Rachel Schoenberg

CS 5001

Fall 2024

Final Project: Drip Labs Data

Project Description:

"Drip" is a startup at the Roux Institute that developed an app enabling customers to "punch" small business loyalty cards by tapping a sensor, earning rewards based on their loyalty tiers. Businesses participating in this program seek to understand their customers better through data analysis. The data includes transaction dates, spending amounts, payment types, and customer identifiers. Using this data, my program categorizes customers, identifies trends, and presents insights visually. My project focuses on developing a coding program to analyze customer data from participating businesses and generate a one-page visual report. This report provides actionable insights into customer behavior, loyalty patterns, and spending trends to optimize business strategies

Changes:

Throughout the project, numerous adjustments were made as challenges emerged and new requirements were introduced:

Customer Types Redefined
The first version of this program classified customers into these categories (which I defined from my customer service experience):

- Super Regulars: Visit five or more times a month.
- Regulars: Visit two to four times a month.
- Occasionals: Visit once a month.
- Inactive Regulars: Regulars who have not visited in the past month.
- Inactive Occasionals: Occasionals who have not visited in the past month.

Midway through development, Wheeler introduced the concept of RFM Analysis (Recency, Frequency, and Monetary Value). This led to redefining the customer categories:

- Active Regulars: Visit more than six times a month and visited in the past month.
- Inactive Regulars: Visited more than six times in the past but did not visit in October.
- Active Occasionals: Visit fewer than six times a month and visited in the past month.
- Inactive Occasionals: Visit fewer than six times a month in the past but did not visit in October.

Adjusting to this new framework required significant changes to the logic for merging recency and frequency data. Additional conditions were added to ensure the revised categories accurately represented customer behavior.

Adding a New Visualization

Wheeler requested a time-of-day by time-of-week visualization to track customer order patterns. For this I utilized a heatmap, which is a visualization that shows data values as colors in a grid-like layout. The created heatmap displays order counts based on hour and weekday. Creating this visualization involved:

- Data Preparation: Grouping transactions by day and time.
- Structuring: The grouped data was reorganized into a pivot table format, where:

  - Rows represent the hours of the day.

  - Columns represent the days of the week.

  - The values in the table represent the order counts for each hour-day combination.

- Visualization: Generating a Seaborn heatmap with labeled axes and annotations for clarity.

Data Loading Logic

Initially, the program was designed to load data from a single Google Sheet by downloading a CSV file. However, “Drip” organizes data across multiple worksheets within the same file. This required implementing dynamic logic to:

- Loop through worksheets.
- Load data from each sheet.
- Merge all data into a unified dataset.

Handling inconsistent structures across worksheets while maintaining data integrity was a significant challenge.

Customer Categorization Logic
Early versions of the program struggled to correctly classify active and inactive customer types. For example, customers with recent visits but low frequency were often misclassified. Updating the logic to align with the new RFM-based categories resolved these issues and significantly improved analysis accuracy.

Commenting Style Changes
My original commenting style was monotonous, confusing, and repetitive (hindereding readability). I revised my commenting style to focus on concise comments that addressed:

- Purpose: The intent of each method.
- Syntax: Highlighting key libraries or techniques used.
- Summary: A short explanation of how the method works.

This change streamlined the code and made debugging more efficient.

New Syntax Adoption
Several libraries and techniques were critical for the project:

- Pandas: Used extensively for data processing tasks such as grouping, aggregating, and pivoting datasets.
- Matplotlib: Created bar charts, pie charts, and other visuals for the report.
- Gspread: Managed data extraction from Google Sheets.
- Seaborn: Enhanced visualizations, particularly with heatmaps to illustrate time-based order trends.

Visual Adjustments
Creating a cohesive one-page visual report posed design challenges. The need to display multiple insights clearly required frequent adjustments to:

- Spacing and alignment of visuals.
- Font sizes for readability.
- Chart layouts for clarity and balance.

Reflection:

What I Learned
Implementing this project deepened my understanding of data processing and visualization. I learned:

- How to integrate multiple data sources using the libraries Pandas and Gspread.
- The importance of clear and actionable visualizations in conveying insights.
- How RFM Analysis can help businesses better understand and segment their customers.
- The challenges of adapting to changing requirements and how to build flexible code to accommodate future changes.

I also gained valuable experience in balancing technical precision with creative design. For example, aligning multiple visuals on one page required both programming skills and an eye for design.

What I Would Do Differently
If I could start over, I would:

1. Plan Visuals Early:
   I would design the report layout and visuals at the beginning of the project. This would help ensure the code is structured to produce consistent and well-aligned outputs.
2. Modularize the Code Further:
   While the current code is functional, some methods could be broken into smaller, more modular components. This would make the code easier to test, debug, and extend.
3. Anticipate Data Variability:
   Early in the project, I assumed all data sources would have the same structure. If I had anticipated variability, I could have built more robust preprocessing logic from the start.
4. Start with RFM Analysis:
   Incorporating RFM-based customer categories from the beginning would have saved time and avoided rewriting parts of the code.

Challenges

1. Loading Data from Google Sheets:
   Authentication and handling multiple worksheets were time-consuming. Developing dynamic loading logic was critical to overcoming this challenge.
2. Categorizing Customers:
   Redefining customer types mid-project required rewriting key logic. Ensuring the new categories aligned with business goals while maintaining accurate results was challenging but rewarding.
3. Designing Neat Visuals:
   Ensuring the visuals were intuitive and aligned on one page took several iterations. The addition of the time-of-day by time-of-week heatmap added complexity but ultimately enhanced the report.

Acknowledgments/Citations:

CS 5001-

- Dr. Molly Domino's Lectures (Notes from)
- Modules

People -

- Frank Chen, Co-founder of Drip Labs
  (Specification of what his company needed)
- Wheeler Boyd, Co-founder of Drip Lab
  (Specifications on RFM Analysis)
- Puneet Ramini, Project Management Student (Boston) Co-Op
  (An idea of how Drip could utilize data for their company)
- Dan O'Brien, Align Data Science Student
  (How to use Jupyter Notebooks - didn't end up using)

Web -

- https://www.w3schools.com/python/python_pip.asp (Pip)
- https://www.geeksforgeeks.org/how-to-install-python-pandas-on-windows-and-linux/ (Installing Pandas)
- https://docs.python.org/3/library/venv.html (Virtual Environments)
- https://www.youtube.com/watch?v=2uvysYbKdjM (Pandas)
- https://pandas.pydata.org/docs/reference/testing.html (Testing Pandas)
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html (Dataframes)
- https://www.youtube.com/watch?v=t6WSY9D_ORQ (Loading Data from GoogleSheets)
- https://skills.ai/blog/import-google-sheets-to-pandas/ (Loading Google Sheet Data to Pandas)

- https://www.bpwebs.com/import-data-from-one-google-sheet-to-another/ (More than 1 Sheet)
- https://developers.google.com/sheets/api/guides/concepts (Google Sheets)
- https://skills.ai/blog/import-google-sheets-to-pandas/#:~:text=Chunk%20Processing%3A%20If%20your%20dataset,like%20float32%20instead%20of%20float64%20. (Importing Google Sheets to Pandas)
- https://colab.research.google.com/#scrollTo=-gE-Ez1qtyIA (Google Colab)
- https://www.youtube.com/watch?v=DkjCaAMBGWM (Pandas for Data Analysis)
- https://www.youtube.com/watch?v=bqw5-8f-cEI (Jupyder Notebooks & Data Science)
- https://pandas.pydata.org/docs/reference/api/pandas.to_datetime.html (Pandas DateTimes)
- https://www.geeksforgeeks.org/jupyter-notebook-cell-magic-functions/ (Magic Commands)
- https://www.geeksforgeeks.org/useful-ipython-magic-commands/ (Magic Commands)
- https://www.youtube.com/watch?v=WvmFjuGxneI (RFM Analysis)
- https://youtu.be/9wxWrERZvss?si=7jjFzWbzdproC_jK (RFM Analysis)
- https://medium.com/@hhuseyincosgun/customer-segmentation-rfm-analysis-recency-frequency-monetary-5b29d5d45e35 (RFM Analysis)
- https://www.putler.com/rfm-analysis/(RFM Analysis)
- https://docs.python.org/3/library/datetime.html (Datetime)
- https://www.w3schools.com/python/python_file_handling.asp (Files)
- https://pandas.pydata.org/docs/reference/general_functions.html (Pandas Functions)
- https://matplotlib.org/stable/gallery/pie_and_polar_charts/pie_features.html (Matplotlib Pie Chart)
- https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.bar.html (Matplotlib Bar Chart)
- https://www.geeksforgeeks.org/how-to-create-a-pie-chart-in-seaborn/ (Create Pie Chart Using Seaborn)
- https://seaborn.pydata.org/index.html (Seaborn)
- https://docs.python.org/3/library/unittest.html (Unittest)
- https://www.pythontutorial.net/python-unit-testing/python-unittest-mock/ (Mock Unittest)