Lecture 3:
SVM, Logistic Regression, Neural Nets, Evaluation Metrics
Announcements

• HW1 Due Tomorrow
  – Will be graded in about a week

• HW2 Released Tonight/Tomorrow
  – Due Jan 23rd at 9pm

• Recitation Thursday
  – Linear Algebra (& Vector Calculus)
  – Annenberg 105
Recap: Basic Recipe

• Training Data: \( S = \{(x_i, y_i)\}_{i=1}^N \)  
  \[ x \in \mathbb{R}^D \]
  \[ y \in \{-1,+1\} \]

• Model Class: \( f(x \mid w, b) = w^T x - b \)  
  Linear Models

• Loss Function: \( L(a,b) = (a - b)^2 \)  
  Squared Loss

• Learning Objective: \( \argmin_{w,b} \sum_{i=1}^N L(y_i, f(x_i \mid w, b)) \)  
  Optimization Problem
Recap: Bias-Variance Trade-off
Recap: Complete Pipeline

\[ S = \{(x_i, y_i)\}_{i=1}^{N} \]

Training Data

\[ f(x \mid w, b) = w^T x - b \]

Model Class(es)

\[ L(a, b) = (a - b)^2 \]

Loss Function

\[ \arg\min_{w, b} \sum_{i=1}^{N} L(y_i, f(x_i \mid w, b)) \quad \text{SGD!} \]

Cross Validation & Model Selection

Profit!
Today

• Beyond Basic Linear Models
  – Support Vector Machines
  – Logistic Regression
  – Feed-forward Neural Networks
  – Different ways to interpret models

• Different Evaluation Metrics
Today

• Beyond Basic Linear Models
  – Support Vector Machines
  – Logistic Regression
  – Feed-forward Neural Networks
  – Different ways to interpret models

• Different Evaluation Metrics
How to compute gradient for 0/1 Loss?

\[ \partial_w \sum_{i=1}^{N} L(y_i, f(x_i \mid w, b)) \]

\[ \argmin_{w,b} \sum_{i=1}^{N} L(y_i, f(x_i \mid w, b)) \]
Recap: 0/1 Loss is Intractable

• 0/1 Loss is flat or discontinuous everywhere

• VERY difficult to optimize using gradient descent

• **Solution:** Optimize surrogate Loss
  – Today: Hinge Loss (...eventually)
Support Vector Machines
aka Max-Margin Classifiers
Which Line is the Best Classifier?

Which Line is the Best Classifier?

Recall: Hyperplane Distance

• Line is a 1D, Plane is 2D
• Hyperplane is many D
  – Includes Line and Plane
• Defined by \((w, b)\)

• Distance:
  \[
  \frac{|w^T x - b|}{\|w\|}
  \]

• Signed Distance:
  \[
  \frac{w^T x - b}{\|w\|}
  \]

Linear Model = un-normalized signed distance!
Recall: Margin

\[ \gamma = \max_w \min_{(x,y)} \frac{y(w^T x)}{\|w\|} \]
How to Maximize Margin?
(Assume Linearly Separable)

Choose \( w \) that maximizes:

\[
\arg\max_{w,b} \min_{(x,y)} \frac{y(w^T x - b)}{\|w\|}
\]
How to Maximize Margin?
(Assume Linearly Separable)

\[
\arg\max_{w, b} \left[ \min_{(x, y)} \frac{y(w^T x - b)}{\|w\|} \right]
\]

\[\equiv \arg\max_{w, b: \|w\|=1} \left[ \min_{(x, y)} y(w^T x - b) \right]\]

Hold Denominator Fixed

Suppose we instead enforce:

\[
\min_{(x, y)} y(w^T x - b) = 1
\]

Hold Numerator Fixed

Then:

\[= \arg\min_{w, b} \|w\| \equiv \arg\min_{w, b} \|w\|^2\]
Max Margin Classifier (Support Vector Machine)

"Margin"

\[
\arg\min_{w,b} \frac{1}{2} w^T w \equiv \frac{1}{2} \|w\|^2
\]

\[
\forall i : y_i (w^T x_i - b) \geq 1
\]

Better generalization to unseen test examples
(beyond scope of course*)
(only training data on margin matter)

"Linearly Separable"


Soft-Margin Support Vector Machine

\[
\text{argmin}_{w,b,\xi} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i \\
\forall i : y_i (w^T x_i - b) \geq 1 - \xi_i \\
\forall i : \xi_i \geq 0
\]

Size of Margin vs Size of Margin Violations (C controls trade-off)

Hinge Loss

\[
\text{argmin}_{w,b,\xi} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i \\
\forall i : y_i \left( w^T x_i - b \right) \geq 1 - \xi_i \\
\forall i : \xi_i \geq 0
\]

\[
L(y_i, f(x_i)) = \max(0, 1 - y_i f(x_i)) = \xi_i
\]
Hinge Loss vs Squared Loss

Squared Loss

Hinge Loss

0/1 Loss

Target y
Recall: Perceptron Learning Algorithm
(Linear Classification Model)

- \( w^1 = 0, \ b^1 = 0 \)

- For \( t = 1 \ldots \)
  - Receive example \((x, y)\)
  - If \( f(x \mid w^t, b^t) = y \)
    - \([w^{t+1}, b^{t+1}] = [w^t, b^t]\)
  - Else
    - \(w^{t+1} = w^t + yx\)
    - \(b^{t+1} = b^t - y\)

\[ f(x \mid w) = \text{sign}(w^T x - b) \]

Training Set:
\[ S = \{(x_i, y_i)\}_{i=1}^N \]
\( y \in \{+1,-1\} \)

Go through training set in arbitrary order (e.g., randomly)
Comparison with Perceptron “Loss”

Perceptron
\[ \max \{0, -y_i f(x_i | w, b)\} \]

SVM/Hinge
\[ \max \{0, 1 - y_i f(x_i | w, b)\} \]
Support Vector Machine

• 2 Interpretations

• Geometric
  – Margin vs Margin Violations

• Loss Minimization
  – Model complexity vs Hinge Loss
  – (Will discuss in depth next lecture)

• Equivalent!

\[
\arg\min_{w, b, \xi} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i \\
\forall i : y_i (w^T x_i - b) \geq 1 - \xi_i \\
\forall i : \xi_i \geq 0
\]
Comment on Optimization

• Hinge Loss is not smooth
  – Not differentiable

• How to optimize?

• Stochastic (Sub-)Gradient Descent still works!
  – Sub-gradients discussed next lecture

\[
\begin{align*}
\arg\min_{w,b,\xi} & \quad \frac{1}{2} w^T w + \frac{C}{N} \sum_{i} \xi_i \\
\forall i & \quad y_i (w^T x_i - b) \geq 1 - \xi_i \\
\forall i & \quad \xi_i \geq 0
\end{align*}
\]

Logistic Regression
aka “Log-Linear” Models
Logistic Regression

\[
P(y \mid x, w, b) = \frac{1}{e^{2y(w^T x - b)} + e^{-2y(w^T x - b)}}
\]

Also known as sigmoid function: \( \sigma(a) = \frac{e^a}{1 + e^a} \)

\( y \in \{-1, +1\} \)
Maximum Likelihood Training

- Training set: 
  \[ S = \{(x_i, y_i)\}_{i=1}^{N} \] 
  \( x \in \mathbb{R}^D \) 
  \( y \in \{-1,+1\} \)

- Maximum Likelihood: 
  \[ \text{argmax}_{w,b} \prod_{i} P(y_i | x_i, w, b) \] 
  – (Why?)

- Each \((x, y)\) in \(S\) sampled independently! 
  – Discussed further in Probably Recitation
Why Use Logistic Regression?

- SVMs often better at classification
  - Assuming margin exists...

- Calibrated Probabilities?

- Increase in SVM score....
  - ...similar increase in \( P(y=+1|x) \)?
  - Not well calibrated!

- Logistic Regression!

*Figure above is for Boosted Decision Trees (SVMs have similar effect)*

Log Loss

\[
P(y \mid x, w, b) = \frac{\frac{1}{2}y(w^T x - b)}{e^{\frac{1}{2}y(w^T x - b)}} = \frac{\frac{1}{2}yf(x \mid w, b)}{e^{\frac{1}{2}yf(x \mid w, b)}} + e^{\frac{1}{2}yf(x \mid w, b)}
\]

\[
\arg\max_{w, b} \prod_i P(y_i \mid x_i, w, b) = \arg\min_{w, b} \sum_i -\ln P(y_i \mid x_i, w, b)
\]

\[
L(y, f(x)) = -\ln \left( \frac{\frac{1}{2}yf(x)}{e^{\frac{1}{2}yf(x)}} \right) + e^{\frac{1}{2}yf(x)}
\]

Solve using (Stoch.) Gradient Descent
Log Loss vs Hinge Loss

$L(y, f(x)) = \max(0, 1 - yf(x))$

$L(y, f(x)) = -\ln \left( \frac{e^{\frac{1}{2}yf(x)}}{e^{\frac{1}{2}yf(x)} - e^{-\frac{1}{2}yf(x)}} + e^{\frac{1}{2}yf(x)} \right)$
Log-Loss Gradient
(For One Example)

\[ \frac{\partial_w}{\partial w} - \ln P(y_i | x_i) = -\frac{1}{2} y_i f(x_i | w, b) - \ln \left( e^{\frac{1}{2}y_i f(x_i | w, b)} + e^{-\frac{1}{2}y_i f(x_i | w, b)} \right) \]

\[ = -\frac{1}{2} y_i x_i + \frac{1}{e^{\frac{1}{2}y_i f(x_i | w, b)} + e^{-\frac{1}{2}y_i f(x_i | w, b)}} \frac{1}{2} y_i f(x_i | w, b) \frac{1}{e^{\frac{1}{2}y_i f(x_i | w, b)} + e^{-\frac{1}{2}y_i f(x_i | w, b)}} \frac{1}{2} y_i f(x_i | w, b) \]

\[ = \left( -1 + \frac{1}{e^{\frac{1}{2}y_i f(x_i | w, b)} + e^{-\frac{1}{2}y_i f(x_i | w, b)}} \frac{1}{2} y_i f(x_i | w, b) - e^{-\frac{1}{2}y_i f(x_i | w, b)} \right) \frac{1}{2} y_i x_i \]

\[ = \left( -1 + P(y_i | x_i) - P(-y_i | x_i) \right) \frac{1}{2} y_i x_i \]

\[ = -P(-y_i | x_i) y_i x_i = -(1 - P(y_i | x_i)) y_i x_i \]

\[ P(y | x, w, b) = \frac{e^{\frac{1}{2}y f(x | w, b)}}{e^{\frac{1}{2}y f(x | w, b)} + e^{-\frac{1}{2}y f(x | w, b)}} \]
Logistic Regression

- Two Interpretations
- Maximizing Likelihood
- Minimizing Log Loss
- Equivalent!
Feed-Forward Neural Networks
aka Not Quite Deep Learning
1 Layer Neural Network

• 1 Neuron
  – Takes input $x$
  – Outputs $y$

$$f(x|w,b) = w^T x - b$$
$$= w_1 x_1 + w_2 x_2 + w_3 x_3 - b$$

• ~Logistic Regression!
  – Solve via Gradient Descent

$y = \sigma(f(x))$
2 Layer Neural Network

• 2 Layers of Neurons
  – 1\textsuperscript{st} Layer takes input \(x\)
  – 2\textsuperscript{nd} Layer takes output of 1\textsuperscript{st} layer

• Can approximate arbitrary functions
  – Provided hidden layer is large enough
  – “fat” 2-Layer Network
Aside: Deep Neural Networks

• Why prefer Deep over a “Fat” 2-Layer?
  – Compact model
    • (exponentially large “fat” model)
  – Easier to train?
  – Discussed further in deep learning lectures

Training Neural Networks

- **Gradient Descent!**
  - Even for Deep Networks*

- **Parameters:**
  - \((w_{11}, b_{11}, w_{12}, b_{12}, w_2, b_2)\)

\[
\partial_{w_2} \sum_{i=1}^{N} L(y_i, \sigma_2) = \sum_{i=1}^{N} \partial_{w_2} L(y_i, \sigma_2) = \sum_{i=1}^{N} \partial_\sigma L(y_i, \sigma_2) \partial_{w_2} \sigma_2 = \sum_{i=1}^{N} \partial_\sigma L(y_i, \sigma_2) \partial_f \sigma_2 \partial_{w_2} f_2
\]

\[
\partial_{w_{1m}} \sum_{i=1}^{N} L(y_i, \sigma_2) = \sum_{i=1}^{N} \partial_\sigma L(y_i, \sigma_2) \partial_f \sigma_2 \partial_{w_{1m}} f_2 = \sum_{i=1}^{N} \partial_\sigma L(y_i, \sigma_2) \partial_f \sigma_2 \partial_\sigma \partial_{w_{1m}} f_1
\]

*more complicated

Backpropagation = Gradient Descent
(lots of chain rules)
Story So Far

• Different Loss Functions
  – Hinge Loss
  – Log Loss
  – Can be derived from different interpretations

• Non-Linear Model Classes
  – Neural Nets
  – Composable with different loss functions

• No closed-form solution for training
  – Must use some form of gradient descent
Today

• Beyond Basic Linear Models
  – Support Vector Machines
  – Logistic Regression
  – Feed-forward Neural Networks
  – Different ways to interpret models

• Different Evaluation Metrics
Evaluation

• 0/1 Loss (Classification)

• Squared Loss (Regression)

• Anything Else?
Example: Cancer Prediction

<table>
<thead>
<tr>
<th>Model</th>
<th>Loss Function</th>
<th>Has Cancer</th>
<th>Doesn’t Have Cancer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicts Cancer</td>
<td>Low</td>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>Predicts No Cancer</td>
<td>OMG Panic!</td>
<td>Low</td>
<td></td>
</tr>
</tbody>
</table>

- Value Positives & Negatives Differently
  - Care much more about positives

- “Cost Matrix”
  - 0/1 Loss is Special Case
Optimizing for Cost-Sensitive Loss

- There is no universally accepted way.

Simplest Approach (Cost Balancing):

\[
\arg\min_{w,b} \left( 1000 \sum_{i:y_i=1} L(y_i, f(x_i | w, b)) + \sum_{i:y_i=-1} L(y_i, f(x_i | w, b)) \right)
\]

<table>
<thead>
<tr>
<th>Loss Function</th>
<th>Has Cancer</th>
<th>Doesn’t Have Cancer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicts Cancer</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Predicts No Cancer</td>
<td>1000</td>
<td>0</td>
</tr>
</tbody>
</table>
**Precision & Recall**

- **Precision** = \( \frac{TP}{TP + FP} \)
- **Recall** = \( \frac{TP}{TP + FN} \)

\[ F1 = \frac{2}{\frac{1}{P} + \frac{1}{R}} \]

**Counts**

<table>
<thead>
<tr>
<th>Model</th>
<th>Has Cancer</th>
<th>Doesn’t Have Cancer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicts Cancer</td>
<td>20 (TP)</td>
<td>30 (FP)</td>
</tr>
<tr>
<td>Predicts No Cancer</td>
<td>5 (FN)</td>
<td>70 (TN)</td>
</tr>
</tbody>
</table>

- **TP** = True Positive, **TN** = True Negative
- **FP** = False Positive, **FN** = False Negative

**Care More About Positives!**

Example: Search Query

- Rank webpages by relevance
Ranking Measures

- Predict a Ranking (of webpages)
  - Users only look at top 4
  - Sort by \( f(x | w, b) \)

- Precision @4 = 1/2
  - Fraction of top 4 relevant

- Recall @4 = 2/3
  - Fraction of relevant in top 4

- Top of Ranking Only!

Pairwise Preferences

2 Pairwise Disagreements
4 Pairwise Agreements
ROC-Area

• ROC-Area
  – Area under ROC Curve
  – Fraction pairwise agreements

• Example: 

  ROC-Area: 0.5  
  #Pairwise Preferences = 6  
  #Agreements = 3

Average Precision

- **Average Precision**
  - Area under P-R Curve
  - P@K for each positive

- **Example:**

  \[
  \text{AP: } \frac{1}{3} \cdot \left( \frac{1}{1} + \frac{2}{3} + \frac{3}{5} \right) \approx 0.76
  \]

ROC-Area versus Average Precision

- ROC-Area Cares about every pairwise preference equally
- Average Precision cares more about top of ranking

<table>
<thead>
<tr>
<th></th>
<th>ROC-Area: 0.5</th>
<th>Average Precision: 0.76</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Green Red Green</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Red Green Red Green</td>
<td>ROC-Area: 0.5</td>
<td>Average Precision: 0.64</td>
</tr>
</tbody>
</table>
Other Challenges

- “Correct” if overlap is large enough
- How to define large enough?
- Duplicate detections?
- What is learning objective?

• Similar challenges in videos:
  • E.g., temporal bounding box around “running” activity
  • Duplicate predictions: break into two separate running activities

• Other examples: heart-rate monitoring

http://slazebni.cs.illinois.edu/publications/iccv15_active.pdf
Summary: Evaluation Measures

- Different Evaluations Measures
  - Different Scenarios

- Large focus on getting positives
  - Large cost of mis-predicting cancer
  - Relevant webpages are rare
    - Aka “Class Imbalance”

- Other challenges:
  - localization in continuous domain
Next Lecture

- Regularization
- Lasso

- Thursday:
  - Recitation on Matrix Linear Algebra (& Calculus)