Perceptron
Q&A
Reminders

- **Homework 2: Decision Trees**
  - Out: Wed, Jan. 22
  - Due: Wed, Feb. 05 at 11:59pm

- **Homework 3: KNN, Perceptron, Lin.Reg.**
  - Out: Wed, Feb. 05 (+ 1 day)
  - Due: Wed, Feb. 12 at 11:59pm

- **Today’s In-Class Poll**
  - [http://p6.mlcourse.org](http://p6.mlcourse.org)
MODEL SELECTION
Model Selection

WARNING:
• In some sense, our discussion of model selection is premature.
• The models we have considered thus far are fairly simple.
• The models and the many decisions available to the data scientist wielding them will grow to be much more complex than what we’ve seen so far.
Model Selection

Statistics
- **Def**: a model defines the data generation process (i.e. a set or family of parametric probability distributions)
- **Def**: model parameters are the values that give rise to a particular probability distribution in the model family
- **Def**: learning (aka. estimation) is the process of finding the parameters that best fit the data
- **Def**: hyperparameters are the parameters of a prior distribution over parameters

Machine Learning
- **Def**: (loosely) a model defines the hypothesis space over which learning performs its search
- **Def**: model parameters are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis
- **Def**: the learning algorithm defines the data-driven search over the hypothesis space (i.e. search for good parameters)
- **Def**: hyperparameters are the tunable aspects of the model, that the learning algorithm does not select
Model Selection

Example: Decision Tree
- model = set of all possible trees, possibly restricted by some hyperparameters (e.g. max depth)
- parameters = structure of a specific decision tree
- learning algorithm = ID3, CART, etc.
- hyperparameters = max-depth, threshold for splitting criterion, etc.

Machine Learning
- Def: (loosely) a model defines the hypothesis space over which learning performs its search
- Def: model parameters are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis
- Def: the learning algorithm defines the data-driven search over the hypothesis space (i.e. search for good parameters)
- Def: hyperparameters are the tunable aspects of the model, that the learning algorithm does not select
Model Selection

Example: k-Nearest Neighbors

- model = set of all possible nearest neighbors classifiers
- parameters = none (KNN is an instance-based or non-parametric method)
- learning algorithm = for naïve setting, just storing the data
- hyperparameters = $k$, the number of neighbors to consider

Machine Learning

- Def: (loosely) a model defines the hypothesis space over which learning performs its search
- Def: model parameters are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis
- Def: the learning algorithm defines the data-driven search over the hypothesis space (i.e. search for good parameters)
- Def: hyperparameters are the tunable aspects of the model, that the learning algorithm does not select
Model Selection

Example: Perceptron

• **model** = set of all linear separators

• **parameters** = vector of weights (one for each feature)

• **learning algorithm** = mistake based updates to the parameters

• **hyperparameters** = none (unless using some variant such as averaged perceptron)

Machine Learning

• **Def:** (loosely) a **model** defines the hypothesis space over which learning performs its search

• **Def:** **model parameters** are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis

• **Def:** the **learning algorithm** defines the data-driven search over the hypothesis space (i.e. search for good parameters)

• **Def:** **hyperparameters** are the tunable aspects of the model, that the learning algorithm does not select
Model Selection

Statistics

- **Def**: a **model** defines the data generation process (i.e. a set or family of parametric probability distributions)

- **Def**: **model parameters** are the values that give rise to a particular probability distribution in the model family

- **Def**: **learning** (aka. estimation) is the process of finding the parameters that best fit the data

- **Def**: **hyperparameters** are the parameters of a prior distribution over parameters

Machine Learning

- **Def**: (loosely) a **model** defines the hypothesis space over which learning performs its search

- **Def**: **model parameters** are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis

- **Def**: the learning algorithm defines the data-driven search over the hypothesis space (i.e. search for good parameters)

- **Def**: **hyperparameters** are the tunable aspects of the model, that the learning algorithm does not select

If “learning” is all about picking the best **parameters** how do we pick the best **hyperparameters**?
Model Selection

• Two very similar definitions:
  – *Def*: **model selection** is the process by which we choose the “best” model from among a set of candidates
  – *Def*: **hyperparameter optimization** is the process by which we choose the “best” hyperparameters from among a set of candidates (**could be called a special case of model selection**)

• **Both** assume access to a function capable of measuring the quality of a model

• **Both** are typically done “outside” the main training algorithm --- typically training is treated as a black box
## Experimental Design

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Training</strong></td>
<td></td>
<td>We pick the best model parameters by learning on the training dataset for a fixed set of hyperparameters</td>
</tr>
<tr>
<td>• training dataset</td>
<td>• best model parameters</td>
<td></td>
</tr>
<tr>
<td>• hyperparameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Hyperparameter Optimization</strong></td>
<td></td>
<td>We pick the best hyperparameters by learning on the training data and evaluating error on the validation error</td>
</tr>
<tr>
<td>• training dataset</td>
<td>• best hyperparameters</td>
<td></td>
</tr>
<tr>
<td>• validation dataset</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Testing</strong></td>
<td></td>
<td>We evaluate a hypothesis corresponding to a decision rule with fixed model parameters on a test dataset to obtain test error</td>
</tr>
<tr>
<td>• test dataset</td>
<td>• test error</td>
<td></td>
</tr>
<tr>
<td>• hypothesis (i.e. fixed model parameters)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Special Cases of k-NN

k=1: Nearest Neighbor

k=N: Majority Vote
Example of Hyperparameter Opt.

Chalkboard:

– Special cases of k-Nearest Neighbors
– Choosing k with validation data
– Choosing k with cross-validation
Cross-Validation

**Cross validation** is a method of estimating loss on held out data

**Input:** training data, learning algorithm, loss function (e.g. 0/1 error)

**Output:** an estimate of loss function on held-out data

**Key idea:** rather than just a single “validation” set, use many!
(ErrorCode is more stable. Slower computation.)

**Algorithm:**

Divide data into folds (e.g. 4)

1. Train on folds \{1,2,3\} and predict on \{4\}
2. Train on folds \{1,2,4\} and predict on \{3\}
3. Train on folds \{1,3,4\} and predict on \{2\}
4. Train on folds \{2,3,4\} and predict on \{1\}

Concatenate all the predictions and evaluate loss (*almost* equivalent to averaging loss over the folds)

**Definition:**

N-fold cross validation = cross validation with N folds
# Experimental Design

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Training</strong></td>
<td>• training dataset</td>
<td>We pick the best model parameters by learning on the training dataset for a fixed set of hyperparameters</td>
</tr>
<tr>
<td>• hyperparameters</td>
<td>• best model parameters</td>
<td></td>
</tr>
<tr>
<td><strong>Hyperparameter Optimization</strong></td>
<td>• training dataset</td>
<td>We pick the best hyperparameters by learning on the training data and evaluating error on the validation error</td>
</tr>
<tr>
<td>• validation dataset</td>
<td>• best hyperparameters</td>
<td></td>
</tr>
<tr>
<td><strong>Cross-Validation</strong></td>
<td>• training dataset</td>
<td>We estimate the error on held out data by repeatedly training on N-1 folds and predicting on the held-out fold</td>
</tr>
<tr>
<td>• validation dataset</td>
<td>• cross-validation error</td>
<td></td>
</tr>
<tr>
<td><strong>Testing</strong></td>
<td>• test dataset</td>
<td>We evaluate a hypothesis corresponding to a decision rule with fixed model parameters on a test dataset to obtain test error</td>
</tr>
<tr>
<td>• hypothesis (i.e. fixed model parameters)</td>
<td>• test error</td>
<td></td>
</tr>
</tbody>
</table>
Q: We pick the best hyperparameters by learning on the training data and evaluating error on the validation error. For our final model, should we then learn from training + validation?

A: Yes.

Let's assume that \{train-original\} is the original training data, and \{test\} is the provided test dataset.

1. Split \{train-original\} into \{train-subset\} and \{validation\}.
2. Pick the hyperparameters that when training on \{train-subset\} give the lowest error on \{validation\}. Call these hyperparameters \{best-hyper\}.
3. Retrain a new model using \{best-hyper\} on \{train-original\} = \{train-subset\} U \{validation\}.
4. Report test error by evaluating on \{test\}.

Alternatively, you could replace Steps 1-2 with the following:

1. Pick the hyperparameters that give the lowest cross-validation error on \{train-original\}. Call these hyperparameters \{best-hyper\}.
**k-NN: Choosing k**

Fisher Iris Data: varying the value of k
k-NN: Choosing k

Gaussian Data: varying the value of k
WARNING (again):
– This section is only scratching the surface!
– Lots of methods for hyperparameter optimization: (to talk about later)
  • Grid search
  • Random search
  • Bayesian optimization
  • Graduate-student descent
  • ...

Main Takeaway:
– Model selection / hyperparameter optimization is just another form of learning
Model Selection Learning Objectives

You should be able to...

• Plan an experiment that uses training, validation, and test datasets to predict the performance of a classifier on unseen data (without cheating)

• Explain the difference between (1) training error, (2) validation error, (3) cross-validation error, (4) test error, and (5) true error

• For a given learning technique, identify the model, learning algorithm, parameters, and hyperparameters

• Define "instance-based learning" or "nonparametric methods"

• Select an appropriate algorithm for optimizing (aka. learning) hyperparameters
THE PERCEPTRON ALGORITHM
Perceptron: History

Imagine you are trying to build a new machine learning technique... your name is Frank Rosenblatt... and the year is 1957
Perceptron: History

Imagine you are trying to build a new machine learning technique... your name is Frank Rosenblatt... and the year is 1957

*The New Yorker, December 6, 1958 P. 44*

Talk story about the perceptron, a new electronic brain which hasn't been built, but which has been successfully simulated on the I.B.M. 704. Talk with Dr. Frank Rosenblatt, of the Cornell Aeronautical Laboratory, who is one of the two men who developed the prodigy; the other man is Dr. Marshall C. Yovits, of the Office of Naval Research, in Washington. Dr. Rosenblatt defined the perceptron as the first non-biological object which will achieve an organization of its external environment in a meaningful way. It interacts with its environment, forming concepts that have not been made ready for it by a human agent. If a triangle is held up, the perceptron's eye picks up the image & conveys it along a random succession of lines to the response units, where the image is registered. It can tell the difference betw. a cat and a dog, although it wouldn't be able to tell whether the dog was to the left or right of the cat. Right now it is of no practical use, Dr. Rosenblatt conceded, but he said that one day it might be useful to send one into outer space to take in impressions for us.
Linear Models for Classification

Key idea: Try to learn this hyperplane directly

-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

Directly modeling the hyperplane would use a decision function:

\[ h(x) = \text{sign}(\theta^T x) \]

for:

\[ y \in \{-1, +1\} \]
Geometry

In-Class Exercise

Draw a picture of the region corresponding to:

\[ w_1x_1 + w_2x_2 + b > 0 \]

where \( w_1 = 2, w_2 = 3, b = 6 \)

Draw the vector

\[ w = [w_1, w_2] \]
Visualizing Dot-Products

Chalkboard:

- vector in 2D
- line in 2D
- adding a bias term
- definition of orthogonality
- vector projection
- hyperplane definition
- half-space definitions
Vector Projection

Question:
Which of the following is the projection of a vector $a$ onto a vector $b$?

A. $\frac{a^T b}{b} \cdot a$

B. $\frac{a \cdot b}{a^T b}$

C. $\frac{(a^T b)}{||b||_2} \cdot b$

D. $\frac{(a \cdot b)}{||b||_2} \cdot b$

E. $\frac{(a^T b)}{||b||_2^2} \cdot b$

F. $\frac{(a^T b)^2}{||b||_2} \cdot b$
Linear Models for Classification

Key idea: Try to learn this hyperplane directly

Directly modeling the hyperplane would use a decision function:

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

for:

$$y \in \{-1, +1\}$$

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
Online vs. Batch Learning

**Batch Learning**
Learn from all the examples at once

**Online Learning**
Gradually learn as each example is received
Online Learning

Examples

1. **Stock market** prediction (what will the value of Alphabet Inc. be tomorrow?)

2. **Email** classification (distribution of both spam and regular mail changes over time, but the target function stays fixed - last year's spam still looks like spam)

3. **Recommendation** systems. Examples: recommending movies; predicting whether a user will be interested in a new news article

4. **Ad placement** in a new market
Online Learning

For $i = 1, 2, 3, \ldots$:

• **Receive** an unlabeled instance $x^{(i)}$
• **Predict** $y' = h_\theta(x^{(i)})$
• **Receive** true label $y^{(i)}$
• **Suffer loss** if a mistake was made, $y' \neq y^{(i)}$
• **Update** parameters $\theta$

Goal:

• **Minimize** the number of mistakes
Perceptron

Chalkboard:

– (Online) Perceptron Algorithm
– Why do we need a bias term?
– Inductive Bias of Perceptron
– Limitations of Linear Models
Perceptron Algorithm: (without the bias term)

- Set $t=1$, start with all-zeroes weight vector $w_1$.
- Given example $x$, predict positive iff $w_t \cdot x \geq 0$.
- On a mistake, update as follows:
  - Mistake on positive, update $w_{t+1} \leftarrow w_t + x$
  - Mistake on negative, update $w_{t+1} \leftarrow w_t - x$

Slide adapted from Nina Balcan