

# SOSC 4300/5500: Prediction

Han Zhang

# Outline

Prediction vs. Explanation

Predicting pandemic

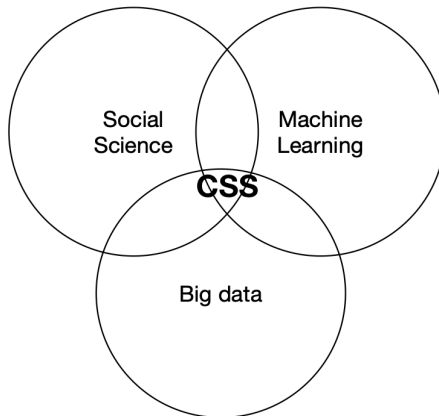
Common ML Algorithms for Prediction

# Logistics

- Grouping?
- Other questions?

# Computational Social Science (CSS)

- We have learned the pros and cons of big data
- Next we focus on using machine learning and big data to make **predictions**



## Prediction vs. Explanation

- Prediction vs explanation
  - Prediction: Whether Trump or Clinton will win the election?
  - Explanation: Why Trump won?
- [In class activities]: Can you give other examples?

## Prediction vs. Explanation: the ideal case

- Ideal case: classical physics, such as Newton's Law of Motion
  - Predictive: we can precisely predict location of planets in solar system
  - Explanative: we have a theory to explain why

- Social worlds are typically too complicated to summarize using several equations
  - We do not have a powerful formula such as  $F = ma$
- Current social science research focus dominantly on **explanation**
  - Testing a theory that looks like “A leads to B”
- But not asking “whether a given theory can predict some outcome of interest”
- There are some pushback toward this overemphasis on explanation and the neglect of prediction

## Pushback 1: Some theories are not useful

- Are our theory really useful?
- Timur Kuran, *Now out of Never: The Element of Surprise in the East European Revolution of 1989*, World Politics **44** (1991), no. 1, 7–48
- In 1987, the American Academy of Arts and Sciences invited a dozen of specialist, including several living in Eastern Europe, to prepare interpretive essays on East European developments. . .
- This was publised in the journal *Daedalus*
- “None foresaw what was to happen”.



## Pushback 1: Some theories are not useful

- Rational choice theory
  - Mancur Olson, *The Logic of Collective Action*, Harvard University Press, 1965
  - People has incentive to free ride
  - So it predicts the **lack** of revolution
- Structural theory: revolution occurs when the state becomes weaker
  - Theda Skocpol, *States and Social Revolutions: A Comparative Analysis of France, Russia and China*, Cambridge University Press, 1979
  - Partially gives a prediction
  - But there are many countries with weak state power but no revolution
  - Eastern European countries were certainly not the countries with the weakest state power then
- Both cannot precisely predict the occurrence of revolution

- Many policy problems are intrinsically a prediction problem
- Judges need to decide whether to give bail to suspects
  - If the decision is correct, saves money
  - If the decision is wrong, suspects commit new crimes or run away
- This is intrinsically a **prediction** problem, because judges are also predicting what a defendant would do if released
- Question: what will be a **explanatory** way of asking questions about judges?

## Pushback 2: Social problems as prediction problems: policy as predictions

- Jon Kleinberg, Himabindu Lakkaraju, Jure Leskovec, Jens Ludwig, and Sendhil Mullainathan, *Human Decisions and Machine Predictions*, The Quarterly Journal of Economics **133** (2018), no. 1, 237–293
- Machines can predict whether to give bails more precisely than human judges

## Some issues of predictions

Jake M. Hofman, Amit Sharma, and Duncan J. Watts, *Prediction and explanation in social systems*, Science **355** (2017), no. 6324, 486–488

- Big data + machine learning -> better predictions (data-driven)
  - Second half of this lecture
- Standards of prediction
  - We can only tell some predictions are good or bad if we agree on a common standard
  - Otherwise, we can easily fall into meaningless debates
    - e.g., someone just tells you that “I have predicted the collapse of the Soviet Union”
  - Will be the focus of next week
- Limits to prediction
  - A cutting-edge research area

## Example of Prediction: Google Search and Flu

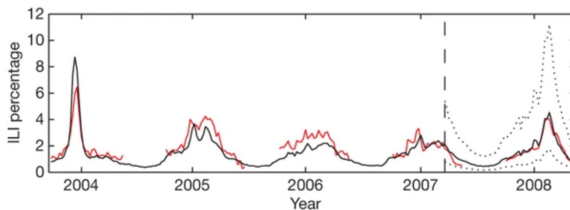
- Can we use big data for prediction?
- Jeremy Ginsberg, Matthew H. Mohebbi, Rajan S. Patel, Lynnette Brammer, Mark S. Smolinski, and Larry Brilliant, *Detecting influenza epidemics using search engine query data*, Nature **457** (2009), no. 7232, 1012–1014
- Background: influenza (flu) tracking system in CDC
  - Patients visit doctors -> doctors make diagnosis -> report to CDC
  - Accurate, but with a lag of weeks
- Using Google Searches to track influenza in real time
  - Intuition: people will search flu-related words, such as “flu symptoms”
  - And the trends of these searches predict ups and downs of flu cases

## Google Flu Trends: Details

- 45 search queries related to Influenza-like illness (ILI)
- $Q(t)$ : ILI-related query fraction at time  $t$ , out of all searches in a geographic region
- $I(t)$ : Number of ILI physician visit at time  $t$
- Model: simple linear regression
- $\text{logit}(I(t)) = \beta \text{logit}(Q(t)) + \epsilon$
- The model was fit using data from 2003 to 2007
- And make predictions for 2008

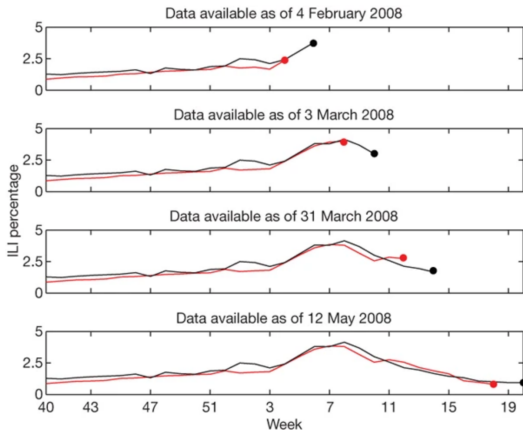
## Google Flu Trends: Results

- Red is Google Search; black is CDC's count
- Correlation in 2008 is 0.95



## Google Flu Trends: nowcasting

- Nowcasting: predict what will happen in the near future/now
- A weaker and more realizable version of forecasting





## Google Flu Trends: discussions

- You can download Google's Flu Trends Data here (till 2015)  
[https://www.google.com/publicdata/explore?ds=z3bsqef7ki44ac\\_](https://www.google.com/publicdata/explore?ds=z3bsqef7ki44ac_)
- [In Class Activities]
  - What else you think Google's search trend can predict?
    - <https://trends.google.com/trends/explore?q=covid&geo=US>
  - What do you think are the potential problems of using search queries to predict influenza counts?

## Google Flu Trends: Critique 1

- Challenge 1: we can actually use old methods and old data to predict flu
  - Sharad Goel, Jake M. Hofman, Sébastien Lahaie, David M. Pennock, and Duncan J. Watts, *Predicting consumer behavior with Web search*, Proceedings of the National Academy of Sciences **107** (2010), no. 41, 17486–17490
  - $I(t) = \alpha + \beta_1 I(t-2) + \beta_1 I(t-3) + \epsilon$
  - The above autoregressive model achieves similar performances
    - But no need to collect big data! Existing statistics from the CDC is enough
  - “search data are comparable in utility to alternative information sources, but not necessarily superior”

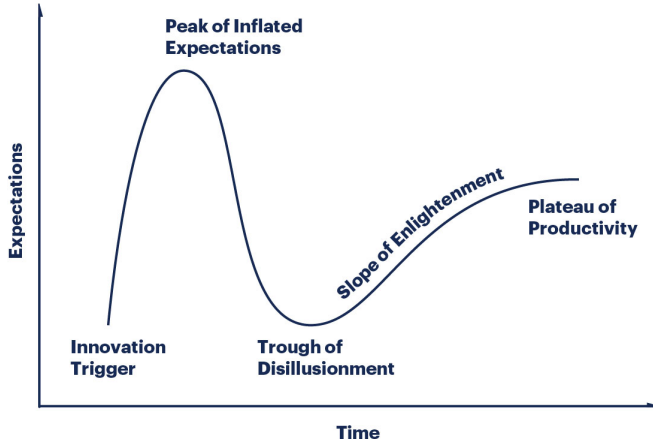
## Google Flu Trends: Critique 2

- Drifting
  - Users may change their search behaviors during pandemic period, leading to overestimation
  - Samantha Cook, Corrie Conrad, Ashley L. Fowlkes, and Matthew H. Mohebbi, *Assessing Google Flu Trends Performance in the United States during the 2009 Influenza Virus A (H1N1) Pandemic*, PLOS ONE **6** (2011), no. 8, e23610
- Algorithm confounding!
  - Google began to suggest related search words
  - David Lazer, Ryan Kennedy, Gary King, and Alessandro Vespignani, *The Parable of Google Flu: Traps in Big Data Analysis*, Science **343** (2014), no. 6176, 1203–1205

## Google Flu Trends: Aftermath

- There are tons of media report titled “Google’s Flu Project Shows the Failings of Big Data”
  - <https://time.com/23782/google-flu-trends-big-data-problems/>
- And Google stopped publishing estimate of ILI counts after 2015
  - <https://ai.googleblog.com/2015/08/the-next-chapter-for-flu-trends.html>

# Hype Cycle of Using Big Data for Prediction



## COVID-19 prediction using search queries

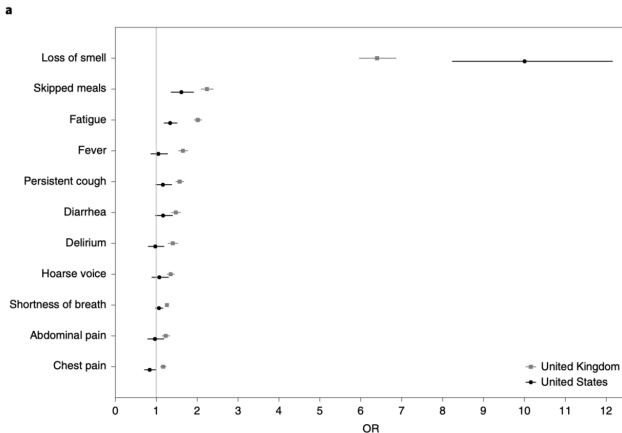
- Google began to release search queries related to COVID-19
  - [https://github.com/google-research/open-covid-19-data/tree/master/data/exports/search\\_trends\\_symptoms\\_dataset](https://github.com/google-research/open-covid-19-data/tree/master/data/exports/search_trends_symptoms_dataset)
- Many recent studies (just google “Google Search Predicts COVID”)
  - [https://www.cghjournal.org/article/S1542-3565\(20\)30922-8/fulltext](https://www.cghjournal.org/article/S1542-3565(20)30922-8/fulltext)
  - Loss of taste and loss of appetite correlated most strongly with the rise in COVID-19 (with a four-week lead)

## COVID-19 prediction using survey data

- Cristina Menni, Ana M. Valdes, Maxim B. Freidin, Carole H. Sudre, Long H. Nguyen, David A. Drew, Sajaysurya Ganesh, Thomas Varsavsky, M. Jorge Cardoso, Julia S. El-Sayed Moustafa, Alessia Visconti, Pirro Hysi, Ruth C. E. Bowyer, Massimo Mangino, Mario Falchi, Jonathan Wolf, Sebastien Ourselin, Andrew T. Chan, Claire J. Steves, and Tim D. Spector, *Real-time tracking of self-reported symptoms to predict potential COVID-19*, Nature Medicine **26** (2020), no. 7, 1037–1040
- 2,450,569 individuals who used an app-based symptom tracker.
- 15,368 had a COVID test
- 6,452 tested positive
- 9,186 tested negative

## COVID-19 prediction using survey data

- Logistic regression model to predict test results



- Can you think of some shortcomings of this article?



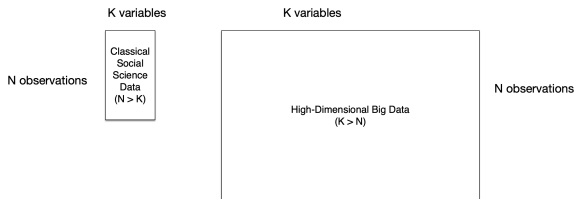
## COVID prediction

- Can you think of pros and cons of using search queries and survey data?
- Can you think of other data/information for predicting COVID infection/trends?
- Can you think of other methodological approaches?

## Short summary

- Prediction is different from explanation
- Current social sciences focus too much on explanation, but theory is often not good at prediction
- Prediction can be useful for real-world policy problems
- Bear a critical mind! Note the limitations in data source/methods

# Why Machine Learning?

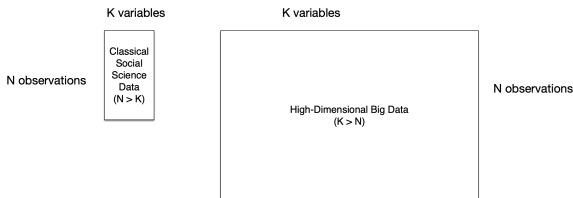


- We have big data (large  $N$ )
- Big data are also high-dimensional ( $K > N$ )
- They together makes applying traditional regression models on big data a difficult problem

## Learning goals

- I am going to introduce intuitions of some important algorithms:
  - LASSO/Ridge/Elastic Net
  - SVM
  - Decision tree and its extensions
    - Bagging trees and random forests
    - Boosting trees
- In tutorial, we will cover how to implement these algorithms in R/Python
- These algorithms are more complex than linear regression; for the same algorithm, you still have different choices to make (called **tuning parameters**)
- Learning goals:
  - [minimum level]: learn how to implement these algorithms in R/Python
  - [for advanced students]: try to understand the math and detailed options of these algorithms as much as possible

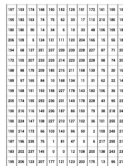
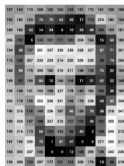
# High-dimensional data



- In the previous Google Flu/COVID prediction cases, there are lots of observations and relatively few variables
  - In these two cases, simple regression models are usually okay
- Other times: there are more variables than observations
  - These are typically called **high-dimensional data** ( $K > N$ )
- Many data are intrinsically high-dimensional, such as text, images, videos, audio, and networks.

## High-dimensional data example: image

### Pixel Representation

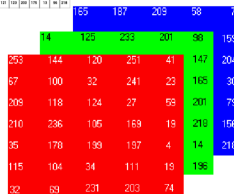


• Grayscale  
Representation of an 16  
\* 12 image

• Each cell is from 0 to  
255 (0 is black and 255  
is white)

• Matrix can further be  
simplified as 16\*12 =  
192 dimension vector

- RGB Representation of an 6\*5 image
- This matrix can then be simplified as a  
6\*5\*3= 90 dimensional vector



- For a typical 800\*600 color image, the dimension  $K$  is  
 $800*600*3=1.44\text{M}$
- If the number of observations excels the dimension, you need  
to have 1.44M images

## Goal of supervised machine learning

- Requirement: a set of input  $X$  and output  $Y$  as **training data**
  - These are like examples you provided to computers; **supervised**
- Goal: find an algorithm  $f(\cdot)$ , “such that for future  $X$  in a **test set**,  $f(X)$  will be a good predictor for  $Y$ ” (Breiman, 2001)
  - There are many different algorithms
  - We will cover some most common ones
    - Focus on intuition, not formal math derivation

## Two types of machine learning in CS

- Predicting **continuous** outcomes is often called **regression** tasks
  - Yes linear regressions are a type of machine learning, the simplest one
- Predicting **categorical** outcomes is called **classification** tasks
- **Caution:** the above are CS notations; they differ from social science terminology.
  - E.g., logistic regression is treated as a classification task in machine learning community



## Simplest ML algorithm: regression

- Linear regression: for continuous outcome  $Y$ 
  - $Y = \beta X$
- Logistic regression: for binary outcome  $Y$ 
  - $Y = \text{logit}^{-1}\beta X$
- Multinomial/ordered logistic regression: categorical/ordinal outcome  $Y$

## LASSO and Ridge

- When data dimension is high (e.g.,  $K > N$ )
  - Linear regression fails because it wants to take consideration of all the variables
  - But usually most variables are not relevant
- To make simple regression works, we can force some variables to be irrelevant:
  - LASSO regression: force the coefficient of some variables to be 0
    - Controlled by a parameter  $\lambda_1$ ; bigger  $\lambda_1$  forces more coefficients to be 0
  - Ridge regression: force the coefficient of some variables to be very small
    - Controlled by a parameter  $\lambda_2$ ; bigger  $\lambda_2$  forces more coefficients to be small
- The idea to explicitly make a model simpler is called **regularization**
- Note that idea is quite counterintuitive: to make a model more effective, sometimes you have to simplify it

## LASSO and Ridge: math

- We have  $p$  variables
- Linear regression minimizes Mean Squared Error (MSE):

$$\hat{\beta}_{OLS} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^n (Y_i - X_i\beta)^2 \quad (1)$$

- Lasso estimator (Tibshirani, 1996, Least Absolute Shrinkage and Selection Operator):

$$\hat{\beta}_{LASSO} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^n (Y_i - X_i\beta)^2 + \lambda_1 \sum_{j=1}^p |\beta_j| \quad (2)$$

- Ridge estimator (Hoerl and Kennard, 1970; Turgenev, 1943):

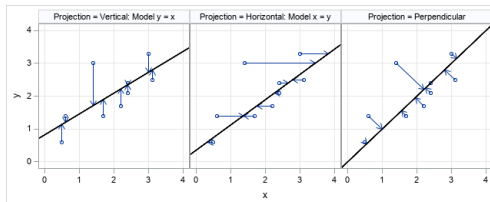
$$\hat{\beta}_{Ridge} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^n (Y_i - X_i\beta)^2 + \lambda_2 \sum_{j=1}^p \beta_j^2 \quad (3)$$

## Elastic Net

- Combine LASSO and Ridge
- With weights  $\lambda_1$  for LASSO and  $\lambda_2$  for Ridge
- How do we choose  $\lambda_1$  for LASSO and  $\lambda_2$  for Ridge?
  - These are classical examples of **tuning parameters**.
  - You have to choose them.
- We will discuss this in detail next week

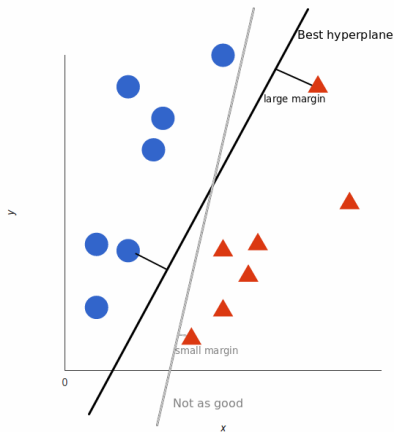
## SVM

- SVM is another popular ML algorithm
- Linear regression project observation points vertically onto the “fitted line”
- The left and middle one are linear regressions
- The right one is the simplest “Support Vector Machine” (SVM)
- SVM try to find a line that maximizes the “margins” between data



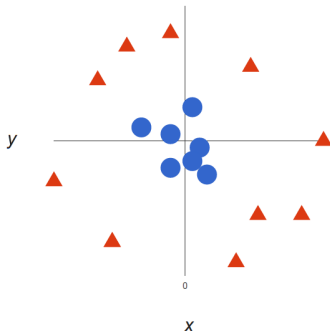
## SVM: linear case

- SVM: maximize margin



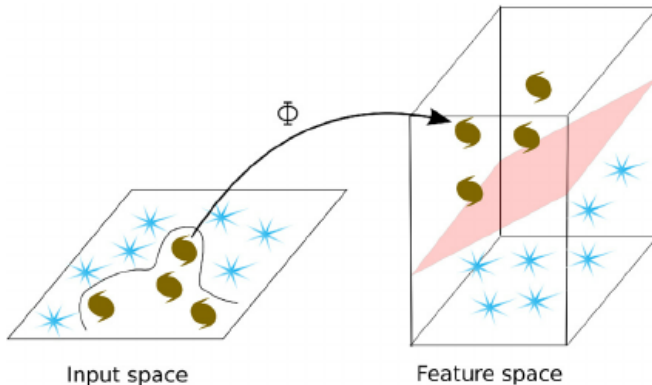
## SVM: nonlinear case

- Some data are not linearly separable
- That is, it's mathematically impossible that you write a linear/logistic regression and use different interactions of  $X$  to perfectly classify  $Y$



## SVM and Kernel Trick

- More complex SVM has a different intuition: transform data from input space (raw inputs) to a higher dimensional feature space that helps the classification
- This transformation is called “kernel trick”



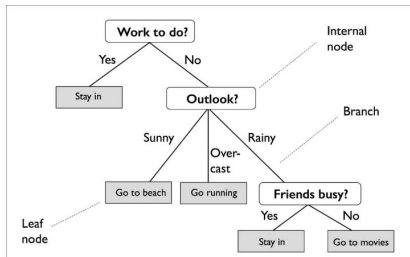


## SVM: practice

- Commonly used kernels
  - Linear/polynomial kernel: less powerful. Not able to project the data onto higher dimension.
    - quicker
  - Radial basis function kernel (RBF): more explicitly project the data onto higher dimension, thus is more powerful
    - much slower
- First developed for binary classification; some extensions are made for multiple
- Has dominated the CS literature for a while (in the 90s and early 00s)

## Decision Tree

- Decision tree visualizes one's **sequential** decisions process ( $Y$ ), based on some predictors (variables)

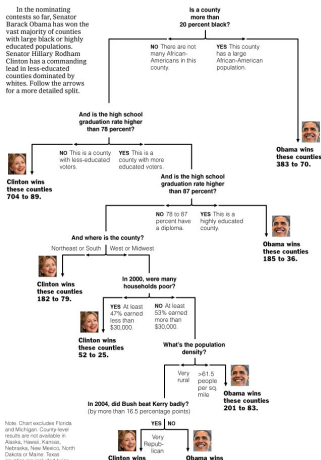


- Decisions (outcomes)  $Y$  are located at leaves
- If you are familiar with linear regression, the rightmost branch has a triple interaction

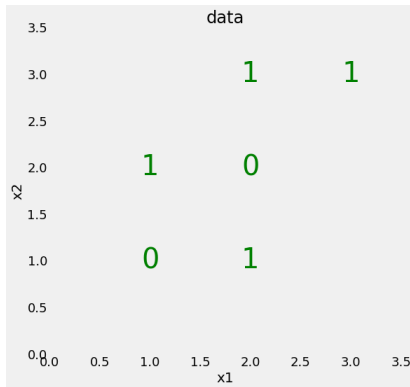
# Decision Tree Example

[https://archive.nytimes.com/www.nytimes.com/imagepages/2008/04/16/us/20080416\\_OBAMA\\_GRAPHIC.html?emc=polb1&nl=pol](https://archive.nytimes.com/www.nytimes.com/imagepages/2008/04/16/us/20080416_OBAMA_GRAPHIC.html?emc=polb1&nl=pol)

Decision Tree: The Obama-Clinton Divide



## Growing a Tree by hand



- The above data cannot easily be separated by drawing a straight line (i.e., simplest linear regression)
- Let us draw a tree by ourselves to distinguish  $Y = 0$  vs.  $Y = 1$ , based on  $X_1$  and  $X_2$ 
  - We want a **binary** tree: split into two branches

## Estimating a tree with software

- Modern statistical software can help you to fit a tree automatically
  - R users: `randomForest` package
  - Python users: `sklearn` package; use `RandomForestClassifier` or `RandomForestRegressor`
- But we have to understand the key decisions these algorithms make:
  - Among many ways to grow a tree, how do we choose a good one?
  - What's the depth of tree?

## Decision tree algorithms: some principles

1. Most algorithms typically assume binary tree. Otherwise:
  - For continuous  $X$ , we can split it in many ways
  - For categorical  $X$ , if the number of levels is large, we can still have a very wide tree
2. What if we there are multiple outcomes on a same leaf?
  - For continuous outcomes, the prediction is the mean
  - For categorical outcomes, the prediction is the mode
3. No need to use all predictors
  - That is, if a variable is not important, no need to use it
4. One predictor can be used multiple times

## Decision Tree Algorithms: formal math

- **challenging topic**
- Decision Tree Algorithms help you to draw a tree from more complex data
- What are the steps we should take?
- Let us first work with continuous outcome  $Y$ : regression tree
- There are two questions to consider:
  - Which variable  $X_j$  to choose first?
  - We will split  $X_j$  into  $X_j < s$  and  $X_j \geq s$ . How do we choose  $s$ ?
- And the intuitive answer is that:
  - You choose choose  $X_j$  and  $s$  that best separates  $Y$  (thus predicts  $Y$  the best)

## Decision Tree Algorithms: formal math

- If we write this intuition down mathematically:
- We have  $p$  predictors:  $X_1, \dots, X_p$
- For each predictor  $X_j$ , calculate its minimum MSE:
  - Consider all its possible cutoffs  $s$ . A particular cutoff  $s$  will split the data into two regions:

$$R_1(j, s) = \{X | X_j < s\} \text{ and } R_2(j, s) = \{X | X_j \geq s\} \quad (4)$$

- We should select a  $s$  that minimizes the MSE

$$\sum_{i: x_i \in R_1(j, s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j, s)} (y_i - \hat{y}_{R_2})^2 \quad (5)$$

- $\hat{y}_{R_1}$  is the mean response for the training observations in region  $R_1(j, s)$
- Select  $X_j$  and its  $s$  whose MSE is the smallest
- Repeat step 2 and 3 multiple times until reaching certain depth



## From One Tree to Many Trees

- **Bagging** tree (or **ensemble** of trees): averaging the predictions of many trees
  1. From the original training data, draw a sample with replacement of equal size
  2. Fit a tree for each sample
  3. Repeat 1 and 2 for some times
- Take the mean of estimates of each tree to produce a single estimate for each test data point

## Random Forest

- Random Forests further extend the idea of bagging
- The key innovation of random forests:
- For each sample from the original training data, randomly select  $m$  variables (not using all  $p$  variables), and grow a tree;
  - A common choice:  $m = \sqrt{p}$
- In other words, we just force  $p - m$  predictors to be non-relevant each time
- Why? High-dimensional data! Needs regularization

## Boosting trees

- Bagging tree and Random Forest create many trees and average them together
  - Each of the tree is independent of the others
- A different idea is to create a **sequence** of trees that gradually improve over each other

## Boosting trees

- Assume you first fit a decision tree  $G_1(X)$
- Bagging trees and random forests: fit another decision  $G_2(X)$ , totally independent of  $G_1$ 
  - Then final prediction  $Y = \frac{G_1(X) + G_2(X)}{2}$
- Boosting trees: find  $G_2(X)$  based on **prediction error** of the first tree
  1. Learn a second tree  $G_2(X)$  to predict  $Y - G_1(X)$
  2. Then final prediction  $Y = G_1(X) + G_2(X)$
  3. Repeat Step 1 and 2: learn a new tree  $G_3(X) = Y - G_1(X) - G_2(X)$ , and so on.
- Essentially, boosting trees find data points that previous algorithms **are most likely to be wrong**, and improve the algorithm on these points.

## Boosting trees vs Random Forests

- You may hear many different variants of trees
  - AdaBoost is the first and Gradient Boosting Tree is the most successful
- Gradient Boosting Tree (GBT) and Random Forests (RF) are typically the two best methods you can get
  - GBT typically works well when the dimension is not that high
  - RF works well when the dimension is very high

## Next week

- More discussions on regularization
- Evaluation and selection of ML algorithms