IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Glory Trizza
4th June 2022

Outline

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

- Space X advertises Falcon 9 rocket launch on its website with a cost of 62 million dollars; other providers cost 165 million dollars and above each. Space X can reuse the first stage and hence the cheaper cost. 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 factors determine if the rocket will land successfully?
- The interaction amongst various features that determine the success rate of a successful landing.
- What operating conditions needs to be in place to ensure a successful landing program.

METHODOLOGY

Section 1

Methodology

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Data wrangling
  - One-hot encoding was applied to categorical features
- Exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

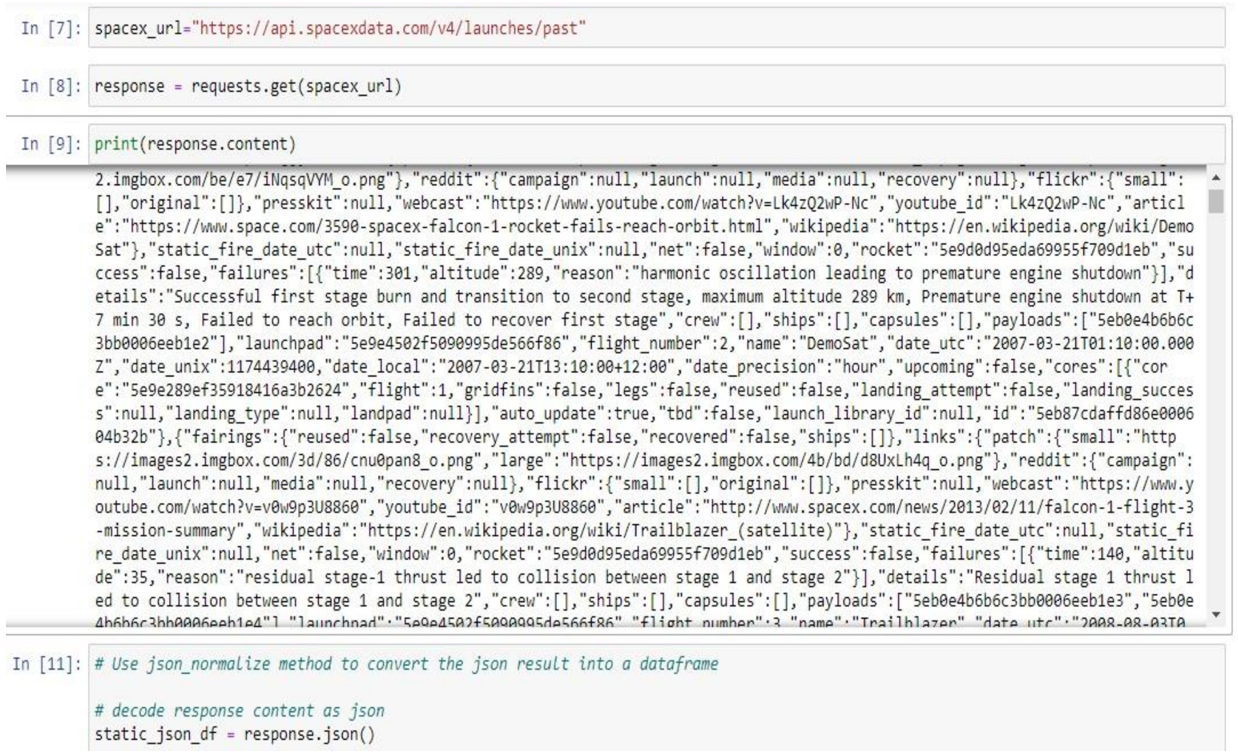
Data Collection

- Data collection was done using get request to the SpaceX API.
- Next, I decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
- I then cleaned the data, checked for missing values and filled in missing values where necessary.
- The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

I 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 the notebook is: https://github.com/GloryTrizza/IBM-DS-Capstone-Space-X/blob/main/Space_X_DS_Collecting_Data.ipynb

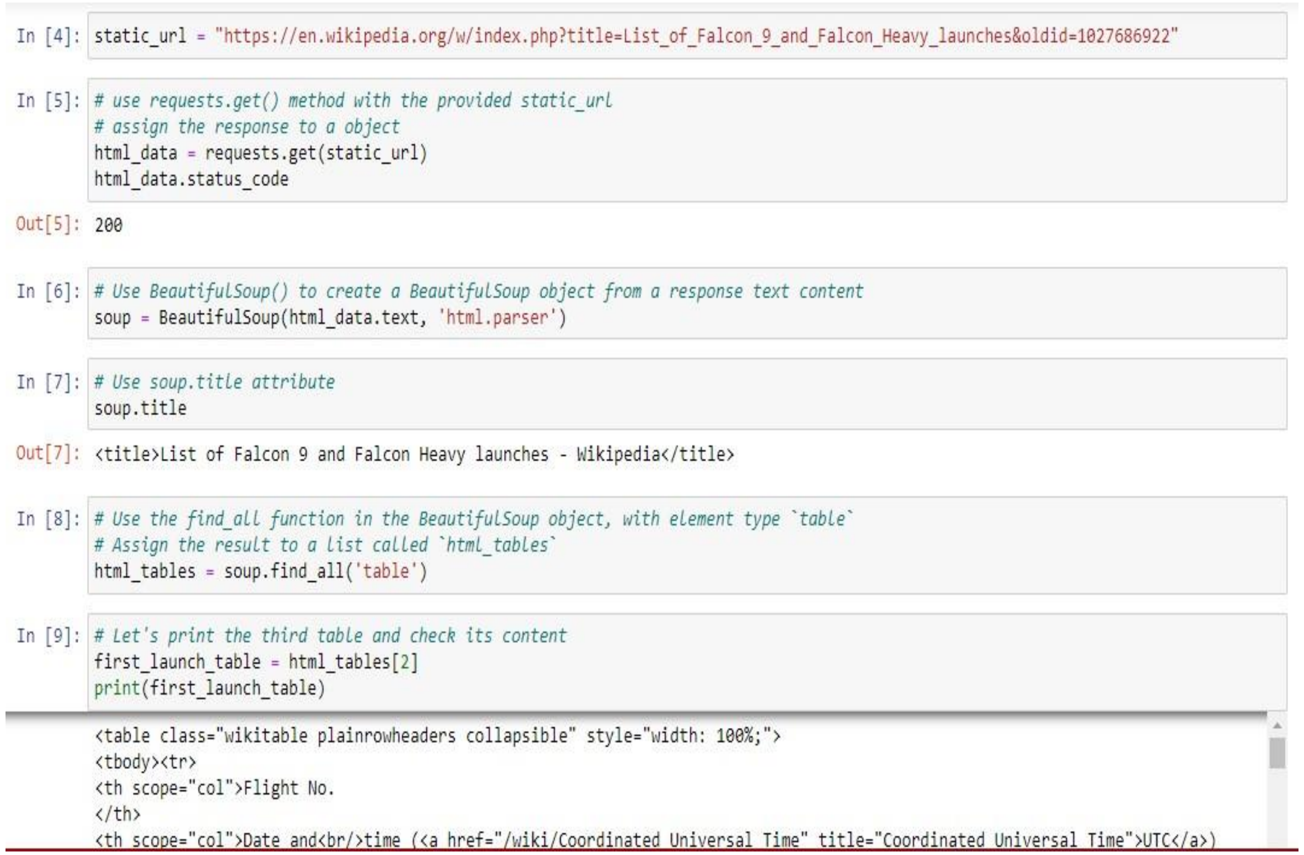


```
In [7]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [8]: response = requests.get(spacex_url)

In [9]: print(response.content)
```

```
2.imgbox.com/be/e7/iNqsqVYM_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":
[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=Lk4zQ2wP-Nc","youtube_id":"Lk4zQ2wP-Nc","articl
e":"https://www.space.com/3590-spacex-falcon-1-rocket-fails-reach-orbit.html","wikipedia":"https://en.wikipedia.org/wiki/Demo
Sat"},"static_fire_date_utc":null,"static_fire_date_unix":null,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","su
ccess":false,"failures":[{"time":301,"altitude":289,"reason":"harmonic oscillation leading to premature engine shutdown"}],"d
etails":"Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown at T+
7 min 30 s, Failed to reach orbit, Failed to recover first stage","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b6b6c
3bb0006eeb1e2"],"launchpad":"5e9e4502f5090995de566f86","flight_number":2,"name":"DemoSat","date_utc":"2007-03-21T01:10:00.000
Z","date_unix":1174439400,"date_local":"2007-03-21T13:10:00+12:00","date_precision":"hour","upcoming":false,"cores":[{"cor
e":"5e9e289ef35918416a3b2624","flight":1,"gridfins":false,"legs":false,"reused":false,"landing_attempt":false,"landing_succes
s":null,"landing_type":null,"landpad":null}],"auto_update":true,"tbd":false,"launch_library_id":null,"id":"5eb87cdaffd86e0006
04b32b"},{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"http
s://images2.imgbox.com/3d/86/cnu0pan8_o.png","large":"https://images2.imgbox.com/4b/bd/d8UxLh4q_o.png"},"reddit":{"campaign":
null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.y
outube.com/watch?v=v0w9p3U8860","youtube_id":"v0w9p3U8860","article":"http://www.spacex.com/news/2013/02/11/falcon-1-flight-3
-mission-summary","wikipedia":"https://en.wikipedia.org/wiki/Trailblazer_(satellite)"},"static_fire_date_utc":null,"static_fi
re_date_unix":null,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":140,"altitu
de":35,"reason":"residual stage-1 thrust led to collision between stage 1 and stage 2"}],"details":"Residual stage 1 thrust l
ed to collision between stage 1 and stage 2","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b6b6c3bb0006eeb1e3","5eb0e
4b6b6c3bb0006eeb1e4"],"launchpad":"5e9e4502f5090995de566f86","flight_number":3,"name":"Trailblazer","date_utc":"2008-08-03T0
```

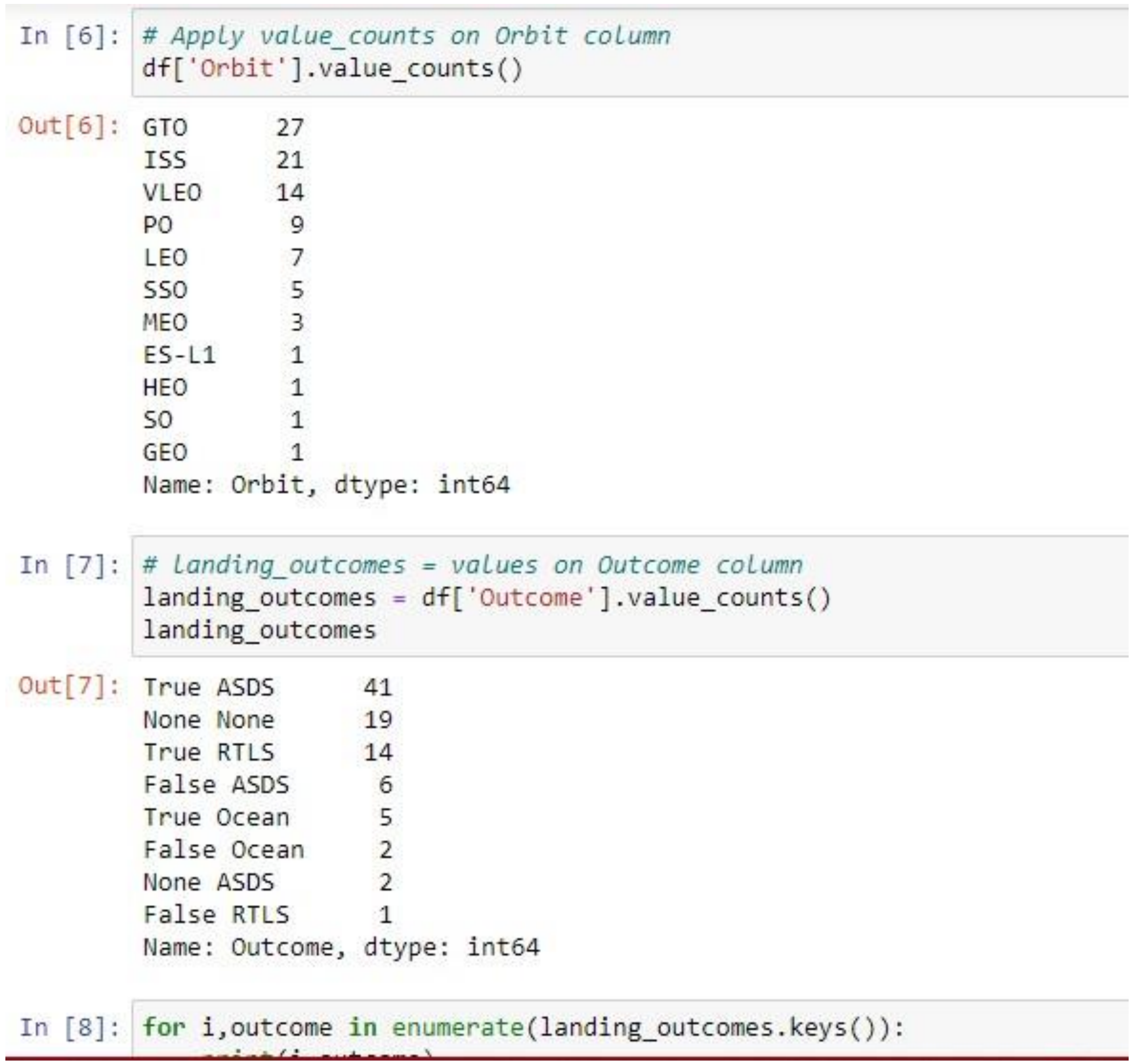
```
In [11]: # Use json_normalize method to convert the json result into a dataframe

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

Data Collection - Scraping



```
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

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

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

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

In [8]: # Use the find_all function in the BeautifulSoup object, with element type `table`
        # Assign the result to a list called `html_tables`
        html_tables = soup.find_all('table')

In [9]: # Let's print the third table and check its content
        first_launch_table = html_tables[2]
        print(first_launch_table)

        <table class="wikitable plainrowheaders collapsible" style="width: 100%;">
        <tbody><tr>
        <th scope="col">Flight No.
        </th>
        <th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
```

- I applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- I parsed the table and converted it into a pandas dataframe.
- The link to the notebook is: https://github.com/GloryTrizza/IBM-DS-Capstone-Space-X/blob/main/DC_Web_Scraping.ipynb



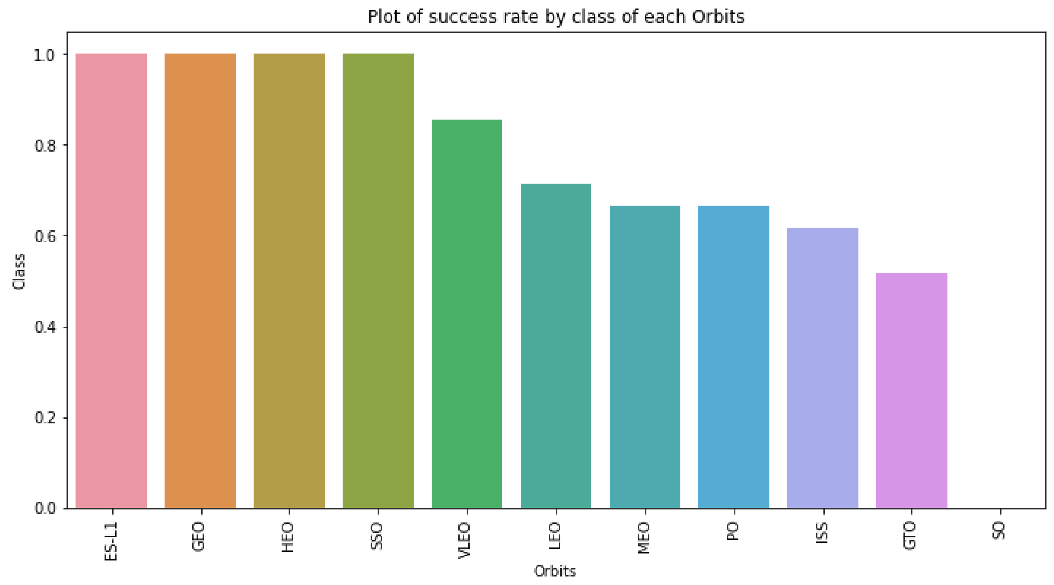
```
In [6]: # Apply value_counts on Orbit column
        df['Orbit'].value_counts()

Out[6]: GTO      27
        ISS      21
        VLEO     14
        PO        9
        LEO       7
        SSO       5
        MEO       3
        ES-L1     1
        HEO       1
        SO        1
        GEO       1
        Name: Orbit, dtype: int64

In [7]: # landing_outcomes = values on Outcome column
        landing_outcomes = df['Outcome'].value_counts()
        landing_outcomes

Out[7]: True ASDS      41
        None None      19
        True RTLS      14
        False ASDS      6
        True Ocean      5
        False Ocean     2
        None ASDS       2
        False RTLS      1
        Name: Outcome, dtype: int64

In [8]: for i,outcome in enumerate(landing_outcomes.keys()):
```

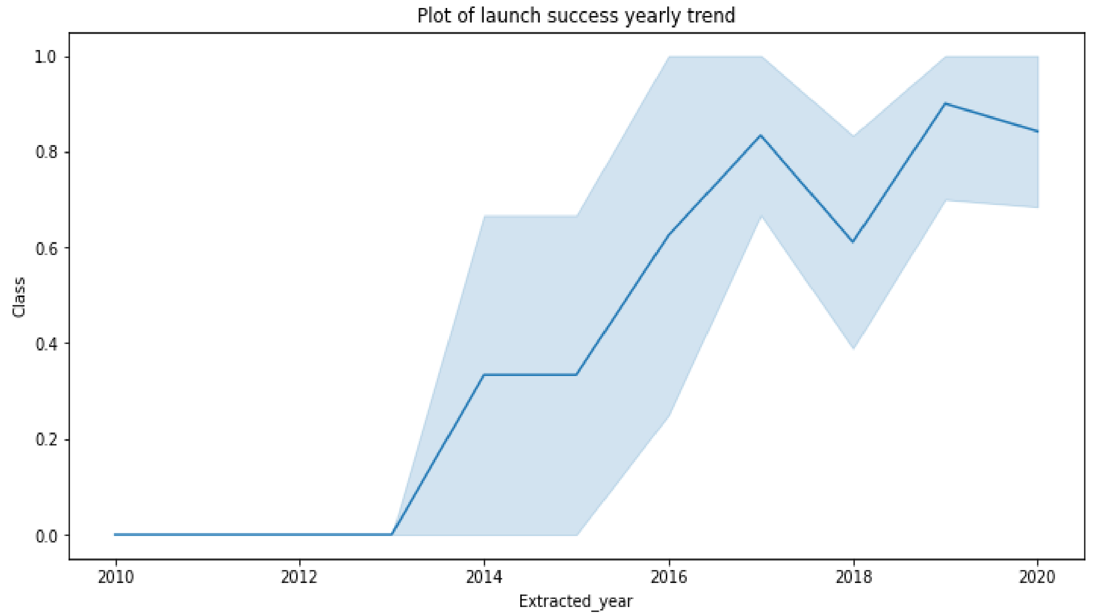
Data Wrangling

- I performed exploratory data analysis and determined the training labels.
- I calculated the number of launches at each site, and the number and occurrence of each orbits
- I then created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/GloryTrizza/IBM-DS-Capstone-Space-X/blob/main/Data_Wrangling.ipynb

EDA with Data Visualization

- I 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 the notebook is: https://github.com/GloryTrizza/IBM-DS-Capstone-Space-X/blob/main/EDA_with_Data_Visualization.ipynb





EDA with SQL

- I loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- I applied EDA with SQL to get insight from the data. I 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.

Build an Interactive Map with Folium

- I 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.
- I assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- I calculated the distances between a launch site to its proximities. I answered some question for instance:
  - Are launch sites near railways, highways and coastlines?
  - Do launch sites keep certain distance away from cities?

Build a Dashboard with Plotly Dash

- I built an interactive dashboard with Plotly dash
- I plotted pie charts showing the total launches by a certain sites
- I plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The link to the notebook is: https://github.com/GloryTrizza/IBM-DS-Capstone-Space-X/blob/main/Dashboard.py

Predictive Analysis (Classification)

- I loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- I built different machine learning models and tune different hyperparameters using GridSearchCV.
- I used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- I found the best performing classification model.
- The link to the notebook is: https://github.com/GloryTrizza/IBM-DS-Capstone-Space-X/blob/main/Machine_Learning_Prediction.ipynb

Results

Exploratory data analysis results

Interactive analytics demo in screenshots

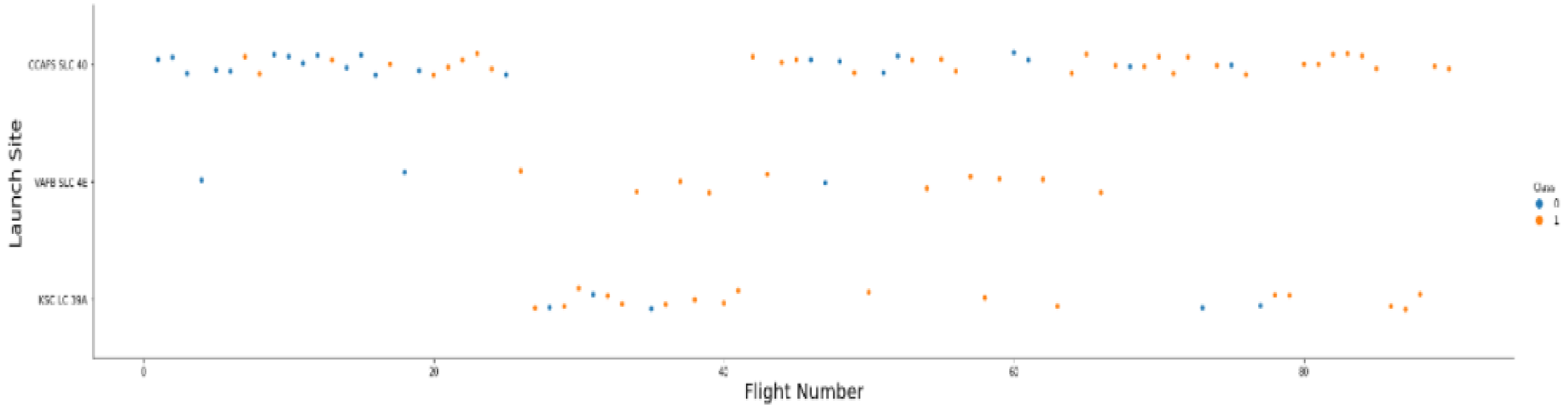
Predictive analysis results

INSIGHTS DRAWN FROM EDA

Section 2

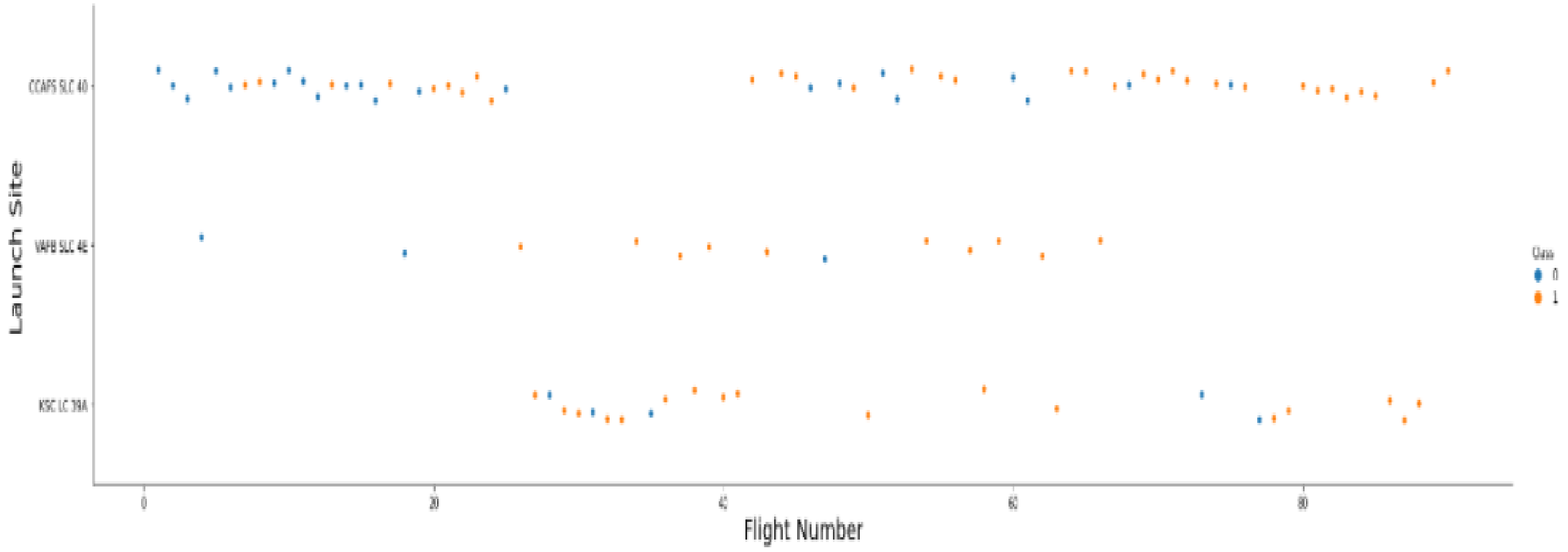
Flight Number vs. Launch Site

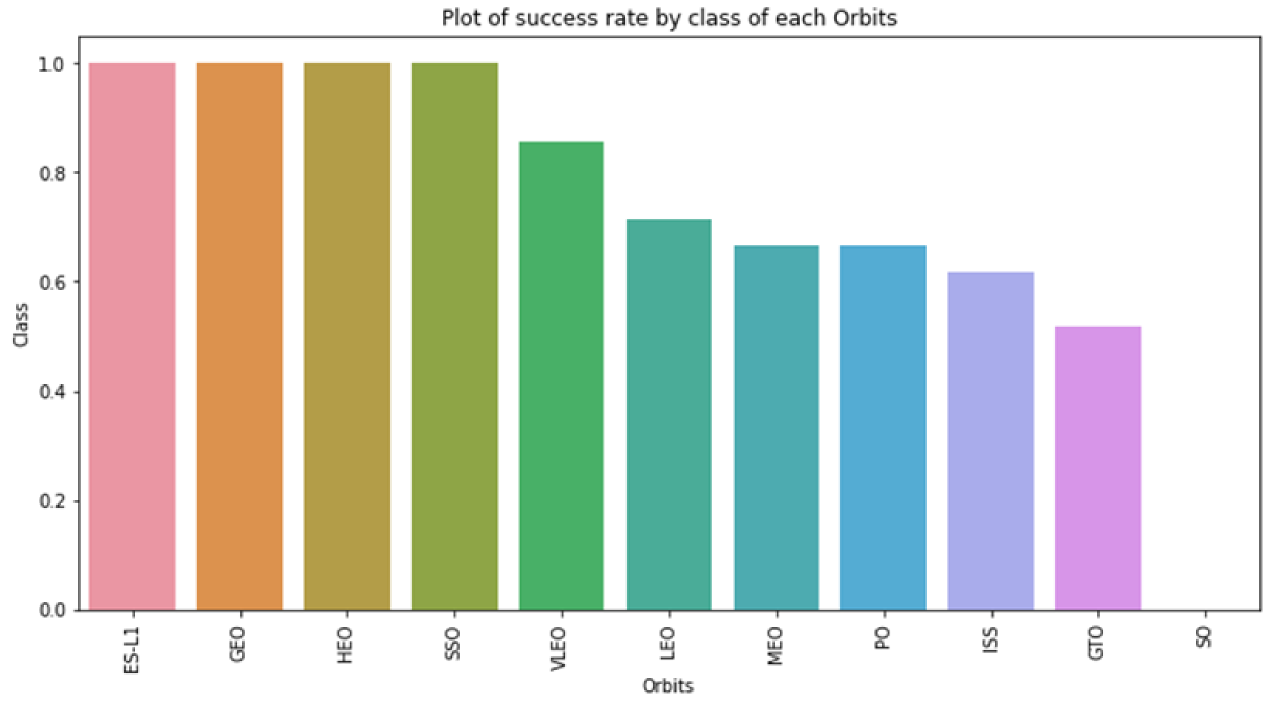
- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



Payload vs. Launch Site

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

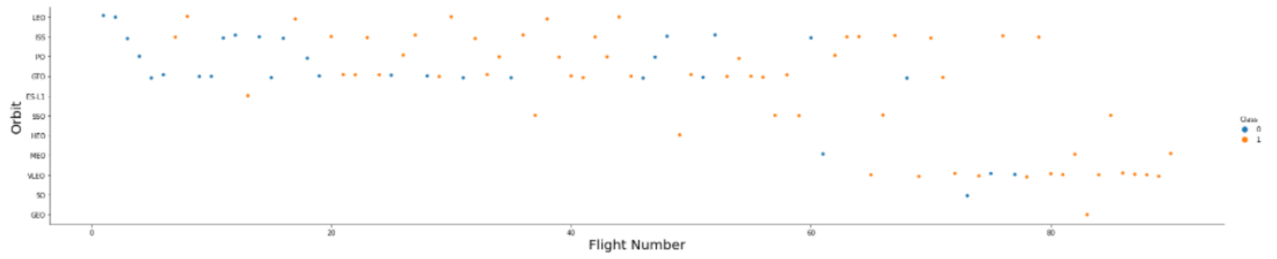




Success Rate vs. Orbit Type

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

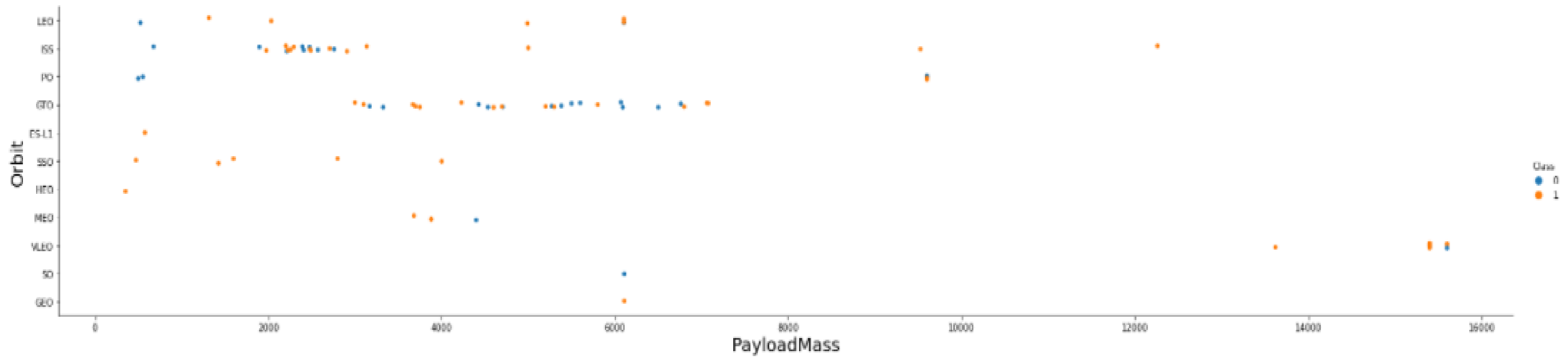
Flight Number vs. Orbit Type

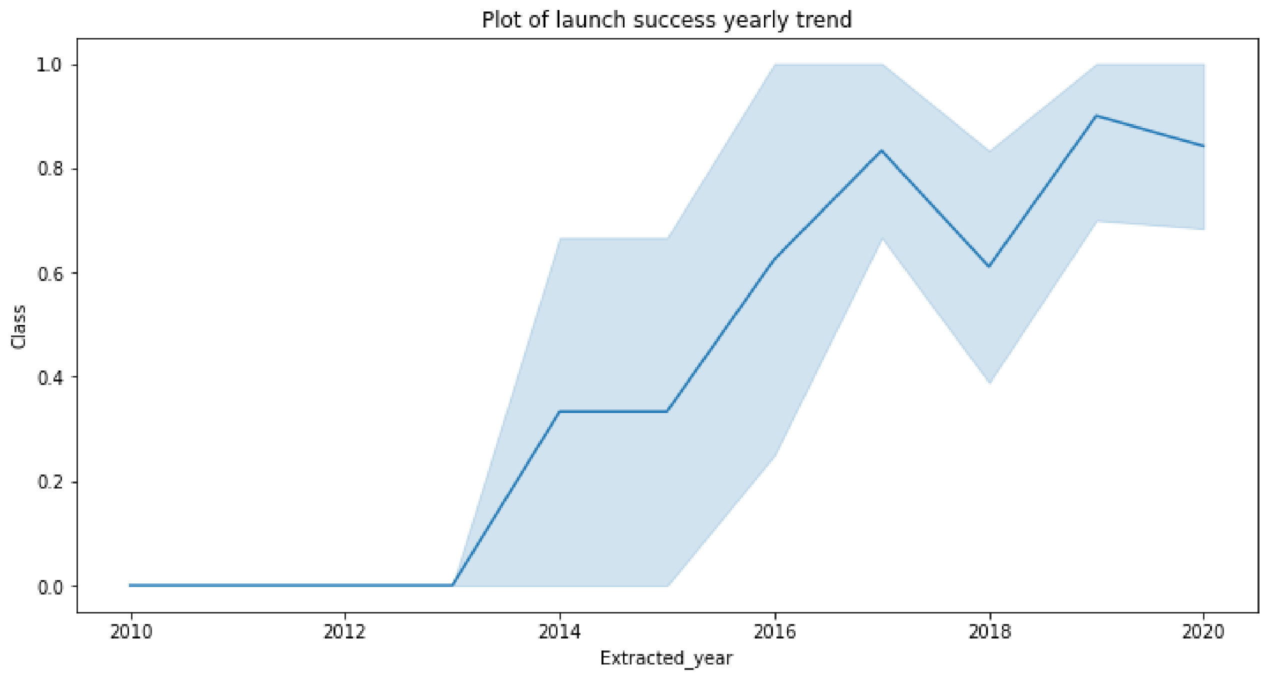


- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

Payload vs. Orbit Type

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

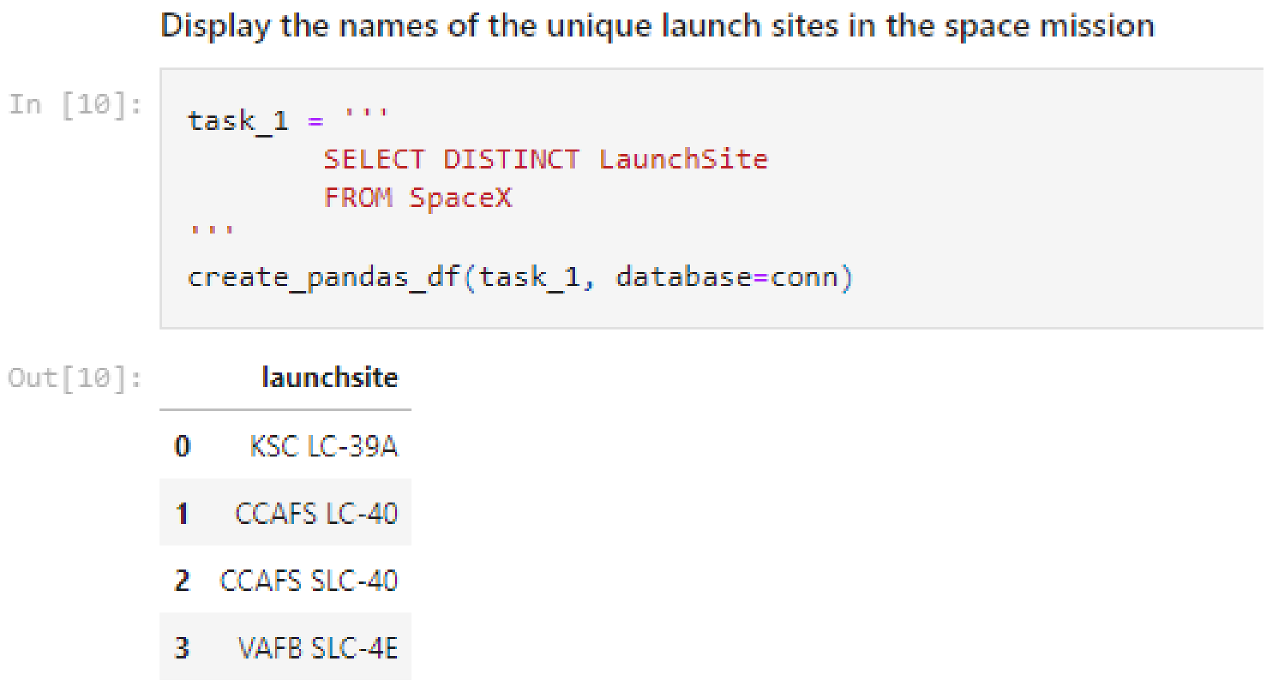




Launch Success Yearly Trend

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

All Launch Site Names



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

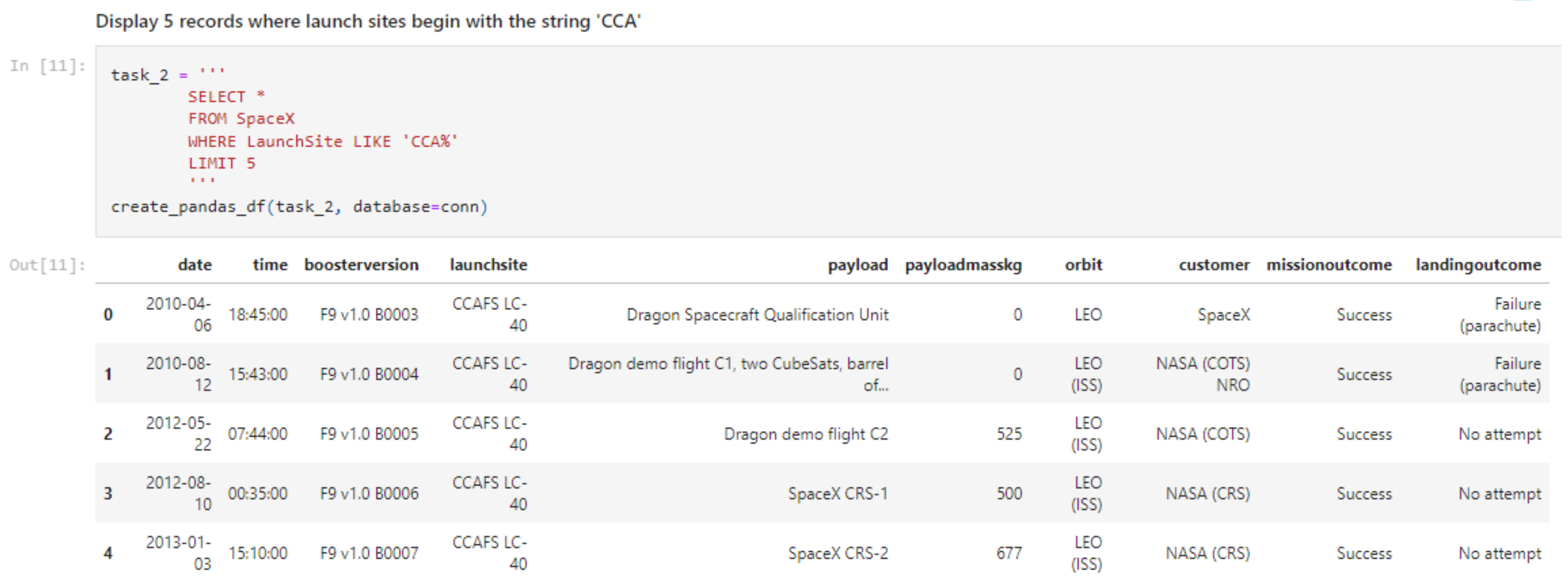
```
In [10]:
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

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)
```

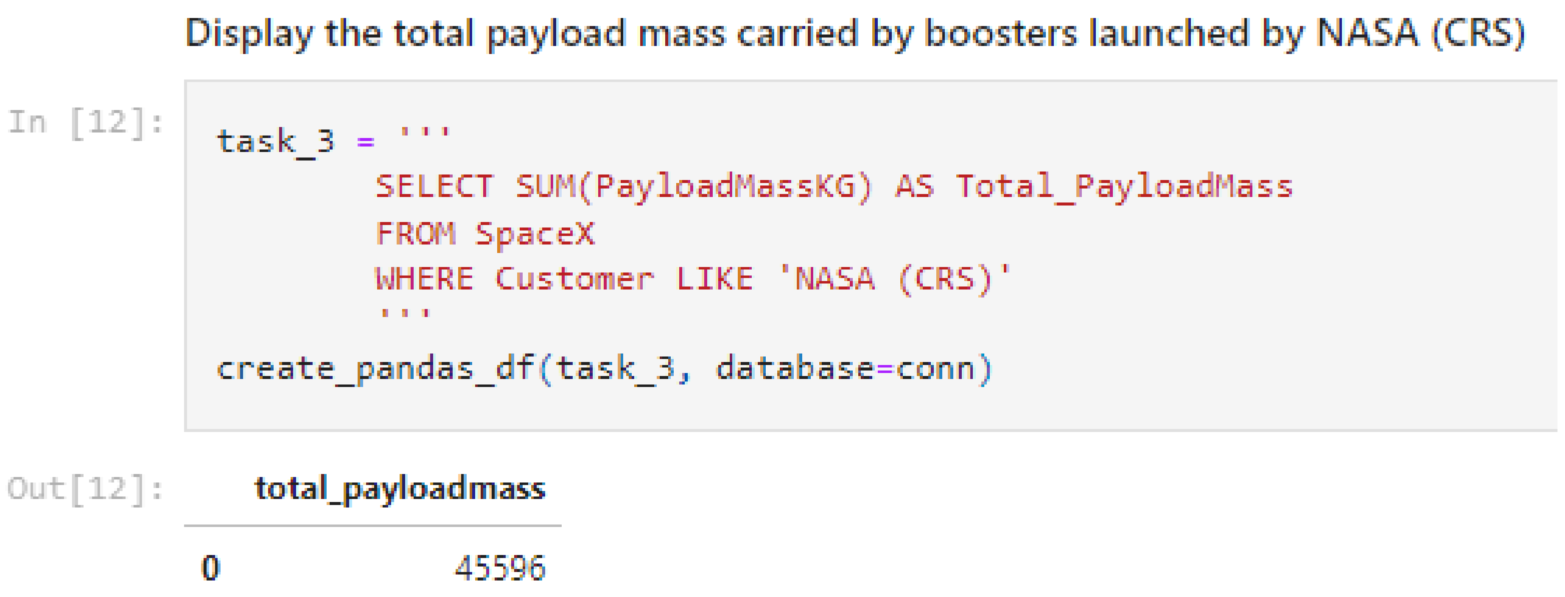
Out[11]:

	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

I calculated the total payload carried by boosters from NASA as 45596 using the query below

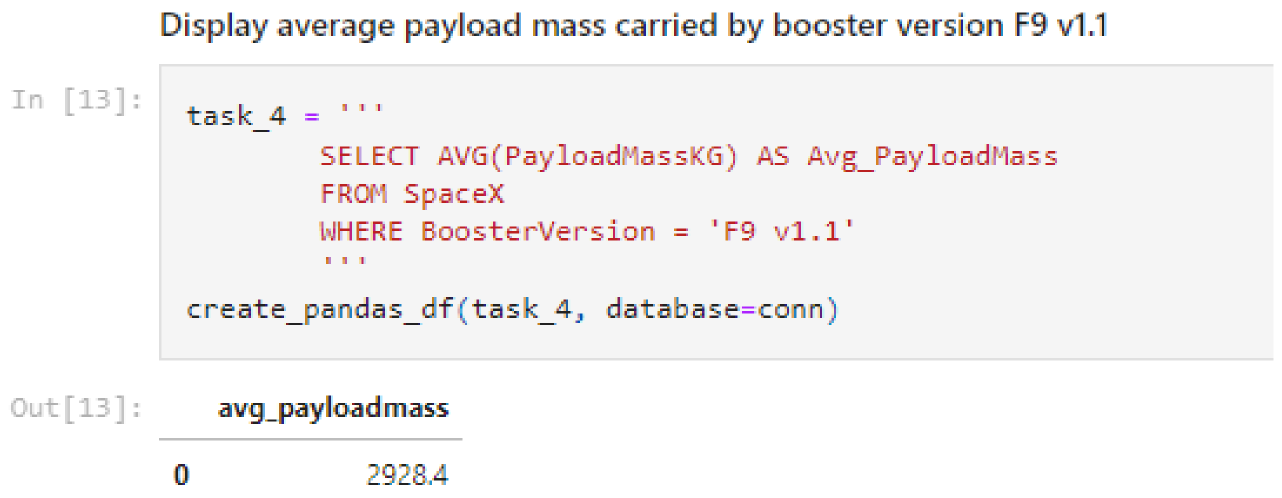


Display the total payload mass carried by boosters launched by NASA (CRS)

```
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
```

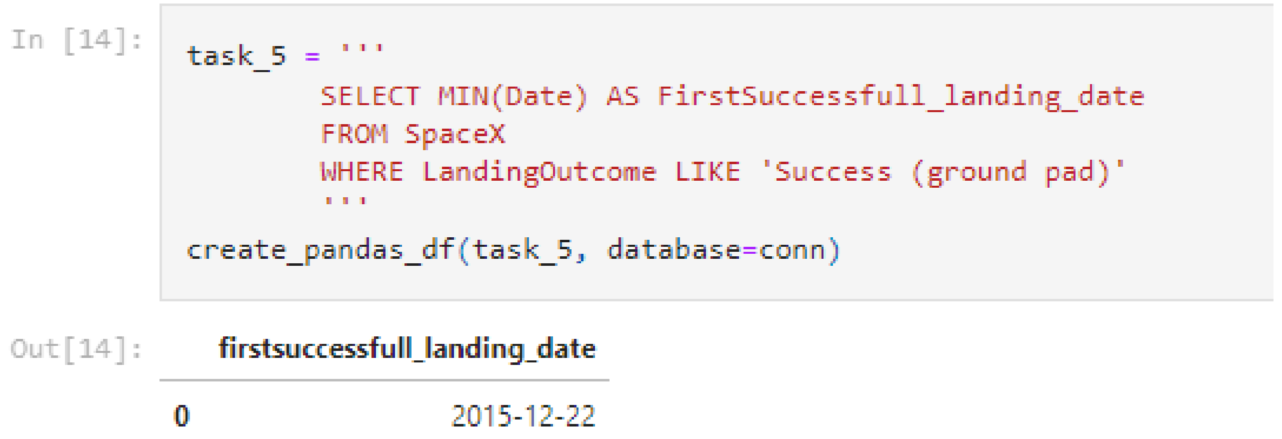
Average Payload Mass by F9 v1.1


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
```

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

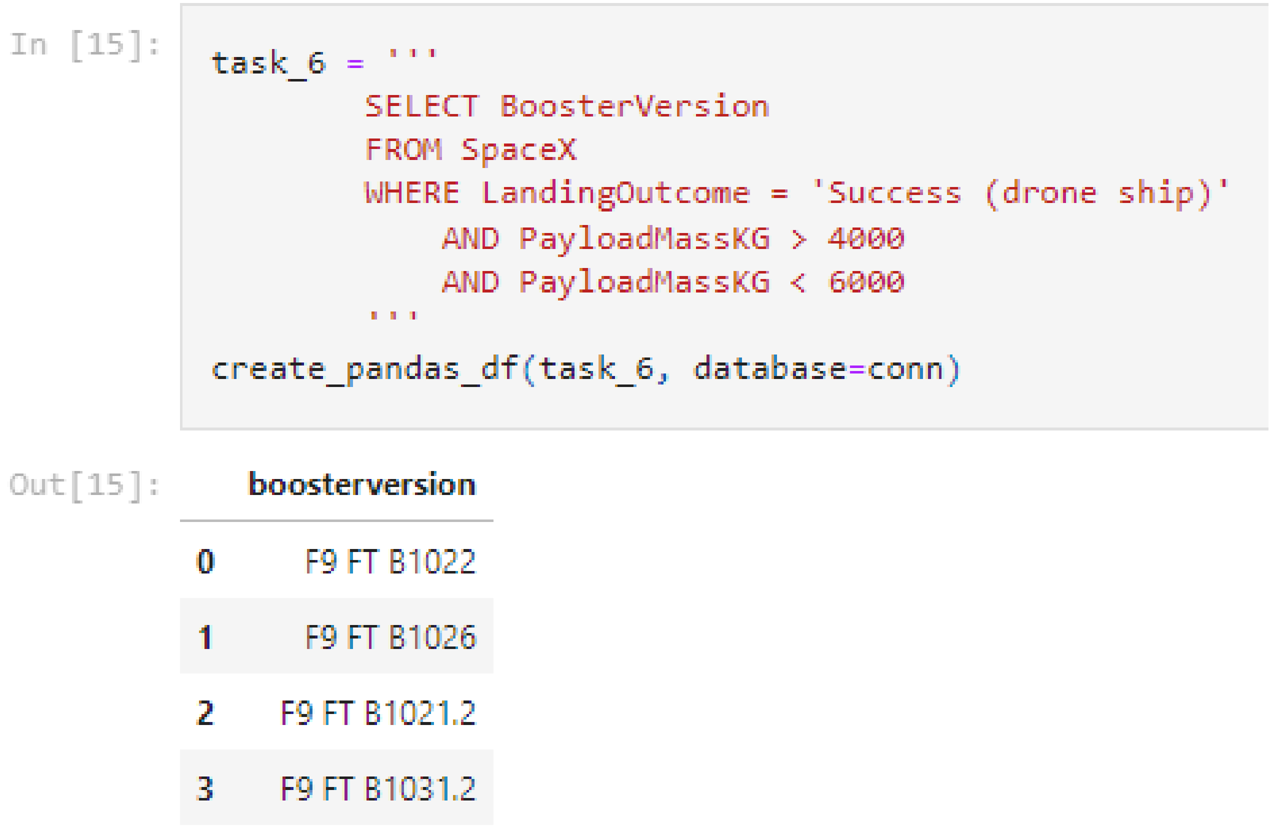
First Successful Ground Landing Date



```
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
```

- I 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

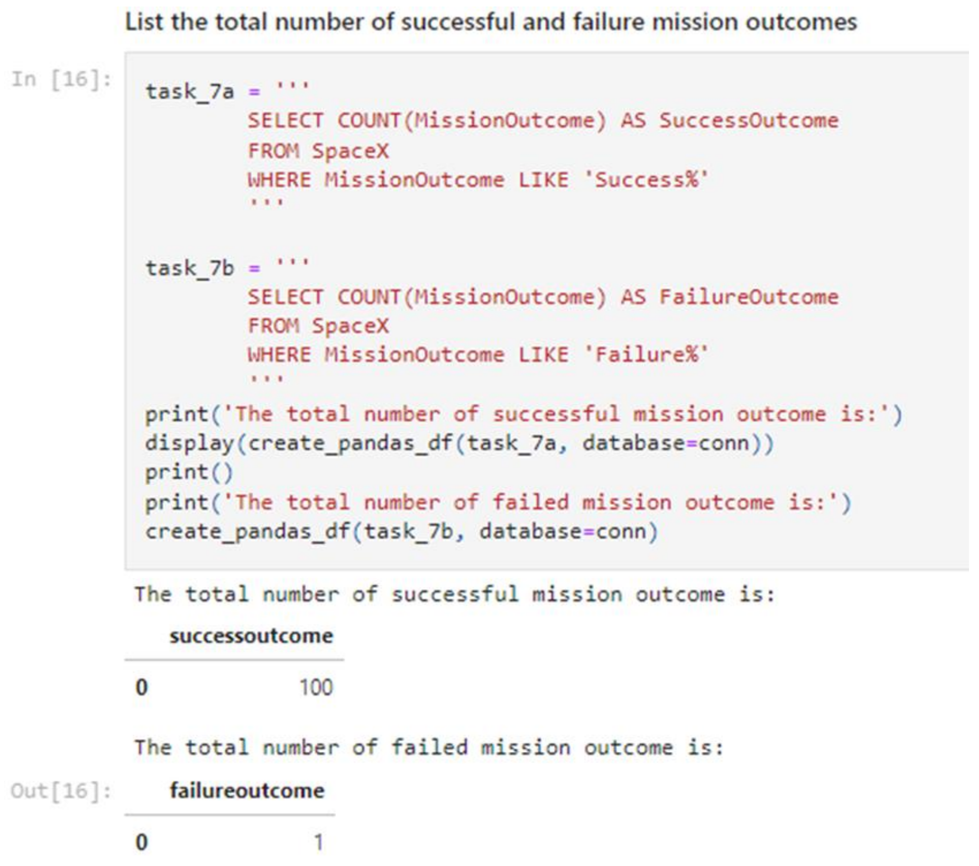


```
In [15]:  task_6 = '''
                  SELECT BoosterVersion
                  FROM SpaceX
                  WHERE LandingOutcome = 'Success (drone ship)'
                      AND PayloadMassKG > 4000
                      AND PayloadMassKG < 6000
                  '''
          create_pandas_df(task_6,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

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

- I 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


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

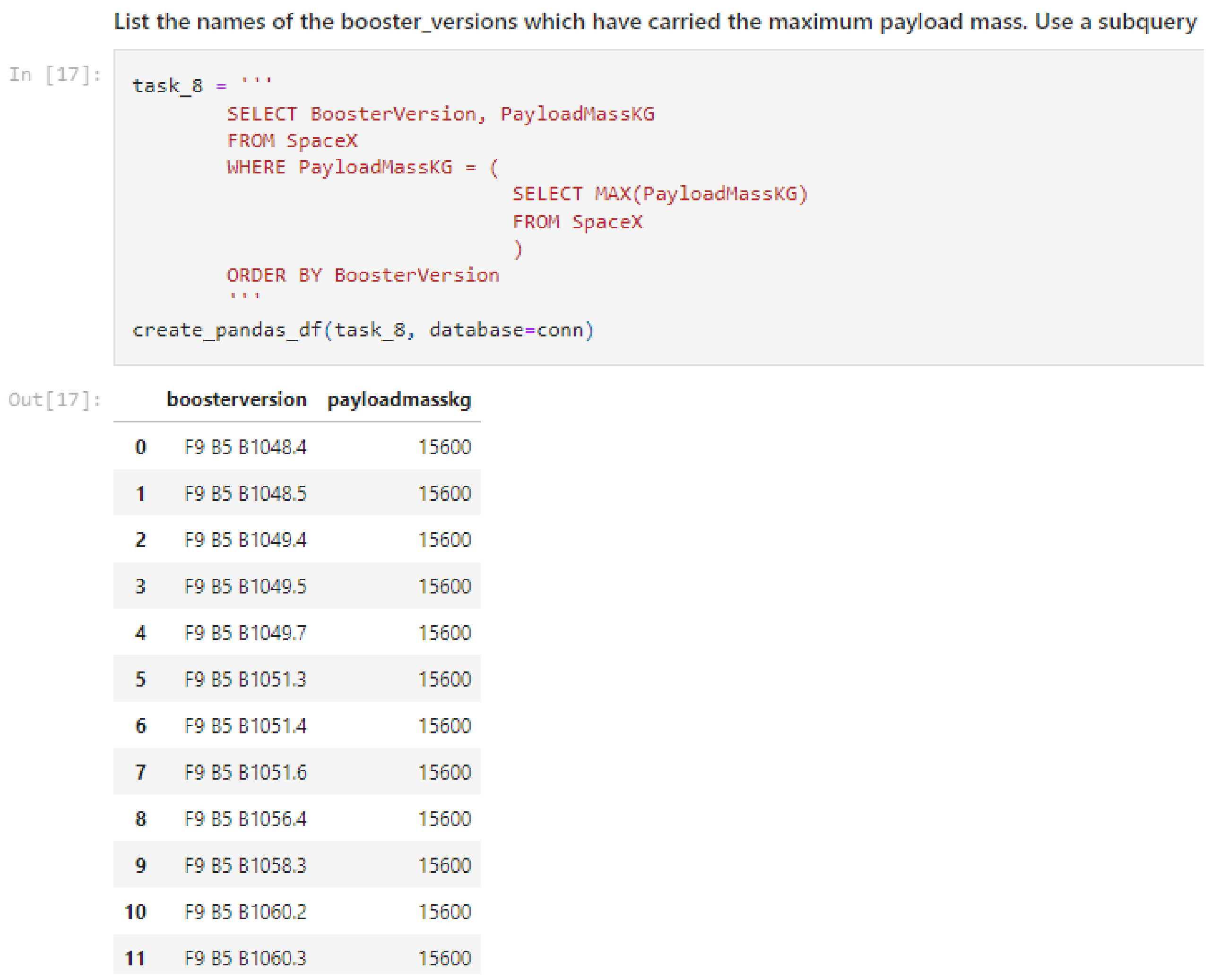
```
The total number of failed mission outcome is:
```

Out[16]:

	failureoutcome
0	1

Total Number of Successful and Failure Mission Outcomes

- I used wildcards like '%' to filter for **WHERE** MissionOutcome was a success or a failure.


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)
```

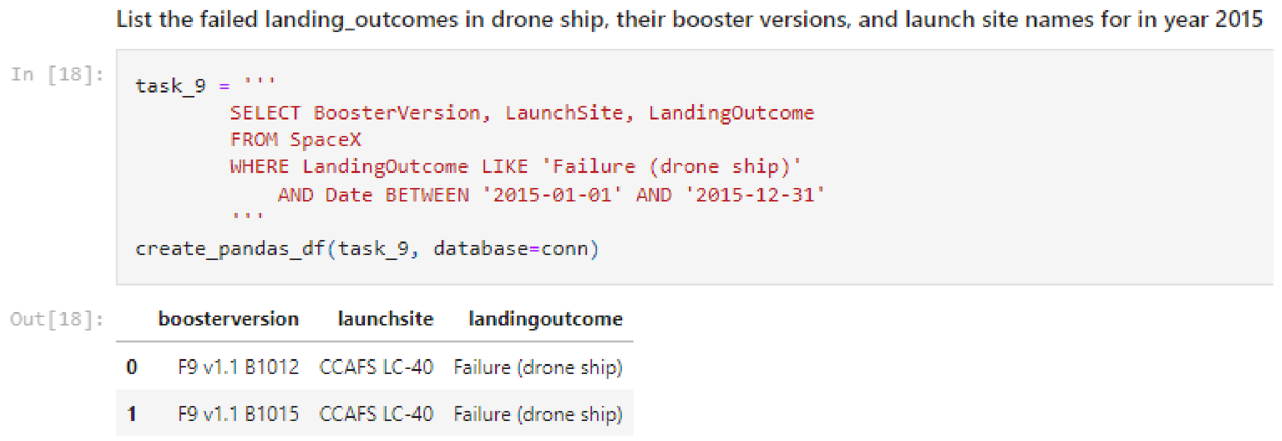
Out[17]:

	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

Boosters Carried Maximum Payload

- Determined 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

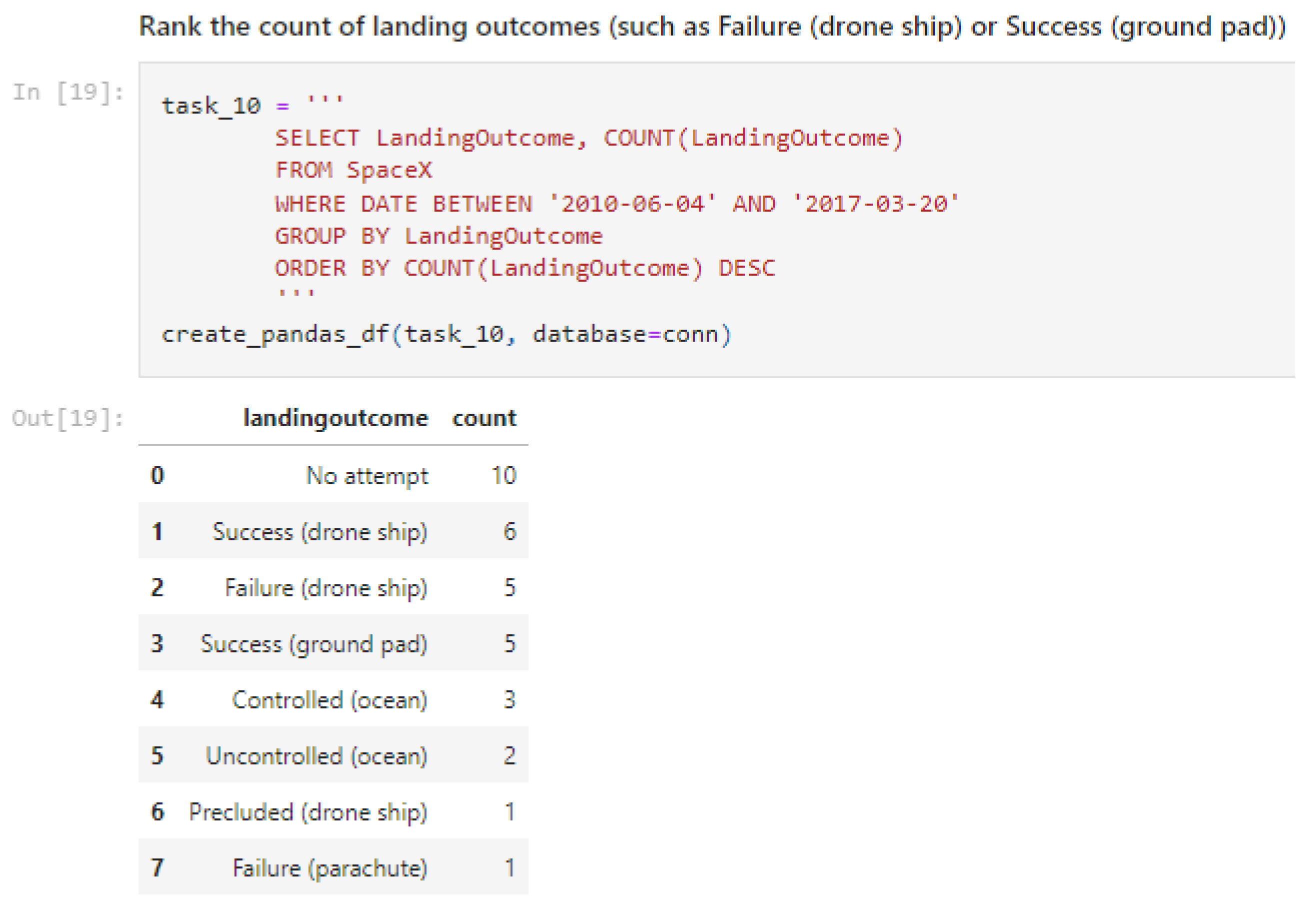
```
In [18]: task_9 = '''
                 SELECT BoosterVersion, LaunchSite, LandingOutcome
                 FROM SpaceX
                 WHERE LandingOutcome LIKE 'Failure (drone ship)'
                     AND Date BETWEEN '2015-01-01' AND '2015-12-31'
                 '''
         create_pandas_df(task_9,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



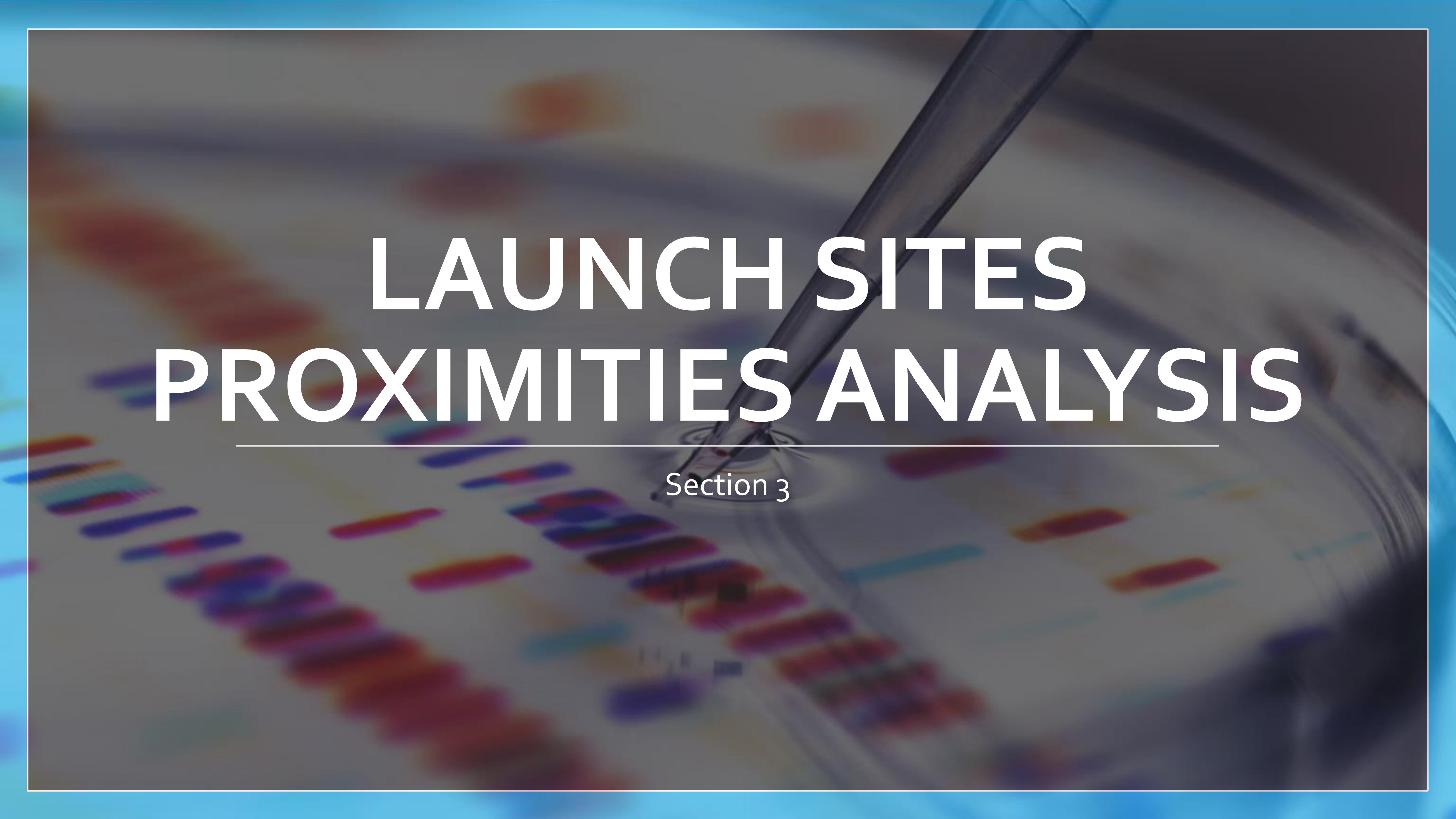
Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

```
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

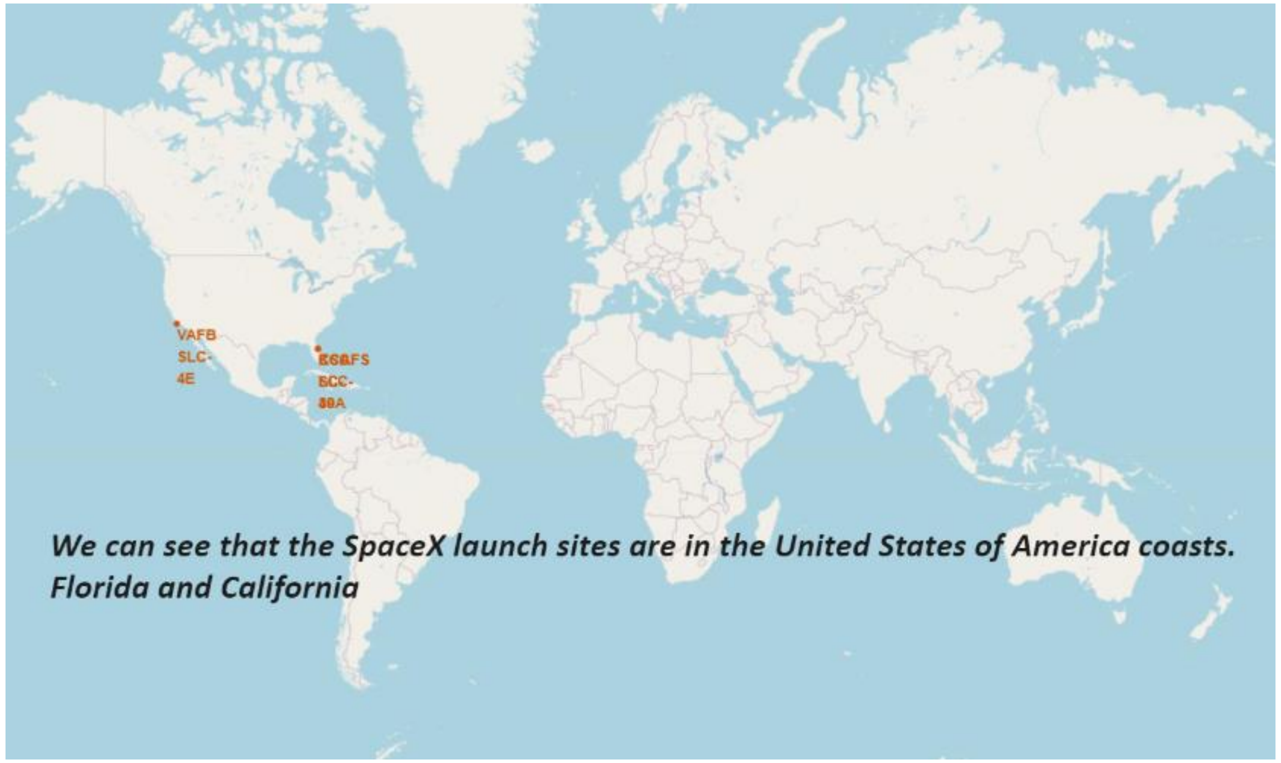
- I selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- I applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing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



LAUNCH SITES PROXIMITIES ANALYSIS

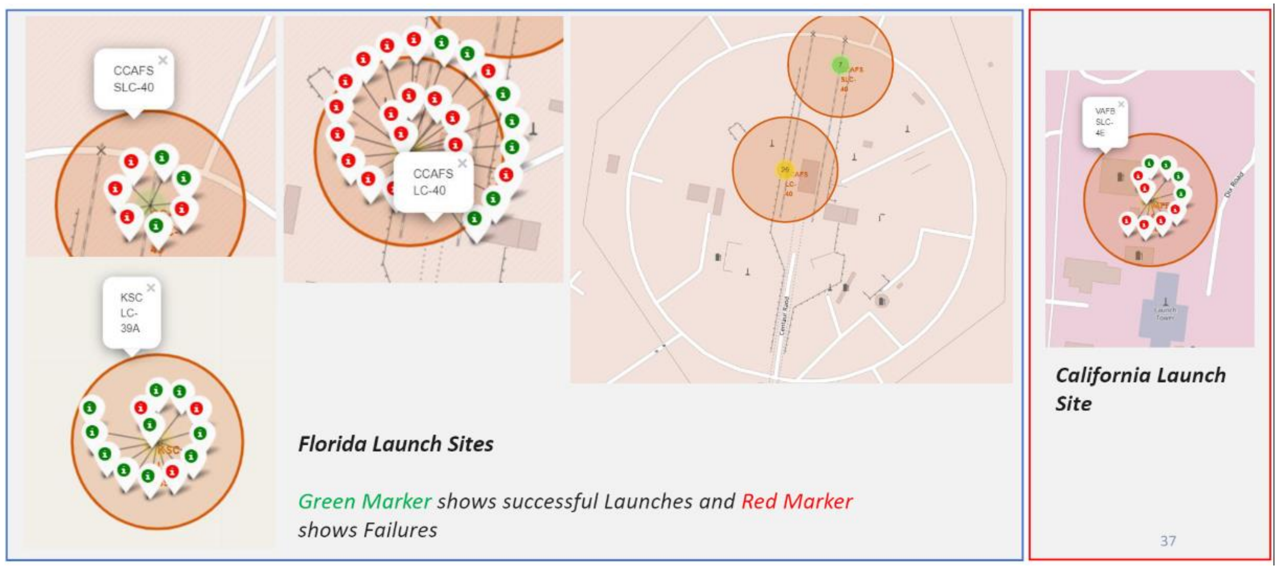
Section 3

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



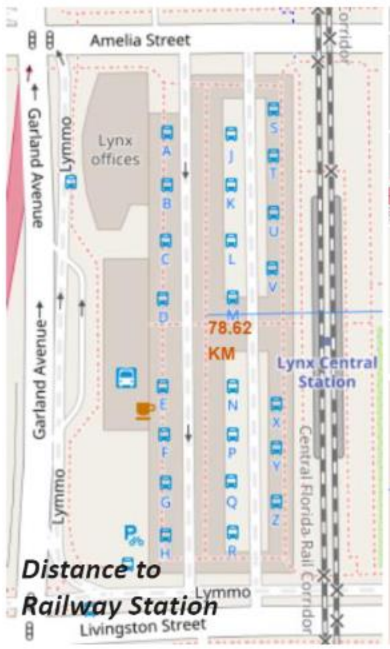
Florida Launch Sites

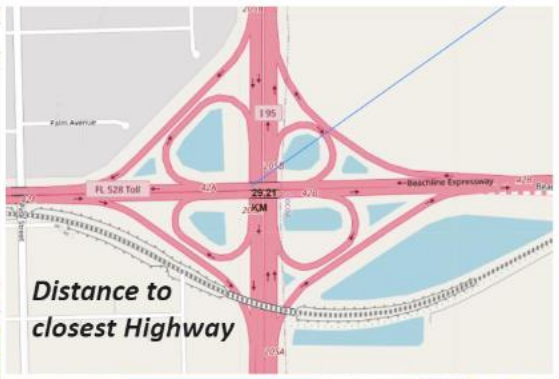
Green Marker shows successful Launches and Red Marker shows Failures

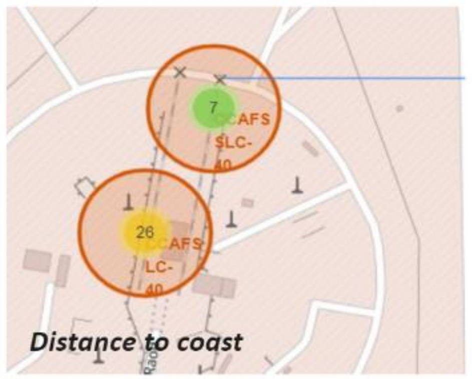
California Launch Site

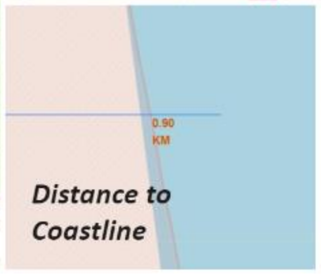
37

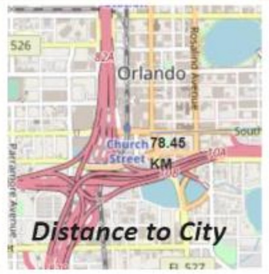
Launch Site distance to landmarks









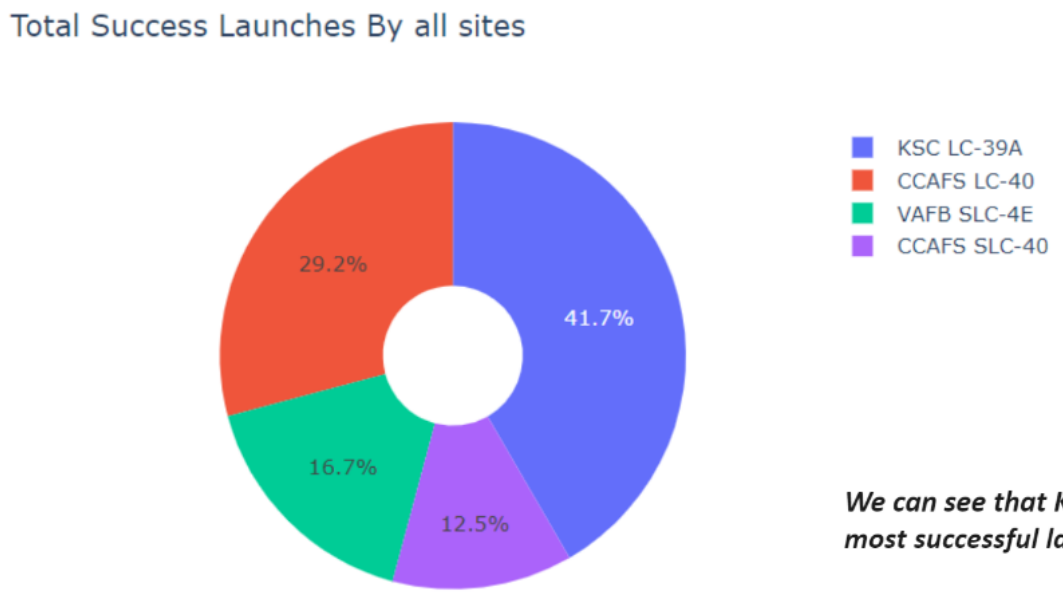


```
•Are launch sites in close proximity to
railways? No
•Are launch sites in close proximity to
highways? No
•Are launch sites in close proximity to
coastline? Yes
•Do launch sites keep certain distance
away from cities? Yes
```

BUILD A DASHBOARD WITH PLOTLY DASH

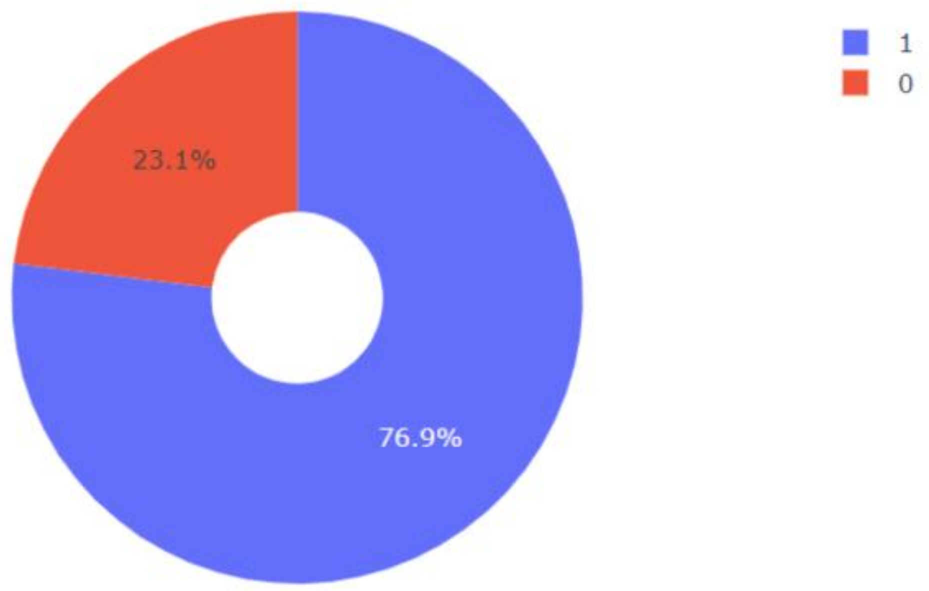
Section 4

Pie chart showing the success percentage achieved by each launch site



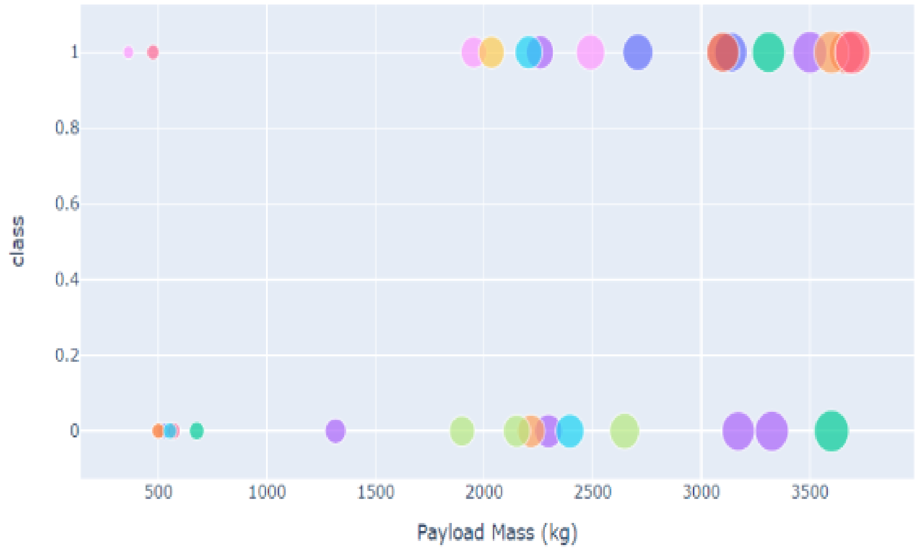
We can see that KSC LC-39A had the most successful launches from all the sites

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

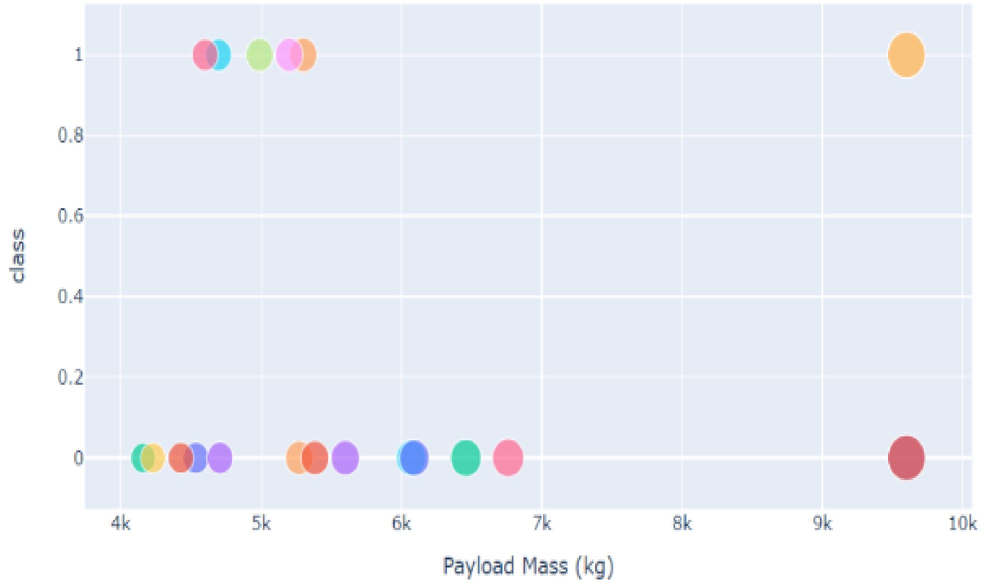


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg



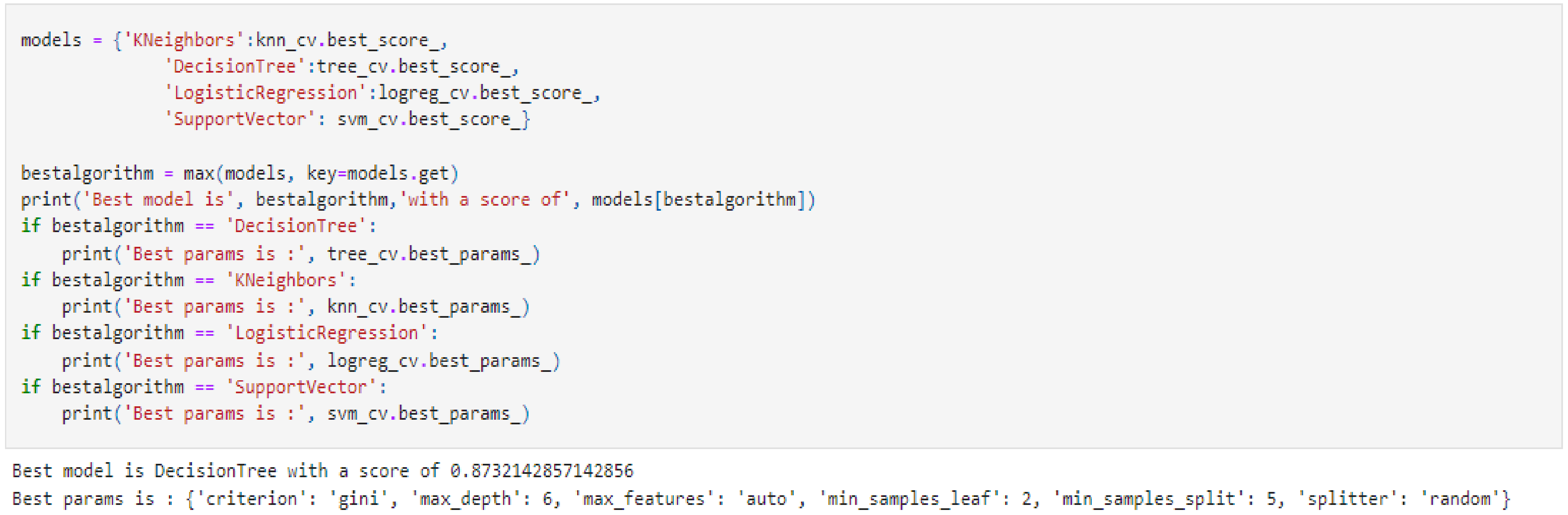
SCATTER PLOT OF PAYLOAD VS LAUNCH OUTCOME FOR ALL SITES, WITH DIFFERENT PAYLOAD SELECTED IN THE RANGE SLIDER

PREDICTIVE ANALYSIS(CLASSIFICATION)

Section 5

Classification Accuracy

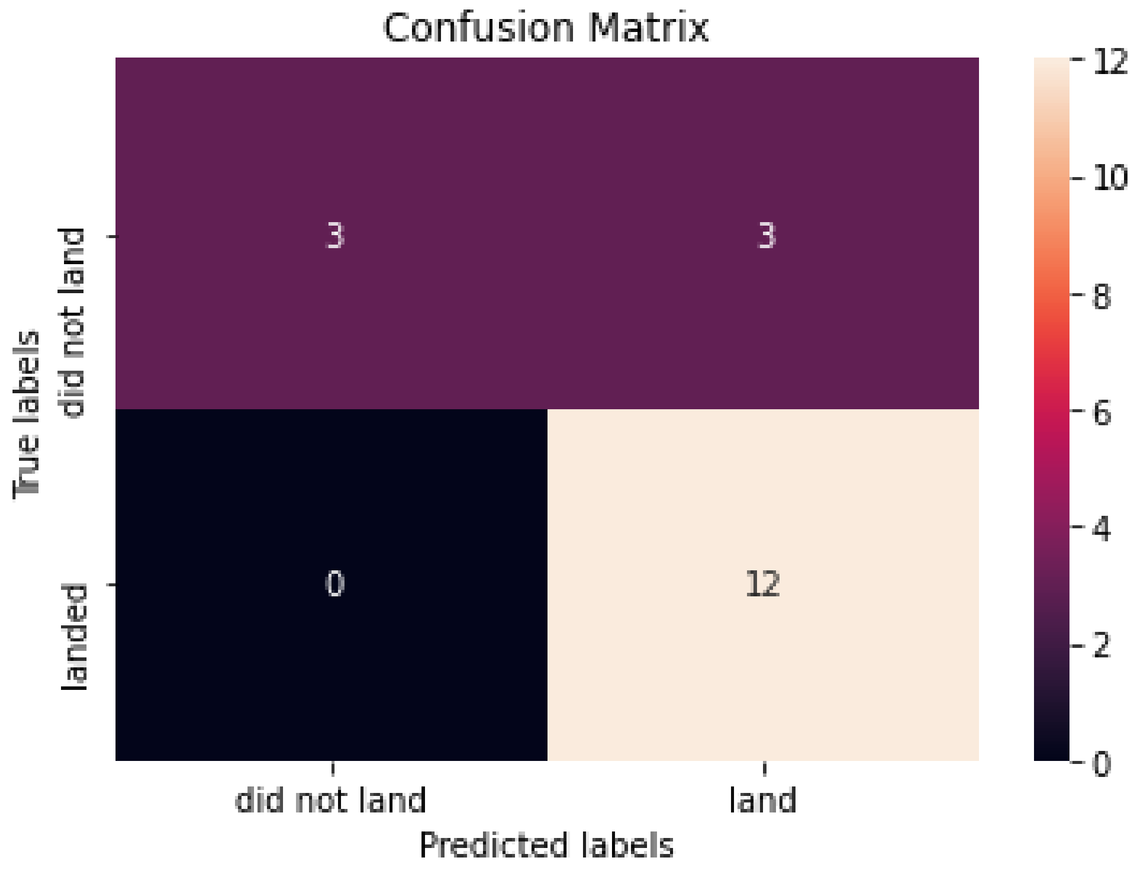
- The decision tree classifier is the model with the highest classification accuracy



```
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.

THANK YOU!