JOHNS HOPKINS UNIVERSITY

EN.600.475 Machine Learning

Introduction to ML

Raman Arora
Lecture 2
February 1, 2017

- Course overview
- Formal Introduction

Slides courtesy: Mark Dredze, Aarti Singh,

Announcements

- HW0 Out, Due next Wed, Feb 8
- Auditors allowed only if there is space
- Waitlist cleared for section (2)

Machine Learning Paradigms

Broad categories -

- **Supervised learning**

  Classification, Regression
- **Unsupervised learning**

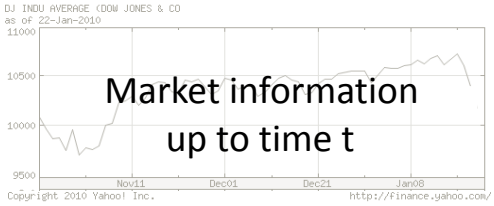
  Density estimation, Clustering, Dimensionality reduction
- Semi-supervised learning
- Active learning
- Reinforcement learning
- Many more …

Supervised Learning

Feature Space $\mathcal{X}$

Label Space $\mathcal{Y}$

"Sports"
"News"
"Science"
…



Share Price
"$ 24.50"

Task: Given $X \in \mathcal{X}$, predict $Y \in \mathcal{Y}$.

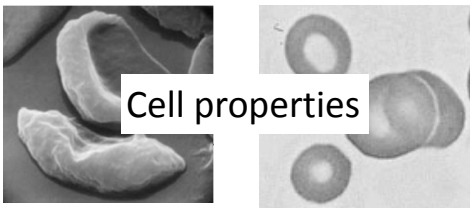
Supervised Learning - Classification

Feature Space $\mathcal{X}$

Label Space $\mathcal{Y}$

⇨

"Sports"
"News"
"Science"
…



⇨

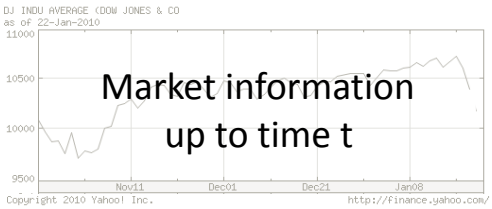
"Anemic cell"
"Healthy cell"

Discrete Labels

Supervised Learning - Regression

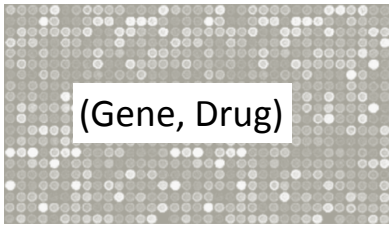
Feature Space $\mathcal{X}$

Label Space $\mathcal{Y}$



⇨

Share Price
"$ 24.50"



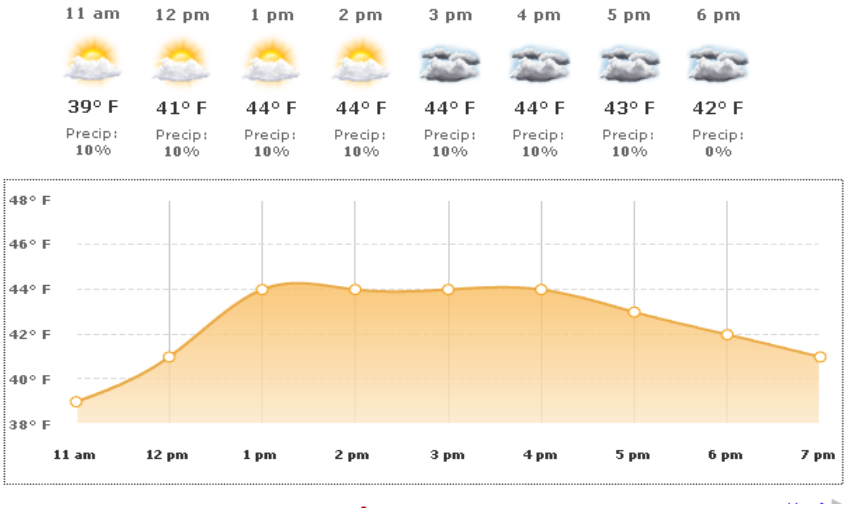
⇨

Expression level
"0.01"

Continuous Labels

Supervised Learning problems

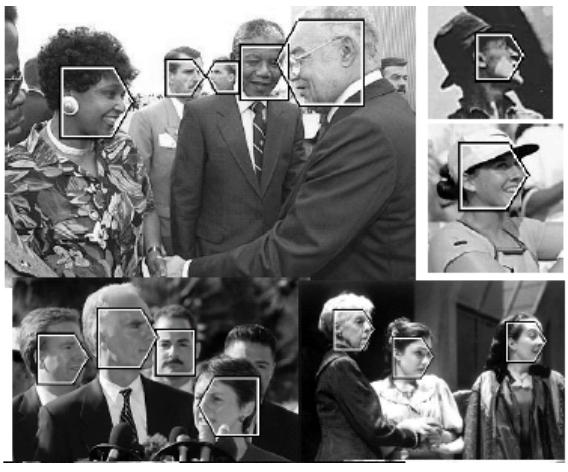
Features? **Labels?** **Classification/Regression?**



Temperature/Weather prediction

Supervised Learning problems

Features? **Labels?** **Classification/Regression?**



Face Detection

Supervised Learning problems

Features? **Labels?** **Classification/Regression?**

Environmental Mapping

Supervised Learning problems

Features? **Labels?** **Classification/Regression?**

Robotic Control

Unsupervised Learning

Aka "learning without a teacher"

Feature Space $\mathcal{X}$

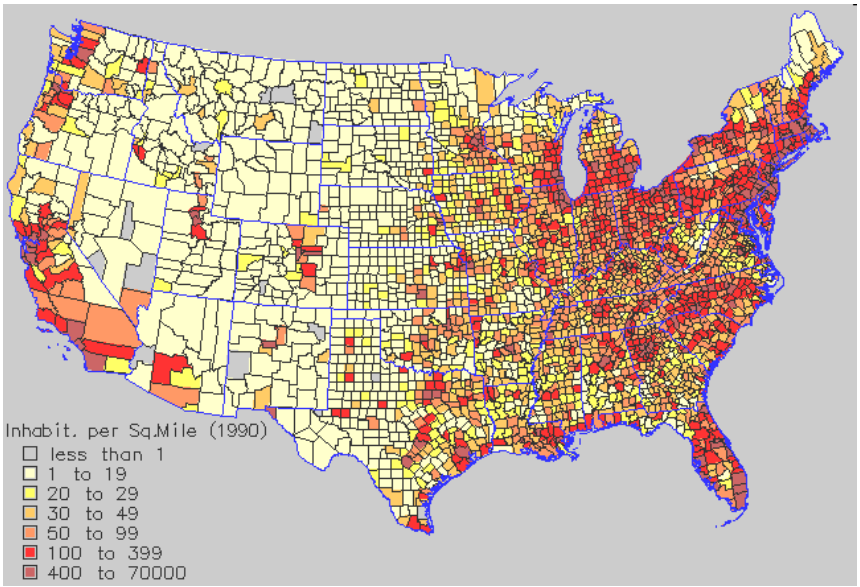
Words in a document

Word distribution
(Probability of a word)

Task: Given $X \in \mathcal{X}$, learn $f(X)$.

Unsupervised Learning – Density Estimation

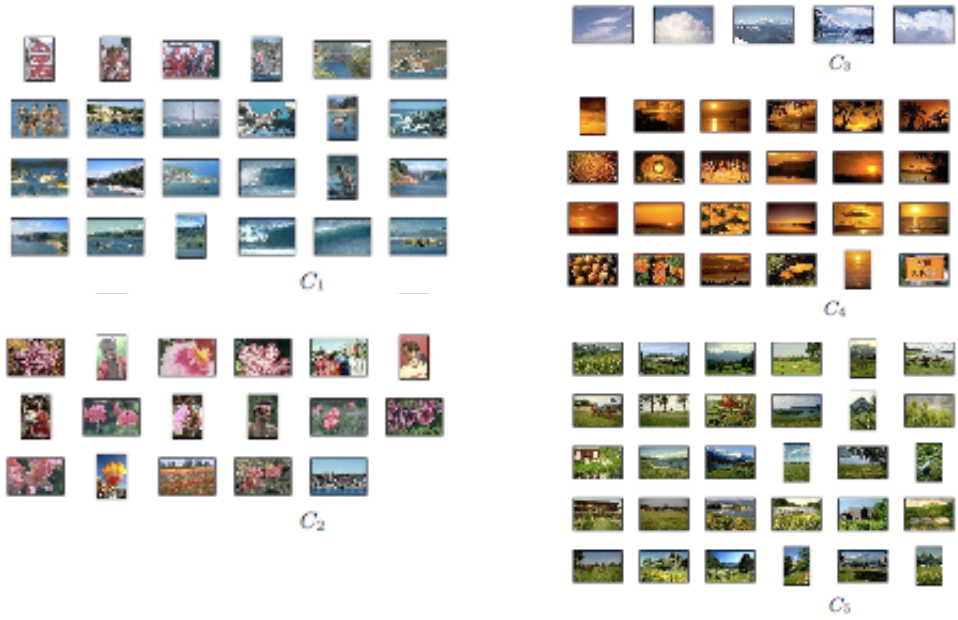
Population density



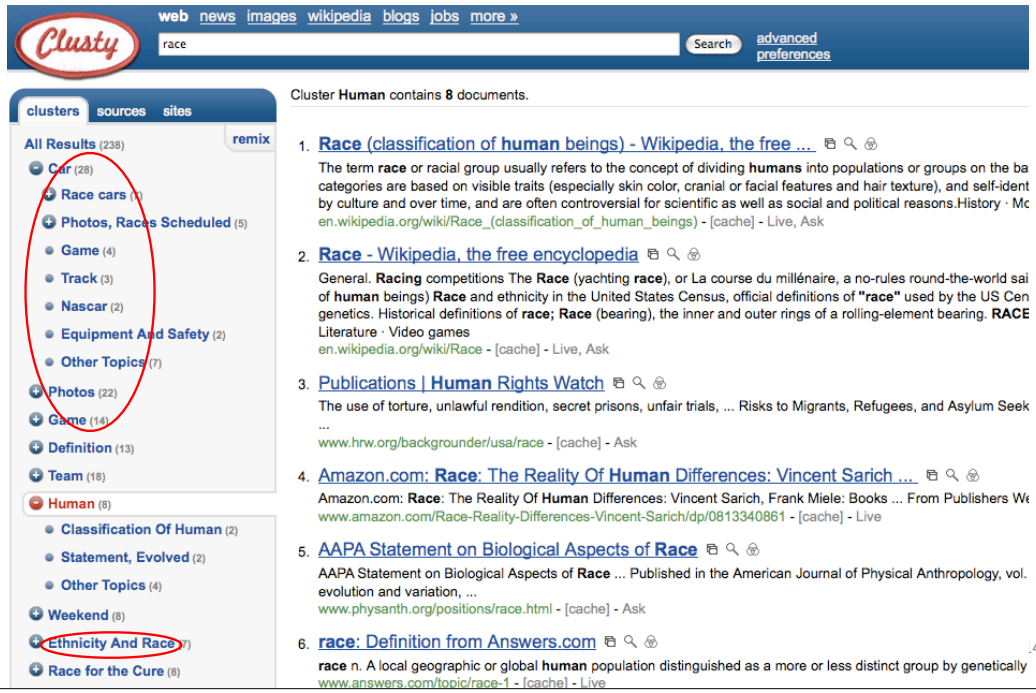
Unsupervised Learning – clustering

Group similar things e.g. images

[Goldberger et al.]



Unsupervised Learning – clustering web search results



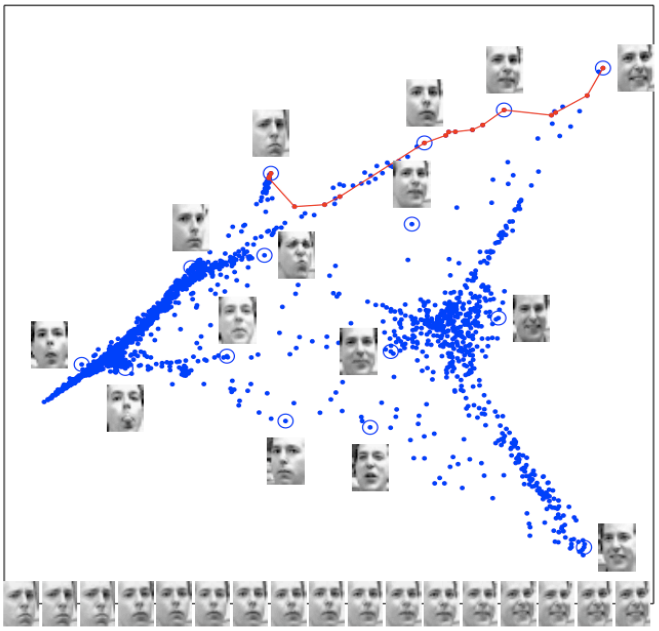
Unsupervised Learning - Embedding

Dimensionality Reduction

[Saul & Roweis '03]

Images have thousands or millions of pixels.

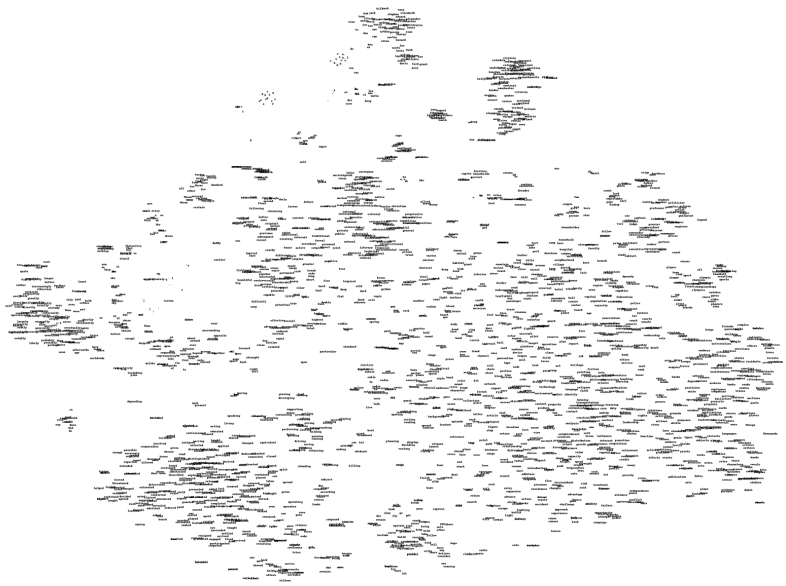
Can we give each image a coordinate,
such that similar images are near each other?



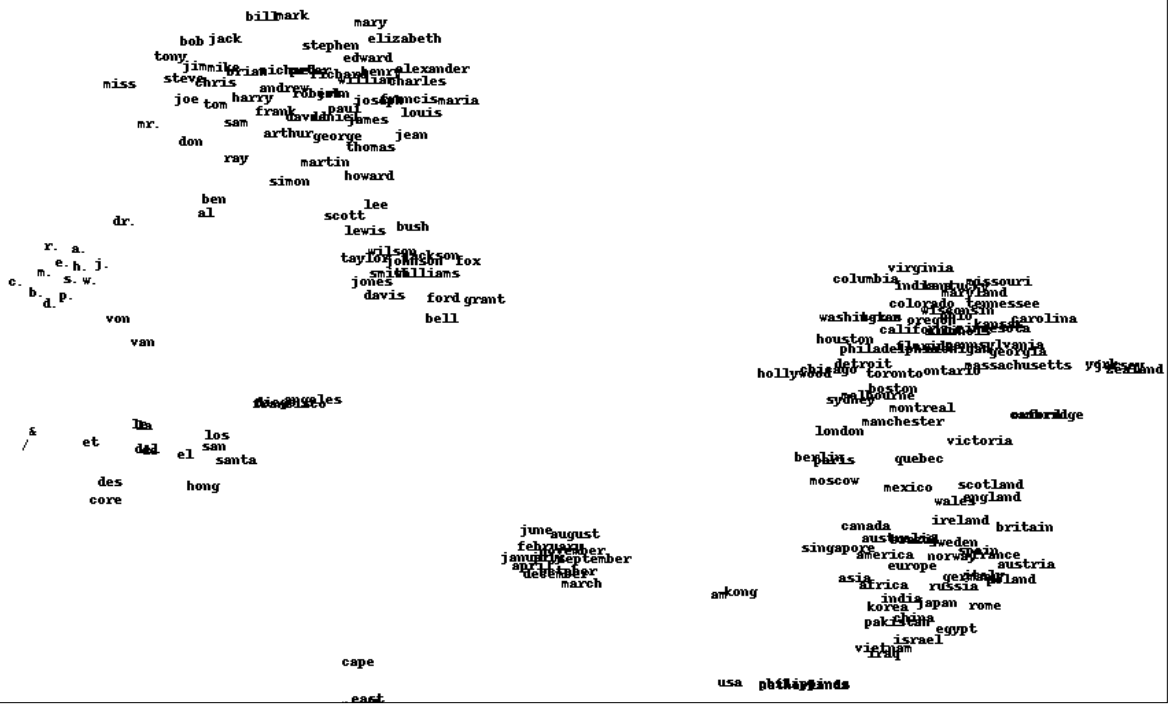
Unsupervised Learning - Embedding

Dimensionality Reduction - words

[Joseph Turian]



Unsupervised Learning - Embedding



Machine Learning Paradigms

Broad categories -

- **Supervised learning**

  Classification, Regression

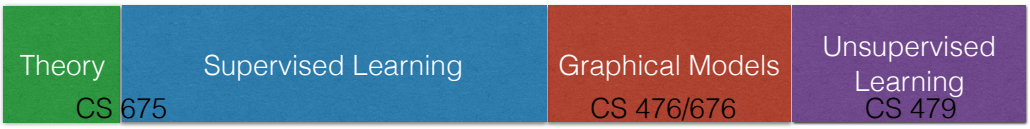
- **Unsupervised learning**

  Density estimation, Clustering, Dimensionality reduction

- Semi-supervised learning
- Active learning
- Reinforcement learning
- Many more …

About the course

- Machine Learning **Algorithms and Principles**



  - Classification: kNN, naive Bayes, logistic regression, perceptrons, large margin, kernels, support vector machines
  - Regression: Linear regression, Gaussian Process, Kernel regression
  - Kernel density estimation, Hidden Markov Models, Graphical Models, k-means clustering, dimensionality reduction, neural networks, deep belief networks, Boosting, Decision Trees, etc.
  - Optimization, Theory, Model selection, overfitting, bias-variance tradeoffs
- **It's going to be fun and hard work** ☺
- See **tentative** lecture outline on Piazza – MAY CHANGE
- Material: Class slides + Reading material

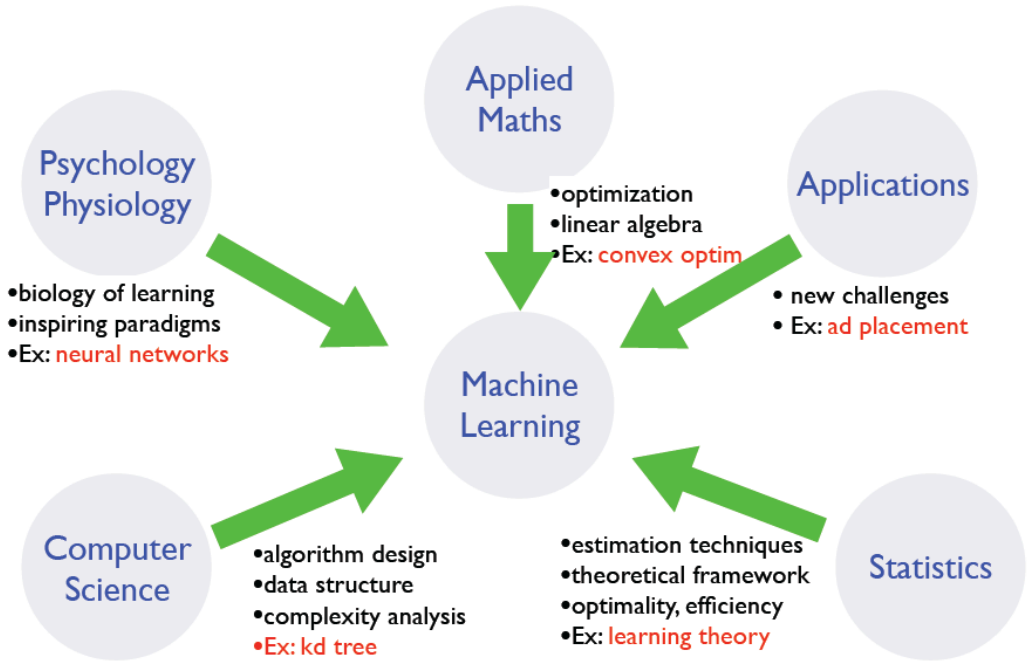
ML vs other "data sciences"

- **Data mining:** emphasis on analysis and producing output (e.g. data summary) for human consumption (e.g., visualization)
- **Statistics:**
  - more focused on explaining rather than predicting future outcomes;
  - more emphasis on management of uncertainty (e.g., confidence intervals) than loss reduction;
  - more emphasis on theoretical results than empirical performance on benchmarks;
  - more emphasis on small data regime, and less focus on computational issues
- **Data science:** combination of statistics, data mining, ML and engineering for 'data wrangling"

ML and computer science

- Emphasis on fully automatic methods
- Increasing focus on computational efficiency as typical available data sets increase in size
  - 10^6 to > 10^{12} examples
  - > 10^6 dimensions
  - expectation of real time prediction
- Connections to foundations of computer science, information theory, and communication
  - learnability
  - data complexity vs computational complexity
  - learning and compression

Where does ML fit in?



Slide Credit: Fei Sha

Frameworks for Learning

- Function Approximation
- Probabilistic Approach
- Information Theoretic Approach

Theoretical Questions

- How many examples do we need to learn?
- How do we quantify our ability to generalize to unseen data?
- Which algorithms are better suited to specific learning settings?

Prerequisites

Math pre-requisites

Linear algebra (vector spaces, orthogonality, singular value decomposition)
Multivariate calculus (partial derivative, gradient, Hessian, Jacobian)
Probability and Statistics (random variables, probability distributions, expectations, mean, variance, covariance, conditional probability, law of large numbers, Bayes rule, MLE)

CS pre-requisites

Algorithms (Dynamic programming, basic data structures, complexity…)
Programming (Mostly your choice of language; Python will be very useful)

General requirement

Ability to deal with "**abstract mathematical concepts**"

We provide some background, but the class will be fast paced

Machine Learning Class webpage

- http://www.cs475.org
- Email: cs475@cs.jhu.edu

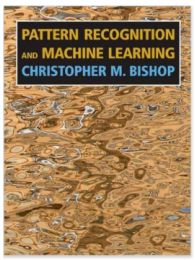
Recitations

- Strongly recommended
  - Brush up pre-requisites
  - Review material (difficult topics, clear misunderstandings, extra new topics)
  - Ask questions related to homework, projects, midterm

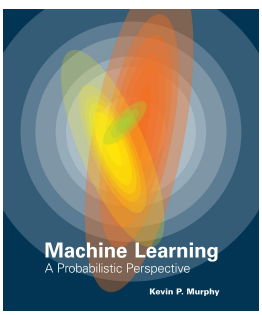
- Friday, Feb 3: Probability & Statistics
- Friday, Feb 10: Linear Algebra
- Friday, Feb 10: Python, scikit-learn

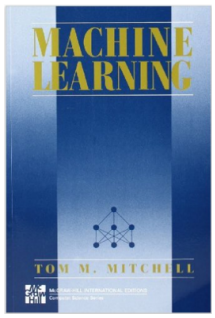
Textbooks

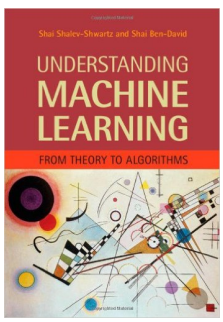
- Recommended Text: Christopher M. Bishop. Pattern recognition and machine learning. 2009

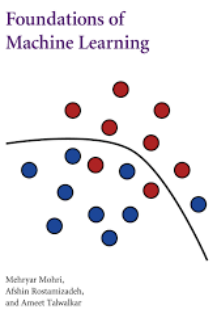


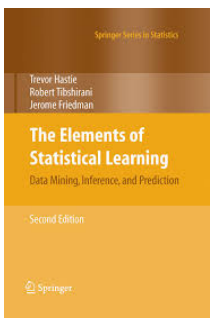
- Reference books





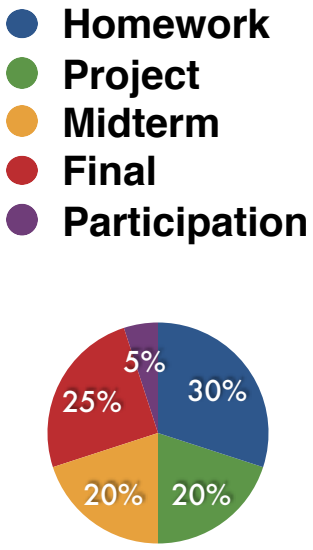






Grading

- 5-6 Homework assignments (30%)
  - First one already out (watch email)
  - Start early, ask questions, discuss
- Three short course projects (20%)
  - Recommend teams of 3-4
  - In-class kaggle competitions
- Midterm (20%)
  - Friday, March 17 in class
- Final exam (25%)
  - Saturday, May 13, 2-5PM
- In-class participation (5%)



Homeworks

- Homework are hard, start early ☺
- Due in the beginning of class
- Submit electronically to Gradescope: https://gradescope.com
- 72 late hours for the semester
- After late days are used up:
  - Half credit within 48 hours
  - Zero credit after 48 hours
- At least 4 homework assignemnts **must be handed in**

Homeworks

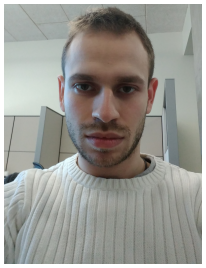
- Collaboration
  - You may **discuss** the questions
  - Each student writes their own answers
  - Each student must write their own code for the programming part
  - **Please don't search for answers on the web, Google, previous years' homework, etc.**
    - please ask us if you are not sure if you can use a particular reference
  - Post questions on Piazza. We will not respond to emails.

Teaching/Course Assistants


Pushpendre Rastogi


Akshay Rangamani


Poorya Mianjy


Teodor V. Marinov


Eric W. Bridgeford


Alex Ahn

Enjoy!

- ML is becoming ubiquitous in science, engineering and beyond
- This class should give you the basic foundation for applying ML and developing new methods
- The fun begins…

What is Machine Learning? (Formally)

What is Machine Learning?

Study of algorithms that

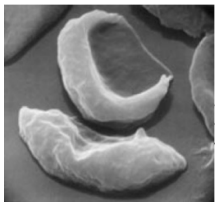
- improve their performance
- at some task
- with experience

Data (experience) ⇒  (task) ⇒ Understanding (performance)

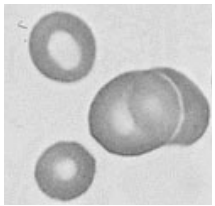
Supervised Learning Task

Task: Given $X \in \mathcal{X}$, predict $Y \in \mathcal{Y}$. X - test data

$\equiv$ Construct **prediction rule** $f : \mathcal{X} \to \mathcal{Y}$



⇒ "Anemic cell (0)"

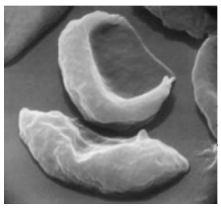


⇒ "Healthy cell (1)"

Performance Measures

Performance:

$\text{loss}(Y, f(X))$- Measure of closeness between true label *Y* and prediction *f*(*X*)

X	*Y*	*f*(*X*)	$\text{loss}(Y, f(X))$
	"Anemic cell"	"Anemic cell"	0
		"Healthy cell"	1

$$\text{loss}(Y, f(X)) = 1_{\{f(X) \neq Y\}}$$ **0/1 loss**

Performance Measures

Performance:

$\text{loss}(Y, f(X))$- Measure of closeness between true label *Y* and prediction *f*(*X*)

X	Share price, *Y*	*f*(*X*)	$\text{loss}(Y, f(X))$
Past performance, trade volume etc. as of Sept 8, 2010	"$24.50"	"$24.50"	0
		"$26.00"	1?
		"$26.10"	2?

$$\text{loss}(Y, f(X)) = (f(X) - Y)^2$$ **square loss**

Performance Measures

Performance:

$\text{loss}(Y, f(X))$- Measure of closeness between true label *Y* and prediction *f*(*X*)

Don't just want label of one test data (cell image), but any cell image $X \in \mathcal{X}$

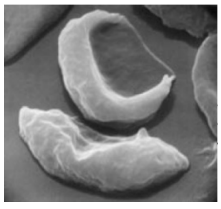
$$(X, Y) \sim P_{XY}$$

Given a cell image drawn randomly from the collection of all cell images, how well does the predictor perform on average?

$$\text{Risk } R(f) \equiv \mathbb{E}_{XY}\left[\text{loss}(Y, f(X))\right]$$

Performance Measures

Performance: $\text{Risk } R(f) \equiv \mathbb{E}_{XY}\left[\text{loss}(Y, f(X))\right]$



⇨ "Anemic cell"

⇨ Share Price "$ 24.50"

$\text{loss}(Y, f(X))$	Risk $R(f)$
$1_{\{f(X) \neq Y\}}$	$P(f(X) \neq Y)$
0/1 loss	**Probability of Error**
$(f(X) - Y)^2$	$\mathbb{E}[(f(X) - Y)^2]$
square loss	**Mean Square Error**

Bayes Optimal Rule

Ideal goal: Construct **prediction rule** $f^* : \mathcal{X} \to \mathcal{Y}$

$$f^* = \arg\min_f \mathbb{E}_{XY}\left[\text{loss}(Y, f(X))\right]$$

Bayes optimal rule

Best possible performance:

Bayes Risk $\quad R(f^*) \leq R(f)$ for all f

BUT… Optimal rule is not computable - depends on unknown P_{XY} !

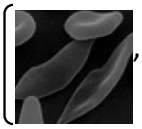
Experience - Training Data

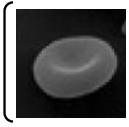
Can't minimize risk since P_{XY} unknown!

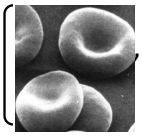
Training data (experience) provides a glimpse of P_{XY}

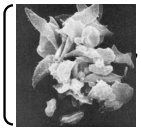
(observed) $\{(X_i, Y_i)\}_{i=1}^n \overset{i.i.d.}{\sim} P_{XY}$ **(unknown)**

independent, identically distributed

 , Anemic cell

 , Healthy cell

 , Healthy cell

 , Anemic cell

Provided by expert, measuring device, some experiment, …

Supervised Learning

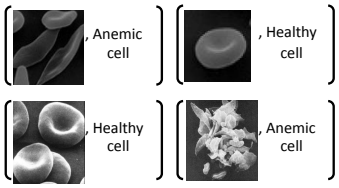
Task: Given $X \in \mathcal{X}$, predict $Y \in \mathcal{Y}$.

$\equiv$ Construct **prediction rule** $f : \mathcal{X} \to \mathcal{Y}$

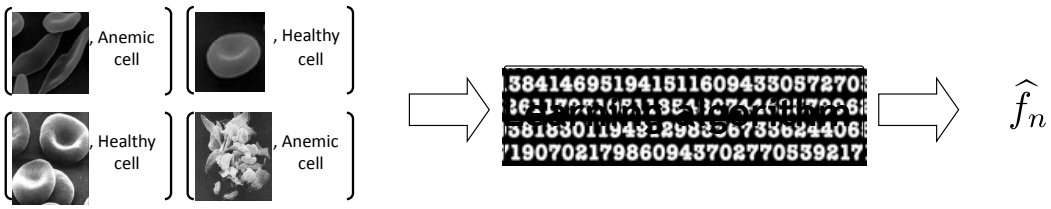
Performance: Risk $R(f) \equiv \mathbb{E}_{XY}\left[\text{loss}(Y, f(X))\right]$

$(X, Y) \sim P_{XY}$

Experience: Training data $\{(X_i, Y_i)\}_{i=1}^n \overset{i.i.d.}{\sim} P_{XY}$ **(unknown)**

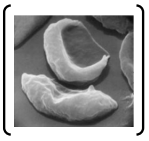


Machine Learning Algorithm



Training data $\{(X_i, Y_i)\}_{i=1}^n$

$\widehat{f}_n$ is a mapping from $\mathcal{X} \to \mathcal{Y}$



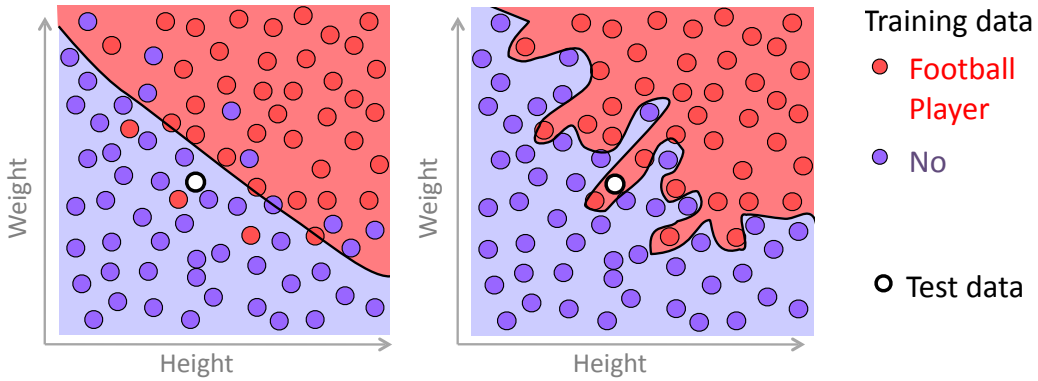
$\widehat{f}_n$ = "Anemic cell"

Test data X

Note: test data ≠ training data

Issues in ML

- A good machine learning algorithm
  - Does not **overfit** training data



  - **Generalizes** well to test data

More later …

Performance Revisited

Performance: (of a learning algorithm)

How well does the algorithm do on average

1. for a test cell image X drawn at random, and
2. for a set of training images and labels $D_n = \{(X_i, Y_i)\}_{i=1}^n$ drawn at random

Expected Risk (aka Generalization Error)

$$\mathbb{E}_{D_n}\left[R(\hat{f}_n)\right] \equiv \mathbb{E}_{D_n}\left[\mathbb{E}_{XY}\left[\text{loss}(Y, \hat{f}_n(X))\right]\right]$$

Supervised Learning

Ideal goal: Construct prediction rule $f^* : \mathcal{X} \to \mathcal{Y}$

$$f^* = \arg\min_f \mathbb{E}_{XY}\left[\text{loss}(Y, f(X))\right]$$

Bayes optimal rule

Practical goal: Given $\{X_i, Y_i\}_{i=1}^n$, **learn** prediction rule $\widehat{f}_n : \mathcal{X} \to \mathcal{Y}$

Often: $$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \left[\text{loss}(Y_i, f(X_i))\right]$$

Empirical Risk minimizer

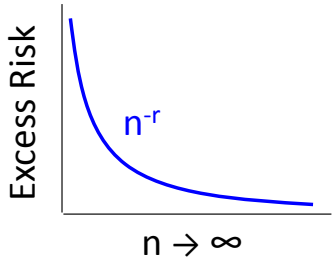
$$\frac{1}{n}\sum_{i=1}^{n} \left[\text{loss}(Y_i, f(X_i))\right] \xrightarrow[\text{Numbers}]{\text{Law of Large}} \mathbb{E}_{XY}\left[\text{loss}(Y, f(X))\right]$$

Consistency and Rate of Convergence

- How does the performance of the algorithm compare with ideal performance?

Excess Risk $$\mathbb{E}_{D_n}\left[R(\widehat{f}_n)\right] - R(f^*)$$

- **Consistent** algorithm if Excess Risk → 0 as n → ∞
- **Rate of Convergence**



More later …

How to sense Generalization Error?

- Can't compute generalization error. How can we get a sense of how well algorithm is performing in practice?
- One approach -
  - Split available data into two sets $\{(X_i, Y_i)\}_{i=1}^n \{(X_i', Y_i')\}_{i=1}^n$
  - Training Data – used for training the algorithm, E.g.

$$\{(X_i, Y_i)\}_{i=1}^n \Rightarrow \boxed{\text{learning algorithm}} \Rightarrow \hat{f}_n$$

  - Test Data (a.k.a. Validation Data, Hold-out Data)

Test Error = $\frac{1}{n}\sum_{i=1}^{n}\left[\text{loss}(Y_i', \hat{f}_n(X_i'))\right]$ **= Estimate of Generalization Error**

Choices in ML Formulation

Often, the same task can be formulated in more than one way:

- Ex. 1: Loan applications
  - creditworthiness/score (regression)
  - probability of default (density estimation)
  - loan decision (classification)
- Ex. 2: Chess
  - Nature of available training examples/experience:
    - expert advice (painful to experts)
    - games against experts (less painful but limited, and not much control)
    - experts' games (almost unlimited, but only "found data" – no control)
    - games against self (unlimited, flexible, but can you learn this way?)
  - Choice of target function: board -> move vs. board -> score

How to approach a Machine Learning Problem

1. Consider your goal -> definition of task **T**
   – E.g. make good loan decisions, win chess competitions, …

2. Consider the nature of available (or potential) experience **E**
   – How much data can you get? What would it cost (in money, time or effort)?

3. Choose type of output **O** to learn
   – (Numerical? Category? Probability? Plan?)

4. Choose the Performance measure **P** (error/loss function)

5. Choose a representation for the input **X**

6. Choose a set of possible solutions **H** (hypothesis space)
   – set of functions h: X -> O
   – (often, by choosing a representation for them)

7. Choose or design a learning algorithm
   – for using examples (E) to converge on a member of H that optimizes P