IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Belkis Ceri
November 29th, 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API: Performing a get request using the requests library to obtain the launch data
  - Data Collection with via Web Scraping: Using Python BeautifulSoup Package for Web Scraping related Wiki pages.
  - Data Wrangling: Cleaning the dataset and create meaningful data (e.g. dealing with with missings, removing nulls, handling outliers and duplicates),
  - Exploratory Data Analysis with SQL: Connecting database and create queries.
  - Exploratory Data Analysis (EDA) with Data Visualization: Creating catplot, barchart, line chart, scatter plot via Seaborn and Matplotlib.
  - Interactive Visual Analytics with Folium: Preparing interactive dashboard and adding filters, zoom in & out, search features on the map by using Folium and Plotly Dash
  - Machine Learning Prediction: Applying Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors.
- Summary of all results
  - Exploratory Data Analysis (EDA)result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context

  SpaceX promotes Falcon 9 rocket launches on its website at a price of 62 million dollars, while other providers charge over 165 million dollars per launch. A significant portion of the cost savings stems from SpaceX's ability to reuse the first stage. Consequently, if we can predict whether the first stage will successfully land, we can estimate the cost of a launch. This knowledge becomes valuable when other companies consider bidding against SpaceX for a rocket launch. The objective of this project is to establish a machine learning pipeline for predicting the successful landing of the first stage.

- Problems you want to find answers

  - Which elements influence the successful landing of the rocket?
  - The correlation between different factors that impact the success rate of a landing.
  - What operational conditions must be established to guarantee the success of the landing program?

Section 1

Methodology

Methodology

Executive Summary

In gathering data, I used the SpaceX API and scraped information from Wikipedia using web techniques. Then, I organized the data by handling categorical features through a process called one-hot encoding. To better understand the data, I conducted exploratory data analysis (EDA) using both visualization and SQL. For a more interactive exploration, I employed Folium and Plotly Dash for visual analytics. Moving on to predictive analysis, I created, fine-tuned, and evaluated various classification models. This multifaceted approach is designed to provide a comprehensive understanding of the data, fostering better decision-making and strategic insights.

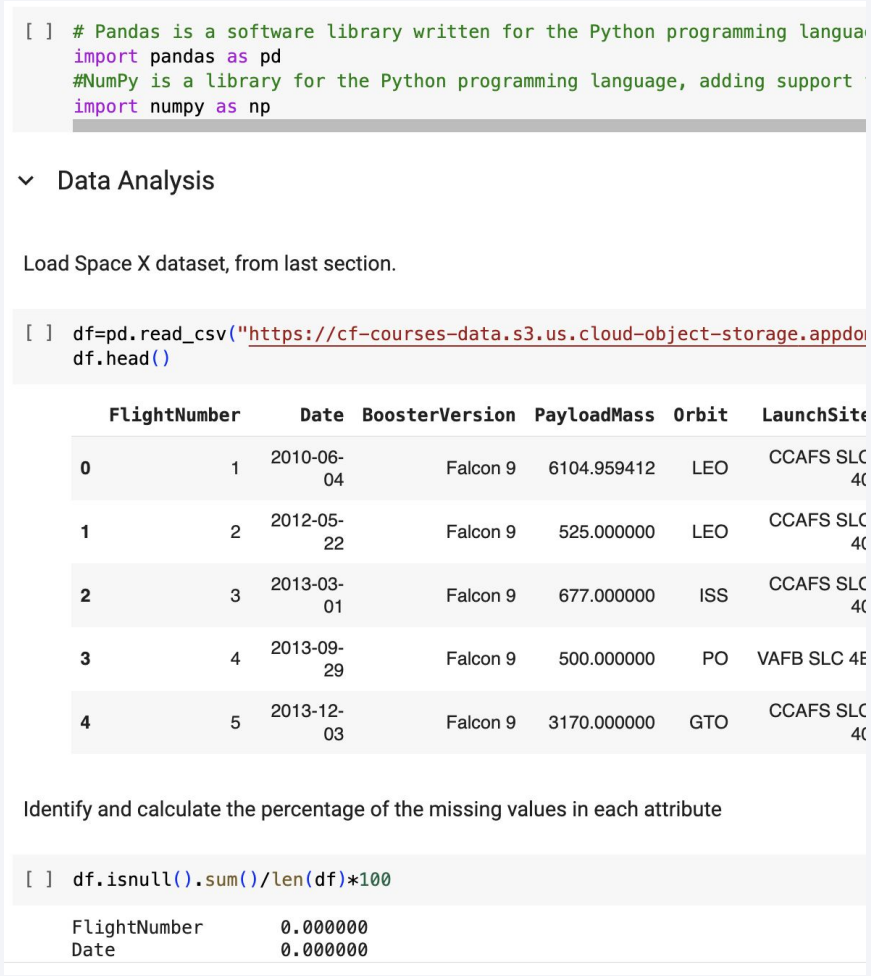
The capstone project GitHub link: https://github.com/BelkisCeri/Capstone-Project

Data Collection

- I collected the data using a combination of methods. Initially, I accessed the SpaceX API through get requests, decoding the response content into a JSON format and then transforming it into a pandas dataframe using .json_normalize(). Following this, I engaged in data cleaning, addressing missing values through appropriate measures. In parallel, I conducted web scraping on Wikipedia to extract Falcon 9 launch records. Using BeautifulSoup, I targeted the launch records presented as an HTML table, parsed the data, and converted it into a pandas dataframe, ensuring a comprehensive dataset for subsequent analysis.
- The data collection jupiter link: https://github.com/BelkisCeri/Capstone-Project/blob/main/LAB_1_jupyter-labs-spacex-data-collection-api.ipynb

Data Collection – SpaceX API

- Utilizing the get request to the SpaceX API, we gathered data, performed cleaning on the acquired information, and carried out essential data wrangling and formatting.
- The link to the notebook is
- https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab-3_jupyter-spacex-Data%20wrangling.ipynb



```
[ ] # Pandas is a software library written for the Python programming langua
    import pandas as pd
    #NumPy is a library for the Python programming language, adding support
    import numpy as np
```

Data Analysis

Load Space X dataset, from last section.

```
[ ] df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdo
    df.head()
```

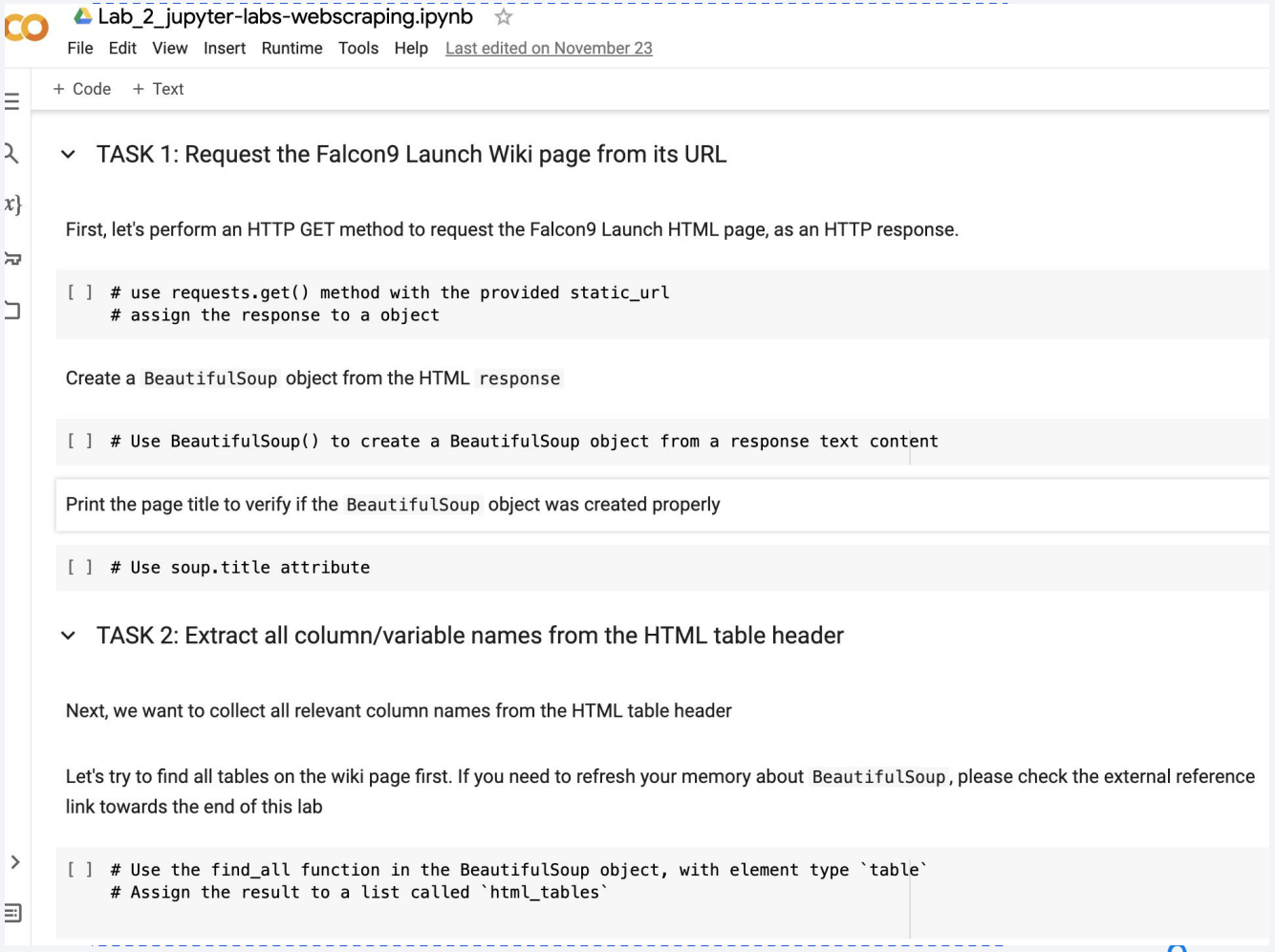
	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 4(
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 4(
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 4(
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 4(

Identify and calculate the percentage of the missing values in each attribute

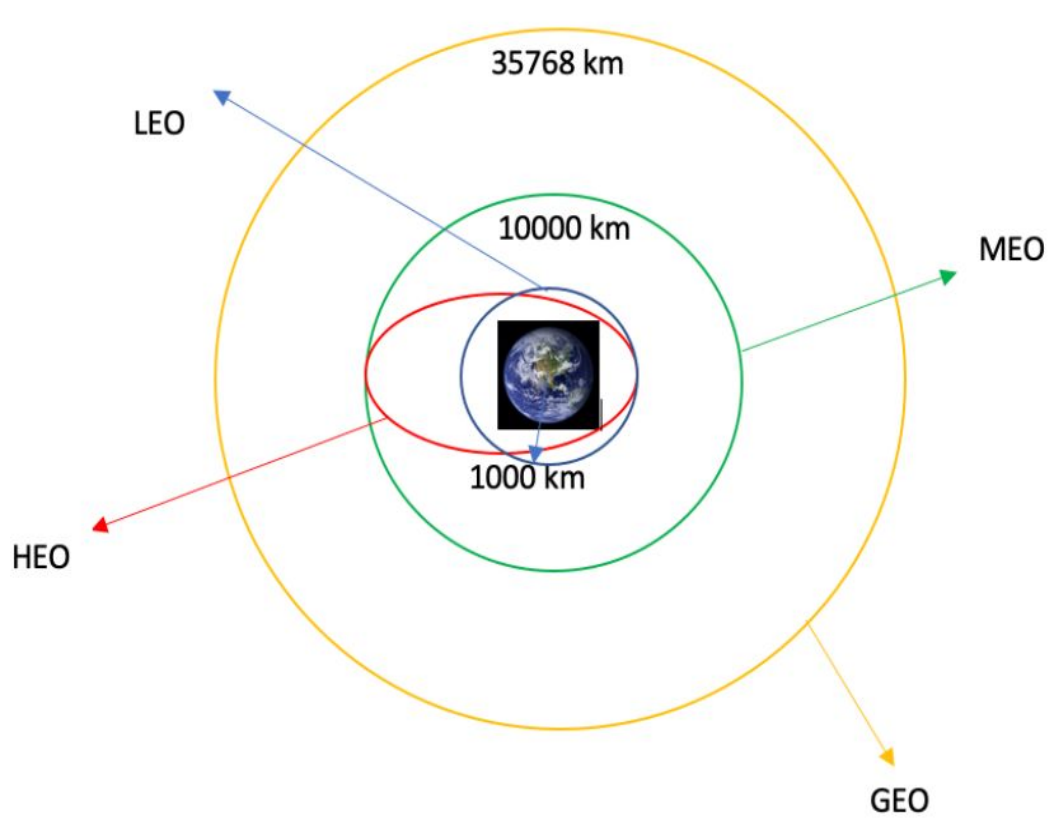
```
[ ] df.isnull().sum()/len(df)*100

    FlightNumber      0.000000
    Date              0.000000
```

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is https://github.com/BelkisCeri/Capstone-Project/tree/main



Data Wrangling



- I performed exploratory data analysis and determined the training labels. Then, I calculated the number of launches at each site, and the number and occurrence of each orbits. Finally, the landing outcome label from outcome column and exported the results to csv is created as seen in the link to the notebook below
- https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab-3_jupyter-spacex-Data%20wrangling.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, and the launch success yearly trend.

The link to the notebook is https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab_5_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab_6_lab_jupyter_launch_site_location.jupyterlite.ipynb

EDA with SQL

The SpaceX dataset was loaded into a PostgreSQL database without leaving the jupyter notebook. Then, I applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:

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 acheived.
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.

- The link to the notebook is https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab_4_jupyter-labs-(Explanatory%20Data%20Analysis)%20EDA-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- I marked all the launch sites on the map and incorporated map elements like markers, circles, and lines to visually represent the success or failure of launches at each site using Folium. The launch outcomes, categorized as failure (class 0) or success (class 1), were assigned accordingly.
- By utilizing color-coded marker clusters, I determined which launch sites exhibited a relatively high success rate. Additionally, I computed the distances between launch sites and their surroundings, addressing questions such as whether launch sites are in proximity to railways, highways, coastlines, and if they maintain a specific distance from cities.
- The link to the notebook is

https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab_6_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- I developed an engaging dashboard using Plotly Dash. The dashboard features pie charts that visualize the total number of launches conducted at specific sites. Additionally, I incorporated scatter graphs to illustrate the correlation between the outcome of launches and the payload mass (in kilograms) across various booster versions.
- The link to the notebook is:
- https://github.com/BelkisCeri/Capstone-Project/blob/main/Dashboard_python.ipynb

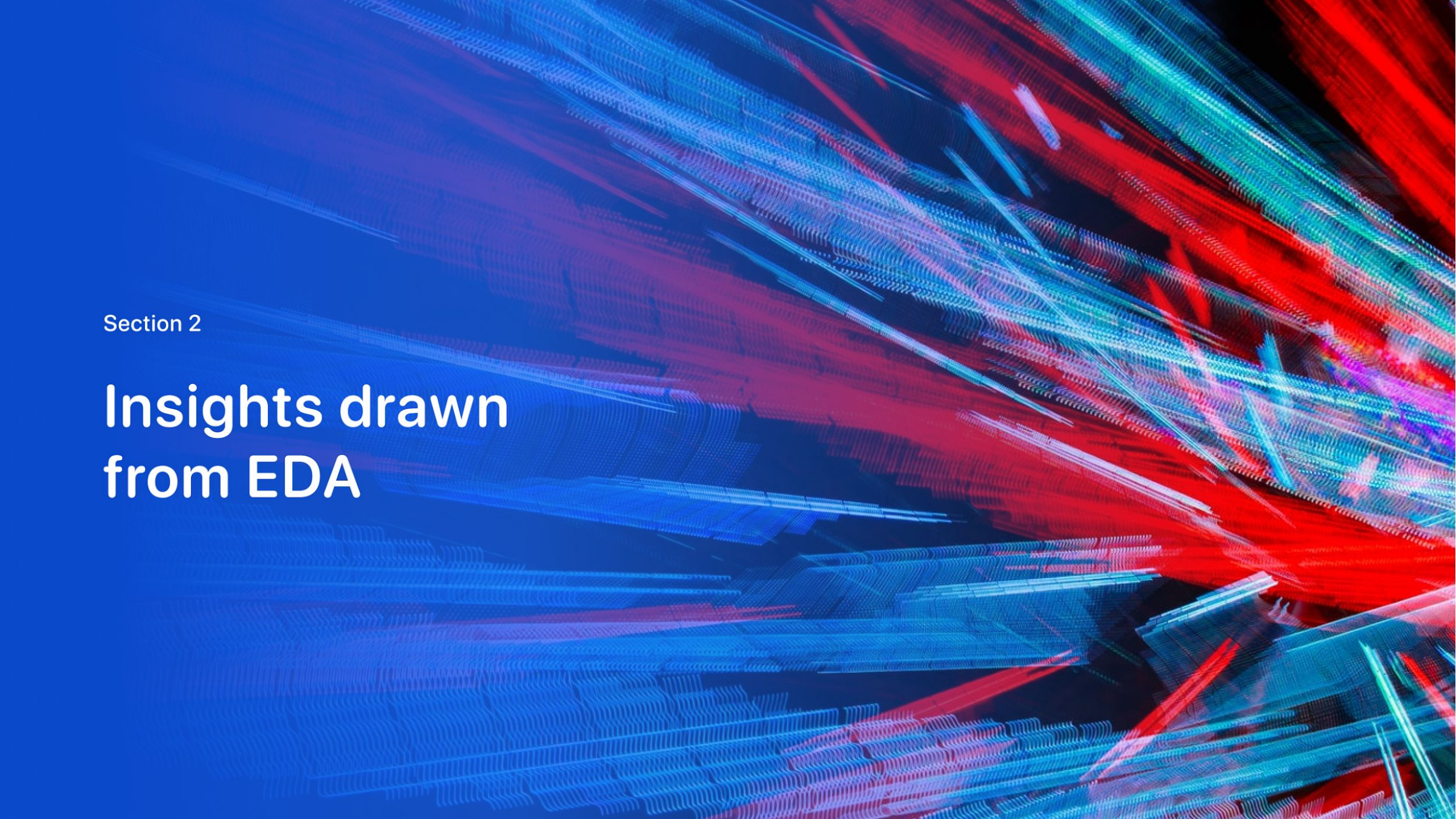
```
import pandas as pd
import dash
import dash_html_components as html
import dash_core_components as dcc
from dash.dependencies import Input, Output
import plotly.express as px


spacex_df = pd.read_csv("spacex_launch_dash.csv")
max_payload = spacex_df['Payload Mass (kg)'].max()
min_payload = spacex_df['Payload Mass (kg)'].min()


app = dash.Dash(__name__)
server = app.server

uniquelaunchsites = spacex_df['Launch Site'].unique().tolist()
lsites = []
lsites.append({'label': 'All Sites', 'value': 'All Sites'})
for site in uniquelaunchsites:
 lsites.append({'label': site, 'value': site})



app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color': '#503D36',
                                               'font-size': 40}),




                                dcc.Dropdown(id='site_dropdown',options=lsites,placeholder='Select a Launch Si
                                html.Br(),
```

Predictive Analysis (Classification)

- I utilized NumPy and Pandas to load and manipulate the data. Following that, I divided the data into training and testing sets. Next, I constructed various machine learning models and fine-tuned their hyperparameters using GridSearchCV. To assess the model's performance, accuracy was employed as the primary metric. Further enhancements were made through feature engineering and algorithm tuning. Ultimately, we identified the most effective classification model.
- The link to the notebook is https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab_8_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

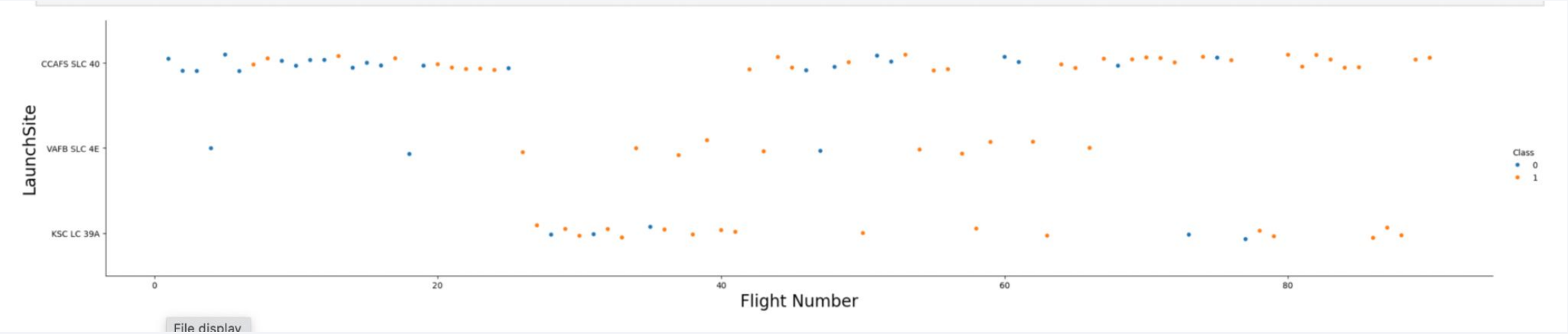
Results

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


Section 2
Insights drawn from EDA

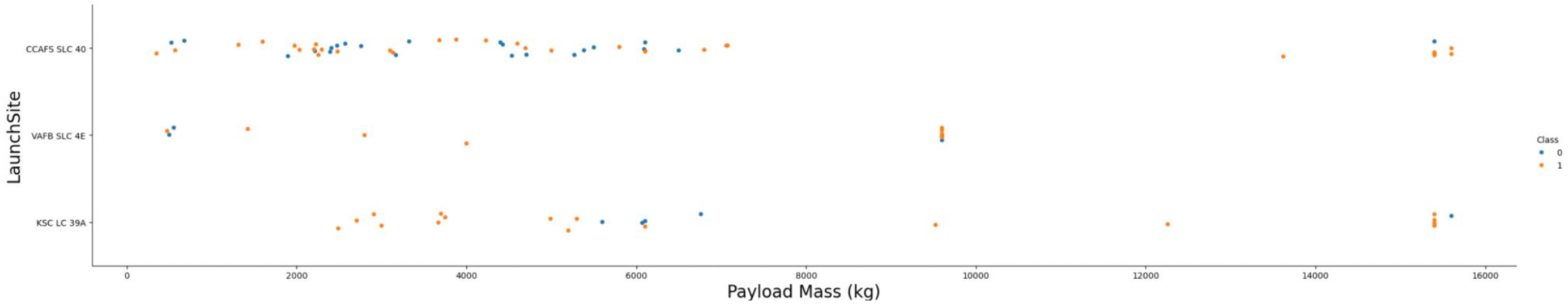
Flight Number vs. Launch Site

- Based on the plot, it was observed that as the number of flights at a launch site increases, the success rate at that launch site also tends to rise.



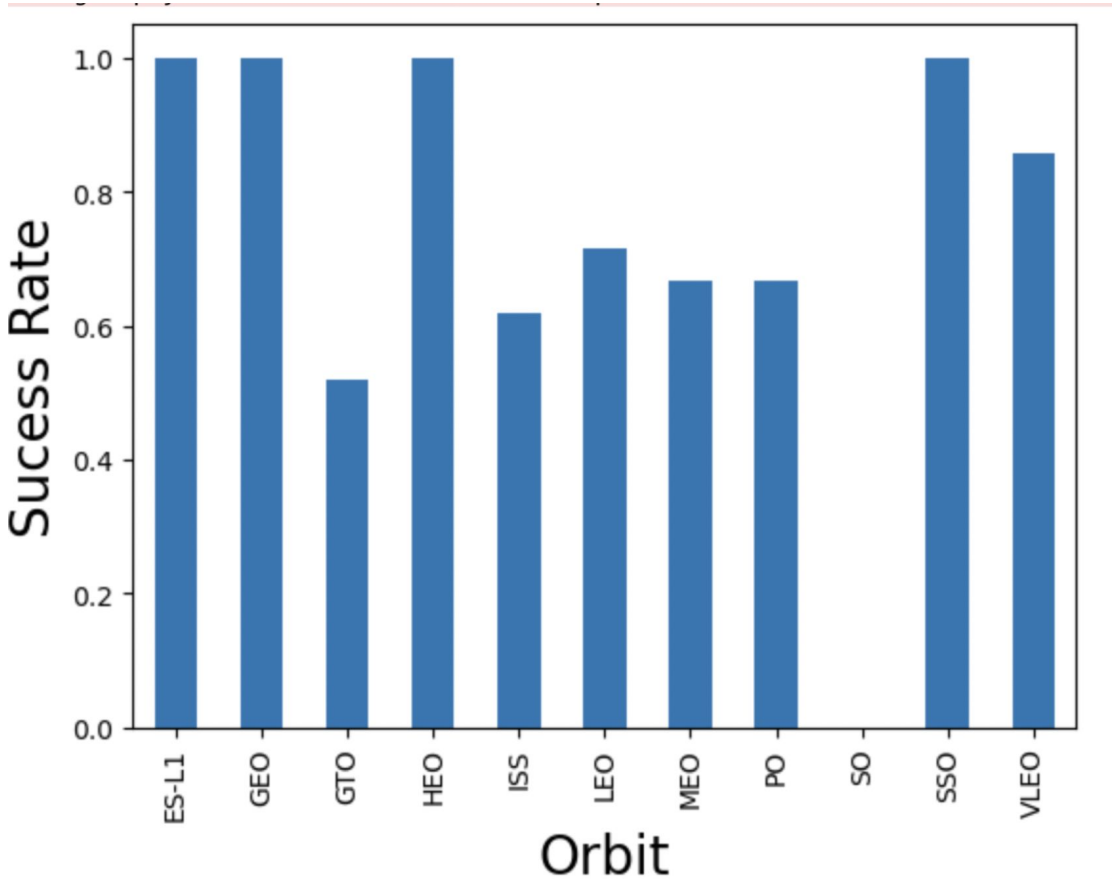
Payload vs. Launch Site

As the payload mass increases for the launch site CCAF5 SLC-40, a corresponding rise in the success rate for rocket launches is evident.



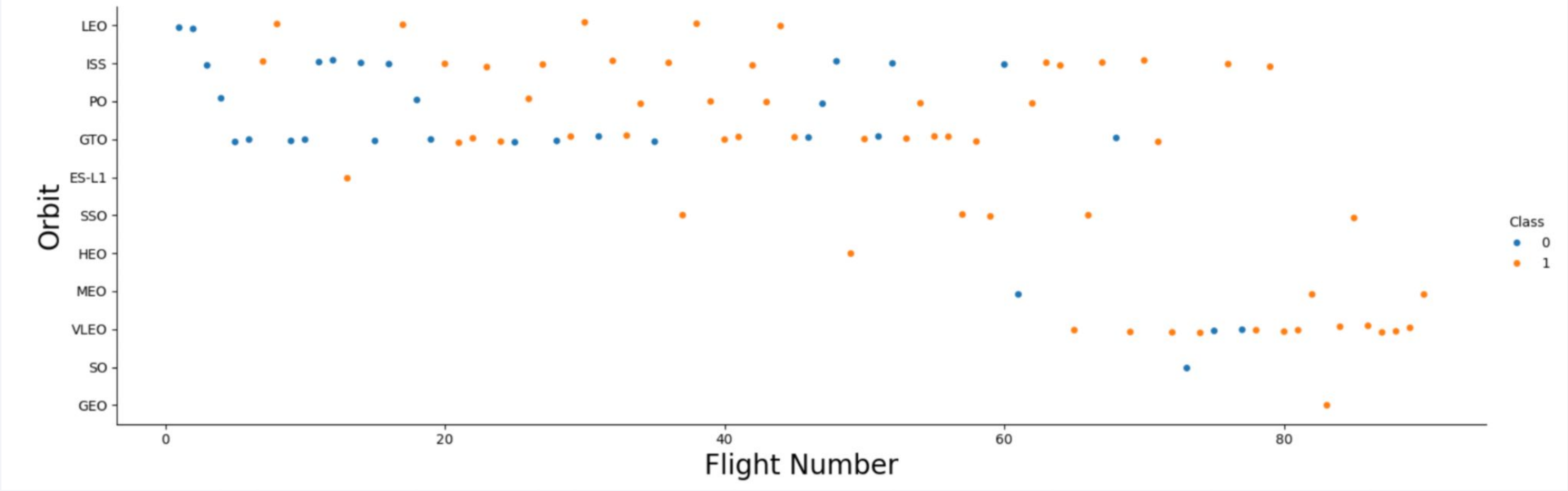
Success Rate vs. Orbit Type

- From the plot, it can be sees that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



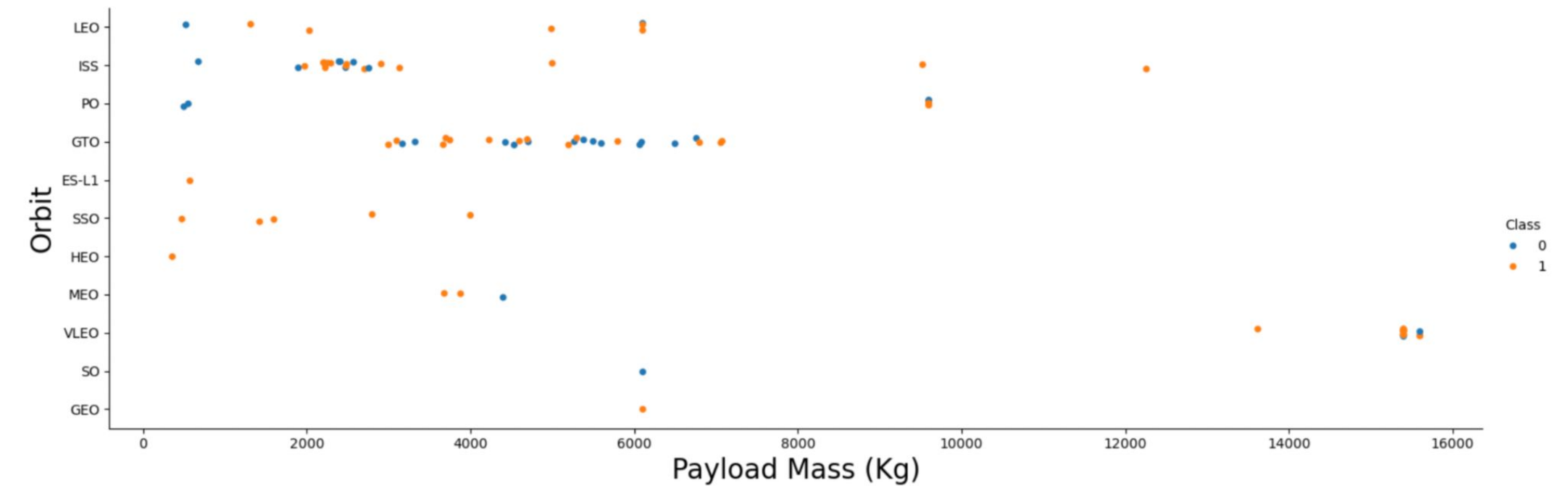
Flight Number vs. Orbit Type

- The chart presented illustrates the correlation between Flight Number and Orbit type. It is apparent that success in the Low Earth Orbit (LEO) is linked to the number of flights, whereas in the Geostationary Transfer Orbit (GTO), there is no discernible connection between flight number and orbit.



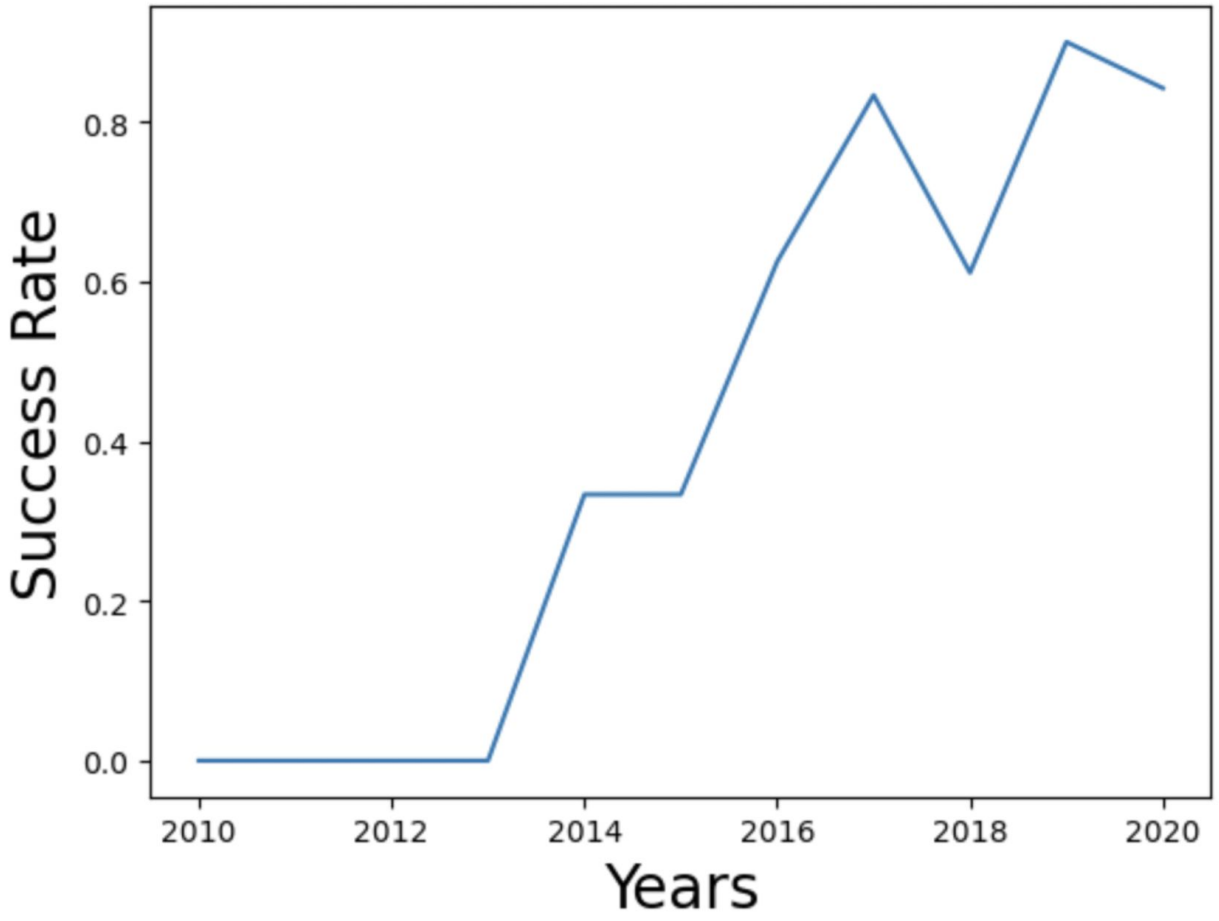
Payload vs. Orbit Type

- It can be noted that for Payload Orbits (PO), Low Earth Orbits (LEO), and International Space Station (ISS) orbits, successful landings are more frequent when dealing with heavier payloads.



Launch Success Yearly Trend

- Based on the graph, it is apparent that the success rate has been consistently rising from 2013 to 2020.



All Launch Site Names

- I used the **DISTINCT** to show only unique launch sites from the SpaceX data.

https://github.com/BelkisCeri/Capstone-Project/blob/main/Lab_4_jupyter-labs-(Explanatory%20Data%20Analysis)%20EDA-sql-coursera_sqllite.ipynb

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

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

Out[10]:

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

Launch Site Names Begin with 'CCA'

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- It was used the query above to display 5 records where launch sites begin with `CCA`

Total Payload Mass

- There are a calculated 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

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

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

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

In [14]:

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

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

Out[15]:

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

- The WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

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

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

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

	successoutcome
0	100

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

Out[16]:

	failureoutcome
0	1

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

Boosters Carried Maximum Payload

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

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

In [17]:
```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

- The combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

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

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

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

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

In [19]:
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

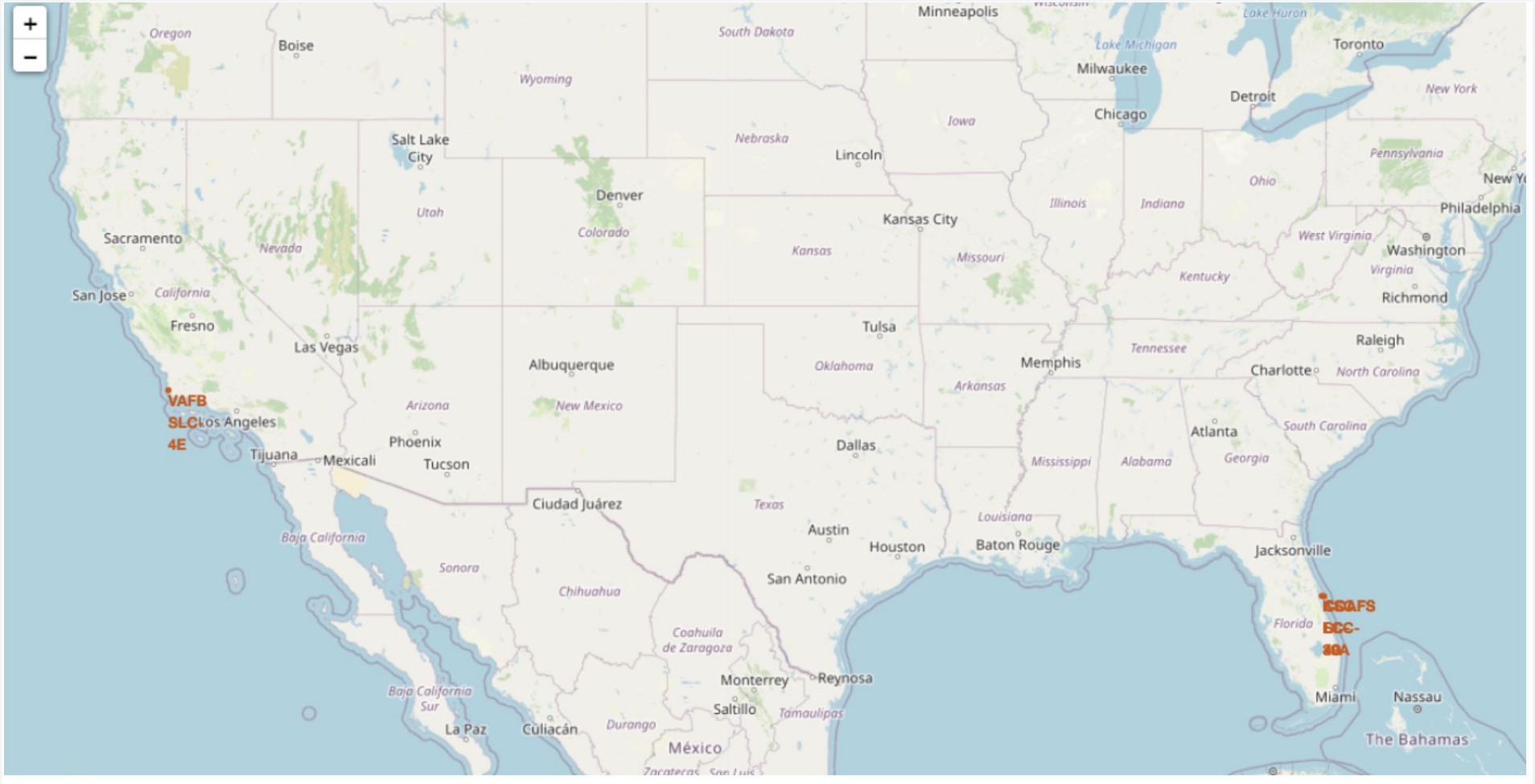
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

- It was chosen the Landing outcomes and the COUNT of landing outcomes from the dataset. Using the WHERE clause, we filtered for landing outcomes within the time range of June 4, 2010, to March 20, 2010. Subsequently, we utilized the GROUP BY clause to group the landing outcomes and applied the ORDER BY clause to arrange the grouped landing outcomes in descending order.

Section 4

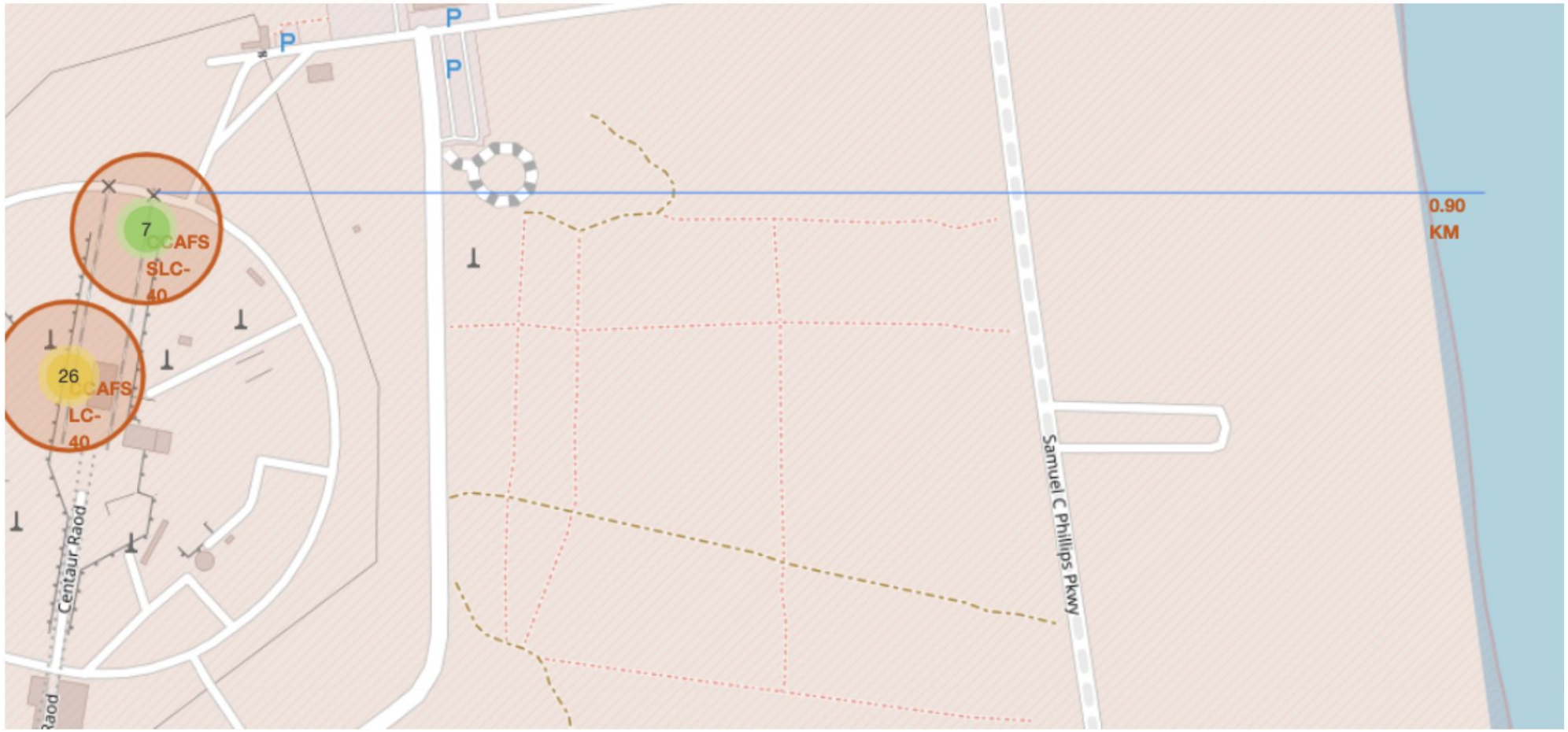
Launch Sites Proximities Analysis

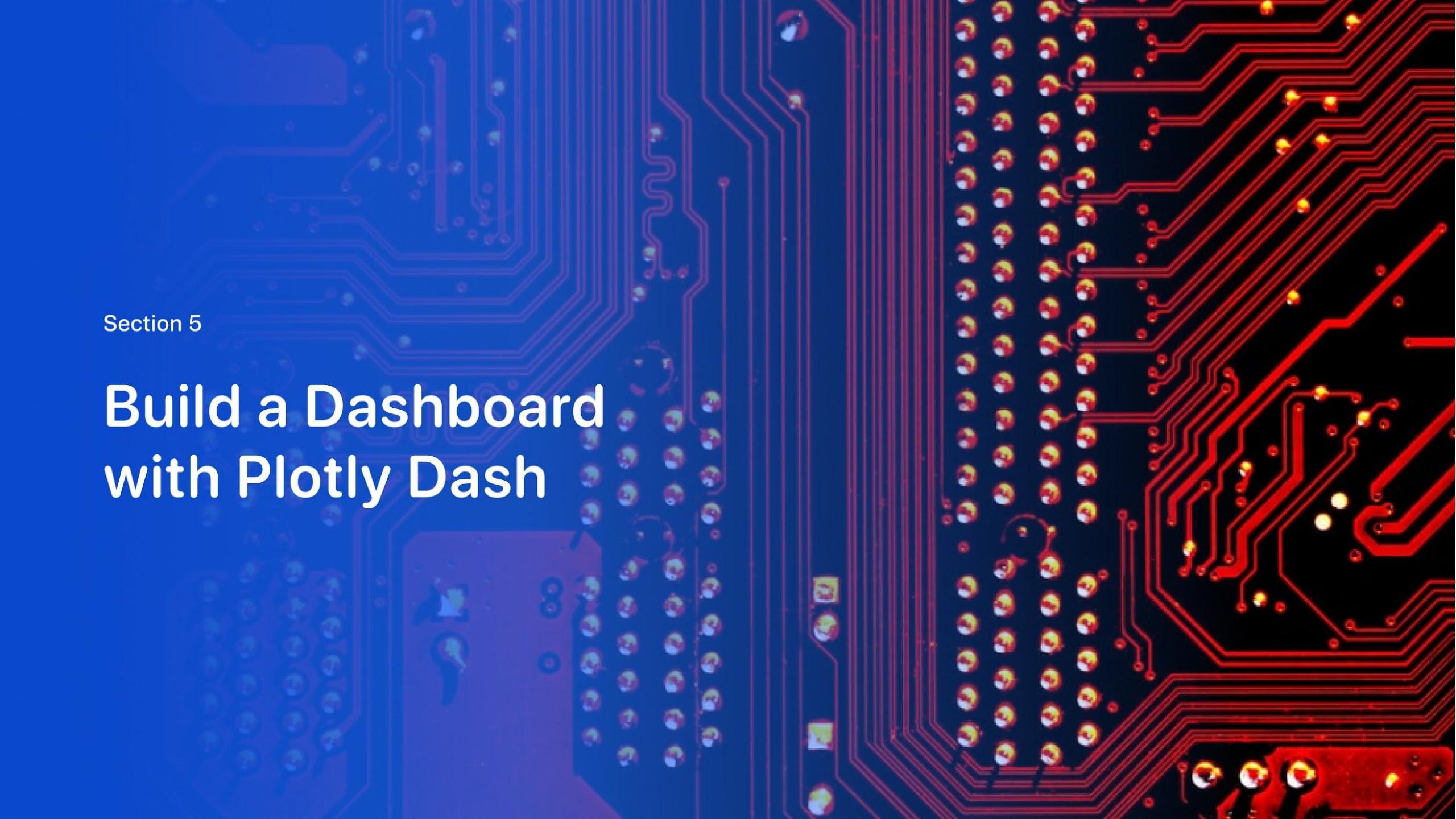
All launch sites global map markers



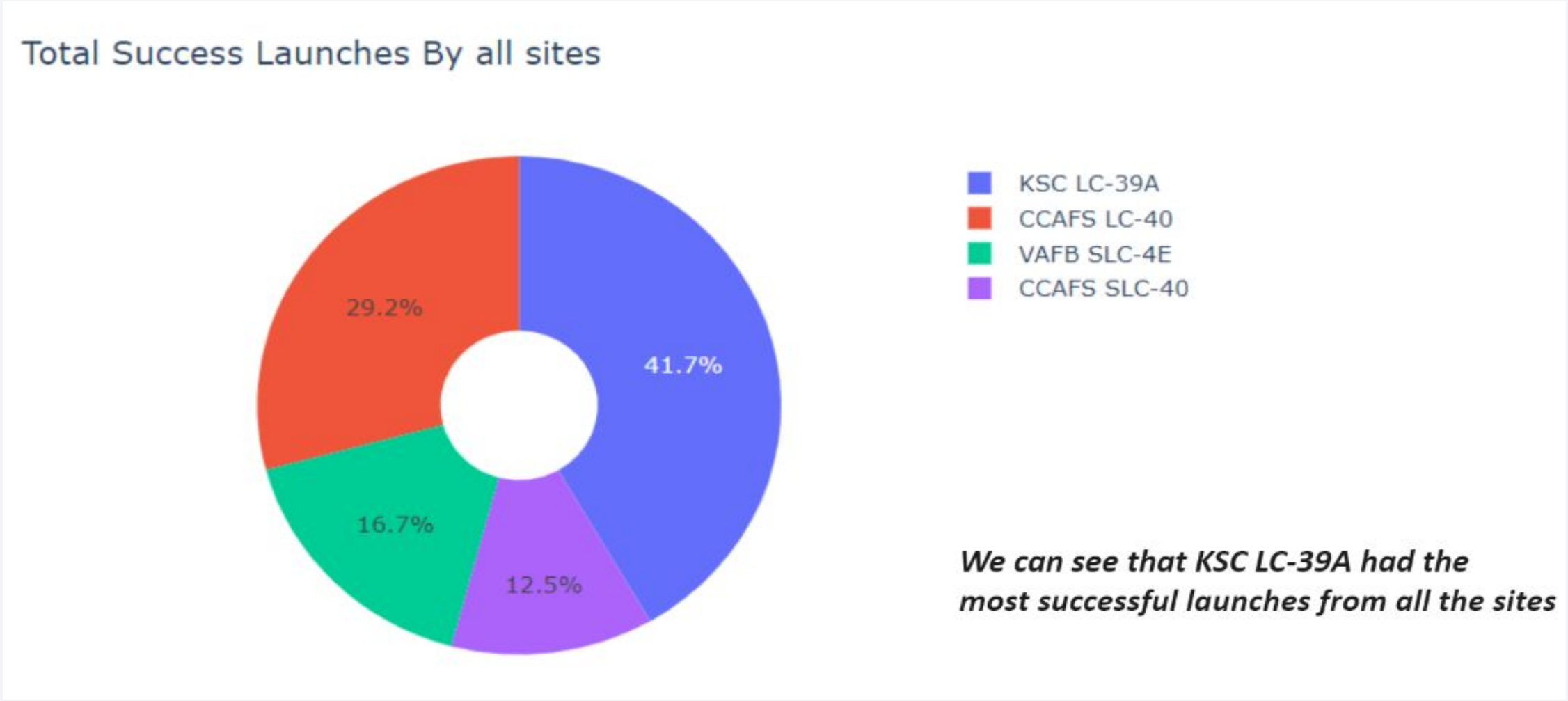
Markers showing launch sites with color labels

Launch Site distance to landmarks



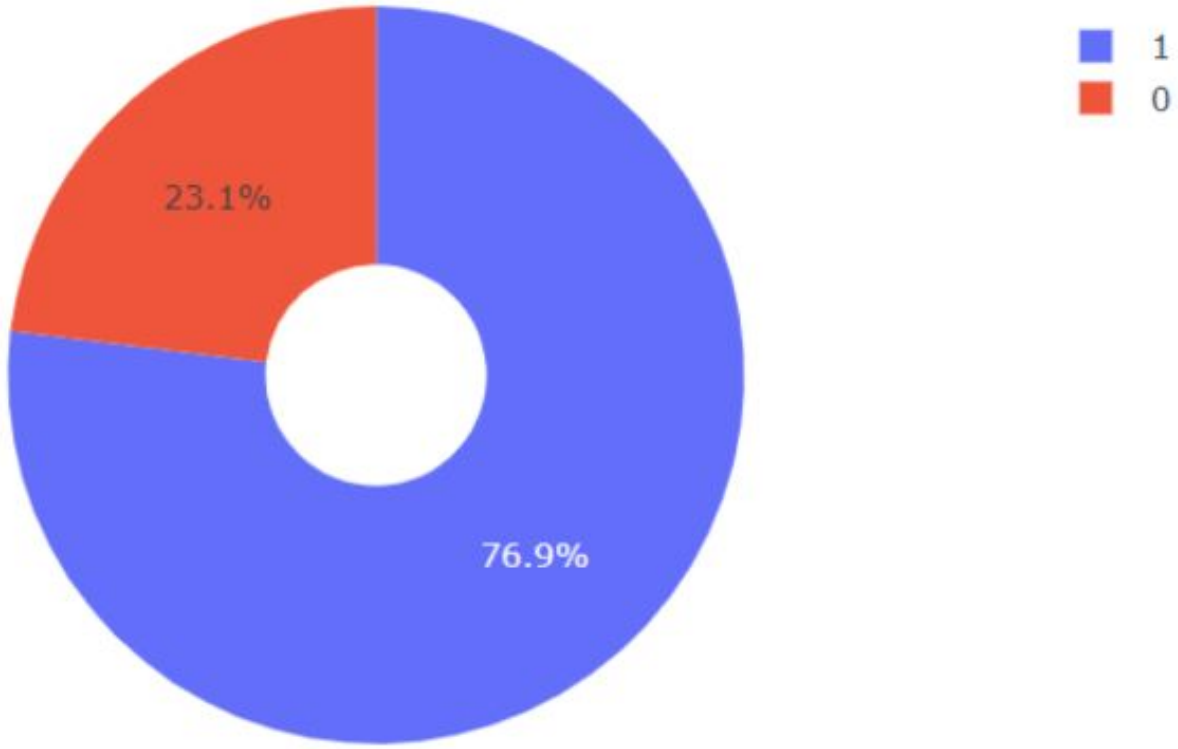

Section 5
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



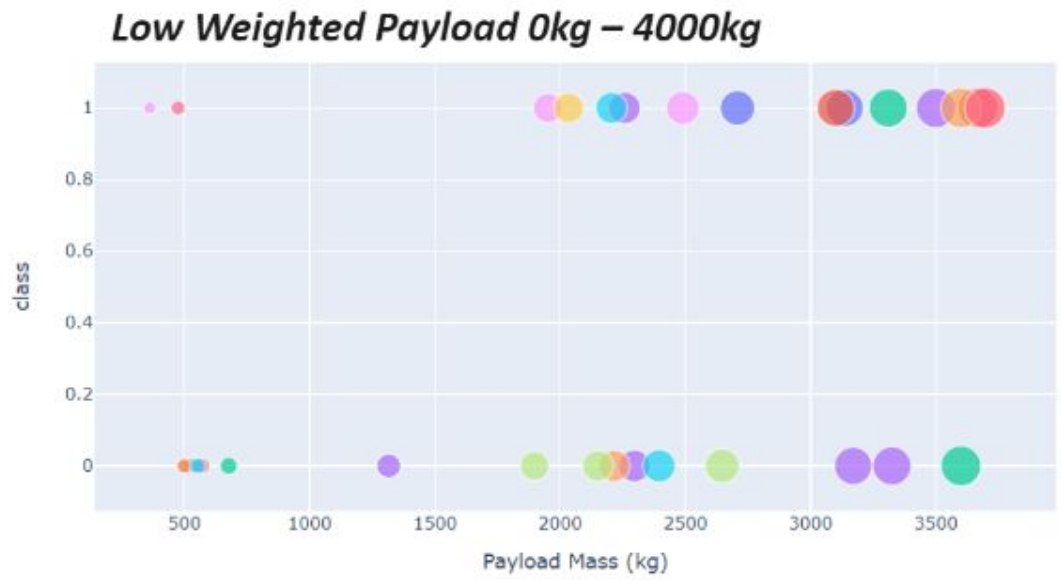
We can see that KSC LC-39A had the most successful launches from all the sites

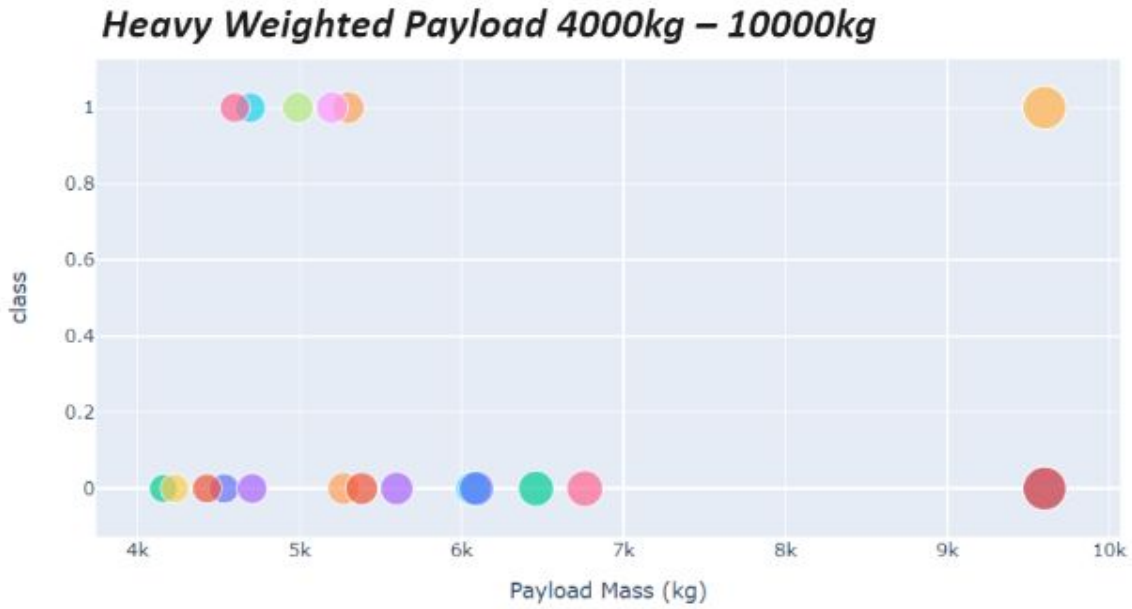
Pie chart showing the Launch site with the highest launch success ratio



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

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





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

Section 6

Predictive Analysis (Classification)

Classification Accuracy

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

Create a decision tree classifier object then create a `GridSearchCV` object `tree_cv` with cv = 10. Fit the object to f parameters from the dictionary `parameters`.

[80]:
```
parameters = {
     'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]
           }

tree = DecisionTreeClassifier()
```

[81]:
```
tree_cv = GridSearchCV(tree, parameters, cv=10)
```

[83]:
```
# tree_cv.fit(X_train,Y_train)
```

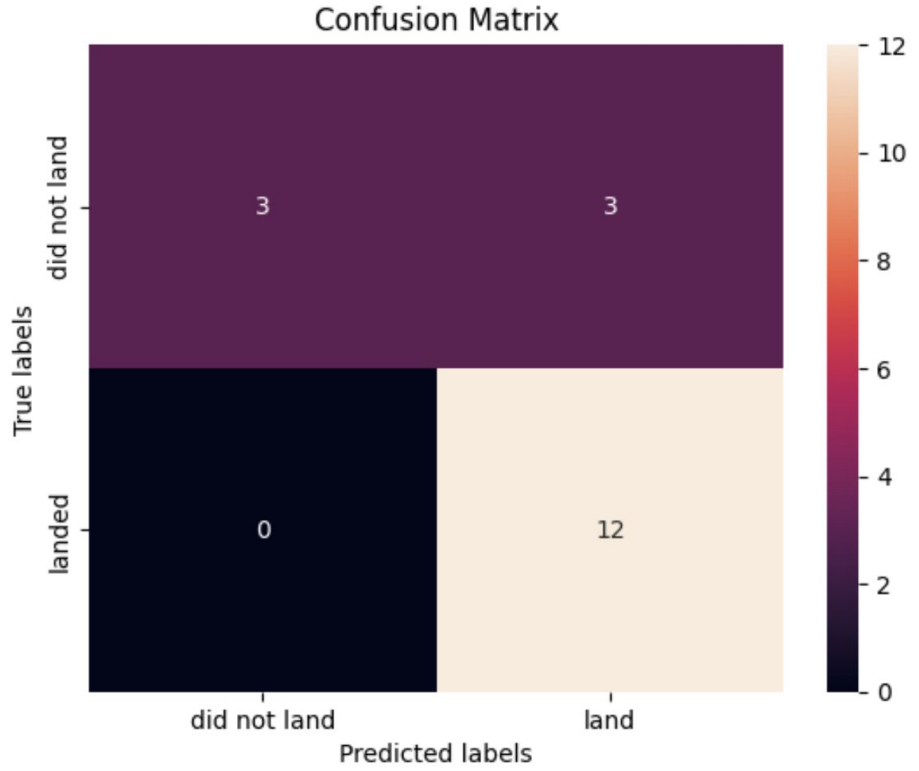
[84]:
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

Confusion Matrix

- The decision tree classifier's confusion matrix reveals its capability to differentiate between distinct classes. However, a significant challenge arises from false positives, specifically instances where the classifier incorrectly identifies unsuccessful landings as successful.

We can plot the confusion matrix

[86]:
```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

In summary:

- A positive correlation exists between the number of flights at a launch site and the success rate, indicating that higher flight amounts correspond to greater success rates at launch sites.
- The launch success rate exhibited a consistent upward trend from 2013 to 2020.
- Orbits ES-L1, GEO, HEO, SSO, and VLEO demonstrated the highest success rates.
- KSC LC-39A emerged as the launch site with the highest number of successful launches among all sites.
- Among the machine learning algorithms assessed, the Decision Tree classifier proved to be the most effective for this particular task.

Thank you!