STA 314: Statistical Methods for Machine Learning I

Lecture 1 - Introduction

Xin Bing

Department of Statistical Sciences
University of Toronto

About this course

- This course is a broad introduction to machine learning from a statistical perspective (aka statistical learning). We put emphasis on intuition and basic mathematical derivations of how (and why) popular machine learning methods work.

- We will focus on understanding methodology rather than implementing complicated machine learning algorithms or delving into deep theory.

- You will learn examples of applying popular machine learning methods to real data sets in R.

About this course

We cover two types of learning problems:

- Supervised learning
  - Regression
  - Classification
- Unsupervised learning
  - Dimension reduction
  - Clustering
  - Matrix factorization
- This includes a variety of important methods:
  - linear regression, logistic regression, non-parametric regression, nearest neighbours, decision trees, bagging, boosting, random forests, SVMs
  - PCA, K-means, matrix completion, topic modeling

Do I have the appropriate background?

Coursework is aimed at advanced undergrads and graduate students. We will use multivariate calculus, probability, and linear algebra.

- Linear algebra: vector/matrix operations such as eigenvalues and eigenvectors, eigen and singular value decompositions, inverse, trace, norms.
- Calculus: partial derivatives/gradient.
- Probability & Statistics: expectation, variance, covariance; Bayes' theorem; common distributions; maximum likelihood estimation, simple linear regression, point and interval estimation, hypothesis testing, p-values.

Do I have the appropriate background?

- Programming language: we are using R in this course.
  - Useful resources: `https://cran.r-project.org/`. A good review of some basic R commands is in Chapter 2.3 of the textbook.
  - How much do you need to know?
    - Basic knowledge on R is required (e.g., load data, create a vector or matrix, etc.)
    - The tutorials will provide you demonstrations of using R.
    - The emphasis of coding will be on the use of the various R packages and on the implementation of the key subroutines of ML methods.
    - You will not be required to **implement complicated machine learning algorithms** nor to **write an entire R package**.

Textbook and other suggested readings

We mainly use the textbook

- Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*.
- You can find it via `https://www.statlearning.com`.

You are only responsible for the material covered in lectures, tutorials, and homeworks. You may also find the following references useful throughout the course:

- Hastie, Tibshirani, and Friedman. *The Elements of Statistical Learning*.
- Christopher Bishop. *Pattern Recognition and Machine Learning*.
- Kevin Murphy. *Machine Learning: a Probabilistic Perspective*.

There are lots of other freely available, high-quality ML resources.

Course Information

Quercus: the main source of course information.

- Syllabus
- Announcements, slides & tutorial notes
- Homework releasing & submission, grades

Piazza: the main place for discussions

- Sign up: `https://piazza.com/utoronto.ca/fall2022/sta314`
- Your grade does not depend on your participation on Piazza. It's a good way for asking questions, discussing with the course community.
- Mainly discuss **non-technical** questions on course materials/assignments/exams.
- **Technical** questions are encoured to ask during office hours.

Email policy

- Please refrain from sending emails except for urgent requests. For such urgent emails, I aim to reply within 24 hours.

- For non-urgent questions / requests,
  - if it is technical and related with the course material including homeworks and exams, you are encouraged to use the office hours first;
  - otherwise you are encouraged to use Piazza for communication with the instructors either privately or publicly.

Course delivery instructions

- All sections (LEC0101 and LEC0201) have the same delivery layout.

Weekly	delivery mode
2-hour lecture	in-person
1-hour tutorial	in-person
4-hour office hour (1 + 3)	zoom / in-person

- Tutorials are not required but highly recommended as they contain supplementary materials to the lectures. Some weeks might not have tutorials.
- While cell phones and other electronics are not prohibited in lecture, talking, recording or taking pictures in class is strictly prohibited without the consent of your instructor. Please ask before doing!

Course Information

All information is in the syllabus on Quercus.
If you remember just one thing:

Check Quercus regularly.

Grading

- (60%) 4 assignments
  - Combination of pen & paper derivations and light-weight programming exercises.
  - Weighted equally.
  - Hand-in on Quercus.
- (20%) Midterm test
  - 2-hour held during normal class time
  - See the syllabus for exact time and date.
  - Taken in the same classroom.
- (20%) Final test
  - 2-hour held during the final assessment period
  - Date, time and location are to be annouced

More on Assignments

Students are required to work on the assignments and submit their handouts alone. Discussion with instructors and other students is allowed. If you choose to do so, then you:

- Must include a statement in your submission that includes the name of the student that you discussed with and what part of your submission is involved.
- Must not share proofs, pseudocode, code, or simulation results.
- Must do your own work.

Violation of this policy is an academic offence and will be investigated and reported as such.

Assignments should be handed in by deadline; a late penalty of 10% of the total credit of the assignment per day will be assessed thereafter (up to 3 days, then submission is blocked).

Extensions will be granted only in special situations, and you will need to complete an absence declaration form and notify us to request special consideration, or otherwise have a written request approved by the course instructors at most one week after the due date.

Similar course: CSC311–Introduction to Machine Learning

- **STA314 takes a more statistical perspective than CSC311** while their core contents share the same machine learning methods.
  - The course will focus on the methodology and statistical insight rather than algorithm (or coding).
  - We do not cover neural networks nor reinforcement learning.
  - We will cover model selection, high-dimensional statistics, bootstrap, topic models.

Advanced Courses

This course will help prepare you for the following courses.

- **STA414** (Statistical Methods for Machine Learning II)
  - This course is the follow-up course, which delves deeper into the probabilistic interpretation of machine learning.
- **CSC413** (Neural Networks and Deep Learning)
  - This course covers deep learning and automatic differentiation.
- **CSC412** (Probabilistic Learning and Reasoning)
  - The CSC analogue of STA414.

Questions?

What is learning?

"The activity or process of gaining knowledge or skill by studying, practicing, being taught, or experiencing something."

Merriam Webster dictionary

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Tom Mitchell

What is machine learning?

- For many problems, it's difficult to program the correct behavior by hand
  - recognizing people and objects
  - understanding human speech
- Machine learning approach: program an algorithm to automatically learn from data, or from experience
- Why might you want to use a learning algorithm?
  - hard to code up a solution by hand (e.g. vision, speech)
  - system needs to adapt to a changing environment (e.g. spam detection)
  - want the system to perform *better* than the human programmers
  - privacy/fairness (e.g. ranking search results)

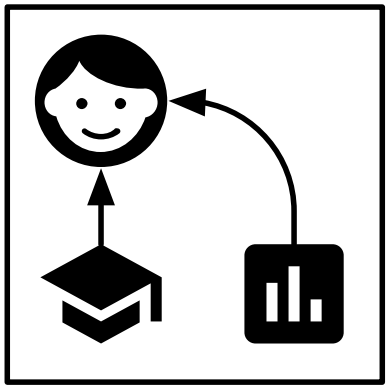
Relations to AI

- Nowadays, "machine learning" is often brought up with "artificial intelligence" (AI)

- AI does not always imply a learning based system
  - Symbolic reasoning
  - Rule based system
  - Tree search
  - etc.

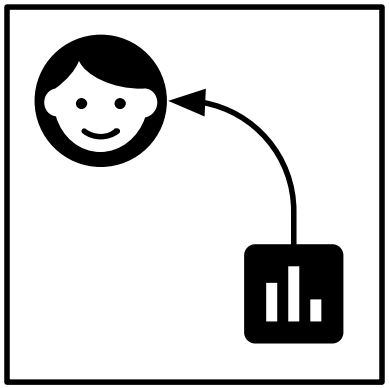
Types of machine learning problems

Supervised Learning



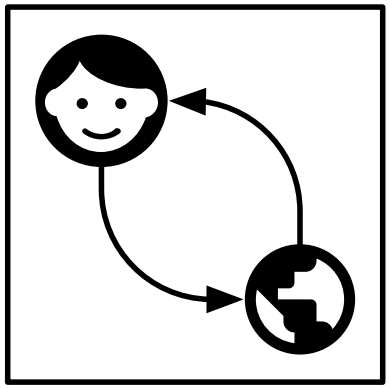
Machine is given data and examples of what to predict.

Unsupervised Learning



Machine is given data, but not what to predict.

Reinforcement Learning



Machine gets data by interacting with an environment and tries to minimize a cost.

Computer vision: Object detection, semantic segmentation, pose estimation, and almost every other task is done with ML.

DAQUAR 1553

What is there in front of the sofa?

Ground truth: table

IMG+BOW: table (0.74)

2-VIS+BLSTM: table (0.88)

LSTM: chair (0.47)

COCOQA 5078

How many leftover donuts is the red bicycle holding?

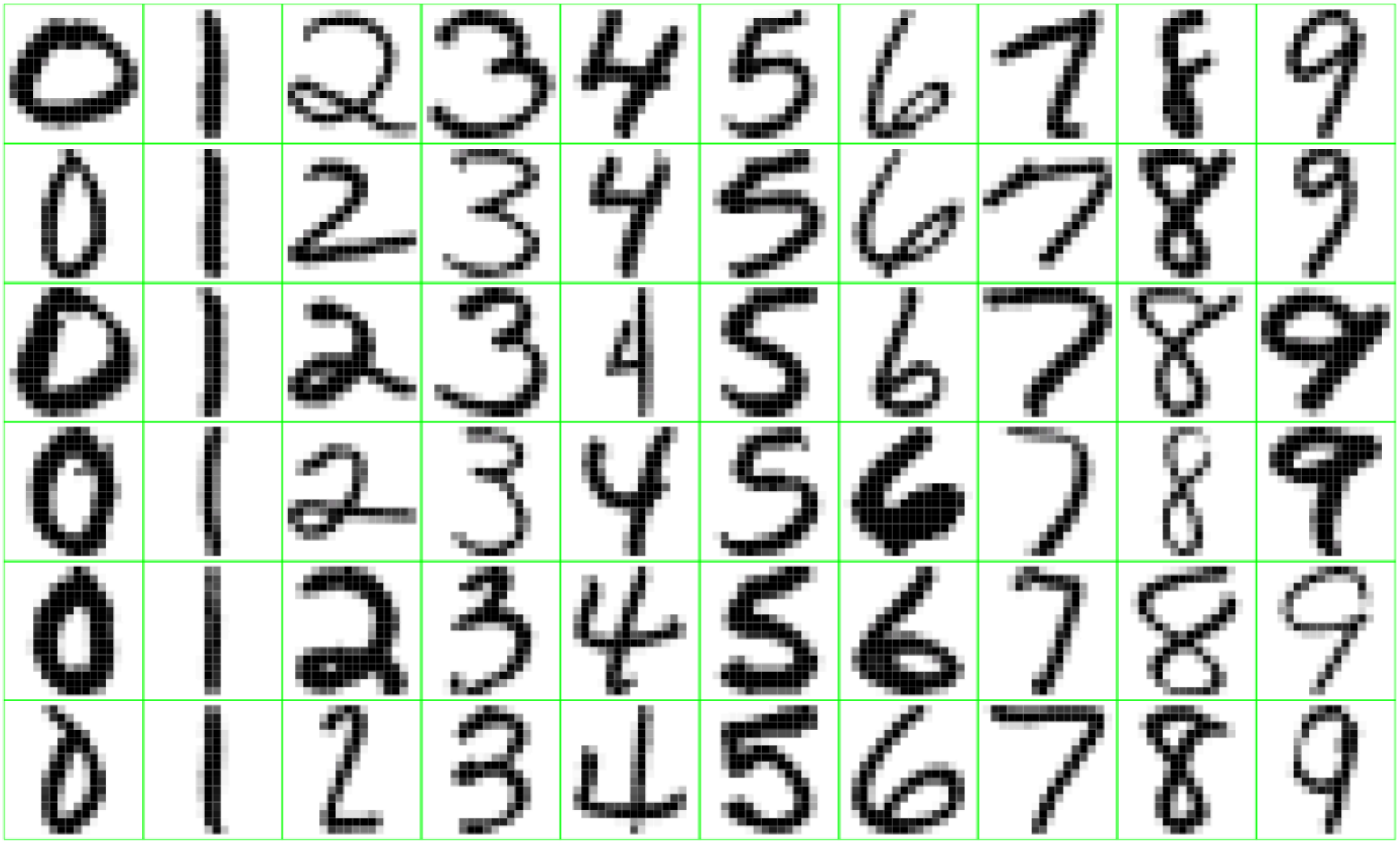
Ground truth: three

IMG+BOW: two (0.51)

2-VIS+BLSTM: three (0.27)

BOW: one (0.29)

Detect numbers in a handwritten zip code



Speech: speech to text, personal assistants, speaker identification...

Natural language processing: machine translation, sentiment analysis, topic modelling.

Real world example: The New York Times

LDA analysis of 1.8M New York Times articles:

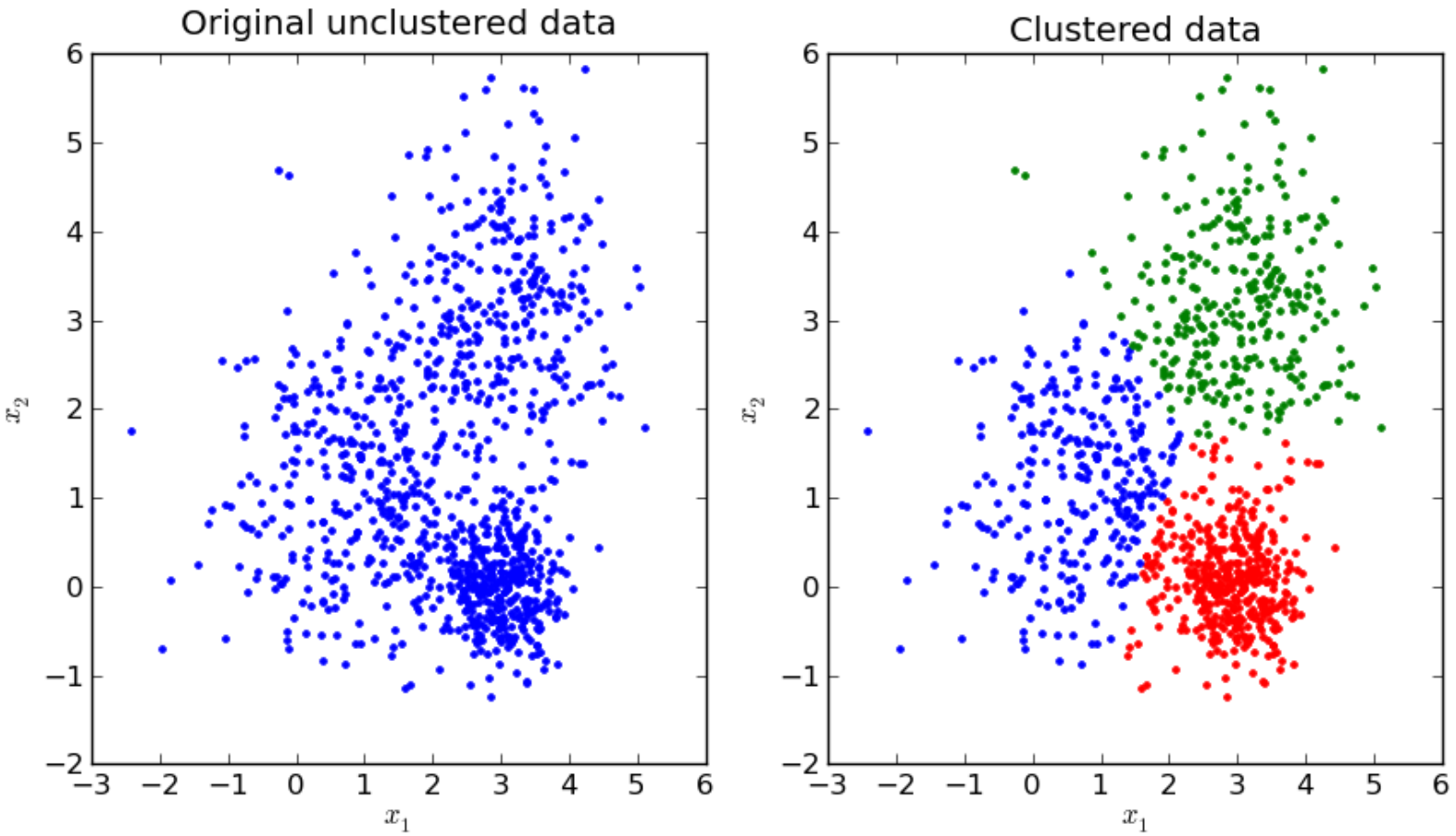
music band songs rock album jazz pop song singer night	book life novel story books man stories love children family	art museum show exhibition artist artists paintings painting century works	game knicks nets points team season play games night coach	show film television movie series says life man character know
theater play production show stage street broadway director musical directed	clinton bush campaign gore political republican dole presidential senator house	stock market percent fund investors funds companies stocks investment trading	restaurant sauce menu food dishes street dining dinner chicken served	budget tax governor county mayor billion taxes plan legislature fiscal

Spam emails detection

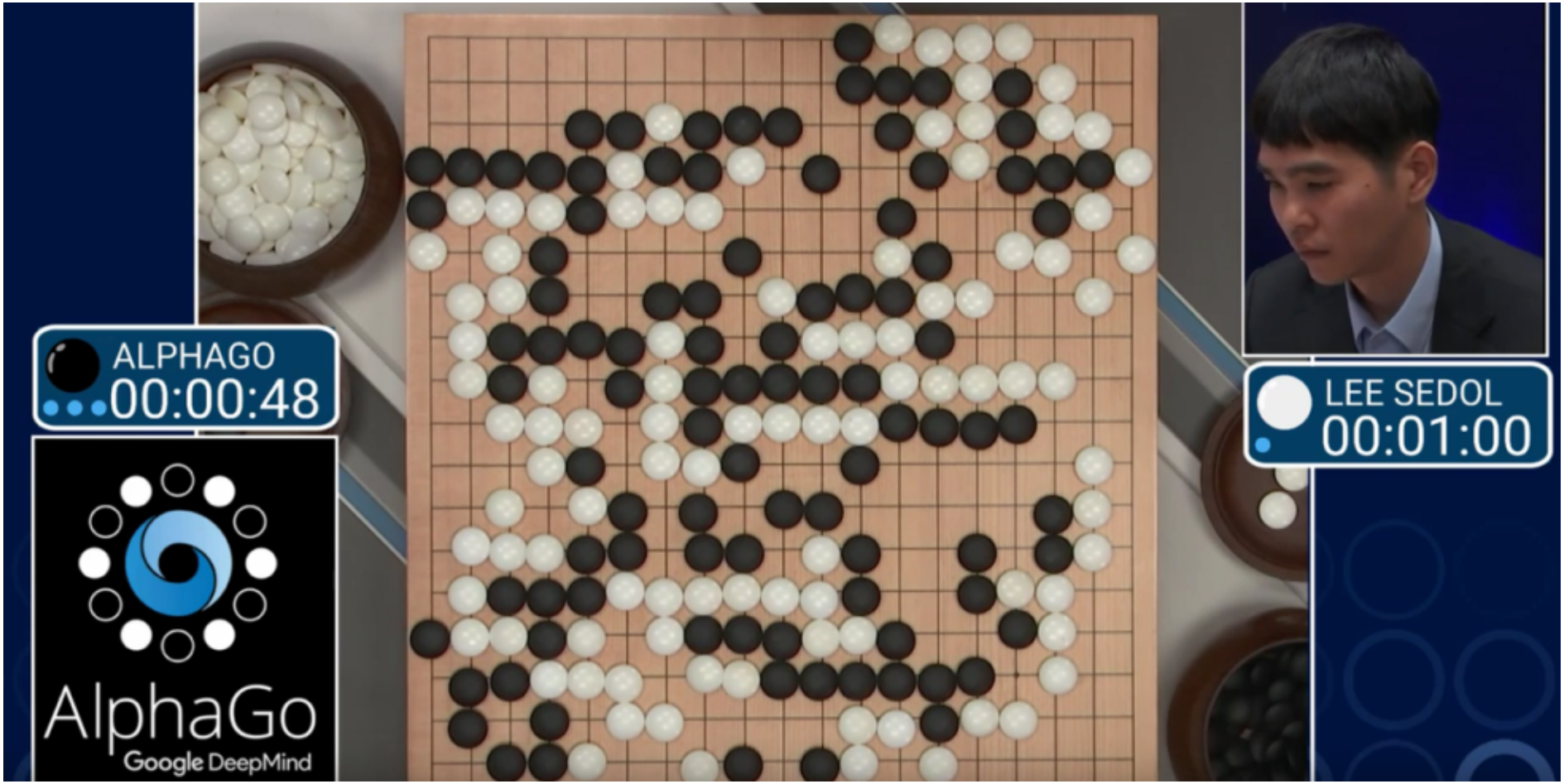
- data from 4601 emails sent to an individual (named George, at HP labs). Each is labeled as **spam** or **email**.
- goal: build a customized spam filter
- input features: relative frequency of 57 words and punctuation marks in the email message

	george	you	hp	free	!	edu	remove
spam	0.00	2.26	0.02	0.54	0.51	0.01	0.28
email	1.27	1.27	0.90	0.07	0.11	0.29	0.01

Subgroup detection / clustering



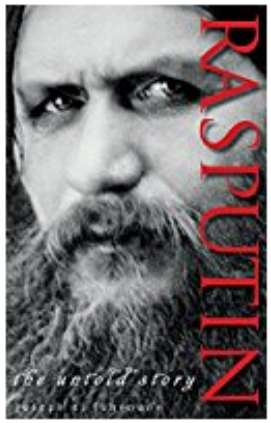
Playing Games

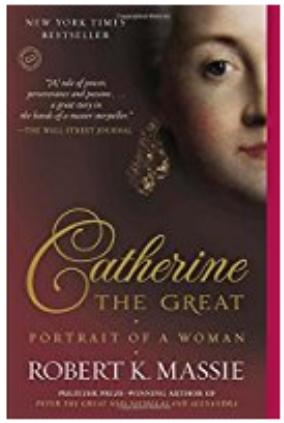


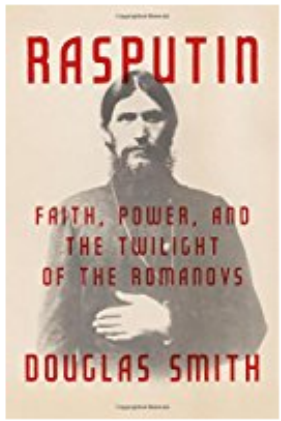
DOTA2 - ▸ Link

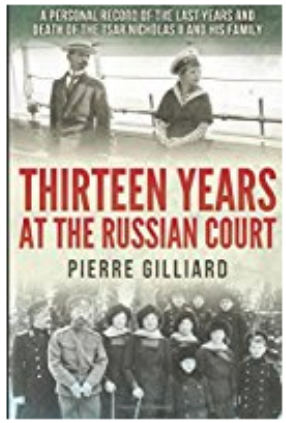
Recommender Systems : Amazon, Netflix, ...

Inspired by your shopping trends

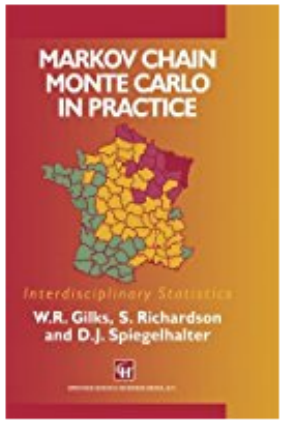


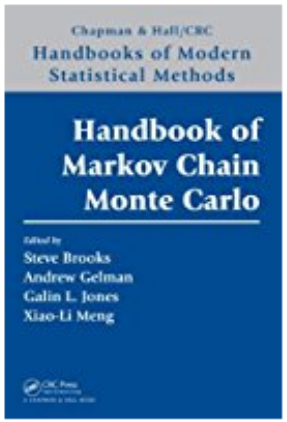


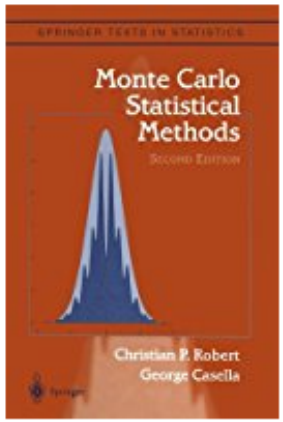


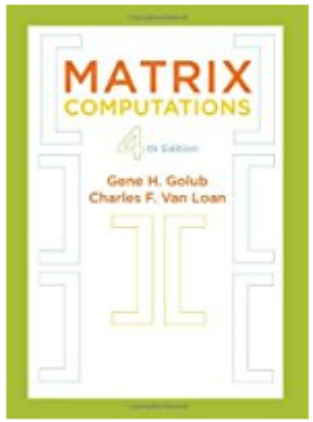


Related to items you've viewed See more









Statistical Learning versus Machine Learning

- Machine Learning (ML) is a subfield of AI while Statistical Learning (SL) is a subfield of Statistics.
- They both try to uncover patterns in data.
- Both fields draw heavily on calculus, probability, and linear algebra, and share many of the same core algorithms
- ML puts more emphasis on algorithms and prediction accuracy while SL emphasizes more on models and their interpretability, and how to evaluate uncertainty of the learning procedure.
- This course focuses on **Statistical Learning**.

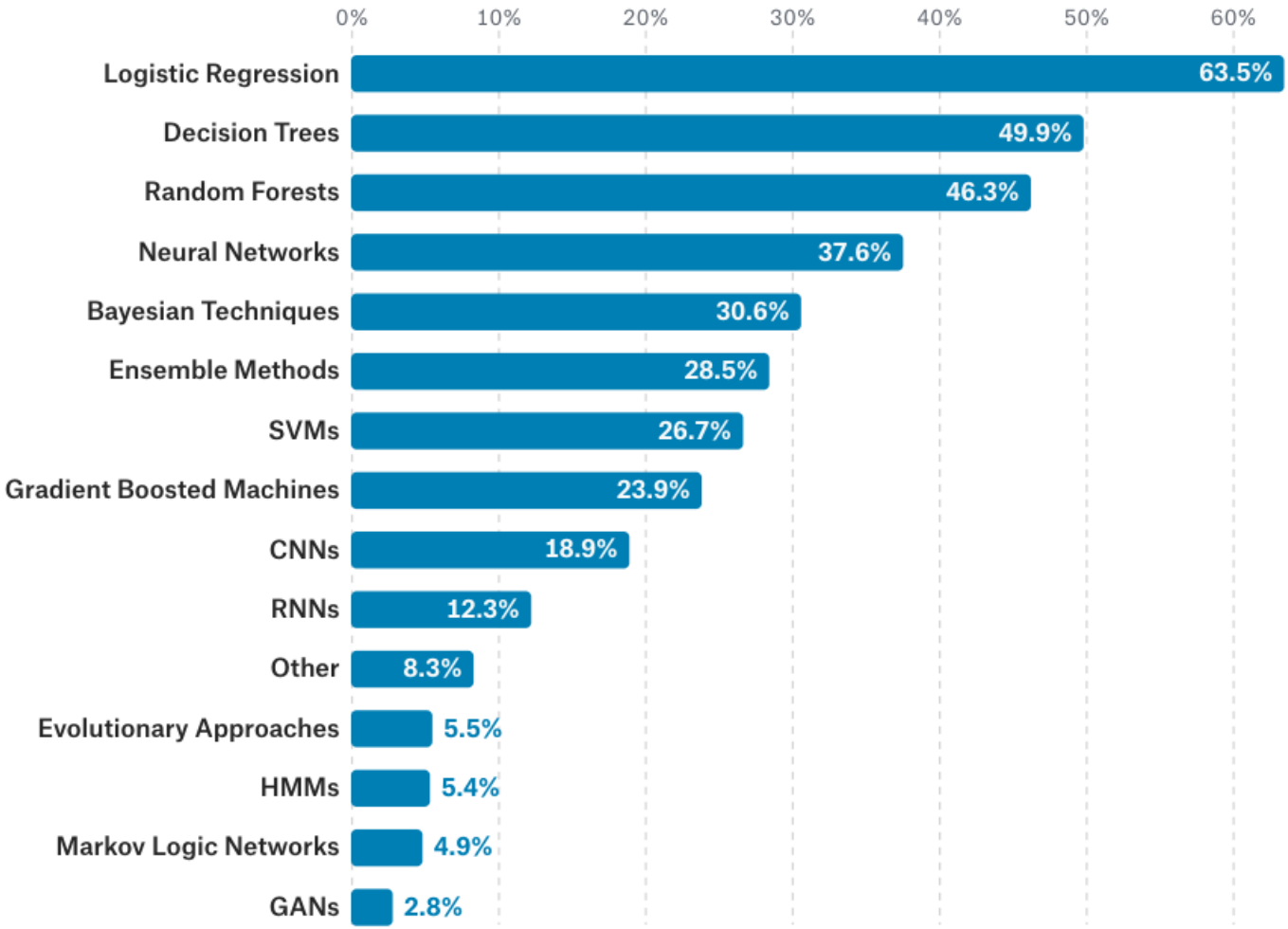
Why this class?

"I've heard that neural networks solve everything, can we just learn those?"

- There's a whole world of problems where neural nets do not work.
- The techniques in this course are still the first things to try for a new ML problem.
  - E.g., try logistic regression before building a deep neural net!
- The principles you learn in this course will be essential to understand and apply neural nets.

Why this class?

2017 Kaggle survey of data science and ML practitioners: what data science methods do you use at work?



Why this class?

- It is important to understand the ideas behind the various techniques, in order to know how and when to use them.
- Advanced algorithms are built on the simpler ones.
- It is important to accurately assess the performance of a method, to know how well or how badly it works or will work
- This is an exciting research area, having important applications in science, industry and finance.
- Statistical learning is a fundamental ingredient in the training of a modern data scientist.

Introduction to Statistical Machine Learning

Supervised Learning

- Outcome measurement Y (also called dependent variable, response, target).
- Vector of p predictor measurements X (also called inputs, regressors, covariates, features, independent variables).
- In **regression** problems, Y is quantitative (e.g price, blood pressure).
- In **classification** problems, Y takes values in a finite, unordered set (survived/died, digit 0-9, cancer class of tissue sample).
- We have training data $(x_1, y_1), ..., (x_n, y_n)$. These are observations (instances, realizations) of the measurement (X, Y).

Main tasks in Supervised Learning

On the basis of the training data we would like to:

- Prediction: accurately predict future outcome (Y).
- Estimation: understand how features (X) affect the outcome (Y).
- Model selection: find the best model for the outcome (Y) or which features (X) affect the outcome (Y).
- Inference: assess the quality of our prediction, estimation and model selection.

Unsupervised Learning

No outcome variable Y, just a set of features X measured on a set of samples.

- objective is more fuzzy – find groups of samples that behave similarly, find features that behave similarly, find linear combinations of features with the most variation.
- difficulty in model assessment: difficult to quantify how well you are doing.
- different from supervised learning, but can be useful as a pre-processing step for supervised learning.

Introduction to Supervised Learning

- Today (and for much of this course) we focus on supervised learning.
- For many tasks of interest we are given some data consisting of features X and outcome Y, and we are interested in predicting some unseen Y by using its corresponding X.

Task	Input data (X, Y)	Outcome (Y) to predict
object recognition	image + object	object category
image captioning	image + caption	caption
document classification	text	document category
speech-to-text	audio waveform	text
⋮	⋮	⋮

- Supervised learning is applicable when we have outcome instances, i.e., we can supervise the learner by telling it exactly what to predict.

Introduction to Supervised Learning

More precisely, in supervised learning, we are given

- training set consisting of n samples of
  - features X and corresponding
  - outcome Y.

Our goal is to learn the underlying generating mechanism from $X \rightarrow Y$.

Introduction to Supervised Learning

Mathematically, write the underlying generating mechanism between Y and X as

$$Y = f(X) + \epsilon$$

where ϵ captures measurement errors and other discrepancies.

We are given the training data $\mathcal{D}^{train}$ consisting of n i.i.d. samples of (X, Y) following the above model, that is, $(x_1, y_1), \ldots, (x_n, y_n)$.

Our goal is to estimate / learn the mapping / function f based on $\mathcal{D}^{train}$.

Why estimate f?

- Prediction: Given a new point $X = x$, $f(x)$ is typically a good prediction, and it is in fact the best prediction one can hope for with respect to the mean squared error.

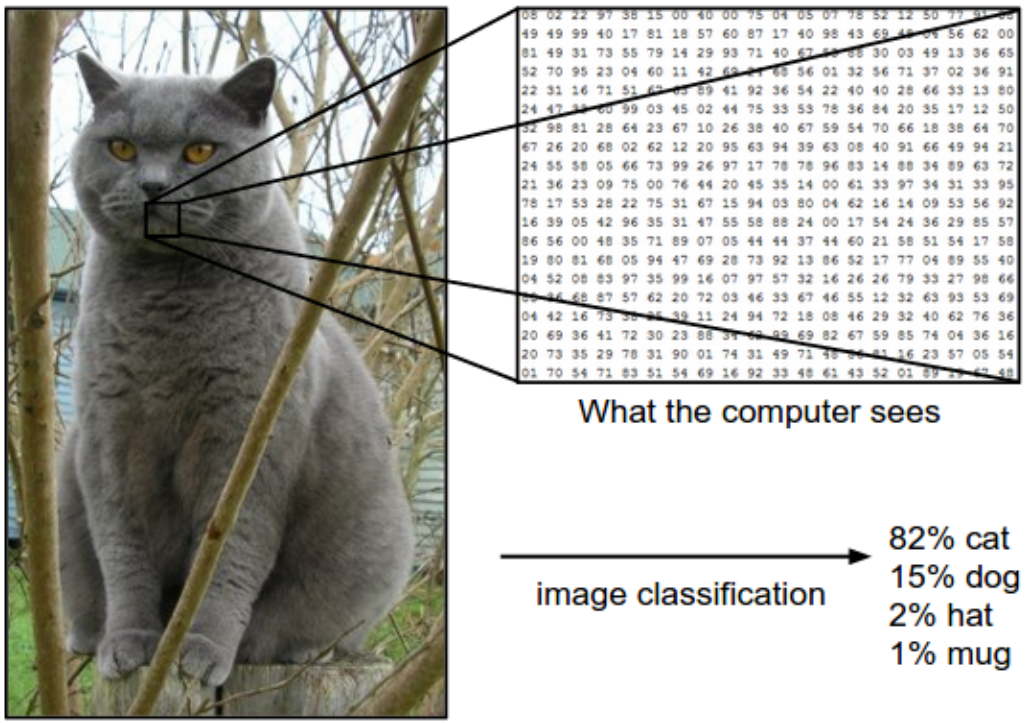
- Inference:
  - We can understand which components of $X = (X_1, X_2, ..., X_p)$ are important in explaining Y, and which are irrelevant.
  - Depending on the complexity of f, we may be able to understand how each component X_j of X affects Y.

Example 1: image recognition

For image data, we observe n images with anotated labels

$$y_i \in \{\text{cat, dog, hat, mug}\}, \qquad 1 \le i \le n.$$

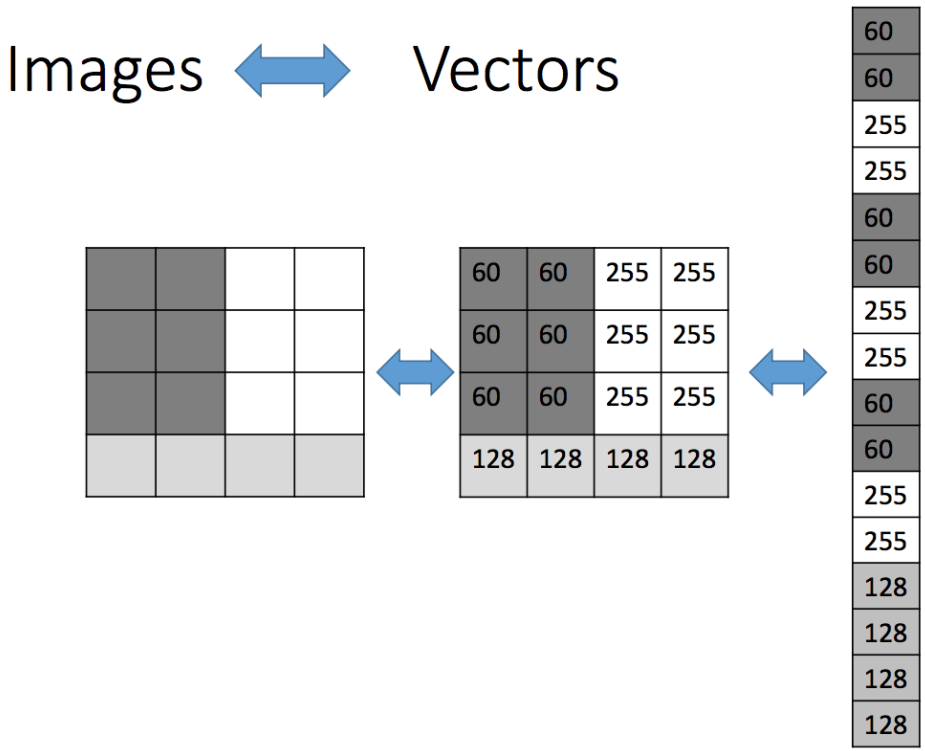
What an image looks like to the computer:



[Image credit: Andrej Karpathy]

Example 1: image recognition

Predictors x_i are represented as a vector



The goal is to let the machine learn the function $f : x_i \rightarrow y_i$ to predict the labels for unanotated images.

Example 2: Sales prediction and inference

We are given information of n products: each of them consists

- y_i: **Sales** of product i in 200 different markets
- x_{i1}: **TV** budget of product i
- x_{i2}: **radio** budget of product i
- x_{i3}: **newspaper** budget of product i

Suppose that the true generating model is

$$Y = f(X) + \epsilon = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \epsilon.$$

Knowing f helps us understand how **Sales** *changes if one increases one unit of* **TV** *budget* (X_1).

How to estimate f?

Notation:

- Let x_{ij} denote the value of the jth feature for observation i, where $i = 1, 2, ..., n$ and $j = 1, 2, ..., p$.
- Let y_i denote the response variable for the ith observation.
- Training data consist of $\mathcal{D}^{train} = \{(x_1, y_1), ..., (x_n, y_n)\}$, where $x_i = (x_{i1}, ..., x_{ip})^T$.

We would like to estimate f based on the training data.

Two different approaches:

- **Parametric method**
- **Non-parametric method**

Parametric method

Assume parametric form of f!

Example (Linear model & predictor)

The linear model is an important example of a parametric model:

$$Y = f(X) + \epsilon, \quad \text{with} \quad f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p.$$

Correspondingly, we would estimate f by

$$\hat{f}_{linear}(X) = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p.$$

Remark: constructing $\hat{f}_{linear}$ reduces the problem of estimating a function to that of estimating $(p+1)$ parameters $(\beta_0, \beta_1, \ldots, \beta_p)$.

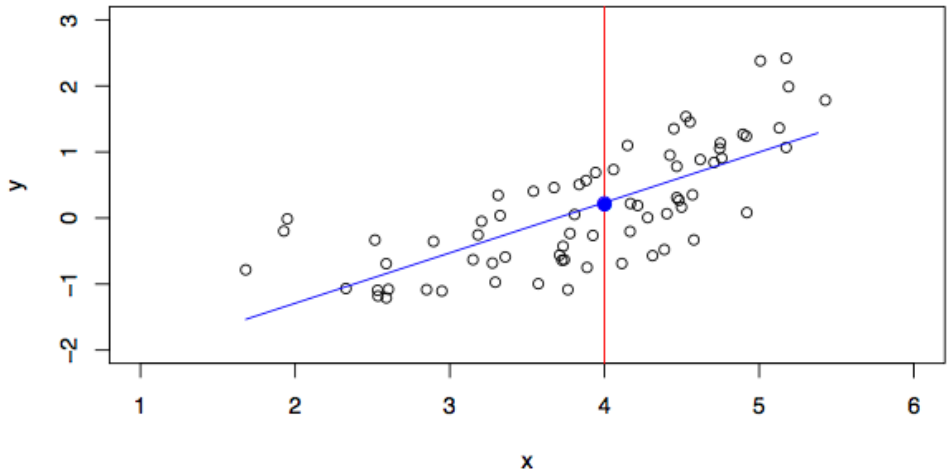
More on linear model

$$\hat{f}_{linear}(X) = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p.$$

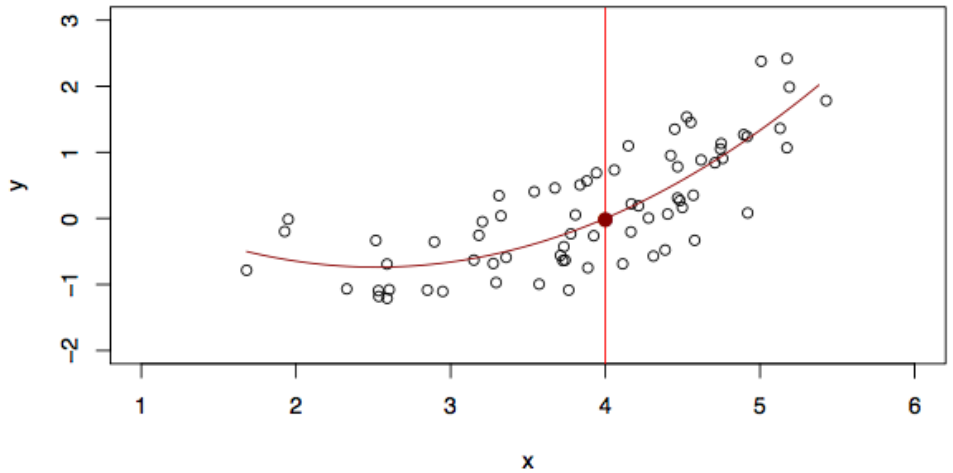
- The procedure of obtaining $\hat{\beta}_0, \ldots, \hat{\beta}_p$ is called **fitting**, i.e. fitting the model to training data.
- There are different ways of fitting! We will get back to this later.
- Although a linear model is almost never correct, it often serves as a good and interpretable approximation to the unknown true function $f(X)$.

A toy example of linear predictor

A fitted linear model $\hat{f}_{linear}(X) = \hat{\beta}_0 + \hat{\beta}_1 X$ via the least squares approach.



A more flexible model $\hat{f}quad(X) = \hat{\beta}_0 + \hat{\beta}_1 X + \hat{\beta}_2 X^2$ gives a slightly better fit



More complex parametric methods

As we have seen, if the linear predictor does a poor job, one can consider more complex forms of f, such as:

- Quadratic form: $f(X) = \beta_0 + \beta_1 X + \beta_2 X^2$
- Step-wise form: $f(X) = \beta_0 + \beta_1 1\{X <= 0.5\} + \beta_2 1\{X > 0.5\}$.
- Polynomial form: $f(X) = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_d X^d$
- Two layer neural net: $f(X) = \sigma\left(W_1 \sigma(W_0 X + b_0) + b_1\right)$
- You can keep adding complexity by considering more complicated f

Remark: f gets less interpretable as its form gets more complicated!

Non-parametric method

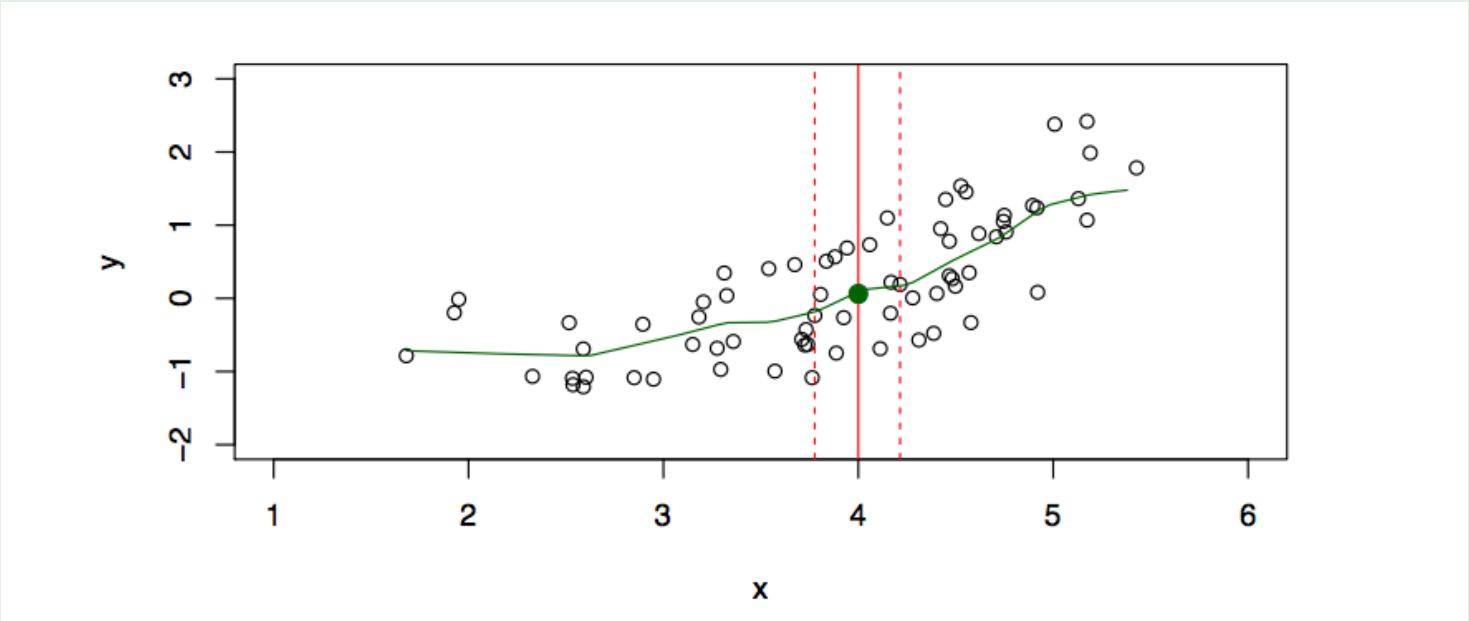
Make no / little assumption on f.

Example (Nearest neighbours)

To predict at $X = x$, consider the local average,

$$\hat{f}(x) = Ave(Y|X \in \mathcal{N}(x))$$

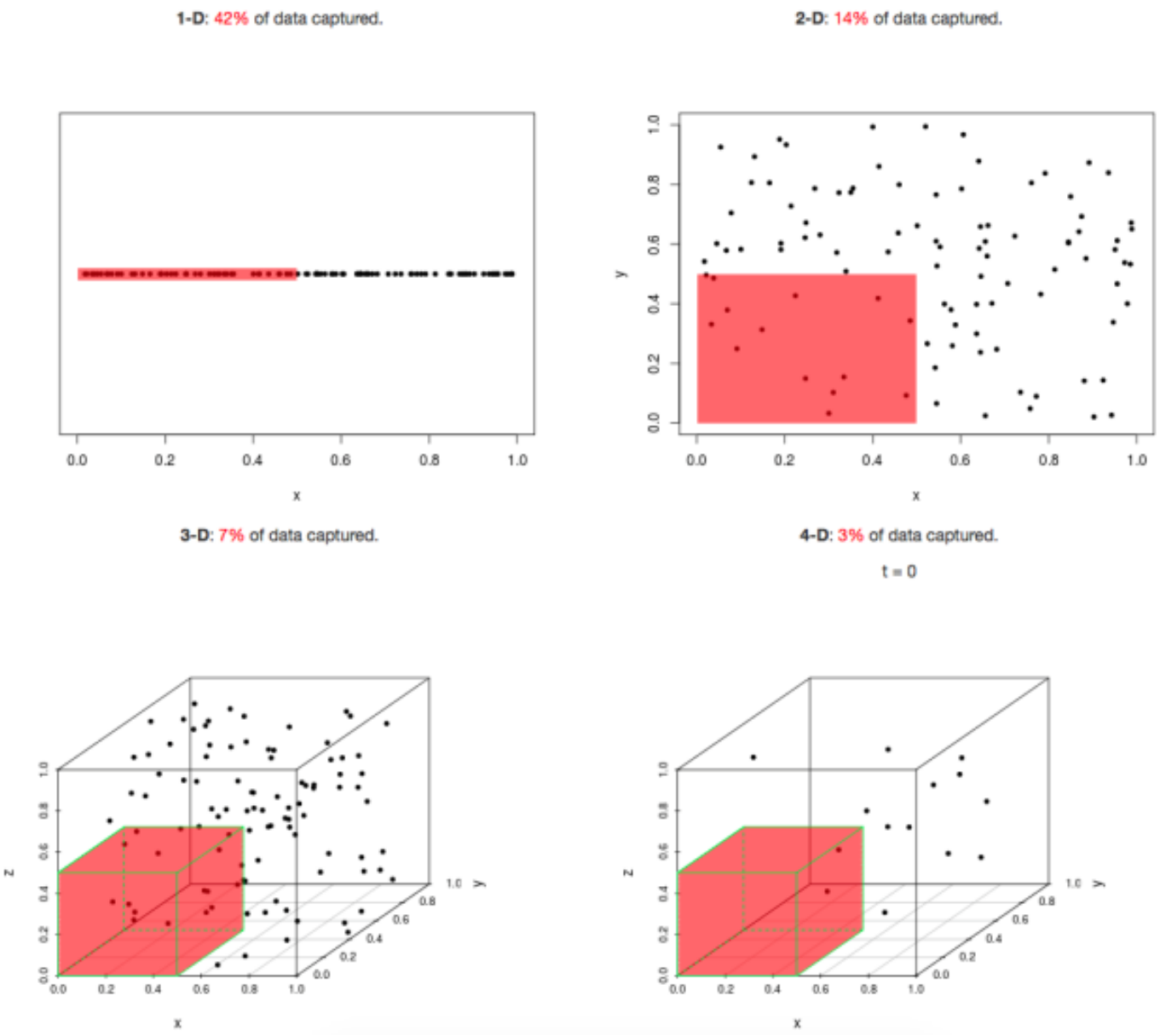
where $\mathcal{N}(x)$ is some neighborhood of x.



General comments on non-parametric methods

- Nearest neighbor averaging can be pretty good for small p i.e. $p \leq 4$ and large n.
- More sophisticated versions, e.g. kernel estimator, spline estimator.
- **Curse of dimensionality**: There are very few data points in the nearby neighbors when p is large.

Curse of Dimensionality



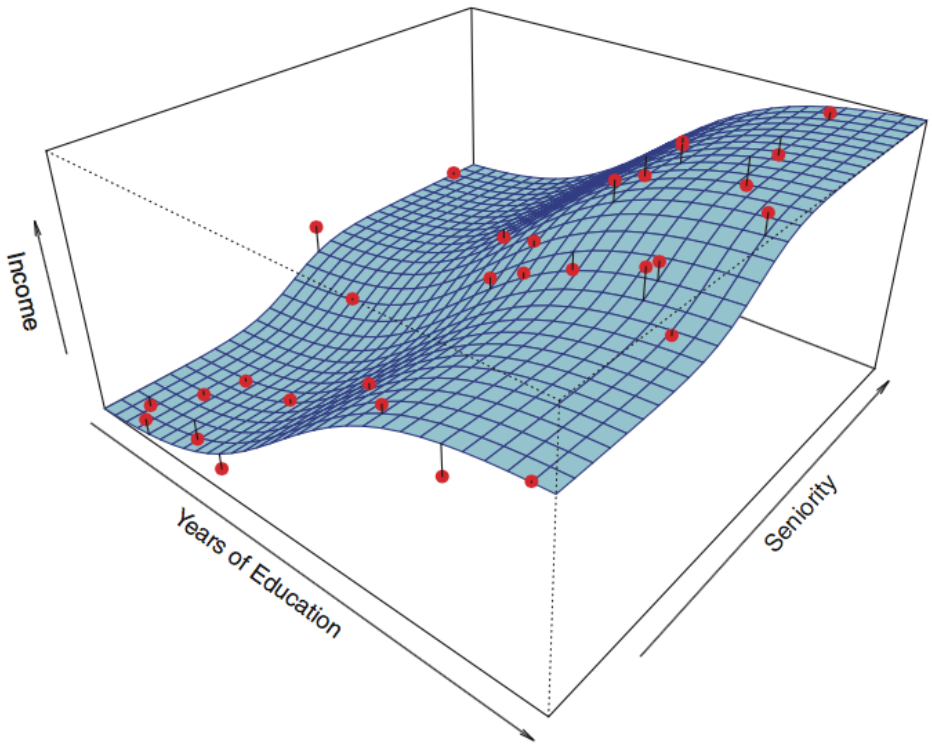
Overfitting

- In parametric/nonparametric approach, fitting a more complex model requires estimating a greater number of parameters, requiring a larger amount of training data points.

- These more complex models can lead to a phenomenon known as **overfitting** the data, which essentially means they follow the errors, or noise, too closely.

- A simple example of overfitting: $\hat{f}(x_i) = y_i$ for all $1 \leq i \leq n$.

Simulated example

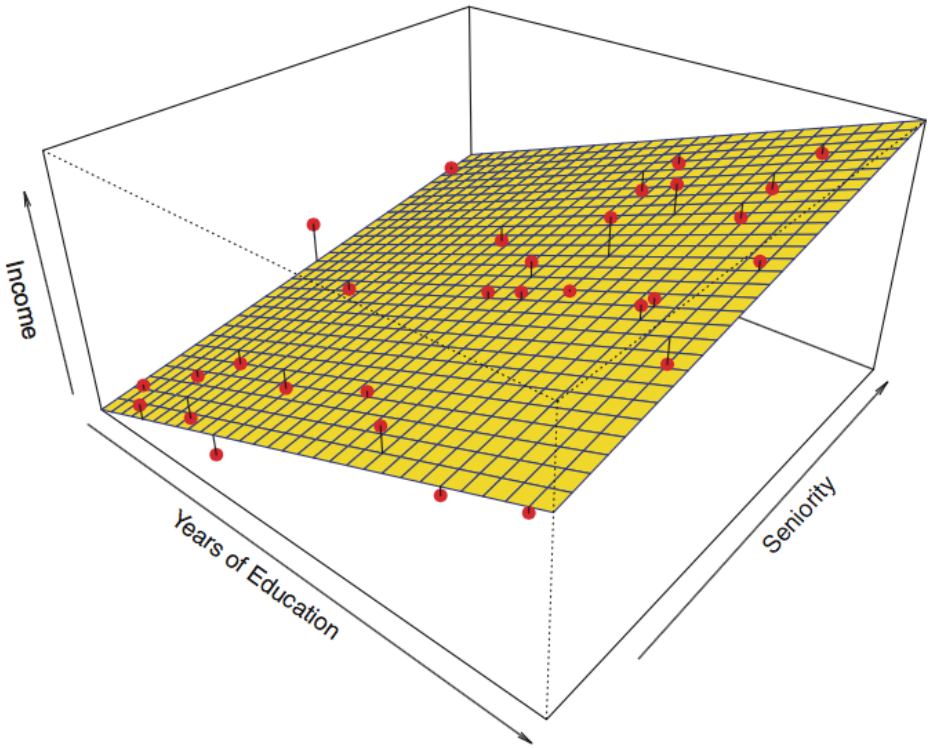


Red points are simulated values for income from the model

$$income = f(education, seniority) + \epsilon$$

where f is the blue surface.

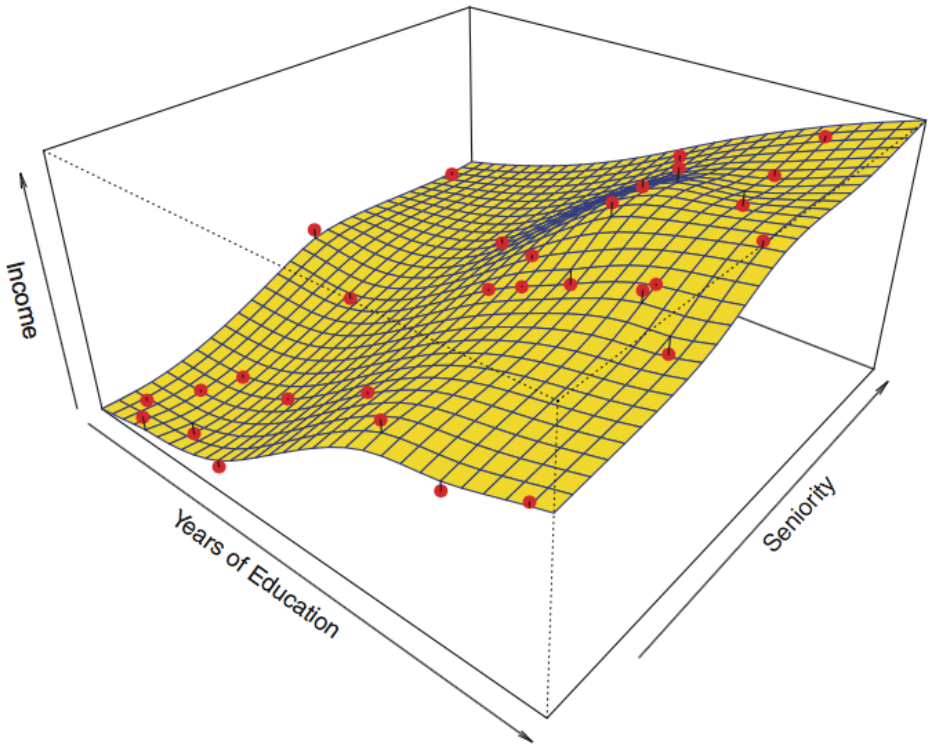
Example



Linear regression model

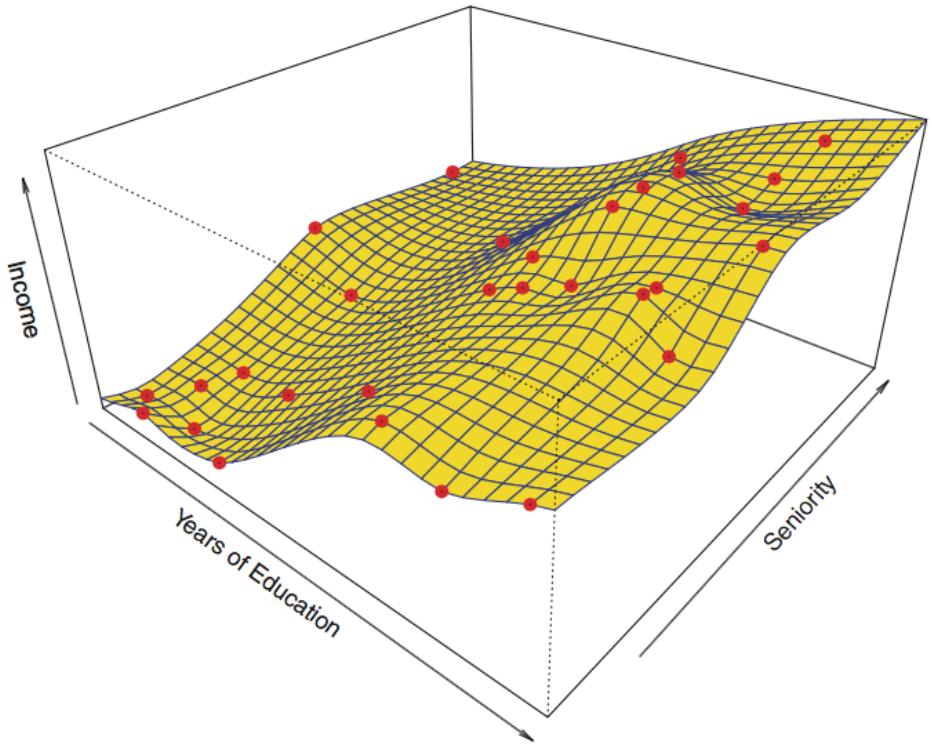
$$\hat{f}_L(\textit{education}, \textit{seniority}) = \hat{\beta}_0 + \hat{\beta}_1 \times \textit{education} + \hat{\beta}_2 \times \textit{seniority}$$

Example



A more flexible model (good nonparametric model).

Example

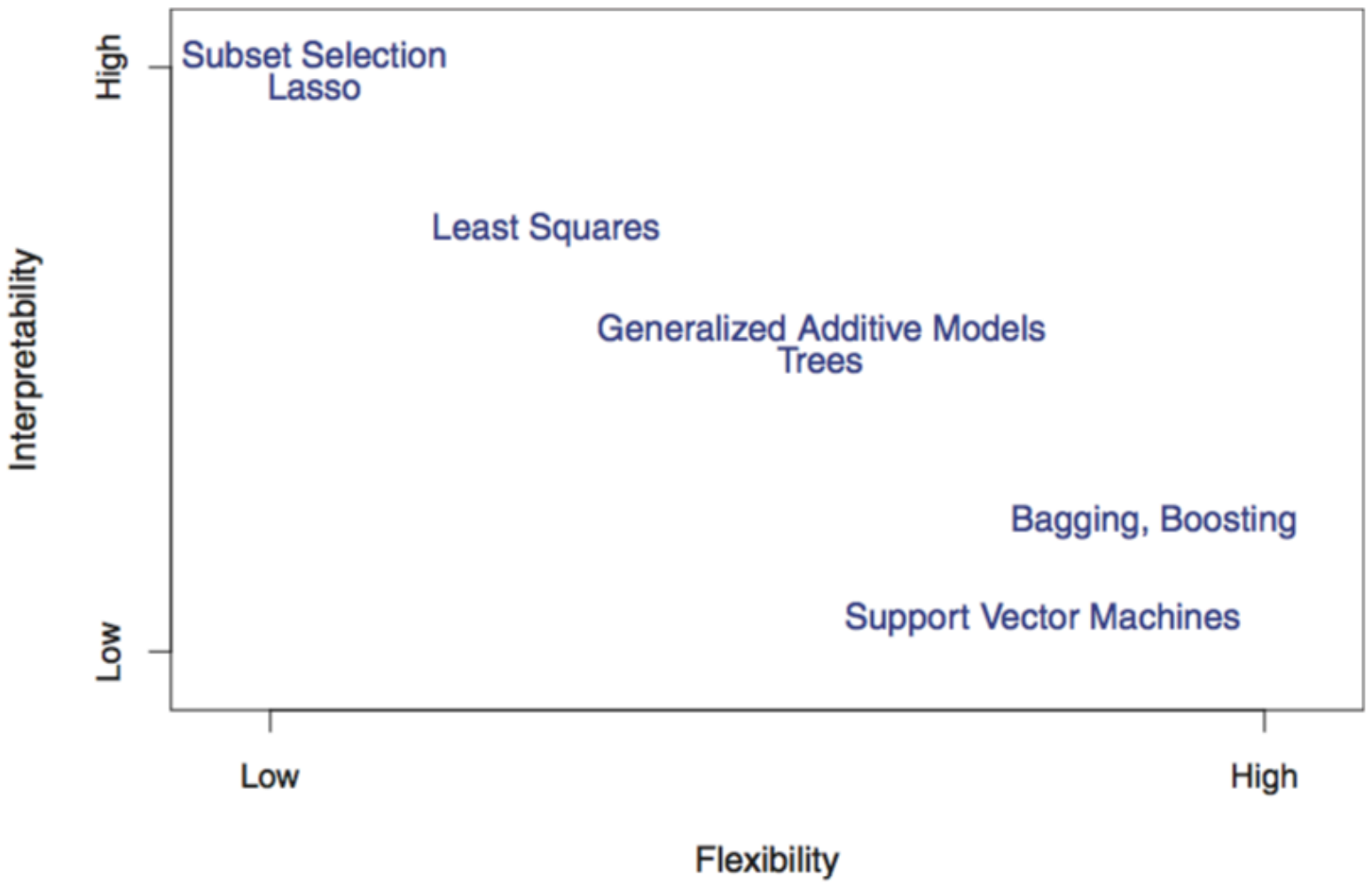


A even more flexible model (nonparametric model with a bad choice of tuning parameters). This fit makes zero errors on the training data! Also known as **overfitting**. Overfit models tend to have larger variability!

Some Trade-off

- Model complexity (fitting flexibility) versus interpretability.
  - The more complex, the more flexible to fit f, but less interpretable.
  - Linear models are easy to interpret; neural nets are not.
  - Interpretation means to understand how the predictors X contribute to predicting Y.
- Parsimony versus black-box
  - Given the same fit, we often prefer a simpler model involving fewer variables over a black-box predictor involving them all.
- Good fit versus over-fit or under-fit.
  - How do we know when the fit is just right?

Flexibility versus interpretability



What's next?

We will learn

- metrics that are used to evaluate the performance of $\hat{f}$
- the bias and variance tradeoff of $\hat{f}$
- the linear predictor (linear model)

Questions?