IBM Developer
SKILLS NETWORK

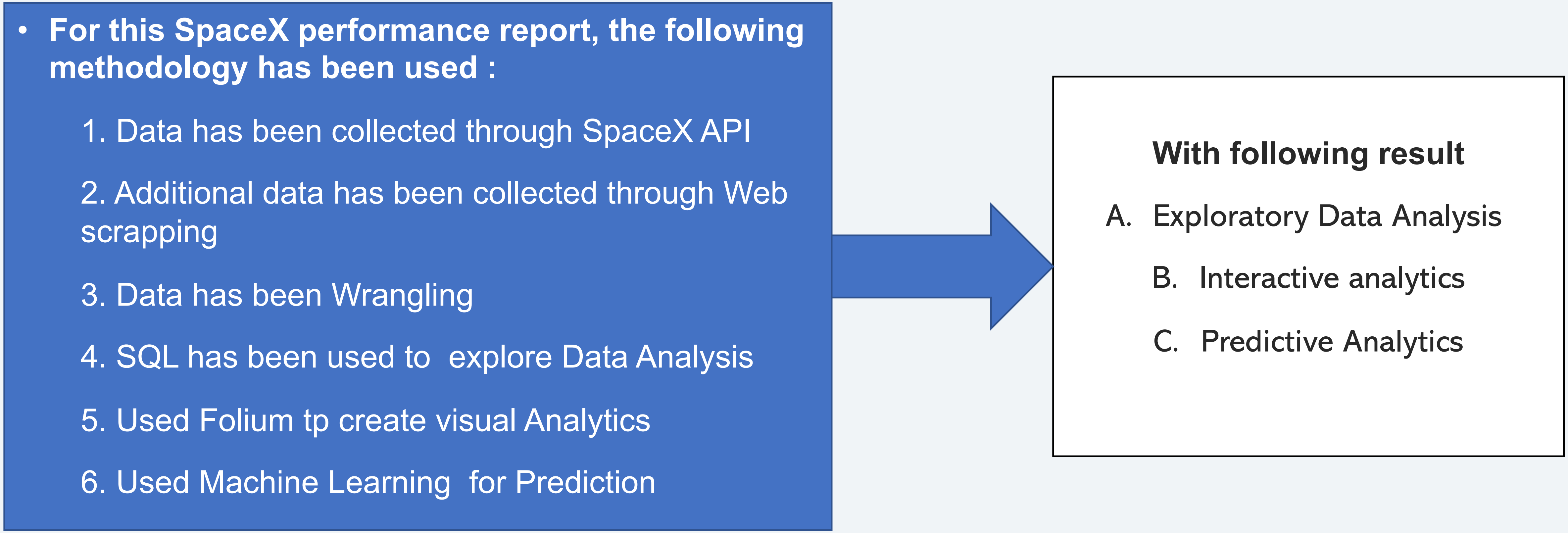
Winning Space Race with Data Science

Samson Ngov
July 24th 2022

Outline

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

Executive Summary



- **For this SpaceX performance report, the following methodology has been used :**

  1. Data has been collected through SpaceX API
  2. Additional data has been collected through Web scrapping
  3. Data has been Wrangling
  4. SQL has been used to explore Data Analysis
  5. Used Folium tp create visual Analytics
  6. Used Machine Learning for Prediction

With following result

A. Exploratory Data Analysis

B. Interactive analytics

C. Predictive Analytics

Introduction

- Project background and context

  Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

  - What are the factors that will determine if the rocket will land successfully?
  - What are the effect of the relationship between the rocket variable and the outcome ?
  - What Conditions needs to be in place to ensure the best result ?

Section 1

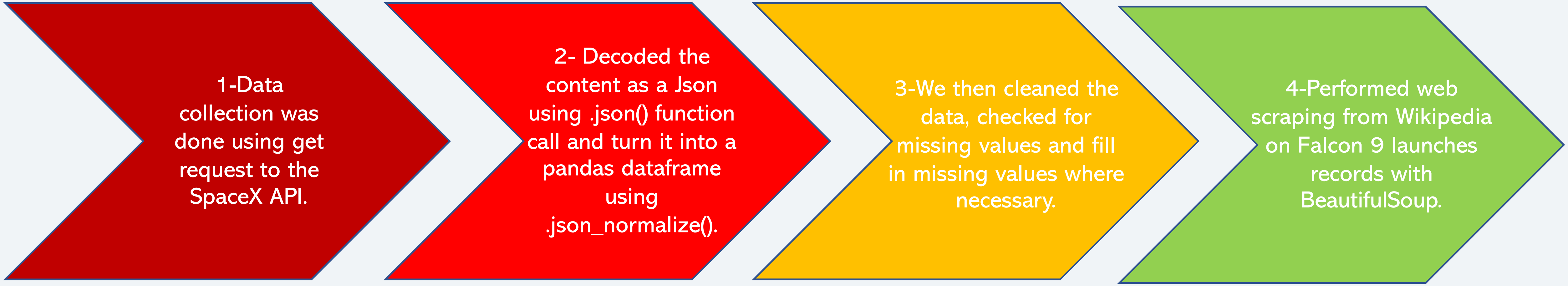
Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected webscraping from Wikipedia and using Space X API
- Perform data wrangling
  - We proceed a one hot encoding to simplify the feature by categorized them
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - We proceed with an study on different algorithm theory and defined what will be the most accurate one to be used.

Data Collection – SpaceX API



1-Data collection was done using get request to the SpaceX API.

2- Decoded the content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().

3-We then cleaned the data, checked for missing values and fill in missing values where necessary.

4-Performed web scraping from Wikipedia on Falcon 9 launches records with BeautifulSoup.

Data Collection – SpaceX API

1- Connect with SpaceX API

2- Collect the datas

3- Cleaned the data

4- Wrangling and Formating

Codes could be find on:

https://github.com/samsonngov/SpaceX-assigment/blob/main/SpaceX%20assignment%20data%20collection%20final.ipynb

Data Collection - Scraping

1-web scrapping to webscrap Falcon 9 launch records with BeautifulSoup

2- parsed the table and converted it into a pandas dataframe.

Codes could be find on:

https://github.com/samsonngov/SpaceX-assigment/blob/main/SpaceX%20assigment%20data%20wrangling.ipynb

Data Wrangling

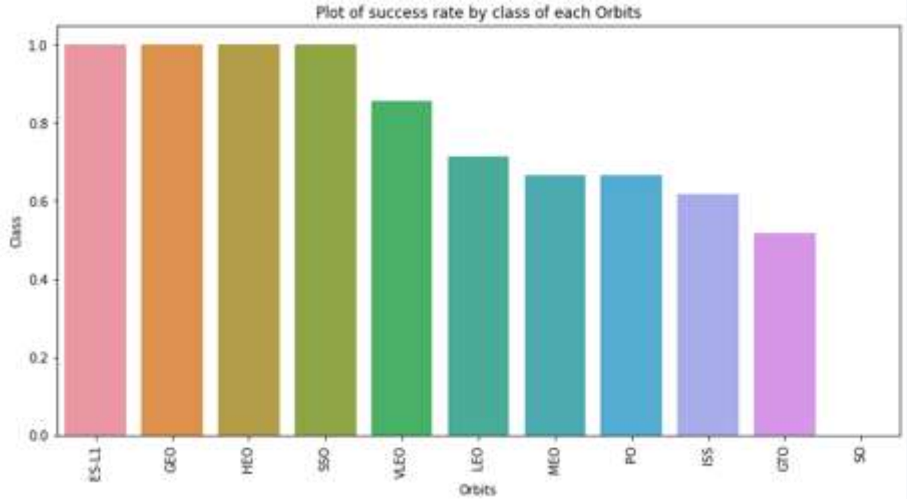
- Performed exploratory data analysis and determined the training labels.
  - Calculated the number of launches at each site, and the number and occurrence of each orbits
- Created landing outcome label from outcome column and exported the results to csv.

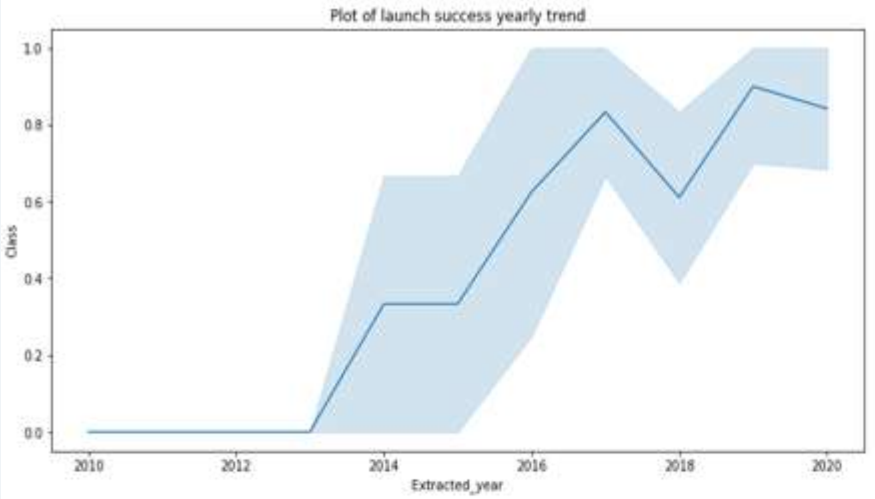
Codes could be find on:

https://github.com/samsonngov/SpaceX-assigment/blob/main/SpaceX%20assigment%20data%20wrangling.ipynb

EDA with Data Visualization

- Data exploration by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.





Codes could be find on:

https://github.com/samsonngov/SpaceX-assigment/blob/main/SpaceX%20assigment%20Dataviz.ipynb

EDA with SQL

Loaded the SpaceX dataset into a PostgreSQL database

Applied EDA with SQL to get insight from the data.

Wrote queries to find out for instance:

Codes could be find on:

https://github.com/samsonngov/SpaceX-assigment/blob/main/Space%20X%20assigment%20SQL.ipynb

Build an Interactive Map with Folium

- **MARKED** all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- **ASSIGNED** the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- **COLOR LABELED** marker clusters, we identified which launch sites have relatively high success rate.
- **CALCULATED** the distances between a launch site to its proximities.

Build a Dashboard with Plotly Dash

- BUILT an interactive dashboard with Plotly dash
- PLOTTED pie charts showing the total launches by different sites
- PLOTTED scatter graph showing the relationship between Outcome and Payload Mass (Kg) for the different booster version.

Predictive Analysis (Classification)

- **LOADED** the data using numpy and pandas, transformed the data, split our data into training and testing.
- **BUILT** different machine learning models and tune different hyperparameters using GridSearchCV.
- **USED** accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- **FOUND** the best performing classification model.

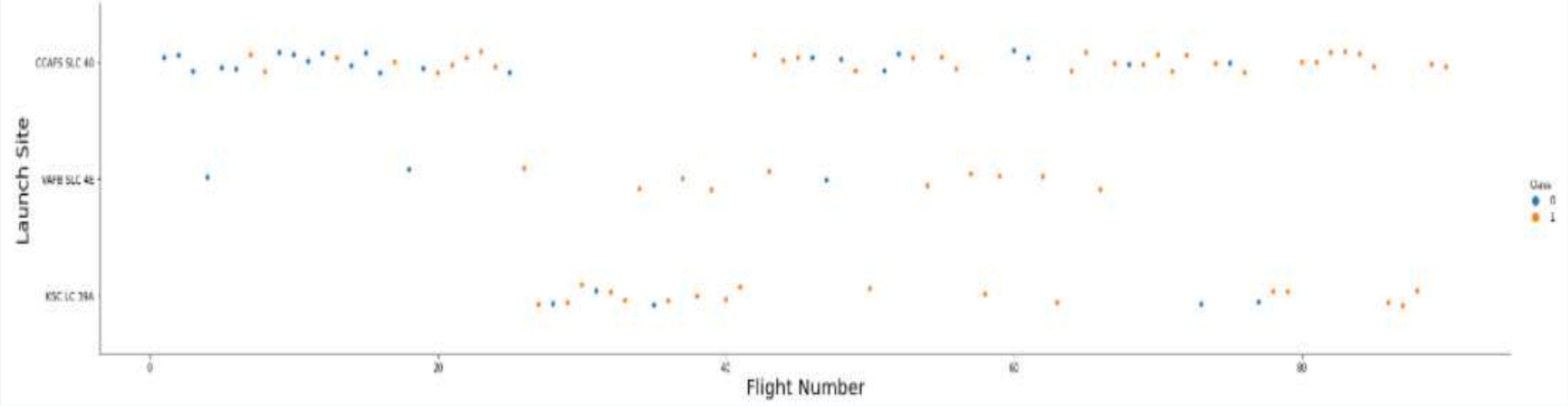
Results

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

Section 2

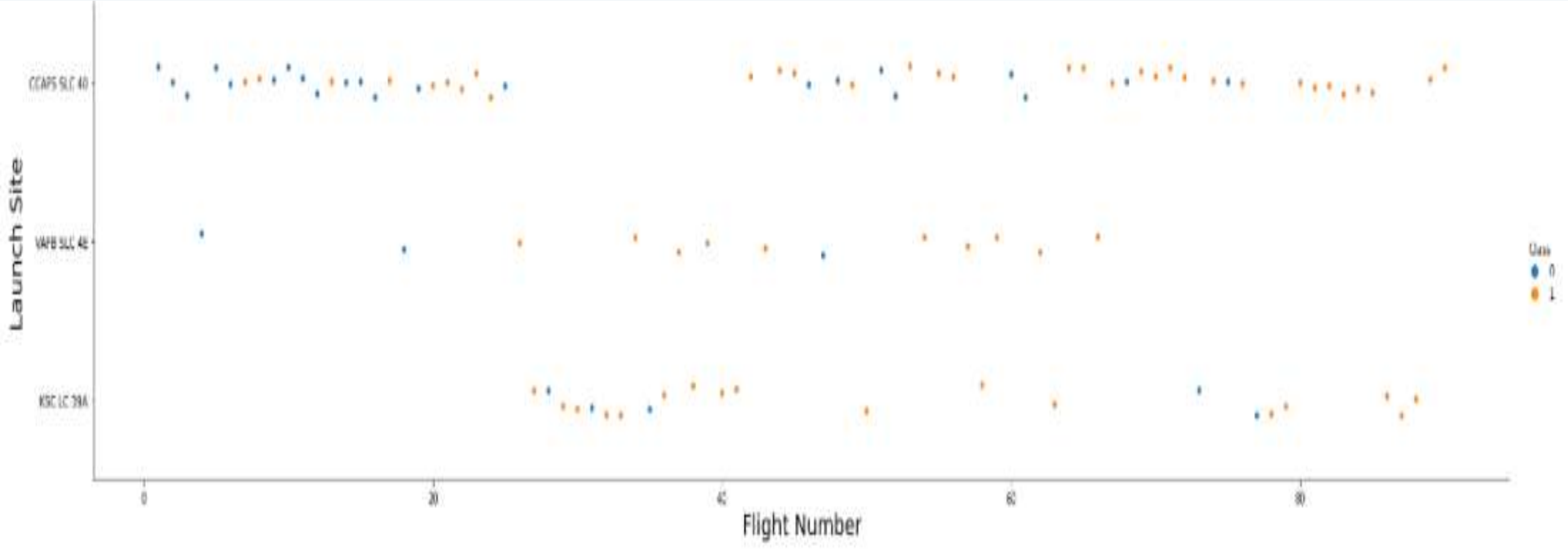
Insights drawn from EDA

Flight Number vs. Launch Site

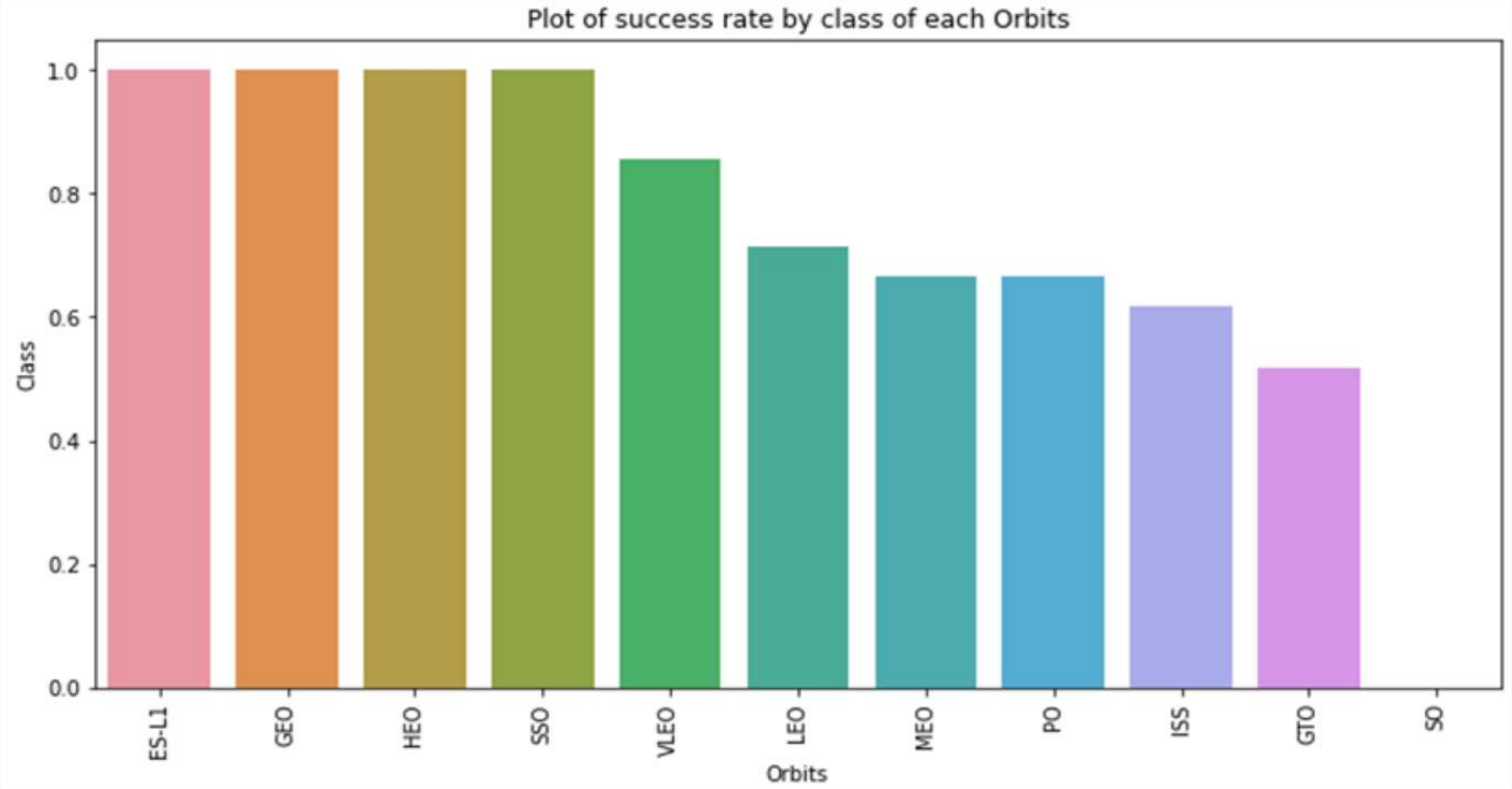


Scatter plot of Flight Number vs. Launch Site

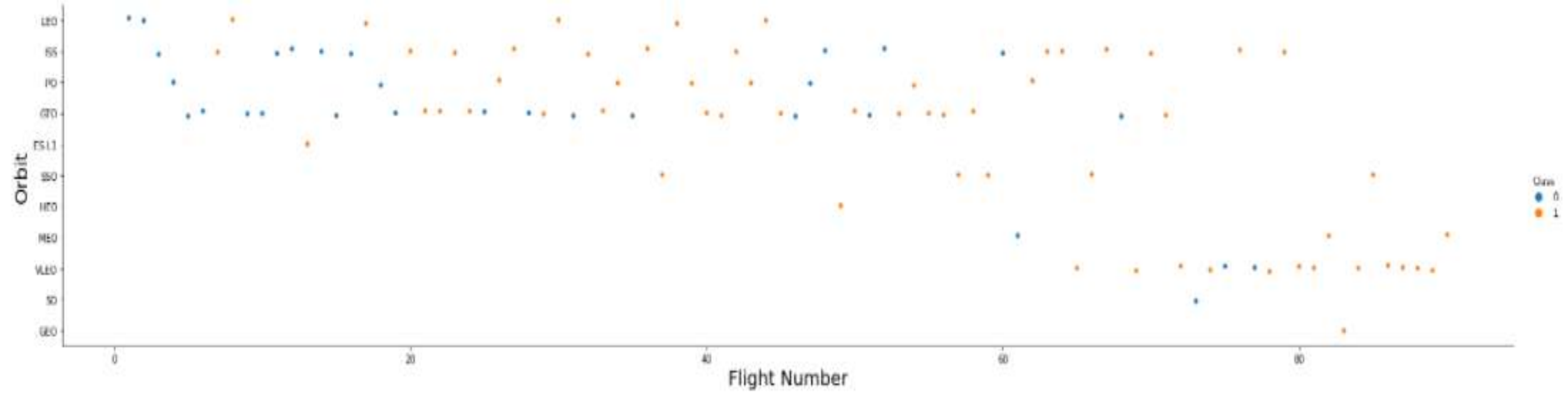
Payload vs. Launch Site



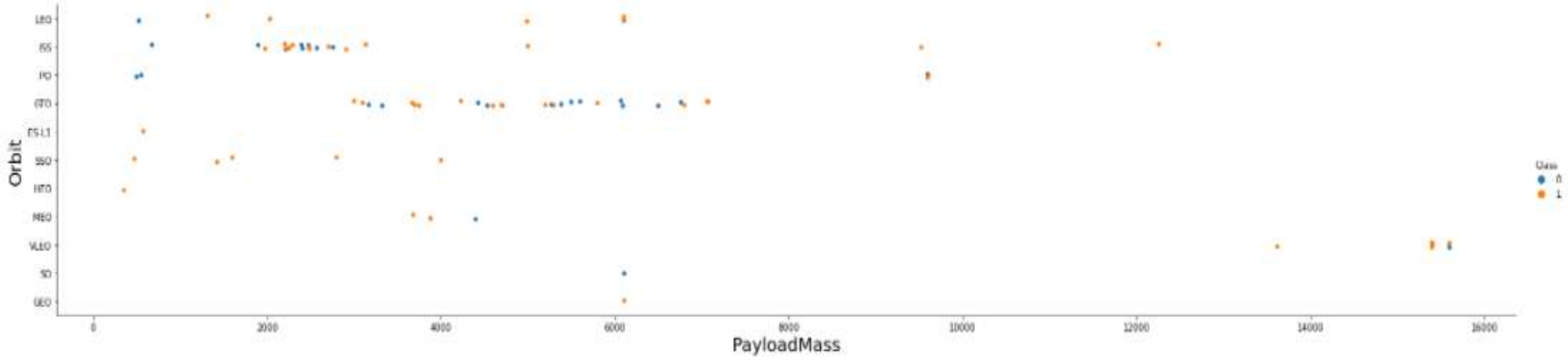
Success Rate vs. Orbit Type



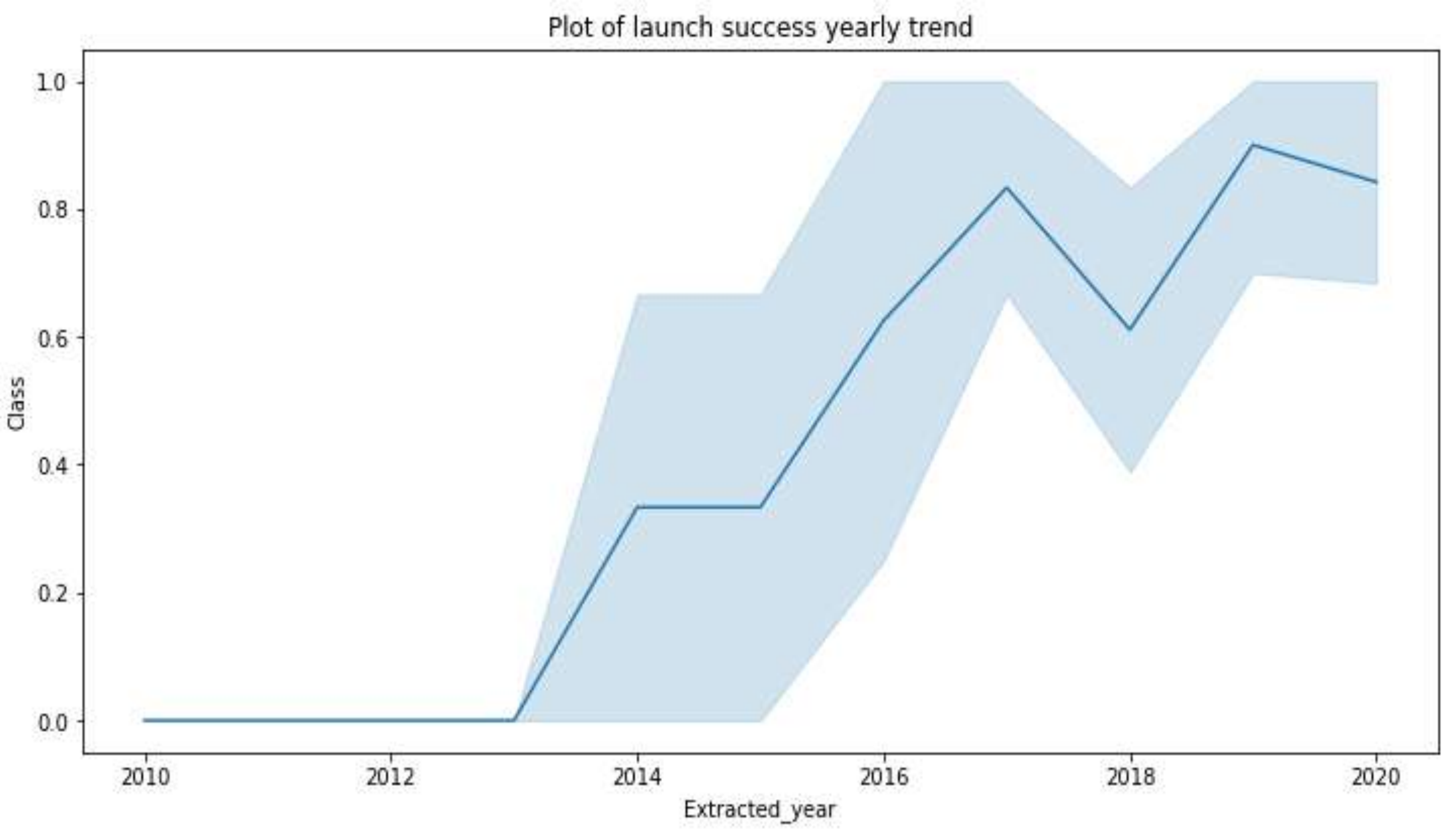
Flight Number vs. Orbit Type



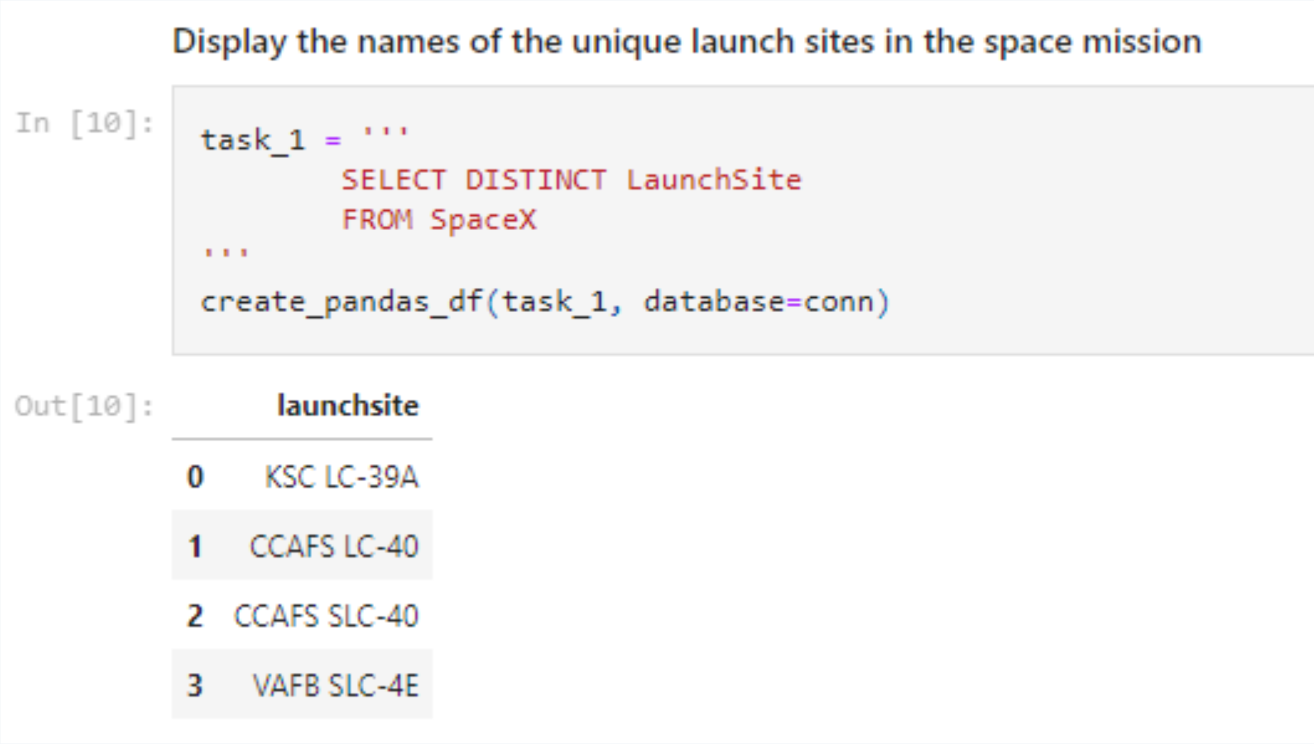
Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names



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

In [10]:

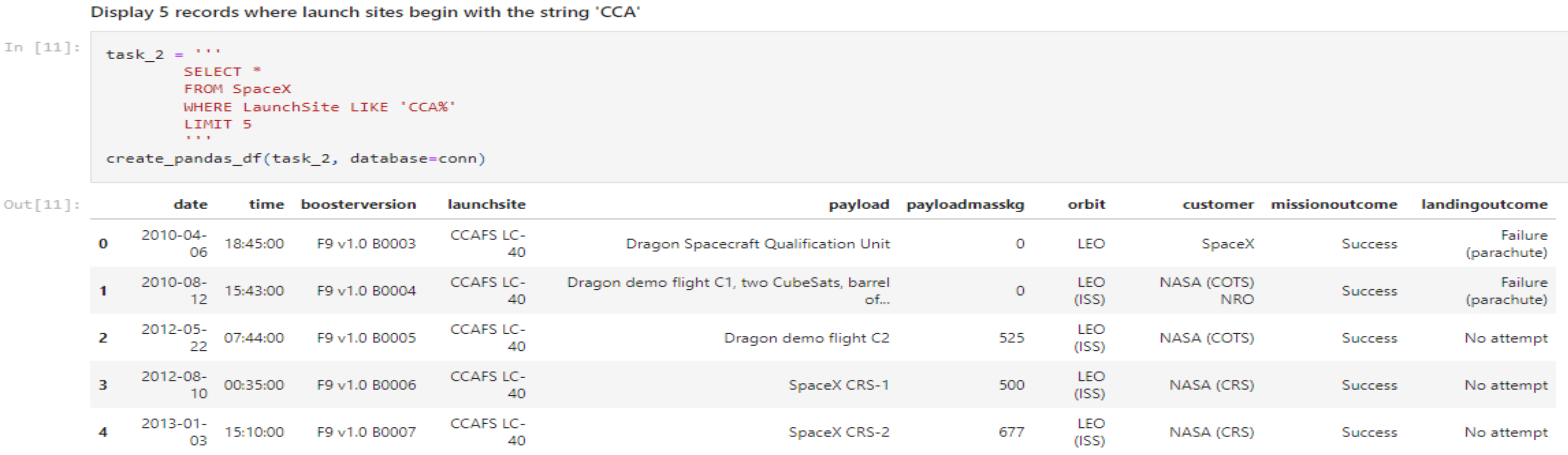
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

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

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 string 'CCA'

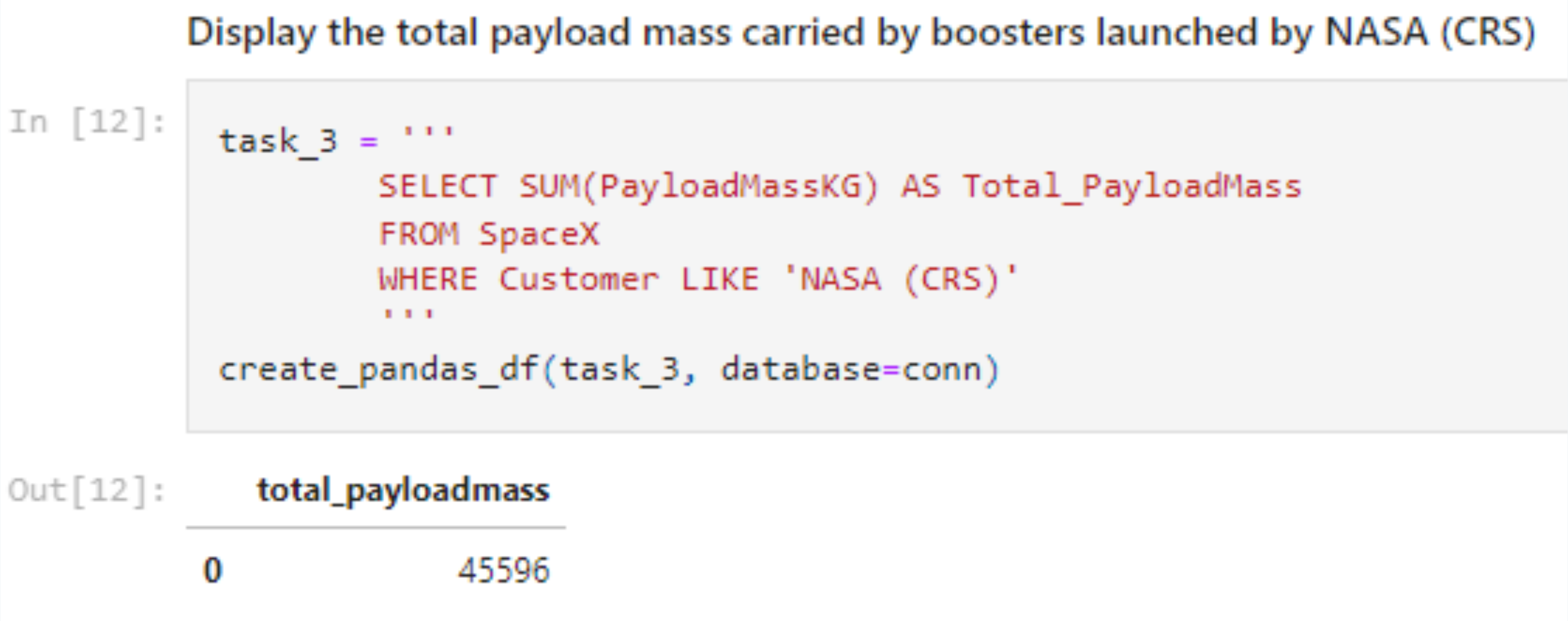
```
In [11]:  task_2 = '''
                  SELECT *
                  FROM SpaceX
                  WHERE LaunchSite LIKE 'CCA%'
                  LIMIT 5
                  '''
          create_pandas_df(task_2, database=conn)
```

Out[11]:

	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

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

Total Payload Mass



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

In [12]:

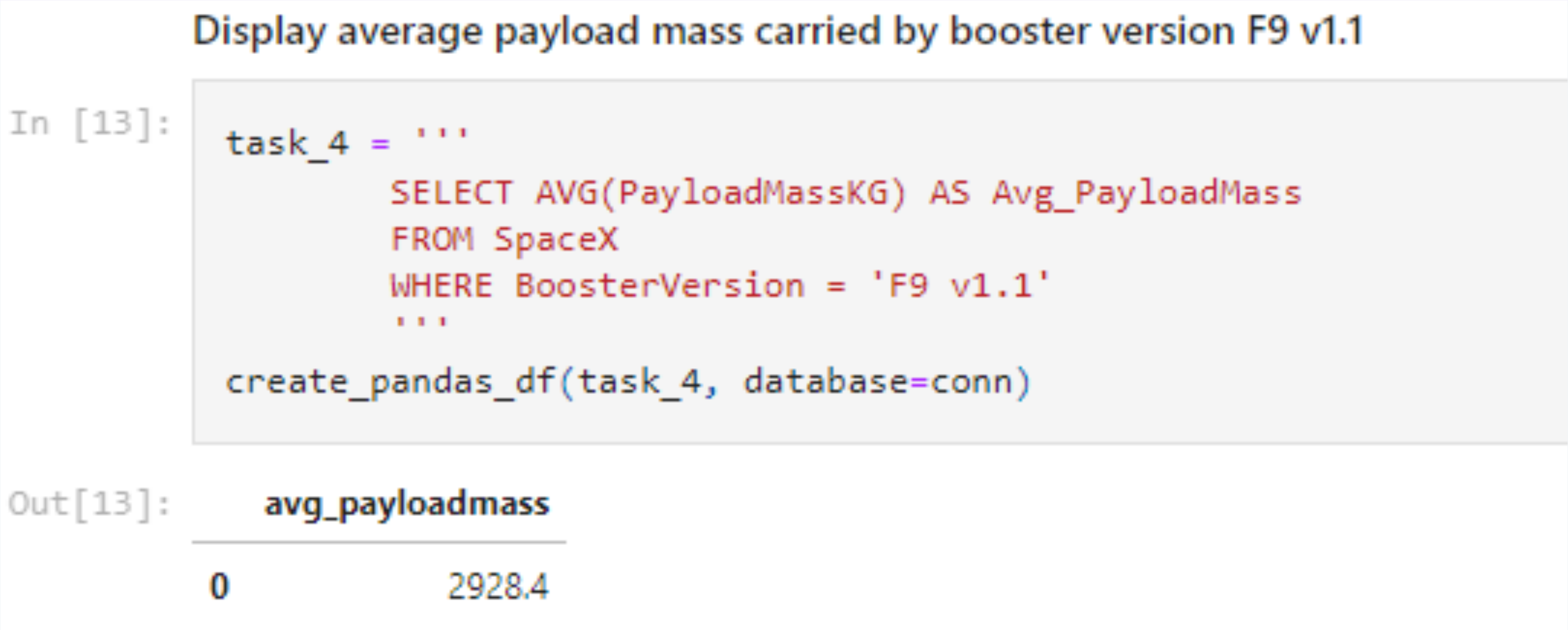
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

- **CALCULATED** the total payload carried by boosters from NASA as 45596 using the query below

Average Payload Mass by F9 v1.1



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

In [13]:

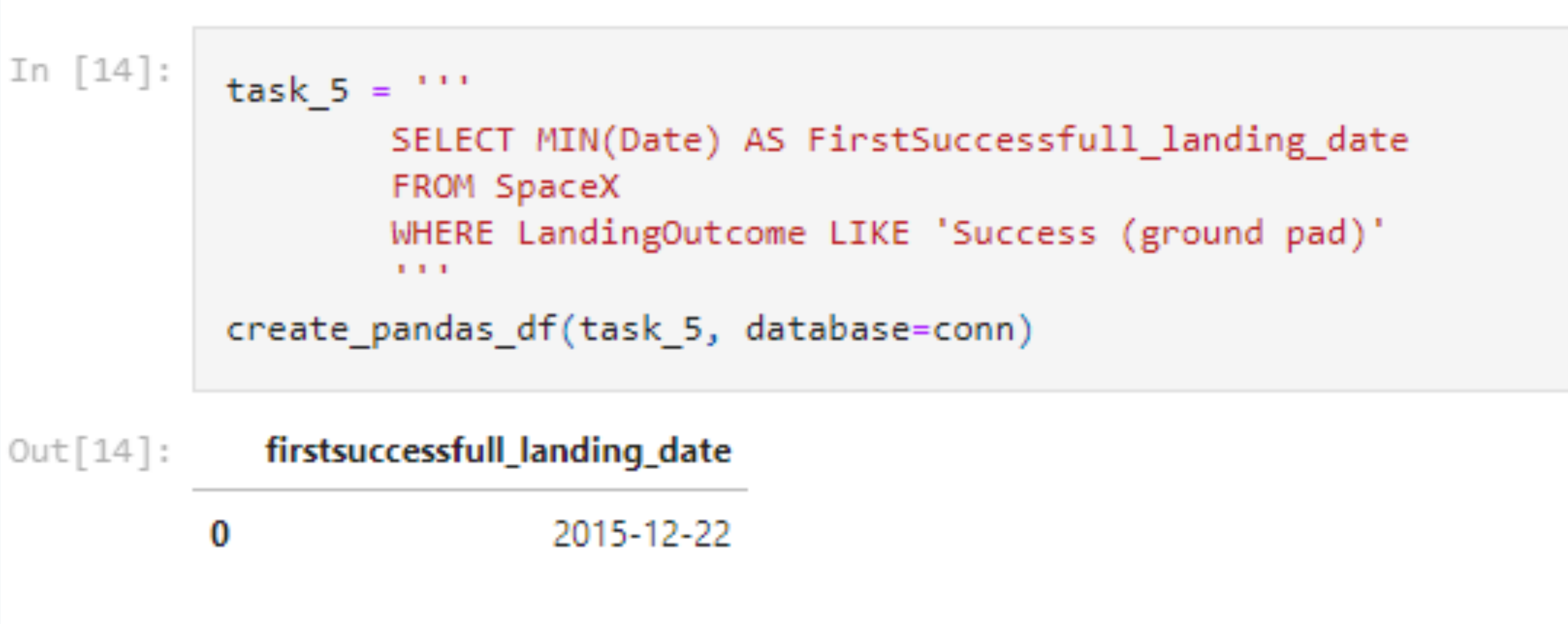
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

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

First Successful Ground Landing Date !

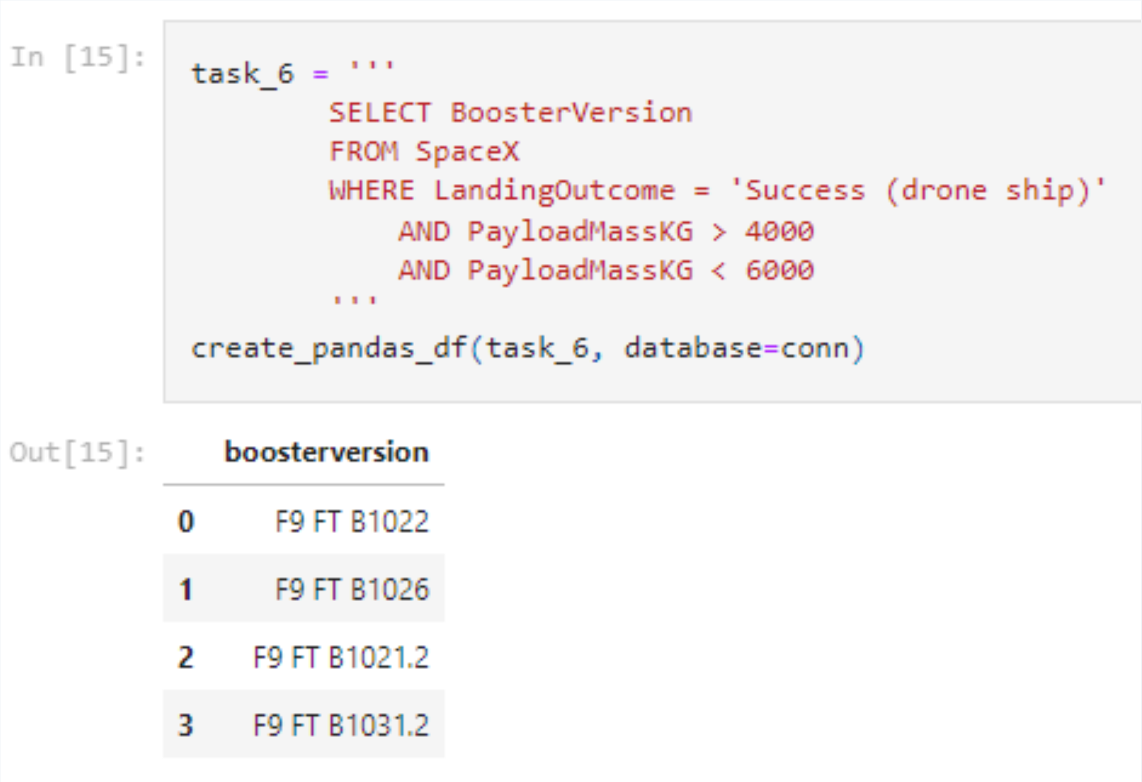


```
In [14]:  task_5 = '''
                  SELECT MIN(Date) AS FirstSuccessfull_landing_date
                  FROM SpaceX
                  WHERE LandingOutcome LIKE 'Success (ground pad)'
                  '''
          create_pandas_df(task_5, database=conn)
```

```
Out[14]:     firstsuccessfull_landing_date
          0                     2015-12-22
```

- **OBSERVED** 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

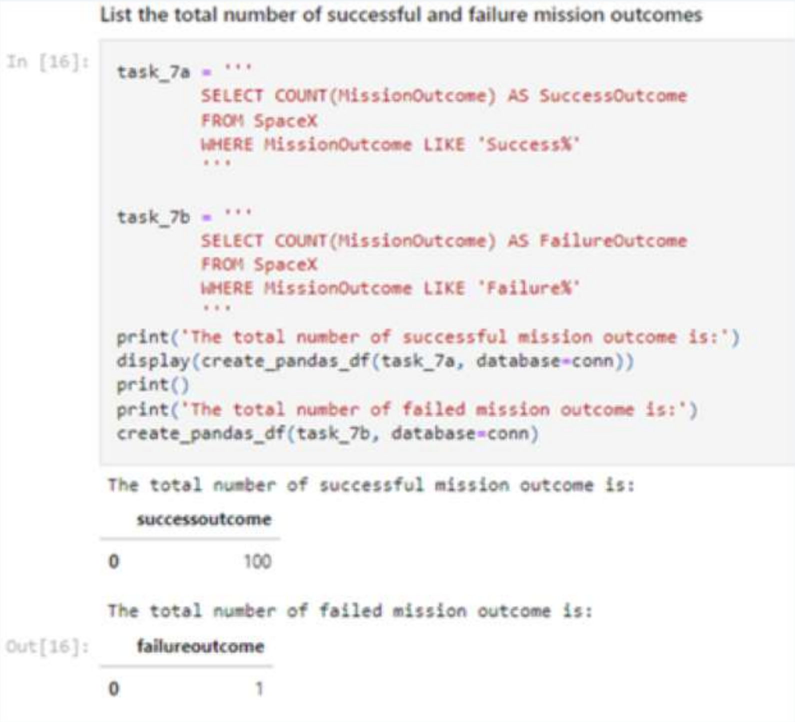


```
In [15]:  task_6 = '''
                  SELECT BoosterVersion
                  FROM SpaceX
                  WHERE LandingOutcome = 'Success (drone ship)'
                      AND PayloadMassKG > 4000
                      AND PayloadMassKG < 6000
                  '''
          create_pandas_df(task_6, database=conn)
```

```
Out[15]:    boosterversion
          0    F9 FT B1022
          1    F9 FT B1026
          2  F9 FT B1021.2
          3  F9 FT B1031.2
```

- 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

Total Number of Successful and Failure Mission Outcomes



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

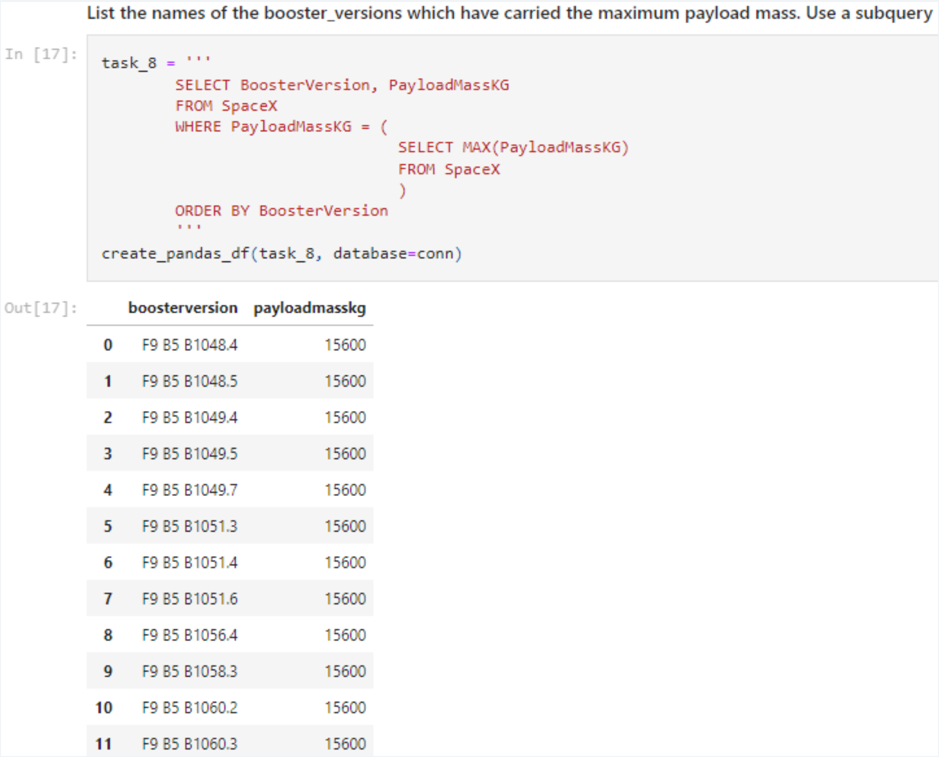
```
The total number of failed mission outcome is:
```

Out[16]:

	failureoutcome
0	1

- USED wildcard like '%' to filter for **WHERE** MissionOutcome was a success or a failure.

Boosters Carried Maximum Payload



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

In [17]:

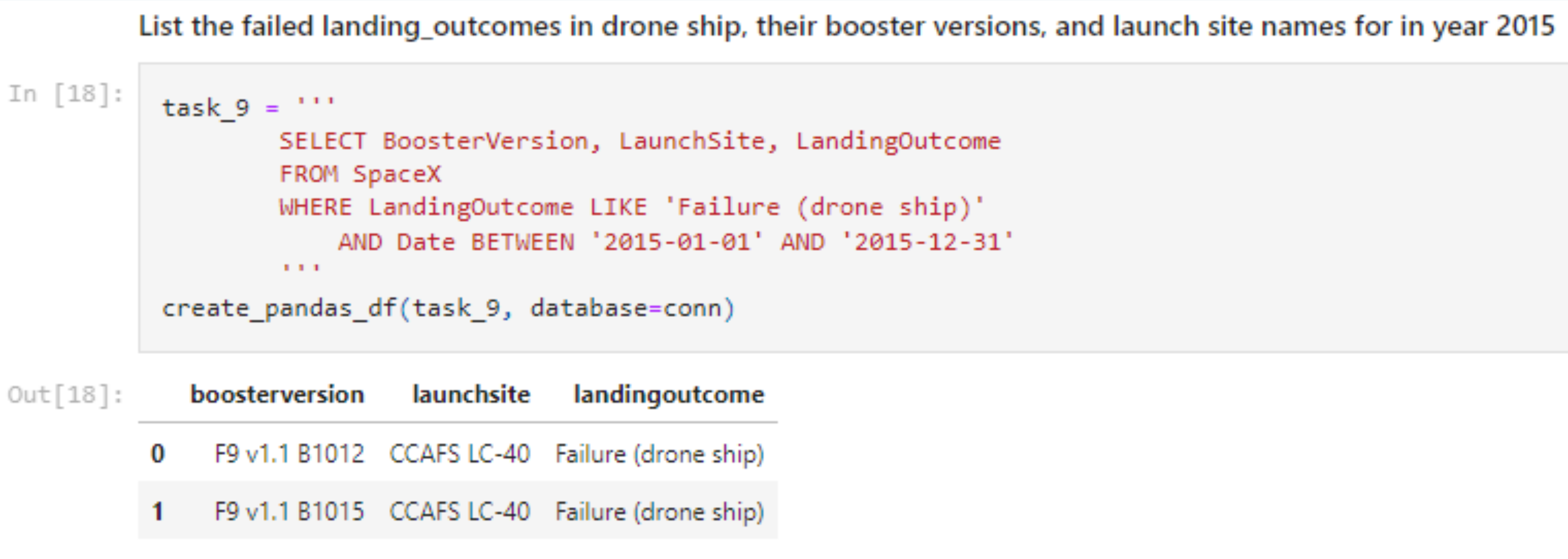
```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, 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

- DERTMINED the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

2015 Launch Records



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

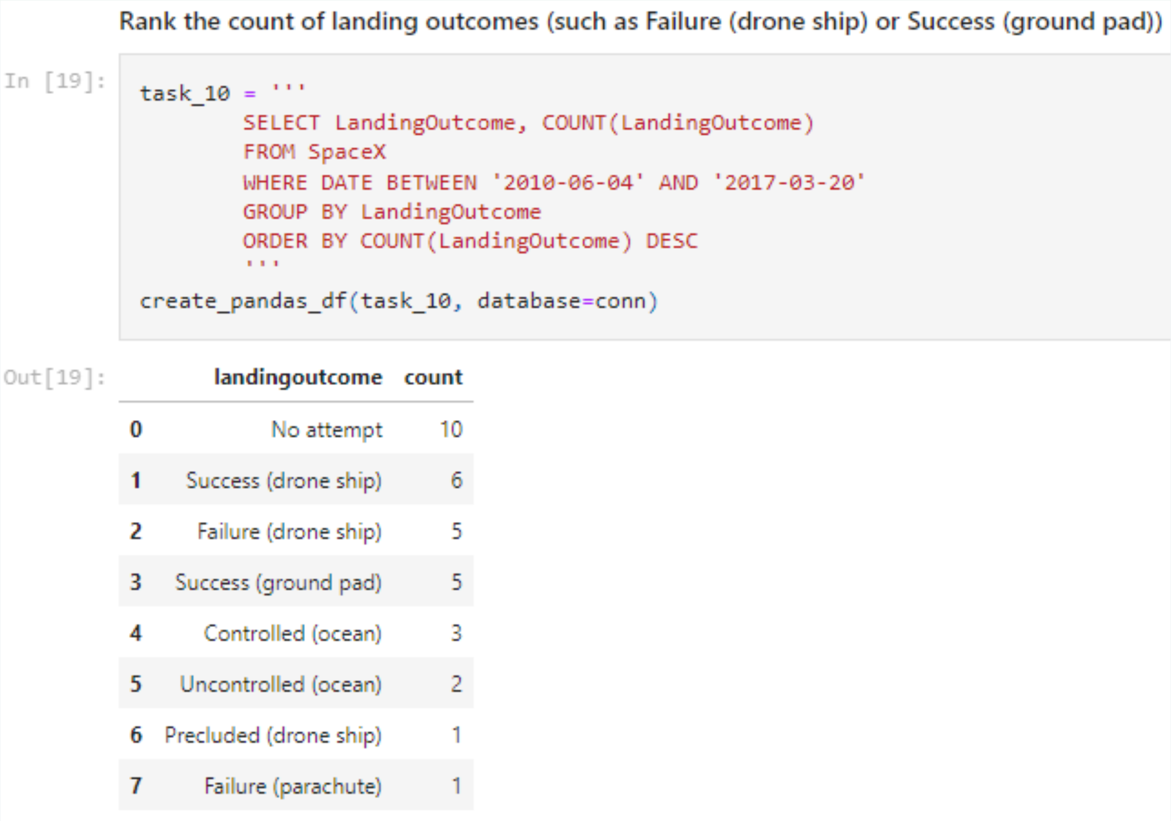
```
In [18]:

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)

- **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

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



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]:

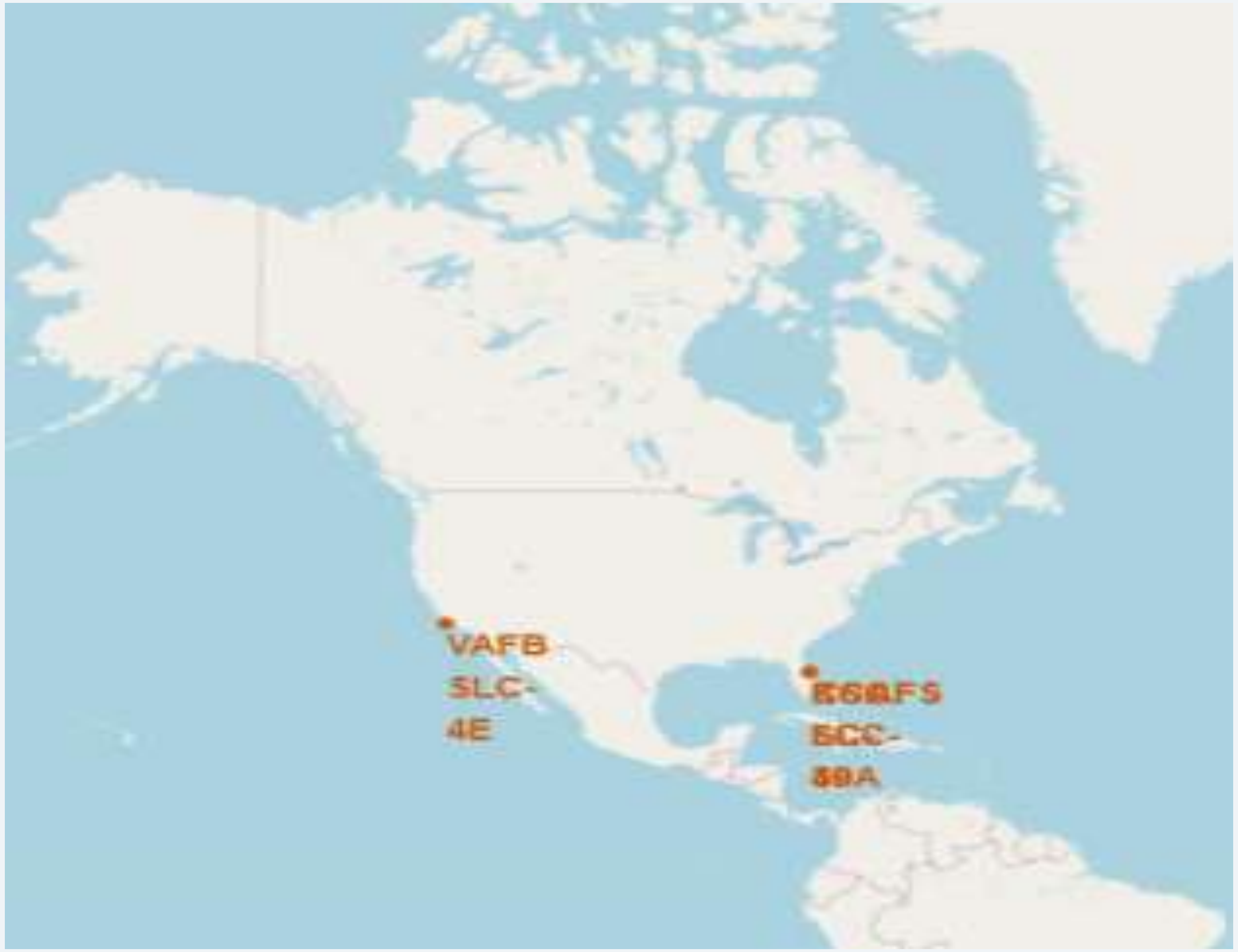
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

- 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.
- APPLIED the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

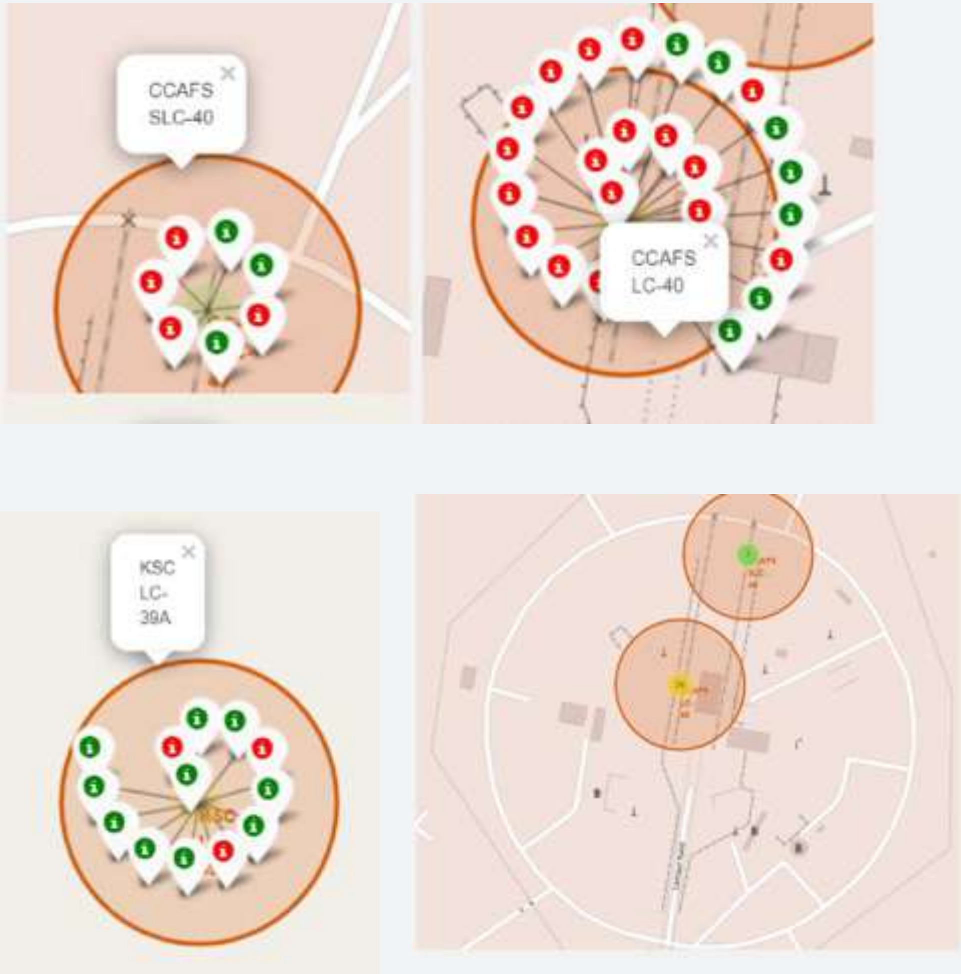
Section 3

Launch Sites Proximities Analysis

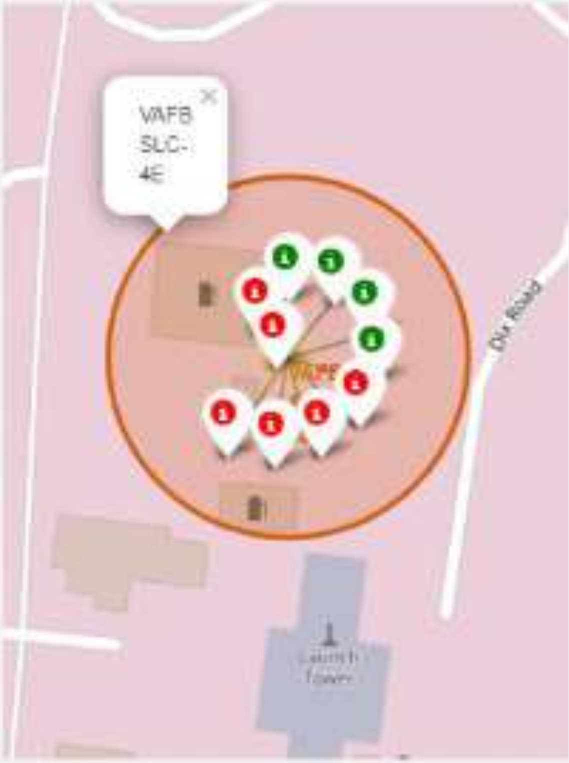
American launch site



LAUNCH SITE : California and Florida

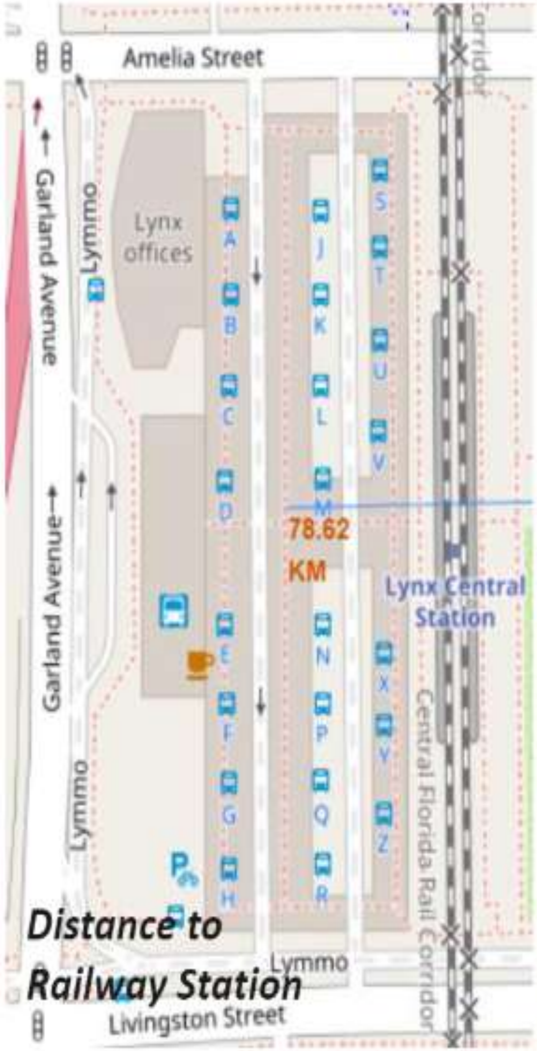


FLORIDA LAUNCH SITE

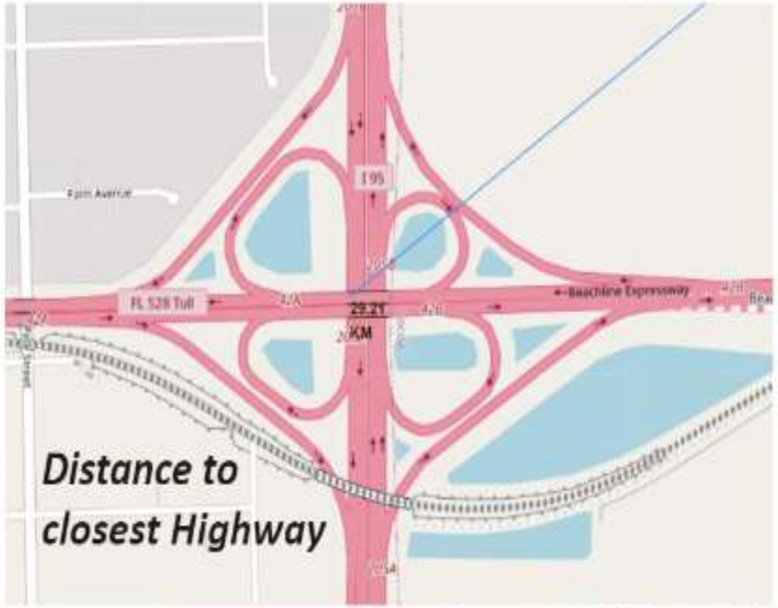


CALIFORNIA LAUNCH SITE

Launch site distance from landmark



Distance to Railway Station

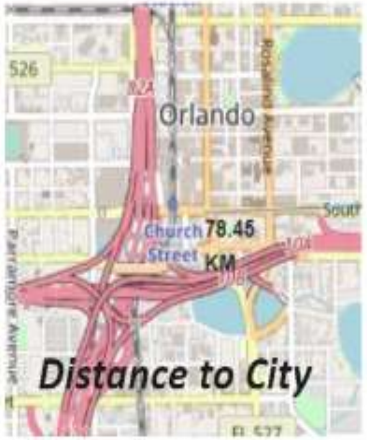


Distance to closest Highway

7
CAFS SLC-40
26
CAFS LC-40

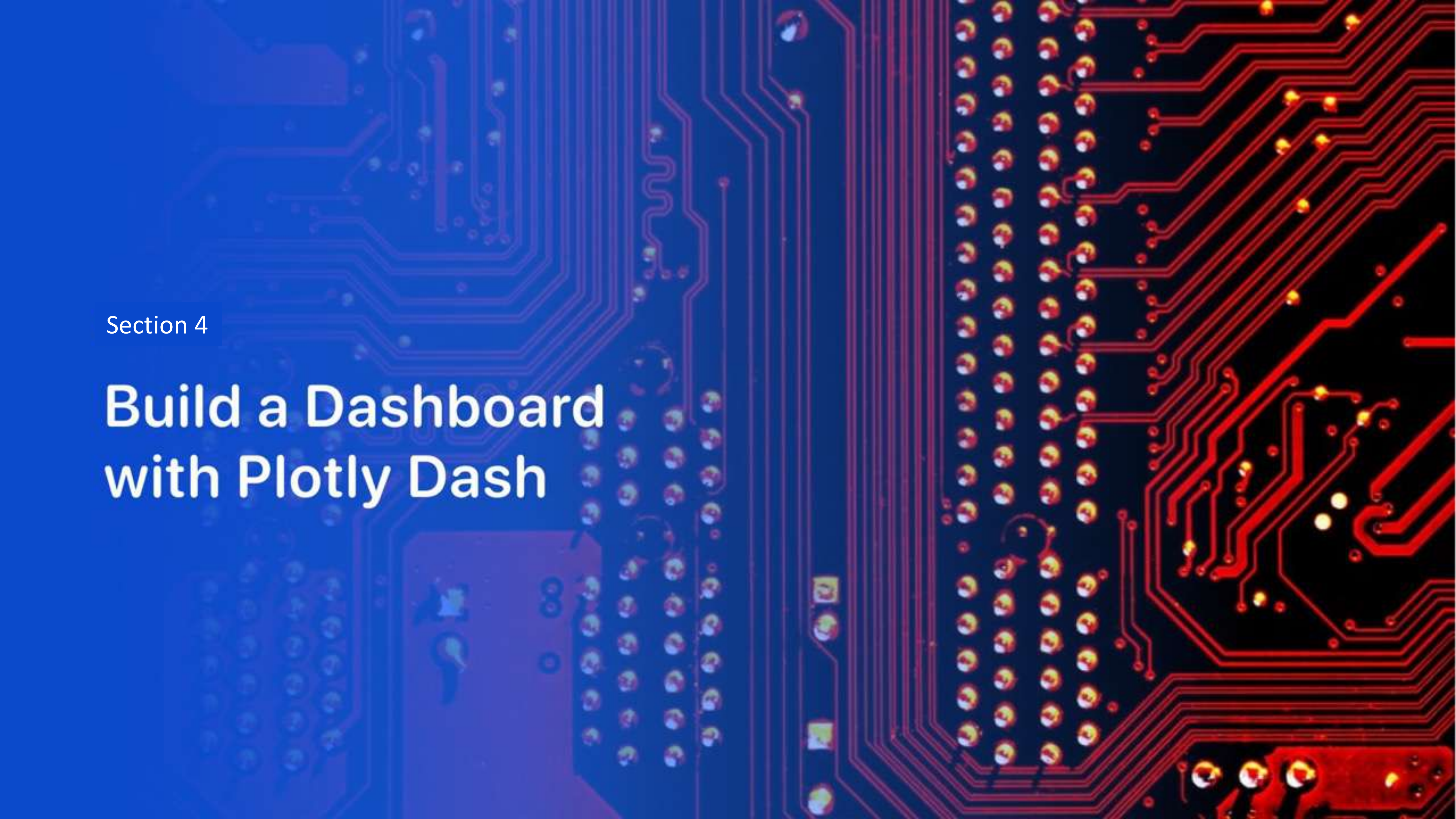
Distance to coast

Distance to Coastline



Distance to City

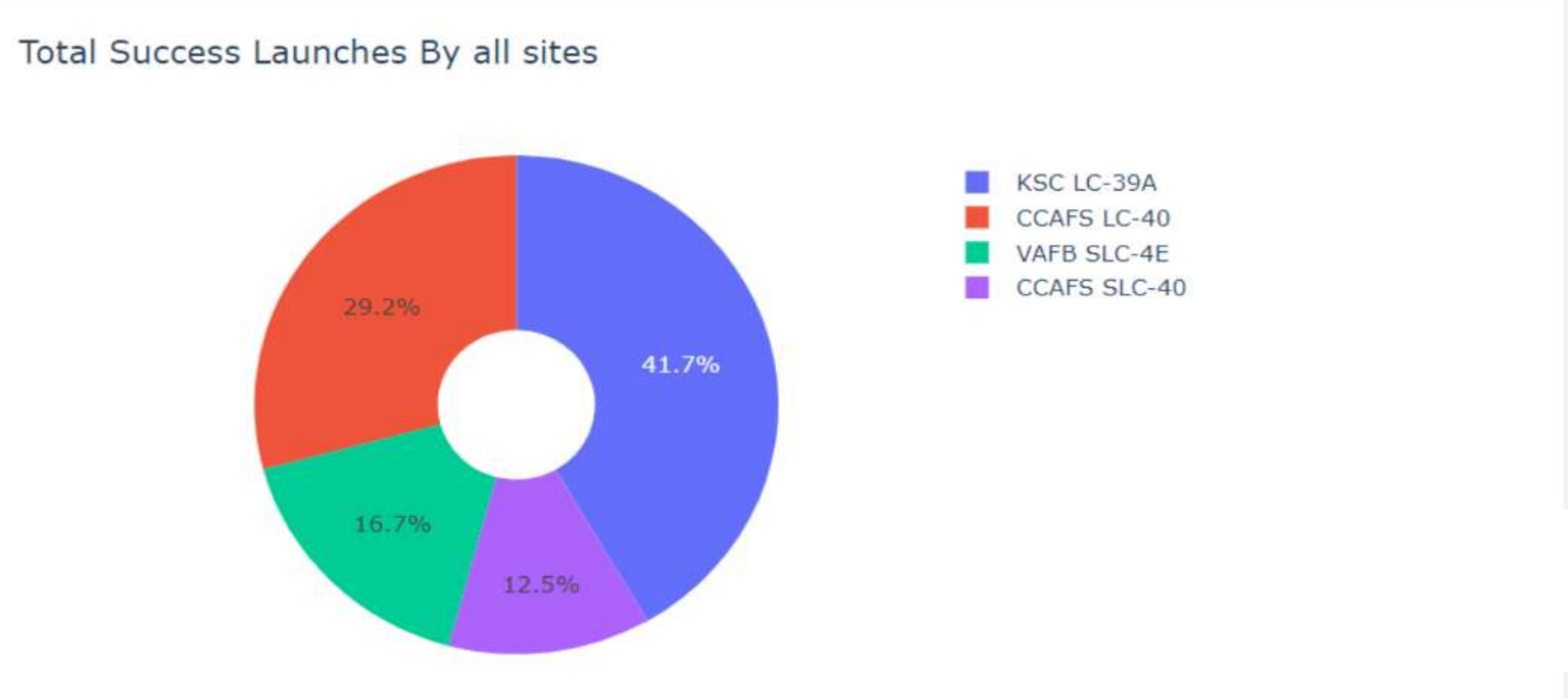
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



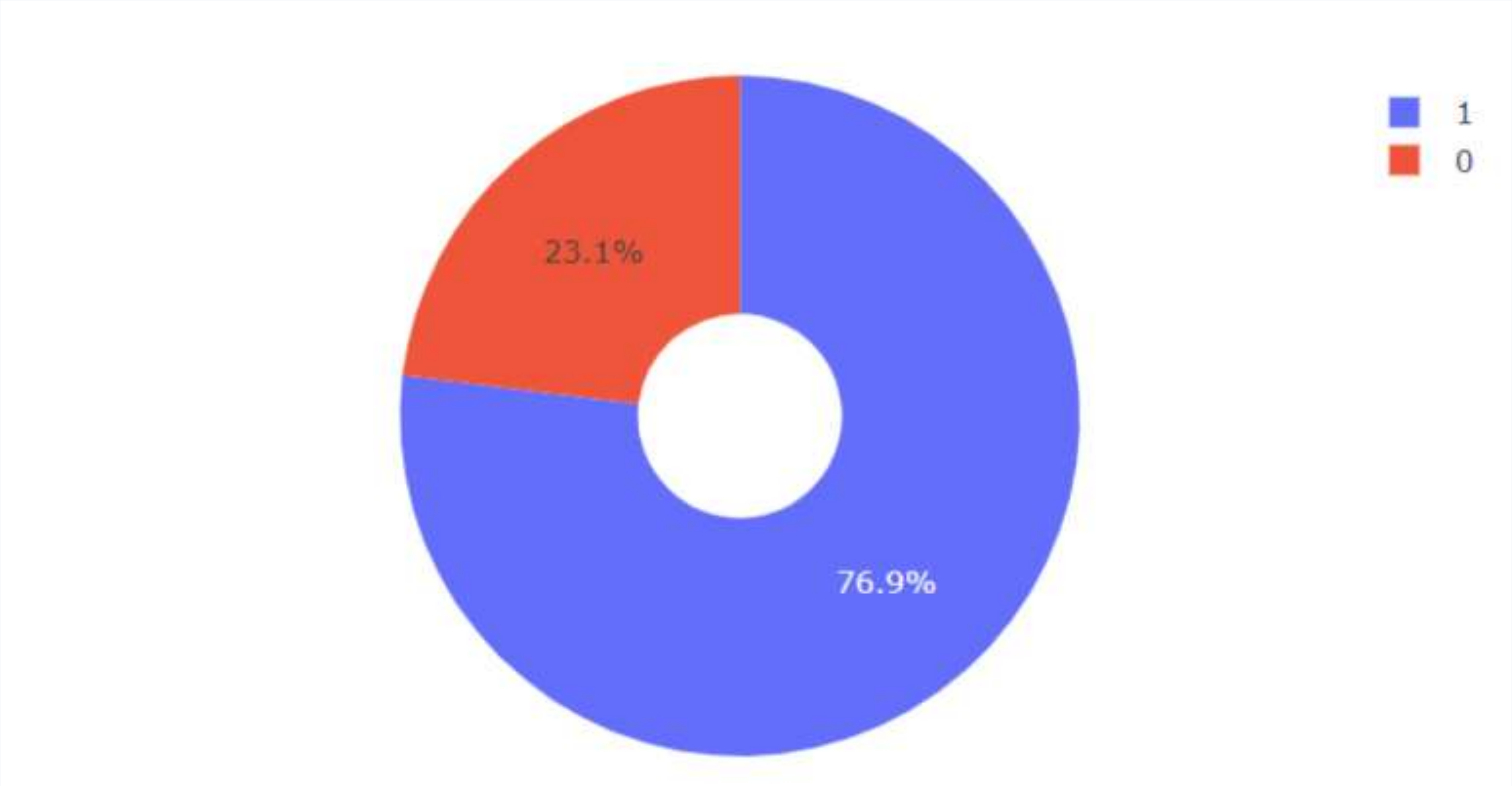
Section 4

Build a Dashboard with Plotly Dash

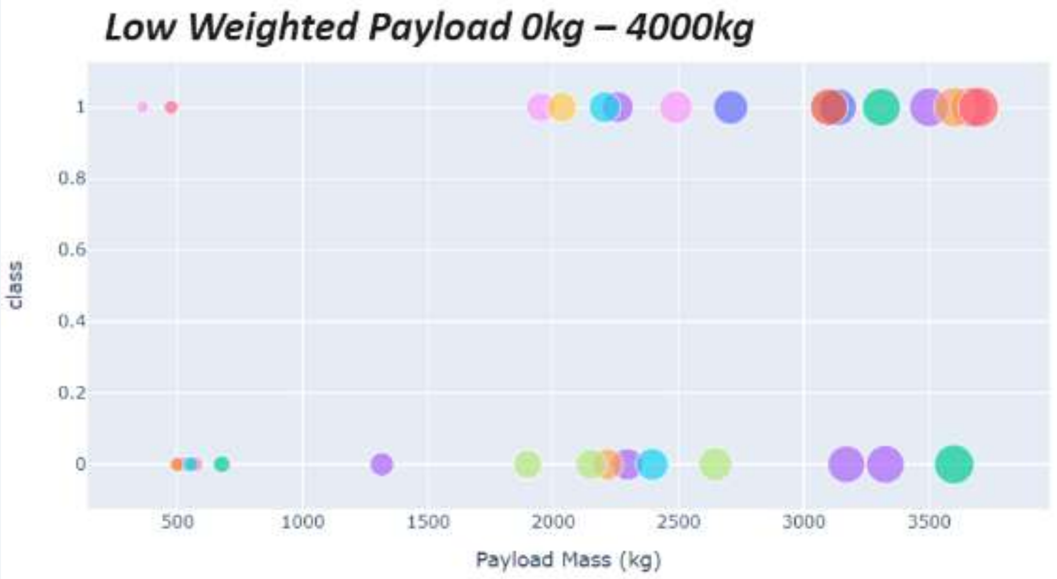
Success rate of landing by launch site

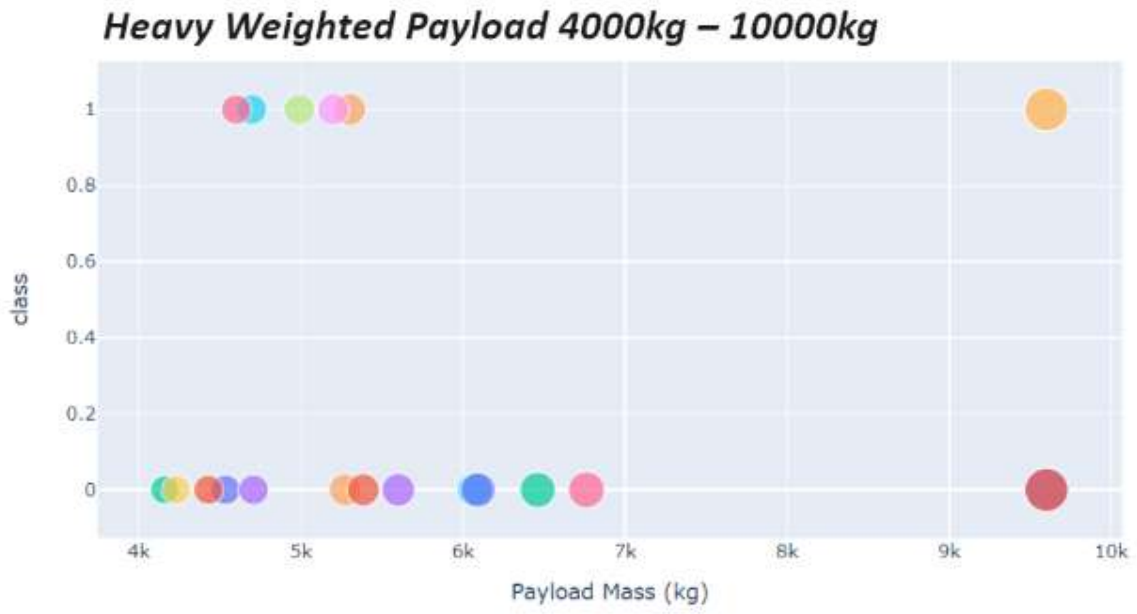


LAUNCH SITE WITH HIGHEST SUCCESS RATIO



PAYLOAD VS LAUNCH OUTCOME OVERALL

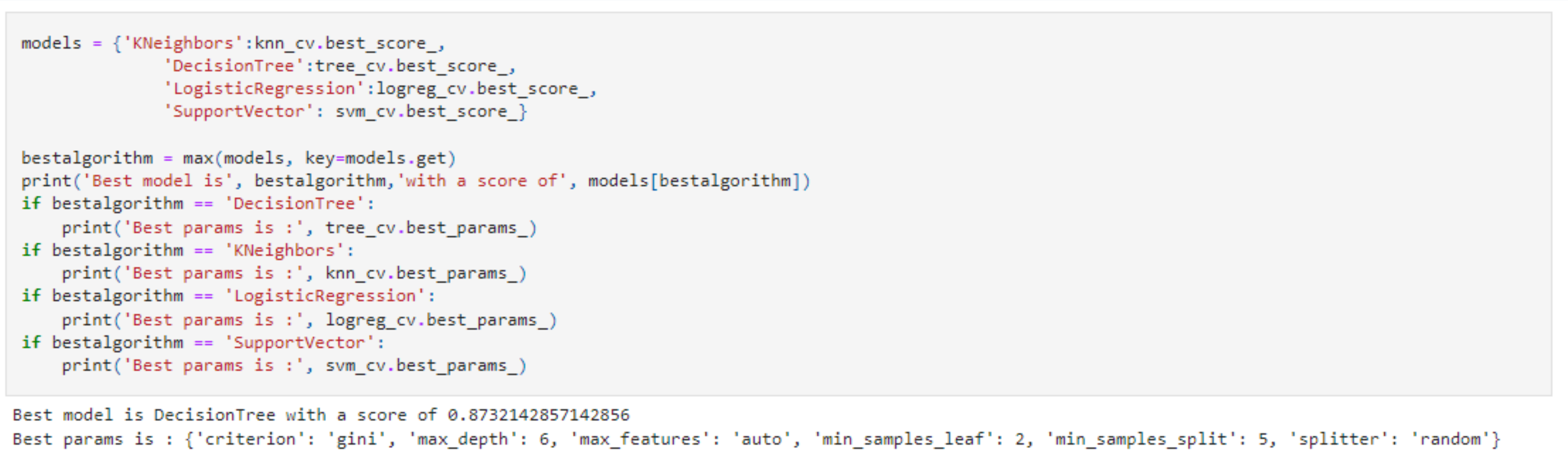

Low Weighted Payload 0kg – 4000kg


Heavy Weighted Payload 4000kg – 10000kg

Section 5

Predictive Analysis (Classification)

Classification Accuracy

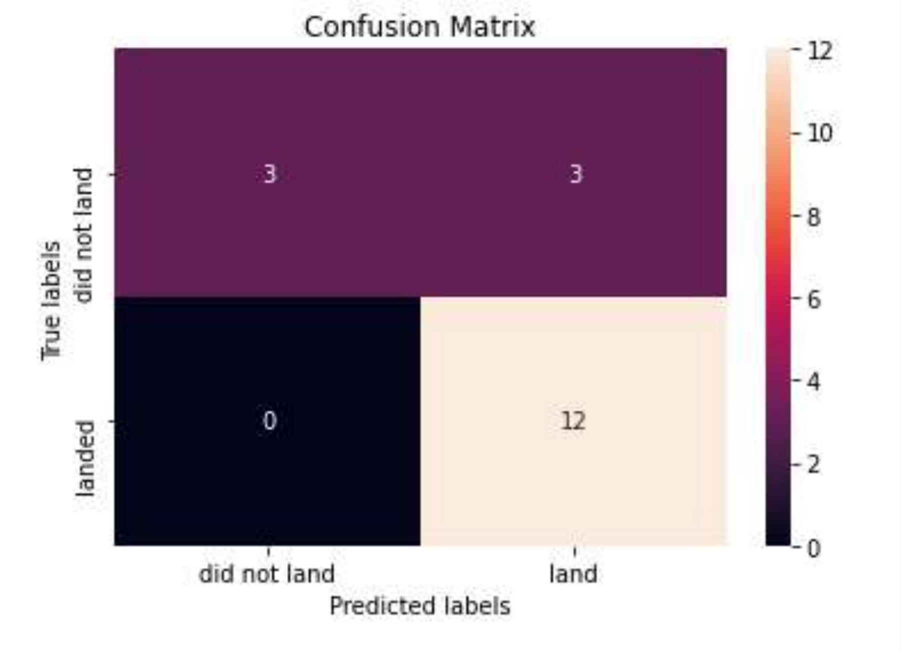


```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

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': 'random'}
```

The decision tree classifier is the model with the highest classification accuracy

Confusion Matrix



Confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes.

Conclusions

1. Launch success rate started to increase from 2013 till 2020.
2. Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
3. KSC LC-39A had the most successful launches of any sites.
4. The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!