Winning Space Race with Data Science

Evan Jager

July 4th, 2023

Outline

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

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

SpaceX, a prominent player in the space industry, is committed to making space travel accessible to a wider audience. The company has achieved significant milestones such as delivering spacecraft to the international space station, deploying a satellite constellation for global internet access, and conducting crewed missions to space. SpaceX's ability to offer relatively affordable rocket launches, priced at $62 million per launch, can be attributed to its innovative approach of reusing the first stage of the Falcon 9 rocket. In contrast, other providers that lack this reusability feature charge upwards of $165 million per launch. By determining the successful landing of the first stage, we can accurately calculate the cost of the launch. To accomplish this, we can leverage publicly available data and employ machine learning models to predict the likelihood of SpaceX, or a competing company, being able to reuse the first stage.

Problems you want to find answers

- What are the main characteristics of a successful or failed landing ?
- What are the effects of each relationship of the rocket variables on the success or failure of a landing ?
- What are the conditions which will allow SpaceX to achieve the best landing success rate ?

Section 1

Methodology

Methodology

Executive Summary

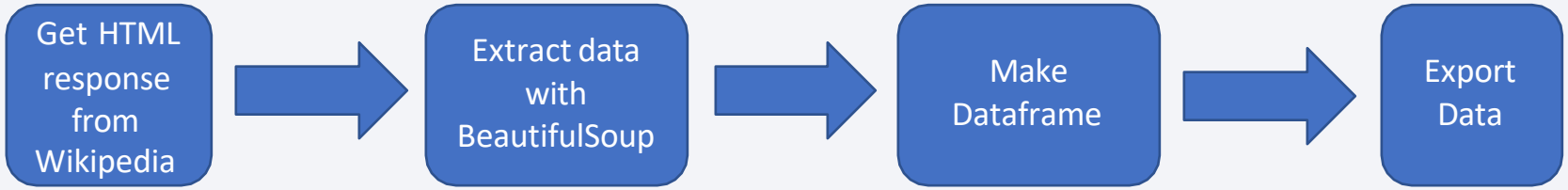
- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Datasets are acquired from the SpaceX REST API and Web scraping relevant information from Wikipedia.

SpaceX API Data collection Flow Chart:

Wikipedia Web scraping flow chart



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)
```

Link to code

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

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

8. Export to file

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

Link to code

Data Wrangling

Within the dataset, numerous instances exist where the booster did not achieve a successful landing.,

"True Ocean," "True RTLS," and "True ASDS" indicate a successful mission

"False Ocean," "False RTLS," and "False ASDS" signify a mission failure.

To accomplish our objective, we aim to convert these string variables into categorical variables, assigning a value of 1 to successful missions and 0 to failed missions.

1. Calculate number of launches for each site

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

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

2. 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
```

3. Calculate number and occurrence of mission outcomes 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
```

4. Create landing outcome label

```
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
```

5. Export to file

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

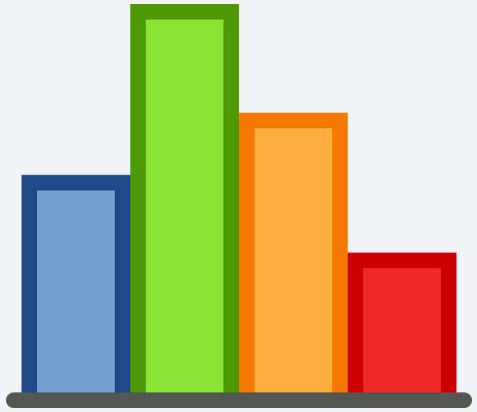
Link to code

EDA with Data Visualization

Bar Graph

A bar graph is a visual representation that uses rectangular bars to compare and present categorical data or discrete variables.

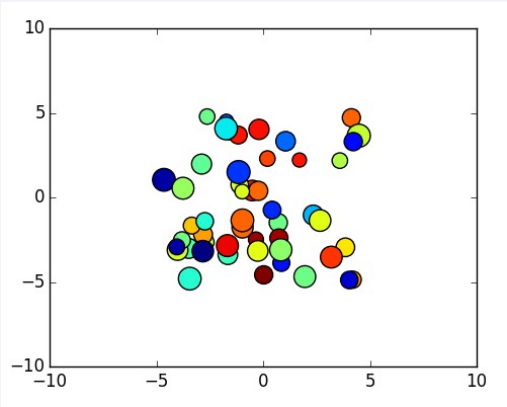
Success rate vs. Orbit



Scatter Plots

A scatter plot is a graphical representation that displays the relationship between two variables using individual data points on a Cartesian coordinate system.

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



Link to code

Line Graph

A line graph is a visual depiction that illustrates the trend or pattern of data over a continuous period or interval by connecting data points with straight lines.

Success rate vs. Year

EDA with SQL

We performed SQL queries to gather and understand data from dataset:

- Showcase the distinct names of launch sites involved in space missions.
- Exhibit 5 records where launch sites commence with the string 'CCA.'
- Present the total payload mass carried by NASA (CRS) boosters during their launches.
- Display the average payload mass transported by booster version F9 v1.1.
- Provide the date when the first successful landing occurred on a ground pad.
- Enumerate the names of boosters that achieved success on a drone ship and carried a payload mass between 4000 and 6000.
- Enumerate the total count of successful and failed mission outcomes.
- List the names of booster versions that carried the maximum payload mass.
- Retrieve records displaying the month names, failure landing outcomes on a drone ship, booster versions, and launch sites for the months in the year 2015.
- Rank the count of successful landing outcomes between the dates 04-06-2010 and 20-03-2017 in descending order.

Link to code

Build an Interactive Map with Folium

- Red circle at NASA Johnson Space Center's coordinate with label showing its name (folium.Circle, folium.map.Marker).
- Red circles at each launch site coordinates with label showing launch site name (folium.Circle, folium.map.Marker, folium.features.DivIcon).
- The grouping of points in a cluster to display multiple and different information for the same coordinates (folium.plugins.MarkerCluster).
- Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing (folium.map.Marker, folium.Icon).
- Markers to show distance between launch site to key locations (railway, highway, coastway, city) and plot a line between them (folium.map.Marker, folium.PolyLine, folium.features.DivIcon).

These objects are created in order to understand better the problem and the data. We can show easily all launch sites, their surroundings and the number of successful and unsuccessful landings.

Link to code

Build a Dashboard with Plotly Dash

The dashboard comprises several components including a dropdown, pie chart, range slider, and scatter plot:

•The dropdown component, implemented with dash_core_components.Dropdown, enables users to select either a specific launch site or all launch sites.

•The pie chart, created using plotly.express.pie, visualizes the total success and failure outcomes for the launch site chosen through the dropdown component.

•The range slider, implemented with dash_core_components.RangeSlider, allows users to specify a payload mass within a predefined range.

•The scatter plot, generated using plotly.express.scatter, depicts the relationship between two variables, specifically Success vs Payload Mass.

Link to code

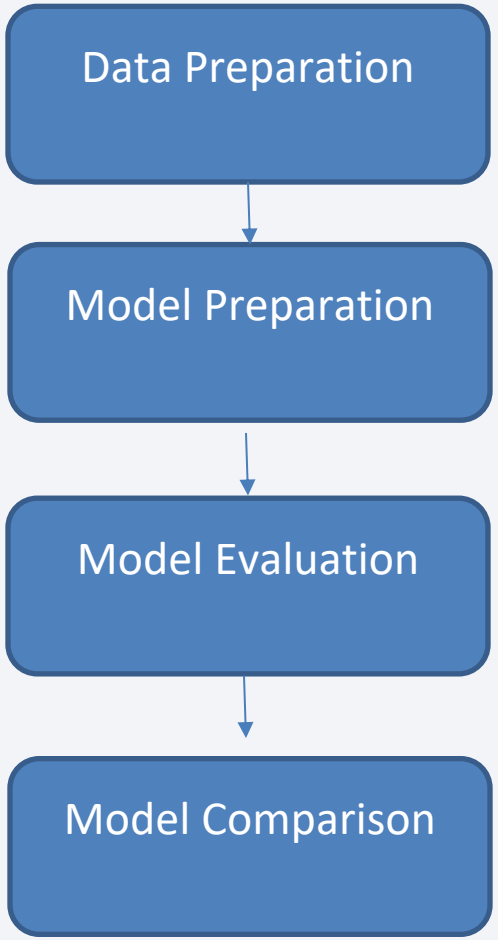
Predictive Analysis (Classification)

Data Preparation:
- Load the dataset into the environment.
- Normalize the data to ensure consistent scaling and avoid bias towards specific features.
- Split the data into training and test sets to assess the model's performance.

Model Preparation:
- Select appropriate machine learning algorithms suitable for the dataset and task.
- Set the parameters for each algorithm using GridSearchCV to optimize their performance.
- Train the GridSearchCV models using the training dataset.

Model Evaluation:
- Retrieve the best hyperparameters for each model determined by the GridSearchCV.
- Calculate the accuracy of each model using the test dataset.
- Visualize the Confusion Matrix to assess the model's predictive performance.

Model Comparison:
- Compare the accuracy of different models.
- Choose the model with the highest accuracy as the preferred choice. Refer to the Notebook for detailed results.



Link to code

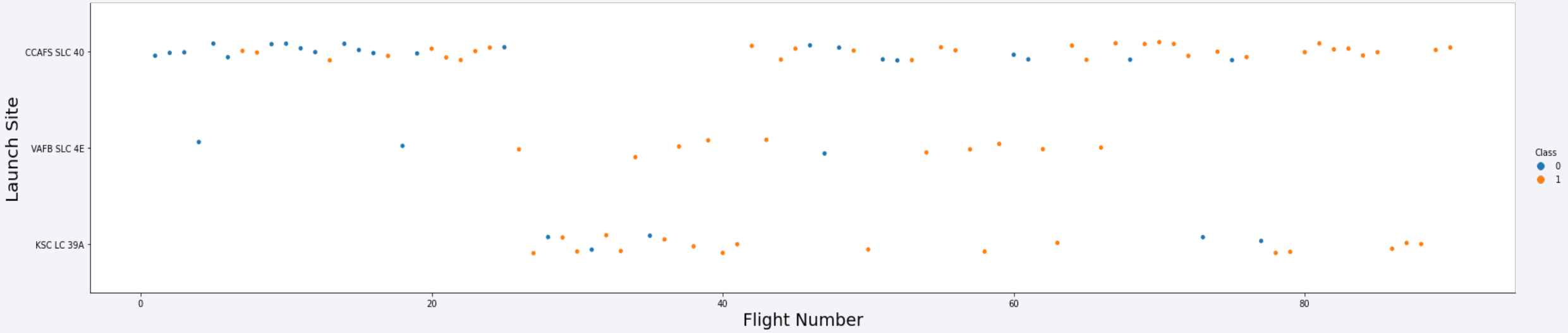
Results

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

Section 2

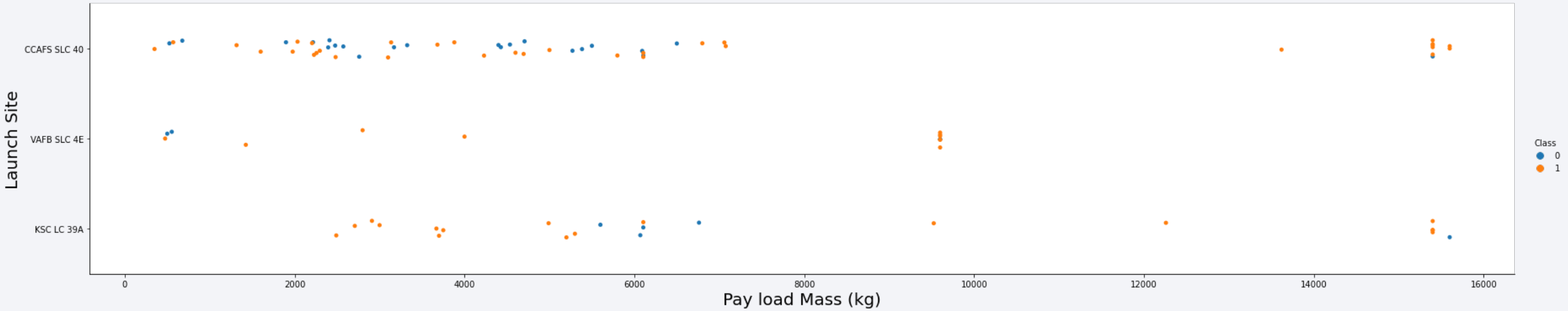
Insights drawn from EDA

Flight Number vs. Launch Site



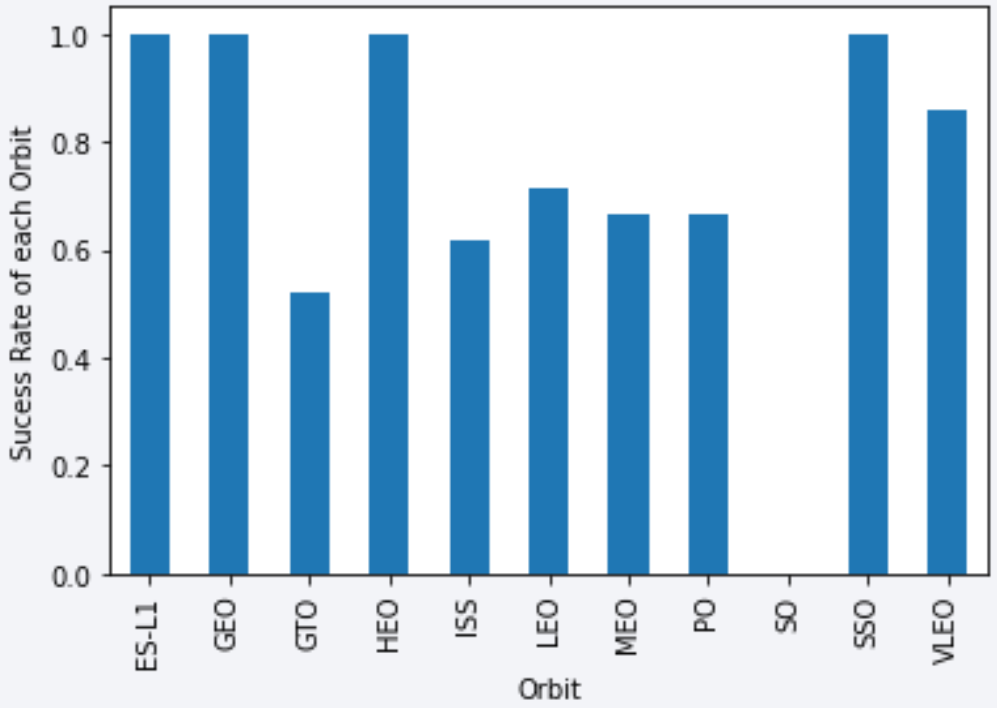
We observe that, for each site, the success rate is increasing.

Payload vs. Launch Site



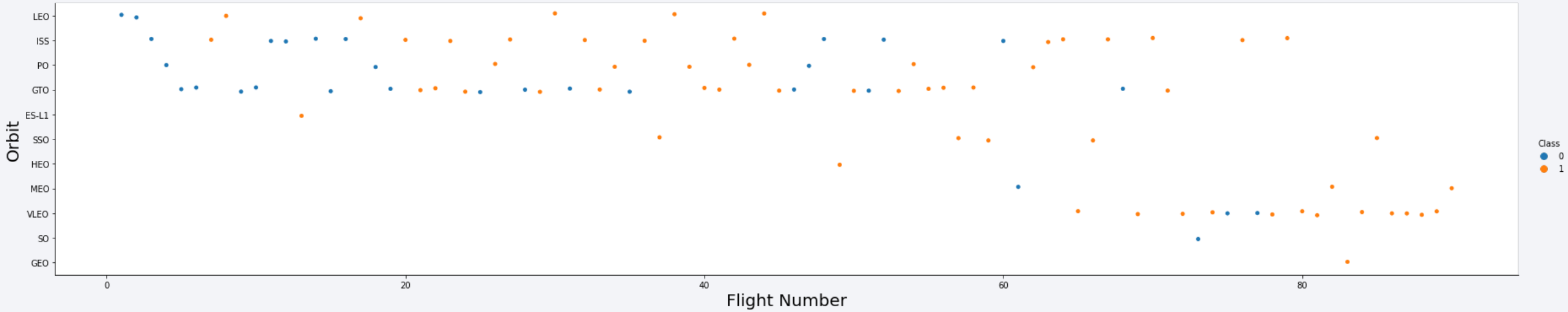
Depending on the launch site, a heavier payload may be a consideration for a successful landing. On the other hand, a too heavy payload can make a landing fail.

Success Rate vs. Orbit Type



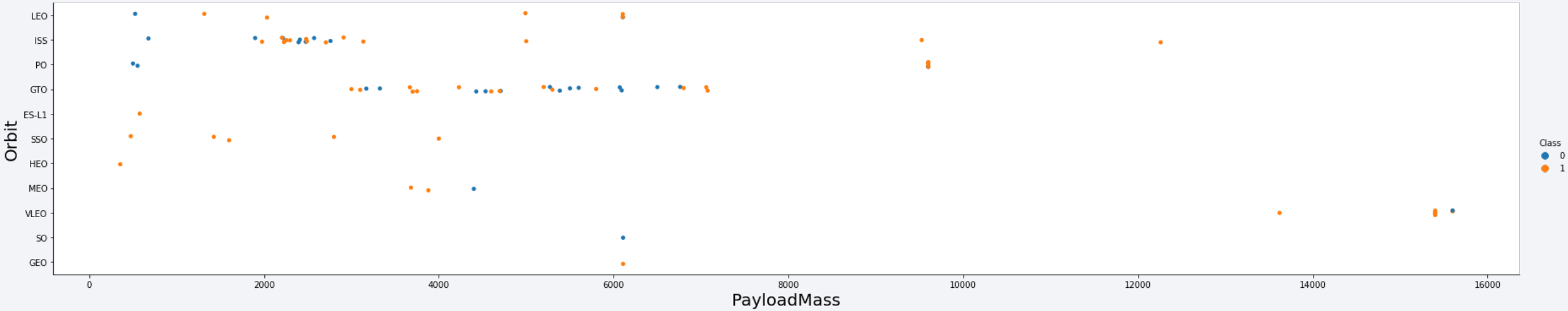
With this plot, we can see success rate for different orbit types. We note that ES-L1, GEO, HEO, SSO have the best success rate.

Flight Number vs. Orbit Type



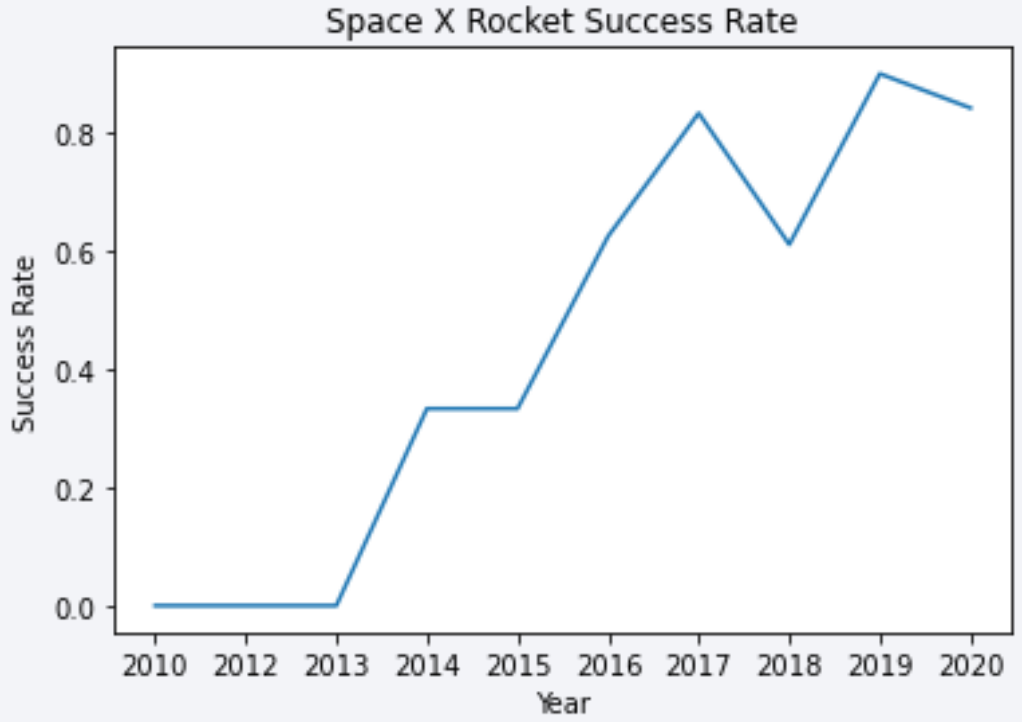
We notice that the success rate increases with the number of flights for the LEO orbit. For some orbits like GTO, there is no relation between the success rate and the number of flights. But we can suppose that the high success rate of some orbits like SSO or HEO is due to the knowledge learned during former launches for other orbits.

Payload vs. Orbit Type



The weight of the payloads can have a great influence on the success rate of the launches in certain orbits. For example, heavier payloads improve the success rate for the LEO orbit. Another finding is that decreasing the payload weight for a GTO orbit improves the success of a launch.

Launch Success Yearly Trend



Since 2013, we can see an increase in the Space X Rocket success rate.

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

Explanation

By incorporating the DISTINCT keyword in the query, duplicate values for the LAUNCH_SITE field can be eliminated.

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)

Explanation

The combination of the WHERE clause and LIKE clause in the query helps to filter the launch sites that contain the substring "CCA". Additionally, the LIMIT 5 statement limits the result set to only display 5 records from the filtered data.

Total Payload Mass

SQL Query

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

Results

SUM("PAYLOAD_MASS__KG_")
45596

Explanation

This query calculates the sum of all payload masses 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

Explanation

This query calculates the average of all payload masses where the booster version contains 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

Explanation

This query selects the oldest successful landing by applying filters to the dataset to keep only the records where the landing was successful. The MIN function is then used to select the record with the oldest 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

Explanation

This query retrieves the booster version for records where the landing was successful and the payload mass falls between 4000 and 6000 kg. The WHERE clause, along with the AND clause, filters the dataset accordingly.

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

Explanation

The first SELECT statement displays the subqueries that produce results. The first subquery counts the number of successful missions, while the second subquery counts the number of unsuccessful missions. The WHERE clause, along with the LIKE clause, filters the mission outcomes. The COUNT function is then used to count the records that match the specified 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

Explanation

In the query, we utilized a subquery to filter the data by retrieving only the record with the maximum payload mass using the MAX function. The main query then utilizes the results of the subquery to return the unique booster version (using 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

Explanation

This query retrieves the month, booster version, and launch site for records where the landing was unsuccessful and the landing date occurred in 2015. The Substr function is used to extract the month or year from the DATE column. Substr(DATE, 4, 2) extracts the month, and Substr(DATE, 7, 4) extracts the year.

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

SQL Query

```
%sql SELECT Landing_Outcome as "Landing_Outcome", COUNT(Landing_Outcome) AS "Total_Count" FROM SPACEXTBL \
WHERE Date >= '04/06/2010' AND Date <='20/03/2017' \
GROUP BY  Landing_Outcome \
ORDER BY COUNT(Landing_Outcome) DESC ;
```

Results

Landing_Outcome	Total_Count
Success	20
No attempt	9
Success (drone ship)	8
Success (ground pad)	7
Failure (drone ship)	3
Failure	3
Failure (parachute)	2
Controlled (ocean)	2
No attempt	1

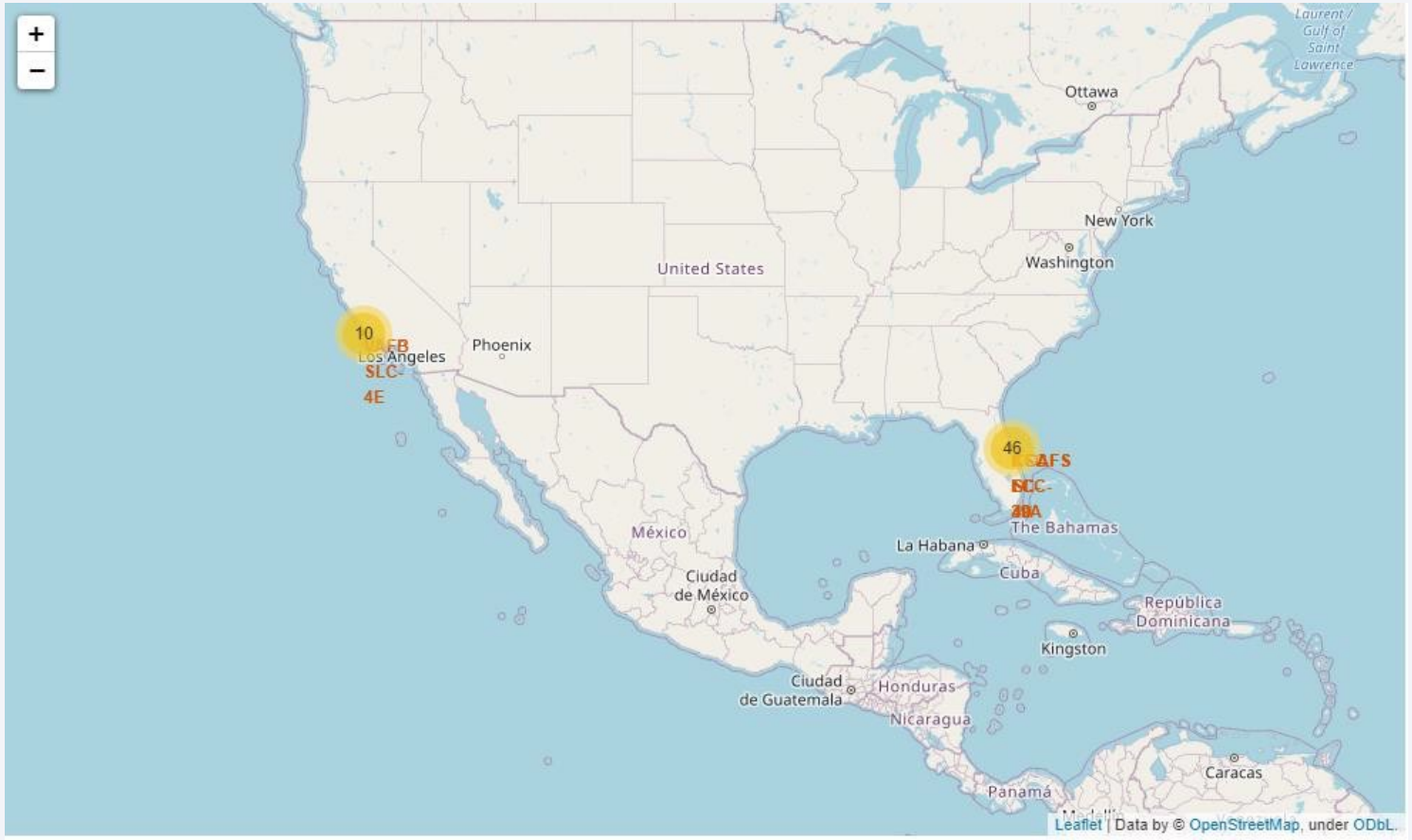
Explanation

This query retrieves the landing outcomes and their count of occurrences between the dates 04/06/2010 and 20/03/2017. The GROUP BY clause is used to group the results by landing outcome, and the ORDER BY COUNT DESC sorts the results in descending order based on the count of occurrences.

Section 4

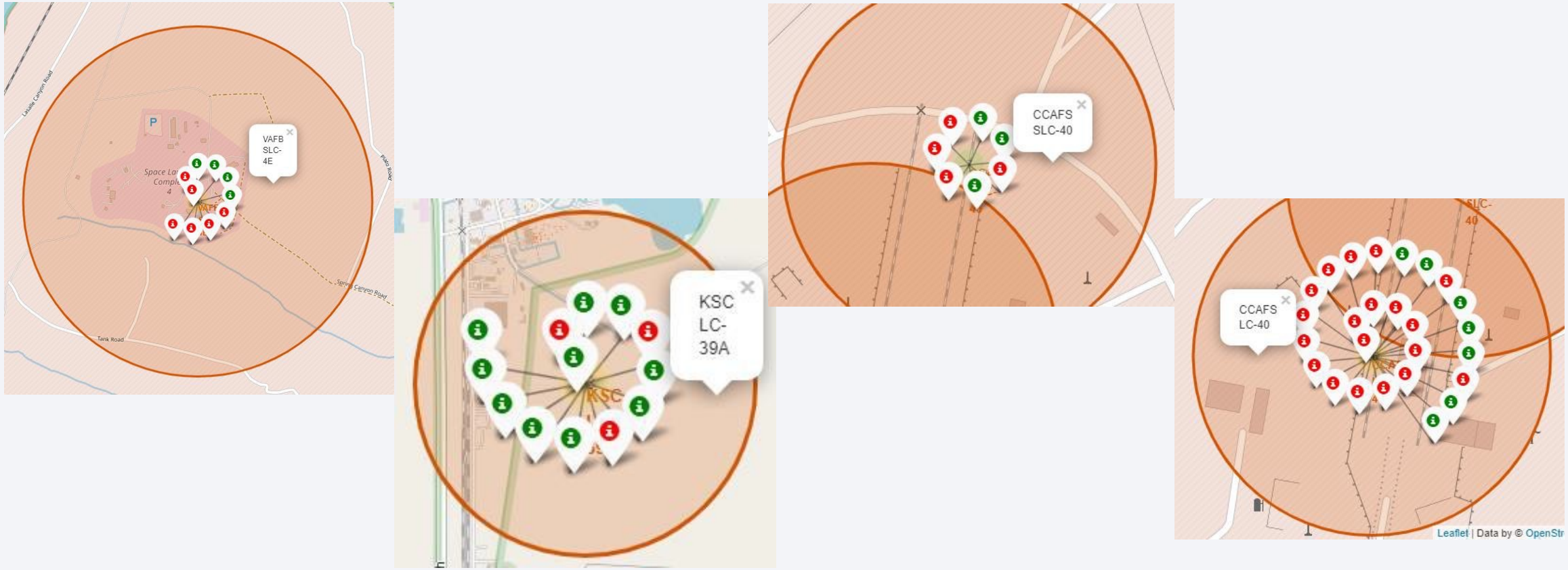
Launch Sites Proximities Analysis

Folium map - Ground stations



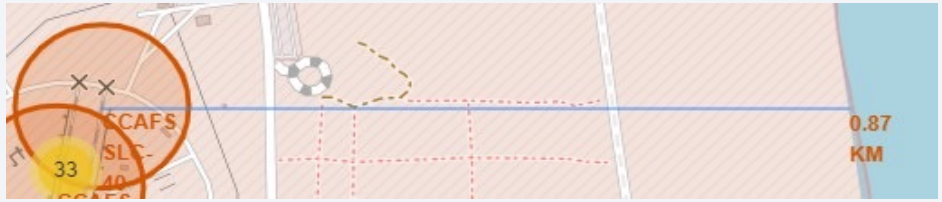
We see that Space X launch sites are located on the coast of the United States

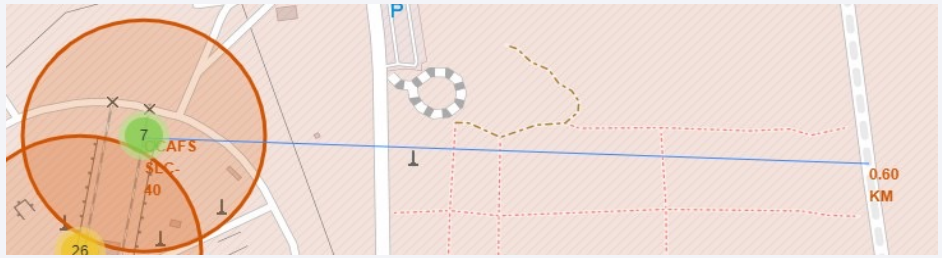
Folium map - Color Labeled Markers



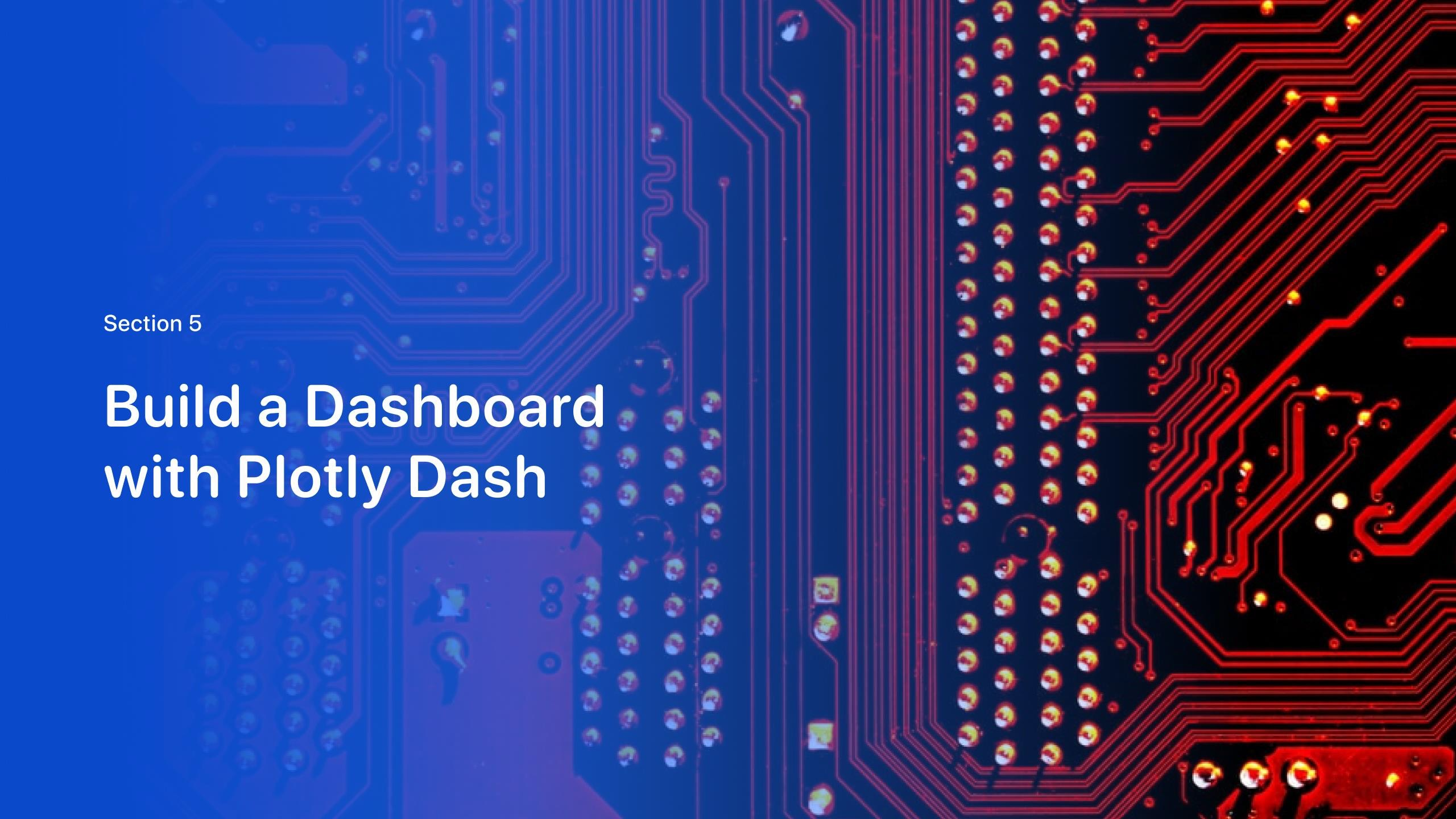
Successful launches are represented by green markers, while unsuccessful launches are represented by red markers. It is observed that KSC LC-39A has a higher launch success rate compared to other launch sites.

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

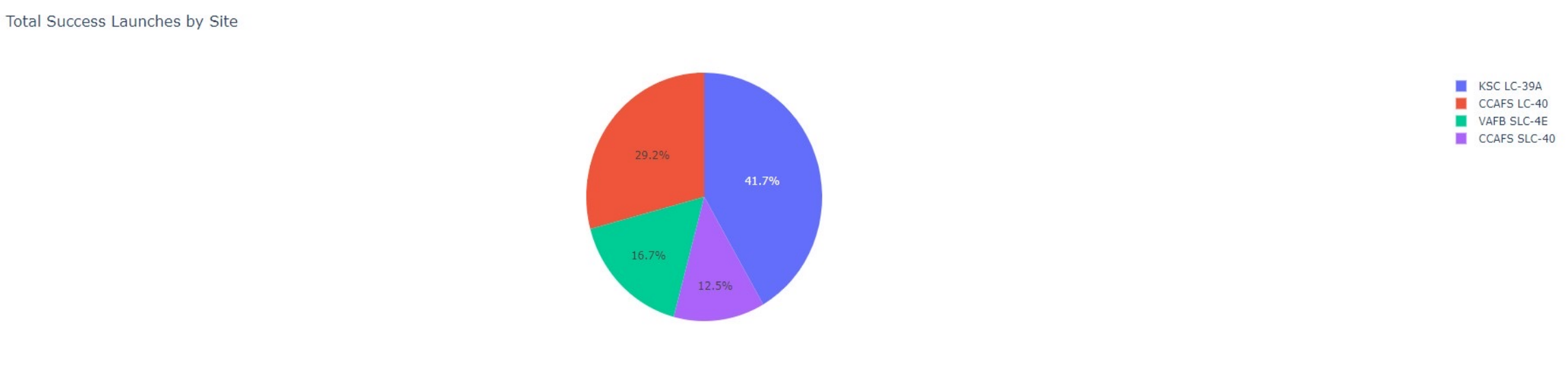




Is CCAFS SLC-40 in close proximity to railways ?
Yes
Is CCAFS SLC-40 in close proximity to highways ?
Yes
Is CCAFS SLC-40 in close proximity to coastline ?
Yes
Do CCAFS SLC-40 keeps certain distance away from cities?
No

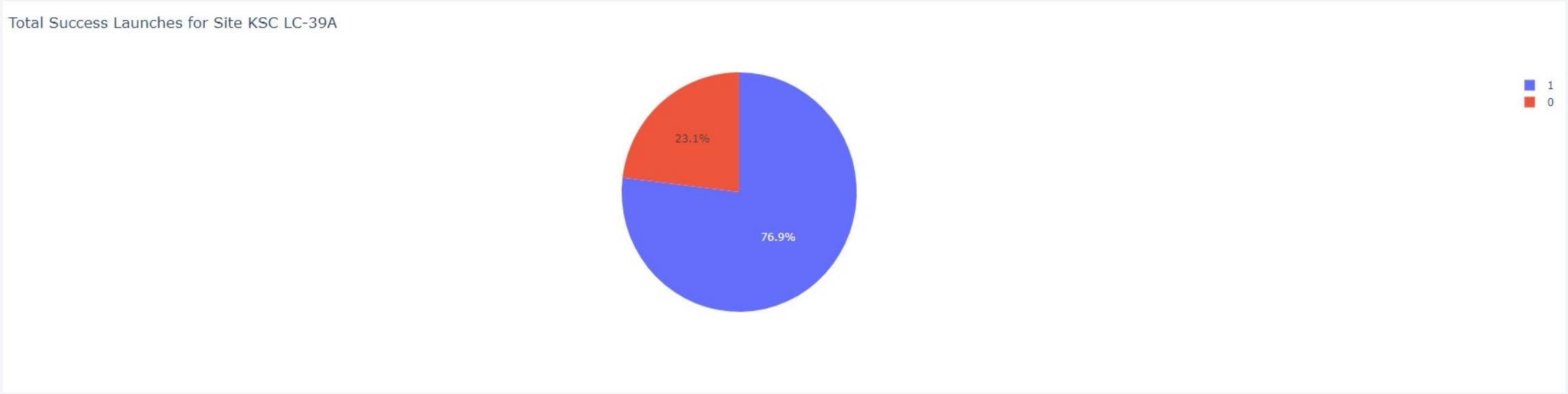

Section 5
Build a Dashboard with Plotly Dash

Dashboard - Total success by Site



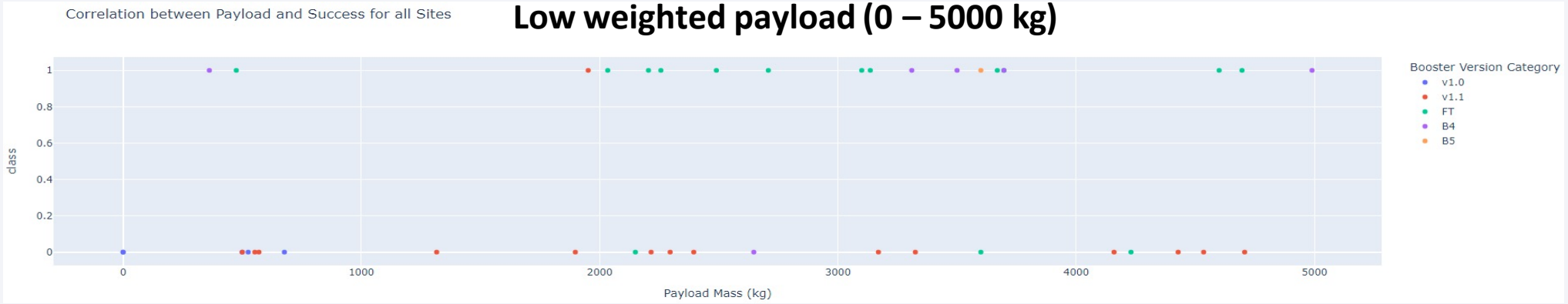
We see that KSC LC-39A has the best success rate of launches.

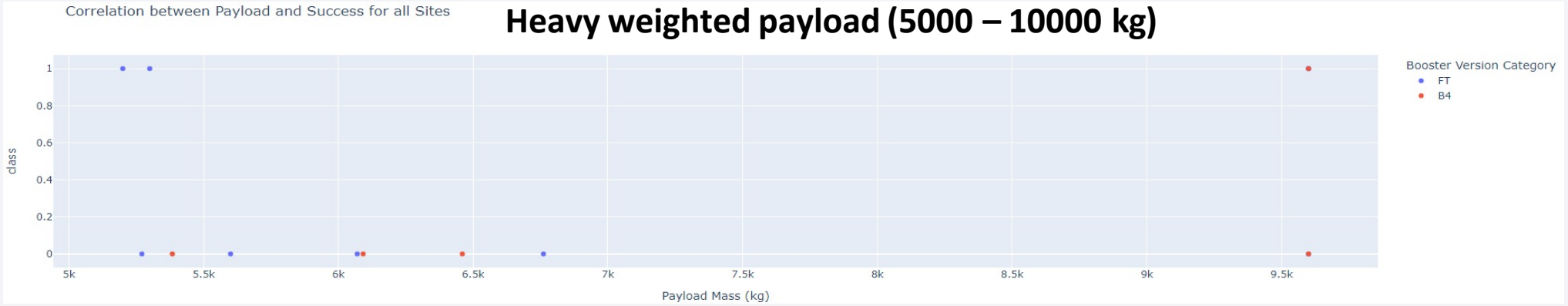
Dashboard – Total success launches for Site KSC LC-39A



We see that KSC LC-39A has achieved a 76.9% success rate while getting a 23.1% failure rate.

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





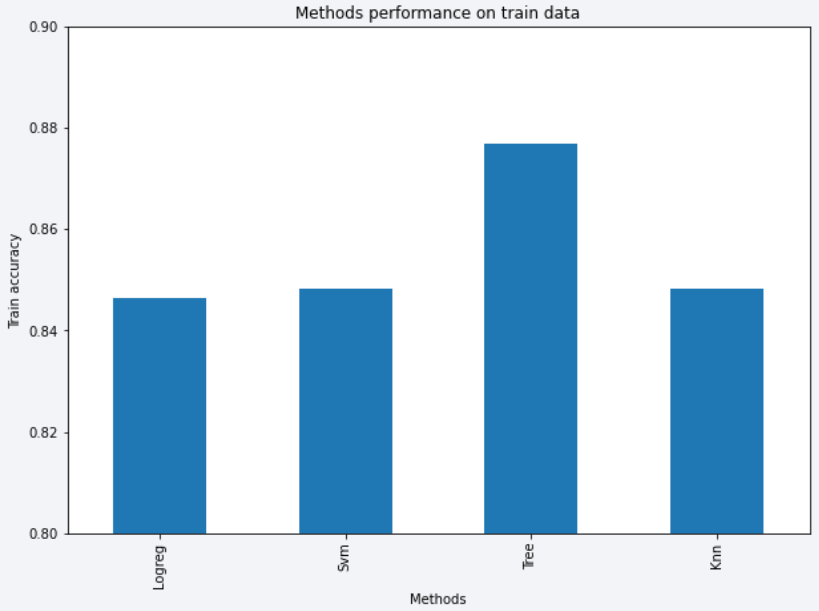
Low weighted payloads have a better success rate than the heavy weighted payloads.

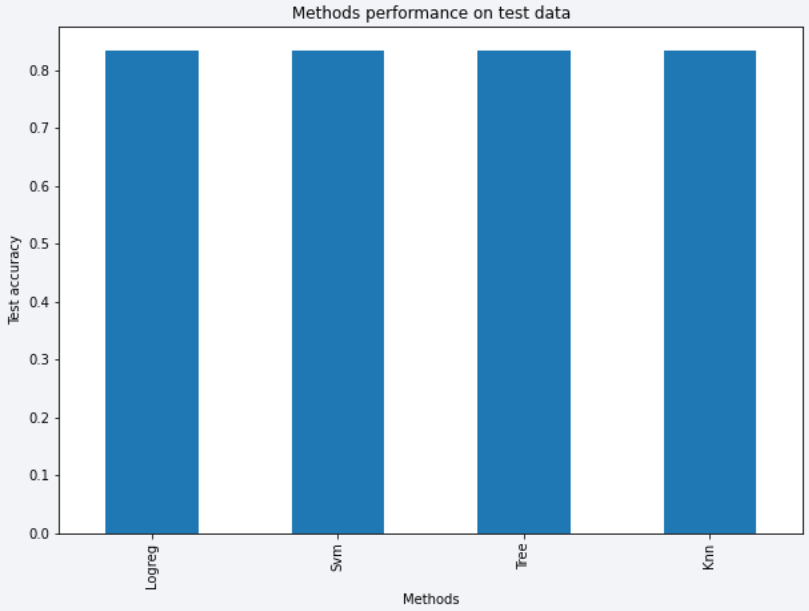
Section 6

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





For accuracy test, all methods performed similar. We could get more test data to decide between them. But if we really need to choose one right now, we would take the decision tree.

Decision tree best parameters

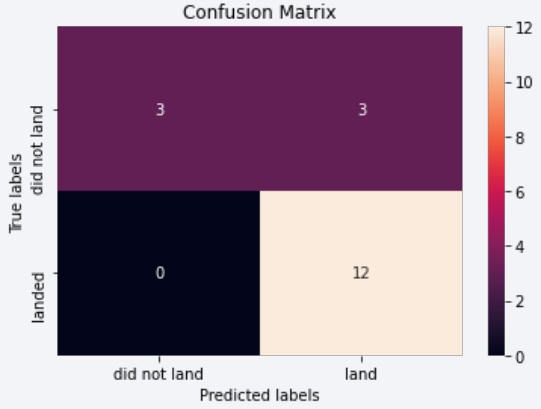
```
tuned hyperparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'sqrt', 'min_samples_leaf':
4, 'min_samples_split': 2, 'splitter': 'random'}
```

Confusion Matrix

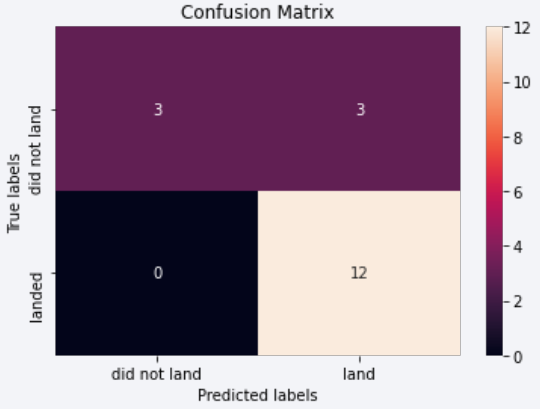
Logistic regression

Decision Tree

kNN



SVM



As the test accuracy are all equal, the confusion matrices are also identical. The main problem of these models are false positives.

		Actual values 1	Actual values 0
Predicted values	1	TP	FP
	0	FN	TN

Conclusions

- The success of a mission can be attributed to various factors, including the launch site, orbit, and the number of previous launches. Knowledge gained from previous launches likely contributes to the improvement in success rates over time.

- The orbits with the highest success rates are GEO, HEO, SSO, and ES-L1. Payload mass is also a crucial factor for mission success, with different orbits requiring specific payload masses. Generally, lighter payloads tend to have better success rates compared to heavier ones.

- Although the current dataset does not provide a clear explanation for why certain launch sites perform better than others (such as KSC LC-39A being the top site), obtaining additional atmospheric or relevant data could help shed light on this matter.

- Based on the dataset, the Decision Tree Algorithm is chosen as the best model, even though all the models used have the same test accuracy. The decision is made due to the higher train accuracy of the Decision Tree Algorithm.

Winning Space Race with Data Science

Thank You