IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vinícius Castro
19/07/2023

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
  - The objective of this project is to accurately predict the successful landing of the Falcon 9 first stage. According to SpaceX's official website, the cost of launching a Falcon 9 rocket is estimated at 62 million dollars, which is significantly lower than the prices offered by other providers, reaching upwards of 165 million dollars per launch. The notable cost difference arises from SpaceX's innovative capability to reuse the first stage of the rocket. By accurately determining the landing outcome of the first stage, we can precisely assess the overall launch cost. This valuable information would be of great interest to any company aiming to compete with SpaceX in the rocket launch industry.
- Problems you want to find answers
  - What are the key factors that contribute to a successful or failed landing?
  - How do different rocket variables affect the likelihood of a successful or failed landing?
  - What are the optimal conditions that enable SpaceX to achieve the highest rate of successful landings?

Section 1

Methodology

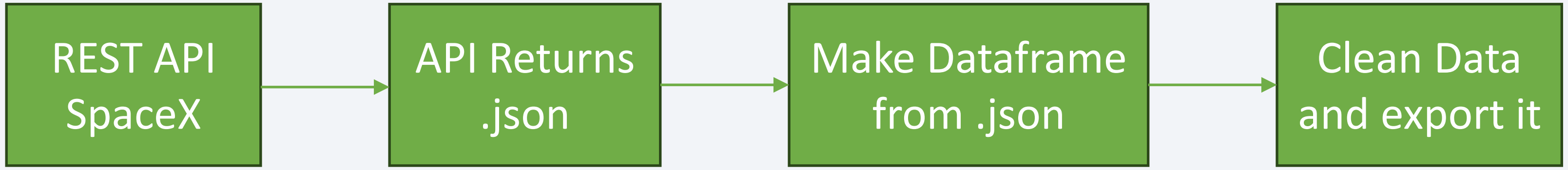
Methodology

Executive Summary

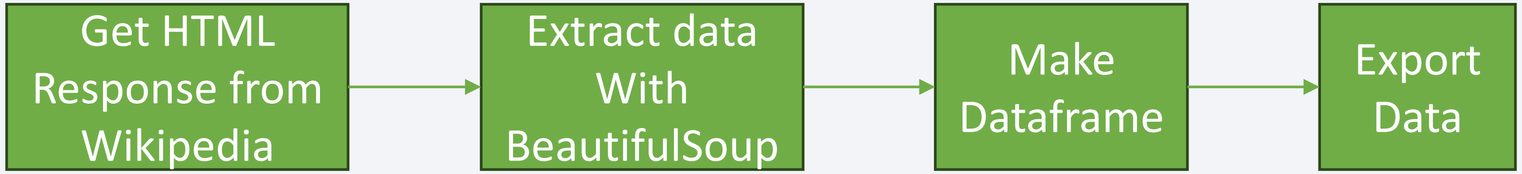
- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using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

Data Collection

- Datasets are collected from both the REST SpaceX API and web scraping Wikipedia.
  - The information obtained through the API includes data on rockets, launches, and payload details.



- The information obtained through web scraping from [Wikipedia] includes data on launches, landings, and payload information.



Data Collection – SpaceX API

[Link GitHub]

Getting response from API

```python
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```python
response = requests.get(spacex_url)
```

Convert response to JSON file

```python
# Use json_normalize meethod to convert
data = response.json()
data = pd.json_normalize(data)
```

Transform Data

```python
# Call getBoosterVersion
getBoosterVersion(data)
```

```python
# Call getLaunchSite
getLaunchSite(data)
```

```python
# Call getPayloadData
getPayloadData(data)
```

```python
# Call getCoreData
getCoreData(data)
```

Create dictionary with Data

```python
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}
```

Create Dataframe

```python
# Create a data from launch_dict
data = pd.DataFrame({key:pd.Series(value) for key, value in launch_dict.items()})
```

Filter Dataframe

```python
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

Export to file

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

Data Collection - Scraping

Getting response from HTML

```
# assign the response to a object
response = requests.get(static_url)
```

Create BeautifulSoup Object

```
# Use BeautifulSoup() to create a BeautifulSoup
soup = BeautifulSoup(response.text, "html5lib")
```

Find all tables

```
# Assign the result to a list called `html_tables`
html_tables = soup.findAll('table')
```

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)
```

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']=[]
```

Add Data to Keys

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
```

Create Dataframe from dictionary

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

Export to file

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

Link GitHub

Data Wrangling

- The dataset contains multiple instances where the booster did not achieve a successful landing.
  - True Ocean, True RTLS, True ASDS means the mission has been successful
  - False Ocean, False RTLS, False ASDS means the mission was a failure.
- We need to convert string variables representing mission outcomes into categorical variables, where "1" signifies a successful mission and "0" indicates a mission failure.

1. Calculate launches number for each site

```
# Apply value_counts() on column LaunchSite
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

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()

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

3. 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
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

4. create landing outcome label from Outcome column

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

5. Export to file

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

[Link GitHub]

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

Scatter plots visually display the relationship between variables, showcasing their correlation.

Bar graphs, on the other hand, illustrate the relationship between numeric and categorical variables.

Line graphs are effective in depicting data variables and their trends. They can provide insights into global behavior and aid in making predictions for unseen data.

Link GitHub

EDA with SQL

- We performed SQL queries to gather and understand data from dataset:
  - Displaying the names of the unique launch sites in the space mission.
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS).
  - Display average payload mass carried by booster version F9 v1.1.
  - List the date when the first successful landing outcome in ground pad was achieved.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - List the total number of successful and failure mission outcomes.
  - List the names of the booster_versions which have carried the maximum payload mass.
  - List the records which will display the month names, faiilure landing_ouutcomes in drone ship, booster versions, launch_site for the months in year 2015.
  - Rank the count of successful landiing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order

Link GitHub

Build an Interactive Map with Folium

- The Folium map object represents a map centered on the NASA Johnson Space Center in Houston, Texas.
  - It displays a red circle at the coordinates of the NASA Johnson Space Center, labeled with its name (implemented using folium.Circle and folium.map.Marker).
  - It also shows red circles at the coordinates of each launch site, with labels indicating the launch site names (achieved using folium.Circle, folium.map.Marker, and folium.features.DivIcon).
  - To present multiple and different information for the same coordinates, the points are grouped into clusters using folium.plugins.MarkerCluster.
  - Markers are used to indicate successful and unsuccessful landings, with green representing successful landings and red representing unsuccessful landings (implemented using folium.map.Marker and folium.Icon).
  - Additionally, markers are placed to illustrate the distance between each launch site and key locations such as railways, highways, coastways, and cities. Lines are plotted between the markers to connect them (achieved using folium.map.Marker, folium.PolyLine, and folium.features.DivIcon).

Link GitHub

Build a Dashboard with Plotly Dash

- The dashboard consists of several components, including a dropdown, pie chart, rangeslider, and scatter plot.
  - The dropdown component (dash_core_components.Dropdown) enables the user to select a specific launch site or view data for all launch sites.
  - The pie chart (plotly.express.pie) displays the total number of successful and unsuccessful launches for the chosen launch site from the dropdown component.
  - The rangeslider (dash_core_components.RangeSlider) allows the user to select a payload mass within a predefined range.
  - The scatter plot (plotly.express.scatter) illustrates the relationship between two variables, specifically the success of launches versus the payload mass.

[Link GitHub]()

Predictive Analysis (Classification)

I. Data Preparation:

I. Load the dataset.

II. Normalize the data to ensure consistent scaling across features.

III. Split the data into training and test sets to evaluate model performance.

II. Model Preparation:

I. Select appropriate machine learning algorithms for the task.

II. Set parameters for each algorithm using GridSearchCV to optimize their performance.

III. Train the GridSearchCV models using the training dataset.

III. Model Evaluation:

I. Retrieve the best hyperparameters for each type of model determined by GridSearchCV.

II. Calculate the accuracy of each model using the test dataset.

III. Plot the Confusion Matrix to visualize the model's performance.

IV. Model Comparison:

I. Compare the models based on their accuracy metrics.

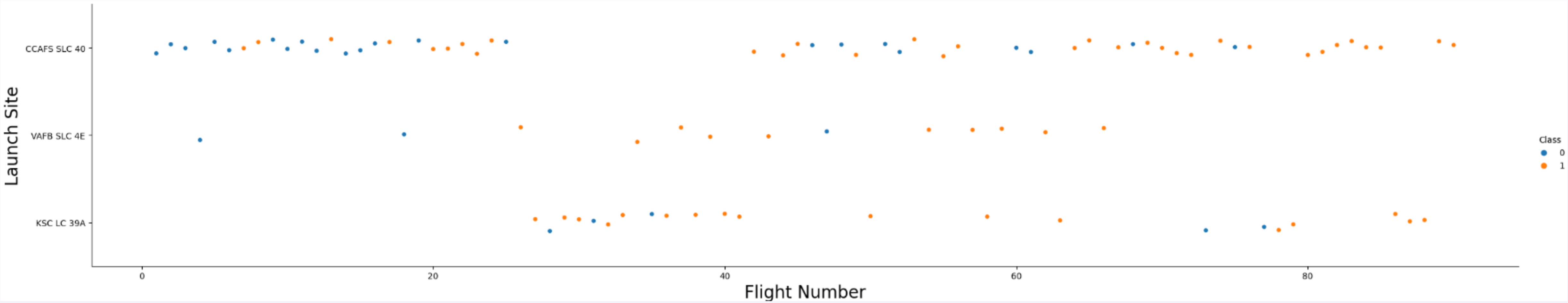
II. Identify the model with the highest accuracy as the top performer (refer to the Notebook for the specific results).

[Link GitHub]

Section 2

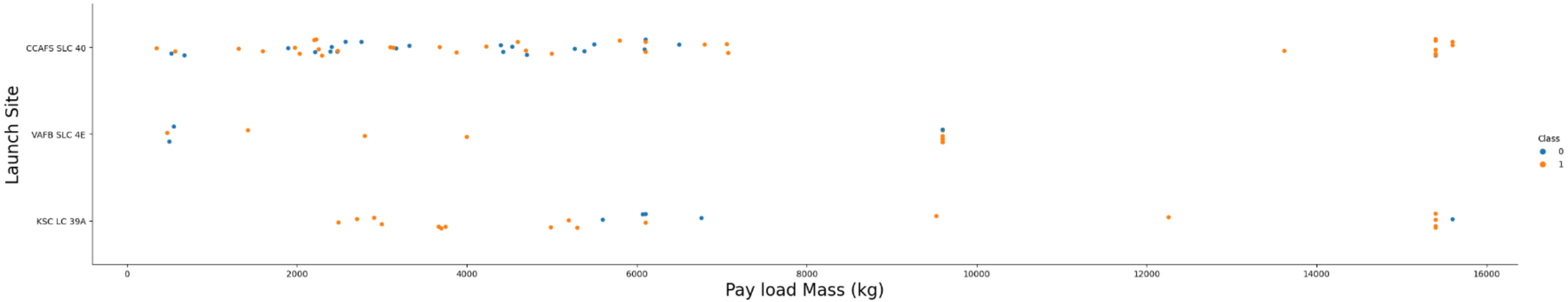
Insights drawn from EDA

Flight Number vs. Launch Site



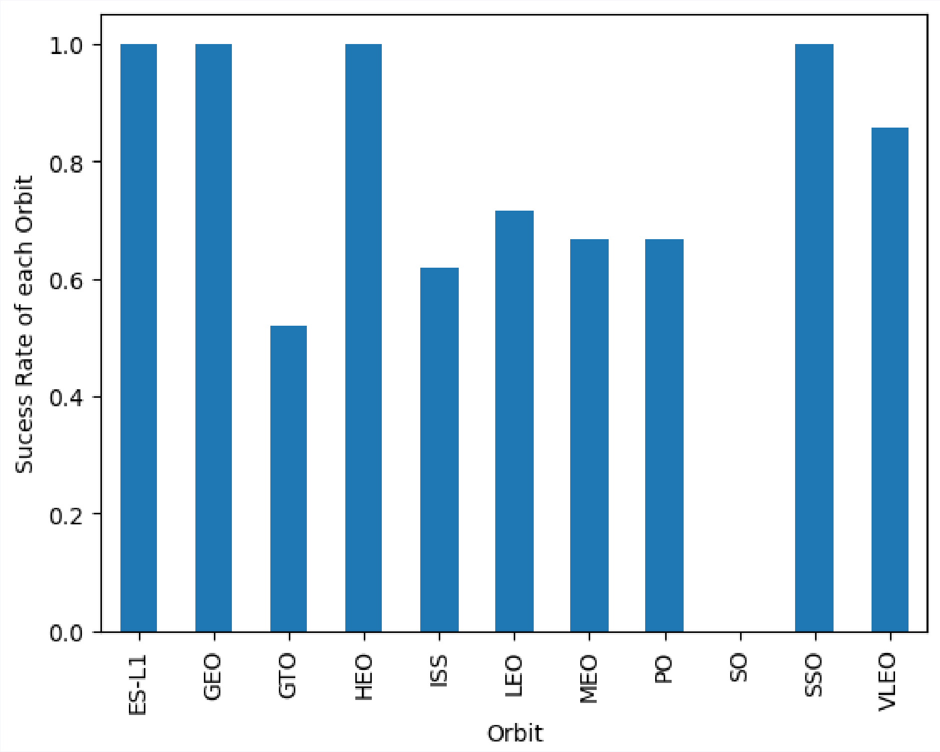
The observation that the success rate is increasing for each launch site indicates a positive trend in the performance of the launches over time. This finding suggests that the space agency or organization conducting these launches has been improving their processes, technologies, or decision-making, leading to a higher rate of successful missions.

Payload vs. Launch Site



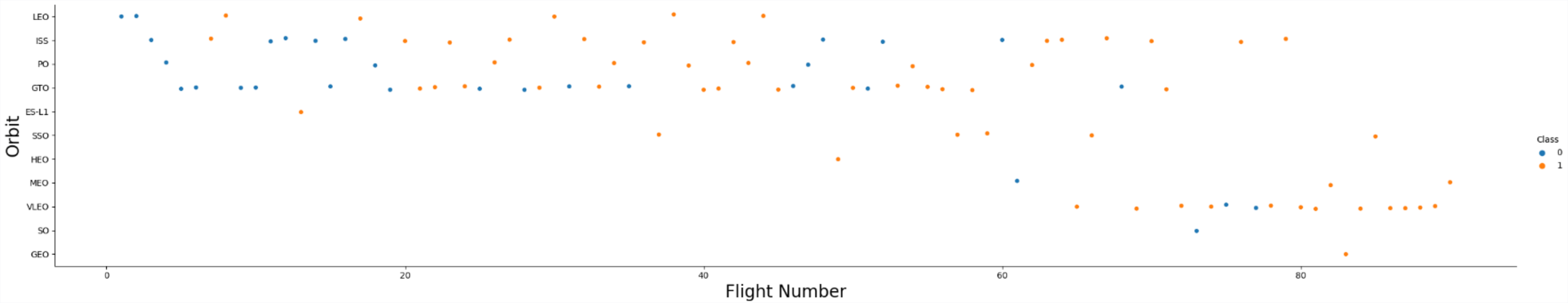
Indeed, the payload weight plays a crucial role in the success or failure of a rocket landing, and it is essential to find the right balance for each specific launch site and mission. Both an excessively heavy and too light payload can cause issues during landing.

Success Rate vs. Orbit Type



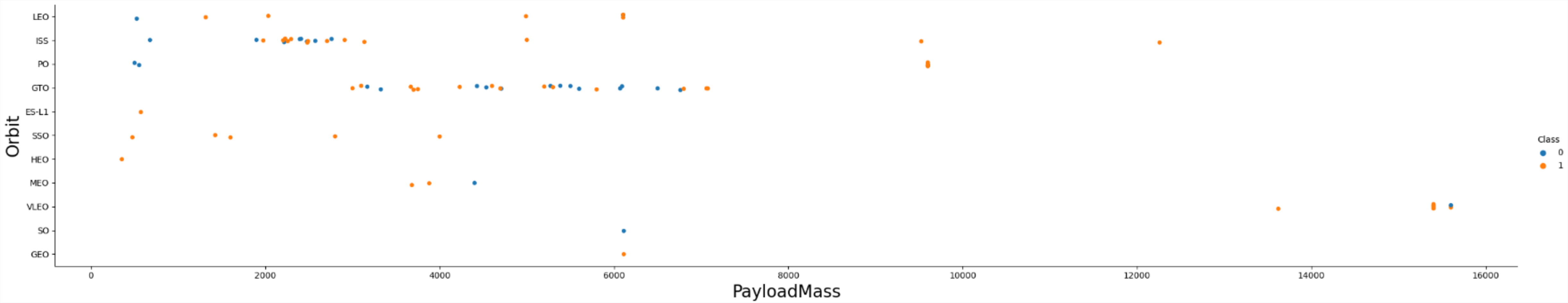
The plot provides valuable insights into the success rates for different orbit types, highlighting that ES-L1, GEO, HEO, and SSO orbits have the highest success rates.

Flight Number vs. Orbit Type



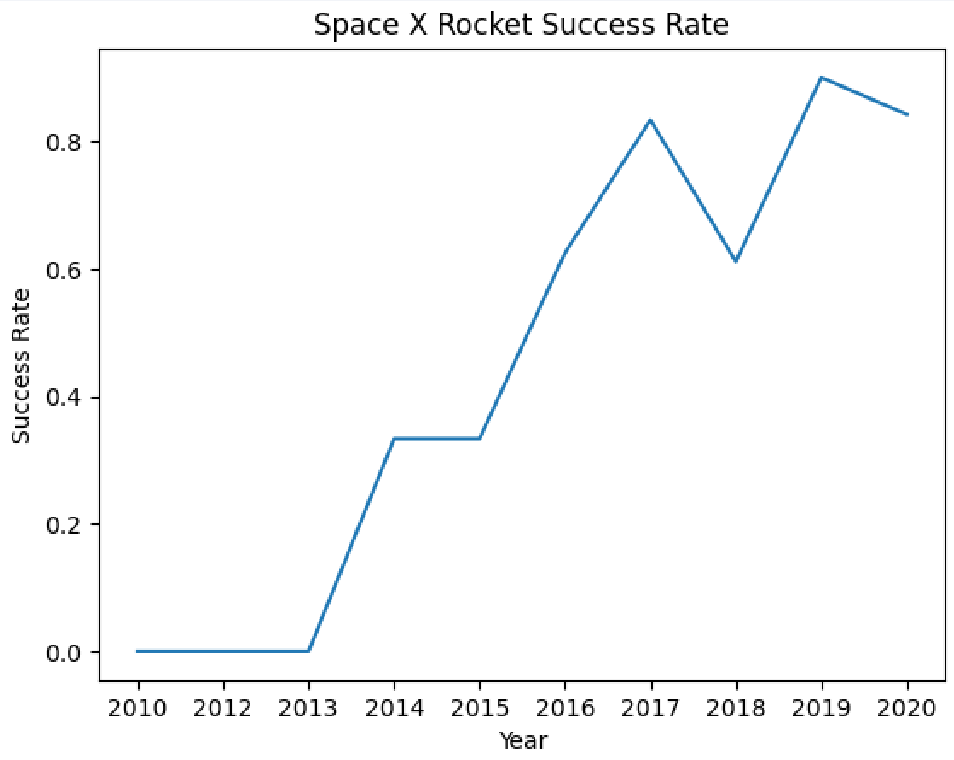
The observations we've made regarding the success rate and the number of flights for different orbits provide valuable insights into the complexities of space missions and how experience plays a significant role in achieving success.

Payload vs. Orbit Type



The influence of payload weight on the success rate of launches is indeed a critical factor to consider, and it can vary significantly depending on the specific orbit type.

Launch Success Yearly Trend



The observation of an increasing success rate for SpaceX rockets since 2013 aligns with the company's efforts, achievements, and continuous improvements over the years.

All Launch Site Names

The use of the DISTINCT keyword in a query allows for the removal of duplicate values in the specified column or columns. Specifically, in the context of we query involving LAUNCH_SITE, using DISTINCT will ensure that only unique values for the LAUNCH_SITE column are returned in the result set.

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

```
%sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

```
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

The WHERE clause, along with the LIKE clause, filters launch sites containing the substring "CCA.“

LIMIT 5 displays five records resulting from the filtering.

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

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

```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

This query calculates the sum of all payload masses where the customer is NASA with the designation "CRS."

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

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

```
 * sqlite:///my_data1.db
Done.
```

SUM("PAYLOAD_MASS__KG_")
45596.0

Average Payload Mass by F9 v1.1

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

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

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

```
 * sqlite:///my_data1.db
Done.
```

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

First Successful Ground Landing Date

Using this query, we aim to identify the earliest successful rocket landing. The WHERE clause is utilized to filter the dataset, ensuring that we retain only the records with a successful landing status. By applying the MIN function to the "date" column, we extract the record with the oldest date among the successful landings.

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

Hint:Use min function

```
%sql SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing_Outcome" LIKE '%Success%'
```

```
 * sqlite:///my_data1.db
Done.
```

MIN("DATE")
01/07/2020

Successful Drone Ship Landing with Payload between 4000 and 6000

This query retrieves the booster versions for which the landing was successful, and the payload mass falls within the range of 4000 to 6000 kg. The WHERE and AND clauses are used to filter the dataset, ensuring that only records meeting both conditions are included in the result.

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

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

```
 * sqlite:///my_data1.db
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

Using the first SELECT statement, we display the subqueries that produce results. The initial subquery calculates the count of successful missions, while the second subquery calculates the count of unsuccessful missions. The WHERE clause, followed by the LIKE clause, filters the mission outcomes. The COUNT function then tallies the records that meet the specified filtering conditions.

List the total number of successful and failure mission outcomes

```
%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
```

```
 * sqlite:///my_data1.db
Done.
```

SUCCESS	FAILURE
100	1

Boosters Carried Maximum Payload

We employed a subquery to filter the data, retrieving only the heaviest payload mass using the MAX function. The primary query utilizes the results from the subquery and returns distinct booster versions (SELECT DISTINCT) associated with the heaviest payload mass.

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

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

```
 * sqlite:///my_data1.db
Done.
```

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

2015 Launch Records

The purpose of this query is to obtain the month, booster version, and launch site where the landing was unsuccessful, and the landing date occurred in the year 2015. The Substr function is utilized to process the date in order to extract either the month or the year. Specifically, Substr(DATE, 4, 2) is used to retrieve the month, and Substr(DATE, 7, 4) is used to extract the year from the date.

```
%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'
```

```
 * sqlite:///my_data1.db
Done.
```

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

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

The purpose of this query is to retrieve the landing outcomes and their corresponding counts for successful missions that occurred between 04/06/2010 and 20/03/2017. The GROUP BY clause is utilized to group the results by landing outcome, and the ORDER BY COUNT DESC arranges the results in descending order based on their count.

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.

```
%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 ;
```

```
 * sqlite:///my_data1.db
Done.
```

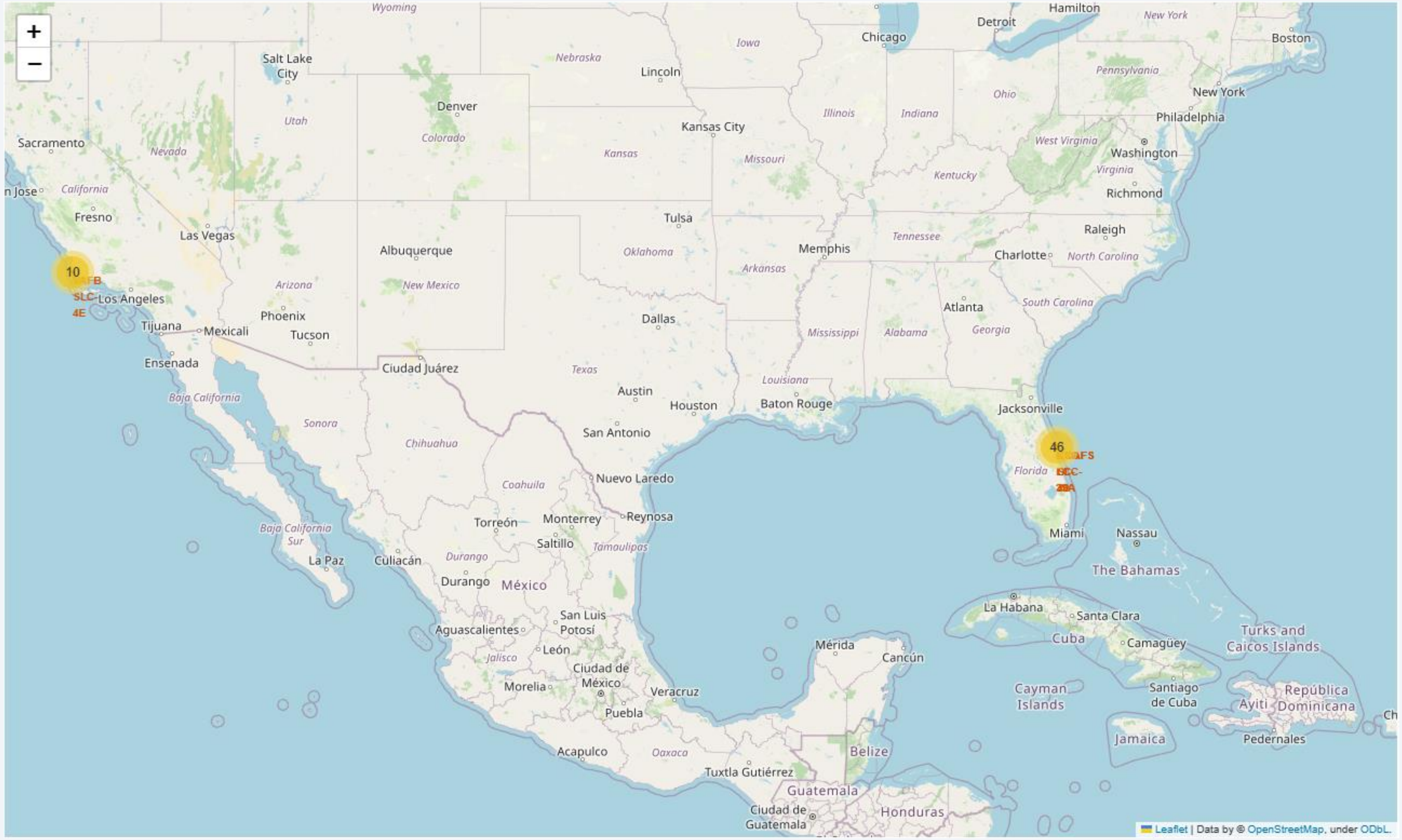
Landing_Outcome	COUNT("LANDING_OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	7

Section 3

Launch Sites Proximities Analysis

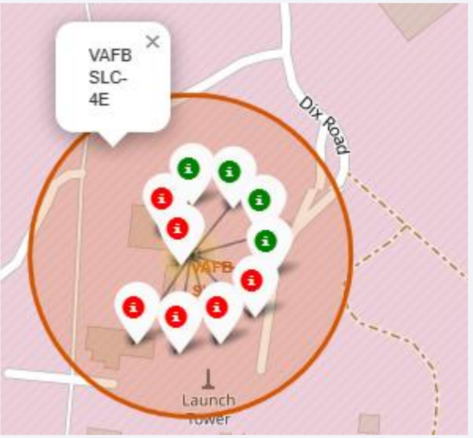
Folium Map Displaying Ground Stations

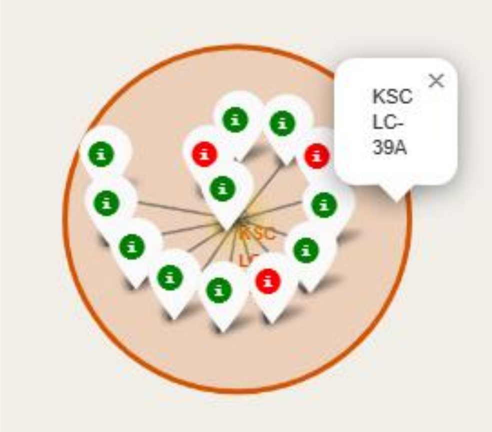
It is observed that Space X launch sites are situated along the coast of the United States.



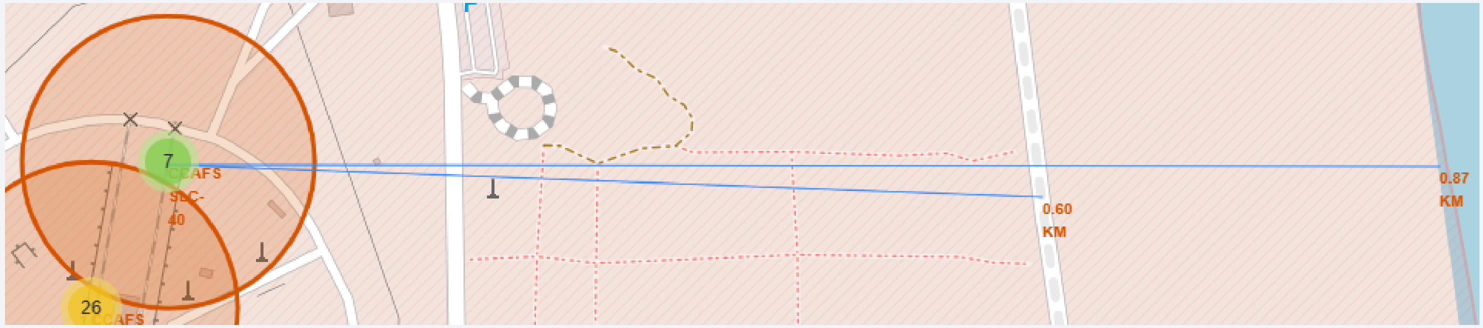
Folium Map – Color Labeled Markers

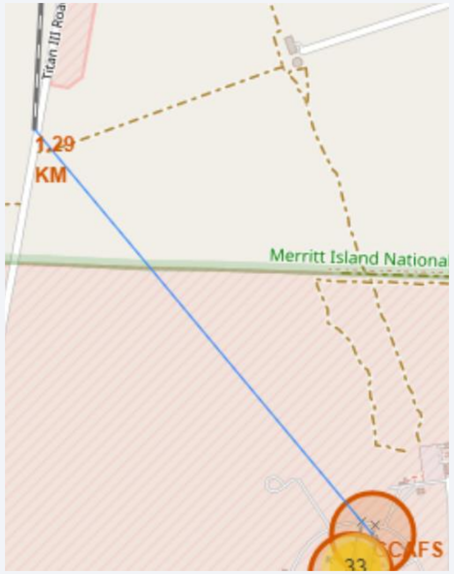
The Folium map exhibits ground stations, where **green** markers indicate successful launches, and **red** markers represent unsuccessful launches. It is evident that KSC LC-39A has a higher launch success rate compared to other ground stations.





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

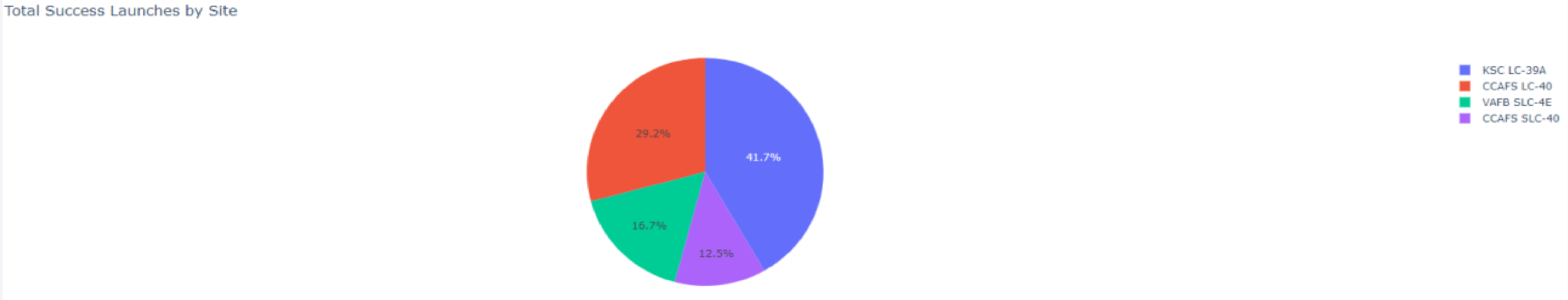
Do CCAFS SLC-40 keeps certain distance away from cities? No

Section 4

Build a Dashboard with Plotly Dash

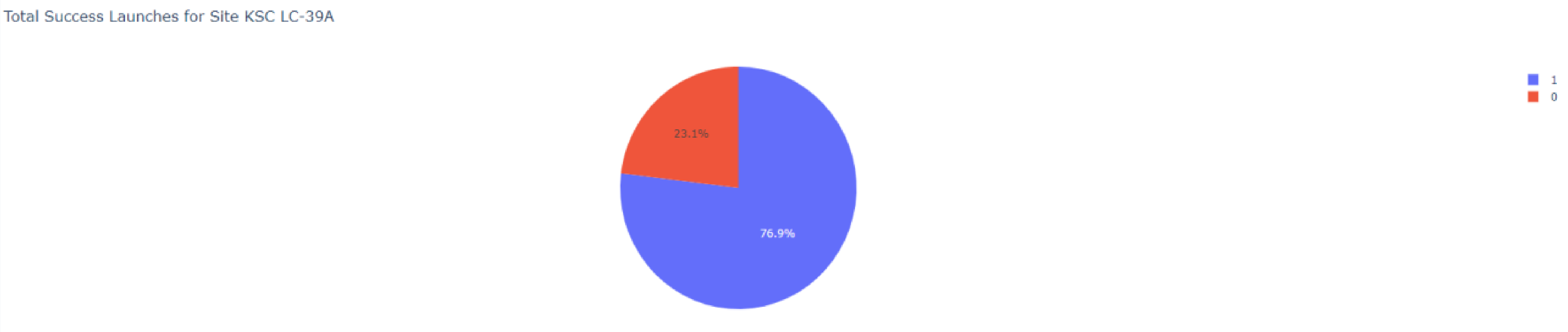
Dashboard – Total Success by Site

Indeed, it is observed that KSC LC-39A has the highest success rate of launches compared to other launch sites.



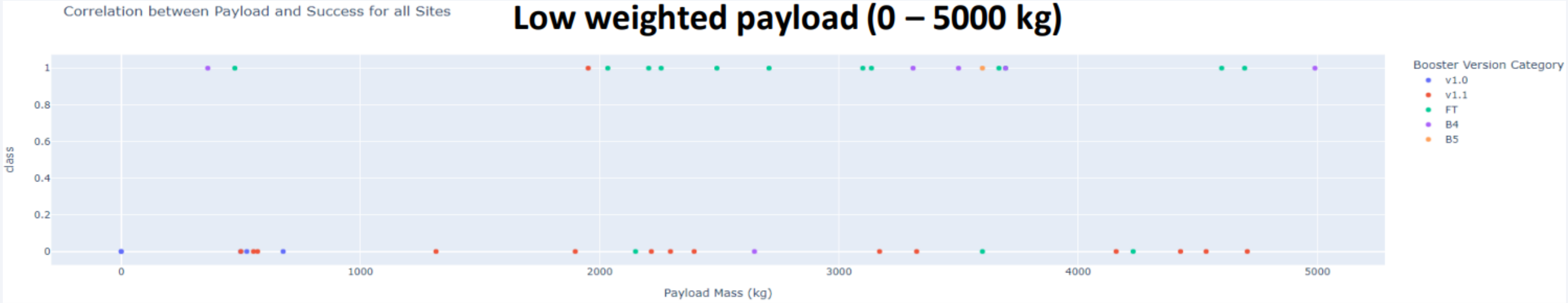
Dashboard – Total success launches for Site KSC LC-39A

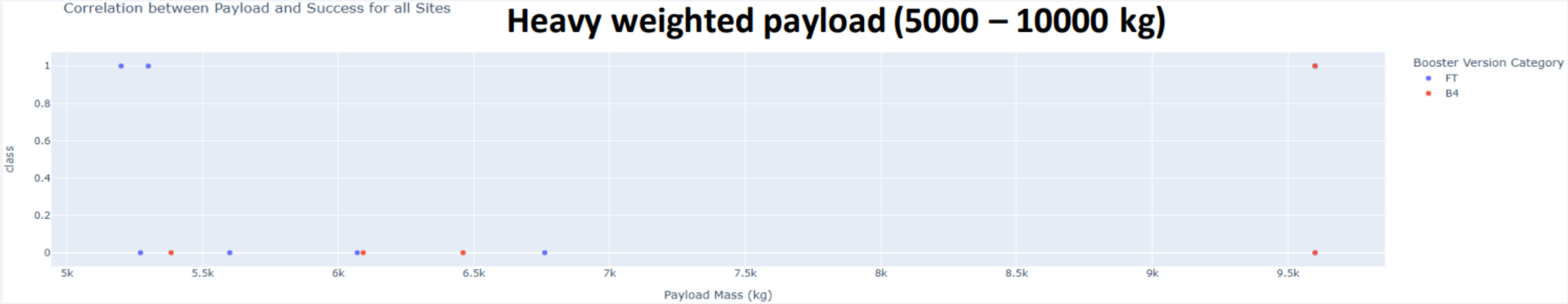
As per the data, KSC LC-39A has achieved a success rate of 76.9% for its launches, and concurrently, it has experienced a failure rate of 23.1%.



Dashboard – Payload Mass vs Outcome for all Sites with Different Payload Mass Selected

Payloads with lower weight exhibit a higher success rate compared to heavier payloads.



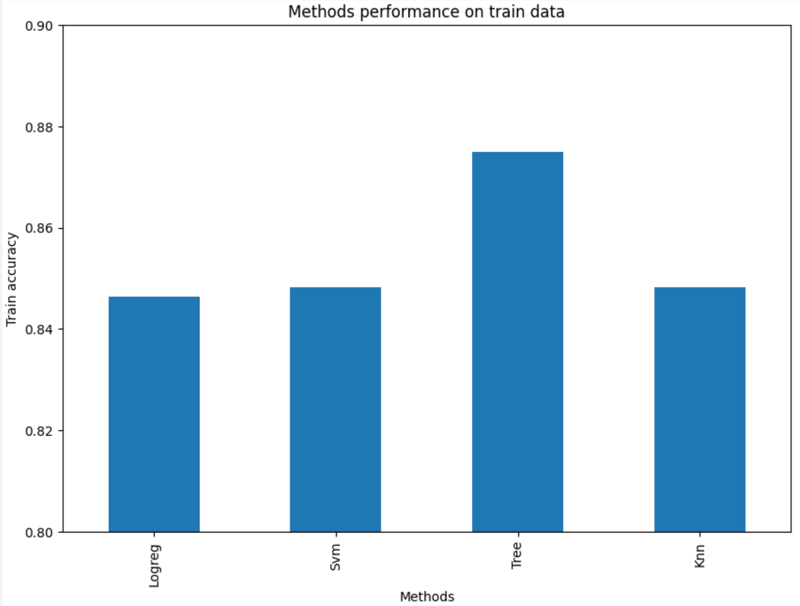


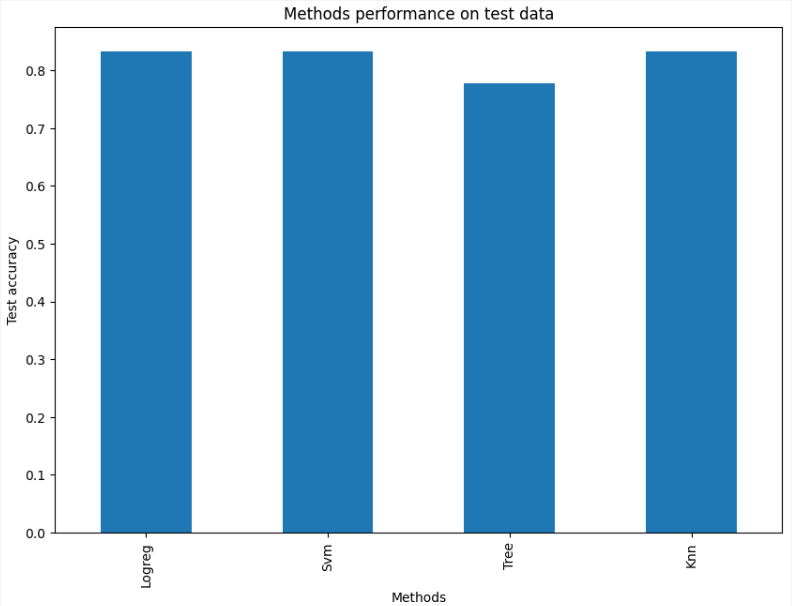
Section 5

Predictive Analysis (Classification)

Classification Accuracy

	Accuracy Train	Accuracy Test
Tree	0.875000	0.777778
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333





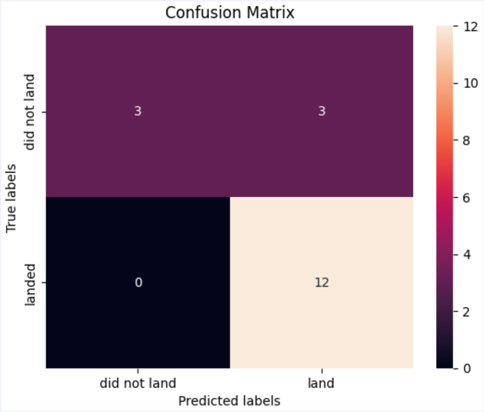
In the accuracy train, all methods demonstrated similar performance, making it challenging to make a definitive decision based solely on the existing data. While obtaining more test data could help further evaluate and compare the methods, the need to select one immediately necessitates a choice.

Given the current circumstances, the preference is to opt for either the KNN (K-Nearest Neighbors) or SVM (Support Vector Machine) method.

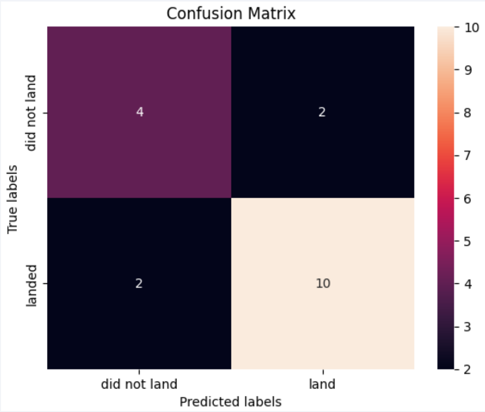
Confusion Matrix

If the test accuracy for all models is equal and the confusion matrices are identical, it suggests that the models are performing similarly in terms of overall accuracy. However, the identification of a common issue with false positives indicates that there is a problem with the models' ability to correctly classify negative instances.

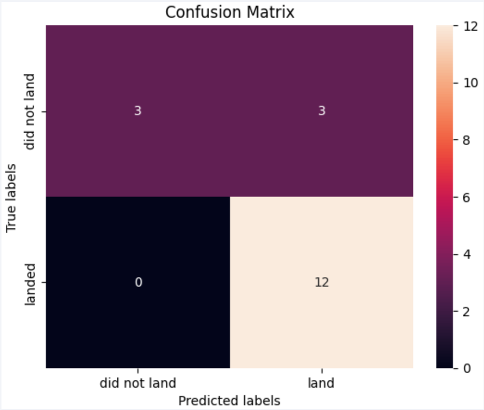
Logistic Regression



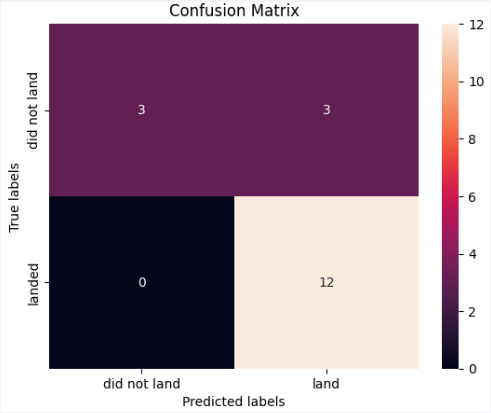
Decision Tree



KNN



SVM



Conclusions

1. The success of a mission can be attributed to various factors, such as the launch site, the orbit, and particularly the number of previous launches. Indeed, we can assume that knowledge gained from past launches contributes to transitioning from launch failures to successful missions.

2. The orbits with the highest success rates are GEO, HEO, SSO, and ES-L1.

3. Depending on the orbit, the payload mass can significantly influence the success of a mission. Some orbits require light or heavy payload masses. However, in general, missions with lower payload masses tend to perform better than those with heavier payloads.

4. Based on the available data, we cannot determine the reasons why some launch sites perform better than others (e.g., KSC LC-39A being the best launch site). To address this issue, we could acquire additional atmospheric or other relevant data for analysis.

5. For this dataset, we have chosen the KNN (K-Nearest Neighbors) or SVM (Support Vector Machine) algorithms as the best model, even though the test accuracy is the same across all the models used. The decision was made based on the Decision Tree Algorithm's superior train accuracy.

Thank you!