IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Hossam Daoud

25-06-2024

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

Project background and context:

- SpaceX advertises Falcon 9 rocket launches on its website at a cost of 62 million dollars, whereas other providers charge upwards of 165 million dollars per launch. The significant cost savings SpaceX achieves is primarily due to the reuse of the first stage of the rocket. Therefore, accurately predicting whether the first stage will land successfully is crucial for estimating launch costs. This information can be valuable for alternate companies bidding against SpaceX for rocket launches. The goal of this project is to create a machine learning pipeline to predict the likelihood of the first stage landing successfully.

Problems you want to find answers:

- Factors Influencing Successful Landing:
  - Identify and analyze the key factors that determine whether the rocket's first stage will land successfully.
- Interaction Among Features:
  - Examine the interactions among various features to understand their combined effect on the success rate of a landing.
- Operating Conditions for Successful Landing:
  - Determine the optimal operating conditions that need to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - Data from SpaceX was obtained from 2 sources:
    - SpaceXAPI (https://api.spacexdata.com/v4/rockets/)
    - WebScraping (https://en.wikipedia.org/wiki/List_of_Falcon/_9/_and_Falcon_Heavy_launches)
- Perform data wrangling

  Data Cleaning - Feature Engineering - Data Transformation - Data Enrichment - Data Validation

Perform exploratory data analysis (EDA) using visualization and SQL

Perform interactive visual analytics using Folium and Plotly Dash

Perform predictive analysis using classification models

Steps for Data Preparation and Model Evaluation:

1. Normalization
2. Splitting the Data
3. Model Evaluation

Data Collection

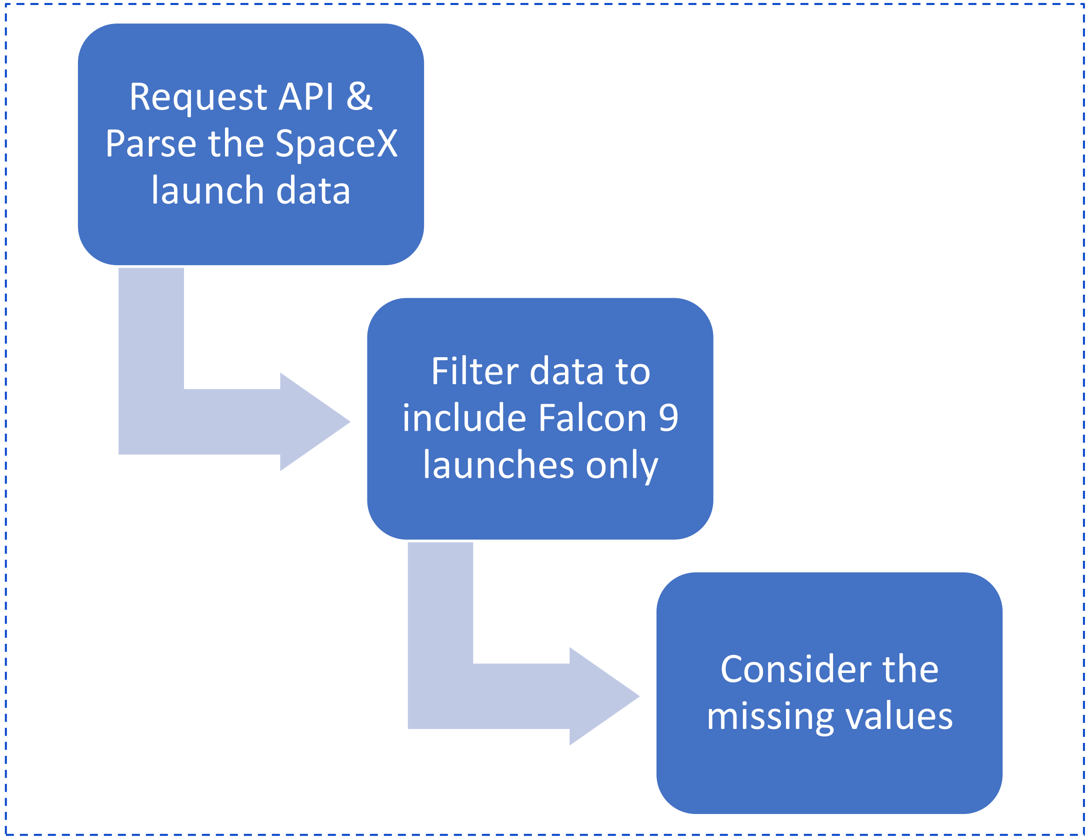
- Data Collection
  - Datasets were collected from two primary sources: the SpaceX API and Wikipedia.
  - SpaceX API:
    - The SpaceX API (https://api.spacexdata.com/v4/rockets/) was used to gather detailed information about the rockets, including technical specifications, launch history, and landing outcomes. The API provides a structured and comprehensive dataset that is regularly updated by SpaceX.
  - Wikipedia:
    - Additional data was collected from Wikipedia (https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches) using web scraping techniques. This included information on launch dates, mission outcomes, and other relevant historical data that may not be covered by the SpaceX API.

Data Collection – SpaceX API

Present your data collection with SpaceX REST calls using key phrases and flowcharts

Add the GitHub URL of the completed SpaceX API calls notebook as an external reference and peer-review purpose

https://github.com/HossamDaoud83/applied-data-science-capstone



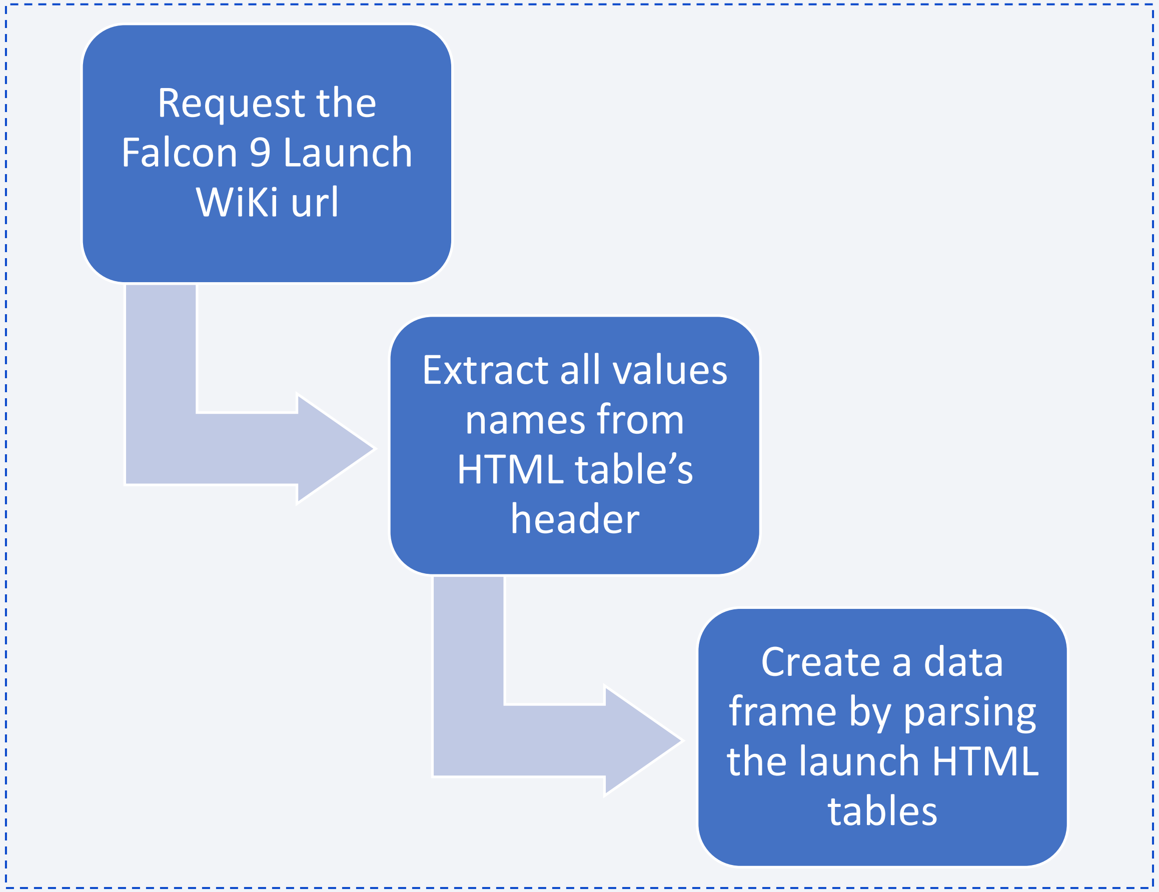
Request API & Parse the SpaceX launch data

Filter data to include Falcon 9 launches only

Consider the missing values

Data Collection - Scraping

- Data on SpaceX launches is also available from Wikipedia.Data is downloaded from Wikipedia following a specific flowchart and then saved for further analysis.
- Add the GitHub URL of the completed web scraping notebook, https://github.com/HossamDaoud83/applied-data-science-capstone/blob/main/jupyter-labs-webscraping.ipynb



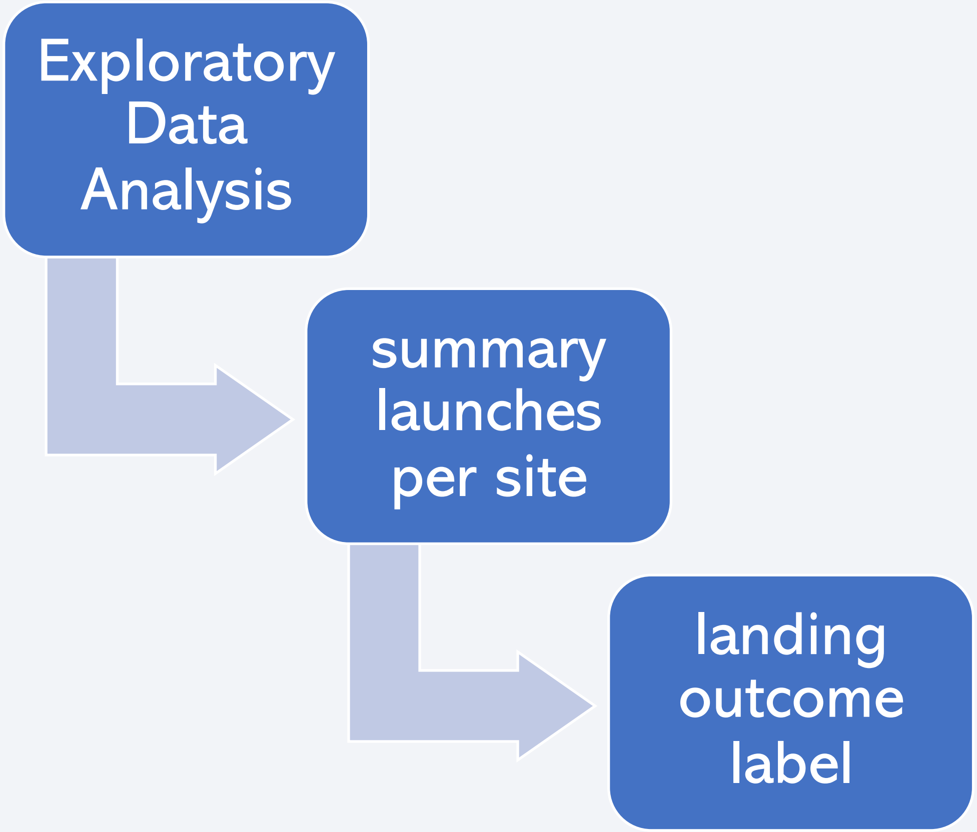
Request the Falcon 9 Launch WiKi url

Extract all values names from HTML table's header

Create a data frame by parsing the launch HTML tables

Data Wrangling

- Initially, some exploratory data analysis (EDA) was performed on the dataset.
- The summary launches per site, occurrences of each orbit, and occurrences of mission outcomes per orbit type were then determined.
- Finally, the landing outcome label was derived from the Outcome column.
- Add the GitHub URL of your completed data wrangling related notebooks, https://github.com/HossamDaoud83/applied-data-science-capstone/blob/main/Data_Wrangling.ipynb

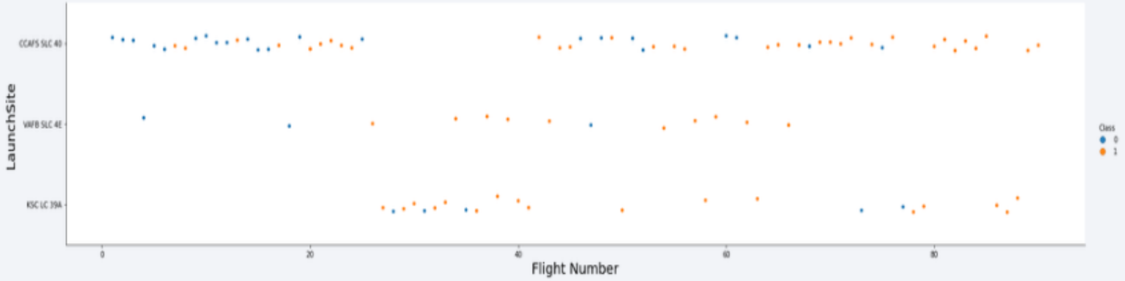


EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
  - Names of the distinct launch locations in the space project;
  - Top 5 launch locations whose names start with the letters 'CCA';
  - The total payload mass carried by boosters flown by NASA (CRS);
  - The average payload mass carried by rocket type F9 v1.1
  - The date when the first successful landing on the ground pad occurred;
  - Names of rockets that have succeeded on drone ships and have payload masses ranging from 4000 to 6000 kg;
  - Total number of successful and failed mission results.
  - Names of the booster models that carried the highest payload mass;
  - Failed landing out comes in droneships, booster variants, and launch site names for the year 2015. and
  - Rank of the count of landing outcomes (such as failure (droneship) or success (ground pad).
- Add the GitHub URL https://github.com/HossamDaoud83/applied-data-science-capstone/blob/main/exploratory_data_analysis_EDA.ipynb

EDA with SQL

- To investigate data, scatterplots and bar plots were employed to visualize the relationship between two pairs of attributes.
- Payload Mass X Flight Number, Launch Site X Flight Number, Launch Site X Payload Mass, Orbit, and Flight Number.



- Add the GitHub URL of your completed EDA with SQL notebook, https://github.com/HossamDaoud83/applied-data-science-capstone/blob/6892547fdf078ef692eba4ef2c9016b2b7128324/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Folium Maps included markers, circles, lines, and marker clusters.
- Markers represent points, circles highlight areas around specific coordinates (e.g., NASA Johnson Space Centre), marker clusters represent groups of events (e.g., launches at a launch site), and lines show distances between coordinates.
- Add the GitHub URL of your completed interactive map with Folium map, https://github.com/HossamDaoud83/applied-data-science-capstone/blob/main/SpaceX_FoliumLab.ipynb

Build a Dashboard with Plotly Dash

- I used Plotly dash to create an interactive dashboard.
- I created pie charts that displayed each site's total launches.
- With each booster version, i created a scatter graph that displayed the link between the outcome and payload mass.
- Add the GitHub URL of your completed Plotly Dash lab, https://github.com/HossamDaoud83/applied-data-science-capstone/blob/main/MainApp.py

Predictive Analysis (Classification)

Summary of Model Development Process

1. Building the Model

2. Evaluating the Model

3. Improving the Model

4. Finding the Best Performing Model

Model Development Process Using Key Phrases and Flowchart

Key Phrases:

- Data Collection
- Data Preprocessing
- Feature Selection
- Model Selection
- Performance Metrics
- Cross-Validation
- Hyperparameter Tuning
- Feature Engineering
- Model Comparison
- Best Model Selection

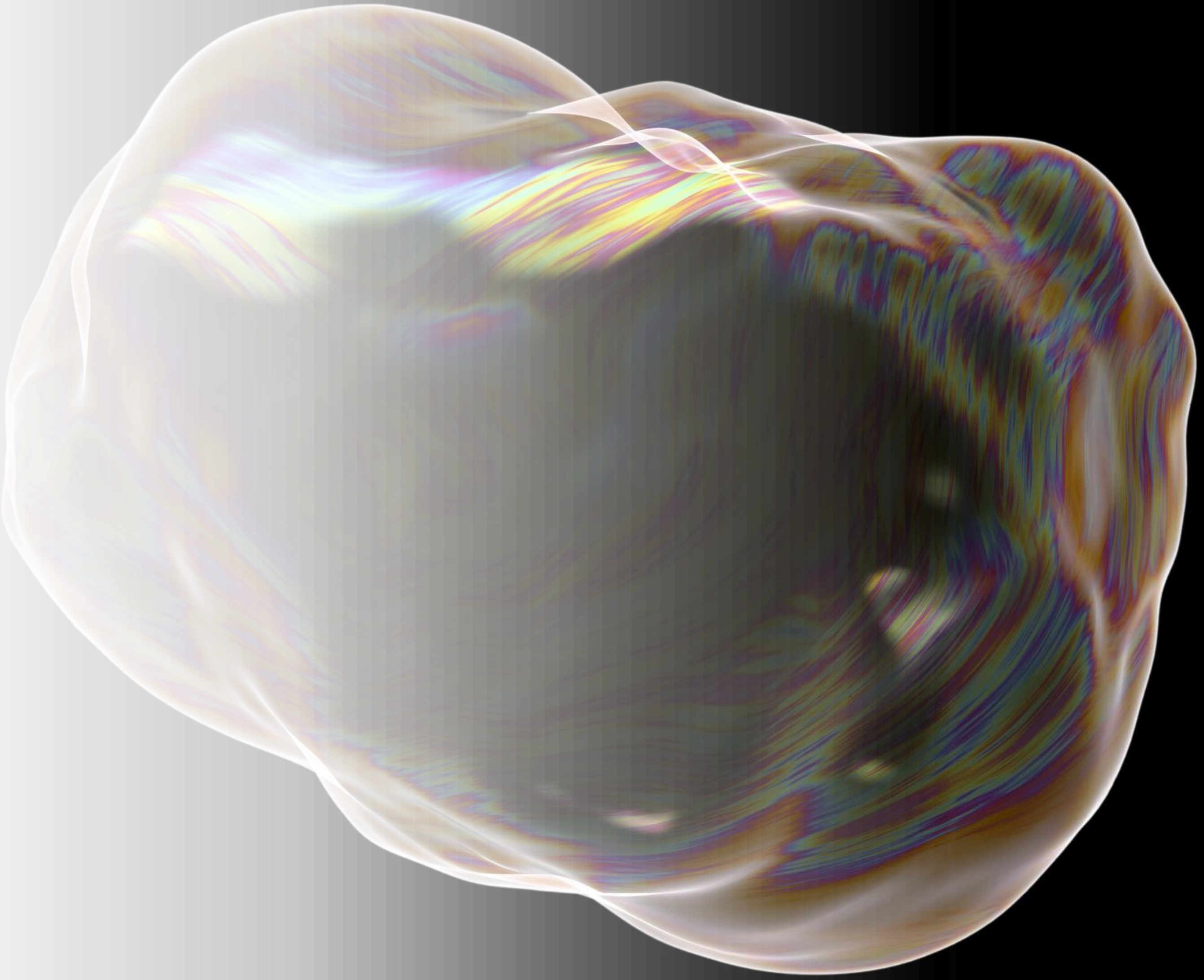
- Add the GitHub URL of your completed predictive analysis lab, https://github.com/HossamDaoud83/applied-data-science-capstone/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

- Results of the exploratory data analysis: Space X launches from four distinct locations; NASA and Space X conducted the initial launches;
- The F9 v1.1 booster's average payload weighs 2,928 kg;
- Five years after the initial launch, in 2015, the first successful landing result occurred;
- Numerous Falcon 9 booster variants with payloads over average were successful in landing on drone ships;
- Nearly all mission outcomes were accomplished successfully;
- 2015 saw the failure of two booster versions, F9 v1.1 B1012 and F9 v1.1 B1015, to land on drone ships;
- With the passing of the years, the number of landing outcomes improved.

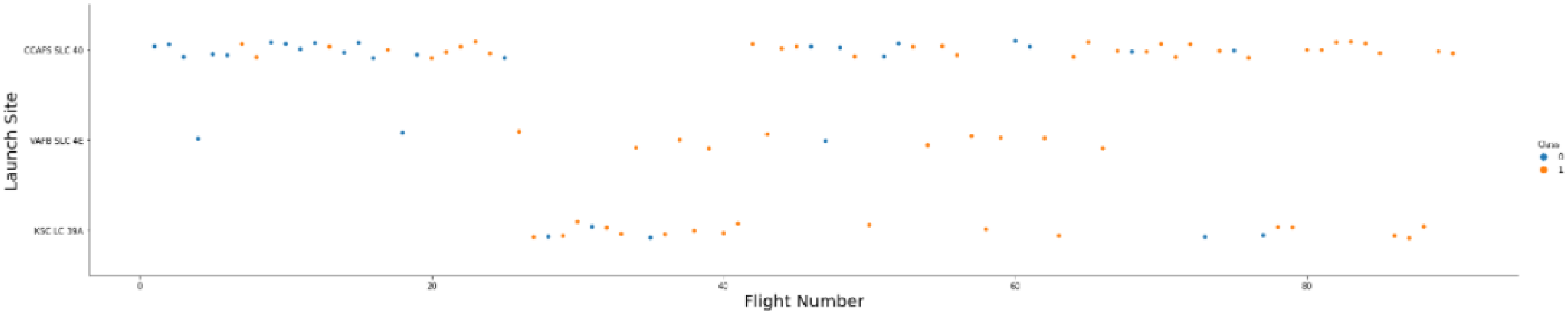
Section 2

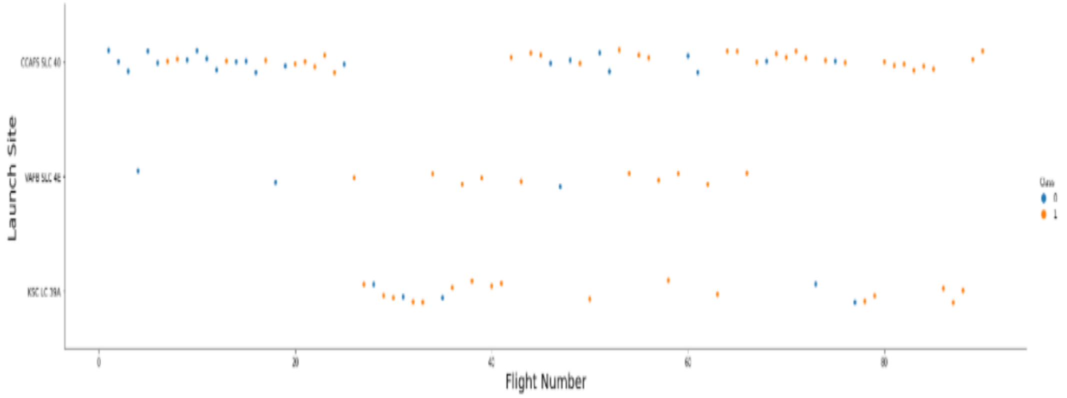
Insight Drawn from EDA



Flight Number vs. Launch Site

- We deduced from the plot that a launch site's success rate increases with the number of flights conducted there.



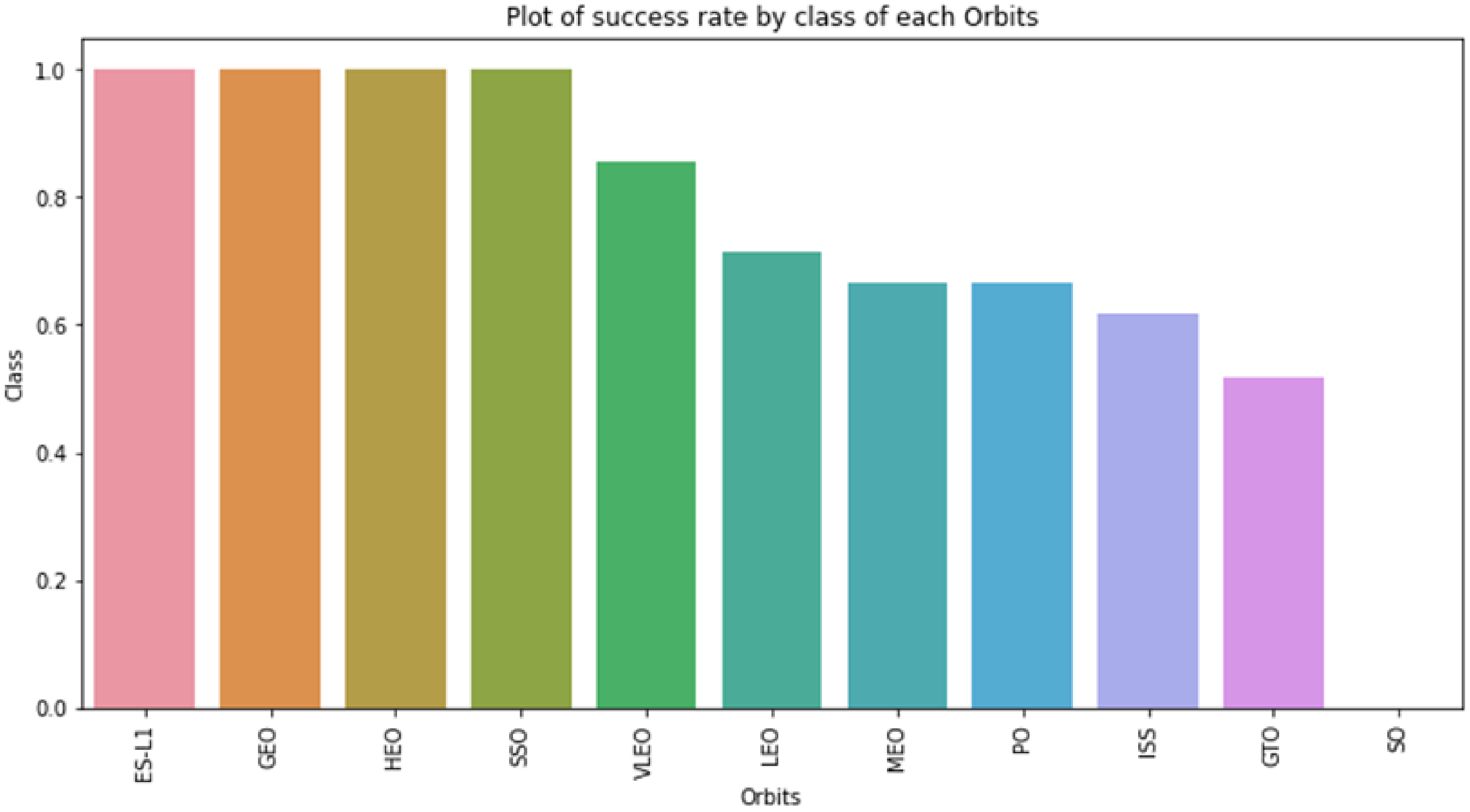


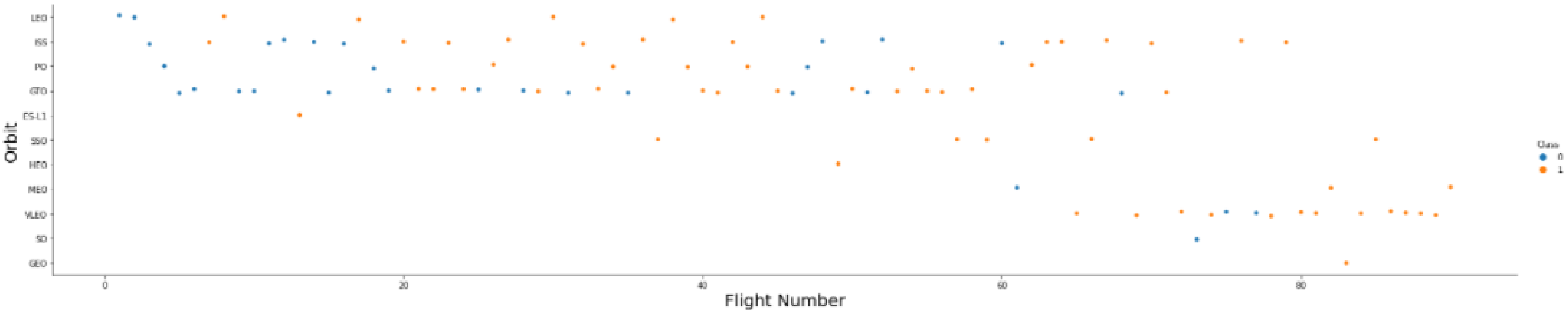
The greater the payload mass for launch site CCAFS SLC 40 the higher the success rate for the rocket.

Payload vs. Launch Site

Success Rate vs. Orbit Type

The plot indicates that the highest success rates were attained by ES-L1, GEO, HEO, SSO, and VLEO.



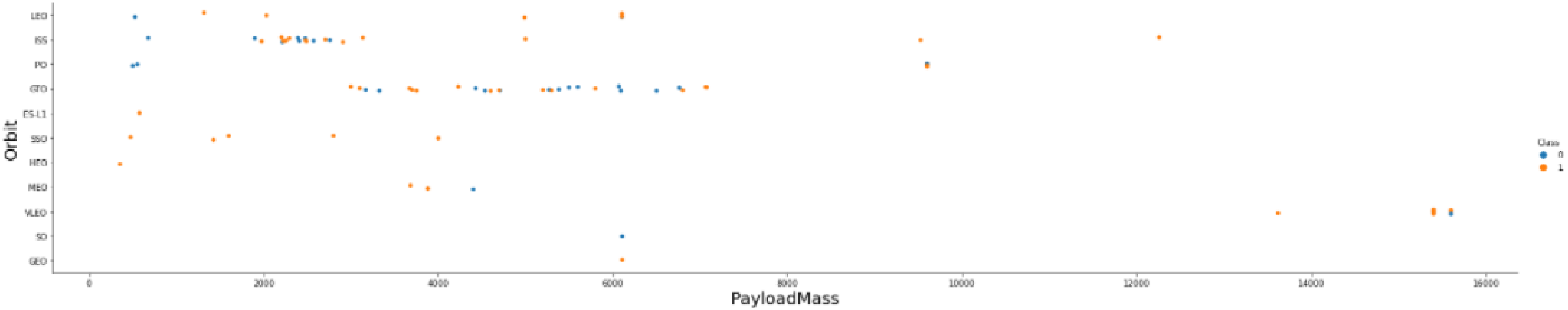


Flight Number vs. Orbit Type

- The Flight Number vs. Orbit type figure is presented below. We see that while there is no correlation between flight number and orbit in the GTO orbit, success in the LEO orbit is correlated with the number of flights.

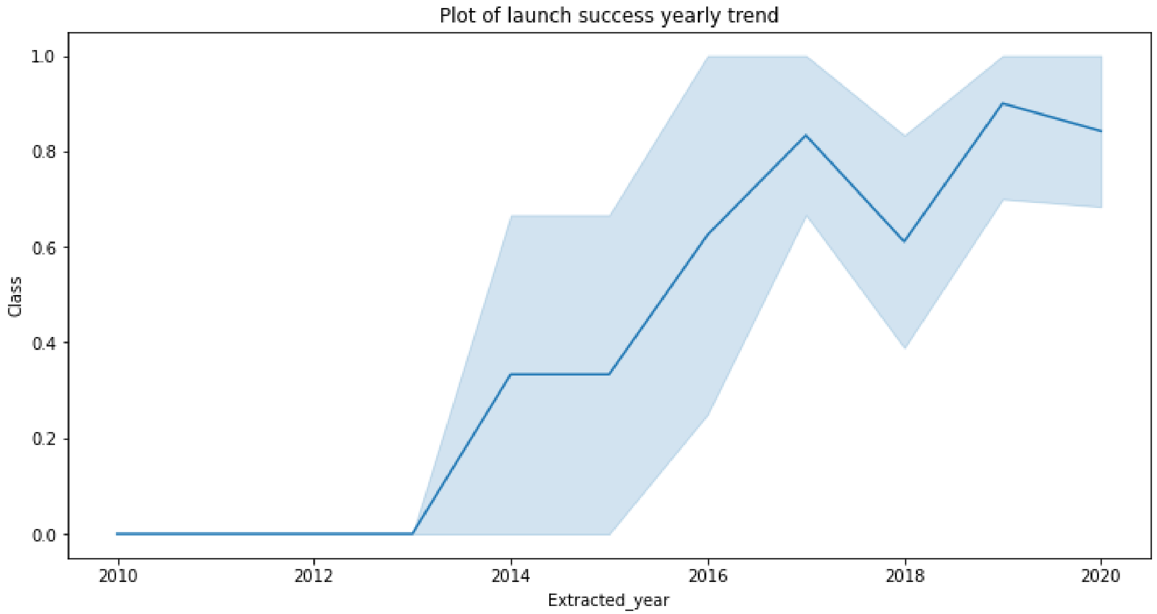
Payload vs. Orbit Type

We can see that more successful landings with large cargoes occur in PO, LEO, and ISS orbits.



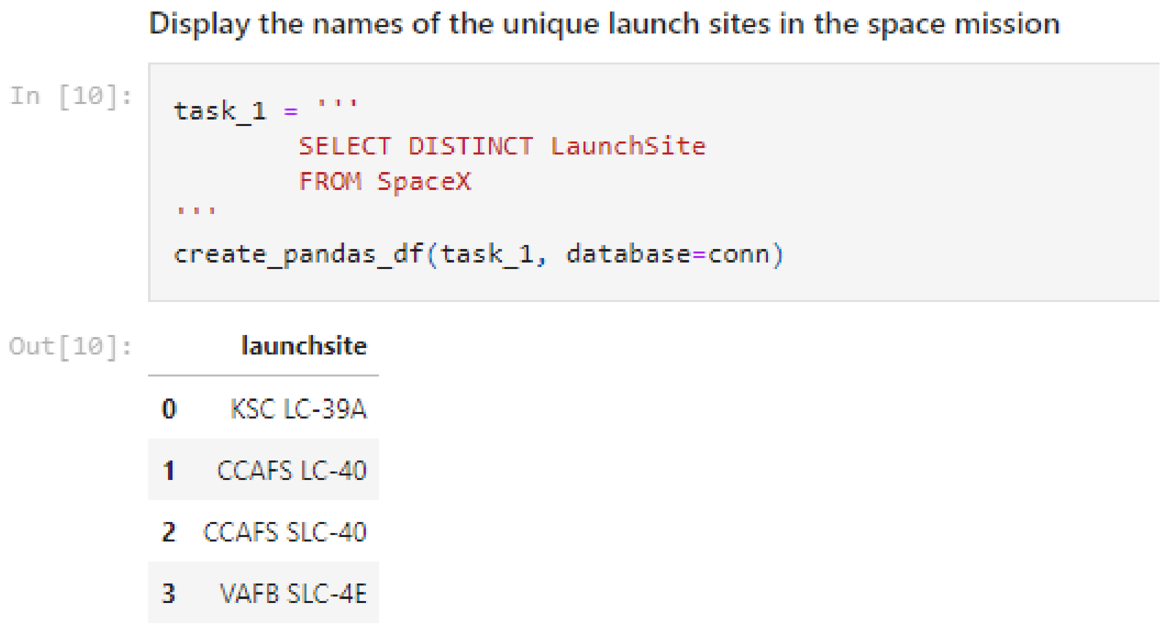
We can see from the plot that, starting in 2013, the success rate continued to rise until 2020.

Launch Success Yearly Trend



To display only distinct launch sites from the SpaceX data, we employed the keyword DISTINCT.

All Launch Site Names

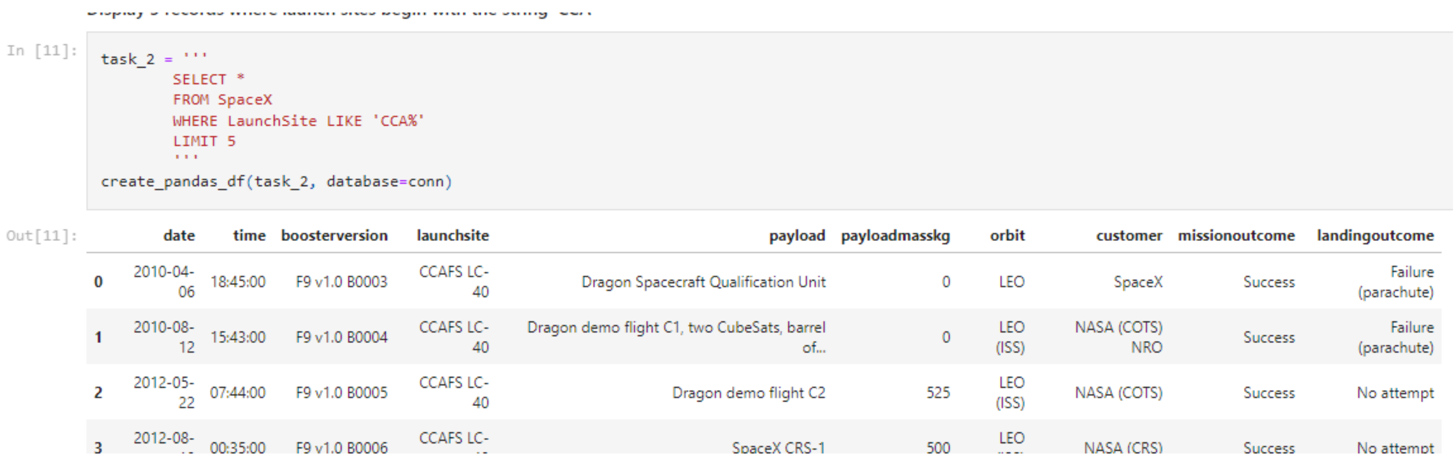


Display the names of the unique launch sites in the space mission

```
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
```

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)
```

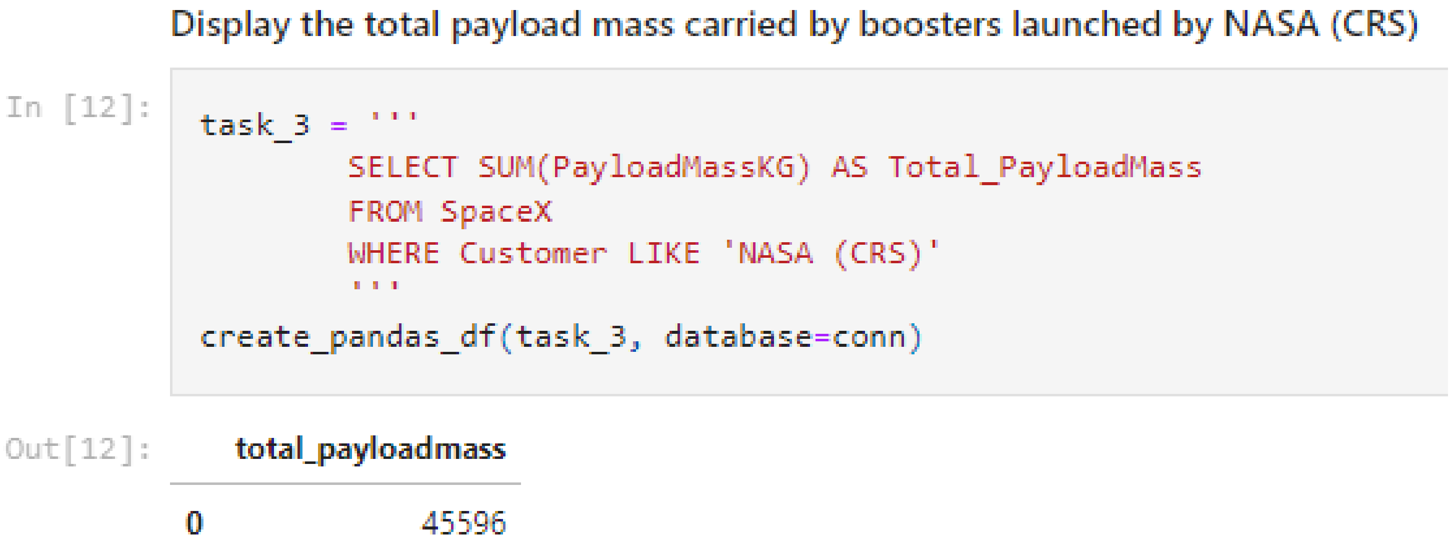
Out[11]:

	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-	00:35:00	F9 v1.0 B0006	CCAFS LC-	SpaceX CRS-1	500	LEO	NASA (CRS)	Success	No attempt

- Using the aforementioned query, we were able to show 5 records whose launch sites start with {CCA}.

Total Payload Mass

- Using the following query, we were able to compute the total payload carried by NASA's boosters to be 45596.



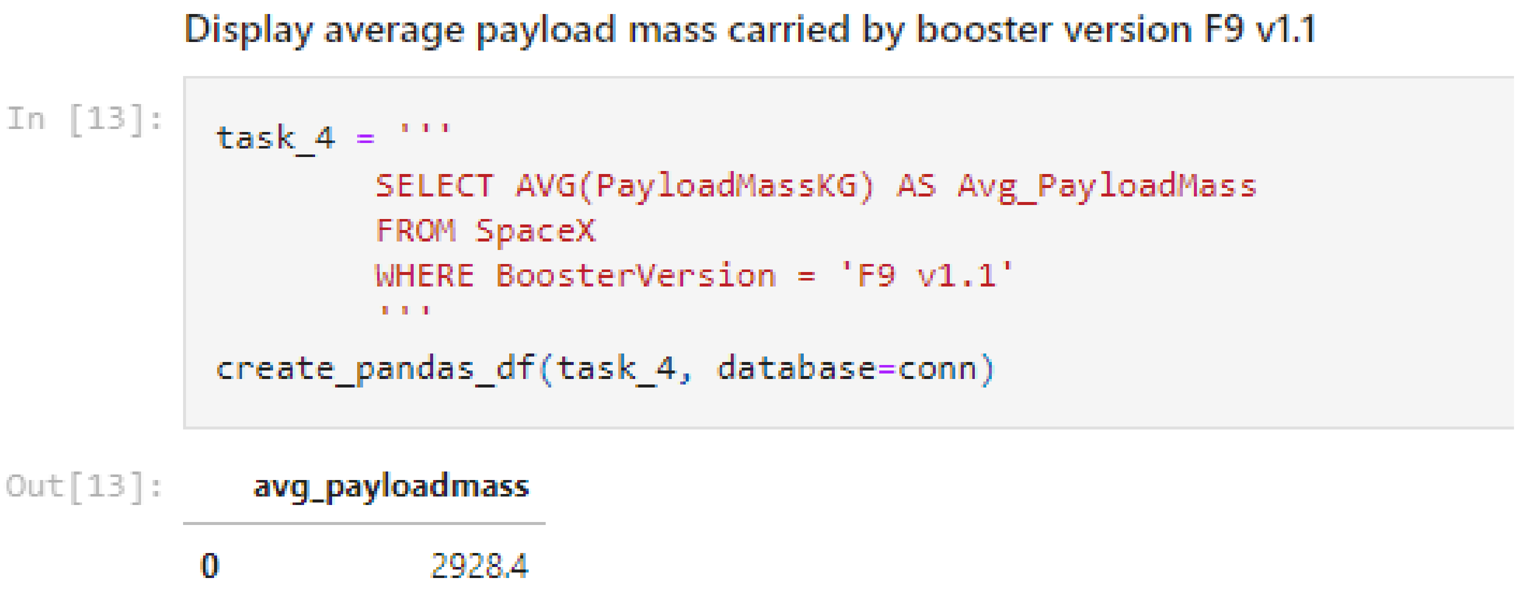
Display the total payload mass carried by boosters launched by NASA (CRS)

```
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
```

Average Payload Mass by F9 v1.1

We determined that the booster version F9 v1.1's average payload mass was 2928.4.



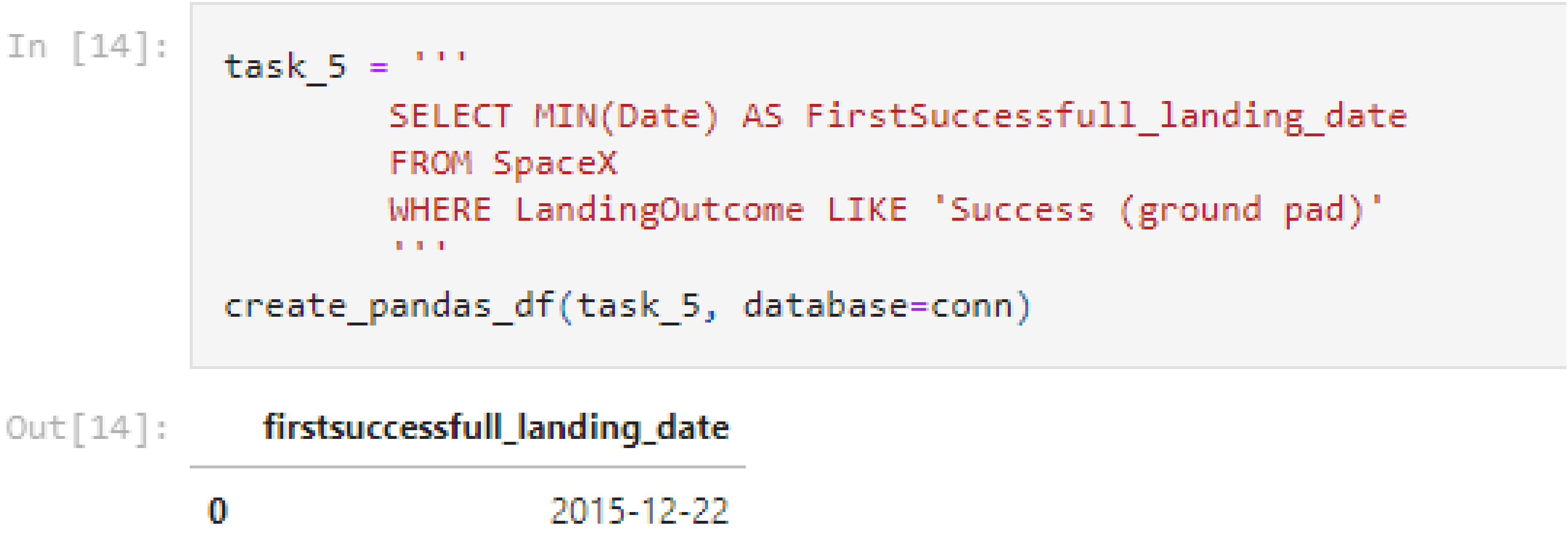
Display average payload mass carried by booster version F9 v1.1

```
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
```

First Successful Ground Landing Date

We noted that December 22, 2015, was the day of the first successful landing on a ground pad.

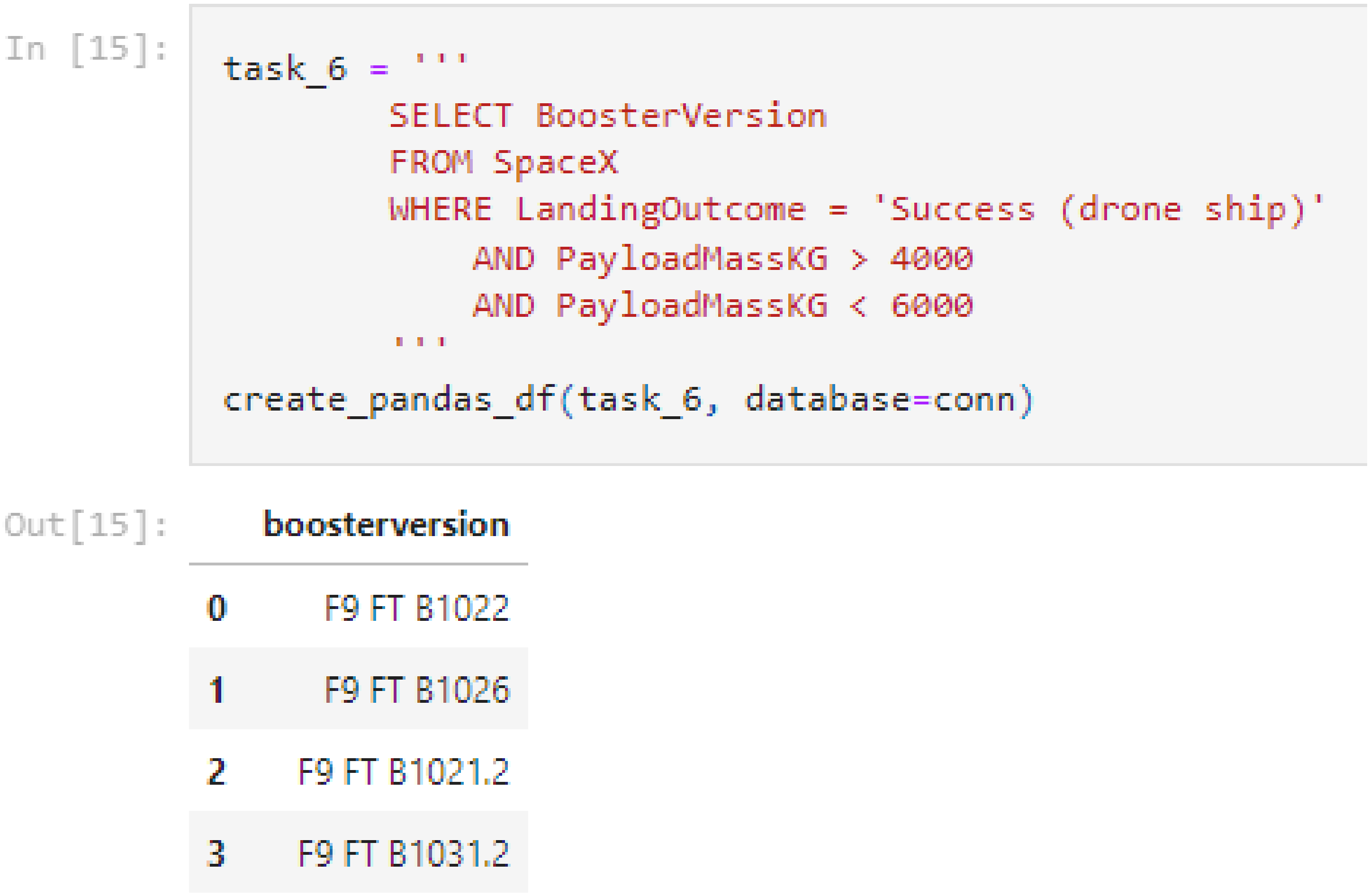


```
In [14]:  task_5 = '''
                  SELECT MIN(Date) AS FirstSuccessfull_landing_date
                  FROM SpaceX
                  WHERE LandingOutcome LIKE 'Success (ground pad)'
                  '''
          create_pandas_df(task_5, database=conn)
```

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

Successful Drone Ship Landing with Payload between 4000 and 6000

- In order to identify successful landings with payload masses larger than 4000 but less than 6000, we employed the AND condition after using the WHERE clause to filter for boosters that have successfully landed on drone ships.



In [15]:

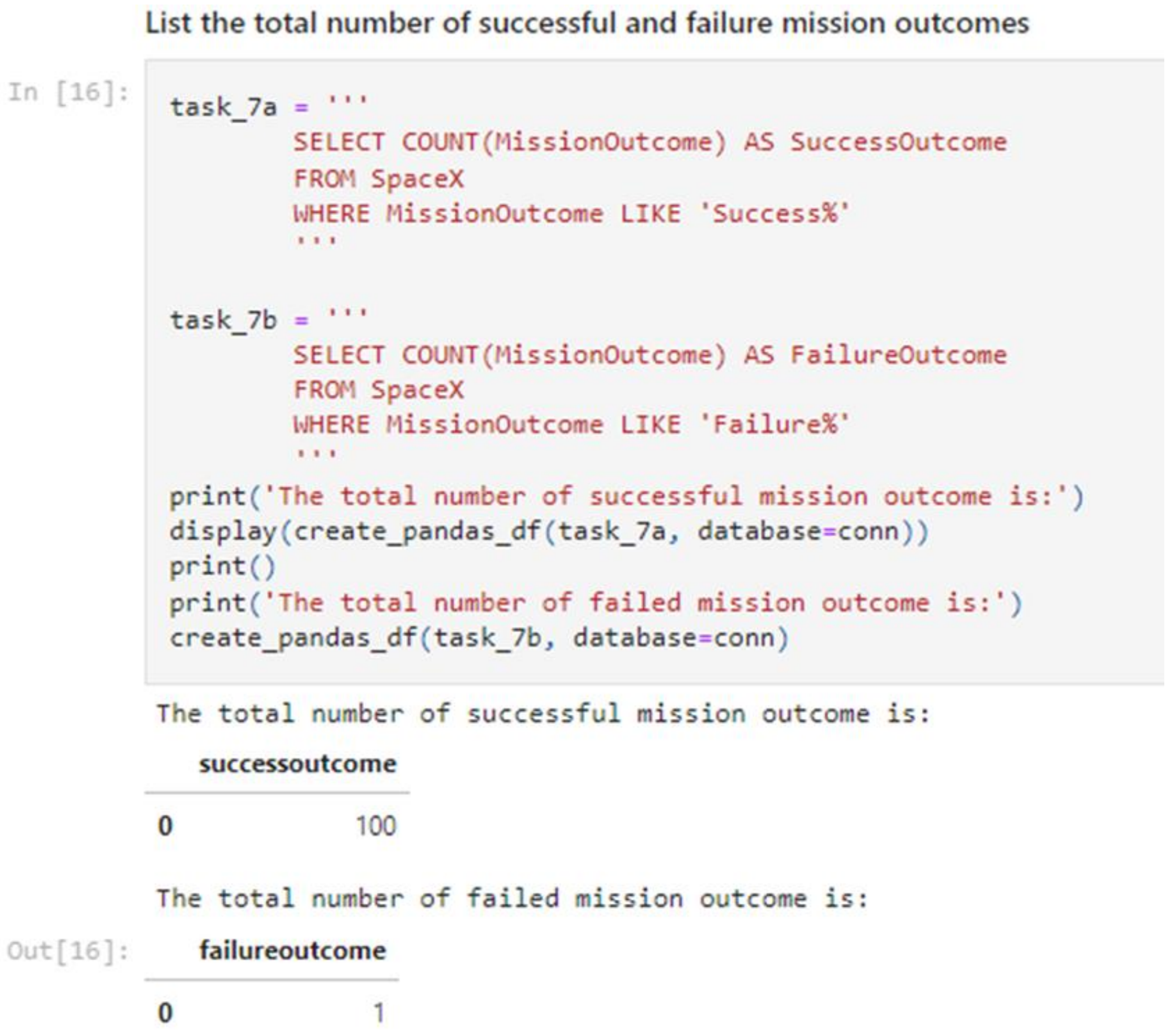
```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

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

To filter for WHERE MissionOutcome was successful or unsuccessful, we utilised wildcards like %.



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

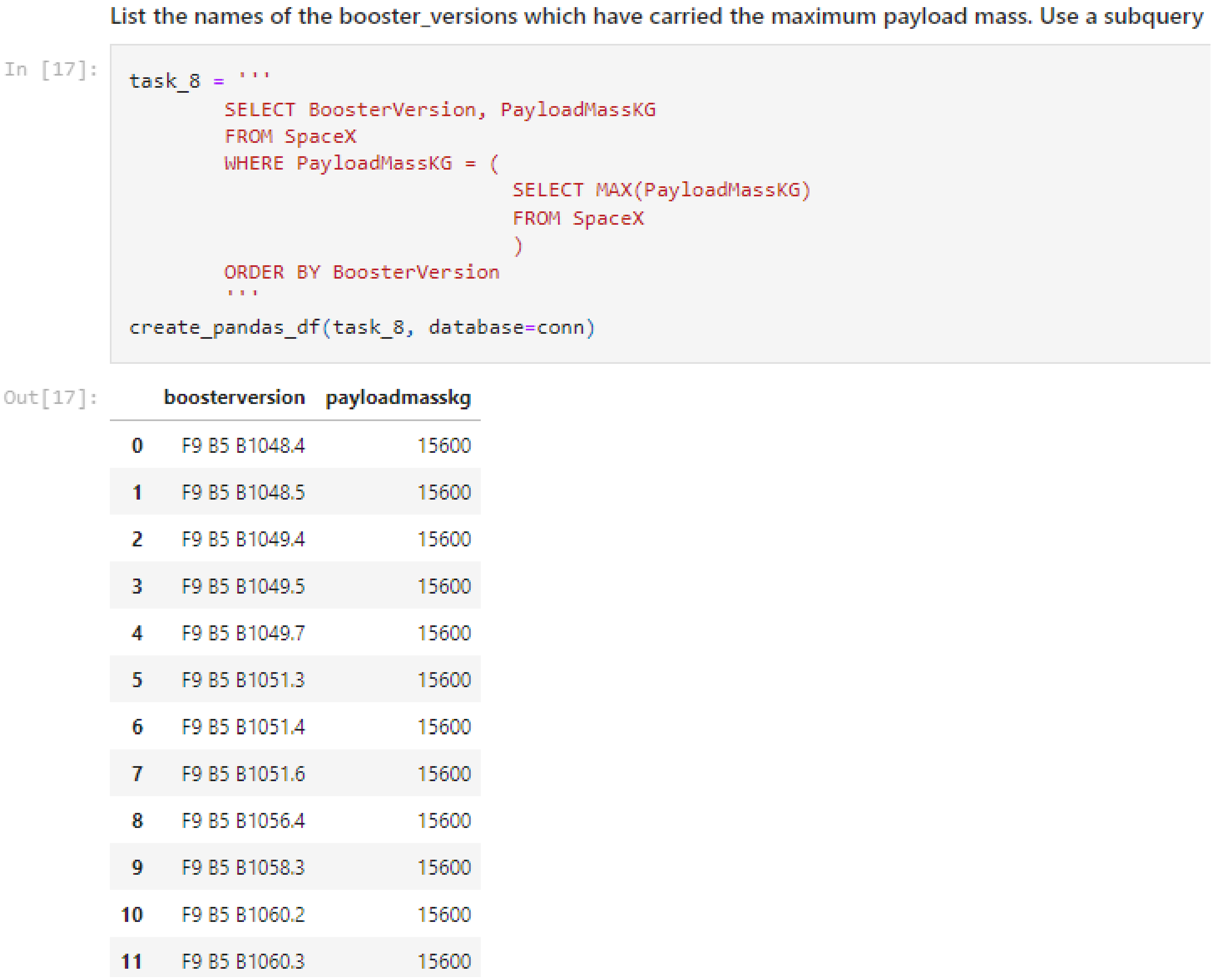
```
The total number of failed mission outcome is:
```

Out[16]:

	failureoutcome
0	1

Boosters Carried Maximum Payload

- Using a subquery in the WHERE clause and the MAX() method, we were able to identify the booster that had transported the maximum payload.



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

In [17]:

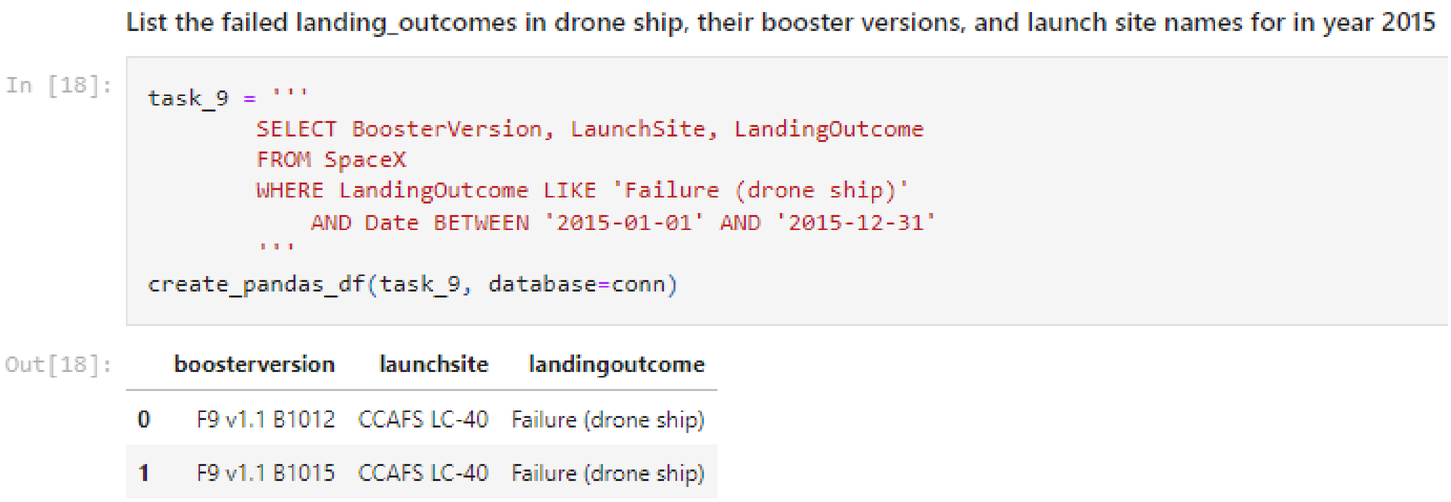
```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8,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

2015 Launch Records

- In order to filter for failure landing outcomes in drone ship, their booster versions, and launch location names for the year 2015, we combined the WHERE clause, LIKE, AND, and BETWEEN conditions.



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

In [18]:

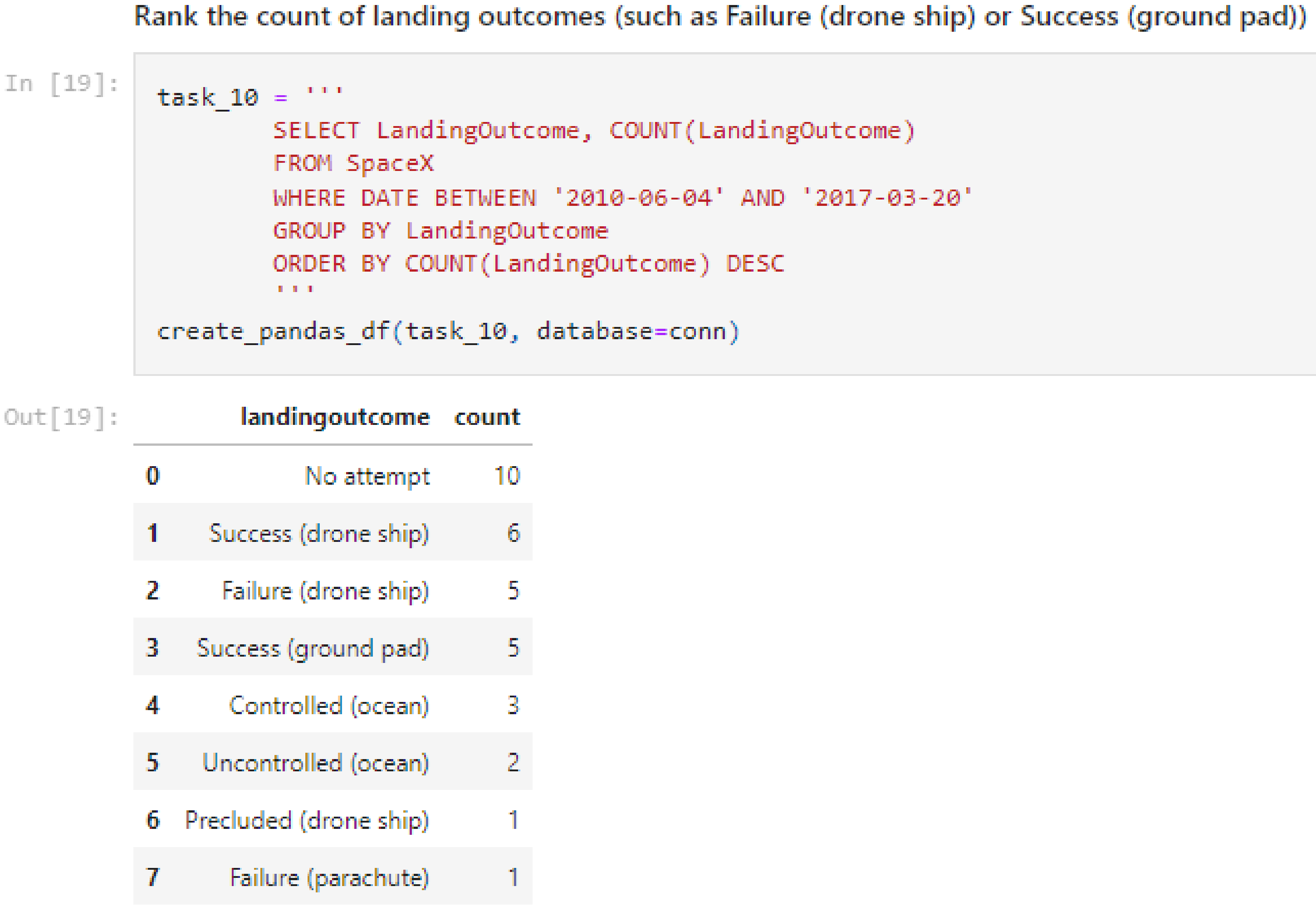
```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

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

- Using the WHERE clause, we filtered the data for landing outcomes BETWEEN 2010-06-04 and 2010-03-20. We also picked the landing outcomes and the COUNT of landing outcomes.
- The landing outcomes were sorted using the GROUP BY clause, and the grouped landing outcomes were then arranged in descending order using the ORDER BY clause.



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)
```

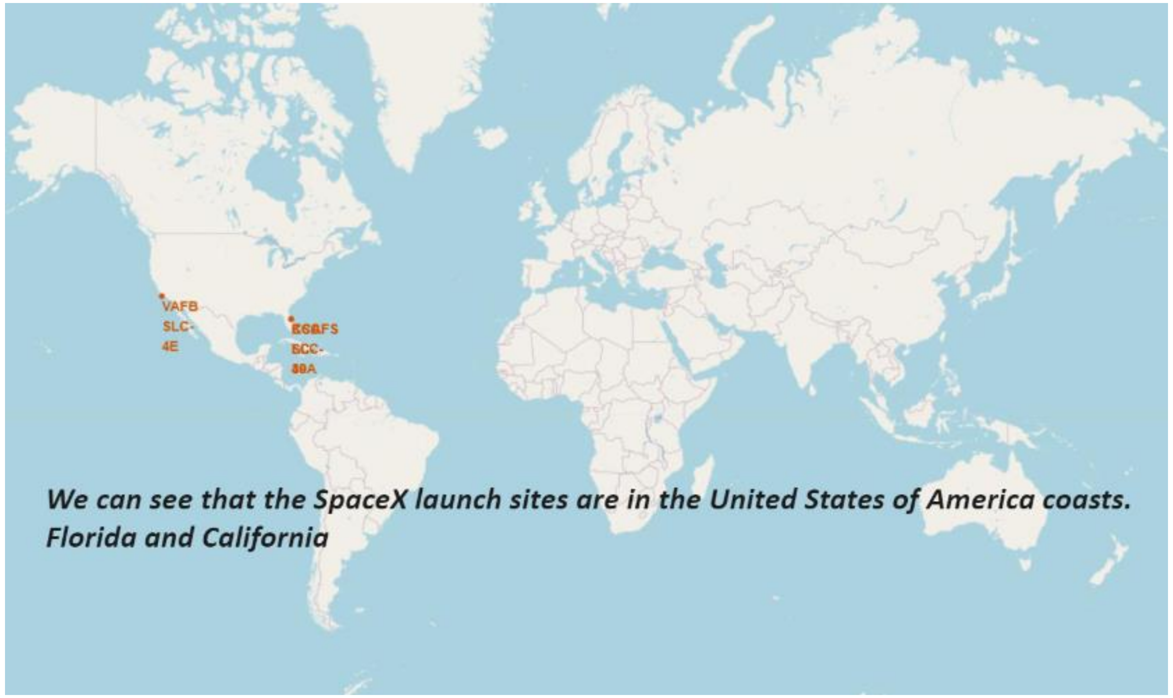
Out[19]:

	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

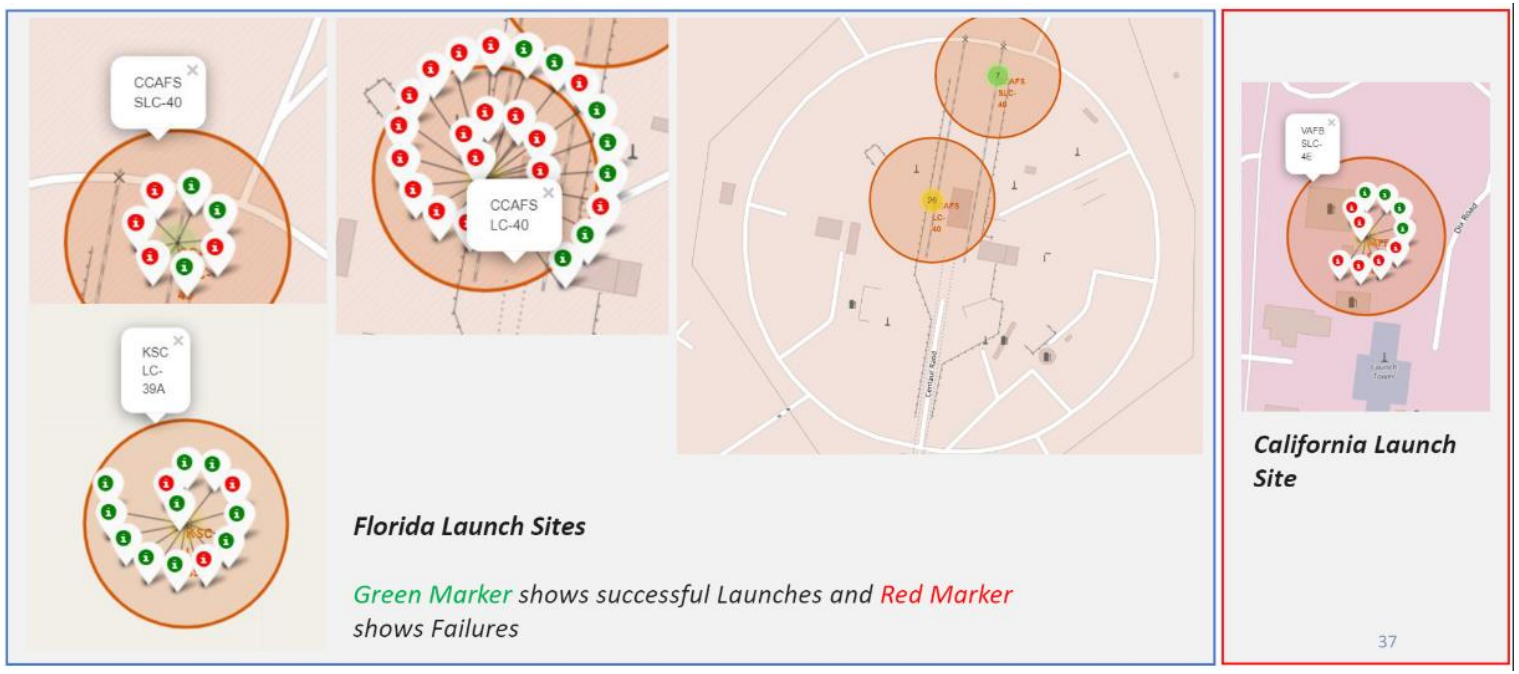
Launch Sites Proximities Analysis

All launch sites global map markers



We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

Markers showing launch sites with color labels



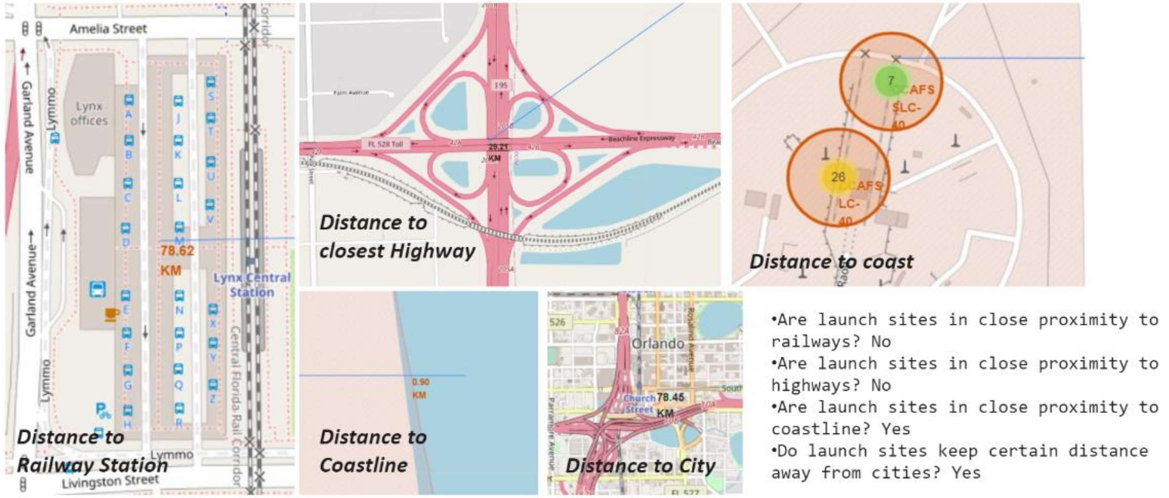
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

37

Launch Site distance to landmarks

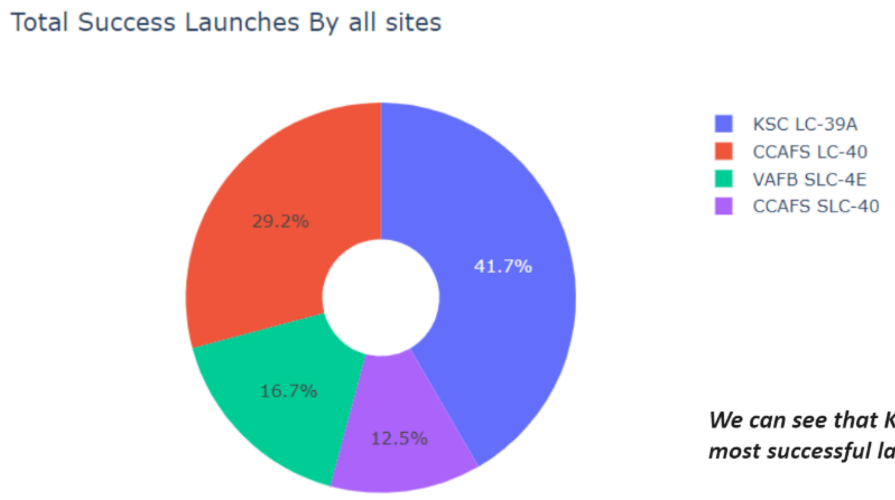


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

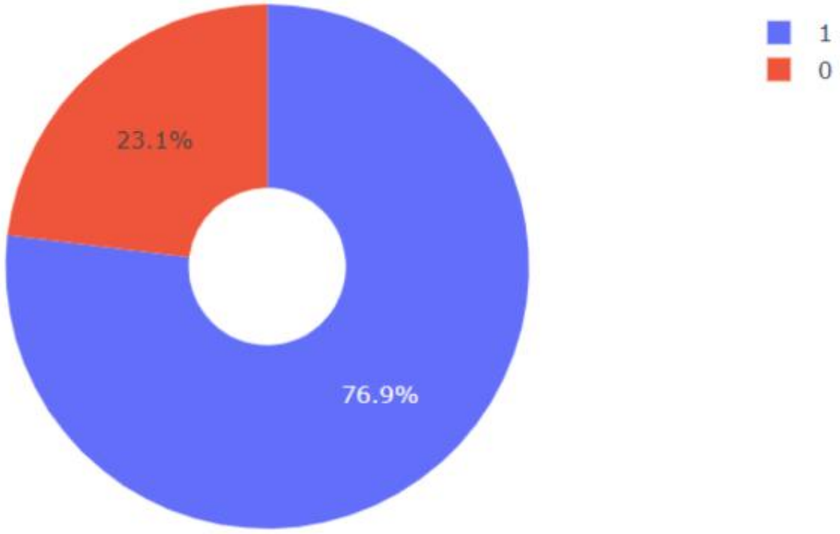
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



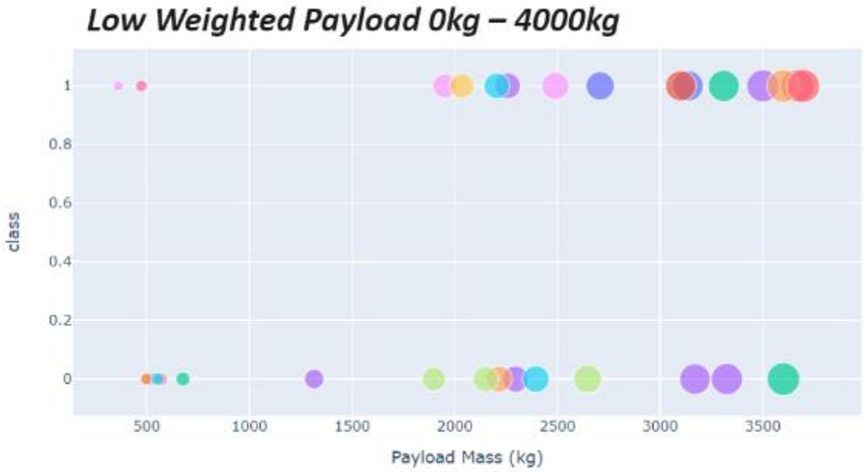
We can see that KSC LC-39A had the most successful launches from all the sites

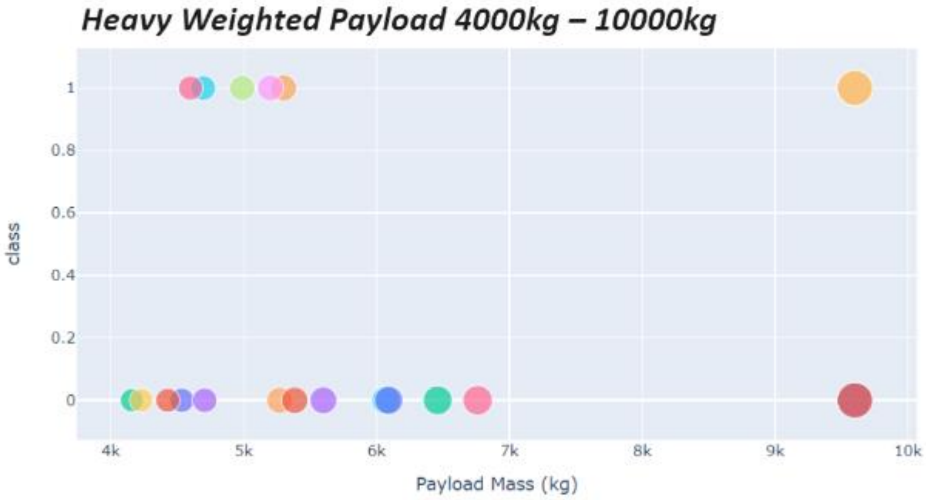
Pie chart showing the Launch site with the highest launch success ratio



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





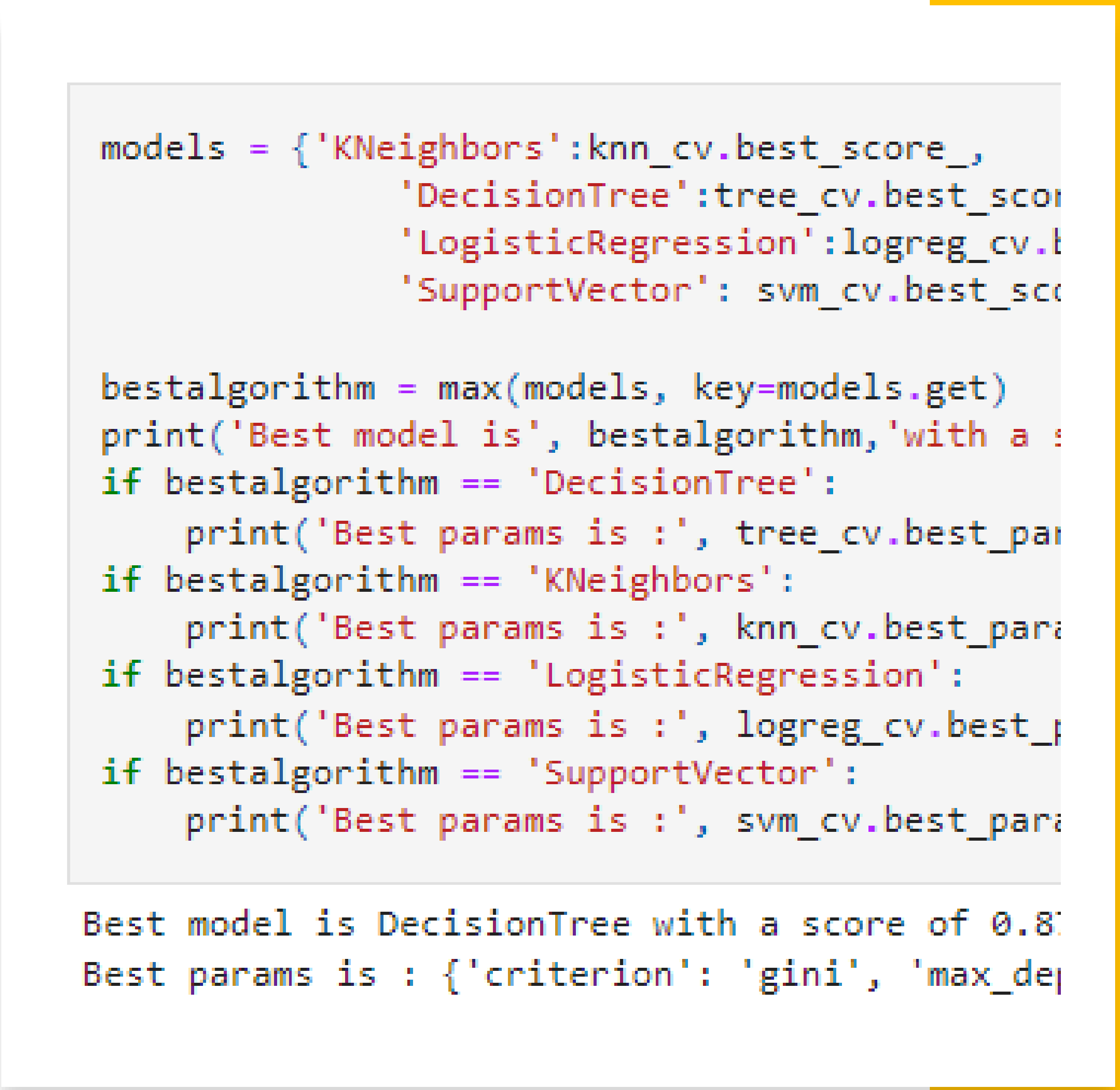
We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

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



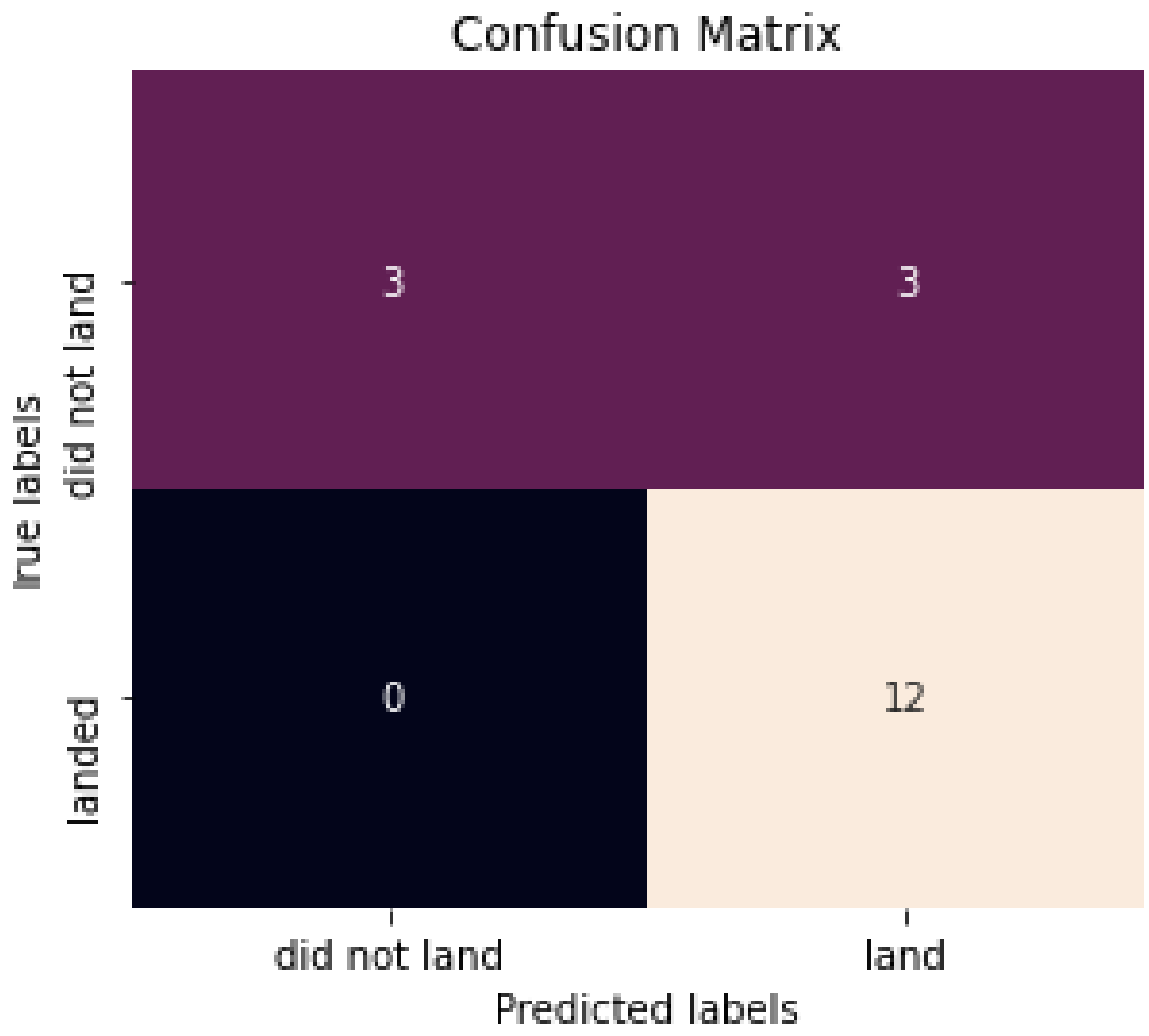
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_scor
              'LogisticRegression':logreg_cv.b
              'SupportVector': svm_cv.best_sco

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a s
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_par
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_para
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_p
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_para
```

```
Best model is DecisionTree with a score of 0.87
Best params is : {'criterion': 'gini', 'max_dep
```

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 draw the following conclusion:
- A launch site's success rate increases with the number of flights conducted there.
- The launch success rate increased from 2013 to 2020.
- Orbits with the highest success rate were ES-L1, GEO, HEO, SSO, and VLEO.
- Out of all the sites, KSC LC-39A had the most successful launches.
- For this problem, the optimal machine learning algorithm is the decision tree classifier.

Thank You