Midterm Exam Review
+ Multinomial Logistic Reg.
  + Feature Engineering
  + Regularization
Reminders

• Homework 4: Logistic Regression
  – Out: Fri, Feb 15
  – Due: Fri, Mar 1 at 11:59pm

• Midterm Exam 1
  – Thu, Feb 21, 6:30pm – 8:00pm

• Today’s In-Class Poll
  – http://p10.mlcourse.org

• Reading on Probabilistic Learning is reused later in the course for MLE/MAP
Outline

• Midterm Exam Logistics
• Sample Questions
• Classification and Regression: The Big Picture
• Q&A
MIDTERM EXAM LOGISTICS
Midterm Exam

- **Time / Location**
  - **Time:** Evening Exam
    Thu, Feb. 21 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 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 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
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

• 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’)

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>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</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.

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?
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 $D^{(2)}$. 

(b) [2 pts.] Suppose $(x)$ is an arbitrary feature mapping from input $x \in X$ to $(x) \in \mathbb{R}^N$ and let $K(x, z) = (x)^\cdot (z)$. Then $K(x, z)$ will always be a kernel function.

(c) [2 pts.] Given the same training data, in which the points are linearly separable, the margin of the decision boundary produced by SVM will always be greater than or equal to the margin of the decision boundary produced by Perceptron.

4.2 Multiple Choice

(a) [3 pt.] If the data is linearly separable, SVM minimizes $\|w\|^2$ subject to the constraints $8_i y_i w \cdot x_i \geq 1$. In the linearly separable case, which of the following may happen to the decision boundary if one of the training samples is removed? Circle all that apply.

• Shifts toward the point removed
• Shifts away from the point removed
• Does not change

(b) [3 pt.] Recall that when the data are not linearly separable, SVM minimizes $\|w\|^2 + C \sum_i \varepsilon_i$ subject to the constraint that $8_i y_i w \cdot x_i \geq \varepsilon_i$ and $\varepsilon_i \geq 0$. Which of the following may happen to the size of the margin if the tradeoff parameter $C$ is increased? Circle all that apply.

• Increases
• Decreases
• Remains the same
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 $S^{\text{new}}$.

(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 $S_{\text{new}}$, 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 $S_{\text{new}}$.

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.

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

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.
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

1. SGD for Logistic Regression
   \[ h_\theta(x) = p(y|x) \]

2. Least Mean Squares
   \[ h_\theta(x) = \theta^T x \]

3. Perceptron
   \[ h_\theta(x) = \text{sign}(\theta^T x) \]

4. \[ \theta_k \leftarrow \theta_k + (h_\theta(x^{(i)}) - y^{(i)}) \]

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

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
MULTINOMIAL LOGISTIC REGRESSION
Multinomial Logistic Regression

Chalkboard

– Background: Multinomial distribution
– Definition: Multi-class classification
– Geometric intuitions
– Multinomial logistic regression model
– Generative story
– Reduction to binary logistic regression
– Partial derivatives and gradients
– Applying Gradient Descent and SGD
– Implementation w/ sparse features
In-Class Exercise: Think-Pair-Share

Debug the following program which is (incorrectly) attempting to run SGD for multinomial logistic regression

Buggy Program:

```python
while not converged:
    for i in shuffle([1,...,N]):
        for k in [1,...,K]:
            theta[k] = theta[k] - lambda * 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 \(\theta[k]\). \(\lambda\) is the learning rate. \(N = \# \text{ of examples.} \ K = \# \text{ of output classes.} \ M = \# \text{ of features.} \ \theta\) is a \(K \times M\) matrix.
FEATURE ENGINEERING
Handcrafted Features

\[ p(y|x) \propto \exp(\Theta_y \cdot f) \]
Where do features come from?

- First word before M1
- Second word before M1
- Bag-of-words in M1
- Head word of M1
- Other word in between
- First word after M2
- Second word after M2
- Bag-of-words in M2
- Head word of M2
- Bigrams in between
- Words on dependency path
- Country name list
- Personal relative triggers
- Personal title list
- WordNet Tags
- Heads of chunks in between
- Path of phrase labels
- Combination of entity types

Hand-crafted features:
- Sun et al., 2011
- Zhou et al., 2005

Feature Engineering vs. Feature Learning
Where do features come from?

- **Hand-crafted features**
  - Sun et al., 2011
  - Zhou et al., 2005

- **Feature Engineering**
  - Mikolov et al., 2013

- **Feature Learning**
  - CBOW model in Mikolov et al. (2013)

- **Look-up table**
  - Input (context words)
  - Embedding
  - Missing word

- **Classifier**
  - Unsupervised learning

- **Similar words, similar embeddings**

- **Word embeddings**
  - Mikolov et al., 2013

- **Cat:**
  - 0.11
  - 0.23
  - ...
  - -0.45

- **Dog:**
  - 0.13
  - 0.26
  - ...
  - -0.52
Where do features come from?

Feature Engineering

- Hand-crafted features
  - Sun et al., 2011
  - Zhou et al., 2005

Feature Learning

- Word embeddings
  - Mikolov et al., 2013
- String embeddings
  - Socher, 2011
  - Collobert & Weston, 2008

Convolutional Neural Networks
(Collobert and Weston 2008)

Recursive Auto Encoder
(Socher 2011)

The [movie] showed [wars]
Where do features come from?

The diagram illustrates the flow of features from hand-crafted features to feature learning. It shows how features are extracted from a sentence and mapped to different types of embeddings. The sentence "The [movie] showed [wars]" is used as an example. The diagram includes references to various studies:

- Sun et al., 2011
- Zhou et al., 2005
- Mikolov et al., 2013
- Socher et al., 2013
- Hermann & Blunsom, 2013
- Collobert & Weston, 2008
Where do features come from?

- Hand-crafted features:
  - Sun et al., 2011
  - Zhou et al., 2005

- Word embedding features:
  - Turian et al., 2010
  - Koo et al., 2008
  - Mikolov et al., 2013

- Tree embeddings:
  - Socher et al., 2013
  - Hermann & Blunsom, 2013

- String embeddings:
  - Socher, 2011
  - Collobert & Weston, 2008

Refine embedding features with semantic/syntactic info.
Where do features come from?

- **Hand-crafted features**
  - Sun et al., 2011
  - Zhou et al., 2005

- **Word embeddings**
  - Mikolov et al., 2013

- **Tree embeddings**
  - Socher et al., 2013
  - Hermann & Blunsom, 2013

- **String embeddings**
  - Socher, 2011
  - Collobert & Weston, 2008

- **Best of both worlds?**
  - Turian et al., 2010
  - Koo et al., 2008
  - Hermann et al., 2014

**Feature Engineering**

**Feature Learning**
Feature Engineering for NLP

Suppose you build a logistic regression model to predict a part-of-speech (POS) tag for each word in a sentence.

What features should you use?

The movie I watched depicted hope
Feature Engineering for NLP

Per-word Features:

<table>
<thead>
<tr>
<th>Feature</th>
<th>$x^{(1)}$</th>
<th>$x^{(2)}$</th>
<th>$x^{(3)}$</th>
<th>$x^{(4)}$</th>
<th>$x^{(5)}$</th>
<th>$x^{(6)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>is-capital($w_i$)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>endswith($w_i$,”e”)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>endswith($w_i$,”d”)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>endswith($w_i$,”ed”)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$w_i$ == “aardvark”</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$w_i$ == “hope”</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

The movie I watched depicted hope
Feature Engineering for NLP

Context Features:

<table>
<thead>
<tr>
<th></th>
<th>x(1)</th>
<th>x(2)</th>
<th>x(3)</th>
<th>x(4)</th>
<th>x(5)</th>
<th>x(6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>wi == “watched”</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>wi+1 == “watched”</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>wi-1 == “watched”</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>wi+2 == “watched”</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>wi-2 == “watched”</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

The movie I watched depicted hope
Feature Engineering for NLP

Context Features:

<table>
<thead>
<tr>
<th>...</th>
<th>$w_i$</th>
<th>$w_{i+1}$</th>
<th>$w_{i-1}$</th>
<th>$w_{i+2}$</th>
<th>$w_{i-2}$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^{(1)}$</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x^{(2)}$</td>
<td>...</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x^{(3)}$</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$x^{(4)}$</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$x^{(5)}$</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x^{(6)}$</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The movie I watched depicted hope
Feature Engineering for NLP

**Table 3.** Tagging accuracies with different feature templates and other changes on the *WSJ* 19-21 development set.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3GramMemm</td>
<td>See text</td>
<td>248,798</td>
<td>52.07%</td>
<td>96.92%</td>
<td>88.99%</td>
</tr>
<tr>
<td>NAACL 2003</td>
<td>See text and [1]</td>
<td>460,552</td>
<td>55.31%</td>
<td>97.15%</td>
<td>88.61%</td>
</tr>
<tr>
<td>Replication</td>
<td>See text and [1]</td>
<td>460,551</td>
<td>55.62%</td>
<td>97.18%</td>
<td>88.92%</td>
</tr>
<tr>
<td>Replication'</td>
<td>+rareFeatureThresh = 5</td>
<td>482,364</td>
<td>55.67%</td>
<td>97.19%</td>
<td>88.96%</td>
</tr>
<tr>
<td>5w</td>
<td>+⟨t₀, w₋₁⟩, ⟨t₀, w₂⟩</td>
<td>730,178</td>
<td>56.23%</td>
<td>97.20%</td>
<td>89.03%</td>
</tr>
<tr>
<td>5wShapes</td>
<td>+⟨t₀, s₋₁⟩, ⟨t₀, s₀⟩, ⟨t₀, s₊₁⟩</td>
<td>731,661</td>
<td>56.52%</td>
<td>97.25%</td>
<td>89.81%</td>
</tr>
<tr>
<td>5wShapesDS</td>
<td>+ distributional similarity</td>
<td>737,955</td>
<td>56.79%</td>
<td>97.28%</td>
<td>90.46%</td>
</tr>
</tbody>
</table>
Feature Engineering for CV

Edge detection (Canny)

Corner Detection (Harris)

Figures from http://opencv.org
Feature Engineering for CV

Scale Invariant Feature Transform (SIFT)

Figure from Lowe (1999) and Lowe (2004)
NON-LINEAR FEATURES
Nonlinear Features

- aka. “nonlinear basis functions”
- So far, input was always $\mathbf{x} = [x_1, \ldots, x_M]$
- **Key Idea**: let input be some function of $\mathbf{x}$
  - original input: $\mathbf{x} \in \mathbb{R}^M$ where $M' > M$ (usually)
  - new input: $\mathbf{x}' \in \mathbb{R}^{M'}$
  - define $\mathbf{x}' = b(\mathbf{x}) = [b_1(\mathbf{x}), b_2(\mathbf{x}), \ldots, b_{M'}(\mathbf{x})]$
    where $b_i : \mathbb{R}^M \to \mathbb{R}$ is any function
- **Examples**: ($M = 1$)
  - polynomial
    
  - radial basis function
    
  - sigmoid
    
  - log

For a linear model: still a linear function of $b(\mathbf{x})$ even though a nonlinear function of $\mathbf{x}$

**Examples:**
- Perceptron
- Linear regression
- Logistic regression
Example: Linear Regression

Goal: Learn $y = \mathbf{w}^T f(x) + b$
where $f(.)$ is a polynomial basis function

true “unknown” target function is $y = \tanh(x) + \text{noise}$
Example: Linear Regression

**Goal:** Learn $y = w^T f(x) + b$ where $f(.)$ is a polynomial basis function.

The true “unknown” target function is $y = \tanh(x) + \text{noise}$.
**Example: Linear Regression**

**Goal:** Learn $y = w^T f(x) + b$ where $f(.)$ is a polynomial basis function.

true “unknown” target function is $y = \tanh(x) + \text{noise}$
**Example: Linear Regression**

**Goal:** Learn \( y = \mathbf{w}^T f(\mathbf{x}) + b \) where \( f(.) \) is a polynomial basis function.

true “unknown” target function is \( y = \tanh(x) + \text{noise} \)
Example: Linear Regression

**Goal:** Learn \( y = w^T f(x) + b \) where \( f(.) \) is a polynomial basis function

true “unknown” target function is \( y = \tanh(x) + \text{noise} \)
Example: Linear Regression

**Goal:** Learn \( y = \mathbf{w}^T \mathbf{f}(\mathbf{x}) + b \)

where \( \mathbf{f}(\cdot) \) is a polynomial basis function

true “unknown” target function is \( y = \tanh(x) + \text{noise} \)
Example: Linear Regression

**Goal:** Learn \( y = \mathbf{w}^T f(x) + b \) where \( f(.) \) is a polynomial basis function

true “unknown” target function is \( y = \tanh(x) + \text{noise} \)
Example: Linear Regression

**Goal:** Learn \( y = w^T f(x) + b \)
where \( f(.) \) is a polynomial basis function

true "unknown" target function is \( y = \tanh(x) + \text{noise} \)
Example: Linear Regression

**Goal:** Learn $y = \mathbf{w}^T f(x) + b$ where $f(.)$ is a polynomial basis function.

The "unknown" target function is $y = \tanh(x) + \text{noise}$.
**Goal:** Learn $y = w^T f(x) + b$ where $f(.)$ is a polynomial basis function

The true "unknown" target function is $y = \tanh(x) + \text{noise}$
Example: Linear Regression

**Goal:** Learn \( y = \mathbf{w}^T f(x) + b \) where \( f(.) \) is a polynomial basis function.

true “unknown” target function is linear with negative slope and gaussian noise
**Goal:** Learn $y = \mathbf{w}^T f(x) + b$ where $f(.)$ is a polynomial basis function.

The true “unknown” target function is linear with negative slope and Gaussian noise.
Example: Linear Regression

**Goal:** Learn \( y = \mathbf{w}^T f(x) + b \)
where \( f(.) \) is a polynomial basis function

true “unknown” target function is linear with negative slope and gaussian noise
Example: Linear Regression

**Goal:** Learn \( y = \mathbf{w}^T f(x) + b \) where \( f(.) \) is a polynomial basis function.

The true “unknown” target function is linear with negative slope and Gaussian noise.

![Linear Regression Plot](image-url)
**Example: Linear Regression**

**Goal:** Learn $y = \mathbf{w}^T f(x) + b$ where $f(.)$ is a polynomial basis function.

true “unknown” target function is linear with negative slope and gaussian noise.
**Example: Linear Regression**

**Goal:** Learn $y = w^T f(x) + b$ where $f(.)$ is a polynomial basis function.

The true "unknown" target function is linear with negative slope and Gaussian noise.
Example: Linear Regression

**Goal:** Learn \( y = \mathbf{w}^\top f(x) + b \) where \( f(.) \) is a polynomial basis function.

The true “unknown” target function is linear with negative slope and Gaussian noise.
Over-fitting

Root-Mean-Square (RMS) Error: \[ E_{\text{RMS}} = \sqrt{\frac{2E(w^*)}{N}} \]

Slide courtesy of William Cohen
Polynomial Coefficients

<table>
<thead>
<tr>
<th></th>
<th>$M = 0$</th>
<th>$M = 1$</th>
<th>$M = 3$</th>
<th>$M = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\theta_0$</td>
<td>0.19</td>
<td>0.82</td>
<td>0.31</td>
<td>0.35</td>
</tr>
<tr>
<td>$\theta_1$</td>
<td>-1.27</td>
<td>7.99</td>
<td>232.37</td>
<td></td>
</tr>
<tr>
<td>$\theta_2$</td>
<td>-25.43</td>
<td>-5321.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\theta_3$</td>
<td>17.37</td>
<td>48568.31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\theta_4$</td>
<td></td>
<td>-231639.30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\theta_5$</td>
<td></td>
<td>640042.26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\theta_6$</td>
<td></td>
<td></td>
<td>-1061800.52</td>
<td></td>
</tr>
<tr>
<td>$\theta_7$</td>
<td></td>
<td></td>
<td>1042400.18</td>
<td></td>
</tr>
<tr>
<td>$\theta_8$</td>
<td></td>
<td></td>
<td>-557682.99</td>
<td></td>
</tr>
<tr>
<td>$\theta_9$</td>
<td></td>
<td></td>
<td>125201.43</td>
<td></td>
</tr>
</tbody>
</table>

Slide courtesy of William Cohen
Example: Linear Regression

**Goal:** Learn $y = w^T f(x) + b$ where $f(.)$ is a polynomial basis function.

The "unknown" target function is linear with negative slope and Gaussian noise.
Example: Linear Regression

**Goal:** Learn \( y = \mathbf{w}^T f(\mathbf{x}) + b \) where \( f(.) \) is a polynomial basis function.

Same as before, but now with \( N = 100 \) points.

true “unknown” target function is linear with negative slope and gaussian noise.

Linear Regression (poly=9)
REGULARIZATION
Overfitting

Definition: The problem of overfitting is when the model captures the noise in the training data instead of the underlying structure.

Overfitting can occur in all the models we’ve seen so far:

– Decision Trees (e.g. when tree is too deep)
– KNN (e.g. when k is small)
– Perceptron (e.g. when sample isn’t representative)
– Linear Regression (e.g. with nonlinear features)
– Logistic Regression (e.g. with many rare features)
Motivation: Regularization

Example: Stock Prices

• Suppose we wish to predict Google’s stock price at time t+1

• **What features should we use?** (putting all computational concerns aside)
  – Stock prices of all other stocks at times t, t-1, t-2, …, t - k
  – Mentions of Google with positive / negative sentiment words in all newspapers and social media outlets

• Do we believe that **all** of these features are going to be useful?
Motivation: Regularization

• **Occam’s Razor:** prefer the simplest hypothesis

• What does it mean for a hypothesis (or model) to be **simple**?
  1. small number of features (**model selection**)
  2. small number of “important” features (**shrinkage**)

83
Regularization

Chalkboard

– L2, L1, L0 Regularization
– Example: Linear Regression
Don’t Regularize the Bias (Intercept) Parameter!

• In our models so far, the bias / intercept parameter is usually denoted by $\theta_0$ -- that is, the parameter for which we fixed $x_0 = 1$

• Regularizers always avoid penalizing this bias / intercept parameter

• Why? Because otherwise the learning algorithms wouldn’t be invariant to a shift in the y-values

Whitening Data

• It’s common to *whiten* each feature by subtracting its mean and dividing by its variance

• For regularization, this helps all the features be penalized in the same units (e.g. convert both centimeters and kilometers to z-scores)
Regularization:

\[ \ln \lambda = +1.18 \]
### Polynomial Coefficients

<table>
<thead>
<tr>
<th>( w_0^* )</th>
<th>none</th>
<th>exp(18)</th>
<th>huge</th>
</tr>
</thead>
<tbody>
<tr>
<td>( w_1^* )</td>
<td>0.35</td>
<td>0.35</td>
<td>0.13</td>
</tr>
<tr>
<td>( w_2^* )</td>
<td>232.37</td>
<td>4.74</td>
<td>-0.05</td>
</tr>
<tr>
<td>( w_3^* )</td>
<td>-5321.83</td>
<td>-0.77</td>
<td>-0.06</td>
</tr>
<tr>
<td>( w_4^* )</td>
<td>48568.31</td>
<td>-31.97</td>
<td>-0.05</td>
</tr>
<tr>
<td>( w_5^* )</td>
<td>-231639.30</td>
<td>-3.89</td>
<td>-0.03</td>
</tr>
<tr>
<td>( w_6^* )</td>
<td>640042.26</td>
<td>55.28</td>
<td>-0.02</td>
</tr>
<tr>
<td>( w_7^* )</td>
<td>-1061800.52</td>
<td>41.32</td>
<td>-0.01</td>
</tr>
<tr>
<td>( w_8^* )</td>
<td>1042400.18</td>
<td>-45.95</td>
<td>-0.00</td>
</tr>
<tr>
<td>( w_9^* )</td>
<td>-557682.99</td>
<td>-91.53</td>
<td>0.00</td>
</tr>
<tr>
<td>( w_0^* )</td>
<td>125201.43</td>
<td>72.68</td>
<td>0.01</td>
</tr>
</tbody>
</table>
Over Regularization:
Regularization Exercise

In-class Exercise
1. Plot train error vs. # features (cartoon)
2. Plot test error vs. # features (cartoon)
Example: Logistic Regression

Training Data
Example: Logistic Regression
Example: Logistic Regression
Example: Logistic Regression

Classification with Logistic Regression ($\lambda = 1e-05$)
Example: Logistic Regression
Example: Logistic Regression

Classification with Logistic Regression ($\lambda = 0.001$)
Example: Logistic Regression

Classification with Logistic Regression (\(lambda=0.01\))
Example: Logistic Regression

Classification with Logistic Regression (lambda=0.1)
Example: Logistic Regression
Example: Logistic Regression

Classification with Logistic Regression (\text{lambda}=10)
Example: Logistic Regression

Classification with Logistic Regression (lambda=100)
Example: Logistic Regression
Example: Logistic Regression

Classification with Logistic Regression (\(\lambda = 10000\))
Example: Logistic Regression
Example: Logistic Regression
Example: Logistic Regression

Classification with Logistic Regression ($\lambda = 1e+07$)
Example: Logistic Regression
Regularization as MAP

• L1 and L2 regularization can be interpreted as maximum a-posteriori (MAP) estimation of the parameters

• To be discussed later in the course...
Takeaways

1. **Nonlinear basis functions** allow **linear models** (e.g. Linear Regression, Logistic Regression) to capture **nonlinear** aspects of the original input.

2. Nonlinear features are **require no changes to the model** (i.e. just preprocessing).

3. **Regularization** helps to avoid **overfitting**.

4. **Regularization** and **MAP estimation** are equivalent for appropriately chosen priors.
Feature Engineering / Regularization

Objectives

You should be able to...

• Engineer appropriate features for a new task
• Use feature selection techniques to identify and remove irrelevant features
• Identify when a model is overfitting
• Add a regularizer to an existing objective in order to combat overfitting
• Explain why we should not regularize the bias term
• Convert linearly inseparable dataset to a linearly separable dataset in higher dimensions
• Describe feature engineering in common application areas