Fall 2008

Time: T-Th 5:00pm - 6:20pm
Location: GFS118

Instructor: Sofus A. Macskassy (macskass@usc.edu)
Office: SAL 216
Office hours: by appointment

Teaching assistant: Cheol Han
Office hours: TBA

Class web page:
http://www-scf.usc.edu/~csci567/index.html
Administrative

- Email me to get on the mailing list
  - To: macskass@usc.edu
  - Subject: “csci567 student”
- Please stop me if I talk too fast
- Please stop me if you have any questions
- Any feedback is better than no feedback
  - Sooner is better
  - End of the semester is too late to make it easier on you
Administrative

• Class project info
  – Get into teams of size 2-4 as soon as you can
    • Would prefer size 2-3. See me if you want to do a project by yourself
  – I’ll suggest topics in a week or two
    • Topics can be *anything* as long as it is related to ML
  – Pre-proposal due on Sep 23
    • Abstract, lay out topic and scope
  – Proposal due on Oct 9
    • 1-2 pages write up on what you will do
  – Final paper due on Dec 2
  – Presentations on Dec 2 & 4
Administrative

- Questions from last class?
Lecture 2 Outline

• Supervised learning defined
• Hypothesis spaces
• Linear Threshold Algorithms Introduction
Review: Focus for this Course

• Based on *information available*
  – **Supervised** – true labels provided
  – **Reinforcement** – Only indirect labels provided (reward/punishment)
  – **Unsupervised** – No feedback & no labels

• Based on the *role of the learner*
  – **Passive** – given a set of data, produce a model
  – **Online** – given one data point at a time, update model
  – **Active** – ask for specific data points to improve model

• Based on *type of output*
  – **Concept Learning** – Binary output based on +ve/-ve examples
  – **Classification** – Classifying into one among many classes
  – **Regression** – Numeric, ordered output
Supervised Learning
(and appropriate applications)

- Given: training examples $<x, f(x)>$ for some unknown function $f$
- Find: A good approximation to $f$

Appropriate Applications

- There is no human expert
  - E.g., DNA analysis
    - $x$: bond graph of a new molecule
    - $f(x)$: predicted binding strength to AIDS protease molecule
- Humans can perform the task but cannot explain how
  - E.g., character recognition
    - $x$: bitmap picture of hand-written character
    - $f(x)$: ascii code of the character
- Desired function changes frequently
  - E.g., predicting stock prices based on recent trading data
    - $x$: description of stock prices and trades for last 10 days
    - $f(x)$: recommended stock transactions
- Each user needs a customized function $f$
  - E.g., email filtering
    - $x$: incoming email message in some (un)structured format
    - Spam $f(x)$: importance score for presenting to the user (or deleting without presenting)
    - Filter $f(x)$: predicting which email folder to put the email into
Example: A data set for supervised learning

Wisconsin Breast Tumor data set from UC-Irvine Machine Learning repository.

- Thirty real-valued variables per tumor.
- Two variables that have been predicted:
  - Outcome (R=recurrence, N=non-recurrence)
  - Time (until recurrence, for R, time healthy, for N)

<table>
<thead>
<tr>
<th>tumor size</th>
<th>texture</th>
<th>perimeter</th>
<th>outcome</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.02</td>
<td>27.60</td>
<td>117.5</td>
<td>N</td>
<td>31</td>
</tr>
<tr>
<td>17.99</td>
<td>10.38</td>
<td>122.8</td>
<td>N</td>
<td>61</td>
</tr>
<tr>
<td>20.29</td>
<td>14.34</td>
<td>135.1</td>
<td>R</td>
<td>27</td>
</tr>
</tbody>
</table>

...
## Terminology

- Columns are called *input variables* or *features* or *attributes*
- The outcome and time (which we are trying to predict) are called *output variables* or *targets* or *labels*
- A row in the table is called a *training example* (if used to build the model) or *instance*
- The whole table is called *(training, validation, test or evaluation)* *data set*

<table>
<thead>
<tr>
<th>tumor size</th>
<th>texture</th>
<th>perimeter</th>
<th>outcome</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.02</td>
<td>27.60</td>
<td>117.5</td>
<td>N</td>
<td>31</td>
</tr>
<tr>
<td>17.99</td>
<td>10.38</td>
<td>122.8</td>
<td>N</td>
<td>61</td>
</tr>
<tr>
<td>20.29</td>
<td>14.34</td>
<td>135.1</td>
<td>R</td>
<td>27</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Prediction problems

<table>
<thead>
<tr>
<th>tumor size</th>
<th>texture</th>
<th>perimeter</th>
<th>...</th>
<th>outcome</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.02</td>
<td>27.60</td>
<td>117.5</td>
<td></td>
<td>N</td>
<td>31</td>
</tr>
<tr>
<td>17.99</td>
<td>10.38</td>
<td>122.8</td>
<td></td>
<td>N</td>
<td>61</td>
</tr>
<tr>
<td>20.29</td>
<td>14.34</td>
<td>135.1</td>
<td></td>
<td>R</td>
<td>27</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- The problem of predicting the recurrence is called *(binary) classification*
- The problem of predicting the time is called *regression*
More formally

- Instances has the form: \( <x_1, \ldots, x_n, y> \) where \( n \) is the number of attributes
  - We will use the notation \( x_i \) to denote the attributes of the \( i \)-th instances: \( <x_{i,1}, \ldots, x_{i,n}> \).

- Assumptions:
  - There is underlying unknown probability distribution \( P(x,y) \), from which instances are drawn.
  - This distribution is fixed.

- Data for training as well as for future evaluation are drawn independently at random from \( P \).
  - Instances are said to be \( i.i.d. \) (independent and identically distributed)

- Let \( \mathcal{X} \) denote the space of input values
- Let \( \mathcal{Y} \) denote the space of output values
  - If \( \mathcal{Y} = \mathbb{R} \), this problem is called \textit{regression}
  - If \( \mathcal{Y} \) is a finite discrete set, the problem is called \textit{classification}
  - If \( \mathcal{Y} \) has 2 elements, the problem is called \textit{binary classification} or \textit{concept learning}
Supervised Learning Problem

- Training examples are drawn from independently at random according to unknown probability distribution \( P(x, y) \)
- The learning algorithm analyzes the examples and produces a classifier \( f \) or hypothesis \( h \)
- Given a new data point \( <x, y> \) drawn from \( P \) (independently and at random), the classifier is given \( x \) and predicts \( \hat{y} = f(x) \)
- The loss \( L(\hat{y}, y) \) is then measured.
- Goal of the learning algorithm: Find the \( f \) that minimizes the expected loss.
Example Problem: Family Car

- Class C of a “family car”
  - Classify: Is car $x$ a family car?
- Output:
  - Positive (+) and negative (−) examples
- Input representation:
  - $x_1$: price, $x_2$: engine power
Training set $X$

$X = \{x^t, y^t\}_{t=1}^{N}$

$y = \begin{cases} 
1 & \text{if } x \text{ is positive} \\
0 & \text{if } x \text{ is negative} 
\end{cases}$

$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
Class $C$

$\left( p_1 \leq \text{price} \leq p_2 \right)$ AND $\left( e_1 \leq \text{engine power} \leq e_2 \right)$
Hypothesis class $H$

$h(x) = \begin{cases} 
1 & \text{if } h \text{ classifies } x \text{ as positive} \\
0 & \text{if } h \text{ classifies } x \text{ as negative}
\end{cases}$

Error of $h$ on $H$

$E(h | X) = \sum_{t=1}^{N} 1(h(x^t) \neq y^t)$
S, G, and the Version Space

most specific hypothesis, $S$

most general hypothesis, $G$

$h \in H$, between $S$ and $G$ is consistent

and make up the version space

(Mitchell, 1997)
Example Problem: Tumor Recurrence

- \( P(x,y) \): distribution of patients \( x \) and their true labels \( y \) ("recurrence" or "non recurrence")
- training sample: a set of patients that have been labeled by the user
- learning algorithm: that’s what this class is about!
- \( f \): the classifier output by the learning algorithm
- test point: A new patient \( x \) (with its true, but hidden, label \( y \))
- loss function \( L(\hat{y},y) \):

<table>
<thead>
<tr>
<th>predicted label ( \hat{y} )</th>
<th>true label ( y )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>recurrence</td>
</tr>
<tr>
<td>recurrence</td>
<td>0</td>
</tr>
<tr>
<td>non recurrence</td>
<td>100</td>
</tr>
</tbody>
</table>
What is $f(x)$?

- There are three main approaches:
  - Learn a classifier: $f: \mathcal{X} \to \mathcal{Y}$
  - Learn a conditional probability distribution: $P(y | x)$
  - Learn a joint probability distribution: $P(x, y)$

- We will start by studying one example of each approach
  - Classifier: Perceptron
  - Conditional Distribution: Logistic Regression
  - Joint Distribution: Linear Discriminant Analysis (LDA)
Inferring $f(x)$ from $P(y \mid x)$

- Predict the $\hat{y}$ that minimizes the expected loss:

$$f(x) = \arg\min_{\hat{y}} E_{y|X}[L(\hat{y}, y)]$$

$$= \arg\min_{\hat{y}} \sum_{y} P(y|x) L(\hat{y}, y)$$
Example: Making the tumor decision

- Suppose our tumor recurrence detector predicts:
  \[ P(y = \text{"recurrence"} \mid x) = 0.1 \]
  What is the optimal classification decision \( \hat{y} \)?
- Expected loss of \( \hat{y} = \text{"recurrence"} \) is:
  \[ 0 \times 0.1 + 1 \times 0.9 = 0.9 \]
- Expected loss of \( \hat{y} = \text{"non recurrence"} \) is:
  \[ 100 \times 0.1 + 0 \times 0.9 = 10 \]
- Therefore, the optimal prediction is “recurrence”

<table>
<thead>
<tr>
<th>Loss function ( L(\hat{y}, y) ):</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>predicted label</strong> ( \hat{y} )</td>
</tr>
<tr>
<td>-----------------------------------</td>
</tr>
<tr>
<td>recurrence</td>
</tr>
<tr>
<td>recurrence</td>
</tr>
<tr>
<td>non recurrence</td>
</tr>
<tr>
<td>( P(y \mid x) )</td>
</tr>
</tbody>
</table>
Inferring $f(x)$ from $P(x,y)$

- We can compute the conditional distribution according to the definition of conditional probability:

$$P(y = k|x) = \frac{P(x, y = k)}{\sum_j P(x, y = j)}.$$ 

- In words, compute $P(x, y=k)$ for each value of $k$. Then normalize these numbers.

- Compute $\hat{y}$ using the method from the previous slide.
Fundamental Problem of Machine Learning: It is ill-posed

![Diagram of a function taking four inputs (x1, x2, x3, x4) and producing an output y = f(x1, x2, x3, x4).]

<table>
<thead>
<tr>
<th>Example</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Learning Appears Impossible

- There are $2^{16} = 65536$ possible boolean functions over four input features. We can’t figure out which one is correct until we’ve seen every possible input-output pair. After 7 examples, we still have $2^9$ possibilities.
Solution: Work with a restricted hypothesis space

- Either by applying prior knowledge or by guessing, we choose a space of hypotheses $H$ that is smaller than the space of all possible functions:
  - simple conjunctive rules
  - $m$-of-$n$ rules
  - linear functions
  - multivariate Gaussian joint probability distributions
  - etc.
Illustration: Simple Conjunctive Rules

- There are only 16 simple conjunctions (no negation)
- However, no simple rule explains the data. The same is true for simple clauses

<table>
<thead>
<tr>
<th>Example</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rule</th>
<th>Counterexample</th>
</tr>
</thead>
<tbody>
<tr>
<td>true $\iff y$</td>
<td>1</td>
</tr>
<tr>
<td>$x_1$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_2$ $\iff y$</td>
<td>2</td>
</tr>
<tr>
<td>$x_3$ $\iff y$</td>
<td>1</td>
</tr>
<tr>
<td>$x_4$ $\iff y$</td>
<td>7</td>
</tr>
<tr>
<td>$x_1 \land x_2$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_1 \land x_3$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_1 \land x_4$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_2 \land x_3$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_2 \land x_4$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_3 \land x_4$ $\iff y$</td>
<td>4</td>
</tr>
<tr>
<td>$x_1 \land x_2 \land x_3$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_1 \land x_2 \land x_4$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_1 \land x_3 \land x_4$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_2 \land x_3 \land x_4$ $\iff y$</td>
<td>3</td>
</tr>
<tr>
<td>$x_1 \land x_2 \land x_3 \land x_4$ $\iff y$</td>
<td>3</td>
</tr>
</tbody>
</table>
A larger hypothesis space: \textit{m-of-n} rules

- At least $m$ of the $n$ variables must be true
- There are 32 possible rules
- Only one rule is consistent!

\begin{center}
\begin{tabular}{cccc}
\hline
\textbf{example} & \textbf{x}_1 & \textbf{x}_2 & \textbf{x}_3 & \textbf{x}_4 & \textbf{y} \\
\hline
1 & 0 & 0 & 1 & 0 & 0 \\
2 & 0 & 1 & 0 & 0 & 0 \\
3 & 0 & 0 & 1 & 1 & 1 \\
4 & 1 & 0 & 0 & 1 & 1 \\
5 & 0 & 1 & 1 & 0 & 0 \\
6 & 1 & 1 & 0 & 0 & 0 \\
7 & 0 & 1 & 0 & 1 & 0 \\
\hline
\end{tabular}
\end{center}

\begin{center}
\begin{tabular}{|c|c|c|c|}
\hline
\textbf{variables} & \textbf{1-of} & \textbf{2-of} & \textbf{3-of} & \textbf{4-of} \\
\hline
\{x_1\} & 3 & - & - & - \\
\{x_2\} & 2 & - & - & - \\
\{x_3\} & 1 & - & - & - \\
\{x_4\} & 7 & - & - & - \\
\{x_1, x_2\} & 3 & 3 & - & - \\
\{x_1, x_3\} & 4 & 3 & - & - \\
\{x_1, x_4\} & 6 & 3 & - & - \\
\{x_2, x_3\} & 2 & 3 & - & - \\
\{x_2, x_4\} & 2 & 3 & - & - \\
\{x_3, x_4\} & 4 & 4 & - & - \\
\{x_1, x_2, x_3\} & 1 & 3 & 3 & - \\
\{x_1, x_2, x_4\} & 2 & 3 & 3 & - \\
\{x_1, x_3, x_4\} & 1 & *** & 3 & - \\
\{x_2, x_3, x_4\} & 1 & 5 & 3 & - \\
\{x_1, x_2, x_3, x_4\} & 1 & 5 & 3 & 3 \\
\hline
\end{tabular}
\end{center}
Two Views of Learning

• View 1: **Learning is the removal of our remaining uncertainty**
  – Suppose we *knew* that the unknown function was an \( m \)-of-\( n \) boolean function. Then we could use the training examples to *deduce* which function it is.

• View 2: **Learning requires guessing a good, small hypothesis class**
  – We can start with a very small class and enlarge it until it contains an hypothesis that fits the data
We could be wrong!

- Our prior “knowledge” might be wrong
- Our guess of the hypothesis class could be wrong
  - The smaller the class, the more likely we are wrong
Two Strategies for Machine Learning

- Develop Languages for Expressing Prior Knowledge
  - Rule grammars, stochastic models, Bayesian networks
  - (Corresponds to the Prior Knowledge view)
- Develop Flexible Hypothesis Spaces
  - Nested collections of hypotheses: decision trees, neural networks, cases, SVMs
  - (Corresponds to the Guessing view)
- In either case we must develop algorithms for finding an hypothesis that fits the data
Terminology

- **Training example.** An example of the form \( \langle \mathbf{x}, y \rangle \). \( \mathbf{x} \) is usually a vector of features, \( y \) is called the class label. We will index the features by \( j \), hence \( x_j \) is the \( j \)-th feature of \( \mathbf{x} \). The number of features is \( n \).

- **Target function.** The true function \( f \), the true conditional distribution \( P(y | \mathbf{x}) \), or the true joint distribution \( P(\mathbf{x}, y) \).

- **Hypothesis.** A proposed function or distribution \( h \) believed to be similar to \( f \) or \( P \).

- **Concept.** A boolean function. Examples for which \( f(\mathbf{x})=1 \) are called positive examples or positive instances of the concept. Examples for which \( f(\mathbf{x})=0 \) are called negative examples or negative instances.
Terminology

- **Classifier.** A discrete-valued function. The possible values \( f(x) \in \{1, \ldots, K\} \) are called the classes or class labels.

- **Hypothesis space.** The space of all hypotheses that can, in principle, be output by a particular learning algorithm.

- **Version Space.** The space of all hypotheses in the hypothesis space that have not yet been ruled out by a training example.

- **Training Sample (or Training Set or Training Data):** A set of \( N \) training examples drawn according to \( P(x, y) \).

- **Test Set:** A set of training examples used to evaluate a proposed hypothesis \( h \).

- **Validation Set/Development Set:** A set of training examples (typically a subset of the training set) used to guide the learning algorithm and prevent overfitting.
S, G, and the Version Space

most specific hypothesis, $S$

most general hypothesis, $G$

$h \in H$, between $S$ and $G$ is consistent

and make up the version space

(Mitchell, 1997)
Key Issues in Machine Learning

• What are good hypothesis spaces?
  – which spaces have been useful in practical applications?

• What algorithms can work with these spaces?
  – Are there general design principles for learning algorithms?

• How can we optimize accuracy on future data points?
  – This is related to the problem of “overfitting”

• How can we have confidence in the results? (the statistical question)
  – How much training data is required to find an accurate hypotheses?

• Are some learning problems computationally intractable? (the computational question)

• How can we formulate application problems as machine learning problems? (the engineering question)