Stochastic Gradient Descent
+
Probabilistic Learning
(Binary Logistic Regression)
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

• Midterm Exam 1
  – Tue, Feb. 18, 7:00pm – 9:00pm

• Today’s In-Class Poll
MIDTERM EXAM LOGISTICS
Midterm Exam

• **Time / Location**
  – **Time:** Evening Exam  
    *Tue, 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
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.

<table>
<thead>
<tr>
<th>Snowstorm</th>
<th>Holiday</th>
<th>Weekend</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

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.

2. What is the DB? $k = 1$

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 $D^{(1)} = \{(x_1^{(1)}, y_1^{(1)}), \ldots, (x_n^{(1)}, y_n^{(1)})\}$ and $D^{(2)} = \{(x_1^{(2)}, y_1^{(2)}), \ldots, (x_m^{(2)}, y_m^{(2)})\}$ 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.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression line</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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 $S_{\text{new}}$.

(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 $S^{new}$ 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.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression line</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

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

(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 $S_{new}$ 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.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression line</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

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

(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 $S^{\text{new}}$ 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.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression line</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Update Rule</th>
</tr>
</thead>
</table>
| 1. SGD for Logistic Regression  
\[ h_\theta(x) = p(y = 1|x) \] | 4.  \[ \theta_k' \leftarrow \theta_k + (h_\theta(x^{(i)}) - y^{(i)}) \] |
| 2. Least Mean Squares  
\[ h_\theta(x) = \theta^T x \] | 5.  \[ \theta_k' \leftarrow \theta_k + \frac{1}{1 + \exp \lambda(h_\theta(x^{(i)}) - y^{(i)})} \] |
| 3. Perceptron  
\[ h_\theta(x) = \text{sign}(\theta^T x) \] | 6.  \[ \theta_k' \leftarrow \theta_k + \lambda(h_\theta(x^{(i)}) - y^{(i)})x_k^{(i)} \] |
| 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
Q: Why did we focus mostly on the Perceptron mistake bound for \textit{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.

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>+1</td>
<td>+</td>
</tr>
<tr>
<td>+1</td>
<td>-1</td>
<td>-</td>
</tr>
<tr>
<td>-1</td>
<td>+1</td>
<td>-</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>+</td>
</tr>
</tbody>
</table>

\textbf{Exercise:} Add another feature to transform this nonlinearly separable data into linearly separable data.
OPTIMIZATION METHOD #3: STOCHASTIC GRADIENT DESCENT
Gradient Descent

Algorithm 1 Gradient Descent

1: procedure GD($\mathcal{D}, \theta^{(0)}$)
2: \hspace{1em} $\theta \leftarrow \theta^{(0)}$
3: \hspace{1em} while not converged do
4: \hspace{2em} $\theta \leftarrow \theta - \gamma \nabla_{\theta} J(\theta)$
5: \hspace{1em} return $\theta$
Stochastic Gradient Descent (SGD)

We need a per-example objective:

Let \( J(\theta) = \sum_{i=1}^{N} J^{(i)}(\theta) \)
We need a per-example objective:

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

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

1. For GD, only one update per epoch
2. For SGD, $N$ updates per epoch
   $N = (\# \text{ train examples})$

- SGD reduces MSE much more rapidly than GD
- For GD / SGD, training MSE is initially large due to uninformed initialization

Figure adapted from Eric P. Xing
Expected Gradients

\[ \frac{dJ(\theta)}{d\theta_j} = \frac{1}{N} \sum_{i=1}^{N} \frac{d}{d\theta_j} (J_i(\theta)) \]

Recall: for any discrete r.v. \( X \)

\[ \mathbb{E}_x [\mathbb{I}(x = x)] = \sum_{x} \mathbb{P}(x = x) \]

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

Let \( I \sim \text{Uniform}(\{1, \ldots, N\}) \),

\[ \mathbb{P}(I = i) = \frac{1}{N} \quad \text{if } i \in \{1, \ldots, N\} \]

\[ E_I [\nabla J_I(\theta)] = \sum_{i=1}^{N} \mathbb{P}(I = i) \nabla J_i(\theta) \]

\[ = \frac{1}{N} \sum_{i=1}^{N} \nabla J_i(\theta) \]

\[ = \nabla J(\theta) \]
Convergence of Optimizers

**Convergence Analysis:**

**Def:** Convergence is when \( \left| J(\theta) - J(\theta^*) \right| < \varepsilon \)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Steps to Converge</th>
<th>Computation per iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newton's Method</td>
<td>( O(ln lnn/\varepsilon) )</td>
<td>( \nabla J(\theta) \leftarrow O(NM^2) )</td>
</tr>
<tr>
<td>GD</td>
<td>( O(ln \varepsilon) )</td>
<td>( \nabla J(\theta) \leftarrow O(NM) )</td>
</tr>
<tr>
<td>SGD</td>
<td>( O(1/\varepsilon) )</td>
<td>( \nabla J_i(\theta) \leftarrow O(M) )</td>
</tr>
</tbody>
</table>

"almost sure" lots of conditions and assumptions

**Takeaway:** SGD has much slower asymptotic convergence, but is often faster in practice.
SGD FOR
LINEAR REGRESSION
Linear Regression as Function Approximation

1. Assume \( \mathcal{D} \) generated as:
   \[
   \begin{align*}
   x^{(i)} &\sim p^*(\cdot) \\
y^{(i)} &= h^*(x^{(i)})
   \end{align*}
   \]

2. Choose hypothesis space, \( \mathcal{H} \): all linear functions in \( M \)-dimensional space
   \[
   \mathcal{H} = \{ h_\theta : h_\theta(x) = \theta^T x, \theta \in \mathbb{R}^M \}
   \]

3. Choose an objective function: mean squared error (MSE)
   \[
   J(\theta) = \frac{1}{N} \sum_{i=1}^{N} e_i^2 = \frac{1}{N} \sum_{i=1}^{N} \left( y^{(i)} - h_\theta(x^{(i)}) \right)^2 = \frac{1}{N} \sum_{i=1}^{N} \left( y^{(i)} - \theta^T x^{(i)} \right)^2
   \]

4. Solve the unconstrained optimization problem via favorite method:
   - gradient descent
   - closed form
   - stochastic gradient descent
   - ...
   \[
   \hat{\theta} = \arg\min_{\theta} J(\theta)
   \]

5. Test time: given a new \( x \), make prediction \( \hat{y} \)
   \[
   \hat{y} = h_\theta(x) = \hat{\theta}^T x
   \]
Gradient Calculation for Linear Regression

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

Gradient of $J^{(i)}(\theta)$ [used by SGD]
\[
\nabla_{\theta} J^{(i)}(\theta) = \begin{bmatrix}
\frac{d}{d\theta_1} J^{(i)}(\theta) \\
\frac{d}{d\theta_2} J^{(i)}(\theta)
\end{bmatrix} = \begin{bmatrix}
(\theta^T x^{(i)} - y^{(i)}) x_1^{(i)} \\
(\theta^T x^{(i)} - y^{(i)}) x_2^{(i)} \\
\vdots \\
(\theta^T x^{(i)} - y^{(i)}) x_N^{(i)}
\end{bmatrix}
= (\theta^T x^{(i)} - y^{(i)}) x^{(i)}
\]

Derivative of $J(\theta)$:
\[
\frac{d}{d\theta} J(\theta) = \sum_{i=1}^{N} \frac{d}{d\theta_k} J^{(i)}(\theta) \\
= \sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x_k^{(i)}
\]

Gradient of $J(\theta)$ [used by Gradient Descent]
\[
\nabla_{\theta} J(\theta) = \begin{bmatrix}
\frac{d}{d\theta_1} J(\theta) \\
\frac{d}{d\theta_2} J(\theta)
\end{bmatrix} = \begin{bmatrix}
\sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x_1^{(i)} \\
\sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x_2^{(i)} \\
\vdots \\
\sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x_N^{(i)}
\end{bmatrix}
= \sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x^{(i)}
\]
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($\mathcal{D}$, $\theta^{(0)}$)
2:   $\theta \leftarrow \theta^{(0)}$ ▷ Initialize parameters
3:   while not converged do
4:     for $i \in \text{shuffle}([1, 2, \ldots, N])$ do
5:       $g \leftarrow (\theta^T x^{(i)} - y^{(i)}) x^{(i)}$ ▷ Compute gradient
6:       $\theta \leftarrow \theta - \gamma g$ ▷ Update parameters
7:   return $\theta$
GD for Linear Regression

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

---

**Algorithm 1** GD for Linear Regression

1. **procedure** \( \text{GDLR}(\mathcal{D}, \theta^{(0)}) \)
2. \[ \theta \leftarrow \theta^{(0)} \] ▶ Initialize parameters
3. **while** not converged **do**
4. \[ g \leftarrow \sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x^{(i)} \] ▶ Compute gradient
5. \[ \theta \leftarrow \theta - \gamma g \] ▶ Update parameters
6. **return** \( \theta \)
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:

\[ x^{(i)} \sim p^*(\cdot) \]
\[ y^{(i)} = c^*(x^{(i)}) \]

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:

\[ x^{(i)} \sim p^*(\cdot) \]
\[ y^{(i)} \sim p^*(\cdot | x^{(i)}) \]

Our goal is to learn a probability distribution \( p(y|x) \) that best approximates \( p^*(y|x) \)
Robotic Farming

<table>
<thead>
<tr>
<th></th>
<th>Deterministic</th>
<th>Probabilistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>Is this a picture of a wheat kernel?</td>
<td>Is this plant drought resistant?</td>
</tr>
<tr>
<td>(binary output)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regression</td>
<td>How many wheat kernels are in this picture?</td>
<td>What will the yield of this plant be?</td>
</tr>
<tr>
<td>(continuous output)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Bayes Optimal Classifier

Whiteboard

– Bayes Optimal Classifier
– Reducible / irreducible error
– Ex: Bayes Optimal Classifier for 0/1 Loss
Maximum Likelihood Estimation

The principle of Maximum Likelihood estimation (MLE):

Choose parameters that make the data “most likely.”

Assumptions: Data generated iid from distribution $p^*(x|\theta^*)$ and comes from a family of distributions parameterized $	heta \in \Theta$ (set of possible parameters)

Formally:

$\theta_{MLE} = \arg \max_{\theta \in \Theta} p(D|\theta)$

$= \arg \max_{\theta \in \Theta} \log p(D|\theta)$

$= \arg \max_{\theta \in \Theta} \ell(\theta)$

where $\ell(\theta) \equiv \log p(D|\theta)$

“Log-likelihood”

Usually a continuous optimization.

For $\theta$, treat as function $f(\theta)$

where $D$ is constant.
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.

\[
\theta^\text{MLE} = \arg\max_{\theta} \prod_{i=1}^{N} p(x^{(i)} | \theta)
\]

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)