Optimizing Restaurant Menus & Operations with Data Science

Rashad Sarwani (res265) – Independent Project - CS 439 Final Project
Github: https://github.com/rsarwani/finalproject
Dataset: (too large for GitHub - you can also generate a similar dataset using the script on GitHub)
https://drive.google.com/file/d/1NNEeEldWJ6ruZRRx0wdl2aShQBm7WF12/view?usp=sharing
Video: also uploaded in canvas submission
https://drive.google.com/file/d/14KPJj1TxVKqyBOZQkyaxQPrf54mcnw1_/view?usp=sharing

**Note: Please see the attached notebook (in the submission & on GitHub) for all visualizations - only a handful were shown here due to space constraints. It is also shown in the video

1. Introduction

This project is about analyzing sales data from several restaurants and using that to determine trends in customer behaviour and sales, and to predict what sales can be in the future to help with inventory and staff planning.

2. Motivation & Prior Work

As I will explain below, there is nothing like this analysis available in other works or publications. There is not even a dataset similar to the one I have created. Even if a restaurant were to download its POS data, it would not get as detailed a dataset as I have just created.
As for motivation, I was extremely motivated to work on this project! I have worked in restaurants for many years and have family in the industry. I have held many positions such as a line cook, kitchen manager, server, barista, supervisor, and manager. This gave me a holistic overview of the industry, and being a computer science major, I always appreciated the vast amounts of data generated daily. Thousands of data entries for things like inventory, orders, receipts, phone calls, reviews, transactions, and payroll are all being stored and are likely never to be accessed ever again. There is so much potential for data analysis and predictive modeling development to combat the industry's most significant loss - food wastage. You can't know exactly how many raw ingredients to purchase; you may not have enough and have to close your restaurant until you resupply, or you may have too much and throw away spoiled goods. That is, unless you are analyzing the plethora of data that restaurants already have, and using that to your advantage. This project just scratches the surface of what you can do, and the more data you have, the more accurate your predictions can be. (Note: This is an unsupervised project where I mainly analyze the data and draw conclusions on trends and behaviors. However, with the extra day, I decided to use predictive modelling as I was very interested!)

3. Method

3.1 Dataset Generation

This took the most time. I knew what I wanted to do, but finding the data to analyze was impossible. All available datasets that I could find online were incomplete. For example, one dataset would have weather data, another would have information about transactions, and another would have information about menu items/quantity. No single data set had all the information I needed to perform a valid analysis. The lack of available datasets for what I originally wanted to do led me to change my approach. This is where my experience came in. Having worked in restaurants for years and knowing the industry well, I had an excellent idea of what attributes I needed and what the ranges should be. To make it more accurate, I called restaurants to get some information from their sales reports and looked at past sales reports I had from my old job as a restaurant manager. I used this data and my experience to write a Python script to generate an applicable dataset. The data was randomly generated, using specific probabilities for exact values and ranges, to ensure that the data makes sense. Getting this script right took a couple of weeks, as I would find errors after testing and working on other parts of the project. For example, when analyzing tip amounts, I realized that there were unreasonable order quantities; a party of 2 ordered 6 entrees, 11 drinks, etc. That is not valid data to use for an analysis, and not at all realistic. I continued to run into these issues, and eventually got a dataset that accurately represented 5 unique restaurants. Examples of other issues I ran into were unrealistic tip amounts and inaccurate weather data (e.g., below freezing temperatures in LA summers). The script I created realistically assigns values in a sequential order. For example, the weather is determined based on the date and location, and the tip amount is based on the order total and the number of people in the party. This further strengthens the validity of the dataset. Now that I had a dataset, I could now start to analyze the data.

3.2 Analytical Pipeline

To start, I generated my dataset using the script I created. The script generates 5000 unique transactions. Each row in the dataset has the following attributes: restaurant_id, cuisine,transaction_id,transaction_time,
date,day_of_week,time_of_day,party_size,item_name,

quantity,item_price,transaction_amount,payment_method,order_type,is_weekend, weather,temperature_f,tip_amount. Each transaction is made up of 1 or more rows, and each row represents one part of the transaction; for example, row one stores information about the sale of 2 drinks, row 11 stores information about the sale of 1 entree, row 3 stores information about the sale of another entree…etc. Each row has a transaction ID, and all rows with the same transaction ID refer to the same transaction/order. As you would expect, attributes that are "common" to the transaction, like party size, tip amount, weather, date, etc. are always the same for all rows with the same transaction. The script generates 5000 transactions, and for the generated dataset I am using for the demonstration, there are 747721 rows representing those 5000 transactions. The dataset includes information about five restaurants, in 2 cities (New York City and Los Angeles, randomly assigned based on whether the restaurant ID is odd or even), and each restaurant is a different cuisine: Thai, Indian, Italian, Mediterranean, and Mexican. I conducted extensive market research to come up with sample menus and their prices, and even reached out to restaurants similar ot the ones that I wanted to replicate to come up with this data. I import this data as a dataframe (originally it is a CSV) and then start to analyze it. Before jumping into the analysis, I thought it was important to actually see that data.

R0 | R1 | R2 | R3 | R4

Restaurant 0 — NYC, Italian

Breakfast

	Menu Item	Price
0	Breakfast Bruschetta	9.0
1	Cappuccino	4.5
2	Espresso	3.0
3	Fresh Fruit Plate	8.0
4	Frittata	10.0
5	Italian Breakfast Sandwich	9.5
6	Latte	4.0
7	Prosciutto & Melon	12.0
8	Ricotta Pancakes	11.0
9	Yogurt Parfait	7.5

Lunch

	Menu Item	Price
0	Bruschetta	8.5
1	Caprese Salad	11.0
2	Chianti (glass)	9.0
3	Espresso	3.0
4	Fettuccine Alfredo	14.0
5	Gelato	6.0
6	Lasagna	15.0
7	Margherita Pizza	12.0
8	Minestrone Soup	7.0
9	Panini	10.5

Dinner

	Menu Item	Price
0	Affogato	7.5
1	Antipasto Platter	14.0
2	Arancini	9.0
3	Bistecca Fiorentina	35.0
4	Chianti (bottle)	45.0
5	Gnocchi Sorrentina	18.0
6	Limoncello	7.0
7	Osso Buco	28.0
8	Seafood Risotto	24.0
9	Tiramisu	8.0

Restaurant 0 | Restaurant 1 | Restaurant 2 | Restaurant 3 | Restaurant 4

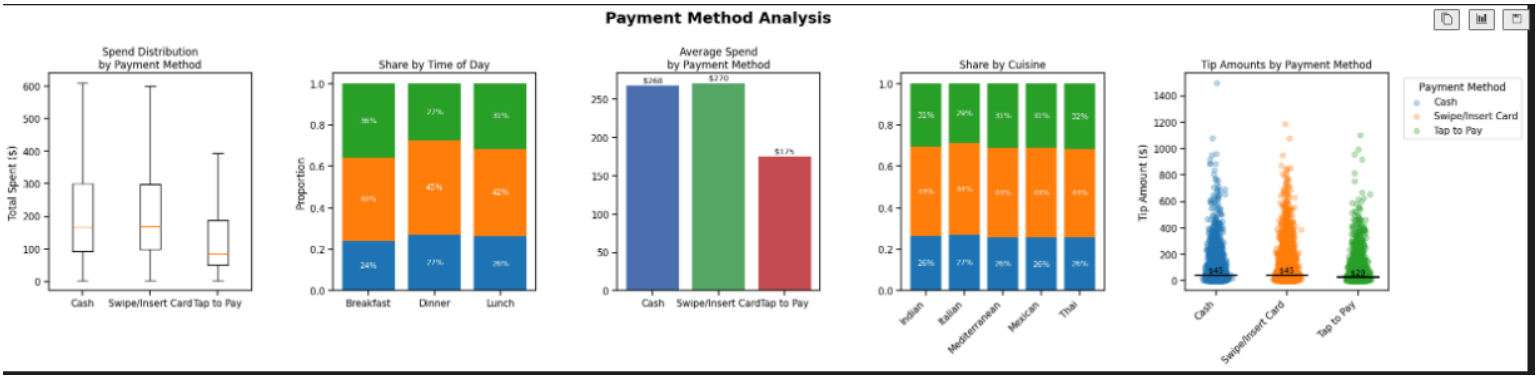
transaction_id	party_size	total_spent	tip_amount	dishes	cuisine	location	is_weekend	weather
006RU4J9E	2	265.0	20.06	Affogato(2), Arancini(2), Bistecca Fiorentina(2), Chianti (bottle)(2), Seafood Risotto(3)	Italian	NYC	False	Snow
00MDD6YHK	6	336.5	43.52	Breakfast Bruschetta(12), Cappuccino(1), Frittata(4), Italian Breakfast Sandwich(5), Prosciutto & Melon(4), Ricotta Pancakes(6), Yogurt Parfait(3)	Italian	NYC	False	Snow
00Q0ZMVA3	2	118.0	9.13	Bruschetta(1), Espresso(2), Fettuccine Alfredo(4), Lasagna(2), Minestrone Soup(1), Panini(1)	Italian	NYC	False	Cloudy
00Q58CHHN	5	196.0	43.83	Breakfast Bruschetta(7), Frittata(5), Italian Breakfast Sandwich(2), Latte(4), Prosciutto & Melon(4)	Italian	NYC	False	Sunny
00T2BDXQG	8	697.0	41.65	Affogato(4), Antipasto Platter(1), Arancini(6), Bistecca Fiorentina(2), Chianti (bottle)(5), Osso Buco(4), Seafood	Italian	NYC	True	Sunny

First, all the required libraries and downloads were imported into the first cell. I then used the *head* function to view the first 5 rows to ensure the dataframe got set up correctly after importing the CSV as DF. Then, I wanted to view the menus for the different restaurants. To do this, I used ipywidgets - a library I found online - that allows you to present your data in a very nice and interactive way. I knew I wanted to separate everything based on each restaurant, and this allowed me to create a tab for each restaurant and then a menu for each meal. I also wanted to be able to view each transaction as a whole, so I created a similar dashboard using the same widget library to show each *whole* transaction in one row. This made it a lot easier to understand the data. Now that I can easily access the data, it is time to start analyzing!

4. Results

4.1 Payment Method Analysis:

This cell analyzes the payment method attribute. Specifically, I wanted to see how payment method is correlated with things like tipping, transaction value, time of day, and cuisine. To do so, I produced several visualizations to show the relationship between the attributes. As you can see in the first (From left to right), most of the tap-to-pay (also includes



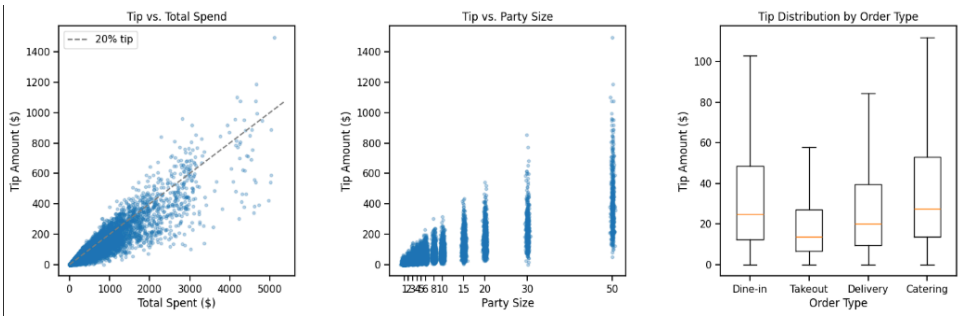
Apple/Android Pay) transactions were lower in value. This shows how when ordering online and or just tapping their phones, customers spend their money more freely. The same figure also shows that when the transactions were large, some people preferred to use a physical card to pay or cash; this suggests that once guests hit “serious” spend levels, they default to traditional payment methods. One unique thing I observed was that when the bill was very high, people would sometimes pay cash. I believe that this is due to the fact that restaurants will often waive transaction fees incurred by the payment processor for cash payments - by paying cash on large orders, customers automatically get a nice discount.

In the next figure, you can see how Tap-to-Pay share spikes at Breakfast (36%) but falls off at Dinner (27%), implying commuters love one-tap coffee orders, whereas sit-down dinners lean on cards/cash - this is also more ideal for tipping as it avoids the award interaction between the customer and server when you have to put in the tip right infront of them. Speaking of tipping, the Average tip on Tap-to-Pay orders is 20% lower than on card transactions—guests may feel less “connected” when tapping versus handing over a card or cash. It’s interesting to see how all cuisines hover around a 25–30% tap, 42–44% card, 30–32% cash split—payment behavior is more spend-driven than cuisine-driven. Going into this, I thought some cuisines would have higher cash or card or tap payments due to certain demographics preferring to use a specific payment method, but I was proven wrong! It was also interesting to see that NYC and LA show virtually identical payment mixes, indicating that these behavioral trends hold across different urban markets (Figure 4). Overall, these patterns suggest opportunities to tailor promotions (for example, a “Tap-to-Pay Happy Hour” discount on small-ticket items) and to possibly restructure fee structures for large cash orders to maximize margin retention and more money going to the business rather than processing fees.

4.2 Tipping Behavior

Something that I was very interested to see was how tipping behavior changed with different party sizes, weather, time of day, and payment methods. As a former server, I had my own hypothesis, but was eager to see a data-driven approach. Similarly, I used my dataframe to create some visualizations. Tips generally rise in proportion with the bill—most points cluster around the 20 % line, showing a strong positive correlation between the total spend and tip amount - this much was expected, as was a handful of outliers tip well above 20 % on large checks, suggesting either particularly generous groups or reaction to exceptional service. Something interesting to me was seeing that larger parties tended to give larger tips - 50-person tables routinely tip $800–$1,200. This is likely due to corporate parties, catering, and events that have larger budgets and can afford to tip generally. Also, it is likely that management stresses service to be above and beyond with these big spenders, so they likely received premium service that deserved the higher tip. I did not include auto grat in my dataset, so these are all voluntary tips.

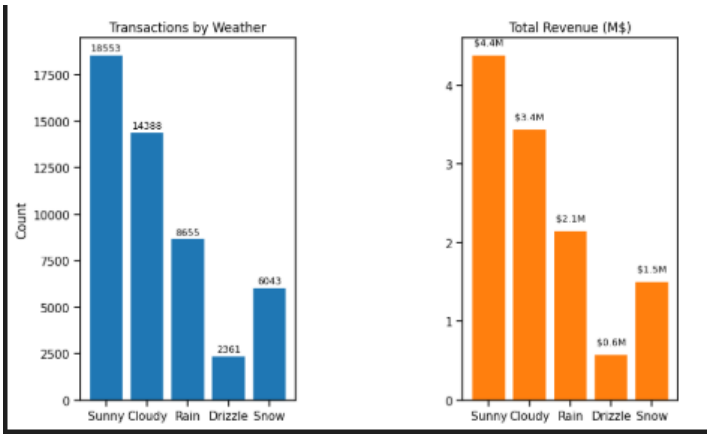
Something that really stood out to me was how the order type, Dine-in/takeout/delivery, affected tipping behavior. **Dine-in** orders seem to show the highest median tip (~20 % of average spend) and the widest spread; guests often reward the full-service experience most. **Delivery** tips are slightly lower on average but still substantial, with a tighter IQR, reflecting consistent appreciation for convenience, which was a pleasant surprise - people are appreciating their delivery drivers! This also stood out because delivery orders are more expensive when using 3rd party services like DoorDash due to a lot of service charges, so a tip on top of a marked-up order is surprising. **Takeout** has the lowest median and narrowest distribution, without table service, tipping norms weaken - this is expected; there is not much service being provided with takeout, so tipping is not



expected but still appreciated. Interestingly, catering displays the highest absolute tips (median around $25) and extreme upper whiskers, driven by large-party auto-gratuities and large corporate party budgets.

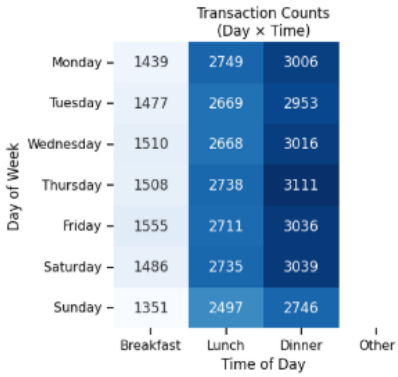
These patterns suggest that tailoring service and implementing upsell strategies have a lot of benefits. It encourages restaurants to reinforce premium diner experiences with subtle tip prompts, optimize delivery incentives to maintain an 18 % baseline, and consider modest “suggested tip” reminders for takeout to boost low-end tips on those orders.

4.3 Weather Impact on Sales & Order Mix



As a former restaurant employee who had worked through many sunny, rainy, and snowy days, I was interested to see, numerically, how the weather seemed to impact sales and operations. I always thought that bad weather meant no one would come in, but we would occasionally get packed. This was somewhat reflected in the data. Looking at the two figures to the right, the first one shows the number of transactions during each type of weather. Unsurprisingly, sunny weather is the highest, with precipitation being the lowest. People don’t always want to go out when the weather is bad, and they want to stay home. However, when the weather is bad, people tend not to want to cook for themselves. In the second graph, you can see how there was almost a 1 million dollar difference in revenue on snowy days vs rainy days! This shows how people would rather want to eat outside food on snowy days compared to slightly “wet” days. Something that stood out to me was how there was not a significant increase in the percentage of delivery orders during inclement weather - I would have expected that to increase significantly, as people do not want to go out.

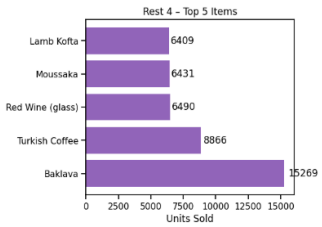
4.4 Time-of-Day & Day-of-Week Dynamics

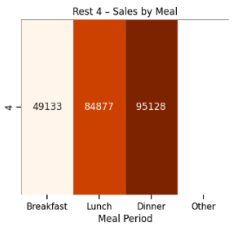


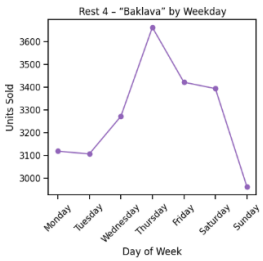
The next part of the dataset I wanted to explore was the time of day/mealtimes. I wanted to see what cuisines generate the most revenue and at what times. This could be very beneficial to the restaurant, as they can see what hours to prioritize in terms of staff and other resources, and what hours of operation need to be further developed. As I had seen firsthand in industry, Dinner was the most profitable meal by far. The average spend per party and the number of transactions are always highest during dinner service. During breakfast and lunch, many people are at work and are not able to go out. It is also expensive to go out for fancy meals, and not all can afford to do so for 3 meals a day. Dinners are often ideal for celebrations and gatherings, which is represented by the data. There are a lot of large parties during dinner hours. One thing that surprised me with this cell was that Thursday was the highest revenue day overall across the dataset. I believe this may be inflated due to some holidays, which were not accounted for in the dataset. The restaurants can use this information to accurately schedule staff, ensure that enough ingredients are prepped, and also predict what their sales would be on each day of the week for each meal service.

4.5 Best-Selling Items & Timing

The next cell analyzes each restaurant's best-selling items. I used a similar widget implementation with tabs to separate each restaurant. There are three graphs here, showing the popularity of each of the top dishes, the sales for each meal, and the purchase trends of the top dish. Here you can see how for Restaurant 4, Baklava is the most popular dish. The third figure also shows when people are buying it the most. This is very beneficial information to have, as it can drastically help you reduce food wastage and prioritize which products to procure. This restaurant knows that it will always be selling lots of baklava, and knows exactly which days will be selling more/less than usual. This also allows the business to see what menu items "Work" and what they should try to replicate other dishes, to get other dishes performing at the same level. It also shows what the competitors like in each result and what their specialty is.







4.6 Inventory Forecasting by Temperature

Additionally, I decided to add a section that showed what the forecasted sales could be, using the data that we have. This is SUCH a powerful tool to have, as it makes ordering inventory, scheduling staff, and managing all parts of the business seamless. It takes the "What if" out of the equation and allows restaurant owners to make data-driven predictions about their sales in the future. I decided to take it a step further by incorporating temperature. Since we have the data, we might as well use it to make more accurate predictions. Now, if there is a cold or warm week forecasted, the restaurant can predict what its sales will be. This is helpful because the more data we consider, we can maek the more accurate our prediction. Using weather data is helpful as many people will get cold things like an iced drink or cold dessert in the warm weather, and the opposite is true in the cold weather. This cell segments transactions into weekly buckets, labels each week as "Hot" or "Cold" based on whether its average temperature sits above or below the historical median, and then computes the top 10 selling menu items for each category. It then forecasts next week's demand for those items by averaging their weekly sales over past hot weeks and over past cold weeks, yielding side-by-side "forecast_hot_week" and "forecast_cold_week" tables. The next cell presents it in a very simple and easy-to-use dashboard, with a tab for each restaurant and each of its dishes, as well as how many units are predicted to be sold that week. There is also a toggle at the top that allows you to choose between a hot or cold week based on your local weather forecast.

Weekly Sales Forecast by Restaurant & Week-Type

Restaurant 0 | Restaurant 1 | Restaurant 2

Week Type: Hot | Cold

Menu Item	Units Forecast
Affogato	709
Antipasto Platter	729
Arancini	731
Bistecca Fiorentina	697
Breakfast Bruschetta	326
Bruschetta	652
Cappuccino	326
Caprese Salad	644
Chianti (bottle)	726
Chianti (glass)	644
Espresso	942
Fettuccine Alfredo	652

5. Discussion, Limitations & Conclusions

The success of this analysis will be gauged by its consistency when applied to multiple datasets generated by the accompanying script. Running the same notebook across various synthetic datasets generated by the same script will allow us to determine if the analytical logic produces comparable insights and identifies similar patterns. This ensures the robustness and generalizability of the analysis framework, independent of random variations within each dataset.

To further enhance the utility and depth of this analysis, future work should focus on increasing the sophistication and accuracy of the dataset generation process - it can be improved to be more accurate and have more attributes. Incorporating more realistic and mathematically derived attributes, such as profit margins for each dish calculated by meticulously modeling labor costs, utility costs, raw material costs, and miscellaneous overheads, would provide valuable insights into the restaurant's financial performance and enable a richer analysis of profitability trends across different restaurants, meal times, and individual menu items daily. The primary limitation encountered during this project was the significant time investment required for dataset development and learning new libraries like ipywidgets. This constraint necessitated an adjustment of the initial project scope due to the lack of readily available real-world data and the considerable effort involved in creating a realistic synthetic dataset without resorting to arbitrary random values. Consequently, several ambitious features, including the incorporation of packaging costs, an interactive dashboard, a menu optimizer, and staff data for labor cost analysis, were not implemented due to time constraints and working alone - it was just not feasible. Addressing these limitations in future work, ideally with more time and through collaboration, would significantly enhance this analysis's scope and practical applicability, leading to a more comprehensive understanding of restaurant operations.

References

- CS 439 Class Notes, Project Guidlines, Class Lectures
- ipywidgets documentation: https://ipywidgets.readthedocs.io
- SciPy General: https://docs.scipy.org/doc/
- Stats submodule: https://docs.scipy.org/doc/scipy/reference/stats.html
- Statsmodels General: https://www.statsmodels.org/stable/index.html
- LOWESS smoothing: https://www.statsmodels.org/stable/generated/statsmodels.nonparametric.smoothers_lowess.lowess.html
- Holt–Winters: https://www.statsmodels.org/stable/generated/statsmodels.tsa.holtwinters.ExponentialSmoothing.html
- Mlxtend General: http://rasbt.github.io/mlxtend/
- Apriori: http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/apriori/
- Association rules: http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/
- Seaborn https://seaborn.pydata.org/
- Pandas https://pandas.pydata.org/docs/
- NumPy https://numpy.org/doc/
- Random (built-in Python) https://docs.python.org/3/library/random.html
- UUID (built-in Python) https://docs.python.org/3/library/uuid.html
- Datetime (built-in Python) https://docs.python.org/3/library/datetime.html
- Scipy.stats https://docs.scipy.org/doc/scipy/reference/stats.html
- Scikit-learn - General: https://scikit-learn.org/stable/
- train_test_split: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
- RandomForestRegressor: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
- Ridge: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html
- KMeans: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
- Ipywidgets https://ipywidgets.readthedocs.io/en/stable/ (**I had never used or seen this before, I had to use Copilot to debug one part - there was a syntax issue with creating the tabs correctly for the cell that shows the menus. As I was working alone this was the only option I had - the professor said it was ok**)
- IPython.display https://ipython.readthedocs.io/en/stable/api/generated/IPython.display.html
- Matplotlib https://matplotlib.org/stable/contents.html