Winning Space Race with Data Science

Sabrina KHEZZANE
11/11/2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection & Wrangling
  - Exploratory Data Analysis (EDA)
  - Interactive Visual Analytics
  - Predictive Analysis
- Summary of all results
  - EDA results
  - Interactive Visuals Analytics results
  - Predictive Models results

Introduction

- Project Background:
  - SpaceX, a leader in space exploration, conducts frequent missions that aim to revolutionize space travel and reduce launch costs. The company collects vast amounts of data during these missions, including launch sites, payload mass, mission success rates, and booster recovery. This project aims to leverage this data to extract valuable insights and predictions for future missions.
- Problems to Find Answers:
  - Which launch sites are most successful?: Identifying the best-performing launch sites based on mission outcomes.
  - What factors influence mission success?: Analyzing key features like payload mass, booster type, and launch site on mission success.
  - How can we predict the success of future missions?: Building a predictive model to assess the likelihood of a mission's success based on historical data.
  - What insights can we derive from visual analytics and interactive dashboards?: Creating tools for stakeholders to explore SpaceX mission data.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - DatafromSpaceXwasobtainedfrom2sources:
    - SpaceXAPI(https://api.spacexdata.com/v4/rockets/)
    - WebScraping (https://en.wikipedia.org/wiki/List_of_Falcon/_9/_and_Falcon_Heavy_launches)
- Perform data wrangling
  - Collected data was enriched by creating a landing outcome label based on outcome data after summarizing and analyzing features

Methodology

Executive Summary

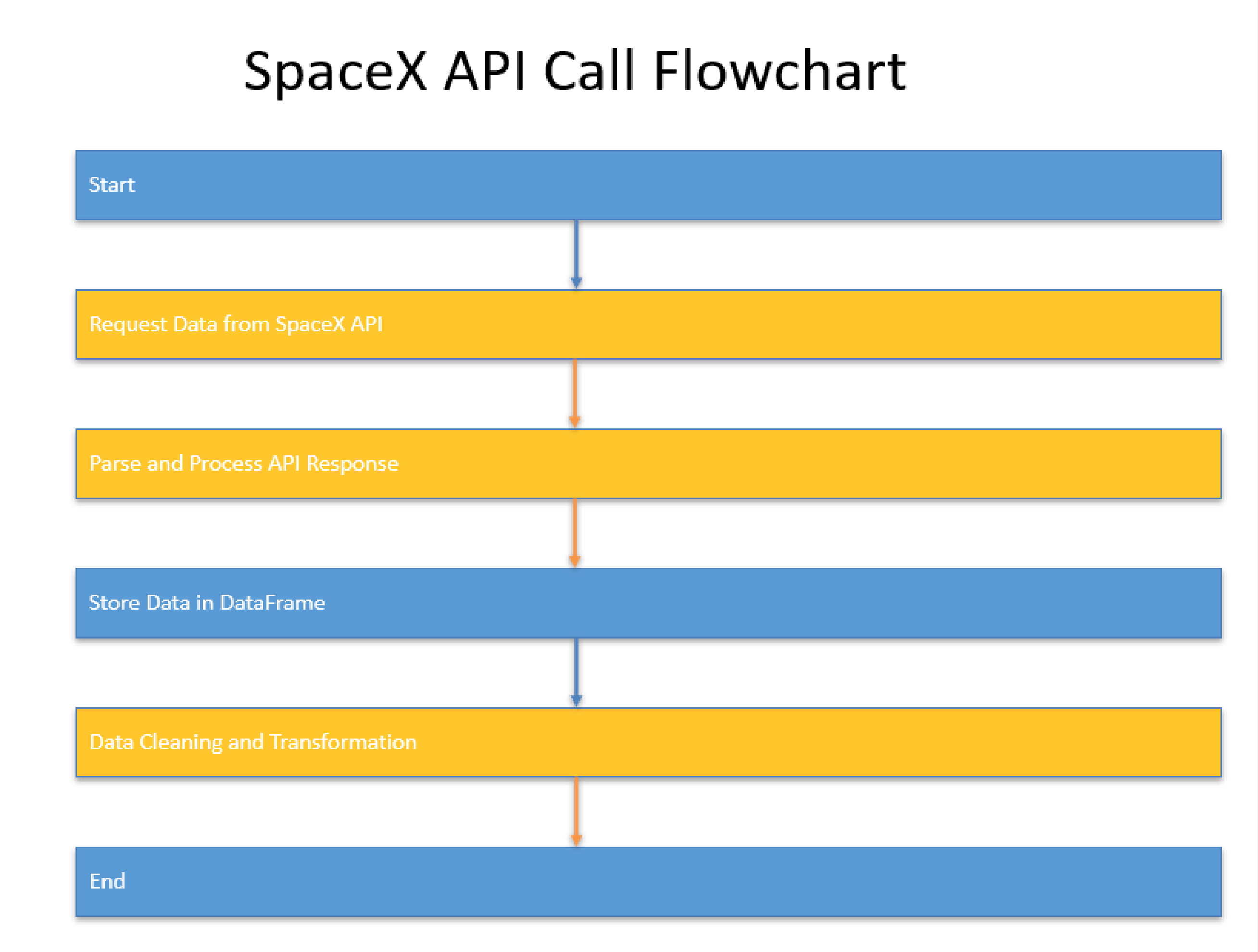
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Data that was collected until this step were normalized, divided in training and test data sets and evaluated by four different classification models, being the accuracy of each model evaluated using different combinations of parameters.

Data Collection

- Describe how data sets were collected.
  - SpaceX API (https://api.spacexdata.com/v4/rockets/)
  - Wikipedia (https://en.wikipedia.org/wiki/List_of_Falcon/_9/_and_Falcon_Heavy_launches)
  - Web Scraping Technics.

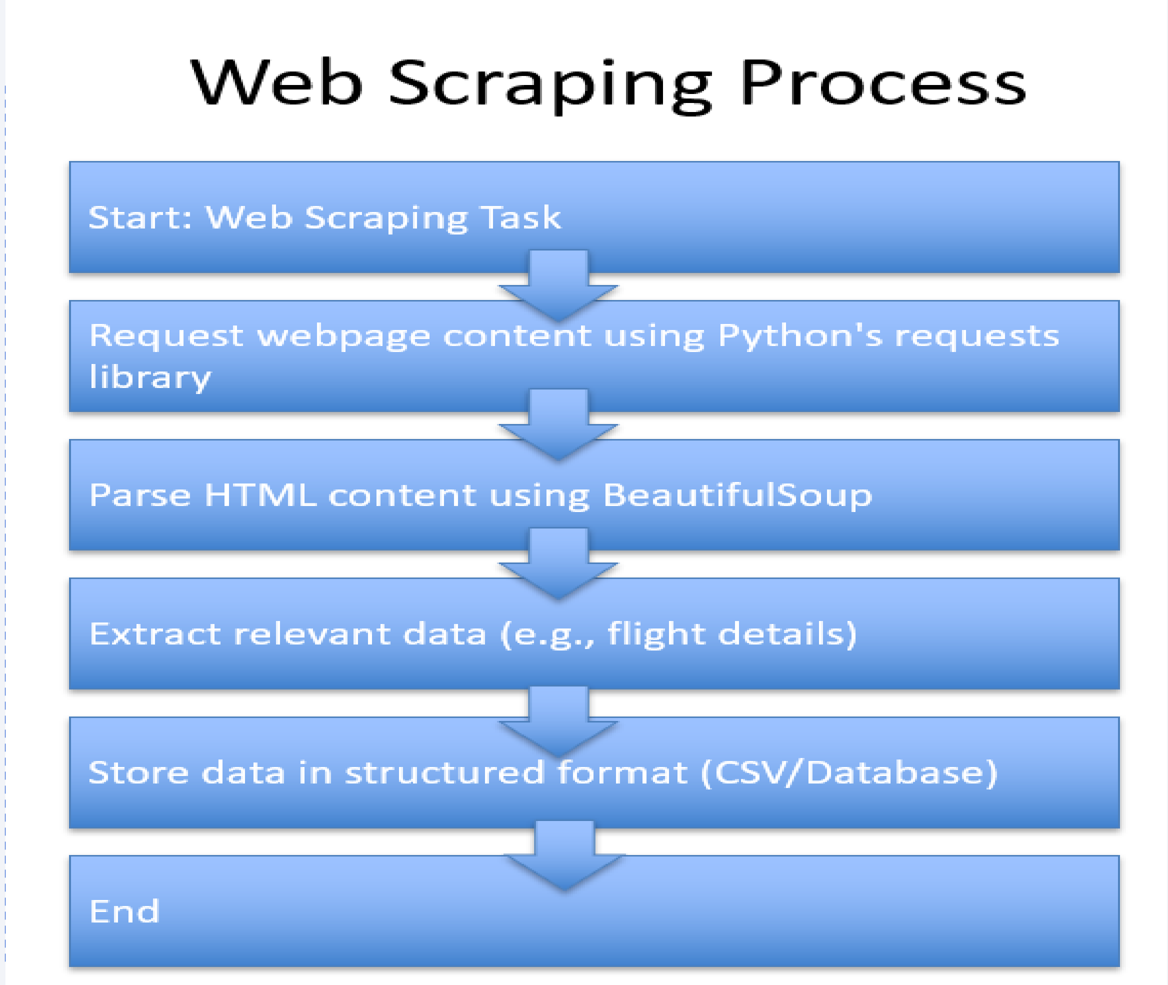
Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.
- The link the git notebook where data is collected is : https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/Capstone1_SpaceX_Data_Collection_API.ipynb



Data Collection - Scraping

- Using our web scraping techniques to the launches as BeautifulSoup, and the pandas features. (parsing table, converting to dataframe)
- The link the git notebook where data is collected is : https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/Capstone1_SpaceX_Data_Collection_API.ipynb



Data Wrangling

I. Collect Data (API, Csv,…)

II. Clean Data

  I. Handle missing data (e.g., drop or impute missing values)

  II. Correct inconsistencies and errors in data

III. Transform Data

  I. Normalize and standardize numerical data

  II. Convert categorical data into numerical representations (e.g., one-hot encoding)

IV. Feature Engineering

  I. Create new features based on existing data

  II. Select relevant features for model training

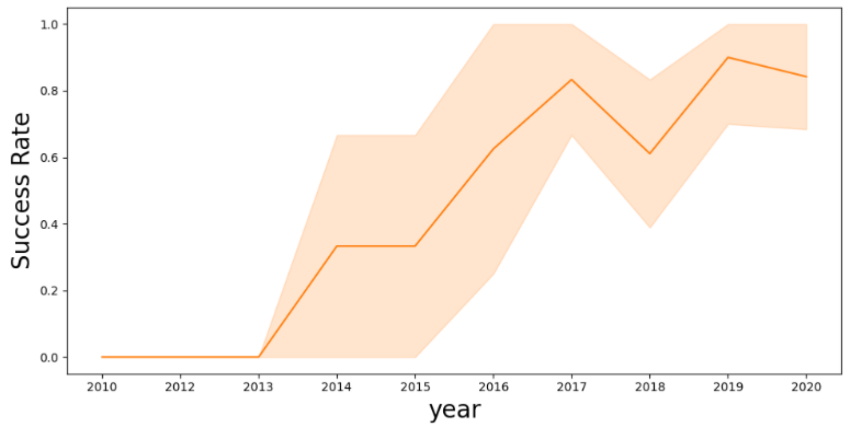
V. Data Splitting

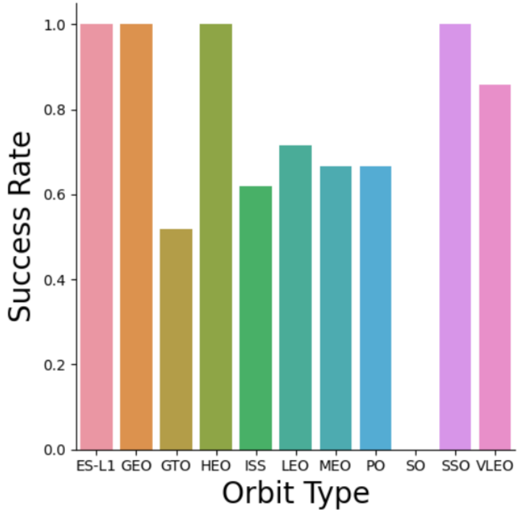
  I. Split the dataset into training and testing sets

- GitHub URL : https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/Capstone3_SpaceX_Data_Wrangling.ipynb

EDA with Data Visualization

- We analyzed the data by visualizing the relationships between flight number and launch site, payload and launch site, success rates for each orbit type, flight number and orbit type, as well as the yearly trends in launch success.
- GitHub URL : https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/Capstone5_SpaceX_EDA_Data_Visualization.ipynb





EDA with SQL

We imported the SpaceX dataset into a PostgreSQL database directly from the Jupyter notebook.We performed exploratory data analysis (EDA) using SQL to gain insights from the data. We crafted queries to uncover information such as:

The unique launch site names in the space missions,

The total payload mass carried by NASA (CRS) boosters,

The average payload mass for booster version F9 v1.1,

The total number of successful and failed mission outcomes,

The failed landing outcomes on the drone ship, along with their corresponding booster versions and launch site names

GitHub URL :
https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/Capstone4_SpaceX_EDA_SQL.ipynb

Build an Interactive Map with Folium

- We marked all launch sites on the map and added various map objects, such as markers, circles, and lines, to indicate the success or failure of launches for each site using Folium. We classified the launch outcomes as 0 for failure and 1 for success.By utilizing color-coded marker clusters, we identified launch sites with relatively high success rates. Additionally, we calculated the distances from each launch site to nearby features and answered questions such as:
  - Are the launch sites located near railways, highways, or coastlines?
  - Do the launch sites maintain a certain distance from urban areas?
- GitHub URL : https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/app.py

Build a Dashboard with Plotly Dash

- We created an interactive dashboard using Plotly Dash.
  - We visualized the total number of launches for each site with pie charts.
  - We displayed a scatter plot illustrating the relationship between the launch outcome and payload mass (kg) for different booster versions.
- GitHub URL : https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/app.py

Predictive Analysis (Classification)

- First, loading the data using numpy and pandas, transforming the data and splitting our data into training and testing sets.
- Then we build machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning **to found the best performing classification model.**
- GitHub URL : https://github.com/SabrinaKhez/IBM_Coursera_Python/blob/main/Capstone8_SpaceX_Predictive_Analytics.ipynb

Results

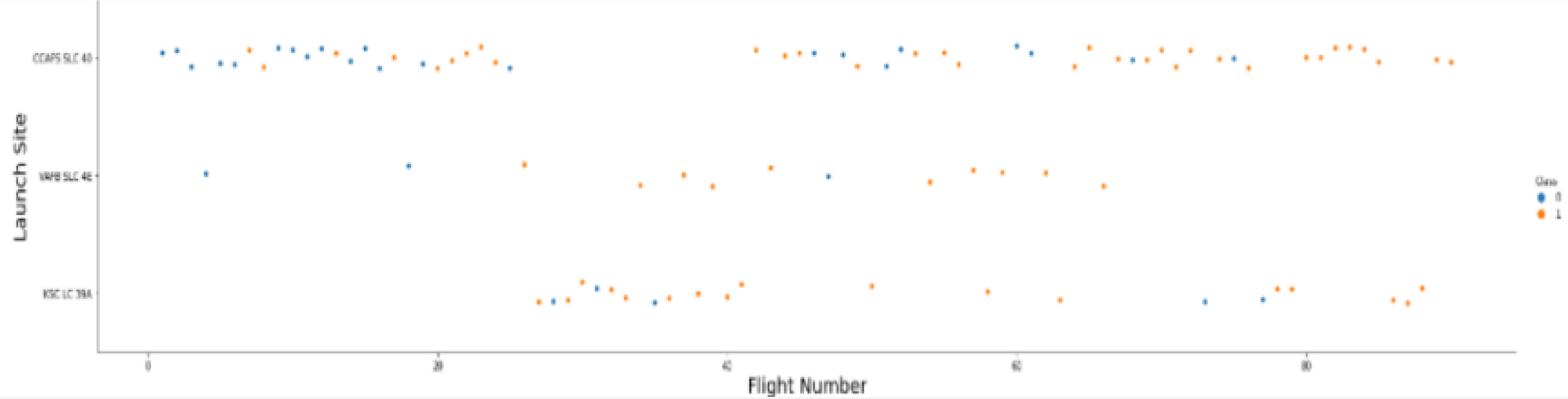
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

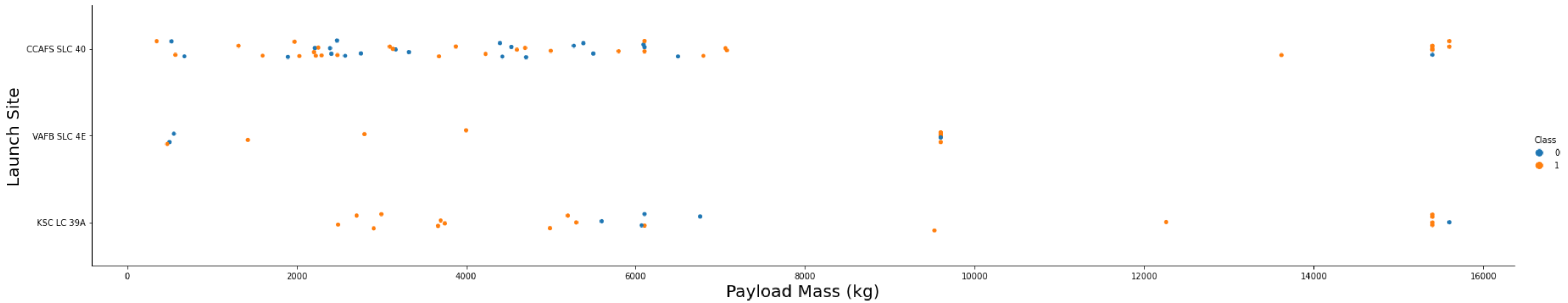
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- Analyzing the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



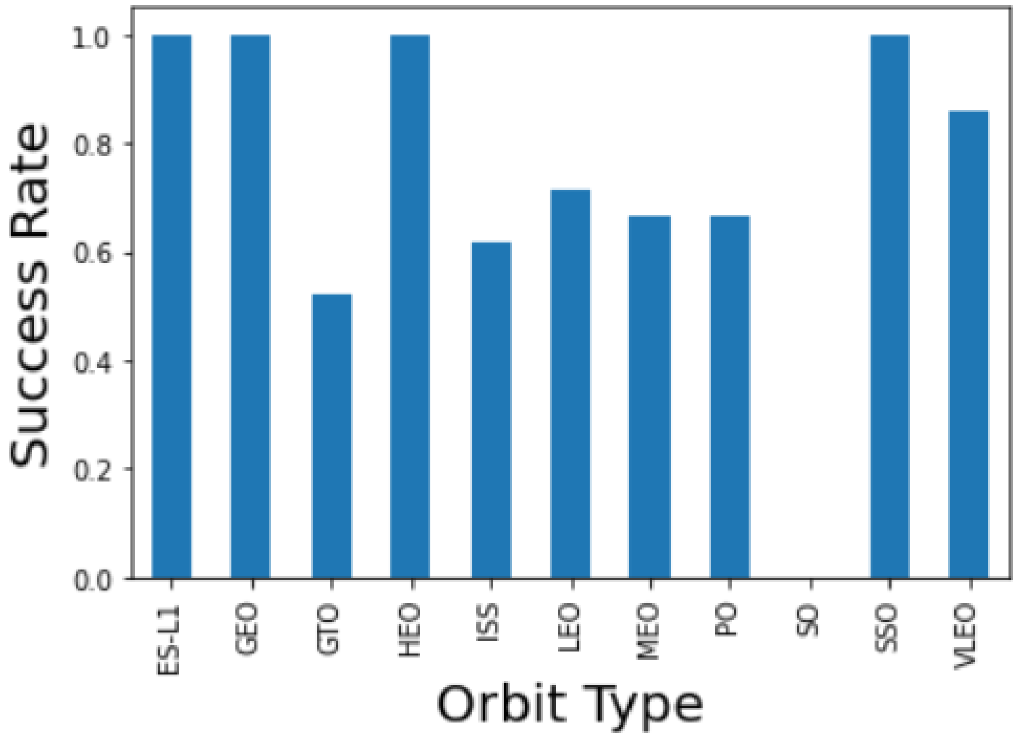


Payload vs. Launch Site

- We noticed that Payloads that approach MAX(Payload) tended to launch from CCAFS SLC 40 & KSC LC 39A
- Payloads less than 8000 kg tended to fail at a higher rate when launched from CCAFS SLC 40, plausibly due to that launch site being used for R&D versus the other two launch it used with less failure-tolerant payloads.

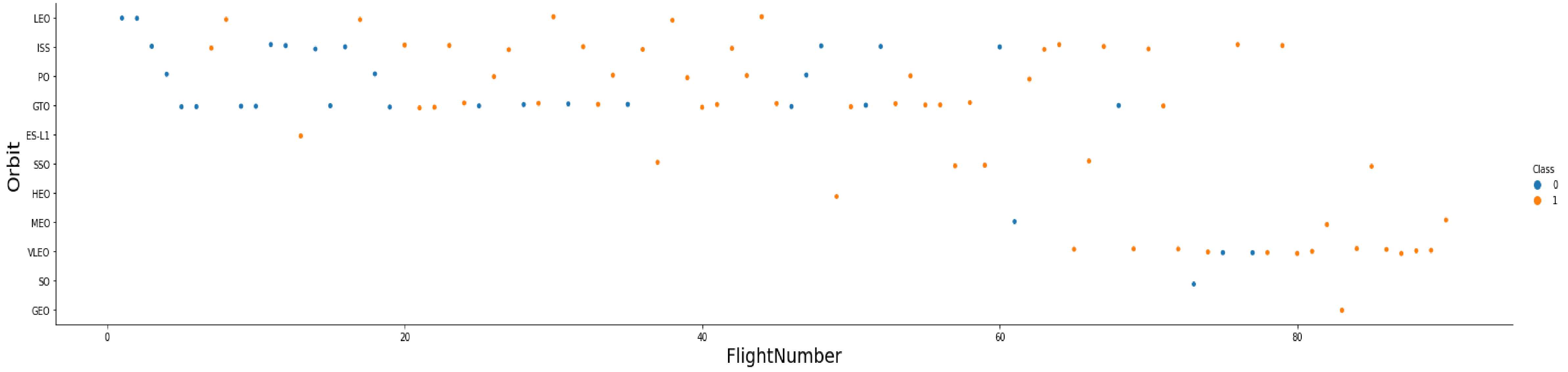
Success Rate vs. Orbit Type

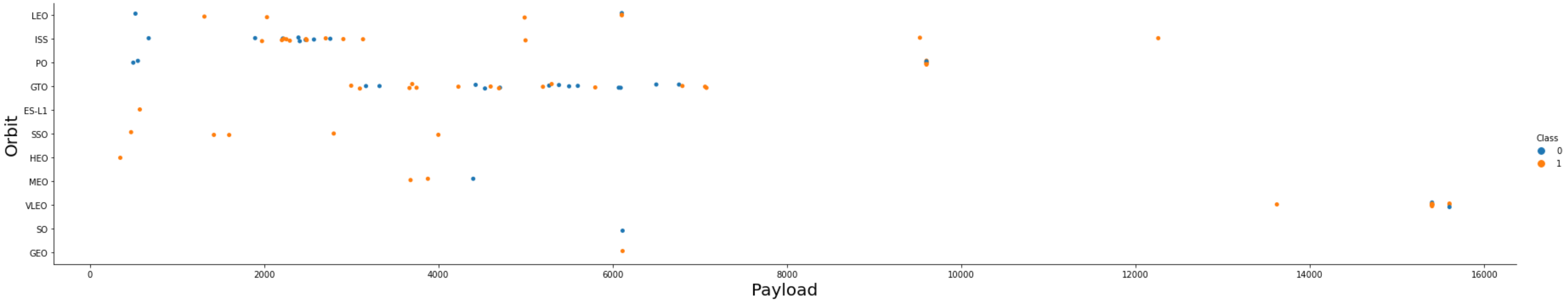
- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



Flight Number vs. Orbit Type

- We noticed that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

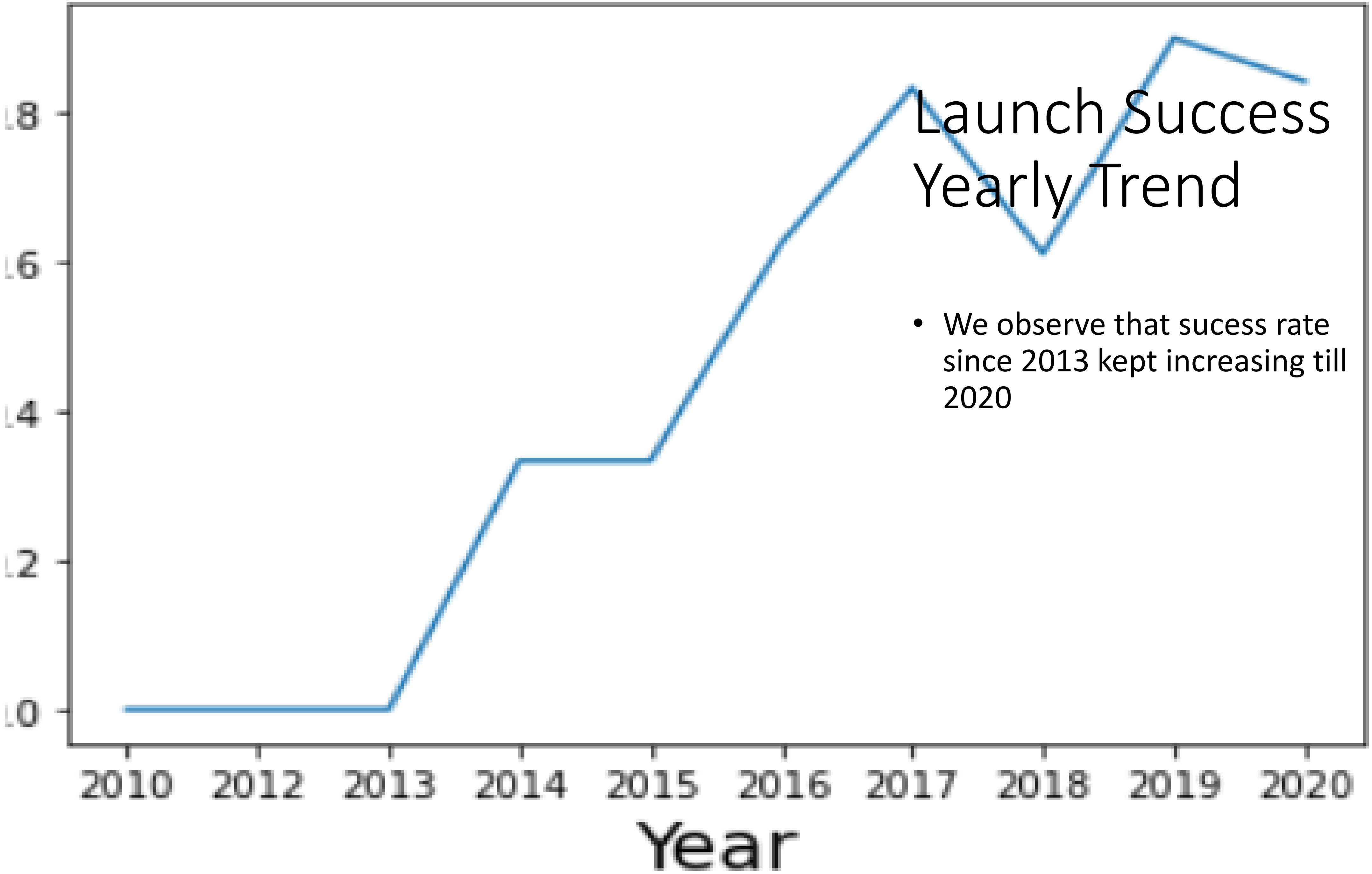




Payload vs. Orbit Type

- We observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

Launch Success Yearly Trend



- We observe that sucess rate since 2013 kept increasing till 2020

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

In [10]:
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

All Launch Site Names

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

Launch Site Names Begin with 'CCA'

Display 5 records where launch sites begin with the

```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

	date	time	boosterversion	launchsite
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40

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

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)

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

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

Average Payload Mass by F9 v1.1

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

```
In [14]: task_5 = '''
                 SELECT MIN(Date) AS FirstSuccessfull_landing_date
                 FROM SpaceX
                 WHERE LandingOutcome LIKE 'Success (ground pad)'
                 '''
         create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015.

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.

In [15]:

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	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

```
In [16]:
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

```
The total number of successful mission outcome is:
```

	successoutcome
0	100

```
The total number of failed mission outcome is:
```

Out[16]:

	failureoutcome
0	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.

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

In [17]:
```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

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.

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [18]:
```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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.

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

In [19]:

```
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)
```

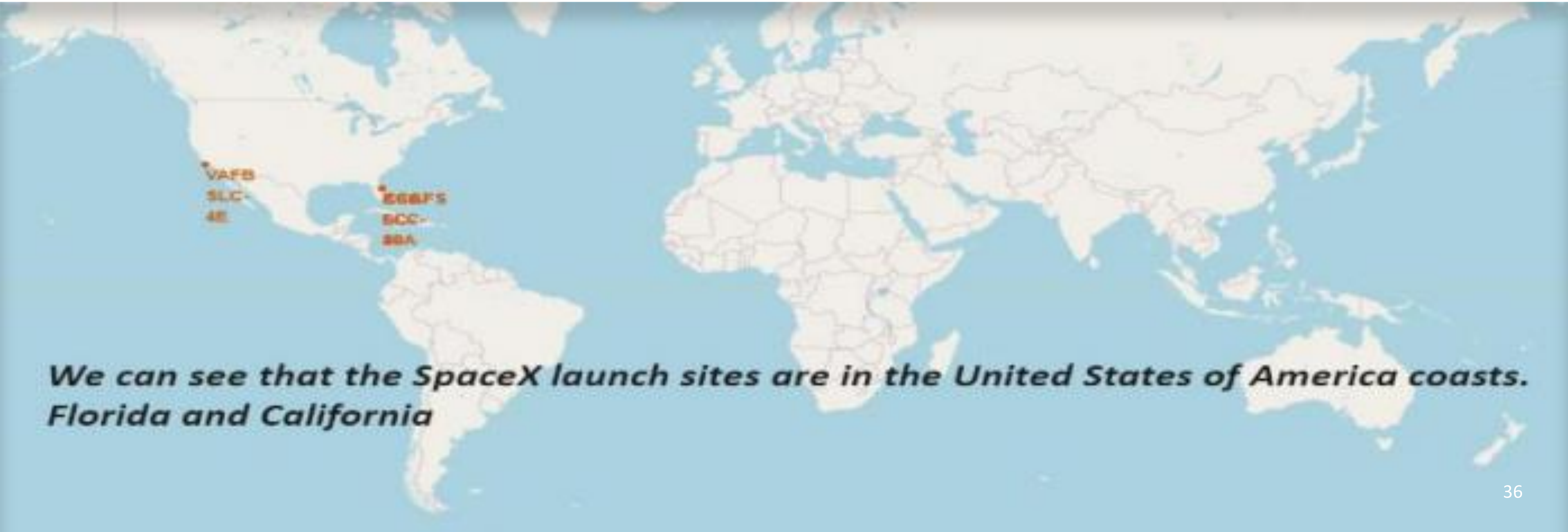
Out[19]:

	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

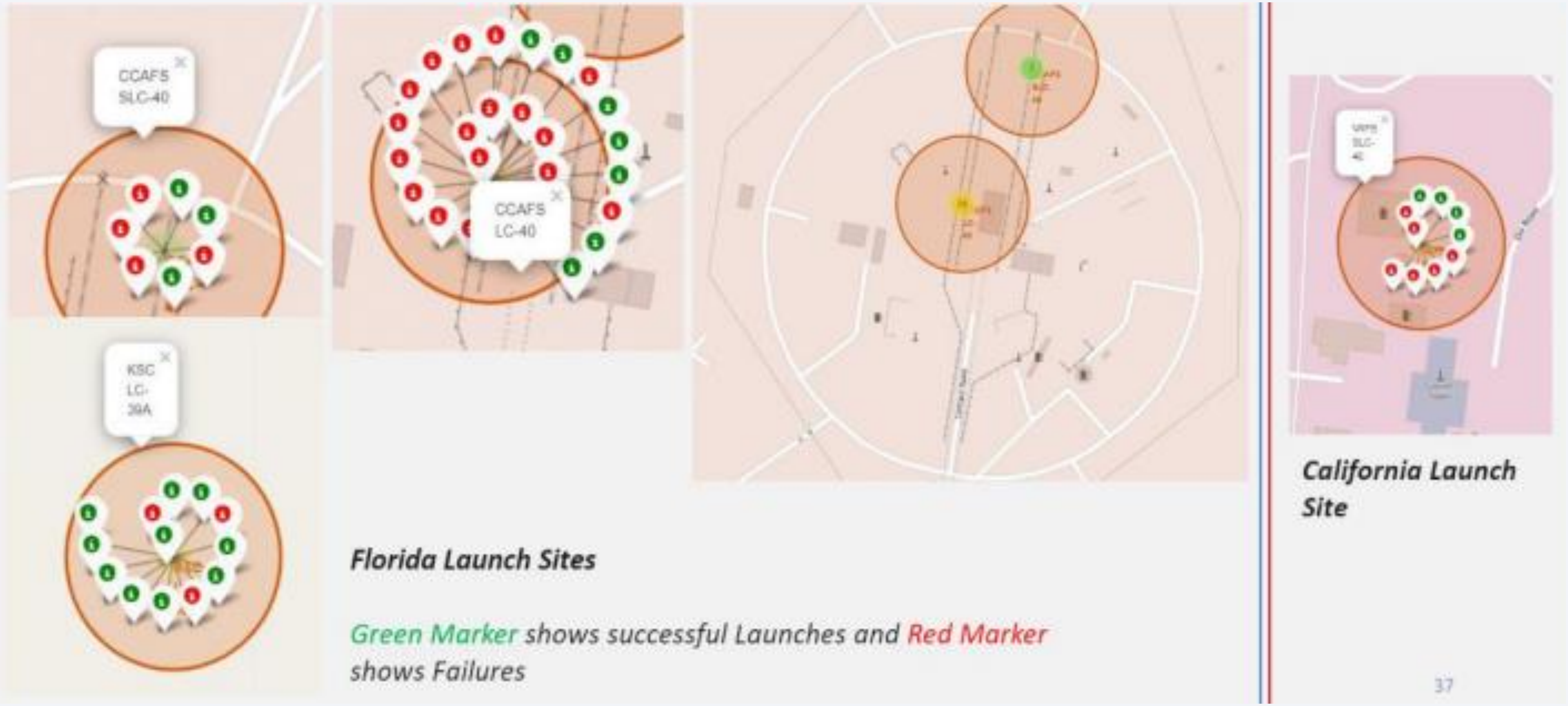
Section 3

Launch Sites Proximities Analysis

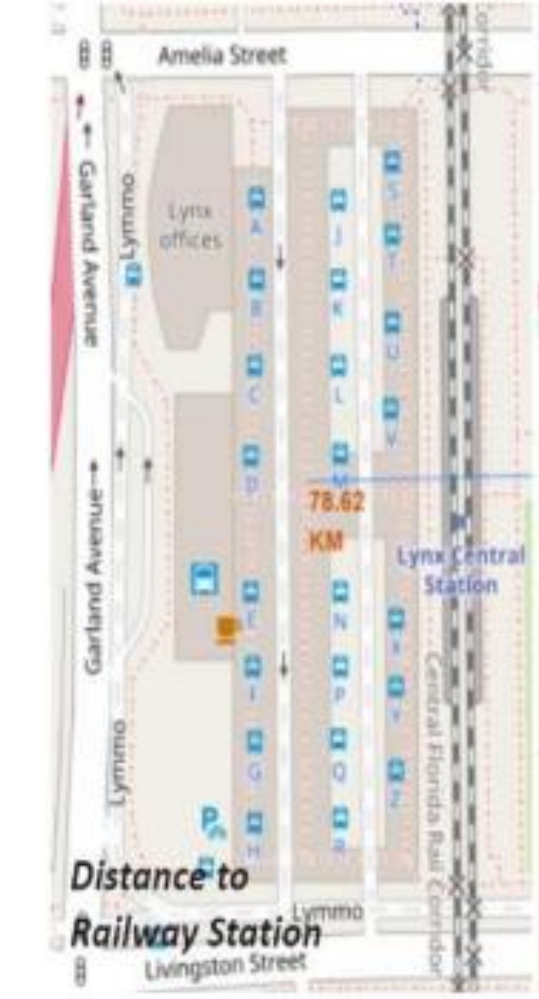
All launch sites global map markers

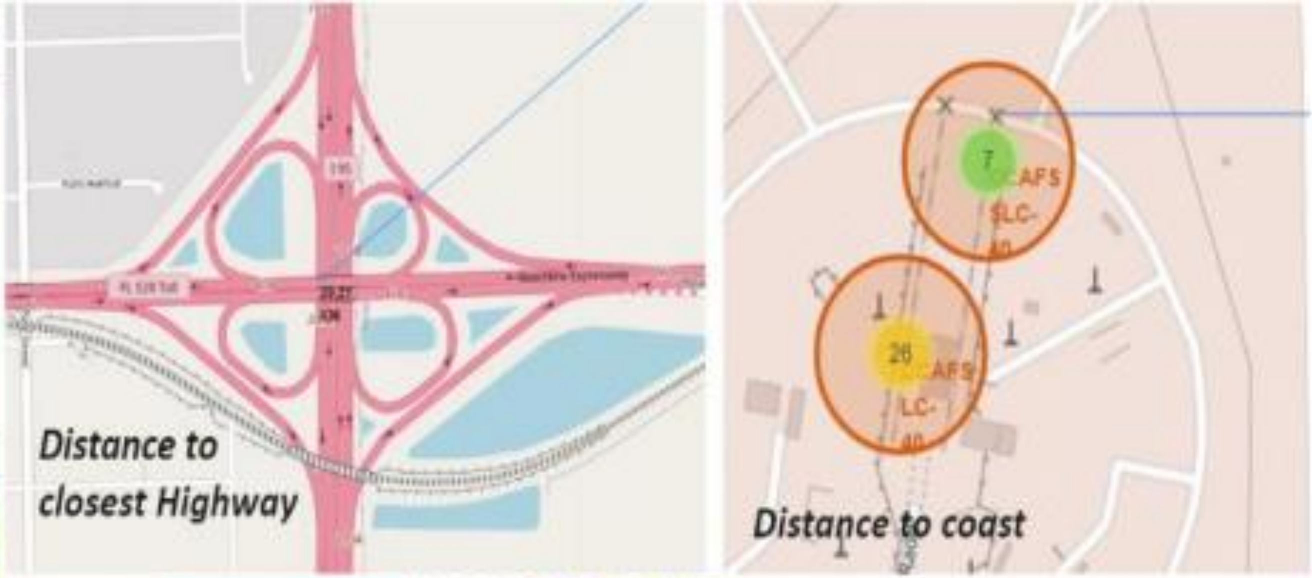


Markers showing launch sites with color labels



Launch Site distance to landmarks



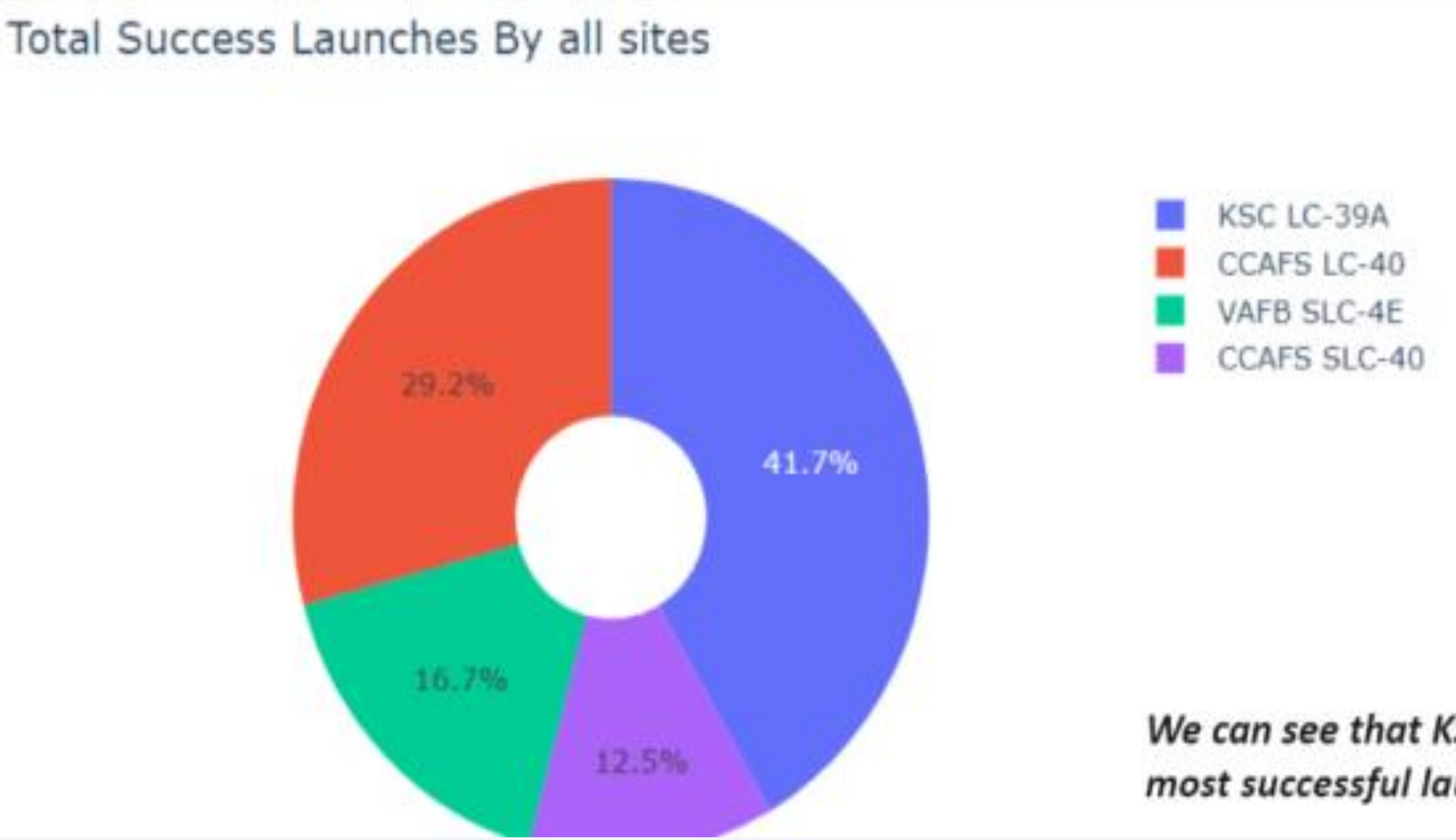


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

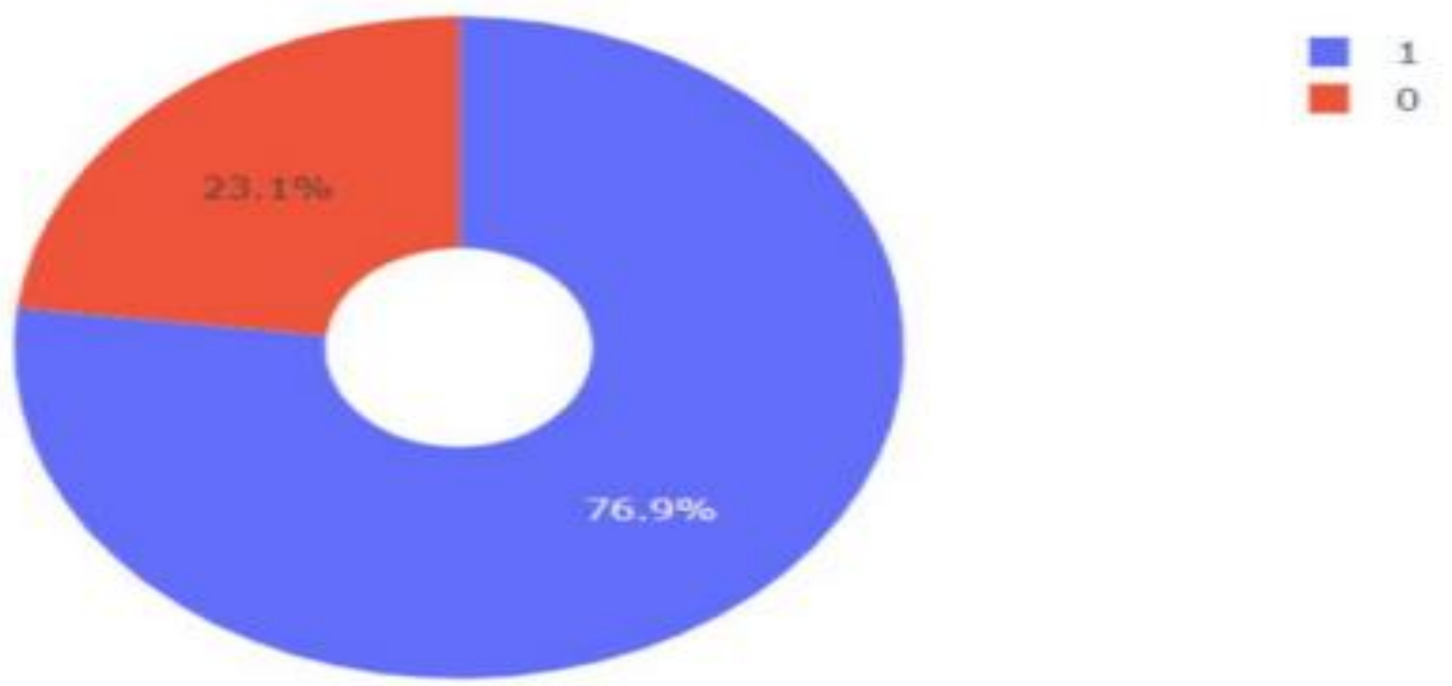
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



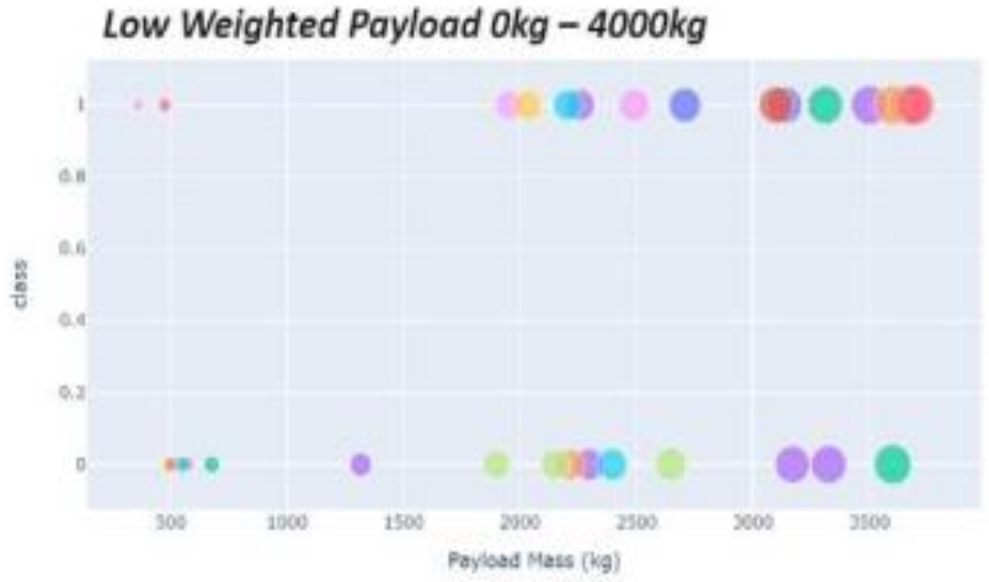
We can see that KSC LC-39A had the most successful launches from all the sites

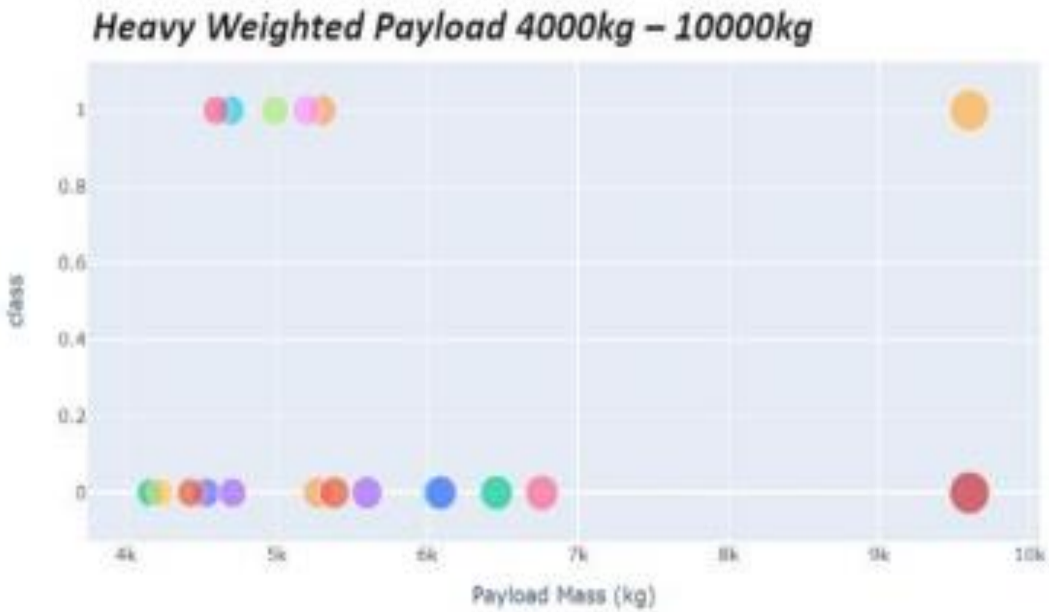
Pie chart showing the Launch site with the highest launch success ratio



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

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





We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

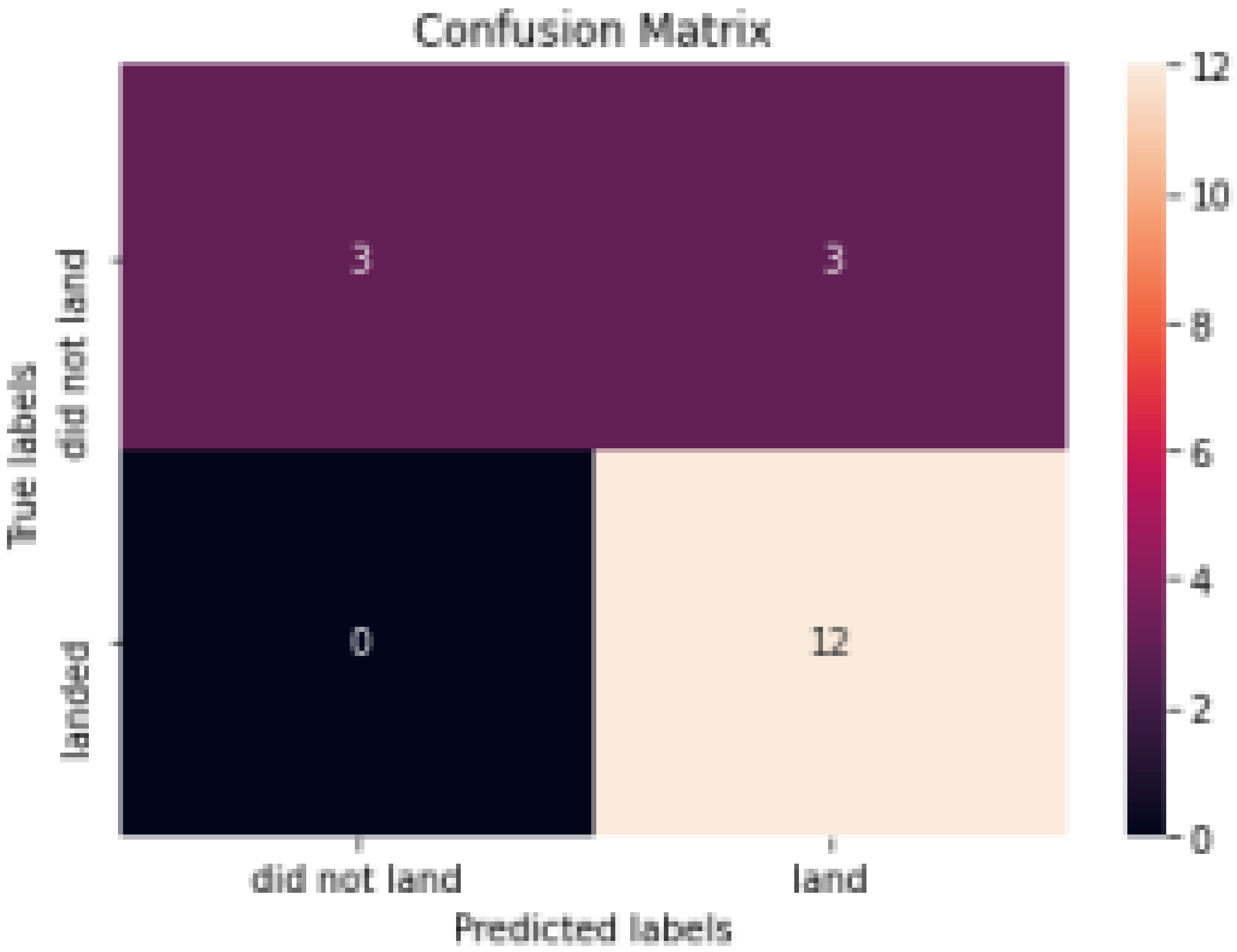
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter
```

Classification Accuracy

The decision tree classifier has the **highest** classification accuracy.

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

- We can conclude that:
  - The larger the flight amount at a launch site, the greater the success rate at a launch site.
  - Launch success rate started to increase in 2013 till 2020.
  - Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
  - KSC LC-39A had the most successful launches of any sites.
  - The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!