IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Mart
18-11-2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- For this project we tackled a couple of problems, of which were:
  - SpaceX Falcon 9 First Stage Landing Prediction
  - Using SQL to get information about Launch sites and more
  - Find more information about Results of the falcon 9 landings, success rates, etc…
  - Doing some Location analyses with Folium
  - And doing some landing predictions using sklearn

Introduction

- In our quest to harness the power of data science, we encountered a multifaceted challenge: How do we effectively gather, clean, and analyze complex datasets to uncover meaningful insights? T

- his journey began with a diverse array of raw data, presenting us with the initial problem of transforming it into a coherent and analyzable format.

- Our primary objective was to navigate through this labyrinth of data to answer critical questions and make data-driven decisions, a task that required meticulous data wrangling, innovative analytical approaches, and a keen understanding of the underlying patterns within the data.

Section 1

Methodology

Methodology

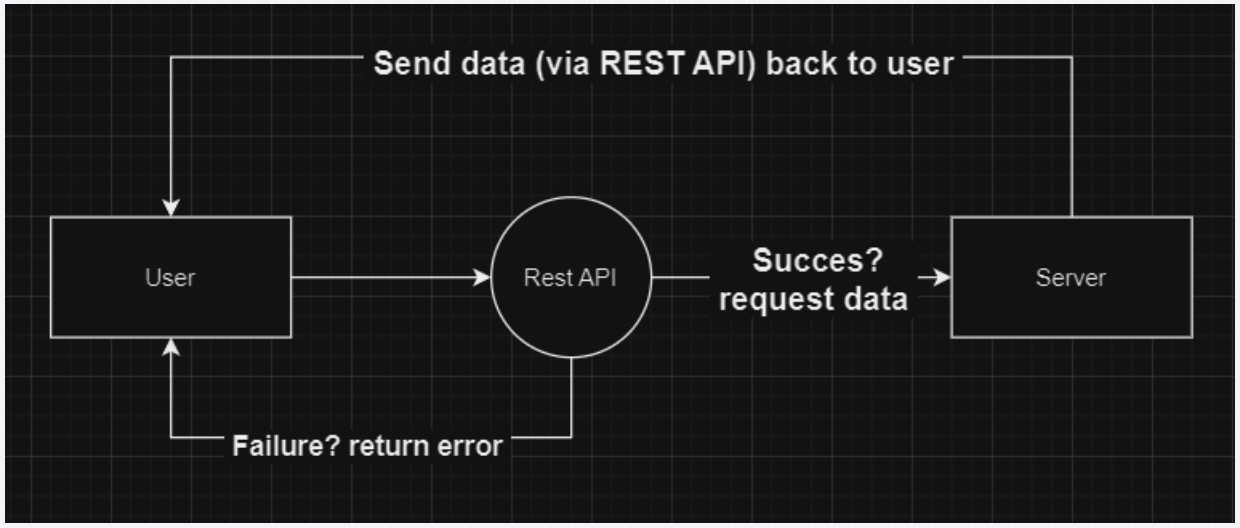
Executive Summary

- Data collection methodology:
  - Data was collected using a REST API call. And SQL queries (scraping).
- Perform data wrangling
  - This data then was converted into a python dataframe
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection – SpaceX API

- So REST API calls are used to interact with web services, allowing a client to send a request to a server, which then performs an action (like retrieving data) and returns a response.

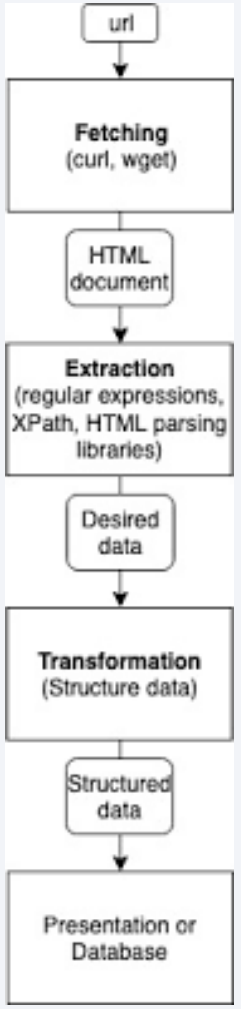
- The github: https://github.com/KingerNL/Coursera_Data_Science



Data Collection - Scraping

- SQL queries, on the other hand, are statements used to perform operations on databases, such as retrieving, updating, or deleting data, by directly interacting with the database management system.

- The github: https://github.com/KingerNL/Coursera_Data_Science/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb



Data Wrangling

- Cleaning, structuring, and enriching raw data into a desired format for better decision making in less time. It involves transforming and mapping data from one "raw" form into another format to make it more appropriate and valuable for a variety of downstream purposes such as analytics.
- For the data used in the notebook, I've conducted:

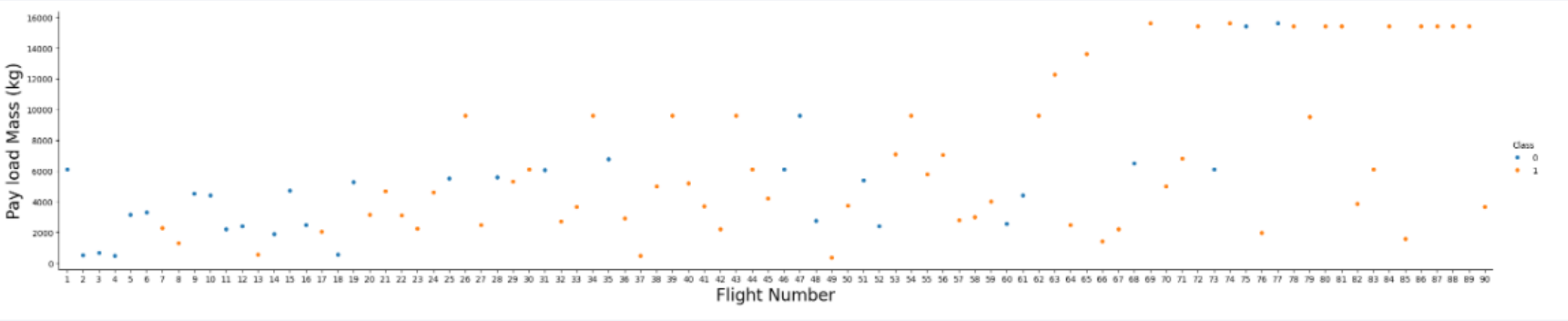
1. Conducted intensive analyses on the data
2. Removed none entries
3. 2. Created landing outcome label from outcome column

- Can be found in the github: https://github.com/KingerNL/Coursera_Data_Science/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- I've used mostly seaborn, because it gave more overhead and made better plots, I've also stuck to trying to compress most the data in a neat graph.

- The github page I've used: https://github.com/KingerNL/Coursera_Data_Science/blob/main/jupyter-labs-eda-dataviz.ipynb



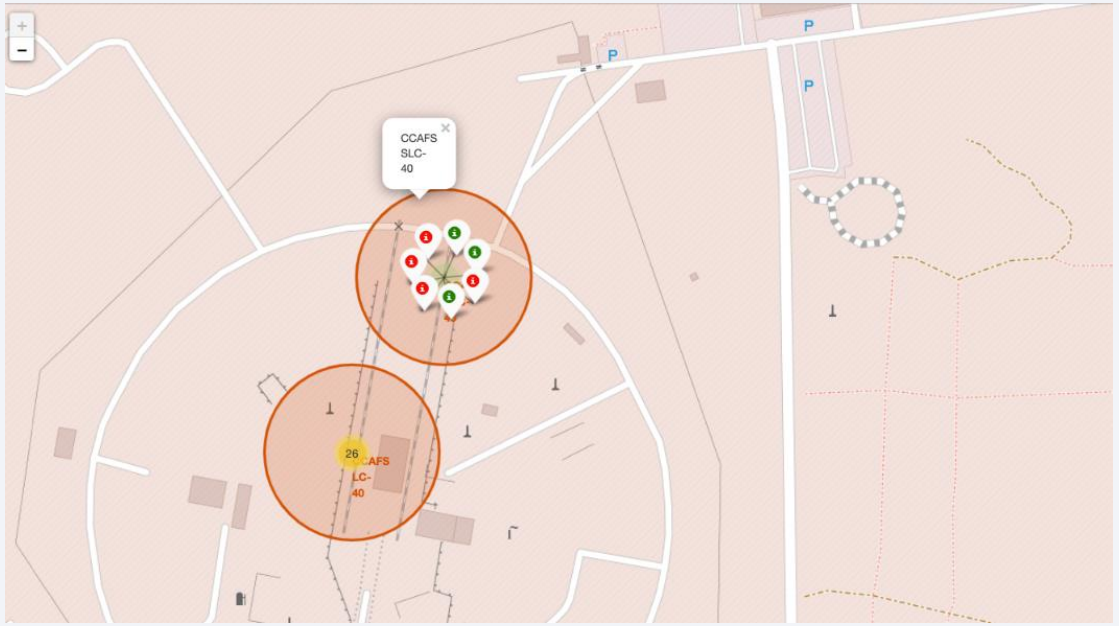
EDA with SQL

- The queries I've performed:
  - Task 1: Distinct
  - Task 2: Limit
  - Task 3: Sum
  - Task 4: Average
  - Task 5: Minimum
  - Task 6: Filter
  - Task 7: Count
  - Task 8: Maximum
  - Task 9: Extract
  - Task 10: Rank

Link to github page:
https://github.com/KingerNL/Coursera_Data_Science/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- The Folium map has objects such as
  - markers
  - circles
  - lines



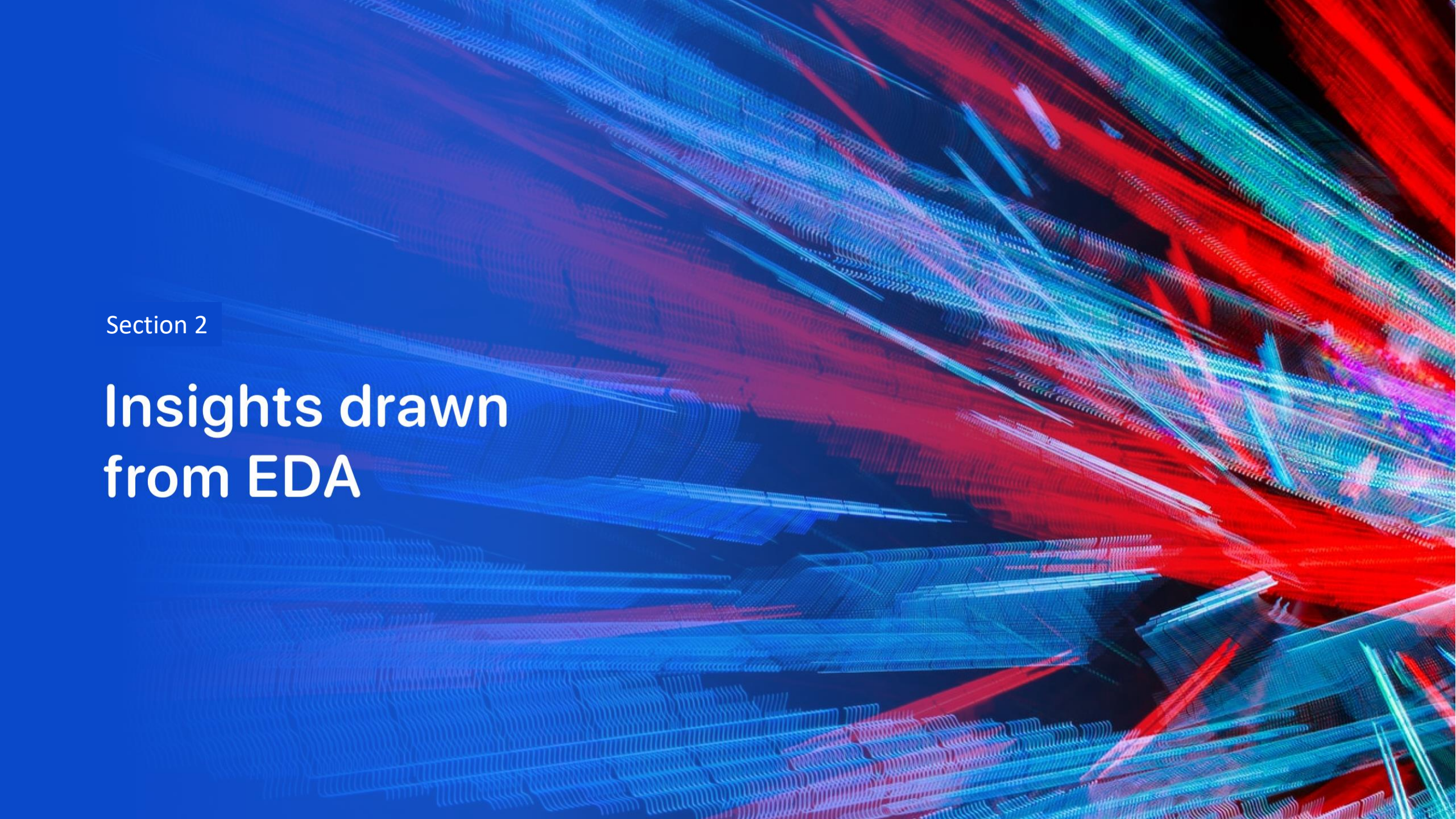
- Link to Github: https://github.com/KingerNL/Coursera_Data_Science/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- I've added a lot of plots for various information
- Link to Github: https://github.com/KingerNL/Coursera_Data_Science/blob/main/jupyter-labs-eda-dataviz.ipynb

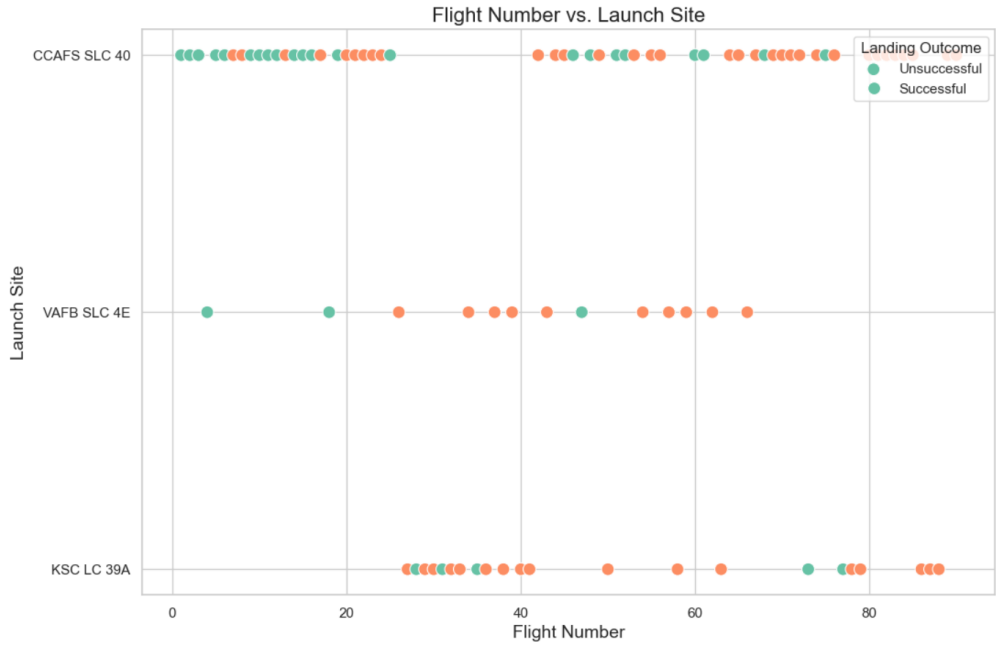
Predictive Analysis (Classification)

- While used the following techniques:
  - LogisticRegression
  - GridSearchCV
  - DecisionTreeClassifier
  - KNeighborsClassifier
- Decision Tree was the best, but more on that later.
- Github: https://github.com/KingerNL/Coursera_Data_Science/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb


Section 2
Insights drawn from EDA

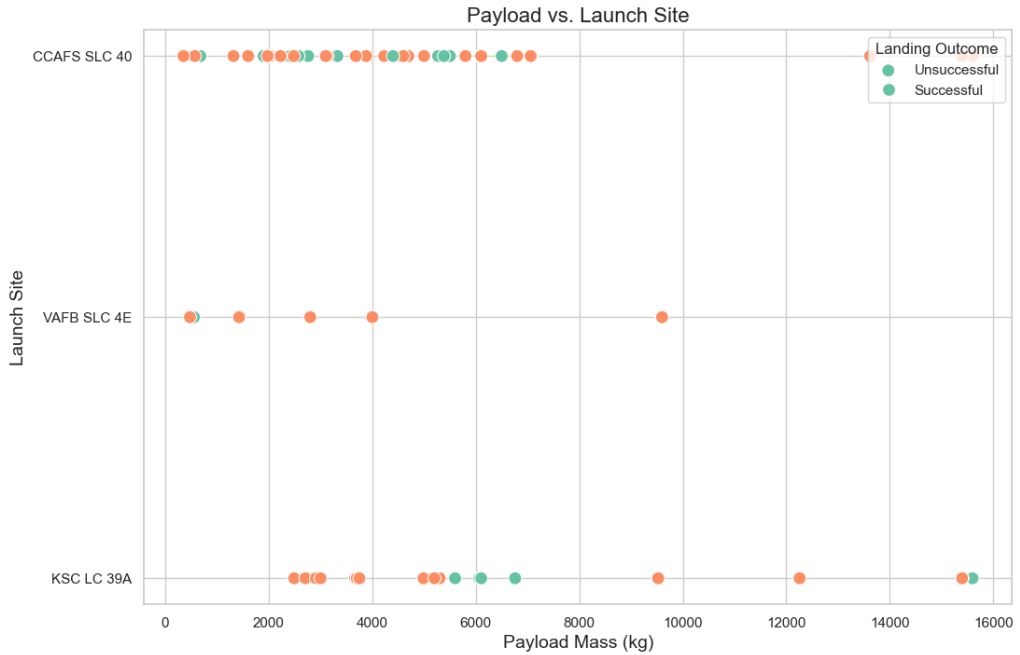
Flight Number vs. Launch Site

This is a seaborn scatterplot for the different launch sites, and the different flight numbers



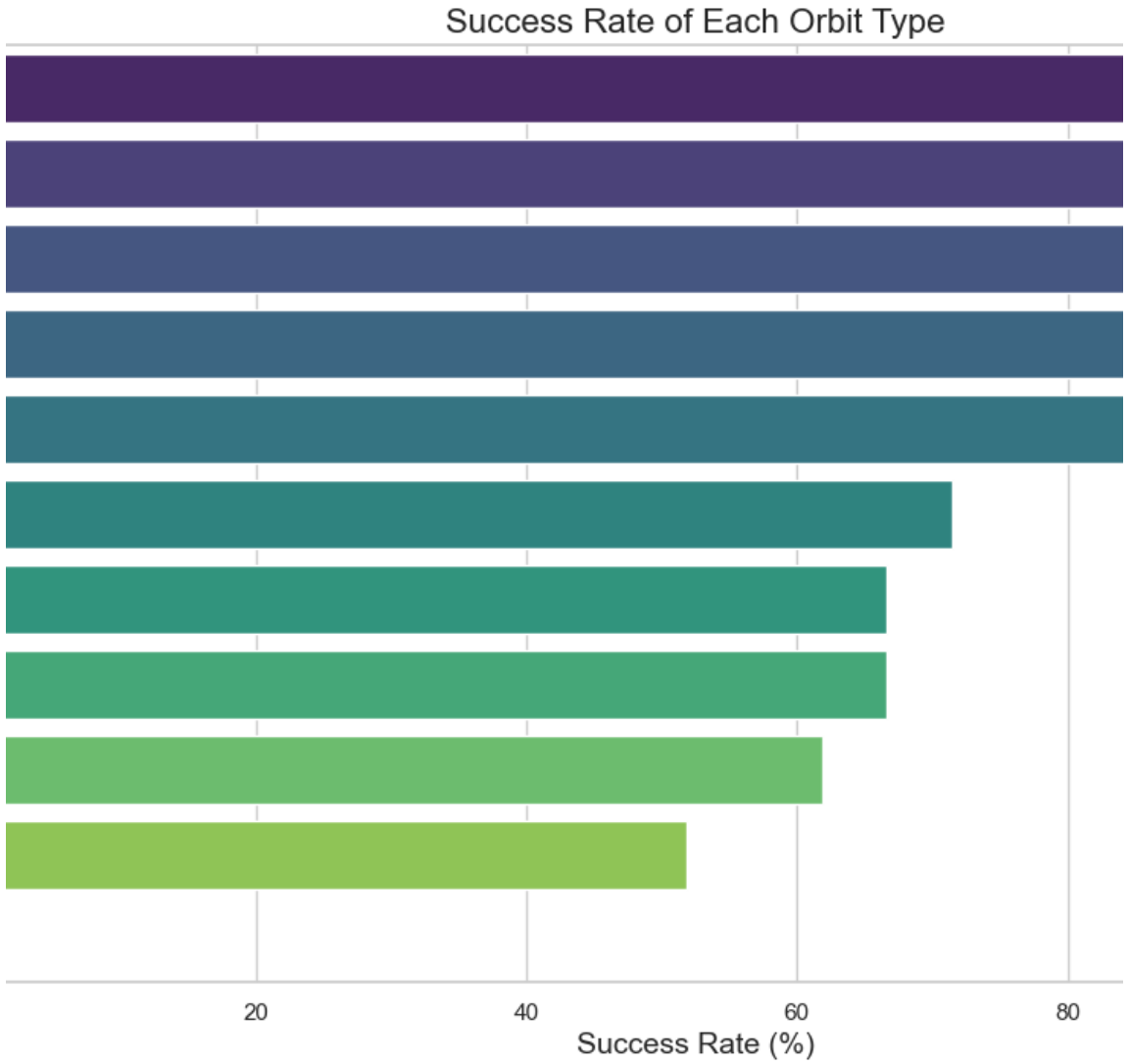
Payload vs. Launch Site

This is a seaborn scatterplot for the different payloads at the different launch sites



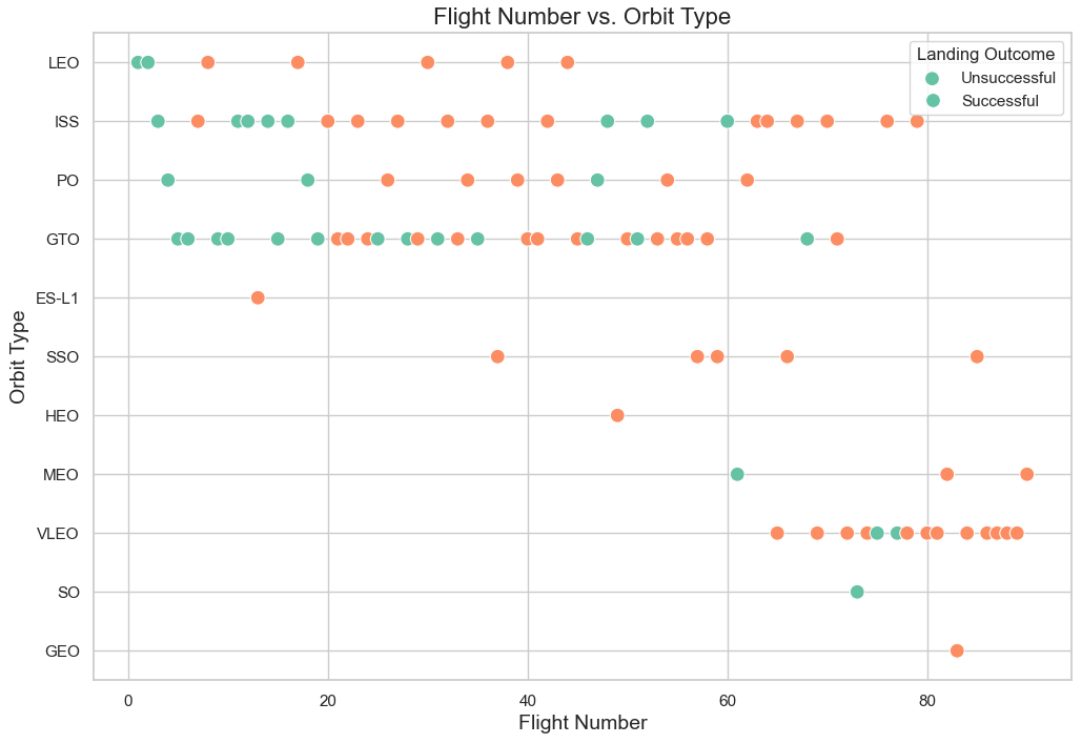
Success Rate vs. Orbit Type

- This is the barr scart with success rate for each orbit type



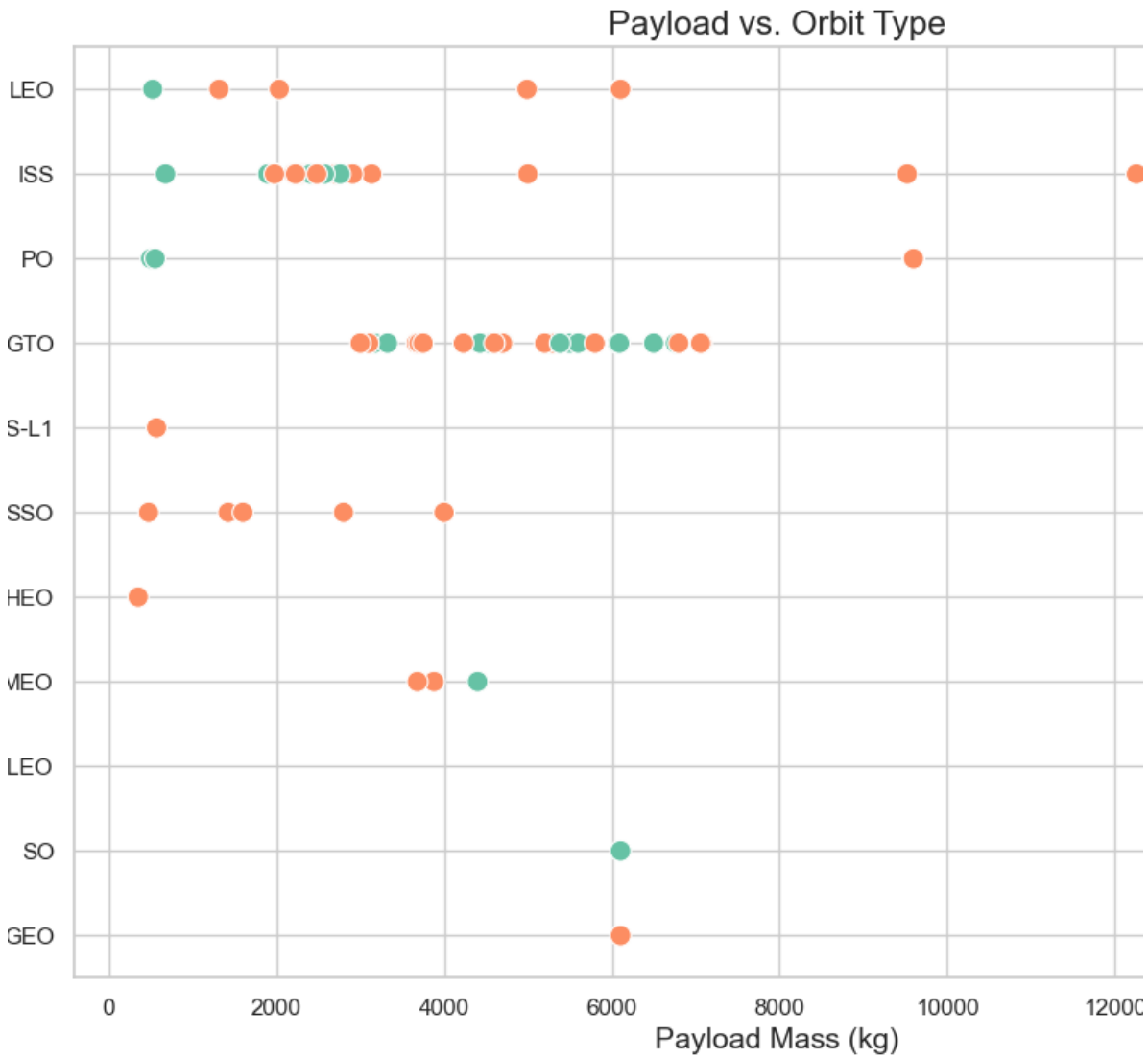
Flight Number vs. Orbit Type

- This is the flight number vs orbit type scatter cart.



Payload vs. Orbit Type

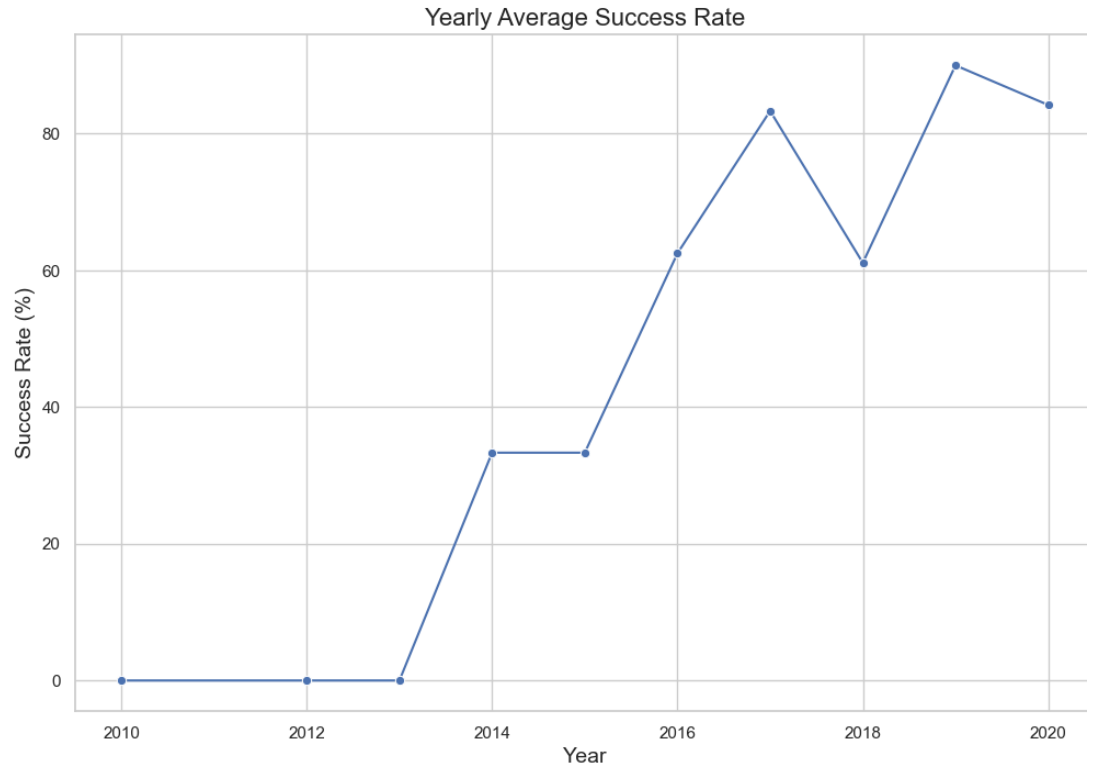
- Here you can see the difference in payload vs orbit type.



Launch Success Yearly Trend

- Here you can see the success rate over the years

- You can see clearly that the success rate increased over the span of 4 years by 80%. From 2013 to 2017.



Launch Site Names

- CCAFS SLC 40' 'VAFUnique Launch Sites:
  - 0 CCAFS SLC 40
  - 1 CCAFS SLC 40
  - 2 CCAFS SLC 40
  - 3 VAFB SLC 4E
  - 4 CCAFS SLC 40
  - …
  - 85 KSC LC 39A
  - 86 KSC LC 39A
  - 87 KSC LC 39A
  - 88 CCAFS SLC 40
  - 89 CCAFS SLC 40'

Unique Launch Site Beginning with 'CCA':

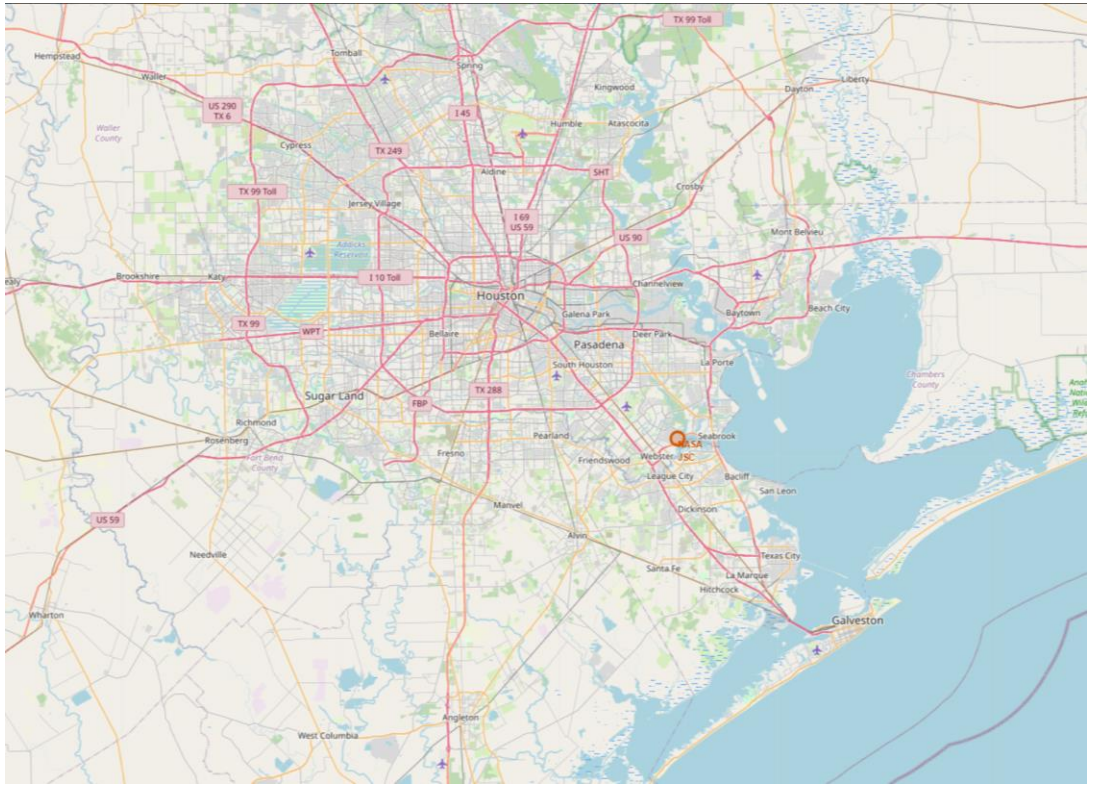
- **'CCAFS SLC 40'**

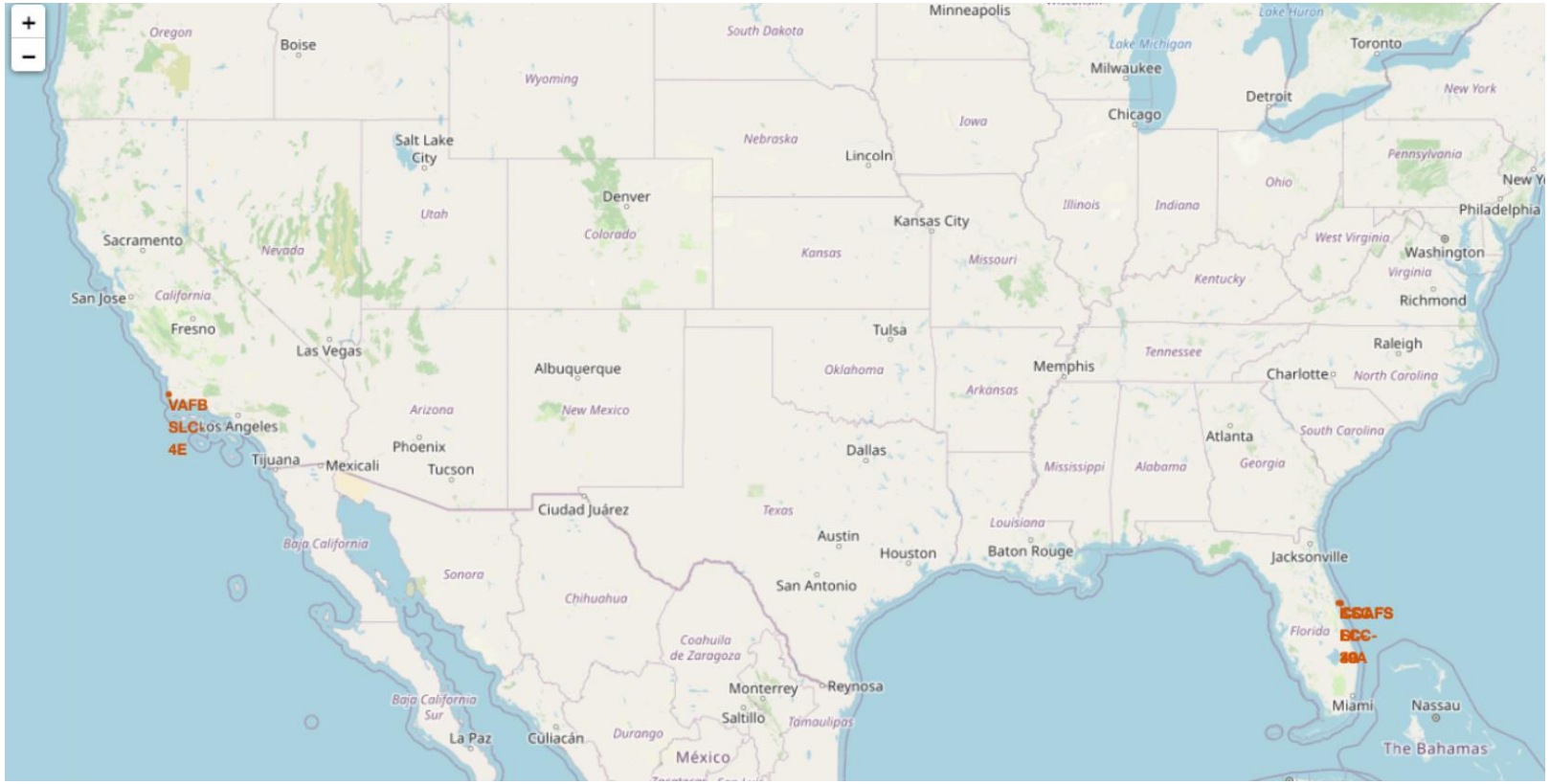
Section 3

Launch Sites Proximities Analysis

NASA JSC Houston

- A screenshot of Houston
- This is were NASA JSC is located



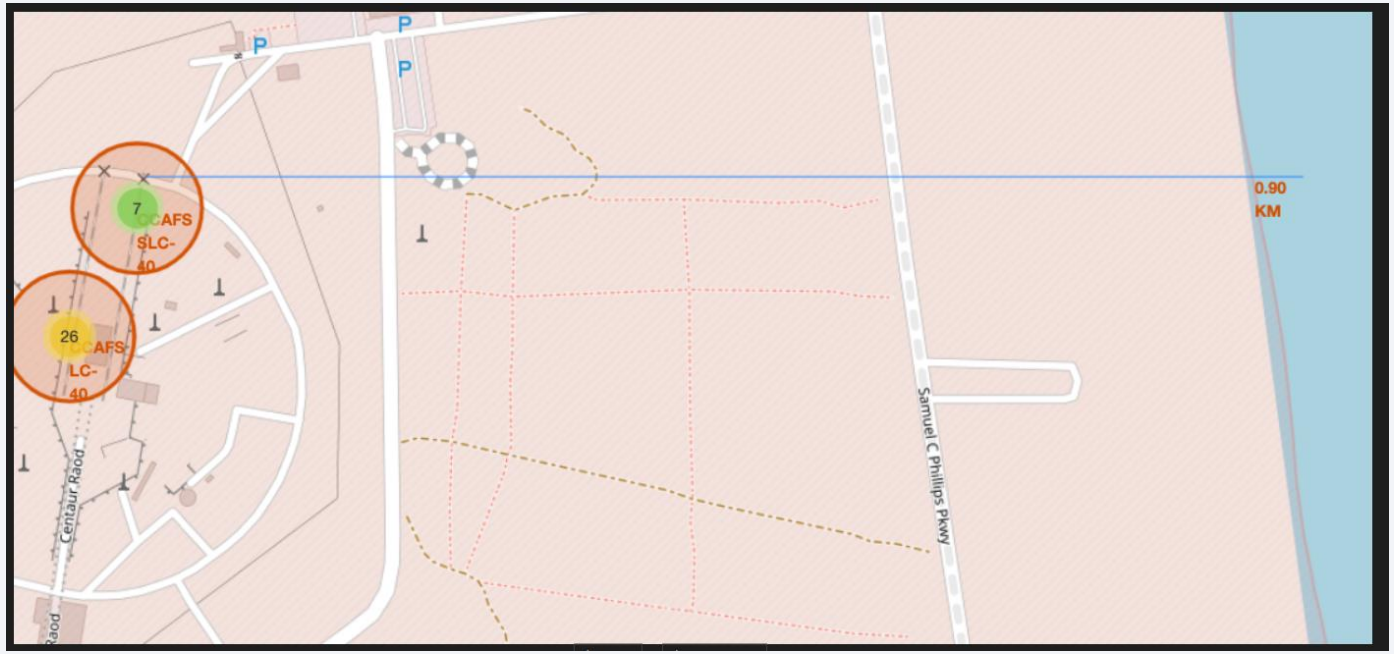


Launch sites

- These were some of the launch sites

A line photo, from water to launch site

- An photo of the updated map with direction line from water to launch site



Section 5

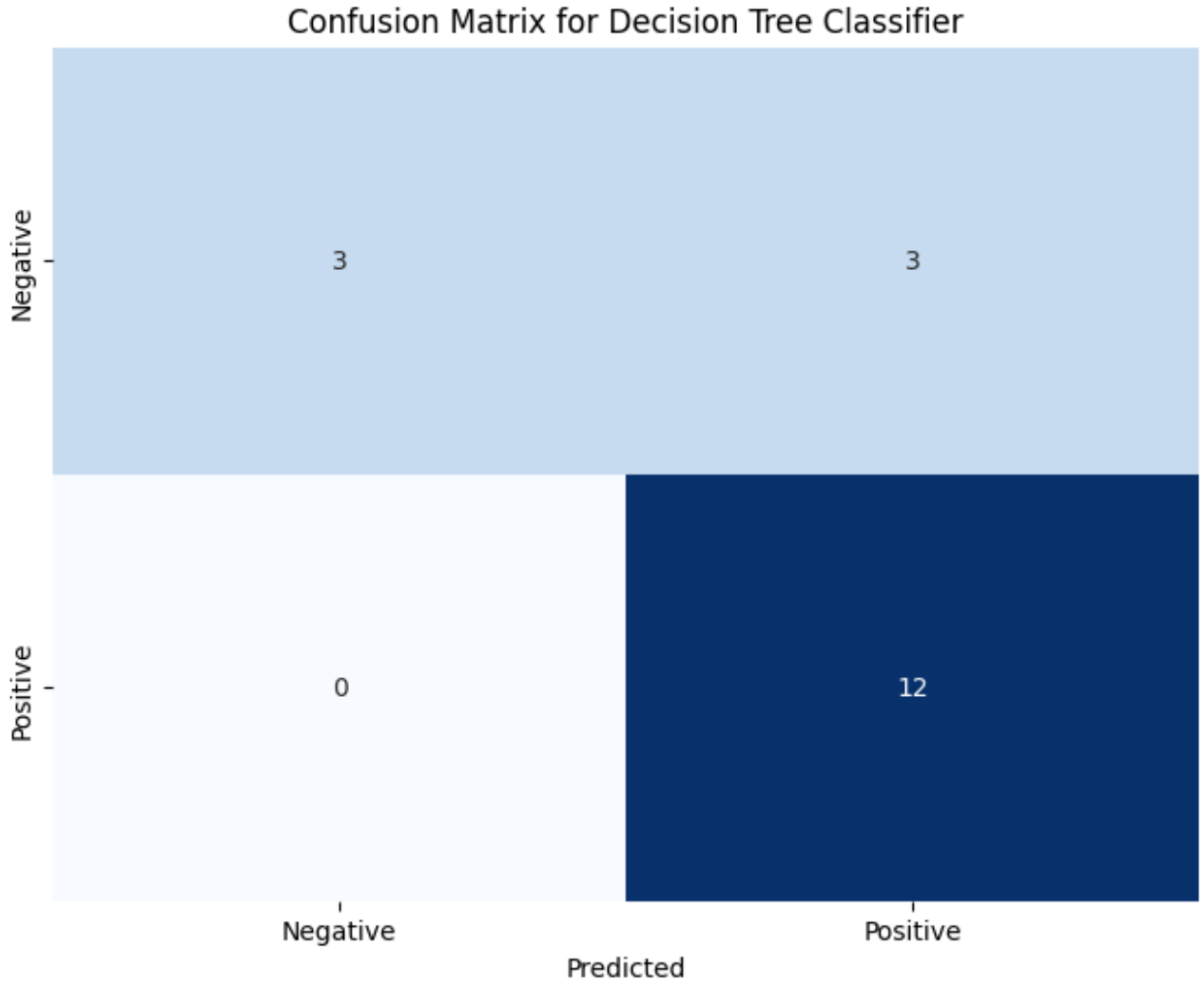
Predictive Analysis (Classification)

Classification Accuracy

- While used the following techniques:
  - LogisticRegression
  - GridSearchCV
  - DecisionTreeClassifier
  - KNeighborsClassifier

- Decision tree had the highest accuracy, with:
  86.07 %

Confusion matrix Decision Tree



Conclusions

- The conclusion was that after doing some model tests, the following statements could be made:
  - Decision tree was the best, followed by;
  - GridSearchCV and KNeighborsClassifier
  - At last place we have LogisticRegression, which scored the worst.

- For more information you can see the notebook at: https://github.com/KingerNL/Coursera_Data_Science/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Thank you!