Winning Space Race with Data Science

Kevin G Aguilar
January 24, 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

The project aims to create a machine-learning pipeline to predict the success of first stage landings for Falcon 9 rocket launches by Space X. This information can be used to determine the cost of a launch and to aid in competition with other providers, whose costs can reach 165 million dollars. The main problems to be addressed are identifying the factors that affect successful landings and understanding the interactions among these factors, as well as determining the necessary operating conditions for a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

The methodology for data collection involved utilizing the SpaceX API and web scraping from Wikipedia. Data was then cleaned and organized through data wrangling techniques such as one-hot encoding for categorical features. Exploratory data analysis (EDA) was conducted using both visualization and SQL. Interactive visual analytics were performed using tools such as Folium and Plotly Dash. Predictive analysis was performed using classification models and methods for building, tuning, and evaluating these models were employed.

Data Collection

The data was gathered using a combination of methods, including making GET requests to the SpaceX API, decoding the response as JSON, converting it to a pandas dataframe, and cleaning, checking for missing values, and filling in any missing data where necessary. Additionally, web scraping was conducted on Wikipedia's Falcon 9 launch records using BeautifulSoup in order to extract the launch records as an HTML table, parse them, and convert them to a pandas dataframe for further analysis.

Data Collection – SpaceX API

We employed GET requests to the SpaceX API to gather data [6,7], and then performed cleaning, basic data wrangling [27], and formatting on the obtained data [11]. The notebook containing these processes can be found at the following link: Github

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

In [11]:
```
# Use json_normalize meethod to convert the json result into a dataframe
rj = response.json()

data = pd.json_normalize(rj)
```

In [27]:
```
# Calculate the mean value of PayloadMass column
pm_mean = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan, pm_mean, inplace=True)
```

Data Collection - Scraping

Using web scraping techniques, we extracted the Falcon 9 launch records from Wikipedia using BeautifulSoup [4, 6]. We then parsed the table [7, 8, 12] and converted it into a Pandas dataframe [14] for further analysis. The notebook containing these processes can be found at the following link: [Github]

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [6]:
```
# use requests.get() method with the provided static_url
response = requests.get(static_url)
# assign the response to a object
```

In [7]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

In [8]:
```
# Use soup.title attribute
soup.title
```

Out[8]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

In [12]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

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

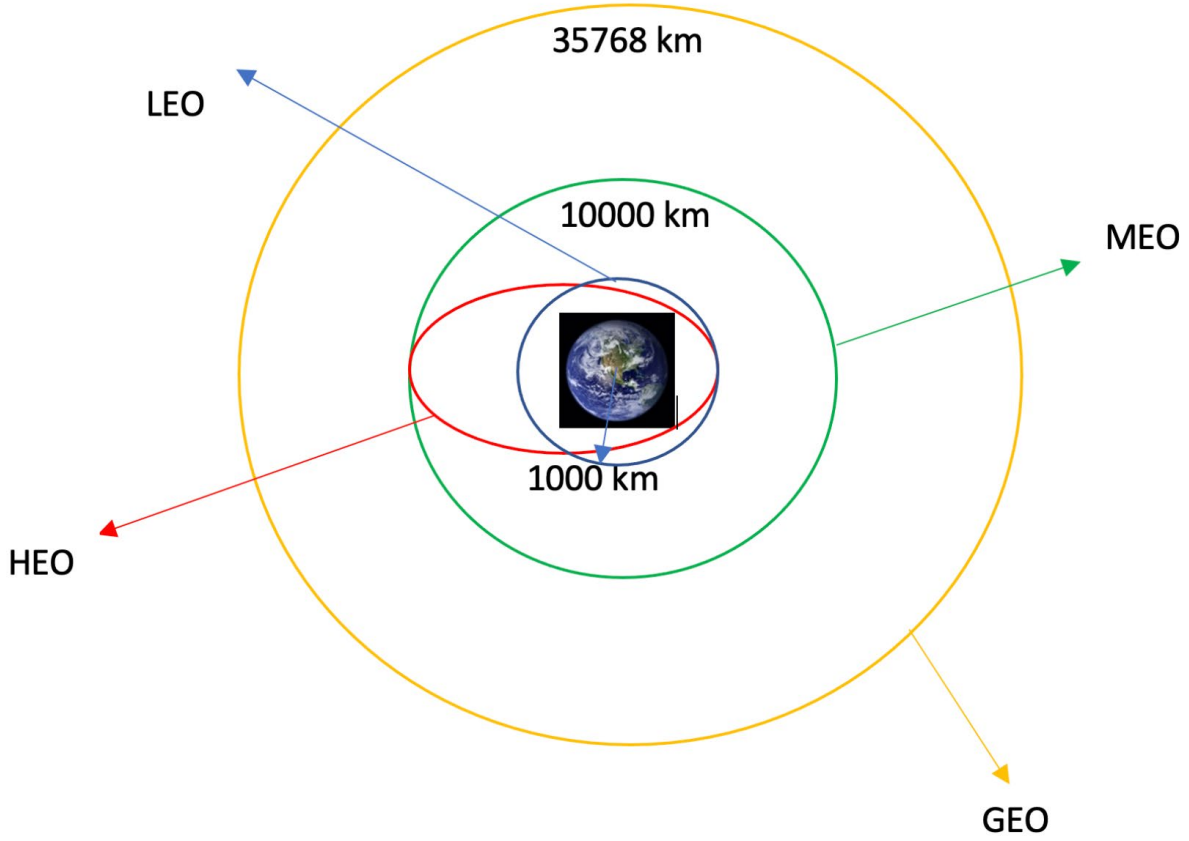
In [14]:
```
launch_dict= dict.fromkeys(column_names)

# 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']=[]
```

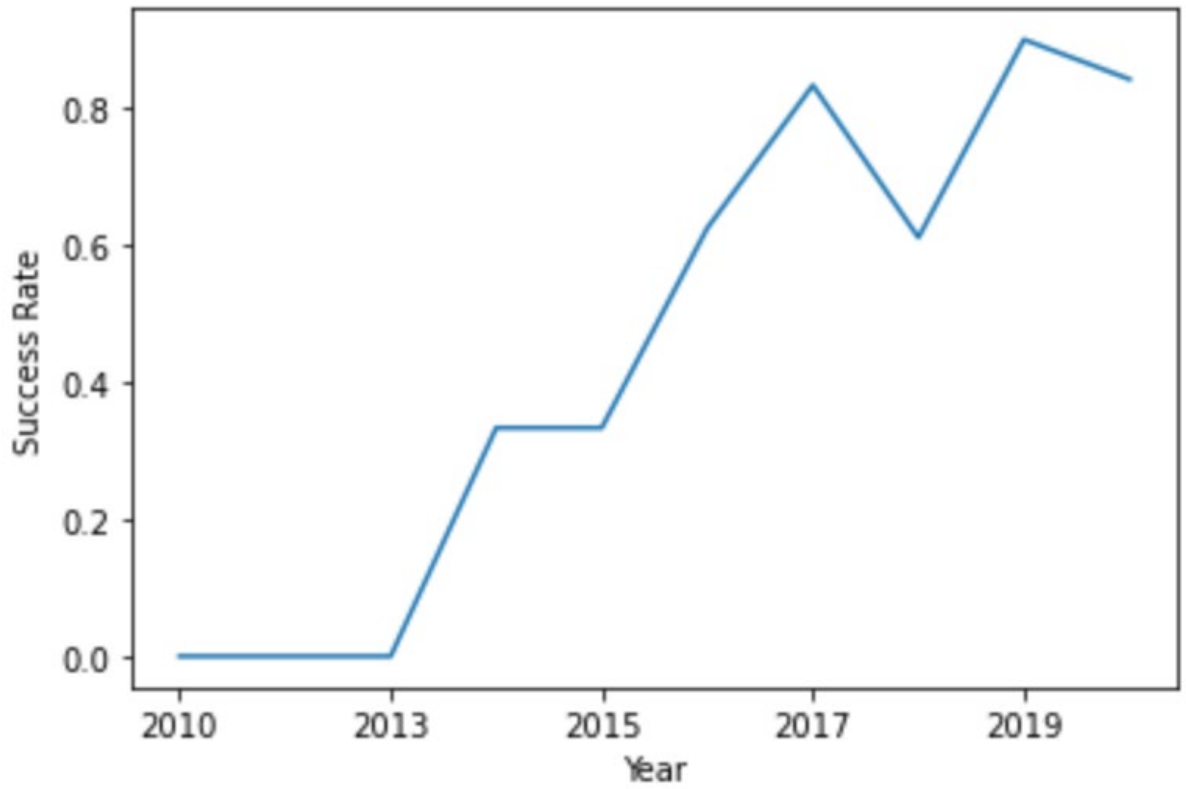
Data Wrangling

After conducting exploratory data analysis, the training labels were established. The number of launches at each site and the frequency of each orbit were calculated. Additionally, a landing outcome label was derived from the outcome column and the results were exported to CSV. The notebook can be accessed here: Github



EDA with Data Visualization

We examined the data by visualizing the correlation between flight number and launch site, payload and launch site, the success rate of each orbit type, flight number and orbit type, and the launch success trend over time. The notebook can be found on Github.



EDA with SQL

We imported the SpaceX dataset into a PostgreSQL database within the Jupyter notebook. Using SQL for exploratory data analysis, we extracted insights from the data by writing queries, such as identifying: unique launch sites in the space mission, total payload mass carried by boosters launched by NASA (CRS), average payload mass carried by booster version F9 v1.1, the total number of successful and failed mission outcomes, and failed landing outcomes on drone ships, their booster version, and launch site names. The notebook can be accessed on Github.

Build an Interactive Map with Folium

In this research, we employed geospatial visualization techniques to explore the distribution of launch sites for the SpaceX mission. We utilized the Folium library to create an interactive map, which was populated with markers, circles, and lines to represent the launch sites and their corresponding success or failure outcomes. The launch outcomes were classified into binary categories, with 0 representing failure and 1 representing success. Through the use of color-labeled marker clusters, we were able to identify launch sites with a relatively high success rate. Additionally, we computed the distances between launch sites and nearby geographical features such as railways, highways, and coastlines, and analyzed if there is a correlation between the proximity to such features and the launch site location in relation to cities. Github

Build a Dashboard with Plotly Dash

- In this analysis, we developed an interactive dashboard using Plotly Dash. The dashboard comprises various visualizations that enable the exploration of the SpaceX launch dataset. Specifically, we utilized pie charts to depict the total number of launches by specific sites. Additionally, we employed a scatter plot to investigate the relationship between the outcome of the launch and the payload mass (in kilograms) for different booster versions. The code and accompanying documentation can be accessed on Github.

Predictive Analysis (Classification)

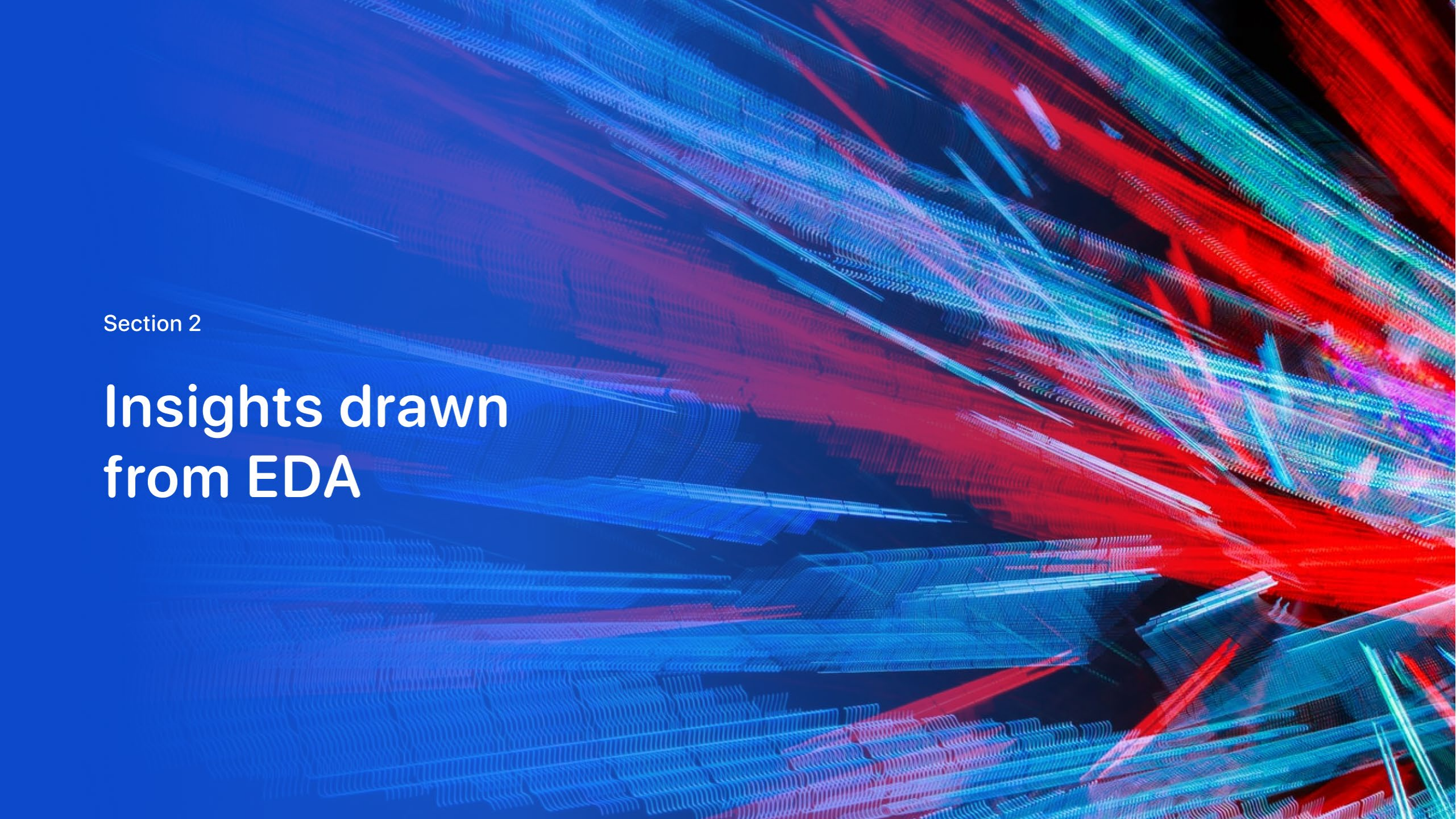
- In this research, we utilized the Python libraries NumPy and pandas to import and preprocess the data. The data was transformed and split into training and testing sets to facilitate the evaluation of machine learning models. Various models were constructed and their hyperparameters were optimized using GridSearchCV. The performance of the models was evaluated using accuracy as the metric. Furthermore, we employed feature engineering and algorithm tuning techniques to improve the performance of the models. Through this process, we were able to identify the best-performing classification model. The code and accompanying documentation can be accessed on Github.

Results

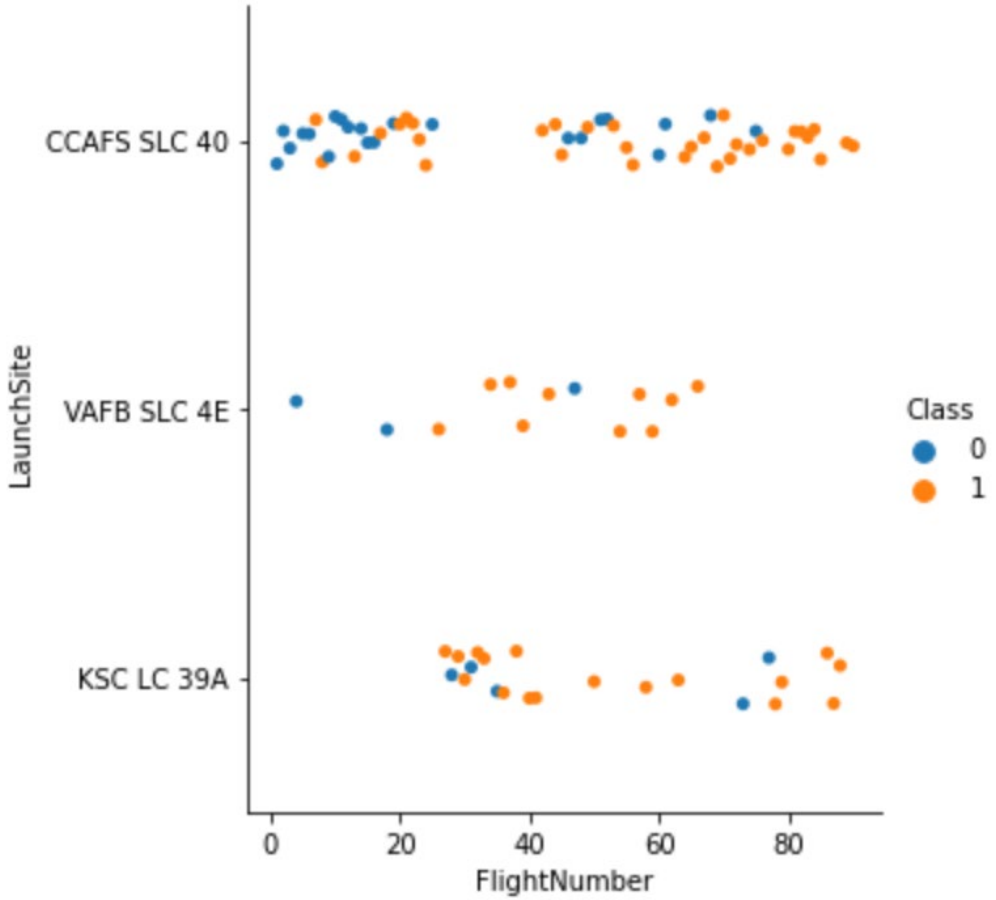
EXPLORATORY DATA ANALYSIS RESULTS

INTERACTIVE ANALYTICS DEMO IN SCREENSHOTS

PREDICTIVE ANALYSIS RESULTS


Section 2
Insights drawn
from EDA

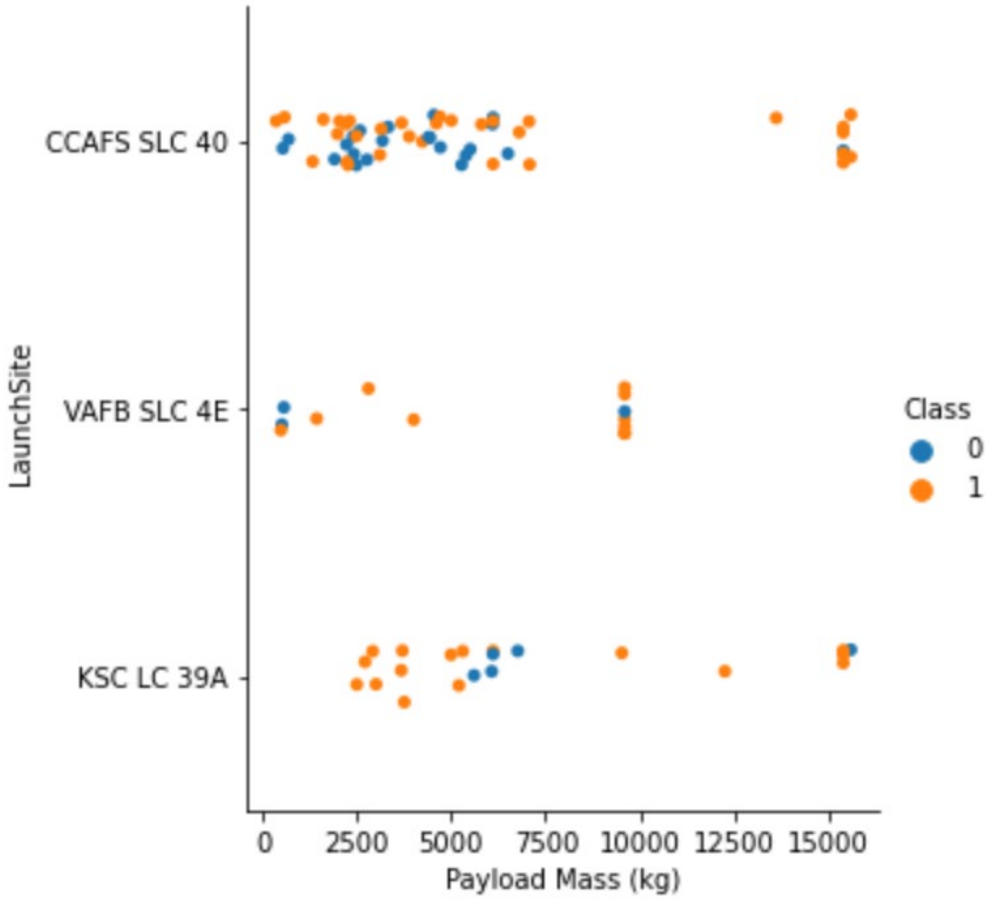
Flight Number vs. Launch Site



We discovered that a correlation exists between the number of flights at a launch site and the success rate at that site, with an increase in the number of flights being indicative of a higher success rate.

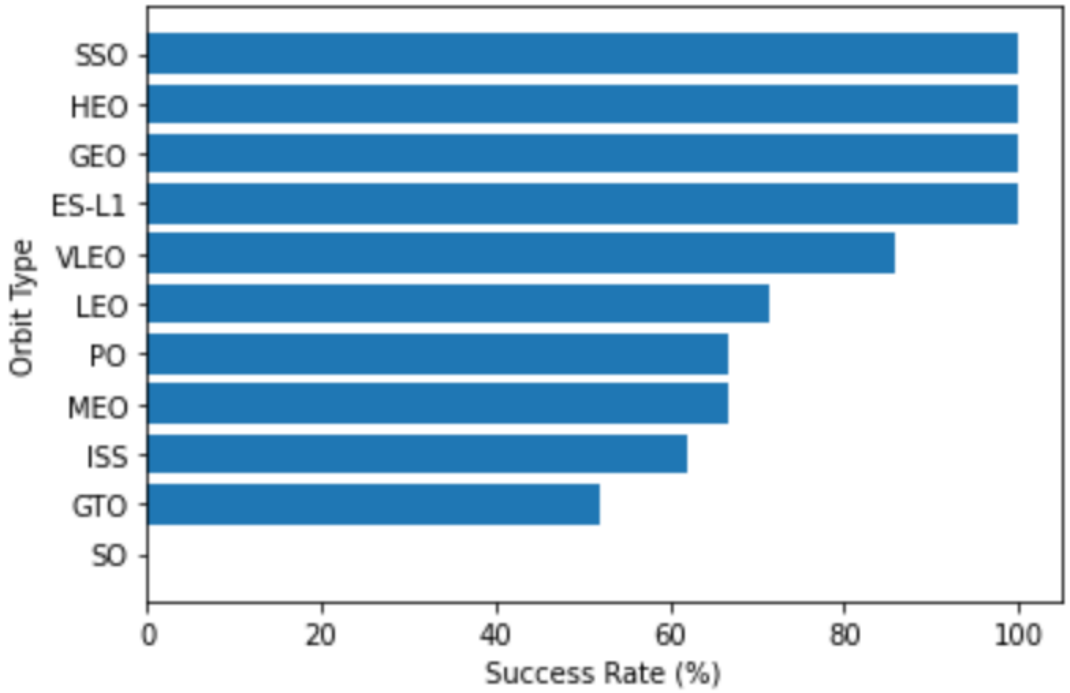
Payload vs. Launch Site

- We established that at Launch Site CCAFS SLC 40, a correlation exists between payload mass and launch success, with an increase in payload mass being associated with a higher success rate.



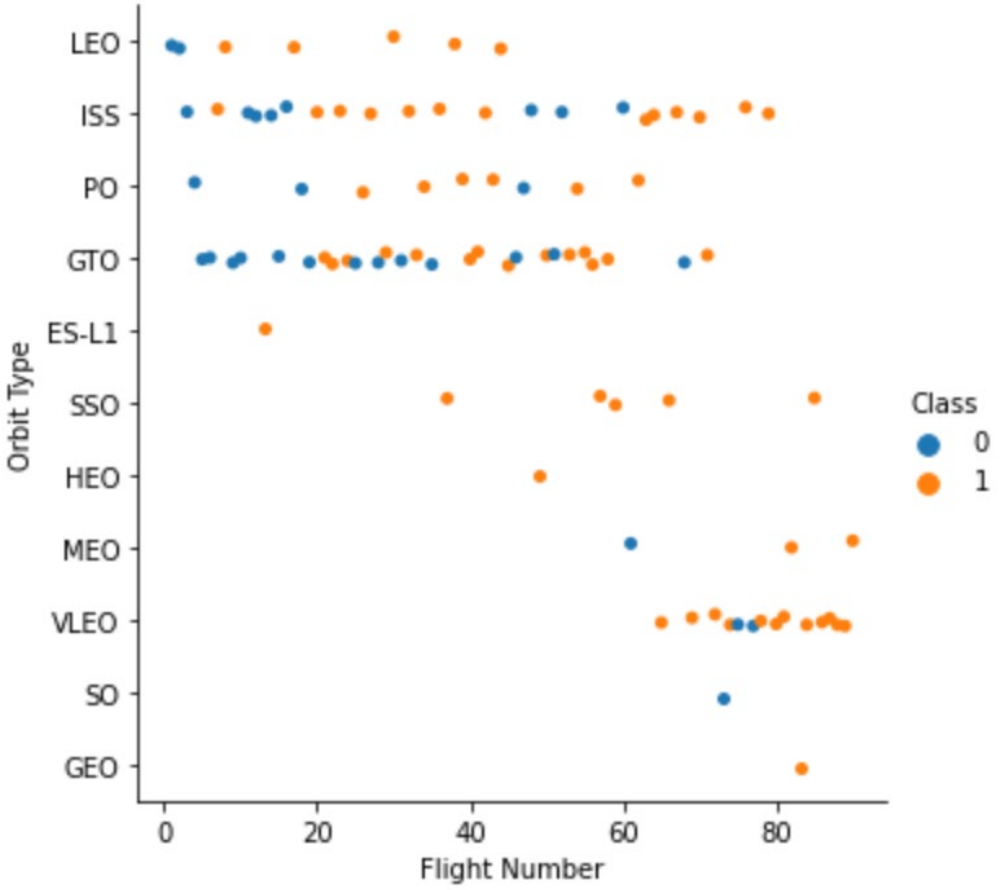
Success Rate vs. Orbit Type

In this data analysis, we focused on examining the success rate of various orbits used by SpaceX. The data was analyzed, and it was found that orbits ES-L1, GEO, HEO, SSO, and VLEO had the highest success rate among the orbits in the dataset. This information can be useful in determining which orbits are more likely to be successful for future missions. Further analysis is needed to understand the underlying factors that contribute to the success rate of these orbits.



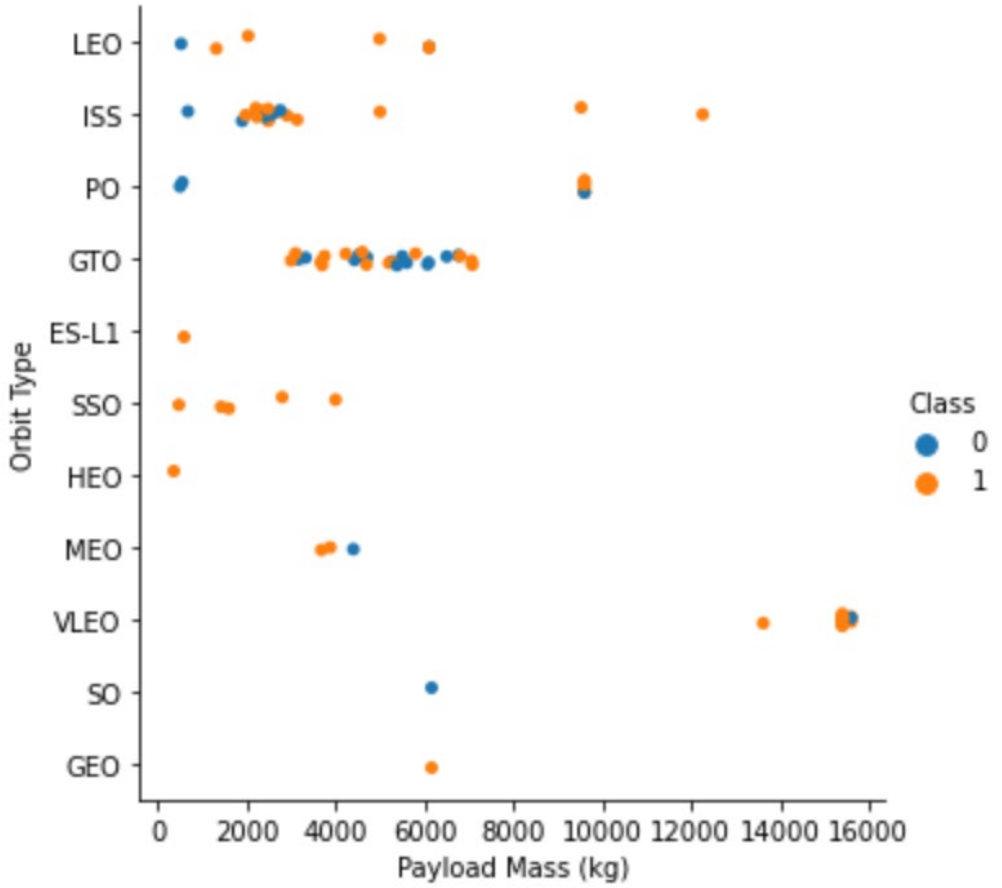
Flight Number vs. Orbit Type

We explored the relationship between flight number and orbit type. We can see that there is a correlation between flight number and success rate in the LEO orbit. As the number of flights increases in the LEO orbit, the success rate also increases. On the other hand, we do not observe any relationship between flight number and orbit type in the GTO orbit. These findings suggest that the number of flights in the LEO orbit may be a significant factor in determining the success rate of missions in that orbit.

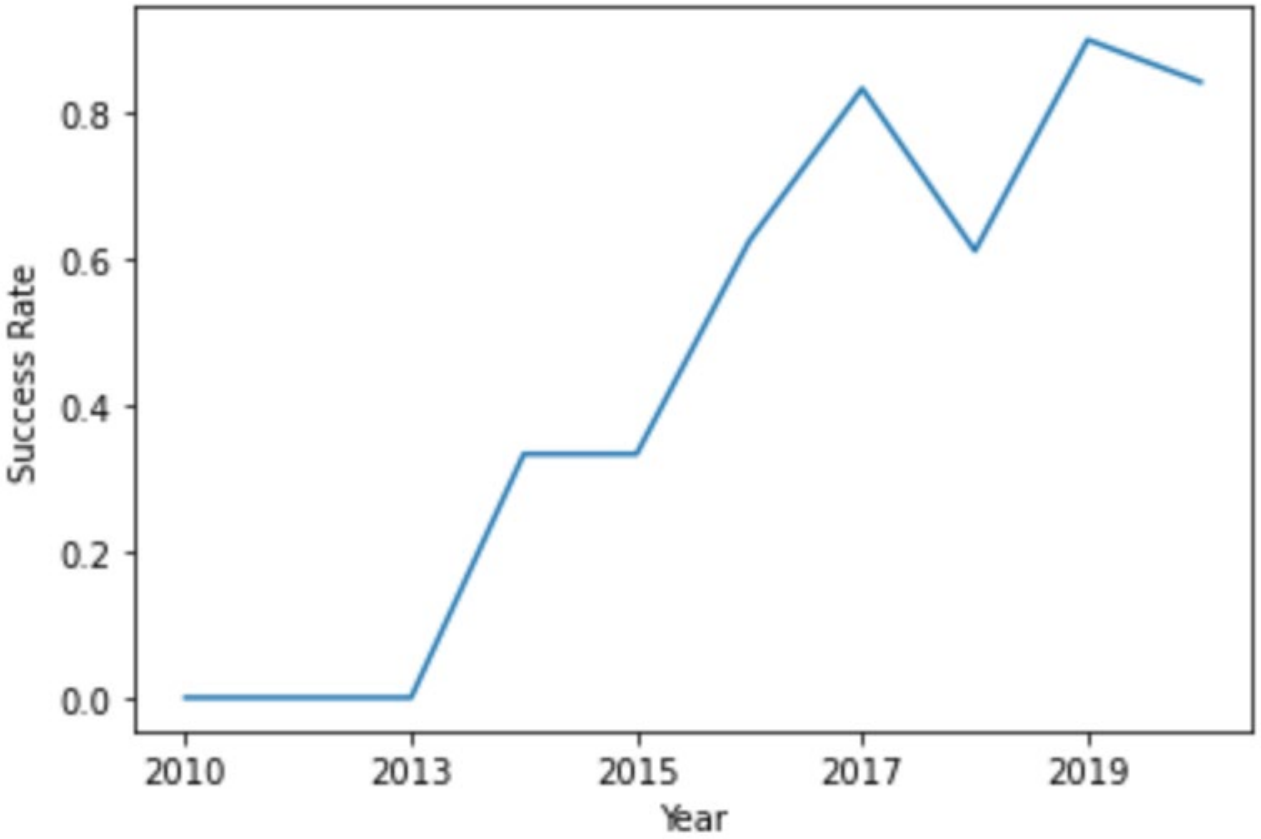


Payload vs. Orbit Type

An analysis of the data revealed that for payloads of significant mass, the success rate of landing is notably higher for PO, LEO, and ISS orbits. This information suggests that these orbits may be better suited for missions involving heavy payloads.



Launch Success Yearly Trend



Analysis of the data revealed that the success rate of launches has been increasing over time. The following plot illustrates this trend, showing that the success rate began to rise in 2013 and has continued to increase through 2020. This information suggests that SpaceX has been improving the success rate of its launches over time, and it could help forecast future success rates and determine the most favorable launch dates.

All Launch Site Names

In order to extract the unique launch sites from the SpaceX data, we employed the SQL DISTINCT statement after the SELECT statement. This statement enabled us to retrieve unique values in the column "launch sites" from the SPACEXTBL dataset and exclude duplicates. This operation was useful in identifying the specific locations where SpaceX launches have taken place.

In [12]:
```
%sql SELECT DISTINCT(LAUNCH_SITE) from SPACEXTBL
```

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

Out[12]:

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

Launch Site Names Begin with 'CCA'

We employed a SQL query to filter and display a subset of records from the SpaceX dataset by using the LIMIT clause in combination with the WHERE clause. Specifically, we used the WHERE clause to filter the "launch sites" column by selecting only those records whose values begin with "CCA" using the LIKE statement. Additionally, we used the LIMIT clause to limit the number of records displayed to five. This query allowed us to extract and display specific launch site records that matched a specific pattern, in this case, launch sites that begin with 'CCA'.

In [13]:
```
%sql SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'CCA%' LIMIT 5
```

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

Out[13]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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	Success	Failure (parachute)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

We used a SQL query to calculate the total payload mass carried by boosters from NASA (CRS) in the SpaceX dataset. The query employs the SUM() aggregate function after the SELECT statement on the "Payload Mass (kg)" column to determine the sum of all payload mass values in this column. Additionally, we used the AS statement to label the resulting column, naming it "Payload Mass kg Total." The query returned a total payload mass of 45,596 kg.

In [14]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as PAYLOAD_MASS_KG_TOTAL FROM SPACEXTBL WHERE Customer = 'NASA (CRS)'
```

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

Out[14]:

PAYLOAD_MASS_KG_TOTAL
45596

Average Payload Mass by F9 v1.1

We employed a SQL query to determine the average payload mass carried by the booster version F9 v1.1 in the SpaceX dataset. The query used the AVG() aggregate function after the SELECT statement on the "Payload Mass (kg)" column, this was done to calculate the average payload mass of all values in this column. The query returned an average payload mass of 2,928.4 kg for booster version F9 v1.1.

```
In [15]: %sql SELECT AVG(PAYLOAD_MASS__KG_) as AVG_PAYLOADMASS FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1'

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

Out[15]:

AVG_PAYLOADMASS
2928.4

First Successful Ground Landing Date

The analysis of the data revealed that the first successful landing outcome on a ground pad occurred on January 05, 2017. This information provides insight into the development of SpaceX's landing capabilities. It can be used as a reference point for future landing milestones.

```
In [23]: %sql SELECT MIN(DATE) as FIRST_SUCCESSFUL_LANDING_DATE FROM SPACEXTBL WHERE [LANDING _OUTCOME] = 'Success (ground pad)'

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

Out[23]:

FIRST_SUCCESSFUL_LANDING_DATE
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

We employed a SQL query to filter for boosters that have successfully landed on a drone ship and have a payload mass greater than 4000 kg but less than 6000 kg. This was achieved by using the WHERE clause and applying the AND condition to filter the data from the "Outcome" and "Payload Mass (kg)" columns. This query allowed us to extract specific records that met the specified criteria and provided insight into the successful landing characteristics of these boosters.

In [26]:
```
%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.
```

Out[26]:

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

Total Number of Successful and Failure Mission Outcomes

We employed a SQL query to group the data based on the "MissionOutcome" column, which was a success or failure. This was achieved by using the GROUP BY clause on the "MissionOutcome" column, and applying the COUNT() function to determine the number of occurrences of each group. Additionally, we used the AS alias on the COUNT() function in order to label the resulting column. This query allowed us to extract and group data based on the "MissionOutcome" column and provided insight into the number of successful and failed outcomes in the dataset.

In [54]:
```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) as TOTAL FROM SPACEXTBL GROUP BY MISSION_OUTCOME
```

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

Out[54]:

Mission_Outcome	TOTAL
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

We utilized a SQL query to identify the booster that has carried the maximum payload in the SpaceX dataset. This was achieved by using a subquery in the WHERE clause, and applying the MAX() function on the "Payload Mass (kg)" column to determine the maximum payload mass. The subquery then compares the maximum payload mass with the payload mass of each booster, thus allowing us to identify the booster that has carried the maximum payload. This query provided insights into the capabilities of different booster versions in terms of payload mass.

In [55]:
```
%sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

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

Out[55]:

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

We employed a SQL query to extract specific records from the SpaceX dataset by using a combination of the WHERE clause, LIKE, AND, and BETWEEN conditions. Specifically, we used the WHERE clause to filter for failed landing outcomes, the LIKE statement to filter for "Drone Ship" in the "Landing Type" column, the AND condition to filter the "Booster Version" column and "Launch Site" column, and the BETWEEN condition to filter for the year 2015 This query allowed us to extract detailed information about failed landing outcomes that occurred on drone ships, their booster versions, and launch site names for the year 2015.

In [65]:
```
%sql SELECT [LANDING _OUTCOME], BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE [LANDING _OUTCOME] LIKE 'Failure 9(drop ship)' AND DATE BETWEEN '201
```

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

Out[65]:

Landing _Outcome	Booster_Version	Launch_Site

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

- We constructed a SQL query to extract and analyze landing outcomes from the SpaceX dataset. The query employed the SELECT statement to select the "Landing Outcomes" column and the COUNT() function to count the number of occurrences of each landing outcome. Additionally, we used the WHERE clause to filter the data by applying the BETWEEN condition on the landing outcome date column between 2010-06-04 and 2010-03-20. Furthermore, we applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcomes in descending order. This query allowed us to extract and analyze detailed information about landing outcomes that occurred within a specific time frame.

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

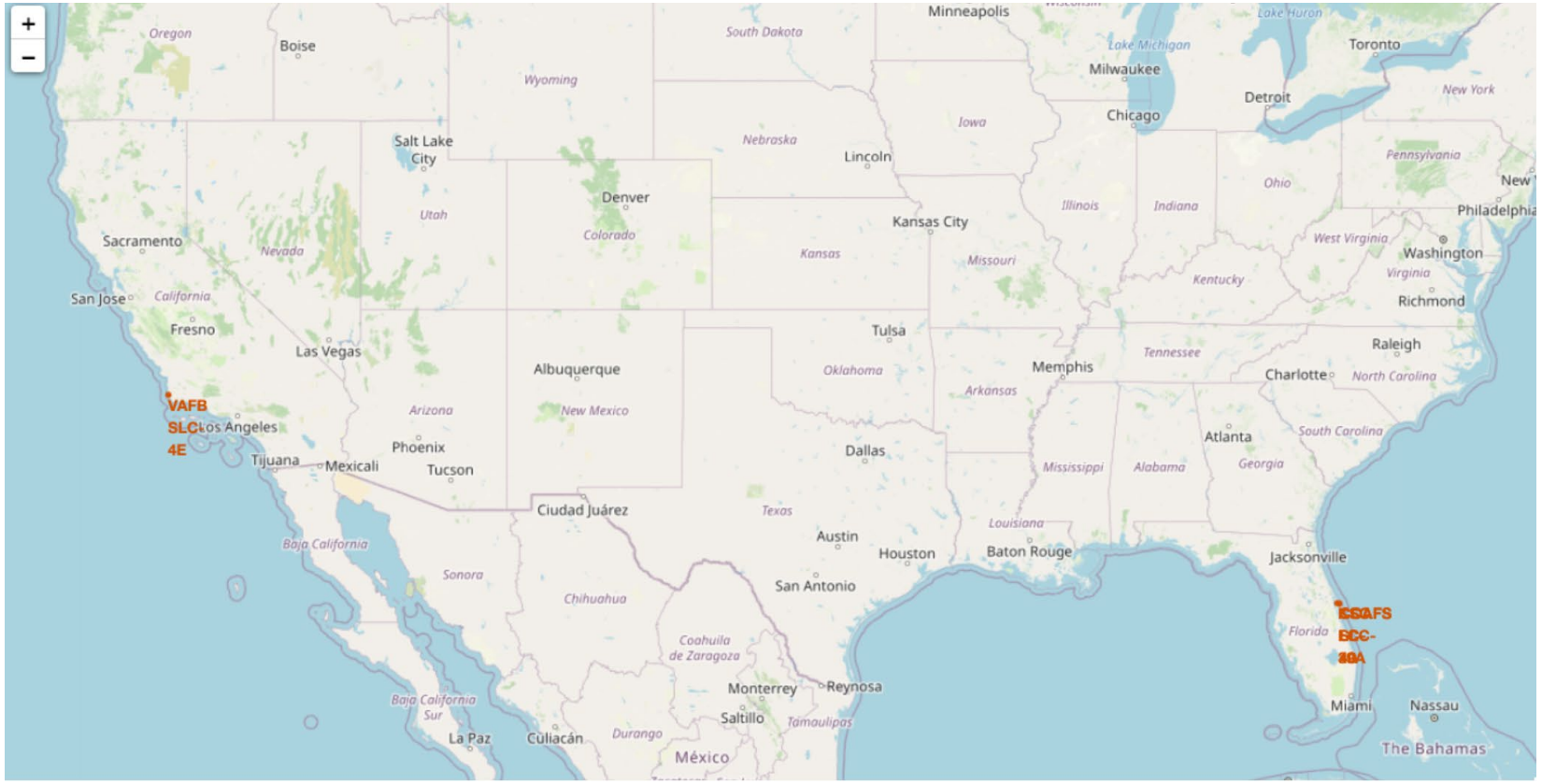
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 4

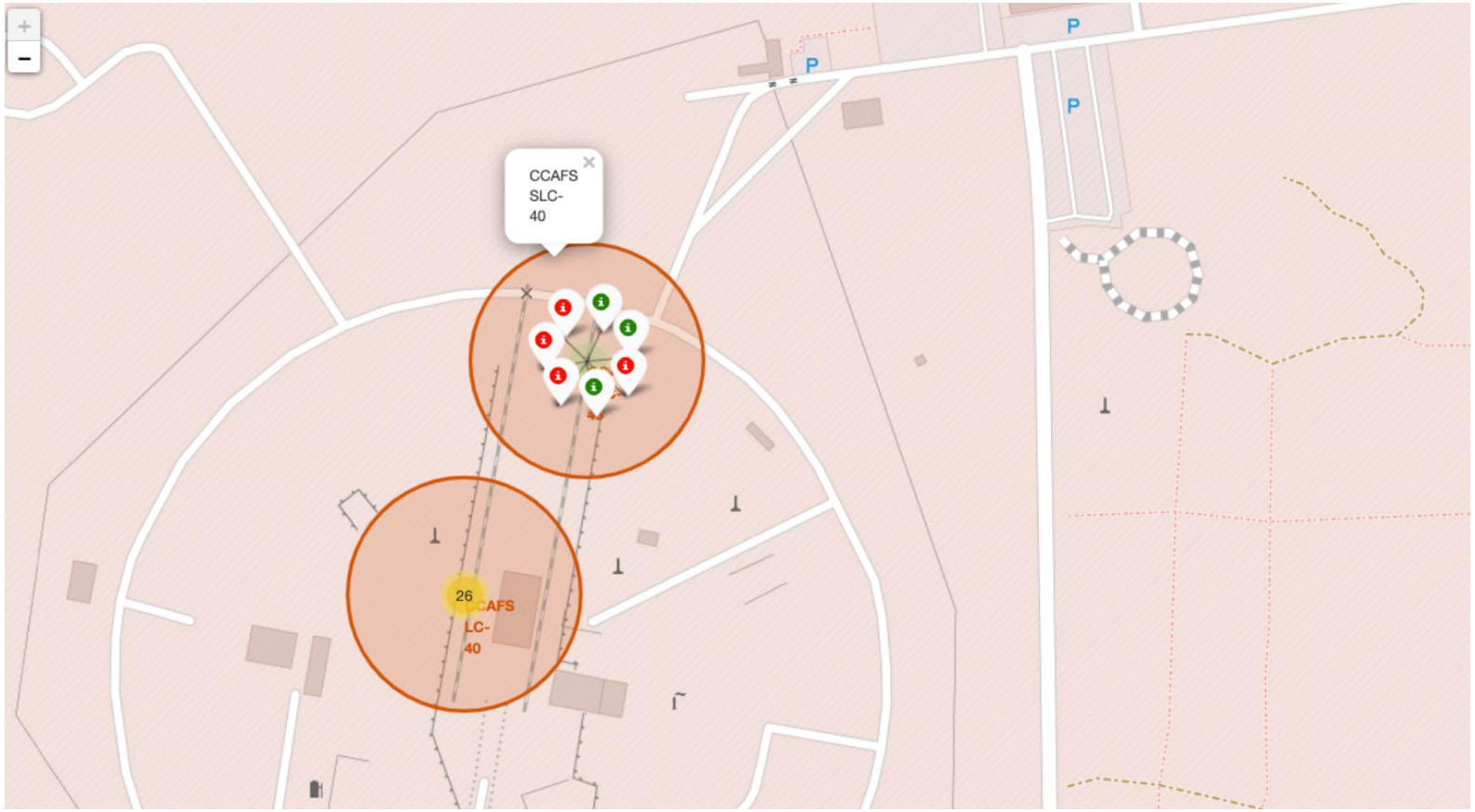
Launch Sites Proximities Analysis

All launch sites global map markers

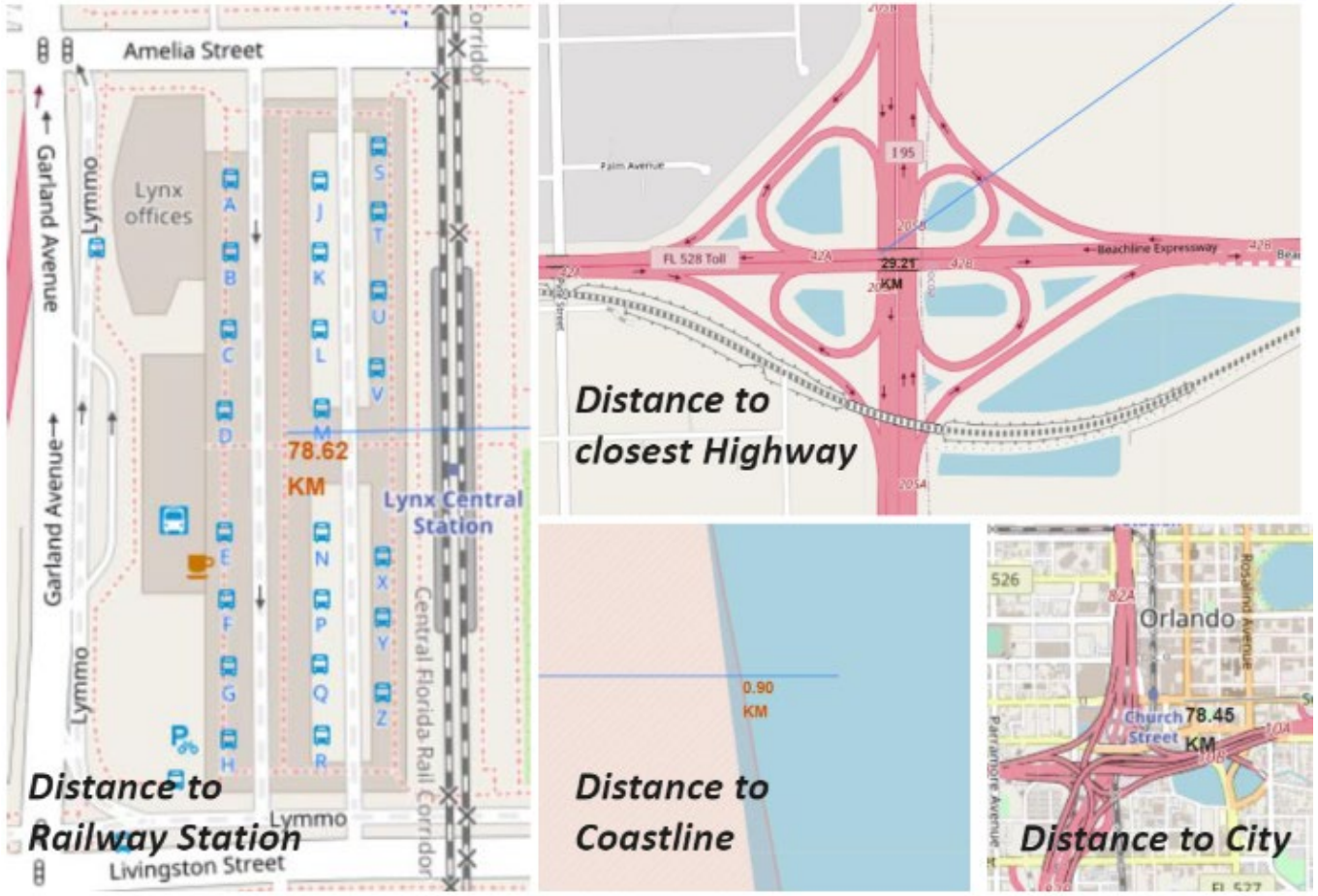
The SpaceX launch sites are located on the east and west coasts of the United States, specifically in the states of California and Florida.

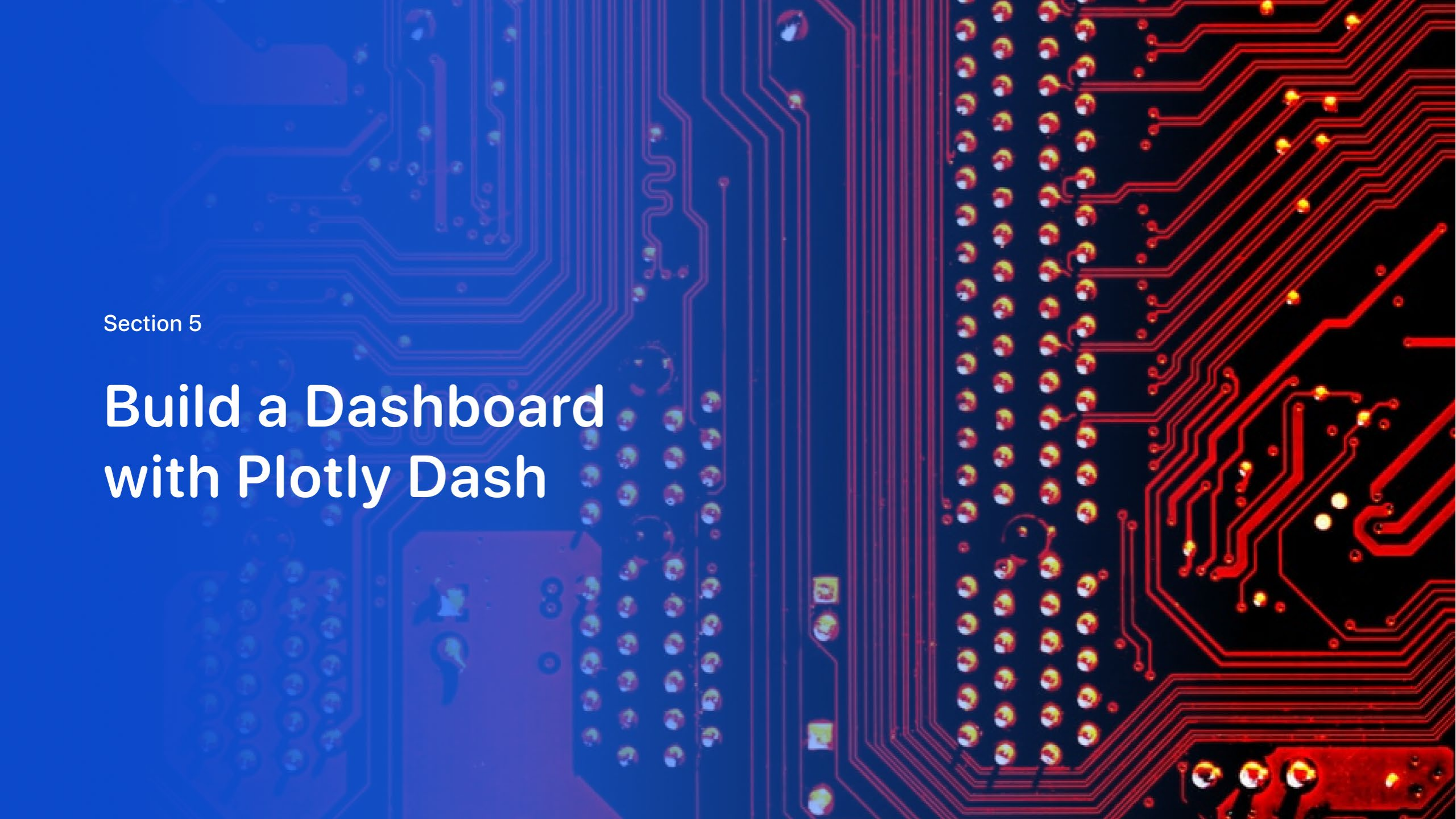


Markers showing launch sites with color labels



Launch Site distance to landmarks



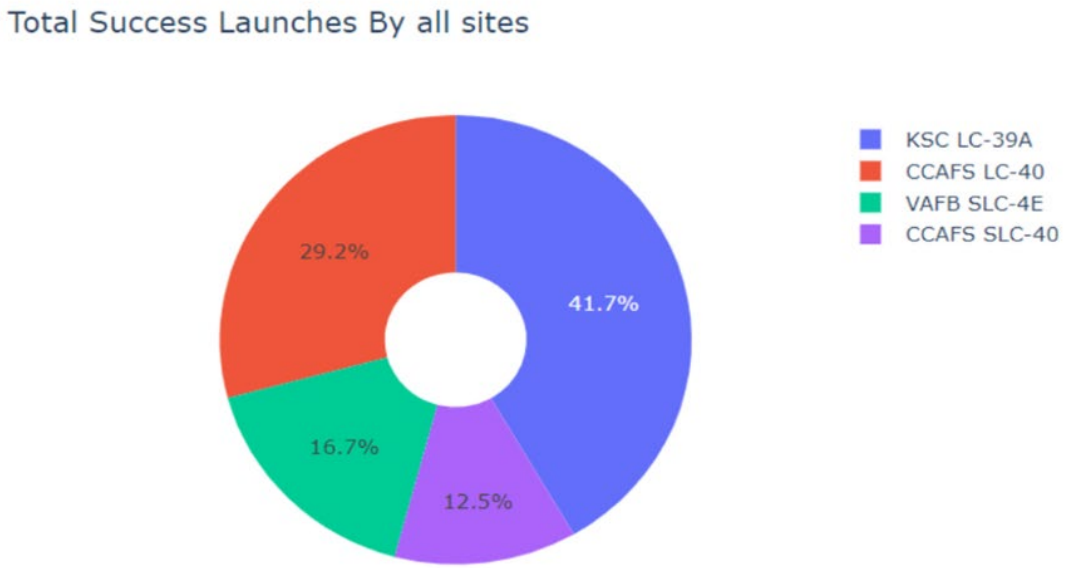


Section 5

Build a Dashboard with Plotly Dash

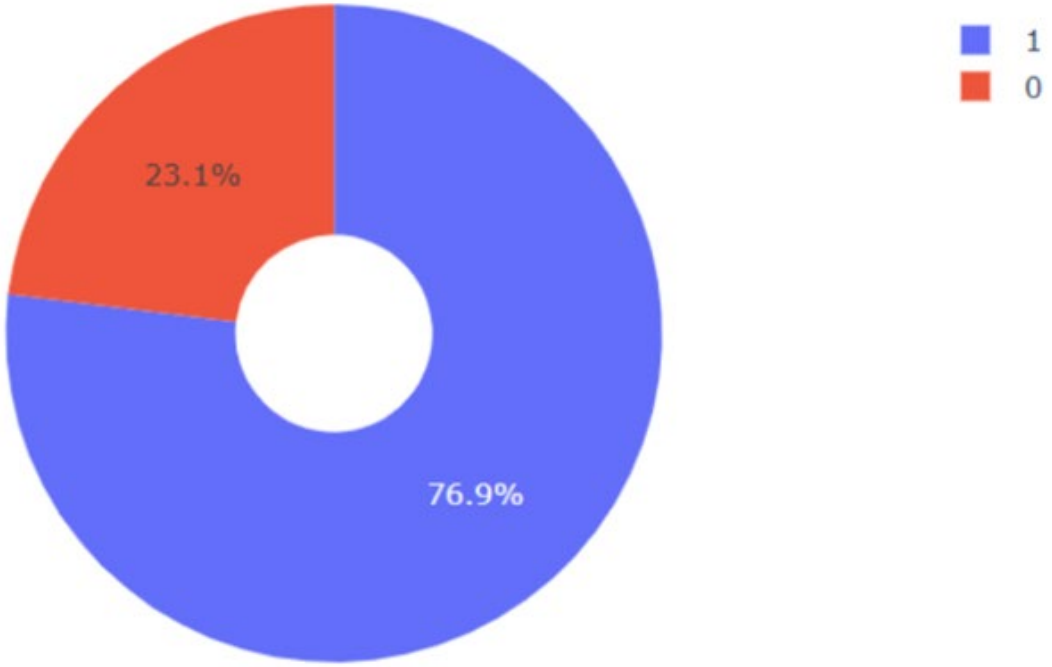
Pie chart showing the success percentage achieved by each launch site

We employed the Plotly Dash to verify that KRC LC-39A had the highest success rate among all launch sites.

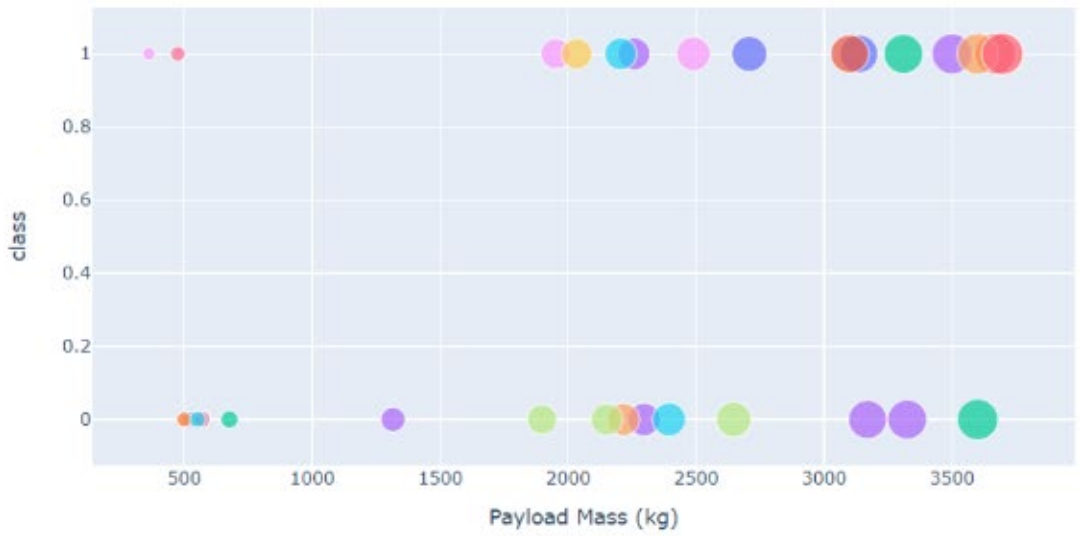


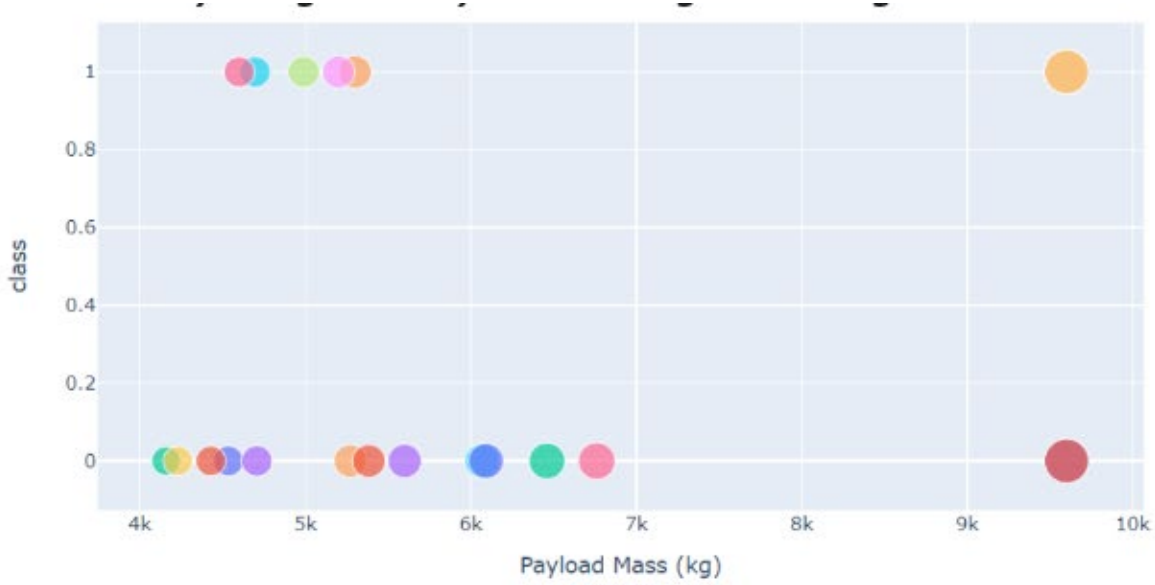
Pie chart showing the Launch site with the highest launch success ratio

At KSC LC-39A, a success rate of 76.9% was observed, while a failure rate of 23.1% was also recorded.



Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider



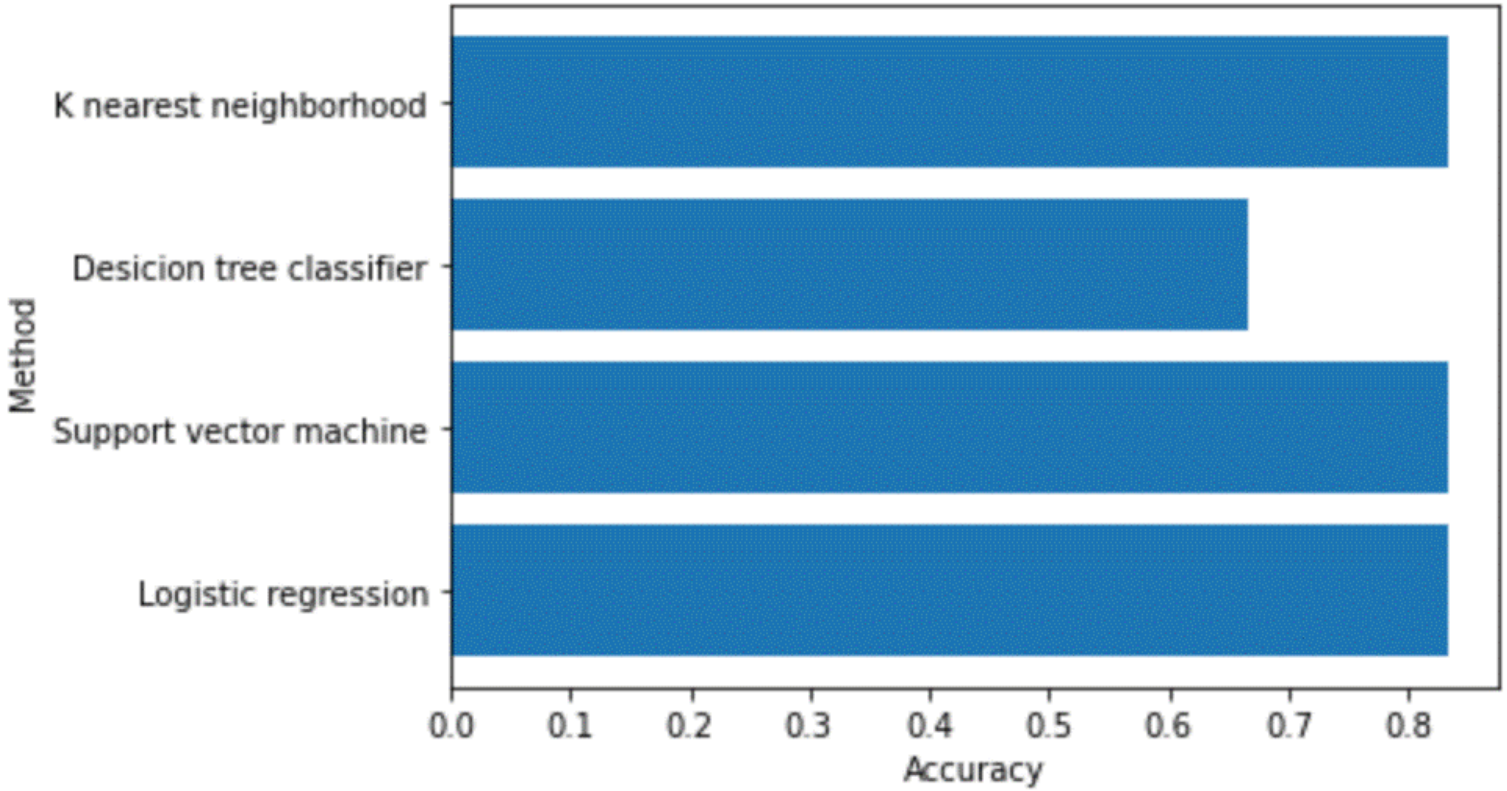


Section 6

Predictive Analysis (Classification)

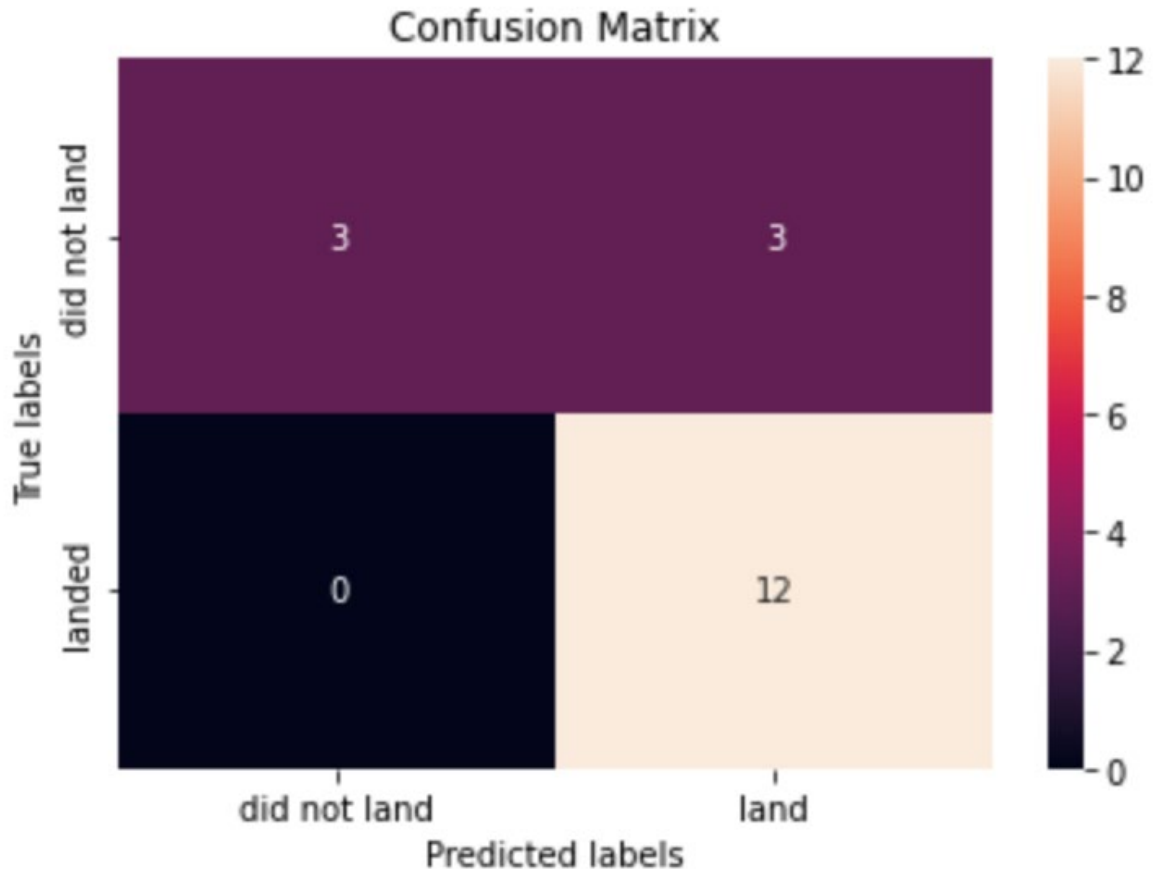
Classification Accuracy

The K-Nearest Neighbors (KNN) algorithm was found to be the model with the highest classification accuracy.



Confusion Matrix

The K-nearest Neighbors classifier demonstrates a high level of classification accuracy. However, the confusion matrix reveals that there is an issue with false positives, where the classifier incorrectly identifies an unsuccessful landing as a successful one.



Conclusions

- Based on the data analysis conducted, it can be inferred that:

- There is a positive correlation between the flight amount at a launch site and the success rate at the same site.
- The success rate of launches has been on the rise since 2013 and continues to increase till 2020.
- The ES-L1, GEO, HEO, SSO, and VLEO orbits have demonstrated the highest success rates among all orbits.
- The KSC LC-39A launch site has recorded the most successful launches among all launch sites.
- The K-Nearest Neighbors classifier was found to be the most effective machine-learning algorithm for the task of predicting launch outcomes based on the data provided.

Thank you!