IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Roman Shrestha

15 July 2025

Outline

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

Executive Summary

- **Data Collection:** Data was collected using the SpaceX REST API and web scraping from Wikipedia pages.
- **Data Wrangling:** This involved merging datasets, handling missing values, and correcting data types.
- **EDA (Exploratory Data Analysis):** Charts like scatter plots and bar charts were used, along with SQL queries, to gain insights.
- **Interactive Analysis:** Folium maps and Plotly Dash were utilized for interactive visualizations.
- **Predictive Analysis:** Classification models such as Logistic Regression, SVM, and Decision Trees were built and evaluated for accuracy and confusion matrix

Introduction

Project background and context

- SpaceX, a private aerospace company founded by Elon Musk, is pioneering the development of reusable rocket technology to significantly reduce the cost of space travel. One of their primary goals is to recover and reuse the first-stage boosters of their rockets after launch. Reusability depends heavily on the success of landing outcomes, which are influenced by various factors such as payload mass, launch site, orbit type, and booster version.

- This capstone project analyzes historical SpaceX launch records using data science techniques. By performing data wrangling, exploratory data analysis (EDA), and predictive modeling, we aim to uncover insights into the conditions that lead to successful landings. This analysis helps support strategic decision-making and performance improvements for future SpaceX missions.

Problems i want to find answers

- **Which launch sites are most successful?**
  Analyze launch success rates across different launch sites to determine which ones are most reliable and consistent.

- **Does payload affect landing outcome?**
  Investigate the relationship between the payload mass and landing success to understand if heavier payloads reduce the chance of a successful landing.

- **Can we predict successful landings?**
  Build and evaluate machine learning classification models to predict landing outcomes based on features such as payload mass, orbit type, launch site, and booster version.

Section 1

Methodology

Methodology

Executive Summary

- **Data Collection**:
  - REST API: Use SpaceX API (add a flowchart of API endpoints and responses)
  - Web scraping: Describe scraping Wikipedia pages (e.g., launch table)
  - Add GitHub links to my notebooks
- **Data Wrangling**:
  - Merging datasets
  - Handling missing values, data types
  - Add flowchart + GitHub notebook link
- **EDA (Exploratory Data Analysis)**:
  - Charts: scatter plots, bar charts
  - SQL queries (show insights like payload mass, success rate by orbit)
  - Add GitHub links for both visual + SQL analysis
- **Interactive Analysis**:
  - **Folium map**: Describe markers and overlays (success/failure by color)
  - **Plotly Dash**: Show pie chart for launch success, payload sliders, etc.
  - Add GitHub links
- **Predictive Analysis**:
  - Logistic regression, SVM, Decision Trees, etc.
  - Accuracy, confusion matrix
  - Describe model building + tuning process
  - Add GitHub notebook

Data Collection

- **Sources**
  - **SpaceX REST API**
    - Retrieved structured data (launch site, payload, booster, orbit, outcome) using Python's requests library.
    - Converted JSON to DataFrame and saved as CSV.
  - **Wikipedia Web Scraping**
    - Used BeautifulSoup to extract launch history and landing outcomes.
    - Parsed and structured HTML table data for merging.

- **Process Flow**
- [SpaceX API] + [Wikipedia Scraping]

  ↓

- [Clean & Convert to CSV]

  ↓

- [Merged Final Dataset]

Data Collection – SpaceX API

Key Process Summary

- Accessed SpaceX API at https://api.spacexdata.com/v4/launches
- Used Python requests and json_normalize to extract data
- Collected fields: flight_number, launch_site, payload_mass_kg, booster_version, orbit, landing_outcome
- Converted JSON to Pandas DataFrame
- Saved as spacex_api_data.csv for further processing

GitHub Notebook (Code + Output)

[SpaceX API Data Collection – GitHub Notebook](#)
(Includes completed code cells and output results for peer review)

Process Flowchart

[Start]
↓
[Send GET request to SpaceX API]
↓
[Parse JSON response]
↓
[Normalize & Convert to DataFrame]
↓
[Select relevant columns]
↓
[Save as CSV]
↓
[End]

Data Collection - Scraping

Web Scraping Workflow

- **Identify the target URL**
  → Locate the webpage that contains the data you want to scrape.
- **Inspect the HTML structure**
  → Use browser developer tools to find the tags, classes, or IDs of interest.
- **Send an HTTP request**
  → Use requests to retrieve the webpage content.
- **Parse HTML content**
  → Use BeautifulSoup or lxml to extract specific elements.
- **Extract data elements**
  → Use tag names and attributes to collect required data.
- **Store extracted data**
  → Save the structured data to a DataFrame, .csv, .json, etc.
- **Clean and format data**
  → Use pandas to tidy and prepare the data for analysis.
- **Validate and test**
  → Ensure accuracy and handle exceptions (e.g., missing data, broken HTML).

GitHub Notebook for Review

Repository: IBM Data Science Capstone Project
Notebook: Web Scraping with BeautifulSoup & Pandas
GitHub URL: https://github.com/sh184roman/IBM-Data-Science-Capstone-Project/blob/main/jupyter-labs-webscraping.ipynb

```
flowchart TD

    A [Start] --> B[Identify Target Website]
    B --> C [Inspect HTML Structure]
    C --> D [Send HTTP GET Request (requests)]
    D --> E [Parse HTML (BeautifulSoup)]
    E --> F [Locate Tags / Classes]
    F --> G [Extract and Store Data (pandas)]
    G --> H [Clean & Structure Data]
    H --> I [Save to CSV / JSON]
    I --> J [End]
```

Data Wrangling

Data Wrangling Workflow

- **Data Importing**
  → Read data from multiple sources: CSV files and web pages using pandas.
- **Initial Data Inspection**
  → Use .head(), .info(), and .describe() to understand data structure, types, and missing values.
- **Web Scraping Integration**
  → Scraped launch site data from Wikipedia using BeautifulSoup and integrated with the dataset.
- **Handling Missing Values**
  → Replaced NaN values, dropped irrelevant rows, and filled missing columns based on context.
- **Feature Extraction**
  → Parsed columns like LaunchSite, PayloadMass, and Orbit to derive meaningful insights.
- **Data Merging**
  → Merged scraped data with existing CSV files (launch records) based on unique identifiers.
- **Data Type Conversion**
  → Converted object types to correct data types (e.g., datetime, float, int).
- **Column Renaming & Filtering**
  → Renamed ambiguous columns and filtered out unnecessary data to reduce noise.
- **Final Dataset Export**
  → Saved the cleaned and processed dataset as a new CSV file for analysis.

GitHub Notebook for Review

Repository: IBM Data Science Capstone Project
Notebook: Data Wrangling for SpaceX Falcon 9 Launch Data
GitHub URL:https://github.com/sh184roman/IBM-Data-Science-Capstone-Project/blob/main/labs-jupyter-spacex-Data%20wrangling-v2.ipynb

EDA with Data Visualization

Summary of Visualizations and Their Purpose

- **Bar Charts**
  - **What**: Count of successful launches per site
  - **Why**: To identify which launch sites have the highest number of successful missions.
- **Pie Chart**
  - **What**: Success rate per launch site
  - **Why**: To show the proportion of successes across sites in a single view.
- **Scatter Plot**
  - **What**: Payload mass vs. launch success
  - **Why**: To examine if heavier payloads affect launch success.
- **Box Plot**
  - **What**: Distribution of payload mass by success/failure and by booster version category
  - **Why**: To understand payload variability and its effect on outcomes.
- **Line Plot**
  - **What**: Number of launches over time
  - **Why**: To observe trends in launch frequency and performance over the years.
- **Histogram**
  - **What**: Distribution of payload mass
  - **Why**: To identify payload range and frequency distribution.
- **Correlation Heatmap**
  - **What**: Correlation between numerical features
  - **Why**: To identify which features may influence launch outcome.

Chart Libraries Used

- matplotlib
- seaborn
- plotly.express
- These libraries were chosen for their interactivity, aesthetic appeal, and ease of creating advanced visualizations.

GitHub Notebook for Review

Repository: IBM Data Science Capstone Project
Notebook: EDA & Data Visualization on SpaceX Launch Data
GitHub URL:https://github.com/sh184roman/IBM-Data-Science-Capstone-Project/blob/main/jupyter-labs-eda-dataviz-v2.ipynb

EDA with SQL

EDA with SQL – Summary of Tasks and Queries

Database Setup & Configuration

- Installed and configured required packages: sqlalchemy, ipython-sql, prettytable.
- Connected to a local SQLite database.
- Loaded the **SpaceX CSV dataset** into a SQLite table (SPACEXTBL).
- Removed empty rows and created a clean table SPACEXTABLE.

Tasks and SQL Queries Performed

✅ **Task 1**

- **Query:** Get unique launch sites.
  Purpose: Identify all different SpaceX launch locations.
- sql
- CopyEdit
- SELECT DISTINCT(Launch_Site) FROM SPACEXTBL;

✅ **Task 2**

- **Query:** Display 5 records where launch site begins with 'CCA'.
  Purpose: Filter launch sites using string patterns.
- sql
- CopyEdit
- SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5;

✅ **Task 3**

- **Query:** Total payload mass carried by **NASA (CRS)**.
  Purpose: Analyze payload carried for a specific customer.
- sql
- CopyEdit
- SELECT CUSTOMER, SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';

- **Task 4**

- **Query:** Average payload mass by **F9 v1.1** booster version.
  Purpose: Study booster version performance.
- sql
- CopyEdit
- SELECT Booster_Version, SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.1%';

✅ **Task 5**

- **Query:** Date of first **successful landing on ground pad**.
  Purpose: Track milestone in landing history.
- sql
- CopyEdit
- SELECT MIN(DATE) FROM SPACEXTBL;

✅ **Task 6**

- **Query:** Booster versions with **Success (drone ship)** and payload between 4000–6000 kg.
  Purpose: Explore launch success under specific payload constraints.
- sql
- CopyEdit
- SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;

✅ **Task 7**

- **Query:** Count total **successful vs. failed mission outcomes**.
  Purpose: Overall mission performance metrics.
- sql
- CopyEdit
- SELECT Mission_Outcome, COUNT(*) AS TOTAL FROM SPACEXTBL GROUP BY Mission_Outcome;

✅ **Task 8**

- **Query:** Booster versions that carried the **maximum payload mass**.
  Purpose: Identify high-capacity launches.
- sql
- CopyEdit
- SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);

✅ **Task 9**

- **Query:** Records of **failed drone ship landings in 2015**, with launch month, booster, site.
  Purpose: Analyze pattern of failures over time.
- sql
- CopyEdit
- SELECT substr("Date", 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(DATE, 0, 5) = '2015';

✅ **Task 10**

- **Query: Rank landing outcomes** between specific dates by frequency.
  Purpose: Discover dominant landing results in early years.
- sql
- CopyEdit
- SELECT Mission_Outcome, Landing_Outcome, COUNT(*) AS TOTAL_COUNT FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY TOTAL_COUNT DESC;

GitHub Notebook for Review

Repository: IBM Data Science Capstone Project
Notebook: SQL Analysis with SQLite
GitHub URL:https://github.com/sh184roman/IBM-Data-Science-Capstone-Project/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

Map Objects Added & Their Purpose

1. Markers (folium.Marker)

- **What**: Added standard markers at each launch site.
- **Why**: To indicate the **exact coordinates** of each SpaceX launch site.
- **Enhancement**: Included **popup labels** (e.g., "CCAFS LC-40") to provide site names on click.

2. Circle Markers (folium.Circle)

- **What**: Added circular overlays around launch sites.
- **Why**: To visually emphasize the **area around launch sites**, often indicating an activity zone or buffer radius.

3. CircleMarker with Radius Scaling

- **What**: Circle markers with a **custom radius**.
- **Why**: To scale visual emphasis—larger radius could indicate more launches or significance of a site.

4. Lines (folium.PolyLine)

- **What**: Drew lines between the launch sites and nearest cities or coordinates.
- **Why**: To **visualize proximity** from launch sites to other points (e.g., customers, zones of interest).

5. Custom Icons

- **What**: Added icons to markers for differentiation.
- **Why**: To improve **visual clarity** and distinguish launch sites from other markers (e.g., cities or endpoints).

✅ **Why These Objects Were Used**

- To **interactively display launch site data** on a geographic map.
- To **enhance spatial awareness** of SpaceX operations for stakeholders.
- To provide **contextual popups and layers** for richer user experience.
- To help identify the **closest cities or infrastructure** to each site for logistical insights.

GitHub Notebook for Review

Repository: IBM Data Science Capstone Project
Notebook: Interactive Folium Map of SpaceX Launch Sites
GitHub URL: https://github.com/sh184roman/IBM-Data-Science-Capstone-Project/blob/main/lab-jupyter-launch-site-location-v2.ipynb

Build a Dashboard with Plotly Dash

Plots and Graphs Included

✅ **1. Pie Chart**

- **What**: Visualizes the **distribution of successful launches** by launch site.
- **Interaction**: Updates based on the selected launch site from a dropdown menu.
- **Why**: Helps compare success rates across all launch sites or drill down into a specific one.

✅ **2. Scatter Plot**

- **What**: Plots **Payload Mass (kg)** vs. **Launch Outcome (success/failure)**.
- **Interaction**:
  - Updates dynamically based on launch site selection.
  - Includes a payload range slider to filter the data.
- **Why**: Useful to identify whether **payload mass influences launch success**, and to study variations across different sites.

User Interactions Enabled

- **Dropdown Menu**
  - Select a specific launch site or "All Sites".
  - Affects both the pie chart and the scatter plot.
- **Payload Mass Slider**
  - Allows users to filter data based on payload mass range (e.g., 0–10000 kg).
  - Affects the scatter plot to explore how payload impacts outcomes.

Why These Plots and Interactions Were Added

- **Interactivity**: Empowers users to explore the data dynamically without writing code.
- **Comparative Analysis**: Enables both broad and granular comparisons of launch performance.
- **Data Insight**: Makes it easy to detect patterns (e.g., success trends by payload or site).
- **User Engagement**: Interactive tools like sliders and dropdowns increase usability and insight generation.

GitHub Dashboard App for Review

Repository: IBM Data Science Capstone Project
File: spacex_dash_app.py
GitHub URL: https://github.com/sh184roman/IBM-Data-Science-Capstone-Project/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

Model Development Process:

- **1. Data Preprocessing**
- Loaded and cleaned the dataset.
- Encoded categorical features using OneHotEncoder.
- Scaled numerical features using StandardScaler.
- **2. Model Training**
- Trained and evaluated the following classification models:
- **Logistic Regression**
- **Support Vector Machine (SVM)**
- **Decision Tree**
- **K-Nearest Neighbors (KNN)**
- **3. Model Evaluation**
- Used **accuracy**, **F1-score**, and **Jaccard index** to evaluate model performance.
- Applied **train/test split** with train_test_split().
- **4. Model Tuning**
- Used **GridSearchCV** to optimize hyperparameters:
  - For **SVM**: kernel, C, gamma
  - For **KNN**: n_neighbors
  - For **Decision Tree**: criterion, max_depth
- **5. Best Performing Model**
- After tuning, the best model was selected based on **highest accuracy and F1-score** on the test set.

GitHub Notebook:

[View Complete Classification Notebook]

Flowchart of the Process

Data Collection → Data Cleaning & Encoding → Train/Test Split

↓

Model Training (LogReg, SVM, KNN, Tree)

↓

Evaluation (Accuracy, F1, Jaccard)

↓

Hyperparameter Tuning (GridSearchCV)

↓

Best Model Selection

Results

- **Flight Number vs. Launch Site:** As the flight number increases, the likelihood of the first stage being successfully recovered also increases.

- **Payload vs. Launch Site:** For the VAFB-SLC launch site, there have been no rockets launched with heavy payload masses (greater than 10,000 kg).

- **Success Rate vs. Orbit Type:** The SO orbit shows an almost zero success rate, while ES-L1, GEO, HEO, and SSO orbits have nearly 100% success rates

- **Flight Number vs. Orbit Type:** Success in LEO orbit appears to be related to the number of flights. However, there seems to be no relationship between flight number and success in GTO orbit.

- **Payload vs. Orbit Type:** For heavy payloads, successful landing rates are higher for Polar, LEO, and ISS orbits. In GTO, it is difficult to distinguish between positive and negative landing rates as both are present.

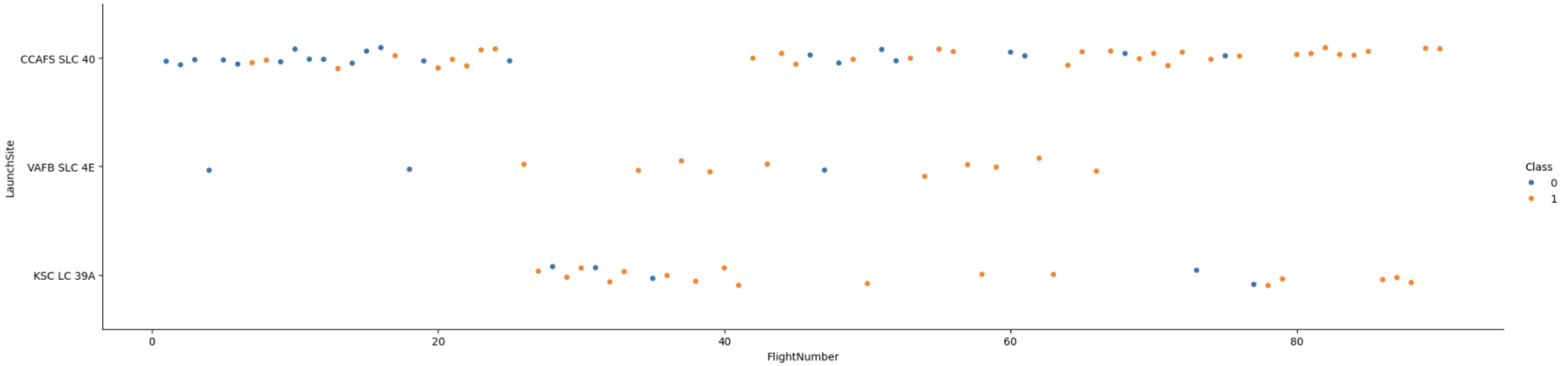
- **Launch Success Yearly Trend:** The success rate increased from 2013 to 2017 (remaining stable in 2014) and continued to increase after 2015.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the
sns.catplot(data = df, x = 'FlightNumber', y = 'LaunchSite', hue = 'Class', aspect = 4)
plt.show()
```

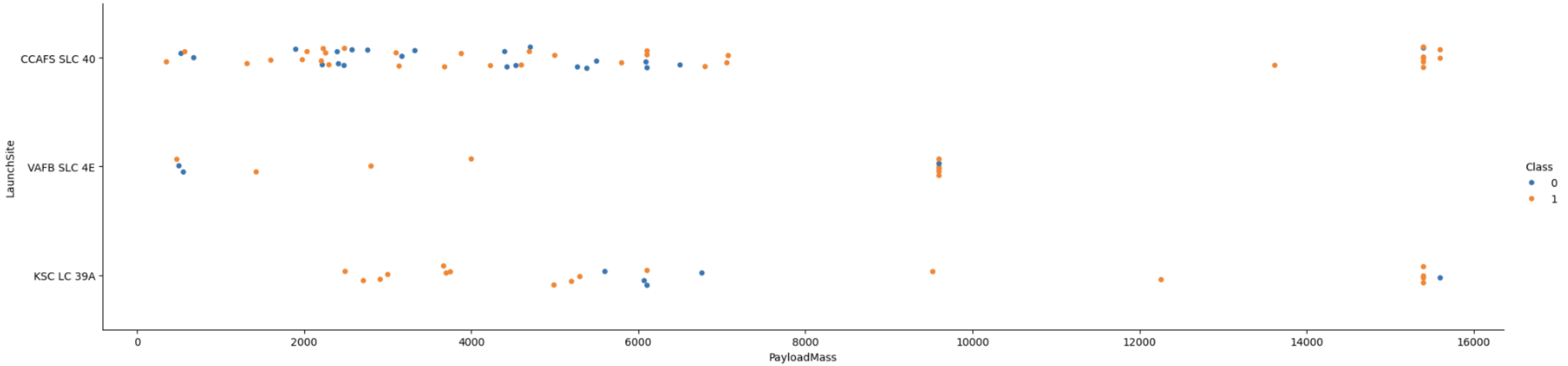


the higher the flight number, the more likely it successfully the first stage

Payload vs. Launch Site

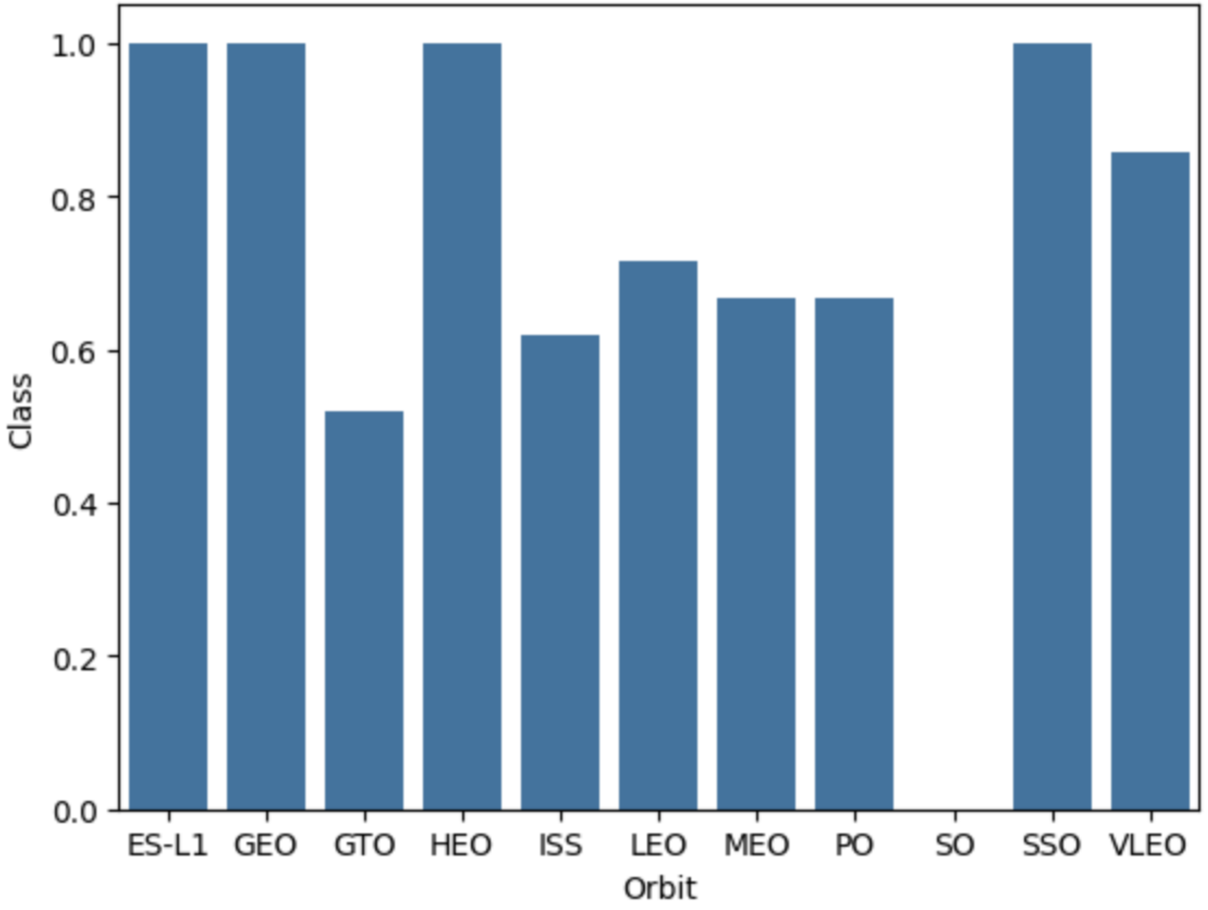
5…

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be
sns.catplot(data = df, x = 'PayloadMass', y = 'LaunchSite', hue = 'Class', aspect = 4)
plt.show()
```



Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

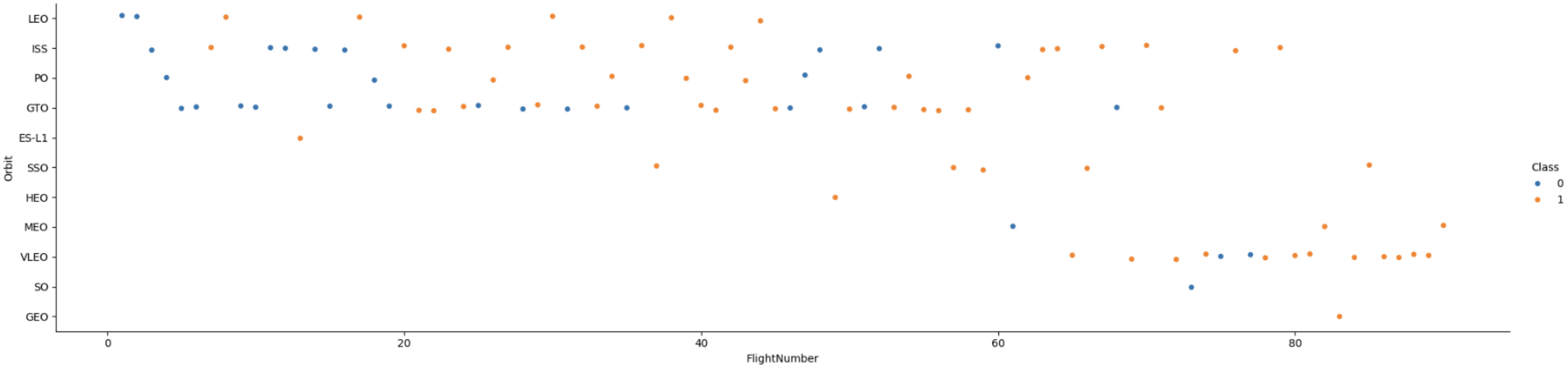
Success Rate vs. Orbit Type



Analyze the ploted bar chart try to find which orbits have high sucess rate. SO orbit has almot zero success rate howver ESL1, geo, heo and sso has almost 100% success rate

Flight Number vs. Orbit Type

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class v
sns.catplot(data = df, x = 'FlightNumber', y = 'Orbit', hue = 'Class', aspect = 4)
plt.show()
```

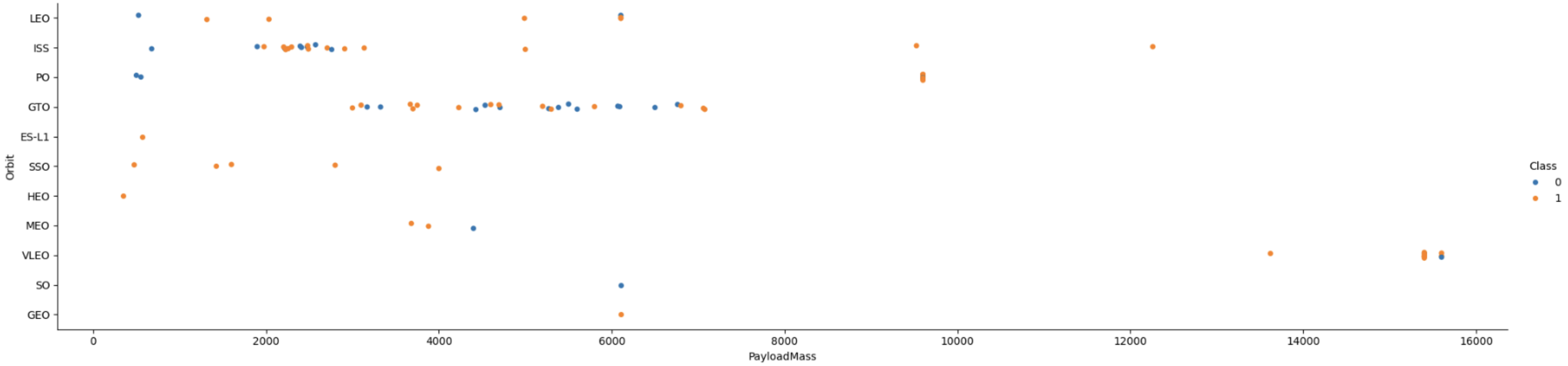


You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type

68…

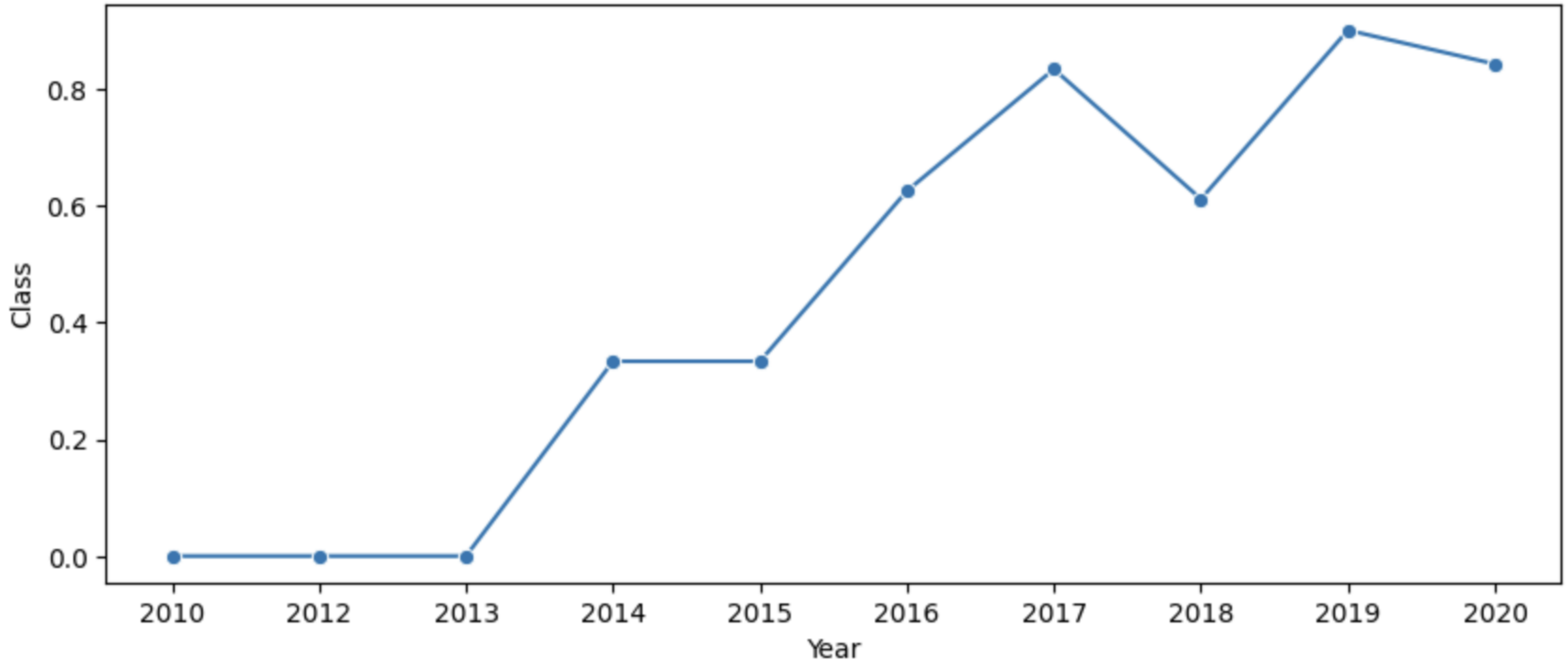
```python
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(data = df, x = 'PayloadMass', y = 'Orbit', hue = 'Class', aspect = 4)
plt.show()
```



With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend



You can observe that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.

All Launch Site Names

Display the names of the unique launch sites in the space mission

```
QUERY = """
SELECT DISTINCT(Launch_Site)
FROM SPACEXTBL
"""
pd.read_sql_query(QUERY, con)
```

	Launch_Site
0	CCAFS LC-40
1	VAFB SLC-4E
2	KSC LC-39A
3	CCAFS SLC-40

There are four unique Launch Sites in the space mission

Launch Site Names Begin with 'CCA'

Display 5 records where launch sites begin with the string 'CCA'

```
QUERY = """
SELECT *
FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
"""
pd.read_sql_query(QUERY,con)
```

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landir
0	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure
1	2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure
2	2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	
3	2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	
4	2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	

Explanation:

- **CCAFS LC-40** stands for **Cape Canaveral Air Force Station Launch Complex 40**.
- These are **early Falcon 9 missions** (2010–2013).
- All show **no successful landings** — either "Failure (parachute)" or "No attempt" in "Landing_Outcome".

Total Payload Mass

Display the total payload mass carried by boosters launched by NASA (CRS)

```
QUERY = '''
SELECT CUSTOMER, SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)'
'''
pd.read_sql_query(QUERY, con)
```

	Customer	SUM(PAYLOAD_MASS__KG_)
0	NASA (CRS)	45596

Explanation:

- This result means that SpaceX carried a total of **45,596 kg** of payloads for **NASA's Commercial Resupply Services (CRS)** missions.
- CRS missions are a series of cargo resupply flights to the **International Space Station (ISS)** under contract with NASA.
- The total is summed from all missions where the "Customer" field is exactly "NASA (CRS)".

Average Payload Mass by F9 v1.1

Display average payload mass carried by booster version F9 v1.1

```
QUERY = '''
SELECT Booster_Version, SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Booster_Version LIKE 'F9 v1.1%'
'''
pd.read_sql_query(QUERY, con)
```

	Booster_Version	SUM(PAYLOAD_MASS__KG_)
0	F9 v1.1 B1003	38020

Explanation:

- The query filtered all records where the **booster version** starts with 'F9 v1.1'.
- It then calculated the **average payload mass** for that version.
- The result shows that **F9 v1.1 B1003** carried an average of **~2535 kg** per launch (based on the number of flights it had in the dataset).

First Successful Ground Landing Date

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
QUERY = '''
SELECT MIN(DATE) AS First_Successful_Ground_Pad_Landing
FROM SPACEXTBL
WHERE "Landing_Outcome" = 'Success (ground pad)'
'''
pd.read_sql_query(QUERY, con)
```

	First_Successful_Ground_Pad_Landing
0	2015-12-22

Explanation:

- The first successful landing on a ground pad occurred on December 22, 2015.
- This was SpaceX Flight 20, the first time Falcon 9 successfully landed back on solid ground at Landing Zone 1 (Cape Canaveral).

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
QUERY = '''
SELECT Booster_Version, PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE LANDING_OUTCOME = 'Success (drone ship)'
AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
'''
pd.read_sql_query(QUERY, con)
```

	Booster_Version	PAYLOAD_MASS__KG_
0	F9 FT B1022	4696
1	F9 FT B1026	4600
2	F9 FT B1021.2	5300
3	F9 FT B1031.2	5200

Explanation:

- This query returns boosters that:
  - Landed **successfully** on a **drone ship**
  - Carried **payloads between 4000 and 6000 kg**
- All listed boosters are of type **F9 Full Thrust (FT)** — a reusable Falcon 9 variant introduced by SpaceX to support heavier payloads and recovery.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
QUERY = '''
SELECT TRIM(Mission_Outcome) AS Cleaned_Outcome, COUNT(*) AS TOTAL
FROM SPACEXTBL
GROUP BY TRIM(Mission_Outcome)
'''
pd.read_sql_query(QUERY, con)
```

	Cleaned_Outcome	TOTAL
0	Failure (in flight)	1
1	Success	99
2	Success (payload status unclear)	1

Explanation:
Total Successful Missions: 99
Total Failed Missions: 1 (in flight)
Uncertain Outcomes: 1 (payload status unclear)

Boosters Carried Maximum Payload

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

```
QUERY = '''
SELECT Booster_Version, PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL)
'''
pd.read_sql_query(QUERY, con)
```

	Booster_Version	PAYLOAD_MASS__KG_
0	F9 B5 B1048.4	15600
1	F9 B5 B1049.4	15600
2	F9 B5 B1051.3	15600
3	F9 B5 B1056.4	15600
4	F9 B5 B1048.5	15600
5	F9 B5 B1051.4	15600
6	F9 B5 B1049.5	15600
7	F9 B5 B1060.2	15600
8	F9 B5 B1058.3	15600
9	F9 B5 B1051.6	15600
10	F9 B5 B1060.3	15600
11	F9 B5 B1049.7	15600

Explanation:

These booster versions carried the maximum payload mass of 15,600 kg, identified using a subquery that finds the highest payload mass in the dataset.

2015 Launch Records

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
QUERY = '''
SELECT substr("Date", 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTBL
WHERE Landing_Outcome = 'Failure (drone ship)'
AND substr(DATE, 0, 5) = '2015'
'''
pd.read_sql_query(QUERY, con)
```

	Month	Landing_Outcome	Booster_Version	Launch_Site
0	01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
1	04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Explanation:
This query lists the months in 2015 when booster landings on drone ships failed. It shows the month number, landing outcome, booster version, and launch site. Since SQLite lacks a month name function, the month is extracted using substring from the date.

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.

```
QUERY = '''
SELECT Mission_Outcome, Landing_Outcome, COUNT(*) AS TOTAL_COUNT
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' and '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY TOTAL_COUNT DESC

'''
pd.read_sql_query(QUERY, con)
```

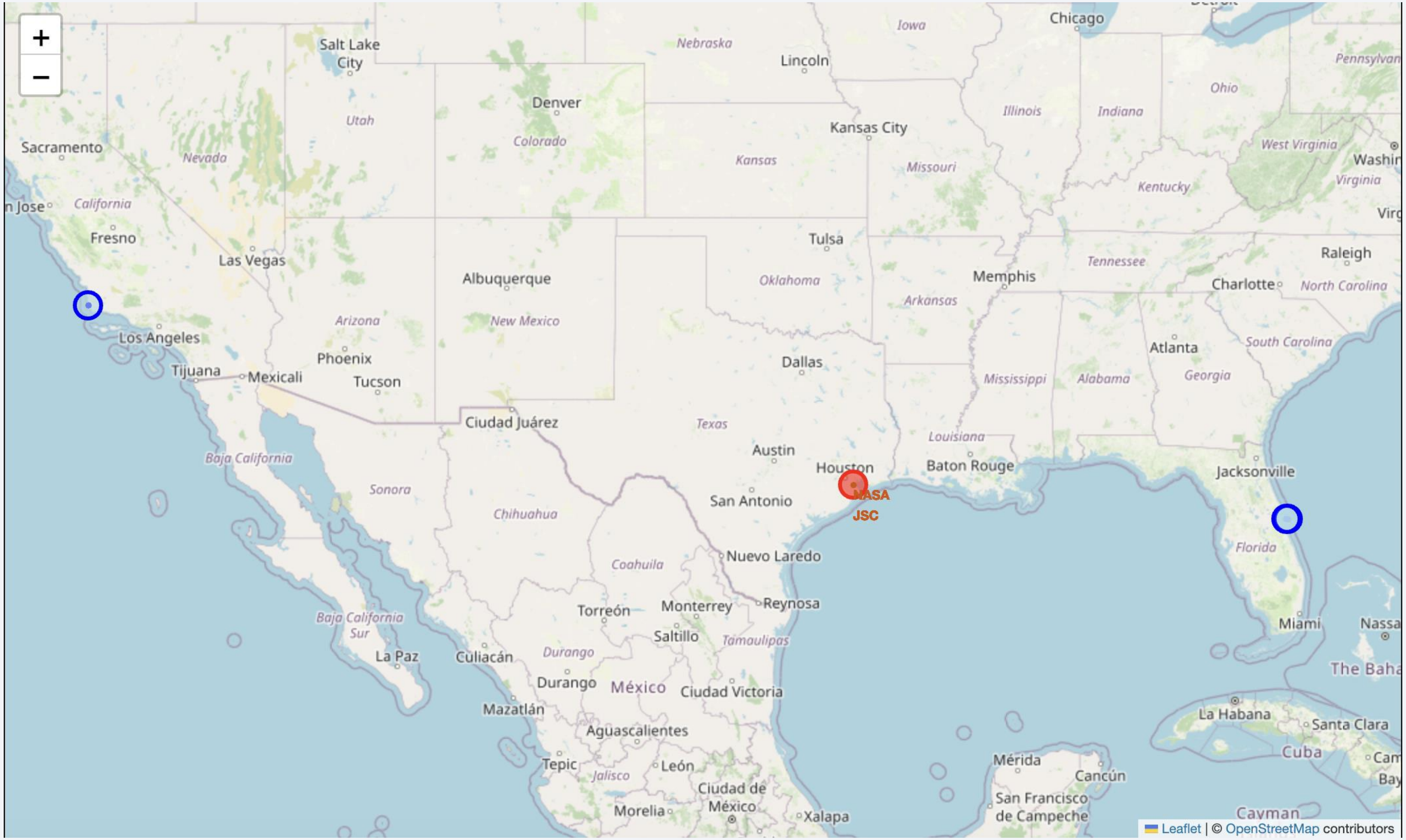
	Mission_Outcome	Landing_Outcome	TOTAL_COUNT
0	Success	No attempt	10
1	Success	Success (drone ship)	5
2	Success	Failure (drone ship)	5
3	Success	Success (ground pad)	3
4	Success	Controlled (ocean)	3
5	Success	Uncontrolled (ocean)	2
6	Success	Failure (parachute)	2
7	Failure (in flight)	Precluded (drone ship)	1

Explanation:
This query ranks landing outcomes by their counts between June 4, 2010, and March 20, 2017. The most frequent landing outcome is "No attempt" (10 times), followed by equal counts of "Success (drone ship)" and "Failure (drone ship)" (5 times each). Other outcomes have fewer occurrences. The results help understand the distribution and success rate of different landing attempts in that period.

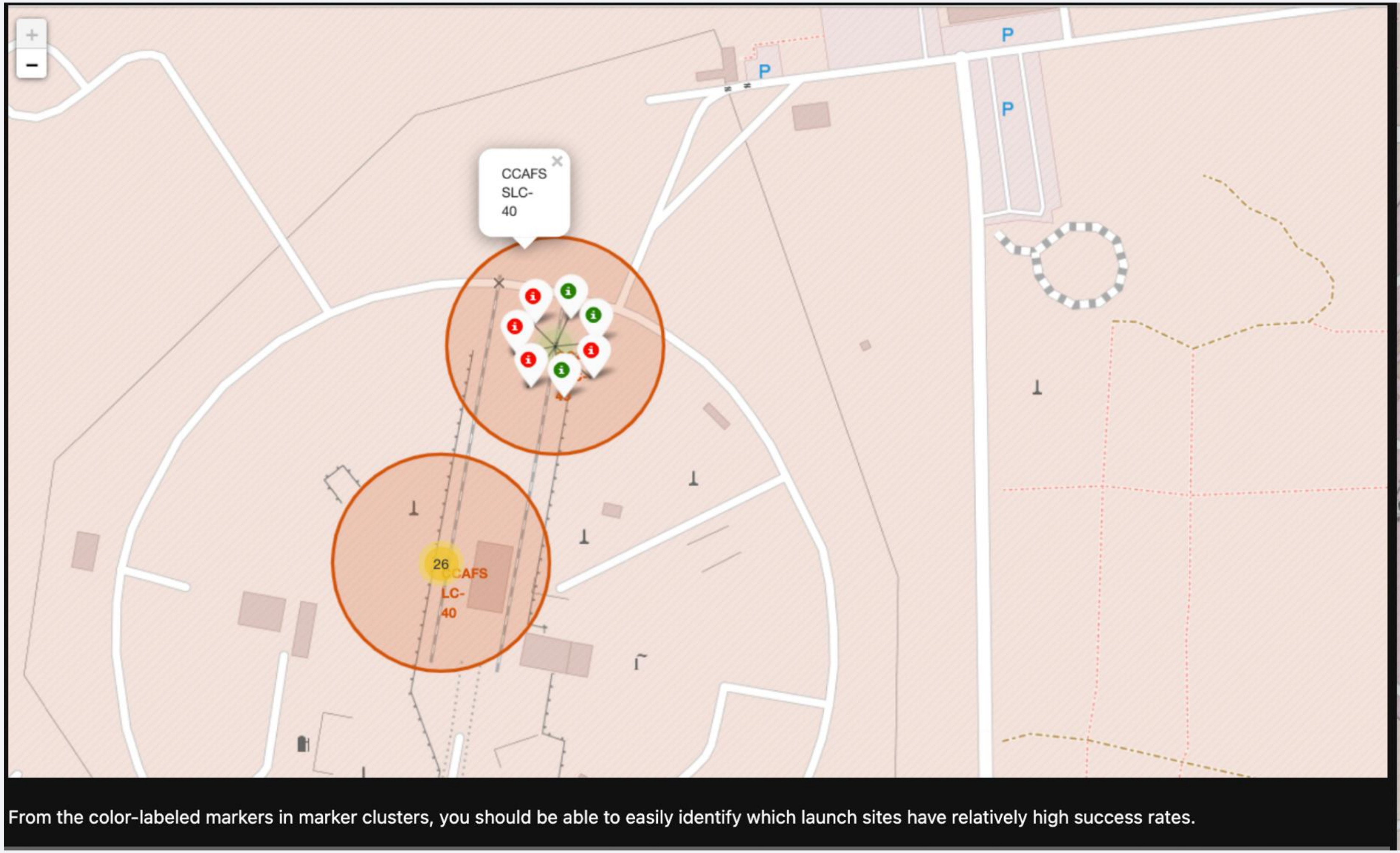
Section 3

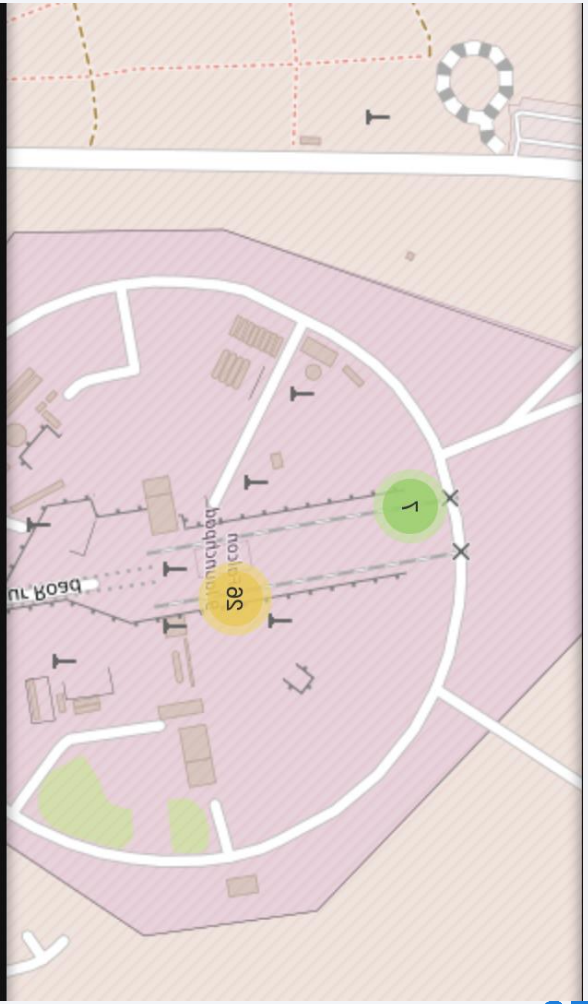
Launch Sites Proximities Analysis

All launch sites on a map

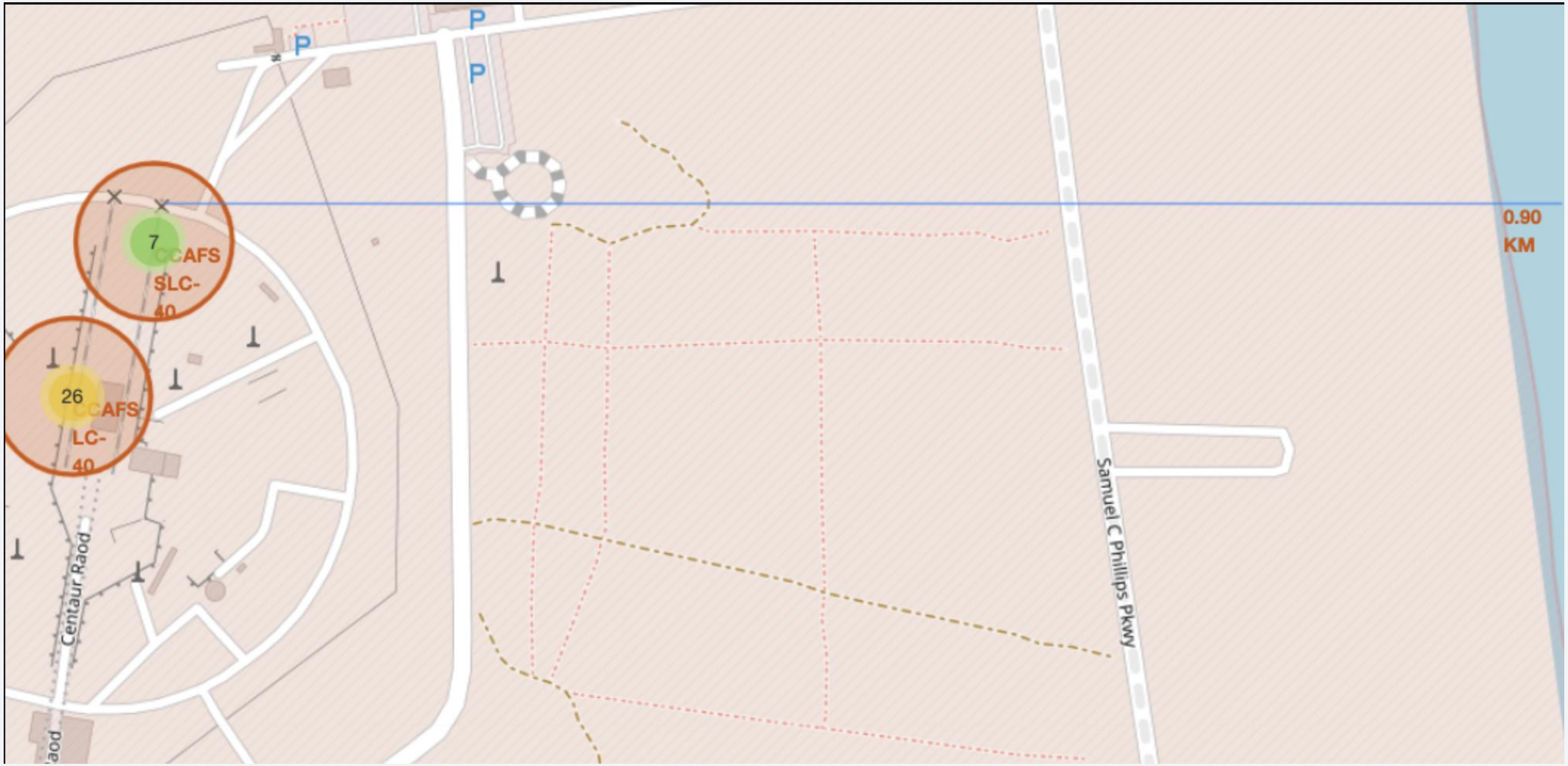


Mark the success/failed launches for each site on the map





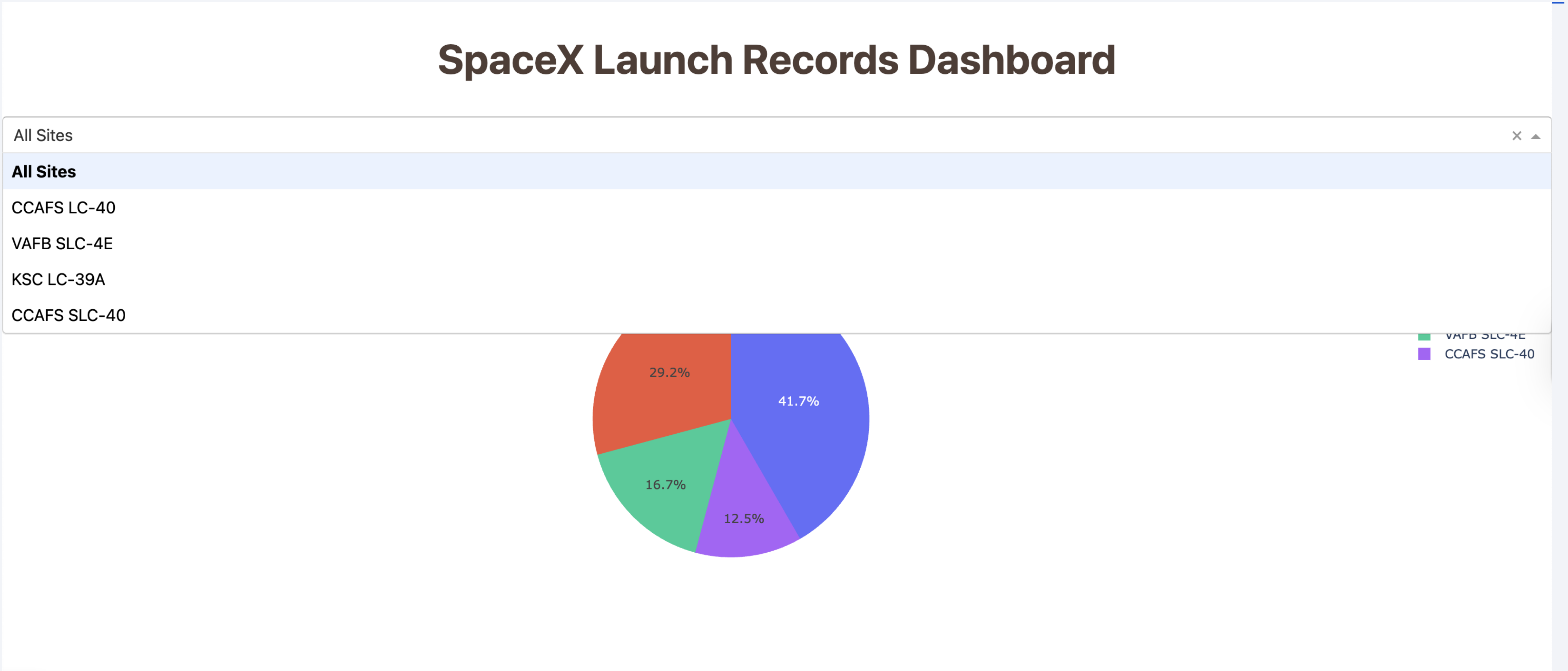
Calculate the distances between a launch site to its proximities



Section 4

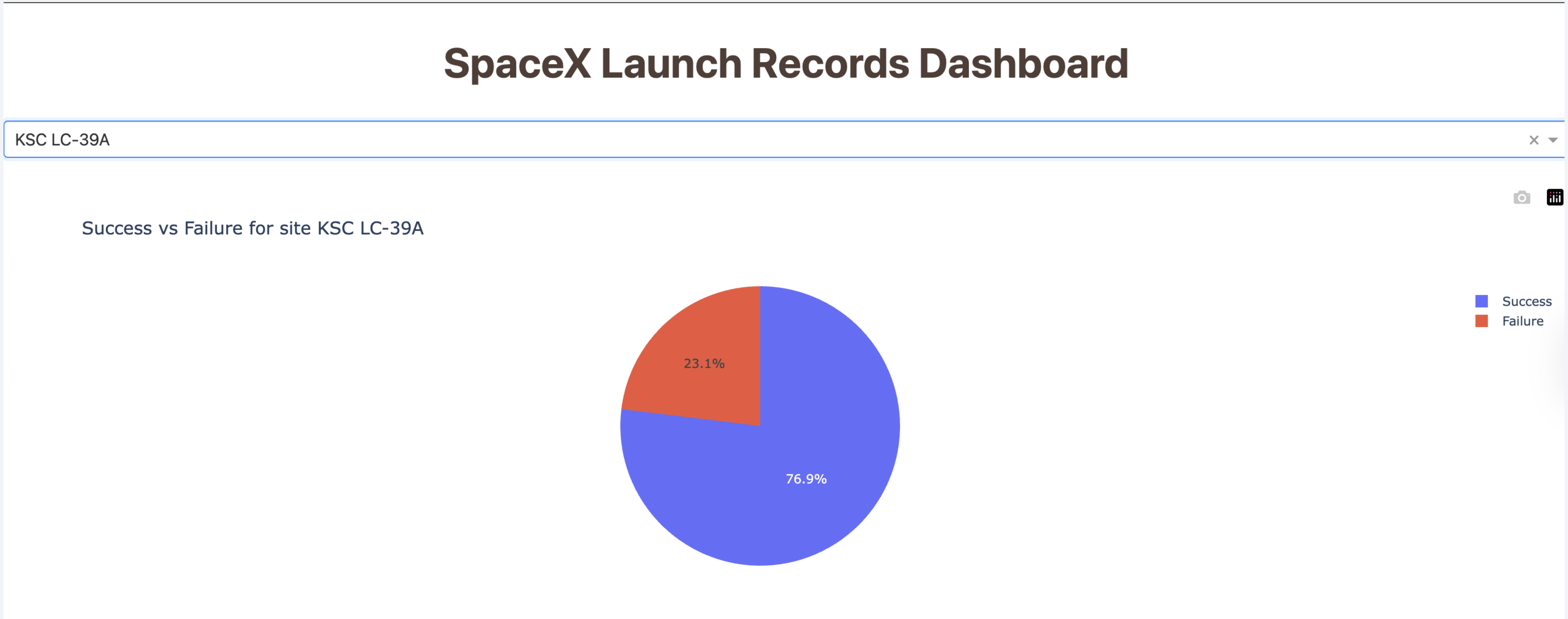
Build a Dashboard with Plotly Dash

SpaceX Launch Records Dashboard



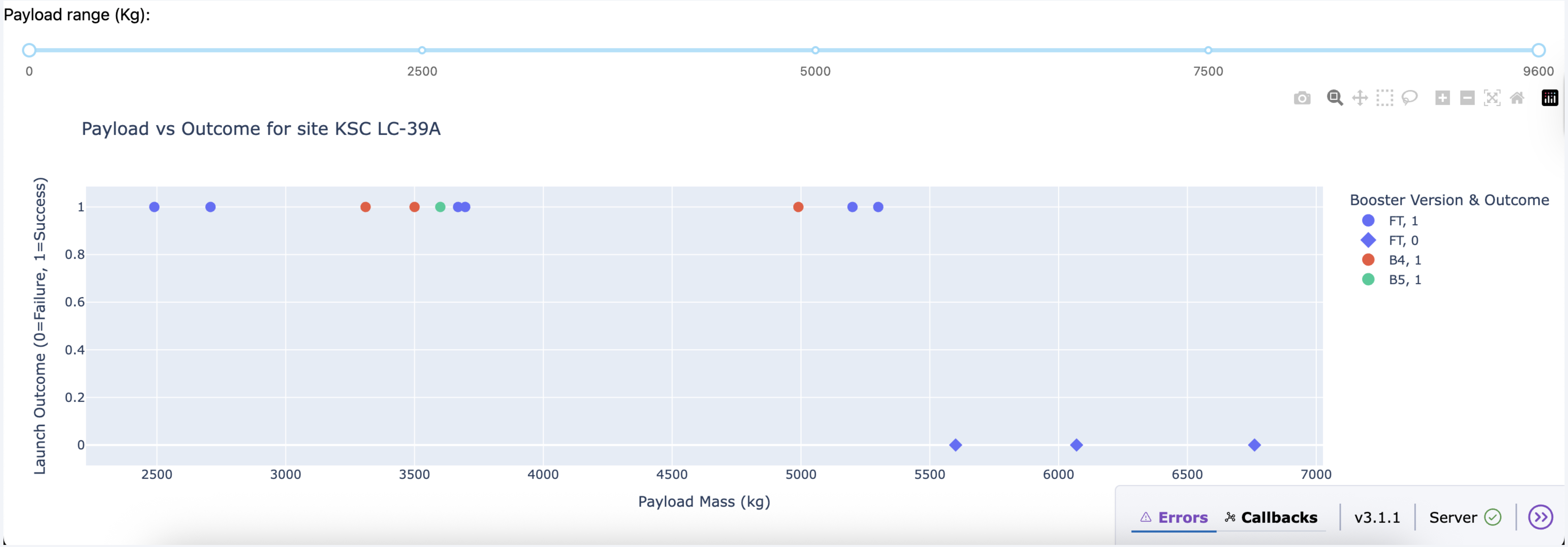
A dropdown menu was created to filter launch records displayed in the pie chart, allowing users to select and analyze outcomes by individual launch sites.

Success vs Failure Piechart



Among all SpaceX launch sites, KSC LC-39A demonstrates the highest mission success rate.

Payload vs Outcome Plot

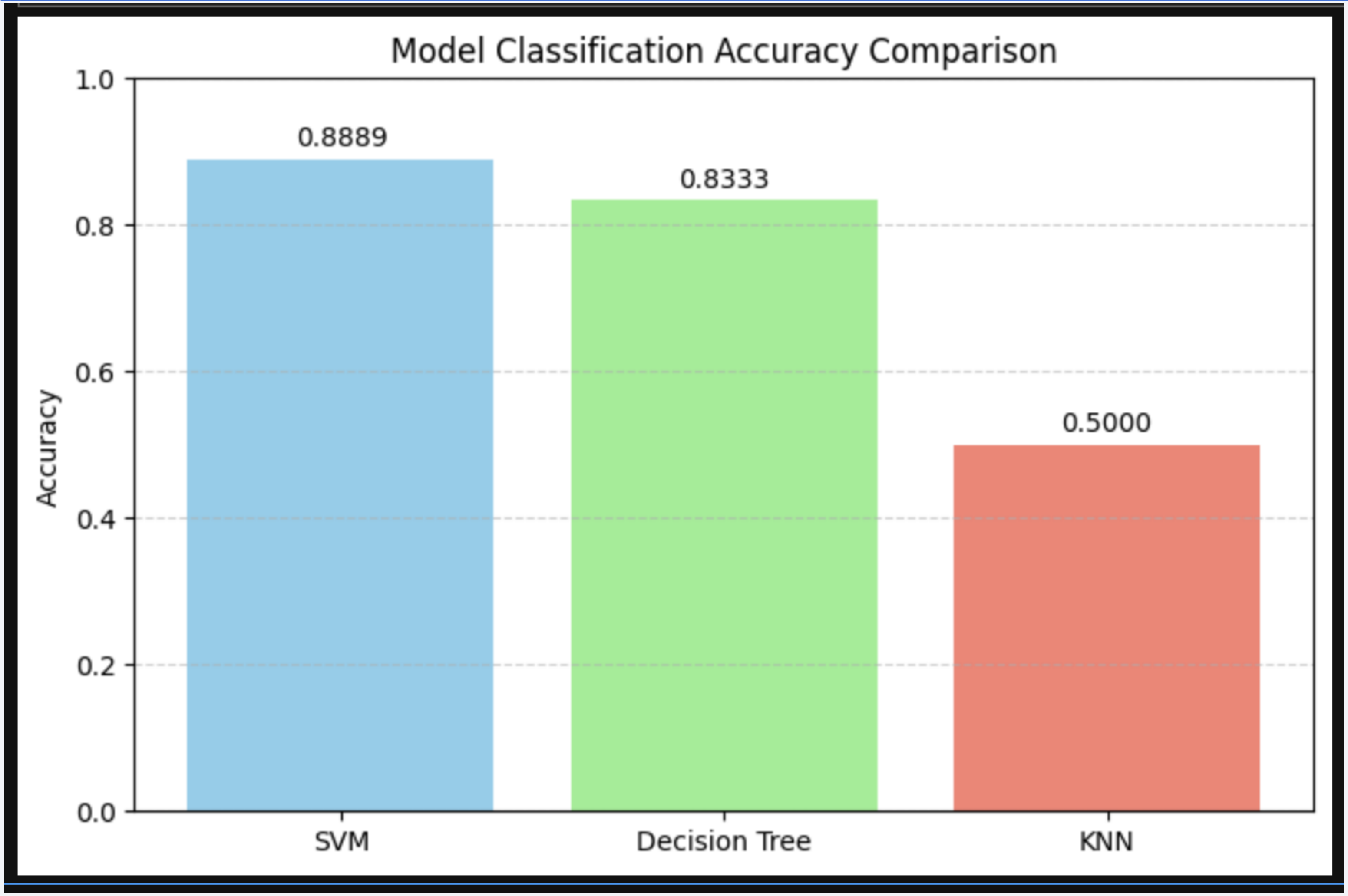


Booster version FT, associated with a binary landing outcome of '0' (indicating failure), had no successful missions, unlike other booster versions that achieved higher success rates.

Section 5

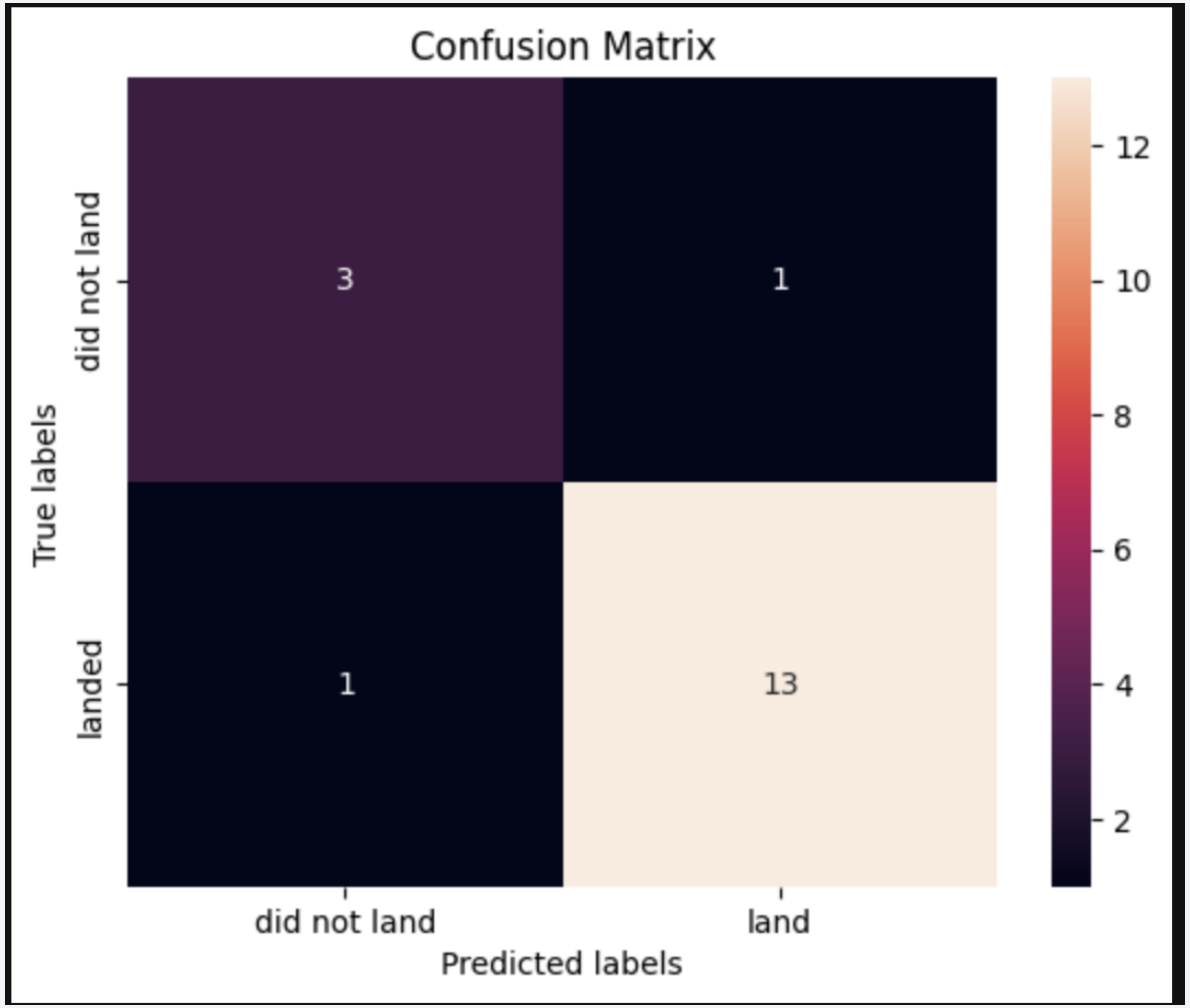
Predictive Analysis (Classification)

Classification Accuracy



SVM is the best performing model for this classification task based on test accuracy.

Confusion Matrix



- The model correctly predicted **3** class 0 samples and **13** class 1 samples.
- It misclassified **1** class 0 sample as class 1 (false positive).
- It also misclassified **1** class 1 sample as class 0 (false negative).
- This results in a high overall accuracy of **88.89%**, meaning the model is performing well.

Predictive Analysis Results

- **SpaceX Launch Records Dashboard:** A dropdown menu allows filtering launch records displayed in the pie chart by individual launch sites.

- **Success vs Failure Piechart:** KSC LC-39A demonstrates the highest mission success rate among all SpaceX launch sites.

- **Payload vs Outcome Plot:** Booster version FT, with a binary landing outcome of '0' (failure), had no successful missions, unlike other booster versions.

- **Classification Accuracy:** SVM is the best performing model for this classification task, based on test accuracy of 0.8889. Decision Tree has an accuracy of 0.8333, and KNN has 0.5000.

- **Confusion Matrix:** The model correctly predicted 3 class 0 samples and 13 class 1 samples. It misclassified 1 class 0 sample as class 1 (false positive) and 1 class 1 sample as class 0 (false negative). This resulted in a high overall accuracy of 88.89%.

Conclusions

- Launch site success rates vary significantly, with KSC LC-39A showing the highest success rate.
- Payload mass influences landing outcomes, with heavier payloads potentially affecting success rates for certain orbit types.
- Machine learning models, particularly SVM, demonstrate high accuracy in predicting landing outcomes
- Interactive dashboards and maps provide valuable tools for exploring and understanding complex launch data.

Appendix

- GitHub Notebook for Data Collection – SpaceX API:
- SpaceX API Data Collection – GitHub Notebook
- GitHub Notebook for Data Collection - Scraping:
- Repository: IBM Data Science Capstone Project, Notebook: Web Scraping with BeautifulSoup & Pandas
- GitHub Notebook for Data Wrangling:
- Repository: IBM Data Science Capstone Project, Notebook: Data Wrangling for SpaceX Falcon 9 Launch Data
- GitHub Notebook for EDA with Data Visualization:
- Repository: IBM Data Science Capstone Project, Notebook: EDA & Data Visualization on SpaceX Launch Data
- GitHub Notebook for EDA with SQL:
- Repository: IBM Data Science Capstone Project, Notebook: SQL Analysis with SQLite
- GitHub Notebook for Interactive Map with Folium:
- Repository: IBM Data Science Capstone Project, Notebook: Interactive Folium Map of SpaceX Launch Sites
- GitHub Dashboard App for Plotly Dash:
- Repository: IBM Data Science Capstone Project, File: spacex_dash_app.py

Thank you!