IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Yung Hsuan Chung
17 July 2025

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 from Machine Learning Lab

Introduction

SpaceX has revolutionized the aerospace industry by dramatically reducing launch costs. While traditional providers charge upwards of $165 million per launch, SpaceX offers Falcon 9 missions for as low as $62 million. This cost advantage is largely attributed to their innovative approach of reusing the first-stage booster by successfully landing it for future missions. This reusability model has the potential to reduce costs even further over time.

As a data scientist working for a startup competing in the commercial space sector, the objective of this project is to develop a machine learning pipeline capable of predicting the landing outcome of the Falcon 9 first stage. This prediction is essential for estimating competitive launch pricing when bidding against SpaceX.

The key challenges addressed in this project include:

- Identifying the factors that most influence landing success.
- Understanding the relationships between those variables and the outcome.
- Determining the optimal conditions that maximize the probability of a successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding for categorical features
- 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

Data collection involves acquiring and preparing information on relevant variables to enable analysis and informed decision-making. In this project, data was obtained through two primary methods: REST API requests and web scraping from Wikipedia.

For the REST API, the process began with issuing a GET request to the SpaceX API. The JSON response was parsed and transformed into a structured pandas DataFrame using json_normalize(). After loading the data, we performed data cleaning by standardizing column names, identifying and handling missing values, and filtering for Falcon 9 launches.

For the web scraping component, BeautifulSoup was used to extract HTML tables from the Wikipedia page listing Falcon 9 and Falcon Heavy launches. The relevant launch data table was parsed and converted into a pandas DataFrame for further preprocessing and analysis.

Data Collection – SpaceX API

1. Get request for rocket launch data using API.

2. Use json_normalize method to convert json result to dataframe.

3. Performed data cleaning and filling the missing value

From: https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Data_Collection.ipynb

Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
[6]
spacex_url="https://api.spacexdata.com/v4/launches/past"
✓ 0.0s
```

```
[7]
response = requests.get(spacex_url)
✓ 0.9s
```

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

```
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

data['date'] = pd.to_datetime(data['date_utc']).dt.date

data = data[data['date'] <= datetime.date(2025, 07, 17)]
```

Data Collection - Scraping

- Request the Falcon9 Launch Wiki page from url
- Create a BeautifulSoup from the HTML response
- Extract all column/variable names from the HTML header

```python

data = requests.get(static_url).text
```

0.4s

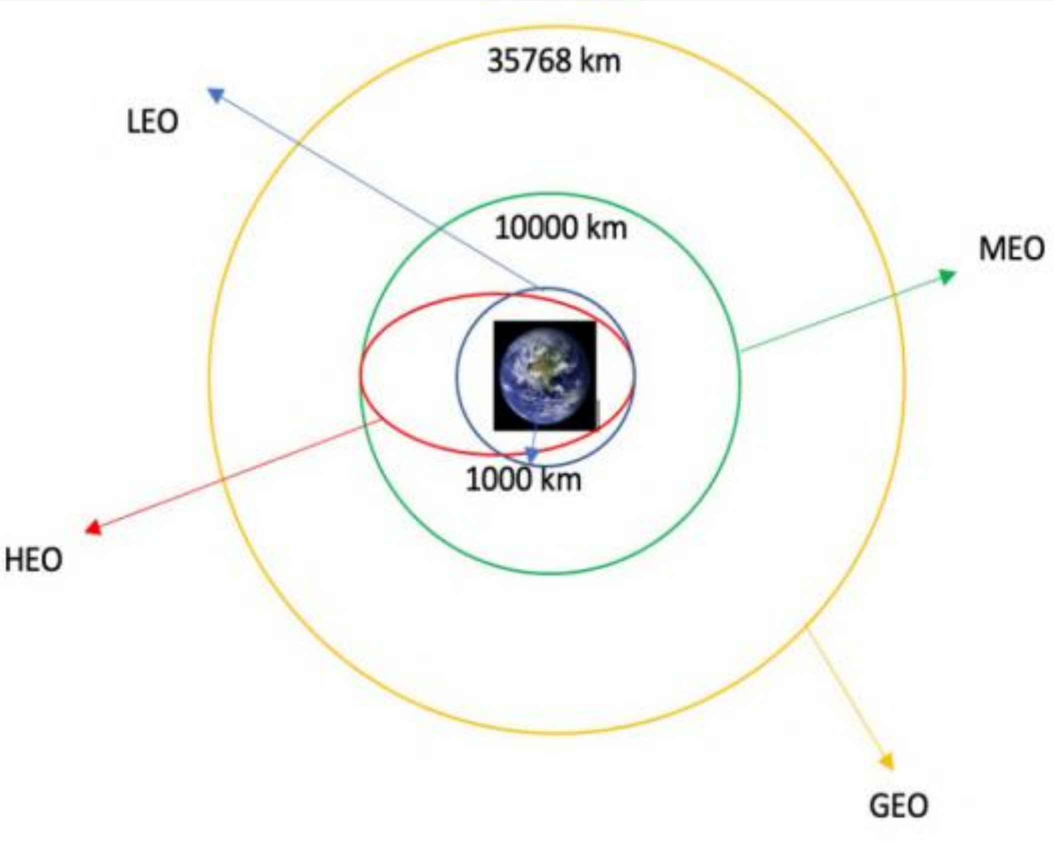
```python
soup = BeautifulSoup(data,'html.parser')
```

```python
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:
            extracted_row += 1
```

From: https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Data_Collection_with_Web_Scraping.ipynb

Data Wrangling

- Data Wrangling is the process of cleaning and unifying messy and complex data sets for easy access and Exploratory Data Analysis (EDA).
- We will first calculate the number of launches on each site, then calculate the number and occurrence of mission outcome per orbit type.
- We then create a landing outcome label from the outcome column. This will make it easier for further analysis, visualization, and ML. Lastly, we will export the result to a CSV.



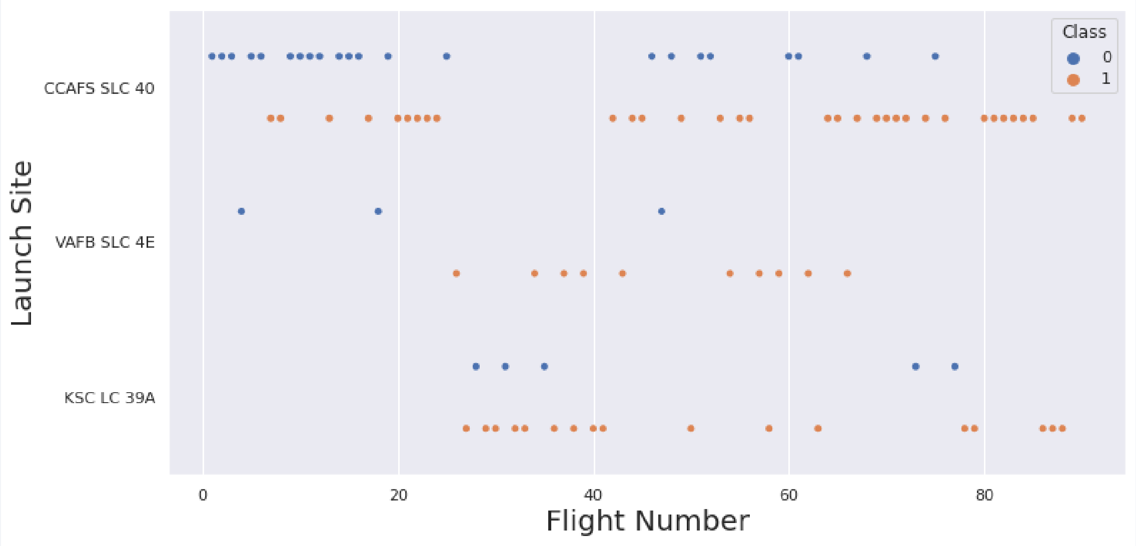
https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Data_Wrangling.ipynb

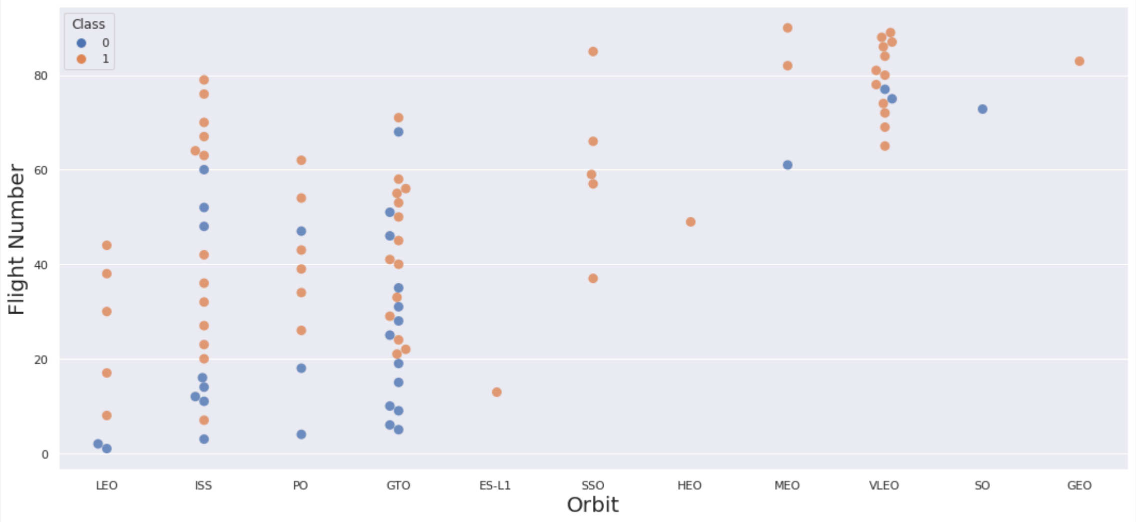
EDA with Data Visualization

- We first started by using scatter graph to find the relationship between the attributes such as between:
    - Payload and Flight Number.
    - Flight Number and Launch Site.
    - Payload and Launch Site.
    - Flight Number and Orbit Type.
    - Payload and Orbit Type.

Scatter plots show dependency of attributes on each other. Once a pattern is determined from the graphs. It's very easy to see which factors affecting the most to the success of the landing outcomes.

https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Exploratory_Data_Analysis_with_Visualisation_Lab_.ipynb

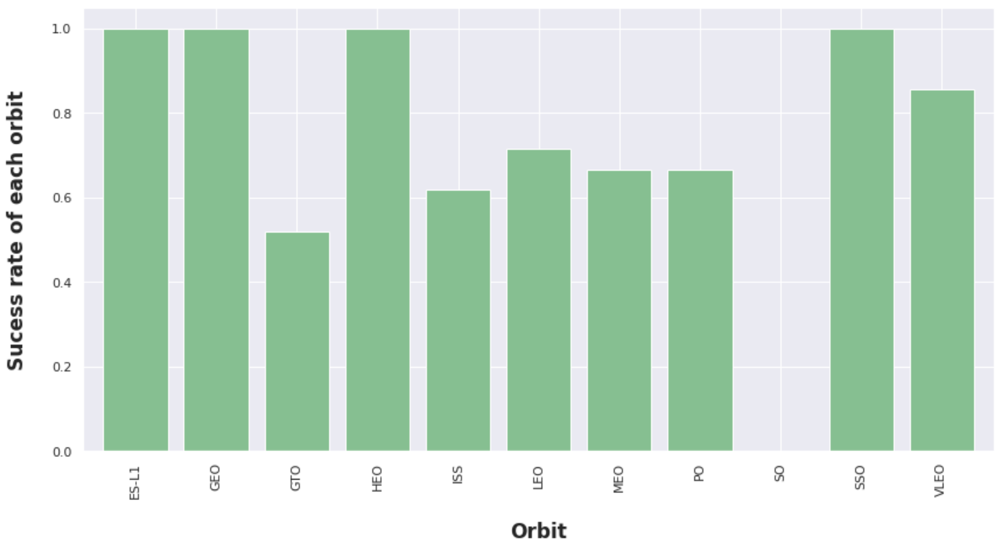


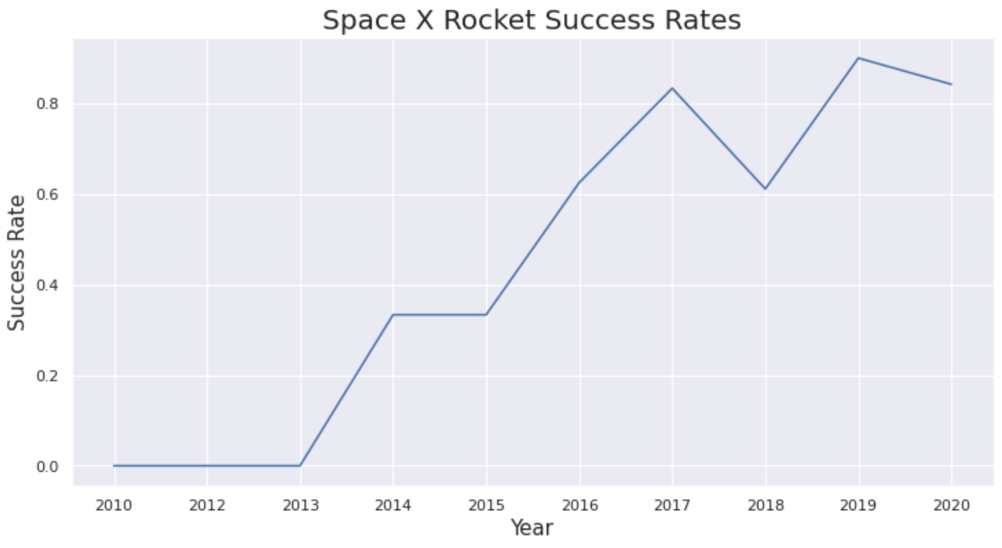


EDA with Data Visualization

- Once we get a hint of the relationships using scatter plot. We will then use further visualization tools such as bar graph and line plots graph for further analysis.
- Bar graphs is one of the easiest way to interpret the relationship between the attributes. In this case, we will use the bar graph to determine which orbits have the highest probability of success.
- We then use the line graph to show a trends or pattern of the attribute over time which in this case, is used for see the launch success yearly trend.
- We then use Feature Engineering to be used in success prediction in the future module by created the dummy variables to categorical columns.

https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Exploratory_Data_Analysis_with_Visualisation_Lab_.ipynb





EDA with SQL

- Using SQL, we had performed many queries to get better understanding of the dataset, Ex:
  - Displaying the names of the launch sites.
  - Displaying 5 records where launch sites begin with the string 'CCA'.
  - Displaying the total payload mass carried by booster launched by NASA (CRS).
  - Displaying the average payload mass carried by booster version F9 v1.1.
  - Listing the date when the first successful landing outcome in ground pad was achieved.
  - Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - Listing the total number of successful and failure mission outcomes.
  - Listing the names of the booster_versions which have carried the maximum payload mass.
  - Listing the failed landing_outcomes in drone ship, their booster versions, and launch sites names for in year 2015.
  - Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order.

https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Exploratory_Data_Analysis_with_SQL.ipynb

Build an Interactive Map with Folium

- To visualize the spatial distribution of SpaceX launch sites, we utilized Folium to create an interactive map. Latitude and longitude coordinates for each launch site were used to place circle markers, each labeled with the corresponding launch site name.
- We then classified launch outcomes into binary classes: 0 = Failure and 1 = Success, and represented them using MarkerCluster with color coding—Red for failed landings and Green for successful ones.
- To perform geospatial analysis, we applied the Haversine formula to compute the distances between each launch site and key landmarks, allowing us to answer questions such as:
- How close are launch sites to railways, highways, and coastlines?
- What is the proximity of launch sites to major cities?
- This geospatial insight helps evaluate the logistical and environmental factors influencing launch operations.

https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Interactive_Visual_Analytics_with_Folium.ipynb

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash which allowing the user to play around with the data as they need.
- We plotted pie charts showing the total launches by a certain sites.
- We then plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version

https://github.com/yungzyx/Applied-Data-Science-Capstone-SpaceX/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

1. Building the Model

We began by loading the cleaned and wrangled dataset into NumPy and Pandas, followed by necessary preprocessing and feature transformation. The data was then split into training and test sets. Based on exploratory insights, we selected suitable classification algorithms and applied GridSearchCV to perform hyperparameter tuning for optimal model configuration.

2. Evaluating the Model

Each model was evaluated using key metrics such as accuracy, confusion matrix, and cross-validation scores. This helped us identify the most effective model and hyperparameter settings for the classification task.

3. Improving the Model

To further enhance model performance, we applied feature engineering techniques (e.g., encoding, scaling, dimensionality reduction) and refined hyperparameters based on iterative validation. This continuous feedback loop helped optimize generalization and robustness.

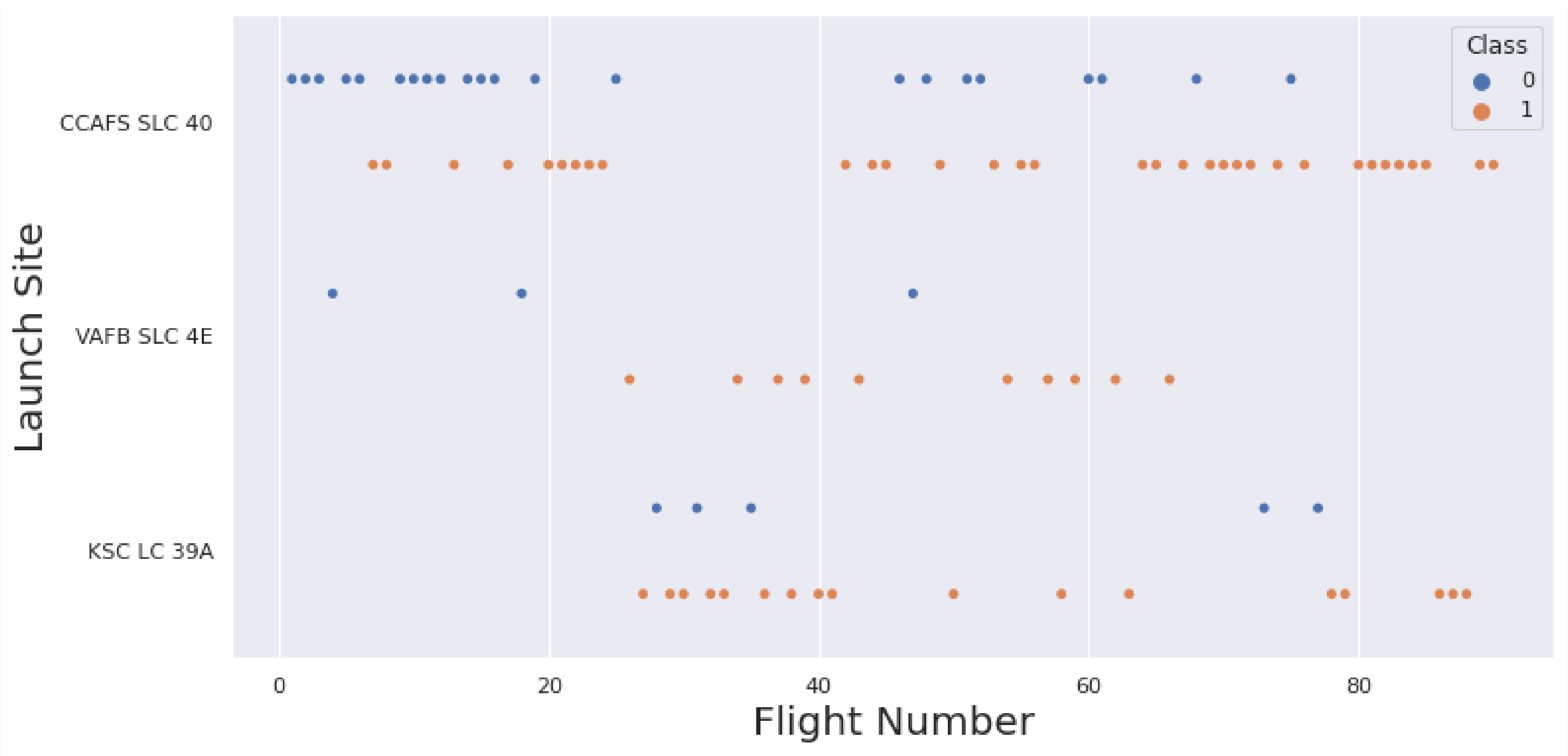
Results

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

Section 2

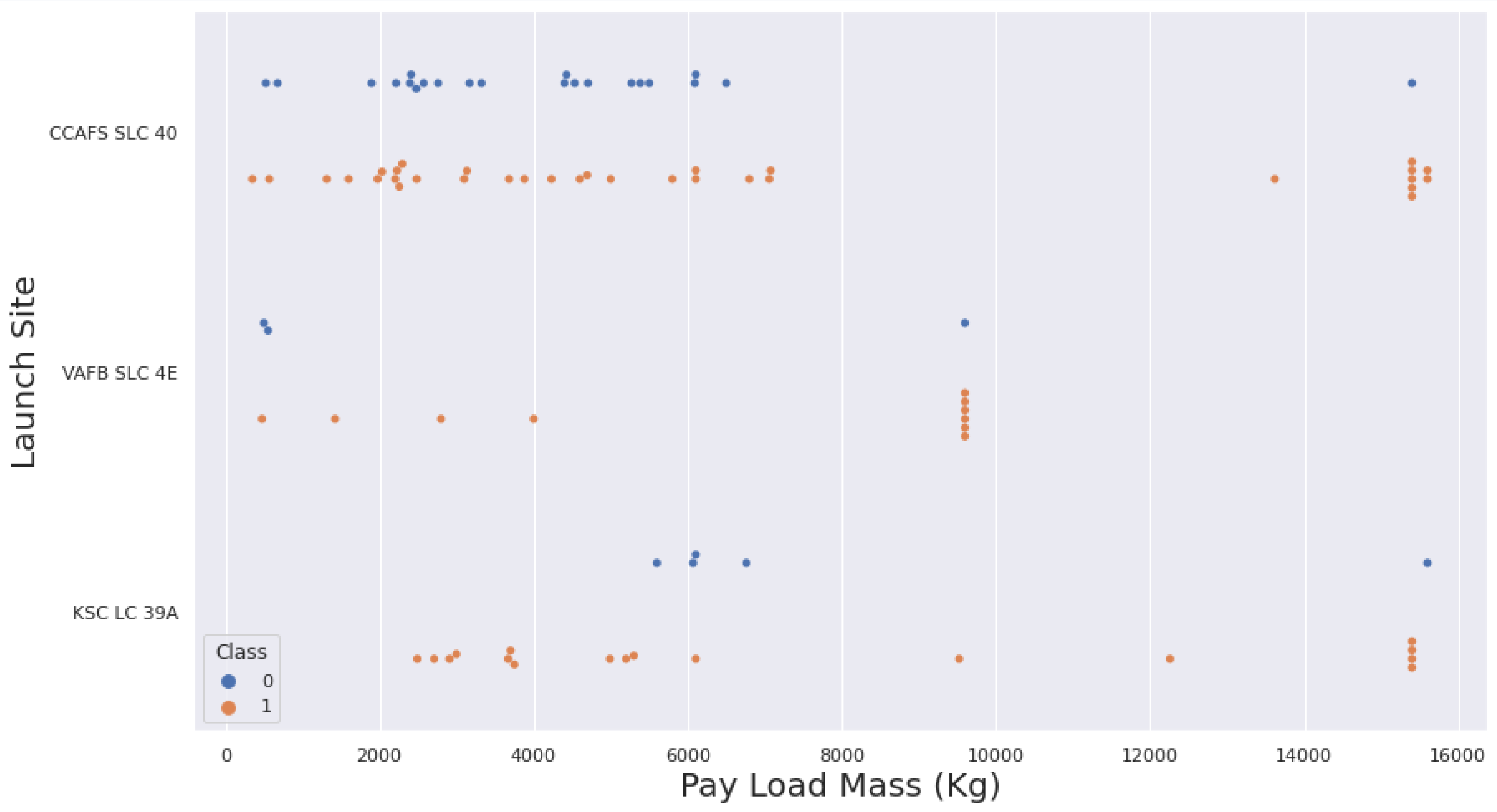
Insights drawn from EDA

Flight Number vs. Launch Site



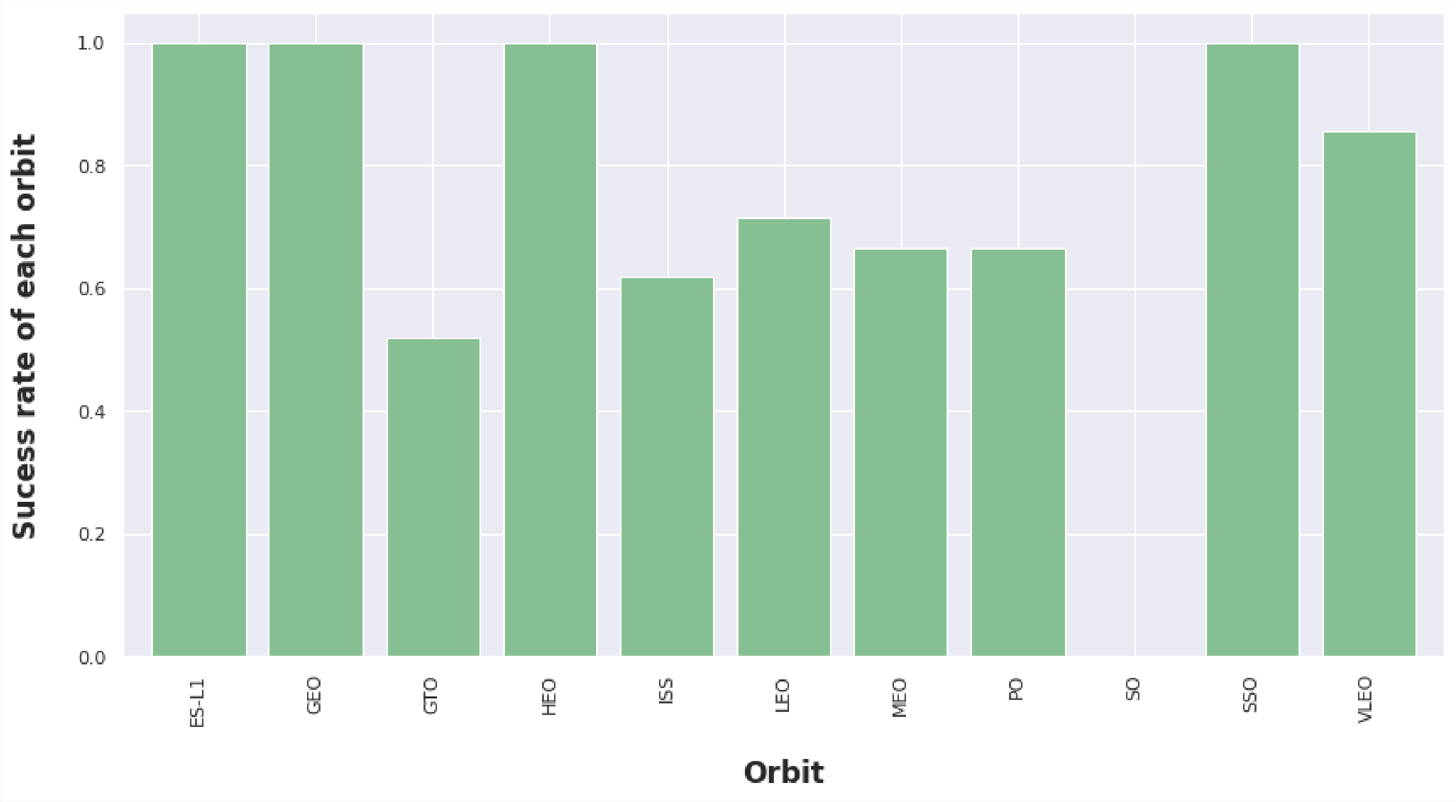
Payload vs. Launch Site

This scatter plot indicates that when the payload mass exceeds 7000 kg, the probability of a successful landing increases significantly. However, there is no clear pattern suggesting that the launch site has a direct dependency on payload mass in determining the success rate. Therefore, while heavier payloads tend to correlate with better outcomes, the launch site's influence remains inconclusive based on this data.



Success Rate vs. Orbit Type

- This figure illustrates the potential influence of orbital types on landing outcomes. Certain orbits, such as SSO, HEO, GEO, and ES-L1, show a 100% success rate, whereas the SO orbit resulted in a 0% success rate.
- However, a closer inspection reveals that some of these orbits, including GEO, SO, HEO, and ES-L1, had only a single recorded launch. Therefore, the current data is insufficient to establish any reliable patterns or trends. Additional data is necessary before making any definitive conclusions regarding the impact of orbit type on landing success.

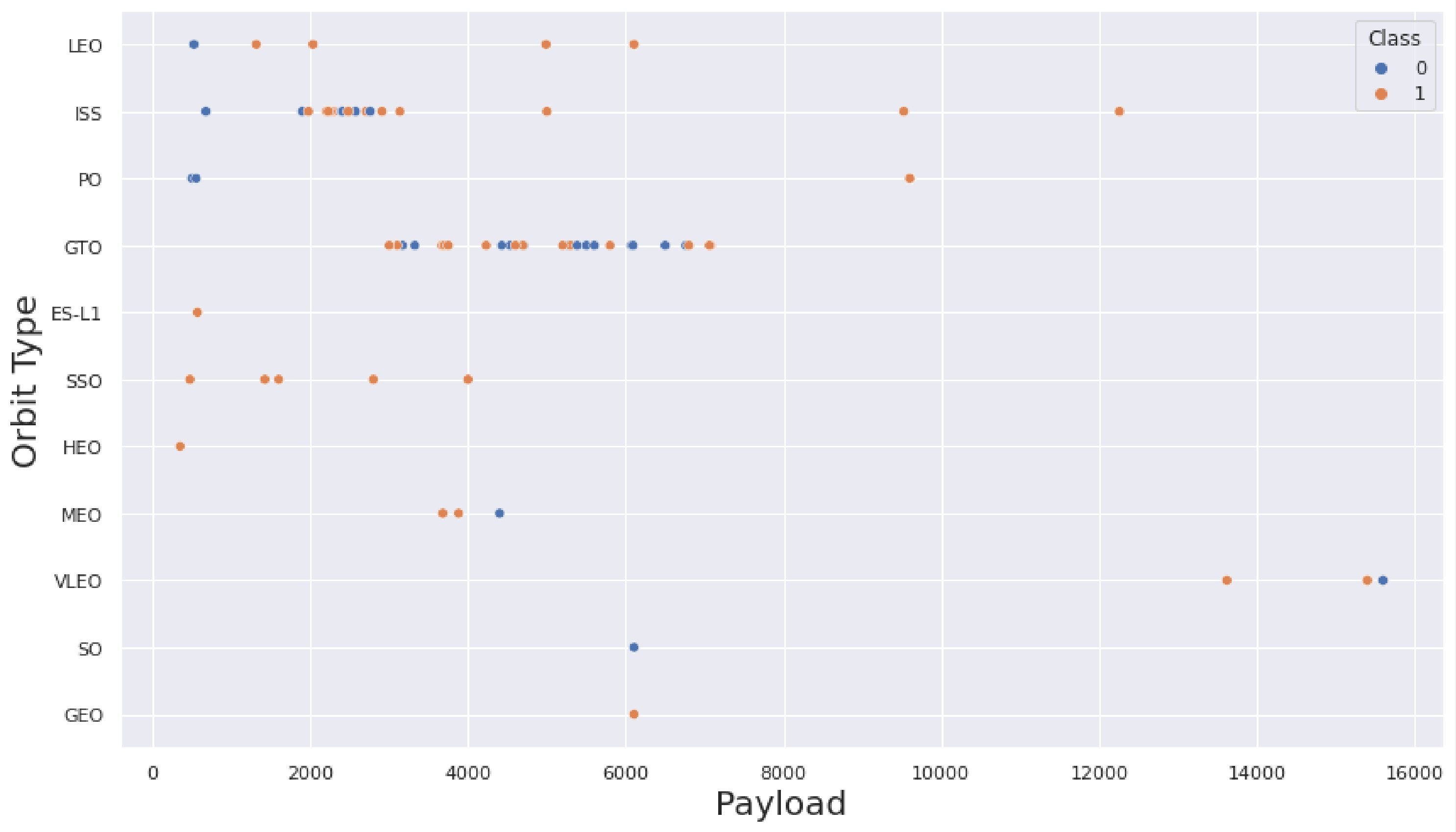


Flight Number vs. Orbit Type

- To visualize the spatial distribution of SpaceX launch sites, we utilized Folium to create an interactive map. Latitude and longitude coordinates for each launch site were used to place circle markers, each labeled with the corresponding launch site name.
- We then classified launch outcomes into binary classes: 0 = Failure and 1 = Success, and represented them using MarkerCluster with color coding Red for failed landings and Green for successful ones.
- To perform geospatial analysis, we applied the Haversine formula to compute the distances between each launch site and key landmarks, allowing us to answer questions such as:
  - How close are launch sites to railways, highways, and coastlines?
  - What is the proximity of launch sites to major cities?
  - This geospatial insight helps evaluate the logistical and environmental factors influencing launch operations.

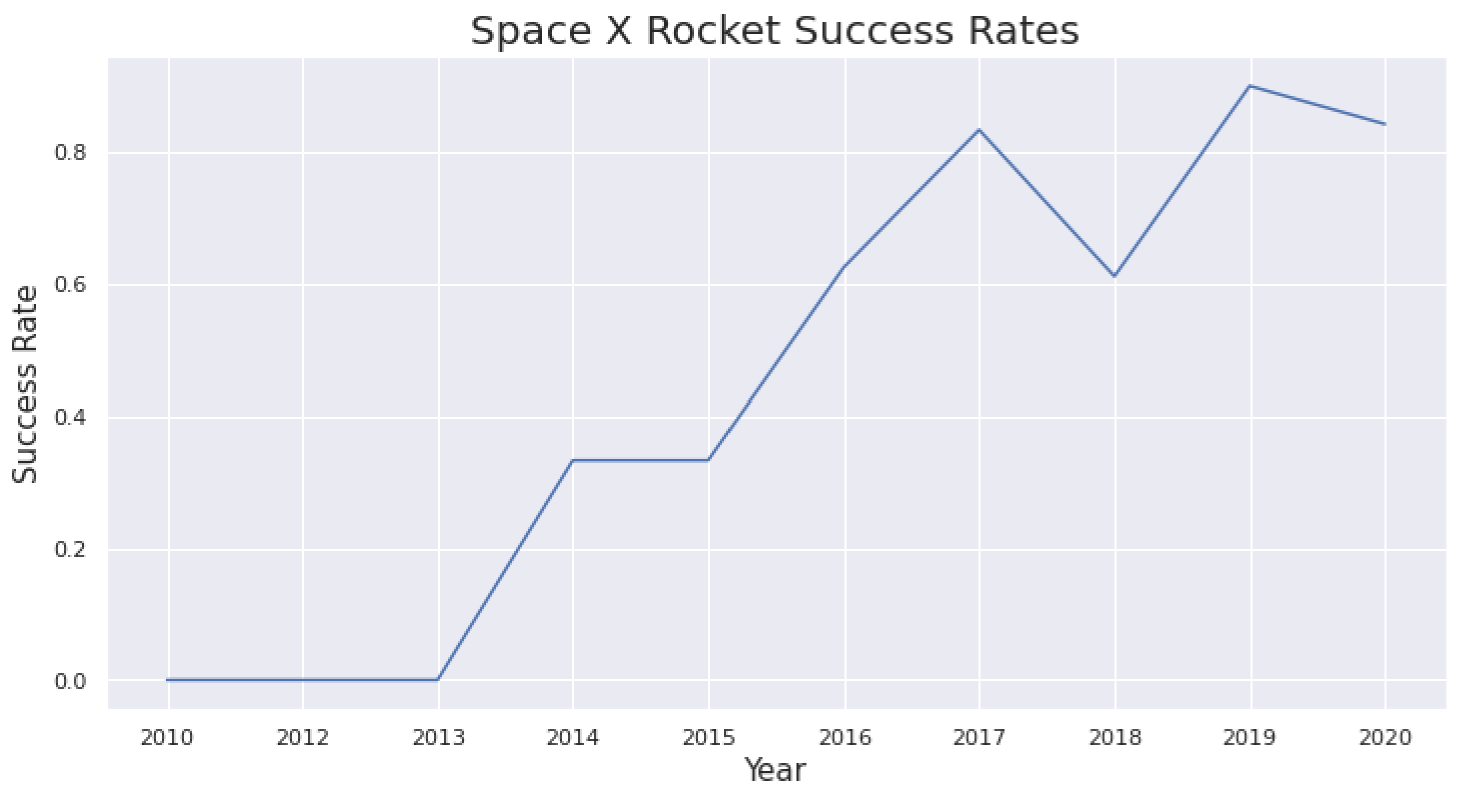
Payload vs. Orbit Type

- Heavier payloads appear to have a positive impact on landing success for LEO, ISS, and PO orbits. In contrast, they tend to have a negative impact on MEO and VLEO orbits.
- For GTO orbits, there is no clear relationship observed between payload mass and landing outcome.
- As with previous observations, SO, GEO, and HEO orbits have too few data points to draw any meaningful conclusions, and therefore require more data to identify reliable patterns or trends.



Launch Success Yearly Trend

- These figures clearly show an increasing trend in landing success rates from 2013 to 2020. If this upward trend continues in the coming years, it is likely that the success rate will continue to improve, potentially reaching near-perfect (100%) reliability in the near future.



All Launch Site Names

We used the key word DISTINCT to show only unique launch sites from the SpaceX data

```
[5]  1  %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
     ✓ 0.0s
```

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

Launch Site Names Begin with 'CCA'

We used the query above to display 5 records where launch sites begin with `CCA`

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

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

✓ 0.0s

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	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 calculated the total payload carried by boosters from NASA as 45596 using the query below

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Payload Mass by NASA (CRS)
45596

Average Payload Mass by F9 v1.1

We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

First Successful Ground Landing Date

We use the min() function to find the result We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.datab
ases.appdomain.cloud:32731/bludb
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

We used wildcard like '%' to filter for WHERE MissionOutcome was a success or a failure.

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Successful Mission
100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Failure Mission
1

Boosters Carried Maximum Payload

We determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

We used a combinations of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20. We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order

```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEX \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

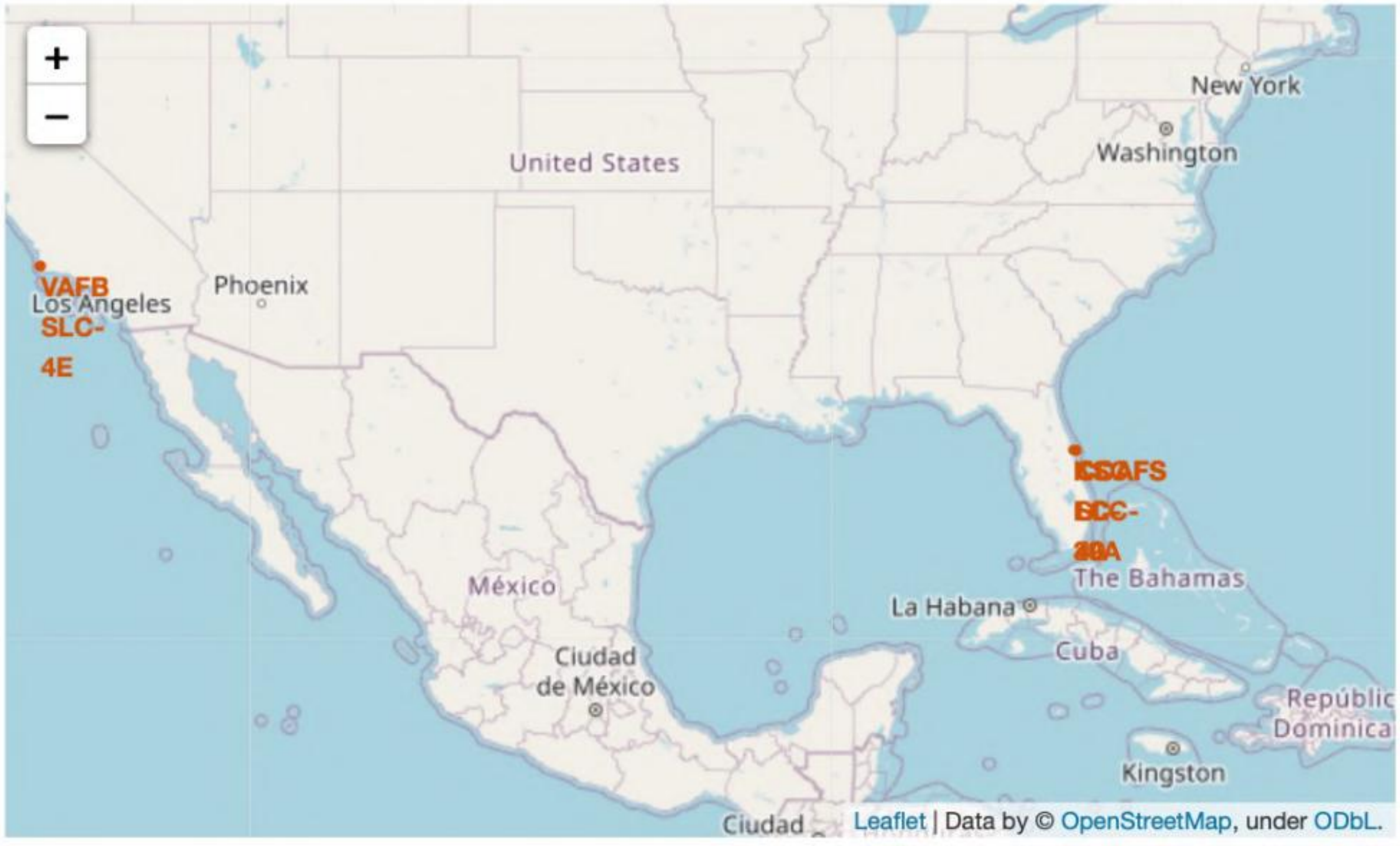
```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.c
loud:32731/bludb
Done.
```

Landing Outcome	Total Count
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

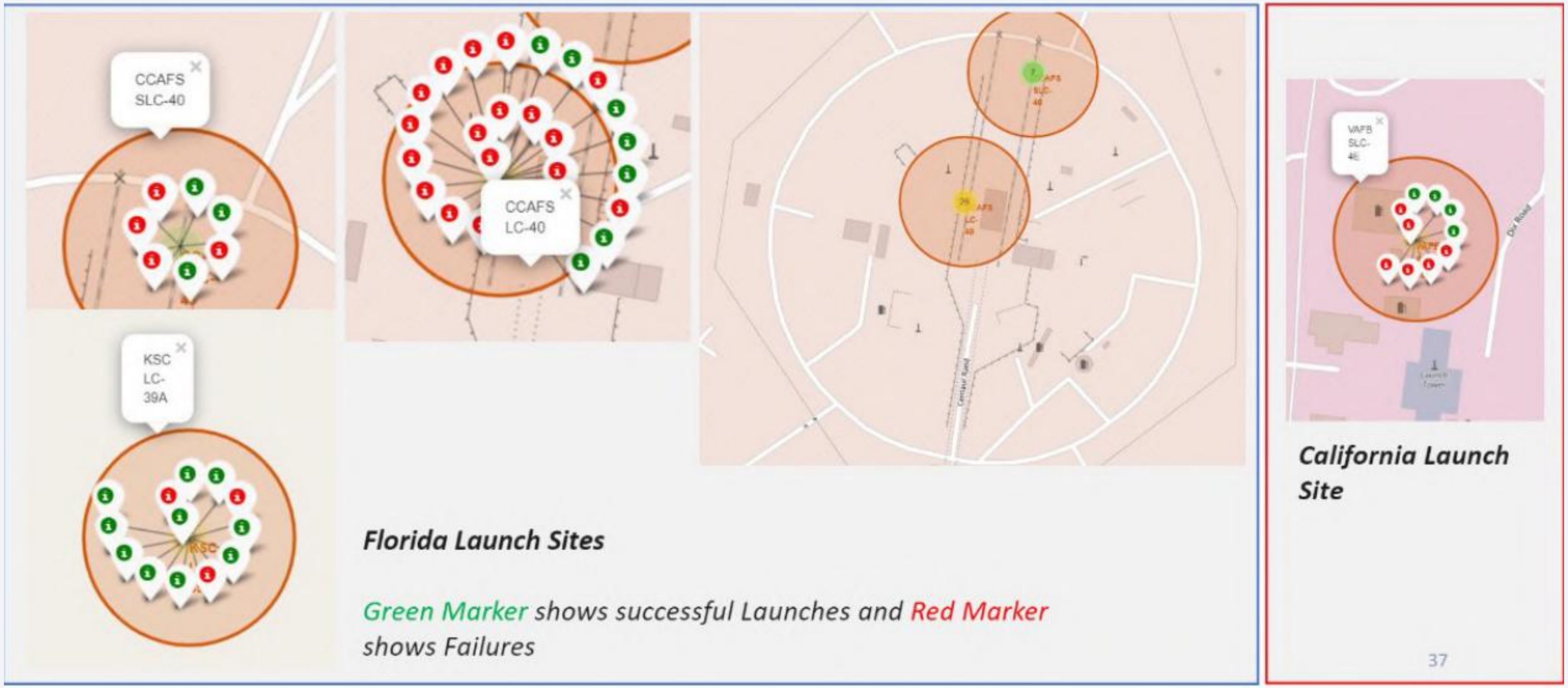
Launch Sites Proximities Analysis

Location of all the Launch Sites



We can see that all the SpaceX launch sites are located inside the United States

Markers showing launch sites with color labels

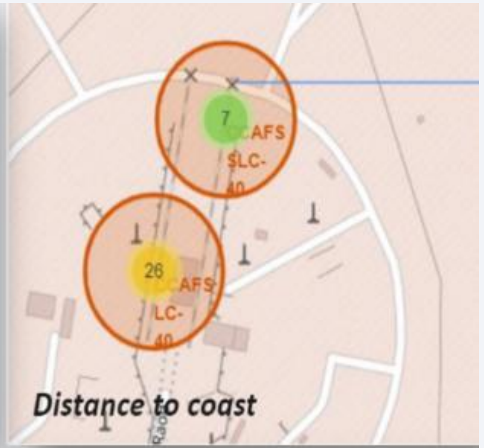


Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

Launch Sites Distance to Landmarks

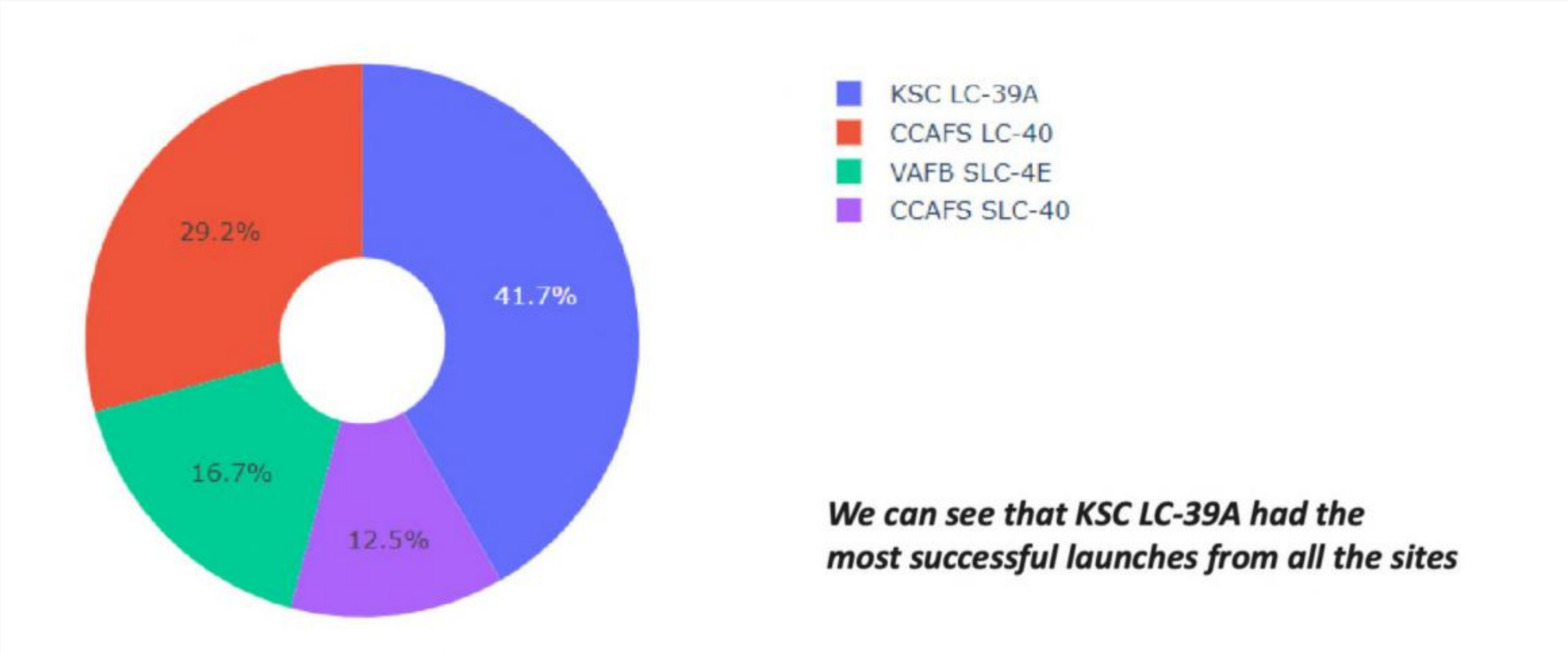


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

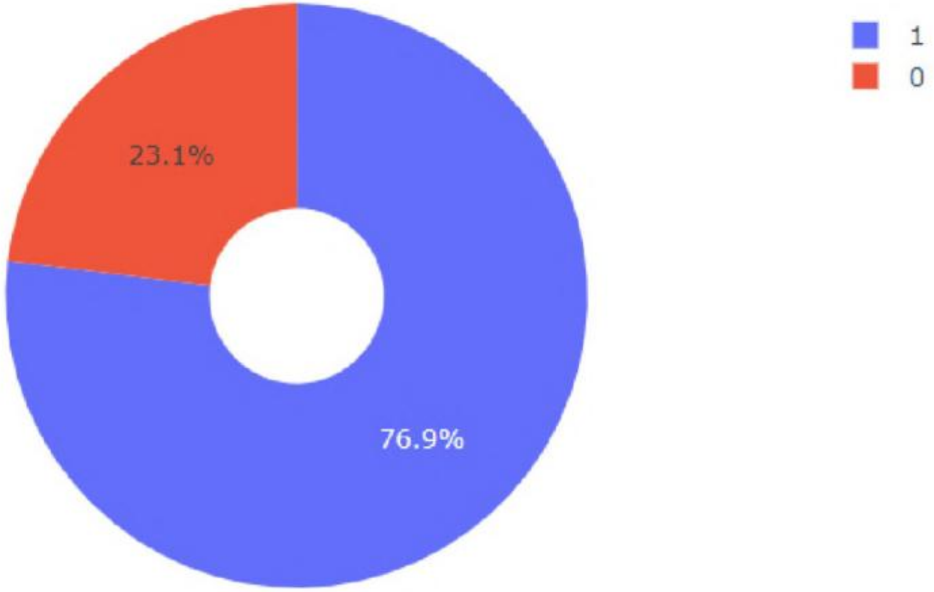
Section 4

Build a Dashboard with Plotly Dash

The success percentage by each sites.



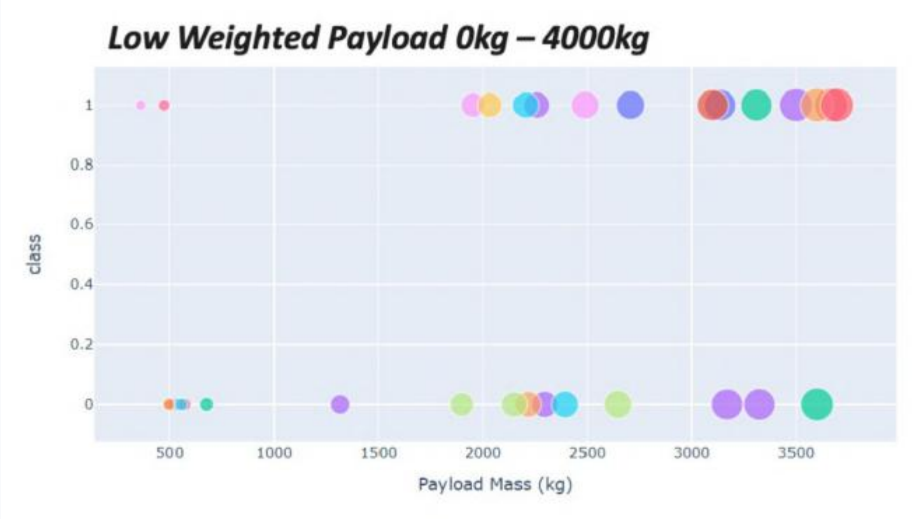
The highest launch-success ratio: KSC LC-39A

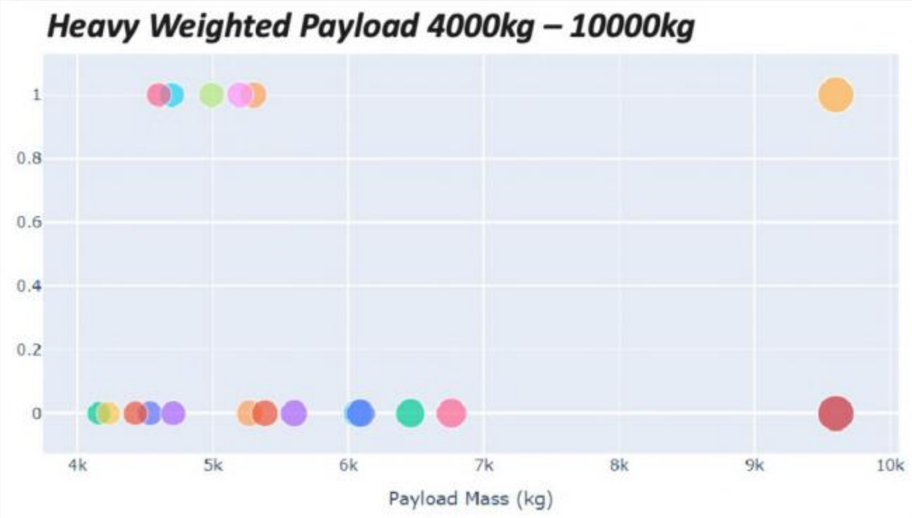


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch Outcome Scatter Plot

We can see that all the success rate for low weighted payload is higher than heavy weighted payload.





Section 5

Predictive Analysis (Classification)

Classification Accuracy

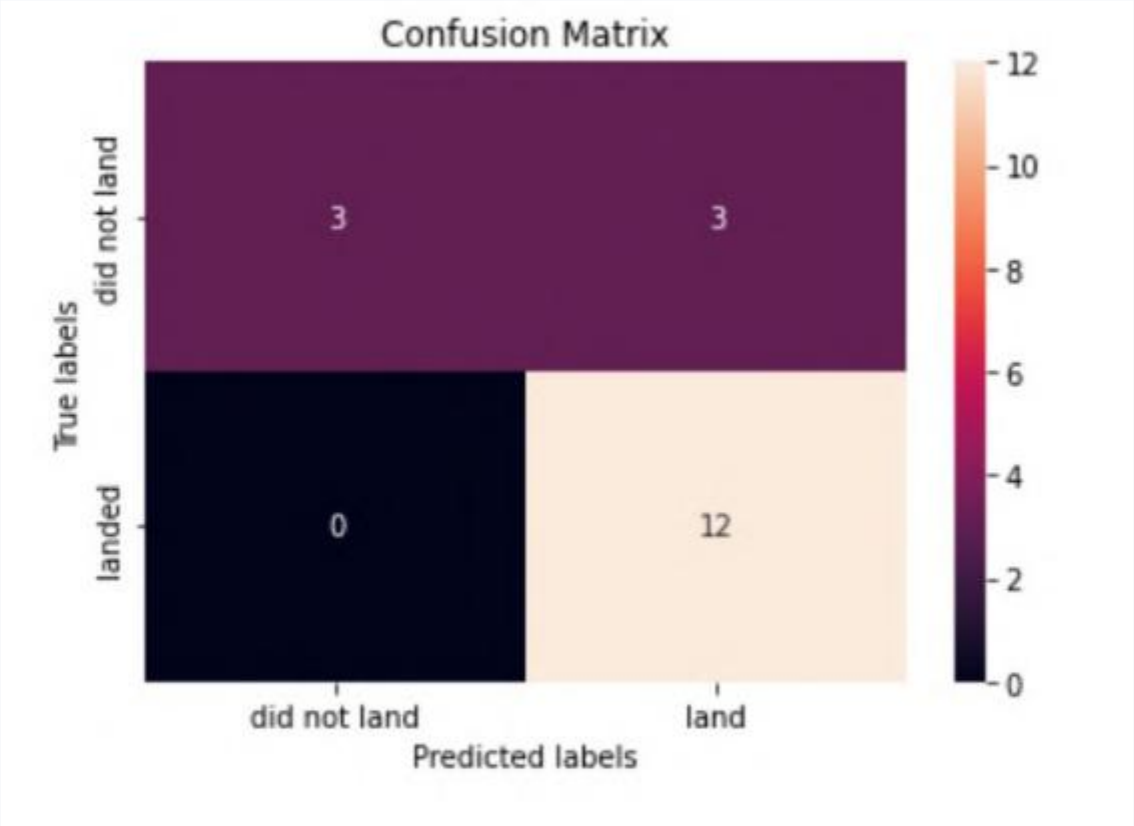
As we can see, by using the code as below: we could identify that the best algorithm to be the Tree Algorithm which have the highest classification accuracy.

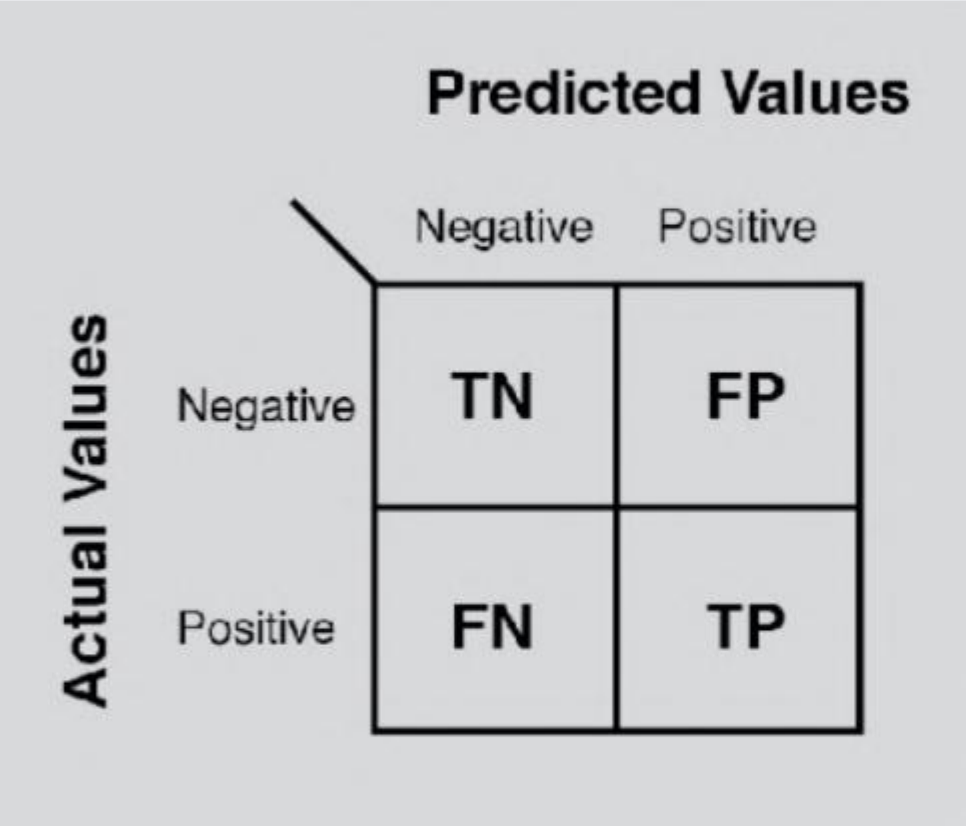
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

Confusion Matrix

The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.





Conclusions

- The Decision Tree Classifier demonstrated the best performance among all machine learning models tested for this dataset.
- Low-weight payloads (≤ 4000 kg) were associated with higher landing success rates compared to heavier payloads.
- Since 2013, SpaceX has shown a consistent increase in launch success rates, indicating a strong positive trend that may lead to near-perfect reliability in the future.
- The KSC LC-39A launch site recorded the highest number of successful launches, with a success rate of 76.9%.
- The SSO (Sun-Synchronous Orbit) had the highest success rate (100%) among orbits with more than one occurrence.

Appendix

This section contains supplementary materials used and generated throughout the project, including:

- Python Code Snippets:
  - GET requests for SpaceX API data collection using requests and json_normalize.
  - Data cleaning and preprocessing scripts with pandas, including handling missing values and transforming data types.
  - Machine Learning model development using scikit-learn, with pipeline setup and hyperparameter tuning via GridSearchCV.
- SQL Queries:
  - Queries for exploratory data analysis, including aggregation by Launch_Site, filtering by BoosterVersion, and grouping by Mission_Outcome.
- Visualizations:
  - matplotlib, seaborn, and plotly charts for EDA and classification model evaluation.
  - Folium interactive maps showing launch site proximity to infrastructure.
  - Confusion matrices for evaluating classification models.
- Dash Application:
  - A custom Plotly Dash dashboard displaying dynamic launch data by site, payload range, and success class.
- Datasets:
  - API-collected launch records from SpaceX and scraped data from Wikipedia.
  - Combined, cleaned, and merged into a structured format for model training and analysis.

Thank you!