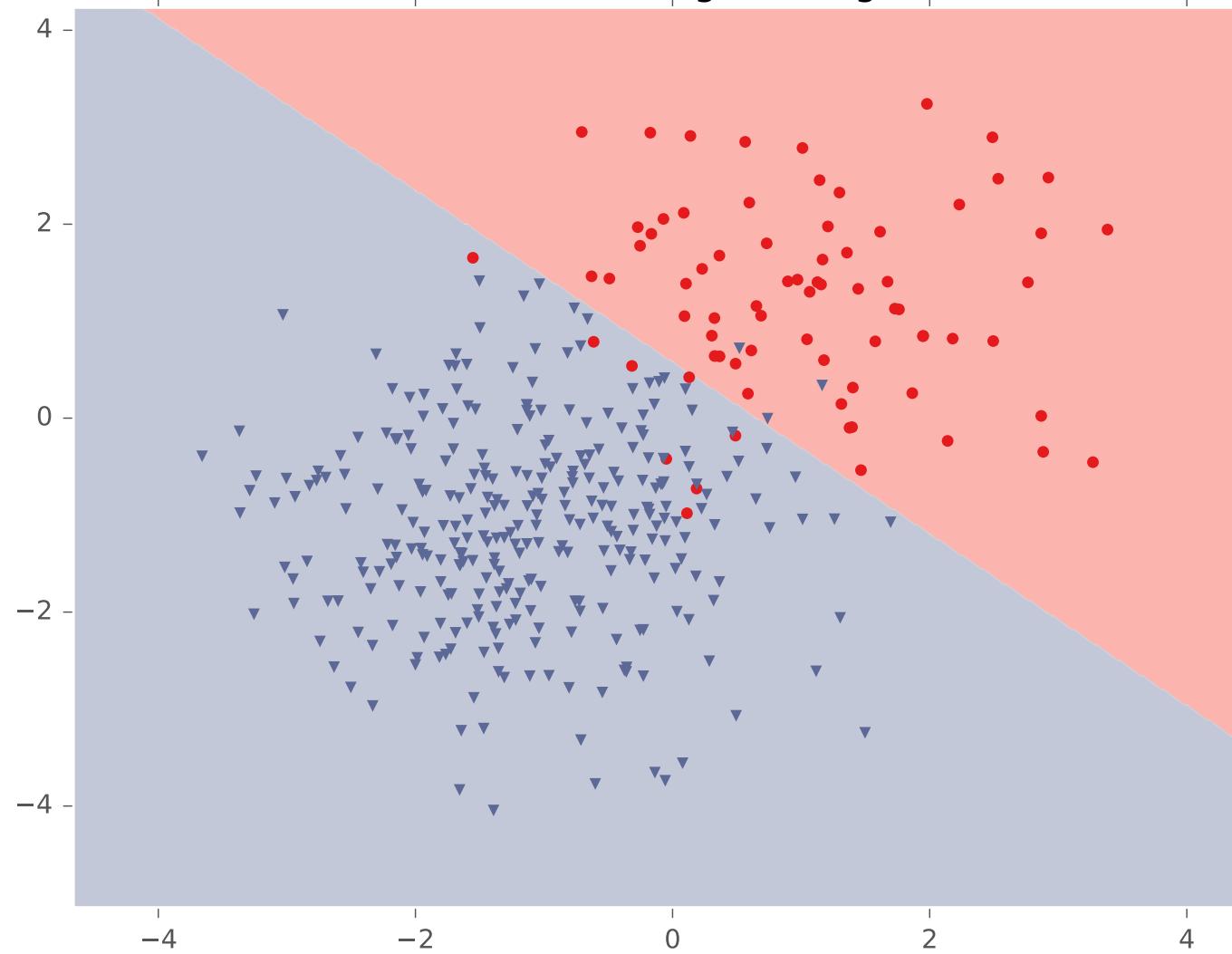


Logistic Regression



Logistic Regression

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

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

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)~~

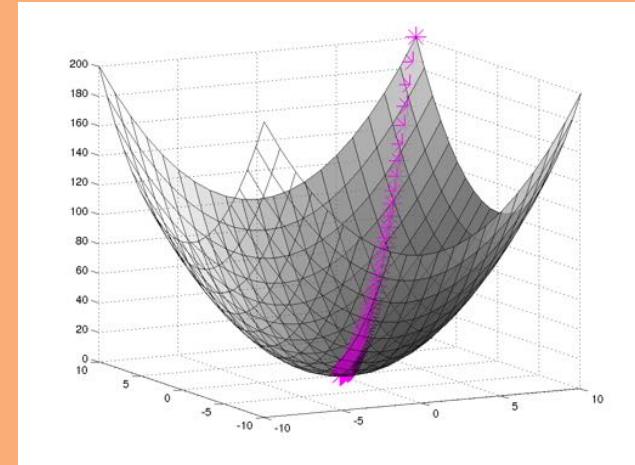
Logistic Regression does not have a closed form solution for MLE parameters.

Gradient Descent

Algorithm 1 Gradient Descent

```

1: procedure GD( $\mathcal{D}$ ,  $\theta^{(0)}$ )
2:    $\theta \leftarrow \theta^{(0)}$ 
3:   while not converged do
4:      $\theta \leftarrow \theta - \lambda \nabla_{\theta} J(\theta)$ 
5:   return  $\theta$ 
```



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

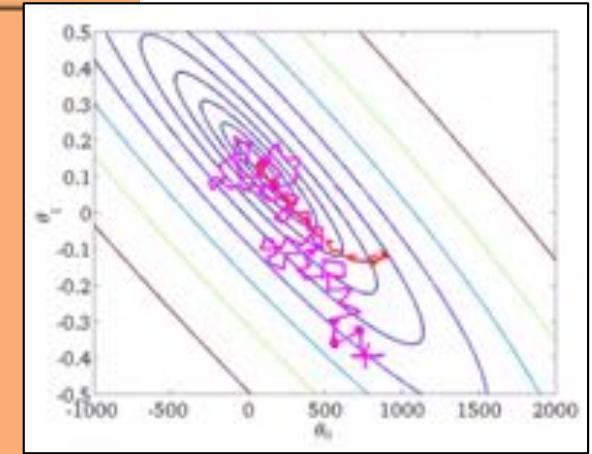
$$\nabla_{\theta} J(\theta) = \begin{bmatrix} \frac{d}{d\theta_1} J(\theta) \\ \frac{d}{d\theta_2} J(\theta) \\ \vdots \\ \frac{d}{d\theta_M} J(\theta) \end{bmatrix}$$

Recall...

Stochastic Gradient Descent (SGD)

Algorithm 1 Stochastic Gradient Descent (SGD)

```
1: procedure SGD( $\mathcal{D}$ ,  $\theta^{(0)}$ )
2:    $\theta \leftarrow \theta^{(0)}$ 
3:   while not converged do
4:     for  $i \in \text{shuffle}(\{1, 2, \dots, N\})$  do
5:        $\theta \leftarrow \theta - \lambda \nabla_{\theta} J^{(i)}(\theta)$ 
6:   return  $\theta$ 
```



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

We need a per-example objective:

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

GRADIENT FOR LOGISTIC REGRESSION

Learning for Logistic Regression

Whiteboard

- Partial derivative for Logistic Regression
- Gradient for Logistic Regression

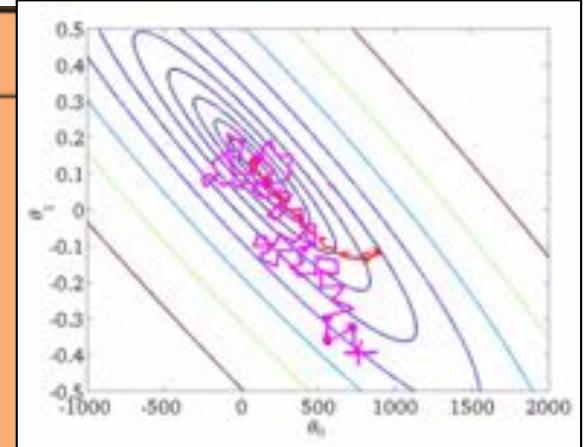
Details: Picking learning rate

- Use grid-search in log-space over small values on a tuning set:
 - e.g., 0.01, 0.001, ...
- Sometimes, decrease after each pass:
 - e.g factor of $1/(1 + dt)$, $t=\text{epoch}$
 - sometimes $1/t^2$
- Fancier techniques I won't talk about:
 - Adaptive gradient: scale gradient differently for each dimension (Adagrad, ADAM,)

SGD for Logistic Regression

Algorithm 1 SGD for Logistic Regression

```
1: procedure SGD( $\mathcal{D}, \theta^{(0)}$ )
2:    $\theta \leftarrow \theta^{(0)}$ 
3:   while not converged do
4:     for  $i \in \text{shuffle}(\{1, 2, \dots, N\})$  do
5:        $\theta \leftarrow \theta - \lambda(y^{(i)} - \rho^{(i)})\mathbf{x}^{(i)}$ 
6:       where  $\rho^{(i)} := 1/(1 + \exp(-\theta^T \mathbf{x}))$ 
7:   return  $\theta$ 
```



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

We need a per-example objective:

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

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)

Probabilistic Interpretation of Linear Regression

Whiteboard

- Conditional Likelihood
- Case #1: 1D Linear Regression
- Case #2: Multiple Linear Regression
- Equivalence: Predictions
- Equivalence: Learning