IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Marcelo Herrera
01/09/2024

Outline

Executive Summary

- Summary of methodologies
  - ❑ Data Collection via API, Web Scraping
  - ❑ Exploratory Data Analysis (EDA) with Data Visualization
  - ❑ EDA with SQL
  - ❑ Interactive Map with Folium
  - ❑ Dashboards with Plotly Dash
  - ❑ Predictive Analysis
- Summary of all results
  - ❑ Exploratory Data Analysis results
  - ❑ Interactive maps and dashboard
  - ❑ Predictive results

Introduction

- Project background and context
- The purpose of this project is to forecast whether the Falcon 9's first stage will achieve a successful landing. SpaceX states that launching a Falcon 9 costs 62 million dollars, whereas other providers charge over 165 million dollars per launch. This cost discrepancy arises from SpaceX's capability to reuse the first stage of the rocket. By predicting the landing outcome, we can assess the potential cost of a launch. This data could be crucial for any company aiming to rival SpaceX in the space launch sector.
- Problems you want to find answers

1. What are the key factors that distinguish a successful landing from an unsuccessful one?
2. How does the interaction between different rocket variables influence the likelihood of a landing succeeding or failing?
3. What specific conditions must be met for SpaceX to maximize the success rate of its rocket landings?

Section 1

Methodology

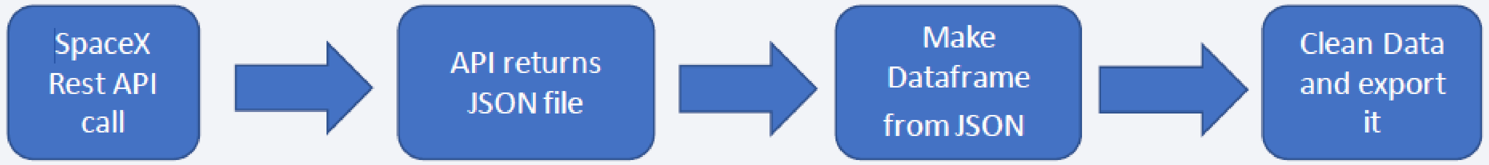
Methodology

Executive Summary

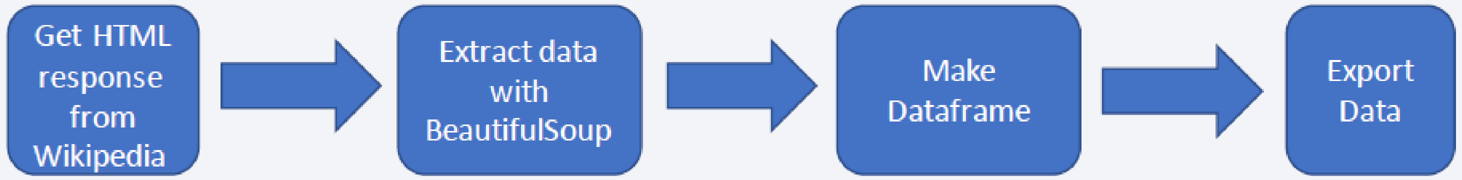
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Datasets are collected from Rest SpaceX API and webscrapping Wikipedia
  - The information obtained by the API are rocket, launches, payload information.
  - The Space XREST API URL is api.spacexdata.com/v4/



- The information obtained by the webscrapping of Wikipedia are launches, landing, payload information.
  - URL is https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922


Get HTML response from Wikipedia
Extract data with BeautifulSoup
Make Dataframe
Export Data

Data Collection – SpaceX API

1. Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

2. Convert Response to JSON File

```
data = response.json()
data = pd.json_normalize(data)
```

3. Transform data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
getBoosterVersion(data)
```

4. Create dictionary with data

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5. Create dataframe

```
data = pd.DataFrame.from_dict(launch_dict)
```

6. Filter dataframe

```
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

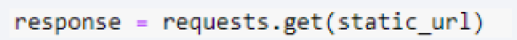
7. Export to file

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

https://github.com/Pugyuru/IBM-Applied-Data-Science-Capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

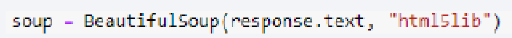
Data Collection - Scraping

1. Getting Response from HTML



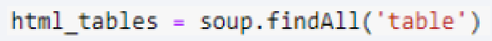
```
response = requests.get(static_url)
```

2. Create BeautifulSoup Object



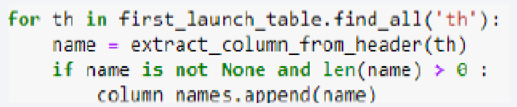
```
soup = BeautifulSoup(response.text, "html5lib")
```

3. Find all tables



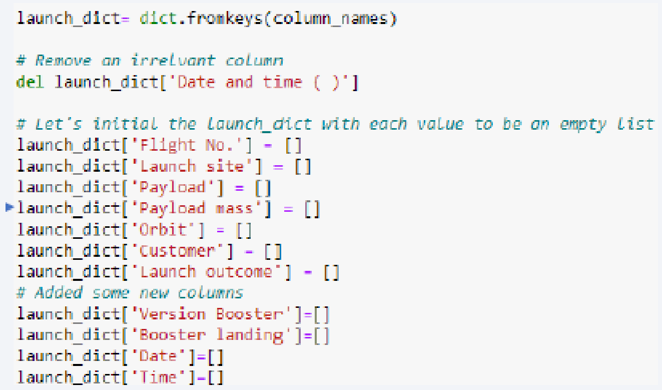
```
html_tables = soup.findAll('table')
```

4. Get column names



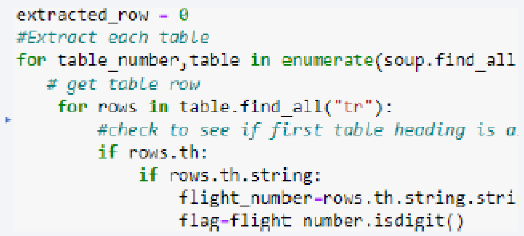
```
for th in first_launch_table.find_all('th'):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0 :
        column_names.append(name)
```

5. Create dictionary



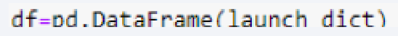
```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Add data to keys



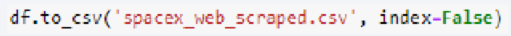
```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is a
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.stri
                flag=flight_number.isdigit()
```

7. Create dataframe from dictionary



```
df=pd.DataFrame(launch_dict)
```

8. Export to file

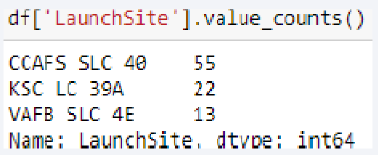


```
df.to_csv('spacex_web_scraped.csv', index=False)
```

https://github.com/Pugyuru/IBM-Applied-Data-Science-Capstone/blob/main/jupyter-labs-webscraping.ipynb

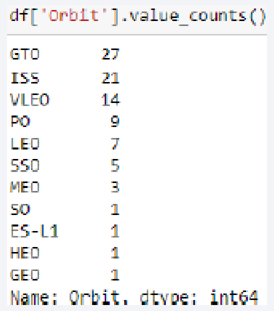
Data Wrangling

- ❑ Calculate launches number for each site



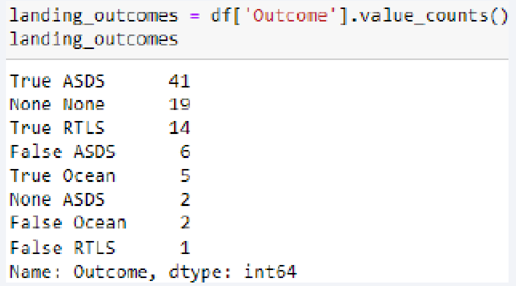
```
df['LaunchSite'].value_counts()

CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

- ❑ Calculate the number and occurence of each orbit



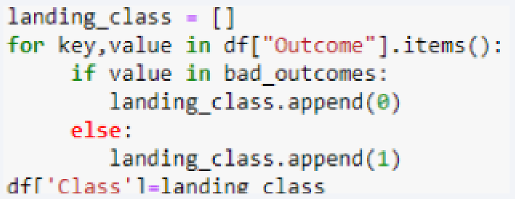
```
df['Orbit'].value_counts()

GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
SO        1
ES-L1     1
HEO       1
GEO       1
Name: Orbit, dtype: int64
```

- ❑ Calculate number and occurrence of mission outcome per orbit type



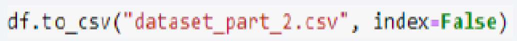
```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes

True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
None ASDS       2
False Ocean     2
False RTLS      1
Name: Outcome, dtype: int64
```

- ❑ Create landing outcome label from Outcome column



```
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
df['Class']=landing_class
```

- ❑ Export to file



```
df.to_csv("dataset_part_2.csv", index=False)
```

https://github.com/Pugyuru/IBM-Applied-Data-Science-Capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Scatter Graphs
  - ✓ Flight Number vs. Payload Mass
  - ✓ Flight Number vs. Launch Site
  - ✓ Payload vs. Launch Site
  - ✓ Orbit vs. Flight Number
  - ✓ Payload vs. Orbit Type
  - ✓ Orbit vs. Payload Mass
- Bar Graph
  - ✓ Success rate vs. Orbit
- Line Graph
  - ✓ Success rate vs. Year

https://github.com/Pugyuru/IBM-Applied-Data-Science-Capstone/blob/main/edadataviz.ipynb

EDA with SQL

https://github.com/Pugyuru/IBM-Applied-Data-Science-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

- We conducted SQL queries to analyze data from the dataset, focusing on the following:

- Extracting the names of all unique launch sites used in the missions.

- Showing five entries where the launch site names start with 'CCA'.

- Calculating the total payload mass carried by boosters used in NASA's CRS missions.

- Finding the average payload mass for the F9 v1.1 booster version.

- Determining the date of the first successful landing on a ground pad.

- Listing the boosters that successfully landed on a drone ship with a payload mass between 4000 and 6000.

- Counting the total number of successful and failed mission outcomes.

- Identifying which booster versions have achieved the highest payload mass.

- Displaying records for the year 2015, including month names, failed landing outcomes on drone ships, booster versions, and launch sites.

- Ranking the number of successful landings from June 4, 2010, to March 20, 2017, in descending order.

Build an Interactive Map with Folium

- The Folium map object is configured to center on NASA's Johnson Space Center in Houston, Texas. It features:

  - A red circle at the coordinates of NASA Johnson Space Center, with a label displaying its name (using `folium.Circle` and `folium.Map`).

  - Red circles at each launch site's coordinates, with labels indicating the launch site names (using `folium.Circle`, `folium.Map.Marker`, and `folium.features.DivIcon`).

  - Clustering of points to aggregate and display multiple pieces of information for overlapping coordinates (using `folium.plugins`).

  - Markers distinguishing successful landings (green) from unsuccessful ones (red) (using `folium.Map.Marker` and `folium`).

  - Markers indicating the distances between launch sites and key locations (such as railways, highways, coastways, and cities), with lines drawn between these points (using `folium.Map.Marker`, `folium.PolyLine`, and `folium.features.DivIcon`).

- These visual elements are designed to enhance understanding of the data and the problem by clearly showing all launch sites, their surroundings, and the success rate of landings.

https://github.com/Pugyuru/IBM-Applied-Data-Science-Capstone/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- The dashboard includes several interactive components:
  - A dropdown menu that lets users select either a specific launch site or all launch sites (`dash_core_components.Dropdown`).
  - A pie chart that displays the total number of successful and failed launches for the selected launch site, based on the dropdown choice (`plotly.express.Pie`).
  - A range slider that enables users to filter payload mass within a specified range (`dash_core_components.RangeSlider`).
  - A scatter plot that illustrates the correlation between two variables, specifically the success rate compared to payload mass (`plotly.express.Scatter`).
- These components are designed to facilitate detailed analysis and visualization of launch data.

Predictive Analysis (Classification)

https://github.com/Pugyuru/IBM-Applied-Data-Science-Capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

- Data Preparation

- Load the Dataset: Import the dataset into the analysis environment.

- Normalize Data: Adjust the data to ensure consistent scaling across features.

- Split Data: Divide the dataset into training and test subsets.

- Model Preparation

- Select Algorithms: Choose appropriate machine learning algorithms for the task.

- Set Parameters: Define parameter ranges for each algorithm and use `GridSearchCV` to explore these.

- Train Models: Use the training data to fit the models specified in the `GridSearchCV`.

- Model Evaluation

- Determine Hyperparameters: Identify the optimal hyperparameters for each model type.

- Compute Accuracy: Evaluate the accuracy of each model using the test dataset.

- Plot Confusion Matrix: Visualize the confusion matrix to assess the performance of each model.

- Model Comparison

- Compare Accuracy: Evaluate and compare the accuracy of different models.

- Select Best Model: Choose the model with the highest accuracy (refer to the Notebook for detailed results).

Results

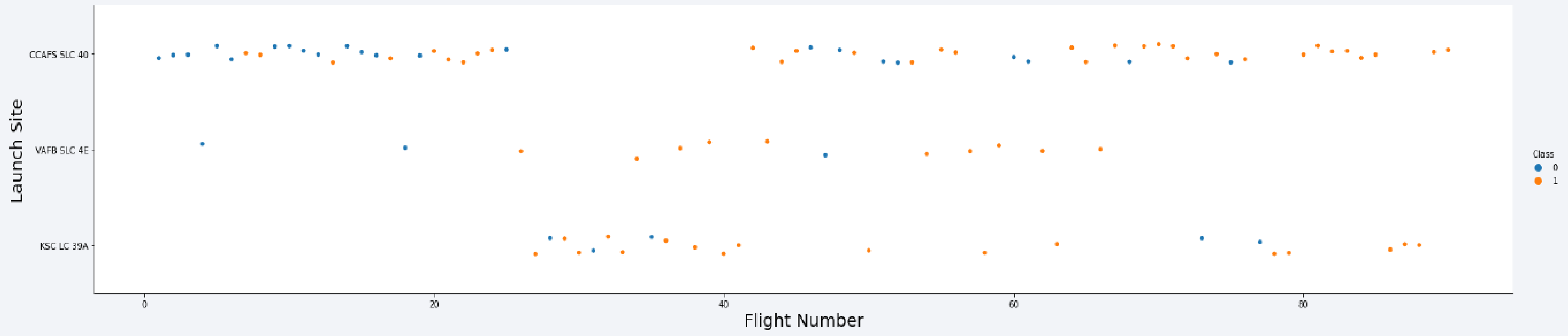
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

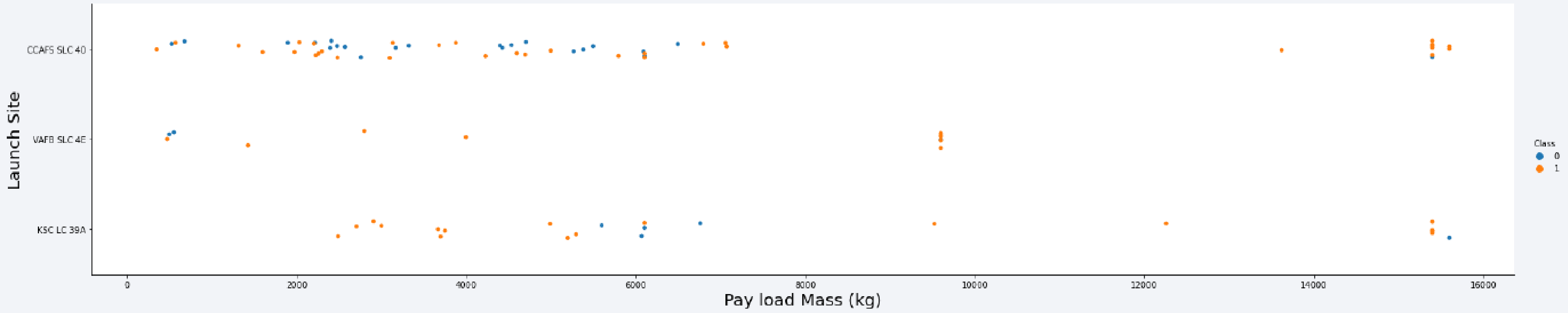
Insights drawn from EDA

Flight Number vs. Launch Site

For each site, the success rate is increasing.

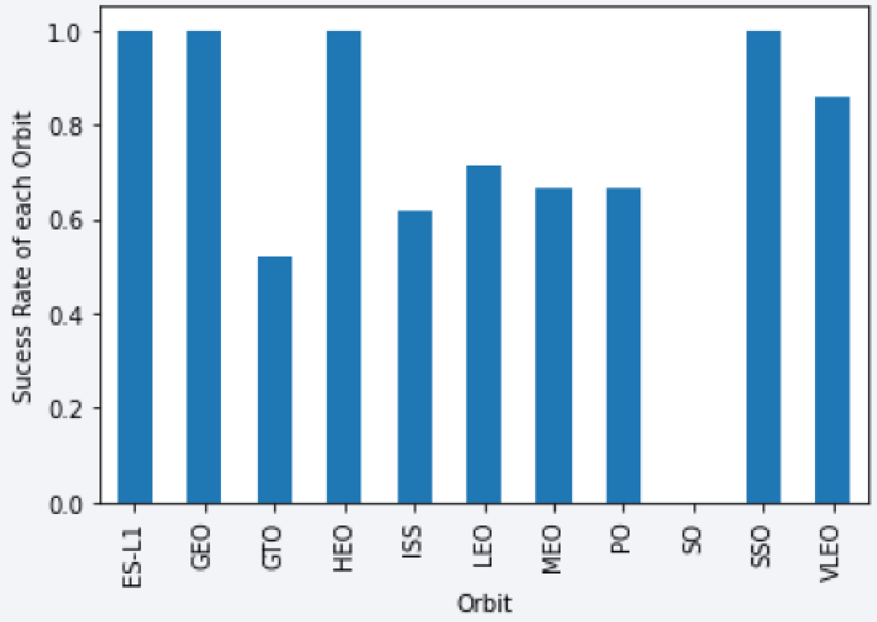


Payload vs. Launch Site



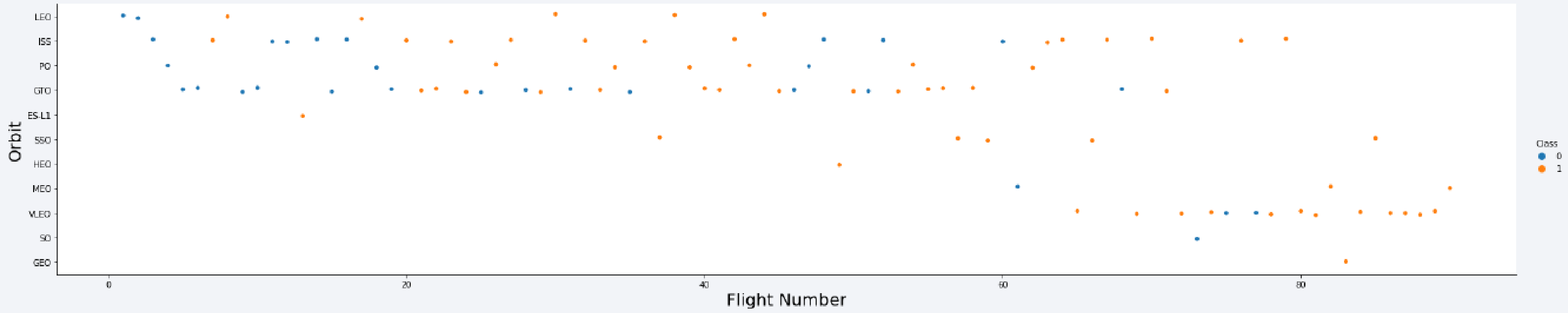
Depending on the launch site, the weight of the payload can impact the success of a landing. A heavier payload might be advantageous for achieving a successful landing at certain sites, as it can optimize performance and efficiency. However, if the payload is excessively heavy, it could lead to a failed landing due to the increased stress and challenges during the descent phase. Balancing the payload weight is crucial for maximizing landing success.

Success Rate vs. Orbit Type



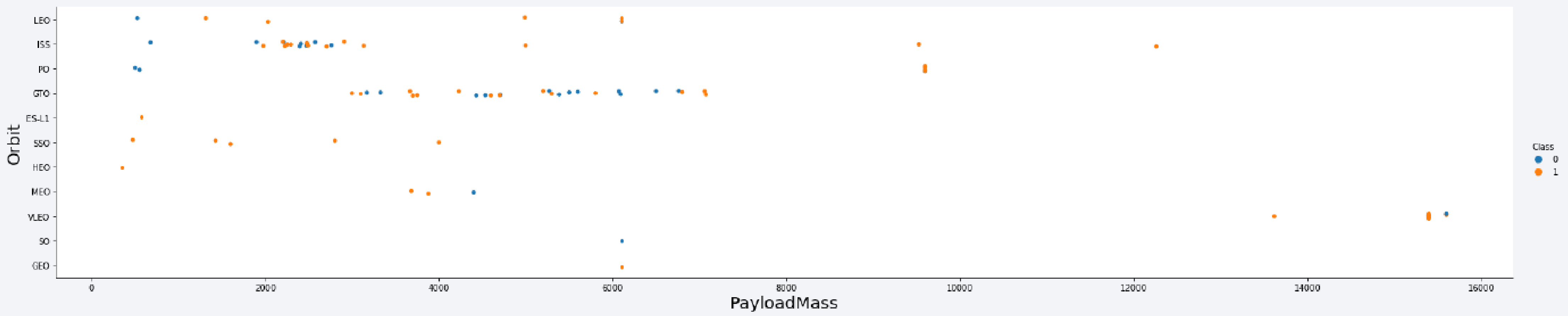
This plot allows us to observe the success rates across various orbit types. It reveals that ES L1, GEO, HEO, and SSO orbits exhibit the highest success rates.

Flight Number vs. Orbit Type



We observe that the success rate for LEO orbits tends to improve with an increasing number of flights. In contrast, for certain orbits such as GTO, there is no apparent relationship between success rate and the number of flights. However, it can be inferred that the high success rates for orbits like SSO and HEO may benefit from the insights gained through previous launches in other orbit types.

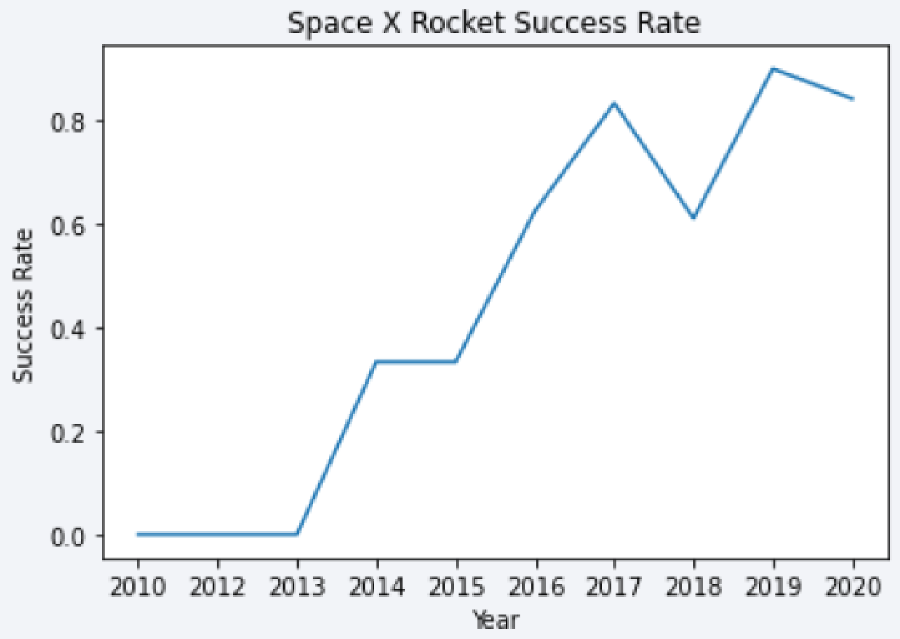
Payload vs. Orbit Type



The weight of the payload can significantly impact the success rate of launches in specific orbits. For instance, heavier payloads tend to enhance the success rate for LEO orbits. Conversely, reducing the payload weight for GTO orbits can improve the likelihood of a successful launch.

Launch Success Yearly Trend

Since 2013, there has been a noticeable increase in the success rate of SpaceX rockets.

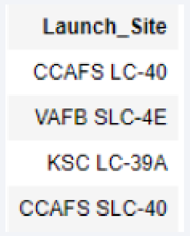


All Launch Site Names

SQL Query

```
SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

Results

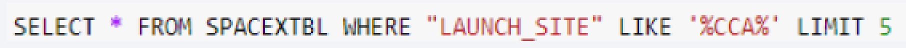


Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Using `DISTINCT` in the query helps eliminate duplicate entries for `LAUNCH_SITE`.

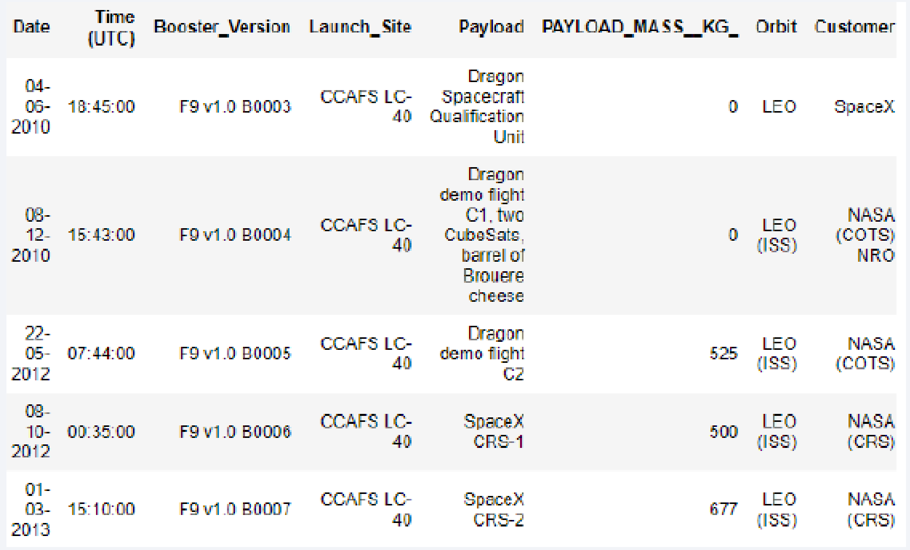
Launch Site Names Begin with 'CCA'

SQL Query



```
SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE '%CCA%' LIMIT 5
```

Results

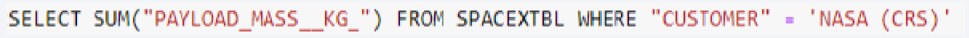


Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
08-12-2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)

The `WHERE` clause combined with the `LIKE` clause filters the launch sites to include only those containing the substring "CCA." The `LIMIT 5` command then restricts the results to just 5 records from this filtered list.

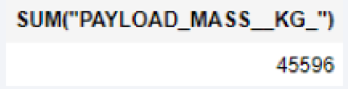
Total Payload Mass

SQL Query



```
SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

Results

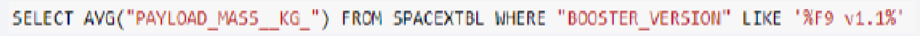


SUM("PAYLOAD_MASS__KG_")
45596

This query calculates the total sum of all payload masses for missions where the customer is NASA (CRS).

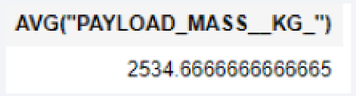
Average Payload Mass by F9 v1.1

SQL Query



```
SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "BOOSTER_VERSION" LIKE '%F9 v1.1%'
```

Results

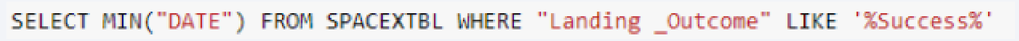


AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

This query calculates the average of all payload masses for missions where the booster version includes the substring "F9 v1.1."

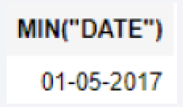
First Successful Ground Landing Date

SQL Query



```
SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing _Outcome" LIKE '%Success%'
```

Results

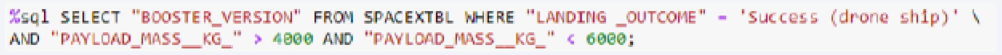


MIN("DATE")
01-05-2017

This query identifies the oldest successful landing. The `WHERE` clause filters the dataset to include only records with successful landings, and the `MIN` function is used to find the record with the earliest date.

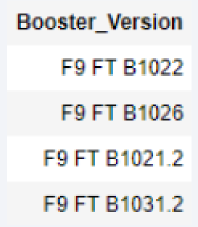
Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query



```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "LANDING _OUTCOME" = 'Success (drone ship)' \
AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000;
```

Results

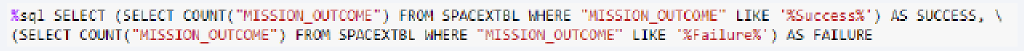


Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

This query retrieves the booster versions for which landings were successful and the payload mass falls between 4000 and 6000 kg. The `WHERE` and `AND` clauses are used to filter the dataset according to these criteria.

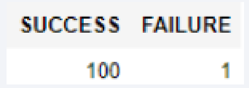
Total Number of Successful and Failure Mission Outcomes

SQL Query



```
%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Success%') AS SUCCESS, \
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS FAILURE
```

Results

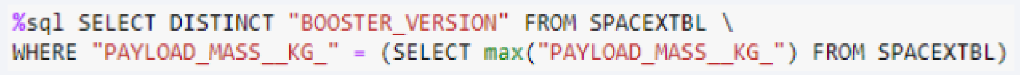


SUCCESS	FAILURE
100	1

In the initial `SELECT` statement, subqueries are used to provide results. The first subquery counts the number of successful missions, while the second subquery counts the number of unsuccessful missions. The `WHERE` clause combined with the `LIKE` clause filters the mission outcomes, and the `COUNT` function tallies the records that match these filters.

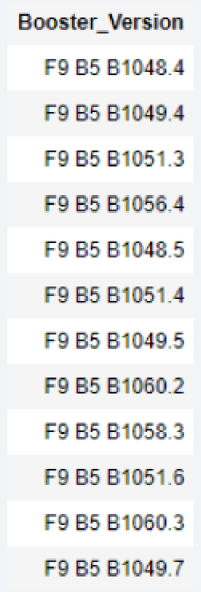
Boosters Carried Maximum Payload

SQL Query



```
%sql SELECT DISTINCT "BOOSTER_VERSION" FROM SPACEXTBL \
WHERE "PAYLOAD_MASS__KG_" = (SELECT max("PAYLOAD_MASS__KG_") FROM SPACEXTBL)
```

Results

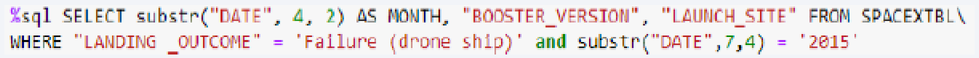


Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

We utilized a subquery to filter the data by identifying the heaviest payload mass using the `MAX` function. The main query then takes the results from this subquery and returns the unique booster version (`SELECT DISTINCT`) associated with the heaviest payload mass.

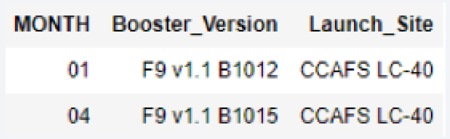
2015 Launch Records

SQL Query



```
%sql SELECT substr("DATE", 4, 2) AS MONTH, "BOOSTER_VERSION", "LAUNCH_SITE" FROM SPACEXTBL\
WHERE "LANDING _OUTCOME" = 'Failure (drone ship)' and substr("DATE",7,4) = '2015'
```

Results

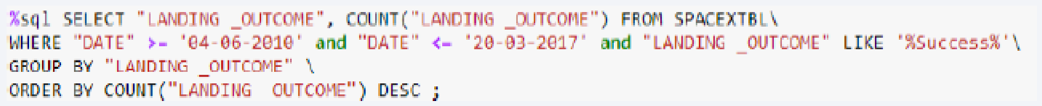


MONTH	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

This query retrieves the month, booster version, and launch site for instances where the landing was unsuccessful and occurred in 2015. The `SUBSTR` function is used to extract specific parts of the date: `SUBSTR(DATE, 4, 2)` extracts the month, while `SUBSTR(DATE, 7, 4)` extracts the year. This allows for filtering the data to include only the relevant records from 2015.

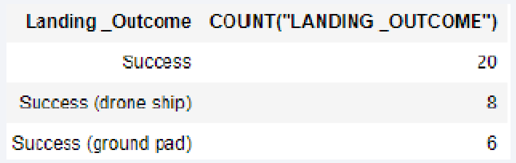
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

SQL Query



```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE '%Success%'\
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING  OUTCOME") DESC ;
```

Results



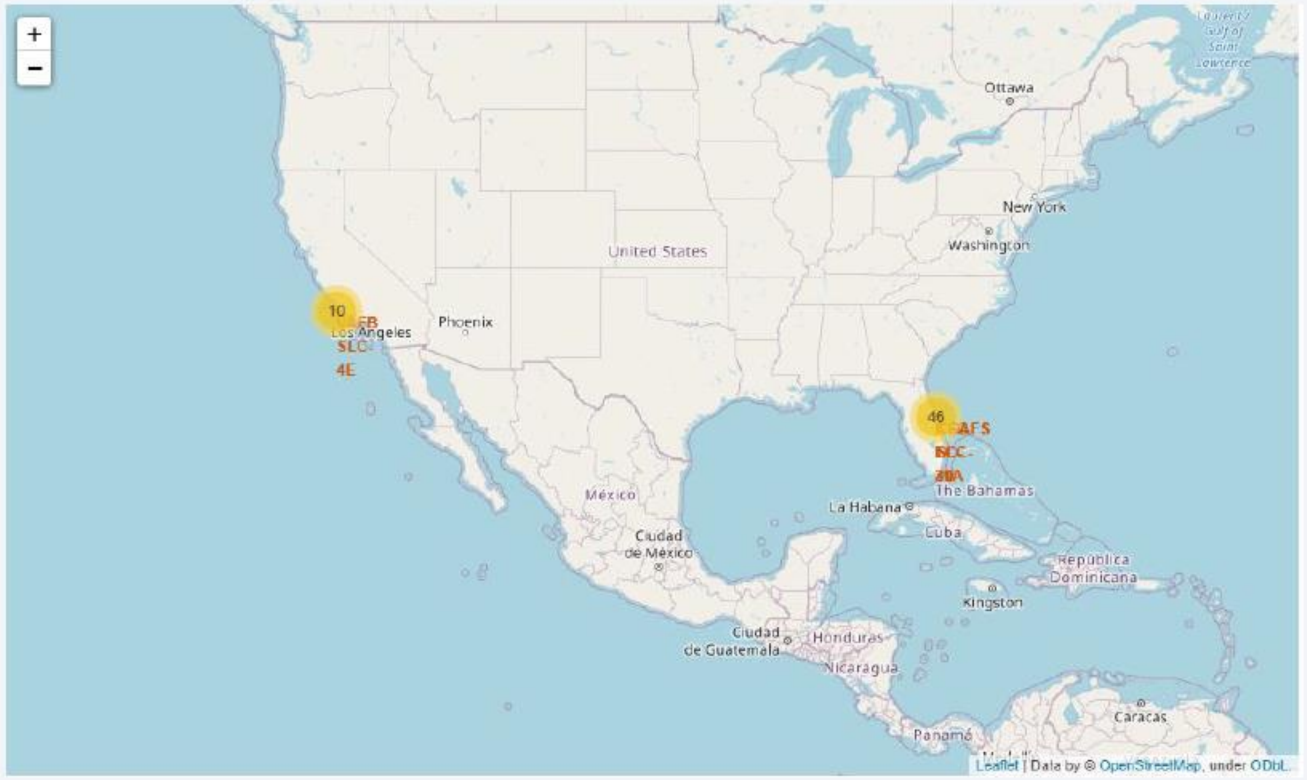
Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

This query retrieves the landing outcomes and their respective counts for missions that were successful between 04/06/2010 and 20/03/2017. The `GROUP BY` clause organizes the results by landing outcome, and the `ORDER BY COUNT DESC` sorts the outcomes by count in descending order.

Section 3

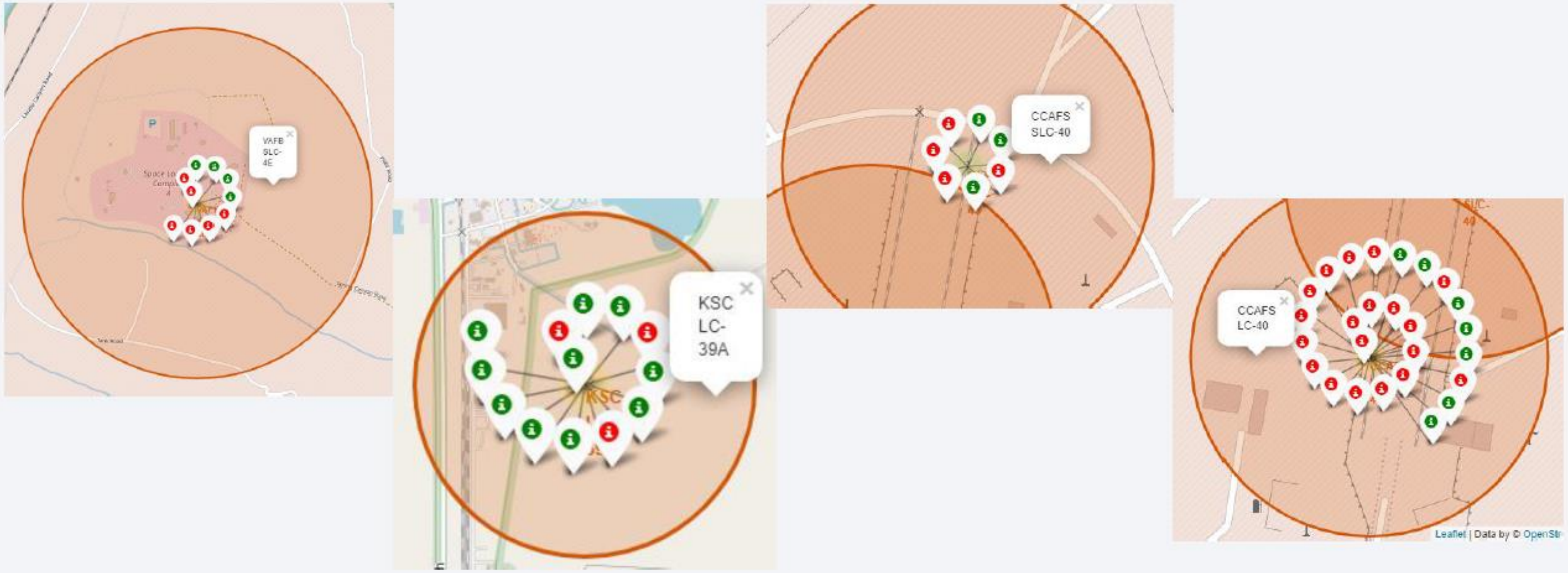
Launch Sites Proximities Analysis

Folium map –Ground stations



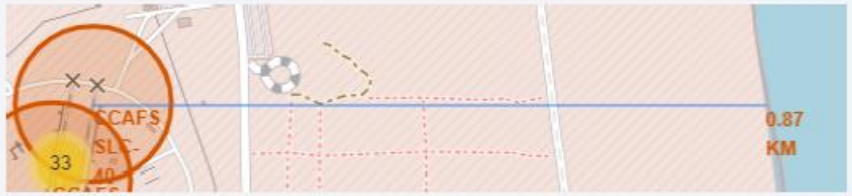
It is evident that SpaceX launch sites are situated along the coast of the United States.

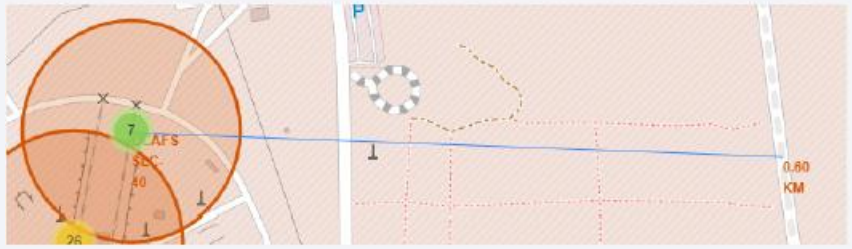
Folium map –Color Labeled Markers

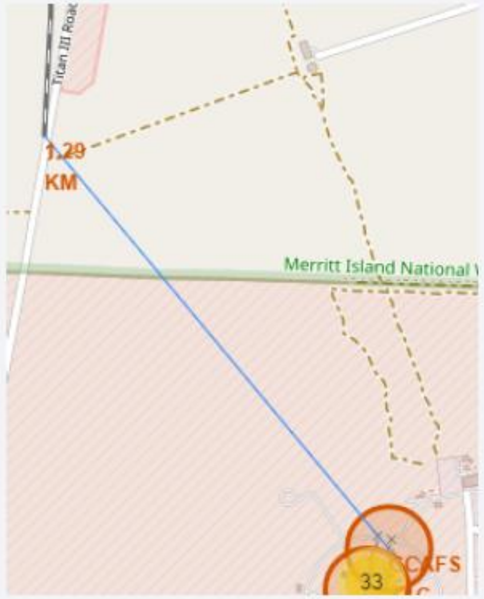


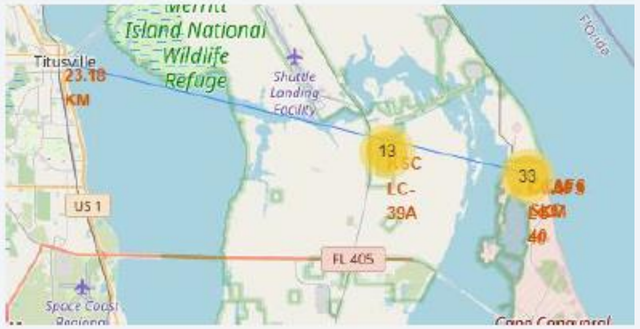
The green markers indicate successful launches, while the red markers represent unsuccessful ones. It's observed that the KSC LC 39A launch site has a higher success rate compared to others.

Folium Map –Distances between CCAFS SLC-40 and its proximities







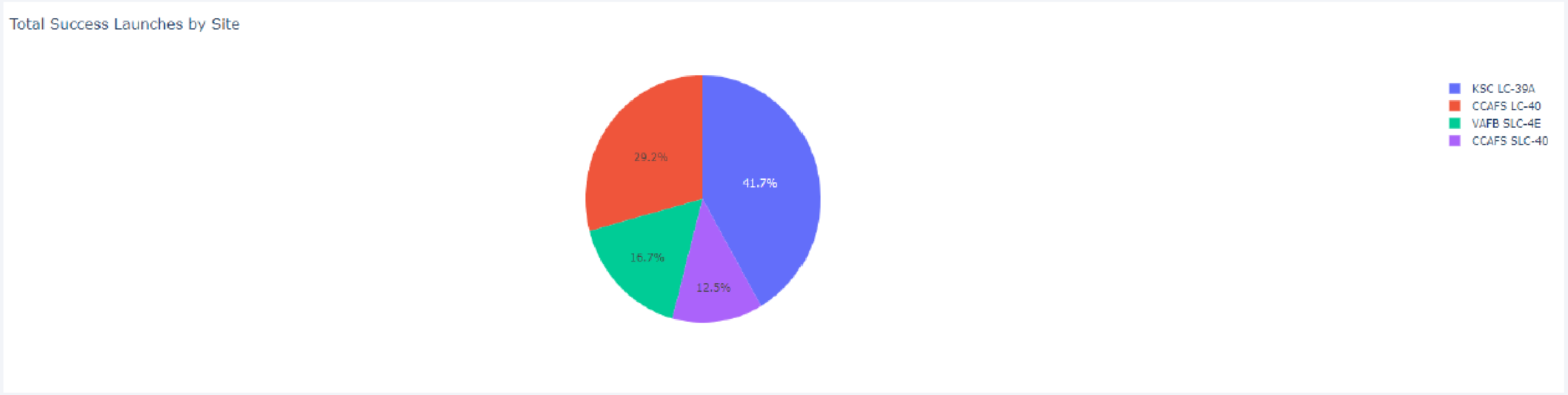


- Is CCAFS SLC 40 close to railways? Yes
- Is CCAFS SLC 40 close to highways? Yes
- Is CCAFS SLC 40 close to the coastline? Yes
- Does CCAFS SLC 40 maintain a certain distance from cities? No

Section 4

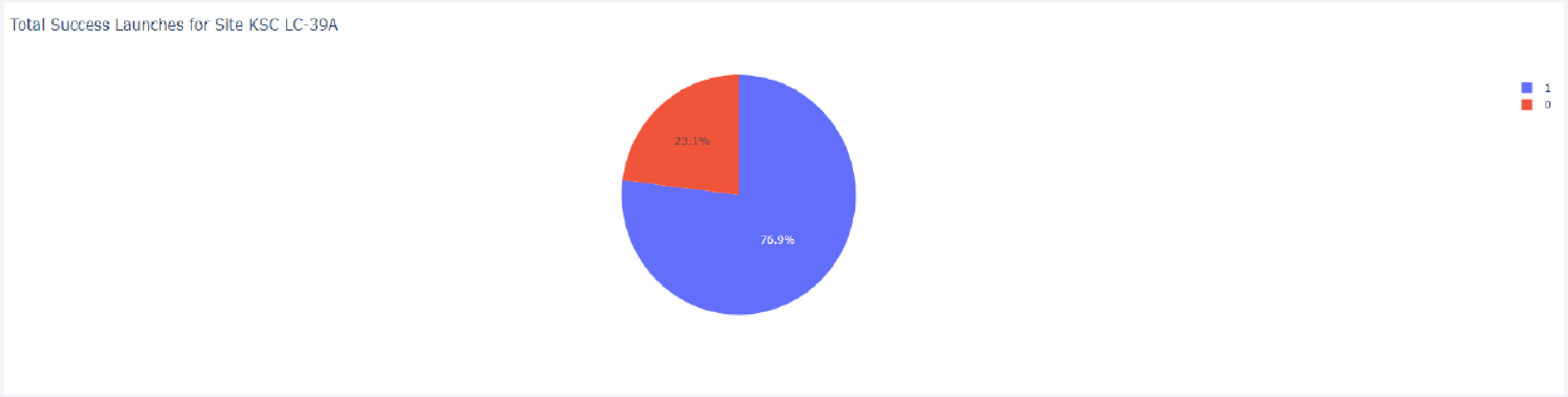
Build a Dashboard with Plotly Dash

Dashboard –Total success by Site



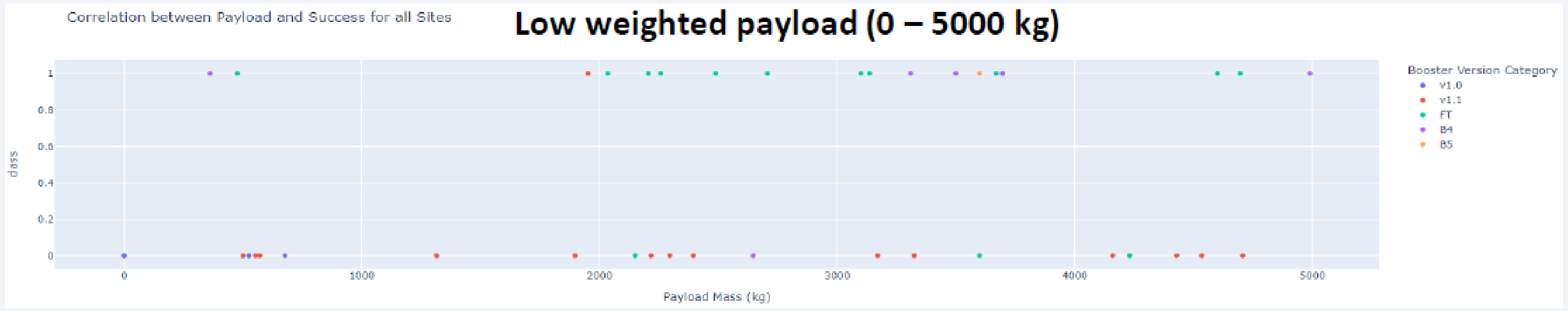
It's clear that KSC LC 39A has the highest success rate for launches.

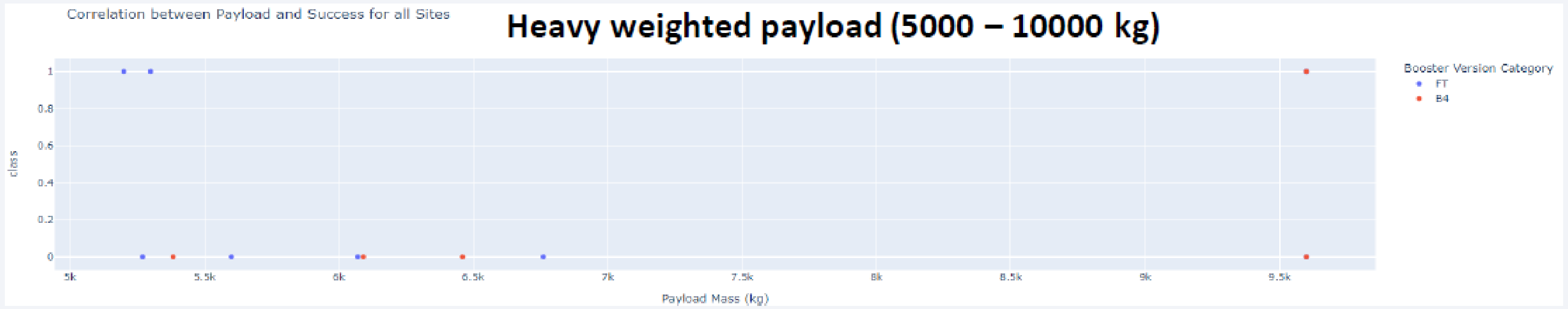
Dashboard –Total success launches for Site KSC LC-39A



KSC LC 39A has achieved a 76.9% success rate, with a corresponding failure rate of 23.1%.

Dashboard –Payload mass vs Outcome for all sites with different payload mass selected



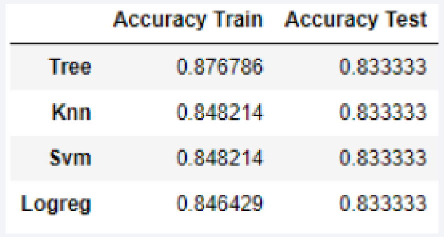


Lower-weight payloads tend to have a higher success rate compared to heavier payloads.

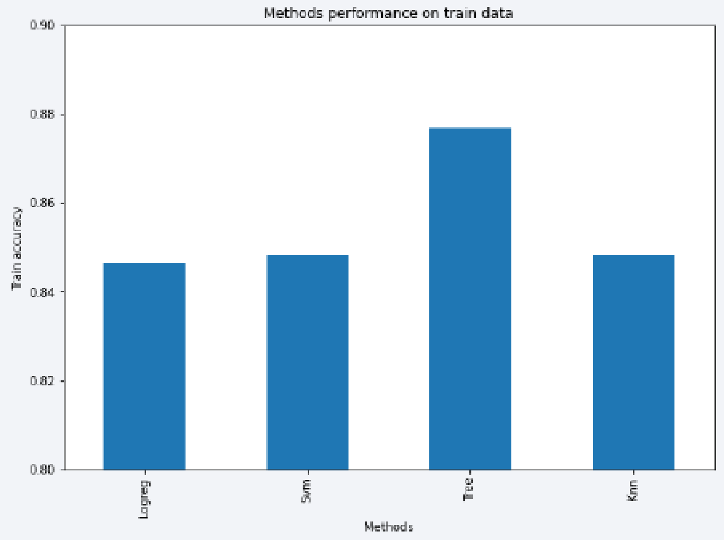
Section 5

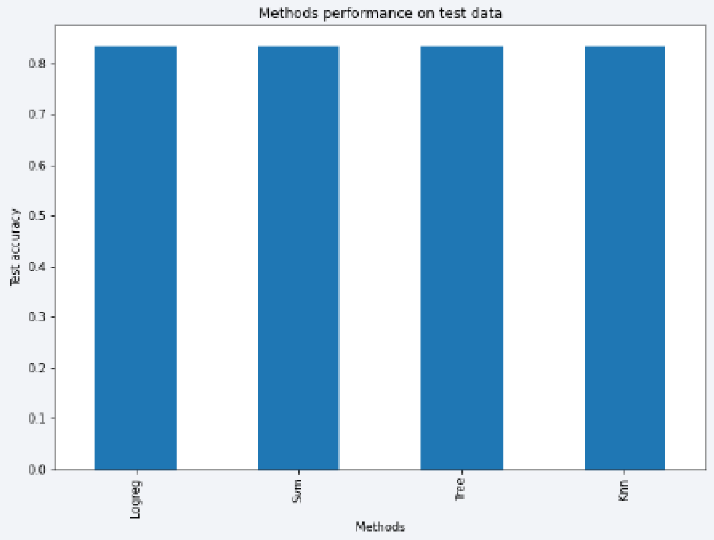
Predictive Analysis (Classification)

Classification Accuracy



	Accuracy Train	Accuracy Test
Tree	0.876786	0.833333
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333

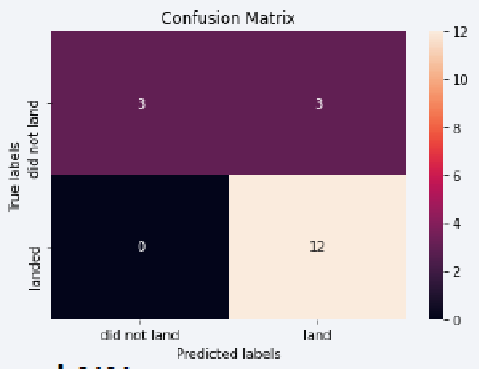




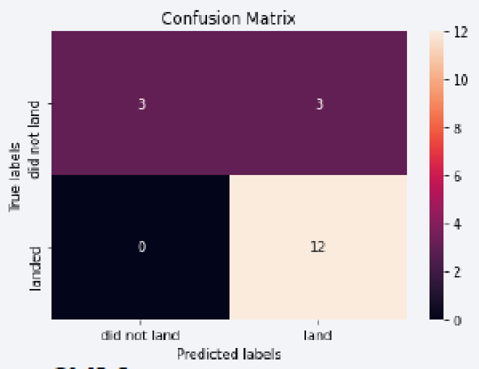
In the accuracy tests, all methods performed similarly. To make a more informed decision, additional test data would be beneficial. However, if a choice is required immediately, the decision tree would be the preferred option.

Confusion Matrix

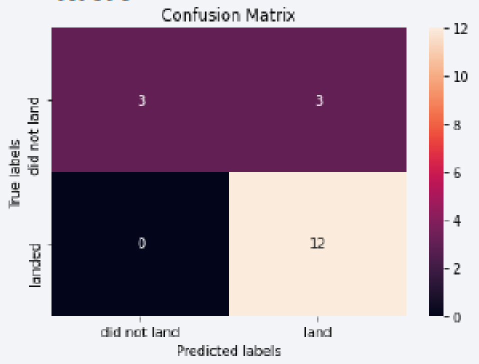
Logistic regression



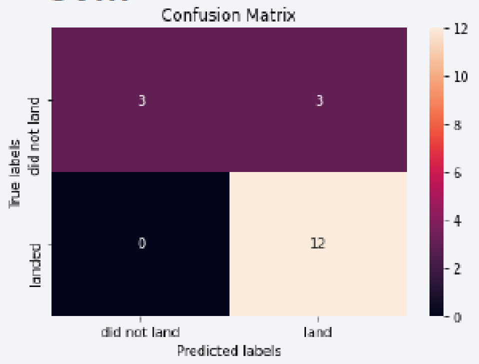
Decision Tree



kNN



SVM



Since the test accuracies are all equal, the confusion matrices are identical as well. The primary issue with these models is the occurrence of false positives.

Conclusions

1. Mission Success Factors: The success of a mission is influenced by various factors, including the launch site, the orbit, and particularly the number of previous launches. It’s reasonable to assume that the experience and knowledge gained from earlier launches contribute to transitioning from failures to successes.

2. Top Success Orbits: The orbits with the highest success rates are GEO, HEO, SSO, and ES L1.

3. Payload Mass Considerations: Depending on the orbit, payload mass can be a critical factor for mission success. Some orbits are better suited for lighter or heavier payloads. However, in general, lighter payloads tend to achieve higher success rates than heavier ones.

4. Launch Site Performance: The current data does not provide a clear explanation for why some launch sites perform better than others (with KSC LC 39A being the top performer). To better understand this, additional data such as atmospheric conditions or other relevant factors would be necessary.

5. Model Selection: For this dataset, the Decision Tree Algorithm was chosen as the best model, despite identical test accuracy across all models. The Decision Tree was preferred because it demonstrated better training accuracy.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!