Sardaana Eginova, Sarika Pasumarthy, Kelly Hu, Collin Duong

Final Review

Midterm 1: Pandas, EDA, Regex, Visualizations, Sampling, Simple Linear Regression, Constant Model
Midterm 2: OLS, Gradient Descent, Cross-Validation, Regularization, Feature Engineering, Random Variables, Bias-Variance Tradeoff, Parameter Inference and Bootstrap
Post midterm 2: SQL, Logistic Regression, Clustering, PCA

Fall 2025

Announcements

- Office hours at http://oh.ds.100.org/
  - Thursday 1-5 pm (today)
  - Friday 10-1 pm (tomorrow)

- Wednesday 12/17
  - Final exam 8-11 am

- Ed support until the morning of the final. Please reach out to us with any questions or concerns!
- Good luck studying, you got this 💪

Logistics + Housekeeping

- The exam prep session will be divided into six 10-15 minute sections:
  - 10:10-10:25 | Pandas, EDA, Regex, Visualizations, Sampling, Simple Linear Regression, Constant Model
  - 10:25-10:40 | OLS, Gradient Descent, Cross-Validation, Regularization, Feature Engineering, Random Variables, Bias-Variance Tradeoff, Parameter Inference and Bootstrap
  - 10:40-10:50 | SQL
  - 10:50-11:05 | Logistic Regression
  - 11:05-11:15 | Clustering
  - 11:15-11:30 | PCA

- This session will focus on topic review for the first half and problem-solving for the last half – we will be doing guided walkthroughs of past exams
- Please reserve any face-to-face questions you may have for after the session, so we stay on schedule!

Pandas

Pandas is a commonly used Python library for processing tabular data

- process data in tables
- perform vectorized operations (fast)
- extract useful information, data science!

Importing Pandas:

```
import pandas as pd
```

Series and DataFrames

- A **Series** is a vector with ***index*** and ***values*** that can be accessed with `s.index` and `s.values`
- A **DataFrame** is a list of Series stacked together horizontally
- A DataFrame is essentially a table class that has efficient storing, retrieving, and modifying methods

- When we call methods on Series and DataFrames, they stack up from left to right: the output of the left method is the input to the right method
  - example: `df.sort_values("a").head(10)` will return first ten rows of the sorted `df` DataFrame

Note: DataFrames can be created from CSV (comma separated values) file, dictionary, JSON file, etc.

Series operations: selection

We can select values from a Series instance **`s`** using:

- a single label: **`s[`**<a string>**`]`**
  - example:

    `s[“apple”]`
- many labels: **`s[`**<a list of strings>**`]`**
  - example:

    `s[[“apple”, “banana”, “orange”]]`
- given a filter (boolean) condition: **`s[`**<a boolean list/a boolean Series>**`]`**
  - example:

    `s[s > 5]`

```
s
```

```
apple     2
banana    3
mango     7
dtype: int64
```

```
[27]: s["apple"]
```

```
[27]: 2
```

```
[29]: s[["apple", "banana"]]
```

```
[29]: apple     2
      banana    3
      dtype: int64
```

```
[31]: s[s > 5]
```

```
[31]: mango    7
      dtype: int64
```

More Series operations

Pandas has many built-in methods to work with Series:

- sort `Series.sort_values(ascending=True)`
- replace values `Series.map({`<old_value>`:` <new_value>`})`

Note: Series/DataFrame operations do **not** occur in-place! You must re-assign the output Series/DataFrame, unless you explictly set `inplace=True` as one of the arguments to the method

DataFrame operations: selection

Many ways to extract data from a DataFrame:

- `.head()`, `.tail()`
- label-based extraction: `.loc[]`
  - list
  - slice
  - single value
- integer-based extraction: `.iloc[]`
  - list
  - slice
  - single value

Note: `.iloc[]` will always look at the underlying DataFrame index, while `.loc[]` will look at the modified DataFrame

DataFrame

```
df
```

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99
2	mango	5	5.99

Many ways to extract data from a DataFrame:

- **label-based extraction: `.loc[]`**
  - **list**
  - slice
  - single value

```
[125]: df.loc[[0, 1]]
```

[125]:

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99

Returns DataFrame

DataFrame

```
df
```

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99
2	mango	5	5.99

Many ways to extract data from a DataFrame:

- **label-based extraction: `.loc[]`**
  - list
  - **slice (inclusive:inclusive)**
  - single value

```
[127]: df.loc[0:1]
```

[127]:

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99

Returns DataFrame

DataFrame

Many ways to extract data from a DataFrame:

- **label-based extraction: `.loc[]`**
  - list
  - slice
  - **single value**

```
df
```

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99
2	mango	5	5.99

```
[137]: df.loc[2]
```

```
[137]: fruit    mango
       stock        5
       price     5.99
       Name: 2, dtype: object
```

Returns Series

DataFrame

Many ways to extract data from a DataFrame:

- **integer-based extraction: `.iloc[]`**
  - **list**
  - slice
  - single value

```
df
```

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99
2	mango	5	5.99

```
[161]: df.sort_values('stock', ascending=False).loc[[0, 1]]
```

[161]:

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99

but

```
[164]: df.sort_values('stock', ascending=False).iloc[[0, 1]]
```

[164]:

	fruit	stock	price
2	mango	5	5.99
1	banana	3	0.99

Returns DataFrame

DataFrame

```
df
```

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99
2	mango	5	5.99

Many ways to extract data from a DataFrame:

- **integer-based extraction: `.iloc[]`**
  - list
  - **slice (inclusive:exclusive)**
  - single value

```
[166]: df.iloc[0:1]
```

[166]:

	fruit	stock	price
0	apple	2	1.99

Returns DataFrame

DataFrame

Many ways to extract data from a DataFrame:

- **integer-based extraction: `.iloc[]`**
  - list
  - slice
  - **single value**

```
df
```

	fruit	stock	price
0	apple	2	1.99
1	banana	3	0.99
2	mango	5	5.99

```
[179]: df.sort_values('stock', ascending=False).loc[2]

[179]: fruit    mango
       stock        5
       price     5.99
       Name: 2, dtype: object
```

but

```
[185]: df.sort_values('stock', ascending=False).iloc[2]

[185]: fruit    apple
       stock        2
       price     1.99
       Name: 0, dtype: object
```

Returns Series

More DataFrame operations

- modify columns
  - create new columns: `df["col_name"] =` <series/array>
  - rename columns: `df = df.rename(columns={...})`
  - remove columns: `df = df.drop(<column_name>, axis=1)`
- replace values: `df.replace({ <old>: <new>, ... })`
  - note: different from `Series.map()`; DataFrame.replace() is out of scope in Data 100
- sort: `df.sort_values(`<column/a list of columns>`, ascending=True/False)`
- join: `left_df.merge(right_df, left_on=`<column>`, right_on=`<column>`, how='inner')`
- group and filter: `df.groupby(...).agg(...)` vs `df.groupby(...).filter(...)`
- pivot table: `df.pivot_table()`

Other methods: `.reset_index()`, `.set_index()`, `.fillna()`, `.isin()`

Grouping

`df.groupby(`<column(s)>`)`

- **`.agg(f)`**: changes granularity of the DataFrame by *aggregating*
  - **`f`** is applied to each column, the input is **`Series`**
  - **`.max(), .sum(), .min()`**
  - **`.size()`**: returns **`Series`**, includes count of rows with NaN
  - **`.count()`**: returns **`DataFrame`**, applies count to each column

- **`.filter(f)`**: does not change granularity of the DataFrame
  - just filters out groups for which the functions returns **`False`**
  - **`f`** is applied to whole table, the input is **`DataFrame`**

Pivot table

`df.pivot_table(values=..., index=..., columns=..., aggfunc='mean', fill_value=None)`: changes granularity of the DataFrame

EDA and four key data properties

1. Structure: ***quantitative*** (numerical) and ***qualitative*** (categorical: *ordinal*/nominal) variable types.
2. Granularity: what each row represents in the DataFrame.
3. Temporality: to use the datetime format, need to use `.dt` accessor.

```
[136]: september_birthdays = pd.Series([
           "September 1, 2025",
           "september 9, 2025",
           "september 10, 2025",
           "september 14, 2025",
           "september 15, 2025"])
       september_birthdays = pd.to_datetime(september_birthdays, errors="coerce")
       display(september_birthdays)
       display(september_birthdays.dt.dayofweek)
```

4. Faithfulness: treating missing data
    a. Keep as NaN
    b. Drop
    c. Impute with mean/median or interpolate

Note: dropping rows can introduce additional bias

Visualizations

In Data 100, we teach two libraries for plotting, **`matplotlib.pyplot`** and **`seaborn`**

Importing both:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

- **`matplotlib.pyplot`** methods take in arrays
  - lower level
- **`seaborn`** methods take in argument **`data`** for **`pandas`** DataFrame; only need to specify column names instead of passing in arrays directly
  - higher level, built on top of **`matplotlib.pyplot`**

Note: conventionally alias as **`sns`** and **`plt`**

One variable

Variable type(s)	Best visualization	What it shows
Quantitative	Histogram	Distribution (shape, center, spread, skew)
Quantitative	Boxplot	Summary statistics (median, IQR, outliers)
Quantitative	Density plot (KDE)	Smoothed estimate of the distribution
Qualitative	Bar chart	Counts/frequency of each category
Qualitative	Pie chart	Proportion of categories

Two variables

Variable type(s)	Best visualization	What it shows
Quant + Quant	Scatter plot	Relationship (correlation, trends, outliers)
Quant + Quant	Hex plot	Relationship (correlation, trends, outliers); reduces overplotting
Quant + Quant	Line plot	Temporal trends over time
Quant + Qual	Boxplot per category	Distribution of values per category
Quant + Qual	Violin plot per category	Shape and distribution of values per category
Qual + Qual	Stacked bar chart	Part-to-whole within categories

Three and more variables

Variable type(s)	Best visualization	What it shows
All Quant	Heatmap e.g. correlation matrix	Strength and direction of variable correlations
Mixed Quant + Qual	Color or shape in scatter plot	Adds third (or more) variables via encoding

Kernel Density Estimation

Idea: approximate true distribution

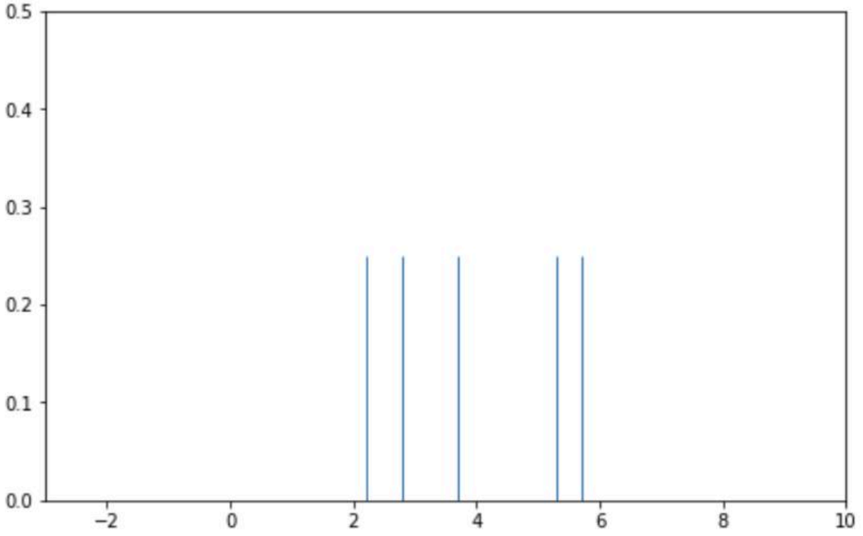
To calculate **KDE curve**, we need to:

1. For each data point, **assign kernel** (error range)
    - we assume raw data is noisy, so treat it as a random sample
2. **Sum up the kernels** across all data points
3. Normalize to have **total area 1** below the curve
    - the total probability should be 1 after we integrate

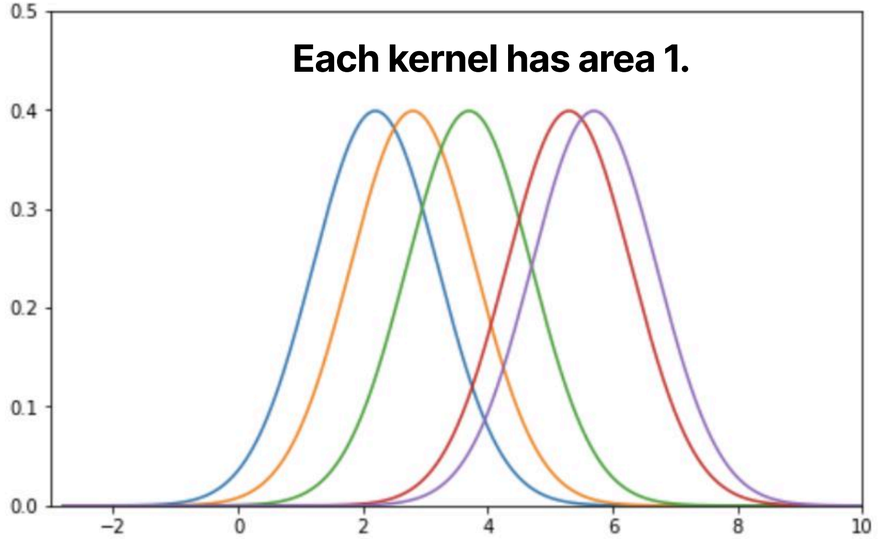
In class, we go over a Gaussian kernel centered at data point x_i with fixed bandwidth 1.
Note: There are many other kernel types! To name some: Boxcar (uniform), Epanechnikov, Cosine.

KDE curve

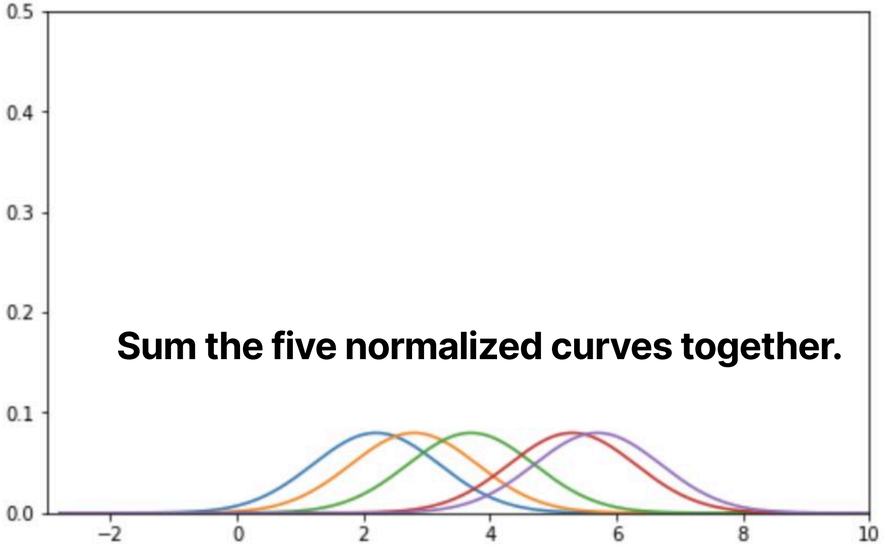
Step-by-step



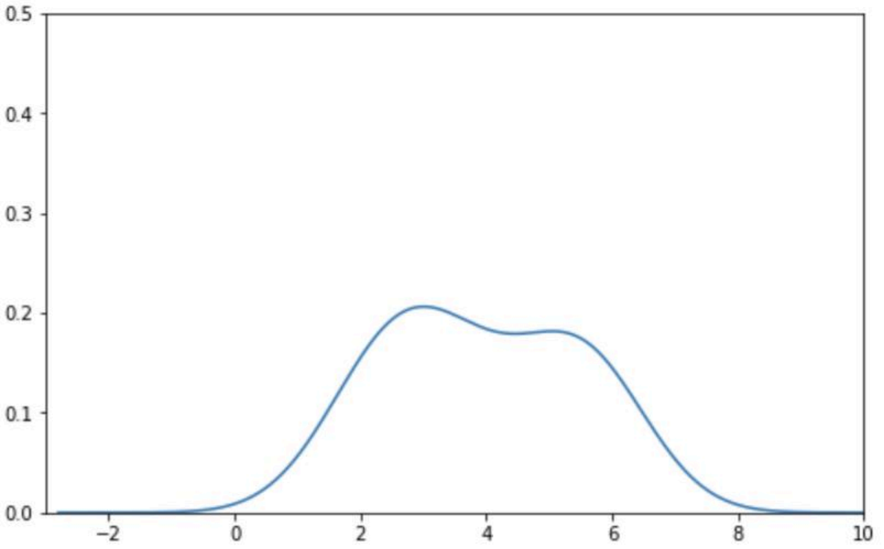
Each line represents a datapoint in the dataset (e.g., one country's HIV rate). This is a **rug plot**.



Place a kernel on top of each datapoint.



Each normalized kernel has density ⅕.

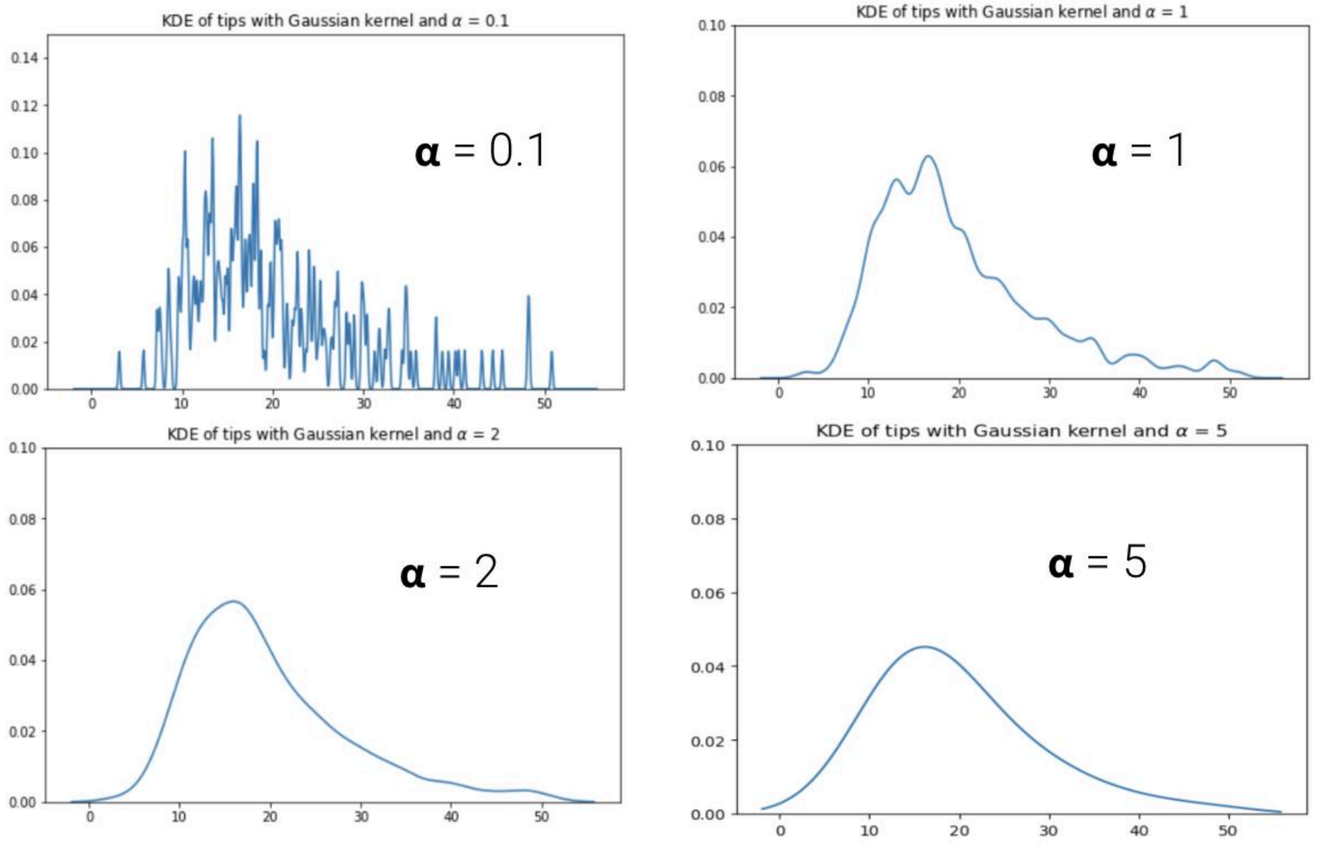


The final KDE curve.

Effect of bandwidth

As we increase bandwidth (=variance), the KDE curve becomes "*smoother*"

- intuitively, we remove more and more noise



String vs Series with strings

Before diving into regex expressions, it's important to clarify how **`pandas`** treats strings

String → String

```
s.lower()
s.split()
len(s)
```

Python assumes we are working with one string at a time

Series of strings → Series of strings

Pandas data accessor **`.str`** method is vectorized, meaning it accesses one string from a row at a time

```
df['col1'].str.lower()
df['col1'].str.split()
df['col1'].str.lower().str.split()
```

`.str` methods are stackable!

Regex

A regular expression is a sequence of characters that specifies a search **pattern**

We use raw strings for regex patterns that follow `r'<pattern>'` format

Order of regex operations (left-to-right): grouping with `()`, `*`, concatenation, `|`

	Meaning	Pattern	Match
	Look for consecutive characters	`r'AABBAB'`	`'AABBAB'` only
`|`	Match one pattern or another	`r'AA|BBAB'`	`'AA'` or `'BBAB'` only
`*`	Match zero or more of a pattern	`r'AB*A'`	Match: `'AA', 'ABBBBA'` Won't match: `'AB', 'ABABAB'`
`()`	Group operations together and apply one regex op to multiple characters	`r'A(A|B)AAB'`	`'AAAAB'`, `'ABAAB'`

	Meaning	Pattern	Match
`.`	Look for any character other than \n (newline)	`r'.U.U.U.'`	Match: `'CUMULUS'` Won't match: `'UUU'`
`+`	One or more character	`r'AB+'`	`'AB'`, `'ABB'`, `'ABBB'`
`{x}`	Repeat exactly x times	`r'AB{2}'`	`'ABB'` only
`{x, y}`	Repeat between x and y times, inclusively	`r'AB{0,2}'`	`'A'`, `'AB'`, `'ABB'`
`?`	Exactly zero or one times	`r'AB?'`	`'A'` or `'AB'` only

`[]`	Define an alphabet aka character class. All characters inside are literal	`r'[A-Za-z]'`
		`r'[a-z0-9]'`
		
		`r'[aeiou]'`
			all letters
			all lowercase letters & digits: `'2a34b'`
			all vowels: `'aaaaiee'`

Note: assume we use re.search(pattern, string)

	Meaning	Pattern	Match
`[^]`	Negate a character class aka matches all characters EXCEPT the following character	`r'[^A-Z]'`	any character that is NOT an uppercase letter: `'2z34b'`
`\`	Read next char literally	`r'a\+b'`	`'a+b'` only (`'+'` is no longer an operation)
`^`	Match beginning of a string	`r'^abc'`	Match: `'abc'` from `'abc123'` Won't match: `'123abc'`
`$`	Match end of a line or a whole string	`r'abc$'`	Match: `'abc'` from `'123abc'` Won't match: `'abc123'`

Greediness

Regex is greedy, meaning it looks for the **first longest continuous** match in a string

```
String = ‘aaaaabaac’

Pattern = r’a+ba*’

Match = ‘aaaaabaa’
```

```
String = "This is an <div>example</div>
of greediness <div>in</div> regular
expressions."

Pattern = r”<div>.*</div>”

Match = “<div>example</div> of greediness
<div>in</div>”
```

Note: there is no difference between `‘’` and `“”` in Python

Non-greediness with +? and *?

We can remove greediness with adding **? tag** to **+ and * operators**

```
r"<div>.*?</div>"
```

(joint `*?` operator is applied to `.`)

```
"This is an <div>example</div> of greediness <div>in</div> regular expressions."
```

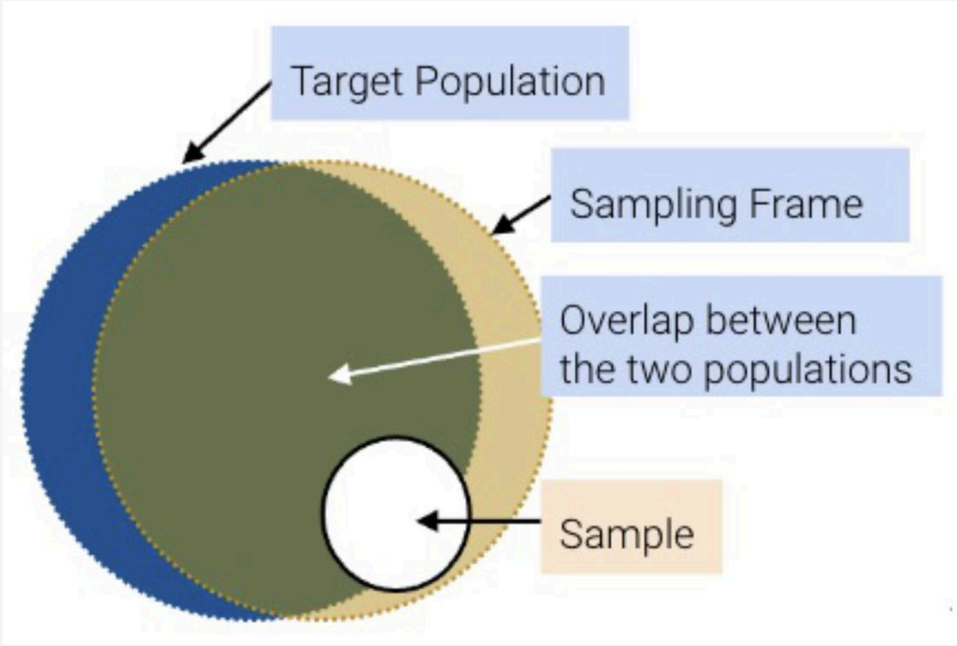
What we are telling regex:

1. Look for the exact string `'<div>'`
2. Grab every character except `\n`
3. Until the FIRST instance of `'</div>'`

Sampling

- sample: a subset of population
- sampling frame: where we draw sample from

We may use **convenience sample** (non-random) and **probability sample** (random).



Sampling errors

Error	What it means	How to avoid
Selection bias	Systematically exclude (or favor) particular groups	Random sampling, increase overlap of sampling frame and population
Response bias	People do not always respond truthfully	Ask better questions with neutral and clear language, ensure anonymity
Non-response bias	People do not always respond	Increase response rate

Note: no sampling method prevents response bias!

Common sampling methods

- uniform random sample with replacement
- uniform random sample without replacement is called **simple random sample (SRS)**
- **stratified random sample**: sampling frame is divided into non-overlapping *strata* according to choose categories, and we carry SRS on each strata, after which we scale it by sampling frame size of each strata
- **probability sample**: any sample that we can find probability of

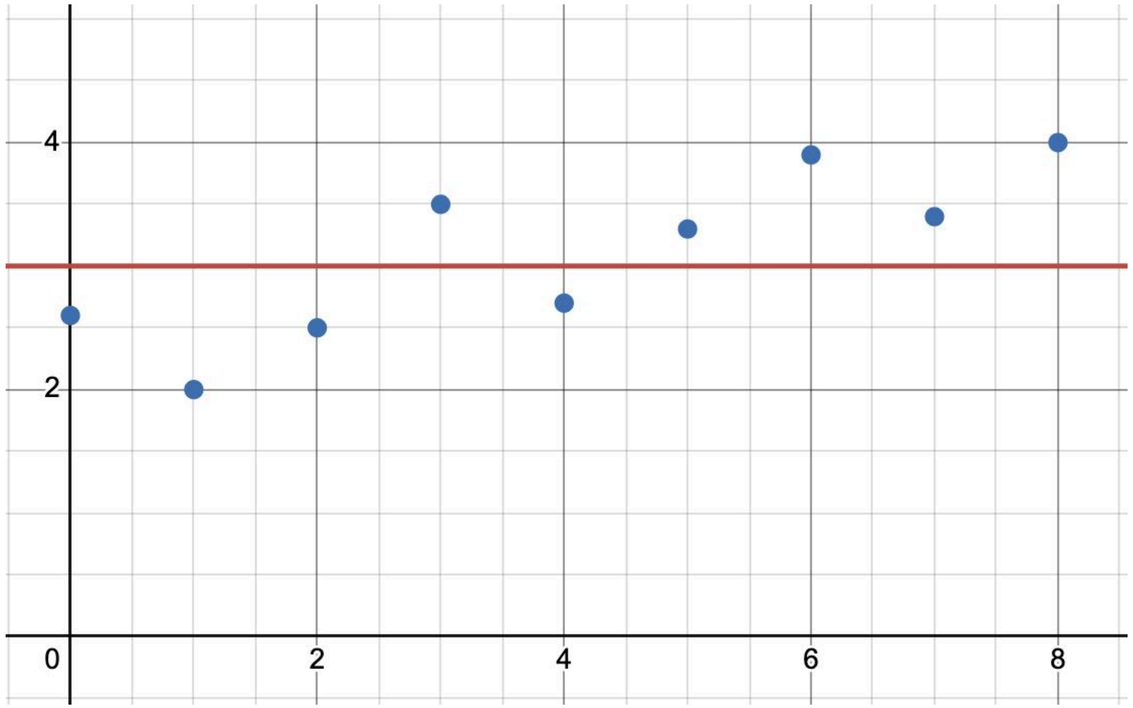
Modeling pipeline

1. Choose a model	Constant model, SLR model, OLS model, etc.
2. Choose a loss function	L1 loss (MAE), L2 loss (MSE), cross-entropy loss, etc.
3. Fit the model	Use the derivative of the cost function (average loss) to find a minimum
4. Evaluate the model	RMSE, residual plots

Constant Model

$$\hat{y} = \theta_0$$

- doesn't depend on input x
- always outputs the same number

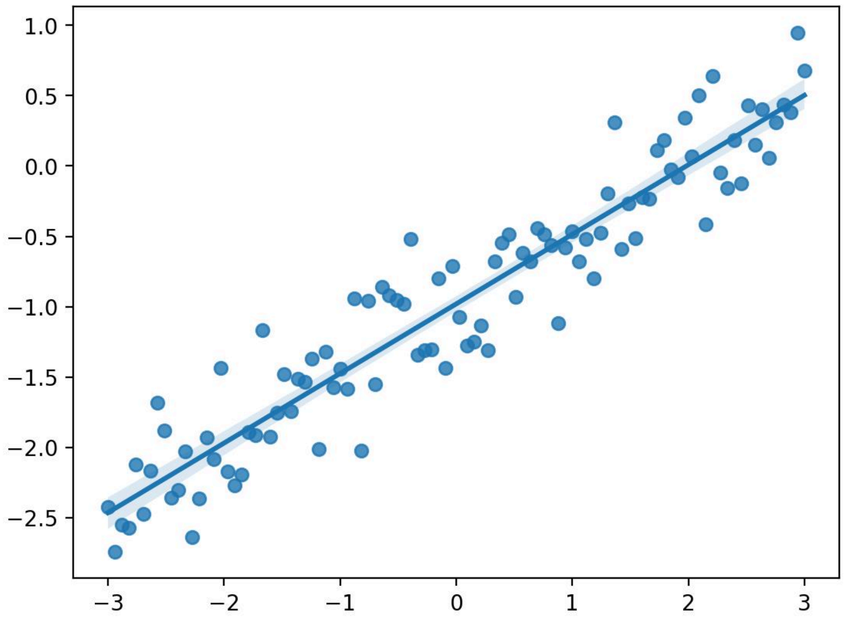


Simple Linear Regression (SLR)

$$\hat{y}_i = \theta_0 + \theta_1 x_i$$

predicted value — parameters — input/feature

$$\hat{\theta}_0 = \bar{y} - \hat{\theta}_1 \bar{x} \qquad \hat{\theta}_1 = r \frac{\sigma_y}{\sigma_x}$$



Note: SLR best fit line is unique if there is nonzero variance in the data points, yes!

Fitting the model

Best model = the one that **minimizes loss** (error)

How to minimize:

1. find **derivative** of loss function
2. find values of theta when derivative is zero
- if the function is convex, the point with **derivative=0** is the global minimum

Example

Define L2 loss function:

$$\frac{d}{d\theta_0} \mathrm{R}(\theta) = \frac{d}{d\theta_0} (\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2)$$

Use constant model for predicted y:

$$\frac{d}{d\theta_0} \mathrm{R}(\theta) = \frac{d}{d\theta_0} (\frac{1}{n} \sum_{i=1}^{n} (y_i - \theta_0)^2)$$

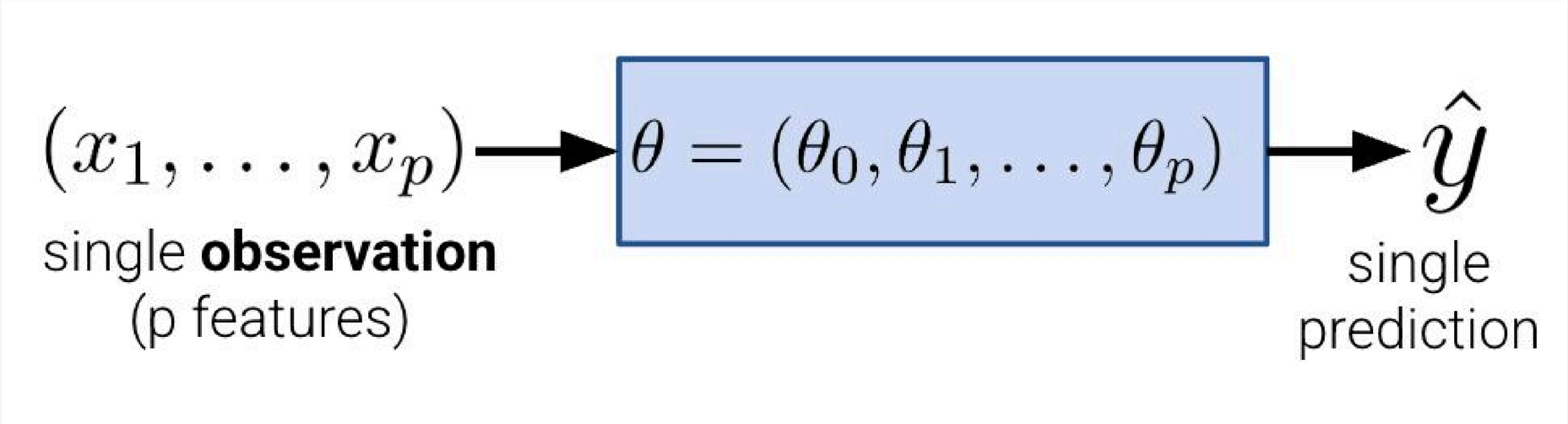
Set derivative to 0 and solve for theta:

$$0 = \frac{-2}{n} \sum_{i=1}^{n} (y_i - \hat{\theta}_0)$$

$$\hat{\theta}_0 = \bar{y}$$

OLS (Ordinary Least Squares)

Motivation: Multiple Linear Regression (We want to include more than 1 feature!)



Measuring Loss with OLS

MSE Metric from SLR, extended to this new matrix model.

$$R(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

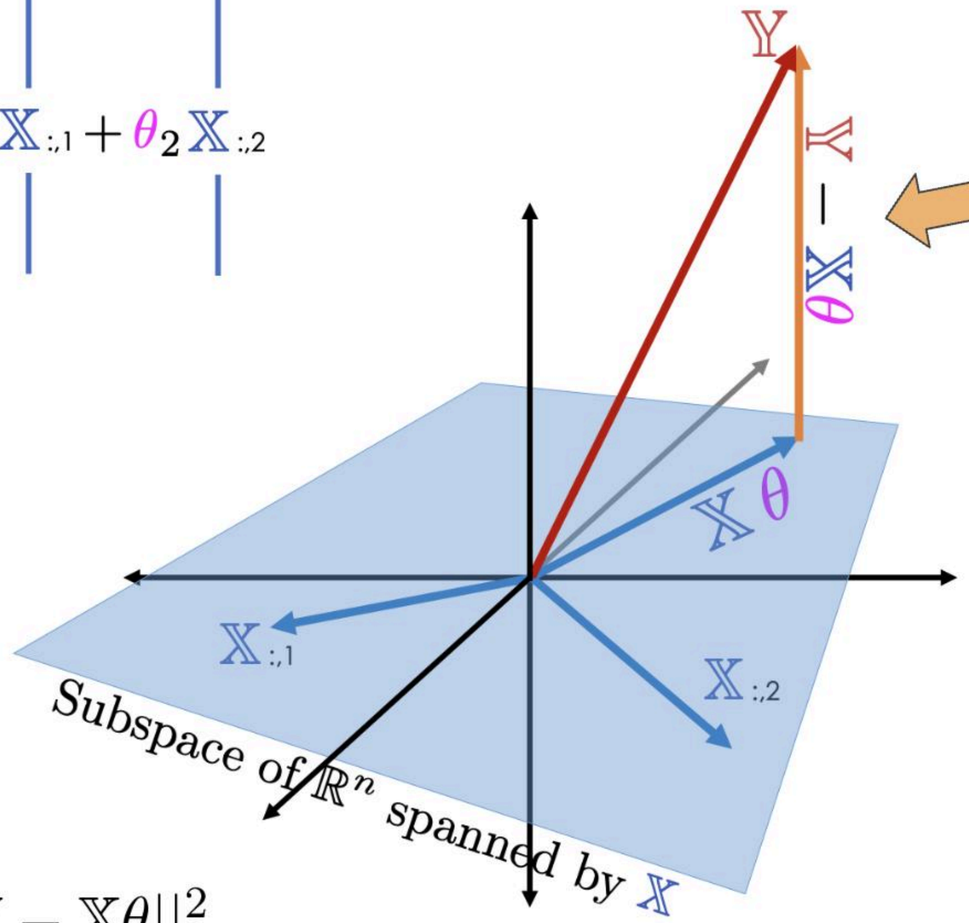
$$= \frac{1}{n}||\mathbb{Y} - \hat{\mathbb{Y}}||_2^2$$

Remember that our prediction vector is our design matrix * parameter vector

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

Geometric View of OLS

$$\begin{matrix} n \\ \\ \\ \\ 1 \end{matrix}\begin{bmatrix} | \\ \hat{\mathbb{Y}} \\ | \end{bmatrix} = \theta_1 \begin{bmatrix} | \\ \mathbb{X}_{:,1} \\ | \end{bmatrix} + \theta_2 \begin{bmatrix} | \\ \mathbb{X}_{:,2} \\ | \end{bmatrix}$$



This is the residual vector, $e = \mathbb{Y} - \hat{\mathbb{Y}}$.

Goal:
Minimize the *L2* norm of the residual vector.
i.e., get the predictions $\hat{\mathbb{Y}}$ to be "as close" to our true y values as possible.

$$R(\theta) = \frac{1}{n}||\mathbb{Y} - \mathbb{X}\theta||_2^2$$

Re

The Normal Equation

What is the optimal value of the parameter vector to minimize L2 Loss?

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

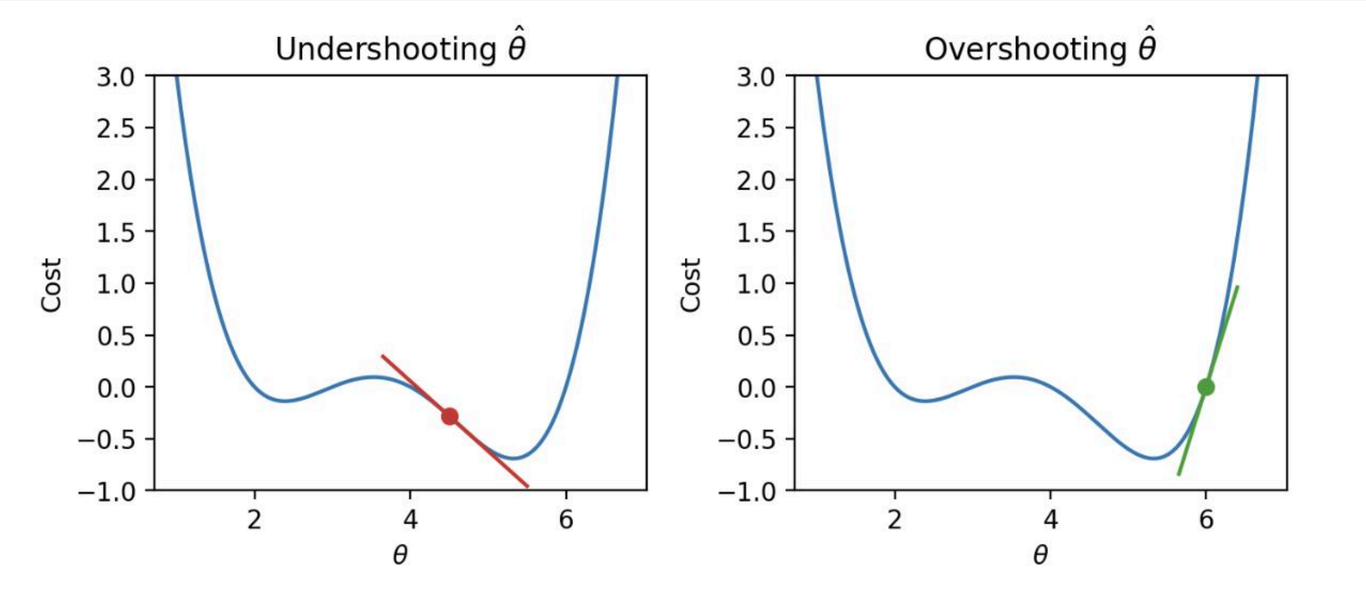
$\mathbb{X}^T\mathbb{X}$ has to be invertible!

	Model	Estimate	Unique?
Constant Model + MSE	$\hat{y} = \theta$	$\hat{\theta} = \mathbf{mean}(y)$	**Yes**. Any set of values has a unique mean.
Constant Model + MAE	$\hat{y} = \theta$	$\hat{\theta} = \mathbf{median}(y)$	**Yes**, if odd. **No**, if even. Return average of middle 2 values.
Simple Linear Regression + MSE	$\hat{y} = a + bx$	$\hat{a} = \bar{y} - \hat{b}\bar{x}$ $\hat{b} = r\frac{\sigma_y}{\sigma_x}$	**Yes**. Any set of non-constant* values has a unique mean, SD, and correlation coefficient.
Ordinary Least Squares (Linear Model + MSE)	$\hat{\mathbb{Y}} = \mathbb{X}\theta$	$\hat{\theta} = (\mathbb{X}^T\mathbb{X})^{-1}\mathbb{X}^T\mathbb{Y}$	**Yes**, if $\mathbb{X}$ is full col rank (all cols lin independent, # datapts >> # feats)

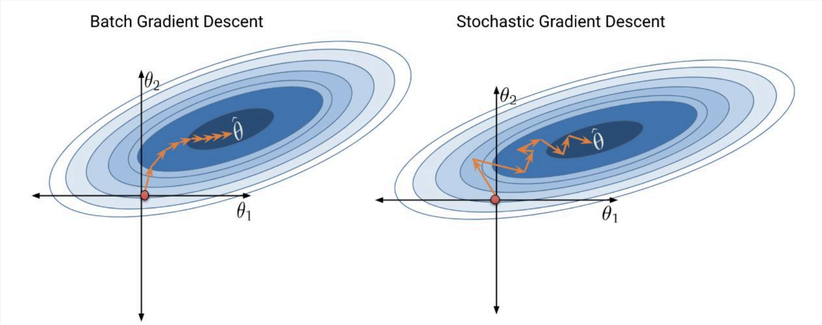
Gradient Descent

Sometimes, the loss function isn't as simple! Gradient descent is a new optimization technique for more complex loss functions.

BIG IDEA: use an iterative algorithm to numerically compute the minimum of the loss.



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



Regularization

- Why do we need regularization? We start to overfit!
  - We use too many features → model complexity increases
  - On the BVT curve, this corresponds to low bias, high variance!

LASSO (L1) Regression:

$$\frac{1}{n}||Y - X\theta||_2^2 + \lambda \sum_{j=1}^{d} |\theta_j|$$

- Encourages sparsity (drives some values of θ to 0)
- No closed-form solution for the optimal θ

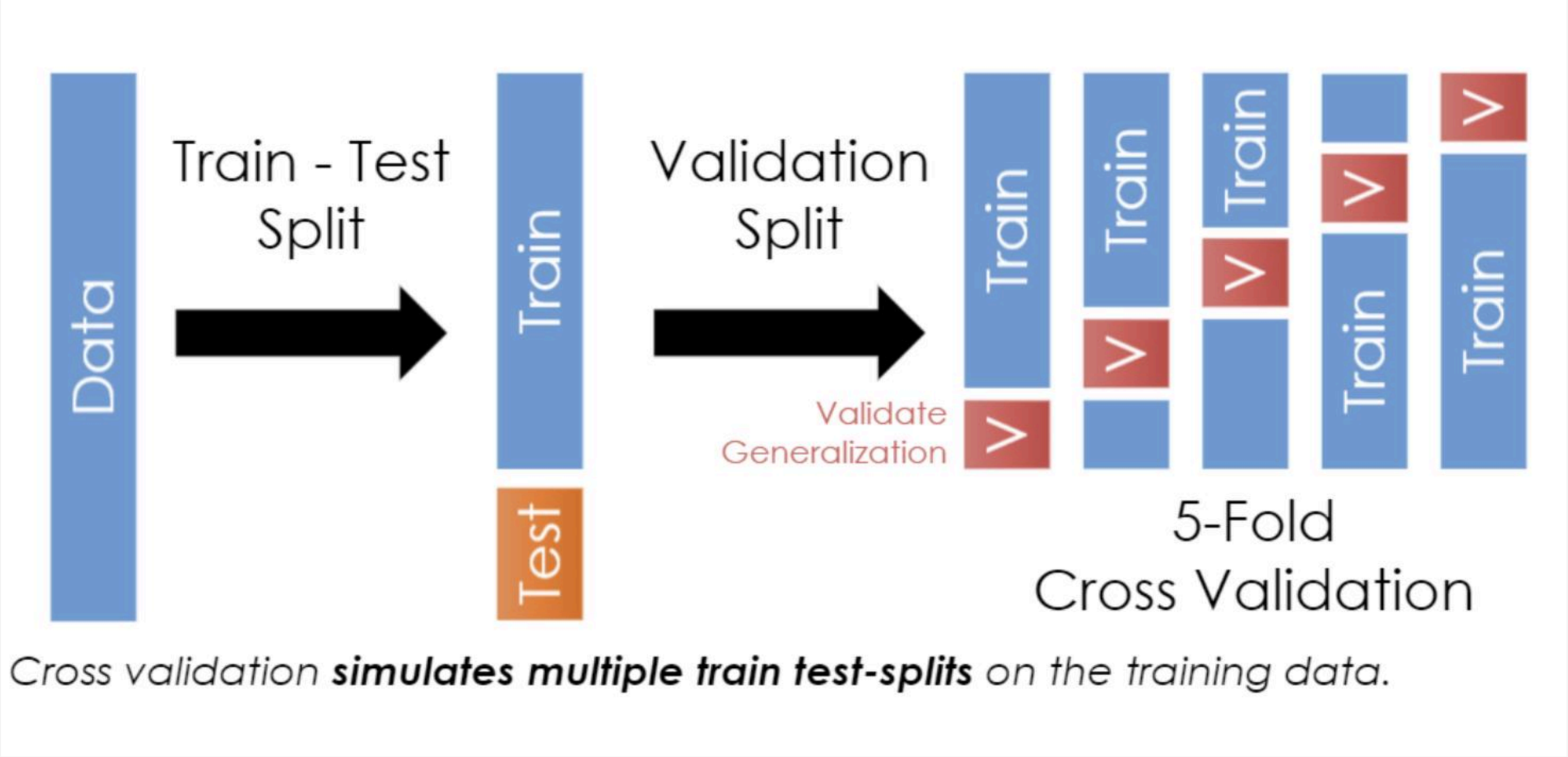
Ridge (L2) Regression:

$$\frac{1}{n}||Y - X\theta||_2^2 + \lambda \sum_{j=1}^{d} \theta_j^2$$

- "Robust" – tends to spread theta weights over many features
- Optimal θ can be found:

$$\hat{\theta} = \left(X^\top X + n\lambda I\right)^{-1} X^T Y$$

Cross Validation (k-fold)



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

Important application: Picking hyperparameters like lamdba (regularization)!

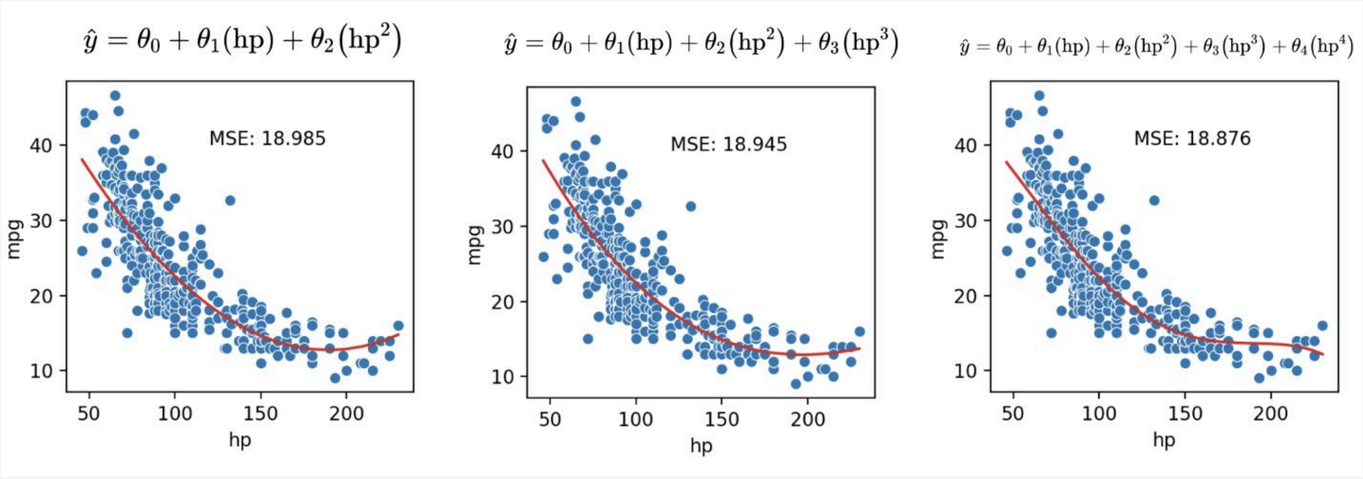
Feature Engineering

Definition: a process of transforming raw features into more informative features for use in modeling

Examples:

- One Hot Encoding
- Polynomial Features

Sunday	Thursday	Friday	Saturday	Bias
1	0	0	0	1
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1



Random Variable

Definition: a random variable (RV) represents the **outcome** of a random experiment

Discrete RV: can take only discrete values

- a listable number of possible outcomes and associated probabilities

Expectation: what outcome we expect from RV

Variance: measure of a RV's chance error

$$\mathbb{E}[X] = \sum_{x \in X} x \cdot \mathbb{P}(X = x)$$

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x \cdot f(x) dx$$

$$\text{var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

Continuous RV: can take only ranges of values.

- an unlistable number of possible outcomes
- probabilities are a continuous function (pdf)

- $I \sim \text{Bernoulli}(p)$, $X = \begin{cases} 1 & \text{w/ prob} = p \\ 0 & \text{w/ prob} = 1-p \end{cases}$ represents a single trial whose outcomes are success or failure. ex: a single coin flip

$P(X=1) = p$

$P(X=0) = 1-p$

- $X \sim \text{Binomial}(n, p)$ represents the number of successes in n independent Bernoulli(p) trials

ex: number of heads in n flips of a p-coin (p-coin = coin w/ prob heads = p)

$P(X=x) = \binom{n}{x} p^x (1-p)^{n-x}$

fair coin: $p = 1/2$, 10 flips, what's the probability I see 5 heads?

$P(X=5) = \binom{10}{5} (1/2)^5 (1-1/2)^{10-5}$

- $X \sim \text{Categorical}(p_1 \dots p_k)$

probability RV takes on value of each possible category $1, 2, \dots i \dots, k$

ex: a dice

$P(X=i) = p_i$

Formulas: Expectation, Variance, and Covariance

$$\mathbb{E}[X] = \sum_x x \cdot \mathbb{P}(X = x)$$

$$\text{var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

$$\mathbb{E}[\alpha X + b] = \alpha\,\mathbb{E}[X] + b$$

$$\text{var}(\alpha X + b) = \alpha^2\,\text{var}(X)$$

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y] \quad \text{linearity of expectation}$$

$$\text{var}(X + Y) = \text{var}(X) + \text{var}(Y) + 2\,\text{cov}(X, Y)$$

$$\text{cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] = \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y]$$

$$\text{cov}(X, X) = \text{var}(X)$$

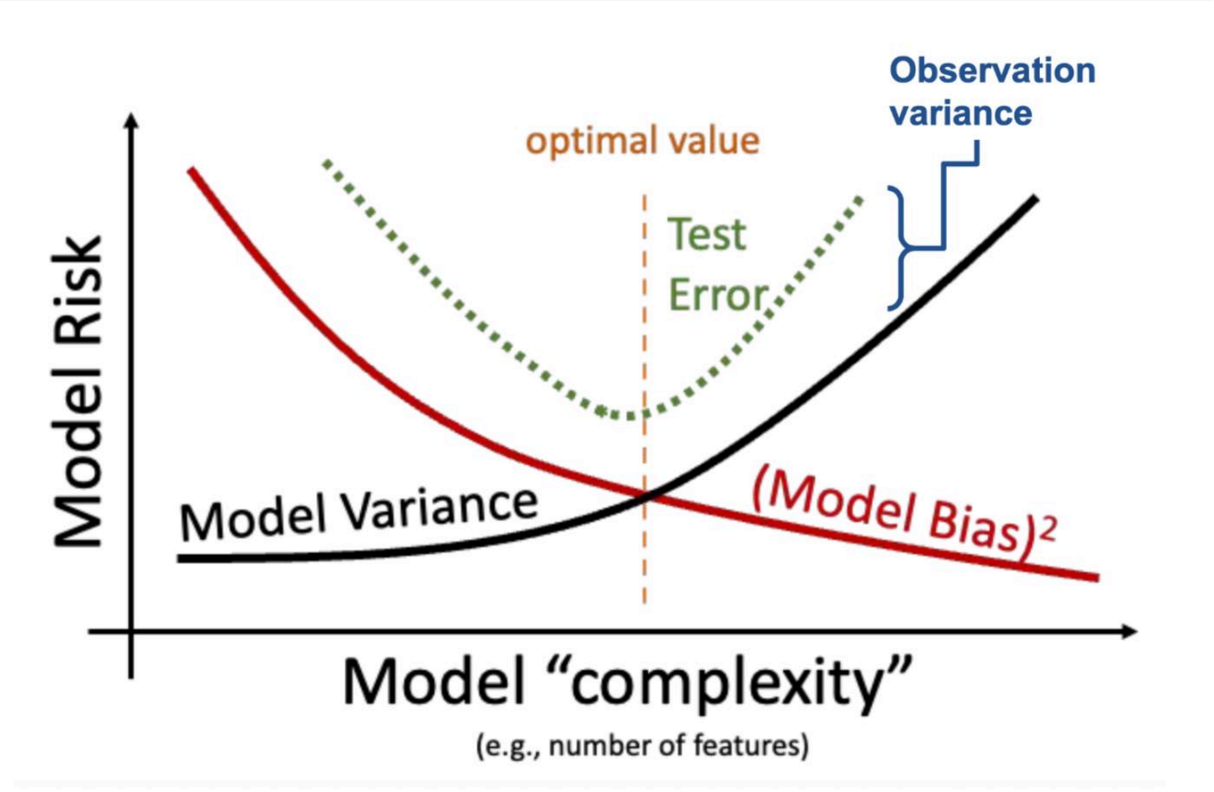
$$\text{cov}(X, Y) = \text{cov}(Y, X)$$

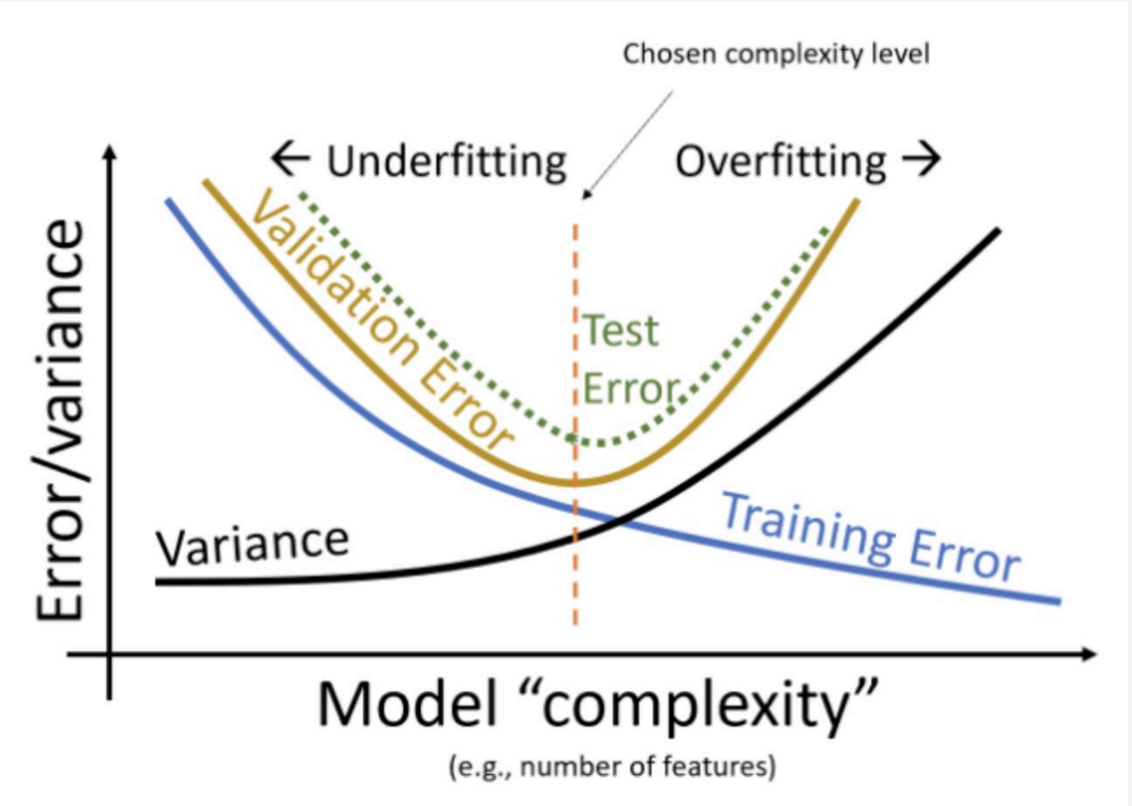
$$\boxed{\textit{If } X, Y \textit{ are independent random variables, then } \mathbb{E}[XY] = \mathbb{E}[X] \cdot \mathbb{E}[Y]}$$

when X, Y are independent, $\text{cov}(X, Y) = 0$ and $\boxed{\text{var}(X - Y) = \text{var}(X) + \text{var}(Y) = \text{var}(X + Y)}$

Bias-Variance Tradeoff

Key idea: as the model complexity increases, model bias decreases and model variance increases





Bias: how well model architecture is suitable for making predictions **Variance:** variability of model predictions

Formulas: Bias-Variance Decomposition

$$\textit{Model risk} = \textit{model bias}^2 + \textit{model variance} + \textit{irreducible error}$$

$$\mathbb{E}[(Y - \hat{Y})^2] = \textit{bias}(\hat{f}(X))^2 + \text{var}(\hat{f}(X)) + \sigma^2$$

$$\begin{aligned}
\mathbb{E}[(Y - \hat{Y})^2] &= \text{var}(Y - \hat{Y}) + (\mathbb{E}[Y - \hat{Y}])^2 \\
&= \text{var}(g(X) + \epsilon - \hat{f}(X)) + (\mathbb{E}[g(X) + \epsilon - \hat{f}(X)])^2 \\
&= \cancel{\text{var}(g(X))} + \text{var}(\epsilon) + \text{var}(\hat{f}(X)) + (\mathbb{E}[g(X)] + \cancel{\mathbb{E}[\epsilon]} - \mathbb{E}[\hat{f}(X)])^2 \\
&= \sigma^2 + \text{var}(\hat{f}(X)) + (g(X) - \mathbb{E}[\hat{f}(X)])^2
\end{aligned}$$

Model risk is $\mathbb{E}[(Y - \hat{Y})^2]$ *(i.e. Mean Squared Error)*

Model bias is $\textit{bias}(\hat{f}(X)) = \mathbb{E}[\hat{f}(X)] - g(X)$

Model variance is $\text{var}(\hat{f}(X)) = \mathbb{E}\left[\left(\hat{f}(X) - \mathbb{E}[\hat{f}(X)]\right)^2\right]$

Bootstrapping [Data 8]

Idea: creating parallel universes

Bootstrapping:

- treat the observed dataset as if it is a population
- repeatedly **sample with replacement** from the dataset to create many *bootstrap samples*
    - compute parameter on each bootstrap sample
- use bootstrapped parameter distribution

We can use bootstrap to estimate a parameter when we don't know the parameter's distribution

- e.g. parameter could be theta, mean, median, feature coefficient in regression, etc.

Parameter Inference

To infer causality, we require **randomization**.
However, it is not *always possible*. For example, we cannot run an experiment of randomly assigning college degrees to students and seeing if lifetime earnings in one group is higher. This is just unethical

Correlational inference: passive observation

- Are homes with granite countertops worth more money?
- Do people with college degrees have higher lifetime earning?
- Are people who smoke more likely to get cancer?

Causational inference: effects of interventions

- How much do granite countertops **raise** the value of a house?
- Does getting a college degree **increase** lifetime earnings?
- Does smoking **cause** cancer?

SQL syntax


```
SELECT *
FROM table1 AS a JOIN table2 AS b ON condition
WHERE condition
ORDER BY column DESC
GROUP BY column HAVING condition
LIMIT n;
```

Defining new column with cases:

```
    CASE WHEN condition THEN value
        [WHEN condition THEN value]
        ELSE value
        END
```

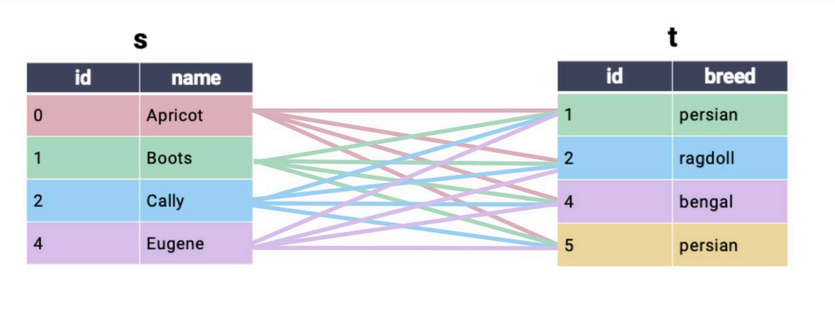
Note: we use single quotes for strings, double quotes and no quotes for table and column names

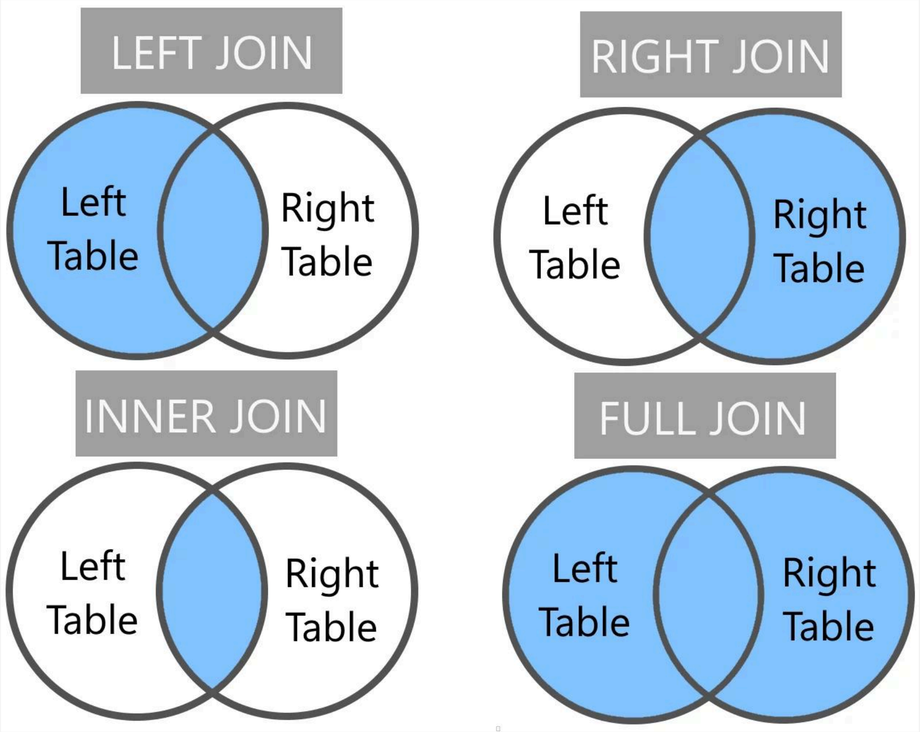
JOIN

Specify joins between tables as part of the FROM statement

```
SELECT *
FROM table1 INNER JOIN table2
ON table1.key = table2.key
```

There are also **cross-joins,** which is every combination of possible rows. Also called a cartesian product.





SQL Practice

Su25 Final Q1d Making Room for pandas

Josh samples information about various rooms in different buildings around Berkeley in preparation for the upcoming semester. He stores the information in a `DataFrame` called `rooms`. The columns of `rooms` are described below:

- `id`: A unique room id from Berkeley's internal database (type = `np.int64`).
- `building`: Name of the building the room is in (type = `str`).
- `number`: The room number listed on the door to the room (type = `str`).
- `college`: The Berkeley college each building is associated with from the select options of: `'CoE'`, `'CDSS'`, `'CNR'`, and `'LNS'` (type = `str`).
- `renovated`: The year the room was last renovated (type = `str`).
  Note: Certain rooms may never have been renovated, in which case `renovated` will contain a value of `None`.

The first five rows of `rooms` are shown below:

	id	building	number	college	renovated
0	100100	Evans	0060	LNS	2014
1	102101	Evans	0010	LNS	None
2	140144	Soda	320	CDSS	2021
3	310523	Gateway	101	CDSS	2025
4	668377	Cory	540AB	CoE	2002

(d) [3 Pts] Michael sends Josh a `DataFrame` called `details` that contains more detailed information about the rooms in `rooms`. Unfortunately, during the data transfer process, some rooms were dropped, so `details` only contains a subset of the rooms in `rooms`.

The columns of `details` are described below:

- `id`: A unique room id from Berkeley's internal database (type = `np.int64`).
- `capacity`: The room's maximum capacity (type = `np.int64`).
- `sqft`: The room's square feet (type = `np.float`).

The first five rows of `details` are shown below:

	id	capacity	sqft
0	100100	102	1504.50
1	140144	237	2075.25
2	888888	9	200.20
3	123456	45	500.00
4	123123	40	345.67

Help Josh join `rooms` and `details` together to create a `SQL` table `detailed_rooms`.

1. Josh does not want to drop any rows of `rooms` in the merge, even if `details` doesn't contain a match.
2. `detailed_rooms` should contain a new column called `size` of the `VARCHAR` (i.e., string) type. The `size` of a room is defined as `'unknown'` if the room has no recorded `capacity`, `'small'` if the `capacity` is less than 30, `'medium'` if the `capacity` is between 30 and 99 (inclusive), and `'large'` otherwise.

Josh does not want to drop any rows of `rooms` in the merge, even if `details` doesn't contain a match.

Fill in the blanks below. Assume that `duckdb` is imported and `rooms` and `details` can be queried as SQL tables. Treat the `None` type in Python as `NULL` in SQL. Treat `DataFrame` names as SQL table names in your query.

```
SELECT r.id, _____(i)_____,
CASE
    _____________
    _____________
    _____(ii)_____
    _____________
    _____________
FROM rooms AS r
_____(iii)_____ JOIN details AS d
    ON _____(iv)_____;
```

Sp21 MT2 Q1

Regular Ice, SQLite Sugar

Marshall Mathers' Data 100 study group is holding a mid-semester social on Memorial Glade, and he is tasked to get bubble tea (i.e., boba). Since it's a Data 100 event, he surveys the students and staff for some information about the boba shops they've been to in Berkeley, and compiles the data into a table named `BobaReview`. Each row is a person's review of a boba shop; the first few rows are shown to the right.

reviewer	role	shop_name	rating	price
Kelly	staff	Boba_Ninja	2	5
Andrew	staff	Asha	4.5	5
Parth	student	Asha	5	5
Kunal	staff	One_Plus	4.5	4.5

(a) [2 Pts] Marshall wants to find all the reviews that rated a shop at least 4 (out of a maximum rating of 5). Help Marshall by writing a one-line SQL query that outputs all rows of `BobaReview` where a boba shop was rated **at least** 4 out of 5.

_____;

(b) [4 Pts] Marshall now wants to find the **5 top-rated boba shops** in Berkeley. Help Marshall by completing the below SQL query, which selects the 5 boba shops from `BobaReview` with the highest average rating. Your query result should have three columns labeled `shop_name`, `avg_rating`, and `avg_price`, which are the shop name, average rating, and average price paid for boba, respectively. Rows should be sorted by the average rating, with the highest rated boba shop as the first row. The first few rows of the query result are shown below:

shop_name	avg_rating	avg_price
Asha	4.75	5.0
One_Plus	4.5	4.5

```
SELECT ____(i)____ FROM BobaReview ____(ii)____;
```

(i) What goes in the blank denoted by `(i)`?

`SELECT` _____

(ii) What goes in the blank denoted by `(ii)`? Use as few/many lines as you need.

Logistic regression

Idea: predict probabilities for each class, and pick class with highest probability

$$\underbrace{\hat{P}_\theta(Y = 1|x)}_{\text{Estimated probability that the response is 1, given the features x}} = \underbrace{\frac{1}{1 + e^{-x^T\theta}}}_{\text{Sigmoid function } \sigma(\ ) \text{ at the value } x^T\theta}$$

Note: logistic regression is a classification model for predicting classes, not a regression model!

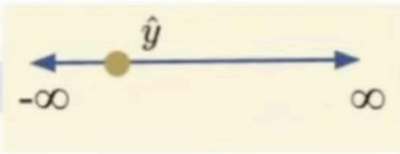
Motivation

GAME_ID	TEAM_NAME	MATCHUP	REB	FTM	TOV	GOAL_DIFF	WON
21700001	Boston Celtics	BOS @ CLE	46	19	12	-0.049	0
21700002	Golden State Warriors	GSW vs. HOU	41	19	17	0.053	0
21700003	Charlotte Hornets	CHA @ DET	47	23	17	-0.030	0
21700004	Indiana Pacers	IND vs. BKN	47	25	14	0.041	1
21700005	Orlando Magic	ORL vs. MIA	50	22	15	0.042	1

Input: numeric features

$x^\top \theta$

Model: linear combination



Output: numeric prediction

GAME_ID	TEAM_NAME	MATCHUP	REB	FTM	TOV	GOAL_DIFF	WON
21700001	Boston Celtics	BOS @ CLE	46	19	12	-0.049	0
21700002	Golden State Warriors	GSW vs. HOU	41	19	17	0.053	0
21700003	Charlotte Hornets	CHA @ DET	47	23	17	-0.030	0
21700004	Indiana Pacers	IND vs. BKN	47	25	14	0.041	1
21700005	Orlando Magic	ORL vs. MIA	50	22	15	0.042	1

Input: numeric features

$p = \sigma(x^\top \theta)$

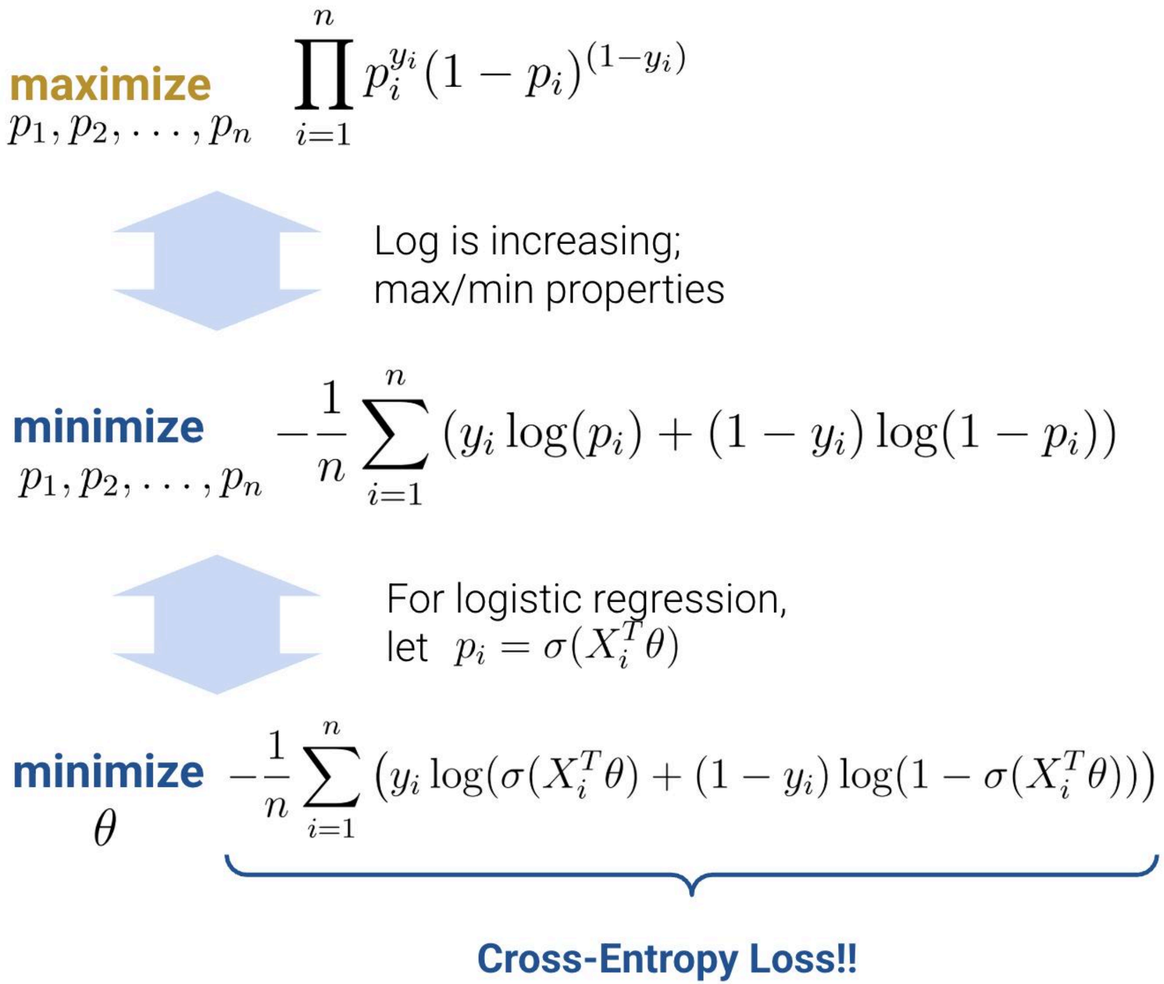
Model: linear combination transformed by non-linear **sigmoid**

Win?
If p > 0.5: predict a win
Other: predict a loss

Decision rule

Output: **class**

Cross-Entropy loss function



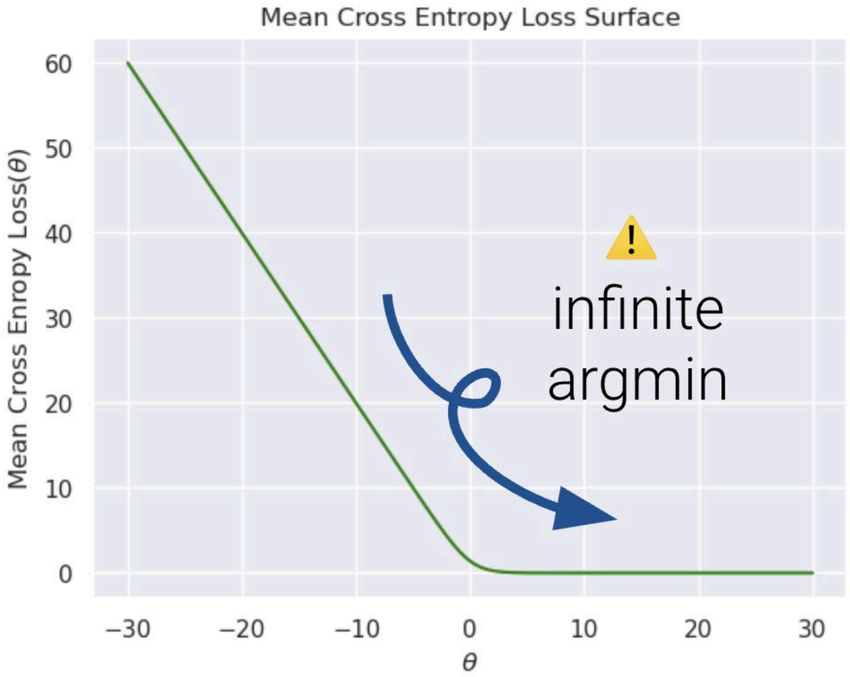
Evaluate a logistic regression model

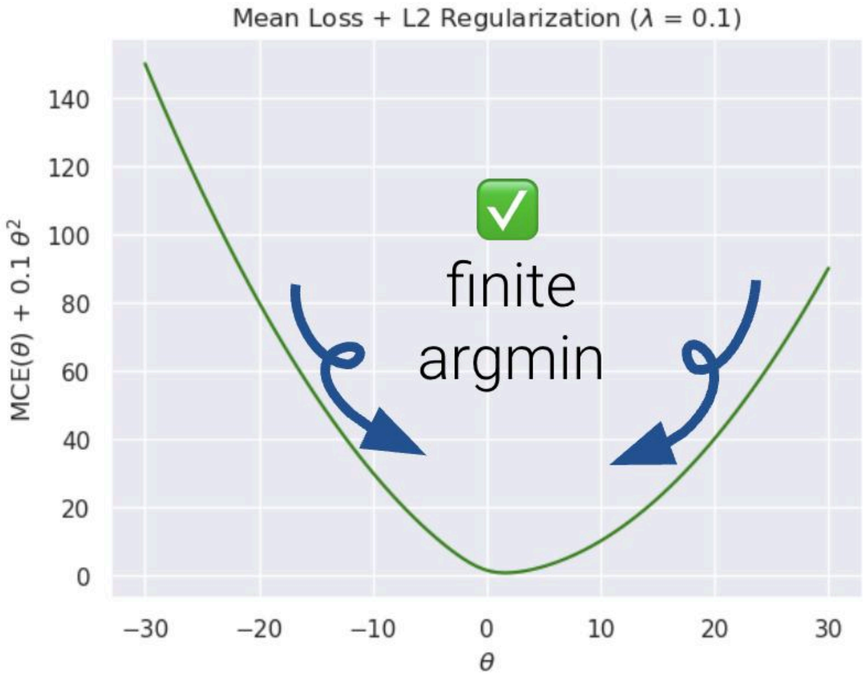
We use **confusion matrix** with FP, FN, TP, and TN and **ROC** curve

Precision: TP / (TP + FP)
Recall: TP / (TP + FN)

Regularized logistic regression

$$\underset{\theta}{\text{argmin}} -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(\sigma(X_i^T\theta) + (1-y_i)\log(1-\sigma(X_i^T\theta))\right) + \lambda\sum_{j=1}^{d}\theta_j^2$$





Logistic Regression Practice

SP22 MT2 Q3a

Edison Takeover

The `DataFrame` `ed_replies` is created for an Ed reply predictor.

The columns of `ed_replies` are described below:

- `ed_id`: The Ed reply's unique id (type = `np.int64`).
- `length`: The number of characters in the reply (type = `np.int64`).
- `likes`: The number of likes the reply received (type = `np.int64`).
- `response_time`: The number of minutes between the time of the reply and the time of its original post (type = `np.int64`).
- `bot`: Indicates if the reply was made by an Ed bot. `0` for a human and `1` for an Ed bot (type = `np.int64`).

The five rows of `ed_replies` are shown below:

	ed_id	length	likes	response_time	bot
0	0	150	0	65	0
1	1	467	2	123	0
2	2	23	5	21	1
3	3	145	0	1	1
4	4	256	1	1	1

(a) [2 Pts] Justin wants to fit a logistic regression model to predict whether a particular Ed reply is from a bot or a real person. Select all statements below that are true.

- ☐ Let $\mathbb{X}$ be the design matrix. The quantity $\mathbb{X}^\top \mathbb{X}$ must be invertible in order to compute the optimal parameter vector.
- ☐ It is possible for the sigmoid function σ to output exactly $P(y_i = 1|x_i) = 1$ for some data point x_i with real-valued features.
- ☐ The model predictions on the probability scale are a nonlinear function of inputted features.
- ☐ If the training dataset is linearly separable, it is impossible for gradient descent to converge on a unique optimal $\hat{\vec{\theta}}$.

SP22 MT2 Q3b

Edison Takeover

(b) [1.5 Pts] Justin fits the following logistic regression model to predict the probability that a reply was made by an Ed bot:

$$P(y = 1|\vec{x}) = \sigma(\hat{\theta}_1 \times \texttt{likes} + \hat{\theta}_2 \times \texttt{response\_time});\ \hat{\theta}_1 = 1,\ \hat{\theta}_2 = 2$$

Based on the fitted model above, what are the estimated **odds** that a reply with `likes = 3` and `response_time = 34` was made by a **human**? *If preferred, feel free to use σ in your solution. You do not need to simplify algebra.*

SP22 MT2 Q3c

Edison Takeover

(c) [3.5 Pts] Justin fits a different classification model on the training data. He applies this model to a test set and gets the following results.

True Label	1	1	1	0	0
Prediction	1	0	1	0	0
Outputted probability (y = 1)	0.88	0.4	0.75	0.3	0.54

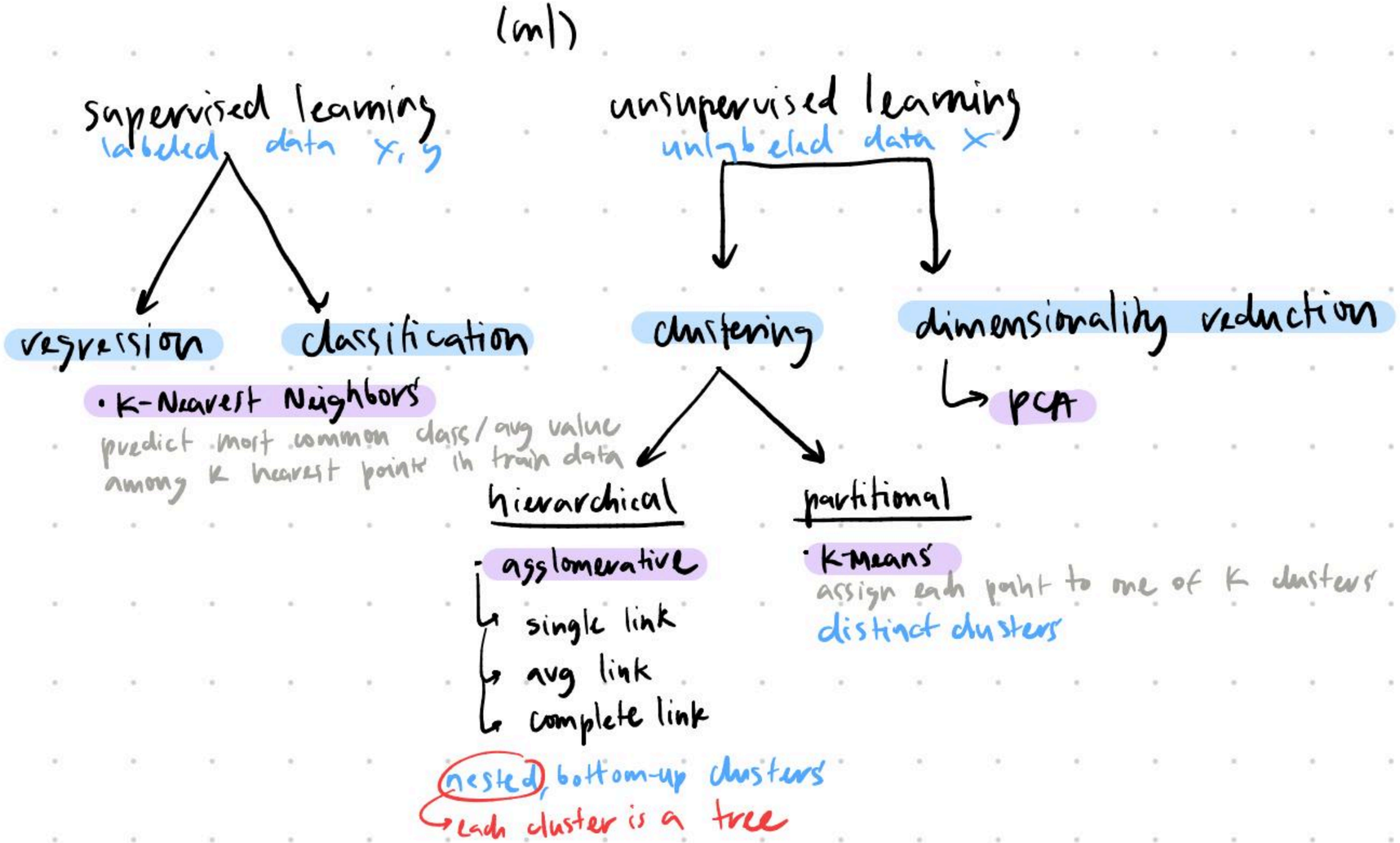
(i) [1 Pts] Given the values in the table above, what are the minimum and maximum possible values of Justin's chosen threshold? *For full credit, your answer does not need to address whether the bounds are inclusive or exclusive.*

(ii) [1 Pts] Calculate the recall of the model on the test set. *You do not need to simplify algebra.*

(iii) [1.5 Pts] Select all statements below that are true.

- ◯ True ◯ False After reducing the classification threshold from 0.9 to 0.1, it is possible for the true positive rate to increase.
- ◯ True ◯ False It is possible for the F_1 score of a classifier to change after changing the threshold.
- ◯ True ◯ False It is possible for AUC-ROC to change after changing the threshold.

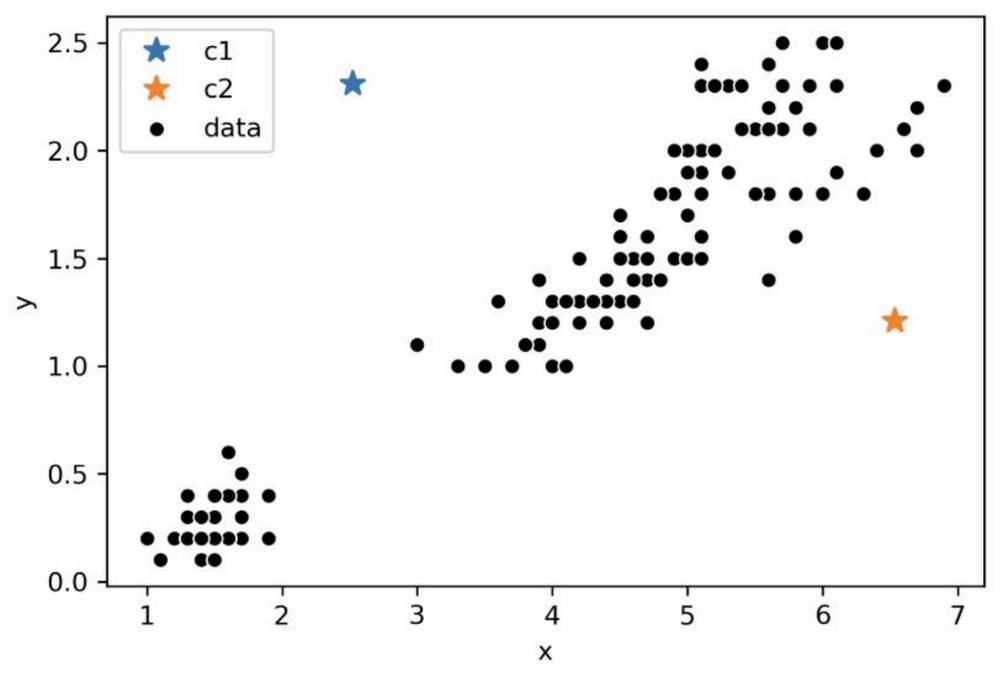
Motivation


(ml)
supervised learning
labeled data x, y
unsupervised learning
unlabeled data x
regression
classification
clustering
dimensionality reduction
PCA
• K-Nearest Neighbors
predict most common class / avg value among K nearest point in train data
hierarchical
partitional
• agglomerative
single link
avg link
complete link
nested, bottom-up clusters
each cluster is a tree
• K-means
assign each point to one of K clusters
distinct clusters

Clustering

K-Means algorithm

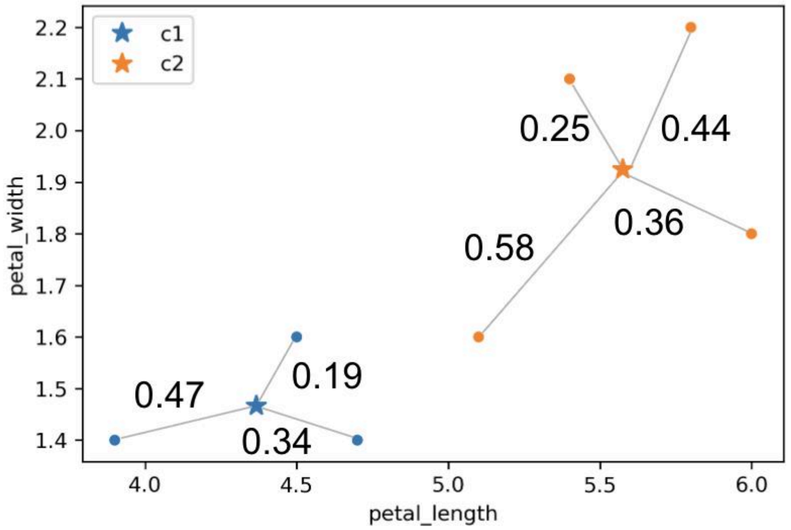
1. Pick K random points as centers of different clusters
2. Repeat until convergence:
    - pick unassigned point, assign to the cluster with the closest center
    - move center for that color to center of points with that color



Evaluate a clustering model

Two common loss functions:

- **Inertia**: sum of squared distance from each data point to its center
- **Distortion**: *weighed* sum of squared distance from each data point to its center, where each point is inversely weighed by total # of points in cluster



Inertia: $0.47^2 + 0.19^2 + 0.34^2 + 0.25^2 + 0.58^2 + 0.36^2 + 0.44^2$

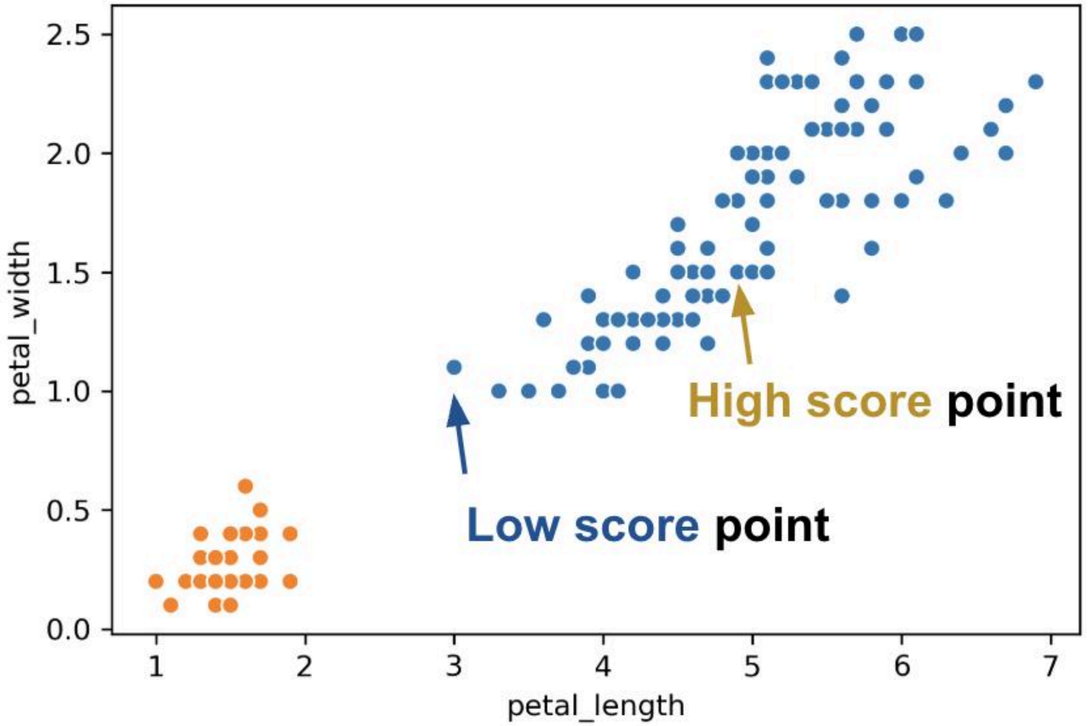
Distortion: $(0.47^2 + 0.19^2 + 0.34^2) / 3 + (0.25^2 + 0.58^2 + 0.36^2 + 0.44^2) / 4$

Note: if we have K = n data points, we overfit and our inertia is 0; K is a hyperparameter

Silhouette score

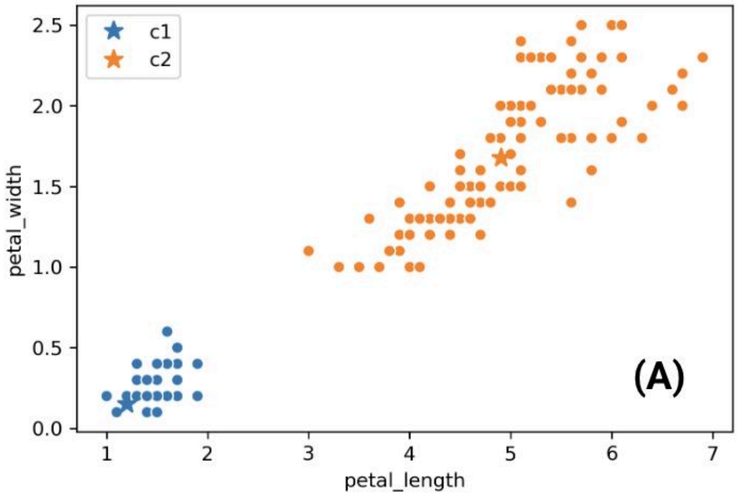
S = (B - A) / max(A, B) [where A is average distance to points in SAME cluster, B = to CLOSEST]

- High score: Close to points in its own cluster. Far from points in other clusters.
- Low score: Far from points in its own cluster. Close to points in other clusters.

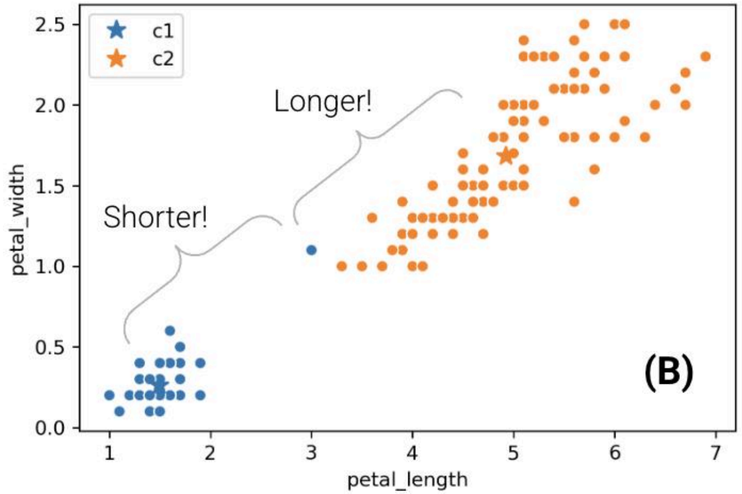


Cons: K-Means is sensitive to outliers

The inertia on the right is lower; however, clusters on the left make more intuitive sense



Inertia: 94.41



Inertia: 87.28

Note: K-Means is best when clusters are spherical shaped

Hierarchical agglomerative clustering algorithm

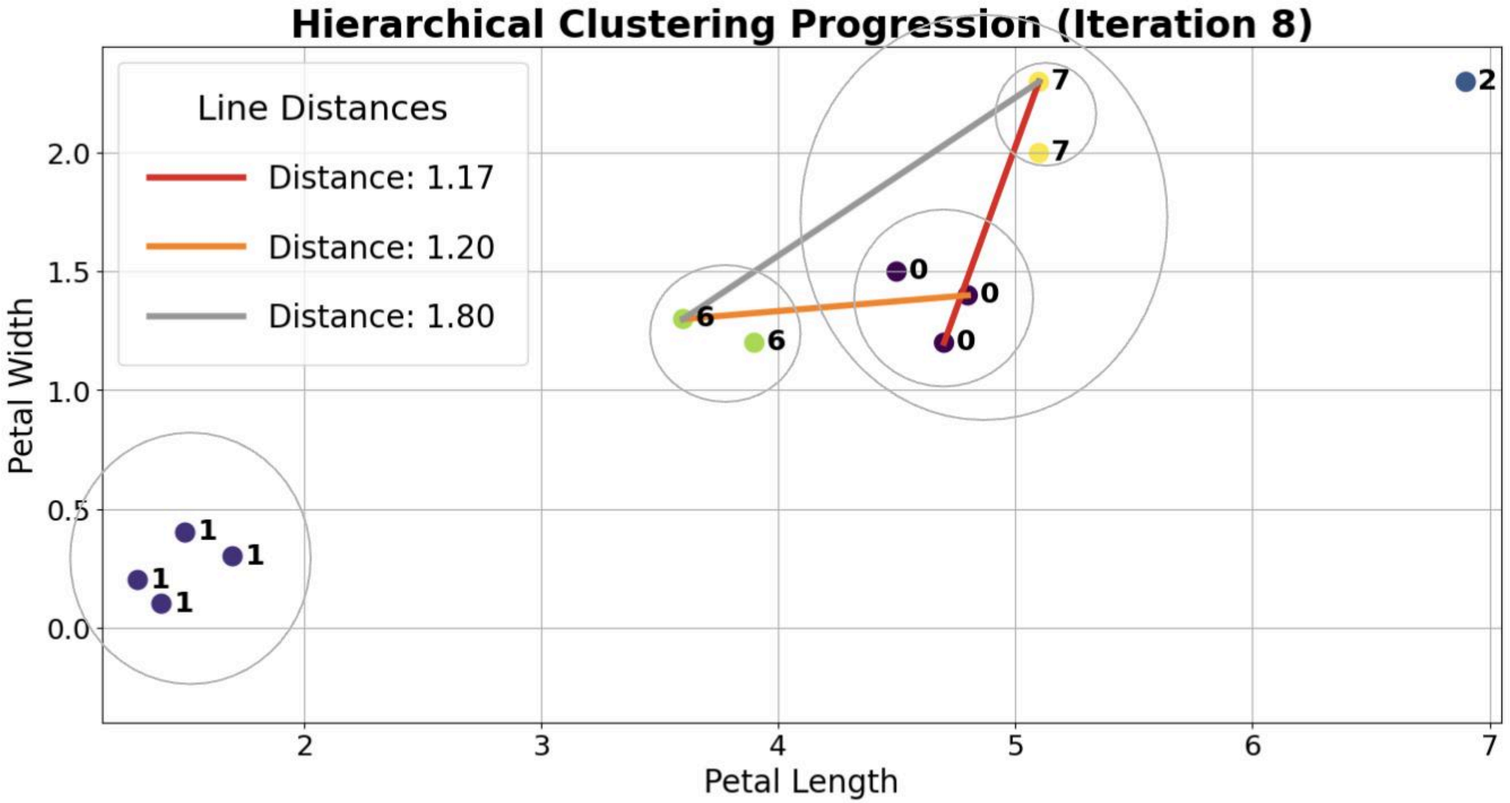
1. Initialize each point as its own cluster
2. Repeat until convergence:
    - merge the most similar *clusters* according to linkage criteria, until we have just one giant cluster

Linkage criteria:

- **Single linkage**: min distance between two points in different clusters "bumpy" shaped clusters
- **Average linkage**: average of all pairwise distance between clusters average between single and complete linkage
- **Complete linkage**: spherically shaped, compact, well separated clusters
    - for every pair of clusters, compute max distance between any two points
    - merge the cluster pair with the smallest max distance

Note: we go *bottom-up*, agglomerating clusters on the go!

Cons: hierarchical agglomerative clustering is also sensitive to outliers



Note: can see hierarchy in relationships!

Clustering Practice

SP23 Final Q11

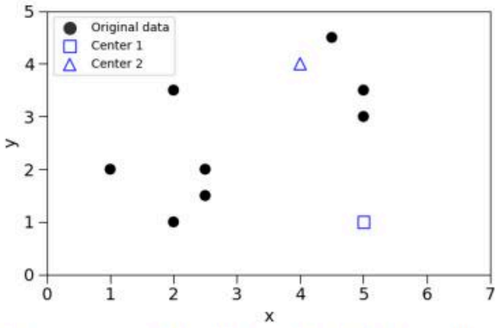
Cluster it All

You are given the following eight (x, y) pairs to perform a clustering task:

$$\{(2,1), (2.5,1.5), (2.5,2), (1,2), (5,3), (4.5,4.5), (2,3.5), (5,3.5)\}$$

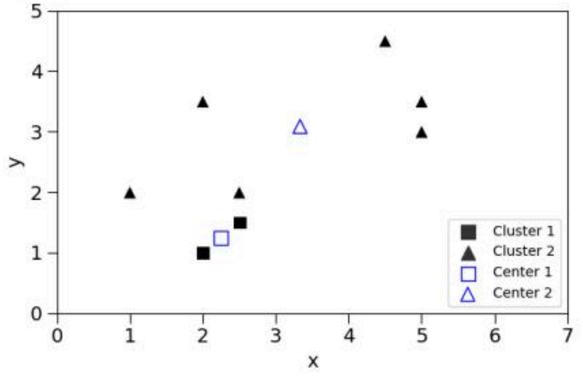
(a) [4 Pts] Suppose you ran the K-Means algorithm to identify TWO clusters in the data, with the initial assignment of cluster centers as Center 1: $(5,1)$ and Center 2: $(4,4)$.

In other words, you have the starting state to the right, where Center 1 and Center 2 are the outlined square and triangle markers, respectively.

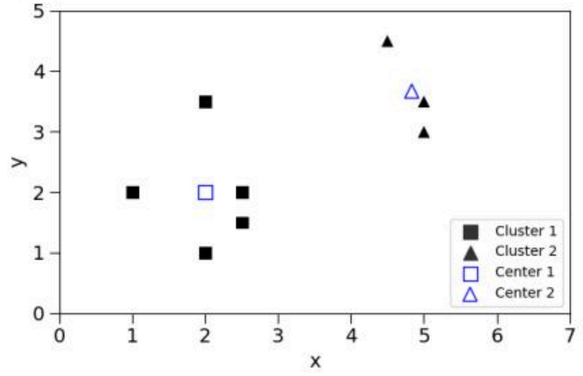


What is the result of K-Means after **ONE** iteration? All figures use the legend in Choice A, which lists the markers for the two clusters and two centers.

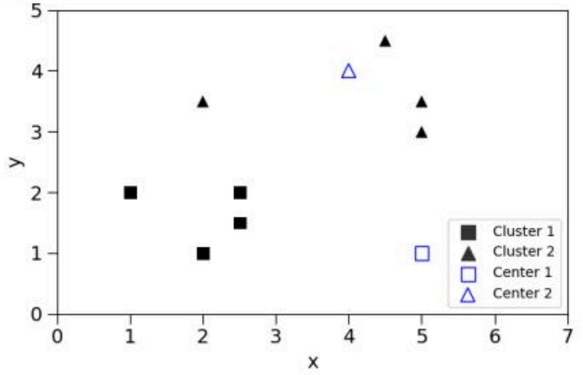
◯ A.



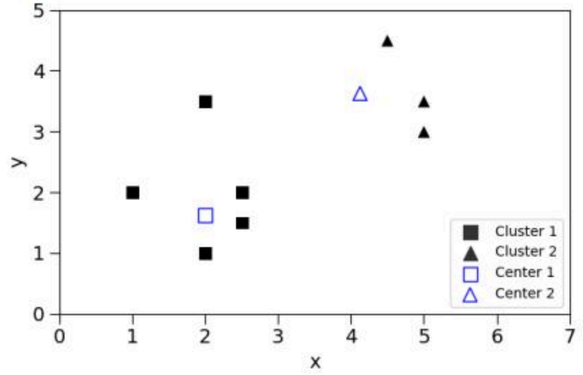
◯ B.



◯ C.



◯ D.

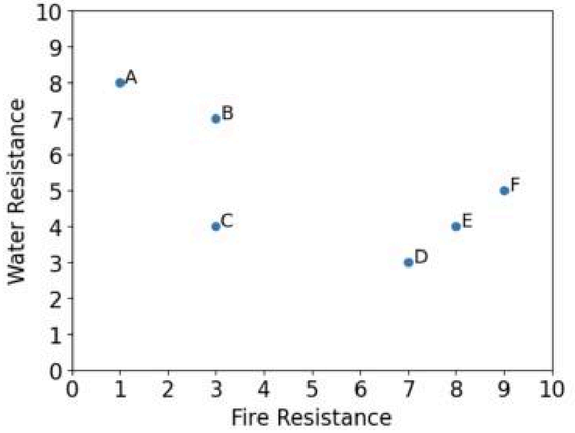


Su24 Final Q2

Pikachu, Charizard, and Arceus

Joining James' adventure in the Safari Zone, Willy wants to use clustering methods to see whether he can group individual Pokémon together. After collecting more data points, he logs two features: a Pokémon's resistance against water and fire, measured on a continuous scale from 1 to 10. The dataframe and a scatter plot have been added for your convenience. For the following questions, assume that distances are calculated using Euclidean distance.

	Fire Resistance	Water Resistance
A	1	8
B	3	7
C	3	4
D	7	3
E	8	4
F	9	5



(e) [1 Pt] Willy decides to perform hierarchical agglomerative clustering using complete linkage.

(i) Which two points are the first to merge into a new cluster? **Write your answer using the provided labels.**

(ii) Willy merges two clusters again. Which points are included in this most recent merged cluster? **Write your answer using the provided labels.**

Su24 Final Q2

Pikachu, Charizard, and Arceus

(f) [2 Pts] Willy decides to start over using K-Means clustering to separate data into two clusters, with the left cluster being centered at $(1, 4)$ and the right cluster being centered at $(6, 7)$.

(i) What points are the left clusters composed of? **Write your answer using the provided labels.**

(ii) What is the new centroid of the left cluster?

(iii) Calculate the inertia of the left cluster, using the centroid from (ii).

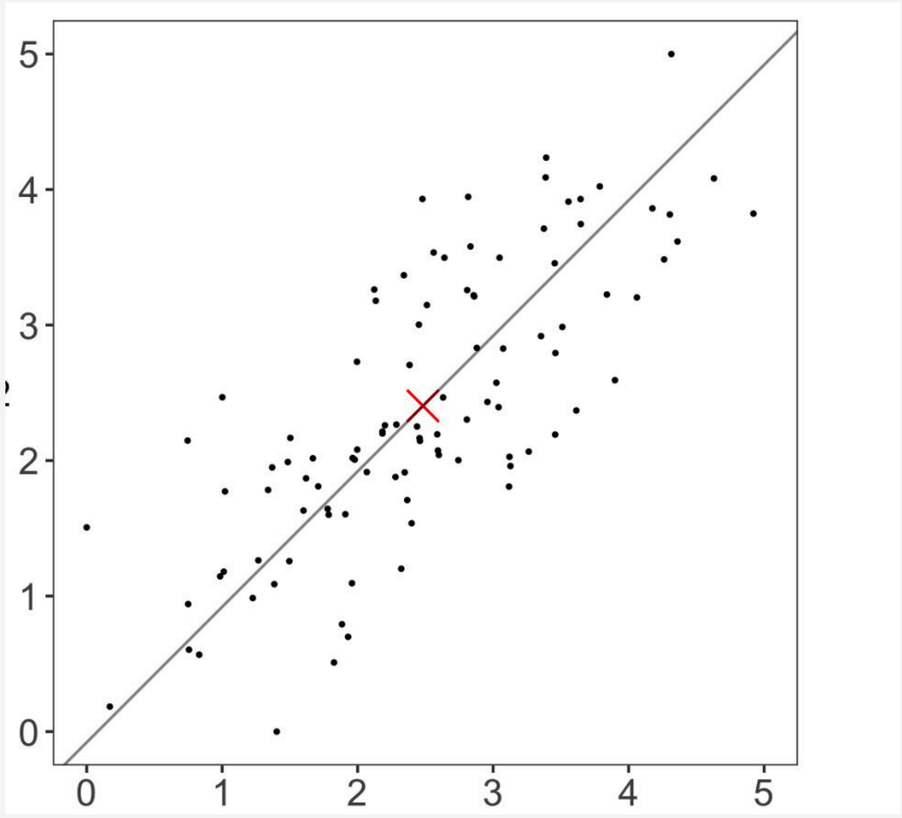
	Fire Resistance	Water Resistance
A	1	8
B	3	7
C	3	4
D	7	3
E	8	4
F	9	5

Principal Component Analysis

Big idea: reduce the dimensionality of our data while staying true to it (capturing max amount of variation).

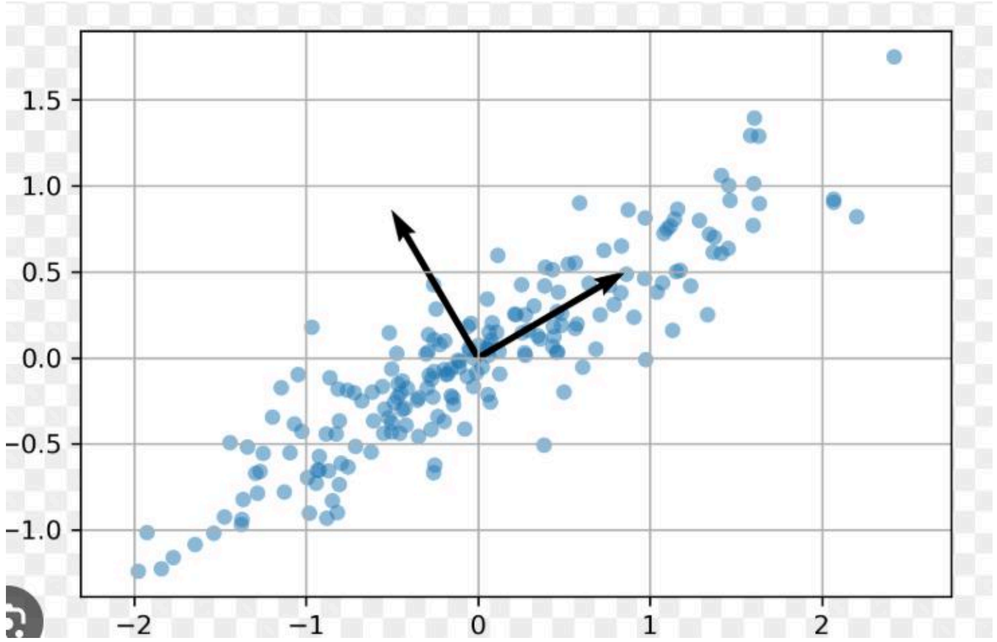
- The data on the right approximately lives on a line even though it is two dimensional
- We want to find ***a change of basis*** of our variables so that each feature we use maximizes the amount of information being represented

Each principal component (PC) tries to capture ***max variance*** of the data towards one direction.



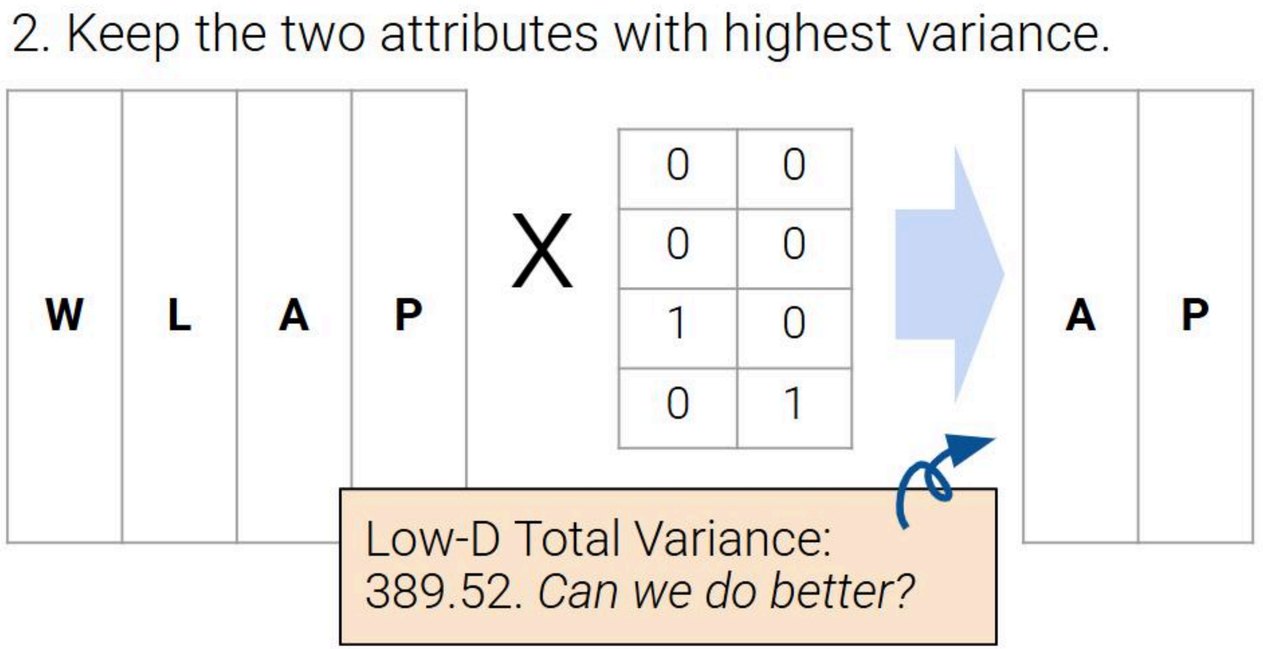
Principal Component Analysis

Task: find the directions in our data that capture the most variance in our data.



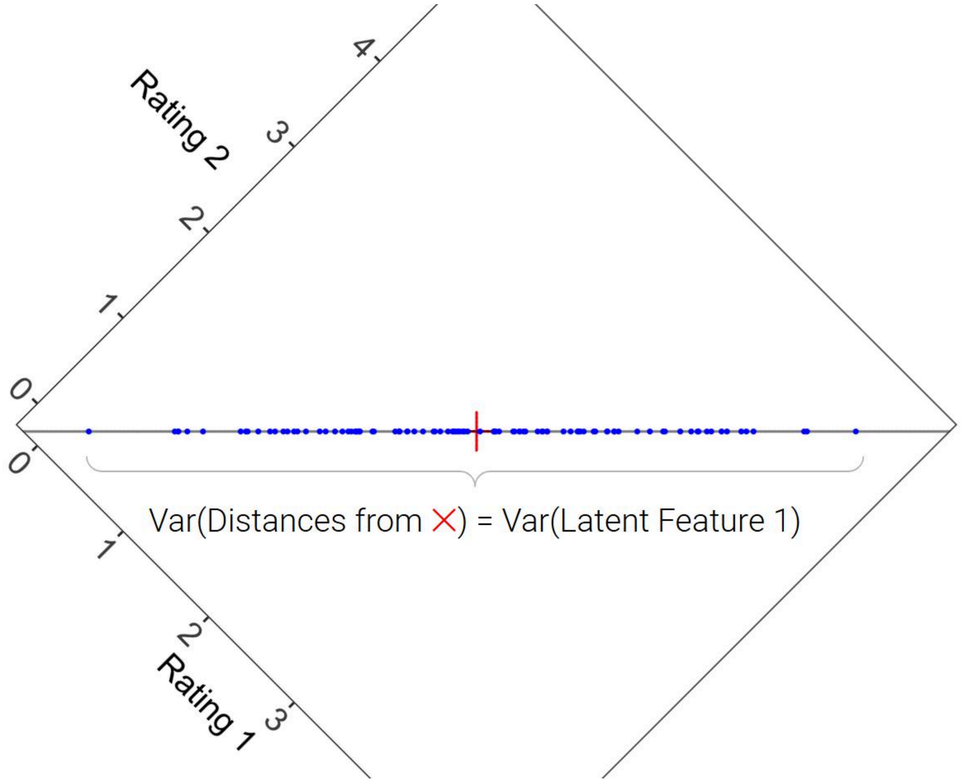
Principal Component Analysis

We can keep the top k directions that capture the most variance. These represent projections of our data onto the most variance-capturing directions in our data.



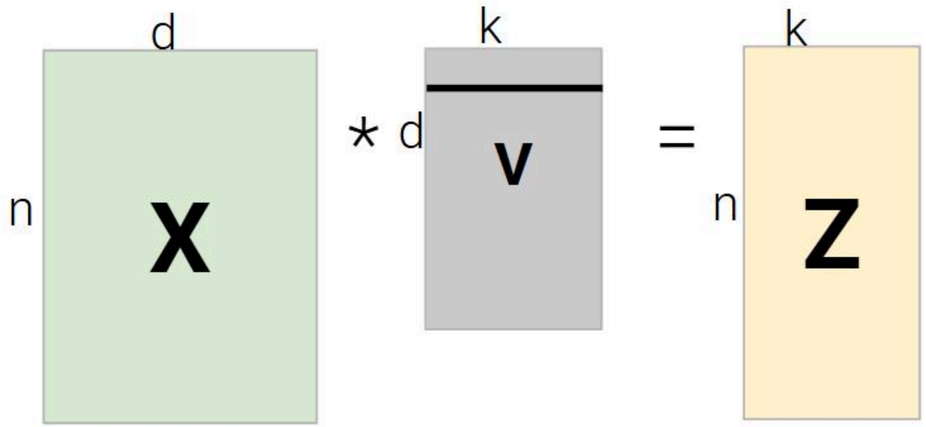
Principal Component Analysis

After finding the directions of maximum variance, we change the basis of our coordinate system to these directions.



Principal Component Analysis

Once we obtain our principle component matrix, we can use this as as change of coordinate transformation.



The i-th row of V indicates **how much feature *i* contributes** to each PC. Cols of V are the PCs ("recipes").

"First row of V = [0.9,-0.44] → PC1 linear combo is 0.9 parts feature 1, and PC2 has -0.44 parts feature 1."

The principal component matrix can be right multiplied to your feature matrix to obtain a matrix of latent features, called Z.

Note: you do not need to know how to compute this matrix in Fall 2025

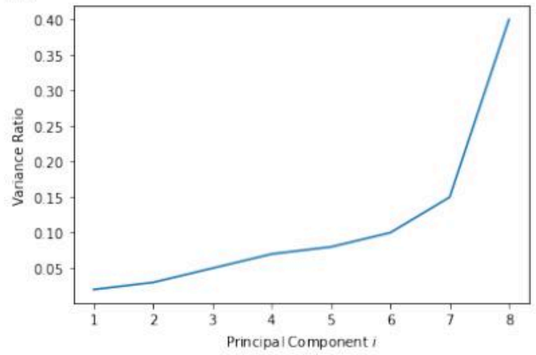
PCA Practice

Fa23 Final Q9

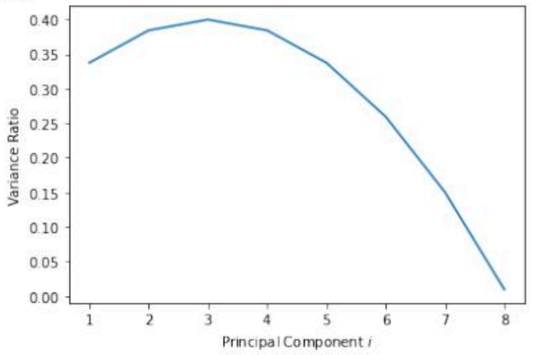
PieCe-A cake

(d) [2 Pts] Milad makes the following plots:

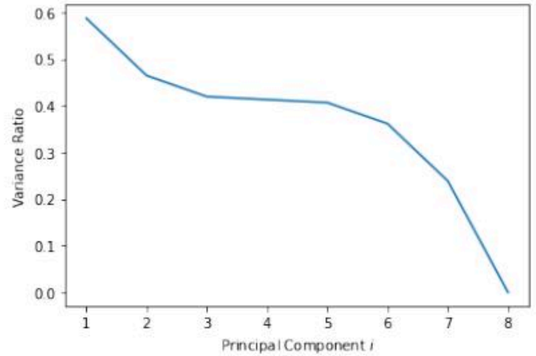
A.



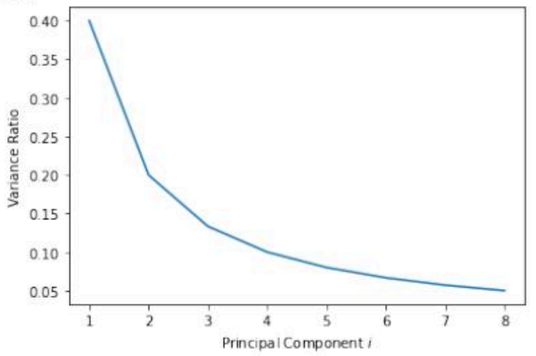
B.



C.



D.



Which of the above plots are **NOT** valid scree plots? **Select all that apply.**

- ☐ A
- ☐ B
- ☐ C
- ☐ D

Fa23 Final Q9

PieCe-A cake

(b) [2 Pts] Use the options below to fill in the blanks and complete the following statement:

Milad trains an OLS model. As he increases the number of principal components used, the training loss will _____(i)_____ at first, then _____(ii)_____ later. The test loss will _____(iii)_____ at first, then _____(iv)_____ later.

(i) Fill in the Blank:

- ◯ A. Increase
- ◯ B. Mostly stays the same
- ◯ C. Decrease

(ii) Fill in the Blank:

- ◯ A. Greatly increase
- ◯ B. Very slightly increase
- ◯ C. Very slightly decrease
- ◯ D. Greatly decrease

(iii) Fill in the Blank:

- ◯ A. Increase
- ◯ B. Stay the same
- ◯ C. Decrease

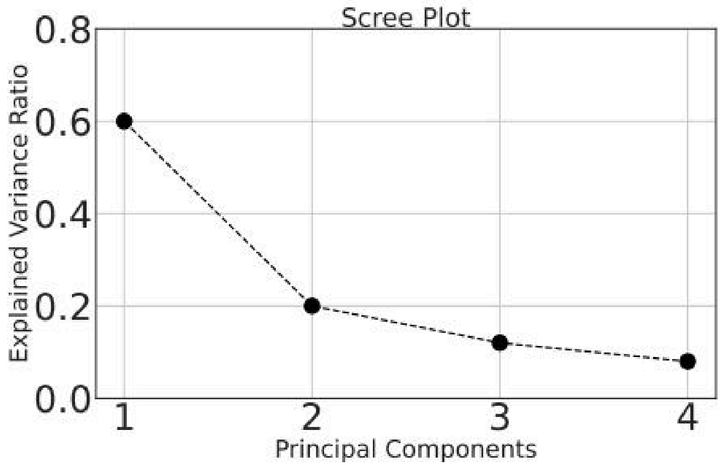
(iv) Fill in the Blank:

- ◯ A. Increase
- ◯ B. Decrease
- ◯ C. Oscillate between increasing and decreasing
- ◯ D. Stay constant

Fa24 Final Q7

PCA in Cooking

(a) [1.5 Pts] Evaluate the statements below based on the following scree plot that Minoli obtained from applying PCA on `cooks`:



◯ True ◯ False We should use 2 principal components for PCA on this dataset.

◯ True ◯ False The rank of this dataset is at least 4.

◯ True ◯ False There can be 5 principal components for `cooks`.

Sardaana Eginova, Sarika Pasumarthy, Kelly Hu, Collin Duong

Thank you for coming!

Note: CCAO and SQL are out of scope