IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ayuri Limje

12-04-2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scrapping
  - Data Wrangling
  - EDA with SQL
  - EDA with Data Visualization
  - Interactive Visual Analysis with Folium
  - Machine Learning Prediction
- Summary of all results
  - EDA Result
  - Interactive Analysis in Screenshot
  - Predictive Analytics result for Machine Learning Lab

Introduction

- SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore, as a Data Scientist using given data and Machine Learning Model we can predict the landing outcome at first stage in the future. This information also can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- Problems included:
  - Identifying the outcomes which influence the landing factor
  - Relationship between variables and how it affecting the outcome
  - Best condition needed to increase the probability of Successful Landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - Describe how data was collected
- Perform data wrangling
    - Describe how data was processed
- 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

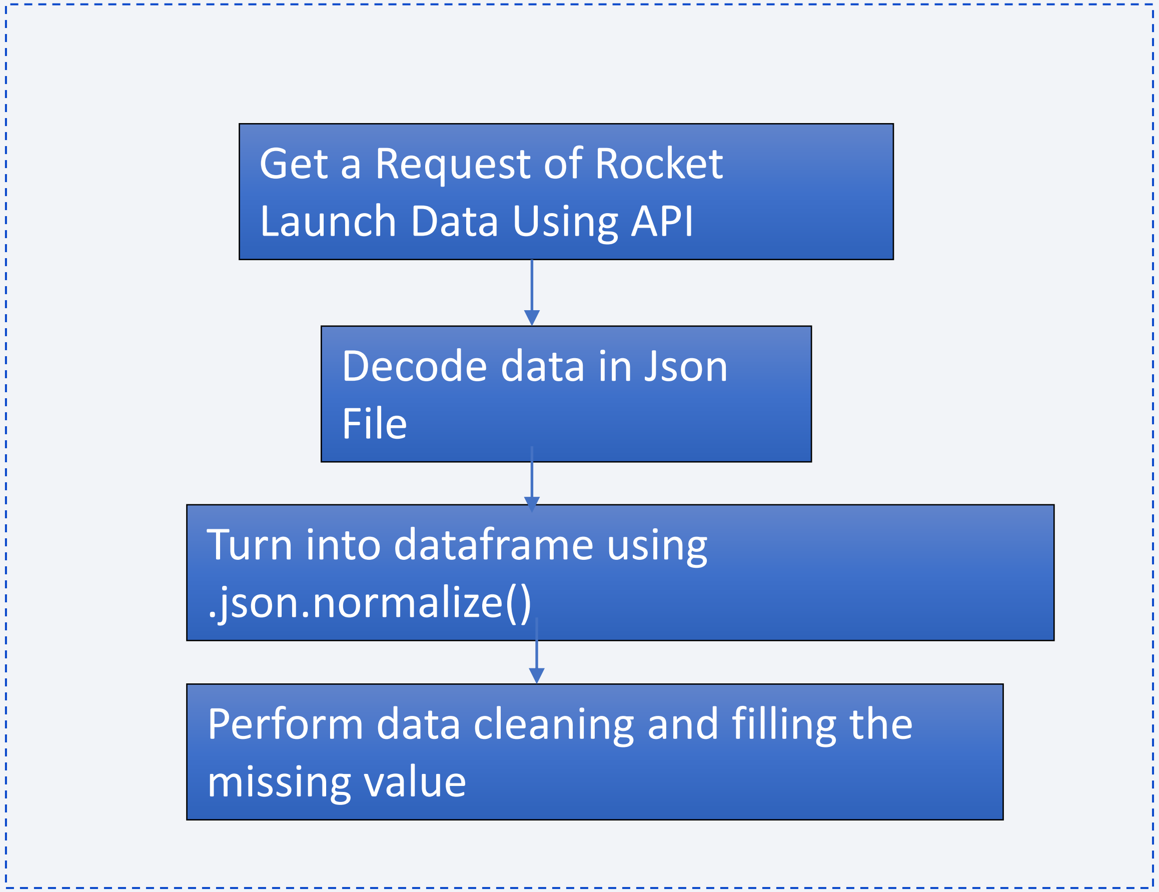
Data Collection

- Data was collected API mainly the SpaceX Rest API and other data source is Falcon 9 launch data from Wiki pages which was extracted using Webscrapping
- In API, starting with getrequest(), later we decoded the data in json file and turning it to data frame using .json.normalize(). Later we removed the unwanted data and creating new data frame with data which we will need. We then cleaned the data, checked for missing values and fill with whatever needed
- For web scrapping, we will use the BeautifulSoup to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for further analysis

Data Collection – SpaceX API

Git URL:

https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Get a Request of Rocket Launch Data Using API

Decode data in Json File

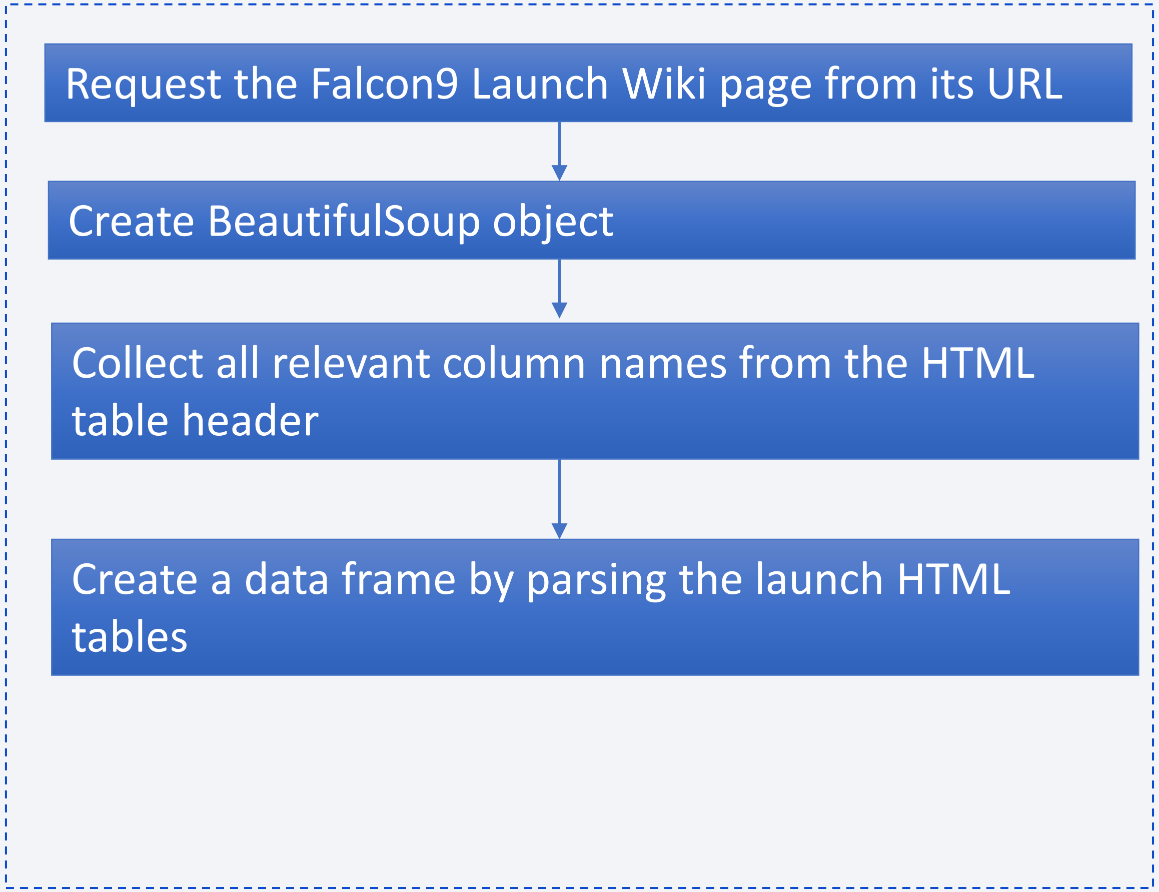
Turn into dataframe using .json.normalize()

Perform data cleaning and filling the missing value

Data Collection - Scraping

Git URL:

https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/jupyter-labs-webscraping.ipynb



Request the Falcon9 Launch Wiki page from its URL

Create BeautifulSoup object

Collect all relevant column names from the HTML table header

Create a data frame by parsing the launch HTML tables

Data Wrangling

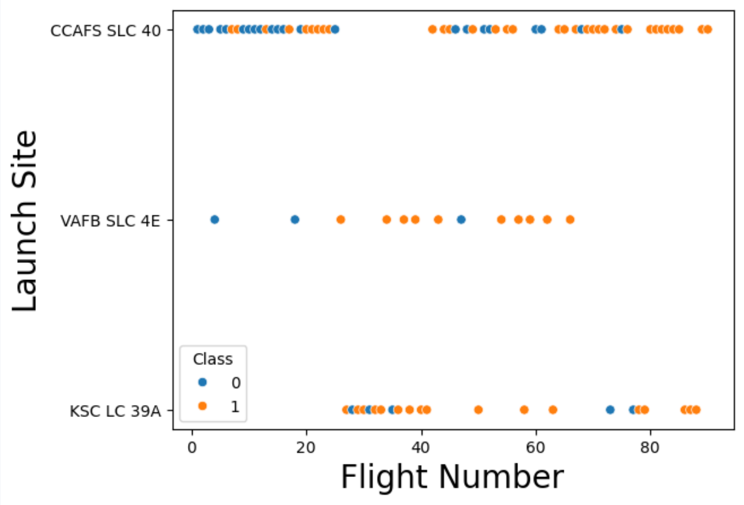
- Data Wrangling is the process of transforming and mapping raw data into a more usable and structured format for analysis and other purposes
- Here, we calculated the % of missing value, type of the columns and afterward calculated the different parameters like number of launches on each site, mission outcomes of the orbits etc using value_count()
- Later we created a classification variable that represent the outcome of the each launch

Git URL: https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

We uses the scatterplot to find the relationship between the attributes:

- Payload and Flight Number
- Flight Number and Launch Site
- Payload and Launch Site
- Flight Number and Orbit Type
- Pay Load and Orbit Type



Scatter plots primarily for visualizing the relationship between two continuous variables in a dataset. Once, the pattern is determined from the graph, its very easy to see which factors affecting the most to the success of landing outcomes.

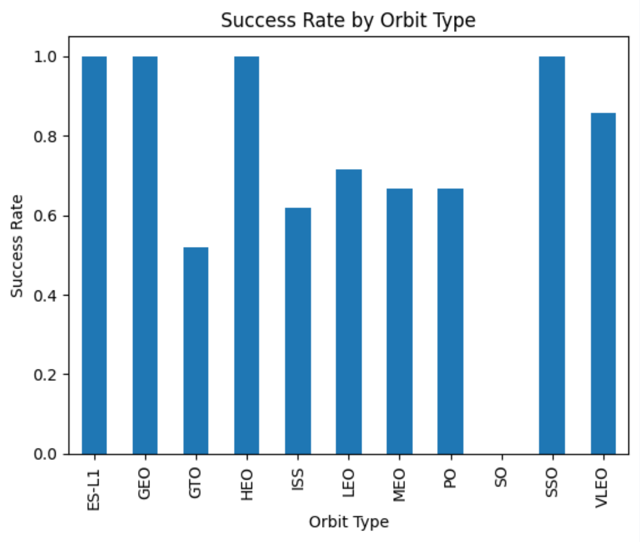
Git URL: https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/edadataviz.ipynb

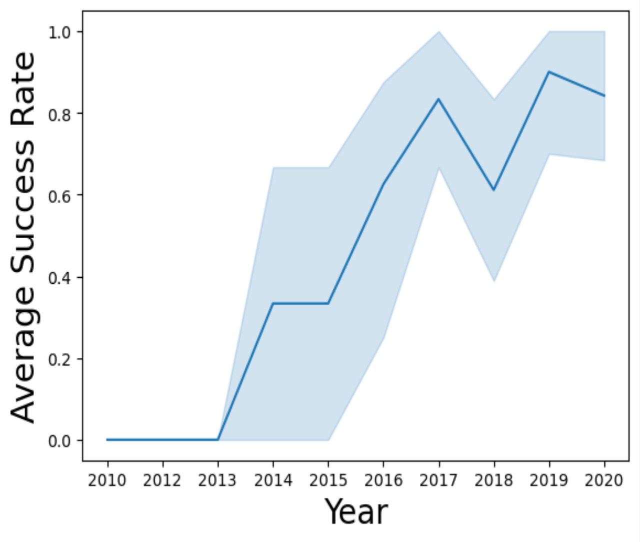
Note : More details on EDA with Data Visualization in below slide

EDA with Data Visualization

- We further than use the Bar chart and Line plot for Visualization
- **Bar chart** we use for visualizing and comparing categorical data. In this case we use the bar chart to determine which orbit has highest probability of success
- **Line chart** we use to visualize the trend or the pattern over the period using different attributes. In this case, we use to visualize the launch success yearly trend
- We then use the **Feature Engineering** to predict the success in future module by creating dummy variables to categorical columns

Git URL: https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/edadataviz.ipynb





EDA with SQL

- Using SQL, we performed EDA to understand data set:
    - Display the names of the unique launch sites in the space mission
    - Display 5 records where launch sites begin with the string 'CCA'
    - Display the total payload mass carried by boosters launched by NASA (CRS)
    - Display average payload mass carried by booster version F9 v1.1
    - List the date when the first successful landing outcome in ground pad was achieved.
    - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
    - List the total number of successful and failure mission outcomes
    - List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
    - List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
    - 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.

Git URL: https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We marked launch sites as well as success and failure launches using map object such as marker and circle. We calculated the distance a launch site to it proximities
- Markers and Circle objects in Folium are used for visualizing specific points of interest (Markers) and areas of influence or coverage (Circles) on interactive maps, enhancing map interactivity and conveying spatial information effectively
- Using color labelled marked cluster, we identified which launch sites have high success rate
- We calculated the distance a launch site to it proximities, while answering some question like:
    - How close the launch site with railways, highways and coastline
    - Do launch site have certain distance from cities

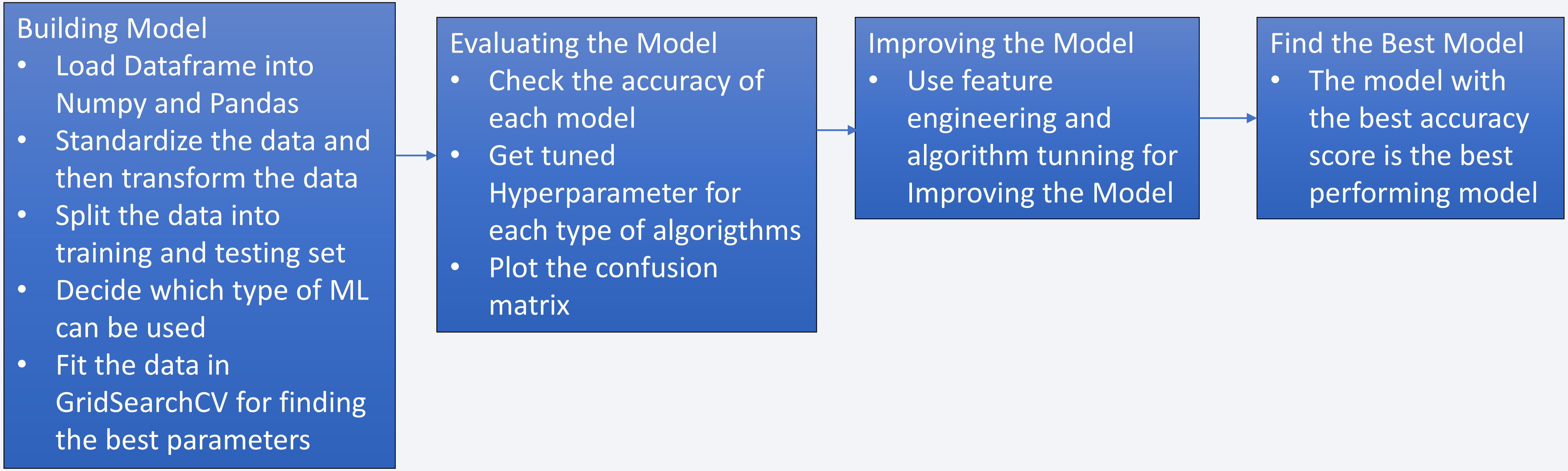
Git URL:
https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- We have use Plotly dash to build interactive dashboard, by adding dropdowns whose value affects the graphs
- We plotted Pie Chart showing the total launches by all sites as well as total launches from a certain cites
- We plotted the Scatter Plot showing the relationship between outcome and payload mass kg for different booster version
- We also used Ranger Slider to select a range of values by sliding handles along a track.

Git URL:
https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)



Building Model

- Load Dataframe into Numpy and Pandas
- Standardize the data and then transform the data
- Split the data into training and testing set
- Decide which type of ML can be used
- Fit the data in GridSearchCV for finding the best parameters

Evaluating the Model

- Check the accuracy of each model
- Get tuned Hyperparameter for each type of algorigthms
- Plot the confusion matrix

Improving the Model

- Use feature engineering and algorithm tunning for Improving the Model

Find the Best Model

- The model with the best accuracy score is the best performing model

Git URL:
https://github.com/ayuri1512/Final_Spacex_Project_Assignment/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

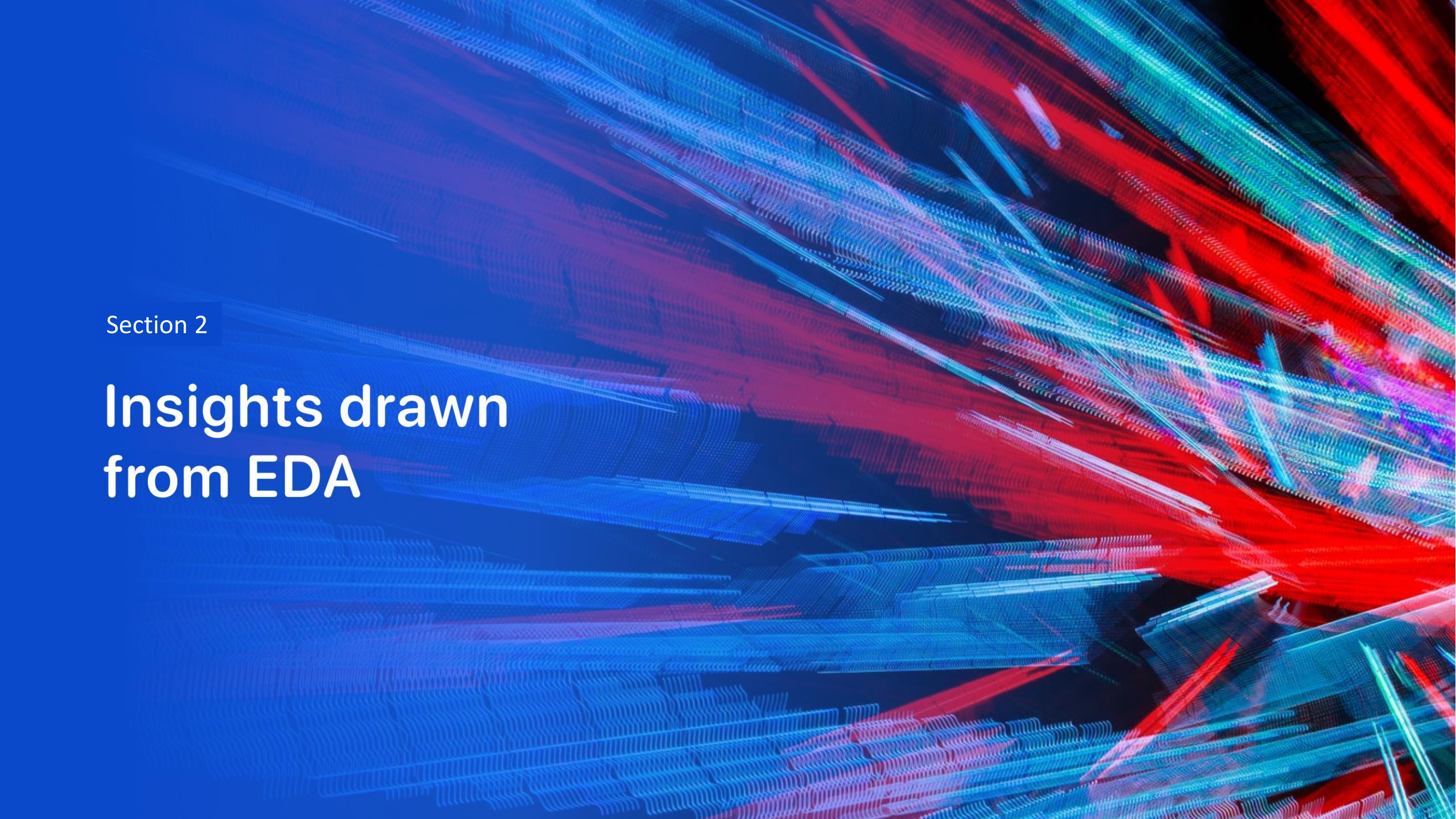
Results

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

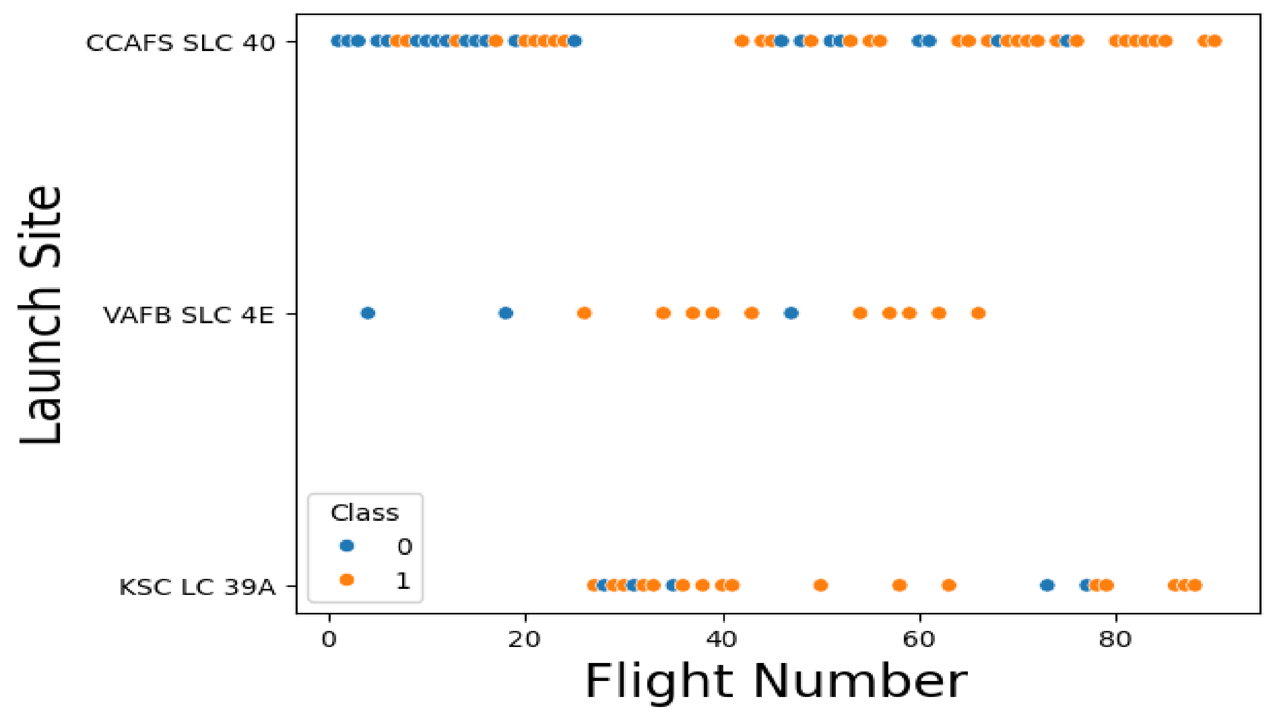
Note: Result are shown in below slides

Section 2

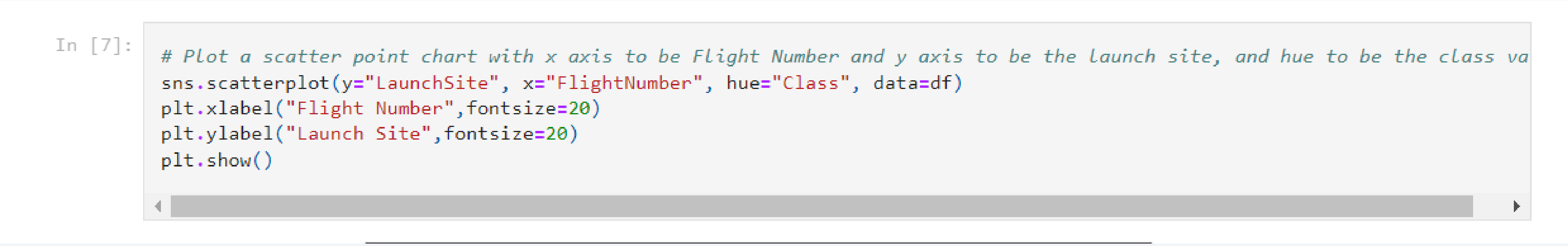
Insights drawn from EDA



Flight Number vs. Launch Site

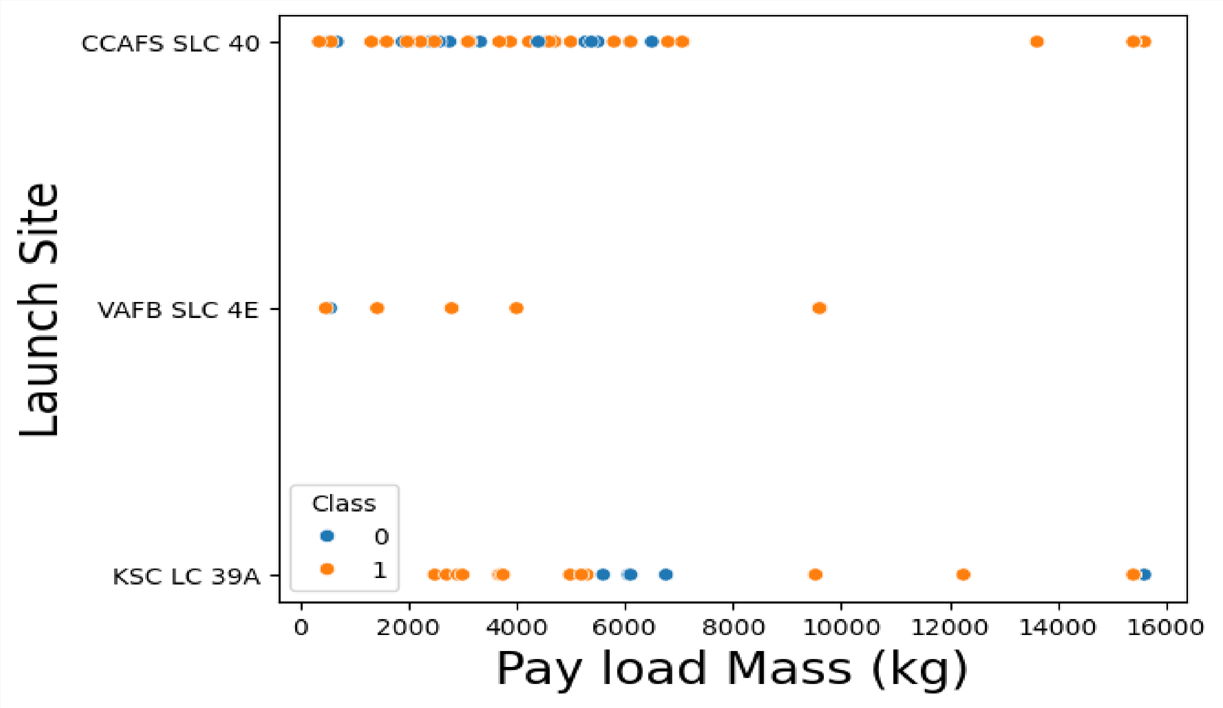


The plot shows that the larger amount of launch has made from the CCAFS SLC 40, while the success rate and failure rate are not much clear as point

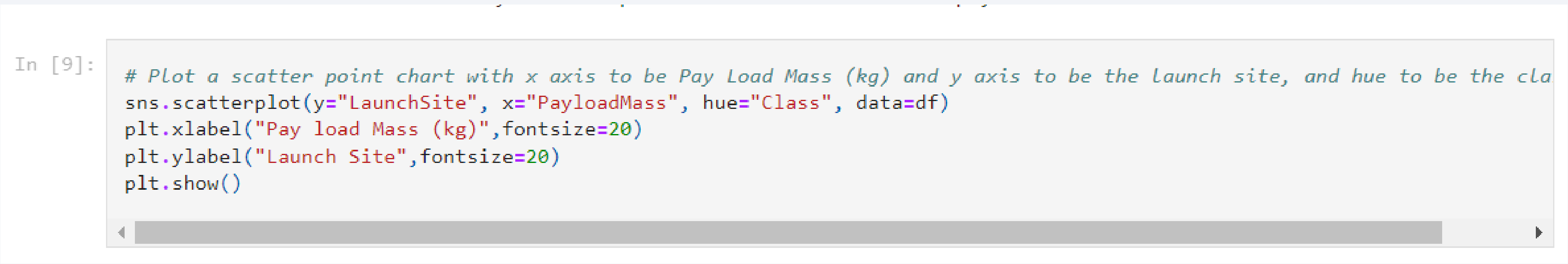


```
In [7]:  # Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class va
         sns.scatterplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df)
         plt.xlabel("Flight Number",fontsize=20)
         plt.ylabel("Launch Site",fontsize=20)
         plt.show()
```

Payload vs. Launch Site



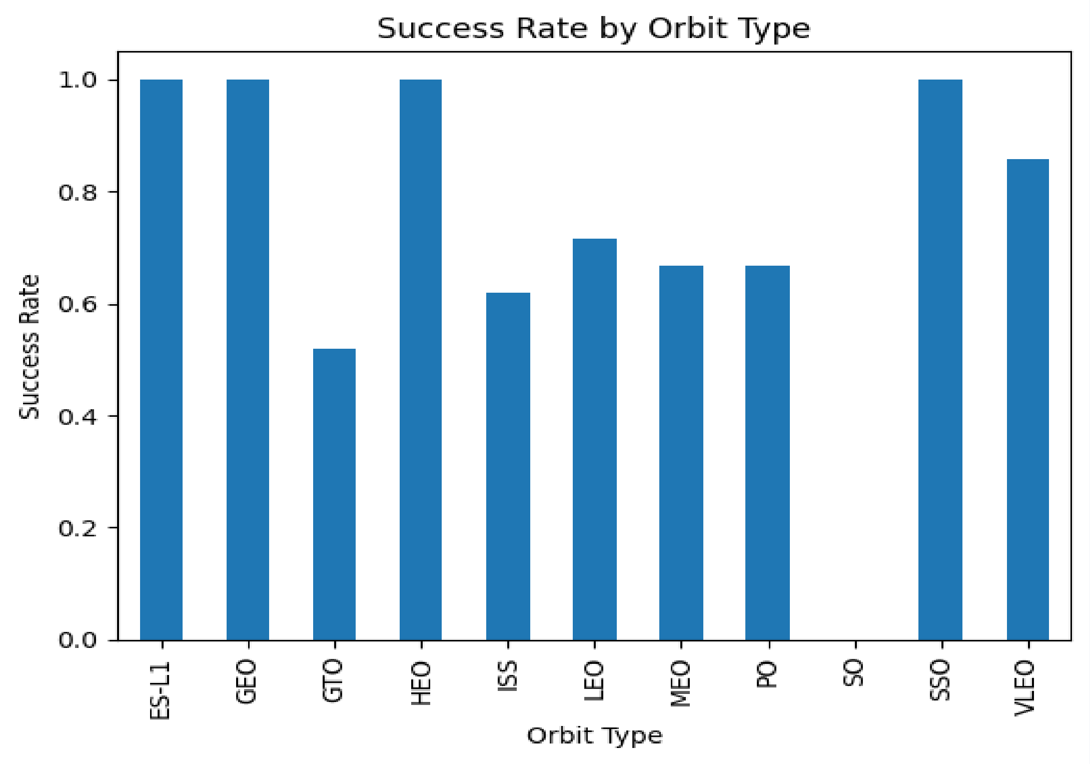
From Payload Vs. Launch Site scatter point chart we can find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).



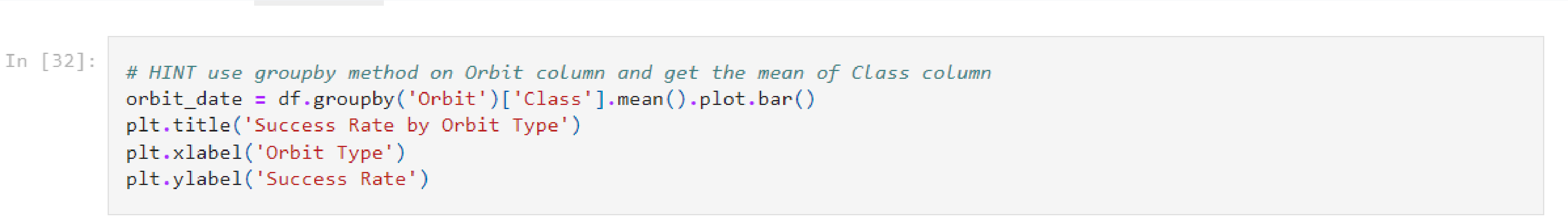
In [9]:

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the cla
sns.scatterplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

Success Rate vs. Orbit Type

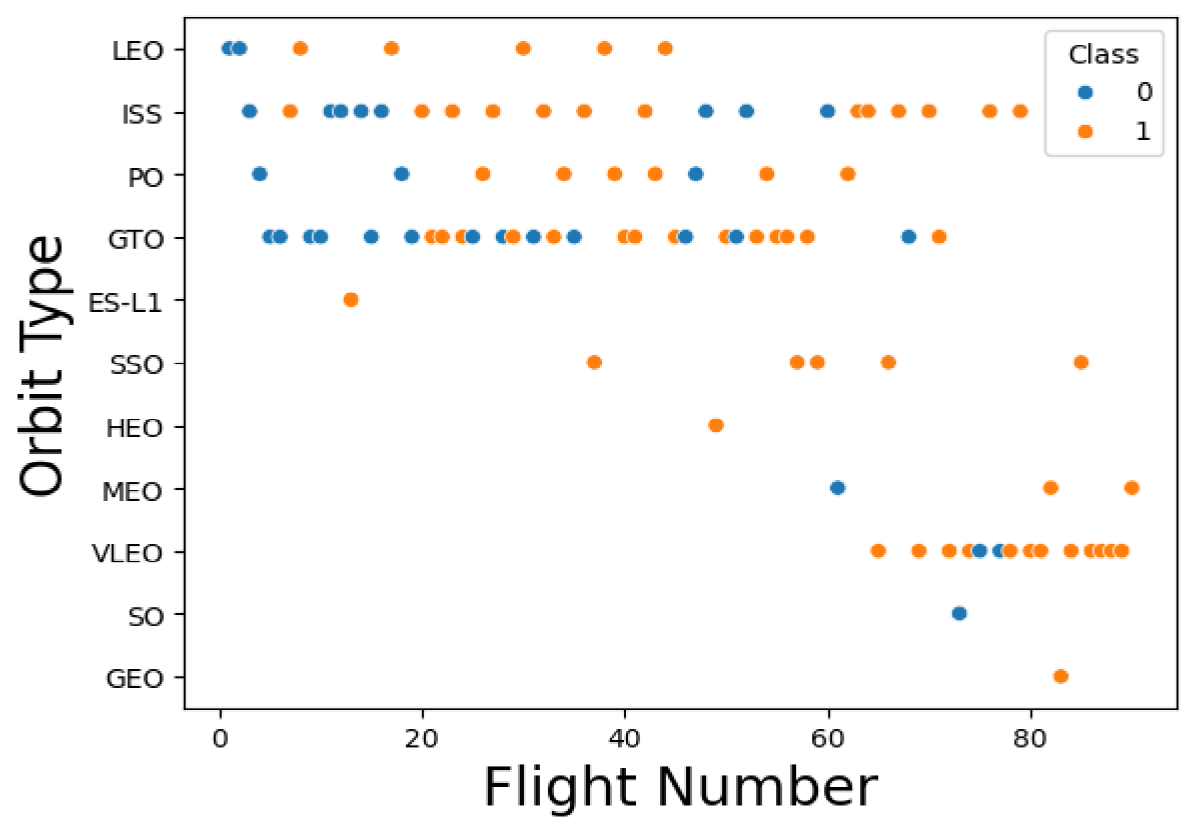


From the chart it can see that the success rate for 4 orbit ES-11, GEO, HEO and SSO is higher as compared to other while SO don’t have the success rate and as only 1 value for SO is there, bar chart is now showing up

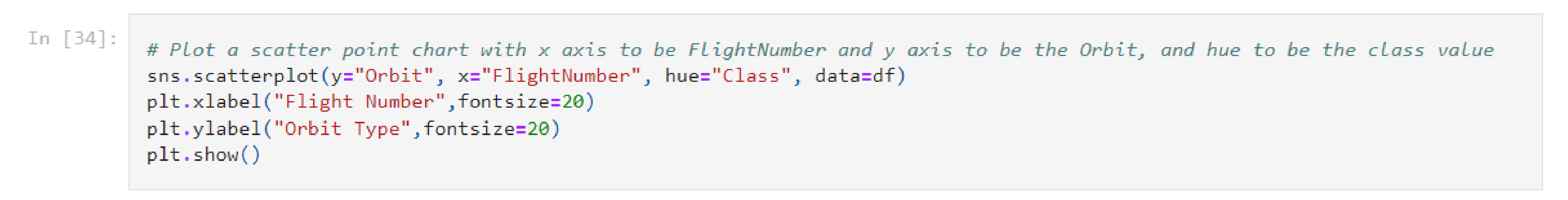


```
In [32]:
# HINT use groupby method on Orbit column and get the mean of Class column
orbit_date = df.groupby('Orbit')['Class'].mean().plot.bar()
plt.title('Success Rate by Orbit Type')
plt.xlabel('Orbit Type')
plt.ylabel('Success Rate')
```

Flight Number vs. Orbit Type

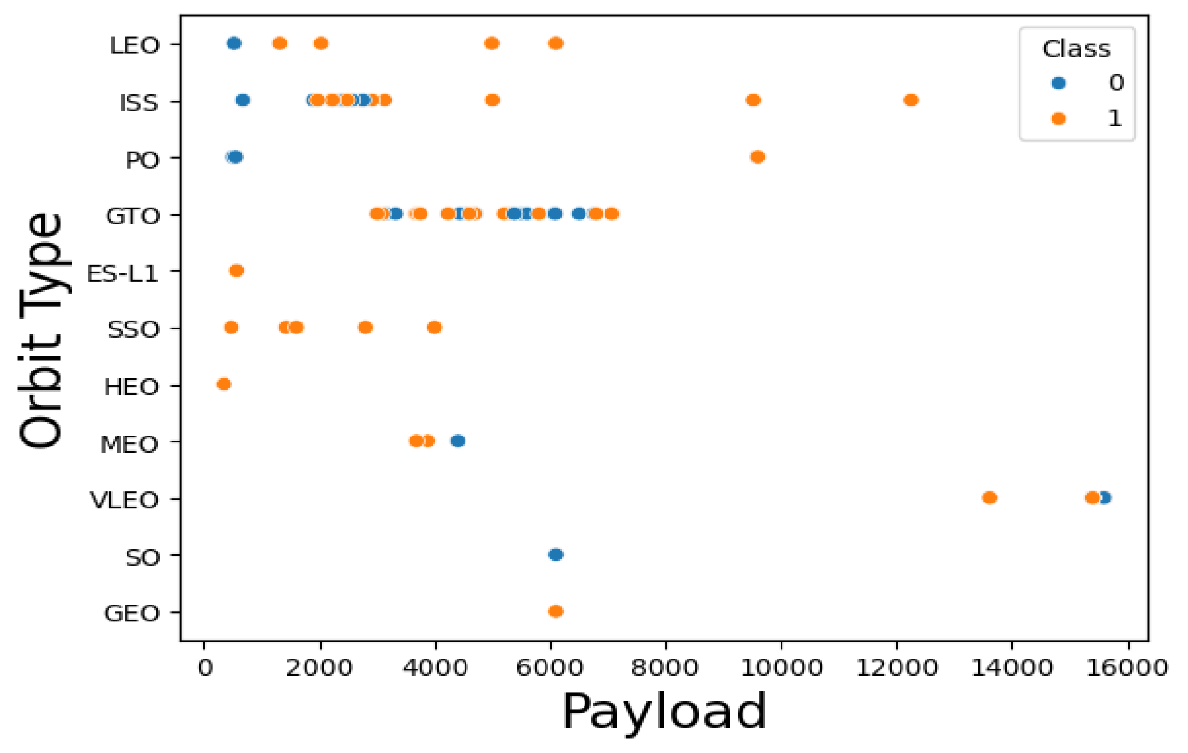


We can see from the plot that the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

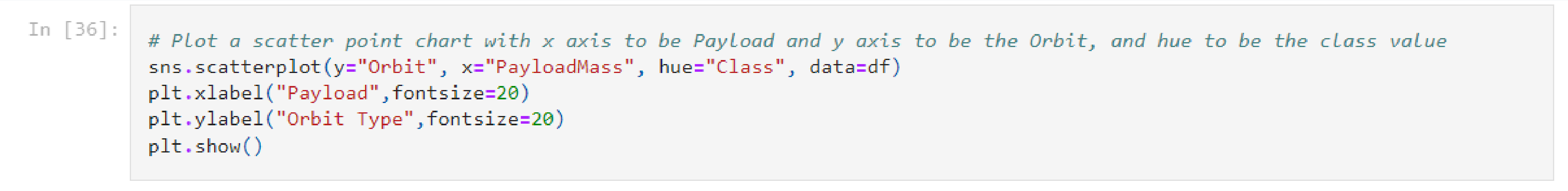


```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.scatterplot(y="Orbit", x="FlightNumber", hue="Class", data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit Type",fontsize=20)
plt.show()
```

Payload vs. Orbit Type



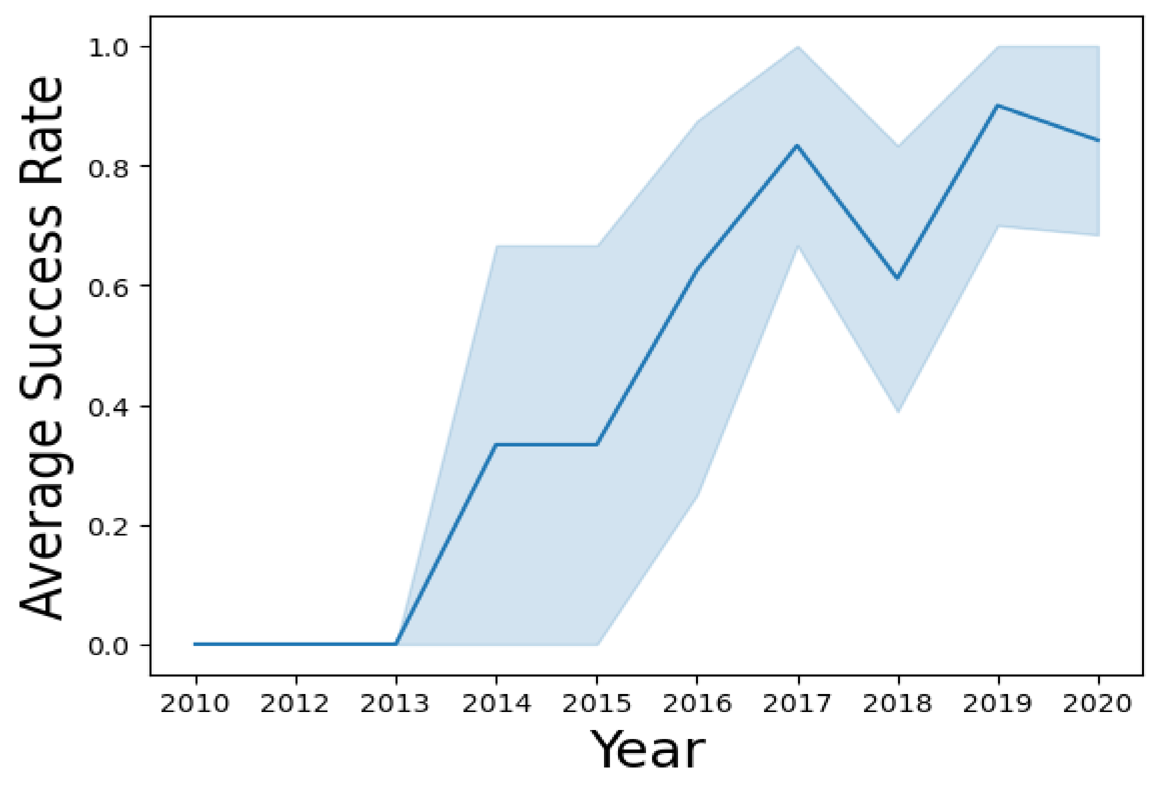
We can visualize that with heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. For GTO we can't visualize the success or failure rate with respect to payload as again points are overlapping



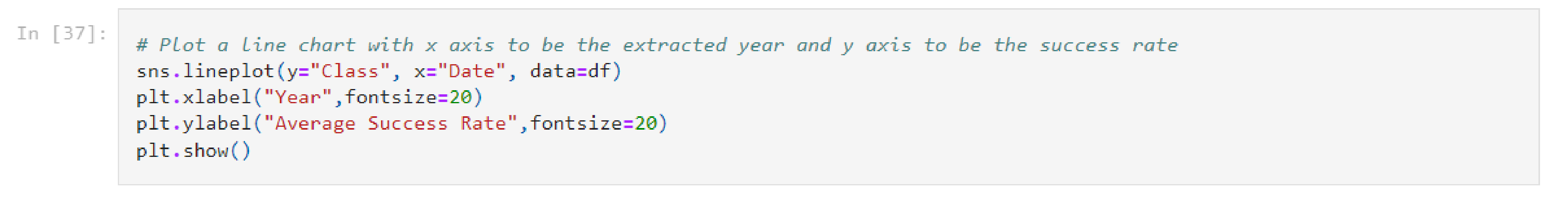
In [36]:

```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.scatterplot(y="Orbit", x="PayloadMass", hue="Class", data=df)
plt.xlabel("Payload",fontsize=20)
plt.ylabel("Orbit Type",fontsize=20)
plt.show()
```

Launch Success Yearly Trend

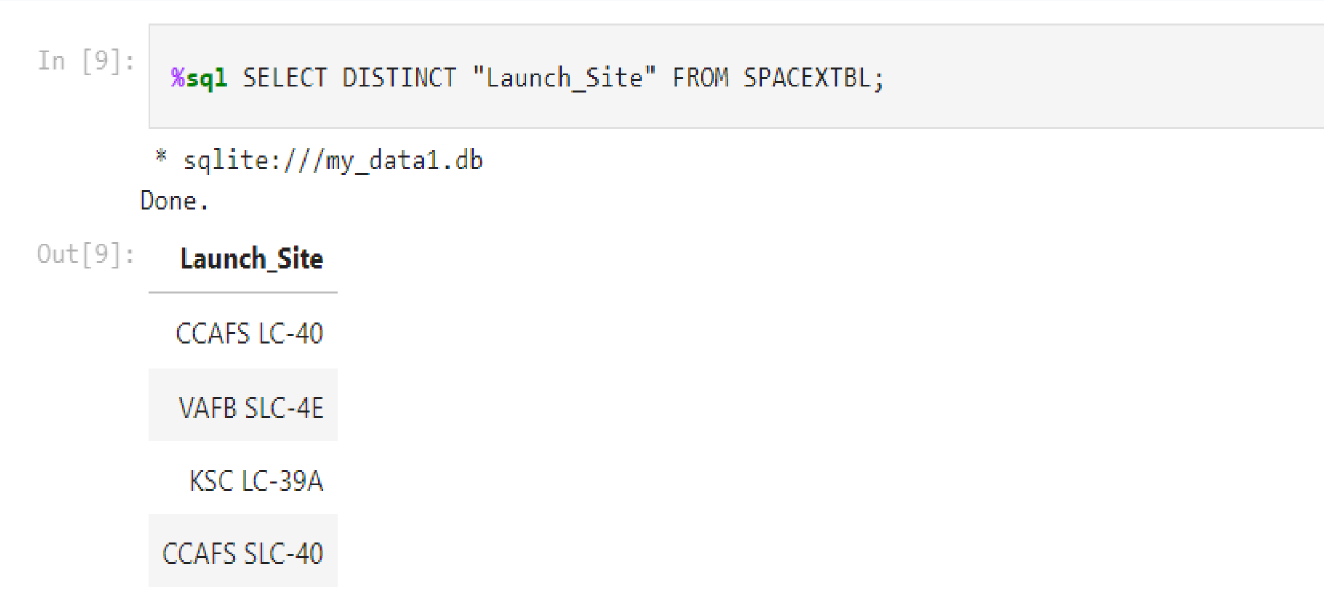


We can see that the success rate since 2013 kept increasing till 2020



```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot(y="Class", x="Date", data=df)
plt.xlabel("Year",fontsize=20)
plt.ylabel("Average Success Rate",fontsize=20)
plt.show()
```

All Launch Site Names



In [9]:

```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTBL;
```

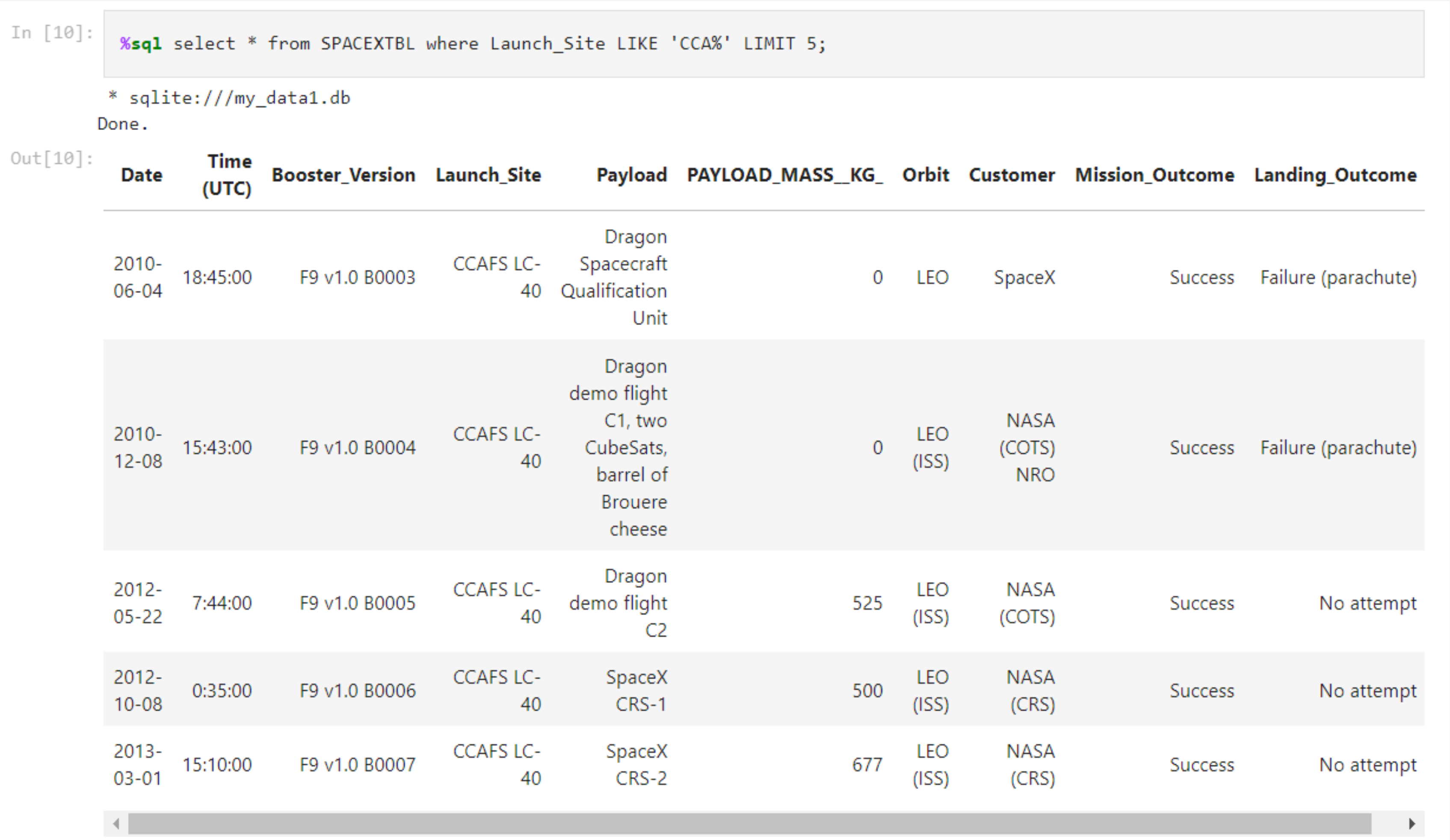
```
 * sqlite:///my_data1.db
Done.
```

Out[9]:

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

We use the DISTINCT keyword to find out the unique launch site names so the repeated data should not show

Launch Site Names Begin with 'CCA'



```
In [10]:  %sql select * from SPACEXTBL where Launch_Site LIKE 'CCA%' LIMIT 5;
```

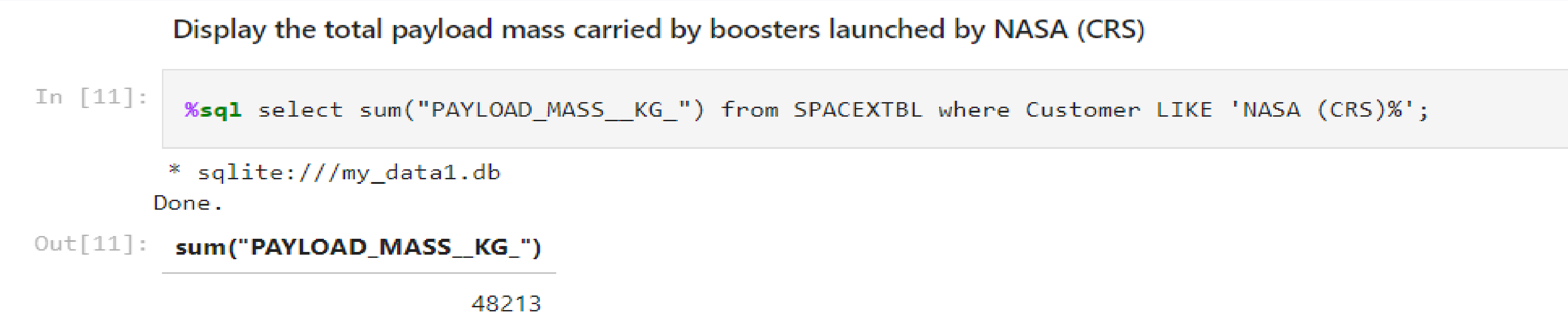
```
 * sqlite:///my_data1.db
Done.
```

Out[10]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Here, we use the LIKE keyword to find the word CCA in Launch site column and showing the 5 rows using LIMIT keyword

Total Payload Mass



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

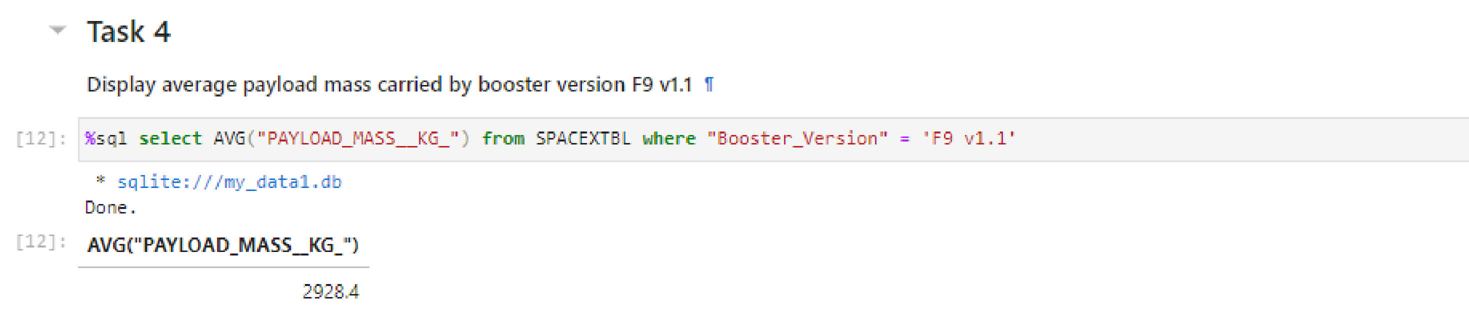
```
In [11]: %sql select sum("PAYLOAD_MASS__KG_") from SPACEXTBL where Customer LIKE 'NASA (CRS)%';
```

```
 * sqlite:///my_data1.db
Done.
```

```
Out[11]: sum("PAYLOAD_MASS__KG_")
                            48213
```

Here we use the Sum(), to find the total payload mass for the customer NASA (CRS) using LIKE keyword, where LIKE keyword will only pick the customer with NASA (CRS)

Average Payload Mass by F9 v1.1



Task 4

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

```
[12]: %sql select AVG("PAYLOAD_MASS__KG_") from SPACEXTBL where "Booster_Version" = 'F9 v1.1'
```

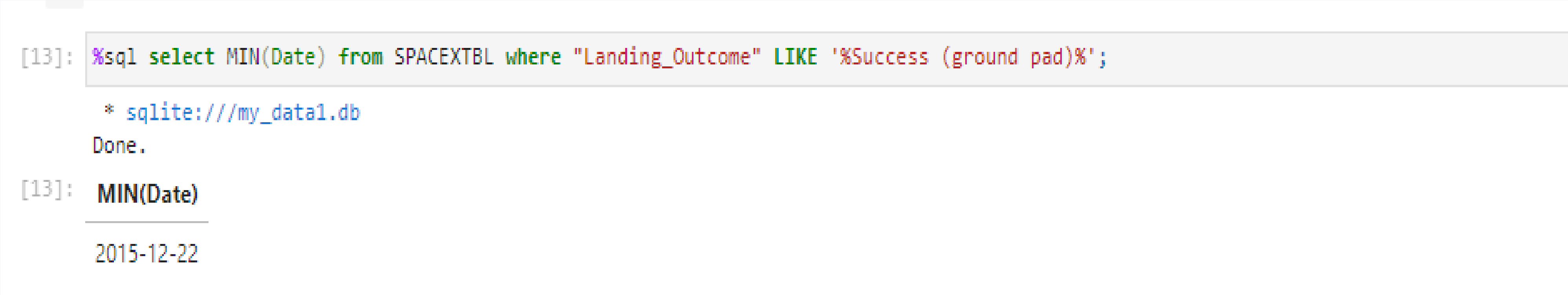
```
 * sqlite:///my_data1.db
Done.
```

[12]:

AVG("PAYLOAD_MASS__KG_")
2928.4

Here using AVG function, we have taken the average where booser version is F(V1.1

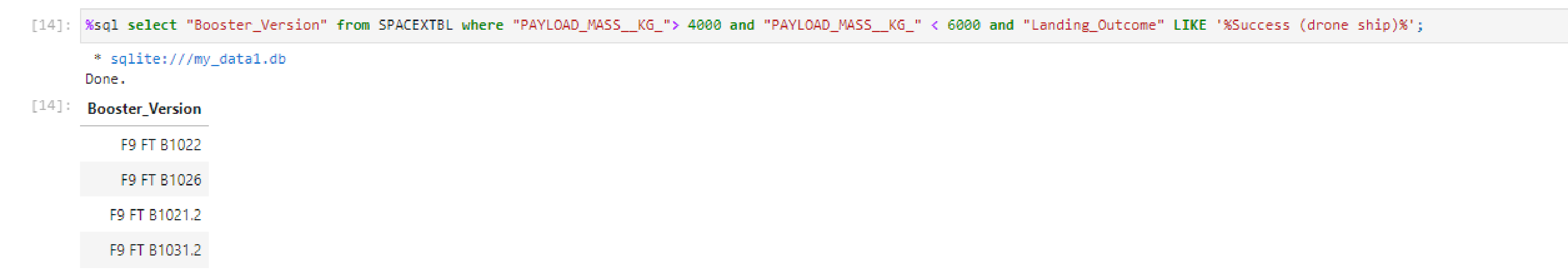
First Successful Ground Landing Date



```
[13]: %sql select MIN(Date) from SPACEXTBL where "Landing_Outcome" LIKE '%Success (ground pad)%';

 * sqlite:///my_data1.db
Done.

[13]: MIN(Date)
      2015-12-22
```

The first successful ground landing happen on 2015-12-22, here we used MIN() function

Successful Drone Ship Landing with Payload between 4000 and 6000



```
[14]: %sql select "Booster_Version" from SPACEXTBL where "PAYLOAD_MASS__KG_"> 4000 and "PAYLOAD_MASS__KG_" < 6000 and "Landing_Outcome" LIKE '%Success (drone ship)%';
```

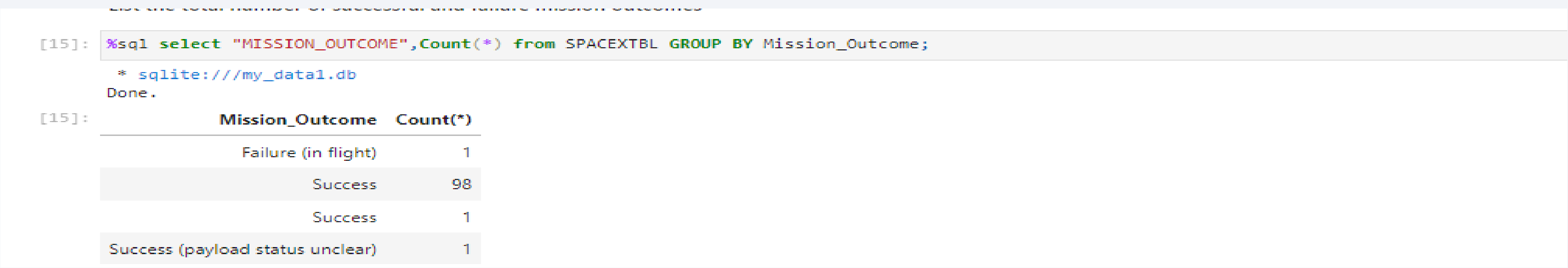
```
 * sqlite:///my_data1.db
Done.
```

[14]:

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Here we found the booster version between the payload 4000 and 6000 having 4 result

Total Number of Successful and Failure Mission Outcomes



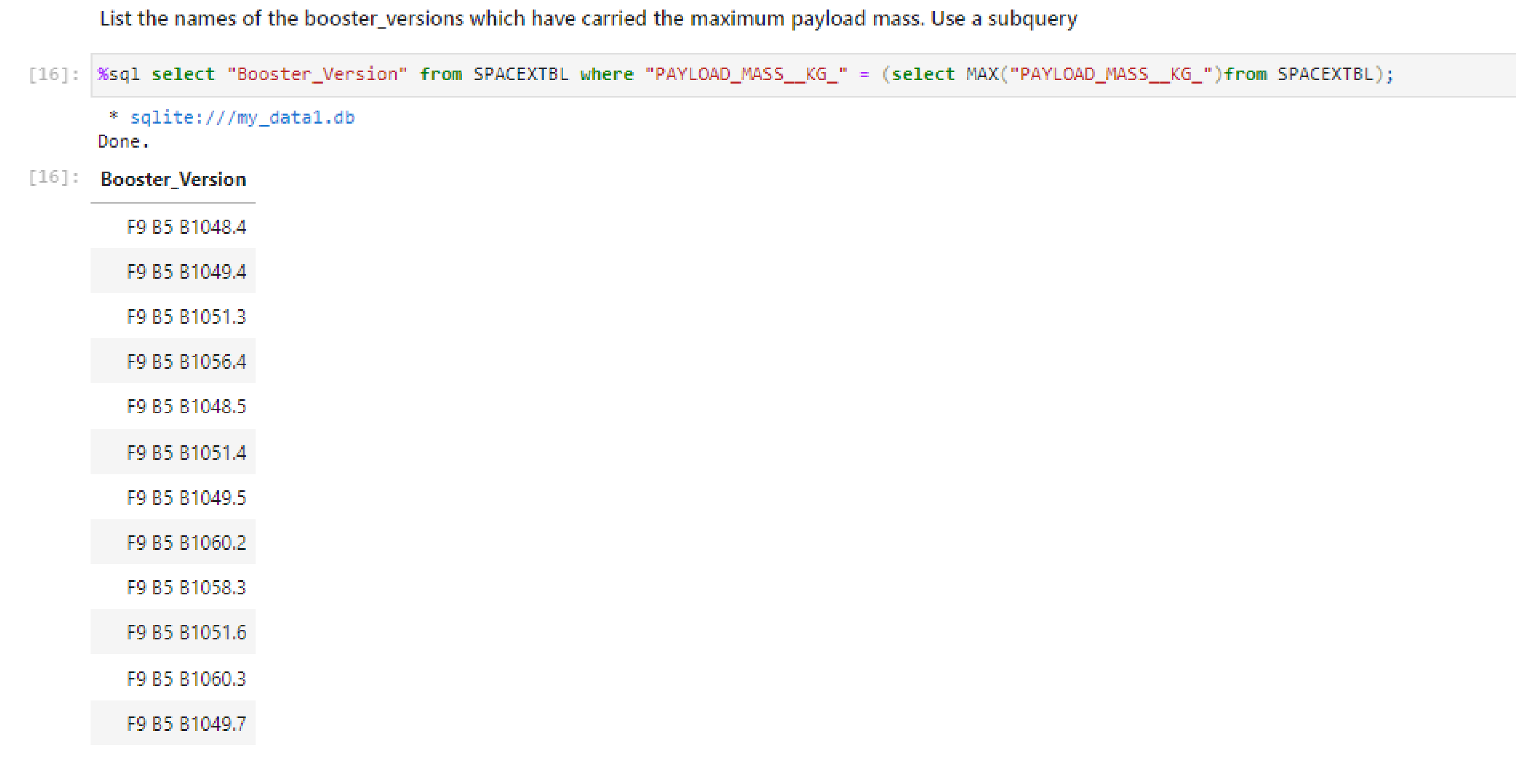
```
[15]: %sql select "MISSION_OUTCOME",Count(*) from SPACEXTBL GROUP BY Mission_Outcome;

       * sqlite:///my_data1.db
      Done.
```

[15]:

Mission_Outcome	Count(*)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Here we can calculate the outcomes using count(*)

Boosters Carried Maximum Payload



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

```
[16]: %sql select "Booster_Version" from SPACEXTBL where "PAYLOAD_MASS__KG_" = (select MAX("PAYLOAD_MASS__KG_")from SPACEXTBL);
```

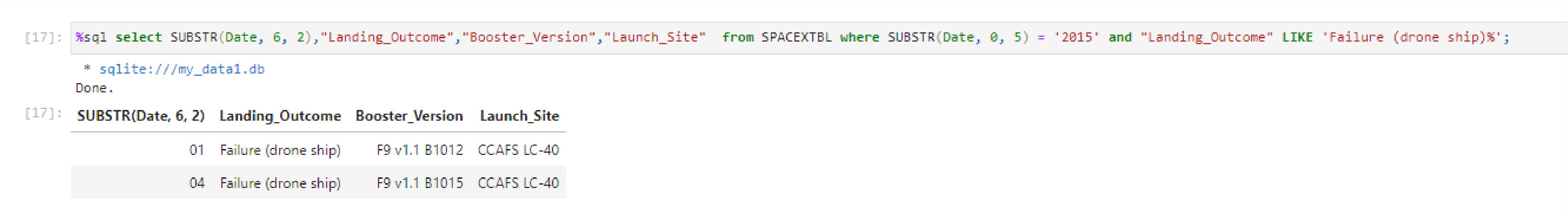
```
 * sqlite:///my_data1.db
Done.
```

[16]:

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

We have used subquery here to find the Max payload using MAX()

2015 Launch Records



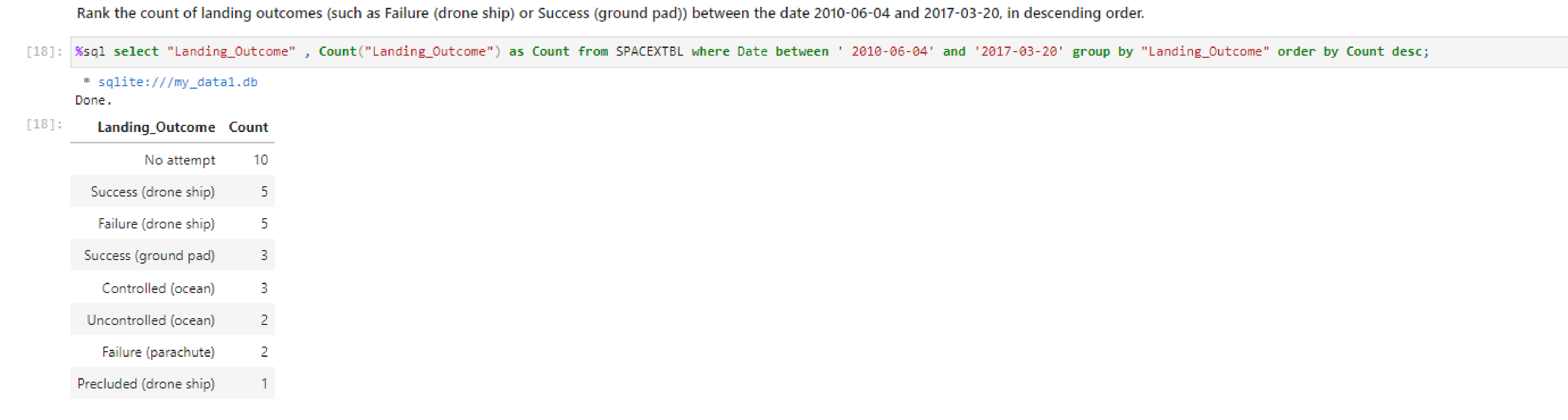
```
[17]: %sql select SUBSTR(Date, 6, 2),"Landing_Outcome","Booster_Version","Launch_Site"  from SPACEXTBL where SUBSTR(Date, 0, 5) = '2015' and "Landing_Outcome" LIKE 'Failure (drone ship)%';

 * sqlite:///my_data1.db
Done.
```

[17]:

SUBSTR(Date, 6, 2)	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Here we used the combination of where clause and different condition to find the data

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.

```
[18]: %sql select "Landing_Outcome" , Count("Landing_Outcome") as Count from SPACEXTBL where Date between ' 2010-06-04' and '2017-03-20' group by "Landing_Outcome" order by Count desc;
```

```
 * sqlite:///my_data1.db
Done.
```

[18]:

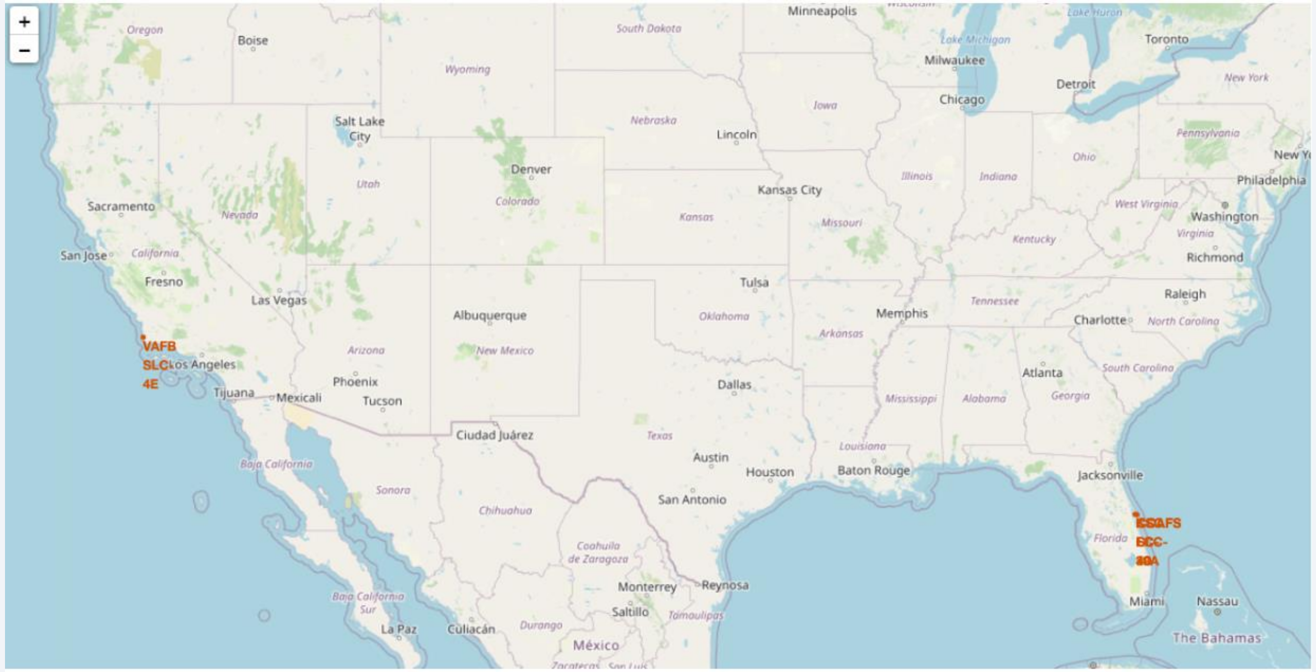
Landing_Outcome	Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Here we have taken the Count of Landing outcome with setting different condition in where and using groupby to group the values and ordering it in descending order

Section 3

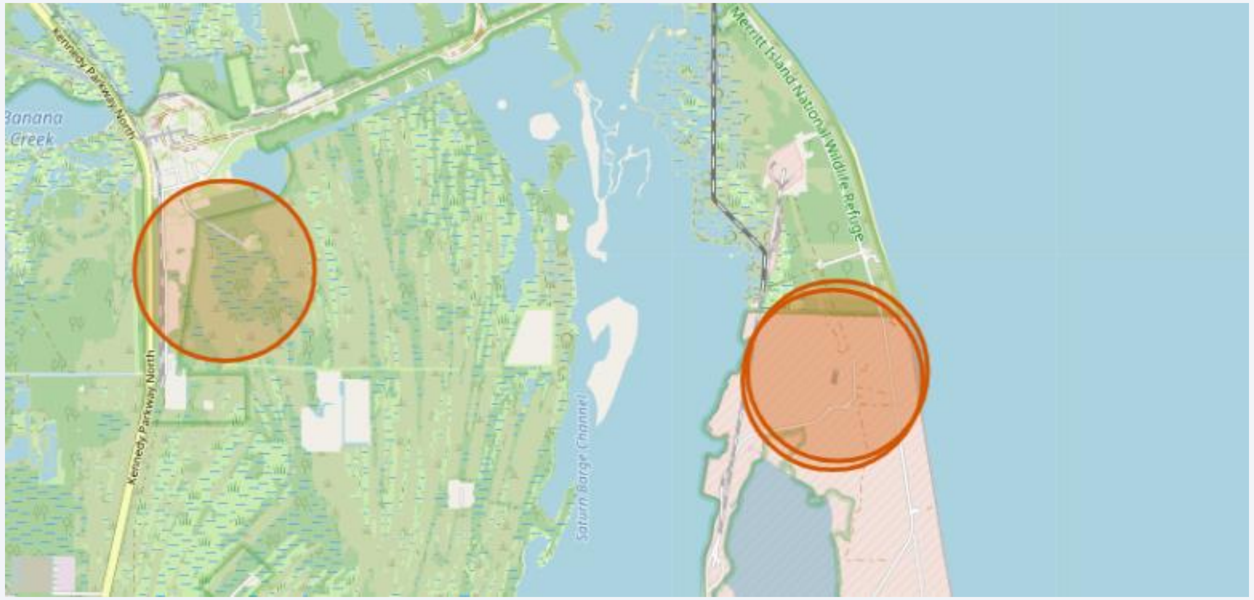
Launch Sites Proximities Analysis

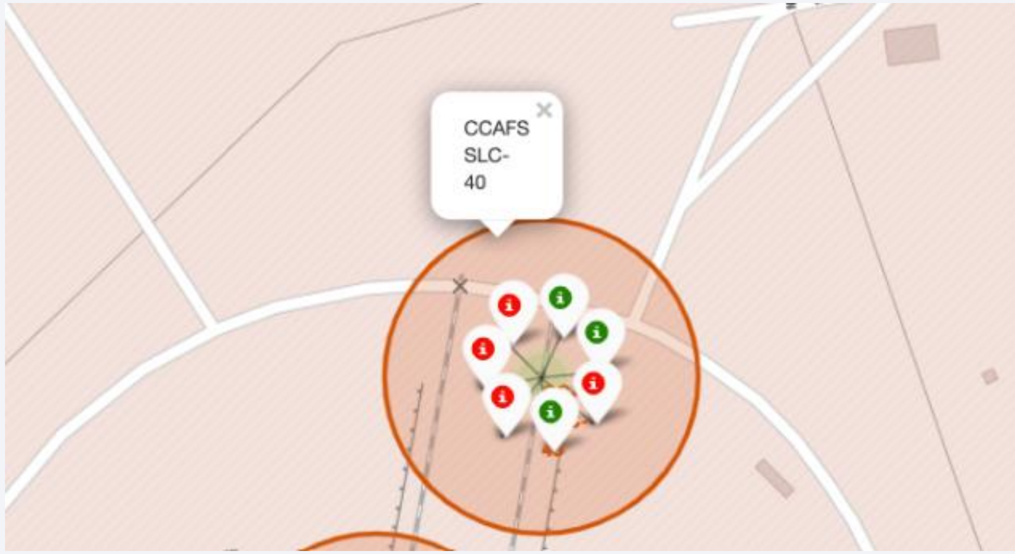
All Launch Site Location Mapping using Marker



We can see the Launch site are in the US

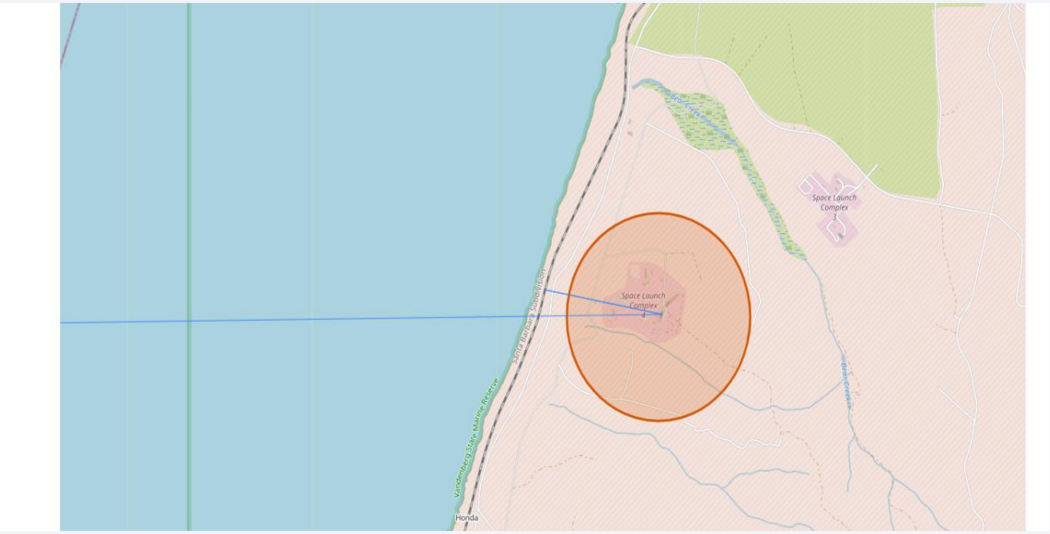
Successful/Failed Launches

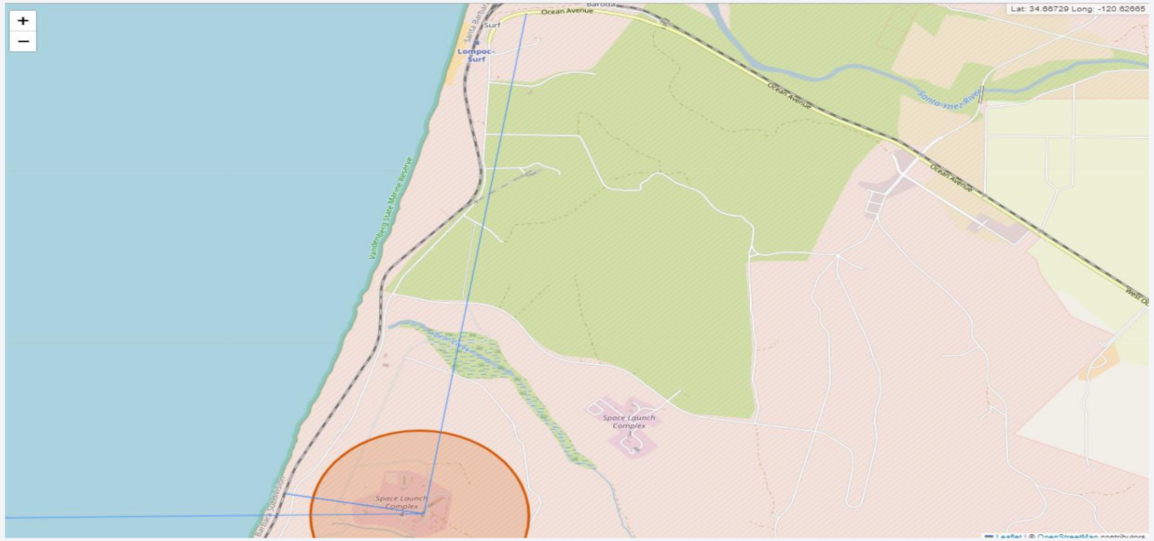




From this we can check the success launch side for green and failure from red

<Folium Map Screenshot 3>

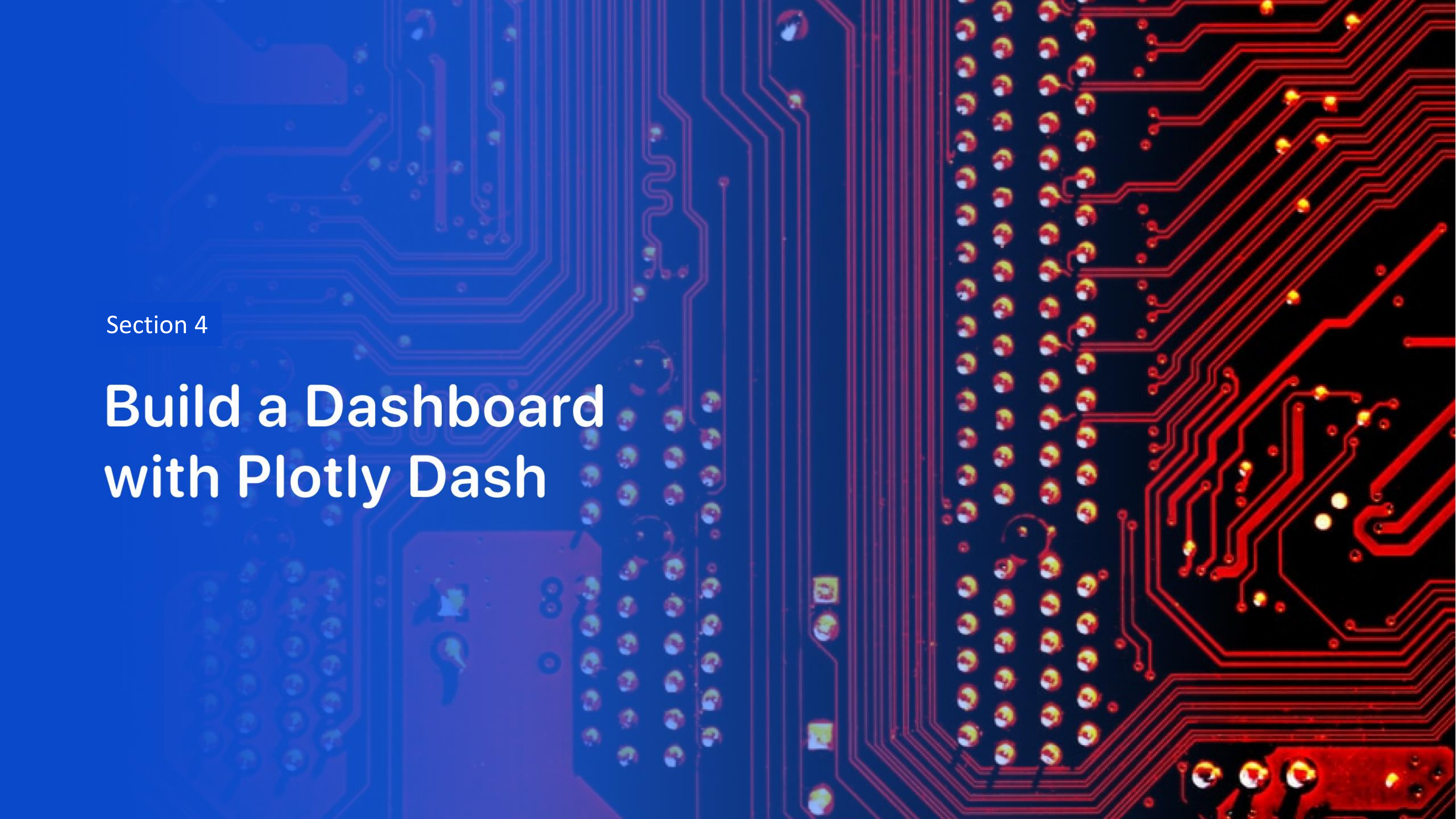



+
−
Lat: 34.66729 Long: -120.62665
Ocean Avenue
Surf
Lompoc-Surf
Ocean Avenue
Ocean Avenue
Vandenberg State Marine Reserve
Space Launch Complex
Space Launch Complex
Leaflet | © OpenStreetMap contributors

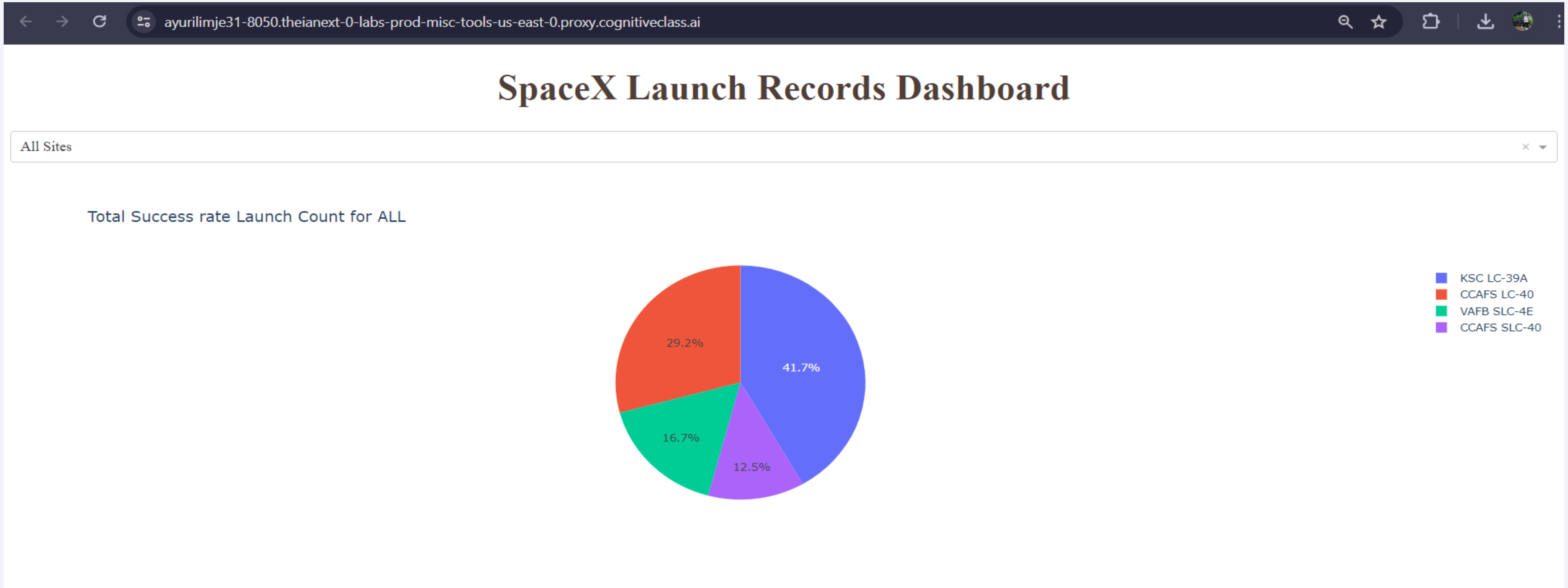
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

Build a Dashboard with Plotly Dash



ALL Launch Site Count using Plotly Dash



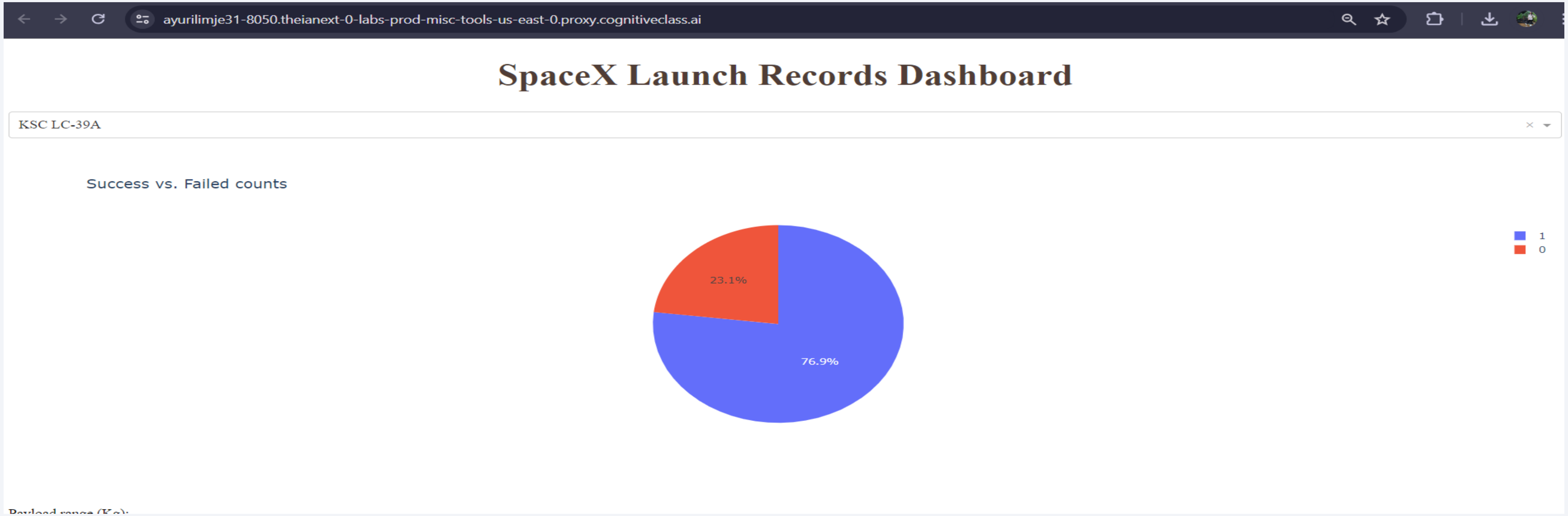
ayurilimje31-8050.theianext-0-labs-prod-misc-tools-us-east-0.proxy.cognitiveclass.ai

SpaceX Launch Records Dashboard

All Sites

From the chart we can visualize the percentage of success rate for each location, where KSC LC-39A has more success rate. Also, here we have use dcc.dropdown for filter while dcc.graph to plot the pie chart by adding the respective pie chart parameter in figure.

Launch Site with Highest Launch Success Ration



ayurilimje31-8050.theianext-0-labs-prod-misc-tools-us-east-0.proxy.cognitiveclass.ai

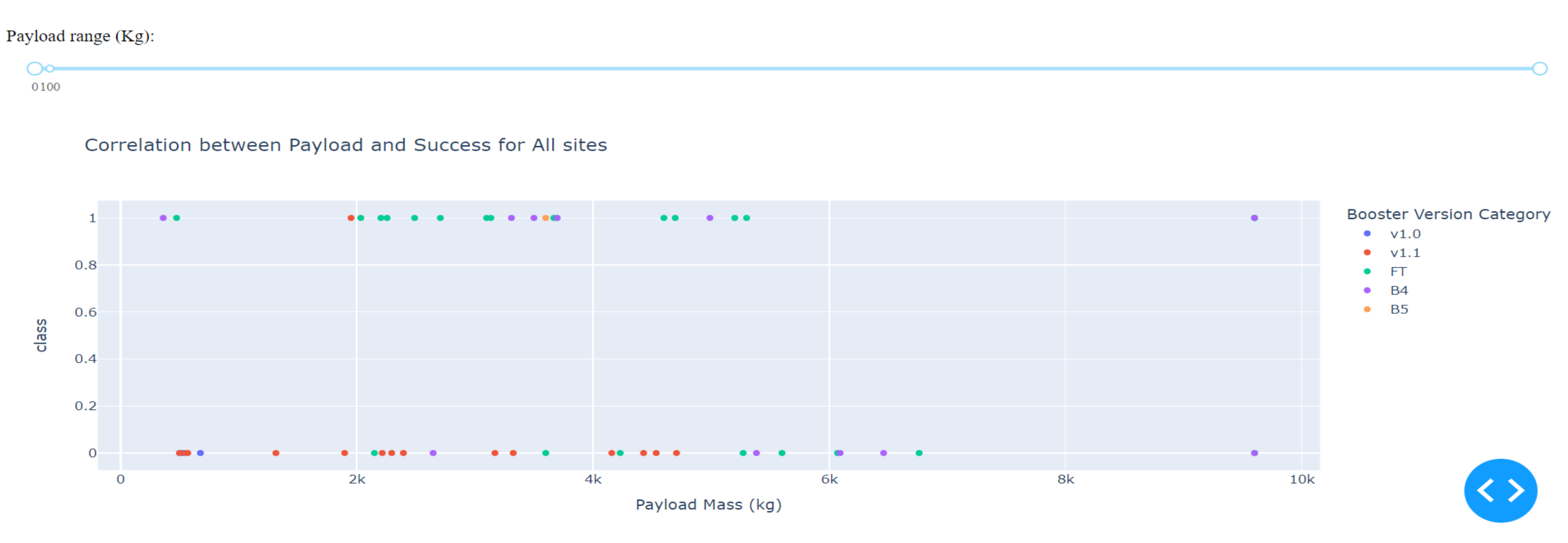
SpaceX Launch Records Dashboard

KSC LC-39A

Payload range (Kg):

From the chart we can visualize the percentage of success rate for KSC LC-39A is 76.9 and failure is 23.1%, which shows that the highest launch success. Also, here we have use dcc.dropdown for filter while dcc.graph to plot the pie chart by adding the respective pie chart parameter in figure.

Payload VS Launch Outcome of ALL Sites

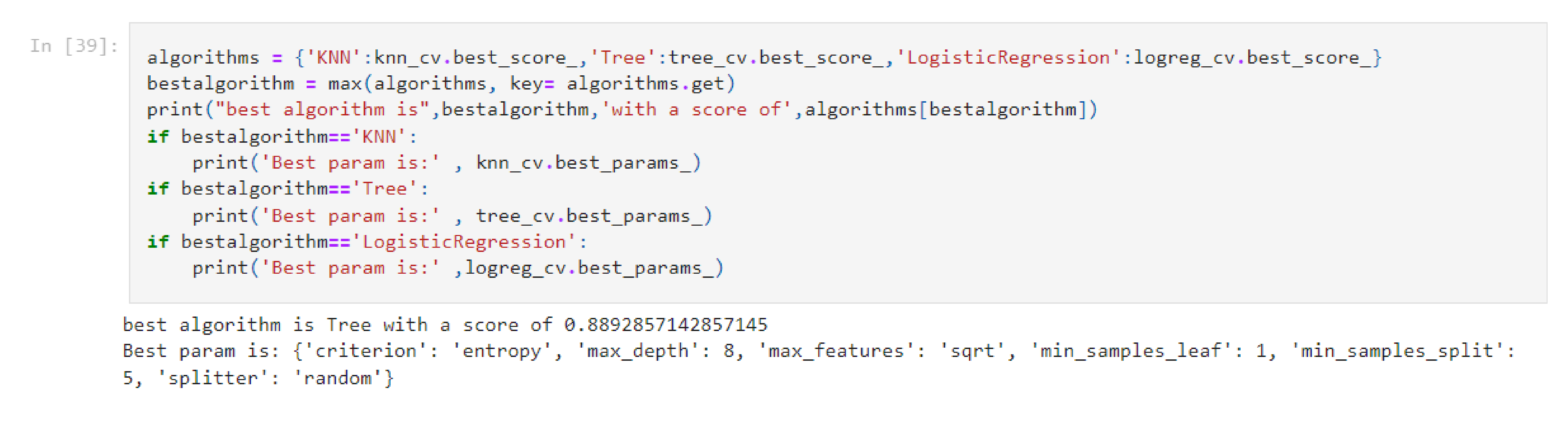


Here we can see the scatter plot showing the relation between the Payload Mass and Booster Version Category with Payload Range in KG. We have used dcc.Rangeslider() for Payload Range and while dcc.graph to plot the scatter plot by adding the respective scatter plot parameter in figure.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

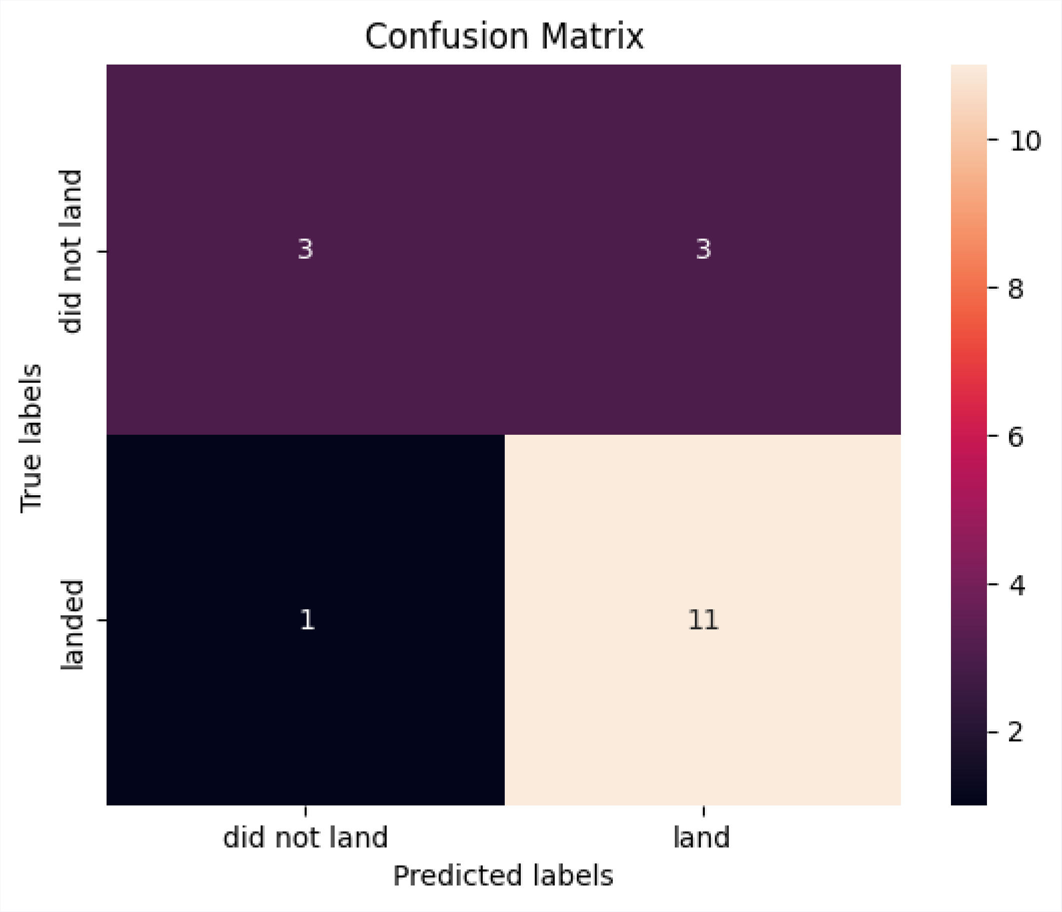


```
In [39]:
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key= algorithms.get)
print("best algorithm is",bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm=='KNN':
    print('Best param is:' , knn_cv.best_params_)
if bestalgorithm=='Tree':
    print('Best param is:' , tree_cv.best_params_)
if bestalgorithm=='LogisticRegression':
    print('Best param is:' ,logreg_cv.best_params_)
```

```
best algorithm is Tree with a score of 0.8892857142857145
Best param is: {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split':
5, 'splitter': 'random'}
```

As we can see the best algorithm is Tree Algorthem

Confusion Matrix

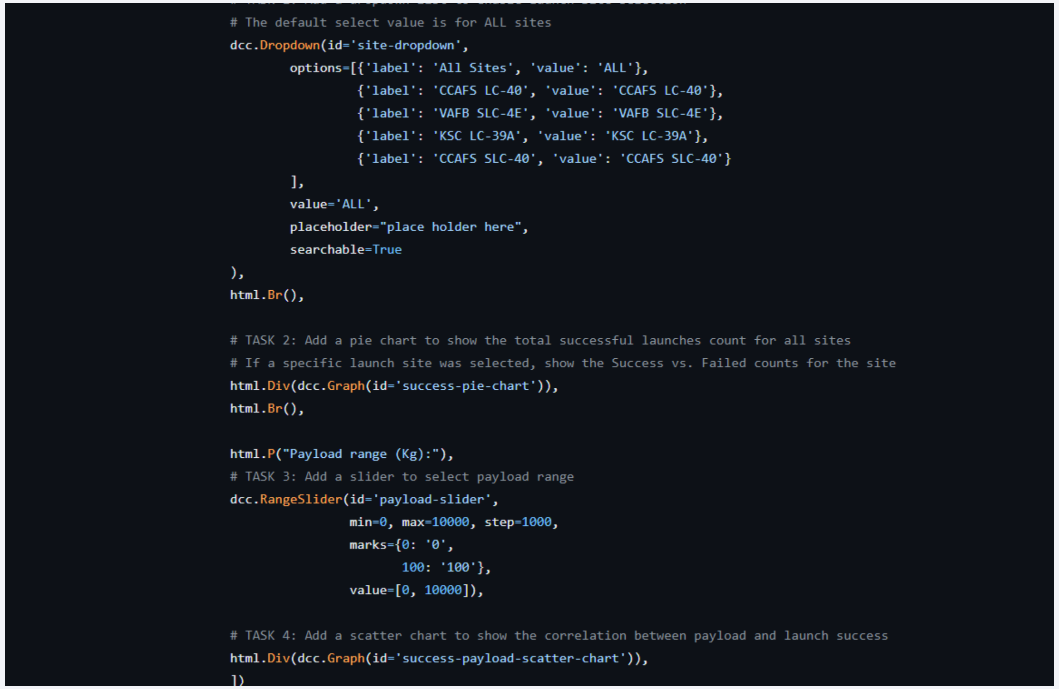


A confusion matrix is a table used to describe the performance of a classification model on a set of test data for which the true values are known. It allows visualization of the performance of an algorithm and helps in understanding how well the model is predicting different classes.

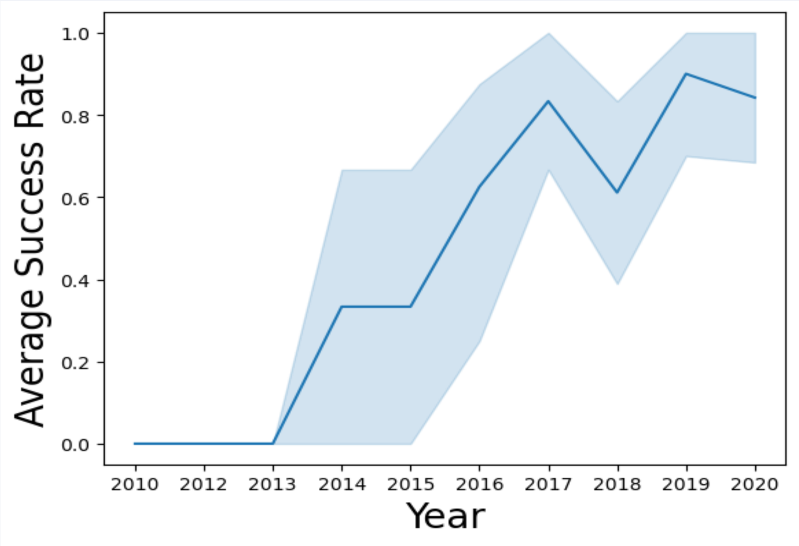
Conclusions

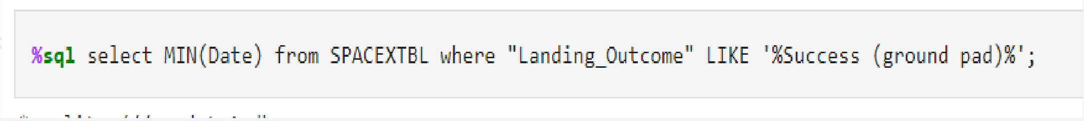
- We can conclude that:
  - The best ML module for this with respect to all the data provided is Tree Algorithm
  - The low wait payloads perform better than the heavy weight payloads
  - From the year 2013 the success rate of Spacex has increased
  - The successful landing or positive landing rate are more for Polar, LEO and ISS
  - The there are no rockets launched for heavypayload mass(greater than 10000) for VAFB-SLC launchsite
  - From the chart we can visualize the percentage of success rate for each location, where KSC LC-39A has more success rate.

Appendix

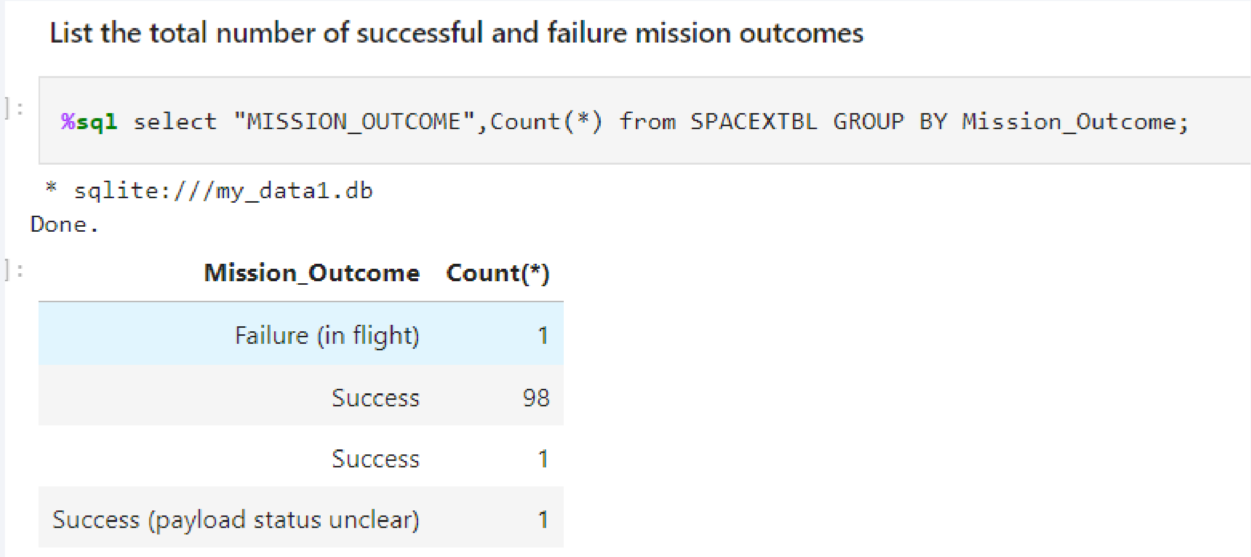


```
# The default select value is for ALL sites
dcc.Dropdown(id='site-dropdown',
        options=[{'label': 'All Sites', 'value': 'ALL'},
                 {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},
                 {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},
                 {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},
                 {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}
        ],
        value='ALL',
        placeholder="place holder here",
        searchable=True
),
html.Br(),

# TASK 2: Add a pie chart to show the total successful launches count for all sites
# If a specific launch site was selected, show the Success vs. Failed counts for the site
html.Div(dcc.Graph(id='success-pie-chart')),
html.Br(),

html.P("Payload range (Kg):"),
# TASK 3: Add a slider to select payload range
dcc.RangeSlider(id='payload-slider',
                min=0, max=10000, step=1000,
                marks={0: '0',
                       100: '100'},
                value=[0, 10000]),

# TASK 4: Add a scatter chart to show the correlation between payload and launch success
html.Div(dcc.Graph(id='success-payload-scatter-chart')),
])
```





```
%sql select MIN(Date) from SPACEXTBL where "Landing_Outcome" LIKE '%Success (ground pad)%';
```



List the total number of successful and failure mission outcomes

```
%sql select "MISSION_OUTCOME",Count(*) from SPACEXTBL GROUP BY Mission_Outcome;
```

```
 * sqlite:///my_data1.db
Done.
```

Mission_Outcome	Count(*)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Thank you!