IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Alexa Gal
March 7, 2022

# Outline

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

# Executive Summary

- Summary of methodologies
  - Data Collection using API
  - Data Collection using 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
  - Interactive analytics
  - Predictive analytics

# Introduction

- Project background and context
  - SpaceX has significantly lower costs (62million USD) of launching a rocket compared to its competitors (up to 165million USD). The major advantage of SpaceX is the reuse of the first stage. We can calculate the cost of a launch by determining if the first stage is successful. This information helps if there is competition against SpaceX for bidding for a contract.
- Problems you want to find answers
  - What are the contributing factors to successfully land a rocket?
  - How do these factors contribute to the success rate of landing a rocket?
  - What are the factors of a successful rocket landing?

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Collected data using SpaceX API and Web Scraping from Wikipedia
- Perform data wrangling
  - Applied one-hot encoding to categorical features
- Perform 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

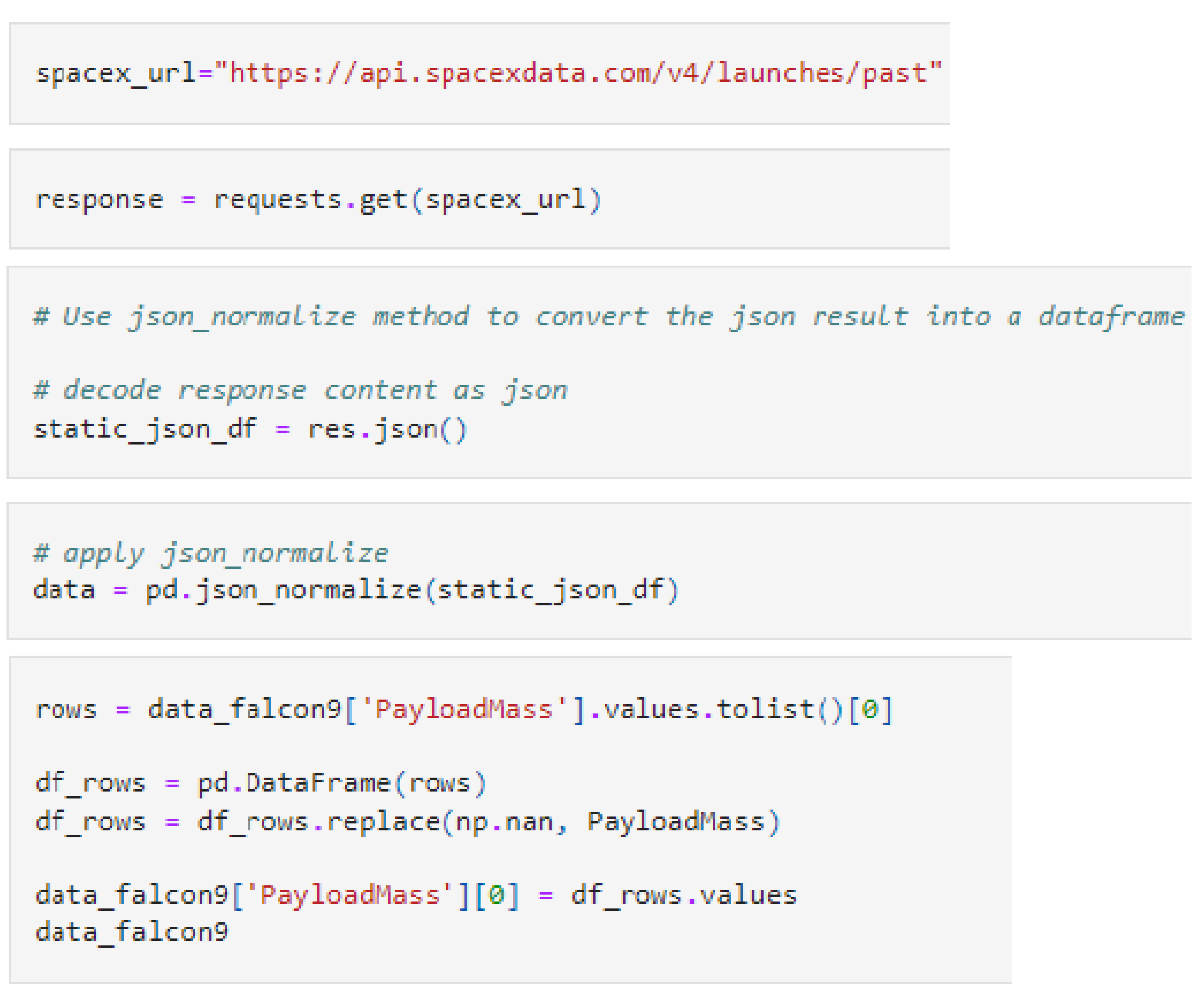
- Collected data sets using:
- Get request to SpaceX API
- .json() function call turned into pandas dataframe using j.son_normalize()
- Cleaned data, checked for and filled in missing values as needed
- Used BeautifulSoup to web scrape from Wikipedia
- Extracted launch records, parsed the HTML table and converted into pandas dataframe

# Data Collection – SpaceX API

- Made a get request to the SpaceX API, then cleaned the data. Then, did some basic data wrangling and formatting.

GitHub URL:

- https://github.com/fridgeraider1/FinalProject/blob/9ab2403eb700d7b9c772abaa9ba38e52aac263e6/Week%201/API.ipynb



```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

```
# Use json_normalize method to convert the json result into a dataframe

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

```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

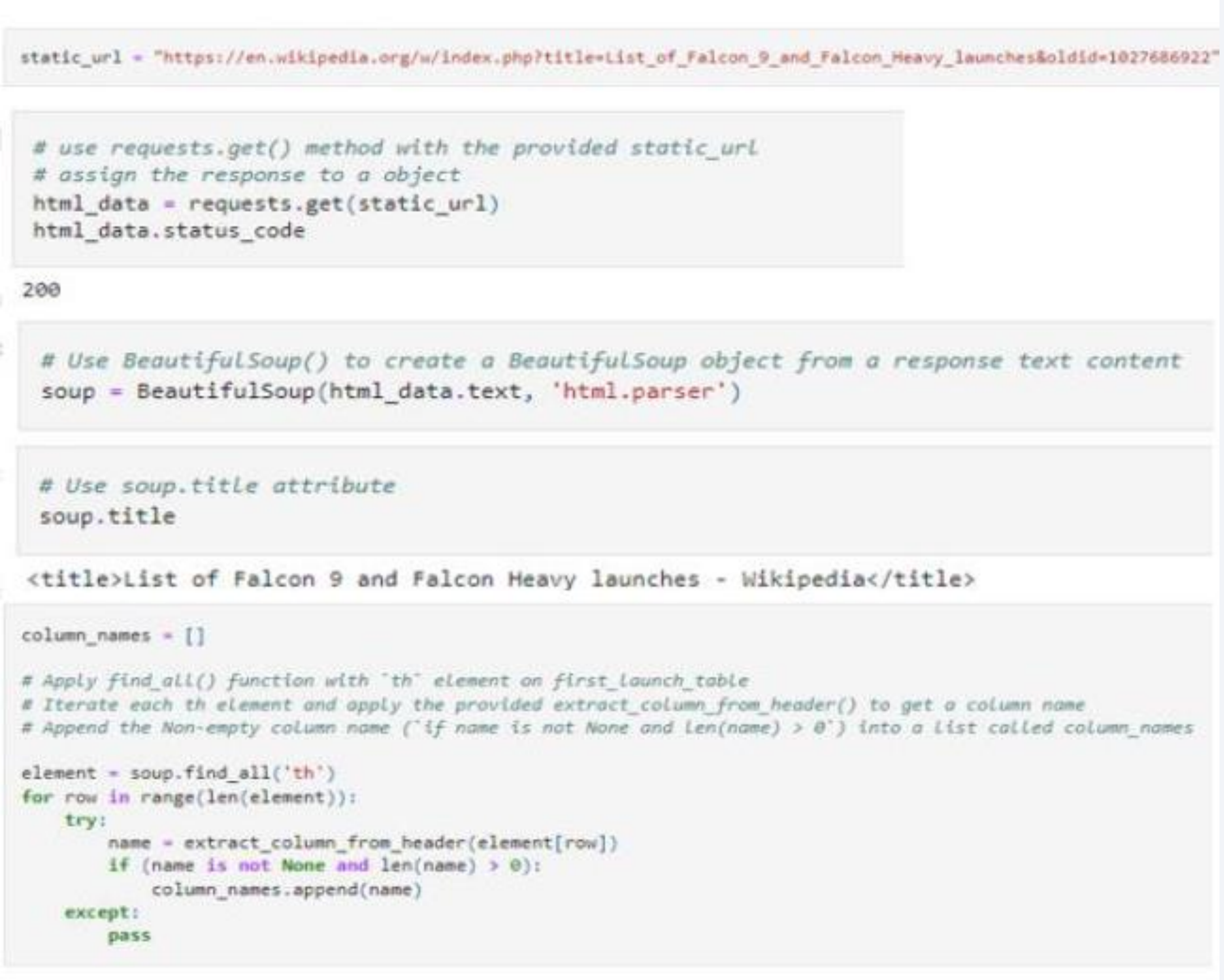
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

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

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

# Data Collection - Scraping

- Used BeautifulSoup web scraping to extract a Falcon 9 launch records HTML table from Wikipedia. Then, parsed the table and converted it into a Pandas data frame.

GitHub URL:

- https://github.com/fridgeraider1/FinalProject/blob/a98bbd3a9e017b7a3333f7fce8a103429cd0b538/Web%20Scraping.ipynb



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

```
# 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
```

```
200
```

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

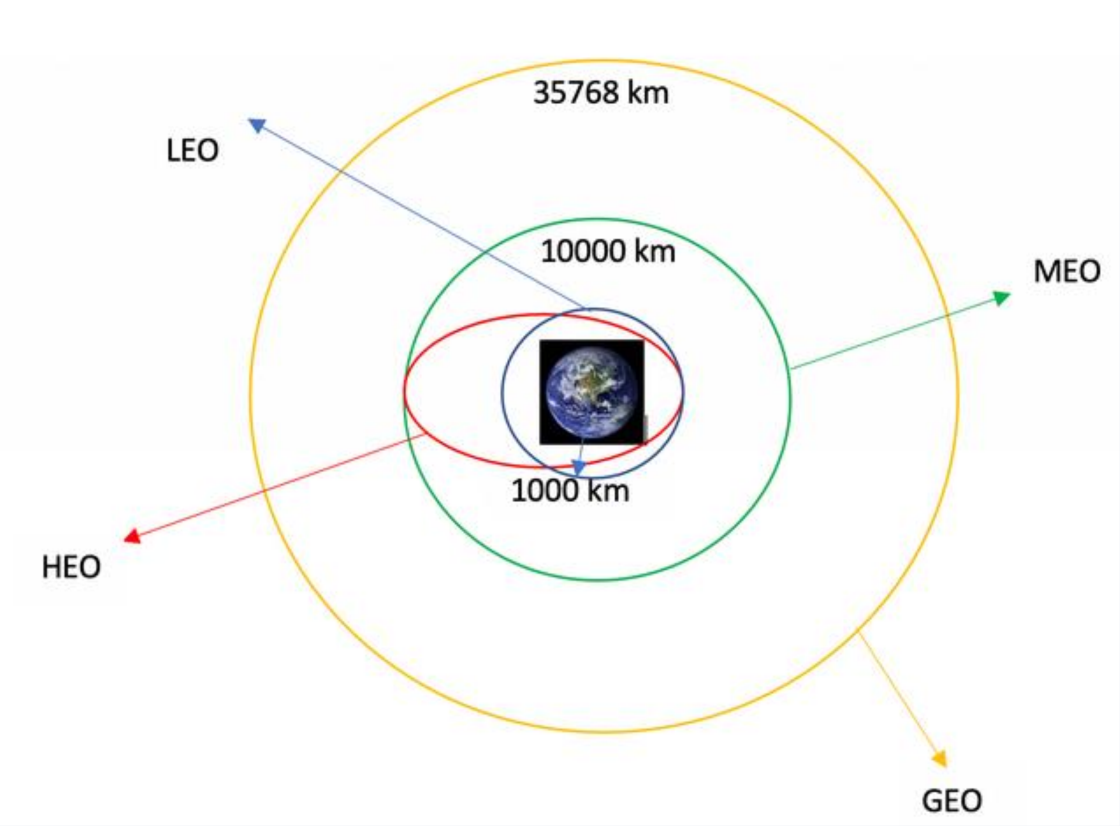
```
# Use soup.title attribute
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

```
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
```

# Data Wrangling

- Performed exploratory data analysis and determined training labels. Then, calculated the number of occurrence of each orbit and the number of occurrence of mission outcome per orbit type. Then, created a landing outcome label from Outcome column.



Add the GitHub URL :

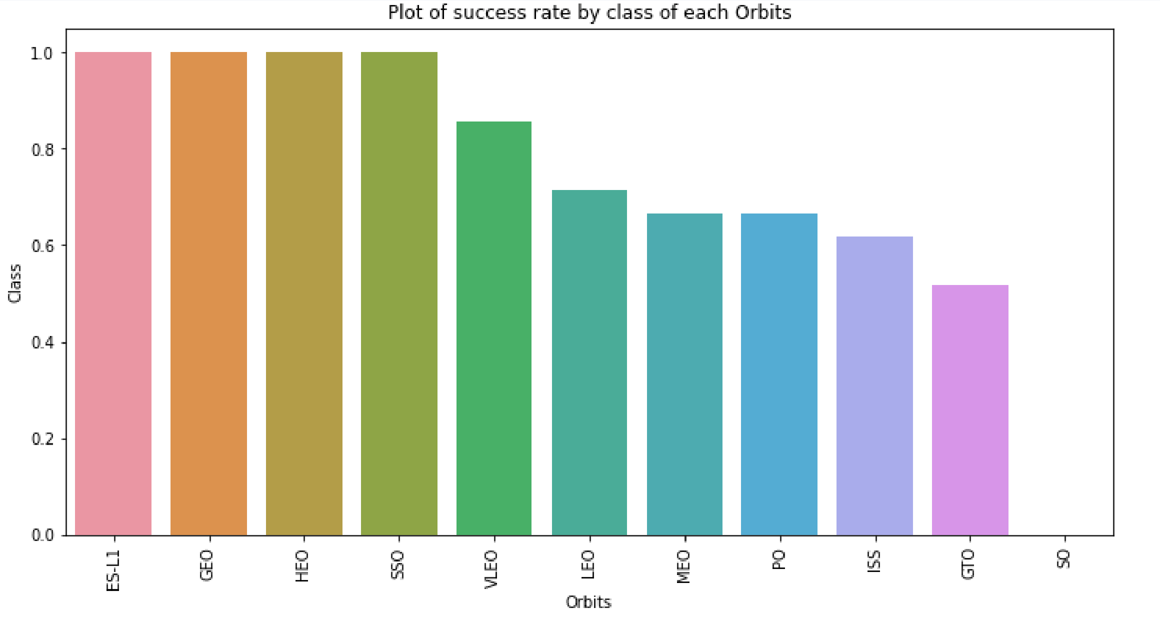
- [https://github.com/fridgeraider1/FinalProject/blob/a98bbd3a9e017b7a3333f7fce8a103429cd0b538/Wrangling.ipynb](https://github.com/fridgeraider1/FinalProject/blob/a98bbd3a9e017b7a3333f7fce8a103429cd0b538/Wrangling.ipynb)

# EDA with Data Visualization

- Plotted a bar chart to try to find which orbits have high success rate.

GitHub URL:

- https://github.com/fridgeraider1/FinalProject/blob/a98bbd3a9e017b7a3333f7fce8a103429cd0b538/Data%20Visualization.ipynb



# EDA with SQL

- Loaded the dataset into the corresponding table in a Db2 database using PostgreSQL. Then created a table, displayed unique launch sites in the space mission with the string 'CCA', displayed total payload mass carries by boosters launched by NASA and average payload mass carried by booster version F9. V1.1. Then, listed date of first successful landing outcome in ground pad was achieved. Listed names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000. Listed the total number of successful and failure mission outcomes. Using a subquery, listed names of the booster_versions with max payload mass and listed failed landing_outcomes in drone ship for year 2015. Finally, ranked the count of landing outcomes in descending order.

GitHub URL:

- https://github.com/fridgeraider1/FinalProject/blob/a98bbd3a9e017b7a3333f7fce8a103429cd0b538/SQL.ipynb

# Build an Interactive Map with Folium

- Marked all launch sites on the map along with the success or failed launches for each site on the map to see which sites have high success rate. Then, calculated the distances between a launch site to its proximities.
- GitHub URL:
- https://github.com/fridgeraider1/FinalProject/blob/a98bbd3a9e017b7a3333f7fce8a103429cd0b538/Launch%20Sites%20Locations.ipynb

# Build a Dashboard with Plotly Dash

- Built an interactive dashboard using Plotly Dash
- Plotted pie charts to show the total launches of sites
- Plotted scatter graph to show the relationship between outcome and payload mass for different booster versions.

# Predictive Analysis (Classification)

- Used NumPy and pandas to create a column for the class, standardize the data in X and used function_train_test_split to split into training data and test data. Then, used logistic regression.Then, created decision tree classifier object to find best parameters. Used K nearest neighbors object to find best performing classification model.

- GitHub URL:
- https://github.com/fridgeraider1/FinalProject/blob/a98bbd3a9e017b7a3333f7fce8a103429cd0b538/Machine%20Learning%20Prediction.ipynb

# Results

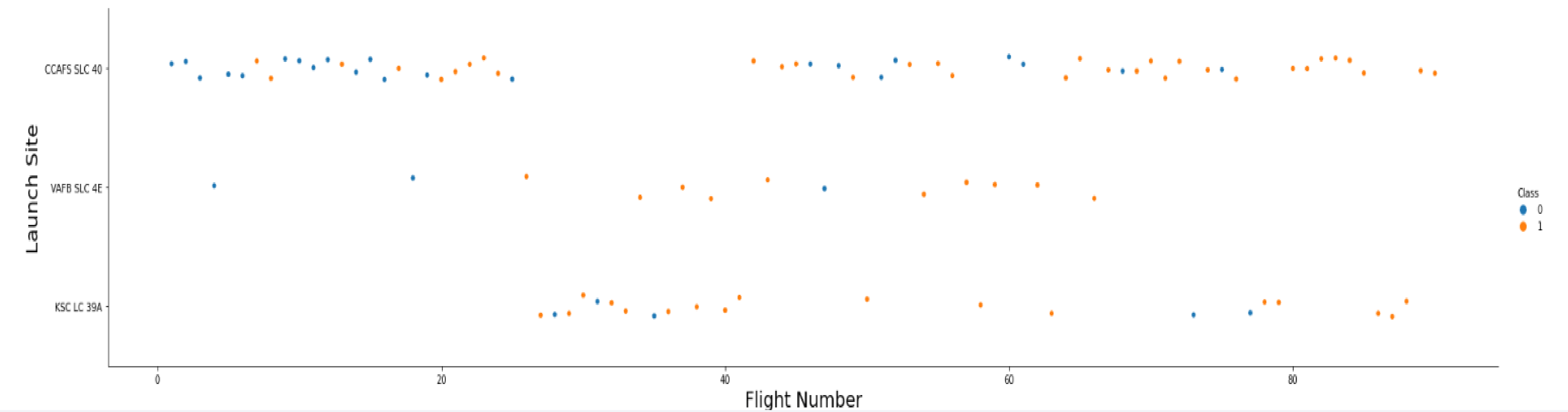
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

# Insights drawn from EDA

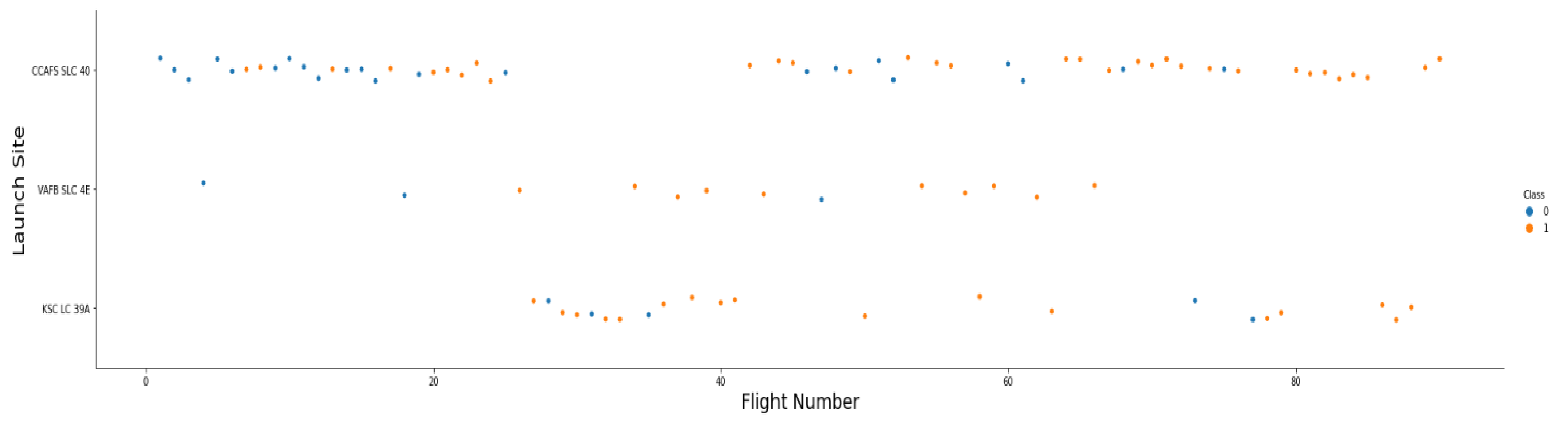
# Flight Number vs. Launch Site

- The success rate at a launch site directly correlates with the flight amount at a launch site. The higher the flight amount, the higher success rate is.



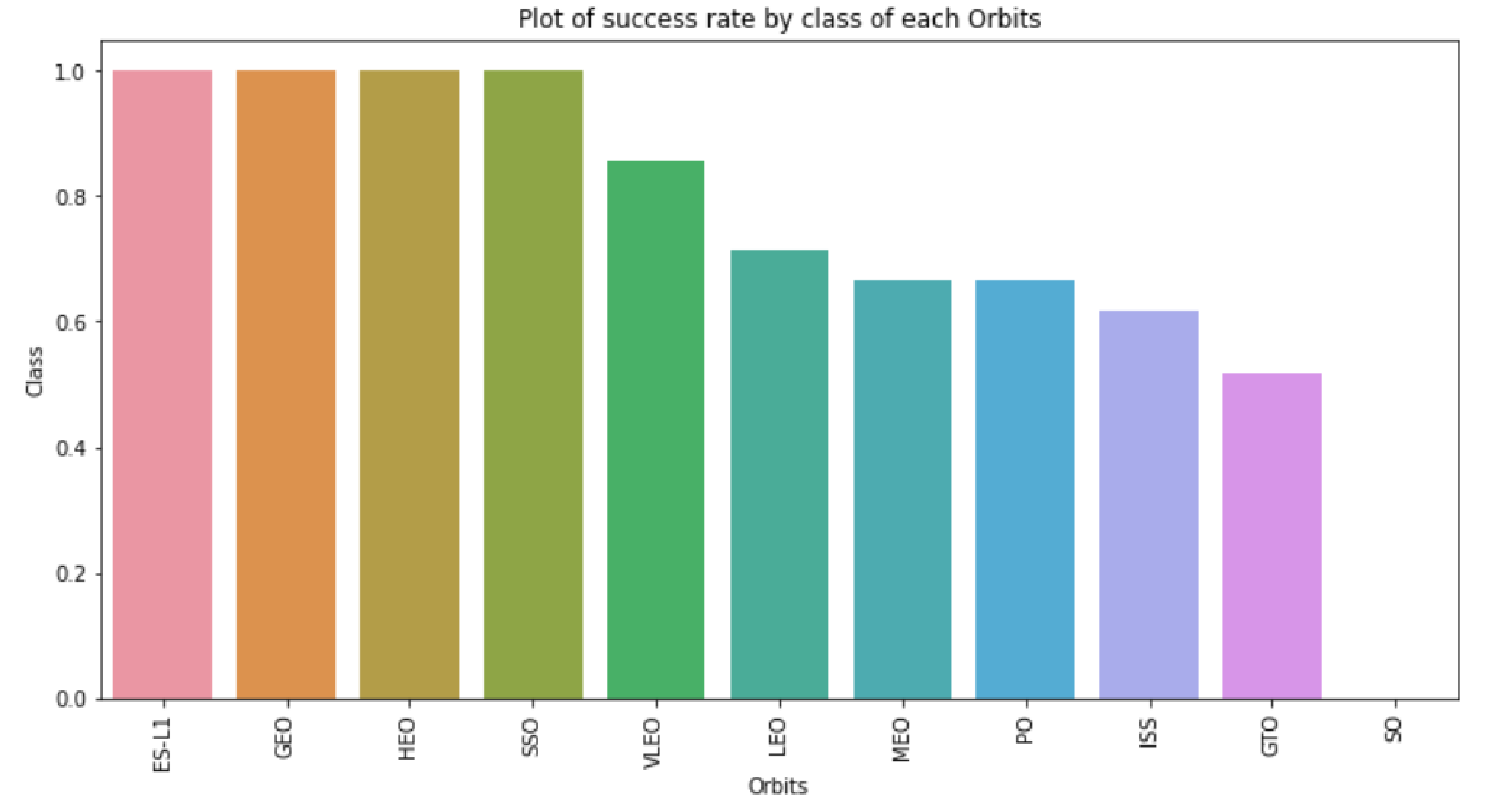
# Payload vs. Launch Site

- As the payload mass increases, the success rate increases too.



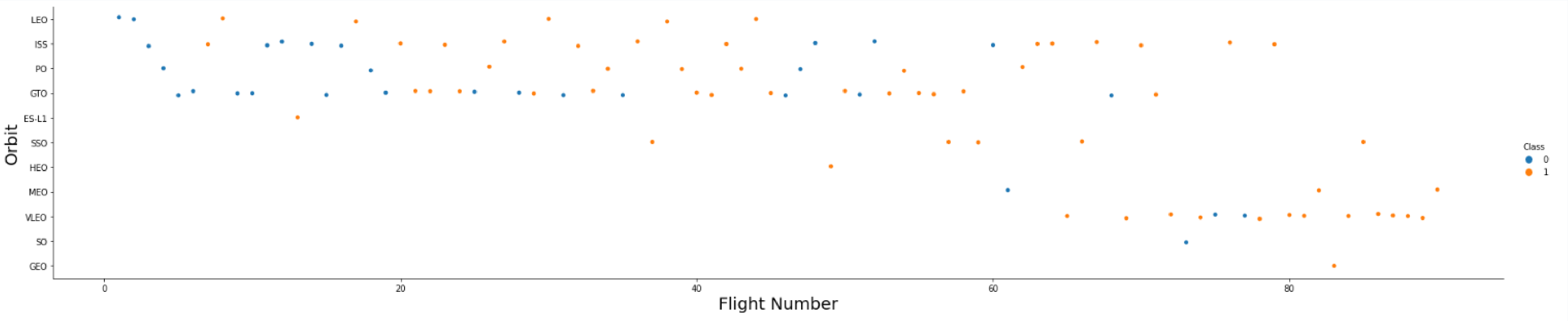
# Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO, and VLEO are the Orbits that have high success rate. The SO has the least success rate amongst the orbits.



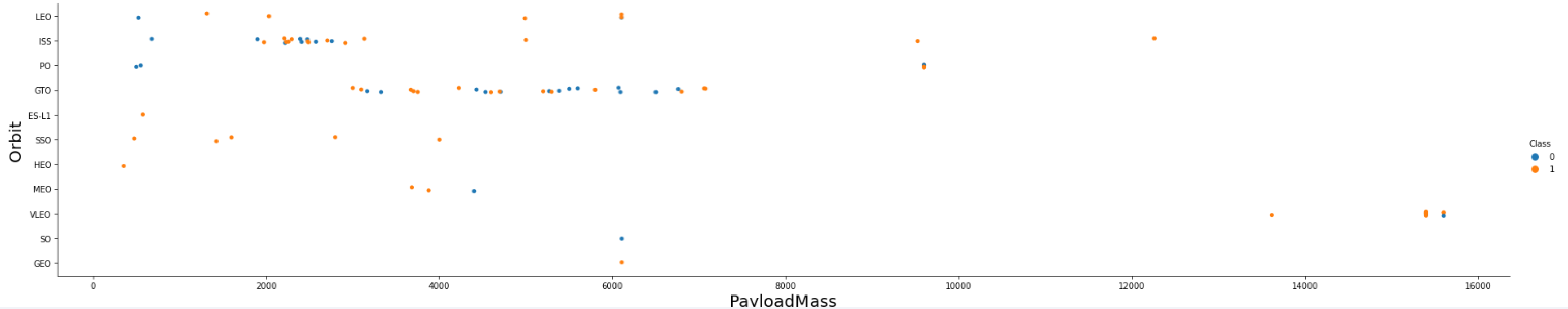
# Flight Number vs. Orbit Type

Success appears related to the number of flights in the LEO orbit. No relationship between flight number when in GTO orbit.



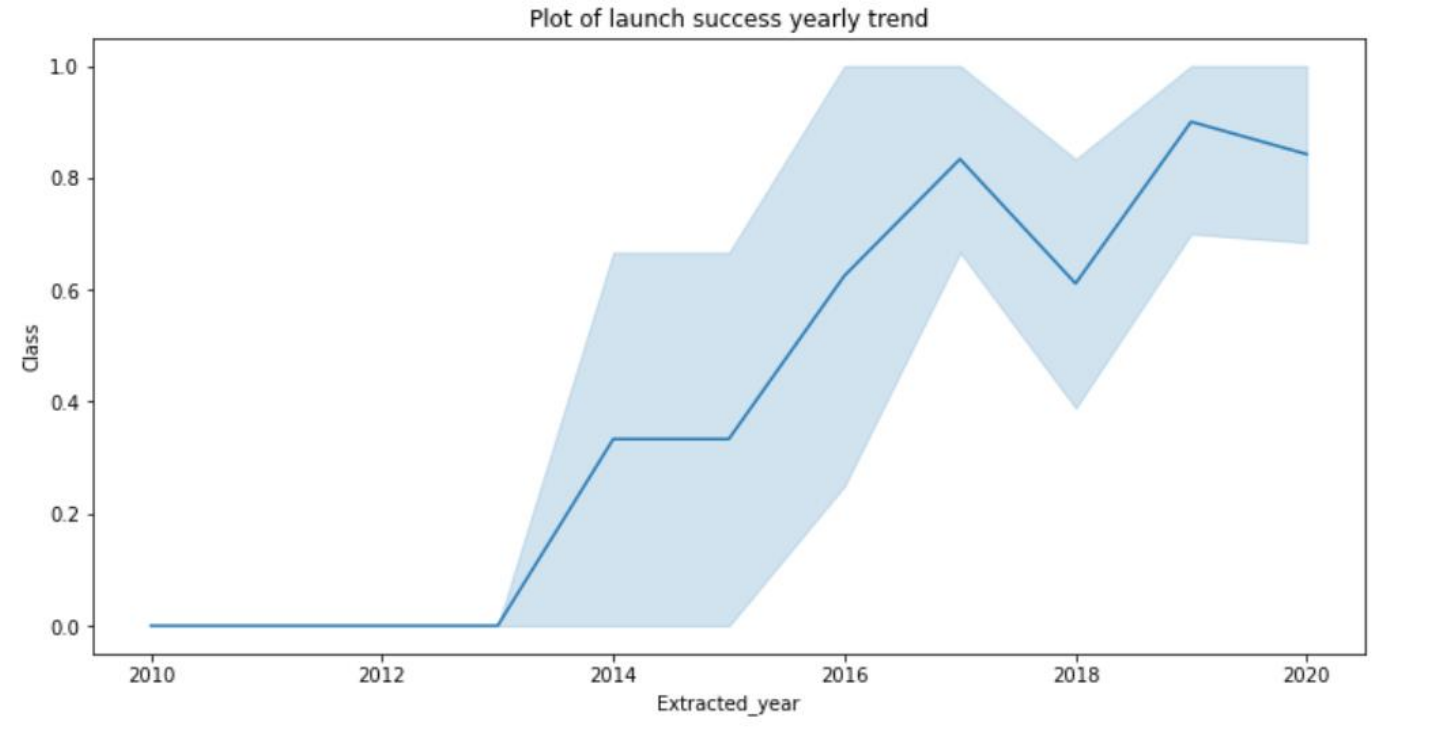
# Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- Can't say the same for GTO as both positive landing rate and negative landing (unsuccessful mission) are both present.



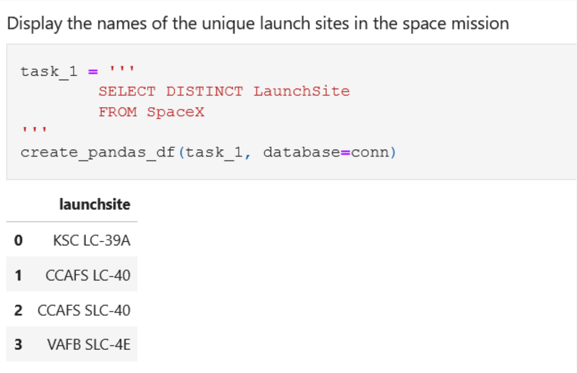
# Launch Success Yearly Trend

- Success rate since 2013 kept increasing till 2020.



# All Launch Site Names

- Used SELECT DISTINCT to display the names of the unique launch sites in the space mission.



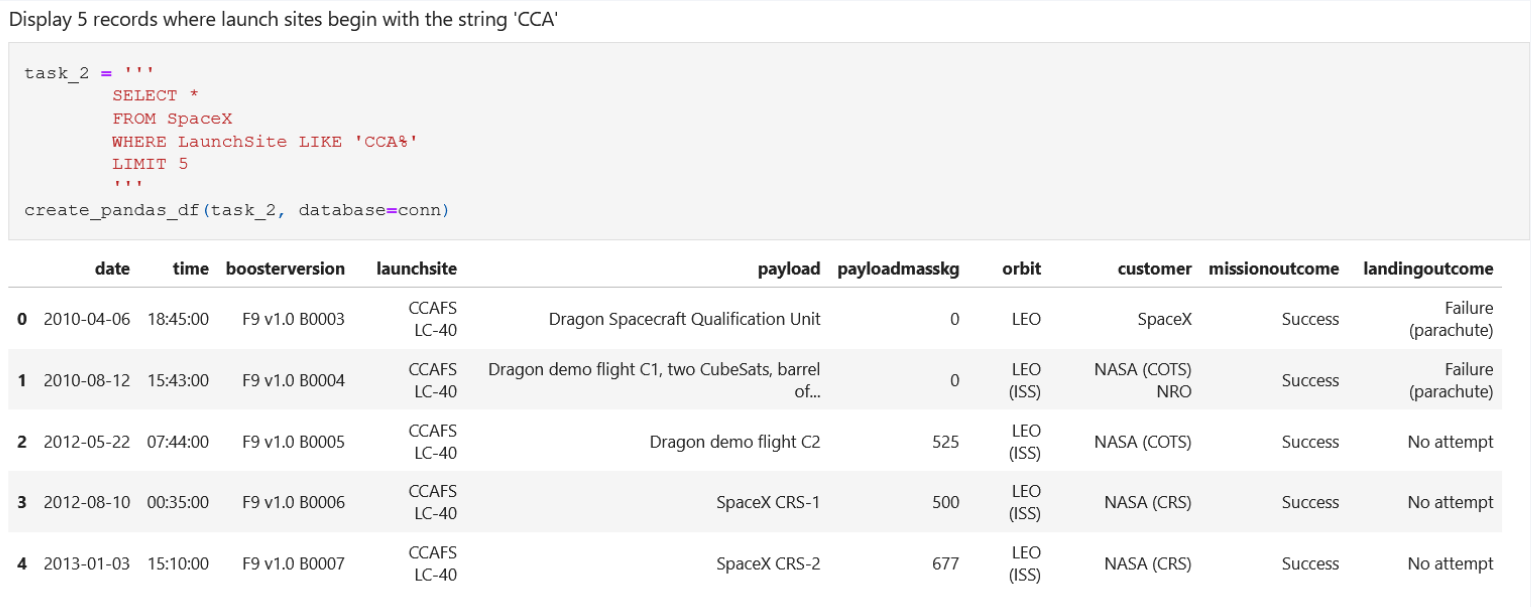
Display the names of the unique launch sites in the space mission

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

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

# Launch Site Names Begin with 'CCA'

Used the below query to display 5 records where launch sites begin with the string 'CCA'



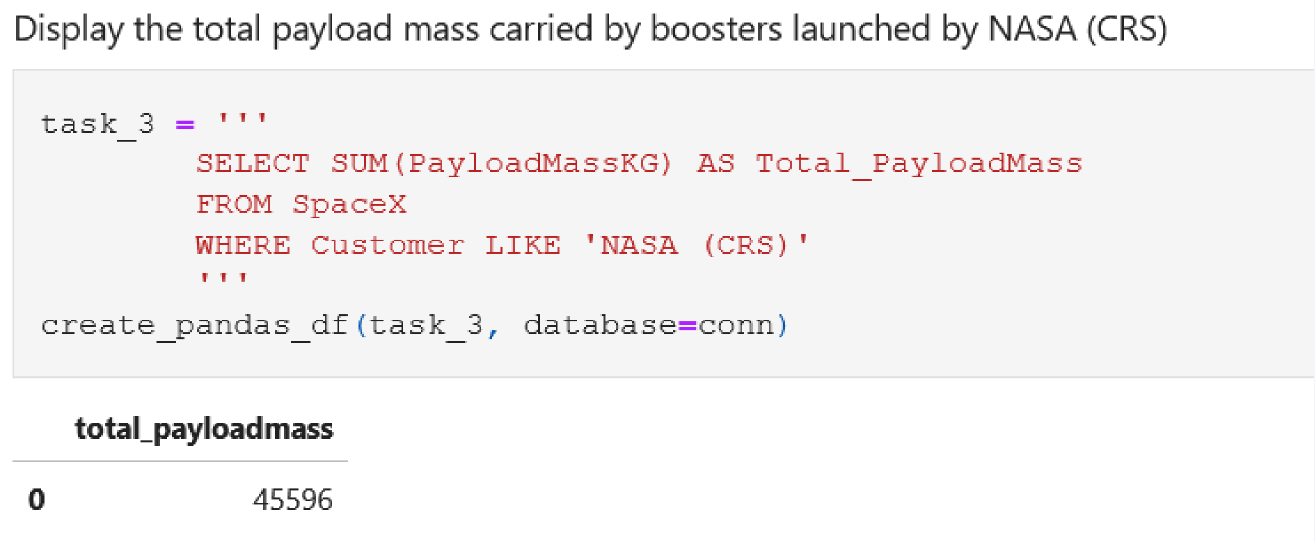
Display 5 records where launch sites begin with the string 'CCA'

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

| | 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 |

# Total Payload Mass

Calculated the total payload mass carried by boosters launched by NASA using the below query.



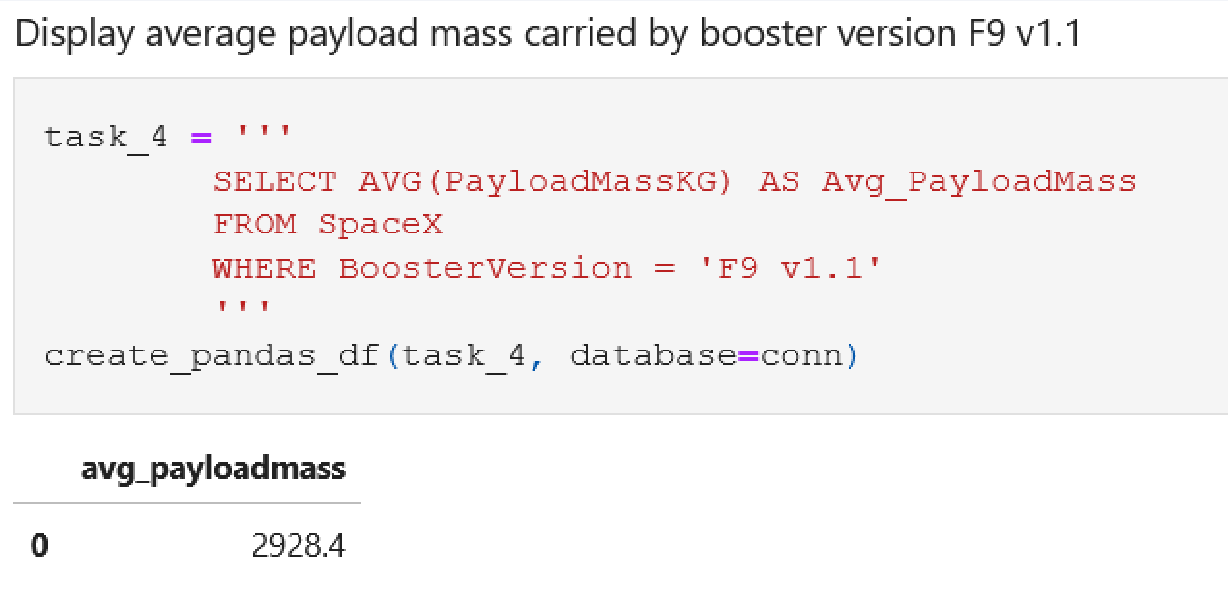
Display the total payload mass carried by boosters launched by NASA (CRS)

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

| | total_payloadmass |
|---|---|
| **0** | 45596 |

# Average Payload Mass by F9 v1.1

Calculated the average payload mass carried by booster version F9 v1.1 by using the below query.



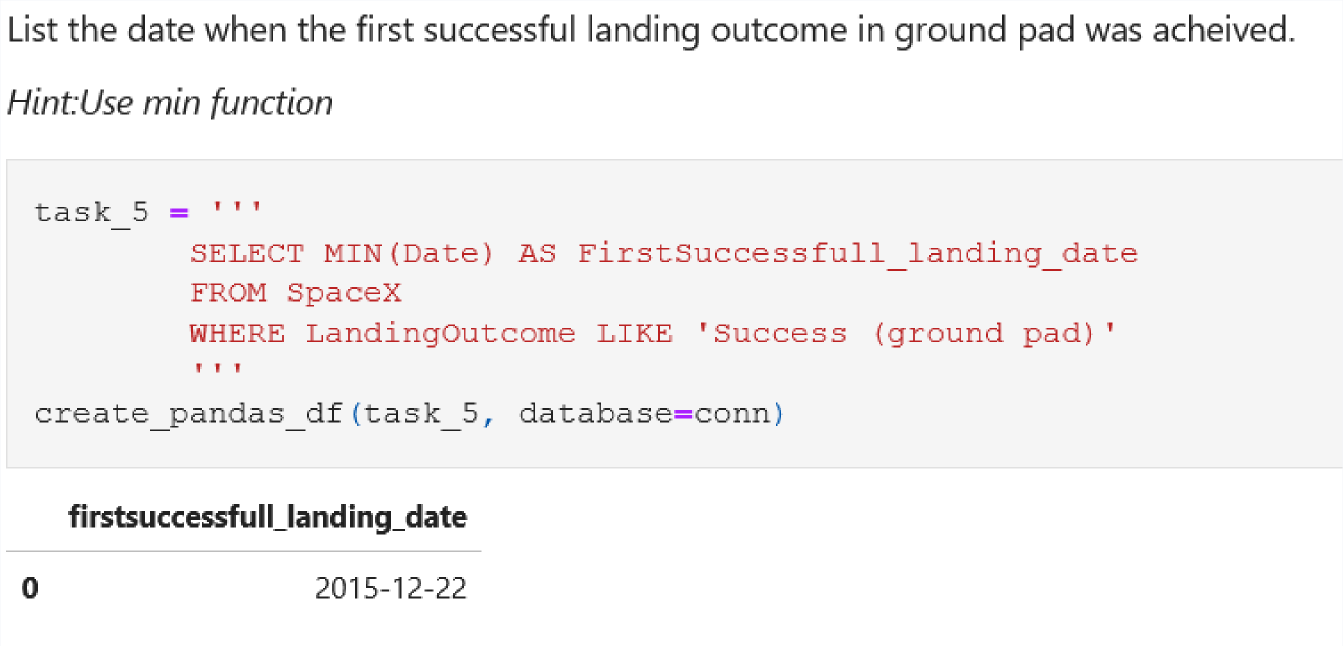
Display average payload mass carried by booster version F9 v1.1

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

|   | avg_payloadmass |
|---|---|
| **0** | 2928.4 |

# First Successful Ground Landing Date

Listed the date when the first successful landing outcome in ground pad was achieved using the below query and the MIN() function.



List the date when the first successful landing outcome in ground pad was acheived.

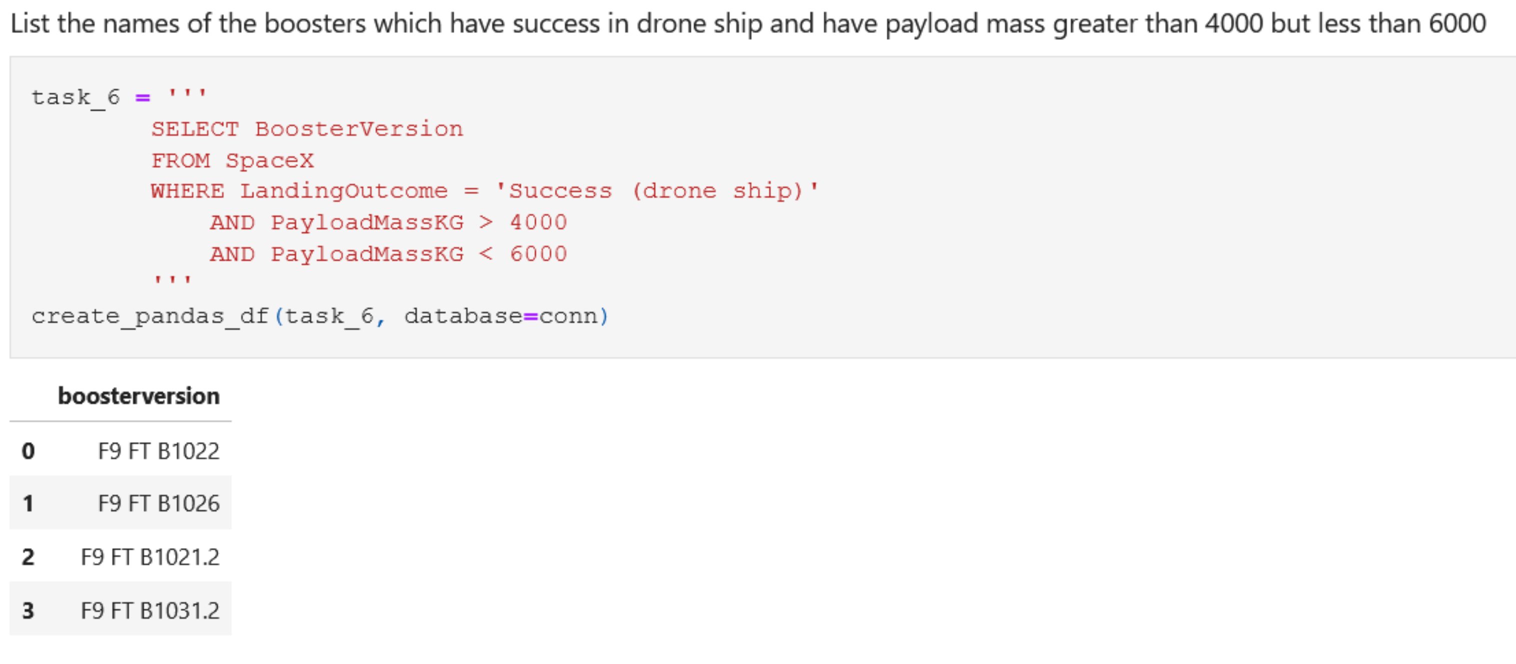
*Hint:Use min function*

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

| | firstsuccessfull_landing_date |
|---|---|
| **0** | 2015-12-22 |

# Successful Drone Ship Landing with Payload between 4000 and 6000

Calculated the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000 using the below query.



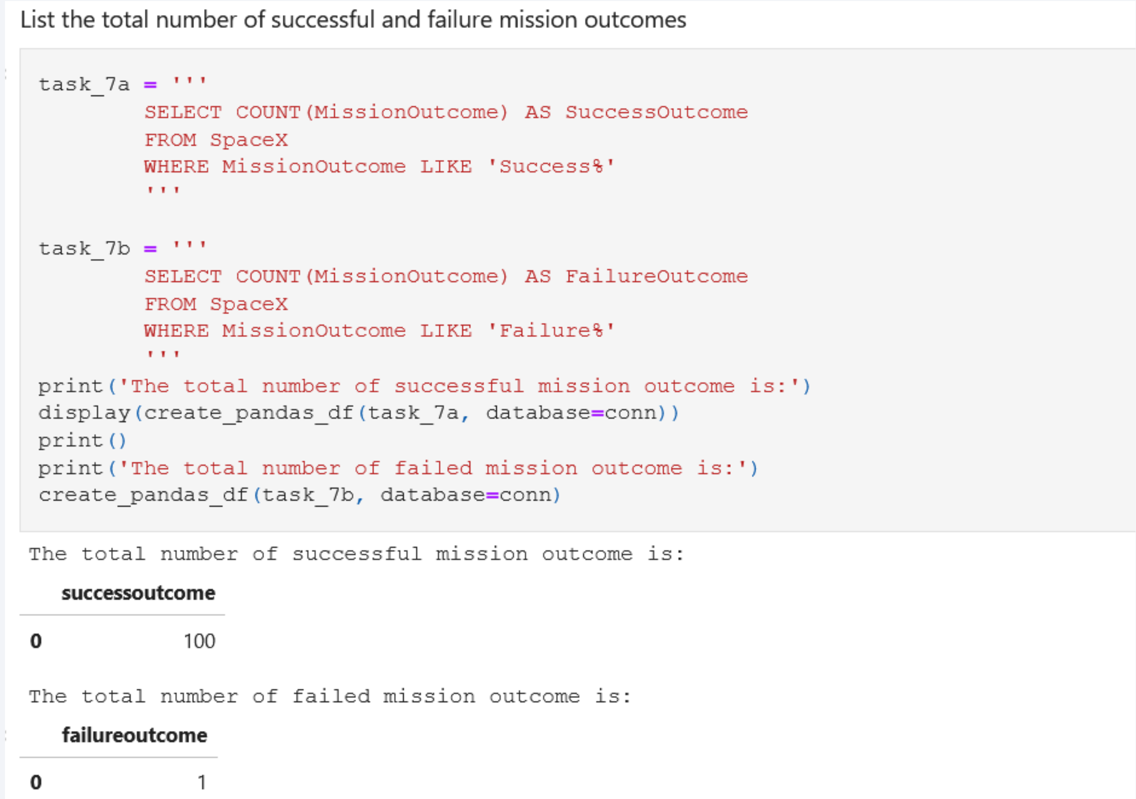
List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

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

|   | 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

Used wildcard '%' to help list the total number of successful and failed mission outcomes.



List the total number of successful and failure mission outcomes

```
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:
```

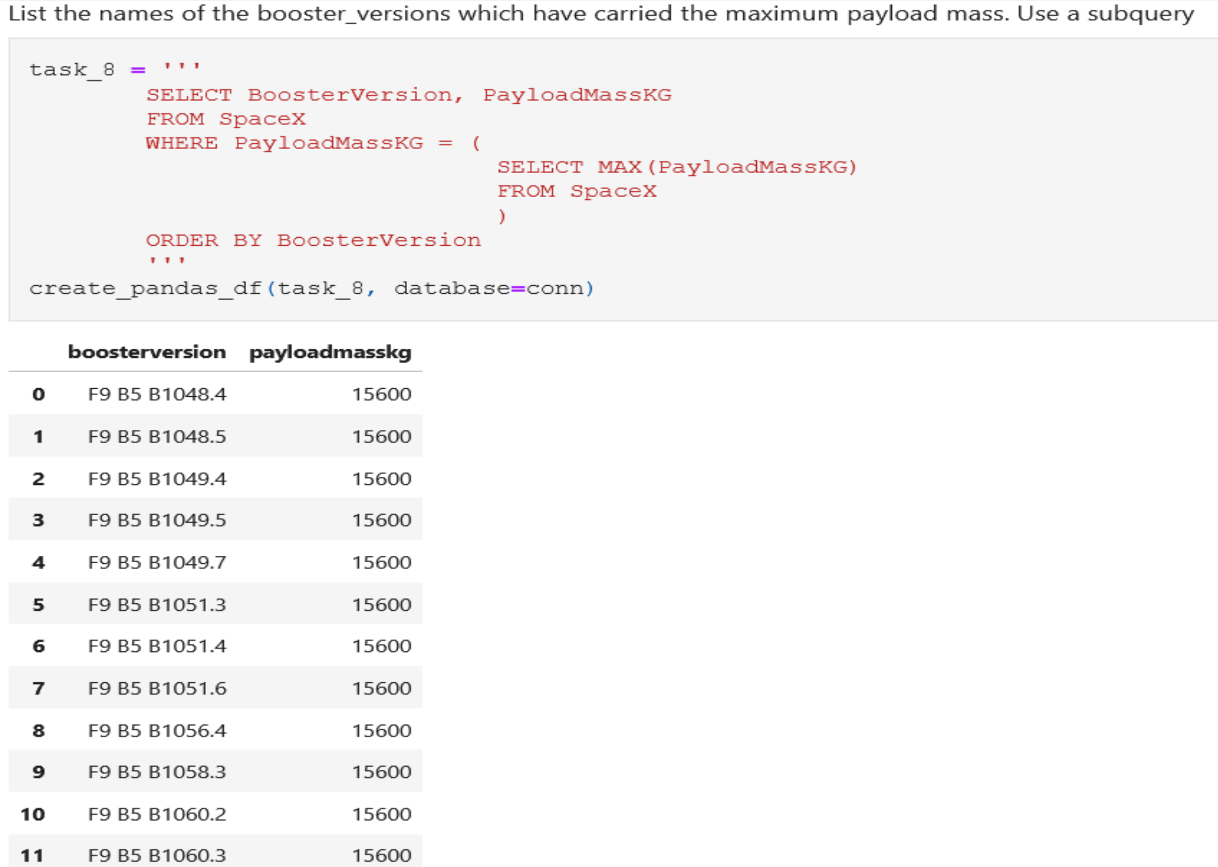
| | successoutcome |
|---|---|
| **0** | 100 |

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

| | failureoutcome |
|---|---|
| **0** | 1 |

# Boosters Carried Maximum Payload

Listed the names of the booster_versions which have carried the max payload mass using the below subquery using MAX() function.

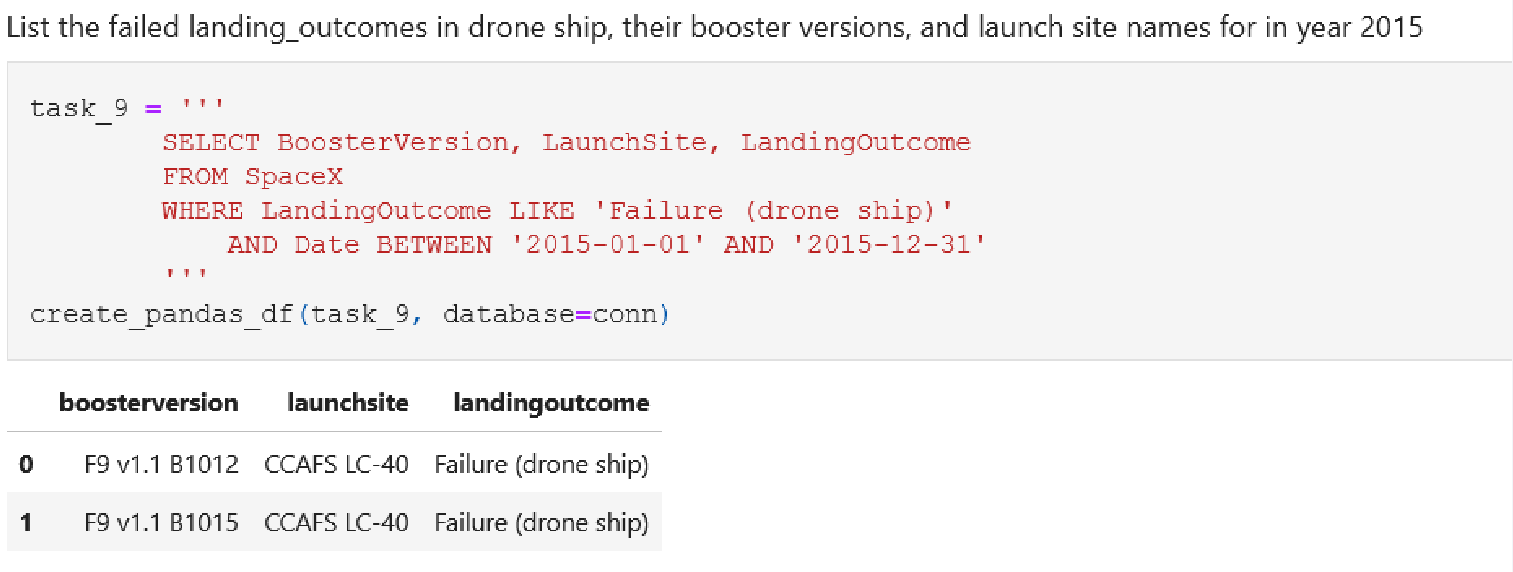


List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

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

| | 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

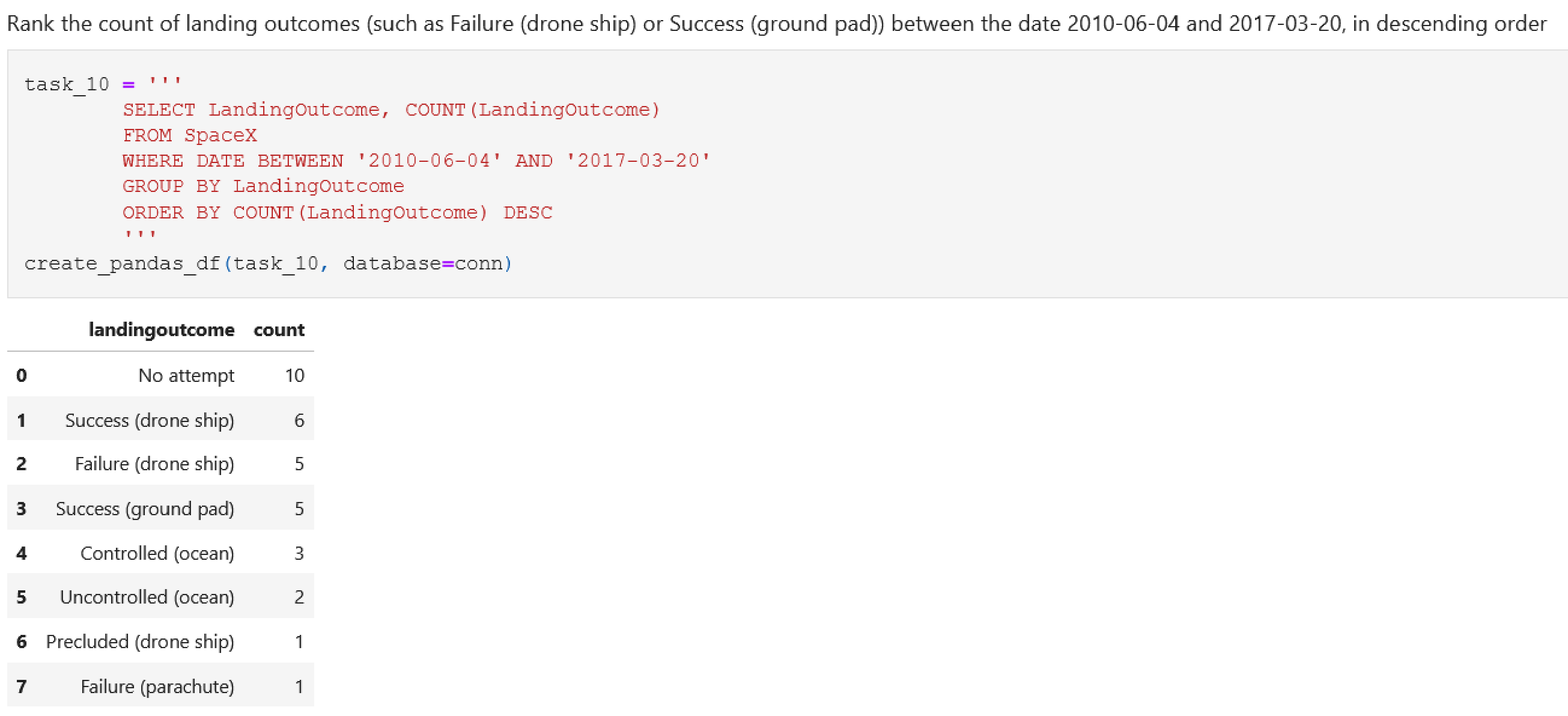


List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
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)
```

| | 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)) between the date 2010-06-04 and 2017-03-20, in descending order

```
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)
```

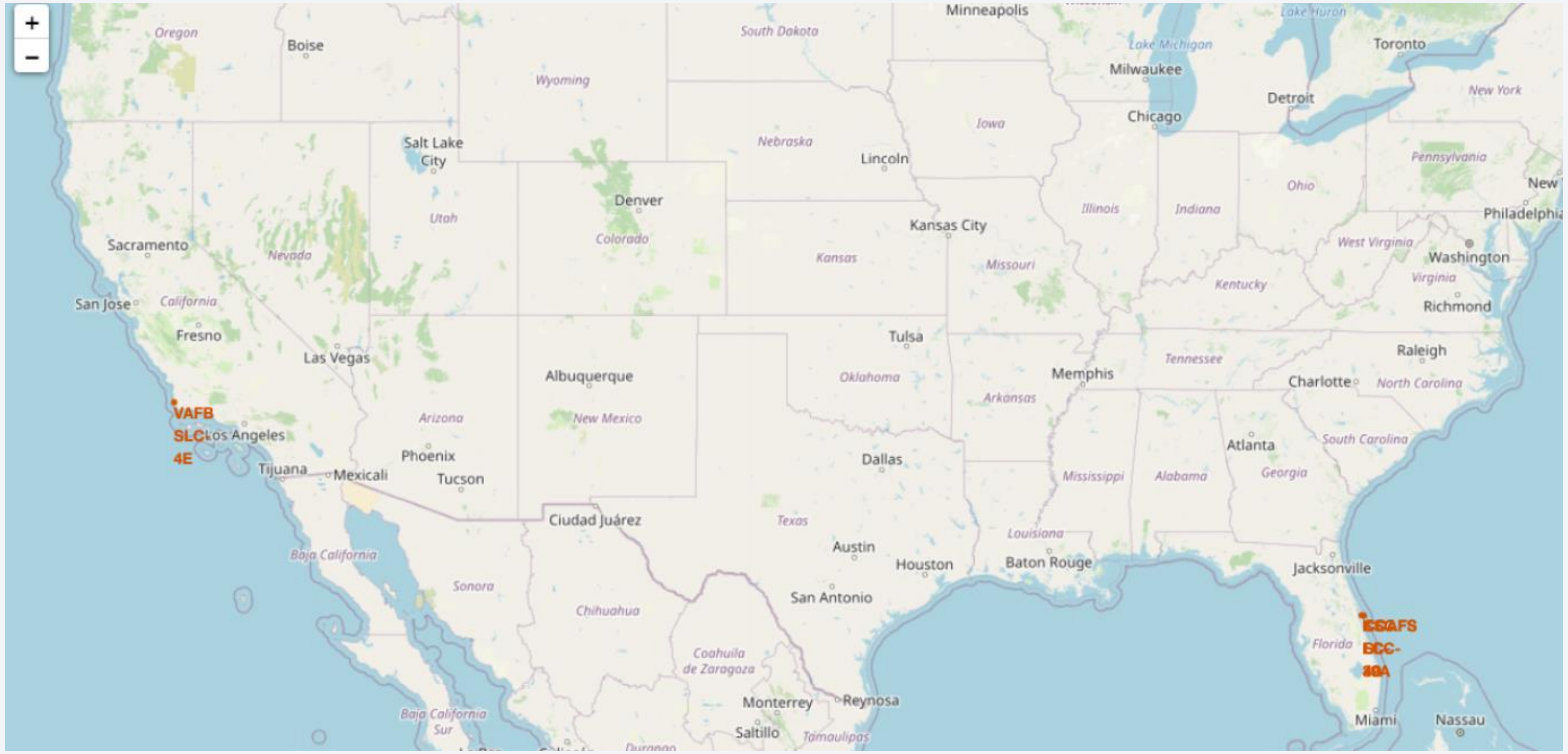
| | landingoutcome | count |
|---|---|---|
| **0** | No attempt | 10 |
| **1** | Success (drone ship) | 6 |
| **2** | Failure (drone ship) | 5 |
| **3** | Success (ground pad) | 5 |
| **4** | Controlled (ocean) | 3 |
| **5** | Uncontrolled (ocean) | 2 |
| **6** | Precluded (drone ship) | 1 |
| **7**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

Section 3

# Launch Sites Proximities Analysis

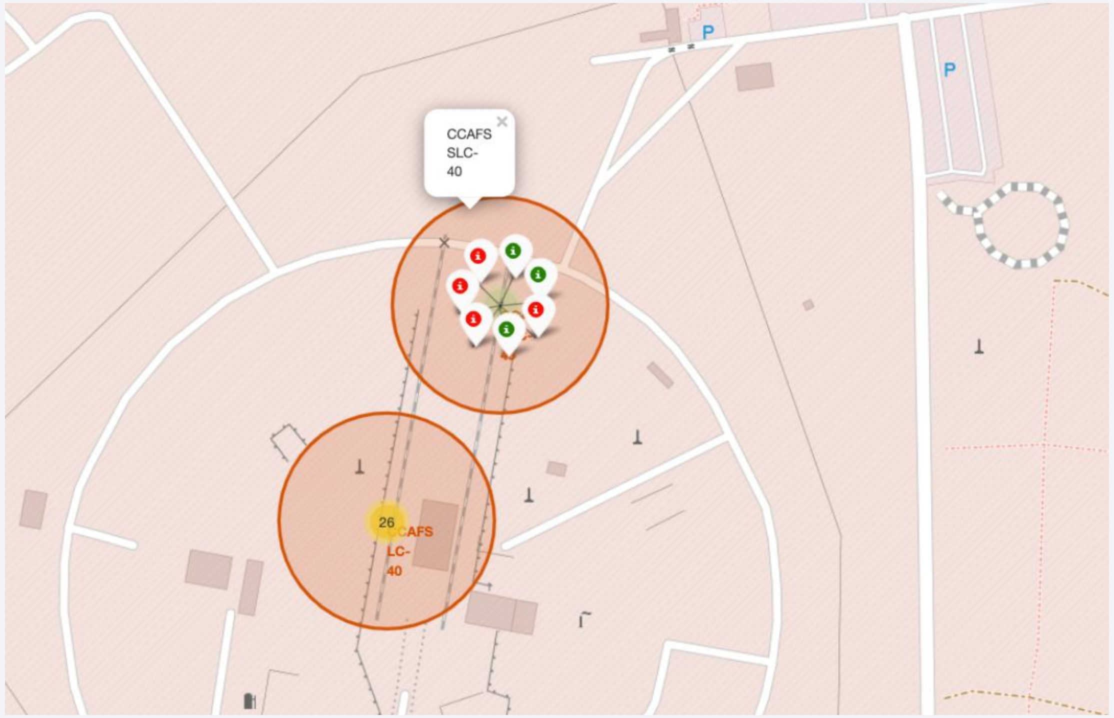
# All Launch Sites Locations with Folium

- There are two launch sites located in the U.S.A - Florida and California.

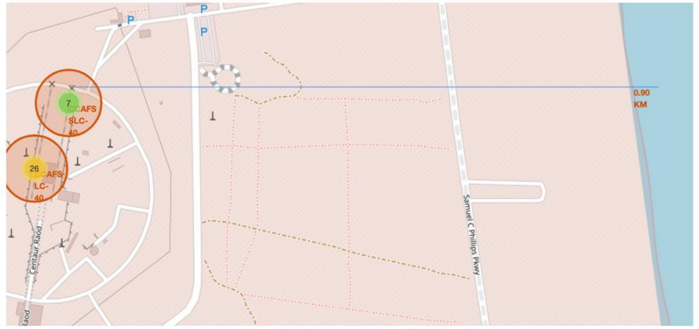


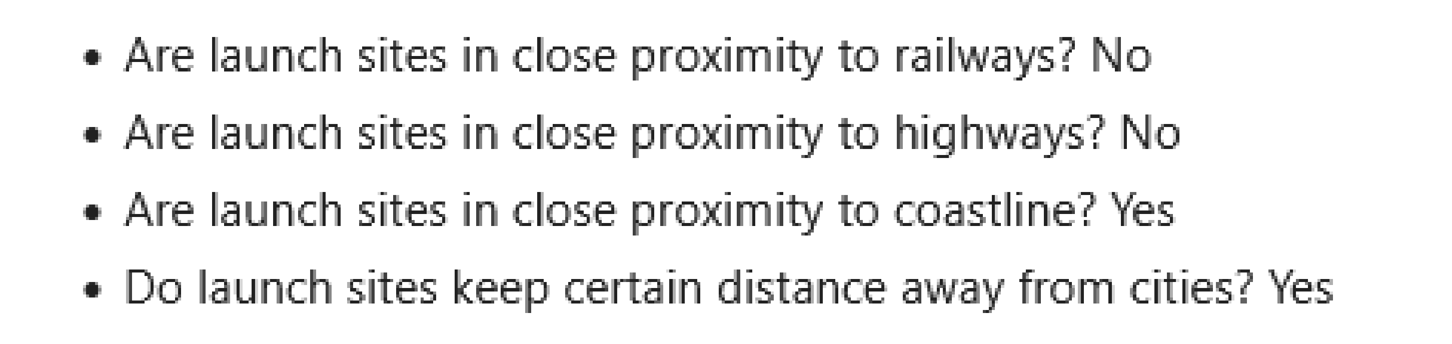
# Launch sites with markers showing color labels

- Green marker – successful launches
- Red marker – failed launches



# Launch Site Distances





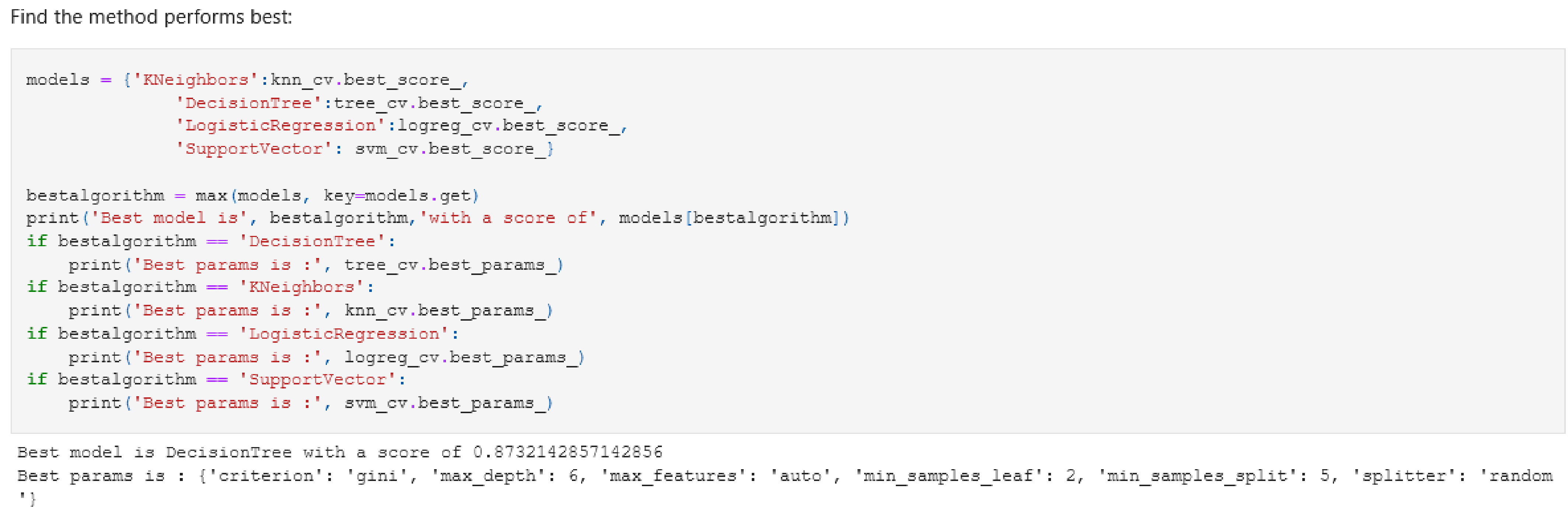
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

- The best model is the Decision Tree which has the highest classification accuracy.



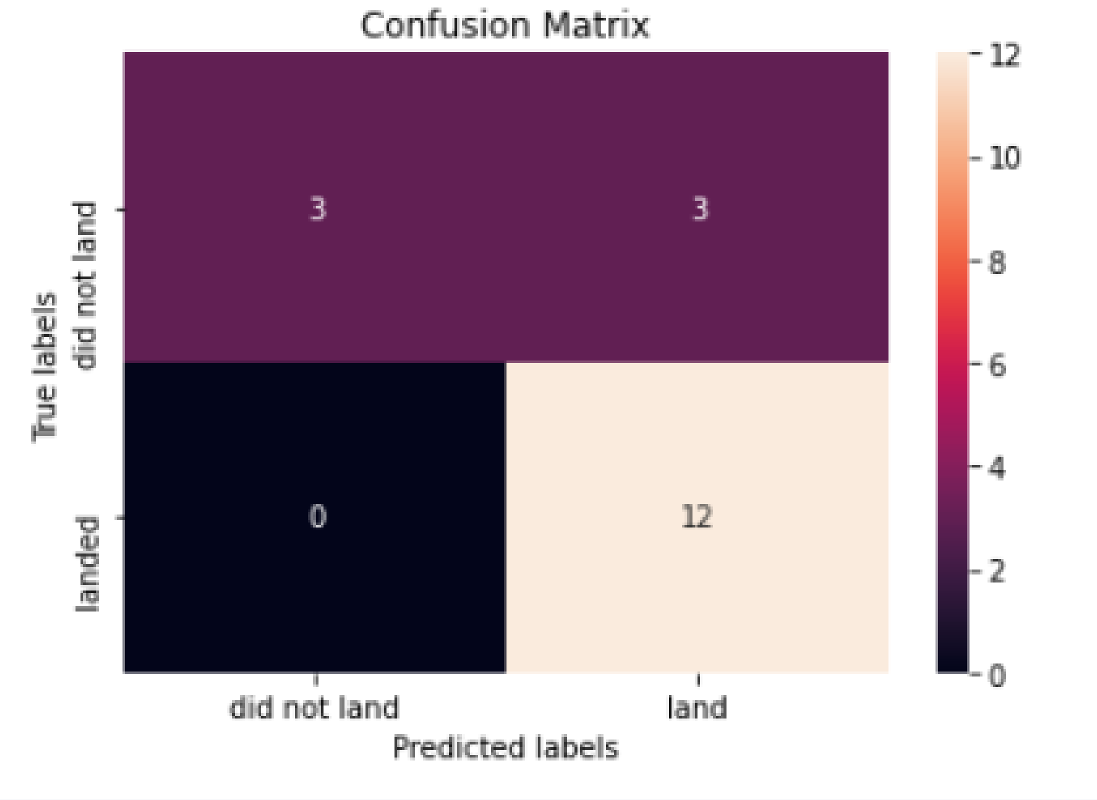
Find the method performs best:

```
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 classifier can distinguish between different classes. Problem with it is the false positives.



# Conclusions

- As the flight amount increases, the success rate increases at a launch site.
- As the payload mass increases, the success rate increases too.
- Launch success rate increased from 2013 until 2020.
- Orbits E-L1, GEO, HEO, SSO, VLEO had the most success rate.
- The SO has the least success rate amongst the orbits.
- KSC LC-39A had the most successful launches at any sites.
- The best machine learning algorithm is the decision tree classifier for this case because it has the highest classification accuracy.

# Thank you!