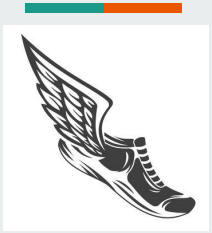


StepAhead

Justin Davis, Thompson Morgan

Company Background

- Our mission is to match our customers with their perfect running shoes by offering recommendations based on individual preferences.
- We are a new company, and after pitching our idea to all the big names in the running shoe business, we were lucky enough to land our first client with Brooks Running.
- One of our main goals is working with even more running shoe brands in the future.
- “StepAhead will pair you up with your sole-mates before taking the next step of your running journey!”

Data

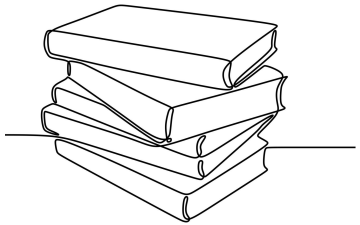
- Brooks Running Shoes dataset
  - Found on Kaggle, created by Hannah Collins (https://www.kaggle.com/datasets/hannahcollins/2020-brooks-running-shoes)
  - Includes data obtained from Brooks (https://www.brooksrunning.com/en_us)
  - Includes attributes such as type, price, support, experience, surface, weight, and arch
- Randomly-generated Customer Preferences dataset
  - Created on Mockaroo, assigned 500 customers a budget and their preferred support type, surface type, and arch type (https://www.mockaroo.com/)
- Quality checks occur after data is cleaned and before it is entered into the database.
- Shoe data we collect is available to the public but all customer data is kept private.

Data Cleaning

BROOKS

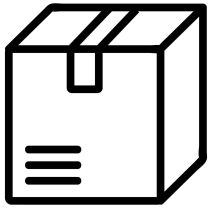
kaggle

mockaroo

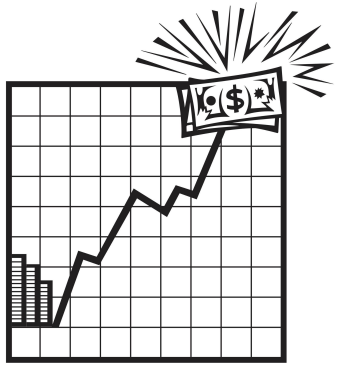


ETL Process

Clean Data with Python in Jupyter Notebook

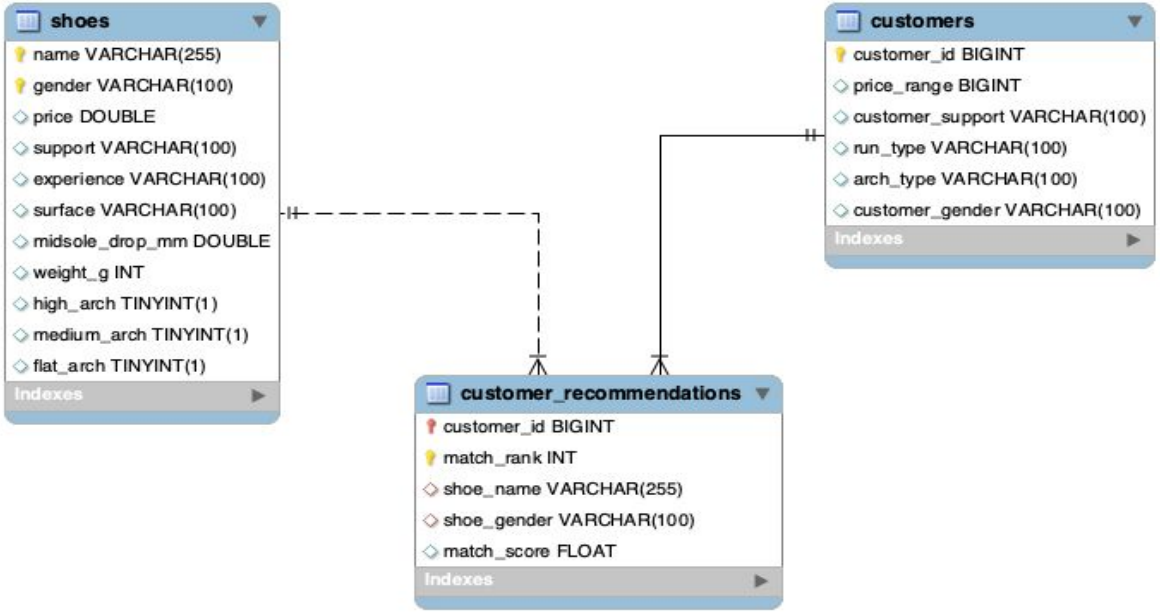


Warehouse



Analysis

Our Database



- Created a scoring system to assign scores for a shoe to a specific customer
  - The score is weighted based on gender > price > support > arch type
  - Only looked at the three highest scores for each customer

Database Storage

- All of our data is stored locally in a MySQL server.
- The database is managed on MySQL Workbench.
- Access to the files used to create the server are only granted to key personnel.
- Final files were shared via company emails to ensure a backup could be downloaded in the future if it was needed.
- In the case of lost data or server failure, there are multiple backups saved between multiple hard drives.

Question 1

- Which shoes appear most frequently across all match ranks?

```
with engine.connect() as connection: # Establish a connection
    # Select from the customer_recommendations table and count(*)
    # And group by shoe name in descending order of count
    question_one = text("""SELECT shoe_name, COUNT(*) AS count
                           FROM customer_recommendations
                           GROUP BY shoe_name
                           ORDER BY count DESC
                        """) # Define the query - text() ensures that the query string is read as a SQL expression
    question_one = pd.read_sql(question_one, connection) # Use pandas to read the sql query with the connection to the database

# Print the results
question_one
```

Graph 1

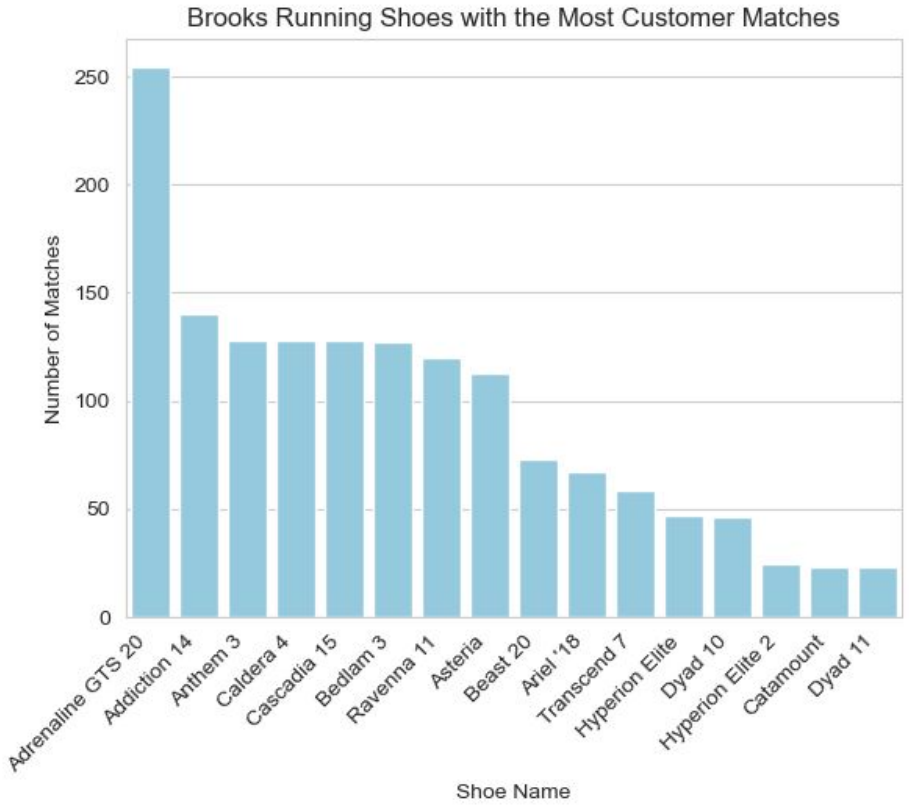


Figure 1: Bar graph illustrating the shoes with the most matches based on each customer's preferences.

Question 2

- Which shoes appear most frequently across all rank one matches?

```
with engine.connect() as connection: # Establish a connection
    # Select from the customer_recommendations table and count(*)
    # Where the match_rank is equal to 1
    # And group by shoe name in descending order of count
    question_two = text("""SELECT shoe_name, COUNT(*) AS count
                           FROM customer_recommendations
                           WHERE match_rank = 1
                           GROUP BY shoe_name
                           ORDER BY count DESC
                       """) # Define the query - text() ensures that the query string is read as a SQL expression
    question_two = pd.read_sql(question_two, connection) # Use pandas to read the sql query with the connection to the database

# Print the results
question_two
```

Graph 2

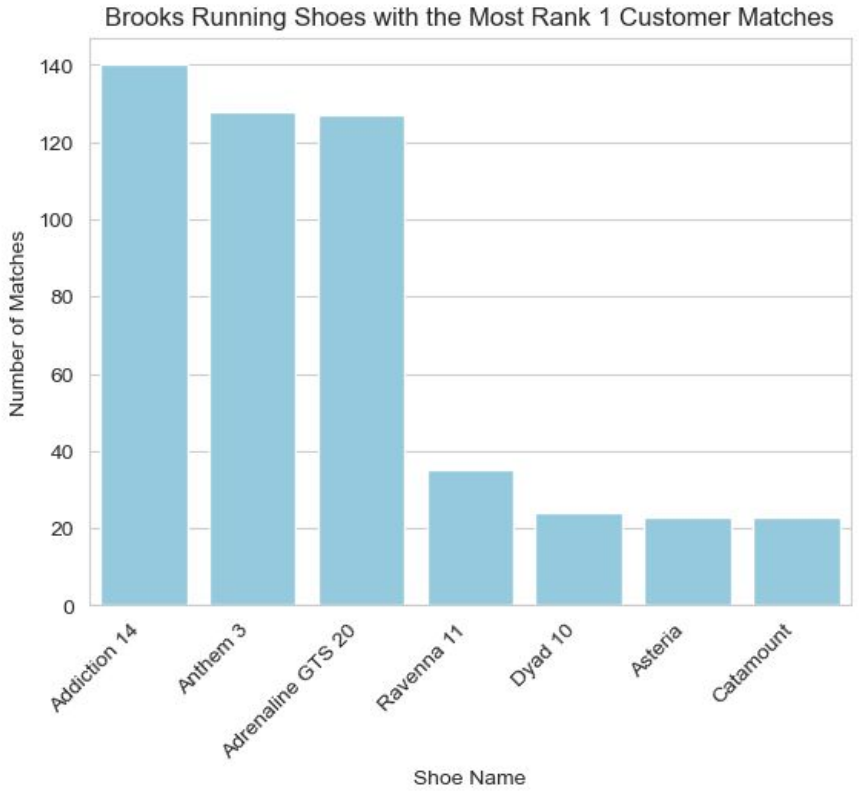


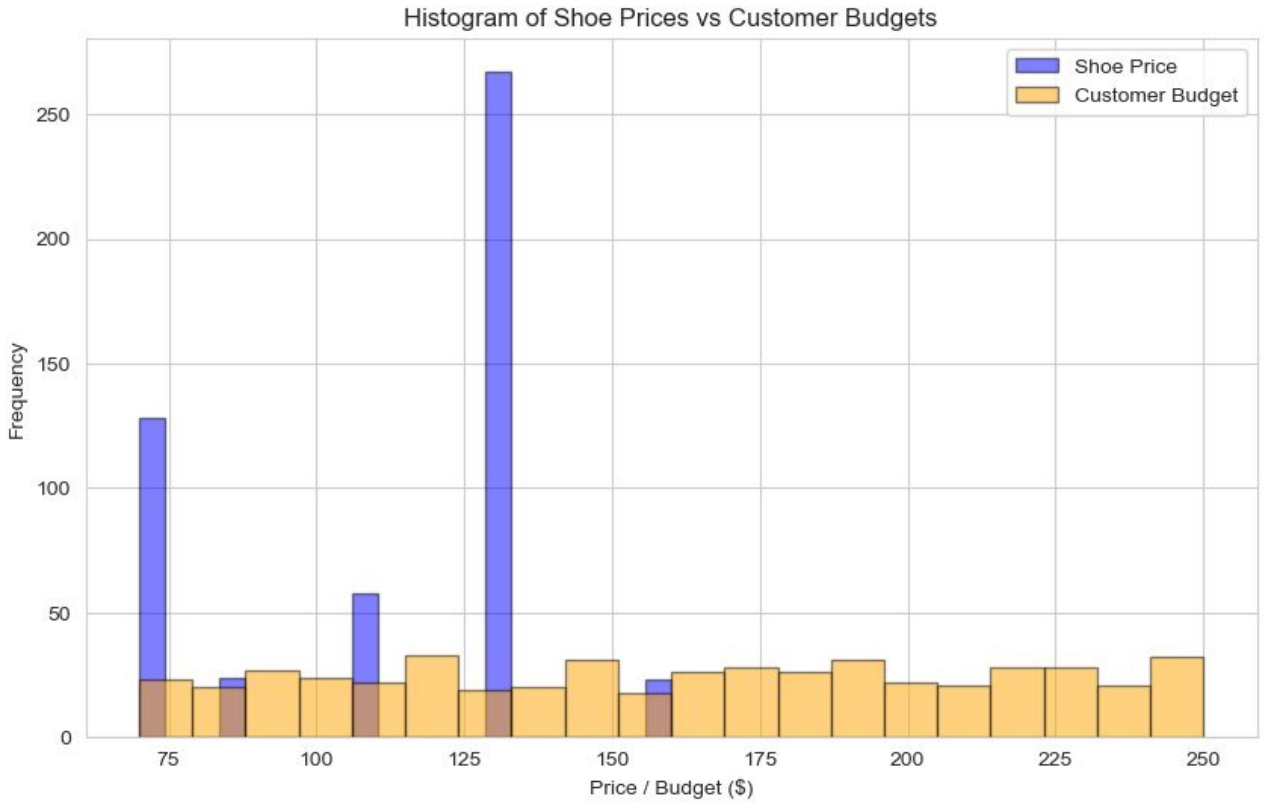
Figure 2: Bar graph illustrating the shoes with the most rank one matches based on each customer's preferences.

Question 3

- How do customers' budgets compare to the price of their top shoe recommendation?

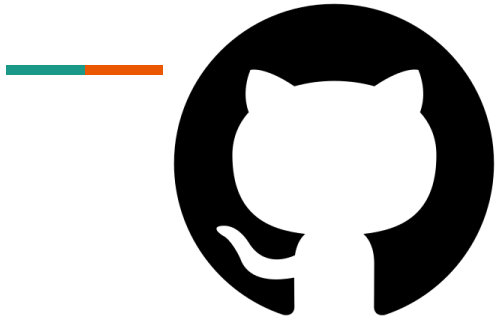
```
with engine.connect() as connection: # Establish a connection
    # Selecting the customer_id, price_range, shoe_name, and the shoe_price
    # Created a case query when the shoe_price is less than price_range (max_budget) then it's underbudget
    # If its the same, then its exactly on budget
    # If the shoe price is greater than the max budget, then its over budget
    # Labeling this new column as budget group
    # Joining the customers and shoes table with the customers_recommendations table
    question_three = text("""SELECT DISTINCT c.customer_id,
                                c.price_range AS max_budget,
                                cr.shoe_name,
                                s.price AS shoe_price,
                            CASE
                                    WHEN s.price < c.price_range THEN 'Under Budget'
                                    WHEN s.price = c.price_range THEN 'Exactly on Budget'
                                    WHEN s.price > c.price_range THEN 'Over Budget'
                                END AS budget_group
                            FROM customer_recommendations cr
                            JOIN customers c ON cr.customer_id = c.customer_id
                            JOIN shoes s ON cr.shoe_name = s.name
                            WHERE cr.match_rank = 1
                            ORDER BY c.customer_id;
                            """) # Define the query - text() ensures that the query string is read as a SQL expression
    question_three = pd.read_sql(question_three, connection) # Use pandas to read the sql query with the connection to the database

# Print the results
question_three
```

Graph 3


Histogram of Shoe Prices vs Customer Budgets
Shoe Price
Customer Budget
Frequency
0
50
100
150
200
250
75
100
125
150
175
200
225
250
Price / Budget ($)

Conclusion

- Our data and scoring system ensures our customers make informed decisions based on their personal needs.
- We hope to work with more shoe companies in the future, and will always be thankful to Brooks for partnering with us at the start.



Justin Davis
DavisJ41

Thompson Morgan
tjmorgan462

Scan to visit our
Github Repository!

https://github.com/DavisJ41/Data_Gathering_Final