Week #1: Introduction
Administrative Stuff
Logistics

• **Instructor:** Asst. Prof. Aykut ERDEM
  [aykut@cs.hacettepe.edu.tr](mailto:aykut@cs.hacettepe.edu.tr)
  Office: 111
  Tel: 297 7500, 146

• **Lectures:** Mon 10:00 - 11:45_D9
  Fri 09:00 - 09:45_D9

• **Office hour:** By appointment
About BBM 406

• This is a undergraduate-level introductory course in machine learning (ML) which will give a broad overview of many concepts and algorithms in ML.

• The goal is to provide students with a deep understanding of the subject matter and skills to apply these concepts to real world problems.
Communication

• The course webpage will be updated regularly throughout the semester with lecture notes, programming and reading assignments and important deadlines.


• We will be using Piazza for course related discussions and announcements. Please enroll the class on Piazza by following the link

  http://piazza.com/class#spring2014/bbm406
Prerequisites

• Basic algorithms, data structures.
• Basic probability and statistics.
• Basic linear algebra and calculus.
• Good programming skills.

• Bayesian Reasoning and Machine Learning, Barber, Cambridge University Press, 2012. (online version available)

• Introduction to Machine Learning (2nd Edition), Alpaydin, MIT Press, 2010

• Pattern Recognition and Machine Learning, Bishop, Springer, 2006

• Machine Learning: A Probabilistic Perspective, Murphy, MIT Press, 2012
Grading Policy

• 30% Problem Sets
• 30% Midterm Exam
• 40% Final Exam
Problem Sets

• There will be 3 problem sets.

• Each one will involve both theoretical and practical exercises.

• Each assignment have to be done individually or in pairs.

• Start your problem sets early!
Course Outline

• Week1  Overview of Machine Learning
• Week2  Linear Regression, Least Squares  PS1 out
• Week3  Machine Learning Methodology
• Week4  Statistical Estimation
• Week5  Classification  PS1 due, PS2 out
• Week6  Support Vector Machines
• Week7  Support Vector Machines (cont'd.)
Course Outline (cont’d.)

• Week 8  *Midterm exam*

• Week 9  Decision Tree Learning  *PS2 due, PS3 out*

• Week 10  Ensemble Methods

• Week 11  Instance-based Methods

• Week 12  Neural Networks  *PS3 due*

• Week 13  Unsupervised Learning

• Week 14  Factor Analysis
Welcome to BBM 406!
This week

• An overview of Machine Learning

• Nearest-neighbor classifier
Machine Learning: An Overview
What I cannot create, I do not understand.

Know how to solve every problem that has been solved.

\[ f = uYr(1, a) \]
\[ g = (1-x)a(x, a) \]
\[ f = 2|u(a)|^2 \]
Two definitions of learning

(1) Learning is the acquisition of knowledge about the world.

   Kupfermann (1985)

(2) Learning is an adaptive change in behavior caused by experience.

   Shepherd (1988)
Empirical Inference

• Drawing conclusions from empirical data (observations, measurements)

• Example 1: Scientific inference

\[ y = \sum a_i k(x, x_i) + b \]

\[ y = a \times x \]

Leibniz, Weyl, Chaitin
Empirical Inference

• Example 2: Perception
Bernhard Schölkopf
Bernhard Schölkopf
Edward H. Adelson
What is machine learning? (by examples)
Spam Filtering

- **ham**
  - Southwest Airlines: Your trip is around the corner! - You're all set for your San Jose trip! My Account | View My Itinerary Online
  - DiscountMags.com: $3.99 Business & Finance Sale... starts now! - Trouble Seeing This Email? View as Webpage STOP these e-r
  - support, Alex (3): Your order has shipped... please send to the address below for an exchange remotesremotes.com(exchange
  - Taesup, Alex, Taesup (3): Happy new year! - Hi Alex, Thanks for your condolence. I will arrive at Berkeley on 16th (wed) night. So, I can

- **spam**
  - mae
  - Dear Valued Customers, Low Interest Rate Loan - Dear Valued Customers, Do you need a loan or funding for any of the following reas
  - Steven Cooke: Congratulations Alex, $150 awaits you - Alex: IMPORTANT - NOTICE OF WINNINGS Please make sure yc
  - First-Class Mail Service: Tracking ID (G)BGD03 849 603 4893 4550 - Fed Ex Order: JN-3339-28891786 Order Date: Thursday, 3 Janua
  - Candy.Li: 中层,不只当老板的代言人
  - Ronan Morgan: Ronan Morgan just sent you a personal message. - LinkedIn Ronan Morgan just sent you a private messag
  - RE/MAX®: 2013 Valueable Offer! - Hello Friend, RE/MAX® has issued 2013 valuable property offer in your resident from
  - newsletter: newsletter WWW2013 - Newsletter 6 - See the Portuguese and Spanish version right after the English versio
  - garjetti: Chinese Journal of Cancer Research (CJCR) has been indexed by Pubmed and PMC - Click here if this e-mail
  - Wayne Smith: Wayne Smith has sent you a message - Linked In Wayne Smith just sent you a message Date: 1/09/2013 Hi
Collaborative Filtering

Don’t mix preferences on Netflix!

Amazon books
Avatar learns from your behavior

Black & White
Lionsgate Studios
Cheque Reading

segment image

Photograph Front of Check:
Place the check on a dark background in a well-lit area, keep the camera steady and align the check's edges with the frame.

Note: Fidelity cannot act on any written instructions

recognize handwriting
Topic Models of Text Documents
“In May 1997, an IBM supercomputer known as Deep Blue beat then chess world champion Garry Kasparov, who had once bragged he would never lose to a machine.

Kasparov and other chess masters blamed the defeat on a single move made by the IBM machine. Either at the end of the first game or the beginning of the second, depending on who's telling the story, the computer made a sacrifice that seemed to hint at its long-term strategy.

Kasparov and many others thought the move was too sophisticated for a computer, suggesting there had been some sort of human intervention during the game. "It was an incredibly refined move, of defending while ahead to cut out any hint of countermoves," grandmaster Yasser Seirawan told Wired.com in 2001, "and it sent Garry into a tizzy."

Fifteen years after the historical match, one of Big Blue's designers says the move was the result of a bug in Deep Blue's software.”

http://www.wired.co.uk/news/archive/2012-10/01/deep-blue-bug
• IBM Watson wins on Jeopardy (February 2011)

• Watson provides cancer treatment options to doctors in seconds (February 2013)
Google Self-Driving Cars

Autonomous Driving

Google's modified Toyota Prius uses an array of sensors to navigate public roads without a human driver. Other components, not shown, include a GPS receiver and an inertial motion sensor.

LIDAR
A rotating sensor on the roof scans more than 200 feet in all directions to generate a precise three-dimensional map of the car's surroundings.

VIDEO CAMERA
A camera mounted near the rear-view mirror detects traffic lights and helps the car's onboard computers recognize moving obstacles like pedestrians and bicyclists.

POSITION ESTIMATOR
A sensor mounted on the left rear wheel measures small movements made by the car and helps to accurately locate its position on the map.

RADAR
Four standard automotive radar sensors, three in front and one in the rear, help determine the positions of distant objects.

Source: Google

• Google’s self-driving car passes 300,000 miles (Forbes, 8/15/2012)
Search Ads

why these ads?
Weather Prediction

Temperature
72° F
Speech Recognition

Given an audio waveform, robustly extract & recognize any spoken words

- Statistical models can be used to
  - Provide greater robustness to noise
  - Adapt to accent of different speakers
  - Learn from training
I need to hide a body
noun, verb, preposition, …
Face Detection

1.2. Supervised learning

Figure 1.6 Example of face detection. (a) Input image (Murphy family, photo taken 5 August 2010 by Bernard Diedrich of Sherwood Studios). (b) Output of classifier, which detected 5 faces at different poses.

This was produced using the online demo at http://demo.pittpatt.com/. The classifier was trained on 1000s of manually labeled images of faces and non-faces, and then was applied to a dense set of overlapping patches in the test image. Only the patches whose probability of containing a face was sufficiently high were returned. Used with kind permission of Pittpatt.com.

This flexibility is both a blessing (since the methods are general purpose) and a curse (since the methods ignore an obviously useful source of information). We will discuss methods for exploiting structure in the input features later in the book.

Face detection and recognition

A harder problem is to find objects within an image; this is called object detection or object localization. An important special case of this is face detection. One approach to this problem is to divide the image into many small overlapping patches at different locations, scales and orientations, and to classify each such patch based on whether it contains face-like texture or not. This is called a sliding window detector. The system then returns those locations where the probability of face is sufficiently high. See Figure 1.6 for an example. Such face detection systems are built-in to most modern digital cameras; the locations of the detected faces are used to determine the center of the auto-focus. Another application is automatically blurring out faces in Google’s StreetView system.

Having found the faces, one can then proceed to perform face recognition, which means estimating the identity of the person (see Figure 1.10(a)). In this case, the number of class labels might be very large. Also, the features one should use are likely to be different than in the face detection problem: for recognition, subtle differences between faces such as hairstyle may be important for determining identity, but for detection, it is important to be invariant to such details, and to just focus on the differences between faces and non-faces. For more information about visual object detection, see e.g., (Szeliski 2010).

Based on classifiers trained from tens of thousands of example faces (Viola & Jones, 2004)
Visual Scene Understanding

- skyscraper
- sky
- trees
- buildings
- temple
- dome
- bell
- temple
- bell

Images showing visual scene understanding with labeled objects.
Learning - revisited

Learning

\[ \text{data} \rightarrow \text{knowledge} \]
Learning - revisited

prior knowledge → Learning → knowledge

data → Learning
Learning - revisited

Learning

prior knowledge

data

knowledge

Crucial open problem: weak intermediate forms of knowledge that support future generalizations

8/25/11
CS 194-10
Fall 2011, Stuart Russell
Programming with Data

• Want adaptive robust and fault tolerant systems
• Rule-based implementation is (often)
  - difficult (for the programmer)
  - brittle (can miss many edge-cases)
  - becomes a nightmare to maintain explicitly
  - often doesn’t work too well (e.g. OCR)

• Usually easy to obtain examples of what we want
  IF x THEN DO y
• Collect many pairs \((x_i, y_i)\)
• Estimate function \(f\) such that \(f(x_i) = y_i\) (supervised learning)
• Detect patterns in data (unsupervised learning)
Objectives of Machine Learning

• **Algorithms:** design of efficient, accurate, and general learning algorithms to
  – deal with large-scale problems.
  – make accurate predictions (unseen examples).
  – handle a variety of different learning problems.

• **Theoretical questions:**
  – what can be learned? Under what conditions?
  – what learning guarantees can be given?
  – what is the algorithmic complexity?
• **Example:** an object, instance of the data used.

• **Features:** the set of attributes, often represented as a vector, associated to an example (e.g., height and weight for gender prediction).

• **Labels:** in classification, category associated to an object (e.g., positive or negative in binary classification); in regression real value.

• **Training data:** data used for training learning algorithm (often labeled data).
Definitions and Terminology (cont’d.)

- **Test data**: data used for testing learning algorithm (unlabeled data).

- **Unsupervised learning**: no labeled data.

- **Supervised learning**: uses labeled data.

- **Semi-supervised learning and transduction**: intermediate scenarios.
Supervised Learning Example - SPAM Detection

- **Input:** email
- **Output:** spam/ham
- **Setup:**
  - Get a large collection of example emails, each labeled “spam” or “ham”
  - Note: someone has to hand label all this data!
  - Want to learn to predict labels of new, future emails
- **Features:** The attributes used to make the ham / spam decision
  - Words: FREE!
  - Text Patterns: $dd, CAPS
  - Non-text: SenderInContacts
  - ...
Unsupervised Learning Example - Segmentation
Important challenges in ML

- How important is the actual learning algorithm and its tuning
- Simple versus complex algorithm
- Overfitting
- Model Selection
- Regularization
The basic machine learning framework

\[ y = f(x) \]

- **Learning**: given a training set of labeled examples \[ \{(x_1, y_1), \ldots, (x_N, y_N)\} \], estimate the parameters of the prediction function \( f \)

- **Inference**: apply \( f \) to a never before seen test example \( x \) and output the predicted value \( y = f(x) \)
Your 1st Classifier: Nearest Neighbor Classifier
• **Definition:** Acquire an operational definition of a general category of objects given positive and negative training examples.

• Also called binary classification, binary supervised learning
Concept Learning Example

<table>
<thead>
<tr>
<th>correct (complete, partial, guessing)</th>
<th>color (yes, no)</th>
<th>original (yes, no)</th>
<th>presentation (clear, unclear, cryptic)</th>
<th>binder (yes, no)</th>
<th>A+</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 complete</td>
<td>yes</td>
<td>yes</td>
<td>clear</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>2 complete</td>
<td>no</td>
<td>yes</td>
<td>clear</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>3 partial</td>
<td>yes</td>
<td>no</td>
<td>unclear</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>4 complete</td>
<td>yes</td>
<td>yes</td>
<td>clear</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

- **Instance Space X**: Set of all possible objects describable by attributes (often called features).

- **Concept c**: Subset of objects from X (c is unknown).

- **Target Function f**: Characteristic function indicating membership in c based on attributes (i.e. label) (f is unknown).

- **Training Data S**: Set of instances labeled with target function.
• Task:
  – Learn (to imitate) a function $f: X \rightarrow \{+1,-1\}$

• Training Examples:
  – Learning algorithm is given the correct value of the function for particular inputs $\rightarrow$ training examples
  – An example is a pair $(x, y)$, where $x$ is the input and $y=f(x)$ is the output of the target function applied to $x$.

• Goal:
  – Find a function
    \[
    h: X \rightarrow \{+1,-1\}
    \]
  that approximates
    \[
    f: X \rightarrow \{+1,-1\}
    \] as well as possible.
Supervised Learning

- **Task:**
  - Learn (to imitate) a function $f: X \rightarrow Y$

- **Training Examples:**
  - Learning algorithm is given the correct value of the function for particular inputs $\rightarrow$ training examples
  - An example is a pair $(x, f(x))$, where $x$ is the input and $y = f(x)$ is the output of the target function applied to $x$.

- **Goal:**
  - Find a function $h: X \rightarrow Y$
  - that approximates $f: X \rightarrow Y$ as well as possible.
Nearest Neighbors

- Table lookup
  For previously seen instance remember label
- Nearest neighbor
  - Pick label of most similar neighbor
  - Slight improvement - use k-nearest neighbors
  - Really useful baseline!
  - Easy to implement for small amounts of data. Why?
K-Nearest Neighbor (kNN)

• Given: Training data \((x_1, y_1), \ldots, (x_n, y_n)\)
  – Attribute vectors: \(x_i \in X\)
  – Labels: \(y_i \in Y\)

• Parameter:
  – Similarity function: \(K : X \times X \to \mathbb{R}\)
  – Number of nearest neighbors to consider: \(k\)

• Prediction rule
  – New example \(x'\)
  – K-nearest neighbors: \(k\) train examples with largest \(K(x_i, x')\)

\[
h(x') = \arg \max_{y \in Y} \left\{ \sum_{i \in \text{knn}(x')} 1[y_i = y] \right\}
\]
K-Nearest Neighbor (kNN)

- How will new examples be classified?
  - Similarity function?
  - Value of $k$?

\[
h(x') = \arg \max_{y \in Y} \left\{ \sum_{i \in \text{knn}(x')} 1[y_i = y] \right\}
\]
Example - Document Classification

Recall: Vector Space Representation

Each document is a vector, one component for each term (= word).

```
Doc 1  Doc 2  Doc 3 ...
Word 1 300 . . .
Word 2 081 . . .
Word 3 12 1 10 ...
... ...
Word n 0 1 3 ...
```

component for each term (= word).

Normalize to unit length.

High-dimensional vector space: terms are axes, 10,000+ dimensions, or even 100,000+

Docs are vectors in this space.

Example

- Sports
- Science
- Arts
Example - Document Classification

Recall: Vector Space Representation

Each document is a vector, one component for each term (= word). Doc 1 Doc 2 Doc 3 ...

Word 1 300 . . . Word 2 081 . . . Word 3 12 1 10 ...

... 0

... 000 . . .

Normalize to unit length.

High-dimensional vector space: Terms are axes, 10,000+ dimensions, or even 100,000+

Docs are vectors in this space

© Eric Xing @ CMU, 2006-2012
Recall: Vector Space Representation

Each document is a vector, one component for each term (= word).

\[
\text{Doc 1 \ Doc 2 \ Doc 3 ...}
\]

\[
\begin{array}{cccc}
\text{Word 1} & \text{300} & \ldots \\
\text{Word 2} & \text{081} & \ldots \\
\text{Word 3} & \text{12} & \text{1} & \text{10} & \ldots \\
\end{array}
\]

\[\vdots\]

Normalize to unit length.

High-dimensional vector space: terms are axes, 10,000+ dimensions, or even 100,000+

Example - Document Classification

Test Document = ?

\[
\begin{align*}
\text{Sports} & \quad \text{Science} & \quad \text{Arts} \\
\end{align*}
\]
Recall: Vector Space Representation

Each document is a vector, one component for each term (= word).

\[ z \]

Normalize to unit length.

High-dimensional vector space:
Terms are axes, 10,000+ dimensions, or even 100,000+

Docs are vectors in this space.

Test Document = ?
Example - Document Classification

Recall: Vector Space

Each document is a vector, one component for each term (= word).

$z_{\text{Doc 1 Doc 2 Doc 3 ... Word 1}} = 300 \ldots$  
$z_{\text{Word 2}} = 081 \ldots$  
$z_{\text{Word 3}} = 12 1 10 \ldots$  

$z$ Normalize to unit length.

$z$ High-dimensional vector space:
Terms are axes, 10,000+ dimensions, or even 100,000+ $z$

Docs are vectors in this space

$S$ports  
$S$cience  
$A$rts

Voting kNN
If we get more data

• 1 Nearest Neighbor
  • Converges to perfect solution if clear separation
  • Twice the minimal error rate $2p(1-p)$ for noisy problems

• $k$-Nearest Neighbor
  • Converges to perfect solution if clear separation (*but needs more data*)
  • Converges to minimal error $\min(p, 1-p)$ for noisy problems if $k$ increases
Weighted K-Nearest Neighbor

- Given: Training data \((x_1, y_1), \ldots, (x_n, y_n)\)
  - Attribute vectors: \(x_i \in X\)
  - Target attribute \(y_i \in Y\)

- Parameter:
  - Similarity function: \(K : X \times X \rightarrow \mathbb{R}\)
  - Number of nearest neighbors to consider: \(k\)

- Prediction rule
  - New example \(x'\)
  - K-nearest neighbors: \(k\) train examples with largest \(K(x_i, x')\)

\[
h(x') = \arg \max_{y \in Y} \left\{ \sum_{i \in \text{knn}(x')} 1_{[y_i = y]} K(x_i, x') \right\}
\]
1-Nearest Neighbor
4-Nearest Neighbors
4-Nearest Neighbors Sign
• Torralba et al., 80 million tiny images: a large dataset for non-parametric object and scene recognition, IEEE TPAMI, 2008
Nearest neighbor search in higher dimensions

Linear Search:
  e.g. scanning 4.5M images!

k-D trees:
  axis parallel partitions of the data
  Only effective in low-dimensional data

Large Scale Approximate Indexing
  Locality Sensitive Hashing (LSH), Spill-Tree, NV-Tree
  All above run on a single machine with all data in memory, and scale to big data

Web-scale Approximate Indexing
  Parallel variant of Spill-tree, NV-tree on distributed systems,
  Scale to Billions of images in disks on multiple machines
Overview of Nearest Neighbors

- Very simple method
- Retain all training data
  - Can be slow in testing
  - Finding NN in high dimensions is slow
- Metrics are very important
- Good baseline