

Data 100: Principles & Techniques of Data Science

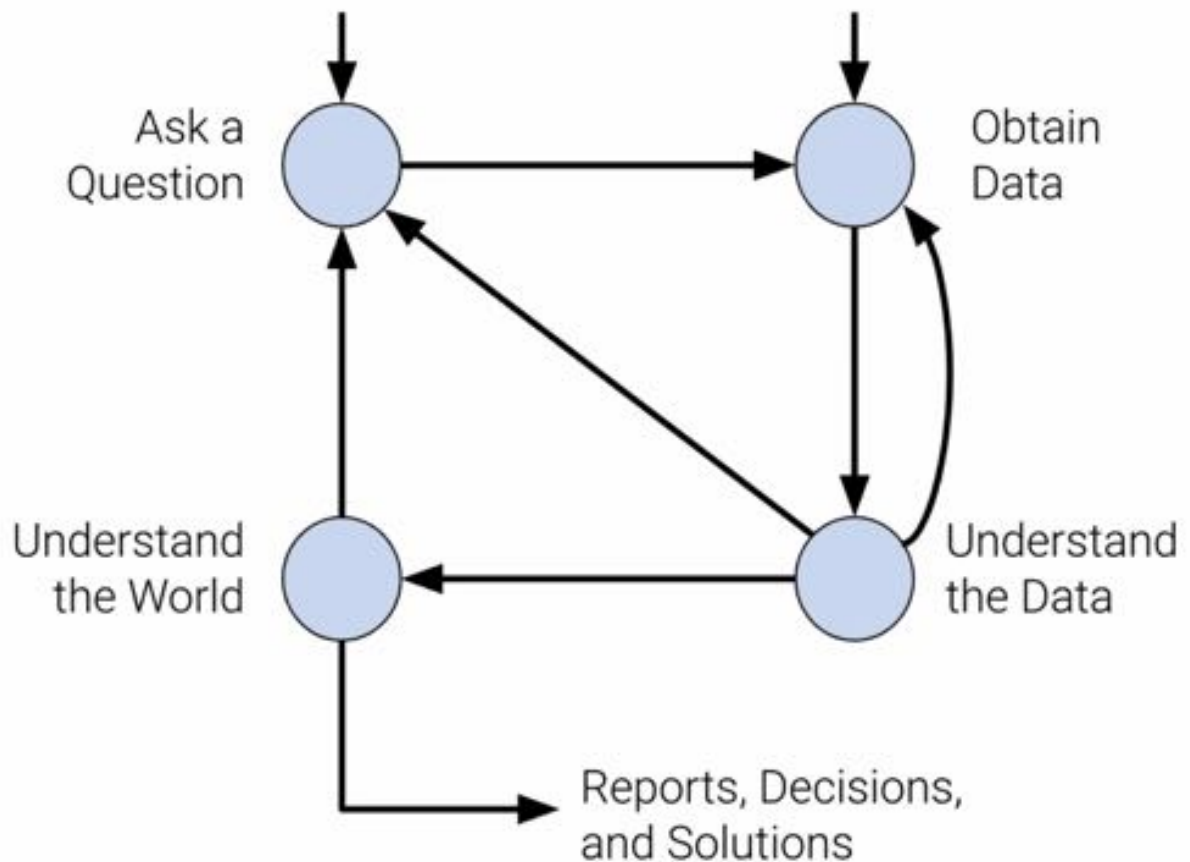
Lecture Notes

Week 1: Lecture 1 Fireside Chat (1/19)

What is Data Science?

- CS, Statistics, Domain Knowledge (Reason about problem domain)

Data Science Lifecycle



Week 1: Lecture 2 Data Sampling and Probability (1/21)

To interpret raw data, we model

- Want to know how they calculated the data, why certain numbers are used
- See what assumptions are made

Census

- Count every single person

Surveys

- Set of questions:
- What is asked and how it is asked, affects how the respondent answers, whether the respondent answers

Sample

- Subset of a population because census is too expensive
- Chance Error: random samples can vary from what is expected, in any direction
- Bias Error: a systematic error in one direction

Convenience samples

- Whoever you can get ahold of
 - Not a good idea

Quota sample

- Specify your desired breakdown of various subgroups
 - Not random,

Quality, not quantity

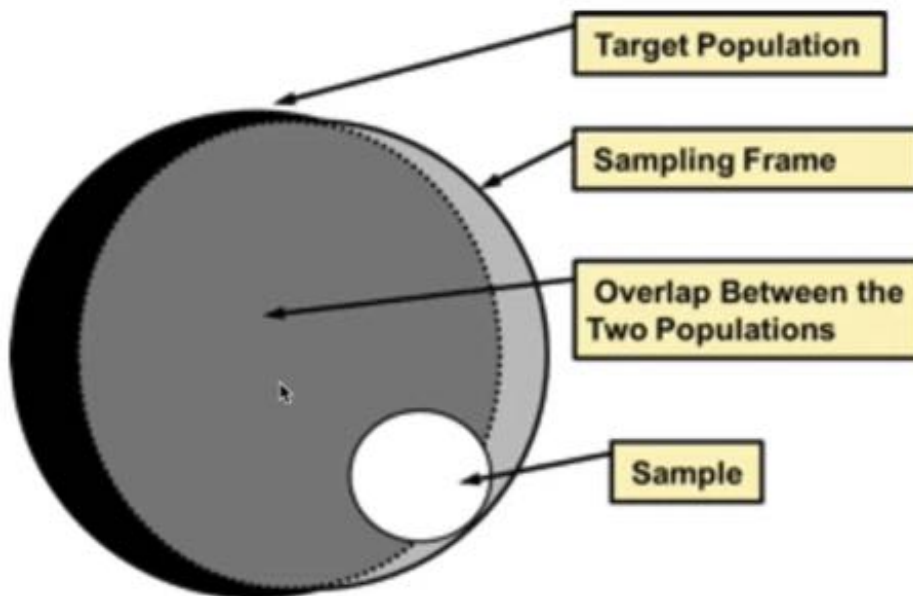
- Try to ensure that the sample is representative of the population
- Don't just try to get the big sample

Case: 1936 Election

- Sent from phone books, magazines
- Predicted Landon would win but Roosevelt actually won
- Not representative, more affluent people, low response rate
- Big sample aren't always good, Gallup had more accurate

Population, samples, sampling frame

- Population: group you want to learn something about
- Sampling Frame: list from which the sample is drawn
- Sample: who you actually end up sampling, subset of sampling frame



Selection Bias

- Systematically excluding particular groups
- Avoid: examine the sampling frame and method of sampling

Response Bias

- People don't always respond truthfully
- Avoid: Examine the nature of questions and method of surveying

Non-response Bias

- People don't always respond
- Avoid: Keep your survey's short and be persistent
 - People who don't respond aren't like the people who do

Probability Sampling

- Reduce bias, estimate the bias and chance error, quantify the uncertainty
- Must be able to provide chance that any specified set of individuals will be in the sample,
- Be able to measure the errors

Some random sampling schemes

- Random sample with replacement uniformly at random with replacement
- Simple random sample is a sample drawn uniformly at random *without* replacement
 - Every individual has same chance of being selected, every pair same chance, every three same chance...

Binomial and Multinomial Probability

The scenario

- Sample at random, with replacement
- Multiply binomial by chance of specific sequence
- Binomial Distribution: n times, probability p, k successes

$$P(k \text{ successes}) = \binom{n}{k} p^k (1 - p)^{n-k}$$

Multinomial probability: n times, proportion p₁

$$\frac{n!}{k_1! k_2! k_3!} p_1^{k_1} p_2^{k_2} p_3^{k_3}$$

Week 2: Lecture 3 Estimation and Bias (1/27)

Statistical Bias

- Difference between your estimate and the truth

Key Concepts

- Population: the set of all units of interest, size N.
- Sampling frame: set of all possible units that can be drawn with the sample
- Sample: subset of the sampling frame size n

Scenario 1: A census

- Population = sampling = sample frame
- Pros: lots of data, no selection bias, easy inference
- Cons: expensive (time, money)
 - Often impossible

Scenario 2: Administrative Data

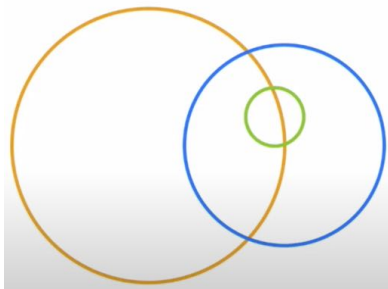
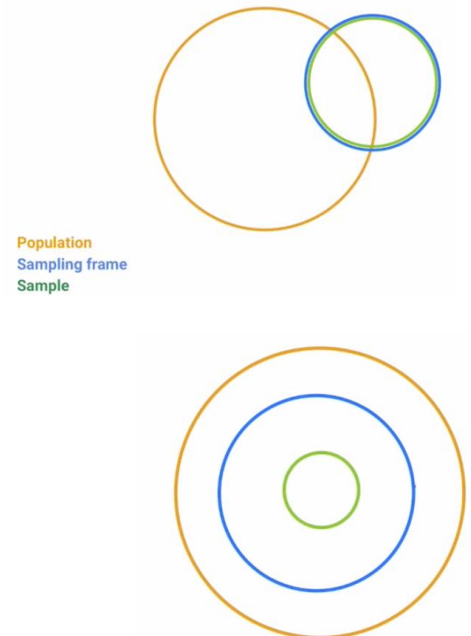
- Sampling frame contains a lot not in population, collect from info

Scenario 3: What we like to think we have

- Sample is representative

Scenario 4: what we usually have

- Sample may be drawn from a skewed frame and may not be representative of population



Case study: 1936 election

- Population: all people who will cast votes in the 1936
- Selection bias: systematically favoring certain groups for inclusion
- Non-response bias: when people who don't respond are non-representative

Case Study - Gender diversity in Data science

- Proportion of data 100 identify as female
- Baby names !=
- Zoom poll: onlookers
- Pre-class survey: people that didn't fill out the survey

Random Variables

- Distributions and Data Generation
- Empirical Distribution:

- Distribution of your sample
- Probability Distribution
 - A model for how the sample is generated

Random Variables

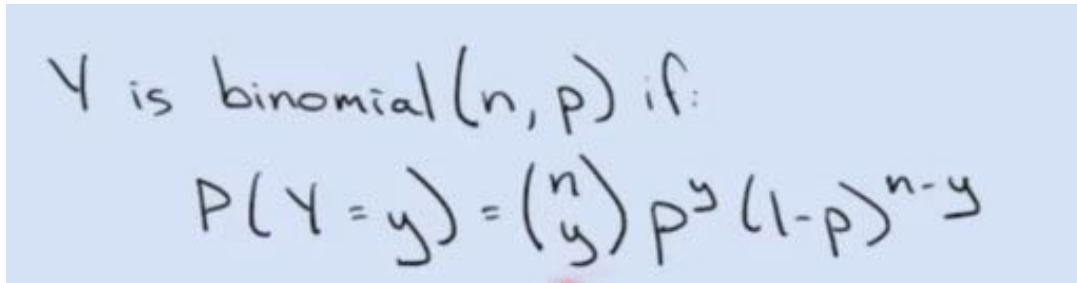
- Variable that can take numerical values with particular probabilities
- Maps name to numerical values
- Value is taken by a lower case value

Bernoulli Distribution

- Random variable that takes the value 1 with probability p and 0 otherwise
 - X is bernoulli(p) if $P(x = 1) = p$
 - $P(x = 0) = 1-p$

Abstracting Random chance

- Can be sums of Bernoulli RVs



Y is binomial(n, p) if:

$$P(Y = y) = \binom{n}{y} p^y (1-p)^{n-y}$$

Common distributions

- Bernoulli (p)
 - Takes on value 1 with probability p and 0 with probability $1-p$
- Binomial(n, p)
- Uniform

Expected Value

- Weighted average of the values of X , where the weights are the probabilities of the values
- Linear transformations: $Z = aX + b$
 - Expected Values: $E(Z) = aE(X) + E(b) = a E(X) + b$
- Additivity: $W = X_1 + X_2$
 - $E(W) = E(X_1) + E(X_2)$
- Linearity of Expectations: $V = aX_1 + bX_2$
 - $E(V) = aE(X_1) + bE(X_2)$
- Binomial
 - Let Y be binomial(n, p): $Y = X_1 + X_2 + \dots + X_n$
 - $E(Y) = np$

Summary

- In order to understand the world, know how data was generated
- Random Variables and distribution formalize

How does estimate differ from truth (Plato's Allegory of the Cave)

- World of forms
 - Non-physical essence of all things
- World of representation
 - The material world that we observe

Metaphor of the Cave

- World of parameters
 - Constants that define the structure of the world
- World of Data/Statistics
 - Observable information generated by RVs and their parameters
 - Statistic: numerical summary of data

Statistic

- A single piece of data
- A numerical summary of a dataset

What is an estimator?

- Can pick our estimate

Estimation error

- Sampling variability

Statistic bias

- Expected value of estimator and truth (parameter)
- $E(\theta) - \theta$
- If data wasn't generated by θ , it will not be representative of population
- What if estimator isn't

Lessons

- Think about how data was generated
- Think about your model/estimator

Week 2: Lecture 4 Relational Databases (1/28)

Overview

- Database is an organized collection of data
- Database Management System (DBMS): software system that stores, manages, and facilitates access to one or more databases

DBMS vs CSV

- Data Storage,
 - Reliable storage, optimize, performance

Relational DBMS Terminology

- In relational database, each table is called a relation
- Row of relation is called record or tuple
- Column of a relation is attribute or field
- Primary key must be unique, Foreign key references another table

SQL Query Syntax

```
SELECT [DISTINCT] <column expression list>  
FROM <list of tables>  
[WHERE <predicate>]  
[GROUP BY <column list>]  
[HAVING <predicate>]  
[ORDER BY <column list>]  
[LIMIT <number of rows>];
```



Cross join, joined all combinations

Inner Join can match id

```
SELECT * FROM s JOIN t ON s.id = t.id;
```

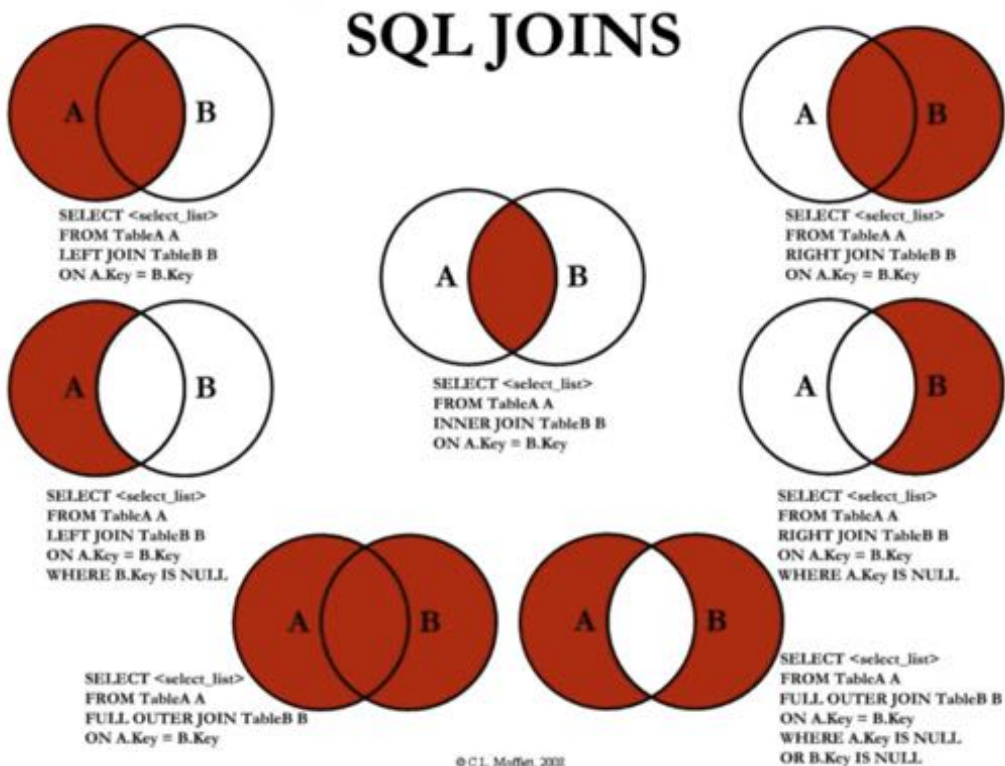
```
SELECT * FROM s INNER JOIN t ON s.id = t.id;
```

```
SELECT * FROM s, t WHERE s.id = t.id;
```

Left Outer Join: every row in first table appears in the result, matching or not

Right Outer Join: every row in second table appears in the result, matching or not

Full Outer Join: every row in both tables appears, matching or not



NULLs in comparators

- Check by IS or IS NOT

SQL Predicates and Casting

In addition to numerical comparisons (=, <, >) IN, LIKE

SQL Casting

- Cast data types

SQL Sampling

Random Sampling

- SELECT * FROM action_movie ORDER BY RANDOM() LIMIT 3
- Declarative language, what not how

Cluster sample

- SELECT * FROM action_movie
- WHERE year IN (SELECT ...)

Subqueries

- Query within another query used to create a temporary table
- In a FROM clause or in a WHERE clause

Common Table Expressions (CTE) create of temporary tables

- Make complex queries more readable
- WITH t2 as (...) can use t2 as variable

CASE Expressions

- CASE Expression chooses among alternative values

SUBSTR

- SUBSTR allows you to extract substrings

Week 3: Lecture 5 Pandas (2/2)

Introduction to Pandas syntax and operators

Data Frame

- From the world, we can take a statistical population to draw samples where each sample has certain features
- Create a DataFrame with rows of data and column with stasis

Data Frames, Series, and Indices

- Series is column types

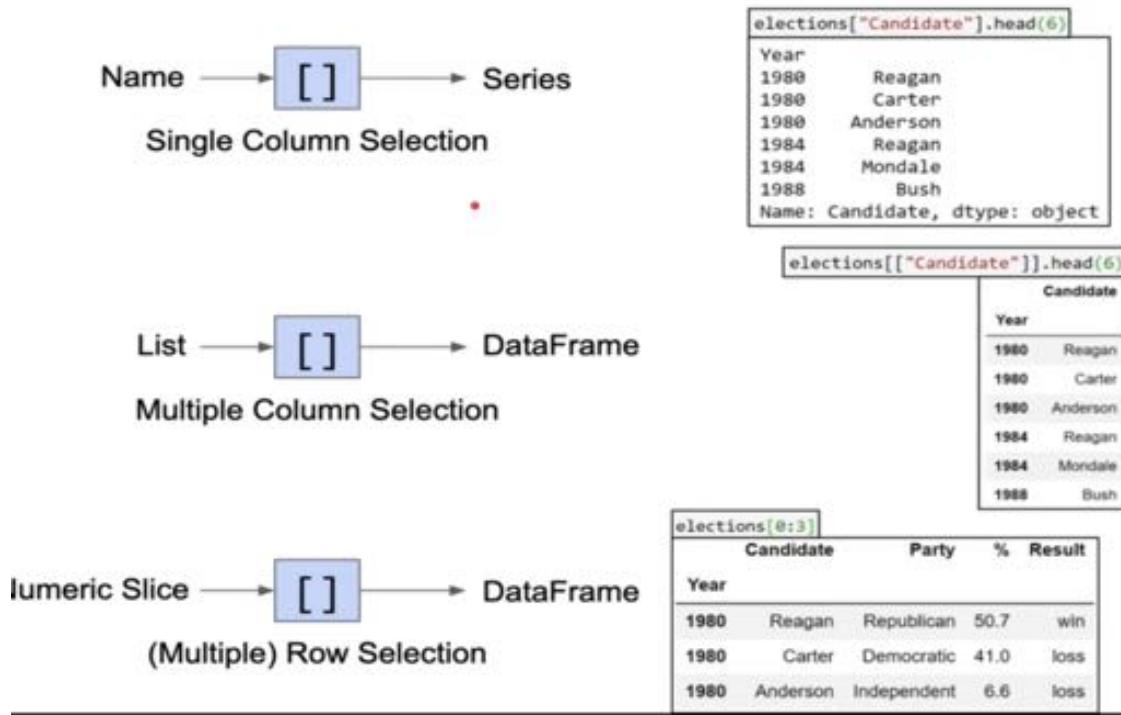
Structure of a Series

- Has internal indices starting from 0 for data
- Index and values

Example

- import pandas as pd
- elections = pd.read_csv("elections.csv")
- Methods
 - head(rows=5): shows first # rows

- tail(rows=5): shows last # rows
- [] operator
 - Series is more plain text
 - DataFrame has particular columns (elections[["Candidate", "Party"]])
 - to_frame(): change series to Data Frame
 - Elections["key"]
 - Elections[0:3] gives you multiple column selection
 -



pd.DataFrame({}): Can make DataFrame out of dictionary

Boolean Array Selection

- Can do boolean mask with [True, False, True, False] for each index in dataframe
- Can filter by: elections[(elections['Result'] == 'win' & (election['Party'] == 'Democrato'))]
- isin(lst) is value in set self
- query(str): specific string where names are columns and can use queried operations based on columns

Loc

Loc is label based object

- Elections.loc[[0,1,2,3,4], ['Candidate', 'Party', 'Year']]
 - Pulls out rows by index, and corresponding columns
- Know difference between index and internal index
- Slices in loc are inclusive on both ends
 - Elections.loc[0:4, 'Candidate':'Year']

Iloc

- Iloc slicing is exclusive
 - Elections[[0,3,5], [0,3]]

Sampling

- Sampling is without replacement and gives you random
- sample(num): number of random samples

Basics

- <data>.size
 - : length * columns
- <data>.shape
 - : rows, columns
- <data>.describe()
 - : count, unique, top, freq
- <data>.index
 - : gives you range of index like a list of the keys, has specific methods
- <data>.columns
 - : gives you columns besides the key column name
- <data>.sort_values(<col_name>)
 - Sorts by column name, can set ascending=True
- <data>[<col>].value_counts()
 - Counts of each value
- <data>[<col>].unique()

Week 3: Lecture 6 Pandas II Advanced (2/5)

Group of series from data frame when joined by indexes

%timeit <code>

- will tell you how long

<series>.str

- Allows you to use string methods

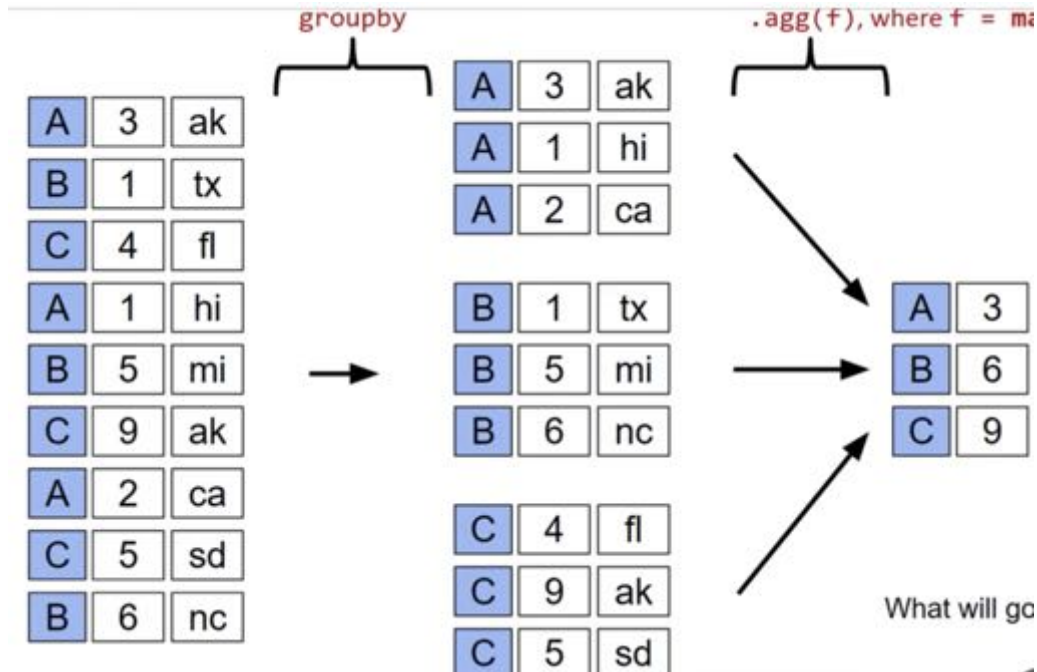
How to sort by length

1. Create a temporary column namelengths
2. Sort
3. Remove temp column

<data>.query(<string with columns>)

<data>.groupby(<Column>)

- Returns similar to dictionary where it combines with the same column
- <groupby>.agg(<func>)
 - Applies function to each next column



Can sort first then `agg(lambda x : x.iloc[0])`

`<idxmax()>` tells you index of max

- `elections.loc[elections.groupby('Party')['%'].idxmax()]`
- `<dataframe>.drop_duplicates([<col_str>], keep='last')`

`groupby.filter(lambda sf: sf["num"] > 6)`

`groupby.size()`

- Size of each group

`display(<var>)`

- Can show more pretty data

`groupby.sum()`

- Same as `agg(sum)`

`groupby.max(), min, median`

`groupby(["year", "Sex"])`

- multi-indexed, index has multiple dimension

`<dataframe>.merge(df2, how="inner", left_on="Candidate", right_on="year")`

Week 4: Lecture 7 Data Cleaning & Exploratory Data Analysis (2/13)

Understanding the Data

- Data Cleaning, Exploratory Data Analysis,

Data Cleaning

- The process of transforming raw data to facilitate subsequent analysis
- Data Cleaning often addresses issues

- structure/formatting, missing or corrupted values, unit conversion, encoding text as numbers
- Data cleaning is a big part

Exploratory Data Analysis

- "Getting to know the data"
- Transforming, visualizing, and summarizing data to
 - build/confirm understanding of the data and its provenance,
 - Identify and address potential issues in the data
 - Inform the subsequent analysis
 - Discover potential hypothesis
- EDA is an open-ended analysis
- John Tukey, FFT,
- Exploratory data analysis is attitude, state of flexibility, a willingness to look for those things that we believe are not there

File Formats and structures

- Structure: shape of a data file
- Granularity: how fine/coarse is each datum
- Scope: how (in) complete is the data
- Temporality
- Faithfulness

Structure Data

- Rectangular data,
 1. Tables
 - a. Named columns with different types
 - b. Manipulated using data transformation languages
 2. Matrices
 - a. Manipulated using linear algebra functions

Data file formats

- Tab separated values (TSV), Comma separated values (CSV), JSON

CSV and TSB

- Comma and Tab separated values Files
- Records delimited by a newline: "\n", "\r\n"

JavaScript Object Notation (JSON)

- Strict quoting formatting
- Not rectangular

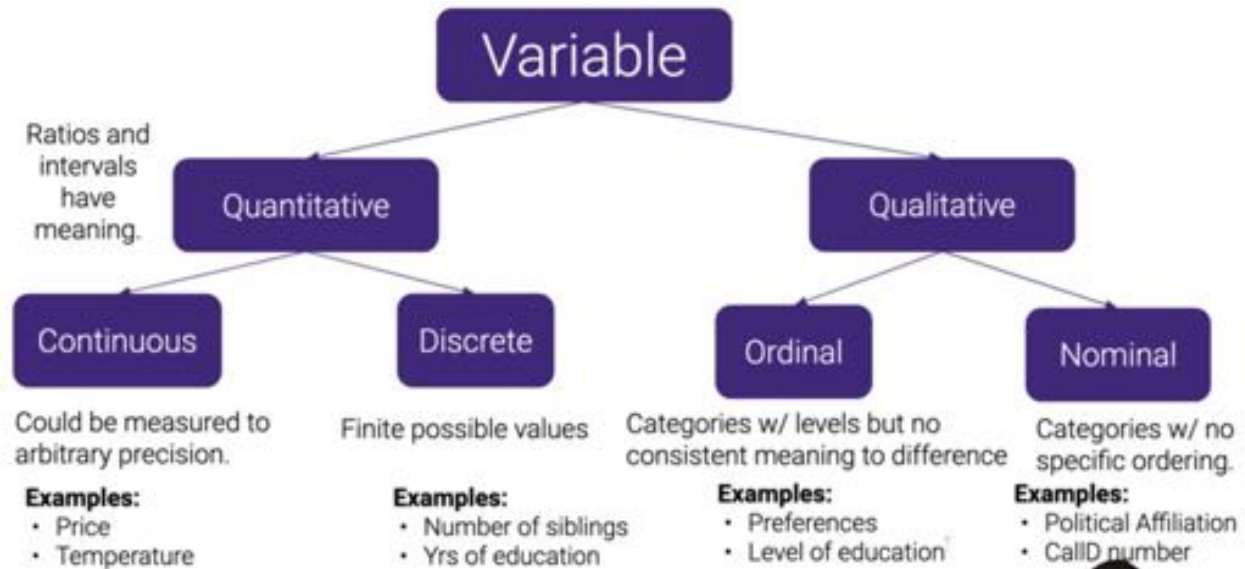
Keys and Joins

- Primary Key: column or set of columns in a table that determine the values of the remaining columns
 - Primary keys unique

- Foreign keys: the columns or sets of columns that reference primary keys of other tables
 - Allows you to do joins

Variable

- Quantitative vs qualitative



Granularity

Fine grained: each individual

Coarse Grained: Group of people

Scope

- Does Data cover my area of interest
- Is my data too expansive
 - Filtering? Poor coverage

Revisiting the sampling frame

The sampling frame is the population from which the data was sampled

How complete/incomplete is the frame

Temporality

- Data Changes over time, when was the data collected
- When the "event" happened
- Time depends on where!
 - Datetime python library

Faithfulness

Do I trust this data?

- Unrealistic or incorrect values
- Data falsification, entered by hand
- Missing Values/Default values
- What to do with Missing Values

- Drop records, check for biases introduced by dropped values
- Infer missing values,

Possibly not faithful

- Missing values, default values
- Truncated data
- Time zone inconsistencies
- Duplicated records or fields
- Spelling errors
- Units not specified or consistent

How to do EDA

1. Examine data and metadata
2. Examine each field/attribute/dimension individually
3. Examine pairs of related dimensions
4. Visualize/summarize the data
 - a. Validate assumptions about data
 - b. Record everything you do

Week 4: Lecture 8 Regular Expressions (2/13)

String Canonicalization

Joining Tables with Mismatched Labels

- Canonicalize: convert data that has more than one possible presentation into a standard form
- Can use lower and replace

Splitting String

Can split at certain places but it is brittle

Regular Expressions

Regular Language

- Formal language that can be described by regular expressions


operation	order	example	matches	does not match
concatenation	3	AABAAB	AABAAB	every other string
or	4	AA BAAB	AA BAAB	every other string
closure (zero or more)	2	AB*A	AA ABBBBBBA	AB ABABA
parenthesis	1	A(A B)AAB	AAAAB ABAAB	every other string
		(AB)*A	A ABABABABA	AA ABBA

- concatenation/or/closure/parenthesis

operation	example	matches	does not match
any character (except newline)	.U.U.U.	CUMULUS JUGULUM	SUCCUBUS TUMULTUOUS
character class	[A-Za-z][a-z]*	word Capitalized	camelCase 4illegal
at least one	jo+hn	john joooooooohn	jhn jjohn
zero or one	joh?n	jon john	any other string
repeated exactly {a} times	j[aeiou]{3}hn	jaoehn jooohn	jhn jaeiouhn
repeated from a to b times: {a,b}	j[ou]{1,2}hn	john juohn	jhn jooohn

- Can be difficult to debug

operation	example	matches	does not match
built-in character classes	\w+ \d+	fawef 231231	this person 423 people
character class negation	[^a-z]+	PEPPERS3982 17211!↑å	porch CLAmS
escape character	cow\.com	cow.com	cowscom

operation	example	matches	does not match
beginning of line	^ark	ark two ark o ark	dark
end of line	ark\$	dark ark o ark	ark two
lazy version of zero or more *?	5.*?5	5005 55	5005005 

Regular expressions in Python

Python regex

- `re.findall(pattern, text)` : returns list of instances
- `re.sub(pattern, "", text)` : substitutes pattern
- `r"ab*"` : reduce number of backslashes

Regular Expression Groups

- Parentheses used to specify a so-called group
- Python, they are returned as tuples
- Parentheses to specify groups

basic python	re	pandas
	<code>re.findall</code>	<code>df.str.findall</code>
<code>str.replace</code>	<code>re.sub</code>	<code>df.str.replace</code>
<code>str.split</code>	<code>re.split</code>	<code>df.str.split</code>
<code>'ab' in str</code>	<code>re.search</code>	<code>df.str.contains</code>
<code>len(str)</code>		<code>df.str.len</code>
<code>str[1:4]</code>		<code>df.str[1:4]</code>

Week 5: Lecture 9 Visualization I (2/16)

Visualization

- Use of computer generated interactive, visual representations of data to amplify cognition, finding artificial memory that supports means of perception

Visualize, then quantify

- Can have the same statistics but looks different

Goals of data visualization

1. Help your own understanding of data/results
 - Key part of exploratory data analysis, modeling, flexible
2. Communicate results/conclusions to others
 - Highly editorial selective
 - Be thoughtful and careful
 - Fine tuned to achieve a communications goal, design, art

Why data visualization?

- Plots help inform human decisions
- Sometime most useful results are visualization
- Many plots possible but few are useful

Encoding

- Mapping from a variable to a visual element
- Not all encoding channels are exchangeable

Distribution

- Frequency at which values of a variable occur
- All values must be accounted for once, and only once

Bar Plot

- Most common way of displaying distribution of a qualitative (categorical) variable
- Lengths encode values
- Color could be a sub-category

Plotting

1. Matplotlib (plt)
 - a. `plt.bar(<categories>, <height>)`
2. Pandas (.plot())
 - a. `<Series>.plot(kind="bar")`
3. Seaborn (sns)
 - a. `sns.barplot(x=<categories>, y=<heights>)`
 - b. More statistics

Data visualizations examples

Rug plot

- Single quantitative variable
- Show each and every value
- Hard to see the bigger picture

Histogram

- Smoothed version of a run plot
- Areas represent proportions
- Horizontal axis: the number line, divided into bins
- Height: proportion per unit on the x-axis
- Proportion in bin = width of bin * height of bar
- Optimal bar width:

The Freedman-Diaconis rule:

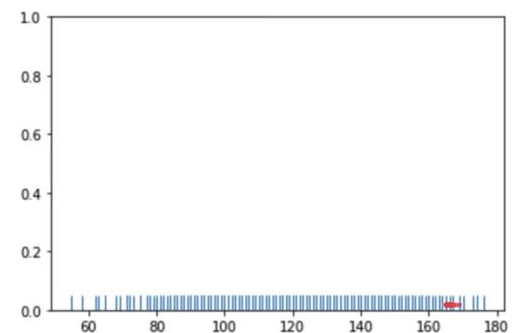
$$\text{Bin width} = 2 \frac{\text{IQR}(x)}{\sqrt[3]{n}}$$

Density curve

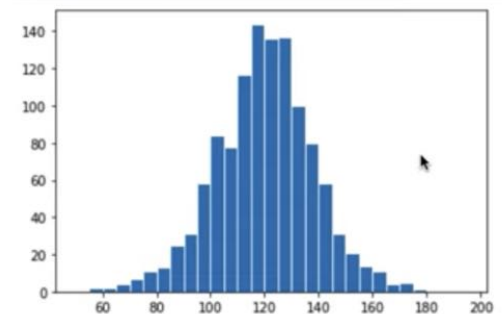
- `sns.kdeplot(weights)`

Statistics

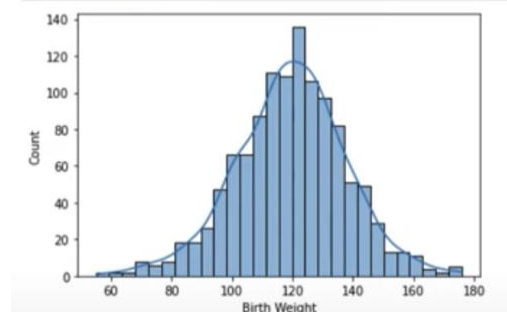
- Modes
 - Distribution is a local or global maximum



```
plt.hist(bweights, bins=bw_bins, ec='w');
```



```
sns.histplot(bweights, kde=True);
```

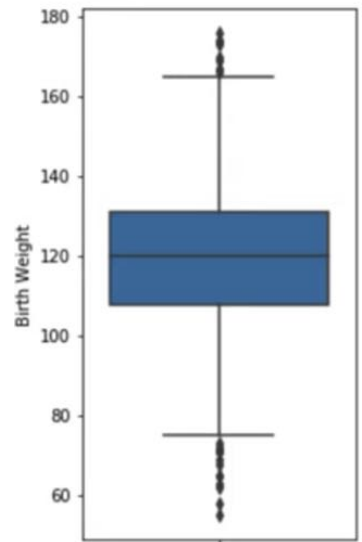


- Need to distinguish between modes and random noise
- Bimodal or single unimodal

- Skew and Tails
 - Long right tail, skewed right
 - Long left tail, skewed left
 - Median, area is equal on both sides

Box Plots

- Quartiles
 - First/lowe quartile 25th percentile
 - Second quartile: 50th percentile (median)
 - Third or upper quartile: 75th percentile
 - [first quartile, third quartile] contains the middle 50% of the data
 - Interquartile range measures spread
 - $IQR = \text{third quartile} - \text{first quartile}$

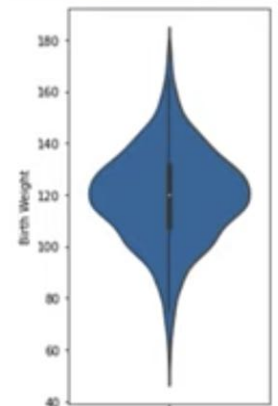


- Box Plots
 - Lower quartile
 - Median
 - Upper quartile
 - "Whiskers" $1.5 * IQR$
 - Outliers

Violin plots

- Show smoothed density curves
- Width of our box now has meaning
- 3 quartiles and whiskers are still present

```
plt.figure(figsize = (3, 6))
sns.violinplot(y=bweights);
```

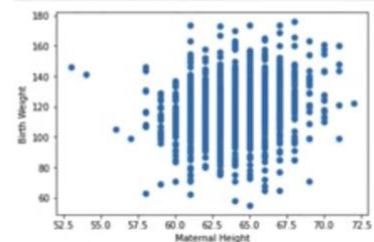


Relationships between Quantitative variables

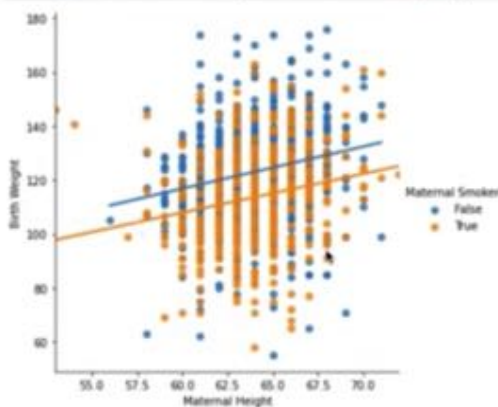
Scatter plots

- Reveal relationship between pairs of numerical data
- Seaborn has many more capabilities than matplotlib
- Can fit line of best fit and separate by hue

```
plt.scatter(data=births, x='Maternal Height', y='Birth Weight');
plt.xlabel('Maternal Height')
plt.ylabel('Birth Weight');
```



```
sns.lmplot(data = births, x = 'Maternal Height', y = 'Birth Weight', ci=False, hue='Maternal Smoker')
```



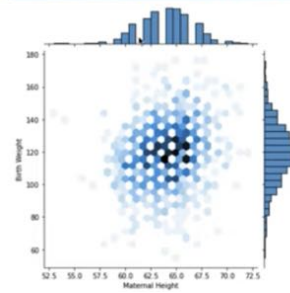
Hex Plots

- Thought of as two dimensional histogram,
- x y plane binned into hexagons
- Darker is more dense

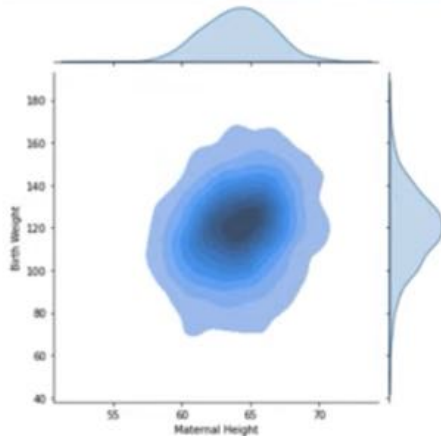
Contour plot

- Two dimensional density curves
- Gradient descent

```
sns.jointplot(data = births, x = 'Maternal Height', y = 'Birth Weight', kind='hex');
```



```
sns.jointplot(data = births, x = 'Maternal Height', y = 'Birth Weight', kind='kde',
```



Summary

- Visualization requires a lot of thought
- Types of variables constrain the charts that you can make
 - Single quantitative: rug plot, histogram, density plot
 - Two quantitative: scatter plot, hex plot, contour plots
 - Combination: bar plot, overlaid histograms/density plots, SBS box/violin

Week 5: Lecture 10 Visualization II (2/19)

Principle of Scale

- Make sure scales are the same
- Do not use two different scales for the same axis
- Can use percentage, normalizing with total services

Reveal the data

- Choose axis limits to fill the visualization
- Zoom in on the bulk of the data, show multiple plots to show different regions of interest

Principle of Conditioning

Use conditioning to aid comparison

- Lines make it easy to see the large effect

Juxtaposition: putting multiple plots side by side with the same scale

Superposition: placing multiple density plots, scatter on top of each other

Perception of Color

Choosing set of colors which work together is a challenging task

Colormaps

- Mapping according to colors, jet/rainbow colormap actively misleads
- Use perceptually uniform colormaps perceptual change is the same as when the data color difference is
- Use turbo if you have to perceptually uniform

Perception of Markings

Lengths are easy to distinguish, angles are hard

- Don't use pie charts
- Areas are harder to distinguish
- Avoid word clouds

Avoid jiggling the baseline

- Harder to calculate inter group comparisons

Overplotting, want to make sure you can still see all the data

- Can make more transparent

Context

Make sure has informative title, axis labels, reference lines, captions, legends

Kernel Density Estimation

Histograms are smoothed version of rug plot

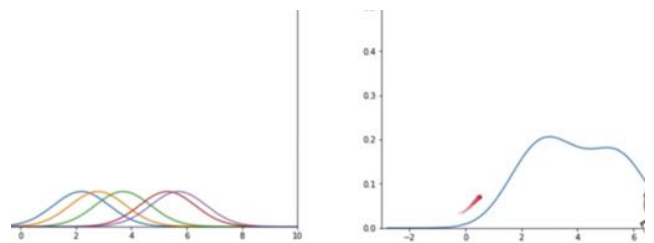
Estimate probability density function from a set of data

- Place a kernel at each data point, normalize kernels so that total area = 1
- Sum all kernels together

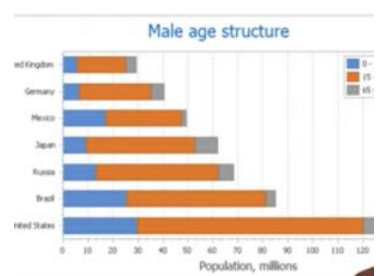
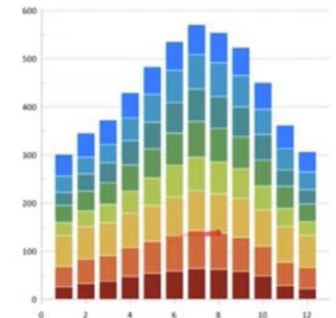
Kernel is valid density function

- Most use gaussian function for each one

$$K_{\alpha}(x, x_i) = \frac{1}{\sqrt{2\pi\alpha^2}} e^{-\frac{(x-x_i)^2}{2\alpha^2}}$$



Alpha is bandwidth parameter that controls smoothness of the kernel, hyperparameter, choose what to set it to



$$f_{\alpha}(x) = \frac{1}{n} \sum_{i=1}^n K_{\alpha}(x, x_i)$$

The "KDE formula" is above.

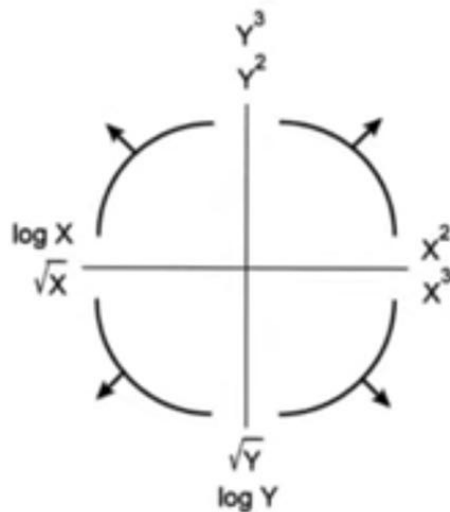
- x represents any number on the number line. It is the input to our function.
- n is the number of observed data points that we have.
- Each x_i (x_1, x_2, \dots, x_n) represents an observed data point. These are what we use to create our KDE.
- α is the bandwidth or smoothing parameter.
- $K_{\alpha}(x, x_i)$ is the kernel centered on the observation i .
 - Each kernel individually has area 1. We multiply by $1/n$ so that the total

Can extend kernel density estimation to two dimensions

Transformations

Can be useful to take a log if data is very spread out

Tukey-Mosteller Budge diagram



Summary

- Choose appropriate scales
- Condition in order to make comparisons more natural
- Choose colors and markings that are easy to interpret correctly
- Add context and captions that help tell the story
- Reveal the data, eliminate unrelated

Week 6: Lecture 11 Modeling (2/23)

Data Wrangling

- Filtering rows, columns, aggregation, pivot tables, string method, joins

Data Visualization

- Allow you to understand structure of data
 - Communicating results to the audience
1. Think about your data
 2. Think about your model

Model

- A model is useful simplification of reality
- George Box: Essentially all models are wrong but some are useful
- To understand the world we live in

Physical Models

- Well established theories of how the world works

Statistical models

- Based upon observation and data

Modeling process: 3 steps

1. Choosing your model
 - a. Constant, linear, non linear model
2. Choose Objective Model
 - a. Prediction: loss function
 - b. Description: likelihood function
3. Fitting model by optimizing your objective function
 - a.

Estimation

- Using data to determine the model parameters

Prediction

- Using fitted model parameters to predict unseen data

Loss Functions

- Quantify how bad a prediction is for a single observation: Cost of doing business (making predictions)
- Metric of how good or bad our predictions are
- Typically actual - predicted : error
- Squared loss L2 loss: $(y - \hat{y})^2$
- Absolute loss L1 loss: $|y - \hat{y}|$

Loss functions and empirical risk

- Average loss across all points, empirical risk, objective function
 - $\frac{1}{n} \sum_{i=1}^n L(y_i, \hat{y}_i)$
- Objective function wasn't to minimize
- Mean squared error (MSE)
- Mean absolute error (MAE)
- Minimize the loss function

$$\hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n (y_i - \theta)^2$$

-
- Value of theta that minimizes the loss
- Exploring MSE

Minimizing MSE

- Substituting \bar{y} , the value of mean squared error is the sample variance

MSE vs MAE

- Mean Squared error:
 - Very smooth:
 - Very sensitive to outliers
- Mean absolute error
 - Not as smooth
 - Robust to outliers

Summary: The modeling process

1. Choose a model
 - a. Parameters define the model
2. Choose a loss function
 - a. L2 (squared) loss and L1 absolute loss, choose
3. Fit the model by minimizing average loss
 - a. Choose optimal parameters that minimize average loss across entire data set, fitting the model to the data
 - b. Different loss functions lead to different optimal parameters

Week 6: Lecture 12 Modeling (2/26)

Simple Linear Model

- Better prediction

Minimizing MSE for Linear

- Substitute

$$R(a, b) = \frac{1}{n} \sum_{i=1}^n (y_i - (a + bx_i))^2$$

Take partial derivatives with respect to both parameters (a, b) set them to 0 and solve

Model interpretation

Slope = $r \cdot \text{sigmay} / \text{sigmax}$

CONSTANT MODEL

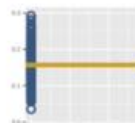
$$\hat{y} = \theta$$

Useful

- Desc: average tip
- Pred: $MSE = .0021$

Simple

- 1 parameter



SIMPLE LINEAR MODEL

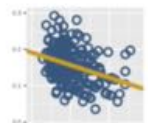
$$\hat{y} = \theta_0 + \theta_1 x$$

Useful

- Desc: negative assoc.
- Pred: $MSE = ?$

Simple

- 2 parameters



- Measures increase in y for 1 unit increase in x

Model shows association, not causation

- Estimated difference

New data needs to be similar to original data

- New data we test our model on looks nothing like the data we fit our model on, there's no guarantee that it will be any good

Visualize, then quantify

- Before modeling you should always visualize your data first

Multiple Linear regression

Terminology

- X: feature, covariate, independent var, predictor, input, regressor
- Y: output, outcome, response, dependent variable

Adding independent variables

- 2 features leads to 3 parameters
- 3d visualization

Multiple linear regression model

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \dots + \theta_p x_p = \theta_0 + \sum_{j=1}^p \theta_j x_j$$

Constant model: $f_{\theta}(x) = \theta$

Simple linear regression model: $f_{\theta}(x) = \theta_0 + \theta_1 x$

Multiple linear regression model: $f_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_p x_p$

Root Mean Square Error

- Square root of MSE

$$\text{RMSE}(y, \hat{y}) = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

- Impossible for RMSE to go up by adding features

R² is proportion of variance in our true y that our fitted values capture "proportion of variance that the model explains"

$$R^2 = \frac{\text{variance of fitted values}}{\text{variance of } y} = \frac{\sigma_{\hat{y}}^2}{\sigma_y^2}$$

As we add more feature, our fitted values tend to become closer and closer to our actual y values. R² increases

Adding features doesn't always mean our model is better though

Week 7: Lecture 13 Linear Regression (3/2)

Fit model by optimizing your objective function

$$\hat{\theta}_0 = \bar{y} - \hat{\theta}_1 \bar{x}; \quad \hat{\theta}_1 = r \frac{SD(x)}{SD(y)}$$

Can be stated as a dot product between two vectors

Design matrix

- Rows are observations
- Columns are features

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_p \end{bmatrix} \quad x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}$$

$$\hat{y} = x^T \theta$$

scalar

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & x_{23} & \dots & x_{2p} \\ 1 & x_{31} & x_{32} & x_{33} & \dots & x_{3p} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} & x_{n3} & \dots & x_{np} \end{bmatrix}$$

Rows correspond to **observations**.
e.g. all features for data point 3

Columns correspond to **features**.
e.g. feature 1, for all data points

$$\hat{Y} = X\theta$$

Vector with dimensions **n x 1**.

Matrix with dimensions **n x (p + 1)**.

Vector with dimensions **(p + 1) x 1**.

Linear model on entire dataset

Predicted response vector

Vector norms

- L2 : square root of the sum of squares of the values (magnitude)
 - "Length" of a vector
- L1 : sum of absolute value of all the values

Residual

- difference between an actual and predicted value, in the regression context
- $E = y_i - \hat{y}_i$
- MSE of a model equal to mean of the squares of residuals
- Can create Residual vector

$$e_i = y_i - \hat{y}_i$$

odel is equal to the m

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n e_i^2$$

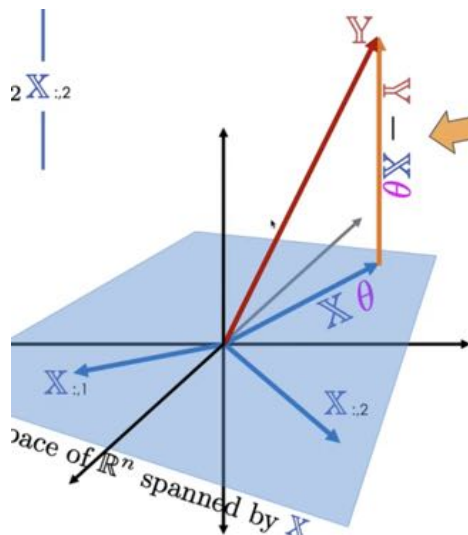
$$e = Y - \hat{Y} = \begin{bmatrix} y_1 - \hat{y}_1 \\ y_2 - \hat{y}_2 \\ \vdots \\ y_n - \hat{y}_n \end{bmatrix}$$

$$R(\theta) = \|Y - X\theta\|_2^2$$

Find value of theta that minimizes the squared L2 norm of the residual vector

Span

- Set of all possible linear combinations of columns of X (column space)
- All the vectors you can reach using columns of X, find vector in span(X) that is closest to Y



Recall, this is residual vector $e = Y - \hat{Y}$.

Our goal is to minimize the norm of the residual vector i.e. we want our predictions to be "as close" to the true y values as possible.

Vector closest is the vector projected onto span(X) orthogonal projection

Orthogonality

- Orthogonal iff dot product is 0
- If vector orthogonal to span(X), dot prod $mv = 0$ is orthogonal to all the vectors

Least squares regression, minimize avg loss

$$\hat{\theta} = (X^T X)^{-1} X^T Y$$

Residual plots

- Simple linear regression, plot residuals vs x
- Residual plot is of residuals vs fitted values

When model has intercept term

- Sum of residuals is 0, mean is also 0
- Avg true y value is equal to avg predicted y value

At least one solution always exists

- Unique solution exists when $X^T X$ is invertible
- $X^T X$ is invertible if and only if it is full rank
 - Each column is linearly independent
- $X^T X$ have the same rank
- Maximum possible rank of $X^T X$ is $p + 1$
- X^T is invertible iff X has rank $p + 1$ (full column rank)
-

Week 8: Lecture 14 Feature Engineering (3/11)

Features are the x_j

Feature Engineering

- Enables you to capture domain knowledge
- Express non-linear relationships
- Encode non-numeric features

Feature functions

- Transform features into new features

$$\hat{y} = f_{\theta}(x) = \sum_{j=0}^p \phi(x)_j \theta_j$$

(phi)ture func

SciKit,

- Fit model and predict, can use packages, variables to find error

Linear model in terms of periodic

- Can add features depending on model, sin, periodic sin
- Add nonlinear features

Encode cartegorical data

- Categorical data used as dummy encoding to have magnitude for each dimension
- First fit dummy with OneHotEncoder

Bag of Words Encoding

- CountVectorizer
- Separate columns for each word
- Remove stop words, loses word order

N gram encoding

- Encode clusters of words, keeps counts of phrases

Week 9: Lecture 15 Bias and Variance (3/16)

Overfitting and Generalization

Building models can create a model overfit on the data

Variance

Random Variables

- Numerical function of a random sample
- Expectation
 - Weighted average of values of X, where weights are probabilities of the values
 - Linearity of Expectation

Definition of variance

- Variance is expected squared deviation from expectation of X
- $Var(X) = E((X - E(X))^2) = E(X^2) - E(X)^2$
- Standard deviation $SD(X) = \sqrt{Var(X)}$
- Chebyshev's inequality
 - Vast majority of probability lies in the interval "expection plus or minus a few SDs"
 - $\mu = E[X], \sigma = SD[X], \text{ then } P(|X - \mu| \geq k\sigma) \leq \frac{1}{k^2}$

LInearity

- $E(aX + b) = aE(X) + b$
- $Var(aX + b) = Var(aX) = a^2Var(X)$
- $SD(aX + b) = |a|SD(X)$

Standarization of random variables

- X in standard units: $X_{su} = \frac{X - E(X)}{SD(X)}$

- Measures X on the scale "number of SDs from expectation, linear transformation

Data Generating Process and Model Risk

Assumptions of Randomness

-

Data Generation Process

- True relation g , for each individual fixed value of x and $g(x)$,
- Random error ϵ
- Observation $Y = g(x) + \epsilon$
- Predicted model $error = Y - \hat{Y}(x)$

Model Risk

- Mean squared error of prediction
 - Model risk = $E((Y - \hat{Y}(x))^2)$

Kinds of errors

- Chance error
 - Randomness alone
- Bias
 - None random error, model different from true function

Components of Prediction Error

Bias

$$g(x) - E(\hat{Y}(x))$$

Decomposition of the prediction error into three pieces:

$$Y - \hat{Y}(x) = \epsilon + (g(x) - E(\hat{Y}(x))) + (E(\hat{Y}(x)) - \hat{Y}(x))$$

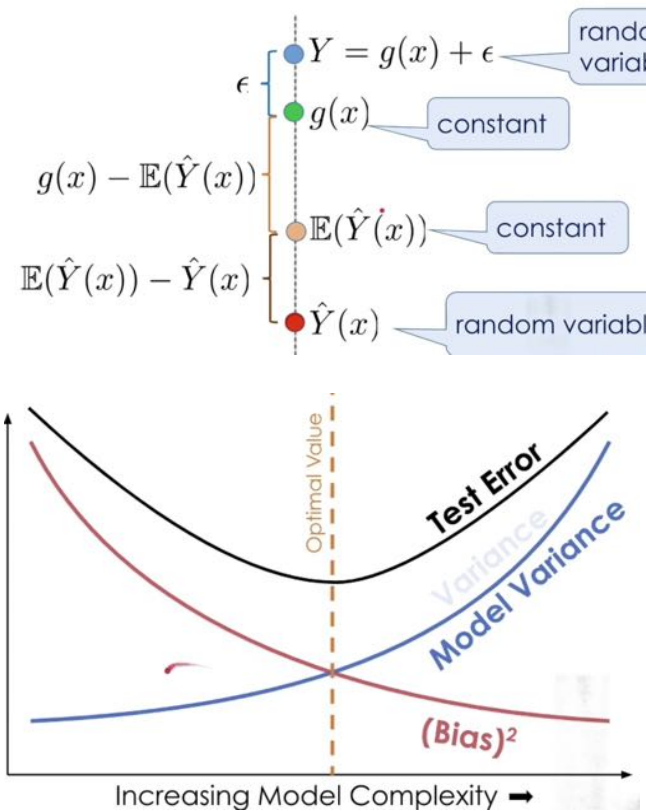
Decomposition of the model risk into three pieces:

$$E((Y - \hat{Y}(x))^2) = E(\epsilon^2) + (g(x) - E(\hat{Y}(x)))^2 + E((E(\hat{Y}(x)) - \hat{Y}(x))^2)$$

The cross-product terms are

$$\text{model risk} = \sigma^2 + (\text{model bias})^2 + \text{model variance}$$

Right model structure matters



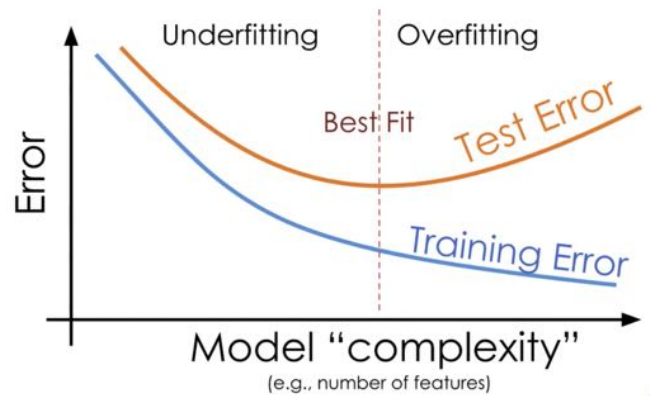
Week 9: Lecture 16 Cross-Validation and Regularization (3/18)

Training Error vs Test Error

- Training loss underestimates real loss
- Increasing complexity of model increases test error

Train=Test Split

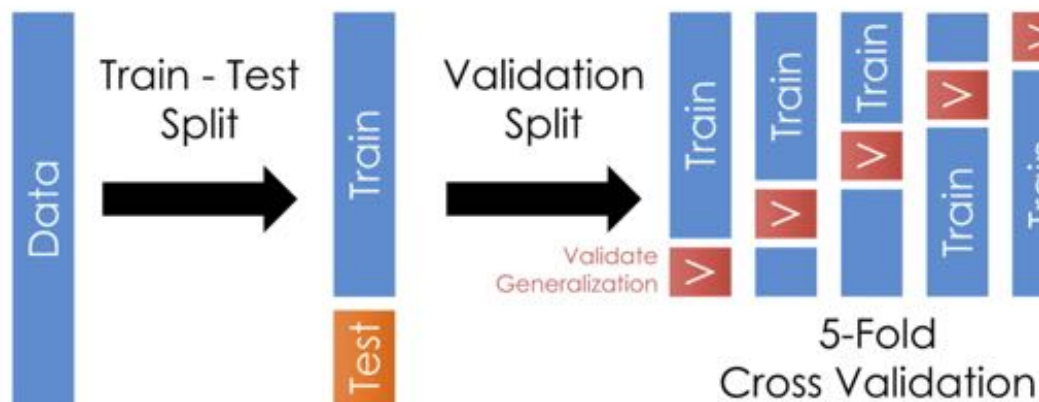
- Training Data: used to fit model
- Check generalization error
- How to split?
 - Randomly, Temporally, Geo
- What side
 - Larger training set -- more complex models
 - Larger test set -- better estimate of generalization errors
 - Between 75%-25%, 90%-10%
- You can only use the test dataset once you decide on the model



Generalization Validation Split

- Use train and have Validation split

Generalization: Validation Split



Cross validation **simulates multiple train test-splits** on the training data.

1. Split your data into training and test sets (90%, 10%)
2. Use only the training data when designing, training, and tuning the model
 - a. Use cross validation to test generalization during this phase
 - b. Do not look at the test data
3. Commit to final model and train once more using only the training data
4. Test the final model using the test data, if accuracy is not acceptable return to 2
5. Train on all available data

Regularization

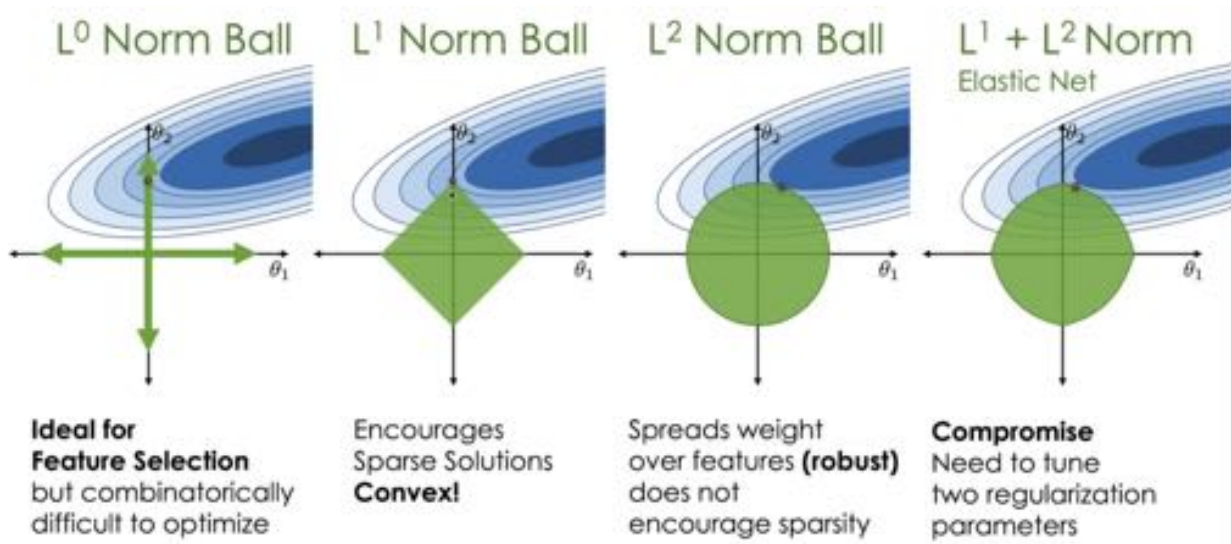
- (parametrically Controlling the model complexity)

$$\hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n \text{Loss}(y_i, f_{\theta}(x_i))$$

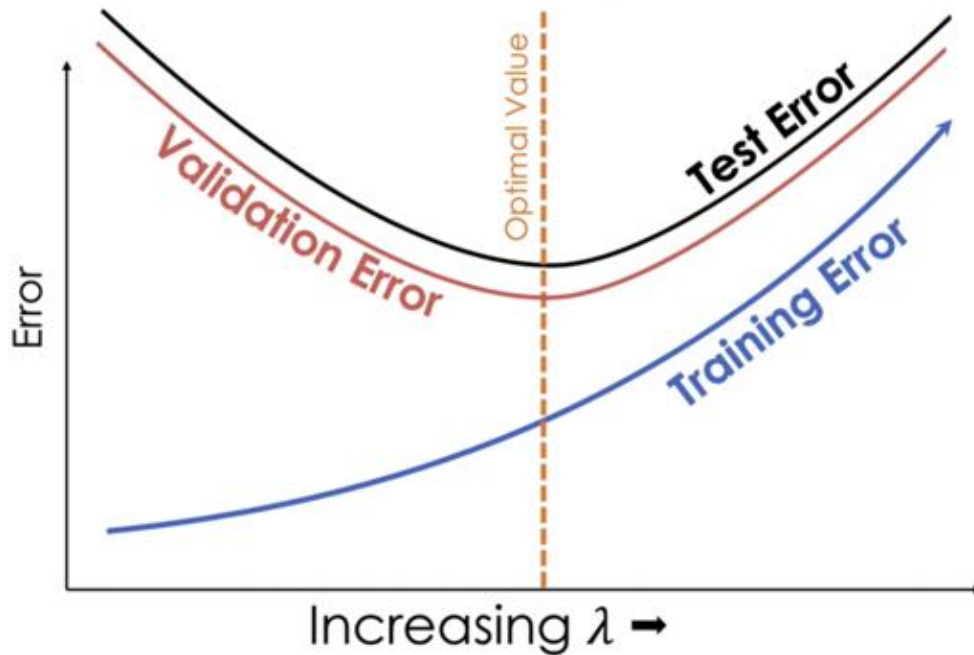
-
- So Complexity(f_{θ}) $\leq \beta$
- Aka number of features
- Want approximation since it would be NP complete

→ θ_1 Convex approximation!

$$\text{Complexity}(f_{\theta}) = \sum_{j=1}^d |\theta_j| \leq \beta$$



Value of λ determines bias-variance tradeoff
Determined through validation



- Standardization and Intercept Term
- Height = $\theta_1 \text{age_in_seconds} + \theta_2$

Ridge Regression

- Term for following specific combination of model, loss and regulation
- Model: $Y = X\theta$
- Loss: Squared Loss
- Regularization: L2 regularization
- Objective function we minimize for ridge regression

$$\hat{\theta}_{\text{ridge}} = \arg \min_{\theta} \frac{1}{n} \|Y - X\theta\|_2^2 + \lambda \sum_{j=1}^d \theta_j^2$$

Ridge Regression

- Closed form solution

$$\hat{\theta}_{\text{ridge}} = (X^T X + n\lambda I)^{-1} X^T Y$$

Identity matrix

- Always has unique optimal parameter vector for ridge regression

LASSO Regression

- Model: $Y=X\theta$
- Loss: squared loss
- Regularization: L1 regularization

Objective function minimize for LASSO is squared loss plus added penalty

$$\hat{\theta}_{\text{LASSO}} = \arg \min_{\theta} \frac{1}{n} \|Y - X\theta\|_2^2 + \lambda \sum_{j=1}^d |\theta_j|$$

Name	Model	Loss	Reg.	Objective	Solution
OLS	$\hat{Y} = X\theta$	Squared loss	None	$\frac{1}{n} \ Y - X\theta\ _2^2$	$\hat{\theta}_{\text{OLS}} = (X^T X)^{-1} X^T Y$
Ridge Regression	$\hat{Y} = X\theta$	Squared loss	L2	$\frac{1}{n} \ Y - X\theta\ _2^2 + \lambda \sum_{j=1}^d \theta_j^2$	$\hat{\theta}_{\text{ridge}} = (X^T X + n\lambda I)^{-1} X^T Y$
LASSO	$\hat{Y} = X\theta$	Squared loss	L1	$\frac{1}{n} \ Y - X\theta\ _2^2 + \lambda \sum_{j=1}^d \theta_j $	No closed form

Use root mean squared error to evaluate models performance

Tune features

Week 11: Lecture 17 Modeling in Context: Fairness in Housing (3/30)

Cook county Assessor's Office

The Problem

Unfair burden, failed to value homes accurately for years, property tax system that harmed poor and helped the rich

- Has appeals but only the rich do appeal
- Redlining: making it difficult or impossible to get a federally-backed mortgage to buy a house in t specific neighborhoods coded as risky
- Real estate industry encoded race as a factor of valuation

The Solution

- Created new Office of Data Science
- Distributional equity in property taxation = properties
- Separate models for each neighborhood, more granular

Key Takeaways

1. Accuracy is a necessary but not sufficient, condition of a fair system
2. Fairness and transparency are context-dependent

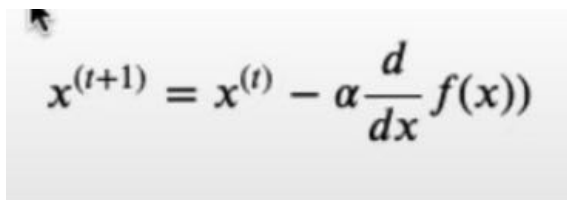
3. Learn to work with contexts, and consider how your data analysis will reshape them
4. Keep in mind the power, and limits of data analysis

Lessons for Data Science Principles

1. Question/Problem Formulation
 - a. Who is responsible for framing the problem?
 - b. Stakeholders, how involved in problem framing?
 - c. What do you bring to the table?
 - d. What are the narratives that you're tapping into?
2. Data Acquisition and Cleaning
 - a. Where does the data come from, who collected it
 - b. What kinds of collecting and recording systems and techniques were used
 - c. How does data used in the past
 - d. What restrictions on accessing to the data
3. Exploratory Data Analysis & Visualization
 - a. What kind of personal or group identities have become salient in this data?
 - b. Which variables salient
 - c. Do relationships make visible lend themselves to arguments that can be harmful
4. Prediction and Inference
 - a. What does the prediction or inference do in the world
 - b. Are results useful for purposes
 - c. Are there benchmarks to compare the results
 - d. How are your predictions and inferences dependent upon the larger system in which your model works

Week 11: Lecture 18 Gradient Descent (4/1)

User derivative to determine how far to move next point



$$x^{(t+1)} = x^{(t)} - \alpha \frac{d}{dx} f(x)$$

α : learning rate

- Too large and algorithm fails to converge
- Too small and takes too long to converge

Different starting points and learning rates can lead to local minimums

Convexity

- For a convex function, any local min is also a global minimum
- F is convex if

$$tf(a) + (1 - t)f(b) \geq f(ta + (1 - t)b)$$

For all a, b in domain of f and $t \in [0, 1]$

Goal

Goal is to find the min MSE

$$\hat{y} = \frac{\sum(x_i y_i)}{\sum(x_i^2)}$$

1. Using calculus the optimal answer is
2. We can also Plot the MSE vs \hat{y} choices and graphically view it
3. Gradient Descent
 - a. Need to calculate derivative of curve and pass into gradient descent
4. Scipy.optimize.minimize
5. sklearn.linear_model.LinearRegression

Gradient descent in 2D

- Need to calculate partial derivatives to be able to find minimums as a vector

Next value for θ

$$\vec{\theta}^{(t+1)} = \vec{\theta}^{(t)} - \alpha \nabla_{\vec{\theta}} L(\vec{\theta}, \mathbb{X}, \vec{y})$$

Gradient of loss wrt θ

Learning rate

Gradient Descent Algorithm

1. Initialize model weights to all zero
2. Update model weights using update rule
3. Repeat until model weights don't change (convergence)

$$\nabla_{\theta} \ell(\vec{\theta}, \vec{x}, y_i) = \begin{bmatrix} -2(y_i - \theta_0 x_0 - \theta_1 x_1)(x_0) \\ -2(y_i - \theta_0 x_0 - \theta_1 x_1)(x_1) \end{bmatrix}$$

Stochastic Gradient Descent

Loss function is really the average loss over a large dataset

$$\nabla_{\theta} \mathbf{L}(\theta) \Big|_{\theta=\theta(\tau)} = \frac{1}{n} \sum_{i=1}^n \nabla_{\theta} \text{loss}(y, f_{\theta}(x)) \Big|_{\theta=\theta(\tau)}$$

Want to sample to quickly find it

Stochastic Gradient Descent

$$\theta^{(0)} \leftarrow \text{initial vector (random, zeros ...)}$$

For τ from 0 to convergence:

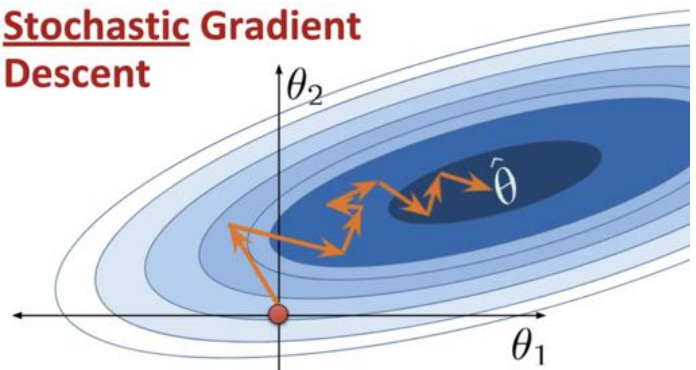
$$\mathcal{B} \sim \text{Random subset of indices}$$

$$\theta^{(\tau+1)} \leftarrow \theta^{(\tau)} - \alpha \left(\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla_{\theta} \mathbf{L}_i(\theta) \Big|_{\theta = \theta^{(\tau)}} \right)$$

Stochastic Gradient descent can help push away from local minimums

- Randomization through taking smaller batches of the data
- Quicker to execute

Stochastic Gradient Descent



Week 12: Lecture 19 Classification (4/6)

Linear Regression

- Goal is to predict a quantitative variable from a set of features
- $\hat{y} = f_{\theta}(x) = x^T \theta$

Classification

- Goal: Predicting some categorical variable
- Binary classification: two classes 0 or 1
- Multiclass classification: many classes

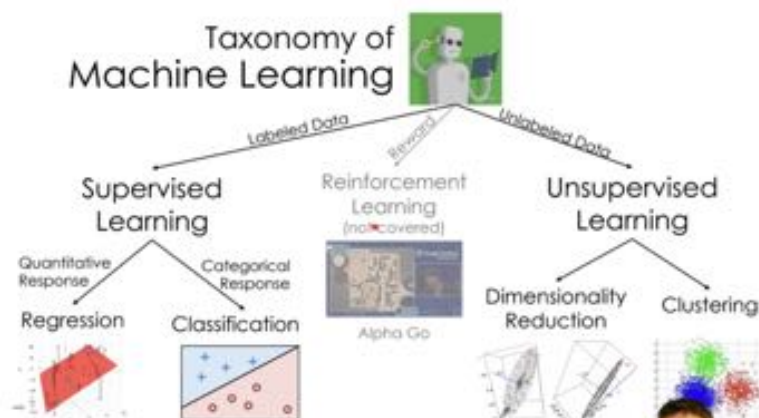
Machine learning taxonomy

- Regression and classification are both forms of supervised
- Design algorithms that make predictions adjusting depending on data

Machine learning taxonomy

Regression and Classification are both forms of **supervised learning**.

Logistic regression, the topic of this lecture, is mostly used for **classification**, even though it has "regression" in the name



Example

- Classes of bins and averaging result for each bin, taking the middle of the
- $odds(p) = \frac{p}{1-p}$

Log-odds

- Roughly linear,
- $logodds(p) = \log\left(\frac{p}{1-p}\right)$

$$t = \log\left(\frac{p}{1-p}\right)$$

$$e^t = \frac{p}{1-p}$$

$$e^t - pe^t = p$$

$$p = \frac{e^t}{1 + e^t} = \frac{1}{1 + e^{-t}}$$

- $\leftarrow \sigma(t)$ function

Logistic regression model (generalized linear model)

$$P(Y = 1|x) = \frac{1}{1 + e^{-x^T\theta}} = \sigma(x^T\theta)$$

Logistic regression model

- predict a binary categorical variable as a linear function of features, passed through the logistic function

Properties of logistic function

- Type of sigmoid,
- Output bounded between 0 and 1
- Mapping real numbers to probabilities

Definition	Range	Inverse
$\sigma(t) = \frac{1}{1 + e^{-t}} = \frac{e^t}{1 + e^t}$	$0 < \sigma(t) < 1$	$t = \sigma^{-1}(p) = \log\left(\frac{p}{1-p}\right)$

Reflection and Symmetry	Derivative
$1 - \sigma(t) = \frac{e^{-t}}{1 + e^{-t}} = \sigma(-t)$	$\frac{d}{dt}\sigma(t) = \sigma(t)(1 - \sigma(t)) = \sigma(t)\sigma(-t)$

Larger theta means steeper, negative starts at 1 and goes to 0

$$P(Y = 1|x) = \sigma(x^T\theta) \quad \leftarrow \quad \log\left(\frac{P(Y = 1|x)}{P(Y = 0|x)}\right) = x^T\theta \quad \leftarrow \quad \frac{P(Y = 1|x)}{P(Y = 0|x)} = e^{x^T\theta}$$

When happens changing x

- $\theta_1 > 0$, the odds increase

- $\theta_1 < 0$, the odds decrease

Exponent 0 when probabilities are equal

Logistic regression with squared loss

- Loss surface of MSE for logistic function, bounded between 0 and 1
- Loss surface not convex, weird things happen

Cross-entropy loss

- $-\log$: Further our prediction, the worse our prediction

Cross-entropy loss or log loss

$$\text{loss} = \begin{cases} -\log(1 - \hat{y}) & y = 0 \\ -\log(\hat{y}) & y = 1 \end{cases}$$

γ written unconditionally as:

$$\text{loss} = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

Mean cross-entropy loss

- Empirical risk for logistical regression model

$$R(\theta) = -\frac{1}{n} \sum_{i=1}^n (y_i \log(\sigma(\mathbb{X}_i^T \theta)) + (1 - y_i) \log(1 - \sigma(\mathbb{X}_i^T \theta)))$$

Modeling recipe

- Squared loss and cross-entropy loss results in different theta hat
- Different optimization problems, different solutions
- Cross-entropy loss is strictly better than squared loss for logistic regression
 - Convex

Likelihood

- If we have coin bias p
- Likelihood function: $L(p) = p^4(1 - p)^6$
- Estimate $\hat{p} = 0.4$

Log likelihoods

- Log is strictly increasing function, for 2 variables

$$\begin{aligned} & \log \left((p_1^{y_1} (1 - p_1)^{1-y_1}) (p_2^{y_2} (1 - p_2)^{1-y_2}) \right) \\ &= y_1 \log(p_1) + (1 - y_1) \log(1 - p_1) + y_2 \log(p_2) + (1 - y_2) \log(1 - p_2) \\ &= \sum_{i=1}^2 (y_i \log(p_i) + (1 - y_i) \log(1 - p_i)) \end{aligned}$$

Starti
famili

Find $\hat{p}_1, \hat{p}_2, \dots, \hat{p}_n$ that maximize $\sum_{i=1}^n (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$

Equivalently:

Find $\hat{p}_1, \hat{p}_2, \dots, \hat{p}_n$ that *minimize* $-\frac{1}{n} \sum_{i=1}^n (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$

Change to negative to find minimize

Maximum likelihood estimation (MLE)

- OLS, Ridge Regression

Summary

- `lm.LogisticRegression`

Week 12: Lecture 20 Logistic Regression, Classification (4/8)

Logistic Regression

- Predicting a binary categorical variable as a linear function of features, passed through the logistic function
- cross-entropy loss is much better (convex)

Quick Note on Cross-Entropy Loss

Thresholding

Classification

- Want to predict binary classification but output is continuous
- Simply classification

$$\text{classify}(x) = \begin{cases} 1, & P(Y = 1|x) \geq 0.5 \\ 0, & P(Y = 1|x) < 0.5 \end{cases}$$

-

- Threshold work the same way in higher dimensions

Evaluating Classifications

Accuracy

$$\text{accuracy} = \frac{\# \text{ of points classified correctly}}{\# \text{ points total}}$$

-

- Widely used, `model.score`

Pitfalls of accuracy

- Can classify everything as spam could still give 95%
- Looking for rare

Types of classification errors

		Prediction	
		0	1
Actual	0	True negative (TN)	False positive (FP)
	1	False negative (FN)	True positive (TP)

- True positives and true negatives classify positive or negative
- False positives are "false alarms"
- False negatives "fail to detect"

Accuracy

- What proportion of points classifier classify correctly

$$\text{accuracy} = \frac{TP + TN}{n}$$

Precision

- What proportion were actually 1, penalizes false positives

$$\text{precision} = \frac{TP}{TP + FP}$$

Recall

- Of observations actually 1, what proportion did predict to be 1, penalizes false negatives

$$\text{recall} = \frac{TP}{TP + FN}$$

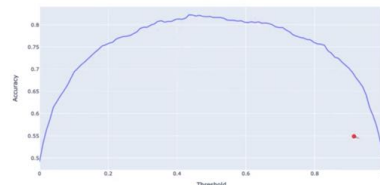
Tradeoff between precision and accuracy

- Adjust threshold
- Higher threshold -- fewer false positives, precision increases
- Lower threshold -- fewer false negatives, recall increases

Visual Threshold

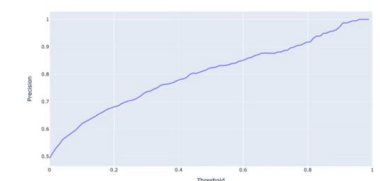
Accuracy vs. Threshold

- Expected accuracy to be maximized when threshold near 0.5



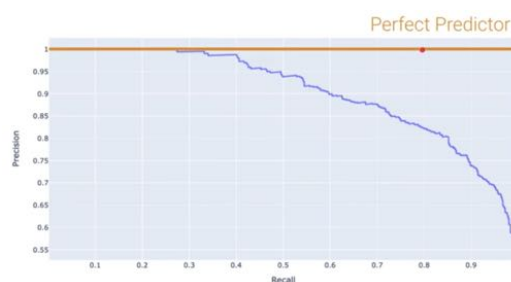
Precision vs. threshold

- Increase threshold, fewer false positives



Precision-recall Curves

- Threshold decreases from top left of bottom right



- Compare model is to use the area under the curve

False positive rate (FPR)

- $FP / (FP + TN)$

True Positive Rate (TPR)

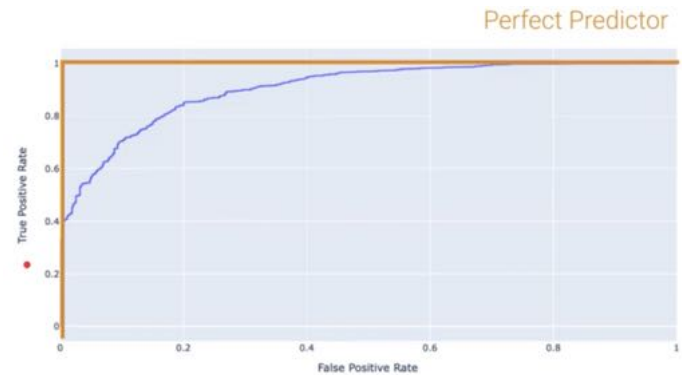
- $TP / (TP + FN)$

Receiver Operating Characteristic (ROC)

- Plots TPR vs FPR

Decision Boundaries

- Decision boundary is line/point/plane



Linear Separability, Regularization

Infinitely large theta is a bad idea

- Cross entropy loss is infinite

Linear Separability

- Linear separable if we can correctly separate the classes with a line
- Degree 0 hyperplane to separate one
- Formally: set of d dimensional points is linearly separable if we can draw a degree -1 hyperplane to separate them

Regularized logistic regression

- Standardize our features before applying regularization
- Scikit learn always applies regularization by default

$$R(\theta) = -\frac{1}{n} \sum_{i=1}^n (y_i \log(\sigma(\mathbb{X}_i^T \theta)) + (1 - y_i) \log(1 - \sigma(\mathbb{X}_i^T \theta))) + \lambda \sum_{i=1}^p \theta_i^2$$

Summary

- Logistic regression models probability of belong to class 1: binary classification
- - employ a threshold, decision rule, different thresholds yield different decision boundaries
- Look at several numeric and visual metrics
 - accuracy , precision, recall
 - PR curves, ROC curves
- Using regularization for logistic regression is a good idea

Week 13: Lecture 21 Decision Trees (4/13)

Decision Trees

- Simple way to classify data, tree of questions that must be answered in sequence

Decision Tree Basics: scikit-learn

- From sklearn import tree
- `tree.DecisionTreeClassifier().fit(data[['data1', 'data2']], [choices])`

- There is a visualizer
 - GraphViz

Decision Trees and Overfitting

- Always achieve 100% with decision trees
- If boundaries aren't clear, becomes too messy and overfitting

Basic Decision Tree Generalization

1. All data starts in root nodes
2. Repeat until every node is either pure or unsplittable
 - a. Pick the best feature x and best split value st that the loss of the resulting split is minimized
 - b. Split data into two nodes

Defining a best feature

- Node Entropy
- $S = - \sum_c p_c \log_2 p_c$
- p_c proportion of data points in node c
- Low entropy means more predictable
- High entropy means more unpredictable

Weighted Entropy as loss function

$$L = \frac{N_1 S(X) + N_2 S(Y)}{N_1 + N_2}$$

Restricting Decision Tree Complexity

- Don't allow to overfit, restrict

Approach 1: Preventing Growth

- Don't split nodes if less than 1%

Approach 2: Pruning

- Set validation set
- Prune using validation set
- If replacing node by most common prediction no impact on validation error

Most typical Approach: Random Forest

- Build many decision trees and have them vote
- For given x/y , use whichever prediction is most popular
- Building many trees
 - Bagging: bootstrapping
 - Resample of training data
- Only use a sample of m features at each split
 - $M = \sqrt{p}$
 - Algo creates individual trees

- Bootstrap training data T times
 - Pick random subset of m features
 - Best feature x and split value beta st that loss is minimized
- Random Forests?
 - Versatile: does both regression and classification
 - Invariant to feature scaling and translation
 - Automatic feature selection
 - Nonlinear decision boundaries without complicated feature engineering
 - Ensemble method: combine knowledge

Decision trees for regression

- Alternate non-linear framework for classification and regression
 - Underlying principle is fundamentally different

Regression classification

Week 13: Lecture 22 Inference for Modeling (4/15)

Inference

- Induction is inference from particular premises to universal conclusion

Statistical Inference

- Process of data analysis to deduce properties of underlying distribution of probability
- Increase our understanding of the phenomena, rather than displaying mastery

Prediction vs. inference

- Prediction is task of using model to make predictions for the response of unseen data
- Inference task of using model to draw conclusions about underlying true relationship

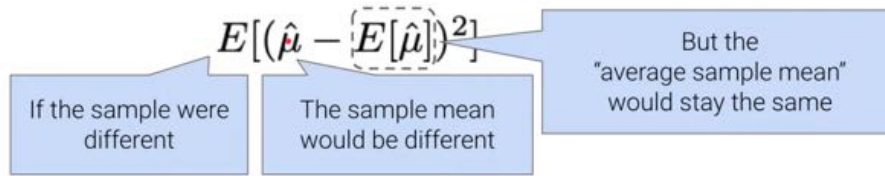
Statistical inference

- Population is a parameter
- Compute a statistic of the random sample
- Inference is drawing conclusion about population parameters, given only a random sample

Terminology

- Parameter: function of a population
 - θ^*
 - Population mean
- Estimator: function of a sample, goal is to estimate a population parameter
 - $\hat{\theta}$
 - Estimators are random variables
- Sample distribution: distribution of estimator values, across all possible samples
- Bias: estimator: difference between estimator's expected value and true value of a parameter being estimated
 - Zero bias: on avg, estimate is correct
 - Non-zero bias: on average, our estimate is consistently too large / too small

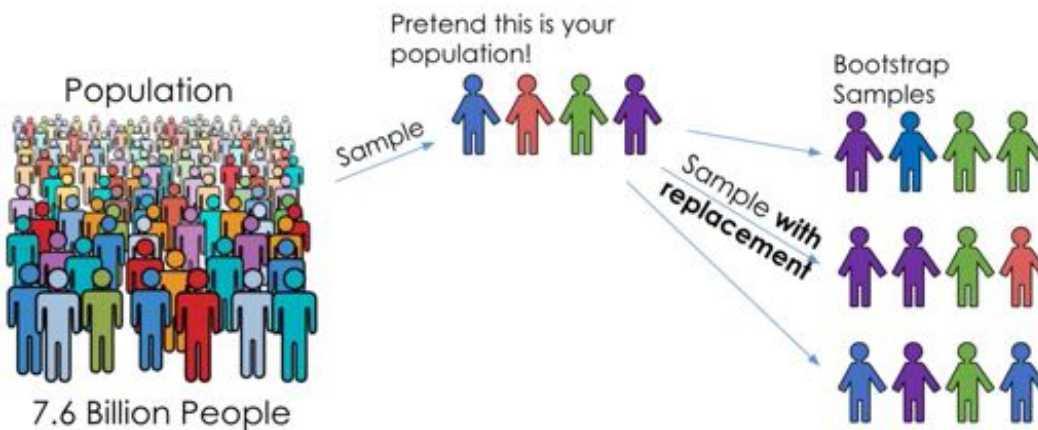
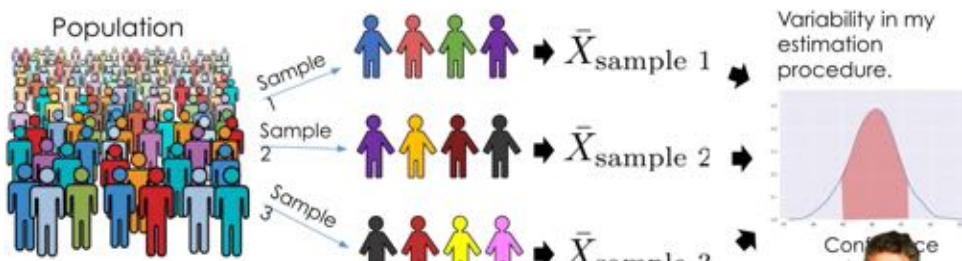
- Bias[$\hat{\theta}, \theta^*$] = $E[\hat{\theta}] - \theta^*$
- Variance: expected square deviation
- Variance of a Sample mean estimator



Bootstrap resampling

- To determine properties of sampling distribution
- Treat our random sample as a population, and resample from it

Resampling the population to estimate the sample distribution.



Lessons from Hesterberg

- Ordinary bootstrap not work well for mean

Bootstrap Confidence intervals

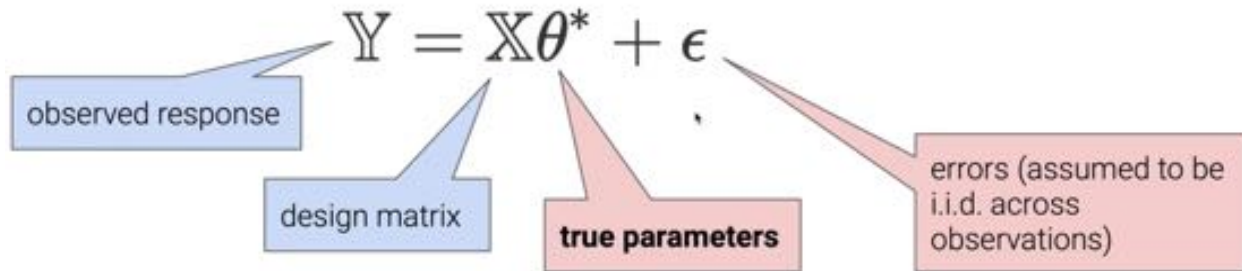
- Estimate a interval where we thinking the population parameter is, based on the center and variance of the estimator
- What does a P% confidence interval mean
 - Take sample from population
 - Compute p% confidence interval for true confidence interval

Confidence intervals

- Estimator f exists to guess the value of unknown parameter θ^*
- Estimator ci for $p\%$ confidence interval takes as sample and returns an interval
- $ci(s, f, P)$
- Statement about procedure
- Percentile bootstrap confidence intervals

The Regression model

- Fitting a linear regression model assuming, only see a noisy version of it



Bootstrapping Model Parameters

Parameter estimates

- Depends on what training data was
- Bootstrap our training data
- Fit a linear model to each resample
- Look at resulting distribution of bootstrapped parameter estimates

Assessing the quality of our model

$$y = f_{\theta^*}(x) + \epsilon = \theta_0^* + \sum_{j=1}^p \theta_j^* x_j + \epsilon$$

assumed underlying model

$$\hat{y} = f_{\hat{\theta}}(x) = \hat{\theta}_0 + \sum_{j=1}^p \hat{\theta}_j x_j$$

how we make predictions

Multicollinearity

- If features are related to each other it might not be possible to have change in one them while - holding the others constant
- When a feature can be predicted fairly accurately by a linear combination of other features
- Perfect: One feature can be written exactly as linear combination

Summary

- Estimators are functions provide estimates of true population parameters
- Use bootstrap to estimate the sampling distribution of an estimator
- Using this bootstrapped sampling distribution compute a confidence interval for our estimator

- Assumption when performing linear regression is some true parameter θ defines a linear relationship between features X and response Y
- Multicollinearity arises when features correlated with one another

Week 14: Lecture 23 Principle Component Analysis (4/20)

Unsupervised Learning

Visualizing High Dimensional Data

- Pick attributes highest variance
- Plot 2 dimensional approx results in linear combination of attributes

Singular Value Decomposition

- Total amount of information stored is less

Diagonal matrix

- Multiplying by a diagonal matrix equivalent to scaling the columns
- Singular values appear in decreasing order
 - Non-negative

U and V

- Orthogonal vectors meet at a right angle, dot prod 0

Principle component

- Columns left in SVD

Data Centering

- Recenter data by subtracting the mean of each column for all values in that column
- To get correct principle components, center data first
- Principle component $U\Sigma$

i th singular value tells how valuable the i th principal component will be

i th singular value tells us how much of the variance is captured by i th principal component

Principal component analysis for exploratory data analysis

- Centered = $\text{body_data} - \text{np.mean}(\text{body_data}, \text{axis} = 0)$
- Account for most of variance - "skee" plot
- Reduce to a few principal components

PCA appropriate for EDA when

- Visually identifying clusters of similar observations in high dimensions
- Still exploring the data
- Reason to believe data are inherently low rank

SVD for PCA

- Matrix decomposition
- $X = U\Sigma V^T$
- Where U and V are orthonormal and Σ diagonal
- X rank r , r non zero values on diagonal
- Values in sigma singular values greatest to least

PCA specific application of SVD

- X is data matrix centered at mean of each column
- Dimensionality reduction
- First n rows of V^T are direction
- Σ columns re principal components
- Largest n singular values are kept

Summary

PCA : to summarize data

- Find directions that minimize projection error
- Maximize captured variance

Use SVD to do PCA

- 2D plot to preserve structure

Week 14: Lecture 24 Clustering (4/22)

Unsupervised Learning

- Identify patterns in unlabeled data

K-Means Clustering

- Pick arbitrary k and randomly place k centers, each a different color
- Repeat until convergence
 - Color points according to the closest center
 - Move center for each color to center of points with that color

K-Means: For clustering

- Assigns each point to one of K clusters

K-Nearest Neighbors: For classification

- Prediction is most common class among k-nearest data points of training set

Minimizing Inertia

Inertia: Sum of squared distance from each data point to its center

Distortion: Weighted sum of squared distances from each data point to its center

- Lowest inertia is best

Best runtime is NP-hard k^n

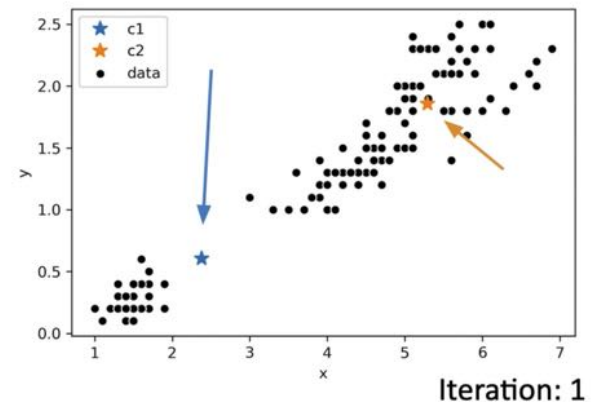
Can do some approximation

Agglomerative clustering

K means optimized for distance not blobbiness

Agglomerative Clustering

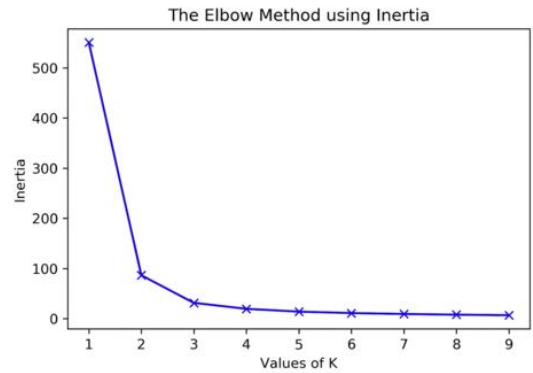
- K Means Attempts to minimize inertia
- Every data point starts out as its own cluster
- Join clusters with neighbors until we have k clusters left
- Distance for cluster is max (or min or avg)
- Hierarchical clustering



- Each cluster is a tree, can keep track of when merged

How to choose k

- Elbow Method
- Plot inertia vs many different K Values



Silhouette Scores

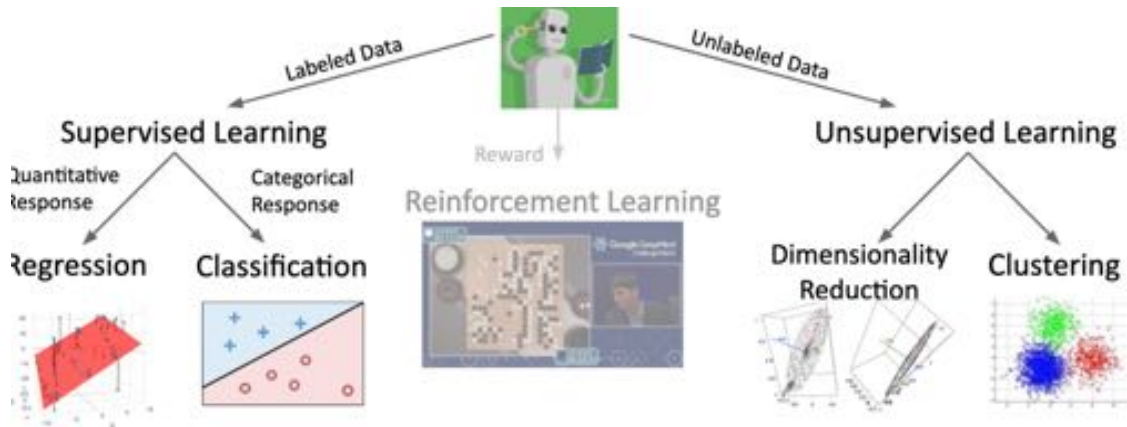
- High Score: near the other points in its X's cluster
- Low Score: far from the other points in its cluster
- A = avg dist to other points in cluster
- B = avg dist to points in closest cluster
- $S = (B - A) / \max(A, B)$
- What is highest possible S
 - 1 when ever point in X's cluster is right on top of X
- Can be negative

To pick K:

- Could need to consider projected costs and sales for 2 different Ks

Summary

- K Means
- Agglomerative
- Linear models for regression and classification

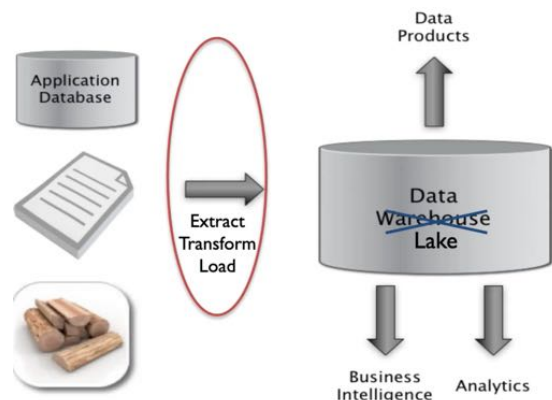


Week 15: Lecture 25 Big Data (4/27)

Big Picture: Actionable Analytics

Big Data

- Data of sizes beyond ability of traditional SW tools to manage, process, need parallel computing tools
- Descriptive, Predictive, Prescriptive: what to do about it



Sources of Big Data

- Web, Mobil, open data sources, Internet of things

Online

- Click, Ad impression, server request

User Generated content

- Web mobile

Graph Data

- Social networks, telecom networks, relationships

Aerospace

- Boeing 787 500GB per flight
- Optimize fleet maintenance operations
- Improve operational performance across flights/fleet

Autonomous Vehicles

- 5 TB/hr/car
- DNN Training: 800 GPUs per test car

Smart Homes

- HVAC: ML at scale

e-Commerce

- Predictive analytics

Precision Medicine

- COVID severity score model
- Built model using biomarkers

Operational Data Stores

- Capture the now, different databases across an organization
- Extract, Transform, Load (ETL)
 - Extract from remote sources,
 - Transformed into std schema, clean and prepare data for analytics
 - Load: into relational DB

Multidimensional Data

- Fact Table
- Other tables for labels
- Dimensions
 - Easy to manage and summarize
 - Reduce data errors
- Star Schema with fact tables, and dimension tables

Sales Fact Table

pid	timeid	locid	sales
11	1	1	25
11	2	1	8
11	3	1	15
12	1	1	30
12	2	1	20
12	3	1	50
12	1	1	8
13	2	1	10
13	3	1	10

Locations

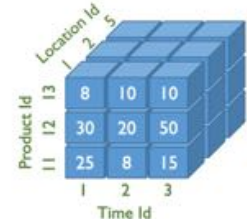
locid	city	state	country
1	Omaha	Nebraska	USA
2	Seoul		Korea
5	Richmond	Virginia	USA

Products

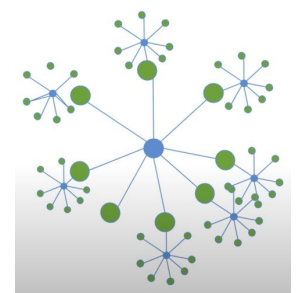
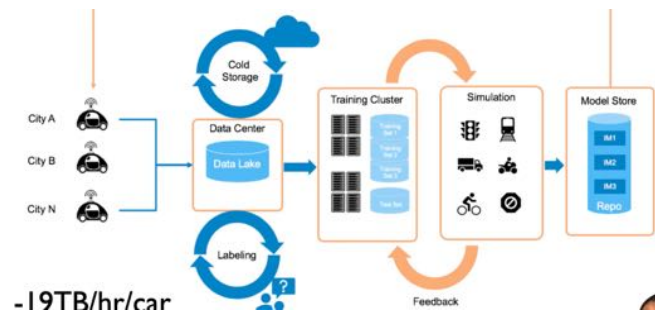
pid	pname	category	price
11	Corn	Food	25
12	Galaxy I	Phones	18
13	Peanuts	Food	2

Time

- Multidimensional "Cube" of data



Online Analytics Processing (OLAP)



- Using graphical tools to construct queries

Data Warehouse

- Collects and organizes historical data from multiple sources
- How to deal with semi-structured and unstructured data

Data Lake

- Store a copy of all the data in one place in original natural form
- Enable data consumers to choose how to transform use data
 - Schema on read

Dark side of data lakes

- Cultural shift: curate -> Save Everything
 - Noise begins to dominate
 - No cleaning and verifications -> lots of dirty data

Data scientists have SQL over large files, data engineer who tracks and manages files

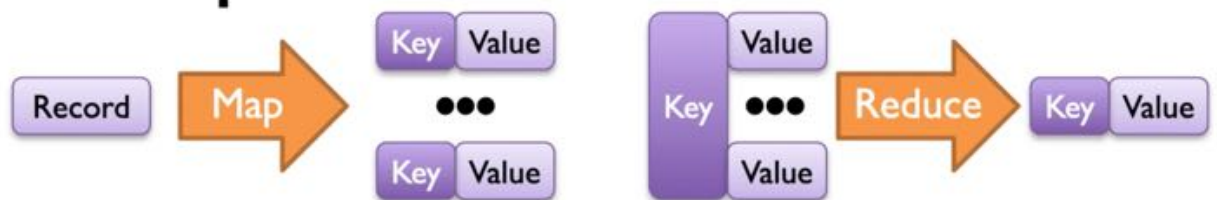
Hardware for Big Data

- Very large files spanning multiple computers
- Distributed data processing quickly and easily
 - Redundancy
- Lots of data lots of CPU
- Distributed computing
 - Load and process files on multiple machines concurrently
- Functional programming -> parallelism

Fault Tolerant Distributed File Systems

- Split file into smaller parts, split into different copies
- Very fast reads of large files

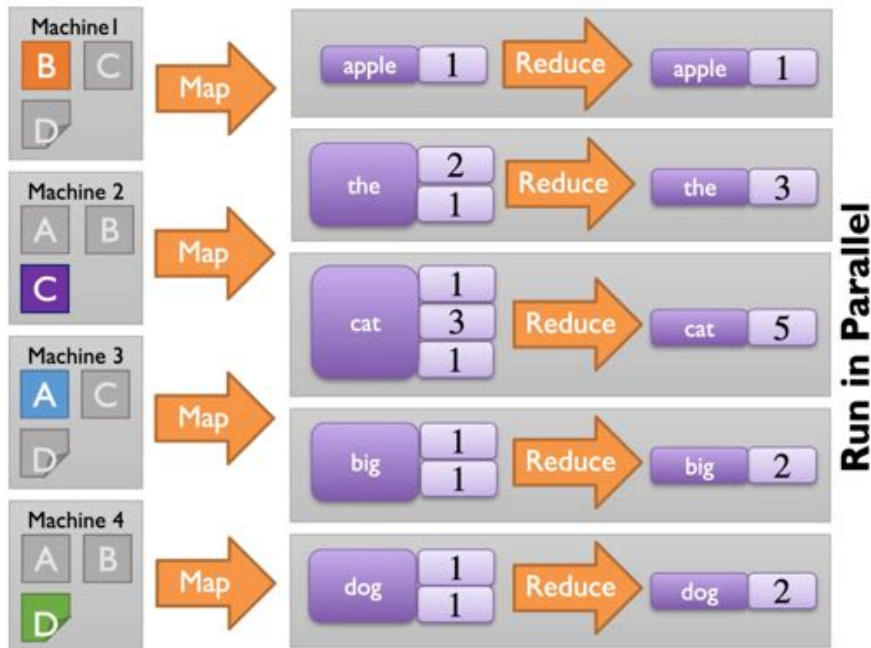
Map-Reduce Distributed Aggregation



- Map separates out function locally to compute and reduce combines values of one key
- Map: deterministic
 - Re execute on failure, same computation every time
 - Use random seed
- Reduce: commutative and Associative
 - Commutative: Allow re order of operations
 - Associative Reduce: allows for regrouping of operations

Executing Map Reduce

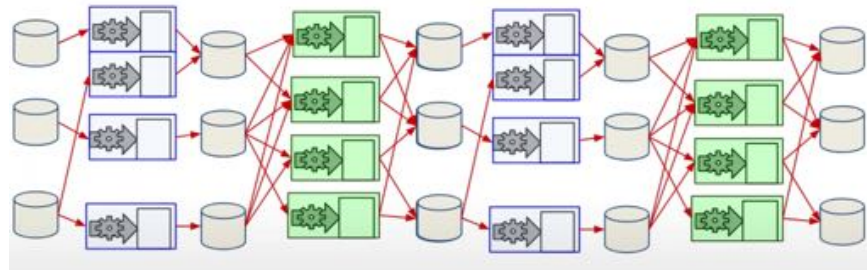
- Map function applied to a local part of the big file to run in parallel



Map Reduce Technologies

Apache Hadoop

- First open source ecosystem
- HDFS: FaDoop File System
- MapReduce: map-reduce compute framework
- YARN: Yet another resource negotiator
- Tedious to use

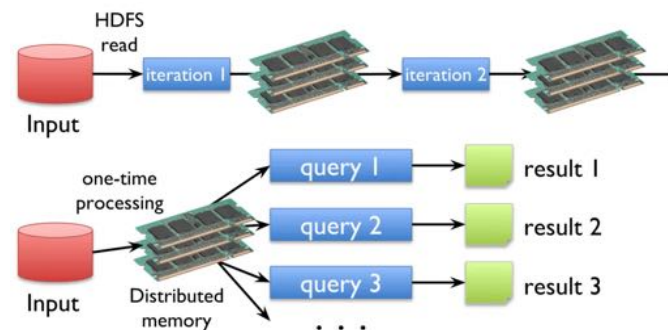


Each stage passes through hard drive:

Disk I/O very slow

Apache Spark : In-Memory Dataflow System

- Keep more data in-memory
- Up to 100x faster than disk network
- Parallel execution engine for big data processing
- General: efficient support for multiple workloads
- Easy to use: 2-5x less code
- Fast: up to 100x faster than HaDoop



Spark Programming Abstraction

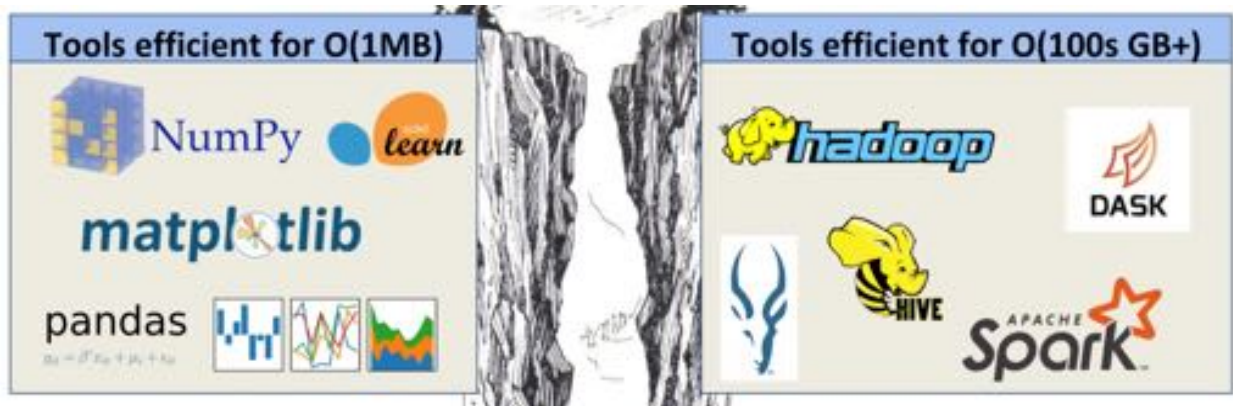
- Write programs in terms of transformations on distributed datasets
- Resilient Distributed Datasets (RDDs)
 - Distributed collections of objects that can be stored in memory or on disk

- Automatically rebuilt on failure
- Transformations and actions
 - Transformations
 - $f(RDD) \rightarrow RDD$
 - Actions
 - Trigger computation

	Hadoop Map Reduce	Spark
Storage	Disk only	In-memory or on disk
Operations	Map and Reduce	Map, Reduce, Join, Sample, etc...
Execution model	Batch	Batch, interactive, streaming
Programming environments	Java	Scala, Java, R, and Python

Log Mining

- Load error messages from a log into memory, then interactively search for various patterns
- Filter, map, cache, filter, count
 - Only actively computes on count



Big tools: harder to debug

- Modin next gen dataframe
- More speedups

Modin

