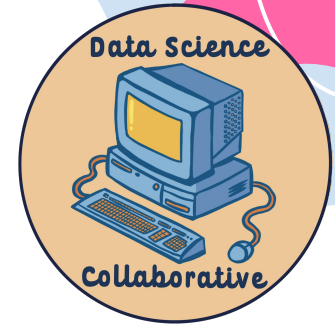
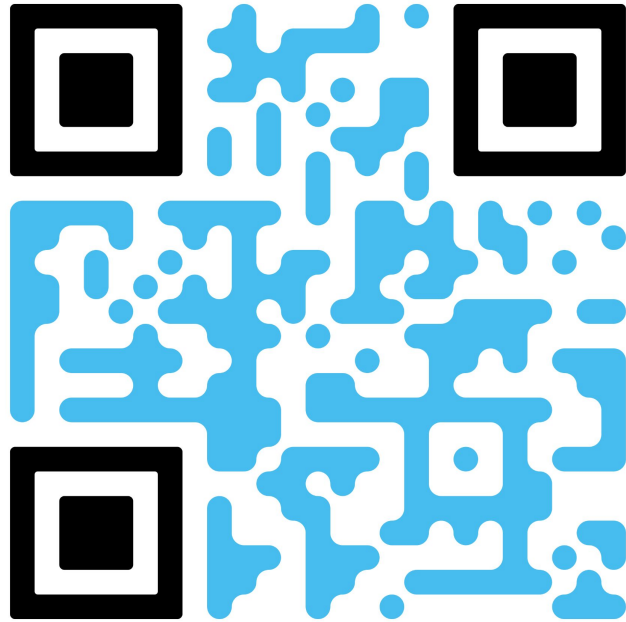


Data Cleaning and Preprocessing GM#4

2 Nov, 2023



Sign In



Why Clean Data?

- To use data for visualization and modeling, data needs to be cleaned.
- Cleaning data is the process of manipulating values into a usable format.
 - You can't plot the string "20 feet" on a graph, but you can plot the number 20.
 - You can't take the average of (20,30,NA) because NA isn't a number.
- Sometimes you want information in a different format than what is given.
 - The ratio of games won / played could be a better predictor than either individually.
- Data cleaning might take up a lot of time in your project.

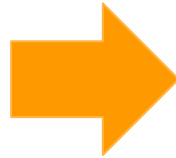


Data Matrices and CSVs

- Recall the “Data Matrix” from Ethan’s Lectures
- A .csv (comma-separated values) file stores data like the data matrix.
 - CSVs will probably be the format you’re provided from Kaggle or other data sources with small to medium size datasets.
- You will have to load this into a dataframe for access later
 - **R:** `data <- read.csv("myfile.csv")`
 - **Python:** `data = pd.read_csv("myfile.csv")`

```
index, c1, c2, c3
1,4,8,blue
2,,9,red
3,2,seven,,
4,,,
```

read CSV




index	c1	c2	c3
1	4	“8”	“blue”
2	None	“9”	“red”
3	2	“seven”	None
4	None	None	None

Manipulating Columns

- Creating new variables from a single column
 - Example: we have a column for distance in feet, we need to convert it to meters.
 - We can do this by taking the column, scaling each item, then putting this new column back into the data matrix.

```
df$meters <- df$Feet * 0.3048
```

Player	Feet	Meters
1	4	1.2192
2	5	1.524
3	7	2.1336
4	18	5.4864
...		
n	w	a



Manipulating Columns

- Creating new variables from multiple columns
 - Example: we want to know the ratio of successes to total attempts.
 - We take the columns, divide each number of successes by each number of attempts, and put the new column back into the data matrix.

```
df$ratio <- data$Wins / data$Attempts
```

Player	Wins	Attempts	Win Ratio
1	2	4	0.5
2	3	5	0.6
3	1	10	0.1
4	53	100	0.53
...
n	w	a	w / a

Manipulating Columns applied

- The `apply` function:
 - Python (Pandas): `DataFrame.apply(function)` returns a Series.
 - **Ex:** `df["ratio"] = df.apply((lambda x: x["wins"] / x["attempts"]), axis=1)`
 - This adds our ratio column in the dataframe, and initializes the values based on our formula.
 - Acts row-by-row (the input into our lambda function is a row of our DataFrame).
 - R: `apply(dataframe, axis, function)` returns a list.
 - **Ex:** `df$ratio <- apply(df, 1, (function (x) x["wins"] / x["attempts"]))`
 - This is equivalent to the Python implementation.



Making your Data Usable

- Data comes in many forms, but generally you want to end up with a **data matrix**.
 - Matrices should contain **categorical** or **numeric** data. It's important to store data according to its data type.
 - Ordinal data is categorical, but sometimes it is best to store it as numeric.
 - Example: Dates: It doesn't make sense to add Nov 2nd, 2023 to August 6, 2002. However, if we pick a "zero" for our date (say, Jan 1, 1970) we can convert days into a "days since" column. Then it DOES make sense to add and subtract them.
 - Counter-Example: Colors can be ordered ROYGBIV, but it's usually best to store color as categorical.



Example: Best Tennis Player

- Suppose you are trying to make a model that is predicting who will win a tennis game. You have p1_skill, player1, player2, and winner columns. How should we reformat the “winner” column so it can be used in a model?

p1_skill	player1	player2	winner
8	John	Dave	John
8	John	Will	Will
10	Will	Mark	Will
9	Mark	Dave	Dave

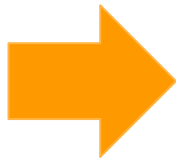


p1_skill	player1	player2	winner
8	John	Dave	player1
8	John	Will	player2
10	Will	Mark	player1
9	Mark	Dave	player2

Example: Tree Height

- Suppose you are making a model that predicts the height of a tree from the diameter of the trunk. The data has the units in terms of feet and inches. How should you manipulate the columns so you can use the data?

diameter	height
2 feet 1 inch	20 feet
1 foot	15 feet 2 inches
11 inches	14 feet



diameter_inch	height_inch
25	240
12	182
11	168

*** This might be a difficult function to code! Think of all the different cases with feet/foot, inches/inch.

Missing Values

- Think: Why might a value be missing from a dataset?
- Types of missing values: MAR, MCAR, MNAR
 - MAR: Missing at Random
 - Values are missing because of other values in the dataset.
 - Ex: Missing income data may depend on education, not the income itself.
 - MCAR: Missing Completely at Random
 - Values are missing for no reason:
 - Ex: Temperature data not present because the machine was broken that day.
 - MNAR: Missing Not at Random
 - Values are missing because of the values themselves.
 - Ex: Someone is embarrassed about their test grade, and decides not to report it.



Dealing with Missing Values

Look out for incorrect values that can skew your analysis!

- If you want to fit a model, no missing values can be present.
- Linear Interpolation
 - Uses linear regression to fill in missing values.
 - Good for **MAR** or **MCAR**.
- Using previous values to fill in
- Add a column or indicator value denoting missingness
 - Good for **MNAR**.
- Dropping columns/rows
 - Only use for **MCAR** or if strictly necessary.
 - **Avoid this if possible!!!** You don't know for sure if it's MCAR, and dropping columns costs valuable data and could result in biases.
- In all cases, explain why your solution was appropriate for your dataset!

Preprocessing: Categorical

- Machines only understand numbers, so we need to convert categories into numbers.
- Solution: One-Hot Encoding
 - Split N categories into N columns.
 - Denote an occurrence with a 1 and an absence with a 0.
- This is usually done as a step in your modeling pipeline (we will talk about this next week).

id	color
1	red
2	blue
3	green
4	blue



id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

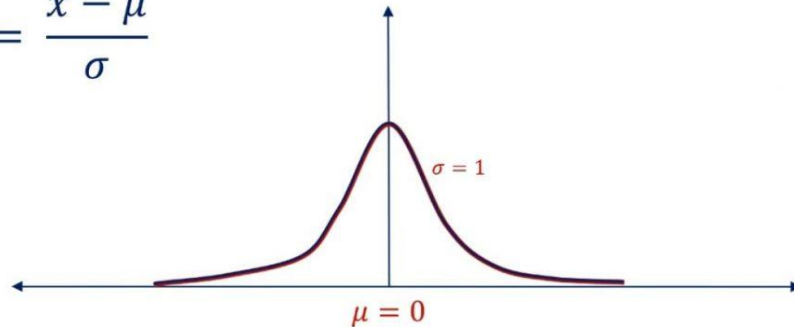


Preprocessing: Numerical

- Certain models will work better with certain distributions of data.
- Standardization is when you subtract the mean of a column, and divide by the standard deviation of that column.
- Usually done as a step in the modeling pipeline.

STANDARDIZATION

$$z = \frac{x - \mu}{\sigma}$$



The background of the slide is a dark blue/black space filled with numerous small, bright blue and white streaks that radiate outwards from the center, creating a sense of motion and depth, reminiscent of a hyperspace jump in Star Wars.

Star Wars RStudio Tutorial



Github Quickstart Tutorial

Project Examples

The slide features a white background with abstract, organic shapes in light blue, pink, and orange. A large, bold, dark blue text 'Project Examples' is centered on the page. There are two orange leaf-like shapes, one in the top left and one in the middle right. The bottom right corner has a large blue and pink shape.



Office Hours + Group Formation

Thank you for coming!



Join our Discord for updates
and connect with peers



Stay tuned to our Instagram
for updates and other
important information :)



Make sure to join our
Shoreline page for event
information!