IBM Developer
SKILLS NETWORK

Cosmic Data Science:

Fueling the Space Race

Ahmad Abdullah Qureshi
27th June, 2023

OUTLINE

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

EXECUTIVE SUMMARY

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

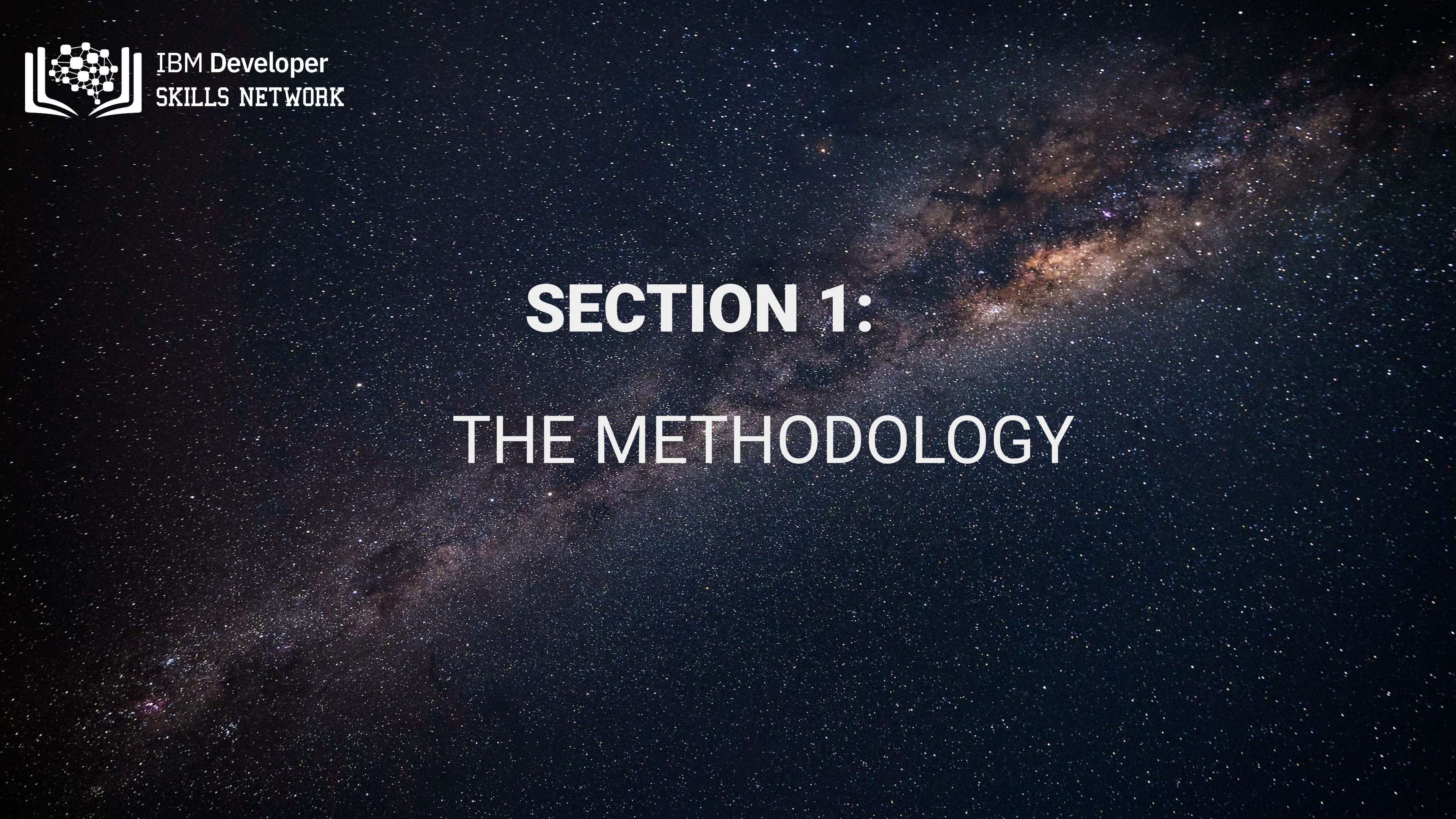
INTRODUCTION

Project Background and Context:

- SpaceX offers Falcon 9 rocket launches at a cost of 62 million dollars on its website, while other providers charge upwards of 165 million dollars per launch. The significant cost savings by SpaceX are attributed to their ability to reuse the first stage of the rocket. Thus, determining the likelihood of a successful landing of the first stage becomes crucial in estimating the overall launch cost. This information becomes valuable for alternate companies bidding against SpaceX for rocket launches. The project's objective is to develop a machine learning pipeline that predicts the probability of a successful first stage landing.

Problems to Address:

- Identifying the key factors influencing the successful landing of the first stage.
- Understanding the interplay between various features in determining the success rate of the landing.
- Determining the necessary operating conditions required for a successful landing program.



SECTION 1:

THE METHODOLOGY

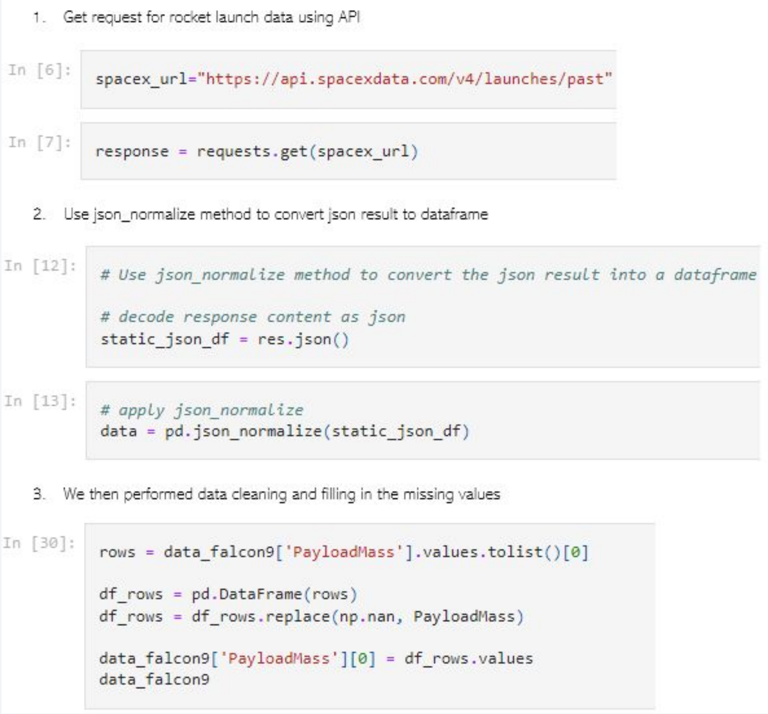
METHODOLOGY

Data Collection Methodology:
Data was obtained by utilizing the SpaceX API and conducting web scraping from Wikipedia.
Data Wrangling:
Categorical features underwent one-hot encoding for data preprocessing.
Exploratory Data Analysis (EDA):
Visualization techniques and SQL were employed to perform in-depth exploration of the data.
Interactive Visual Analytics:
Folium and Plotly Dash were utilized to create interactive visualizations for enhanced data analysis.
Predictive Analysis:
Classification models were implemented to conduct predictive analysis.
Model Building, Tuning, and Evaluation:
The process of constructing, fine-tuning, and evaluating the classification models was outlined.

DATA COLLECTION

Data was gathered through diverse approaches. Initially, data collection involved employing a get request to access the SpaceX API. The subsequent step involved decoding the response content as Json using the .json() function, which was then transformed into a pandas dataframe utilizing .json_normalize(). Subsequently, the data was cleaned, missing values were identified, and appropriate measures were taken to fill in any gaps. Furthermore, web scraping from Wikipedia was conducted specifically for Falcon 9 launch records using BeautifulSoup. The primary aim was to extract the launch records presented as an HTML table, parse the table, and convert it into a pandas dataframe for subsequent analysis.

DATA COLLECTION – SpaceX API



1. Get request for rocket launch data using API

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

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

2. Use json_normalize method to convert json result to dataframe

```
In [12]:  # Use json_normalize method to convert the json result into a dataframe

          # decode response content as json
          static_json_df = res.json()
```

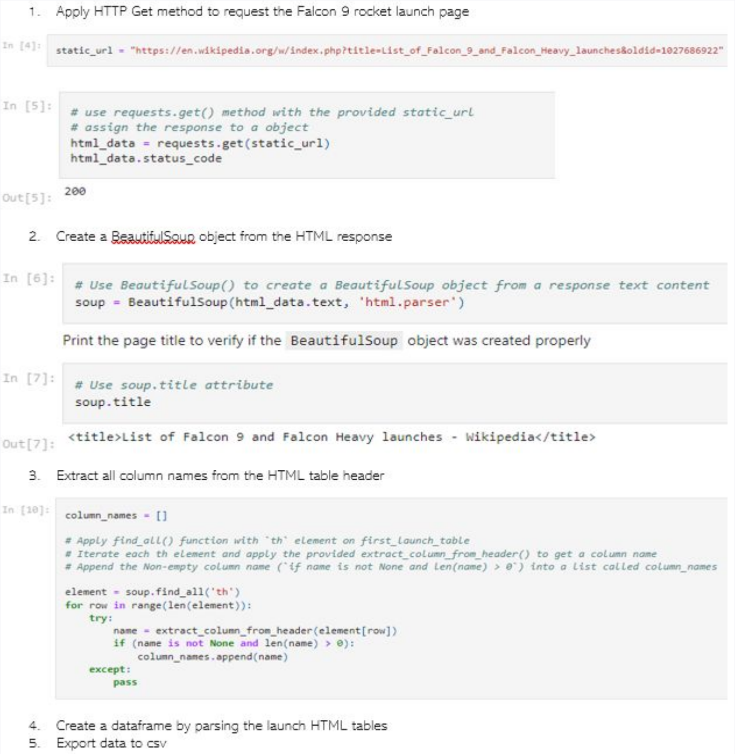
```
In [13]:  # apply json_normalize
          data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

```
In [30]:  rows = data_falcon9['PayloadMass'].values.tolist()[0]

          df_rows = pd.DataFrame(rows)
          df_rows = df_rows.replace(np.nan, PayloadMass)

          data_falcon9['PayloadMass'][0] = df_rows.values
          data_falcon9
```

- 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 to Notebook

DATA COLLECTION - SCRAPING



1. Apply HTTP Get method to request the Falcon 9 rocket launch page

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

```
In [5]: # use requests.get() method with the provided static_url
        # assign the response to a object
        html_data = requests.get(static_url)
        html_data.status_code

Out[5]: 200
```

2. Create a BeautifulSoup object from the HTML response

```
In [6]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
        soup = BeautifulSoup(html_data.text, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
In [7]: # Use soup.title attribute
        soup.title

Out[7]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

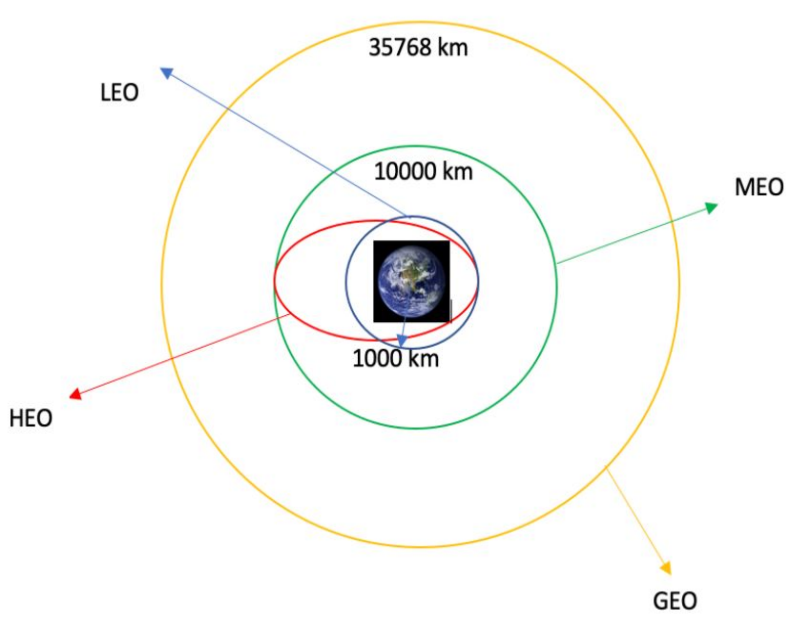
3. Extract all column names from the HTML table header

```
In [10]: column_names = []

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

         element = soup.find_all('th')
         for row in range(len(element)):
             try:
                 name = extract_column_from_header(element[row])
                 if (name is not None and len(name) > 0):
                     column_names.append(name)
             except:
                 pass
```

4. Create a dataframe by parsing the launch HTML tables
5. Export data to csv

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The Link to Notebook

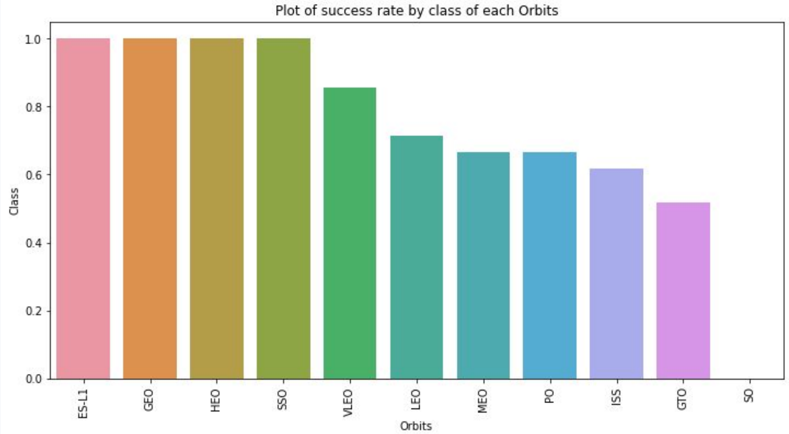
DATA WRANGLING

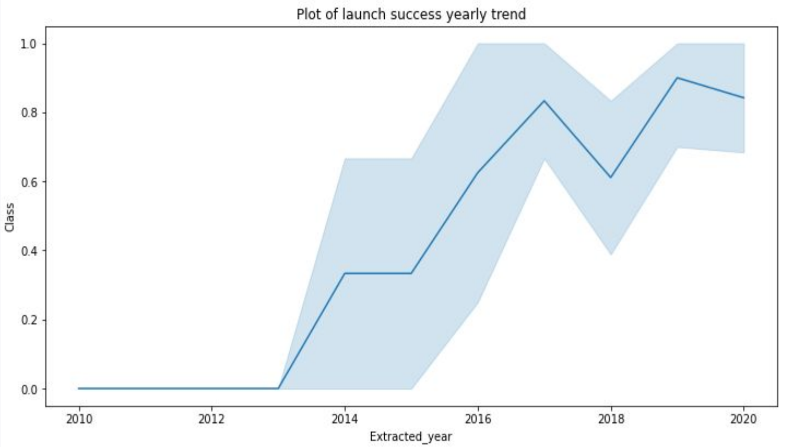


- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
  - [The Link to Notebook]()

EDA with DATA VISUALIZATION

- We explored the data 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.





- [The Link to Notebook]

EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The Link to Notebook

Folium: Interactive Map

Launch sites were meticulously marked on the folium map, with map objects like markers, circles, and lines employed to indicate the success or failure of launches at each site. To facilitate analysis, we assigned the launch outcomes (failure or success) to class 0 and 1, respectively. Consequently, we utilized color-labeled marker clusters to identify launch sites with relatively high success rates. Furthermore, we computed the distances between each launch site and its surrounding areas. By doing so, we addressed pertinent questions such as the proximity of launch sites to railways, highways, and coastlines, as well as whether launch sites maintain a certain distance from cities.

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
  - The Link to Notebook

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different 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.
- We found the best performing classification model.
  - The Link to Notebook

RESULTS

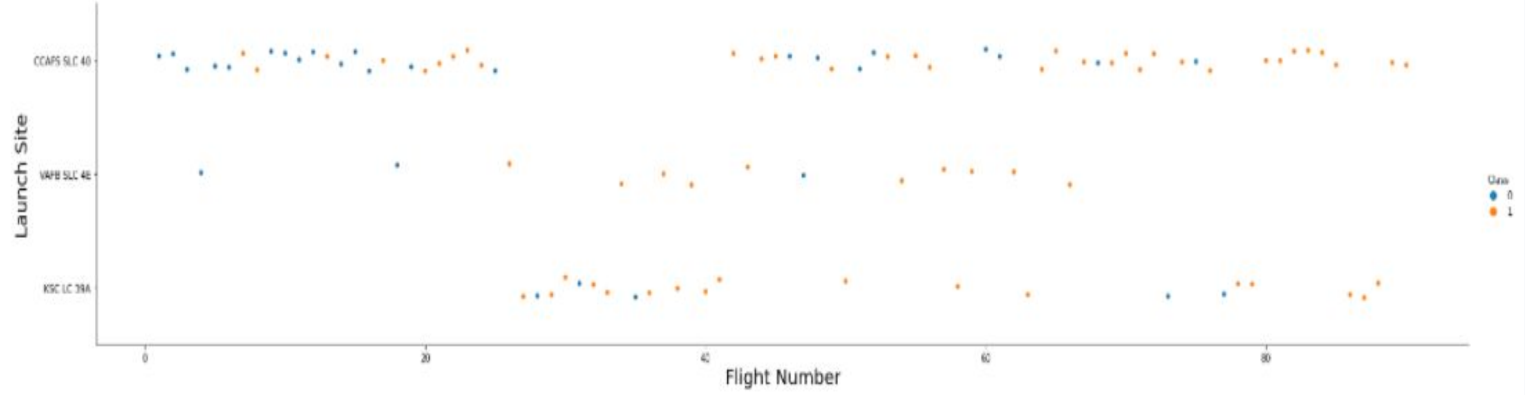
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

SECTION 2:

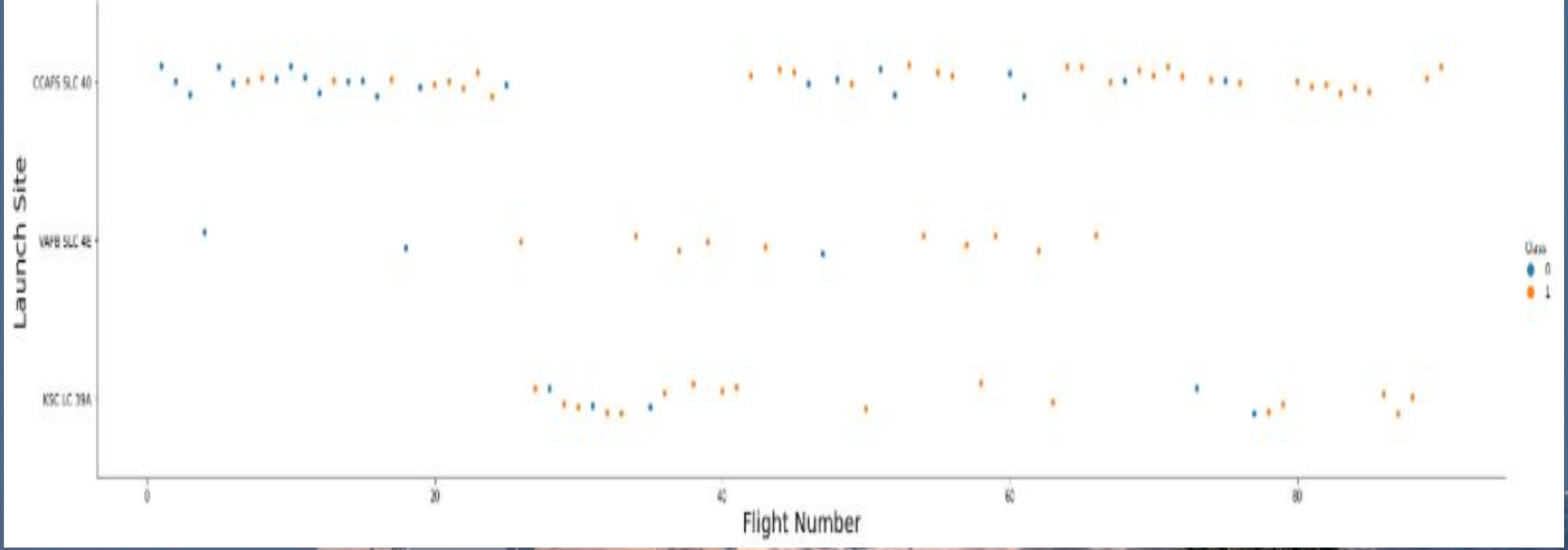
INSIGHTS DERIVED FROM EXPLORATORY DATA ANALYSIS (EDA)

Flight Number vs. Launch Site

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

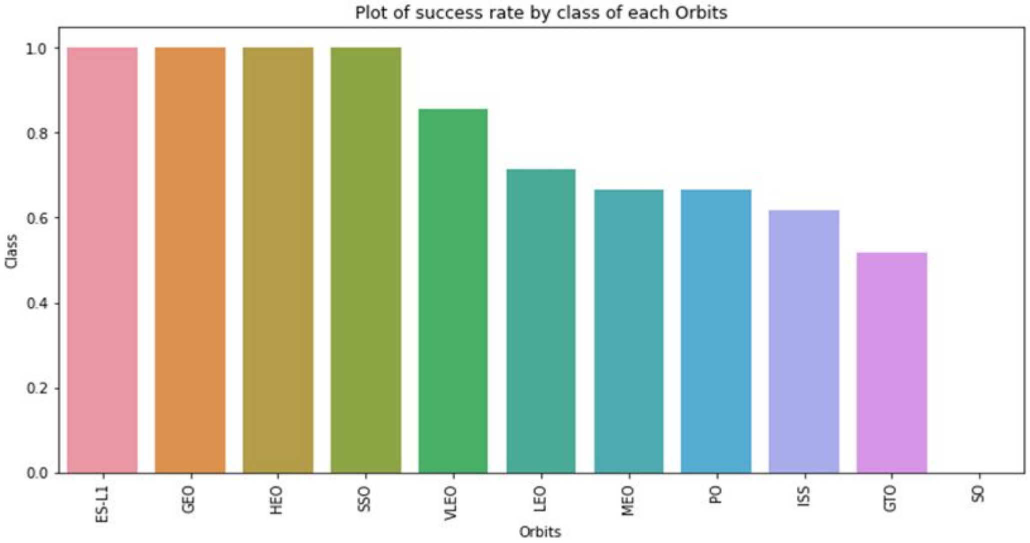


Payload vs. Launch Site



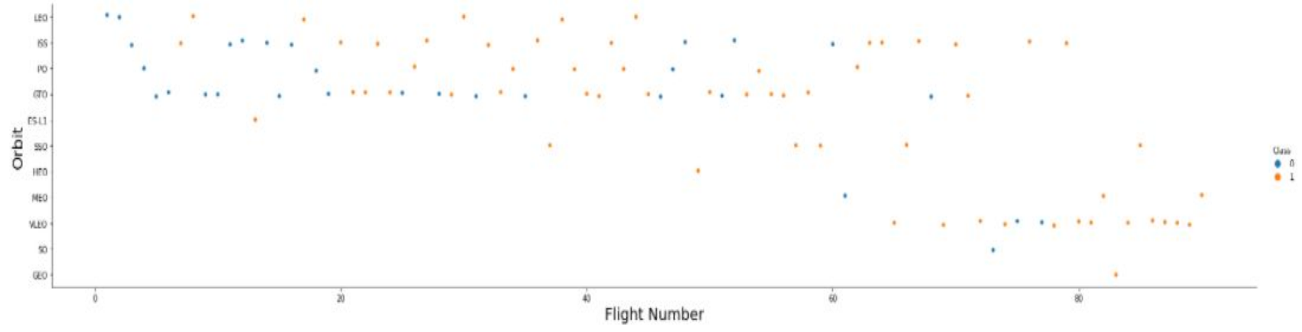
Success Rate vs. Orbit Type

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



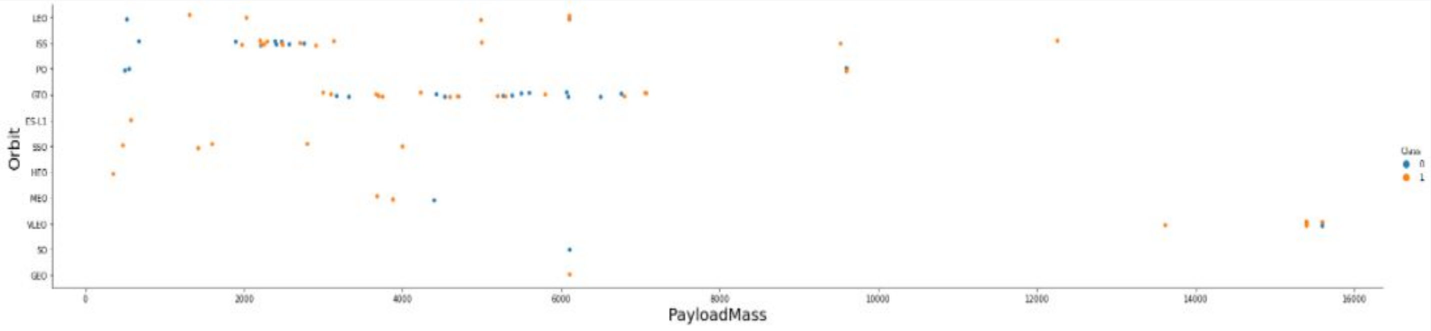
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



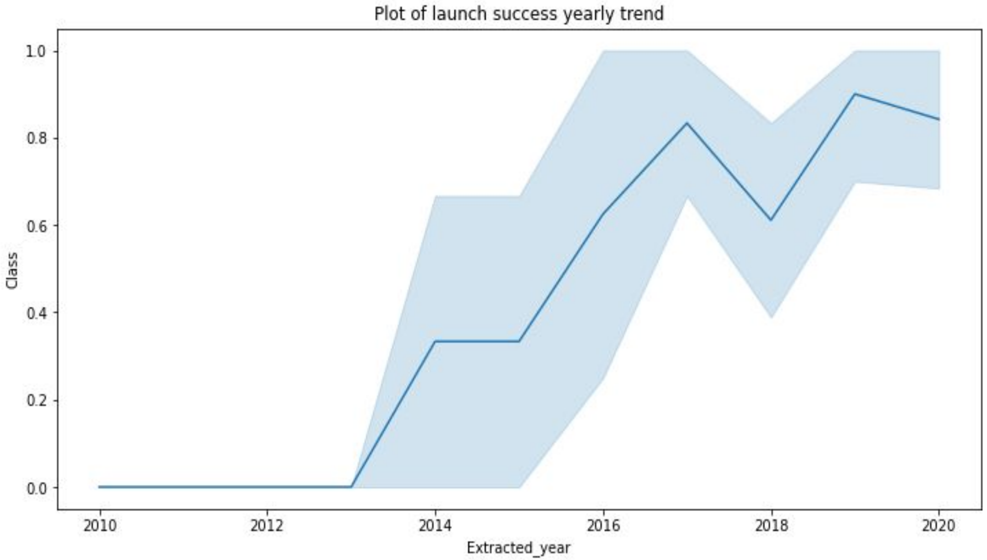
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



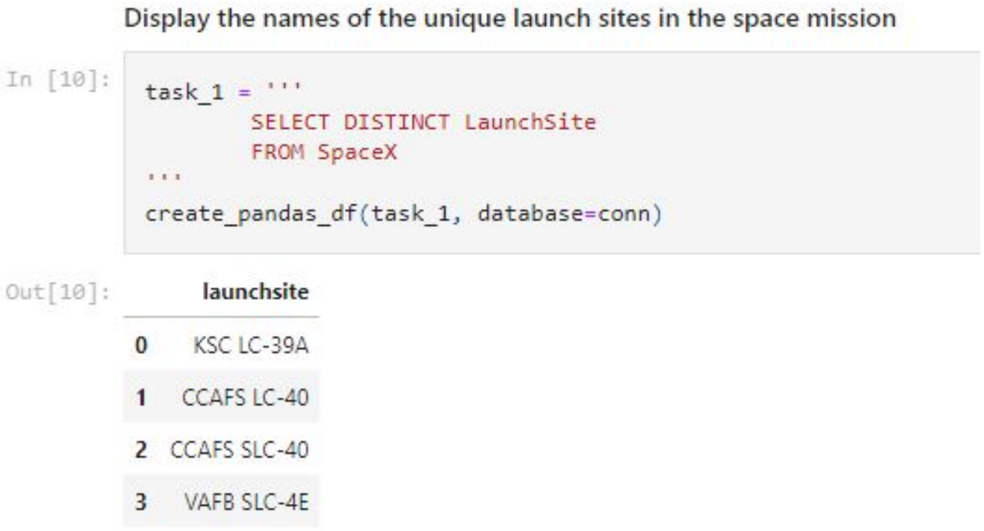
Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

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



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

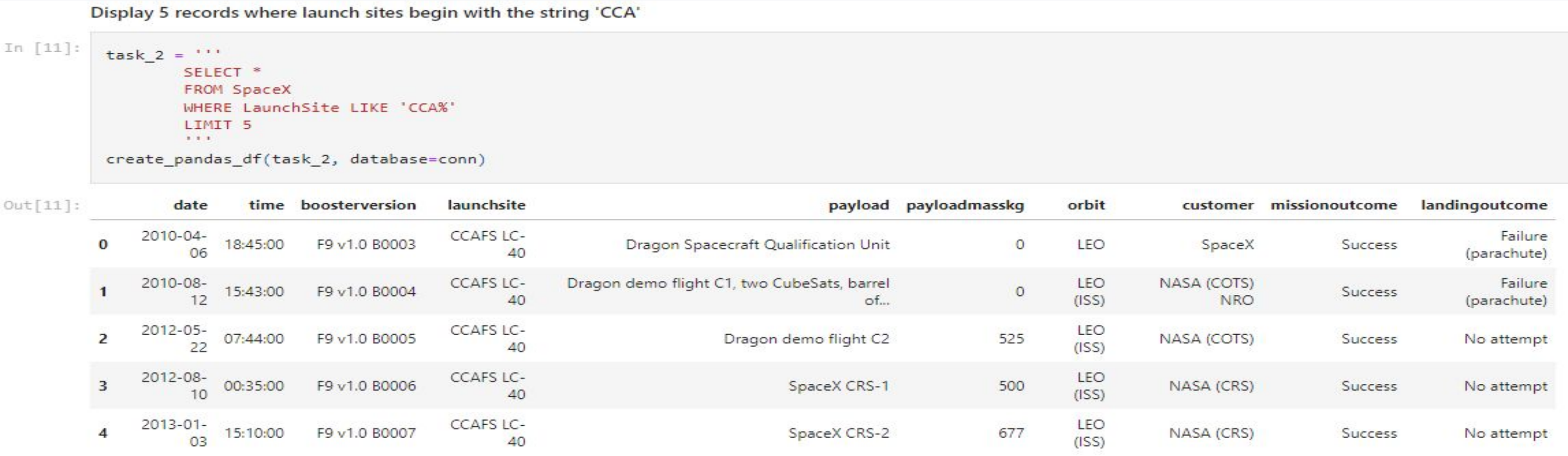
In [10]:

```
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)
```

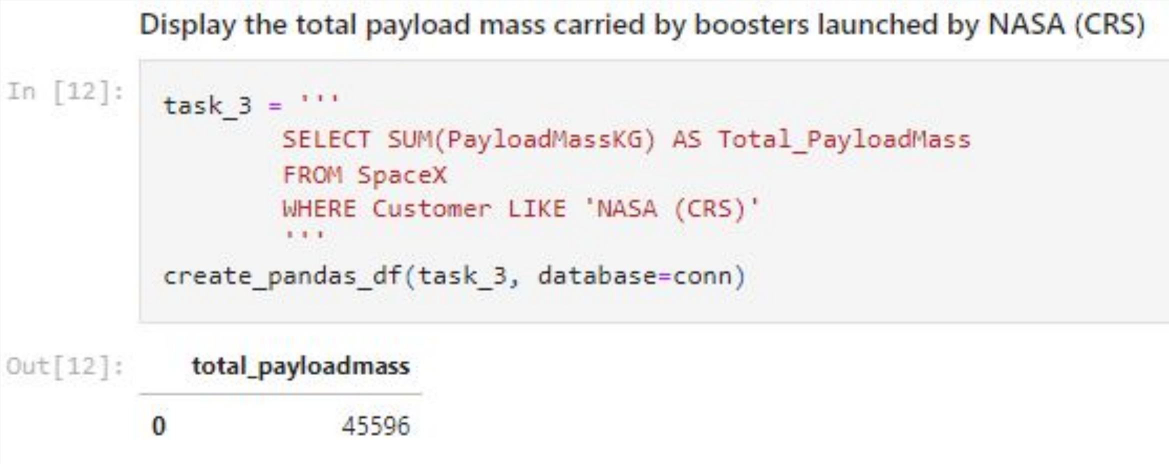
Out[11]:

	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	2012-08-10	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

- 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



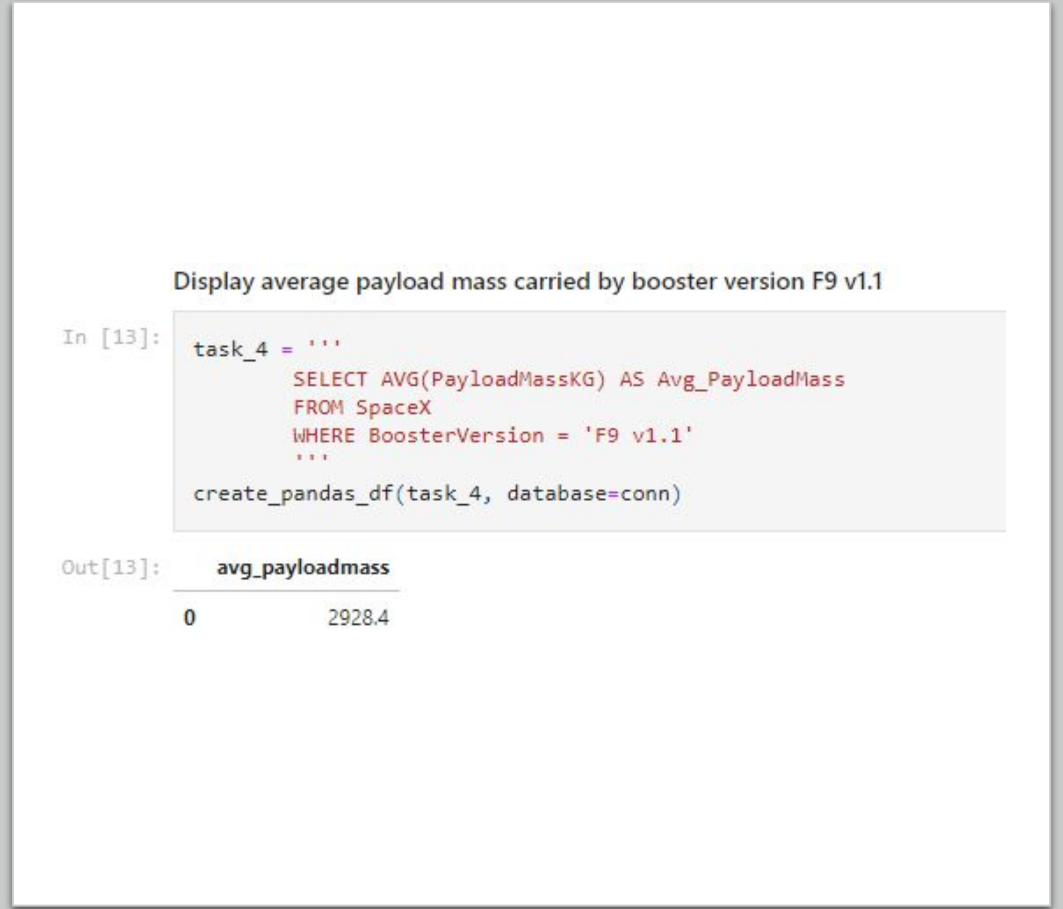
Display the total payload mass carried by boosters launched by NASA (CRS)

```
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
```

Average Payload Mass by F9 v1.1

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



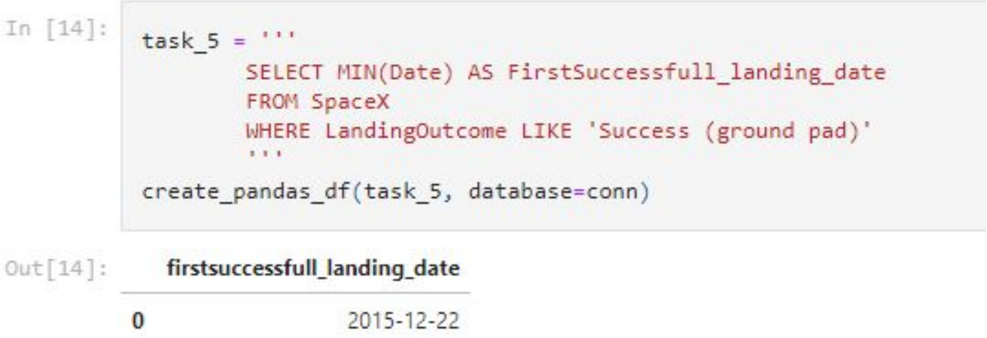
Display average payload mass carried by booster version F9 v1.1

```
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

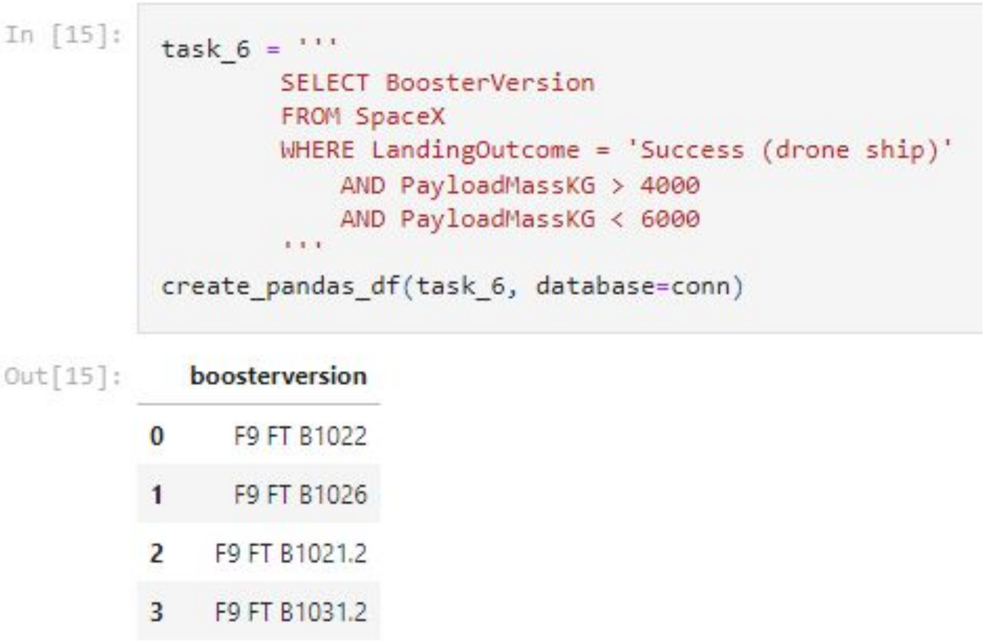
- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015



```
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

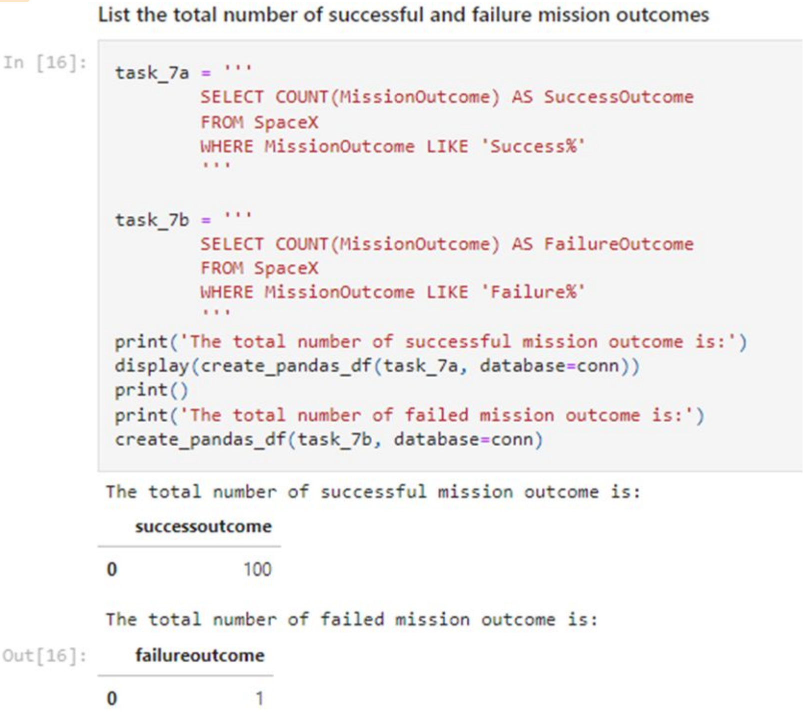


```
In [15]:

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
```

- 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

Total Number of Successful and Failure Mission Outcomes



List the total number of successful and failure mission outcomes

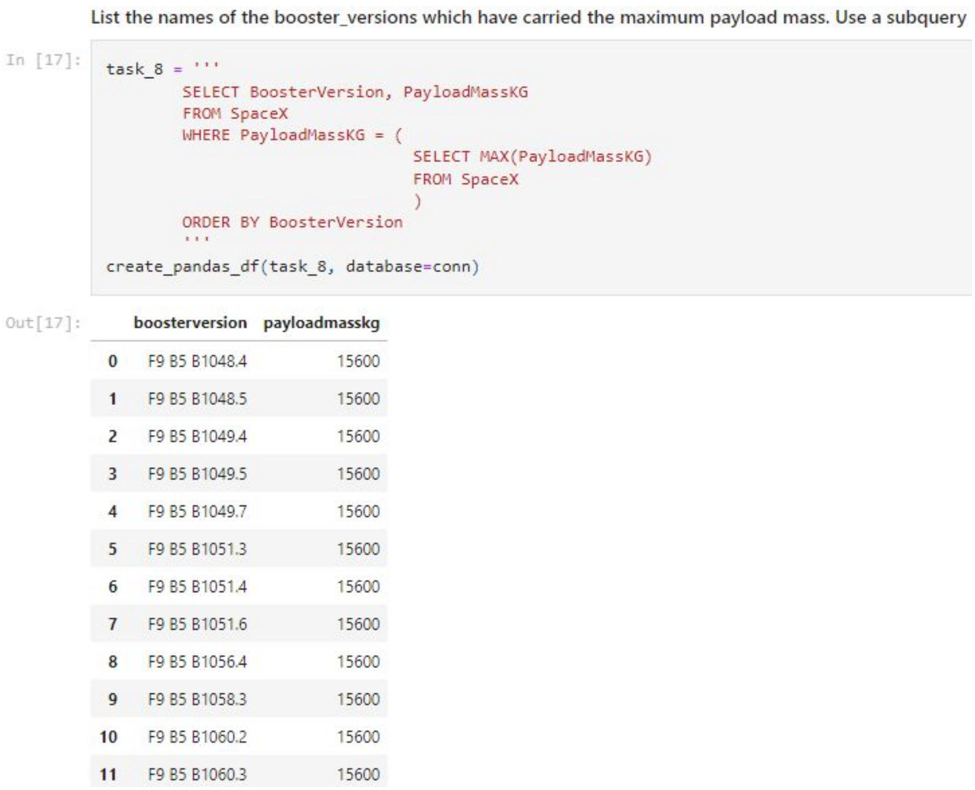
```
In [16]:  task_7a = '''
                  SELECT COUNT(MissionOutcome) AS SuccessOutcome
                  FROM SpaceX
                  WHERE MissionOutcome LIKE 'Success%'
                  '''

          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
```

- We used wildcard like ‘%’ to filter for **WHERE** Mission Outcome was a success or a failure.

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]:

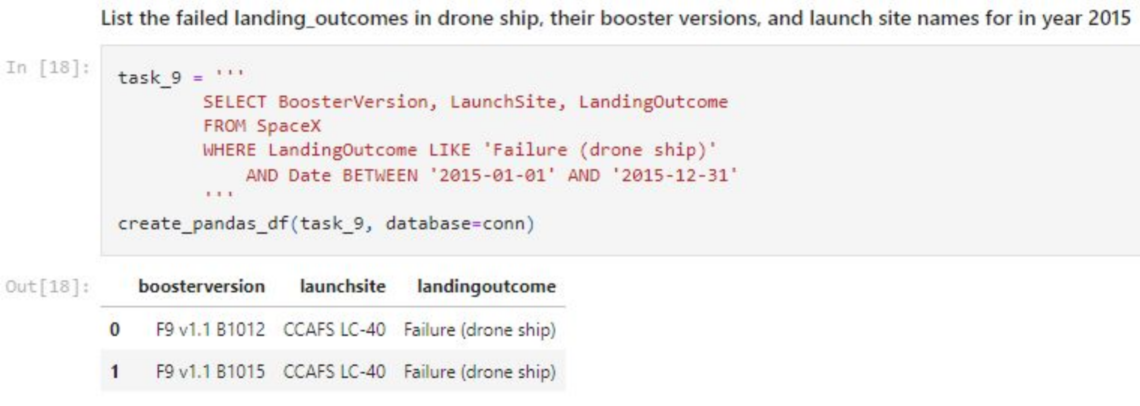
```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8,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

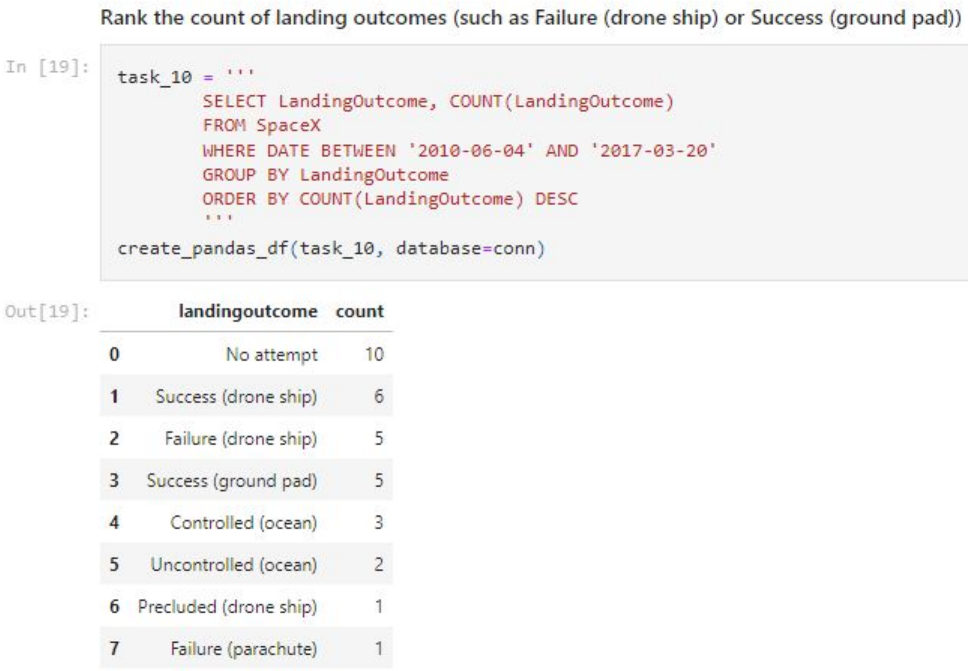
In [18]:

```
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



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

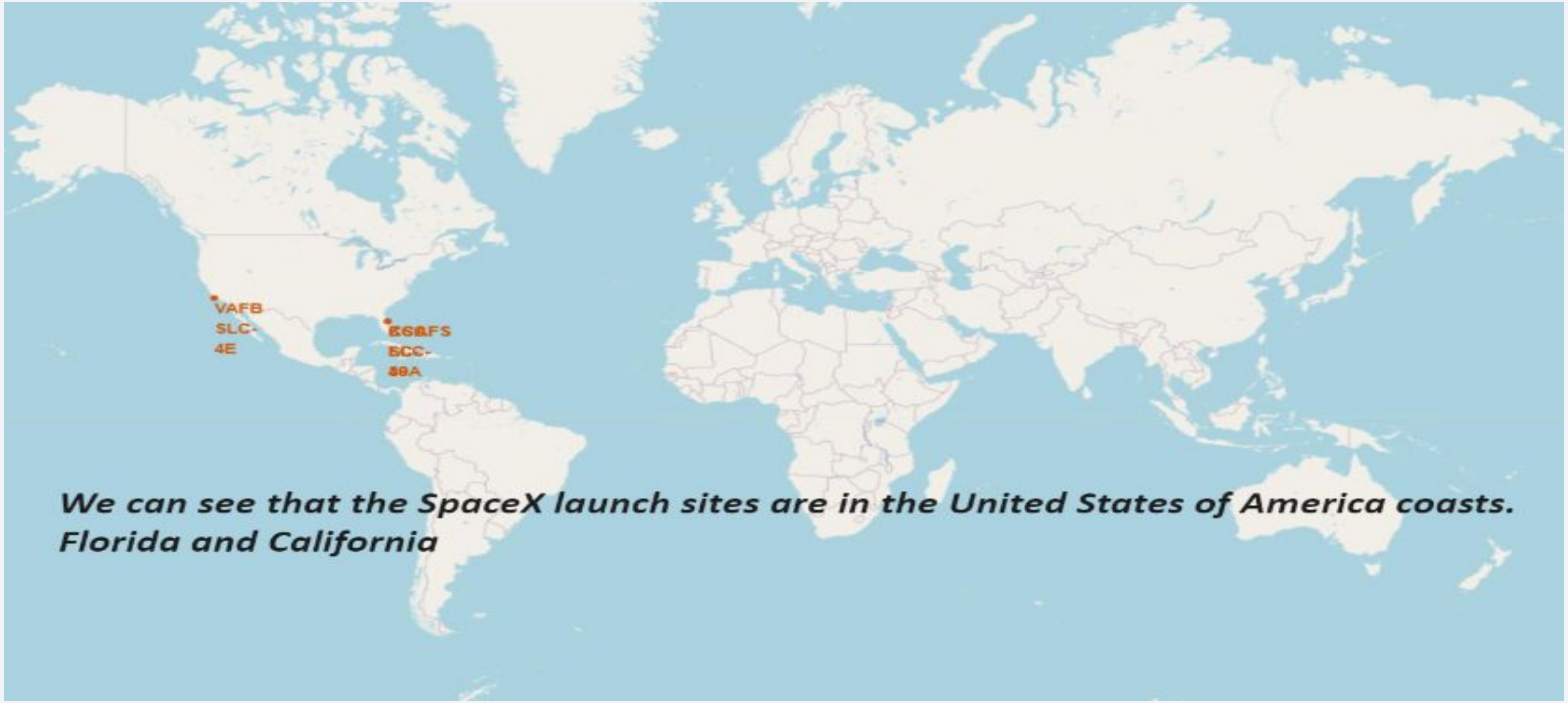
- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing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

IBM Developer
SKILLS NETWORK

SECTION 3:

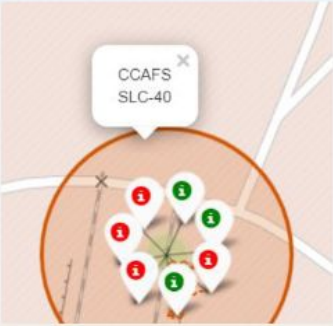
ANALYSIS OF LAUNCH SITE PROXIMITIES

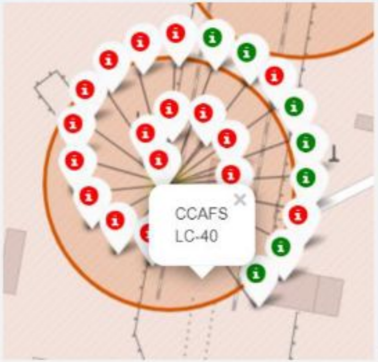
All launch sites global map markers



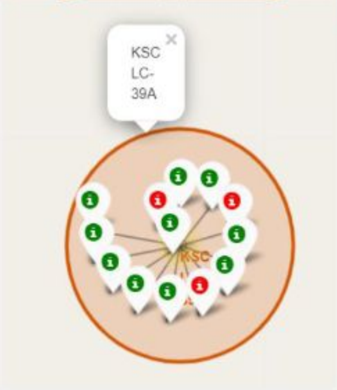
We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

Markers showing launch sites with color labels



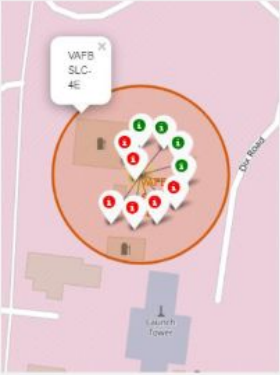


7
CCAFS SLC-40
26
CCAFS LC-40
Centaur Road



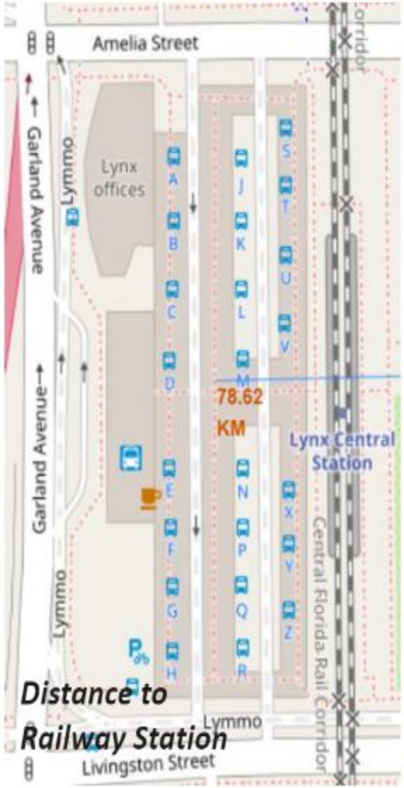
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

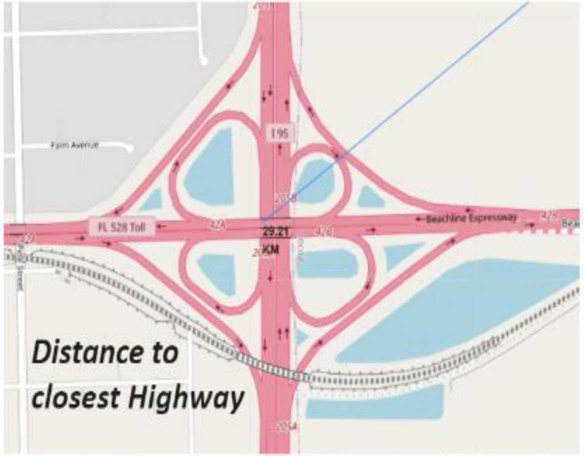


California Launch Site

Launch Site distance to landmarks



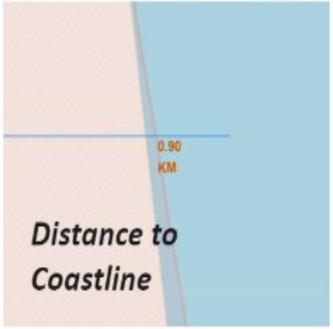
Distance to Railway Station



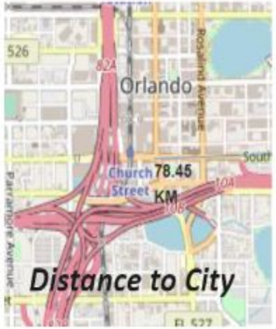
Distance to closest Highway

7
CCAFS SLC-40
26
CCAFS LC-40

Distance to coast



Distance to Coastline



Distance to City

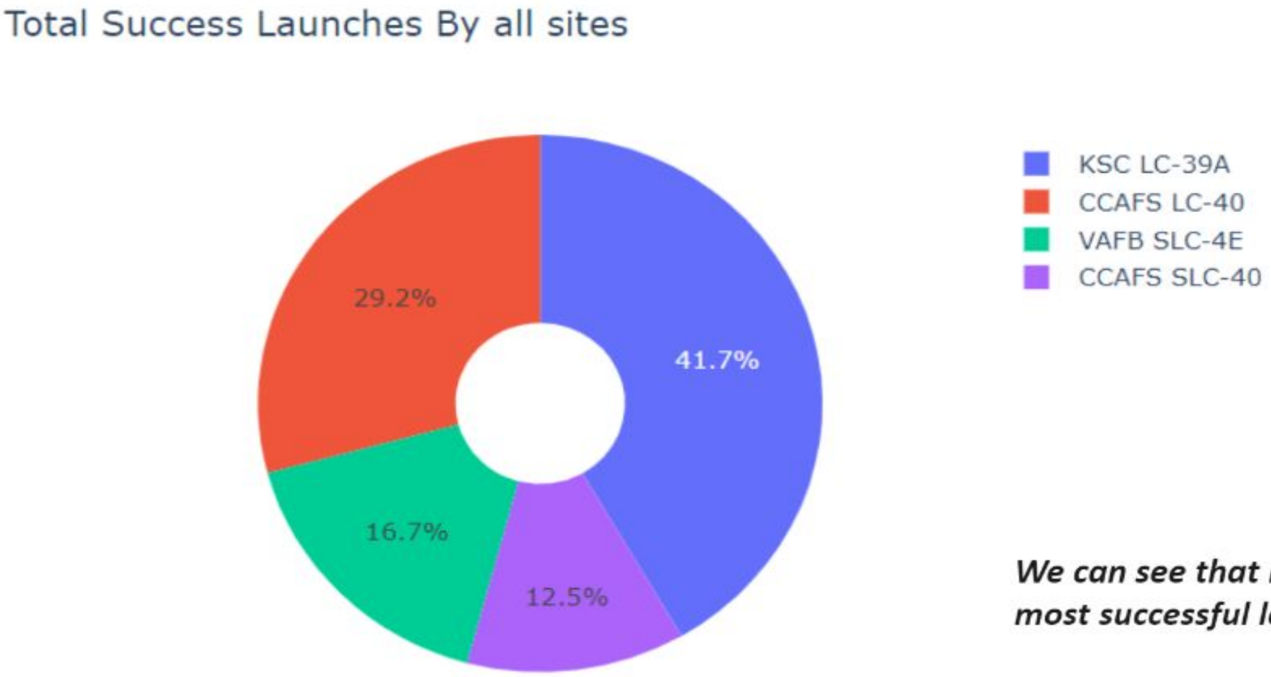
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

IBM Developer
SKILLS NETWORK

SECTION 4:

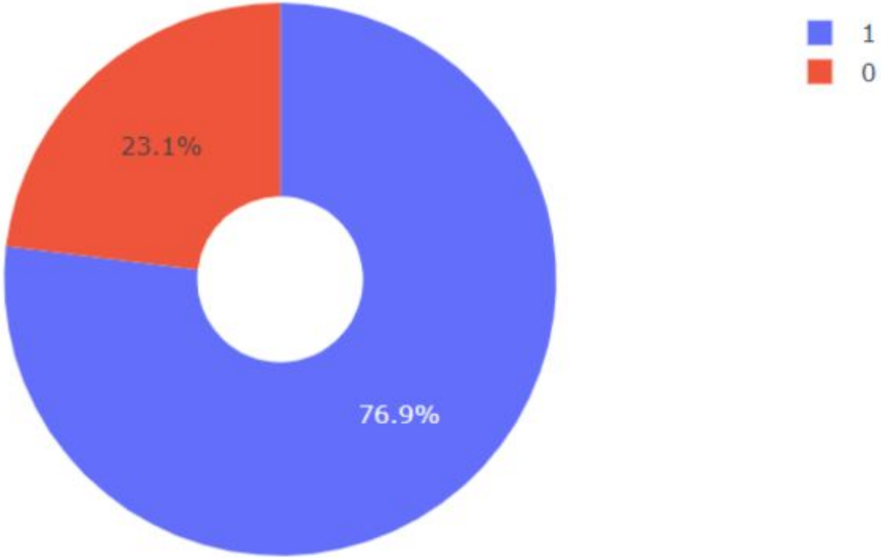
Building the Dashboard

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

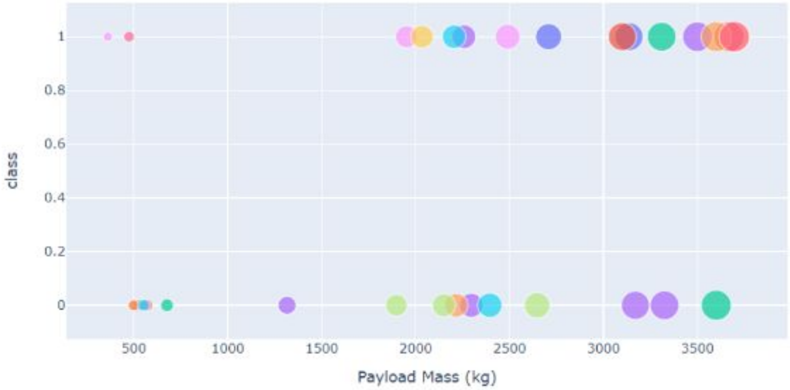
Pie chart showing the Launch site with the highest launch success ratio



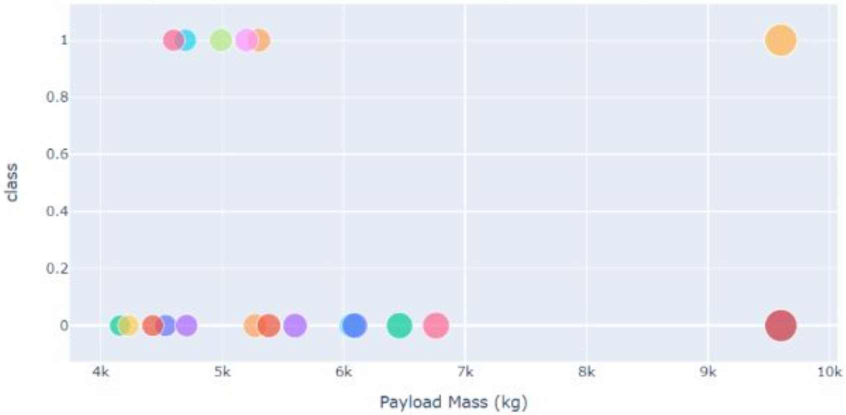
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

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

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg



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

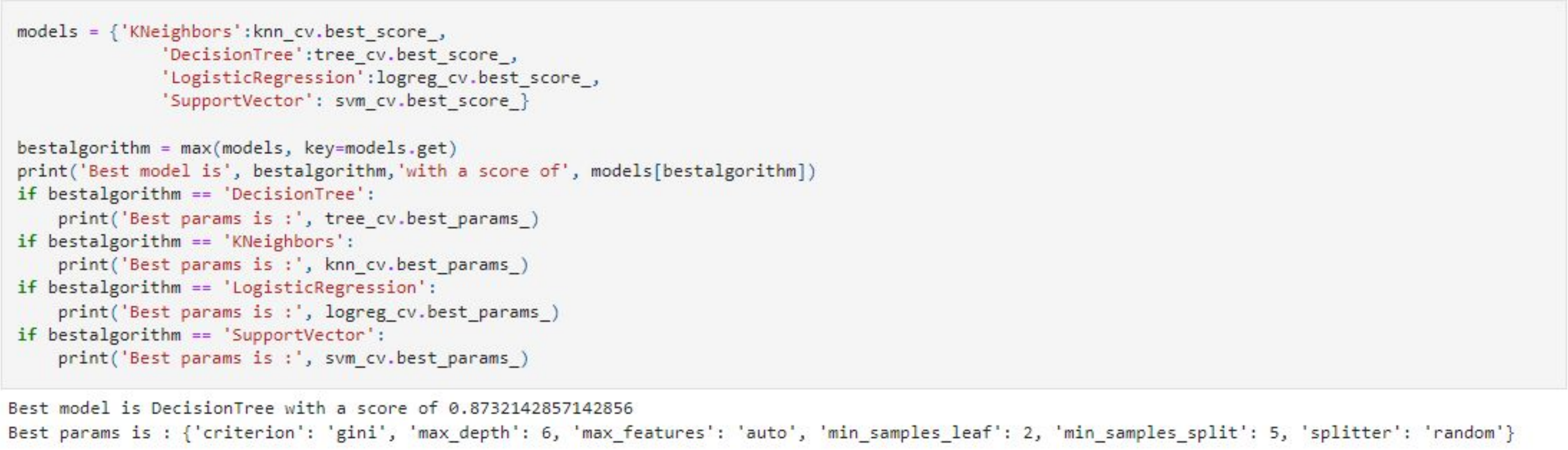
IBM Developer
SKILLS NETWORK

SECTION 5:

Predictive Analysis

Classification Accuracy

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

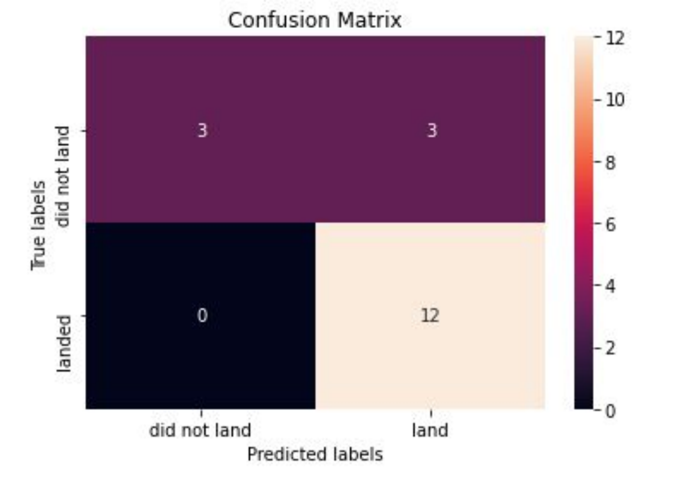


```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

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'}
```

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.

IBM Developer
SKILLS NETWORK

THANK YOU!