IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

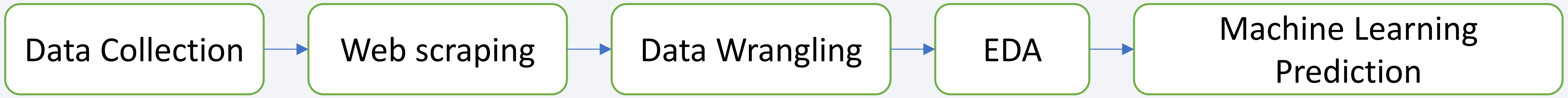
Varun Arya
8/12/2024

Outline

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

Executive Summary

- Summary of methodologies



- Summary of all results
  - All of the code written works perfectly well
  - All of the steps mentioned in the summary of methodologies were executed successfully in code

Introduction

Project Background and Context

1. SpaceX's Reusability Goal: SpaceX aims to reduce the cost of space travel by reusing the first stage of the Falcon 9 rocket, which is designed to return and land after launch.
2. Cost Efficiency: A successful landing of the first stage is crucial in saving millions of dollars per launch, and in making space missions more affordable and competitive.
3. Competitive Edge: SpaceX’s ability to reliably land and reuse rocket stages gives it a significant advantage over other space agencies and private companies, which typically use expendable rockets.
4. Technological Advancement: Predicting the likelihood of a successful landing can help improve the algorithms and technology involved in rocket recover. This in turn contributes to advancements in aerospace technology.
5. Environmental Impact: Reusing rocket stages reduces the environmental footprint of space missions by minimizing the need for new materials and reducing space debris.

Problems You Want to Find Answers To

1. Landing Prediction: Can we accurately predict the likelihood of the Falcon 9’s first stage landing successfully based on historical launch data?
2. Influence of Payload: How does the payload mass and orbit destination affect the success rate of the first stage landing?
3. Impact of Launch Sites: Do different launch sites affect the probability of a successful landing? Which sites have the highest success rates?
4. Effect of Rocket Configuration: How do variations in rocket configuration, such as the use of grid fins or the type of landing pad, influence landing outcomes?
5. Improving Success Rates: What factors can be adjusted or optimized to improve the likelihood of successful landings in future missions?

Section 1

Methodology

Methodology

Executive Summary

- **Data Collection Methodology**
    - In this project I have utilized SpaceX's REST API to gather historical launch data. This includes details on rocket launches, payloads, launch sites, and landing outcomes. Furthermore, I wrote code that extracts and parses data from the API into a structured format suitable for analysis, while focusing on key attributes relevant to the Falcon 9 first stage landings.
- **Perform Data Wrangling**
    - In this project, I also processed raw data by filtering, cleaning, and transforming it to handle missing values and inconsistencies. Moreover, I normalized and structured the data to prepare it for analysis, to ensure that it was in a format that could be used effectively for both exploratory and predictive analyses.

- **Perform Exploratory Data Analysis (EDA) Using Visualization and SQL:**
  - I wrote code that conducts exploratory analysis to uncover trends, patterns, and relationships within the data. Furthermore, I also employed visualization techniques to better understand the data distribution, and SQL queries to extract insights from large datasets.
- **Perform Interactive Visual Analytics Using Folium and Plotly Dash:**
  - I developed interactive visualizations to explore geographic data and landing outcomes, using tools like Folium for mapping and Plotly Dash for dashboard creation. This enables users to interact with the data dynamically, which facilitates deeper insights into the factors affecting landing success.
- **Perform Predictive Analysis Using Classification Models:**
  - I wrote code that builds, tunes, and evaluates classification models to predict the success of Falcon 9 first stage landings. Moreover, I applied machine learning techniques to identify the most significant predictors of landing success, with the goal of improving future landing outcomes.

Data Collection

Describe How Data Sets Were Collected:

1. The data was obtained using SpaceX's REST API via HTTP GET requests, specifically targeting the endpoint for past launches.
2. The JSON response from the API was parsed and normalized into a Pandas DataFrame, which allows easier data manipulation and analysis.
3. Furthermore, the specific attributes such as booster versions, payload masses, and launch site details were extracted using custom auxiliary functions.

Present Your Data Collection Process Using Key Phrases:

1. First, we make API call to SpaceX `/v4/launches/past` endpoint and receive JSON response.
2. Then, we parse and normalize the JSON response into a structured Pandas DataFrame.
3. And finally, we extract and compile specific launch details into a comprehensive dataset using custom helper functions.

Data Collection – SpaceX API

- https://github.com/VAcodequest/Data-Science/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

1. Start
2. Initiate API Request

   Retrieve data using `requests.get()`.
3. Parse JSON Response

   Convert JSON data into a DataFrame using `pd.json_normalize()`.
4. Extract Data Attributes

   Define functions to extract key information (e.g., booster version, payload data).
5. Store Extracted Data

   Save the extracted attributes into a dictionary.
6. Create Final DataFrame

   Convert the dictionary into a Pandas DataFrame.
7. End

Data Collection Scraping

- https://github.com/VAcodequest/Data-Science/blob/main/jupyter-labs-webscraping.ipynb

```
1. Start
2. Send HTTP Request
   Use a library like `requests` to send a GET request to the target URL.
3. Receive Response
   The server returns the HTML content of the webpage.
4. Parse HTML Content
   Use a parser like `BeautifulSoup` to process the HTML data.
5. Extract Data
   Identify and extract the required information (e.g., text, tables, images).
6. Store Data
   Save the extracted data into a structured format like a CSV or
DataFrame.
7. End
```

Data Wrangling

- https://github.com/VAcodequest/Data-Science/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

1. Start
2. Load Data

   Import data from various sources (e.g., CSV, API).
3. Handle Missing Values

   Identify and address missing data (e.g., filling, dropping).
4. Data Transformation

   Convert data types, normalize, and scale features.
5. Feature Engineering

   Create new features or modify existing ones.
6. Filter Data

   Select relevant columns and filter rows as necessary.
7. Store Processed Data

   Save the cleaned and processed data into a new file.
8. End

EDA with Data Visualization

- https://github.com/VAcodequest/Data-Science/blob/main/edadataviz.ipynb

1. Histogram: This is used to understand the distribution of individual variables, particularly for continuous data like payload mass or flight duration.

2. Box Plot: This is employed to identify outliers and understand the spread and skewness of the data.

3. Scatter Plot: This is used to explore relationships between two continuous variables, such as payload mass versus flight success.

EDA with SQL

- https://github.com/VAcodequest/Data-Science/blob/main/jupyter_labs_eda_sql_coursera_sqllite.ipynb

Select Queries:

- This is used to retrieve specific columns and data from the database tables to analyze various metrics.

Join Queries:

- This is used to combine multiple tables based on related columns to create comprehensive datasets for analysis.

Aggregation Queries:

- I also made use of \`GROUP BY\`, \`COUNT\`, \`SUM\`, and \`AVG\` to summarize data, calculate totals, and find average values.

Build an Interactive Map with Folium

- https://github.com/VAcodequest/Data-Science/blob/main/lab_jupyter_launch_site_location.ipynb

Markers: I added markers to indicate specific locations, such as launch sites or areas of interest. This provides a clear visual reference.

- Explanation: Markers help users quickly identify and interact with key locations on the map, making it more engaging and informative.

Circles and Lines: I used circles to highlight regions around certain points and lines to connect related locations, representing flight paths or boundaries.

- Explanation: Circles emphasize the area of influence or impact around a location, while lines visually depict relationships or movements between points, enhancing the map's narrative.

Polygons: I added polygons to demarcate larger areas, such as city boundaries or zones of interest, to provide context and visual grouping.

- Explanation: The polygons offer a clear way to group related areas, helping users understand spatial relationships and the extent of different regions, thus providing better context for the data.

Build a Dashboard with Plotly Dash

Summary of Plots/Graphs and Interactions:

- Pie Chart: To visualize the proportion of successful and failed launches for selected launch sites.
- Range Slider and Scatter Plot: To filter payload mass ranges, which updates a scatter plot to show the correlation between payload mass and launch success.

Explanation of Why These Plots and Interactions Were Added:

- Pie Chart: The pie chart provides an intuitive way to compare the success and failure rates of launches across different sites, offering insights into site performance.
- Range Slider and Scatter Plot: The combination of a range slider and scatter plot enables users to explore how payload size impacts launch outcomes, with the ability to see results for specific payload ranges and site selections.

- https://github.com/VAcodequest/Data-Science/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

Summary of the Model Development Process

1. Model Building: I started with exploratory data analysis to understand the features influencing the success of SpaceX launches. Then I selected relevant features like payload mass, launch site, and orbit for model training. Furthermore, I implemented multiple classification models, including Logistic Regression, Support Vector Machines (SVM), Decision Trees, and K-Nearest Neighbors (KNN).

2. Model Evaluation: For model evaluation I used cross-validation to evaluate model performance, ensuring robustness and avoiding overfitting. Compared models based on their accuracy, with each model's performance logged for both training and test datasets.

https://github.com/VAcodequest/Data-Science/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

3. Model Improvement: I fine-tuned hyperparameters using grid search for each model to optimize performance. Furthermore, I explored different feature scaling techniques and adjusted the model parameters accordingly.

4. Best Model Identification: After evaluation, K-Nearest Neighbors (KNN) was identified as the best-performing model with a consistent accuracy of 83.33% on the test dataset. Also, the final model was further validated to confirm its performance and reliability.

5. Visualization: The entire process was mapped out in a flowchart, highlighting key steps from data preprocessing to model deployment. Visualizations such as confusion matrices and ROC curves were used to interpret model results and guide decision-making.

1. Start
   - Begin the model development process.
2. Exploratory Data Analysis
   - Analyze the data to understand key features and their relationships.
3. Feature Selection
   - Identify the most relevant features for the classification models.
4. Model Implementation
   - Implement various classification models:
     - Logistic Regression
     - Support Vector Machines (SVM)
     - Decision Tree
     - K-Nearest Neighbors (KNN)
5. Cross-Validation and Evaluation
   - Evaluate the performance of each model using cross-validation.
6. Hyperparameter Tuning
   - Optimize each model by tuning hyperparameters to improve accuracy.
7. Model Comparison
   - Compare the performance of all models based on accuracy and other metrics.
8. Best Model Identification - KNN
   - Select K-Nearest Neighbors as the best-performing model.
9. Final Validation and Visualization
   - Validate the selected model and visualize the results with relevant charts and metrics.
10. End
   - Conclude the model development process.

Results

1. Predictive Analysis Results

Logistic Regression:

- Tuned hyperparameters: `{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}`
- Accuracy: `0.8464`
- Test Accuracy: `0.8333`

Support Vector Machine (SVM):

- Tuned hyperparameters: `{'C': 1.0, 'gamma': 0.0316, 'kernel': 'sigmoid'}`
- Accuracy: `0.8482`
- Test Accuracy: `0.8333`

Decision Tree:

- Tuned hyperparameters: `{'criterion': 'gini', 'max_depth': 16, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'best'}`
- Accuracy: `0.8732`
- Test Accuracy: `0.8333`

K-Nearest Neighbors (KNN):

- Tuned hyperparameters: `{'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}`
- Accuracy: `0.8482`
- Test Accuracy: `0.8333`

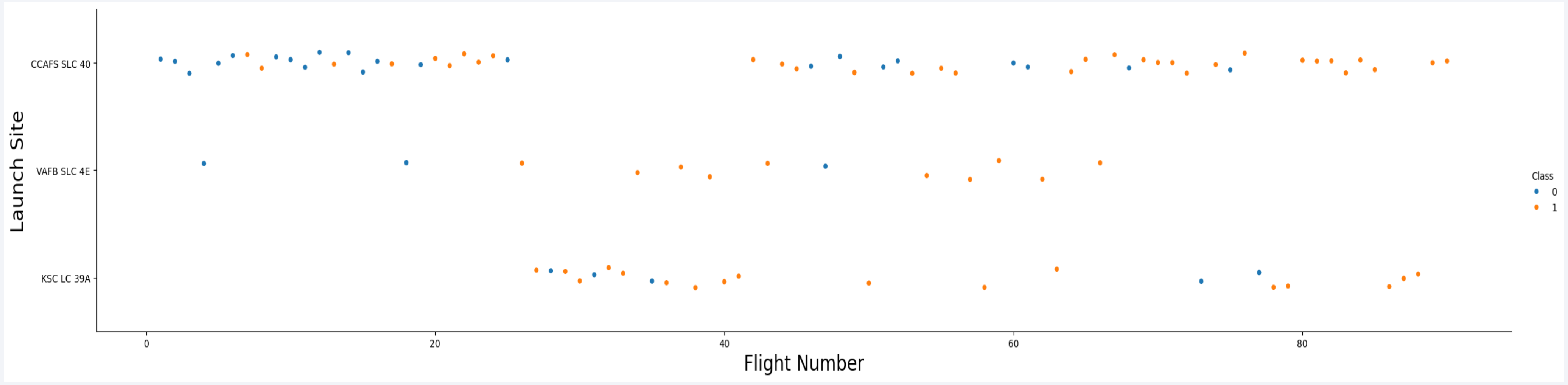
Best Performing Model: K-Nearest Neighbors with an accuracy of `0.8333`.

Section 2

Insights drawn from EDA

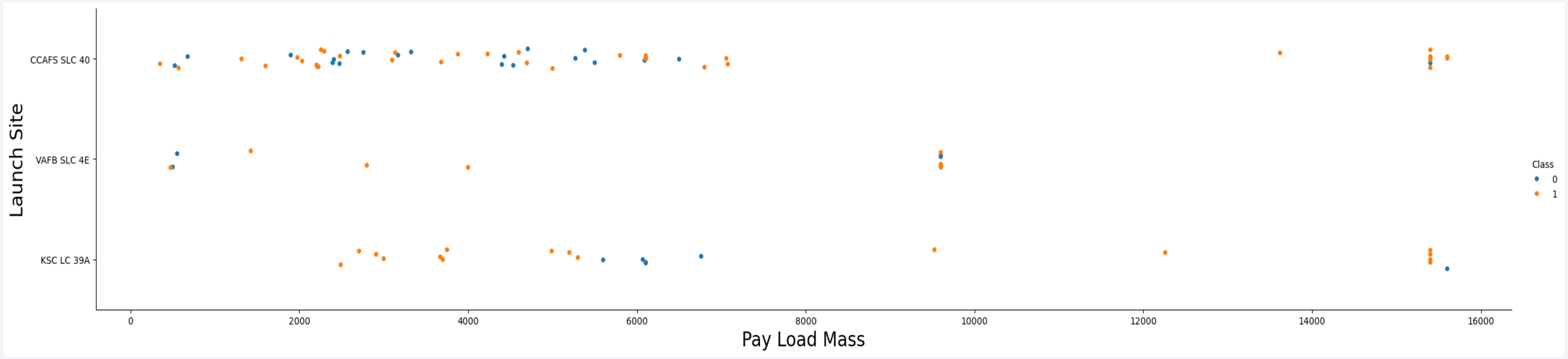
Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site

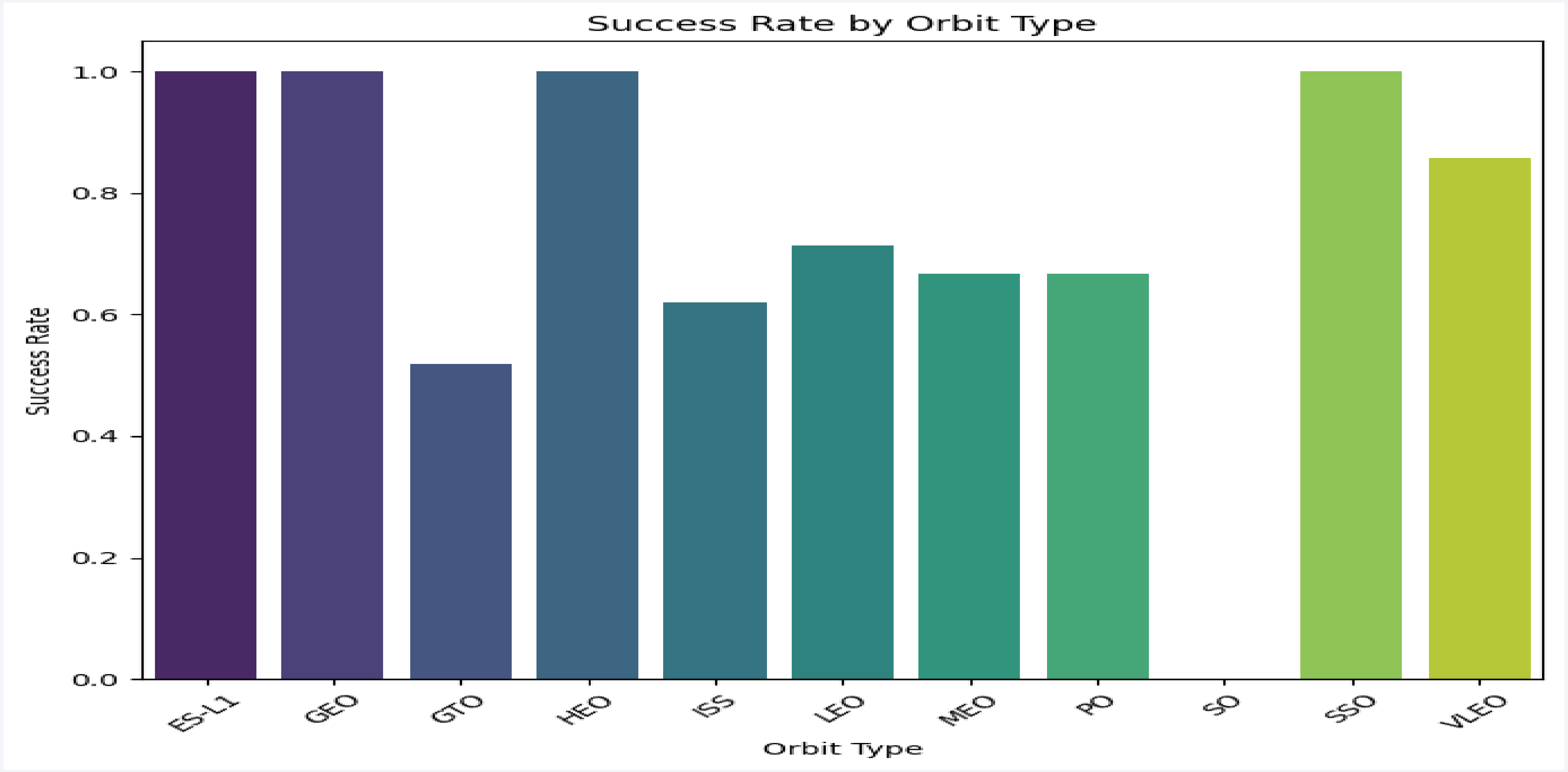


Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site

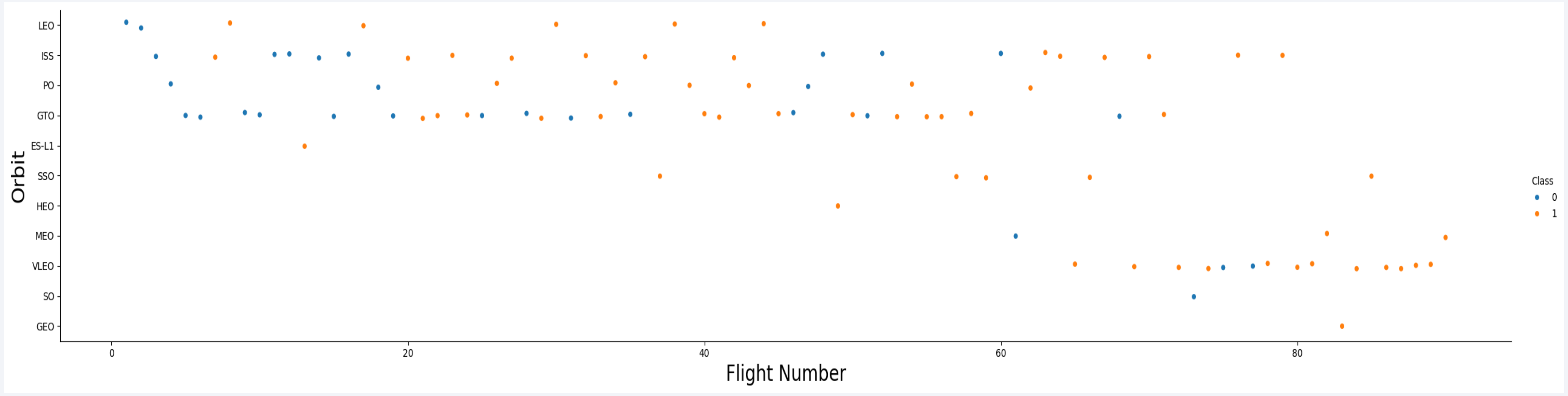


Success Rate vs. Orbit Type



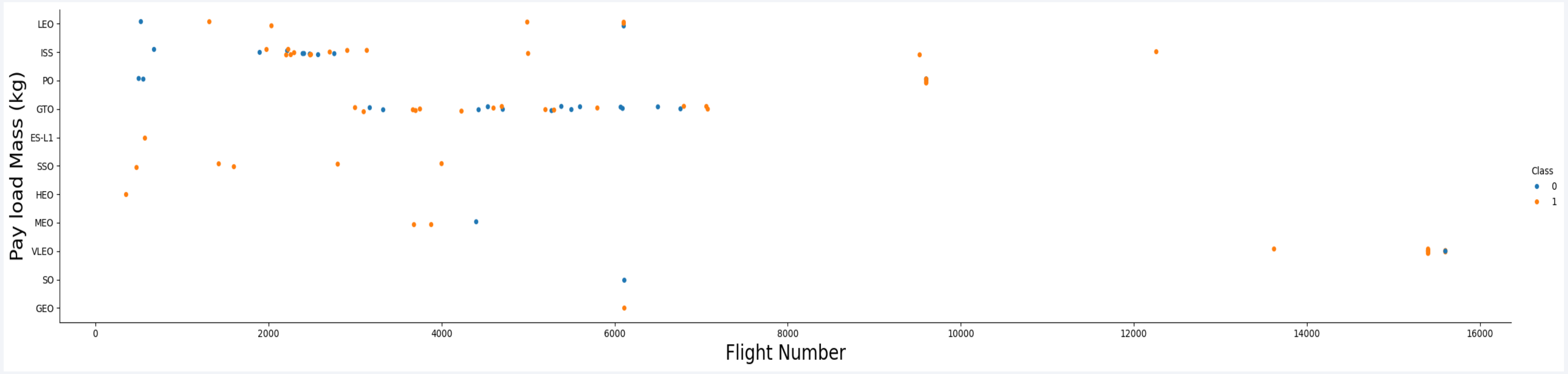
Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type

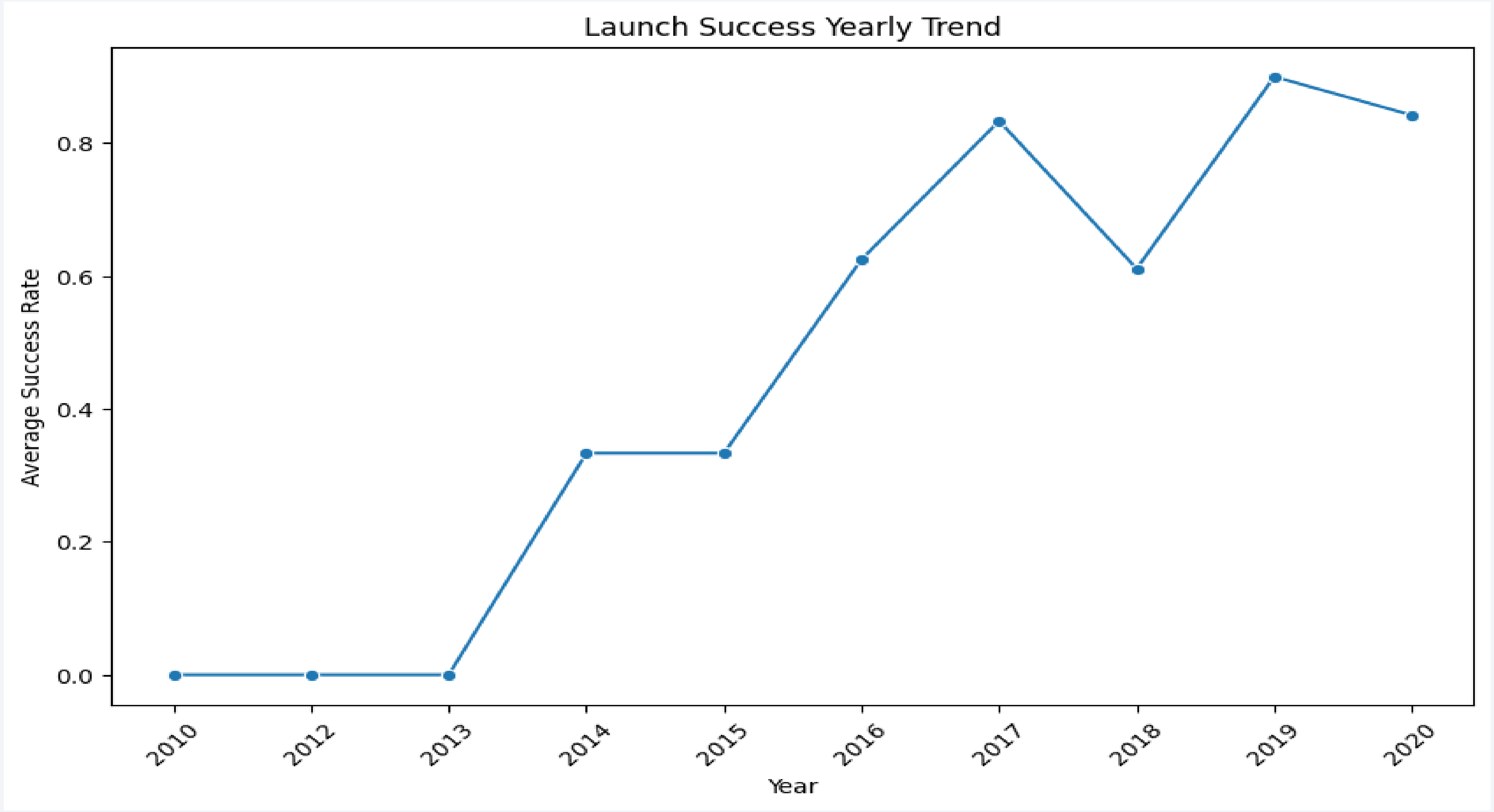


Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type



Launch Success Yearly Trend



All Launch Site Names

- Find the names of the unique launch sites

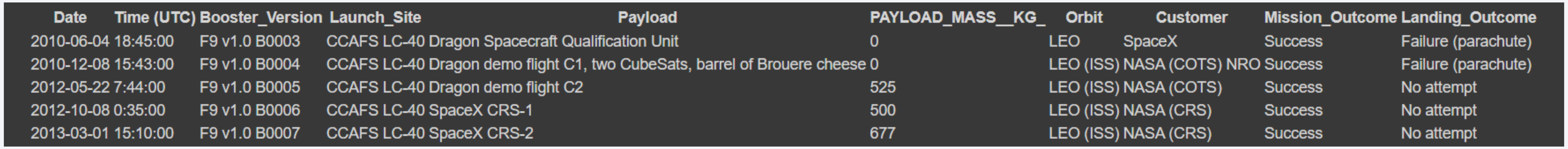
```
Unique Launch Site Names: ['CCAFS SLC 40' 'VAFB SLC 4E' 'KSC LC 39A’]
```

This code will:

- Extract the unique values from the 'Launch Site' column of the DataFrame spacex_df.
- Print these unique values, which represent the different launch sites used.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

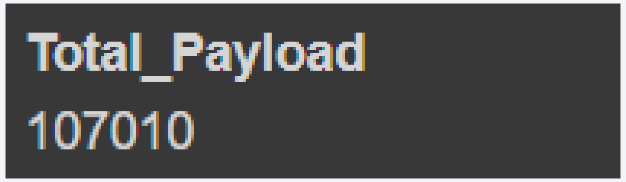


Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- This query fetches the 5 records where launch sites begin with CCA

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

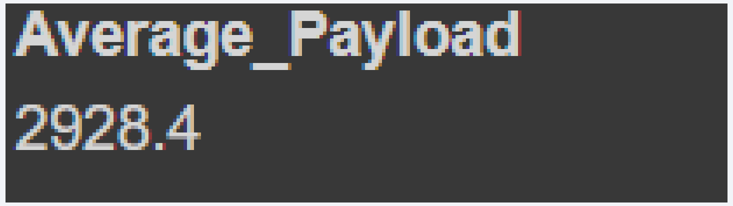


Total_Payload
107010

- For this, I made use of the aggregation concept and calculated the sum

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

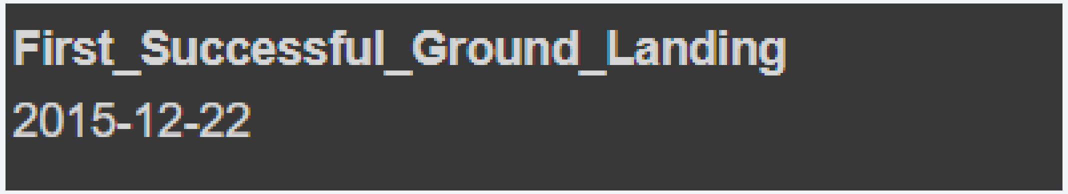


```
Average_Payload
2928.4
```

- For this, I made use of the aggregation concept, using average

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

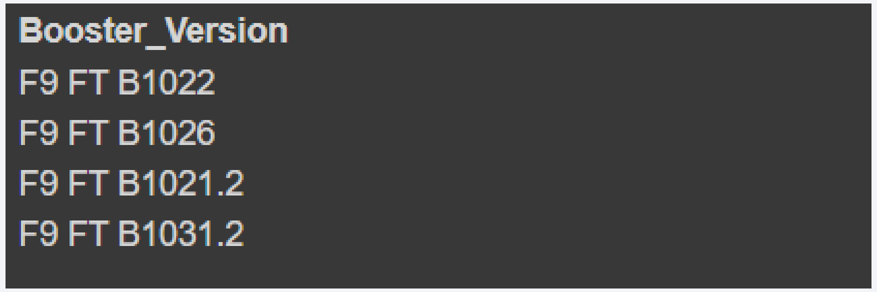


```
First_Successful_Ground_Landing
2015-12-22
```

- The code for this fetches the date that has the first successful landing

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

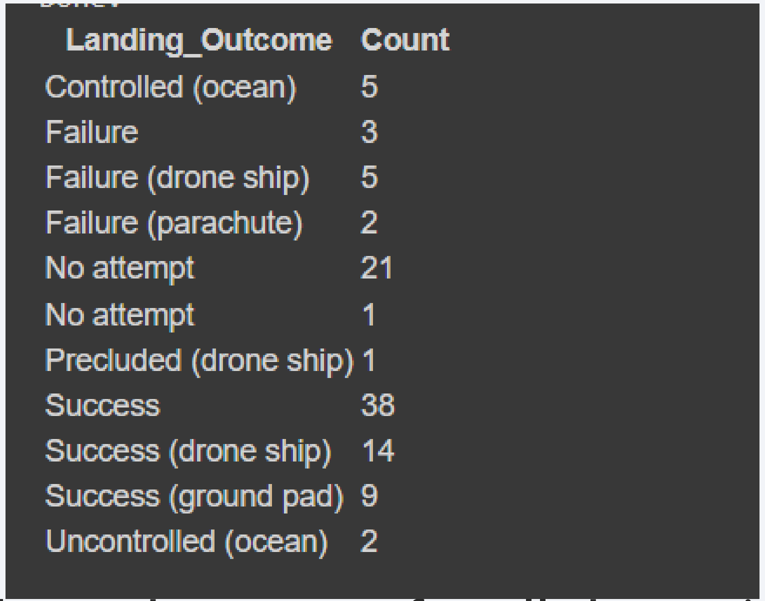


Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

- The code for this makes use of the concept of fetching specific records using a range

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

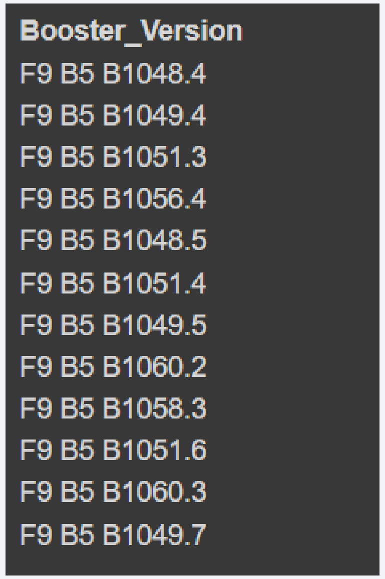


Landing_Outcome	Count
Controlled (ocean)	5
Failure	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	21
No attempt	1
Precluded (drone ship)	1
Success	38
Success (drone ship)	14
Success (ground pad)	9
Uncontrolled (ocean)	2

- This code calculates the counts for all the variable values that can either be categorized under successful or failed

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

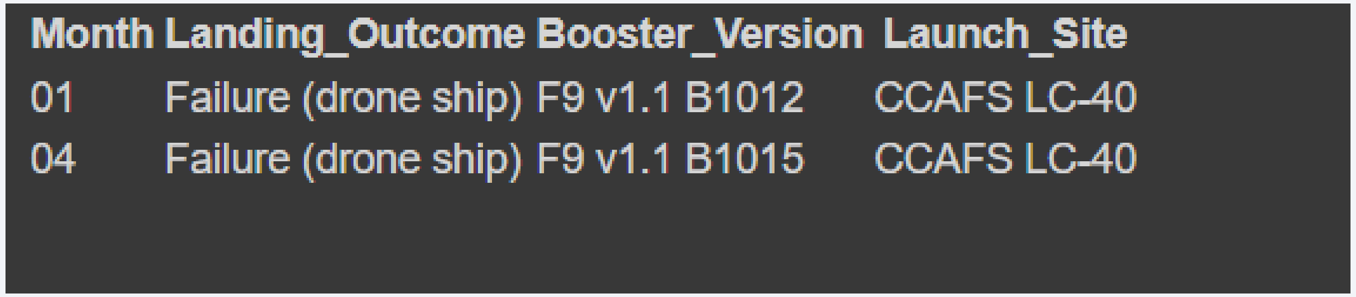


Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

- The code fetches all the unique Booster versions that have carried the maximum load.

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

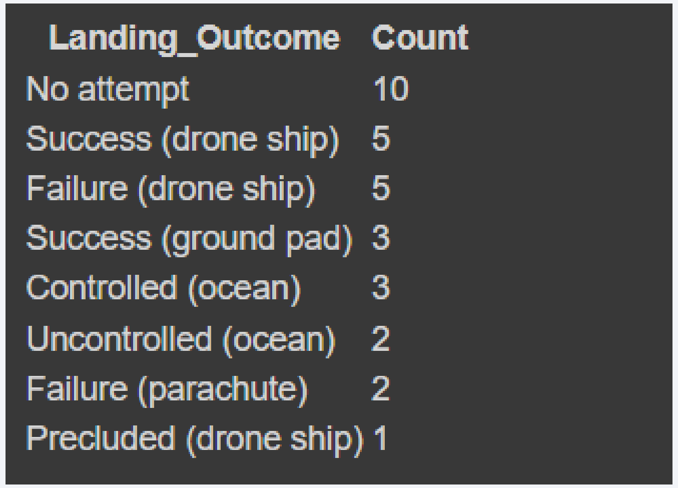


Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- The code works fine

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order



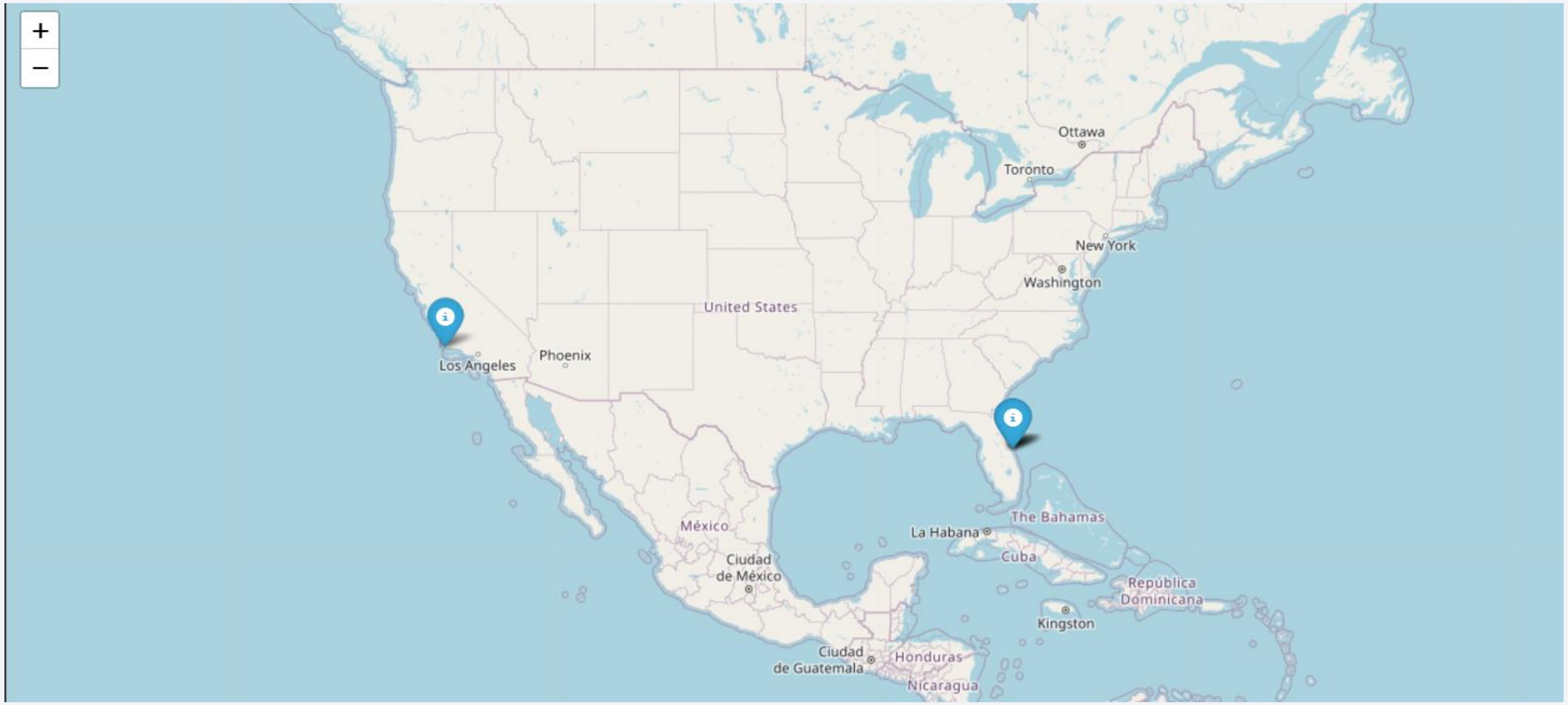
Landing_Outcome	Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

- The code calculates the respective counts

Section 3

Launch Sites Proximities Analysis

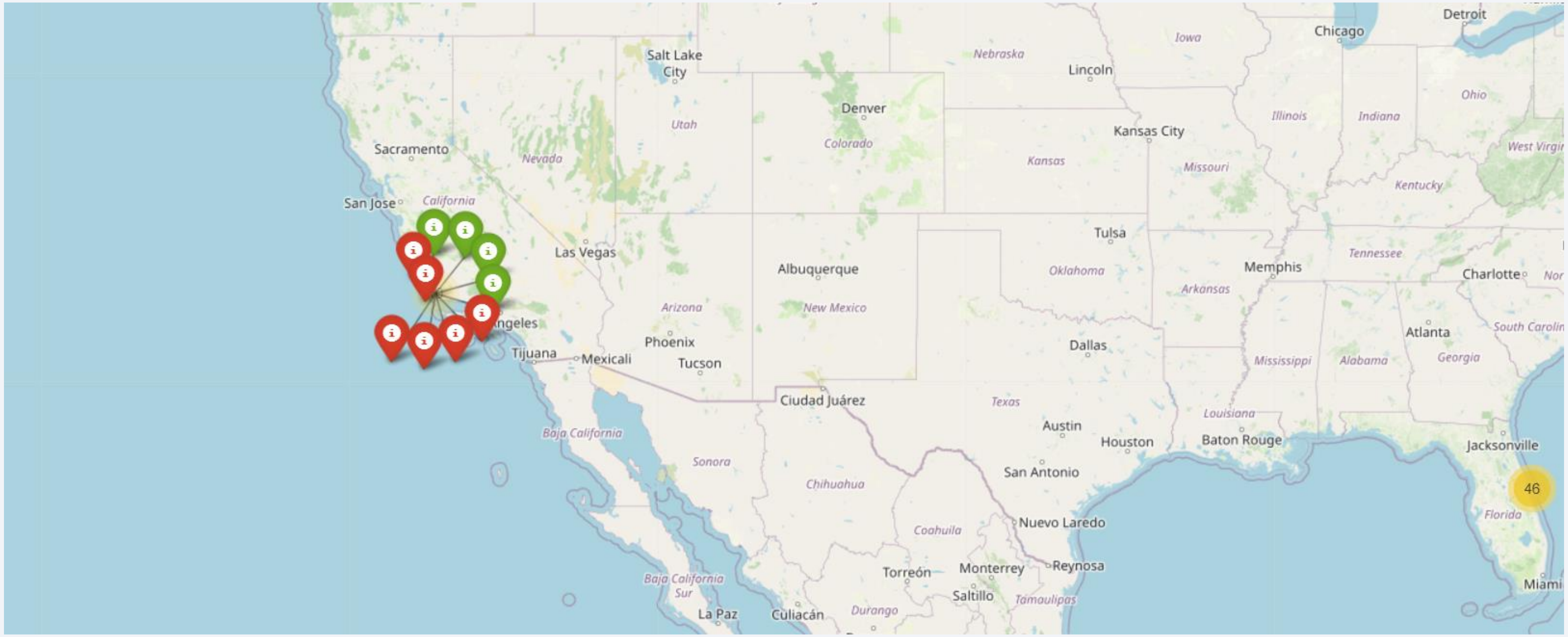
Folium Map of Launch Sites in the USA



Key Elements:

- Locations: This shows the geographical locations of SpaceX launch sites in the United States.
- Markers: These markers represents a launch site, providing a quick visual reference for their locations.
- Interactivity: This map is interactive, allowing users to zoom and explore the surrounding areas of each launch site.

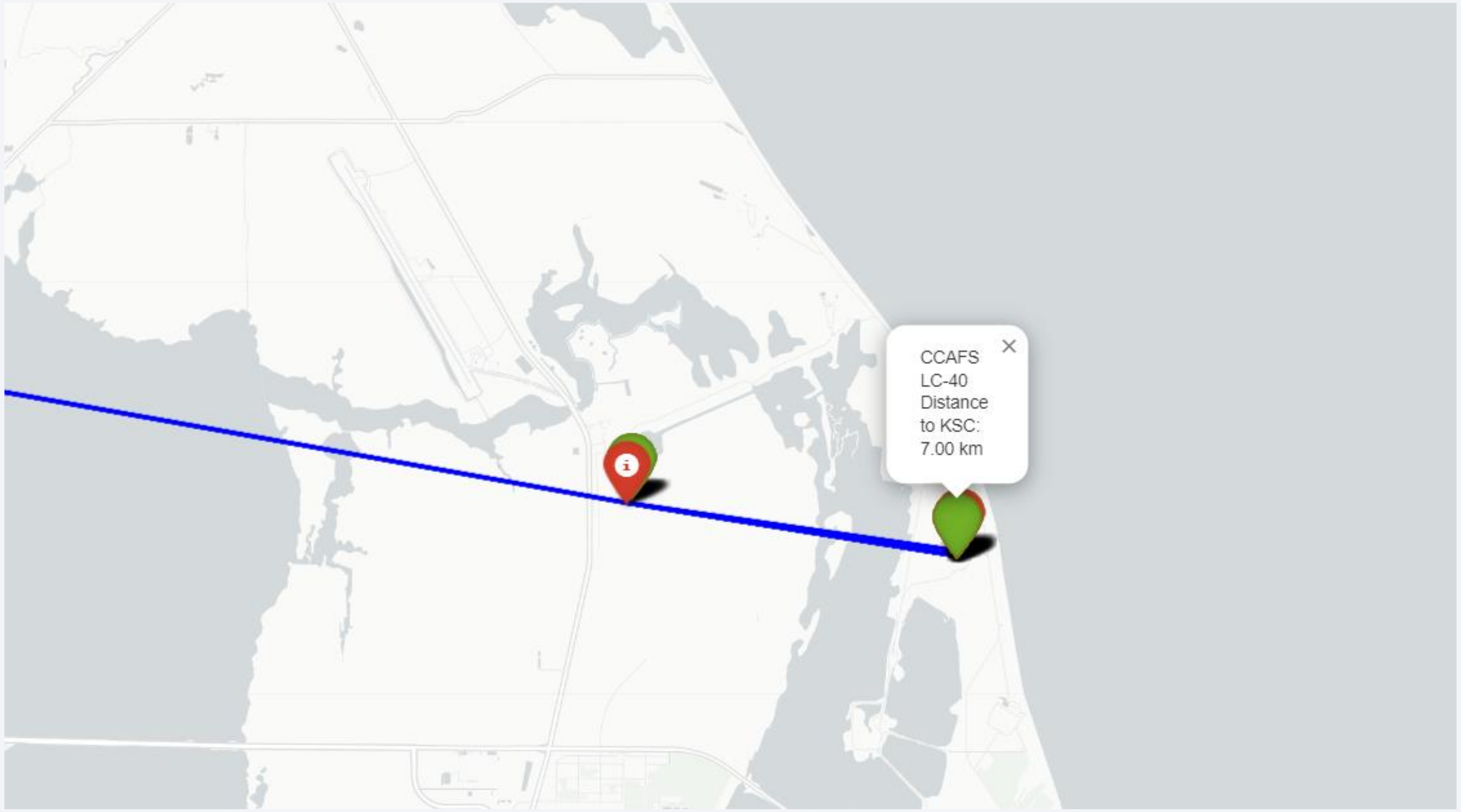
Concentration of Launches in California



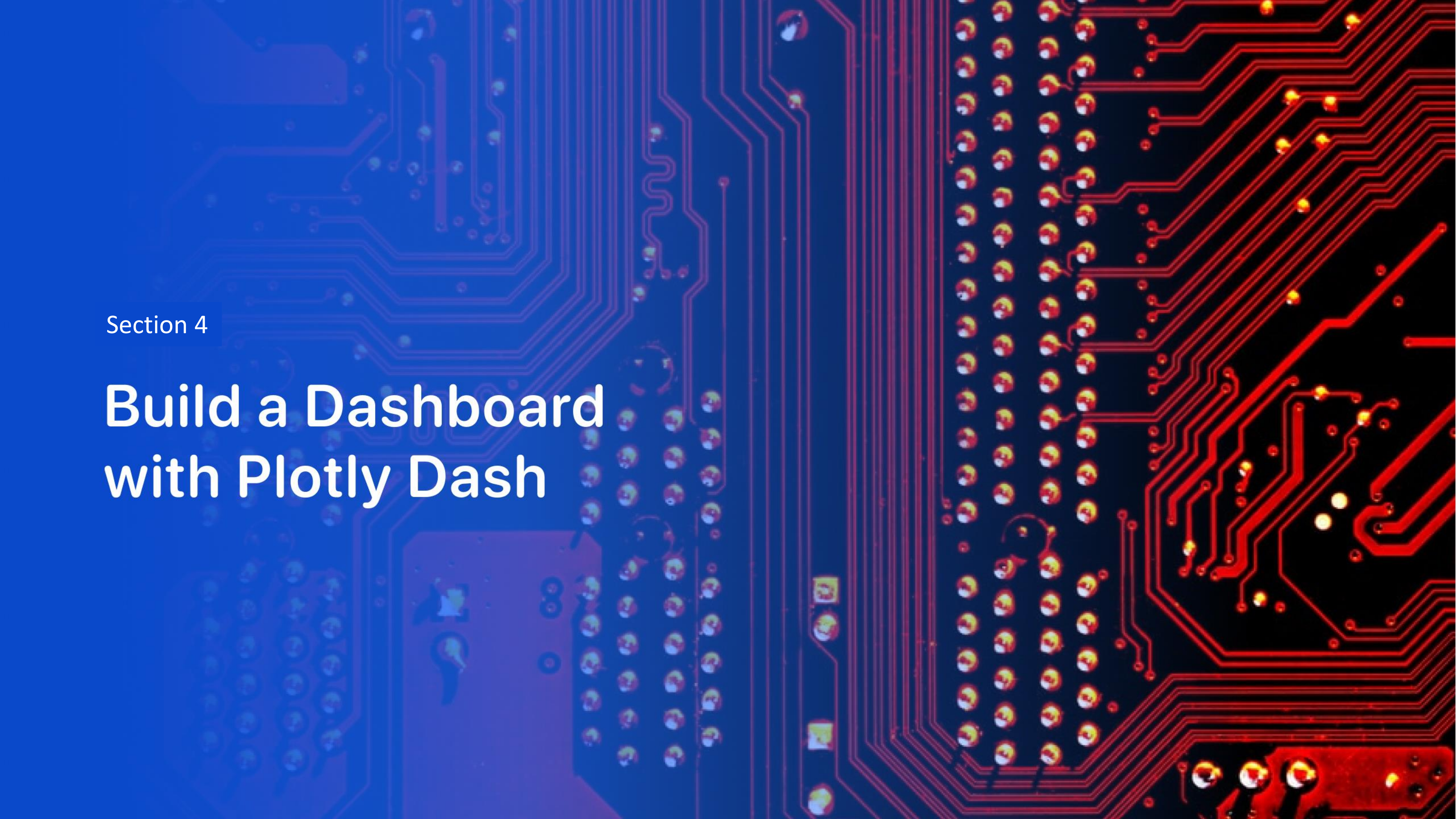
Key Elements:

- Clustered Markers: These indicate a high concentration of launch activities in California, specifically around the Los Angeles area.
- Color-Coded: These markers are color-coded to distinguish between different types of launches or outcomes.
- Data Density: These highlights the significance of California as a primary launch region for SpaceX.

Distance Measurement Between Launch Site



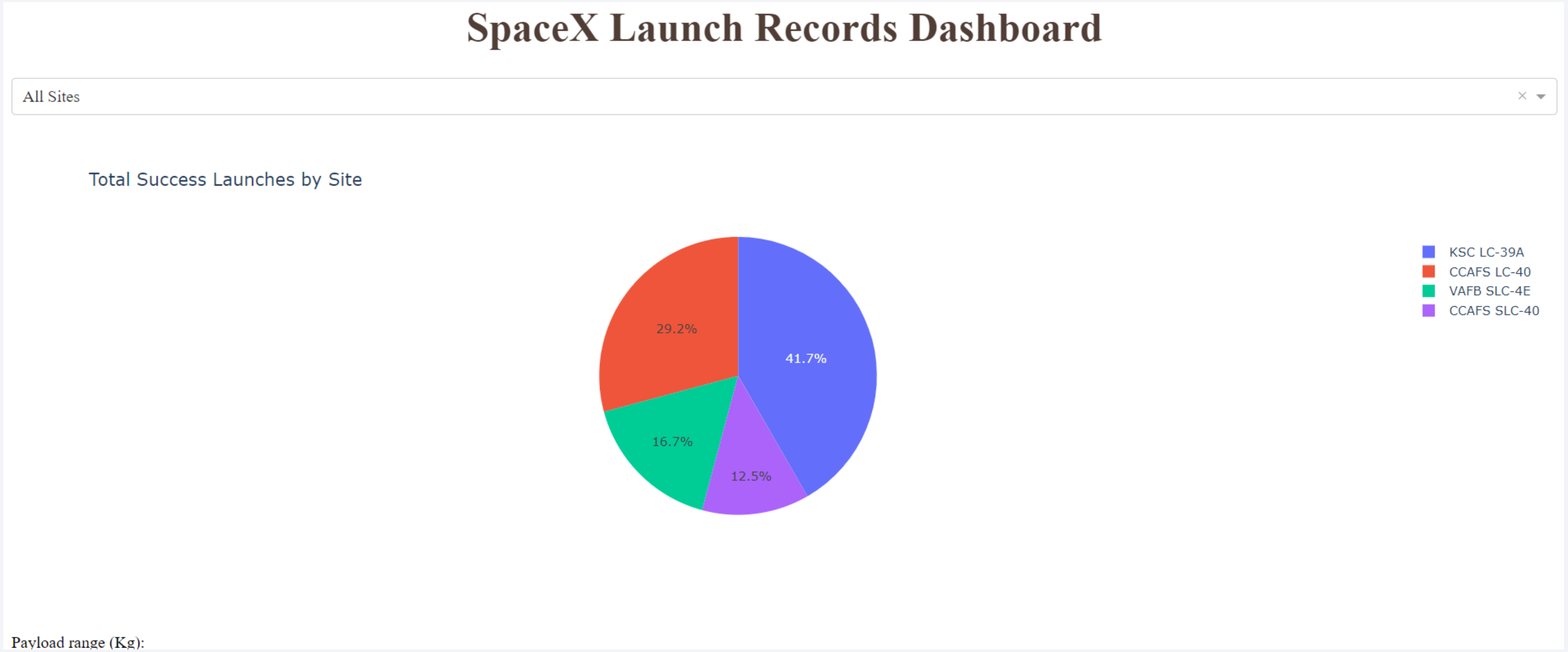
- **Key Elements**:
  - Polyline: The polyline connects two specific points, measuring the distance between the CCAFS LC-40 site and another key location.
  - Distance Display: The distance is clearly marked on this map, showing a 7.00 km separation.
  - Map Detail: This map zooms in on a specific area to give a detailed view of the launch sites and their relative positions.



Section 4

Build a Dashboard with Plotly Dash

Total Success Launches by Site


SpaceX Launch Records Dashboard

All Sites

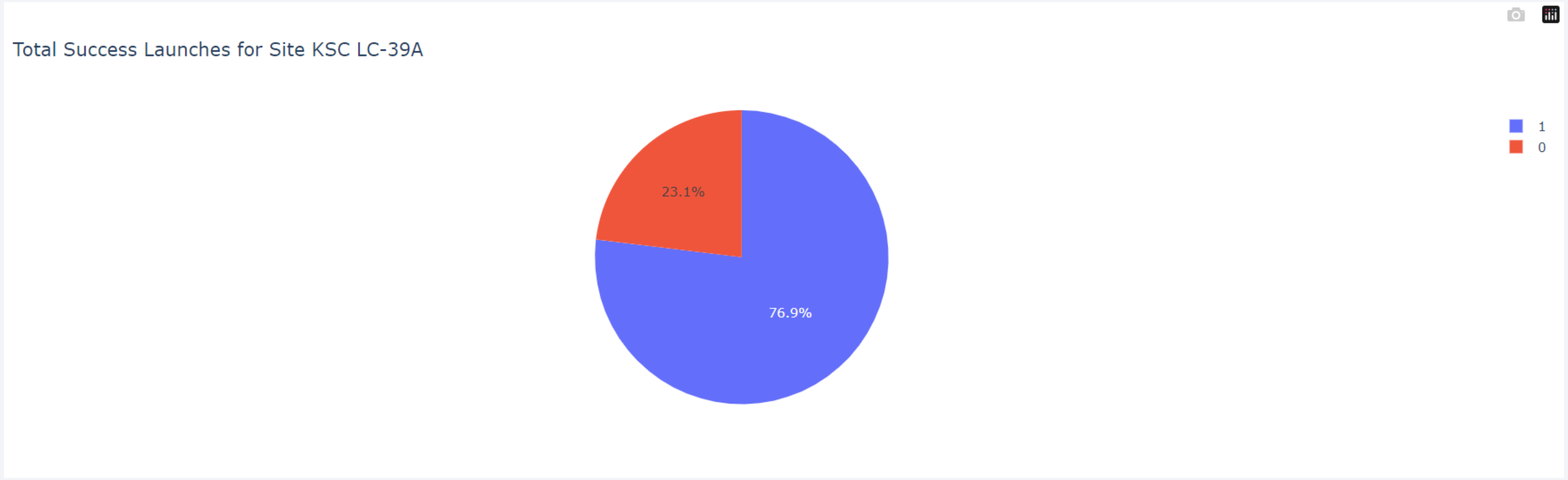
Total Success Launches by Site

Payload range (Kg):

Key Elements:

- Pie Chart: This chart displays the distribution of successful launches across various SpaceX launch sites.
- Site Comparison: KSC LC-39A has the highest percentage of successful launches at 41.7%, followed by CCAFS LC-40 with 29.2%.
- Interactive Dashboard: This dashboard allows filtering by launch site, providing insights into performance across different locations.

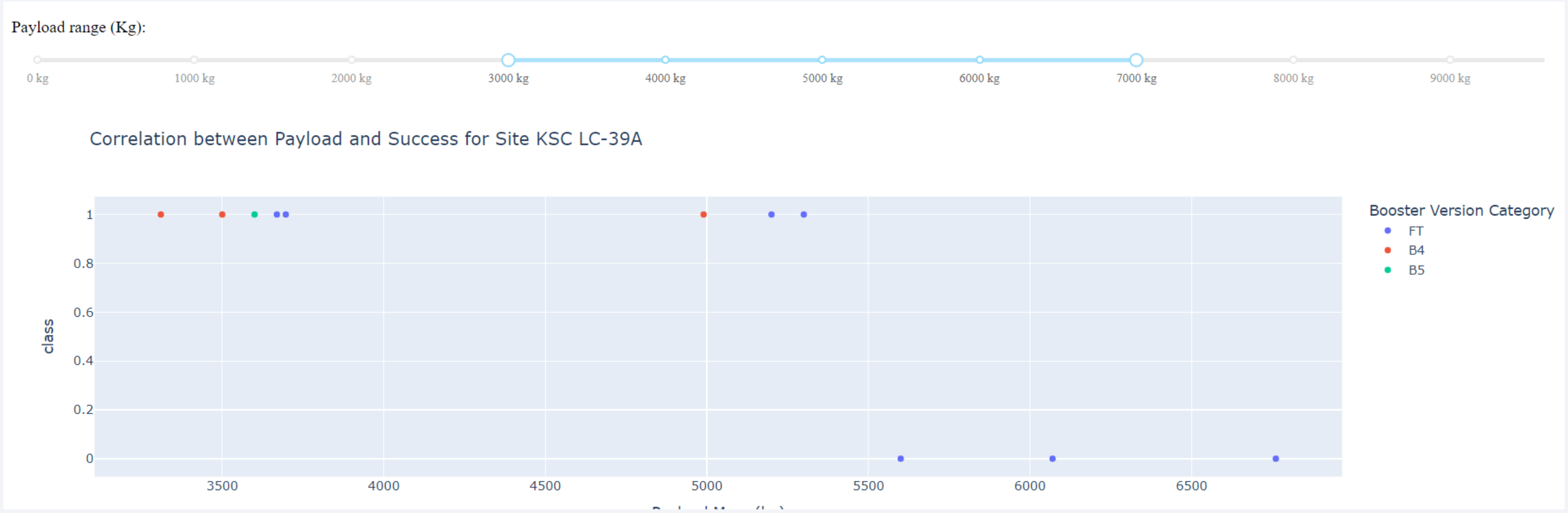
Total Success Launches for KSC LC-39A



Key Elements:

- Success Rate: KSC LC-39A shows a high success rate of 76.9% for launches, with the remaining 23.1% representing failures.
- Focused Analysis: This specific breakdown helps identify the reliability of KSC LC-39A as a launch site.
- Visual Impact: The clear visual differentiation between successful (blue) and unsuccessful (red) launches provides immediate insights.

Correlation Between Payload and Success for KSC LC-39A



Key Elements:

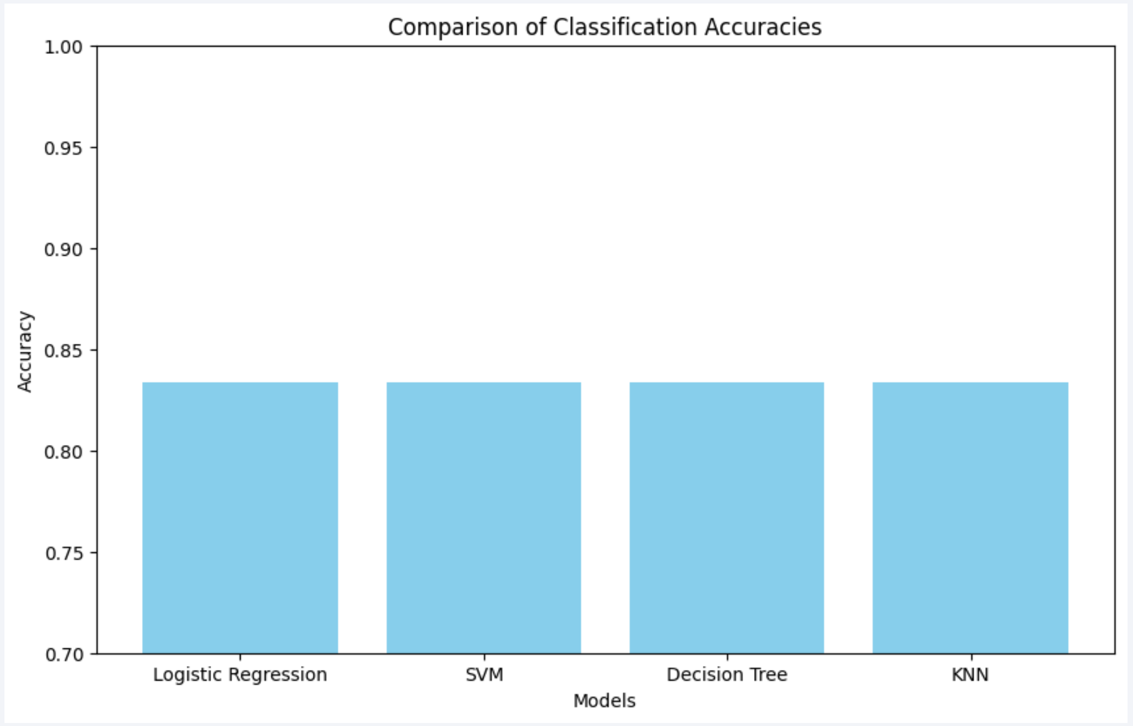
- Scatter Plot: This plot shows the relationship between payload mass and launch success, focusing on the KSC LC-39A site.
- Payload Range Slider: This slider allows users to filter the data by payload mass, making it easier to analyze trends.
- Booster Version Impact: All the different booster versions (FT, B4, B5) are color-coded, helping identify which versions contribute to success or failure.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

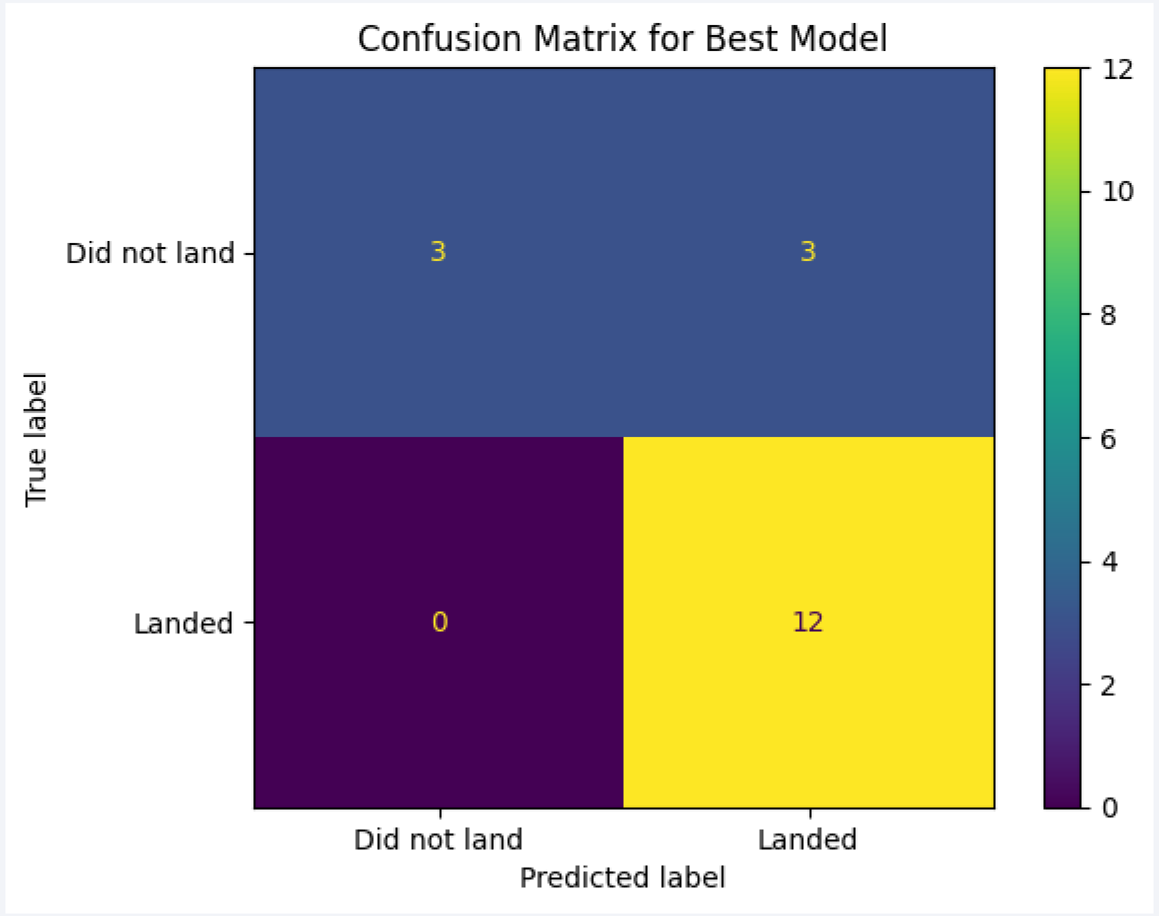
- Visualize the built model accuracy for all built classification models, in a bar chart



- Find which model has the highest classification accuracy KNN

Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



Conclusions

1. Model Performance: All tested models (Logistic Regression, SVM, Decision Tree, and KNN) achieved similar test accuracies, around 83.33%, indicating comparable performance across these algorithms.

2. Best Model: The K-Nearest Neighbors (KNN) model, with the highest validation accuracy of 84.82%, was identified as the best-performing model for this task.

3. Consistency: Despite variations in the best parameters for each model, the test accuracy remained consistent across models, suggesting that the dataset may not significantly favor one algorithm over others.

4. Hyperparameter Tuning: Proper tuning of hyperparameters for each model led to minor differences in accuracy, emphasizing the importance of this step in optimizing model performance.

5. Robustness: The consistent test accuracy across different models suggests that the dataset and feature selection provided a robust foundation for predicting outcomes.

Appendix

- Data wrangling, visualization, web scraping, and API interaction are key focus areas.
- Code snippets include data manipulation, SQL queries, and API data handling.
- Visualizations created using `matplotlib` and `seaborn` illustrate SpaceX launch trends.
- SQL queries manage and retrieve data from an SQLite database.
- Outputs validate the data processing, showing clean data and successful API retrievals.

Thank you!