

#### **10-601 Introduction to Machine Learning**

Machine Learning Department School of Computer Science Carnegie Mellon University

### **Stochastic Gradient Descent**



# **Probabilistic Learning**

(Binary Logistic Regression)

Matt Gormley Lecture 9 Feb. 12, 2020

### Reminders

- Homework 3: KNN, Perceptron, Lin.Reg.
  - Out: Wed, Feb. 05 (+ 1 day)
  - Due: Fri, Feb. 14 at 11:59pm
     last possible moment to submit HW3:
     Sun, Feb. 16 at 11:59pm
- Exam 1 Practice Problems
  - problems + solutions released: Wed, Feb. 12
- Midterm Exam 1
  - Tue, Feb. 18, 7:00pm 9:00pm
- Today's In-Class Poll
  - http://pg.mlcourse.org

### MIDTERM EXAM LOGISTICS

### Midterm Exam

#### Time / Location

- Time: Evening ExamTue, Feb. 18, 7:00pm 9:00pm
- 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 8
- 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 exam practice problems (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
  - KNN Regression

# **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')

#### 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.

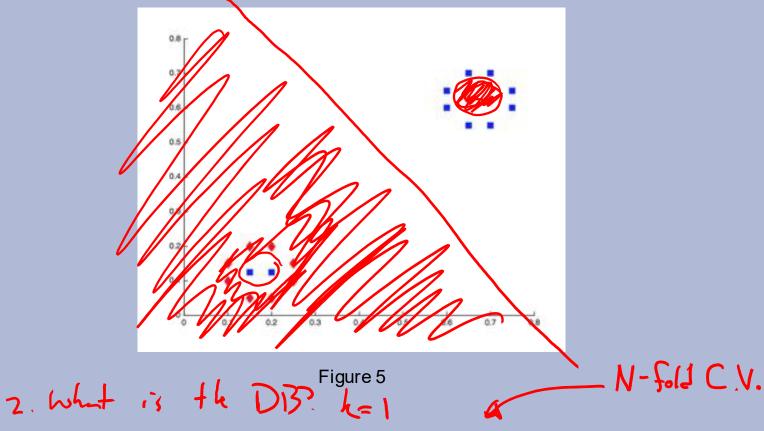
Snowstorm	Holiday	Weekend	Closed
T	T	F	F
T	T	F	$\mathbf{T}$
F	T	F	F
Т	T	F	F
F	F	F	F
F	F	F	$\mathbf{T}$
Т	F	F	$\mathbf{T}$
F X	F X a	F	Т

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<sub>2</sub> 0.75 = -0.4 and log<sub>2</sub> 0.25 = -2)

#### 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.



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?

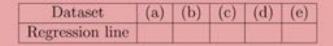
#### 4.1 True or False

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  $D^{(2)}$ .

#### 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.



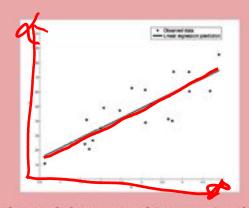


Figure 1: An observed data set and its associated regression line.

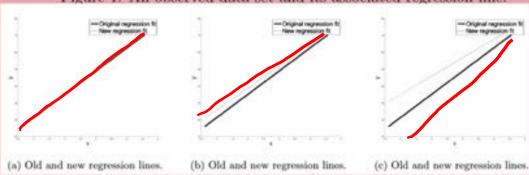
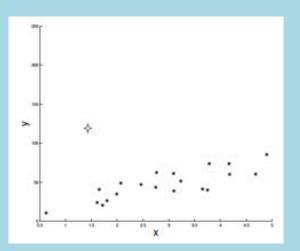


Figure 2: New regression lines for altered data sets  $S^{\text{new}}$ .



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

#### 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					

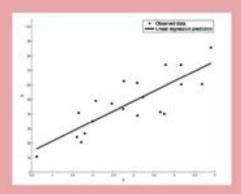


Figure 1: An observed data set and its associated regression line.

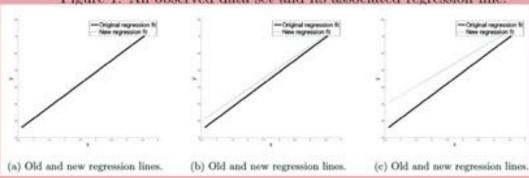
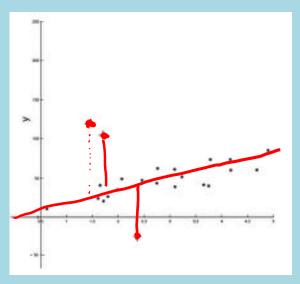


Figure 2: New regression lines for altered data sets  $S^{\text{new}}$ .



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

#### 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					

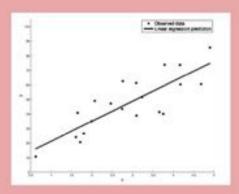


Figure 1: An observed data set and its associated regression line.

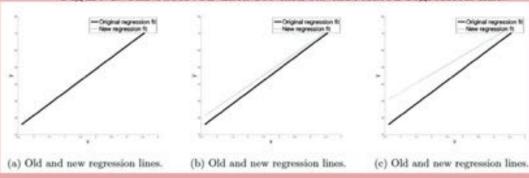
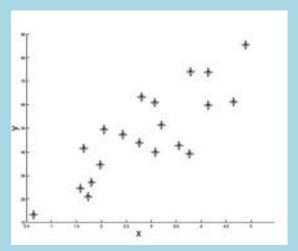


Figure 2: New regression lines for altered data sets  $S^{\text{new}}$ .



(d) Duplicating the original data set.

#### 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					

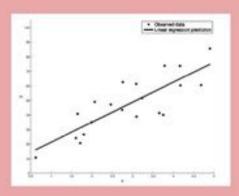


Figure 1: An observed data set and its associated regression line.

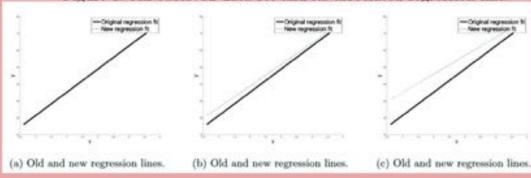
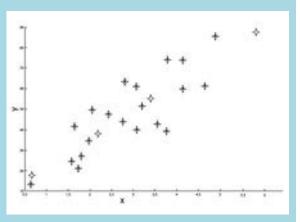


Figure 2: New regression lines for altered data sets  $S^{\text{new}}$ .



(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

$$h_{\theta}(\mathbf{x}) = p(y = 1|\mathbf{x})$$

#### 2. Least Mean Squares

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x}$$

#### 3. Perceptron

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = \operatorname{sign}(\boldsymbol{\theta}^T \mathbf{x})$$

4. 
$$\theta_k \leftarrow \theta_k + (h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})$$

5. 
$$\theta_k \leftarrow \theta_k + \frac{1}{1 + \exp \lambda(h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)})}$$

$$\theta_k \leftarrow \theta_k + \lambda (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)}) x_k^{(i)}$$

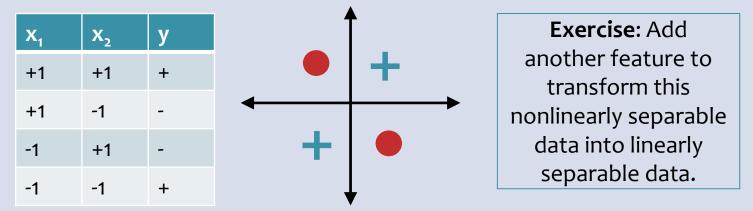
$$C. 1=6, 2=4, 3=4$$

# Q&A

### Q&A

• Why did we focus mostly on the Perceptron mistake bound for **linearly separable data**; isn't that an unrealistic setting?

A: Not at all! Even if your data isn't linearly separable to begin with, we can often add features to make it so.

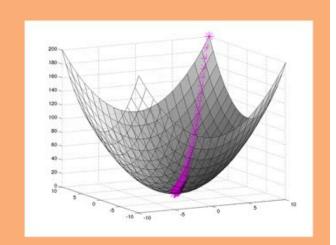


# OPTIMIZATION METHOD #3: STOCHASTIC GRADIENT DESCENT

### **Gradient Descent**

### Algorithm 1 Gradient Descent

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



# Stochastic Gradient Descent (SGD)

#### Algorithm 2 Stochastic Gradient Descent (SGD)

```
1: \operatorname{procedure} \operatorname{SGD}(\mathcal{D}, \theta^{(0)})
2: \theta \leftarrow \theta^{(0)}
3: \operatorname{while} not converged \operatorname{do}
4: i \sim \operatorname{Uniform}(\{1, 2, \dots, N\})
5: \theta \leftarrow \theta - \gamma \nabla_{\theta} J^{(i)}(\theta)
6: \operatorname{return} \theta
```

We need a per-example objective:

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

# Stochastic Gradient Descent (SGD)

#### Algorithm 2 Stochastic Gradient Descent (SGD)

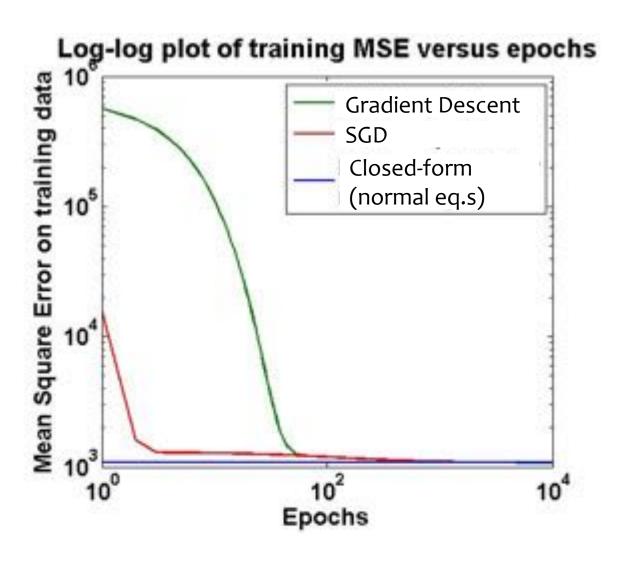
```
1: procedure SGD(\mathcal{D}, \theta^{(0)})
2: \theta \leftarrow \theta^{(0)}
3: while not converged do
4: for i \in \text{shuffle}(\{1, 2, \dots, N\}) do
5: \theta \leftarrow \theta - \gamma \nabla_{\theta} J^{(i)}(\theta)
6: return \theta
```

We need a per-example objective:

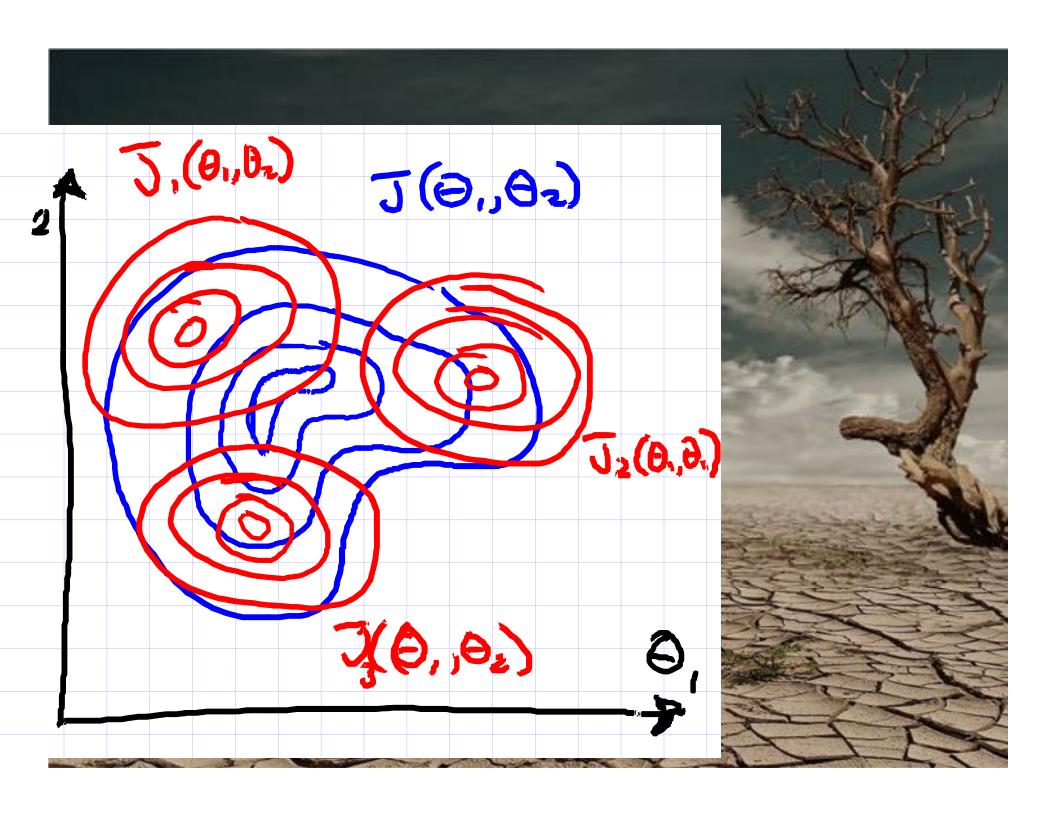
Let 
$$J(\boldsymbol{\theta}) = \sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$

In practice, it is common to implement SGD using sampling without replacement (i.e. shuffle({1,2,...N}), even though most of the theory is for sampling with replacement (i.e. Uniform({1,2,...N}).

# Convergence Curves



- Def: an epoch is a single pass through the training data
- For GD, only one update per epoch
- For SGD, N updates
   per epoch
   N = (# train examples)
- SGD reduces MSE much more rapidly than GD
- For GD / SGD, training MSE is initially large due to uninformed initialization



### **Expectations of Gradients**

$$\frac{dJ(\vec{\Theta})}{d\Theta_j} = \frac{1}{N} \sum_{i=1}^{N} \frac{d}{d\Theta_j} \left( J_i(\vec{\Theta}) \right)$$

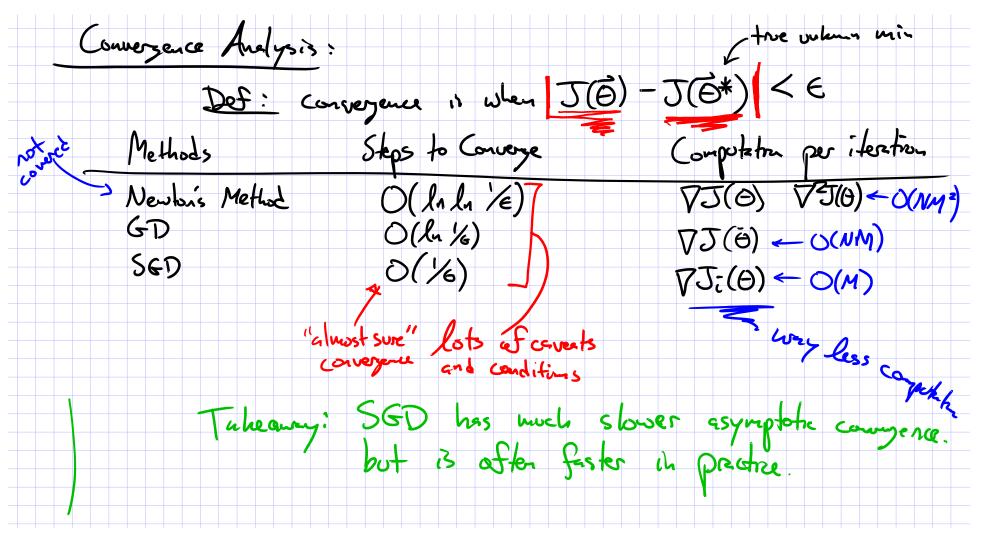
Recall: For any discrete r.v. 
$$X$$

$$E_{X}[+(X)] = \sum_{x} Y(x=x) + (x)$$

Q:What is the expected value of a randomly chosen 
$$\nabla J_i(\Theta)$$
?

Let  $I \sim \text{UniSorm}(\{1,...,N\})$ 
 $\Rightarrow P(I=i) = \frac{1}{N} \text{ if } ie\{1,...N}$ 
 $E_I[\nabla J_I(\Theta)] = \underbrace{\stackrel{N}{=}}_{i=1} P(I=i) \nabla J_i(\Theta)$ 
 $\Rightarrow \underbrace{\stackrel{N}{=}}_{i=1} \nabla J_i(\Theta)$ 
 $\Rightarrow \underbrace{\stackrel{N}{=}}_{i=1} \nabla J_i(\Theta)$ 
 $\Rightarrow \nabla J(\Theta)$ 

# Convergence of Optimizers



# SGD FOR LINEAR REGRESSION

# Linear Regression as Function $\frac{D = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}}{\text{where } \mathbf{x} \in \mathbb{R}^{M} \text{ and } y \in \mathbb{R}}$ Approximation

1. Assume  $\mathcal D$  generated as:

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

2. Choose hypothesis space,  $\mathcal{H}$ : all linear functions in M-dimensional space

$$\mathcal{H} = \{h_{\boldsymbol{\theta}} : h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x}, \boldsymbol{\theta} \in \mathbb{R}^M \}$$

3. Choose an objective function: mean squared error (MSE)

$$J(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} e_i^2$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left( y^{(i)} - h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) \right)^2$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left( y^{(i)} - \boldsymbol{\theta}^T \mathbf{x}^{(i)} \right)^2$$

- Solve the unconstrained optimization problem via favorite method:
  - gradient descent
  - · closed form
  - stochastic gradient descent
  - ...

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

5. Test time: given a new  $\mathbf{x}$ , make prediction  $\hat{y}$ 

$$\hat{y} = h_{\hat{\boldsymbol{\theta}}}(\mathbf{x}) = \hat{\boldsymbol{\theta}}^T \mathbf{x}$$

### Gradient Calculation for Linear Regression

#### Derivative of $J^{(i)}(\theta)$ :

$$\frac{d}{d\theta_k} J^{(i)}(\boldsymbol{\theta}) = \frac{d}{d\theta_k} \frac{1}{2} (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)})^2 
= \frac{1}{2} \frac{d}{d\theta_k} (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)})^2 
= (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) \frac{d}{d\theta_k} (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) 
= (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) \frac{d}{d\theta_k} \left( \sum_{j=1}^K \theta_j x_j^{(i)} - y^{(i)} \right) 
= (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_k^{(i)}$$

#### Derivative of $J(\theta)$ :

$$\frac{d}{d\theta_k} J(\boldsymbol{\theta}) = \sum_{i=1}^N \frac{d}{d\theta_k} J^{(i)}(\boldsymbol{\theta})$$
$$= \sum_{i=1}^N (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_k^{(i)}$$

Gradient of 
$$J^{(i)}(oldsymbol{ heta})$$
 [used by SGD]

$$\nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta}) = \begin{bmatrix} \frac{d}{d\theta_1} J^{(i)}(\boldsymbol{\theta}) \\ \frac{d}{d\theta_2} J^{(i)}(\boldsymbol{\theta}) \\ \vdots \\ \frac{d}{d\theta_M} J^{(i)}(\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_1^{(i)} \\ (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_2^{(i)} \\ \vdots \\ (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_N^{(i)} \end{bmatrix}$$
$$= (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) \mathbf{x}^{(i)}$$

$$\nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta}) = \begin{bmatrix} \begin{bmatrix} \frac{d}{d\theta_1} J^{(i)}(\boldsymbol{\theta}) \\ \frac{d}{d\theta_2} J^{(i)}(\boldsymbol{\theta}) \\ \vdots \\ \frac{d}{d\theta_M} J^{(i)}(\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_1^{(i)} \\ (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_2^{(i)} \\ \vdots \\ (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_N^{(i)} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^N (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_1^{(i)} \\ \sum_{i=1}^N (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_2^{(i)} \\ \vdots \\ \sum_{i=1}^N (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_N^{(i)} \end{bmatrix} = \sum_{i=1}^N (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) \mathbf{x}^{(i)}$$

$$= \sum_{i=1}^N (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) \mathbf{x}^{(i)}$$

# SGD for Linear Regression

SGD applied to Linear Regression is called the "Least Mean Squares" algorithm

```
Algorithm 1 Least Mean Squares (LMS)
                1: procedure LMS(\mathcal{D}, \boldsymbol{\theta}^{(0)})
                                                                         \boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}
                                                                                                                                                                                                                                                                                                                                                                                                                                                          ▷ Initialize parameters
                2:
                                                                        while not converged do
               3:
                                                                                                              for i \in \mathsf{shuffle}(\{1, 2, \dots, N\}) do
              4:
                                                                                                                                             \mathbf{g} \leftarrow (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) \mathbf{x}^{(i)}

    Compute gradient
    Compute gradient

               5:
                                                                                                                                              \theta \leftarrow \theta - \gamma \mathbf{g}

    □ Update parameters

              6:
                                                                          return \theta
              7:
```

# GD for Linear Regression

Gradient Descent for Linear Regression repeatedly takes steps opposite the gradient of the objective function

#### 

### 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(x) that best approximates c\*(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|x) that best approximates  $p^*(y|x)$ 

# 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?



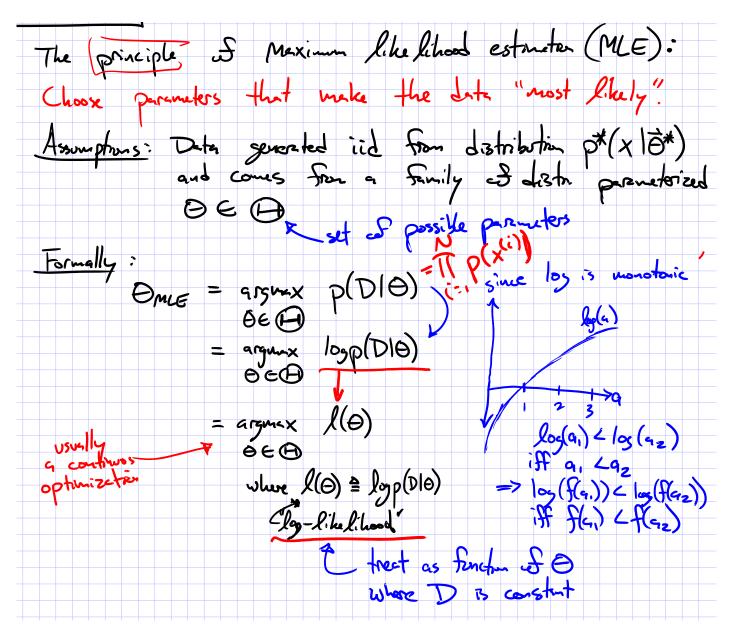


## Bayes Optimal Classifier

#### Whiteboard

- Bayes Optimal Classifier
- Reducible / irreducible error
- Ex: Bayes Optimal Classifier for o/1 Loss

### Maximum Likelihood Estimation



### MLE

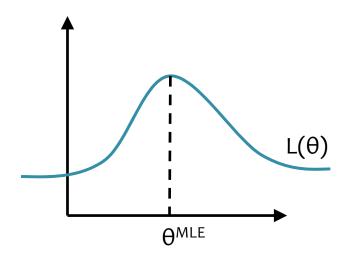
Suppose we have data  $\mathcal{D} = \{x^{(i)}\}_{i=1}^{N}$ 

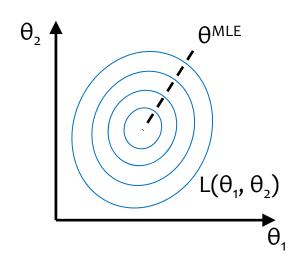
### Principle of Maximum Likelihood Estimation:

Choose the parameters that maximize the likelihood of the data. N

$$\boldsymbol{\theta}^{\mathsf{MLE}} = \operatorname*{argmax}_{\boldsymbol{\theta}} \prod_{i=1} p(\mathbf{x}^{(i)} | \boldsymbol{\theta})$$

Maximum Likelihood Estimate (MLE)





#### MLE

What does maximizing likelihood accomplish?

- There is only a finite amount of probability mass (i.e. sum-to-one constraint)
- MLE tries to allocate as much probability mass as possible to the things we have observed...

... at the expense of the things we have not observed

# 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)