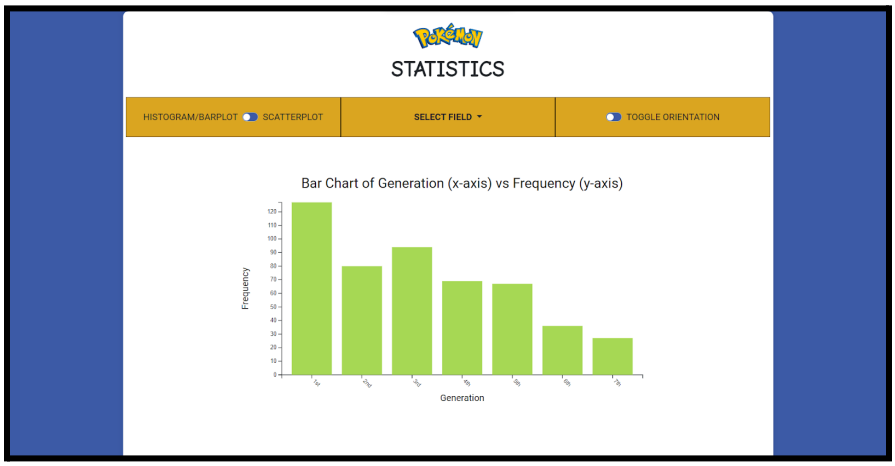
Pokémon Statistics Visualization

CSE 564 Visualization Lab 1

Submitted by:
Chahat Kalsi
SBU ID: 115825394

ABSTRACT

For this mini project, I developed a web application using D3.js to visualize Pokémon statistics, following the guidelines outlined in CSE 564 Lab 1.



The web application

DATASET

DATASETS USED

The datasets used in this project can be obtained from:

- **https://www.kaggle.com/datasets/rounakbanik/pokemon**
  This dataset contains information on all 802 Pokemon from all Seven Generations of Pokemon. The information contained in this dataset include Base Stats, Performance against Other Types, Height, Weight, Classification, Egg Steps, Experience Points, Abilities, etc.
- **https://docs.google.com/spreadsheets/d/1c16Wh4AawHGbTi3Eq1DGZQdM4FMUlJO1YwXJZ_ylRvg/edit#gid=557303698**
  A casual crowdsourced survey where people voted for their favorite Pokemon, details available at https://www.reddit.com/r/pokemon/comments/c0w4s0/favourite_pok%C3%A9mon_survey_results/

PREPROCESSING DATA

I utilized the Pandas library in Python to preprocess the data, following these steps:

- **Feature Selection**: Selected and retained only the crucial features from the entire set.
- **Renaming fields**: Altered the names of fields in the dataset for better intuitiveness.
- **Fusing datasets**: Combined the Pokémon statistics and Pokémon ranks datasets.
- **Cleaning the dataset**: Addressed null values and assigned more intuitive names to categories.
- **Sampling the dataset**: Extracted the top 500 most popular Pokémon.

```python
# -*- coding: utf-8 -*-
"""VISLAB1.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1BbMtO0xr-PbSyg6dbRGgOvEvtz6djDhx
"""

from google.colab import drive
drive.mount('/content/drive')

import pandas as pd

stats = pd.read_csv("/content/drive/MyDrive/vis/hw1/pokemon/pokemon.csv") # pokemon statistics dataset
ranks = pd.read_csv("/content/drive/MyDrive/vis/hw1/pokemon/pokemon_ranks.csv") # pokemon ranks dataset

# pikcing important fields from all fields
stats = stats[['name', 'generation', 'is_legendary',
               'attack','defense', 'speed', 'sp_attack', 'sp_defense',
               'hp', 'percentage_male', 'capture_rate',
               'base_happiness', 'base_total', 'base_egg_steps',
               'type1', 'type2',
               'weight_kg','height_m'
      ]]

# renaming the fields to more readable names
stats.rename(columns={'name':"Pokemon"}, inplace=True)
stats.rename(columns={'generation':"Generation"}, inplace=True)
stats.rename(columns={'is_legendary':"Is Legendary?"}, inplace=True)
stats.rename(columns={'attack':"Attack"}, inplace=True)
stats.rename(columns={'defense':"Defense"}, inplace=True)
stats.rename(columns={'speed':"Speed"}, inplace=True)
stats.rename(columns={'sp_attack':"Special Attack"}, inplace=True)
stats.rename(columns={'sp_defense':"Special Defense"}, inplace=True)
stats.rename(columns={'hp':"HP"}, inplace=True)
stats.rename(columns={'percentage_male':"Percentage Male"}, inplace=True)
stats.rename(columns={'capture_rate':"Capture Rate"}, inplace=True)
stats.rename(columns={'base_happiness':"Base Happiness"}, inplace=True)
stats.rename(columns={'base_total':"Base Total"}, inplace=True)
stats.rename(columns={'base_egg_steps':"Base Egg Steps"}, inplace=True)
stats.rename(columns={'type1':"Primary Type"}, inplace=True)
stats.rename(columns={'type2':"Secondary Type"}, inplace=True)
stats.rename(columns={'weight_kg':"Weight in Kilograms"}, inplace=True)
stats.rename(columns={'height_m':"Height in Meters"}, inplace=True)

# merge the stats and ranks datasets based on pokemon name
merged = pd.merge(stats, ranks, on="Pokemon")

# sorting the rows by their rank
merged.sort_values(by="Rank", inplace=True)

# Changing generation field to strings like 1st, 2nd, 3rd...
merged['Generation'] = merged['Generation'].apply(lambda x: str(x) + ('th' if x > 3 else ['st', 'nd', 'rd'][x-1]) if x < 8 else str(x) + 'th')

# Changing is_legendary field to "True" and "False"
merged['Is Legendary?'] = merged['Is Legendary?'].map({0: "False", 1: "True"})

# Changing NaN values in type2 field to "None"
merged['Secondary Type'].fillna("None", inplace=True)

merged.to_csv("/content/drive/MyDrive/vis/hw1/pokemon/merged.csv",index=False) # saving the whole dataset
merged.head(500).to_csv("/content/drive/MyDrive/vis/hw1/pokemon/merged_sampled.csv",index=False) # saving a subset of the dataset containing top 500 pokemon by Rank
```

The python preprocessing script: vislab1.py

FINAL DATASET

The final preprocessed dataset consists of 500 rows, representing the 500 most popular Pokémon based on the linked survey. It comprises 20 fields, with 5 being categorical and the remaining 15 numerical. All categorical fields (except “Is Legendary?”, which can only be either True or False) have more than 6 categories.

Here are the descriptions of these attributes (note that categorical fields are italicized):

1. ***Pokemon***: The English name of the Pokemon.
2. ***Generation***: The numbered generation which the Pokemon was first introduced.
3. ***Is Legendary?***: Denotes if the Pokemon is legendary (true/false)
4. **Attack**: The Base Attack of the Pokemon.
5. **Defense**: The Base Defense of the Pokemon.
6. **Speed**: The Base Speed of the Pokemon.
7. **Special Attack**: The Base Special Attack of the Pokemon.
8. **Special Defense**: The Base Special Defense of the Pokemon.
9. **HP**: The Base HP of the Pokemon.
10. **Percentage male**: The percentage of the species that are male.
11. **Capture Rate**: Capture Rate of the Pokemon
12. **Base Happiness**: Base Happiness of the Pokemon
13. **Base Total**: Base total statistics of the Pokemon.
14. **Base Egg Steps**: The number of steps required to hatch an egg of the Pokemon.
15. ***Primary Type***: The Primary Type of the Pokemon
16. ***Secondary Type:*** The Secondary Type of the Pokemon
17. **Weight in Kilograms**: The Weight of the Pokemon in kilograms
18. **Height in Meters**: Height of the Pokemon in meters
19. **Number of Votes**: The number of votes the Pokemon received in the popularity survey.
20. **Rank**: The rank of the Pokemon based on the number of votes it received in the survey.

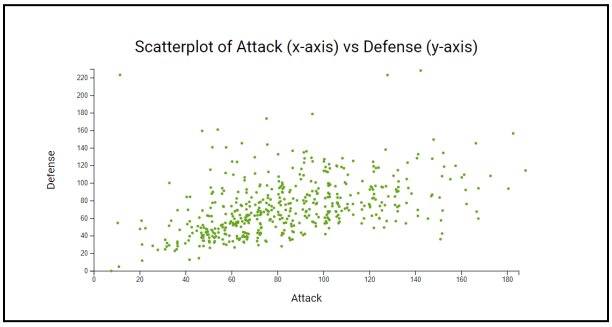
Pokemon	Generation	Is Legendary?	Attack	Defense	Speed	Special Attack	Special Defense	HP	Percentage Male	Capture Rate	Base Happiness	Base Total	Base Egg Steps	Primary Type	Secondary Type	Weight in Kilograms	Height in Meters	Number of votes	Rank
Charizard	1st	FALSE	104	78	100	159	115	78	88.1	45	70	634	5120	fire	flying	90.5	1.7	1107	1
Gengar	1st	FALSE	65	60	130	170	95	60	50	45	70	600	5120	ghost	poison	40.5	1.5	1056	2
Arcanine	1st	FALSE	110	80	95	100	80	90	75.4	75	70	555	5120	fire	None	155	1.9	923	3
Bulbasaur	1st	FALSE	49	49	45	65	65	45	88.1	45	70	318	5120	grass	poison	6.9	0.7	710	4
Blaziken	3rd	FALSE	160	80	100	130	80	80	88.1	45	70	630	5120	fire	fighting	52	1.9	613	5
Umbreon	2nd	FALSE	65	110	65	60	130	95	88.1	45	35	525	8960	dark	None	27	1	607	6
Lucario	4th	FALSE	145	88	112	140	70	70	88.1	45	70	625	6400	fighting	steel	54	1.2	604	7
Gardevoir	3rd	FALSE	85	65	100	165	135	68	50	45	35	618	5120	psychic	fairy	48.4	1.6	585	8
Eevee	1st	FALSE	55	50	55	45	65	55	88.1	45	70	325	8960	normal	None	6.5	0.3	581	9
Dragonite	1st	FALSE	134	95	80	100	100	91	50	45	35	600	10240	dragon	flying	210	2.2	551	10
Absol	3rd	FALSE	150	60	115	115	60	65	50	30	35	565	6400	dark	None	47	1.2	542	11
Typhlosion	2nd	FALSE	84	78	100	109	85	78	88.1	45	70	534	5120	fire	None	79.5	1.7	534	12
Ampharos	2nd	FALSE	95	105	45	165	110	90	50	45	70	610	5120	electric	None	61.5	1.4	529	13
Squirtle	1st	FALSE	48	65	43	50	64	44	88.1	45	70	314	5120	water	None	9	0.5	523	14
Flygon	3rd	FALSE	100	80	100	80	80	80	50	45	70	520	5120	ground	dragon	82	2	510	15
Ninetales	1st	FALSE	67	75	109	81	100	73	24.6	75	70	505	5120	fire	ice			471	16
Tyranitar	2nd	FALSE	164	150	71	95	120	100	50	45	35	700	10240	rock	dark	202	2	451	17
Infernape	4th	FALSE	104	71	108	104	71	76	88.1	45	70	534	5120	fire	fighting	55	1.2	443	18
Snorlax	1st	FALSE	110	65	30	65	110	160	88.1	25	70	540	10240	normal	None	460	2.1	433	19
Torterra	4th	FALSE	109	105	56	75	85	95	88.1	45	70	525	5120	grass	ground	310	2.2	430	20
Luxray	4th	FALSE	120	79	70	95	79	80	50	45	70	523	5120	electric	None	42	1.4	429	21
Scizor	2nd	FALSE	150	140	75	65	100	70	50	25	70	600	6400	bug	steel	118	1.8	424	22
Blastoise	1st	FALSE	103	120	78	135	115	79	88.1	45	70	630	5120	water	None	85.5	1.6	410	23
Garchomp	4th	FALSE	170	115	92	120	95	108	50	45	70	700	10240	dragon	ground	95	1.9	404	24
Mudkip	3rd	FALSE	70	50	40	50	50	50	88.1	45	70	310	5120	water	None	7.6	0.4	404	24
Metagross	3rd	FALSE	145	150	110	105	110	80		3	35	700	10240	steel	psychic	550	1.6	395	26
Scyther	1st	FALSE	110	80	105	55	80	70	50	45	70	500	6400	bug	flying	56	1.5	394	27
Espeon	2nd	FALSE	65	60	110	130	95	65	88.1	45	70	525	8960	psychic	None	26.5	0.9	384	28
Swampert	3rd	FALSE	150	110	70	95	110	100	88.1	45	70	635	5120	water	ground	81.9	1.5	381	29
Cyndaquil	2nd	FALSE	52	43	65	60	50	39	88.1	45	70	309	5120	fire	None	7.9	0.5	377	30
Alakazam	1st	FALSE	50	65	150	175	105	55	75.4	50	70	600	5120	psychic	None	48	1.5	376	31
Charmander	1st	FALSE	52	43	65	60	50	39	88.1	45	70	309	5120	fire	None	8.5	0.6	374	32
Empoleon	4th	FALSE	86	88	60	111	101	84	88.1	45	70	530	5120	water	steel	84.5	1.7	346	33
Sceptile	3rd	FALSE	110	75	145	145	85	70	88.1	45	70	630	5120	grass	None	52.2	1.7	337	34
Jolteon	1st	FALSE	65	60	130	110	95	65	88.1	45	70	525	8960	electric	None	24.5	0.8	335	35
Cubone	1st	FALSE	50	95	35	40	50	50	50	190	70	320	5120	ground	None	6.5	0.4	321	36
Rayquaza	3rd	TRUE	180	100	115	180	100	105		45	0	780	30720	dragon	flying	206.5	7	316	37
Aggron	3rd	FALSE	140	230	50	60	80	70	50	45	35	630	8960	steel	rock	360	2.1	313	38
Vulpix	1st	FALSE	41	40	65	50	65	38	24.6	190	70	299	5120	fire	ice			302	39
Raichu	1st	FALSE	85	50	110	95	85	60	50	75	70	485	2560	electric	electric			301	40
Haunter	1st	FALSE	50	45	95	115	55	45	50	90	70	405	5120	ghost	poison	0.1	1.6	301	40

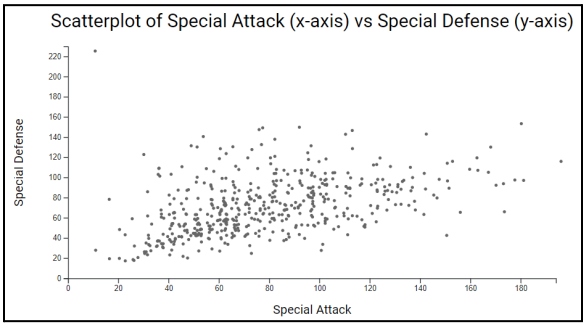
The final dataset used: pokemon_data.csv

INTEREST IN THE DATASET

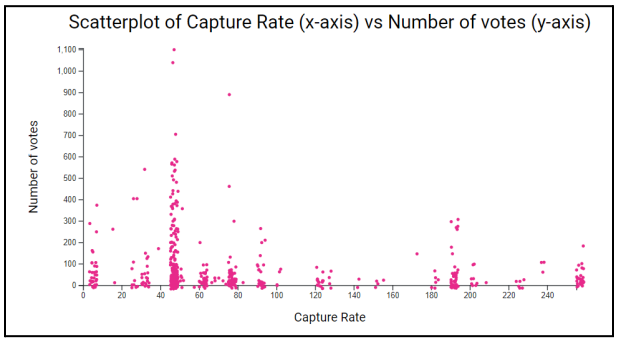
As an enthusiastic fan of the Pokémon games, I stumbled upon a Pokémon statistics dataset and decided to use it for my assignment. It encompassed all the attributes that would captivate any Pokémon aficionado. My particular interest lay in observing how these attributes varied across generations and exploring the correlations between numerical attributes, such as attack and defense.

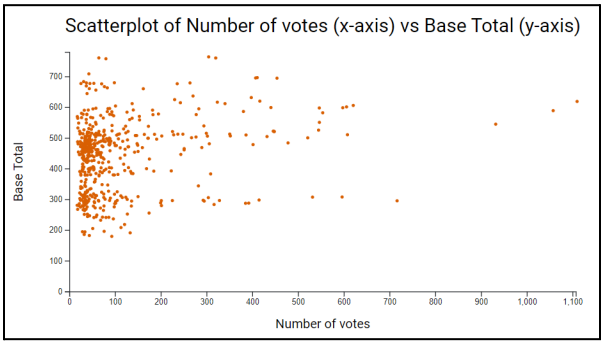
Through the visualizations I created, I discovered numerous relationships between the numerical attributes of Pokémon, such as, that those with higher attack also tended to have higher defense.



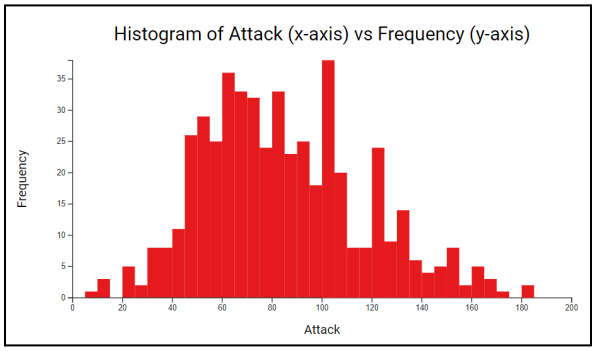


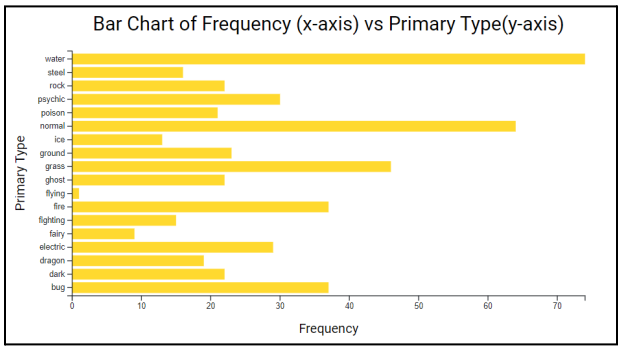
By merging the Pokémon statistics dataset with the Pokémon ranks dataset, I gained insight into what people typically favor in their Pokémon. I observed intriguing patterns, such as the correlation between ease of capture and popularity, as well as the tendency for the most popular Pokémon to possess above-average base statistics.





Finally, insights on how numerical attributes are distributed across the wide pokemon variety and how the categorical attributes vary across their categories were also all very captivating to me.





IMPLEMENTATION

SPECIFICATION

For this assignment, I employed the Node Package Manager (npm) to install D3.js and other necessary dependencies. Additionally, I utilized HTML and CSS, complemented by Bootstrap.js, to design the frontend

RUNNING THE APPLICATION

The visualization is a Node.js application, to run it, navigate into the folder containing the HTML, CSS and JS, and run the following commands:

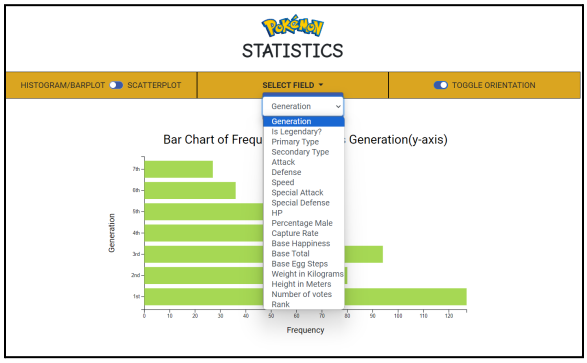
```
npm install
npm start
```

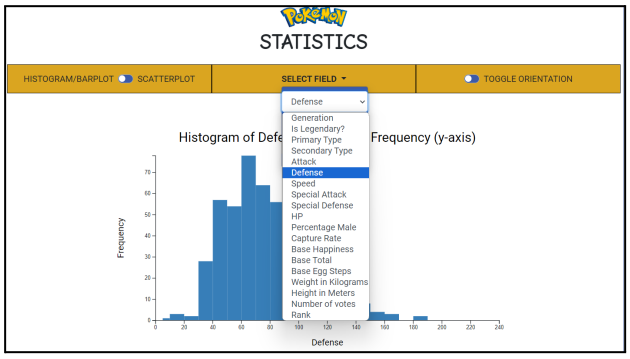
This will create a new http server on your localhost at http://127.0.0.1:8080/
Open this link in the browser to see the application.

DESIGN

The project addresses each of the specific design requirements asked for in the following manner:

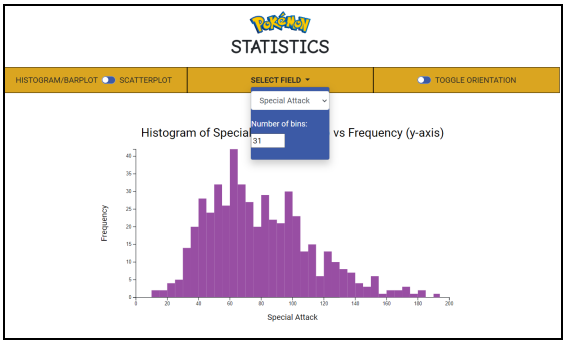
1. A unified dropdown menu to allow the user to select a variable and update the chart based on their selection.

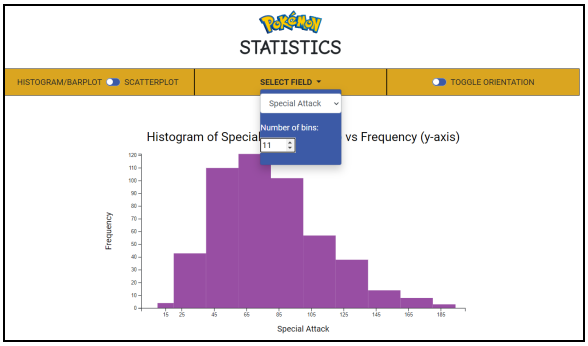




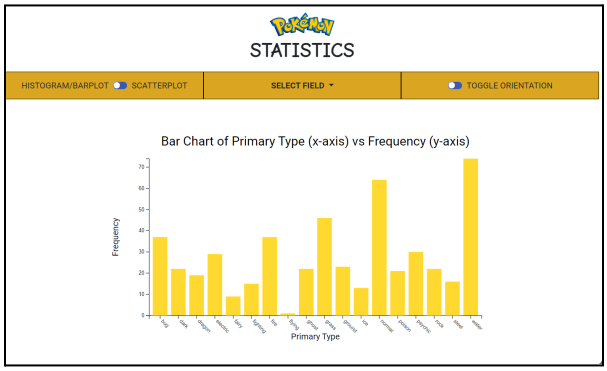
2. Giving the user an option to toggle between drawing a Histogram/Bar plot or a scatterplot and automatically choosing to draw a histogram or a bar plot based on if the user has selected a numeric or a categorical variable respectively.

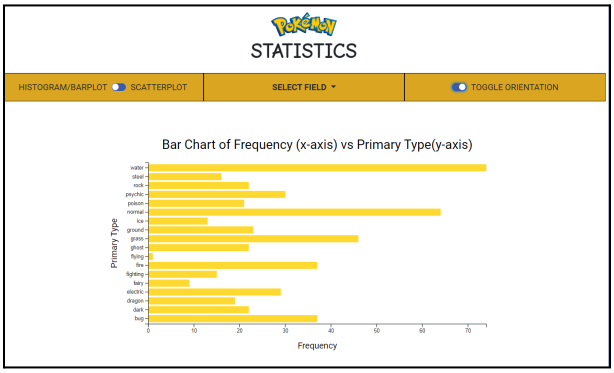
3. Giving the user the option to change the number of bins.



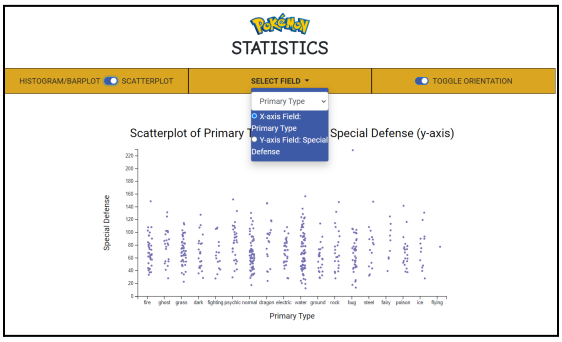


4. A toggle to button to toggle orientation of the chart. It toggles between drawing upright/sideways if the bar plot/histogram is being drawn, otherwise toggling flips the x and the y axes if a scatter plot is being drawn.





5. If the user wishes to draw a scatter plot, giving them the option to pick specific fields for both the axes from the menu based on radio buttons for each axis.



6. Every plot has a clear title specifying the type of graph being rendered and what the axes variables represent. In addition, all graphs have clear axis labels, and histograms have ticks for bin range mids.

7. Jitter has been added to the dots in the scatter plots, as was specified in the requirements.