IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Yusua Agabus>
<15-May-2025>

Outline

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

Executive Summary

- **Summary of methodologies**
  - ✓ Data Collection
  - ✓ Data Wrangling
  - ✓ EDA with SQL
  - ✓ EDA with Visualization
  - ✓ Building Interactive Visual Analytics (Map) with Folum
  - ✓ Building Interactive Dashboard with Plotly Dash
  - ✓ Machine Learning for Predictive Analysis
- **Summary of all results**
  - ✓ Exploratory Data Analysis Results
  - ✓ Interactive Analytics Demo in Screenshots
  - ✓ Predictive Analysis Results

Introduction

- SpaceX advertises Falcon 9 rocket launches on its website with a cost of $62 million dollars, significantly lower than other providers' costs of at least $165 million dollars each. The cost savings is mainly attributed to SpaceX's ability to re-use its Falcon 9 First Stage
- In this Capstone project, we will attempt to predict whether the Falcon 9 First Stage will land successfully, by studying the impact of certain variables & conditions on the success rate of First Stage landing.
- Doing so will help to determine the cost of a launch, which will be useful to an alternative company that wishes to bid against SpaceX for a rocket launch

Section 1

Methodology

Methodology

Executive Summary

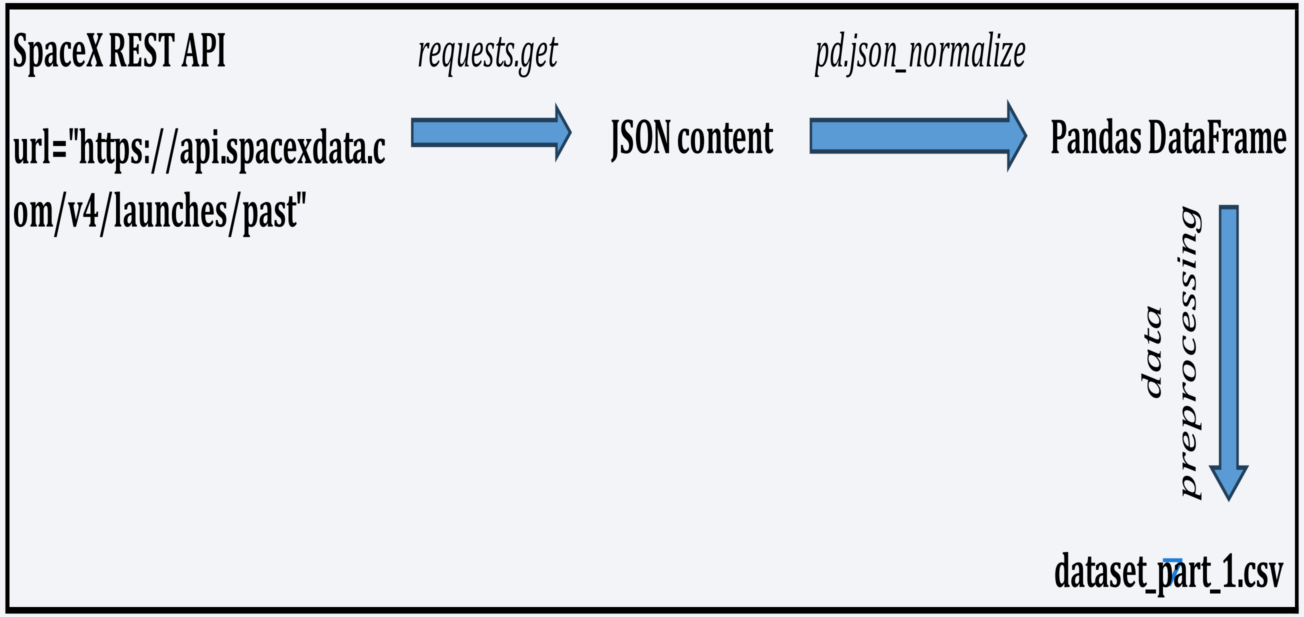
- Data collection methodology: Data is collected from 2 sources: SpaceX's API (using the GET request), and Wikipedia page titled 'List of Falcon 9 and Falcon Heavy launches' (via web scraping)
- Perform data wrangling: data is transformed by categorizing landing outcomes into Bad (0) & Good (1) and storing the binary categories into new column, Class
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models: Run Logistic Regression, Support Vector Machine (SVM), Decision Tree, and K-Nearest Neighbors (KNN) models, using GridSearchCV with 10-fold cross-validation to optimize the hyperparameters

Data Collection – SpaceX API

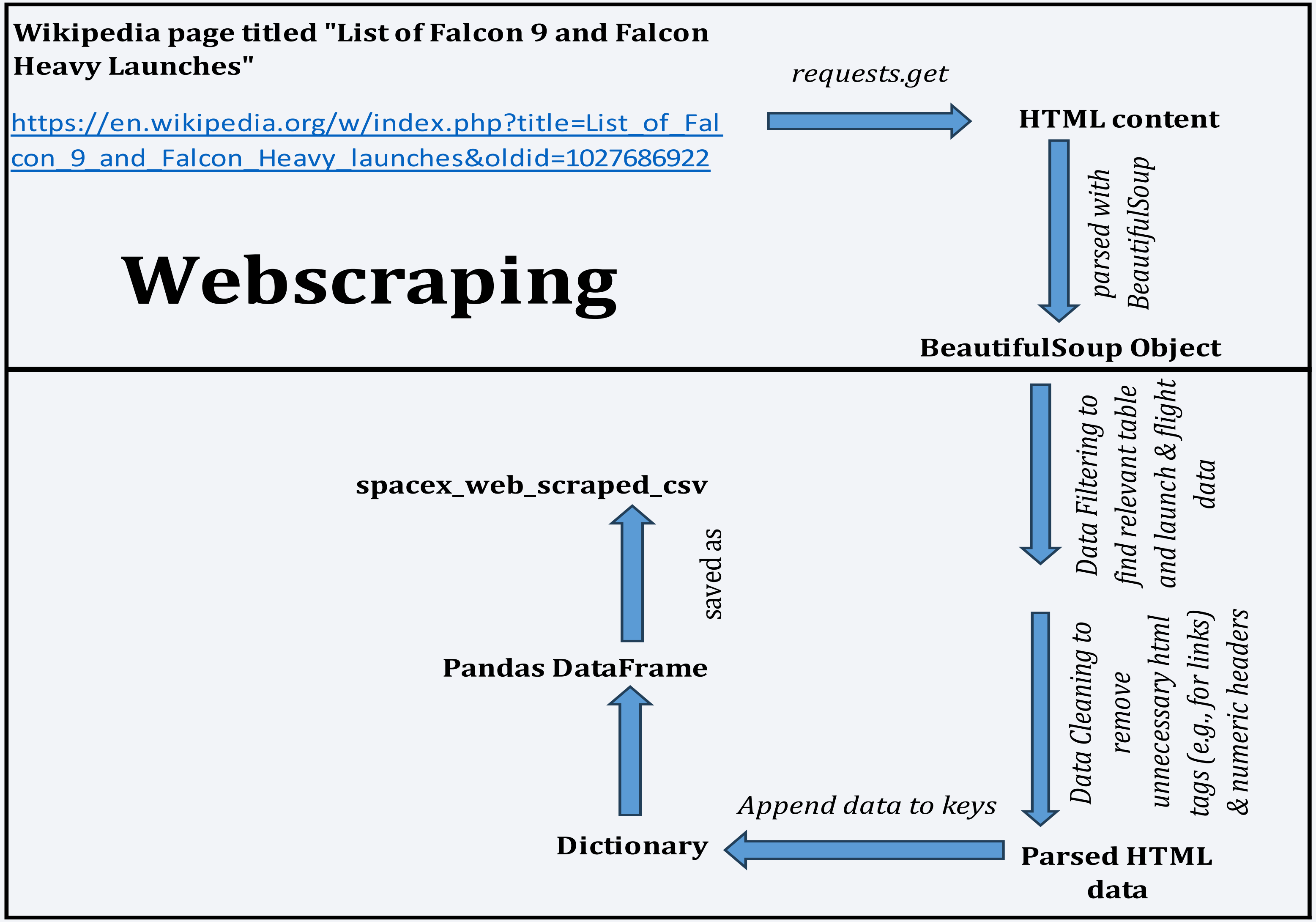
- This project utilizes static historical SpaceX launch data, sourced from a previous API call to SpaceX's website records, and using GET function to fetch the information as a JSON response. The JSON content is then normalized into a Pandas DataFrame (pd) to facilitate subsequent preprocessing
- The pd dataset is (1) filtered to create the desired subset (with rocket, payloads, launchpad, cores, flight no., date columns), remove unwanted features (rockets with multiple payloads & cores), & restrict launches to Falcon 9 and those before 13-Nov-2020 only, (2) time-formatted, and (3) cleaned (replace missing values with mean)
- The resulting dataset is saved as dataset_part_1.csv

The Jupyter notebook containing the data collection and preprocessing steps described can be found in the following Github url:

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M1a_Data%20Collection%20for%20Falcon%209%20First%20Stage%20Landing%20Prediction.ipynb



Data Collection - Scraping



The Jupyter notebook containing the Webscraping and preprocessing steps described can be found in the following Github url:

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M1b_Data%20Collecion%20with%20Web%20Scraping%20for%20Falcon%209%20Records.ipynb

Data Wrangling

Exploratory Data Analysis on Dataset

Load dataset from csv file onto Pandas DataFrame

Inspect data structure

Assess % of missing values in each column

Calculate the number of launches on each site

Calculate the number and occurrence of each orbit

Data Transformation

Categorized Landing Outcomes Into **Bad** (None, False ASDS, False Ocean, None ASDS, False RTLS = 0) and **Good** (True ASDS, True RTLS, True Ocean = 1)

Create a new column, **Class** to store the binary categories (0, 1) of Landing Outcomes

Calculate the success rate of Falcon 9 First Stage Landings

success rate = df['Class'].mean

Export the cleaned & transformed DataFrame to dataset_part_2.csv

The Jupyter notebook containing the Data Wrangling steps described can be found in the following Github url:

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M1c_Data%20Wrangling%20for%20Falcon%209%20First%20Stage%20Landing%20Prediction.ipynb

EDA with SQL

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M2a_Exploratory%20Data%20Analysis%20with%20SQL.ipynb

- Read from Data Source and Write to SQL Tables
- Display the names of unique launch sites ("SELECT DISTINCT Launch_Site FROM SPACEXTABLE;“)
- Display 5 records where launch sites begin with ‘CCA ("SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;")
- Display total payload mass carried by boosters launched by NASA, and average payload mass carried by booster version F9 v1.1
- List the date when the first successful landing on ground pad was achieved
- List the names of boosters with successful landings on drone ship and payload mass of greater than 4,000 kg, but less than 6,000 kg
- List the total number of successful and failure mission outcomes
- List all booster versions that have carried the maximum payload mass, using a subquery
- List the records of month names, failure landing outcomes on drone ship, booster versions, and launch sites in 2015
- Rank the count of landing outcomes (e.g., Failure on drone ship, Success on ground pad) between 2010-06-04 and 2017-03-20, in descending order

EDA with Data Visualization

The following charts were plotted for the EDA with Data Visualization:

- Scatter Plot of Flight Number vs. Launch Site (to identify trends in launch success across sites, whether certain sites have higher probability of success)
- Scatter Plot of Payload Mass vs. Launch Site (to analyze the impact of payload mass on landing success across different launch sites)
- Bar Chart of Success Rate for each Orbit type (to highlight orbit types with higher or lower landing success rates)
- Scatter Plot of Flight Number vs. Orbit (to analyze the relationship between the number of flights and success rate for each orbit type)
- Scatter Plot of Payload Mass vs. Orbit (to analyze the relationship between payload mass and success rate for each orbit type)
- Line Chart of Success Rate Yearly Trend (to track success rate changes over time)

Github url:

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M2b_Exploratory%20Data%20Analysis%20with%20Visualization.ipynb

Build an Interactive Map with Folium

To help visualize the Launch & First Stage Landing Data in an interactive Folium map, the following folium objects were added onto locations on the map (e.g., launch sites), using latitude and longitude coordinates to position these objects on the map:

- Circle (with radius of 1,000m and pop-ups to provide location details) to highlight the zone surrounding each location of interest e.g., launch site
- Marker to pinpoint locations of interest e.g., launch sites. Launch site marker comes with icon (DivIcon) to display text label for launch site name, colors to indicate successful & failed launches at the site, and clustered markers to improve readability
- MousePosition to get the coordinates for a mouse over (location of interest) on map
- PolyLine, to draw a line connecting two locations e.g., launch site and infrastructure (coastline, railway, highway, city) in its proximity. The distance between the 2 points connected by the line can be calculated separately using Haversine formula

Github url:

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M3a_Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- The Dashboard comes with 2 interactive features:
  - ✓ A dropdown menu that allows users to select 'All Sites' or individual launch sites (i.e., CCAFS LC-40, VAFB SLC-4E, CCAFS SLC-40, KSC LC-39A)
  - ✓ A RangeSlider that allows users to indicate the Payload Mass value, with increment of 1,000 kg and marked values of '0', '2500', '5000', '7500', '10000'
- The Dashboard outputs 2 charts based on inputs provided by users:
  - ✓ A Pie chart showing the % contribution of each site to successful launches (when 'All Sites' is selected) or the Success vs. Failure % (when individual site is selected)
  - ✓ A Scatter plot showing Payload vs. Launch Outcome (Failed, Success) for each Booster version at All Sites or individual launch site (selected by users)
- The Dashboard features & output charts allow users to analyze (1) site's success rate & contribution to total successful launches, (2) impact of Payload on Launch success or failure across sites and Booster versions

Github url:

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M3b_Interactive%20Dashboard%20with%20Plotly%20Dash.py

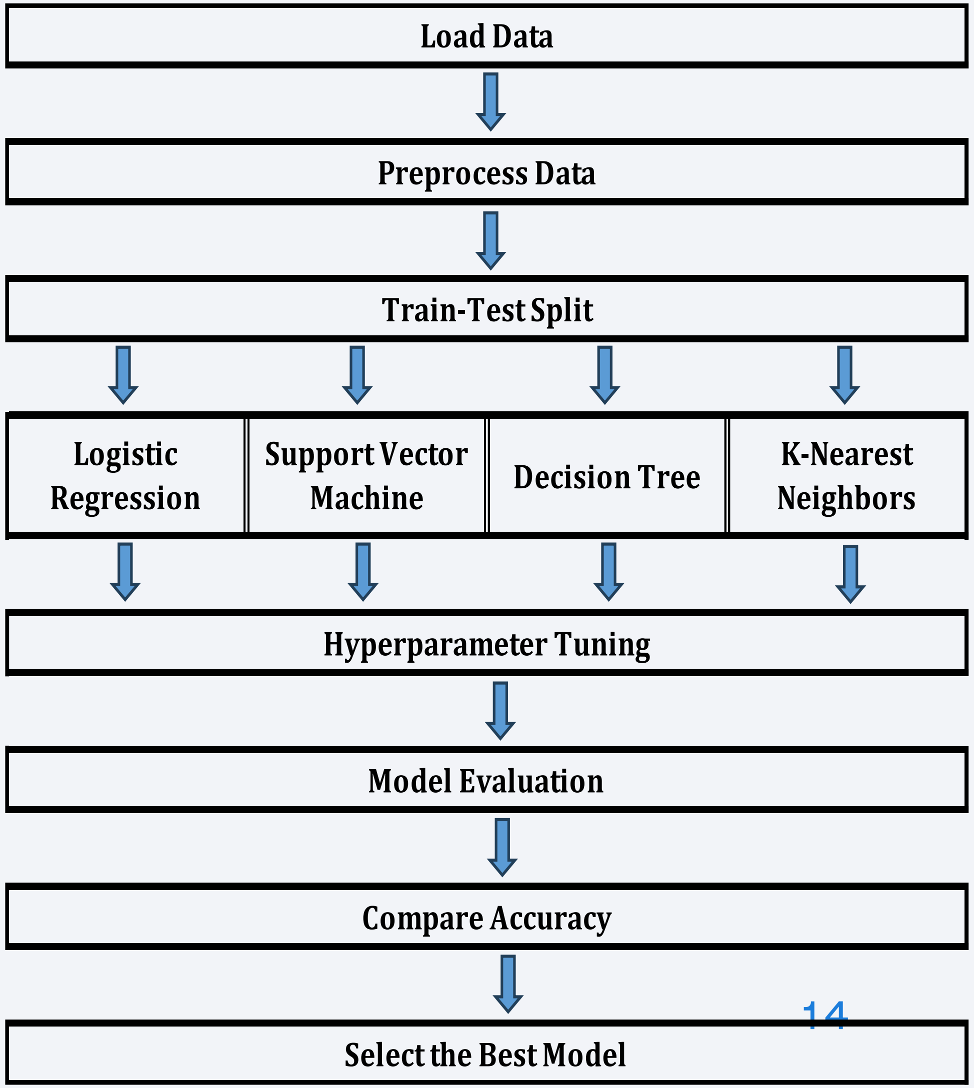
Predictive Analysis (Classification)

Build, evaluate, & find the best performing Classification Model:

1. Load & Preprocess datasets, Standardize features using StandardScaler() to normalize values
2. Separate input features (X) & target output (Y). Apply train_test_split() to create training & test datasets (80/20)
3. Run Logistic Regression, Support Vector Machine (SVM), Decision Tree, and K-Nearest Neighbors (KNN) models. Use GridSearchCV with 10-fold cross-validation to optimize the hyperparameters
4. Compare accuracy scores on test data for each model. Plot Confusion Matrix to visualize Classification performance
5. Compare accuracy results on test data to find the best model

Github url:

https://github.com/yusua2201/YARepo/blob/main/Capstone%20M4_Machine%20Learning%20Prediction%20on%20Falcon%209%20First%20Stage%20Landing.ipynb



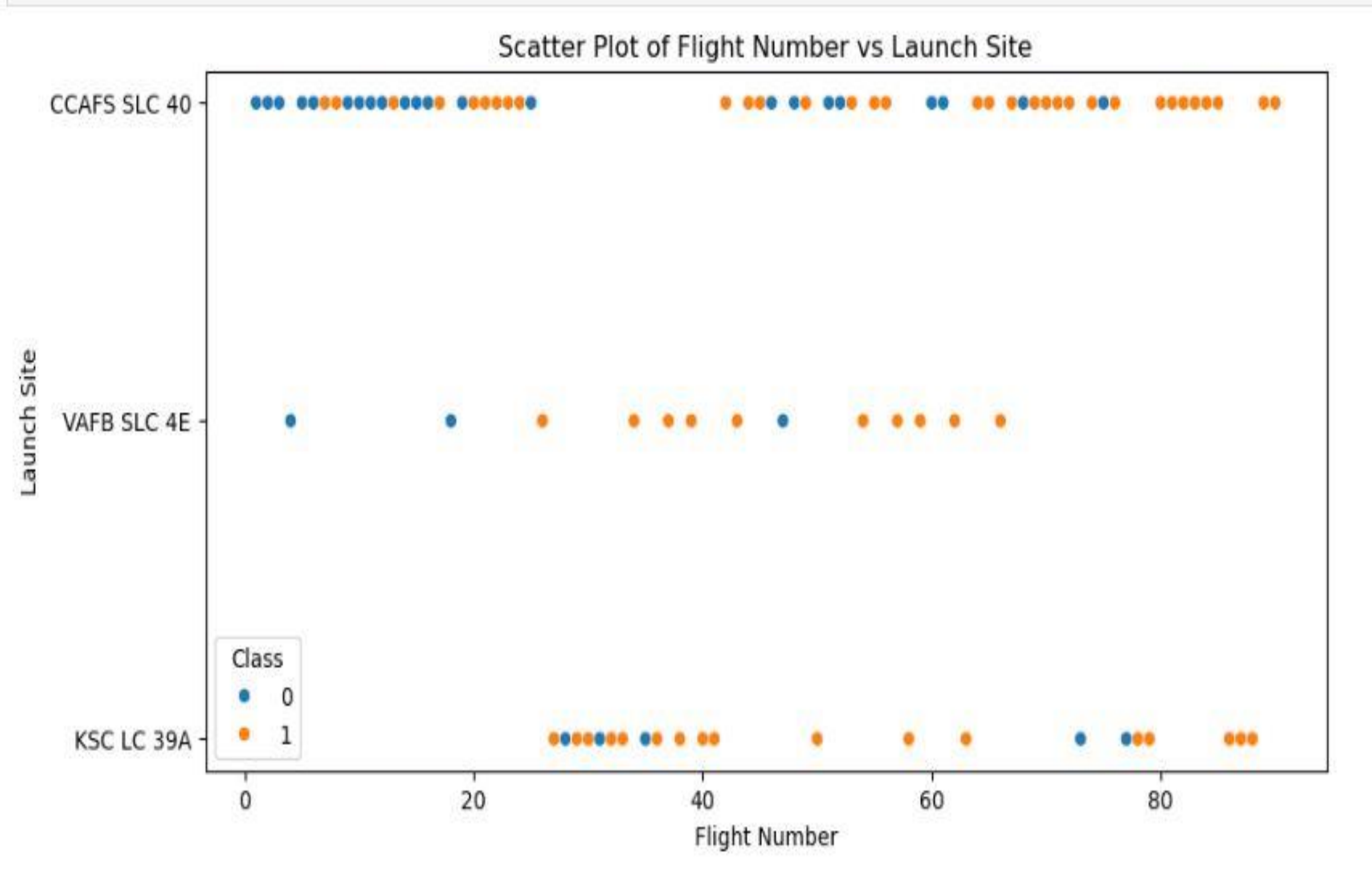
Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

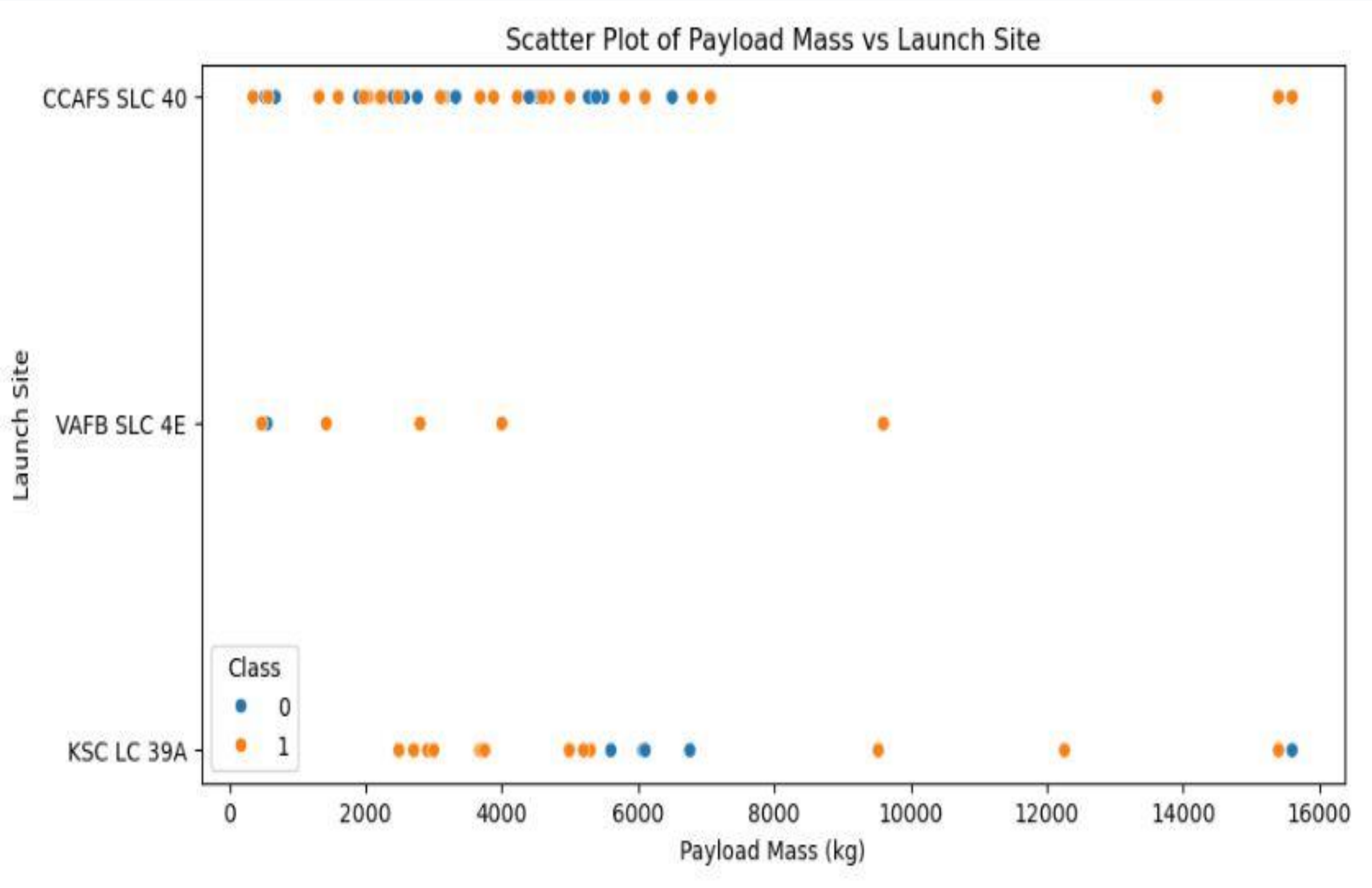
Insights drawn from EDA

Flight Number vs. Launch Site



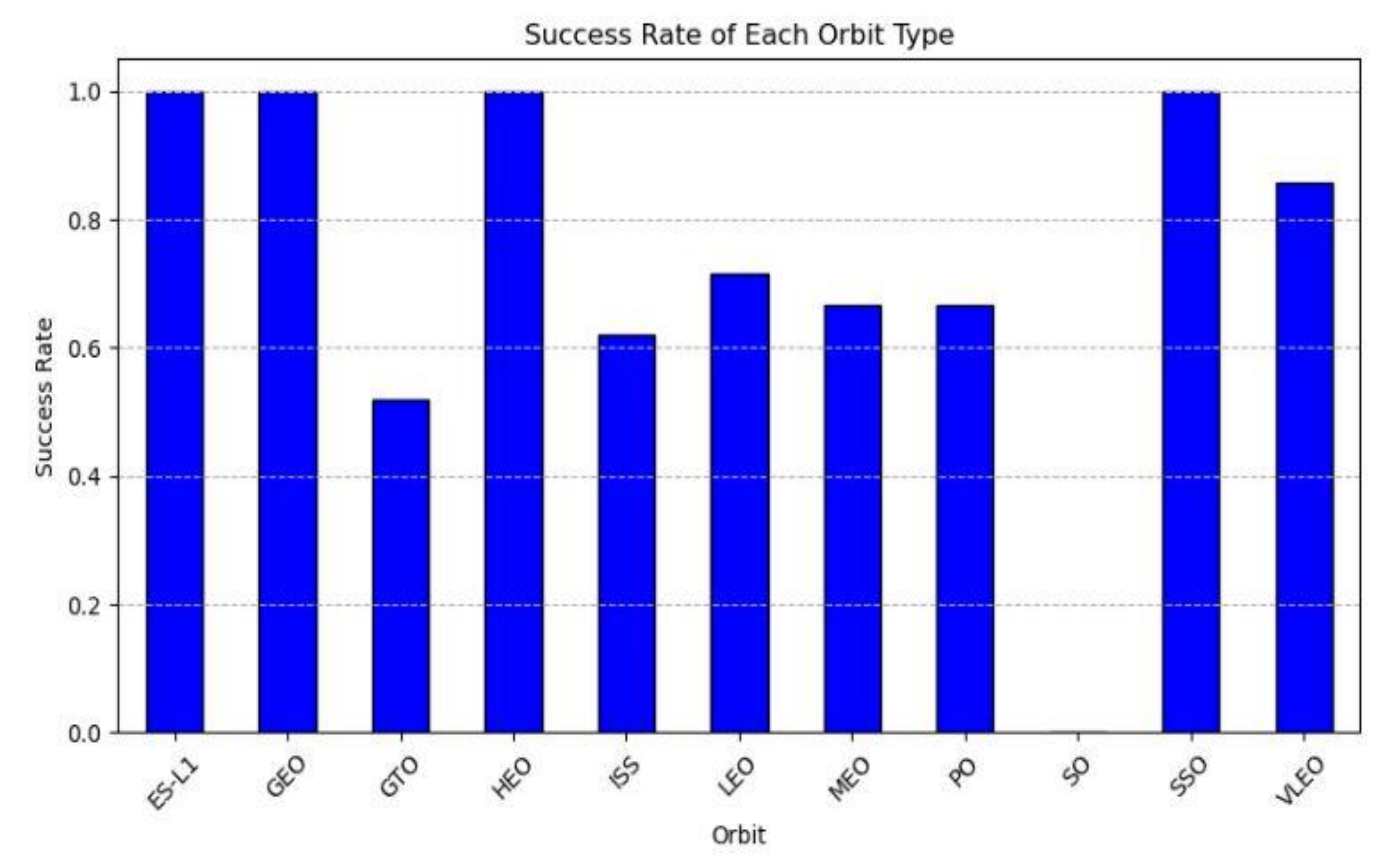
- In general, the greater the Flight number, the higher is the Success rate
- While the above relationship holds true across all launch sites, the correlation is relatively weaker at KSC LC-39A launch site

Payload vs. Launch Site



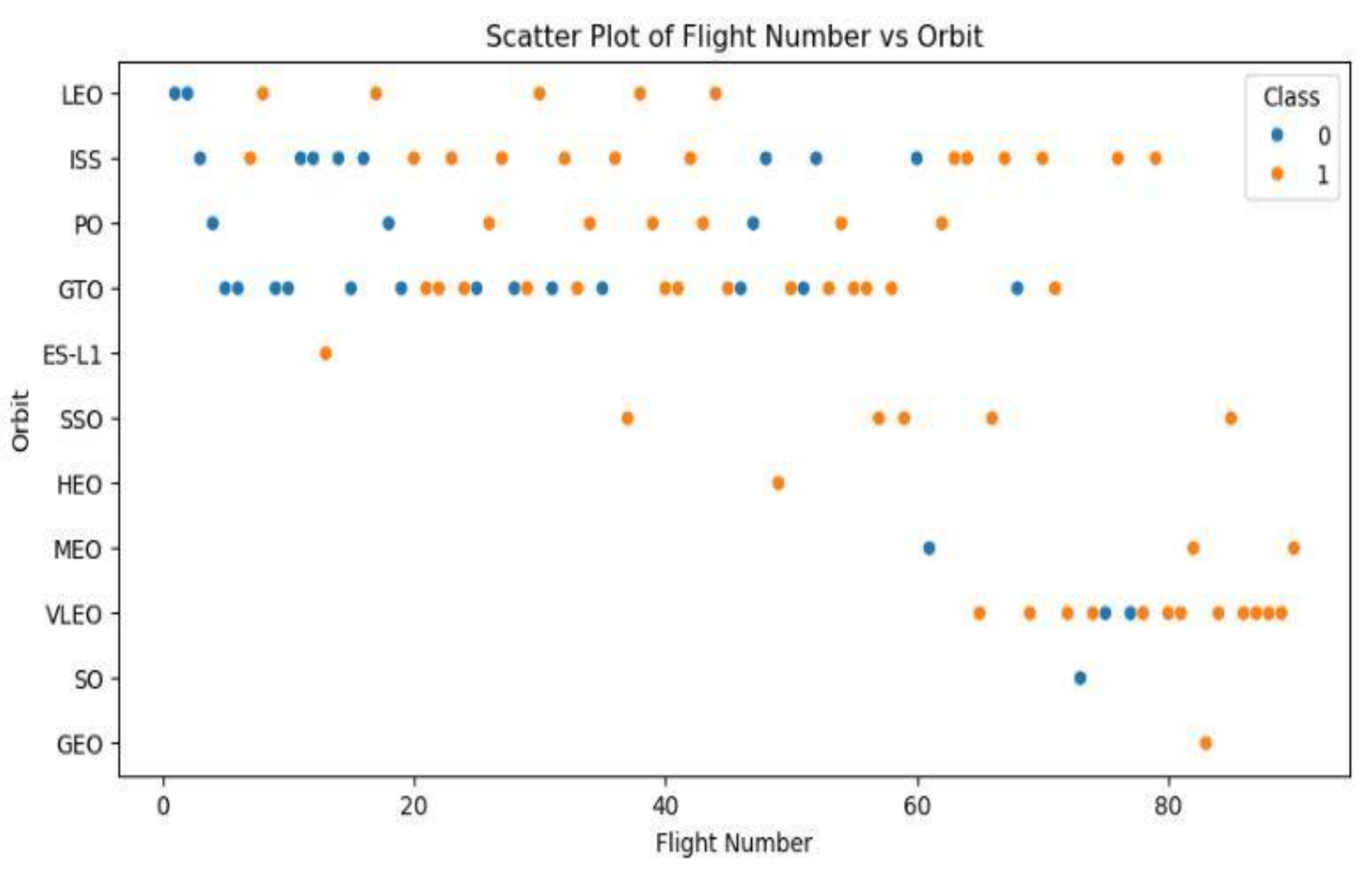
- The relationship between Payload and Succes rate differs across the launch sites
- At CCAFS SLC-40 & VAFB SLC-4E, the greater the Payload, the higher is the Success rate
- At KSC LC-39A site, there is no clear correlation between Payload and Success rate

Success Rate vs. Orbit Type



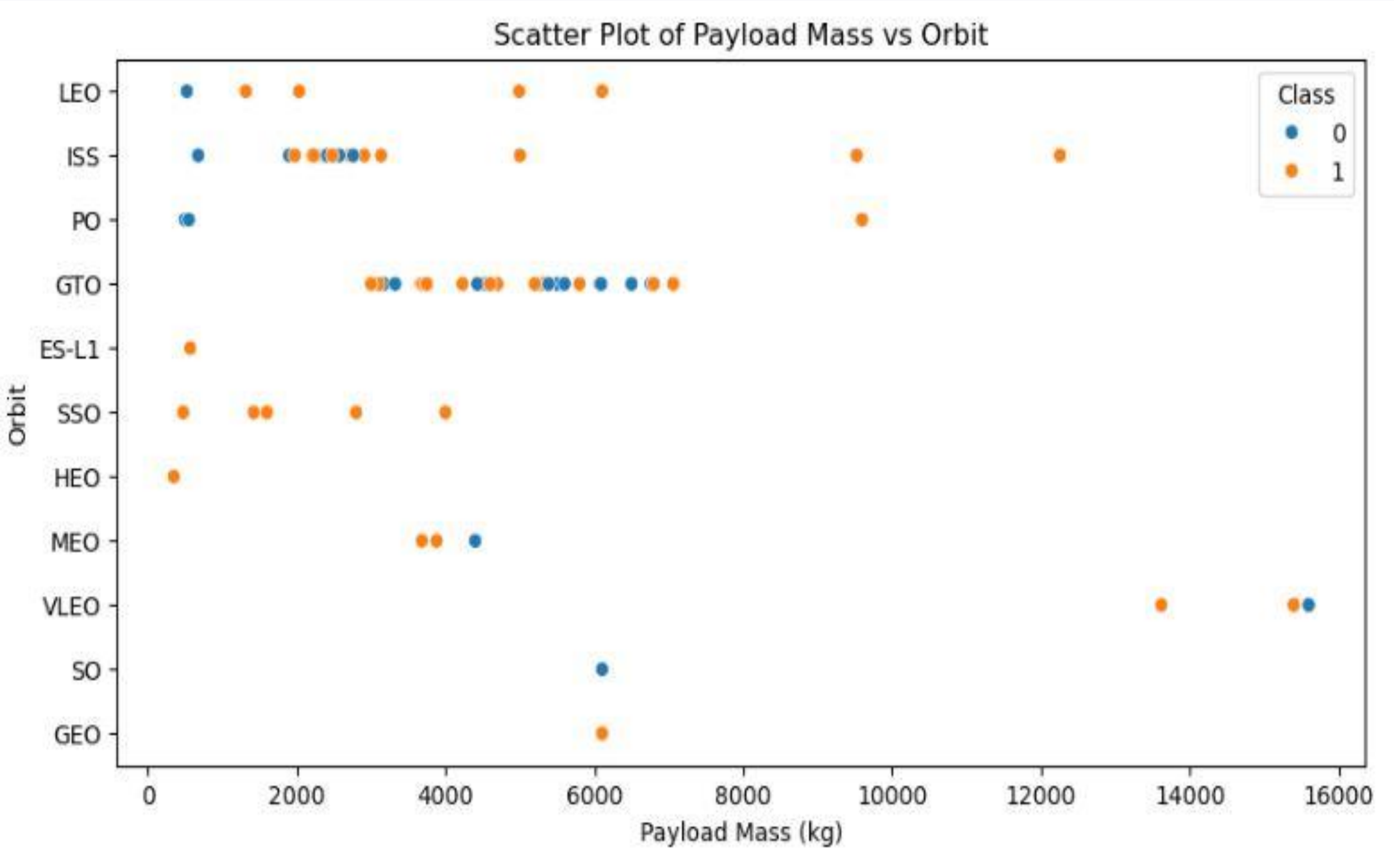
- ES-L1, GEO, HEO, SSO have the highest Success Rate (i.e., 100%), followed by VLEO
- GTO Orbit type has the lowest Success rate

Flight Number vs. Orbit Type



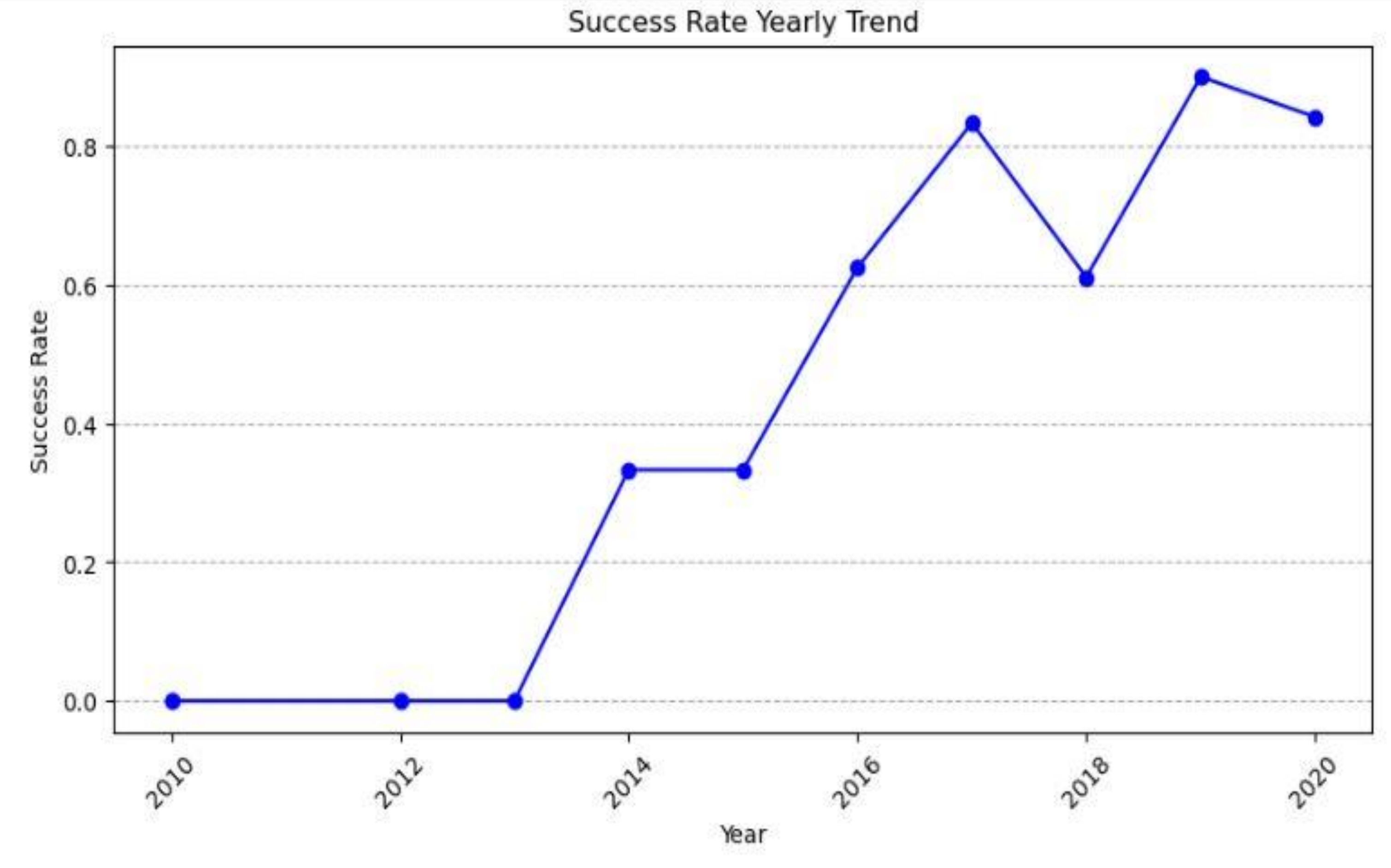
- For Orbit types LEO, ISS, PO, MEO, VLEO, greater Flight number corresponds with higher Success rate
- The correlation between Flight number and Success rate is less clear for GTO
- Other Orbit types have either only 1 data point or 100% Success rate, from which a Correlation trend cannot be deduced

Payload vs. Orbit Type



- For Orbit types LEO, ISS, PO, greater Payload corresponds with higher Success rate
- For Orbit types MEO and VLEO, greater Payload corresponds with lower Success rate
- The correlation between Payload and Success rate is less clear for GTO
- Other Orbit types have either only 1 data point or 100% Success rate, from which a Correlation trend cannot be deduced

Launch Success Yearly Trend



- Since 2013, the Success rate has generally been trending upwards

All Launch Site Names

```
query1 = "SELECT DISTINCT Launch_Site FROM SPACEXTABLE;"
cur.execute(query1)

launch_sites = cur.fetchall()
for site in launch_sites:
    print(site[0])  # Print each unique launch site
```

```
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
```

- Unique Launch Site names are CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, and CCAFS SLC-40
- The SQL query uses '**DISTINCT**' to show only ***the unique values*** of Launch_Site column from SPACEXTABLE

Launch Site Names Begin with 'CCA'

```
query2 = "SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;"
cur.execute(query2)

records = cur.fetchall()
for record in records:
    print(record)
```

```
('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')
```

- The SQL query uses '**LIKE 'CCA%**'' to extract records where Launch_Site names begin with 'CCA', and '**LIMIT 5**' to display only 5 such records

Total Payload Mass

```
query3 = """
SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass
FROM SPACEXTABLE
WHERE Customer = 'NASA (CRS)';
"""
cur.execute(query3)
total_payload_mass = cur.fetchone()[0]
print(f"Total Payload Mass for NASA (CRS): {total_payload_mass} kg")
```

```
Total Payload Mass for NASA (CRS): 45596 kg
```

- The SQL query uses '**SUM**' to sum up the Payload Mass values of the selected records, and '**WHERE Customer = 'NASA (CRS)'**' to select only records where the customer is NASA

Average Payload Mass by F9 v1.1

```
query4 = """
SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass
FROM SPACEXTABLE
WHERE Booster_Version = 'F9 v1.1';
"""
cur.execute(query4)
average_payload_mass = cur.fetchone()[0]
print(f"Average Payload Mass for F9 v1.1: {average_payload_mass} kg")
```

```
Average Payload Mass for F9 v1.1: 2928.4 kg
```

- The SQL query uses ‘**AVG**’ to compute the average of Payload Mass values of the selected records, and ‘**WHERE Booster_Version = ‘F9 v1.1’**’ to select only records where the Booster_Version is ‘F9 v1.1’

First Successful Ground Landing Date

```
query5 = """
SELECT MIN(Date) AS First_Successful_Landing
FROM SPACEXTABLE
WHERE Landing_Outcome LIKE '%ground pad%' AND Mission_Outcome = 'Success';
"""
cur.execute(query5)
first_successful_landing = cur.fetchone()[0]
print(f"First successful landing on ground pad: {first_successful_landing}")
```

```
First successful landing on ground pad: 2015-12-22
```

- The SQL query uses '**MIN(Date)**' to *extract the earliest date* '**WHERE Landing_Outcome LIKE '%ground pad%'** AND '**Mission_Outcome = 'Success''** (i.e., successful landing on ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000

```
query6 = """
SELECT Booster_Version
FROM SPACEXTABLE
WHERE Landing_Outcome LIKE '%drone ship%'
AND Mission_Outcome = 'Success'
AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
"""
cur.execute(query6)
boosters = cur.fetchall()
for booster in boosters:
    print(booster[0])
```

```
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT  B1021.2
F9 FT  B1031.2
```

- The SQL query uses '**SELECT Booster_Version**' to *extract the Booster models* that fulfill the conditions specified by '**WHERE**', namely **Landing_Outcome LIKE '%drone ship%'** AND '**Mission_Outcome = 'Success''** AND '**PAYLOAD_MASS_KG_BETWEEN 4000 AND 6000**'

Total Number of Successful and Failure Mission Outcomes

```
query7 = """
SELECT Mission_Outcome, COUNT(*) AS Total_Count
FROM SPACEXTABLE
GROUP BY Mission_Outcome;
"""
cur.execute(query7)
mission_outcomes = cur.fetchall()
for outcome, count in mission_outcomes:
    print(f"{outcome}: {count} missions")
```

```
Failure (in flight): 1 missions
Success: 98 missions
Success : 1 missions
Success (payload status unclear): 1 missions
```

- The SQL query uses '**GROUP BY Mission_Outcome**' to categorize all missions by their Mission Outcome, and '**COUNT(*)**' to count how many times each unique Mission Outcome occurs

Boosters That Carried Maximum Payload

```
query8 = """
SELECT Booster_Version
FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
"""
cur.execute(query8)
boosters = cur.fetchall()
for booster in boosters:
    print(booster[0])
```

```
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 main (outer) query determines the output column or variable (i.e. '**SELECT Booster_Version**') and the condition that selected records must fulfill (i.e., **WHERE PAYLOAD_MASS_KG_ =**)
- The inner subquery **determines the maximum value in PAYLOAD_MASS_KG_ column** to feed into the condition set by main query

2015 Launch Records

```
query9 = """
SELECT
    CASE substr(Date, 6, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS Month_Name,
    Booster_Version,
    Launch_Site,
    Landing_Outcome
FROM SPACEXTABLE
WHERE substr(Date, 0, 5) = '2015'
AND Landing_Outcome LIKE '%drone ship%'
AND Landing_Outcome LIKE '%Failure%';
"""
cur.execute(query9)
records9 = cur.fetchall()
for record in records9:
    print(record)
```

```
('January', 'F9 v1.1 B1012', 'CCAFS LC-40', 'Failure (drone ship)')
('April', 'F9 v1.1 B1015', 'CCAFS LC-40', 'Failure (drone ship)')
```

- The SQL query uses '**substr(Date, 6, 2)**' to extract the numeric value of month (i.e., the 6th & 7th characters of the string in 'Date' column) and **CASE** statement convert the numeric value extracted into month name
- The query uses '**substr(Date, 0, 5) = '2015'**' AND '**Landing_Outcome LIKE '%drone ship%'**' AND '**Landing_Outcome LIKE '%Failure%'**' to filter for failed landings on Drone Ship in 2015

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

```
query10 = """
SELECT Landing_Outcome, COUNT(*) AS Outcome_Count
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY Outcome_Count DESC;
"""
cur.execute(query10)
landing_outcomes = cur.fetchall()
for outcome, count in landing_outcomes:
    print(f"{outcome}: {count} landings")
```

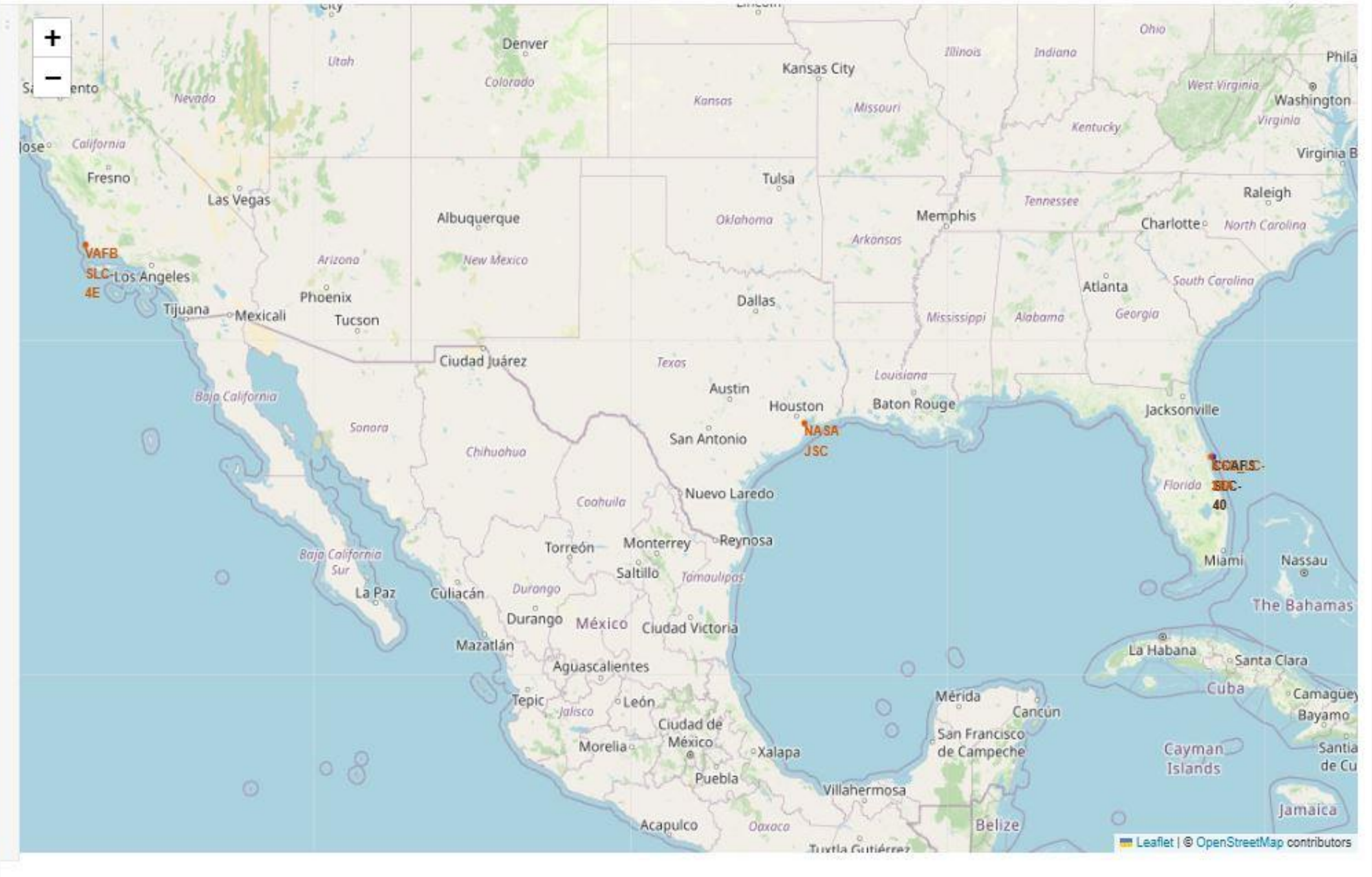
```
No attempt: 10 landings
Success (drone ship): 5 landings
Failure (drone ship): 5 landings
Success (ground pad): 3 landings
Controlled (ocean): 3 landings
Uncontrolled (ocean): 2 landings
Failure (parachute): 2 landings
Precluded (drone ship): 1 landings
```

- The query uses '**WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'**' to filter for only landings that occurred between 4-Jun-2010 and 20-Mar-2017
- The query uses '**COUNT(*)**' and '**GROUP BY Landing_Outcome**' to aggregate data (count) by unique landing outcomes, and ensure that each different landing outcome is counted separately
- The query uses '**ORDER BY Outcome_Count DESC**' to sort the results in descending order

Section 3

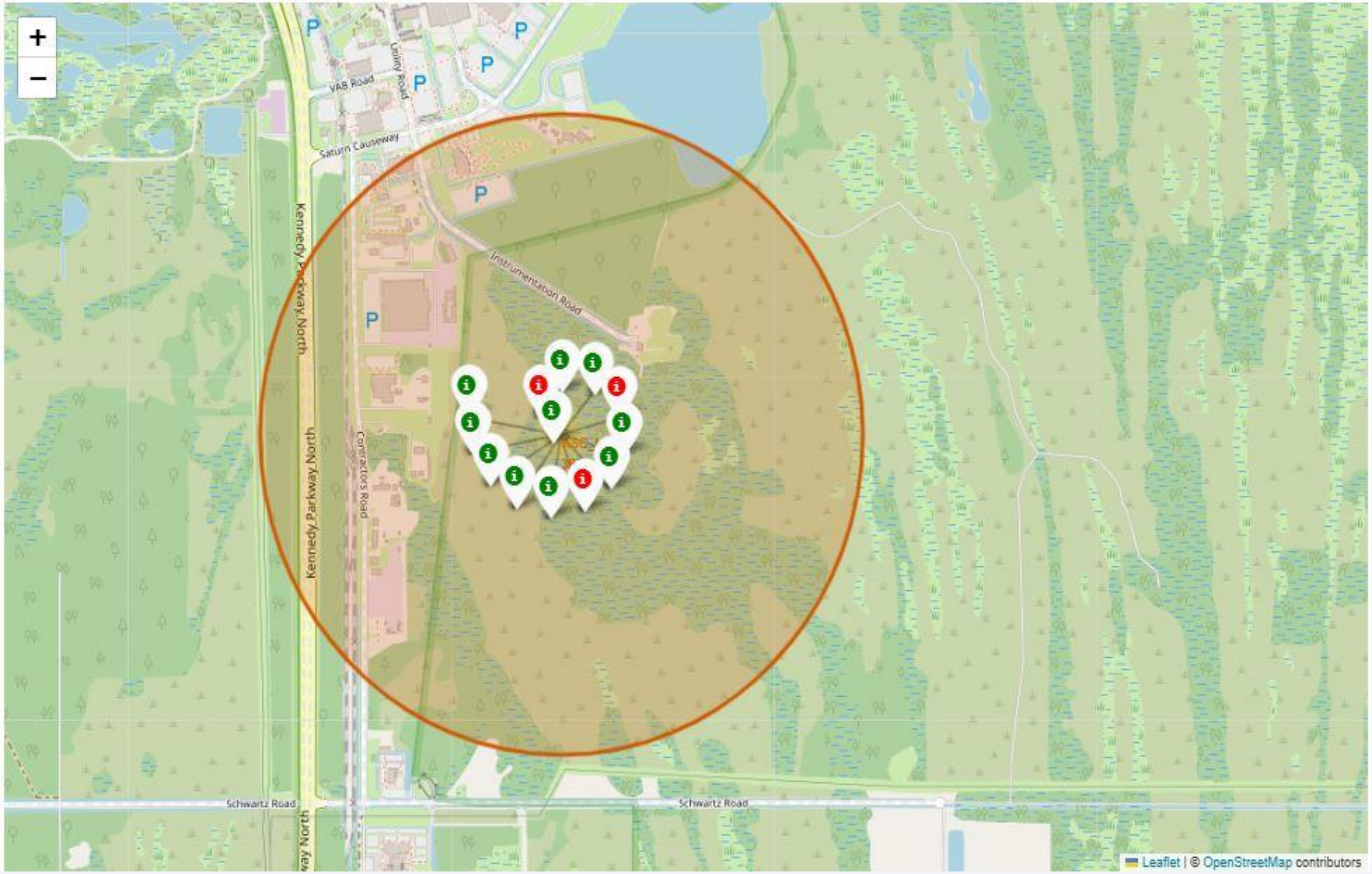
Launch Sites Proximities Analysis

Mark All Launch Sites on Folium Map



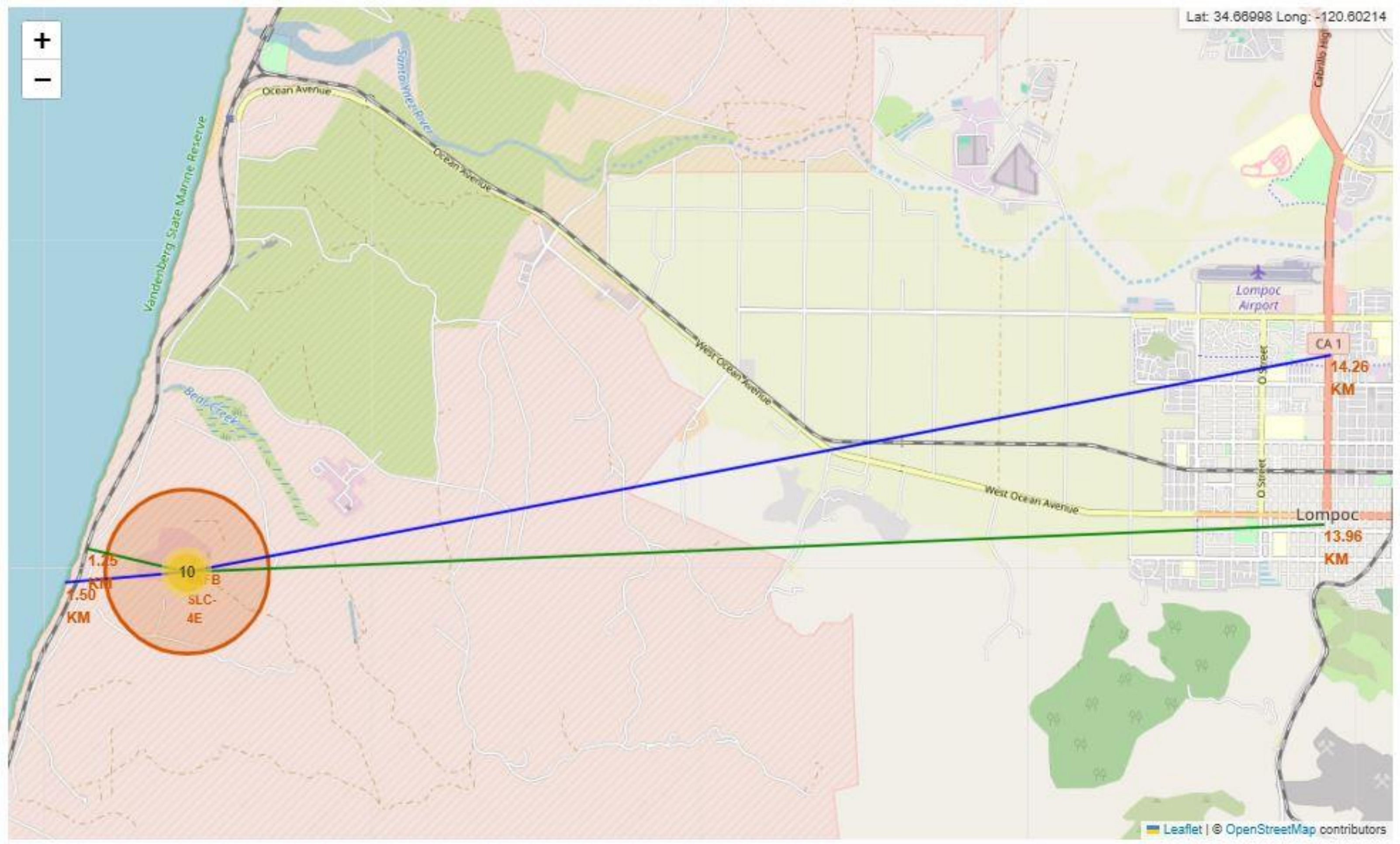
- The map screenshot on the left shows the locations of 4 launch sites, with 3 of them (CCAFS SLC-40, CCAFS LC-40, KSC LC-39A) in Florida, and 1 of them (VAFB SLC-4E) in California
- NASA Johnson Space Center (Texas) is also shown on the map, between the California site and Florida cluster

Mark the Successful & Failed Launches on Each Site



- The map screenshot on the left shows the successful (green, 10 markers) and failed (red, 3 markers) launches at KSC LC-39A
- The orange circle denotes area within the radius of 1,000m of KSC LC-39A

Distance from Launch Sites to Nearby Infrastructure

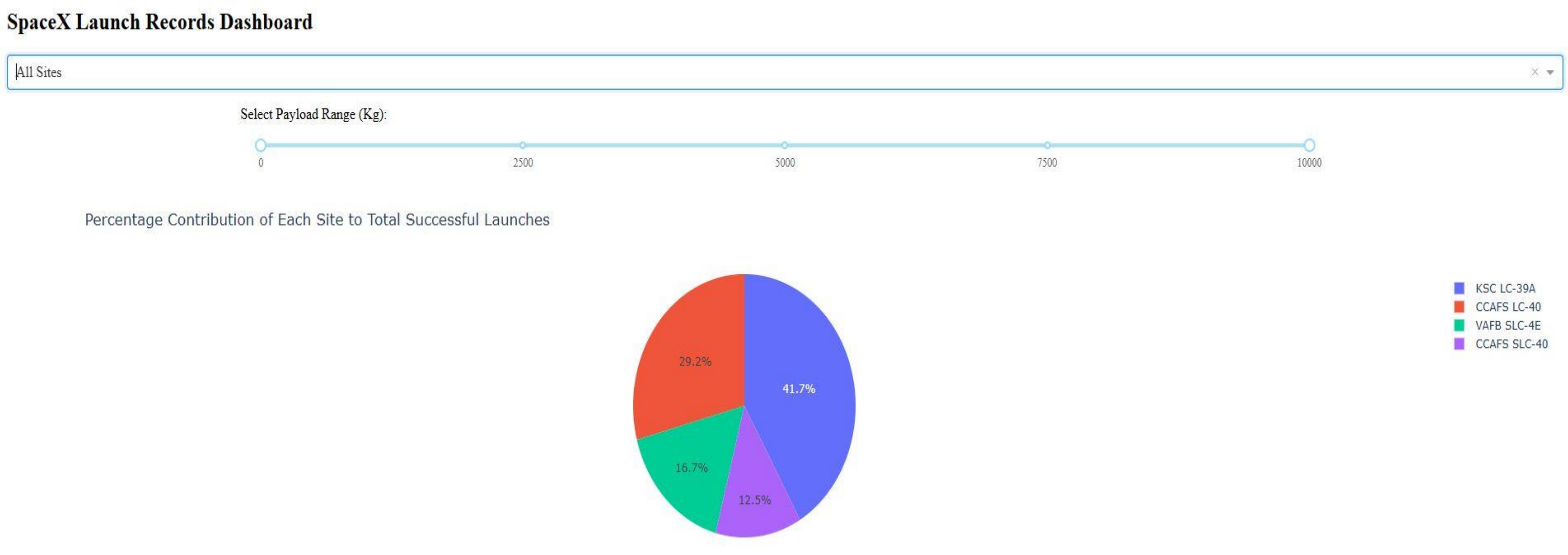


- The map screenshot on the left shows the distance from VAFB SLC-4E launch site to the nearest coastline, railway, highway (CA1), and city (Lompoc)
- Polylines are drawn to connect the launch site to each of the landmarks, with distances between them calculated using Haversine formula

Section 4

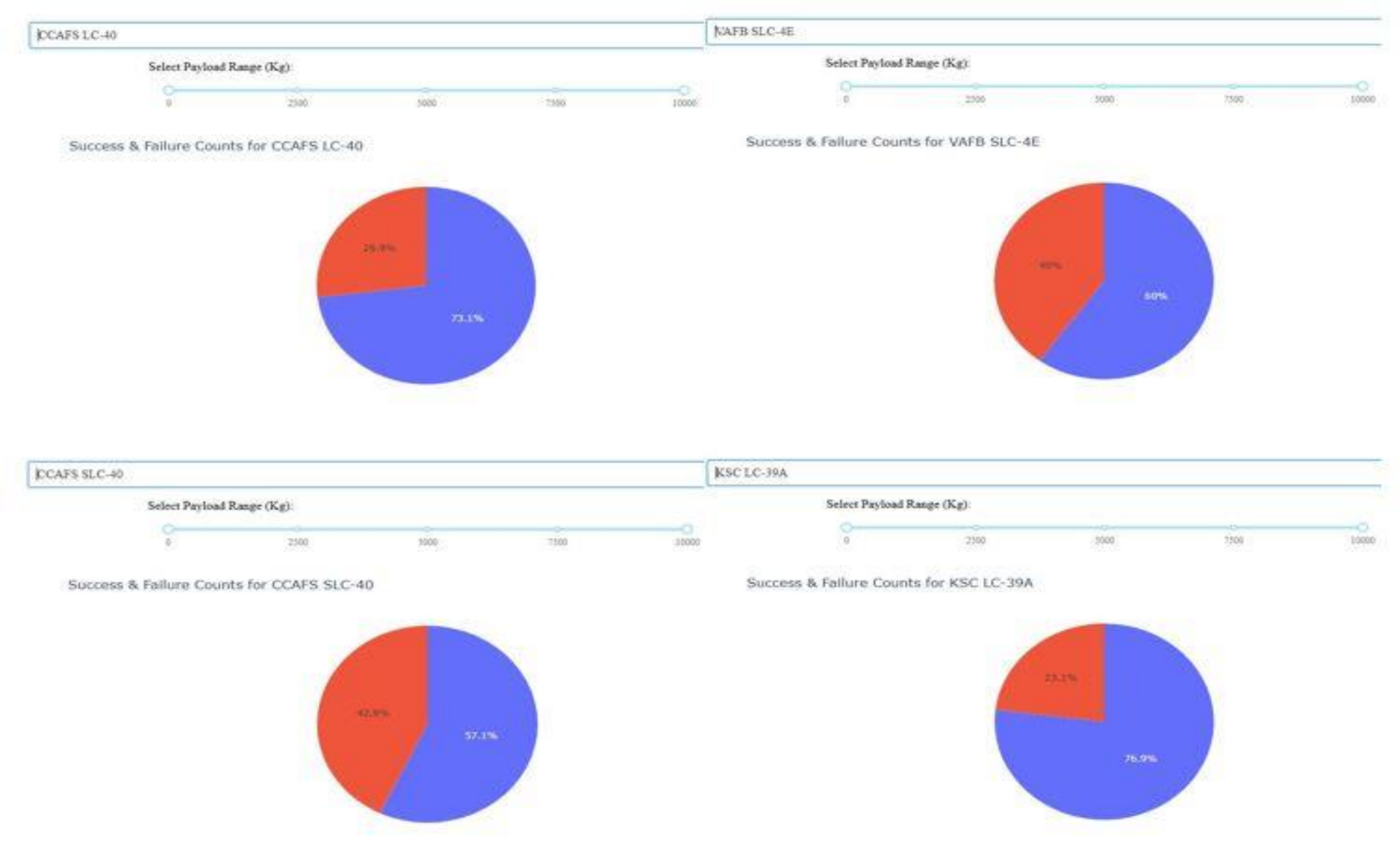
Build a Dashboard with Plotly Dash

Dashboard: % Contribution of Each Site to Total Successful Launches



- The pie chart shows KSC LC-39A as the largest contributor to successful launches
- The % contribution is calculated as count of successful launches at a site divided by sum of successful launch counts at all sites

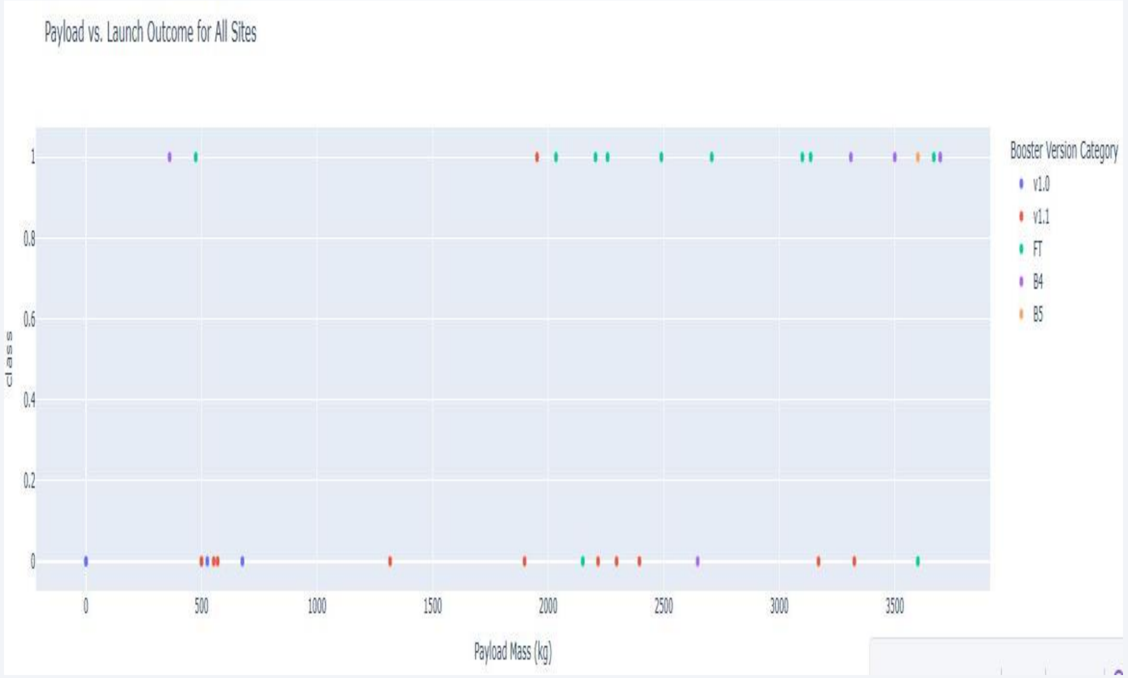
Dashboard: Launch Site with the Highest Success Rate



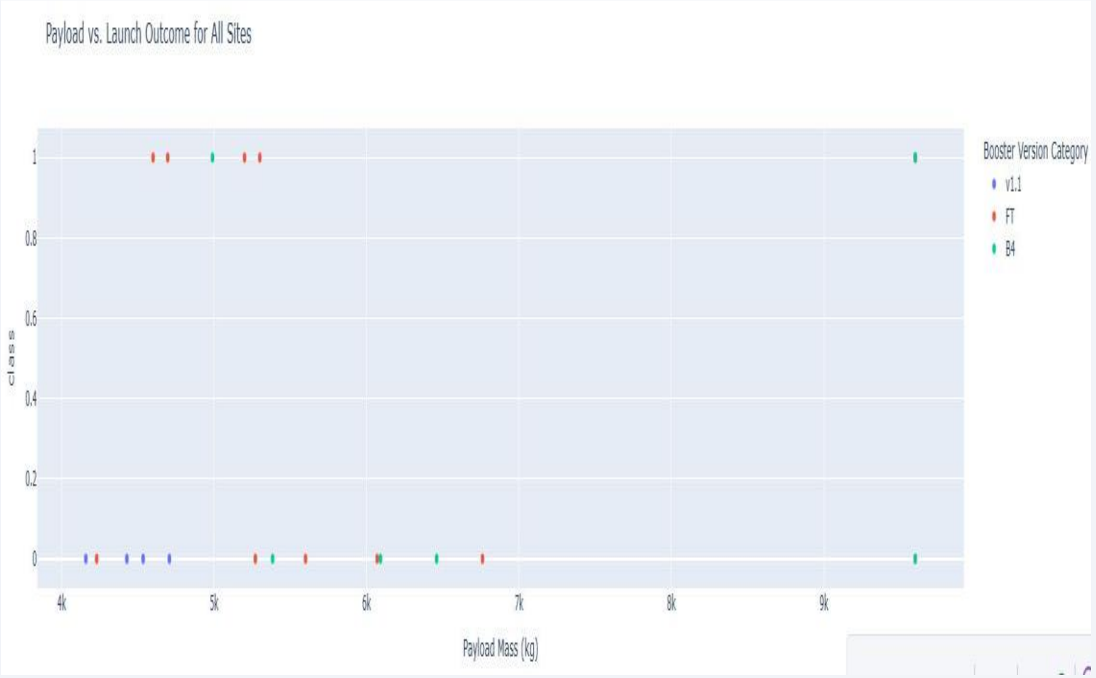
- KSC LC-39A has the highest Success rate at 76.9%
- Purple denotes successful launches and Orange denotes failed launches in the Pie chart

Dashboard: Payload vs. Launch Outcome Scatter Plot

Payload range: 0 to 4,000 kg



Payload range: 4,000 to 10,000 kg

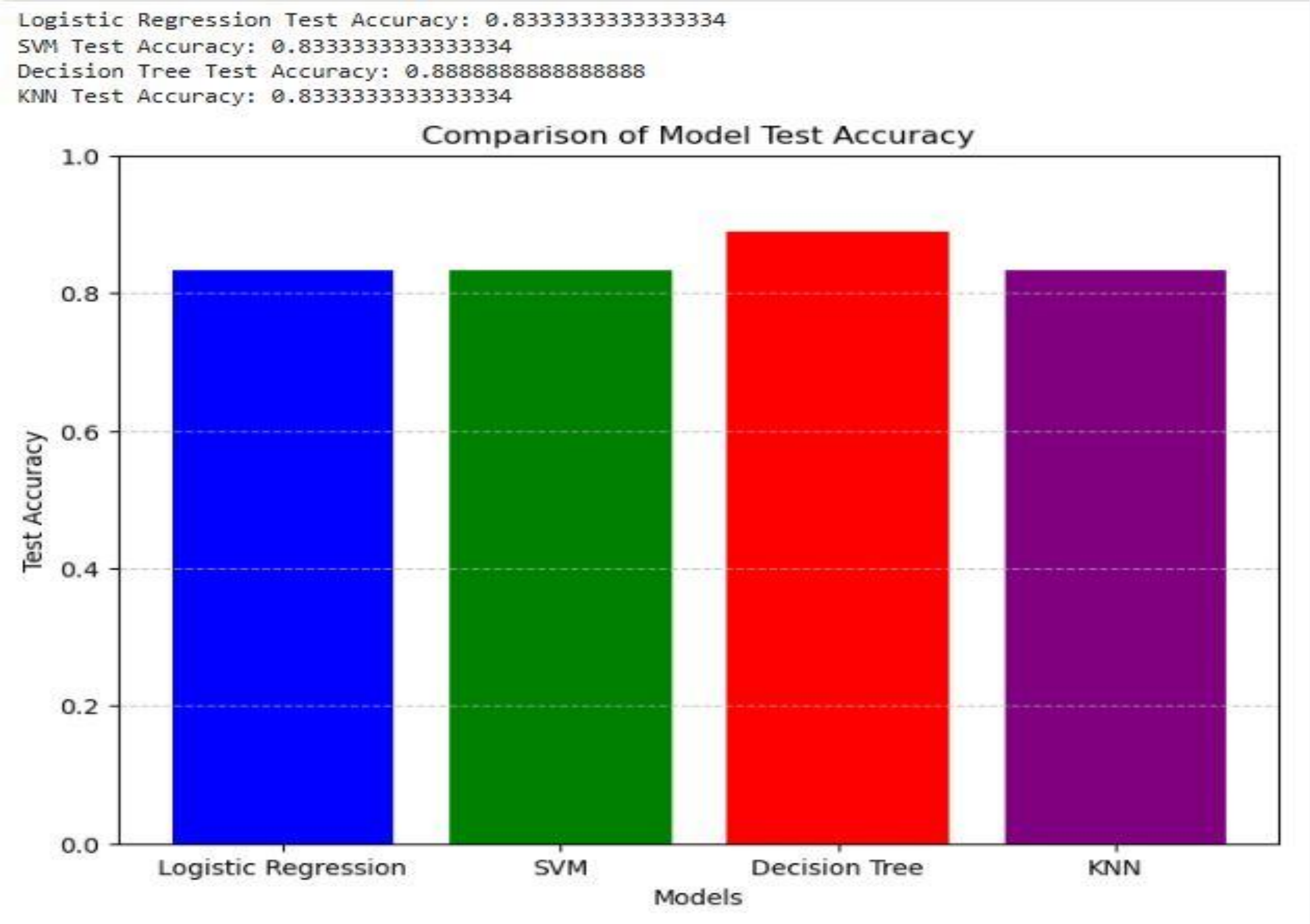


- The success rate for lower Payload range is higher than the success rate for higher Payload range

Section 5

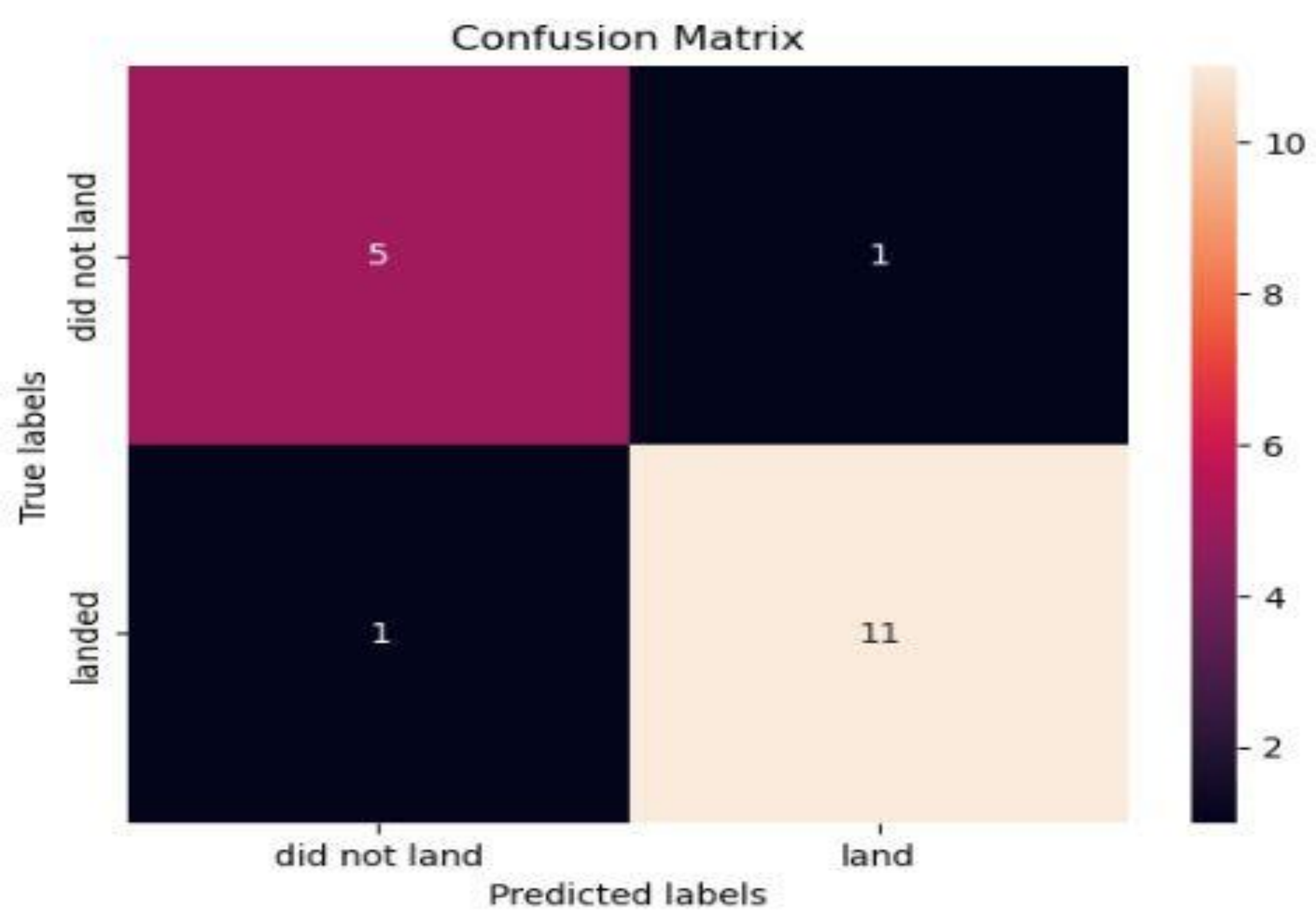
Predictive Analysis (Classification)

Classification Accuracy



- The Decision Tree Classifier has the highest accuracy score on the test data i.e., 88.9%, and is therefore the best-performing model in this study

Confusion Matrix of Best Performing Model



- Confusion Matrix of the best performing model, Decision Tree Classifier, shows 11 True Positives & 5 True Negatives, for a total of 16 correct classifications out of 18 test data points
- The above outperforms the results of the other 3 models, each of which shows 12 True Positives & 3 True Negatives i.e., 15 correct classifications out of 18 test data points

Conclusions

- The Decision Tree Classifier has the highest accuracy score on the test data i.e., 88.9%, and is therefore the best-performing model in predicting the success or failure of Falcon 9 First Stage Landing
- In general, the greater the Flight number, the higher is the Success rate
- Overall, the success rate for lower Payload range (0 – 4,000 kg) is higher than the success rate for higher Payload range (4,000 – 10,000 kg). There are, however, exceptions to this observation, with specific Orbit types (LEO, ISS, PO) showing higher success rate with greater Payload, and GTO Orbit type showing less clear correlation between Payload and success rate
- Since 2013, the success rate has generally been trending upwards

Appendix

- dataset_part_1.csv:
  https://github.com/yusua2201/YARepo/blob/main/dataset_part_1.csv
- dataset_part_2.csv:
  https://github.com/yusua2201/YARepo/blob/main/dataset_part_2.csv
- dataset_part_3.csv:
  https://github.com/yusua2201/YARepo/blob/main/dataset_part_3.csv
- IBM DS Capstone_Winning Space Race with Data Science.pdf:

Thank you!