

## Course overview and general classes of problems

Hal Daumé III

CS5350: Machine Learning

25 August 2009

CS5350

Hal Daumé III (U Utah)

1 / 25

## Course Goals

By the end of the semester, you should be able to:

- ▶ Look at a problem and identify if ML is an appropriate solution
- ▶ If so, identify what types of algorithms might be applicable
- ▶ Apply those algorithms
- ▶ Conquer the world

In order to get there, you will need to:

- ▶ Do a lot of math (calculus, linear algebra, probability)
- ▶ Do a fair amount of programming
- ▶ Work hard (this is a 9-credit class)

CS5350

Hal Daumé III (U Utah)

3 / 25

## What is this course about?

- ▶ Finding (and exploiting) patterns in data
- ▶ Replacing “human writing code” with “human supplying data”
  - ⇒ System figures out what the person wants based on examples
  - ⇒ Need to abstract from “training” examples to “test” examples
  - ⇒ Most central issue in ML: *generalization*

## Why is machine learning so cool?

- ▶ Broad applicability
  - ▶ Finance, robotics, vision, machine translation, medicine, etc.
- ▶ Close connection between theory and practice
- ▶ Open field, lots of room for new work
- ▶ <http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=9026623>

CS5350

Hal Daumé III (U Utah)

2 / 25

## Student comments on this course

- ▶ The field of machine learning seems too wide for any one-semester long course to completely cover everything
- ▶ I think the projects should be changed and different to the previous semester
- ▶ This course shouldn't be CS 5350, it should be MATH 5350!
- ▶ All in all, great class, and clear interest in the needs of the students
- ▶ ... and so on ...

I try to take your comments seriously!

(but some things **won't change...**)

CS5350

Hal Daumé III (U Utah)

4 / 25

- ▶ Supervised learning: learning with a teacher
- ▶ Unsupervised learning: learning without a teacher
- ▶ Complex settings: learning in a complicated world
  - ▶ Time-series models
  - ▶ Structured prediction
  - ▶ Semi-supervised learning
  - ▶ Large-scale learning
- ▶ **Not a zoo tour!**
- ▶ **Not an introduction to tools!**
- ▶ **You will learn how these techniques work and how to implement them.**

CS5350

Hal Daumé III (U Utah)

5 / 25

CS5350

Hal Daumé III (U Utah)

6 / 25

[http://www.cs.utah.edu/~hal/courses/2009F\\_ML/](http://www.cs.utah.edu/~hal/courses/2009F_ML/)

## On Reading and Responsibilities. . .

**Reading: I expect you to do it!**  
(but most are  $\leq 12$  pages, all are  $\leq 20$ )

Class time is for:

- ▶ Discussing questions from the reading
- ▶ Discussing homework assignments
  - ▶ Some questions are starred: these will be presented in class
- ▶ Me providing an insider's view

CS5350

Hal Daumé III (U Utah)

7 / 25

CS5350

Hal Daumé III (U Utah)

8 / 25

## Requirements and Grading

**Programming projects: 40%**

- ▶ Four total
- ▶ Teams of at most three
- ▶ May be 48 hours late, at 50% mark down

**Written homeworks: 35%**

- ▶ One per week, graded 0, 0.5 or 1
- ▶ Completed individually
- ▶ May not be late at all

**Final project: 20%**

- ▶ Canned or your choice, teams
- ▶ Presentations during the final slot

**Class participation: 5%**

### Extra credit: Brain teasers

- ▶ Ten total, each worth 5%
- ▶ CS 6350 students *must* do two
- ▶ Anyone can do up to four (teams of three are allowed)

### Grading complaints

- ▶ Not allowed after one week

### How should you spend your time?

- ▶ 3 hours in class
- ▶ 2 hours reading
- ▶ 2 hours on written assignments
- ▶ 2 hours on programming projects

C:S5350

Hal Daumé III (U Utah)

9 / 25

C:S5350

Hal Daumé III (U Utah)

10 / 25

- ▶ Questions that have already been answered  
Please read the class mailing list and come to class!
- ▶ Fire and forget emails  
I'd *much* rather talk to you in person!

(See HW01 for your opportunity to respond!)

## Things you need to do now!

### Complete Homework 01

- ▶ Due 27 Aug (that's **Thursday!**, by beginning of class)
- ▶ Submit in .pdf format *only* using [handin](#)

### Complete the first reading

- ▶ See syllabus
- ▶ Due by class Thursday (I mean it!)
- ▶ Some parts of the web page are password protected!

### Sign up to get mails

- ▶ Subscribe either to mailing list or RSS feed
- ▶ But be sure to actually read it!

### Read the web page!

C:S5350

Hal Daumé III (U Utah)

11 / 25

C:S5350

Hal Daumé III (U Utah)

12 / 25

# Now, on to some real content. . .

(but first, questions?)

## Classification

- ▶ How would you write a program to distinguish a picture of **me** from a picture of **someone else**?
  - ⇒ Provide examples pictures of **me** and pictures of **other people** and let a *classifier* learn to distinguish the two.
- ▶ How would you write a program to determine whether a **sentence** is **grammatical** or **not**?
  - ⇒ Provide examples of **grammatical** and *ungrammatical* sentences and let a *classifier* learn to distinguish the two.
- ▶ How would you write a program to distinguish **cancerous cells** from **normal cells**?
  - ⇒ Provide examples of **cancerous** and *normal* cells and let a *classifier* learn to distinguish the two.

C:S5350

Hal Daumé III (U Utah)

13 / 25

## Data (“weather” prediction)

### Example dataset:

Class	Outlook	Temperature	Windy?
Play	Sunny	Low	Yes
No play	Sunny	High	Yes
No play	Sunny	High	No
Play	Overcast	Low	Yes
Play	Overcast	High	No
Play	Overcast	Low	No
No play	Rainy	Low	Yes
Play	Rainy	Low	No

### Three principle components:

1. Class label (aka “label”, denoted  $y$ )
2. Features (aka “attributes”)
3. Feature values (aka “attribute values”, denoted  $x$ )
  - ⇒ Features can be **binary**, **nomial** or **continuous**

**A labeled dataset is a collection of  $(x, y)$  pairs**









**Task:**

C:S5350

Hal Daumé III (U Utah)

14 / 25

## Data (face recognition)

Class	Image	Class	Image
Avrim		Tom	
Avrim		Tom	
Avrim		Tom	
Avrim		Tom	

**What is a good representation for images?** Pixel values? Edges?

C:S5350

Hal Daumé III (U Utah)

15 / 25

## Ingredients for classification

**Whole idea:** Inject *your* knowledge into a learning system

### Sources of knowledge:

1. Feature representation
  - ▶ Not typically a focus of machine learning
  - ▶ Typically seen as “problem specific”
  - ▶ *However*, it’s hard to learn on bad representations
2. Training data: labeled examples
  - ▶ Often expensive to label lots of data
  - ▶ Sometimes data is available for “free”
3. Model
  - ▶ No single learning algorithm is always good (“no free lunch”)
  - ▶ Different learning algorithms work with different ways of representing the learned classifier
  - ▶ When the data has nothing to say, which model is better
  - ▶ Typically requires some control over **generalization**

C:S5350

Hal Daumé III (U Utah)

16 / 25

# Regression

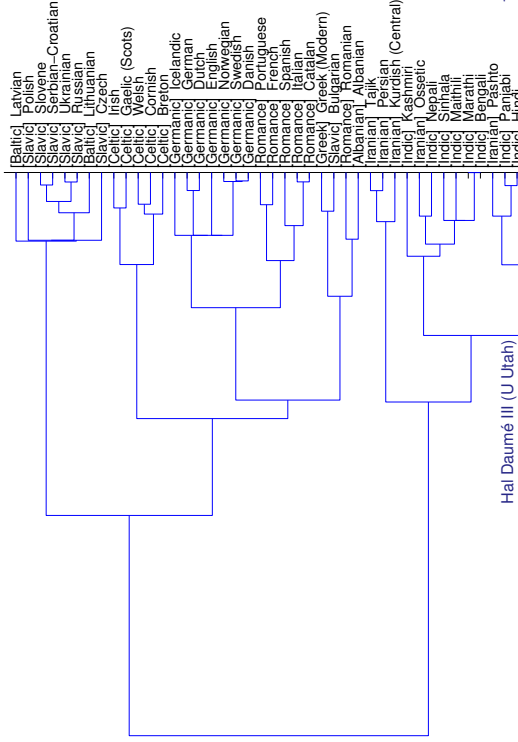
**Regression** is like **classification**, except the labels are real values.

More on generalization later...

**Example applications:**

- ▶ Stock value prediction
- ▶ Income prediction
- ▶ CPU Power consumption
- ▶ **Your grade in CS5350**

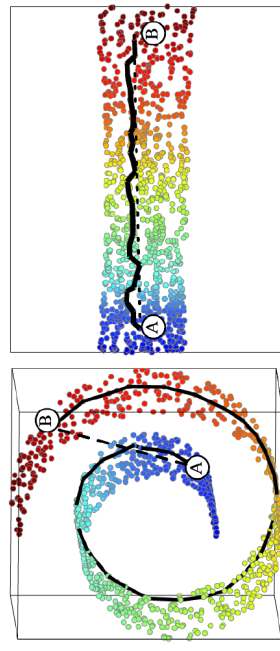
# Unsupervised learning: Clustering



# Unsupervised learning: Manifold learning

Often data that is **really** two dimensional is **embedded** in a higher dimensional space, sometimes warped.

Task is to **recover** the true **geometry** of the underlying data.



- ▶ Usually, replace “two” with “ $d$ ” and “three” with “ $D$ ” for  $d \ll D$
- ▶ Useful for **visualization** (when  $d \in \{2, 3\}$ )
- ▶ Also useful for finding good **representations** for input to classifiers

- ▶ Reinforcement learning is the penultimate ML problem
- ▶ It is “ML-hard”
- ▶ Unlike classification, regression and unsupervised learning, RL does not receive **examples**
- ▶ Rather, it gathers **experience** by interacting with the world
- ▶ RL problems always include **time** as a variable

### Example problems:

1. Chess
  2. Robot control
  3. Taxi driving
- ▶ Key trade-off is **exploration** versus **exploitation**

C:55350

Hal Daumé III (U Utah)

21 / 25

### A (simple) reinforcement learning problem is defined by:

- ▶ A **state space** that defines the *world* that our *agent* inhabits
- ▶ A set of **actions** that an agent can take in any state
- ▶ The **reward** the agent gets for reaching some particular state

The *goal* of the agent is to take a sequence of actions so as to maximize its sum of rewards.

The agent’s output is a *policy* that maps states to actions.

If you want to learn about RL, take CS5300 (AI) this coming Spring!

C:55350

Hal Daumé III (U Utah)

22 / 25

## Why do we care about math?!

- ▶ **Calculus and linear algebra:**
  1. Techniques for finding maxima/minima of functions
  2. Convenient language for high dimensional data analysis

- ▶ **Probability:**
  1. The study of the outcome of repeated experiments
  2. The study of the plausibility of some event

- ▶ **Statistics:**
  1. The analysis and interpretation of data

Statistics makes heavy use of **probability theory**.

C:55350

Hal Daumé III (U Utah)

23 / 25

## Why do we care about probability and statistics?

Recall, **statistics** is the analysis and interpretation of data.

In **machine learning**, we attempt to generalize from one “training” data set to general “rules” that can be applied to “test” data.

### How is machine learning *different* from statistics?

1. Stats cares about the **model**, we care about **predictions**
2. Stats cares about **model fit**, we care about **generalization**
3. Stats tries to **explain the world**, we try to **predict the future**



It all started with a lady drinking tea...



C:55350

Hal Daumé III (U Utah)

24 / 25

## History of ML?

- ▶ Initial attempts at object recognition [Rosenblatt, 1958]
- ▶ Learning to play checker [Samuel, 1959, 1963]
- ▶ Rosenblatt can't learn XOR [Minsky & Pappert, 1969]
- ▶ Symbolic learning, spectroscopy [Winston, 1975; Buchanan 1971]
- ▶ Backpropagation for neural nets [Werbos, 1974; Rummelhart, 1986]
- ▶ PAC model of learning theory [Valiant, 1984]
- ▶ Optimization enters machine learning [Bennett & Mangasarian, 1993]
- ▶ Kernel methods for non-linearity [Cortes & Vapnik, 1995]
- ▶ Machine learning behind day-to-day tasks [2005ish]
- ▶ Machine learning takes over the world [2010ish]