Winning Space Race with Data Science

Abhishek Shrestha
15th December 2022

Outline

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

Executive Summary

- Summary of methodologies

1. Data Collection with API
2. Data Collection with Web Scrapping
3. Data Wrangling
4. Exploratory Data Analysis through EDA with SQL
5. Exploratory Data Analysis through EDA with Visualization
6. Interactive Visual Analytics with Folium
7. Machine Learning Prediction

- Summary of all results

1. Exploratory Data Analysis results
2. Interactive Analytics results
3. Predictive Analytics results

Introduction

- Project background and context

Space X has advertised that Falcone 9 rocket launches on its website with a cost of 62 million dollars; others have provided the cost to be upward of 165 million dollars each, where much of the savings is because Space X will be reusing the first stage. Hence, if we can determine whether the first stage will land or not, we will be more likely to determine the cost of the launch. This information will be used if an alternate company wants to bid against Space X for rocket launches. Moreover, the goal of the project is to create a machine learning pipeline to predict if the first stage of the launch will land successfully.

- Problems you want to find answers

1. What are the factors that will determine the landing of the rocket launches successfully?
2. Determining the relationship amongst the various features on the success rate of each successful landing.
3. What operating conditions does Space X have to achieve to have the best results and ensure successful landings?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected through Space X API and Web Scrapping from Wikipedia
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Performed exploratory data analysis (EDA) using visualization and SQL using scatter and bar graphs to show patterns between the data
- Performed interactive visual analytics using Folium and Plotly Dash
- Performed predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- The data was collected using:

1. Get request to the Space X API
2. Decoding the response as JSON using .json( ) function to call and turn it into a pandas dataframe using .json_normalize( )
3. Cleaned the data then checked for missing values and then filling in the missing values where needed
4. Web Scrapped from Wikipedia for the Falcon 9 launch records through BeautifulSoup
5. Extracted the launch records through html, parser and converted it to pandas dataframe

Data Collection – SpaceX API

- Firstly, we used the get request to get response and collect the data from Space X API.
- Then, we cleaned the requested data to convert json results to a dataframe.
- Lastly, we used custom functions to clean and fill in the missing values in the dataframe for formatting.
- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/Data%20Collection%20API.ipynb

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

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

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
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 – Web Scraping

- Firstly, we webscrapped the Falcon 9 launches page to obtain data with BeautifulSoup.
- Then, we parsed the html response to get column names and converted the dictionary frame to a pandas dataframe.
- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/Data%20Collection%20with%20Web%20Scrapping.ipynb

1. Apply HTTP Get method to request the Falcon 9 rocket launch page

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

Out[5]:
```
200
```

2. Create a BeautifulSoup object from the HTML response

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

Print the page title to verify if the `BeautifulSoup` object was created properly

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

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

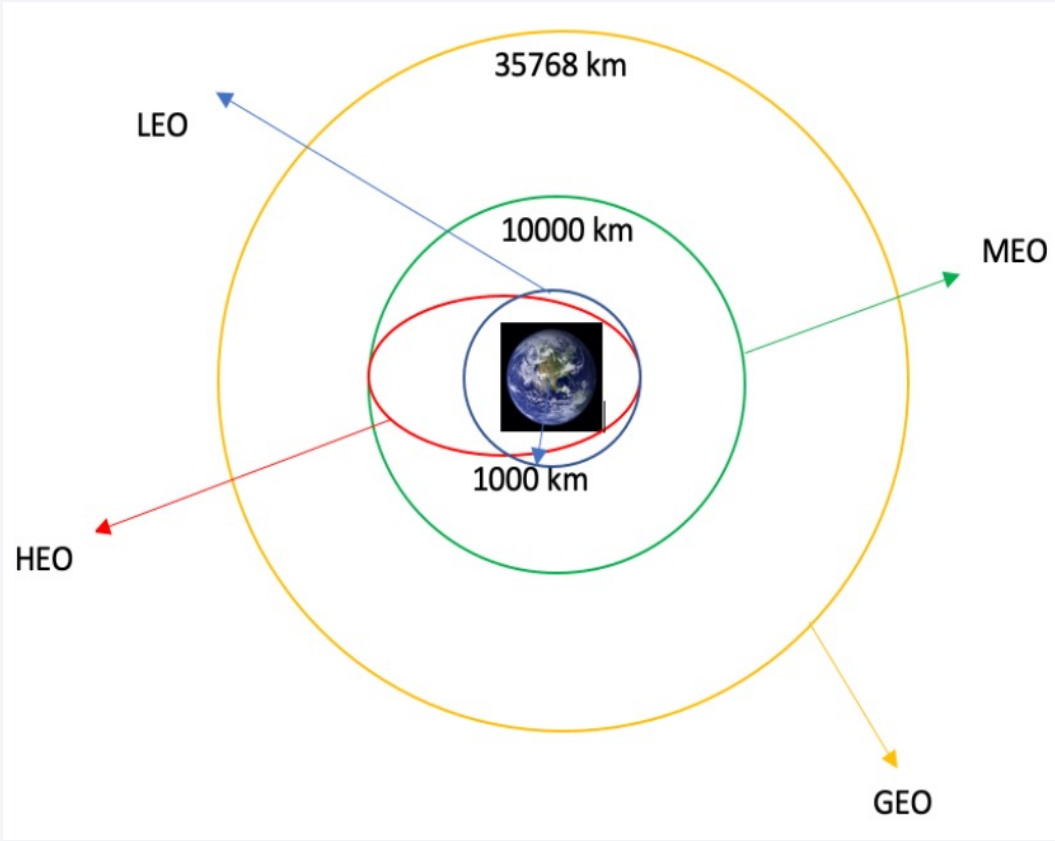
3. Extract all column names from the HTML table header

In [10]:
```
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 by parsing the launch HTML tables
5. Export data to csv

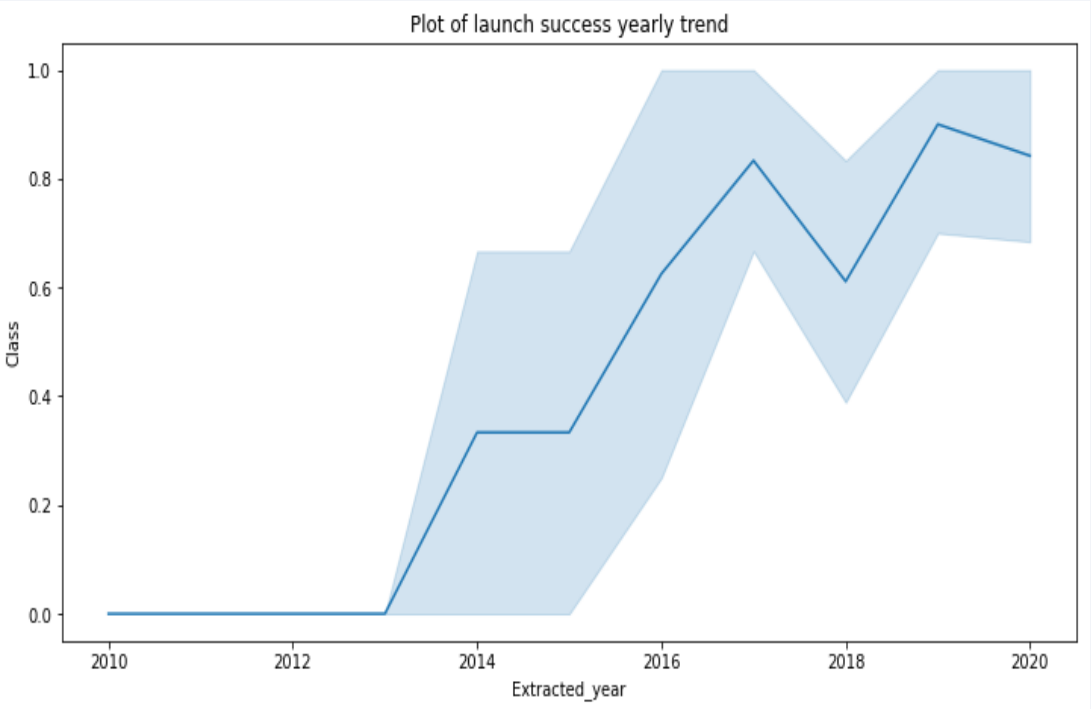
Data Wrangling

- Firstly, we performed an exploratory data analysis to determine the training labels.
- Then, calculated the number of launches on each site in accordance to the number and occurrence of each orbits and,
- Lastly, we worked out success rate for every landing in dataset and creating a landing outcome label from the outcome column to export the results into csv file.
- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/EDA.ipynb



EDA with Data Visualization

- Firstly, we explored the data by using visualization to find the relationship between flight numbers and the launch site.
- Then, we explored the relationship between payload and launch site in regards to the success rate of each orbit type, flight number and orbit type and the launch success trend on a yearly basis.
- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/EDA%20with%20Visualization.ipynb



- Using line graph here are useful as they show the data variables and trends which helps making prediction about results that are yet to be recorded.

EDA with SQL

- To begin with, we used PostgreSQL in the Jupyter notebook to load and assess the Space X dataset.
- To get an insight, we wrote queries to find out:

1. The names of the unique launch sites in the space mission
2. 5 records where launch sites begin with string ‘CCA’
3. The total payload mass carried by boosters launched by NASA (CRS)
4. Average payload mass carried by booster version F9 v1.1
5. The date when the first successful landing outcome in ground pad was achieved
6. The names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. The total number of successful and failure mission outcomes
8. The names of the booster versions which have carried the maximum payload mass
9. The records which will display the month names, failure landing outcomes in drone ship, booster versions, launch sites for the month in the year 2015
10. The count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order

- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Firstly, we started by marking all the launch sites and then adding markers, circles and lines to mark the success or failure of launches in each site.
- Then, we gave attributes of class 0 and 1 for the feature of launch outcomes where 0 being for failure and 1 being for success.
- We, then identified which launch sites had relatively high success rate with the help of color labeled marker clusters and drew lines on the map to measure the distance to landmarks.
- Lastly, we calculated the distance between launch site to its proximities which gave us an outcome to the questions – whether the launch sites were near coastlines or if the launch sites were kept at distance from the cities or not.
- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/Interactive_Dashboard.ipynb

Build a Dashboard with Plotly Dash

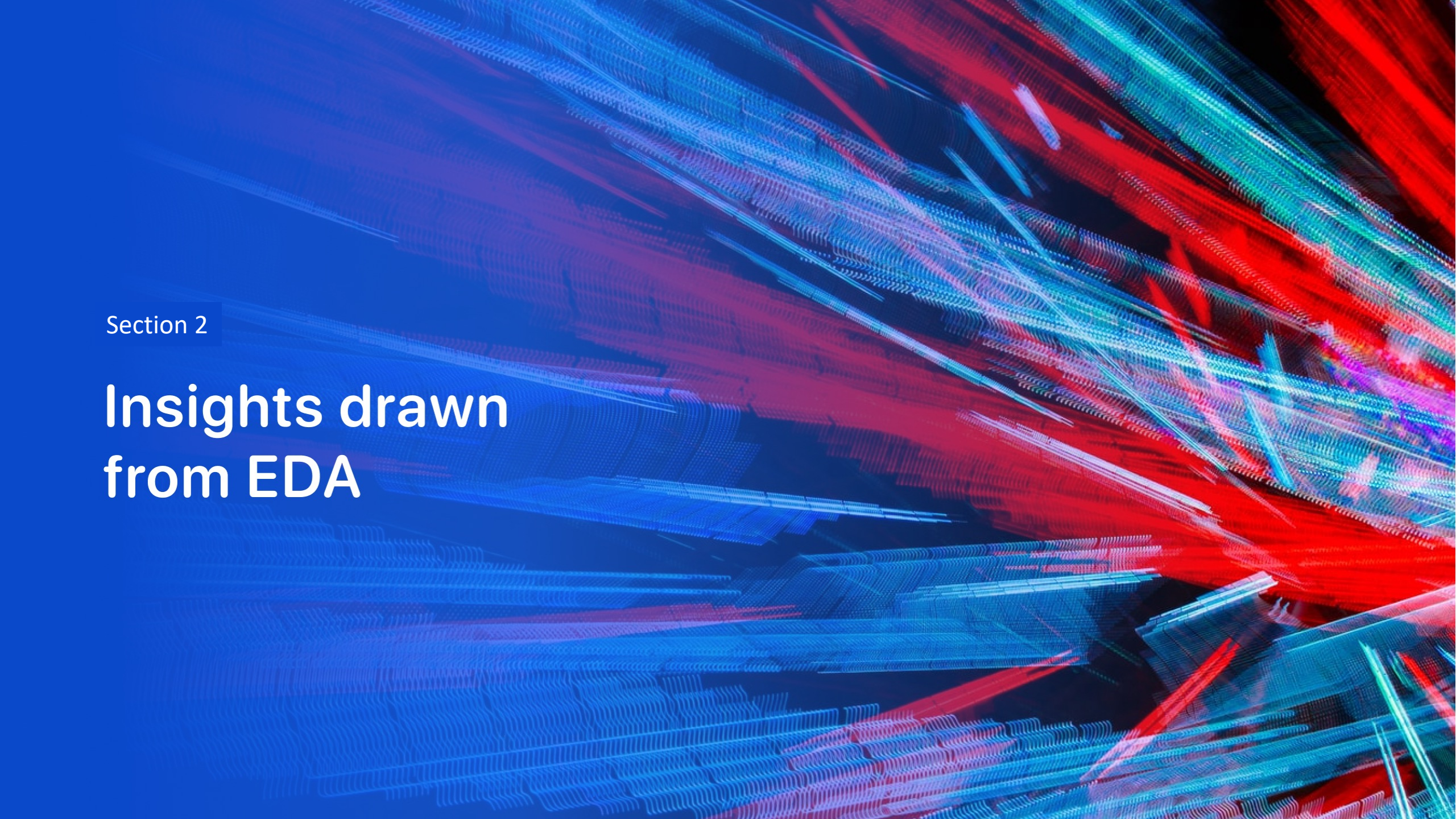
- Firstly, we started by building an interactive dashboard with Plotly Dash that uses flask and dash web framework.
- Then, we plotted the pie charts showing the total launches in accordance to cities which displays the relative proportions of multiple classes of data.
- Lastly, we plotted the scatter graph that shows the relationship with outcome and payload mass for different booster version.
- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/plotlydash.py

Predictive Analysis (Classification)

- Firstly, we loaded the data using numpy and pandas then transformed the data for splitting into training and testing sets to decide which type of machine learning algorithms we wanted to use.
- Then, we made machine learning models and tuned different hyperparameters with the help of GridSearchCV and fit the objects to train the dataset.
- Lastly, we used the accuracy as the metric of our model to evaluate the model. We, then improved the usage of model with feature engineering and algorithm tuning and found the best performing classification model.
- https://github.com/itsabhishake/Data-Science-Space-X-Capstone/blob/main/Machine%20Learning%20Prediction.ipynb

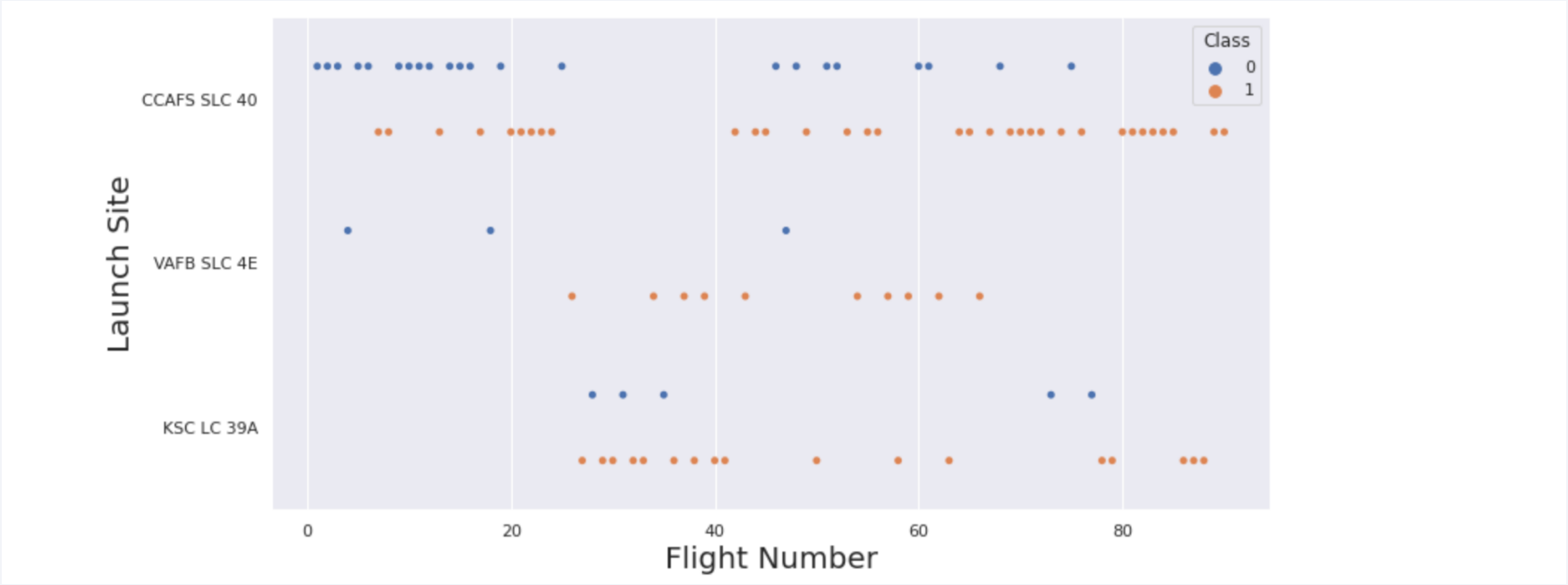
Results

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


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

- We can interpret that the more amount of flight at a launch site the greater the success rate at a launch site.

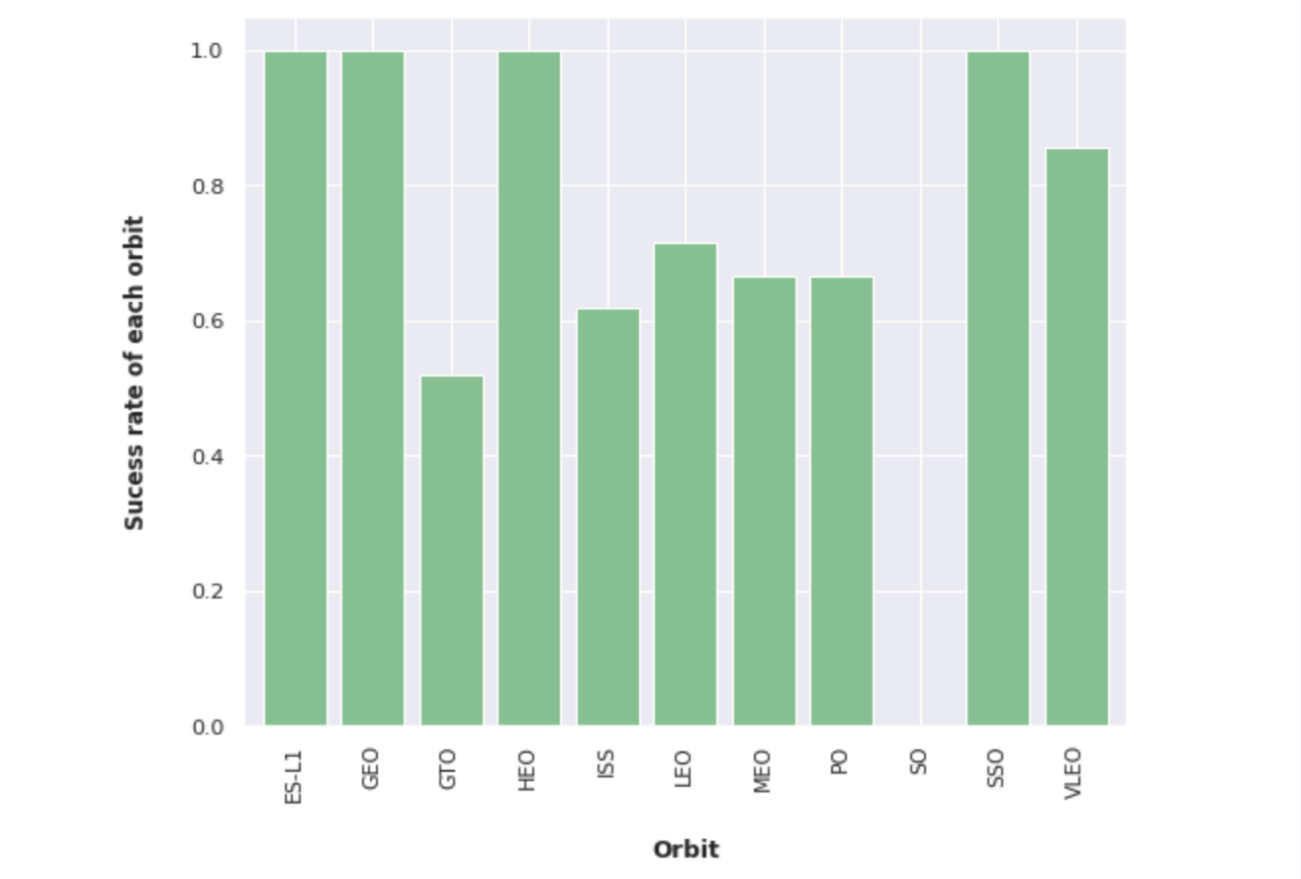


Payload vs. Launch Site

- We can interpret that the greater the payload mass for launch site CCAFS SLC 40, the higher the success rate for the rocket.

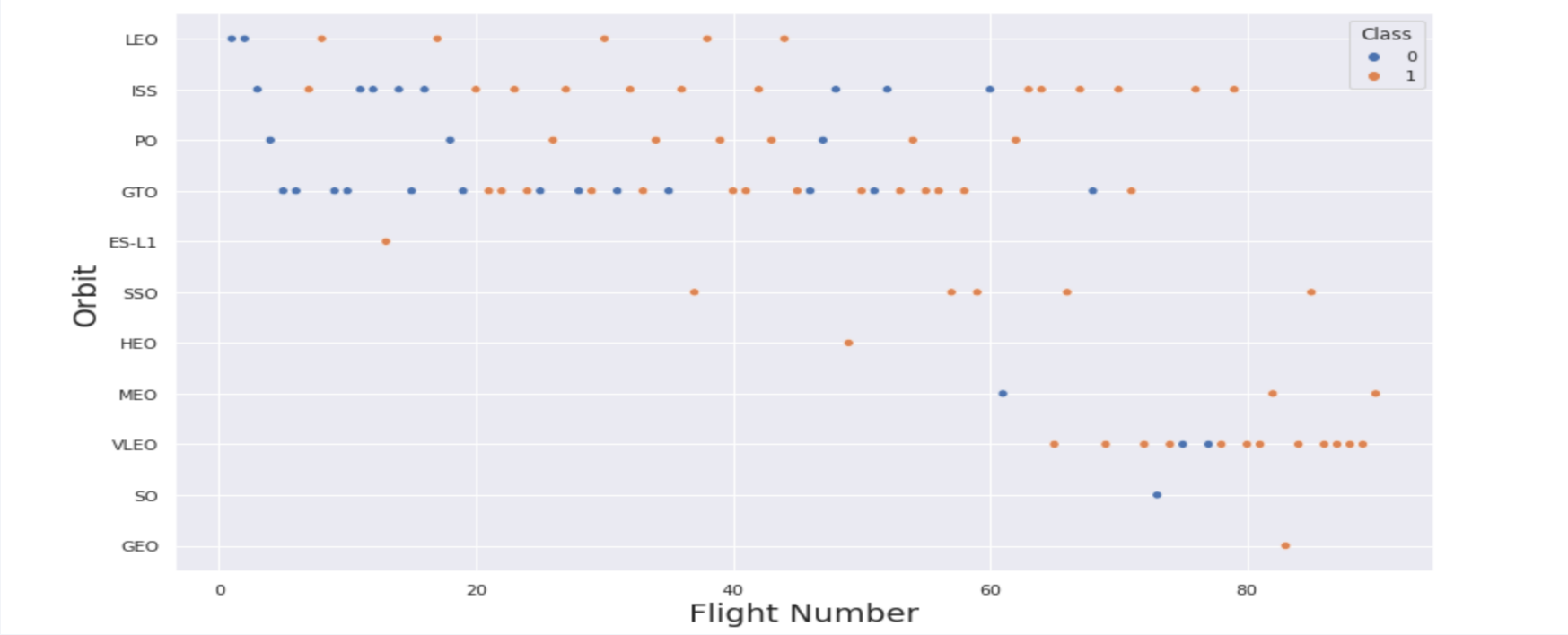
Success Rate vs. Orbit Type

- We can interpret from the plot that ES-1, GEO, HEO, SSO, and VLEO have had the most success rate of each orbit.



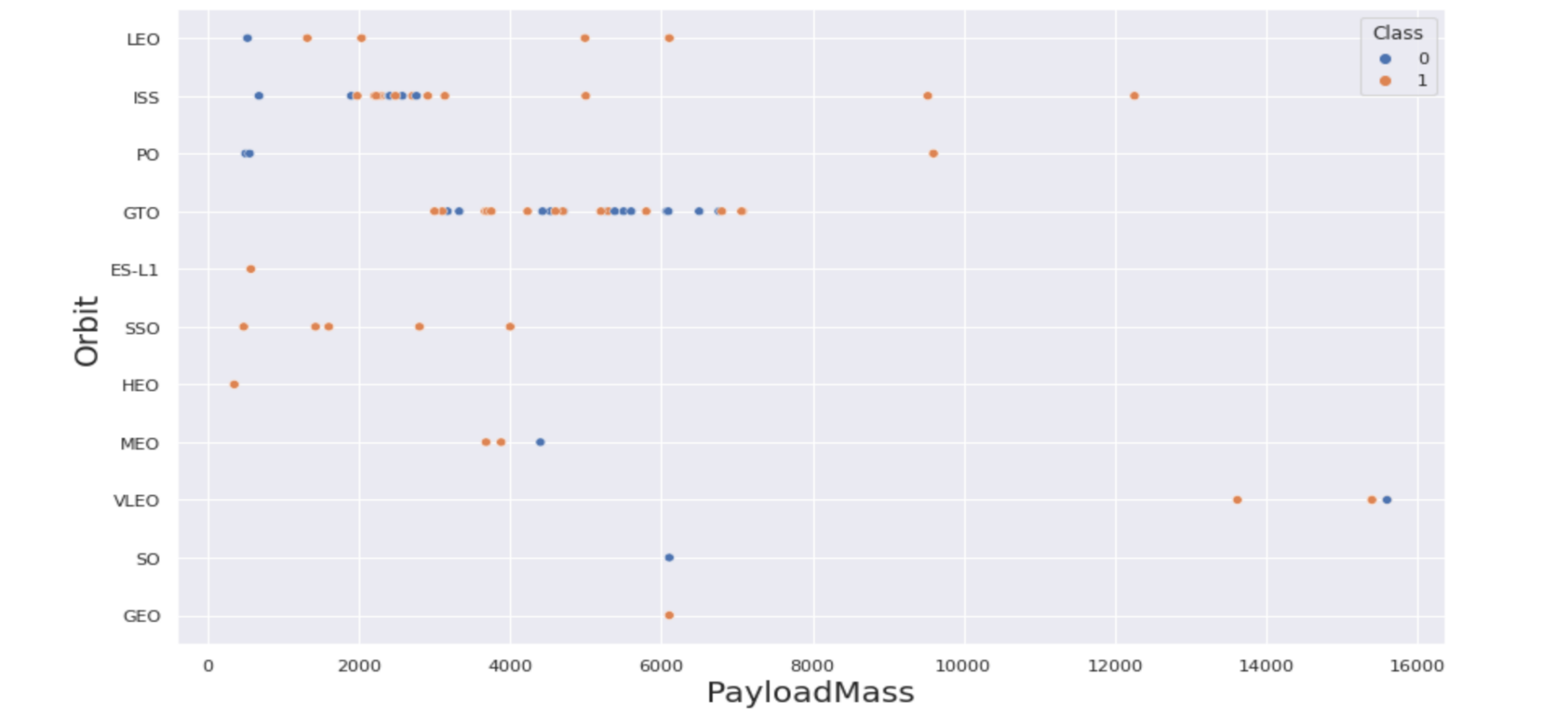
Flight Number vs. Orbit Type

- We can interpret that in LEO orbit, success rate is related to the number of flights where as, in the GTO orbit, there tends to be no relationship between flight number in regards to its orbit.



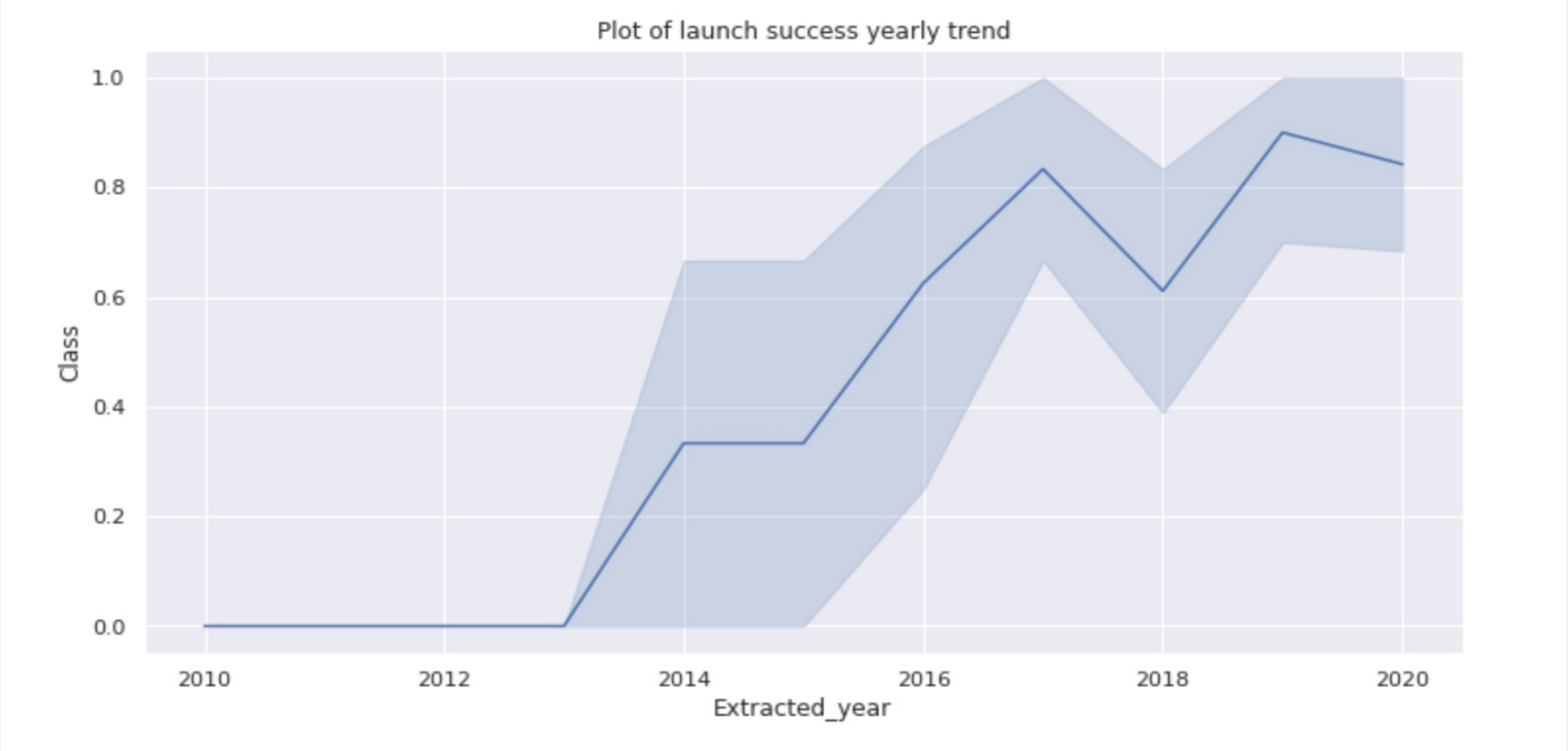
Payload vs. Orbit Type

- We can interpret that with heavy payloads, the success rate for landing are more directed towards PO, LEO, and ISS orbits.



Launch Success Yearly Trend

- We can interpret that the success rate since 2013 has shown an upward trend until 2020.



All Launch Site Names

- We used the function DISTINCT to show only unique launch sites from the Space X data table.

In [20]:
```
%sql SELECT  COUNT(LAUNCH_SITE),LAUNCH_SITE FROM SPACEXTBL GROUP BY LAUNCH_SITE
```

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

Out[20]:

COUNT(LAUNCH_SITE)	Launch_Site
26	CCAFS LC-40
34	CCAFS SLC-40
25	KSC LC-39A
16	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- Here we have used the function LIMIT 5 that will show us only 5 results from the table. We also used ‘CCA’ with percentage sign in the end that will suggest the launch site name which starts with CCA.

In [28]:
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

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

Out[28]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Here we used the function SUM that summates the total in the column of payload mass (kg). We have also used where clause that filters the dataset to perform calculations from the list of customers with NASA (CRS).

```
In [35]: %sql select sum(payload_mass__kg_) as "Total Payload Mass by NASA" from spacextbl where customer = 'NASA (CRS)';
 * sqlite:///my_data1.db
Done.
Out[35]:
```

Total Payload Mass by NASA
45596

Average Payload Mass by F9 v1.1

- Here we have used the function AVG that gives the average in the column of payload mass (kg). Also, the where clause filters the dataset to calculate and separate the booster version F9 v1.1.

```
In [36]: %sql select avg(payload_mass__kg_) as "Average Payload Mass by booster version" from spacextbl where booster_version like 'F9 v1.1%';
 * sqlite:///my_data1.db
Done.
Out[36]:
```

Average Payload Mass by booster version
2534.6666666666665

First Successful Ground Landing Date

- Here we have an error as we were not able to find a column name with Landing_Outcome.

```
In [119…
%sql select DATE from spacextbl where landing__outcome = 'Success (ground pad)' order by Date;

 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: landing__outcome
[SQL: select DATE from spacextbl where landing__outcome = 'Success (ground pad)' order by Date;]
(Background on this error at: http://sqlalche.me/e/13/e3q8)
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- Here we have an error as we were not able to find a column name with Landing_Outcome.

In [50]:

```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDING_OUTCOME
[SQL: SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (drone ship)'  AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS_
_KG_ < 6000;]
(Background on this error at: http://sqlalche.me/e/13/e3q8)
```

Total Number of Successful and Failure Mission Outcomes

- Here we have used sub queries to get the instance. We also used a wildcard like '%' to filter for the clause WHERE mission outcomes were either a success or a failure.

```
In [56]: %sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Success%';

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

Out[56]:

Successful Mission
100

```
In [58]: %sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Failure%';

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

Out[58]:

Failure Mission
1

```
In [59]: %sql SELECT COUNT(MISSION_OUTCOME) AS "Total Number of Successful and Failure Mission" FROM SPACEXTBL \
WHERE MISSION_OUTCOME LIKE 'Success%' OR MISSION_OUTCOME LIKE 'Failure%';

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

Out[59]:

Total Number of Successful and Failure Mission
101

Boosters Carried Maximum Payload

- Here we have used the clause DISTINCT which will only show unique values that are in the booster version column from out Space X table.

In [60]:
```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Out[60]:

Booster Versions which carried the Maximum Payload Mass
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

2015 Launch Records

- Here we have an error as we were not able to find a column name with Landing_Outcome.

In [120…

```
%sql SELECT month(DATE) as Month, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE year(DATE) = '2015' AND \
LANDING_OUTCOME = 'Failure (drone ship)';
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such function: month
[SQL: SELECT month(DATE) as Month, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE year(DATE) = '2015' AND  LANDING_OUTCOME = 'Failure
(drone ship)';]
(Background on this error at: http://sqlalche.me/e/13/e3q8)
```

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

- Here we have an error as we were not able to find a column name with Landing_Outcome.

In [72]:

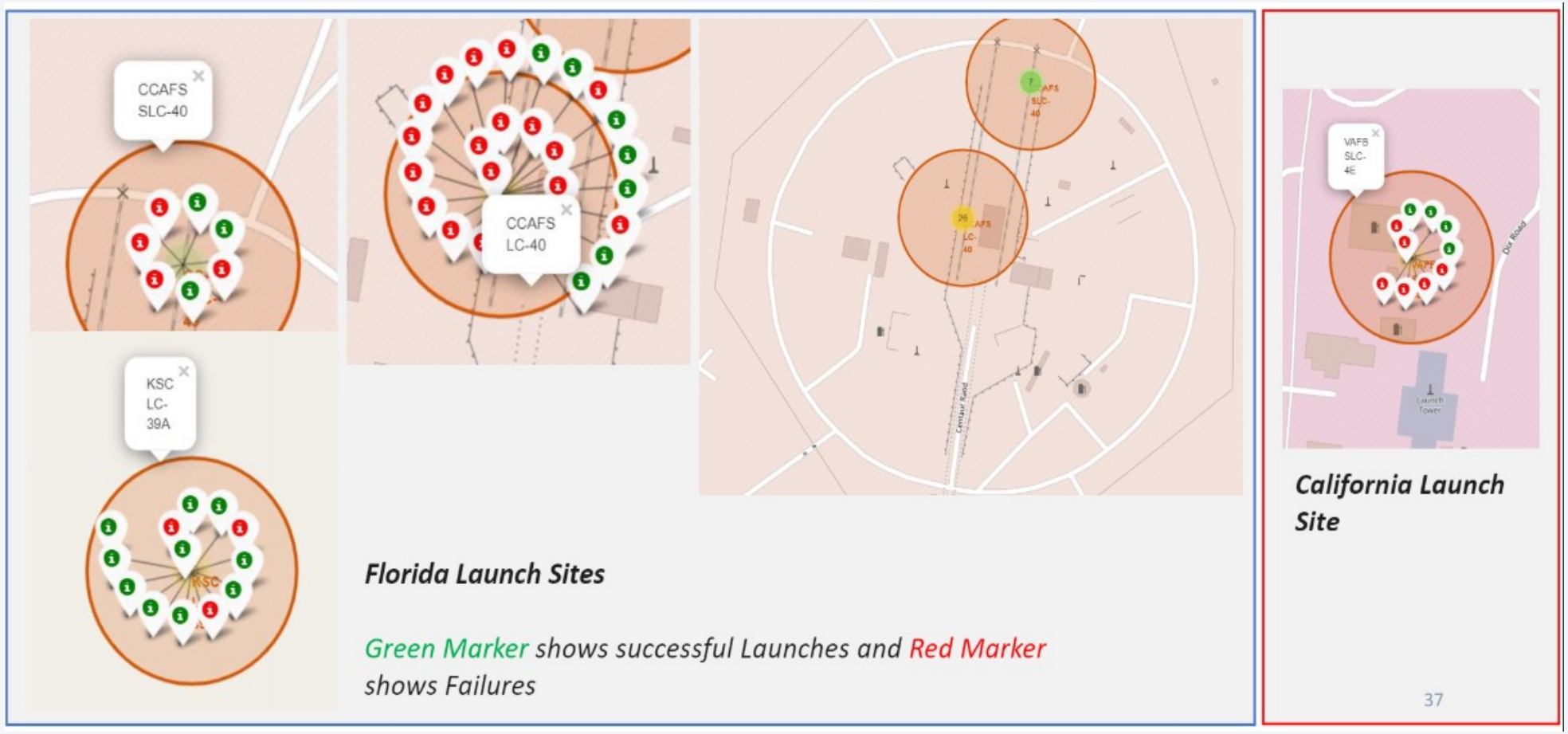
```
%sql SELECT LANDING_OUTCOME as "Landing Outcome", COUNT(LANDING_OUTCOME) AS "Total Count" FROM SPACEXTBL \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING_OUTCOME \
ORDER BY COUNT(LANDING_OUTCOME) DESC ;
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDING_OUTCOME
[SQL: SELECT LANDING_OUTCOME as "Landing Outcome", COUNT(LANDING_OUTCOME) AS "Total Count" FROM SPACEXTBL  WHERE DATE BETWEEN '2010-06-04'
AND '2017-03-20'  GROUP BY  LANDING_OUTCOME  ORDER BY COUNT(LANDING_OUTCOME) DESC ;]
(Background on this error at: http://sqlalche.me/e/13/e3q8)
```

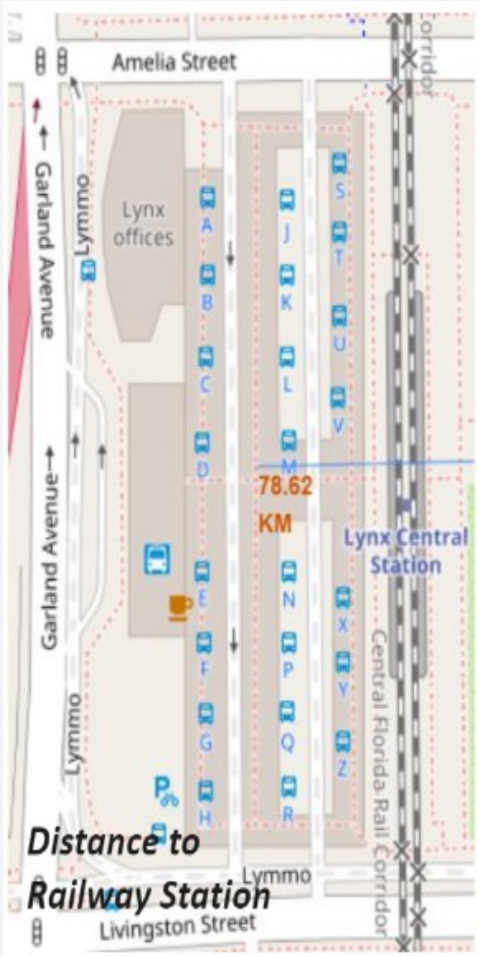

Section 3
Launch Sites
Proximities Analysis

All launch site global map markers

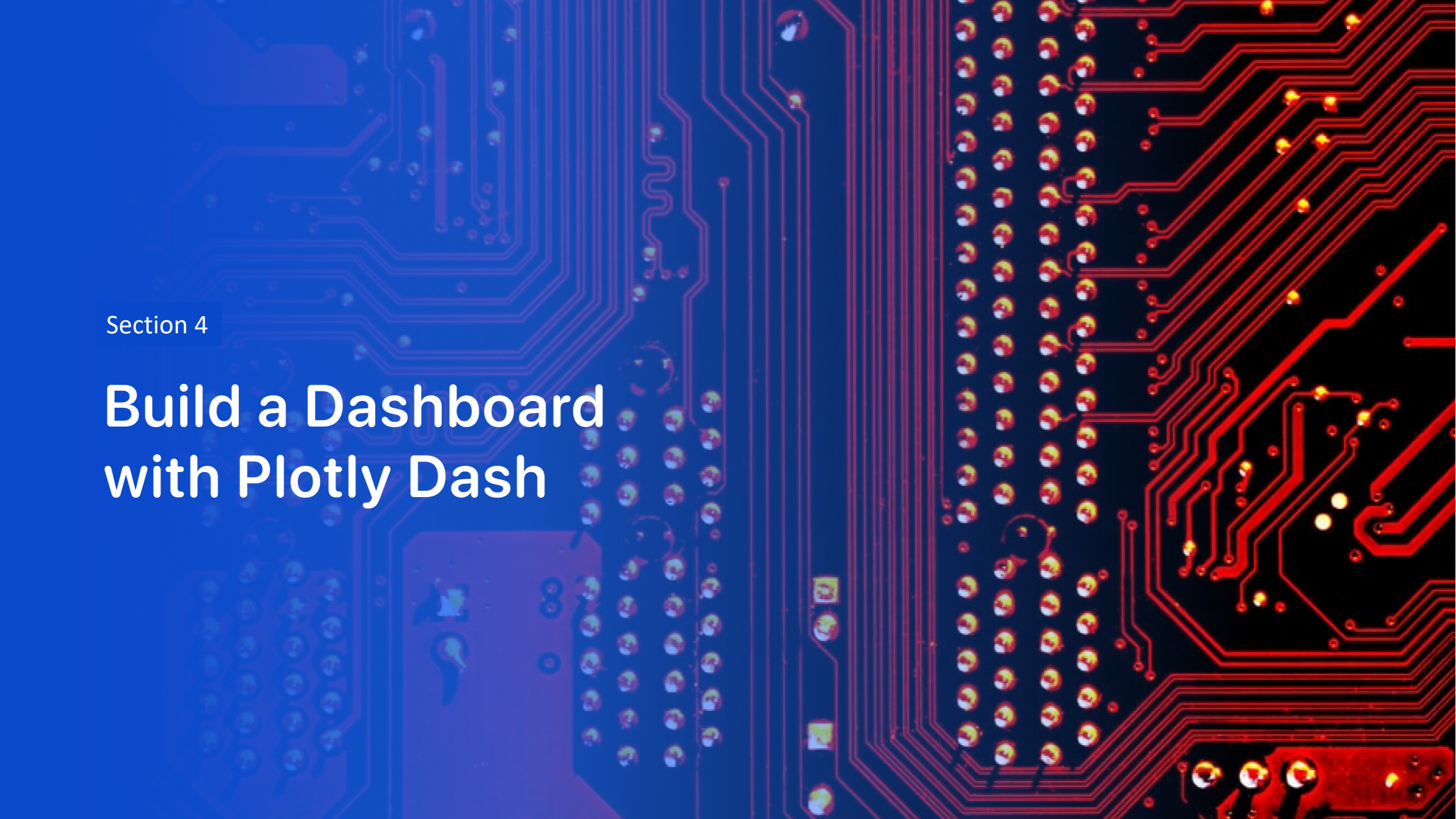
Markers showing launch sites with color labels



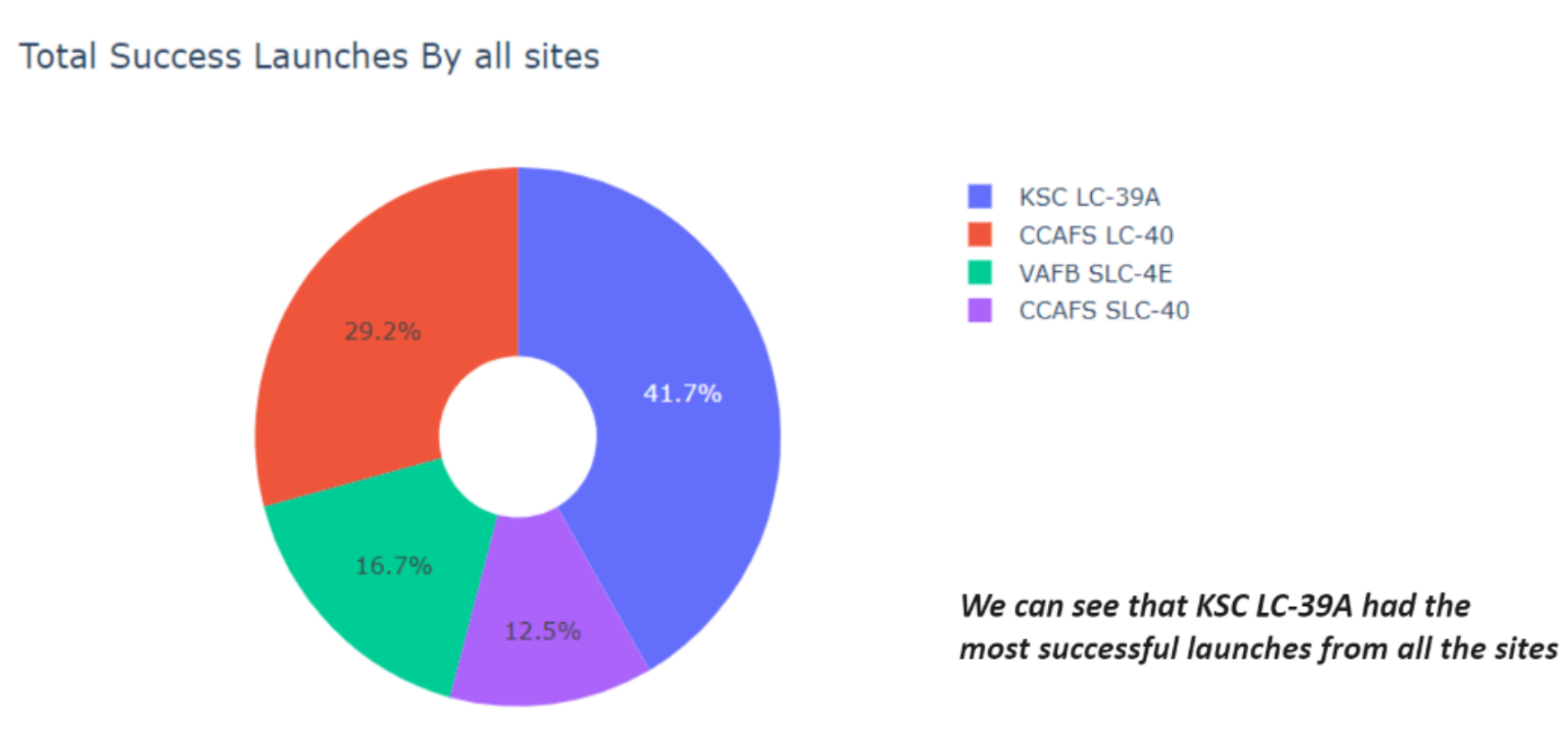
Launch site distance to landmarks



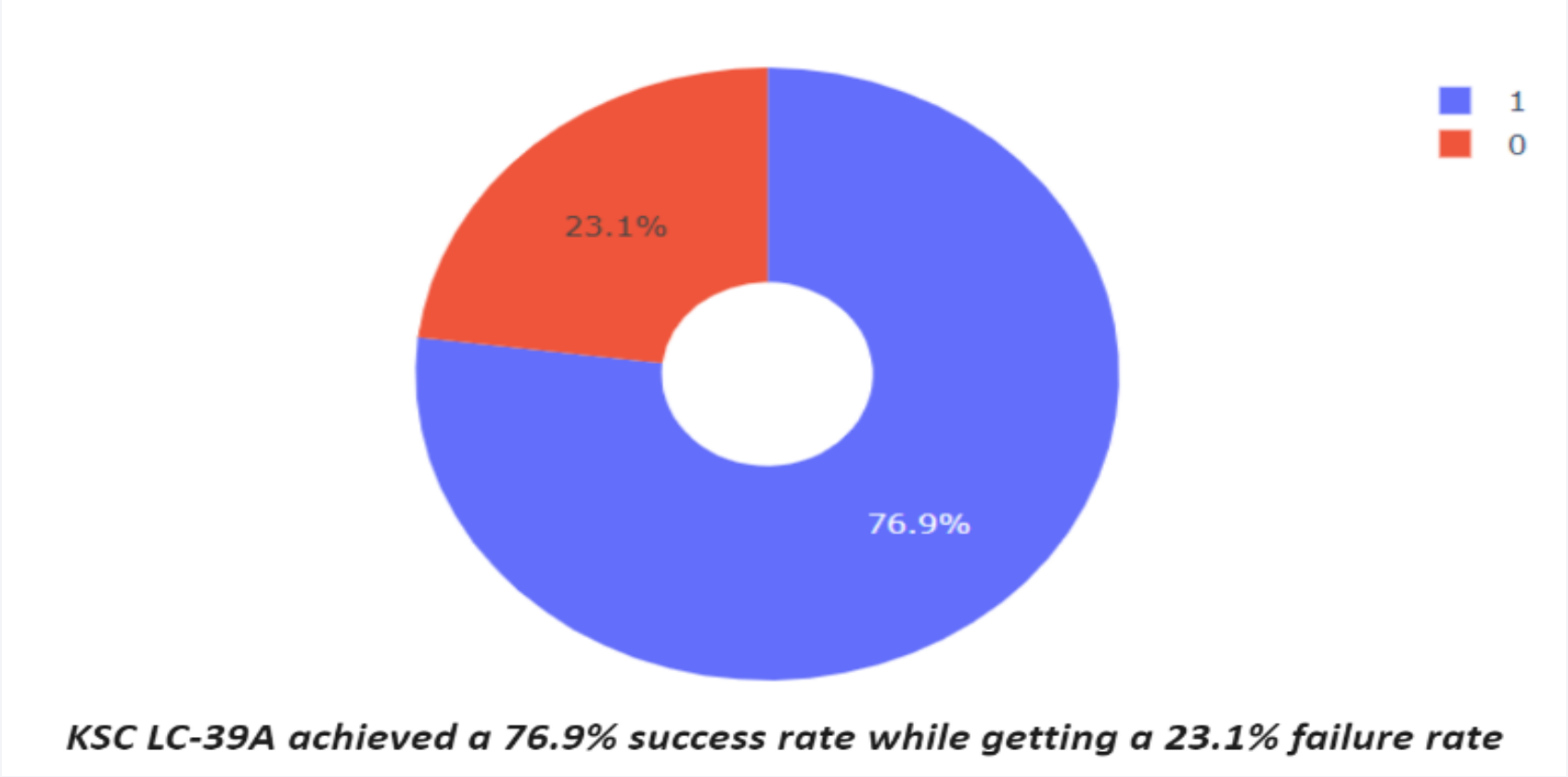
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

Pie Chart showing the success percentage achieved by each launch site

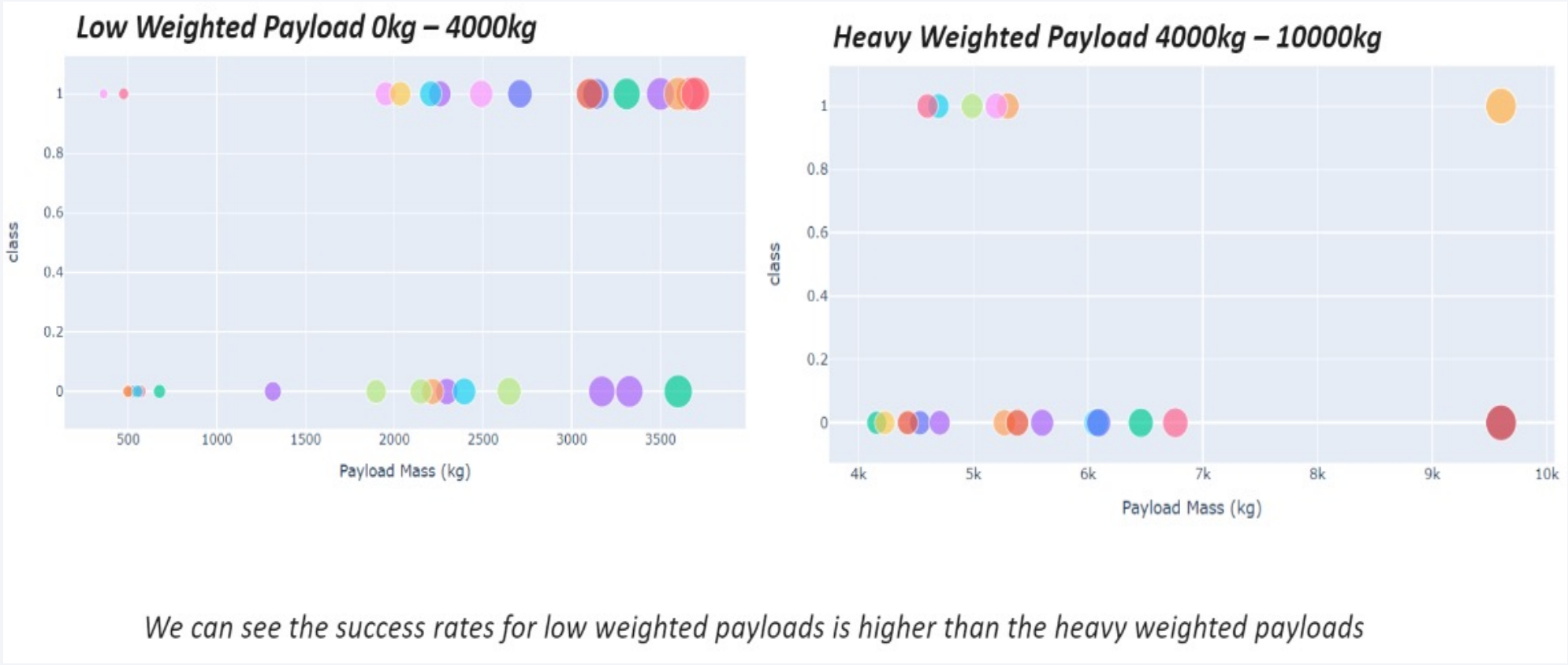


Pie chart showing the launch site with the highest success ratio



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 in the range slider



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

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- As we can interpret, the decision tree classifier is the model with the highest classification accuracy with a score of almost around 89% accuracy.

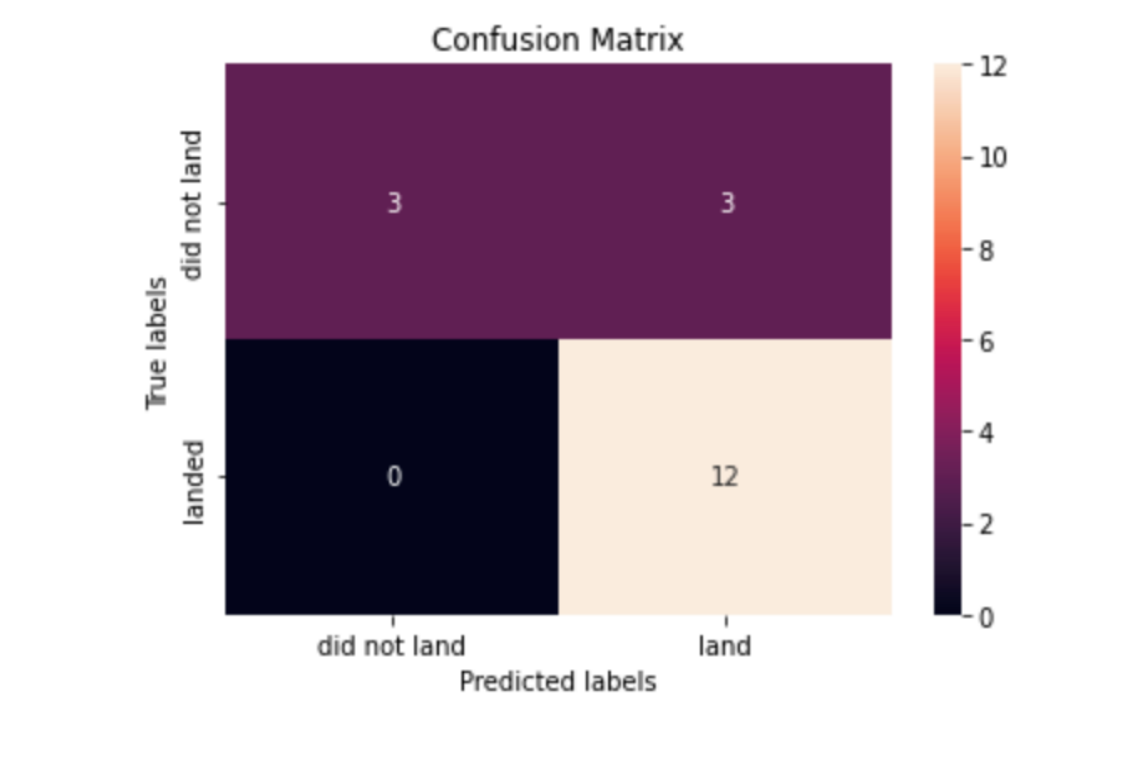
In [33]:

```
algorithms = {'KNN':knn_cv.best_score_,'Decision Tree':tree_cv.best_score_,'Logistic Regression':logreg_cv.best_score_,'SVM':svm_cv.best_s
best_algorithm = max(algorithms, key= lambda x: algorithms[x])

print('The method which performs best is \"',best_algorithm,'\" with a score of',algorithms[best_algorithm])
```

```
The method which performs best is " Decision Tree " with a score of 0.8910714285714285
```

Confusion Matrix

- We can interpret from the confusion matrix that the classifier can distinguish between different classes in the decision tree. However, a false positive could be marked as successful landing by the classifier.



Conclusions

- From these findings, we can conclude that

1. Orbit GEO, HEO, SSO, VLEO, and ES-L1 had the most success rate.
2. The success rate is directly proportional as the larger the flight amount at a launch site, the greater the success rate at a launch site.
3. KSC LC-39A has had the most successful launches of any sites.
4. Launch success rate started to have an upward trend since the year 2013 until the year 2020.
5. The decision tree classifier algorithm is the best for machine learning model for the dataset.

Thank you!