IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Abdelrhman Dameen
28/12/2021

Outline

Executive Summary

Introduction

Methodology

Results

Conclusion

Appendix

Executive Summary

- **Summary of methodologies**

  1. Data collection
  2. Data wrangling
  3. EDA with data visualization
  4. EDA with SQL
     Building an interactive map with plotly Dash
     Predictive analysis

- **Summary of all results**

  1. Exploratory data analysis results
  2. Interactive analytics demo in screenshots
  3. Predictive analysis results

Introduction

Project background and context.

We predicted whether the first stage of Falcon 9 would land successfully. SpaceX announced the Falcon 9 rocket launch on its website, at a cost of $62 million; other suppliers costing over $165 million each. Most of the money saved is due to SpaceX being able to reuse the first stage. Therefore, if we can determine if the first leg will land, we can determine the cost of a single launch. This information can be used if another company wants to bid with SpaceX to launch the rocket.

Problems you want to find answers.

What affects the successful landing of the rocket? The influence of each relationship to certain rocket variables will influence in determining the success rate of a successful landing. What conditions do SpaceX need to meet to get the best results and ensure the best rocket landing speed?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API
- Perform data wrangling
  - perform some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
- 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

Describe how data sets were collected.

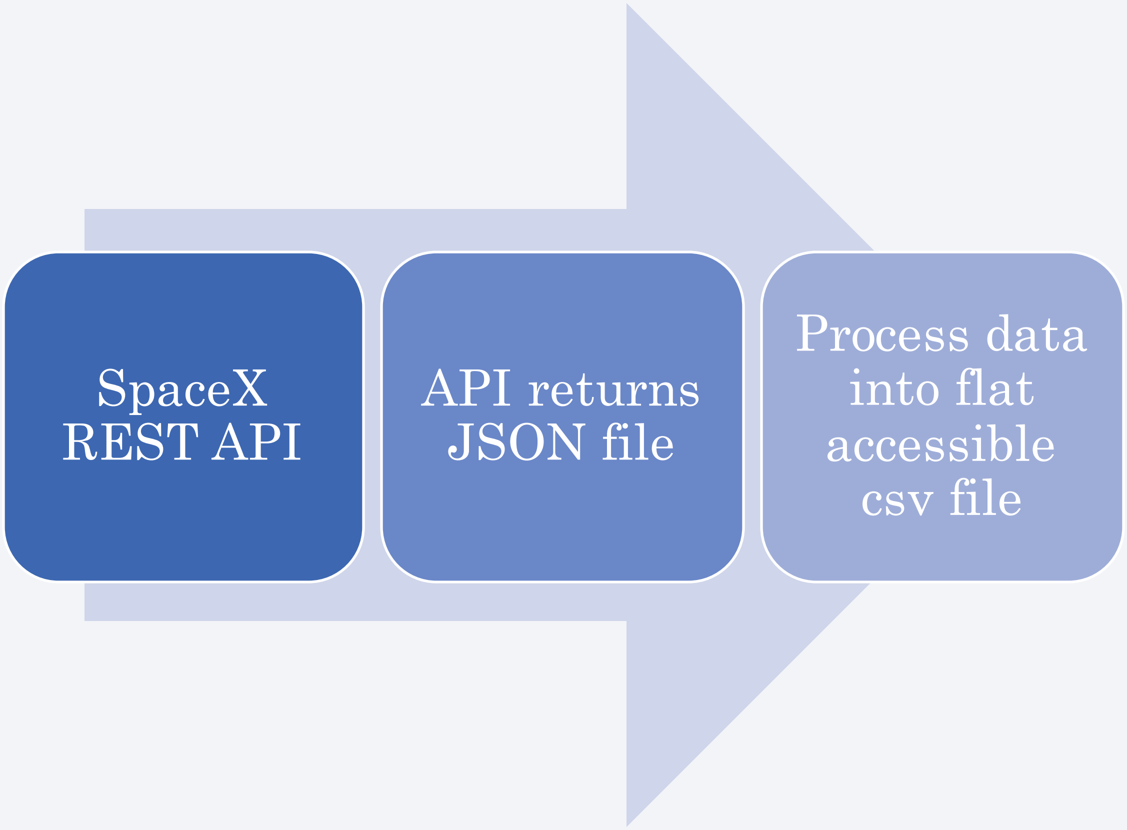
SpaceX launch data was collected using the SpaceX REST API. This data provided us with data on rocket launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcomes.

The goal of this project is to use this data to predict whether SpaceX will attempt to land a rocket.

Data Collection

You need to present your data collection process use key phrases and flowcharts

SpaceX API



SpaceX REST API

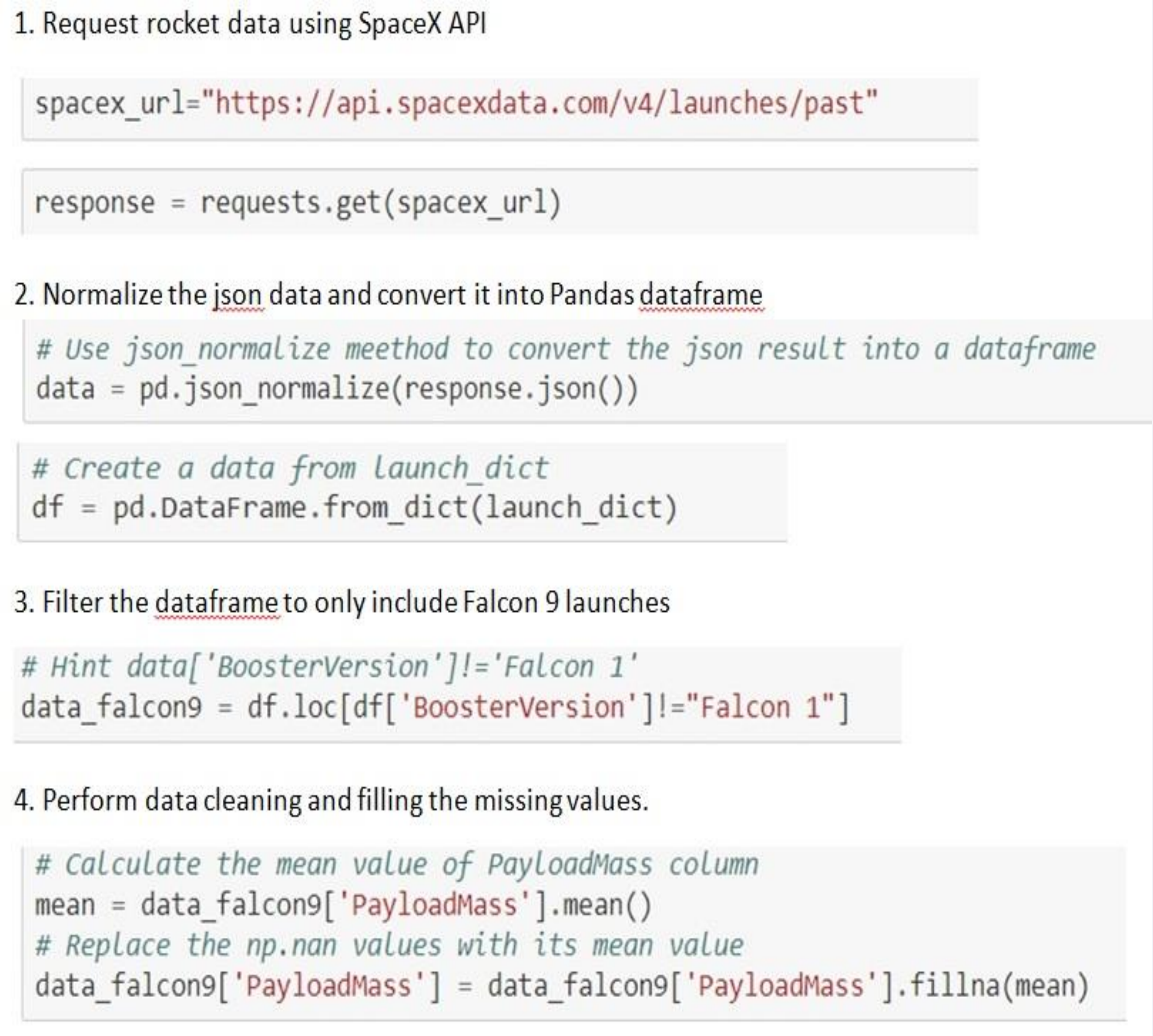
API returns JSON file

Process data into flat accessible csv file

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, then we cleaned the requested data and did some basic data wrangling and formatting.

The Github link (Notebook)



1. Request rocket data using SpaceX API

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

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

2. Normalize the json data and convert it into Pandas dataframe

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

```
# Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

3. Filter the dataframe to only include Falcon 9 launches

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

4. Perform data cleaning and filling the missing values.

```
# Calculate the mean value of PayloadMass column
mean = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].fillna(mean)
```

Data Collection – SpaceX API

SpaceX REST API

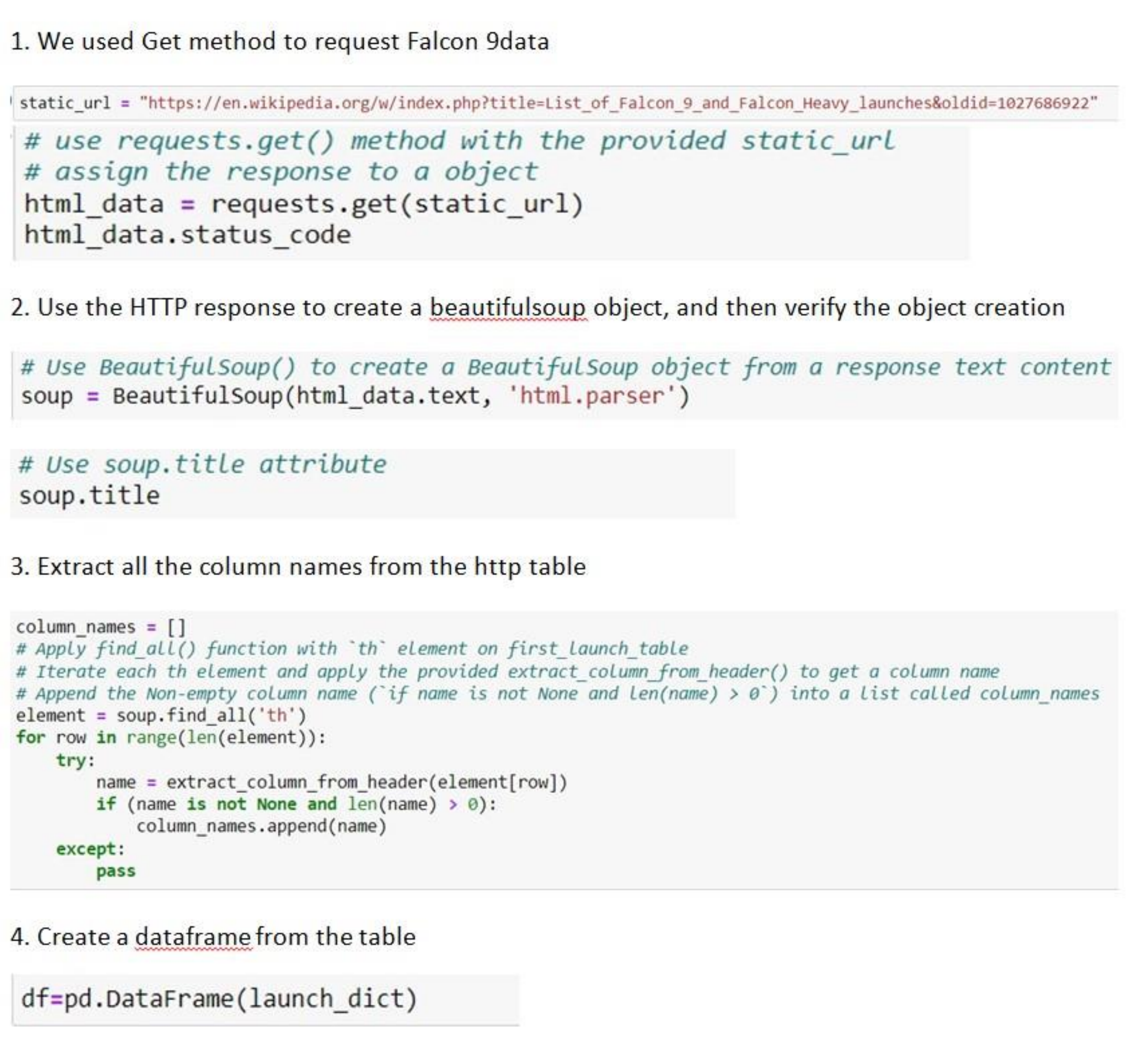
API returns JSON file

Process data into flat accessible csv file

Data Collection - Scraping

- We used beautifulsoup to scrap Falcon 9 records
- We parsed the html file and converted it into pandas.DataFrame

The Github link (Notebook)



1. We used Get method to request Falcon 9data

```
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
```

2. Use the HTTP response to create a beautifulsoup object, and then verify the object creation

```
# 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
```

3. Extract all the column names from the http table

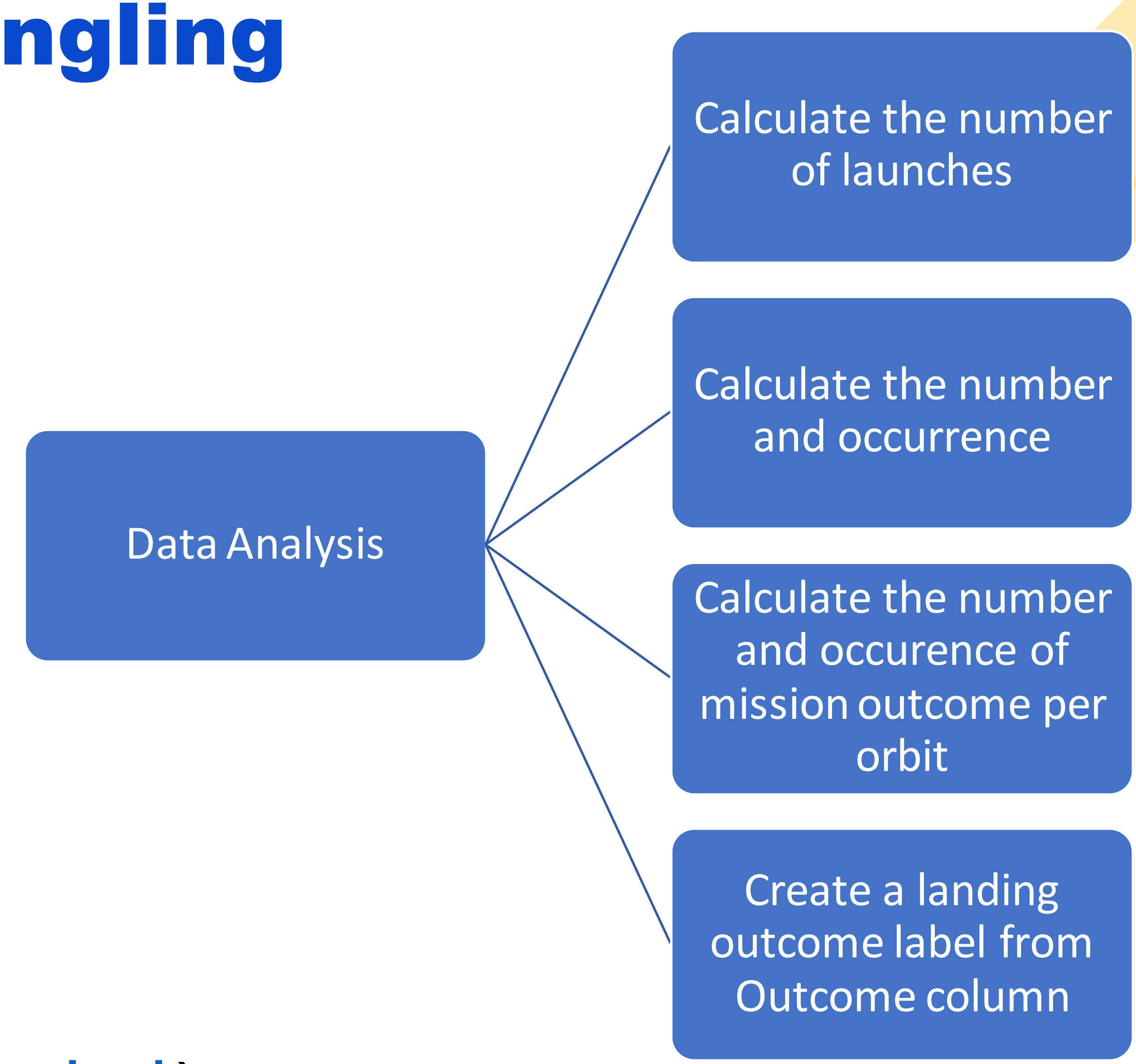
```
column_names = []
# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
element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

```
df=pd.DataFrame(launch_dict)
```

Data Wrangling

- We performed exploring data analysis and determined the training tables.
- We calculated the number of launches at each site and the number of occurrence of each orbits.
- We created a landing outcome label from columns
- Export the results to csv.



Data Analysis

Calculate the number of launches

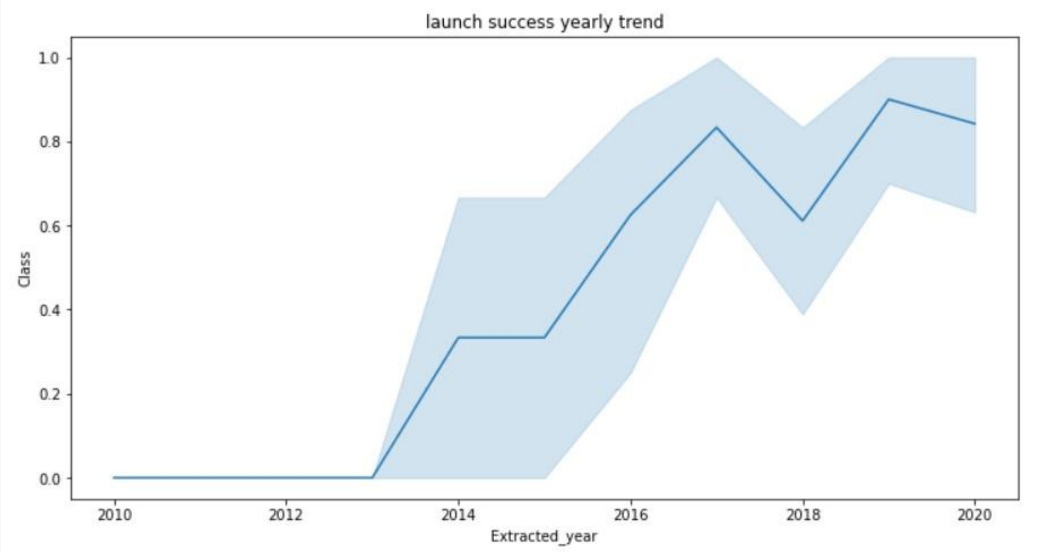
Calculate the number and occurrence

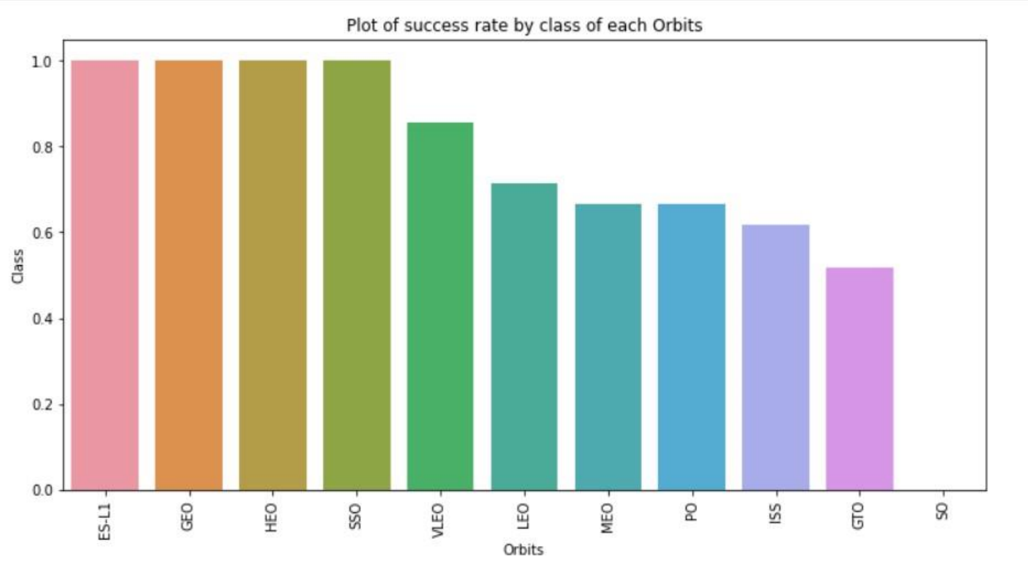
Calculate the number and occurence of mission outcome per orbit

Create a landing outcome label from Outcome column

The Github link (Notebook)

EDA with Data Visualization





We plot a line chart with x axis to be Year and y axis to be average success rate, to get the average launch success trend.

We Analyzed the data by plotting it bar chart to try to find which orbits have high sucess rate.

The Github link (Notebook)

EDA with SQL

- Loading the SpaceX data into PostgreSQL using Jupyter Notebook
- Get insights from the data using EDA with SQL. Some examples of the queries:
  - 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 GitHub link (Notebook)

Build an Interactive Map with Folium

- Marked all launch sites, and added map objects such as markers, circles, lines to mark the success/failure of launches for each site.
- Assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Implement the color-labeled marker clusters, to identify which launch sites has relatively the high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question such as:
  - Are launch sites near railways, highways and coastlines?
  - Do launch sites keep certain distance away from cities?

The Github link (Notebook)

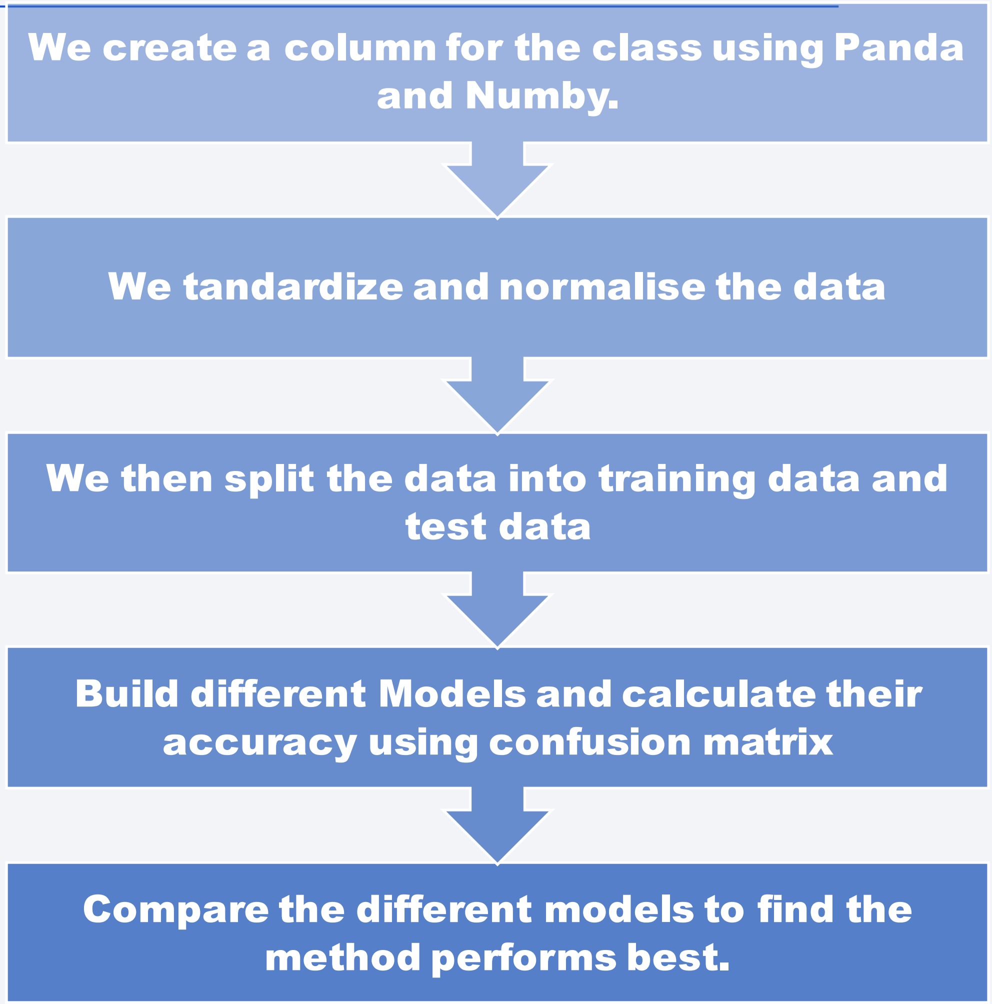
Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly, and Added a Launch Site Drop-down Input Component
- We built (success-pie-chart) 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 Github link (Notebook)

Predictive Analysis (Classification)

- Using numpy and pandas packages we transformed and then split our data into training and testing using sklearn model.
- 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.



We create a column for the class using Panda and Numby.

We tandardize and normalise the data

We then split the data into training data and test data

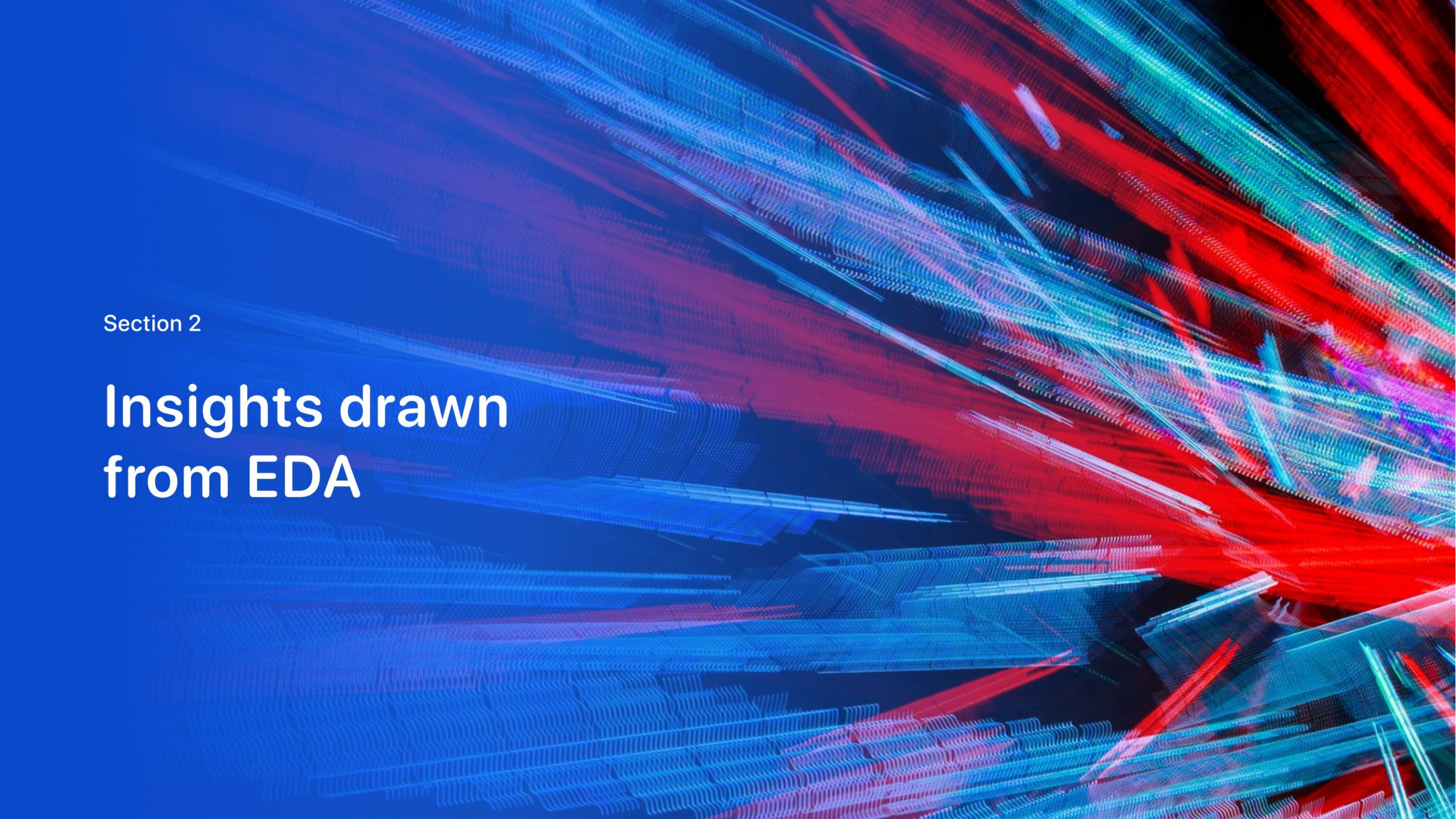
Build different Models and calculate their accuracy using confusion matrix

Compare the different models to find the method performs best.

The Github link (Notebook)

Results

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

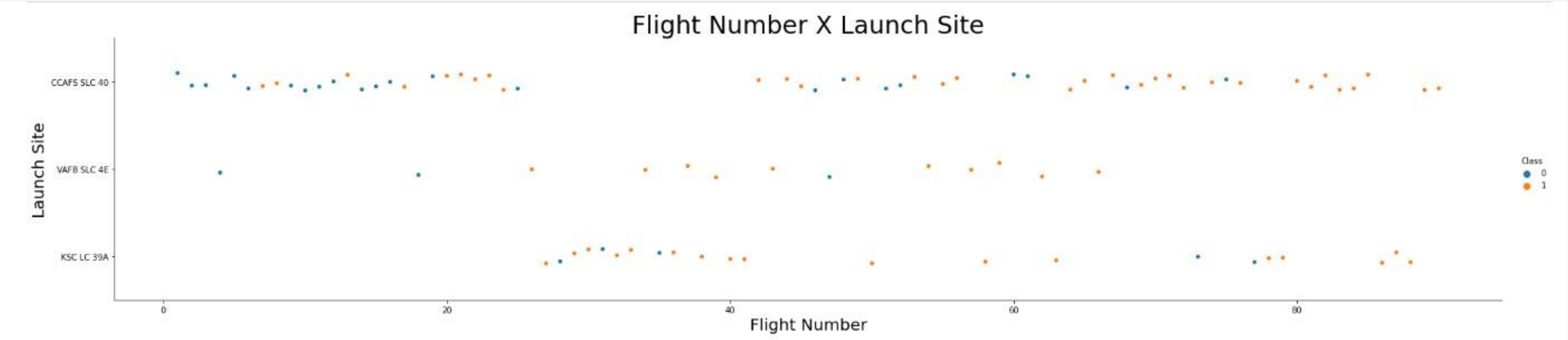


Section 2

Insights drawn from EDA

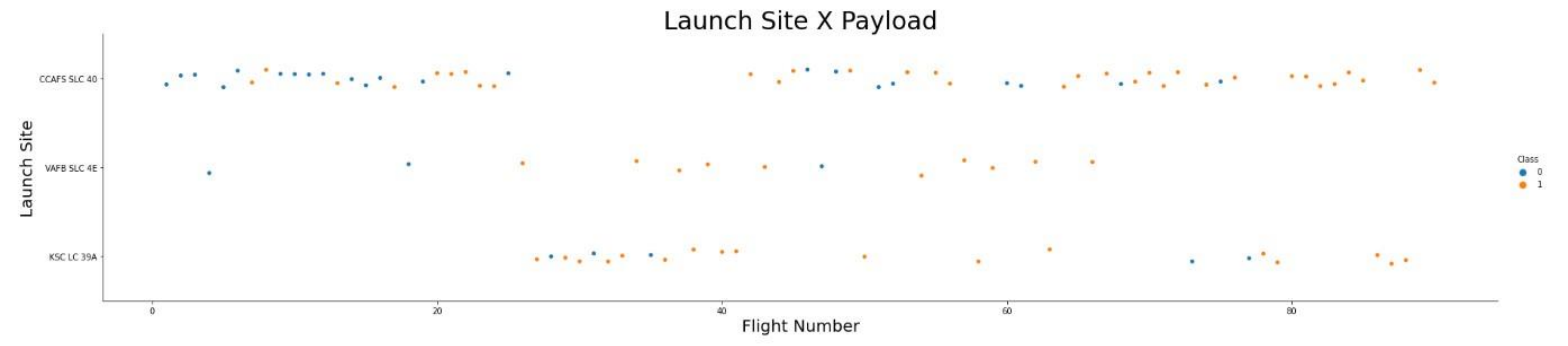
Flight Number vs. Launch Site

- The plot shows the greater the number of flight from a launch site, the higher the success rate.



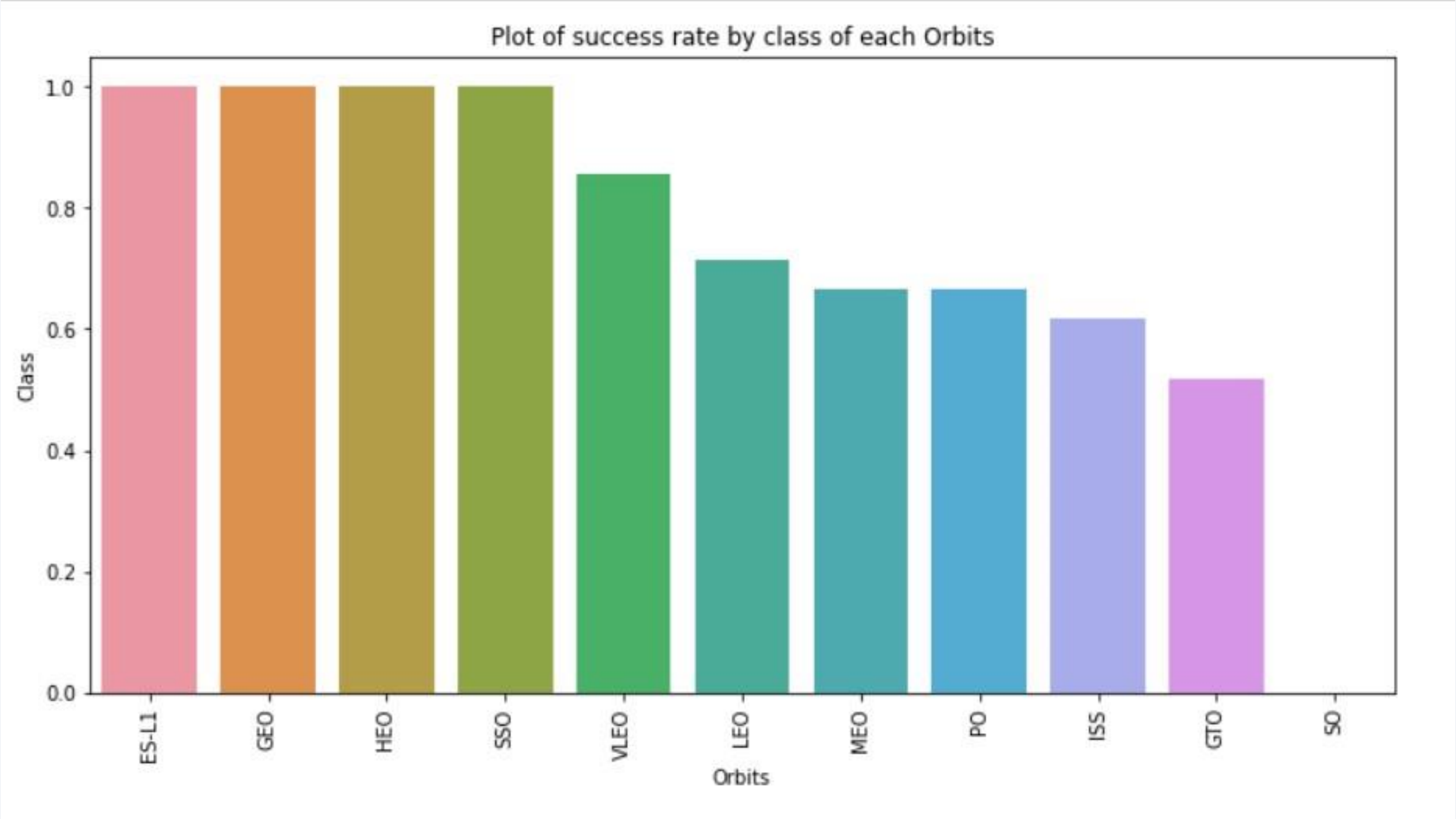
Payload vs. Launch Site

- The plot shows the greater the payload mass for launch site (CCAFS SLC 40), the higher the success rate.



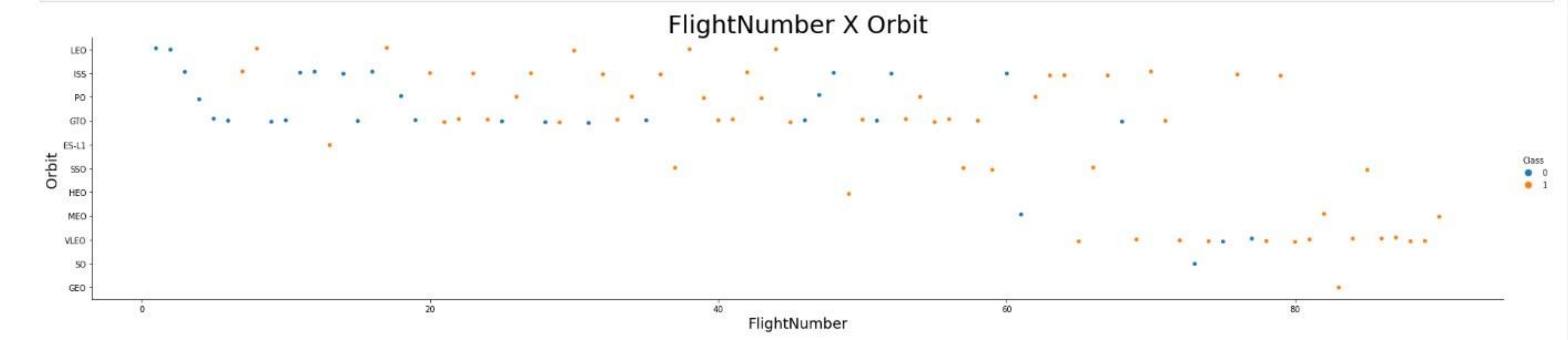
Success Rate vs. Orbit Type

- The chart show that ES-L1, GEO, HEO and SSO had the most success per Orbit



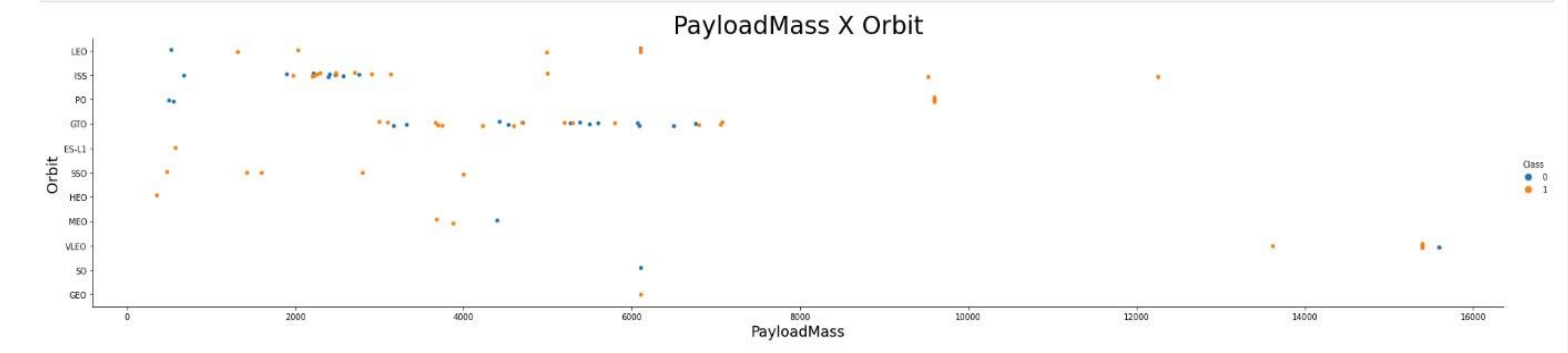
Flight Number vs. Orbit Type

- The plot shows that there is a positive correlation between flight number and Orbit LEO.



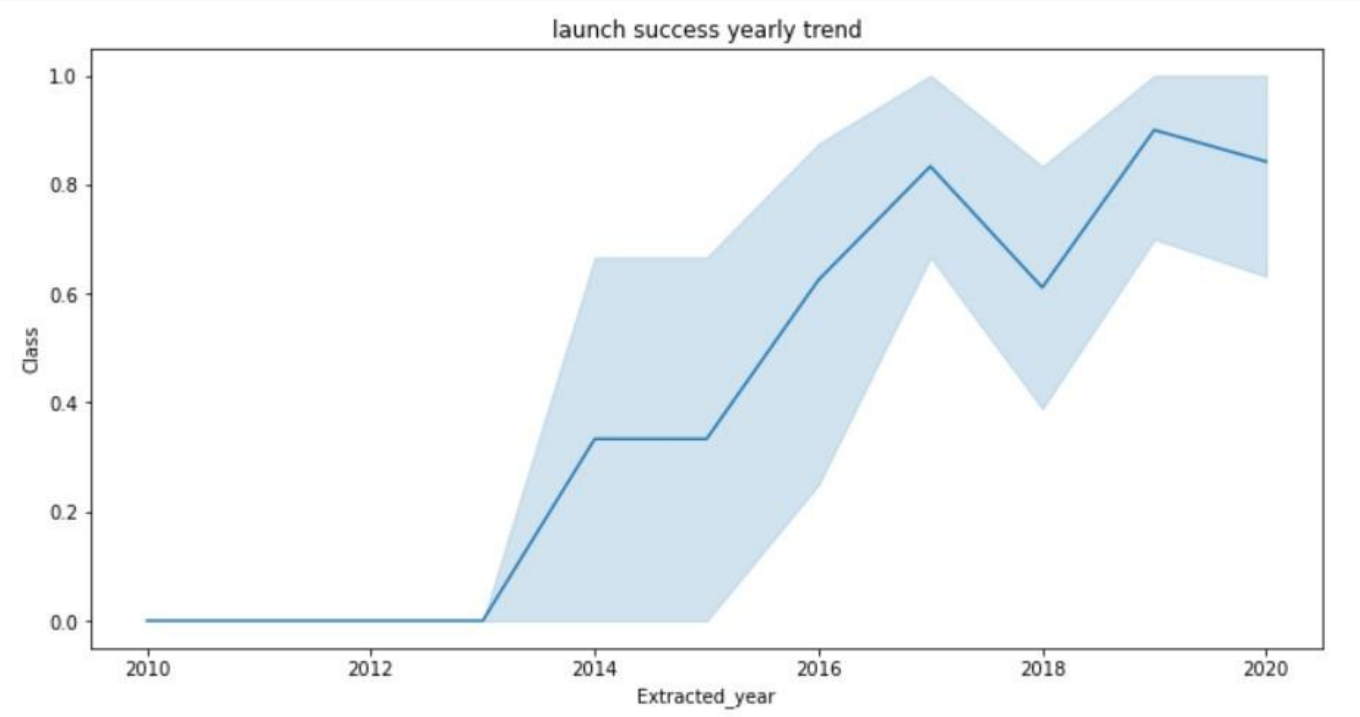
Payload vs. Orbit Type

- The plot shows that there is a correlation between the weight and the landing sucess



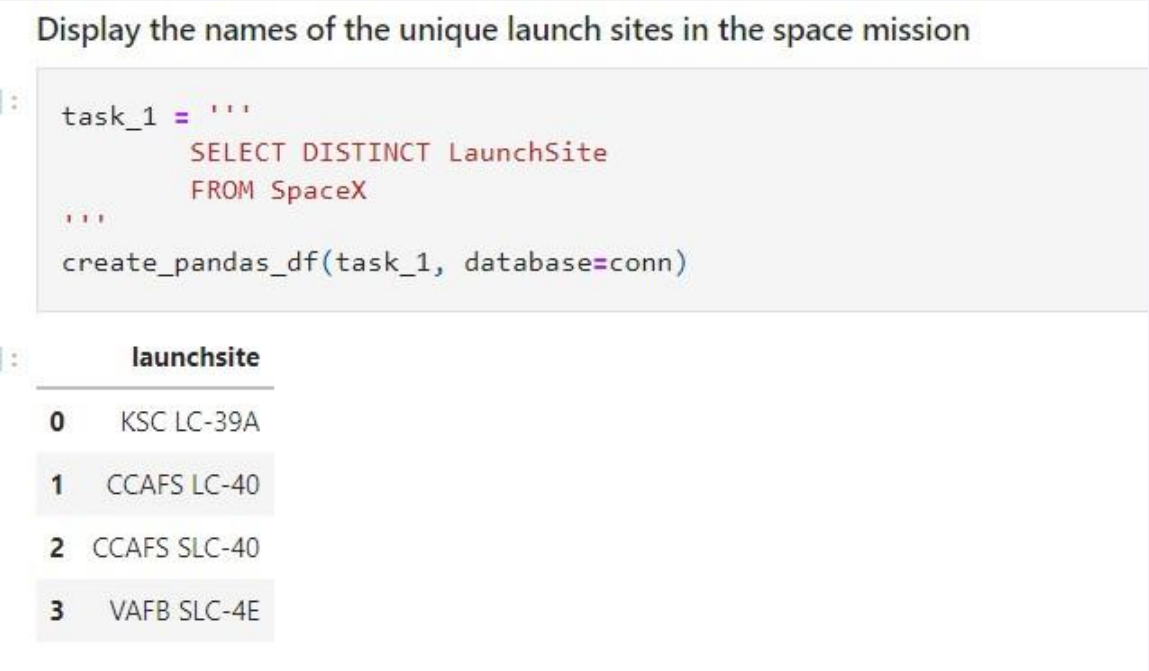
Launch Success Yearly Trend

- The plot shows that there the success rate is increasing each year with a slight drop in 2017



All Launch Site Names

- Using the SQL keyword 'DISTINCT" to show the unique launch sites from SpaceX data



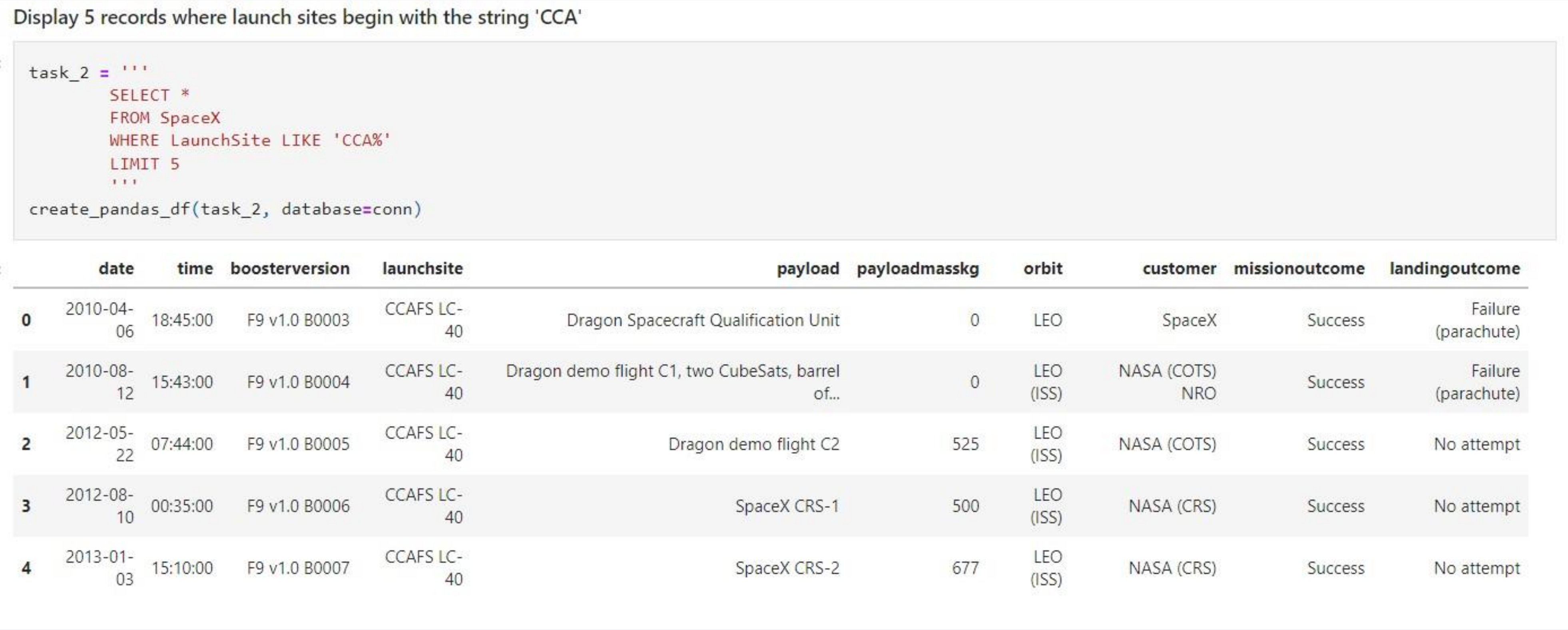
Display the names of the unique launch sites in the space mission

```
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'

- Using the SQL query to display 5 records where the launch site begin with "CCA"



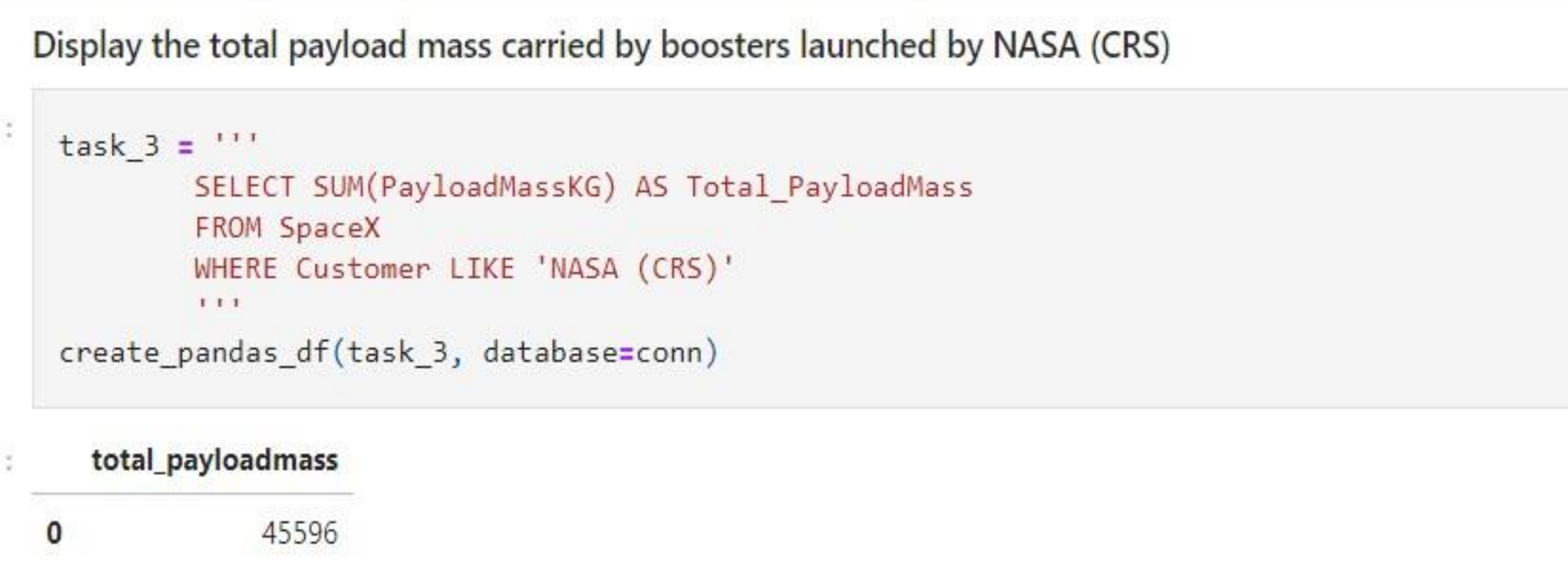
Display 5 records where launch sites begin with the string 'CCA'

```
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

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



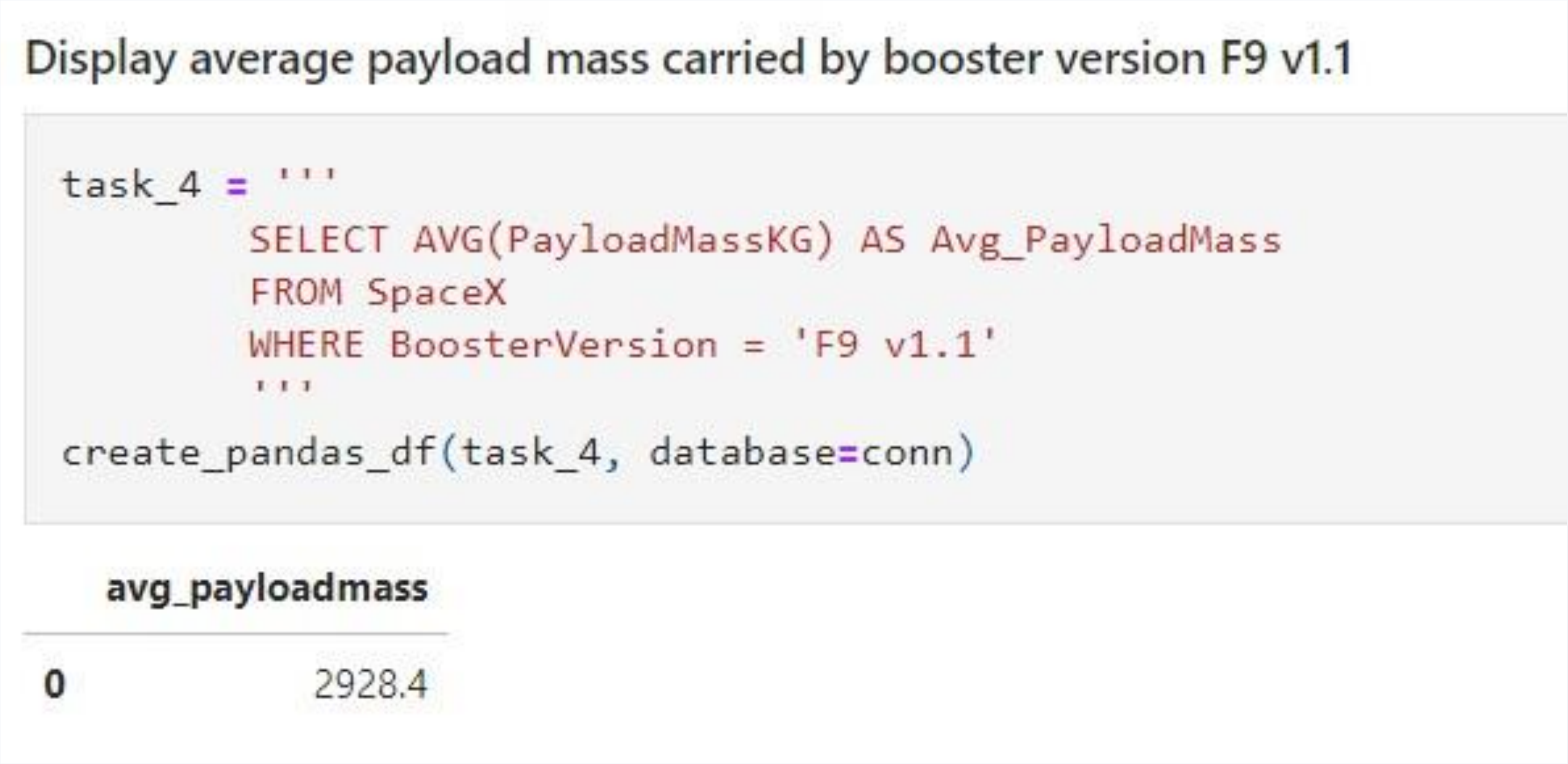
Display the total payload mass carried by boosters launched by NASA (CRS)

```
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

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



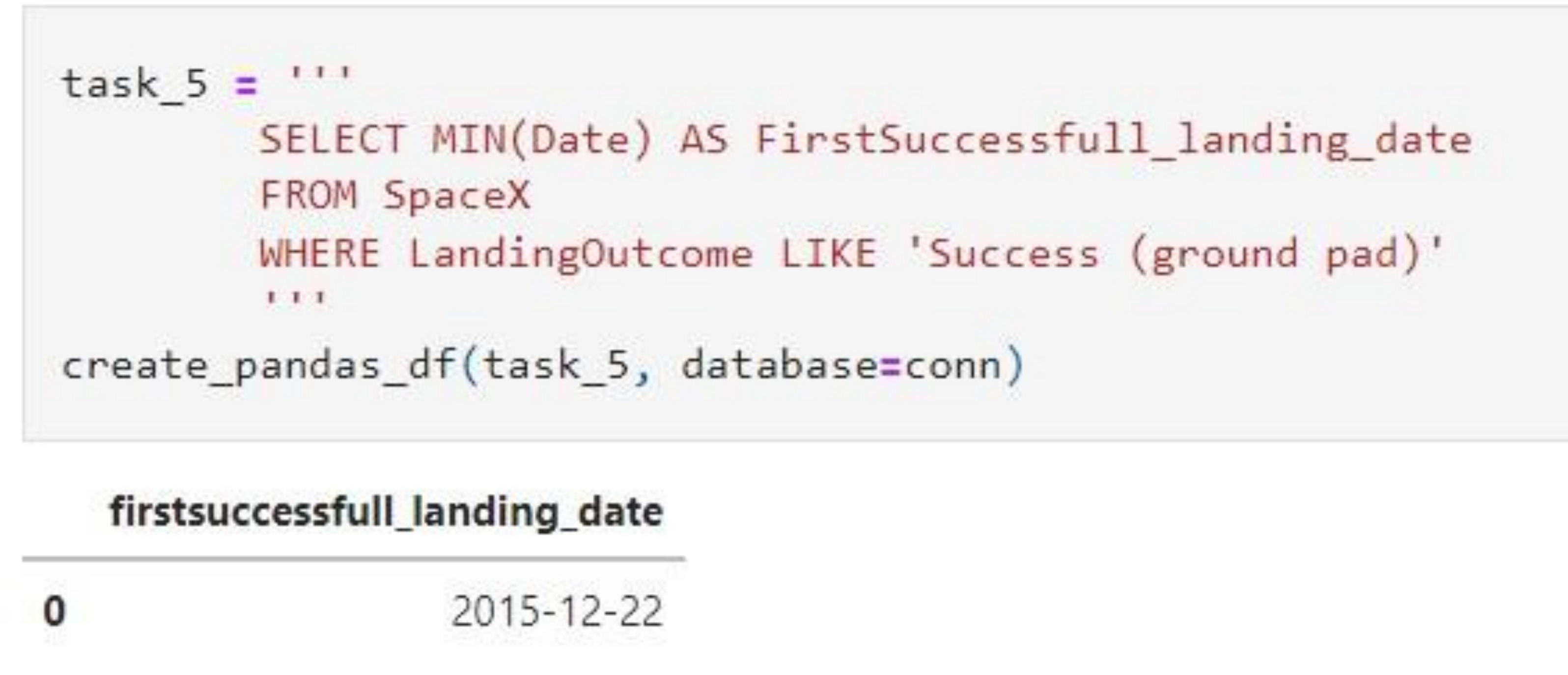
Display average payload mass carried by booster version F9 v1.1

```
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

- The date for the first successful landing outcome on ground pad is 22/12/2015

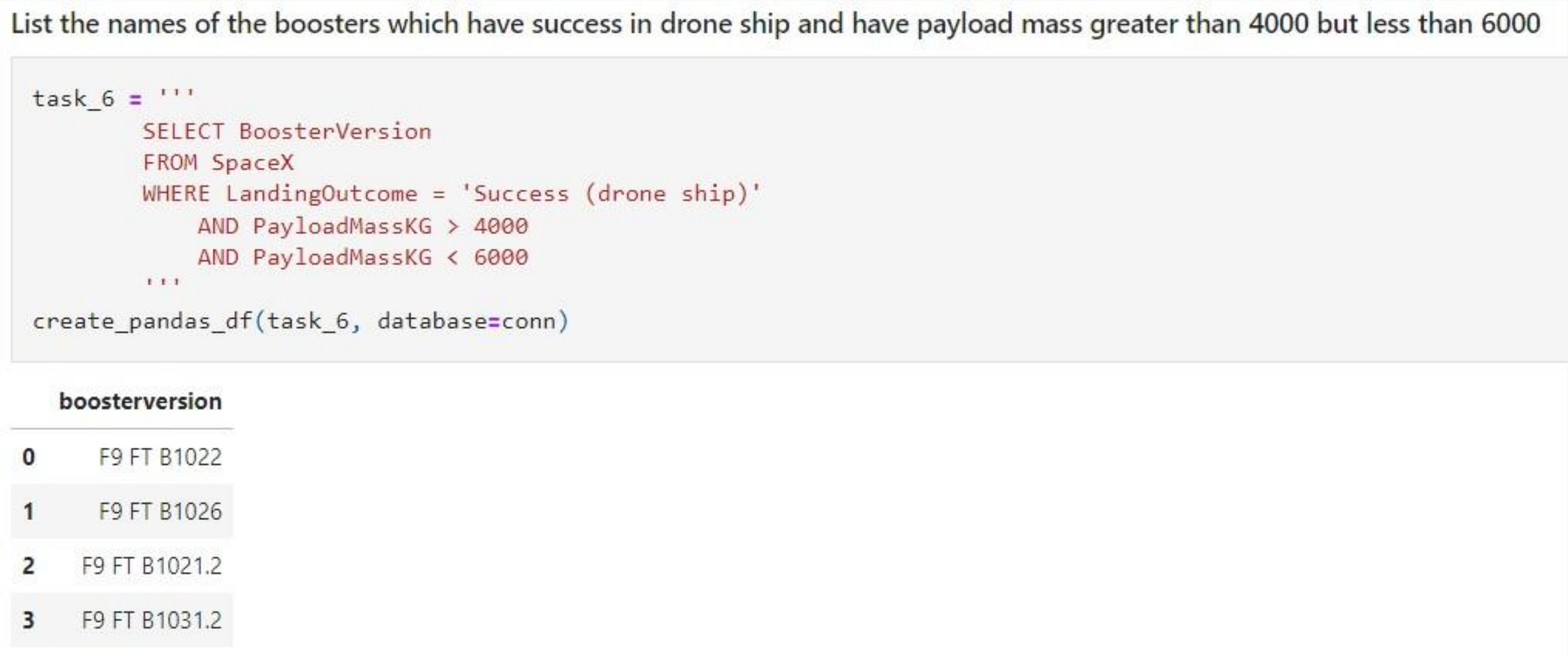


```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used SQL WHERE clause to filter the 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



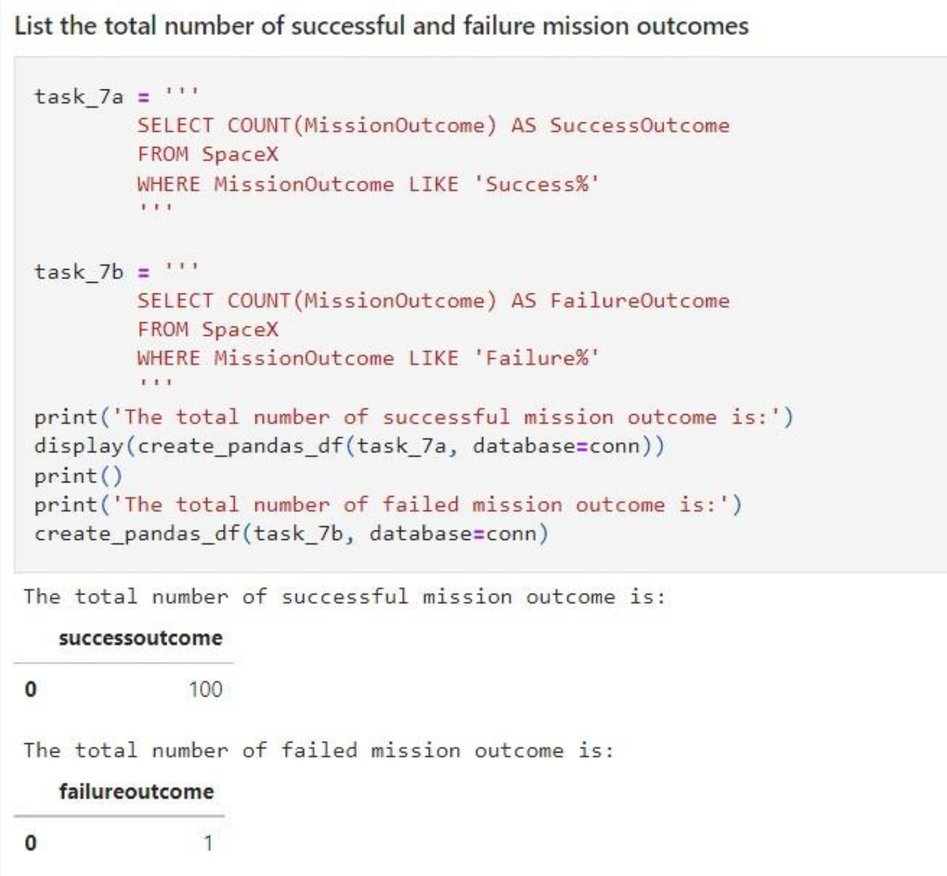
List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
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

- By using SQL wildcard like ‘%’ to filter for WHERE Mission outcome was a success or a failure.



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:
```

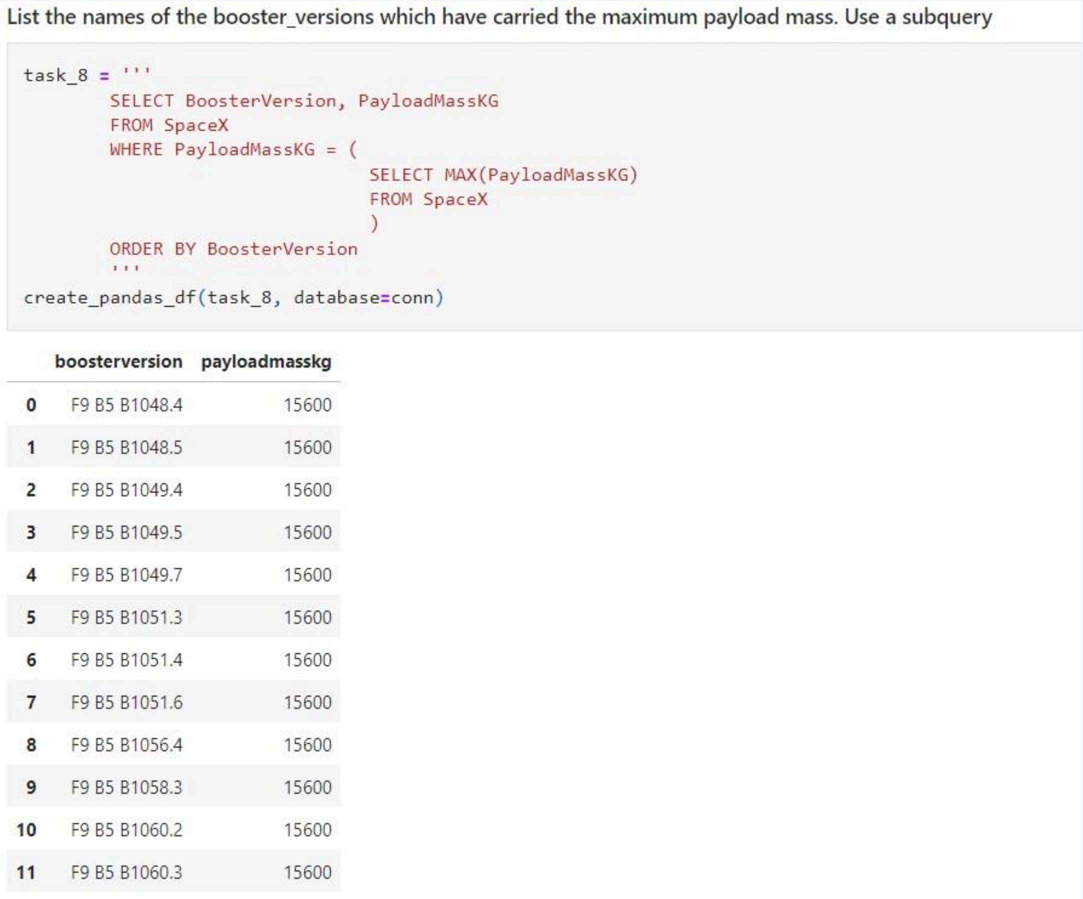
	successoutcome
0	100

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

	failureoutcome
0	1

Boosters Carried Maximum Payload

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



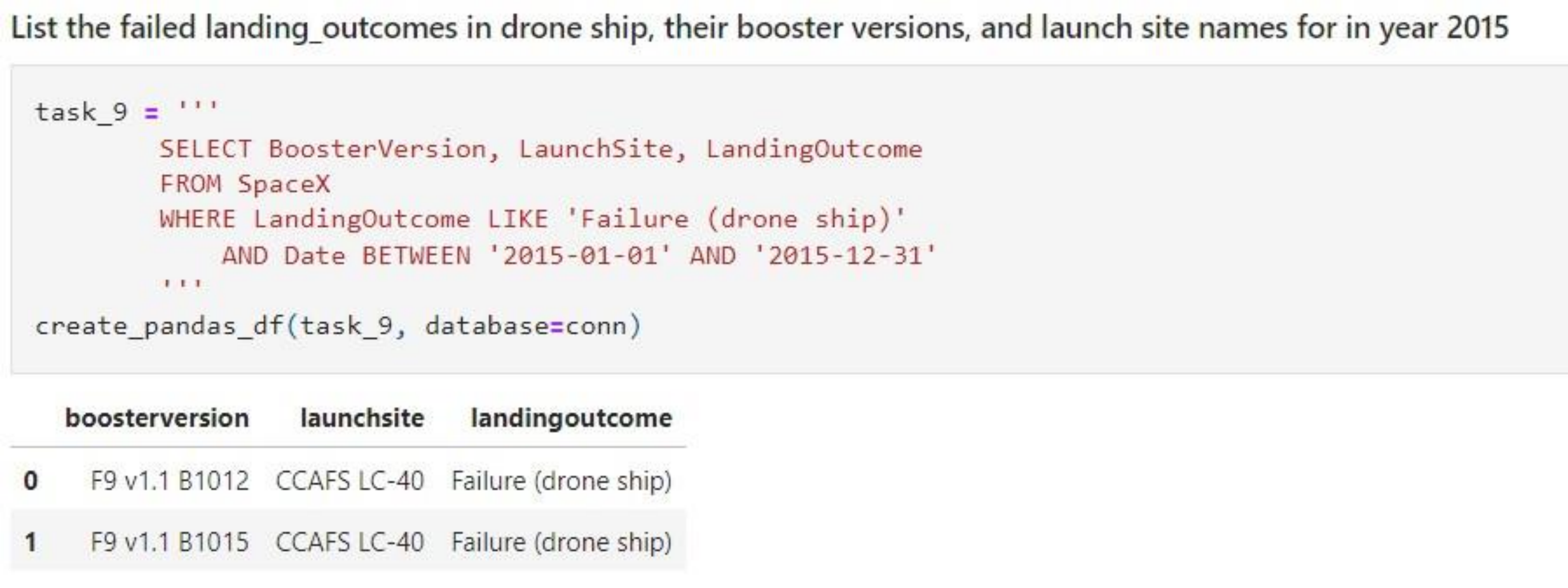
List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
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

- We used a combinations of the SQL 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



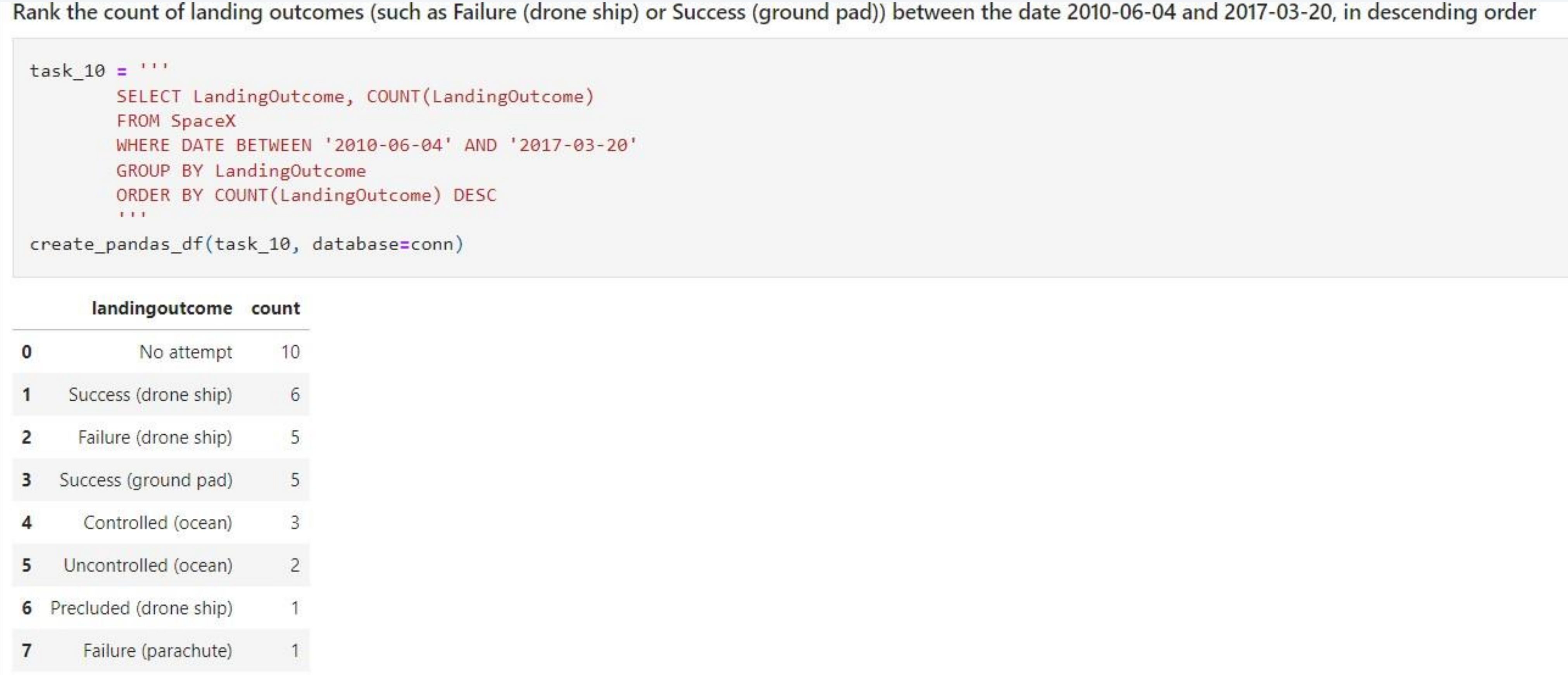
List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
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

- 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.



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)
```

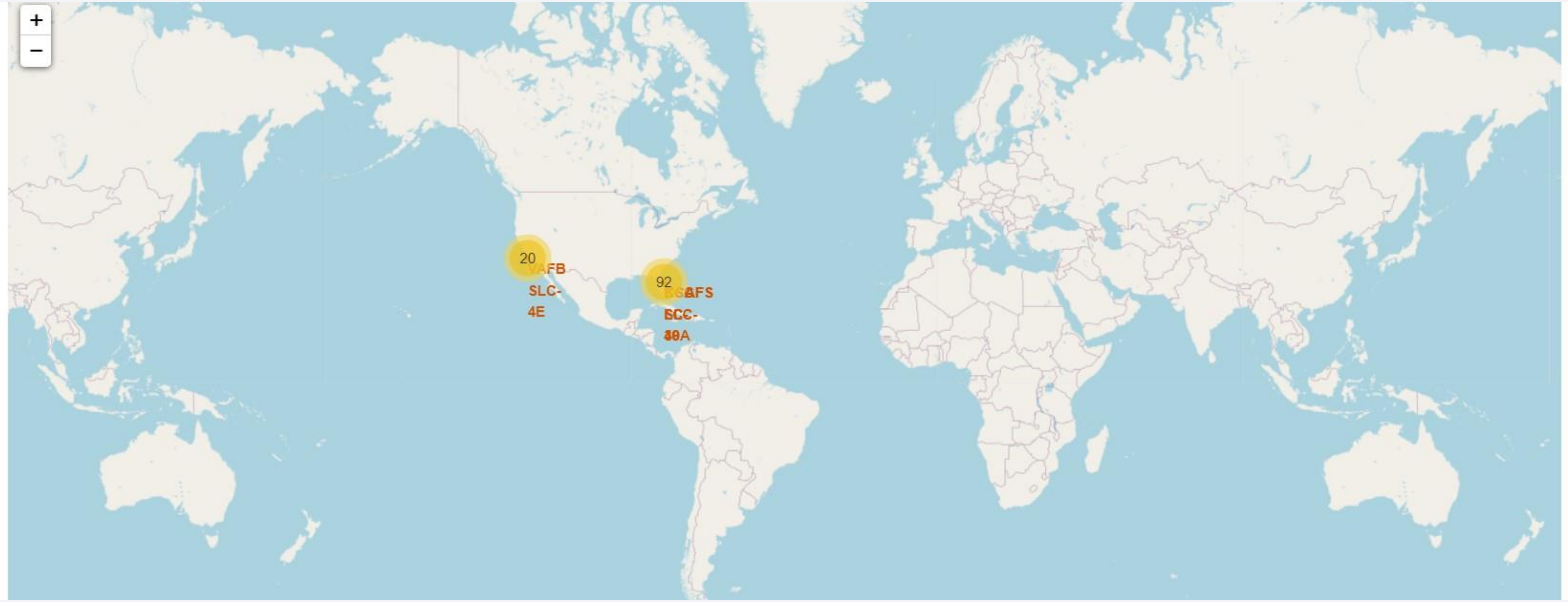
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 4

Launch Sites Proximities Analysis

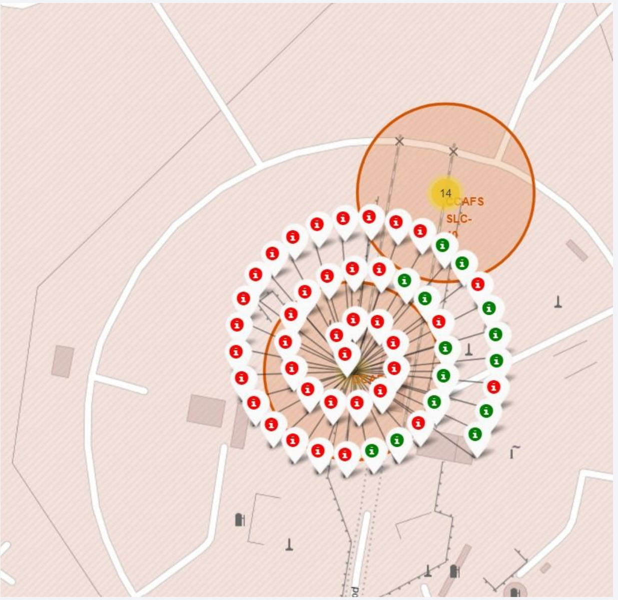
A global map that shows all launch sites markers

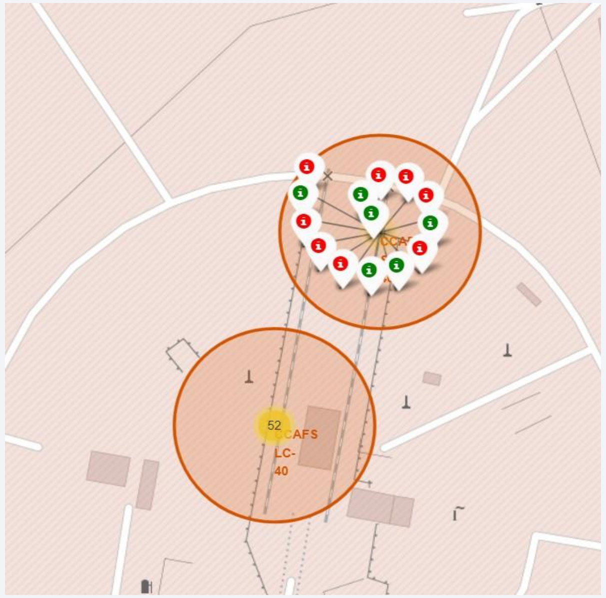
- SpaceX launch sites in the United States

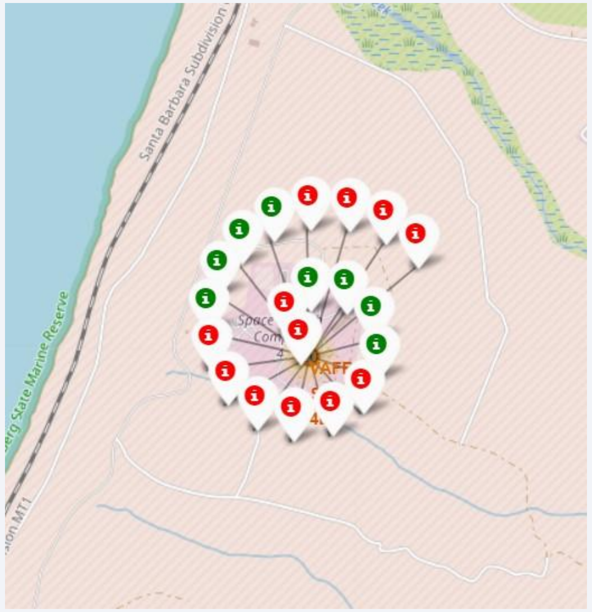


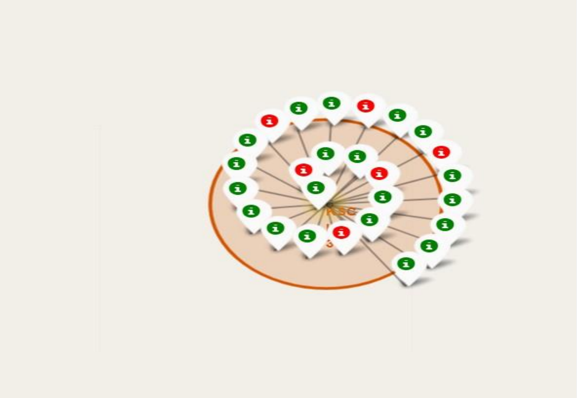
Launch sites labeled and marked with colors Green and Red for success and failure respectively

- Green markers shows a successful launch and Red markers shows a failure



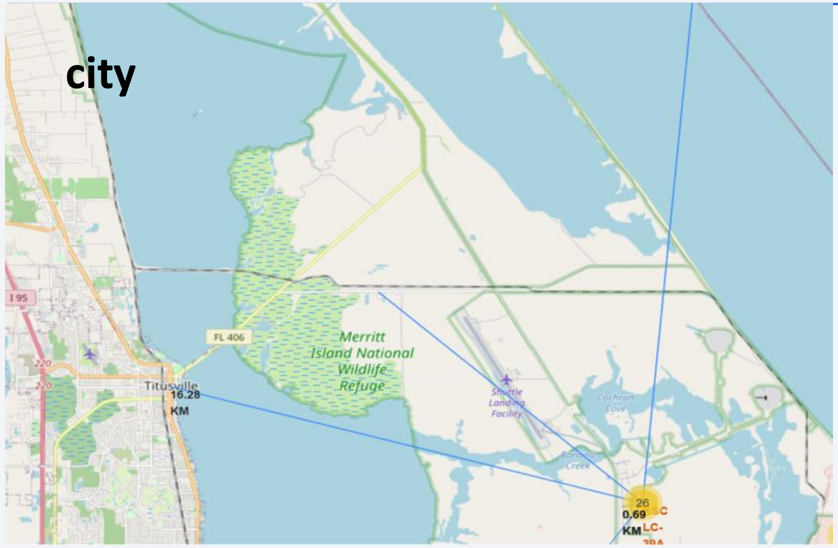




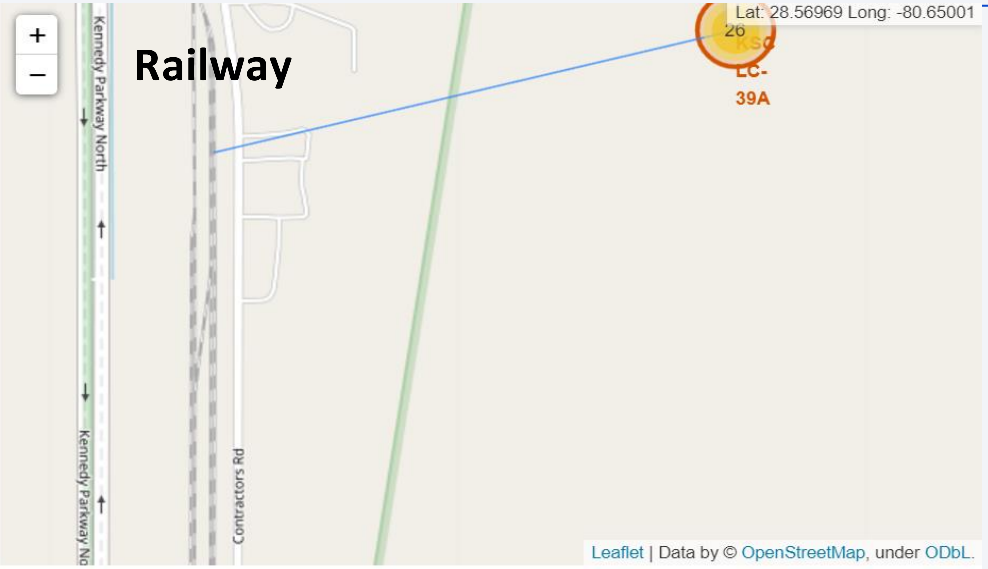


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Launch site approximate distance to landmarks

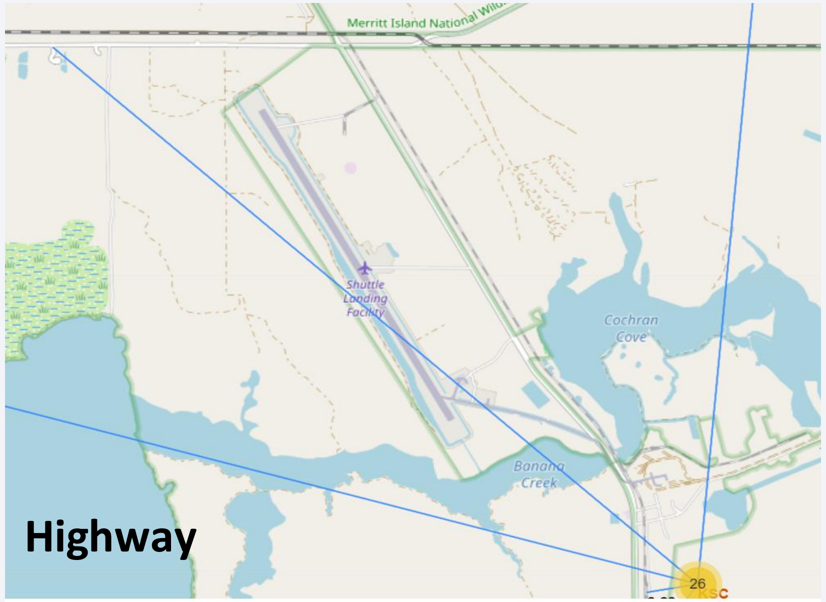


city

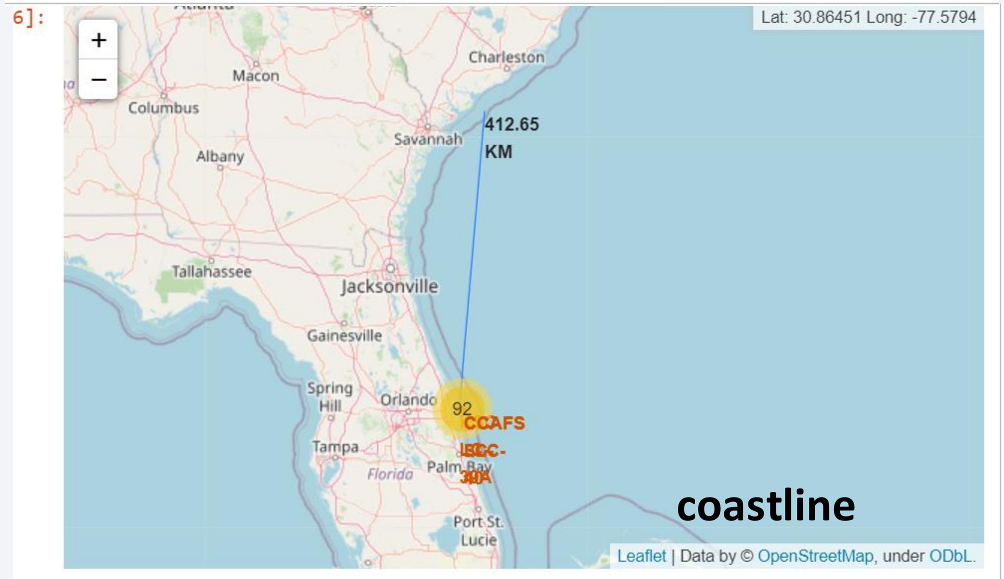


Railway

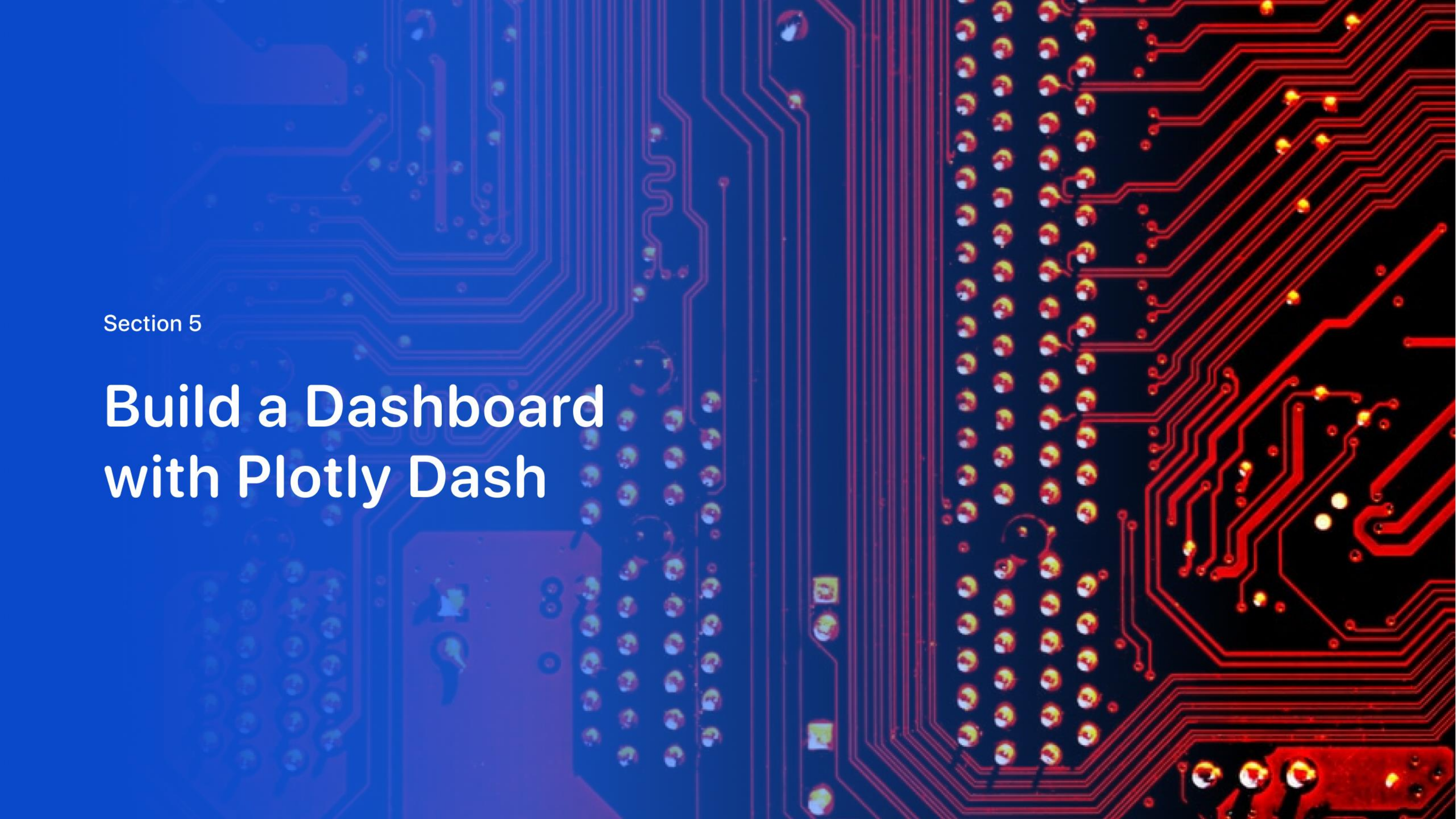
+
−
Lat: 28.56969 Long: -80.65001
26
KSC
LC-
39A
Kennedy Parkway North
Kennedy Parkway No
Contractors Rd
Leaflet | Data by © OpenStreetMap, under ODbL.



Highway



coastline

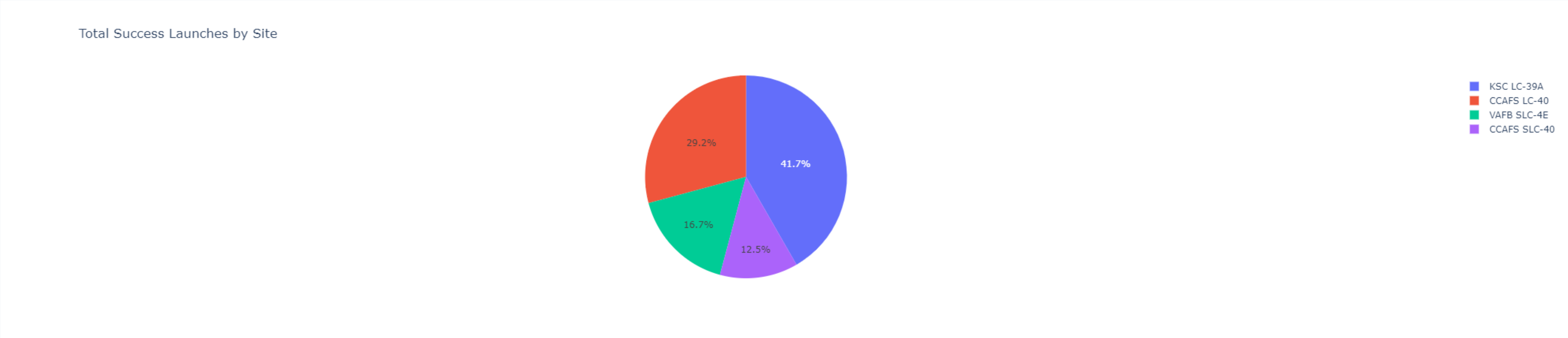


Section 5

Build a Dashboard with Plotly Dash

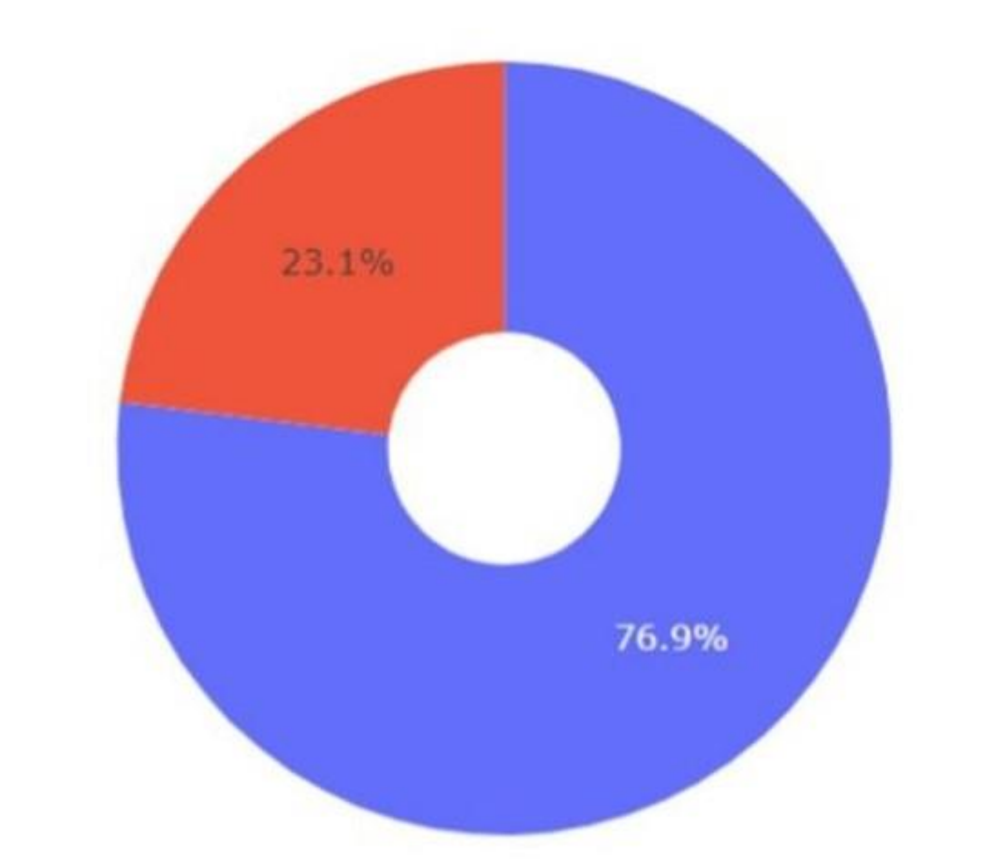
Pie chart showing the success percentage achieved by each launch site

- We can see that KSC LC-39A is the most successful launch site.



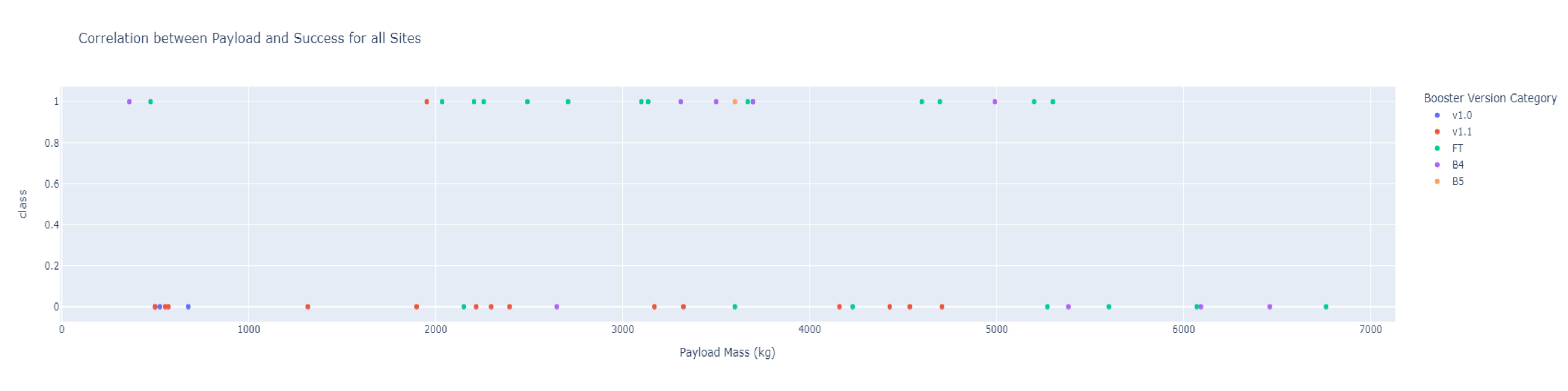
Pie chart showing the Launch site with the highest launch success ratio

- KSC LC-39A achieved the highest success rate with 76.9 successful launches



Payload vs. Launch Outcome scatter plot for all sites

- The scatterplot shows that lighter payload have higher success rate.

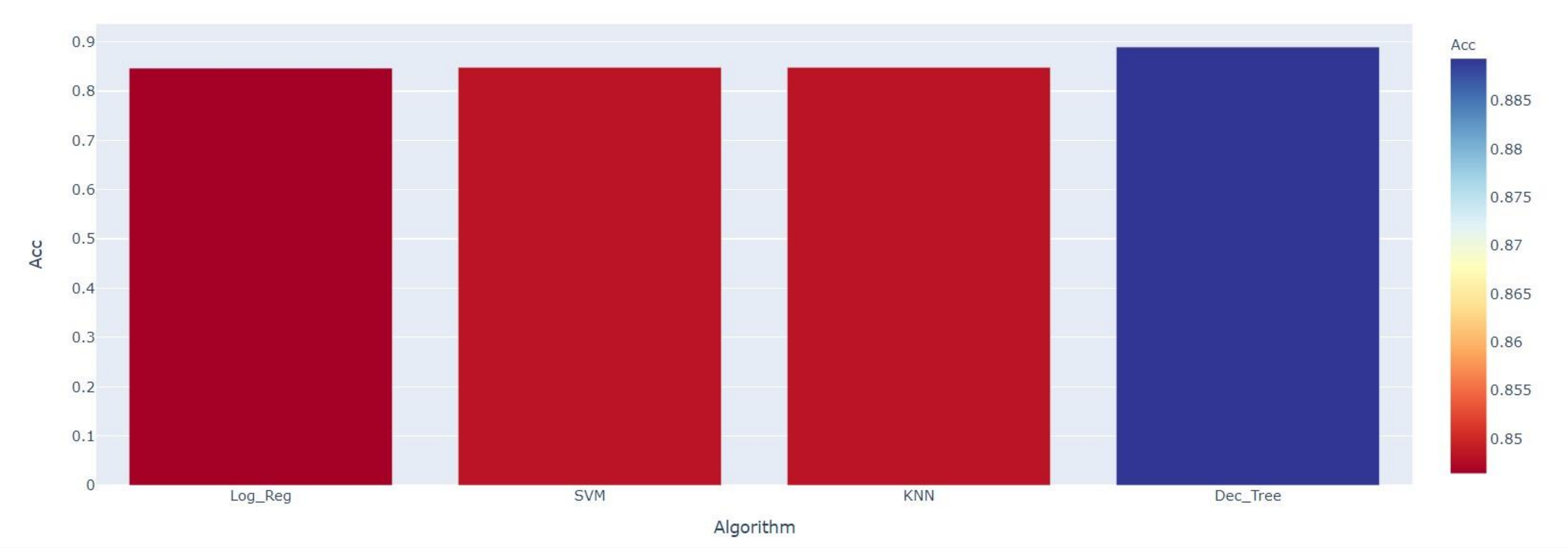


Section 6

Predictive Analysis (Classification)

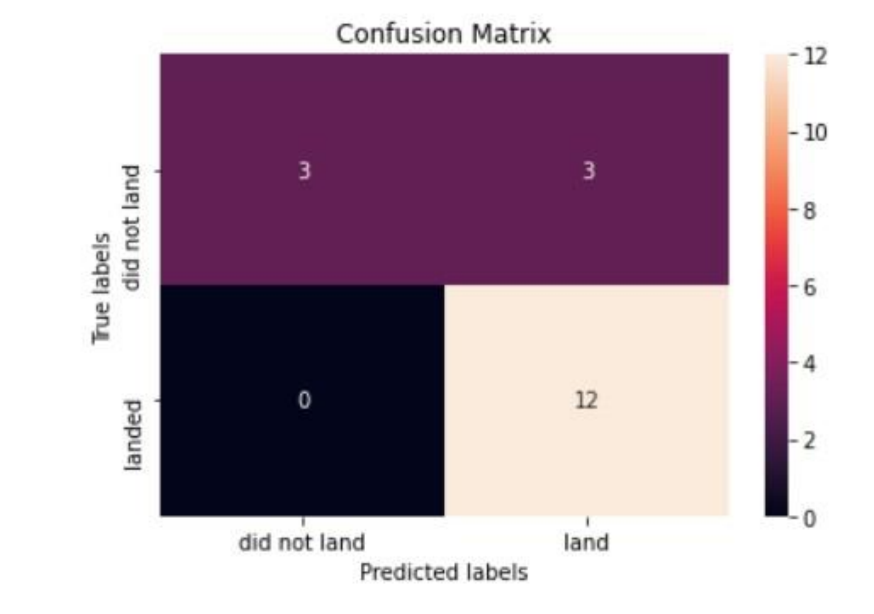
Classification Accuracy

- Bar chart shows the accuracy for all built classification models



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

- **In conclusion:**
  - The greater the number of flight from a launch site, the higher the success rate.
  - Successful launch rate is increasing each year with a slight drop in 2017.
  - ES-L1, GEO, HEO and SSO had the most success per Orbit.
  - KSC LC-39A achieved the highest success rate with 76.9 successful launches.
  - The Decision tree classifier is the best machine learning algorithm

Appendix

- here is a list of links to different GitHub repositories that includes outside resources:
- Datasets 🔗
- Screenshots 🔗
- GitHub Repository 🔗
- App 🔗

Thank you!