Learning from massive data

- Many applications require gaining insights from massive, noisy data sets

**Science**
- Physics (LHC, ...), Astronomy (sky surveys, ...), Neuroscience (fMRI, micro-electrode arrays, ...), Biology (High-throughput microarrays, ...), Geology (sensor arrays, ...), ...
- Social science, economics, ...

**Commercial / civil applications**
- Consumer data (online advertising, viral marketing, ...)
- Health records (evidence based medicine, ...)

**Security / defense related applications**
- Spam filtering / intrusion detection
- Surveillance, ...
Web-scale machine learning

- Predict relevance of search results from click data
- Personalization
- Online advertising
- Machine translation
- Learning to index
- Spam filtering
- Fraud detection
- ...

>21 billion indexed web pages
Analyzing fMRI data

- Predict activation patterns for nouns
- Google’s Trillion word corpus used to measure co-occurrence

Mitchell et al., Science, 2008
Monitoring transients in astronomy [Djorgovski]

Novae, Cataclysmic Variables

Supernovae

Gamma-Ray Bursts

Gravitational Microlensing

Accretion to SMBHs
Data-rich astronomy [Djorgovski]

- Typical digital sky survey now generates ~10 - 100 TB, plus a comparable amount of derived data products
  - PB-scale data sets are on the horizon
- Astronomy today has ~1 - 2 PB of archived data, and generates a few TB/day
  - Both data volumes and data rates grow exponentially, with a doubling time ~ 1.5 years
  - Even more important is the growth of data complexity

For comparison:
- Human memory ~ a few hundred MB
- Human Genome < 1 GB
- 1 TB ~ 2 million books
- Library of Congress (print only) ~ 30 TB
How is the data-rich science different? [Djorgovski]

- The information volume grows exponentially
  
  *Most data will never be seen by humans*

- The need for data storage, network, database-related technologies, standards, etc.

- Information *complexity* is also increasing greatly
  
  *Most data (and data constructs) cannot be comprehended by humans directly*

  The need for data mining, KDD, data understanding technologies, hyperdimensional visualization, AI/Machine-assisted discovery ...

- We need to create a *new scientific methodology* to do the 21st century, computationally enabled, data-rich science...

- ML and AI will be essential components of the new scientific toolkit
Data volume in scientific and industrial applications

[Image showing a graph with data points for various organizations and years, including AT&T, Walmart, eBay, Facebook, Google, Yahoo!, Microsoft, LHC, LSST, NASA, BaBar, and LSST. The X-axis represents the year from 2000 to 2025, and the Y-axis represents petabytes.]

[Meiron et al]
How can we get **gain insight** from massive, noisy data sets?
Key questions

- How can we deal with data sets that don’t fit in main memory of a single machine?
  - Online learning

- Labels are expensive. How can we obtain most informative labels at minimum cost?
  - Active learning

- How can we adapt complexity of classifiers for large data sets?
  - Nonparametric learning
Overview

- Research-oriented advanced topics course
- 3 main topics
  - Online learning (from streaming data)
  - Active learning (for gathering most useful labels)
  - Nonparametric learning (for model selection)
- Both theory and applications
- Handouts etc. on course webpage
  - http://www.cs.caltech.edu/courses/cs253/
Overview

- **Instructors:**
  Andreas Krause ([krausea@caltech.edu](mailto:krausea@caltech.edu)) and Daniel Golovin ([dgolovin@caltech.edu](mailto:dgolovin@caltech.edu))

- **Teaching assistant:**
  Deb Ray ([dray@caltech.edu](mailto:dray@caltech.edu))

- **Administrative assistant:**
  Sheri Garcia ([sheri@cs.caltech.edu](mailto:sheri@cs.caltech.edu))
Formal requirement:
CS/CNS/EE 156a or instructor’s permission
Coursework

- Grading based on
  - 3 homework assignments (one per topic) (50%)
  - Course project (40%)
  - Scribing (10%)

- 3 late days

- Discussing assignments allowed, but everybody must turn in their own solutions

- Start early! 😊
Course project

“Get your hands dirty” with the course material
Implement an algorithm from the course or a paper you read and apply it to some data set
Ideas on the course website (soon)
Application of techniques you learnt to your own research is encouraged
Must be something new (e.g., not work done last term)
Project: Timeline and grading

- Small groups (2-3 students)
- January 20: Project proposals due (1-2 pages); feedback by instructor and TA
- February 10: Project milestone
- March ~10: Poster session (TBA)
- March 15: Project report due

- Grading based on quality of poster (20%), milestone report (20%) and final report (60%)

- We will have a Best Project Award!!
Course overview

- **Online learning** from massive data sets
- **Active learning** to gather most informative labels
- **Nonparametric learning** to adapt model complexity

This lecture: Quick overview over all these topics
Traditional classification task

**Input:** Labeled data set with positive (+) and negative (-) examples

**Output:** Decision rule (e.g., linear separator)
Main memory vs. disk access

**Main memory:**
Fast, random access, expensive

**Secondary memory** (hard disk)
~$10^4$ slower, sequential access, inexpensive

Massive data ➔ Sequential access

How can we learn from streaming data?
Online classification task

- Data arrives sequentially
- Need to classify one data point at a time
- Use a different decision rule (lin. separator) each time
- Can’t remember all data points!
Model: Prediction from expert advice

<table>
<thead>
<tr>
<th>Experts</th>
<th>( I_1 )</th>
<th>( I_2 )</th>
<th>( I_3 )</th>
<th>...</th>
<th>( I_T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( e_1 )</td>
<td></td>
<td>( \times )</td>
<td>( \circ )</td>
<td></td>
<td>( \times )</td>
</tr>
<tr>
<td>( e_2 )</td>
<td></td>
<td></td>
<td>( \times )</td>
<td></td>
<td>( \times )</td>
</tr>
<tr>
<td>( e_3 )</td>
<td></td>
<td></td>
<td>( \times )</td>
<td></td>
<td>( \times )</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td>( \times )</td>
<td></td>
<td>( \times )</td>
</tr>
<tr>
<td>( e_n )</td>
<td></td>
<td></td>
<td>( \times )</td>
<td></td>
<td>( \times )</td>
</tr>
</tbody>
</table>

Loss

\[ \text{Total: } \sum_t I(t, i_t) \rightarrow \min \]

**Expert** = Someone with an opinion (not necessarily someone who knows something)

*Think of an expert as a decision rule (e.g., lin. separator)*
Performance metric: Regret

- Best expert: $i^* = \min_i \sum_t l(t,i)$
- Let $i_1, \ldots, i_T$ be the sequence of experts selected
- Instantaneous regret at time $t$: $r_t = l(t,i_t) - l(t,i^*)$
- Total regret:
  $$R_T = \sum_{t=1}^{T} r_t$$
- Typical goal: Want selection strategy that guarantees
  $$\lim_{T \to \infty} \frac{R_T}{T} = 0$$
Expert selection strategies

- Pick an expert (classifier) uniformly at random?

\[ E[R_T] = \frac{T}{2} \]
\[ \lim_{T \to \infty} \frac{R_T}{T} \neq 0 \]

- Always pick the best expert?

\[ R_T = \frac{T}{2} \]
Randomized weighted majority

Input:
- Learning rate $\eta$

Initialization:
- Associate weight $w_{1,s} = 1$ with every expert $s$

For each round $t$
- Choose expert $s$ with prob. $p_{t,s} = \frac{w_{t,s}}{\sum_{s'} w_{t,s'}}$
- Obtain losses $\ell_{t,s}$
- Update weights: $w_{t+1,s} = w_{t,s} \exp(\eta \ell_{t,s})$
Guarantees for RWM

**Theorem**
For appropriately chosen learning rate, Randomized Weighted Majority obtains sublinear regret:

$$\mathbb{E}[R_T] \leq \sqrt{2T \log n}$$

**Note:** No assumption about how the loss vectors are generated!
Practical problems

- In many applications, number of experts (classifiers) is infinite
  - Online optimization (e.g., online convex programming)

- Often, only partial feedback is available
  (e.g., obtain loss only for chosen classifier)
  - Multi-armed bandits, sequential experimental design

- Many practical problems are high-dimensional
  - Dimension reduction, sketching
Course overview

- **Online learning** from massive data sets
- **Active learning** to gather most informative labels
- **Nonparametric learning** to adapt model complexity

This lecture: Quick overview over all these topics
Spam or Ham?

- Labels are expensive (need to ask expert)
- Which labels should we obtain to maximize classification accuracy?
Learning binary thresholds

- Input domain: $D=[0,1]$
- True concept $c$:
  $c(x) = +1$ if $x \geq t$
  $c(x) = -1$ if $x < t$
- Samples $x_1,...,x_n$ in $D$
  uniform at random
Passive learning

- Input domain: $D=[0,1]$
- True concept $c$:
  \[ c(x) = +1 \text{ if } x \geq t \]
  \[ c(x) = -1 \text{ if } x < t \]

- Passive learning:
  Acquire all labels $y_i \in \{+,-\}$
Active learning

- Input domain: \( D = [0,1] \)
- True concept \( c \):
  \[
  c(x) = +1 \text{ if } x \geq t \\
  c(x) = -1 \text{ if } x < t
  \]

- Passive learning:
  Acquire all labels \( y_i \in \{+, -\} \)

- Active learning:
  **Decide** which labels to obtain
Classification error

- After obtaining $n$ labels, $D_n = \{(x_1, y_1), ..., (x_n, y_n)\}$
  learner outputs hypothesis consistent with labels $D_n$

Classifica$\hat{\text{t}}$ error: $R(h) = E_{x \sim P}[h(x) \neq c(x)]$
Data source P (produces inputs $x_i$)

Active learner assembles data set
$D_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$
by selectively obtaining labels

Learner outputs hypothesis $h$

Classification error $R(h) = \mathbb{E}_{x \sim P}[h(x) \neq c(x)]$

How many labels do we need to ensure that $R(h) \cdot \varepsilon$?
Label complexity for passive learning

Need at least \( \frac{2}{\epsilon} - 1 \)

If fewer, Passive learning needs \( \gtrsim 2 \left( \frac{1}{\epsilon} \right) \) more labels
Label complexity for active learning

Binary search!
$O(\log \frac{1}{\epsilon})$ labels
### Comparison

<table>
<thead>
<tr>
<th></th>
<th>Labels needed to learn with classification error $\varepsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passive learning</td>
<td>$\Omega(1/\varepsilon)$</td>
</tr>
<tr>
<td>Active learning</td>
<td>$O(\log 1/\varepsilon)$</td>
</tr>
</tbody>
</table>

Active learning can exponentially reduce the number of required labels!
Key questions

- For which classification tasks can we provably reduce the number of labels?

- Can we do worse by active learning?

- Can we implement active learning efficiently?
Course overview

- **Online learning** from massive data sets
- **Active learning** to gather most informative labels
- **Nonparametric learning** to adapt model complexity

This lecture: Quick overview over all these topics
Nonlinear classification

How should we adapt the classifier complexity to growing data set size?
Nonlinear classification

How should we adapt the classifier complexity to growing data set size?
How should we adapt the classifier complexity to growing data set size?
Linear classification

\[ \min_w \sum_{t=1}^{T} \ell(w'x_t, y_t) + \lambda \|w\|^2 \]

Loss function
e.g., hinge loss:
\[ \ell(w'x, y) = \max(1 - y \cdot w'x, 0) \]

Complexity penalty
\[ \sum_i w_i^2 \]
From linear to nonlinear classification

**Linear classification**

\[
\min_w \sum_{t=1}^{T} \ell(w' x_t, y_t) + \lambda \|w\|^2
\]

**Nonlinear classification**

\[
\min_f \sum_{t=1}^{T} \ell(f(x_t), y_t) + \lambda \|f\|^2
\]

Complexity penalty for function \( f \)??
1D Example

Nonlinear classification

$$\text{min}_f \sum_{t=1}^{T} \ell(f(x_t), y_t) + \lambda \|f\|^2$$
Representation of function $f$

Solution of
\[
\min_{f} \sum_{t=1}^{T} \ell(f(x_t), y_t) + \lambda \|f\|^2
\]
can be written as
\[
f(x) = \sum_{t=1}^{T} \alpha_t k(x, x_t)
\]
for appropriate choice of $\|f\|$ (Representer Theorem)

Hereby, $k(\cdot, \cdot)$ is called a **kernel function**
(associated with $\|\cdot\|$)
Examples of kernels

Squared exp. kernel
\[ \exp \left( \frac{\|x - x'\|^2}{h^2} \right) \]

Exponential k.
\[ \exp \left( \frac{\|x - x'\|_1}{h} \right) \]

Finite dimensional
\[ \phi(x)^T \phi(x') \]
Nonparametric solution

Solution of

\[
\min_{f} \sum_{t=1}^{T} \ell(f(x_t), y_t) + \lambda \|f\|^{2}
\]

can be written as

\[
f(x) = \sum_{t=1}^{T} \alpha_t k(x, x_t)
\]

Function f has one parameter \(\alpha_t\) for each data point \(x_t\)!
No finite-dimensional representation \(\rightarrow\) “non-parametric”

Large data set \(\rightarrow\) Huge number of parameters!!
Key questions

- How can we determine the right tradeoff between function expressiveness (#parameters) and computational complexity?

- How can we control model complexity in an online fashion?

- How can we quantify uncertainty in nonparametric learning?
Course overview

Online Learning

Response surface methods

Nonparametric Learning

Bandit optimization

Active Learning

Active set selection