IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Achraf Fouhad

25 April 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result from Machine Learning Lab

Introduction

SpaceX is an innovative company that revolutionized the space industry by offering a dedicated Falcon 9 rocket launch for just $62 million. Other providers cost over $165 million each. A large part of this savings is thanks to his SpaceX brilliant idea of reusing the first phase of the launch by relanding the rocket for the next mission. Repeat this process and the price will drop further. As a data scientist at a startup competing with SpaceX, my goal in this project is to build a machine learning pipeline to predict future first-stage landing outcomes. This project is important in determining the appropriate pricing for rocket launches for SpaceX.

The problems included:

- Identifying all factors that influence the landing outcome.
- The relationship between each variables and how it is affecting the outcome.
- The best condition needed to increase the probability of successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding for categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Data collection is the process of collecting and measuring information about target variables in an established system. This allows you to answer relevant questions and evaluate results. As mentioned above, the dataset was collected from Wikipedia via REST API and web scraping.

For REST APIs, it starts with a Get request. I then decoded the response content as json and converted it to a pandas dataframe using json_normalize() . Then I cleaned the data, checked for missing values and filled in what I needed.

Web scraping uses BeautifulSoup to extract the starting record as an HTML table, parse the table and convert it to a pandas dataframe for further analysis.

Data Collection - SpaceX API

Get request for rocket launch data using API

Use json_normalize method to convert json result to dataframe

Performed data cleaning and filling the missing value

From:

https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Data_Collection.ipynb

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

response = requests.get(spacex_url)
```

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

```
# Lets take a subset of our dataframe keeping only the features we want a
nd the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number',
'date_utc']]

# We will remove rows with multiple cores because those are falcon rocket
s with 2 extra rocket boosters and rows that have multiple payloads in a
single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the s
ingle value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then ex
tracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection - Scraping

Request the Falcon9 Launch Wiki page from url

Create a BeautifulSoup from the HTML response

Extract all column/variable names from the HTML header

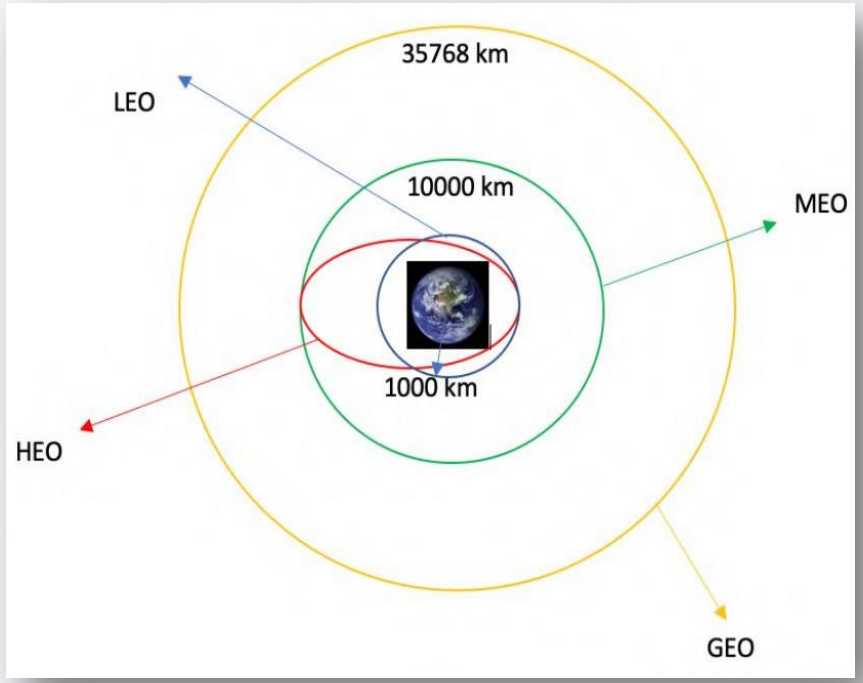
From: https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Data_Collection_with_Web_Scraping.ipynb

```
# use requests.get() method with the provided static_url
# assign the response to a object
data = requests.get(static_url).text
```

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

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plai
nrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding t
o launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

Data Wrangling



From:

https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Data_Wrangling.ipynb

Data wrangling is the process of cleaning up and consolidating messy and complex data sets for easy access and exploratory data analysis.

First calculate the number of launches at each location, then calculate the number and occurrence of mission outcomes by orbit type.

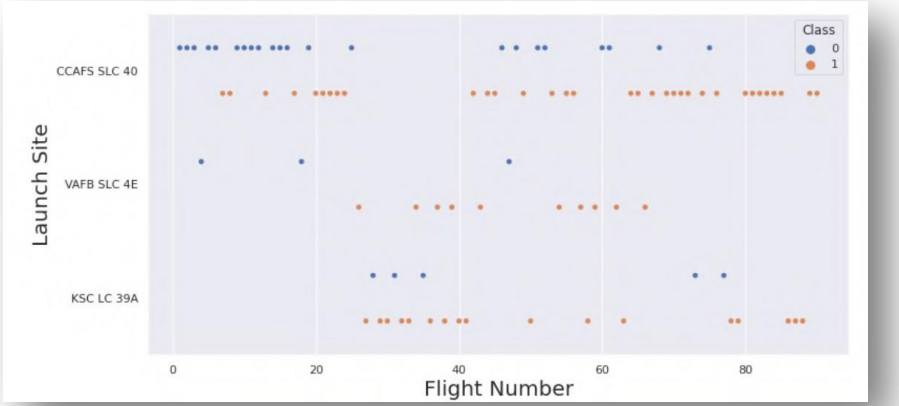
Then create a landing result label from the result column. This facilitates further analysis, visualization, and ML. Finally, export the results to a CSV file.

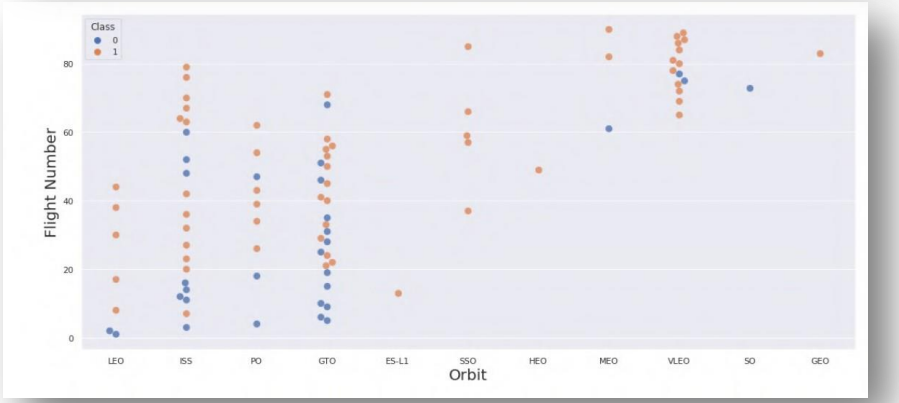
EDA with Data Visualization

Scatterplots were first used to find relationships between attributes like:

- Payload and Flight Number.
- Flight Number and Launch Site.
- Payload and Launch Site.
- Flight Number and Orbit Type.
- Payload and Orbit Type.

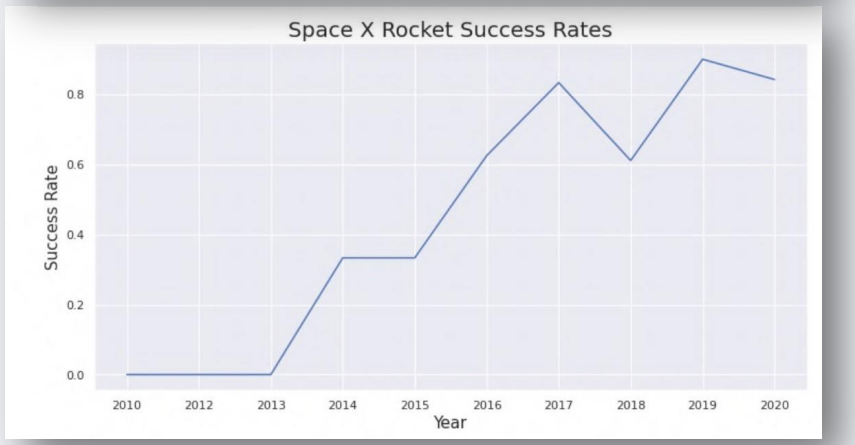
Scatter plots show dependency of attributes on each other. Once a pattern is determined from the graphs. It's very easy tosee which factors affecting the most to the success of the landing outcomes.





https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Exploratory_Data_Analysis_with_Visualisation_Lab.ipynb

EDA with Data Visualization



Once you have shown the relationship using a scatterplot. Then use other visualization tools such as bar charts and line charts for further analysis.

Bar charts are one of the easiest ways to interpret relationships between attributes. In this case, use a bar chart to identify lanes with the highest probability of success.

Then use a line chart to show trends or patterns in attributes over time. In this case, it is used to ascertain year-to-year trends in launch success.

We then use feature engineering to predict success in future modules by creating dummy variables for the categorical columns.

https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Exploratory_Data_Analysis_with_Visualisation_Lab.ipynb

EDA with SQL

I used SQL to run many queries to better understand the dataset. for example:

- Display of the name of the departure city.
- Displays 5 records whose start position starts with the character string "CCA".
- Display of total payload mass carried by NASA (CRS) launch boosters.
- Display of average payload mass carried by booster version F9 v1.1.
- List of dates when the first landing result on the ground pad was achieved.
-Lists the names of boosters that have been successful with drone ships and have a payload mass greater than 4000 and less than 6000.
- List of total number of successful and failed mission outcomes.
- A list of names of booster_versions that carry maximum payload mass.
- List of failed land_outcomes, booster versions, and launch site names for 2015 drones.
- Sort the number of landing results or successes from 04/06/2010 to 20/03/2017 in descending order. https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Exploratory_Data_Analysis_with_SQL.ipynb

Build an Interactive Map with Folium

To view launch data in an interactive map. We take the latitude and longitude coordinates at each launch point and add a circular marker around each launch point along with a label of the launch point name.

Then I assigned the dataframe launch_outcomes(failure,success) to classes 0 and 1 and marked red and green markers on the map with MarkerCluster().

We then used Haversine's formula to calculate the distance from the launch location to various landmarks to find answers to the following questions:

- How close is the launch site to railroads, highways, and beaches?
- What is the distance from the launch site to nearby cities??

From: https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/notebook_Interactive_Visual_Analytics_with_Folium.ipynb

Build a Dashboard with Plotly Dash

- Created interactive dashboards using Plotly Dash, allowing users to interact with the data as desired.
- Created a pie chart showing the total number of launches at a particular location.
- We then created a scatterplot showing the results for the various booster versions versus payload mass (kg).

The link of the app.py:: https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

Building the Model

- Load the dataset into NumPy and Pandas
- Transform the data and then split into training and test datasets
- Decide which type of ML to use
- set the parameters and algorithms to GridSearchCV and fit it to dataset.

Evaluating the Model

- Check the accuracy for each model
- Get tuned hyperparameters for each type of algorithms.
- plot the confusion matrix.

Improving the Model

- Use Feature Engineering and Algorithm Tuning

Find the Best Model

- The model with the best accuracy score will be the best performing model.

From: https://github.com/aka-ds/-Applied-Data-Science-Capstone-SpaceX/blob/main/spacex_dash

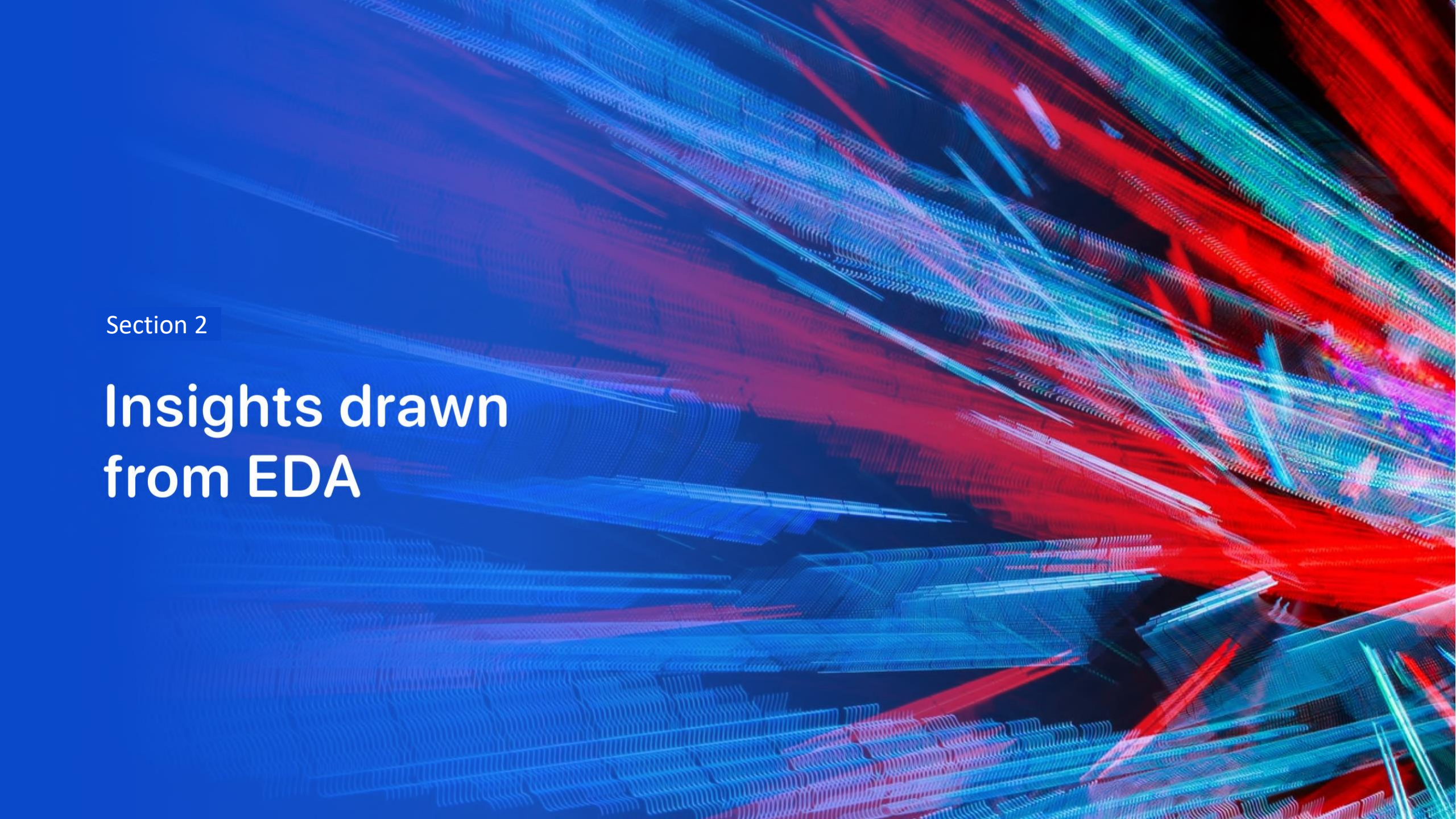
Results

The results will be categorized to 3 main results which is:

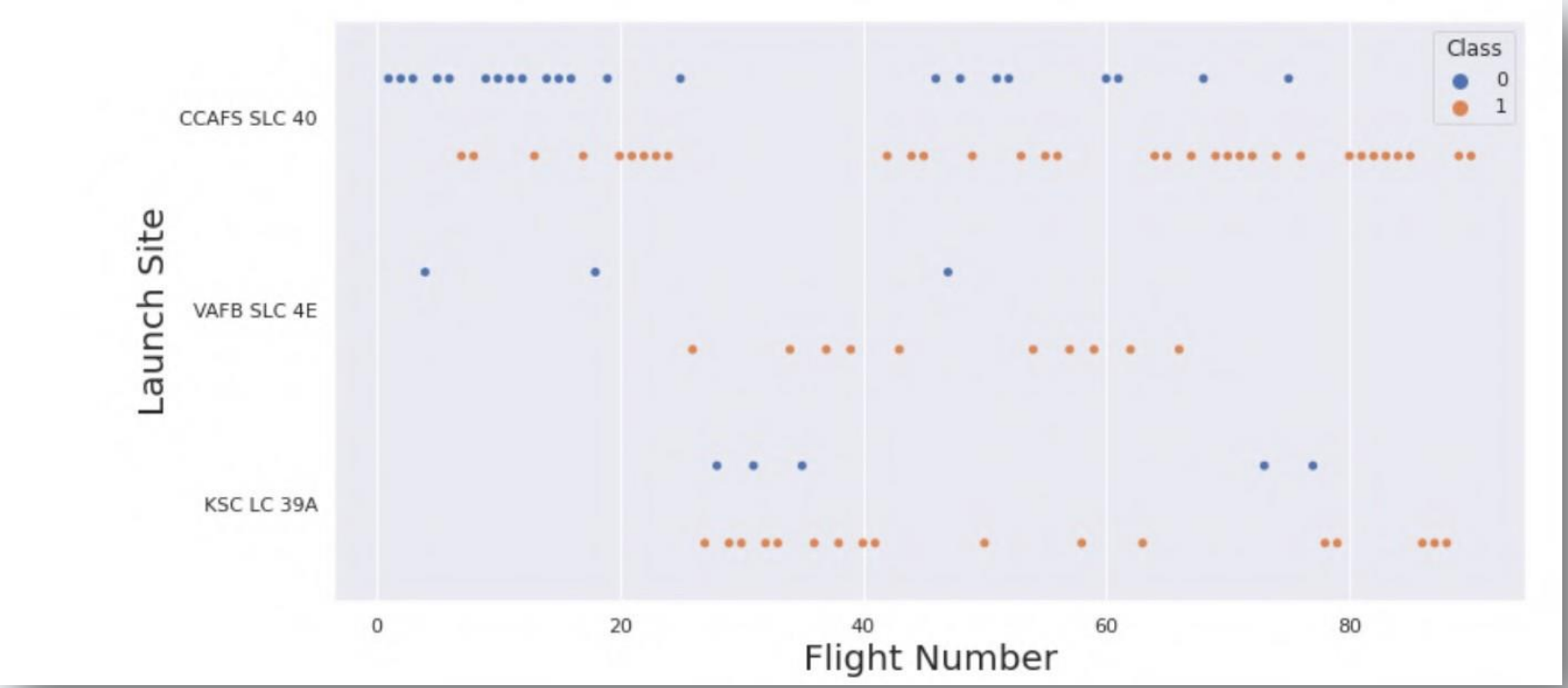
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA



Flight Number vs. Launch Site



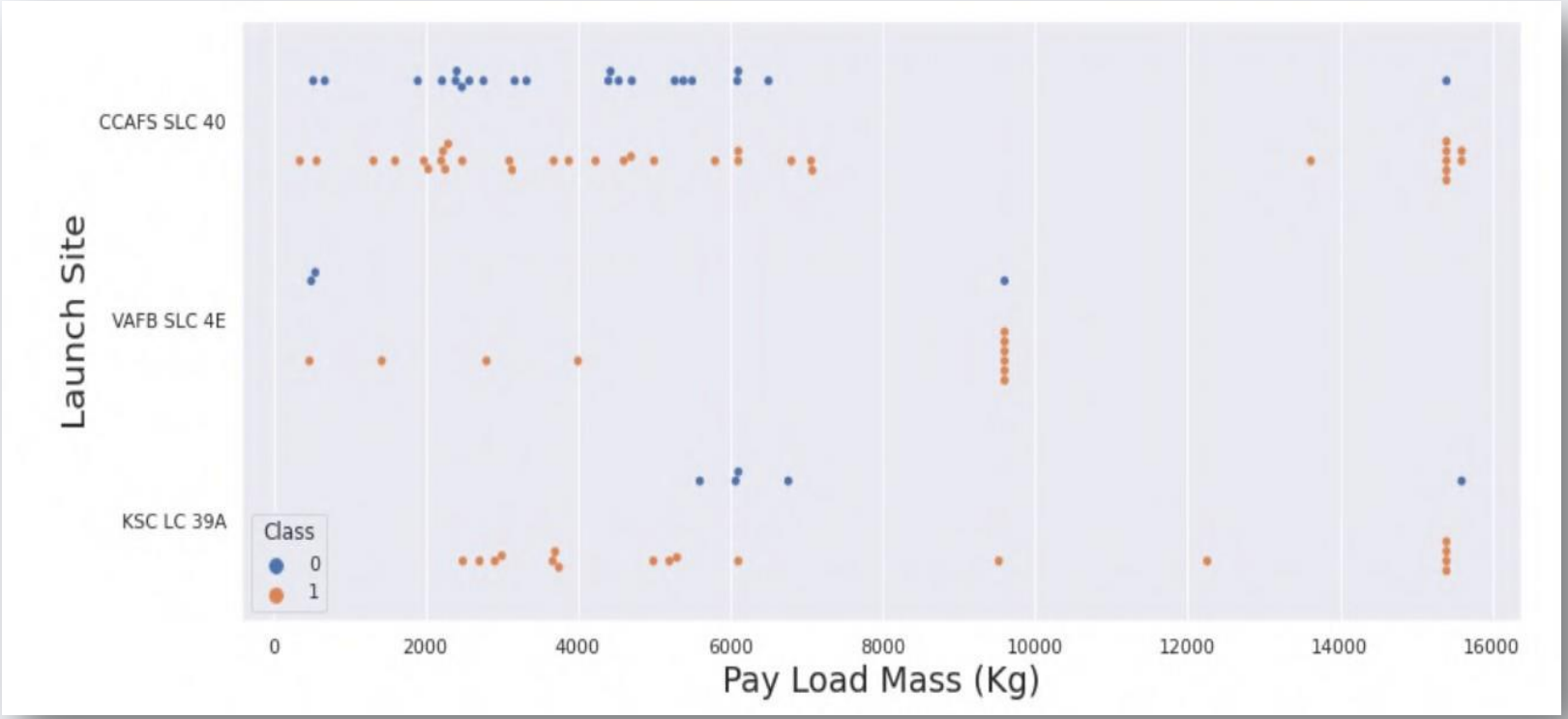
This scatterplot shows that the more flights from the launch site, the higher the success rate.

However, site CCAFS SLC40 shows a slight pattern of that.

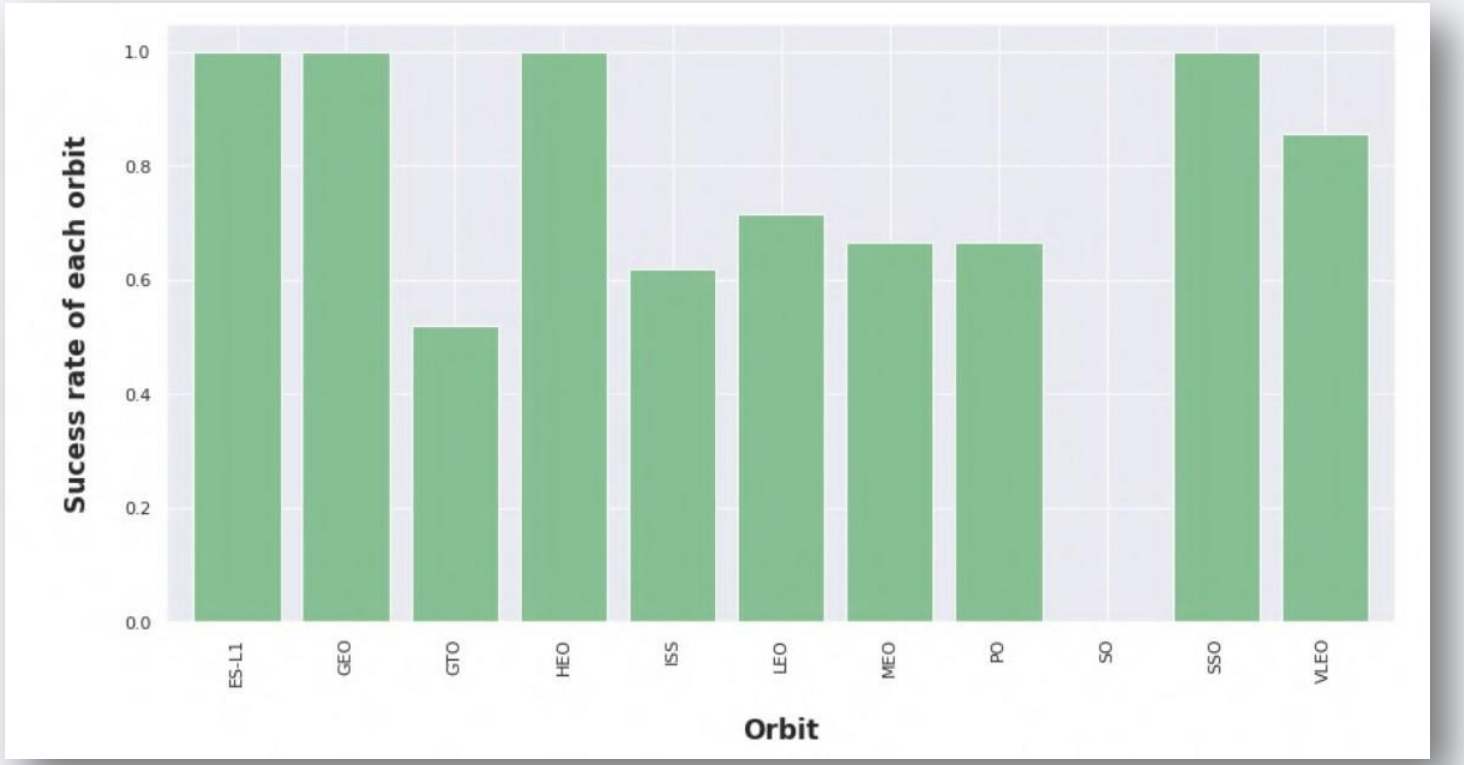
Payload vs. Launch Site

This scatterplot shows that the probability of success increases significantly when the payload mass exceeds 7000 kg.

However, there is no clear pattern that launch site success rate depends on payload mass.



Success Rate vs. Orbit Type



This figure shows that some trajectories have 100% success rate, such as SSO, HEO, GEO, and ES-L1, while SO trajectory has 0% success rate, so trajectory does not affect the landing outcome. It shows the possibilities of giving.

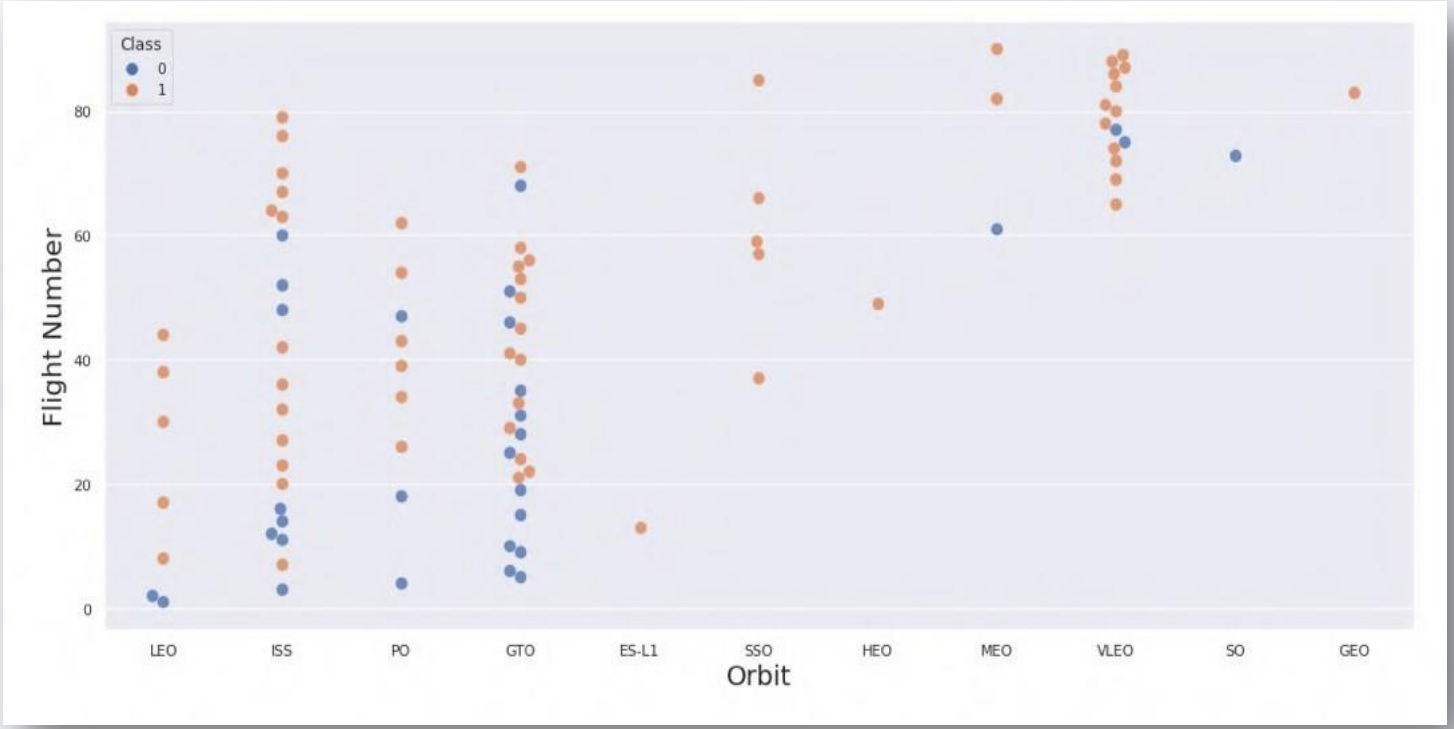
However, a more detailed analysis shows that some of these orbitals occur only once, such as GEO, SO, HEO and ES-L1. In other words, these data require more datasets to see patterns and trends before drawing conclusions.

Flight Number vs. Orbit Type

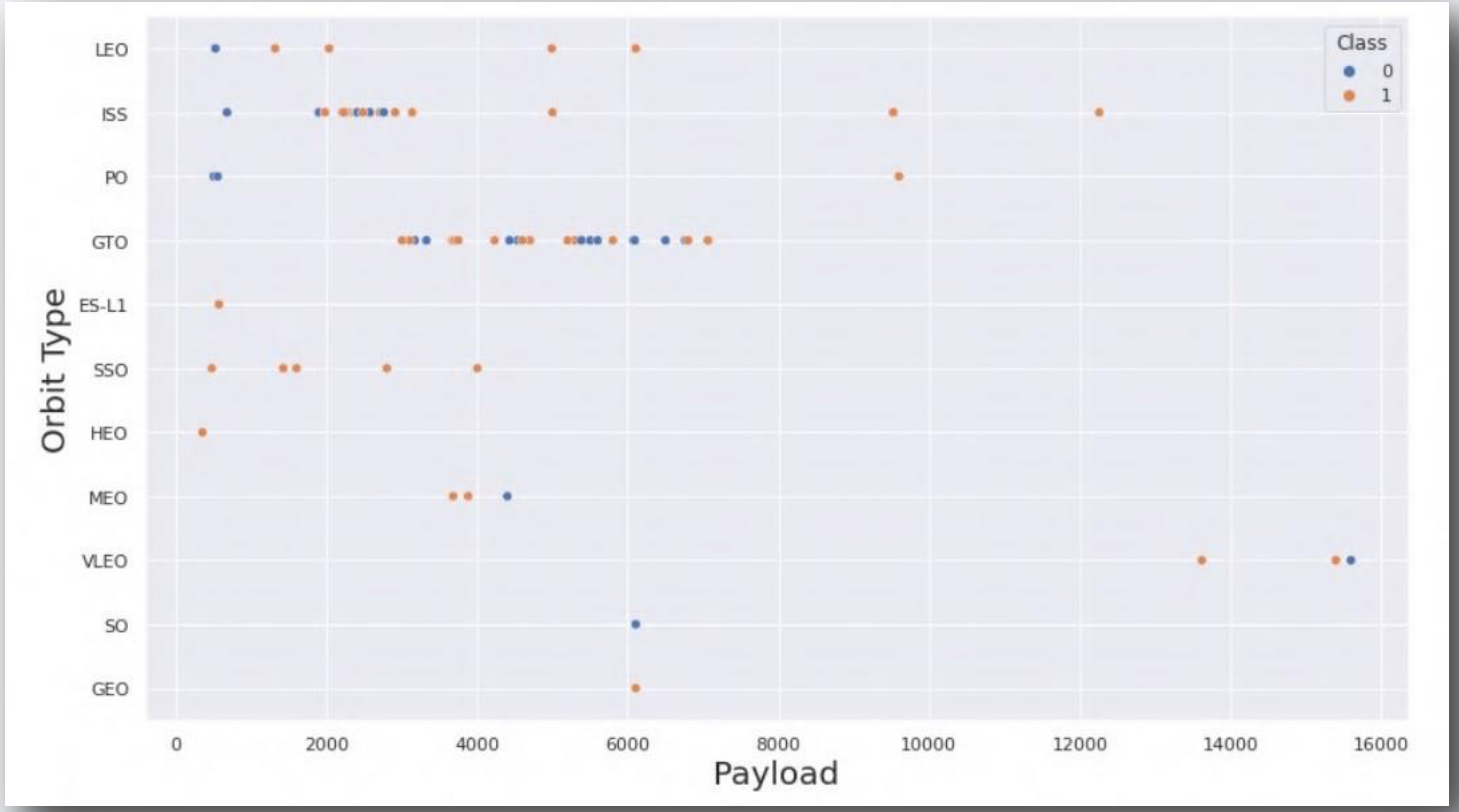
The scatterplot shows that in general, the higher the number of flights, the higher the success rate for each trajectory (particularly the LEO trajectory), except for the GTO trajectory, which shows no relationship between the two attributes.

Trajectories that occur only once must also be excluded from the statement above, as they are required.

more records.



Payload vs. Orbit Type



Heavier payload has positive impact on LEO, ISS and P0 orbit. However, it has negative impact on MEO and VLEO orbit.

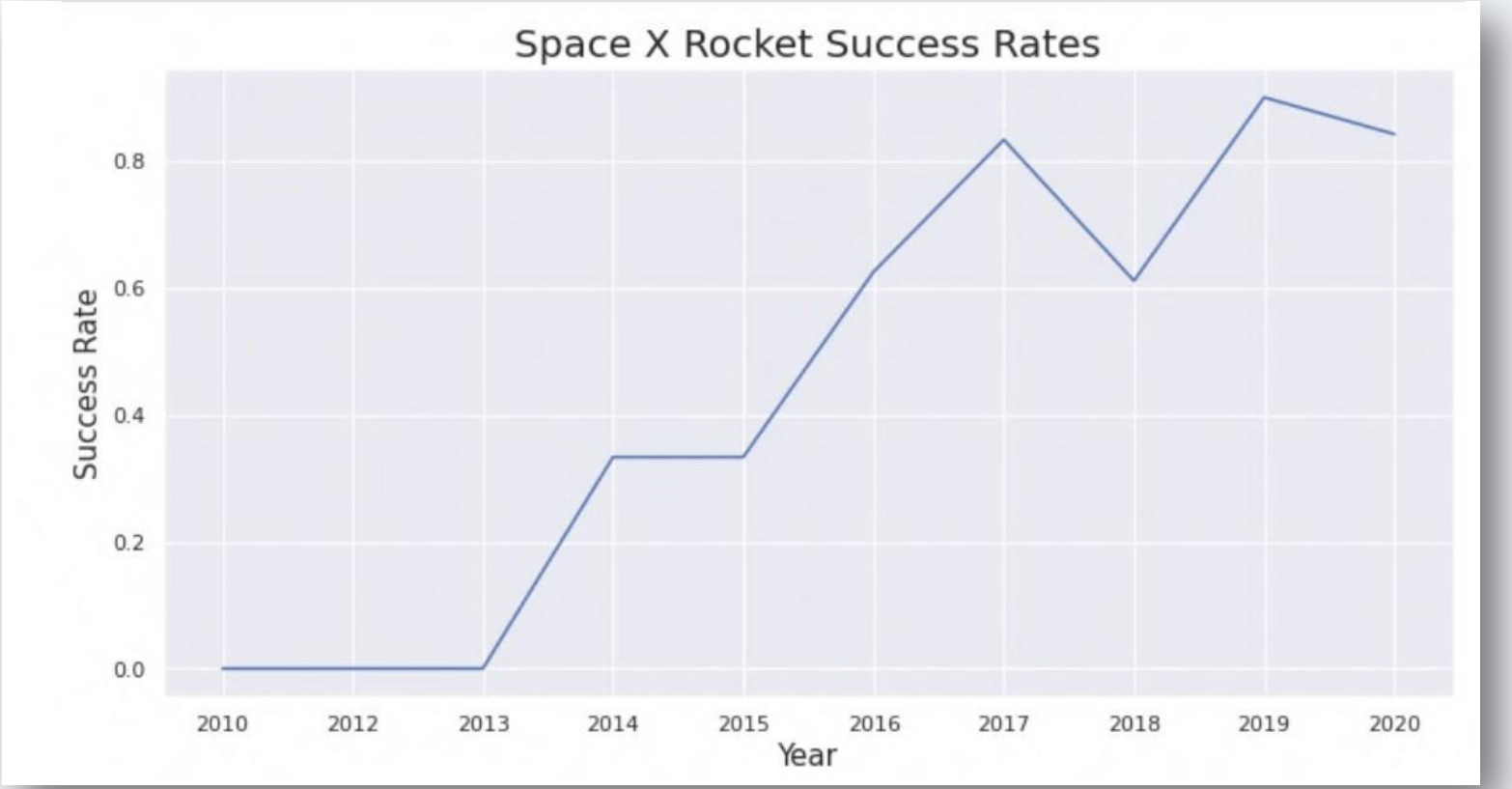
GTO orbit seem to depict no relation between the attributes.

Meanwhile, again, SO, GEO and HEO orbit need more dataset to see any pattern or trend.

Launch Success Yearly Trend

These numbers show a clear upward trend from 2013 to 2020.

If this trend continues next year. The success rate increases steadily until it reaches 1/100%.



All Launch Site Names

We used the DISTINCT keyword to display only unique launch sites from the SpaceX data.

```
In [5]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[5]:

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

Launch Site Names Begin with 'CCA'

Using the query above, I have displayed 5 records with a starting position starting 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

Total Payload Mass

We used the following query to calculate the total payload carried by the NASA booster as 45596.

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Payload Mass by NASA (CRS)
45596

Average Payload Mass by F9 v1.1

The average payload mass of the F9 v1.1 booster version was calculated as 2928.4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

First Successful Ground Landing Date

Use the min() function to find the result

The date of the first successful landing on the ground platform was confirmed as December 22, 2015.

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

A '**WHERE**' clause was used to filter the boosters that landed on the drone ship, and an '**AND**' condition was applied to determine whether landings with a payload mass greater than 4000 and less than 6000 were successful.

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

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

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Successful Mission
100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Failure Mission
1

Boosters Carried Maximum Payload

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

We used subqueries in the **“WHERE”** clause and the **“MAX()”** function to identify boosters carrying the largest payloads.

2015 Launch Records

A combination of **“WHERE”** clauses, **“LIKE”**, **“AND”**, and **“BETWEEN”** conditions was used to filter the 2015 drone ship landing failure results, booster versions, and launch site names.

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.
databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

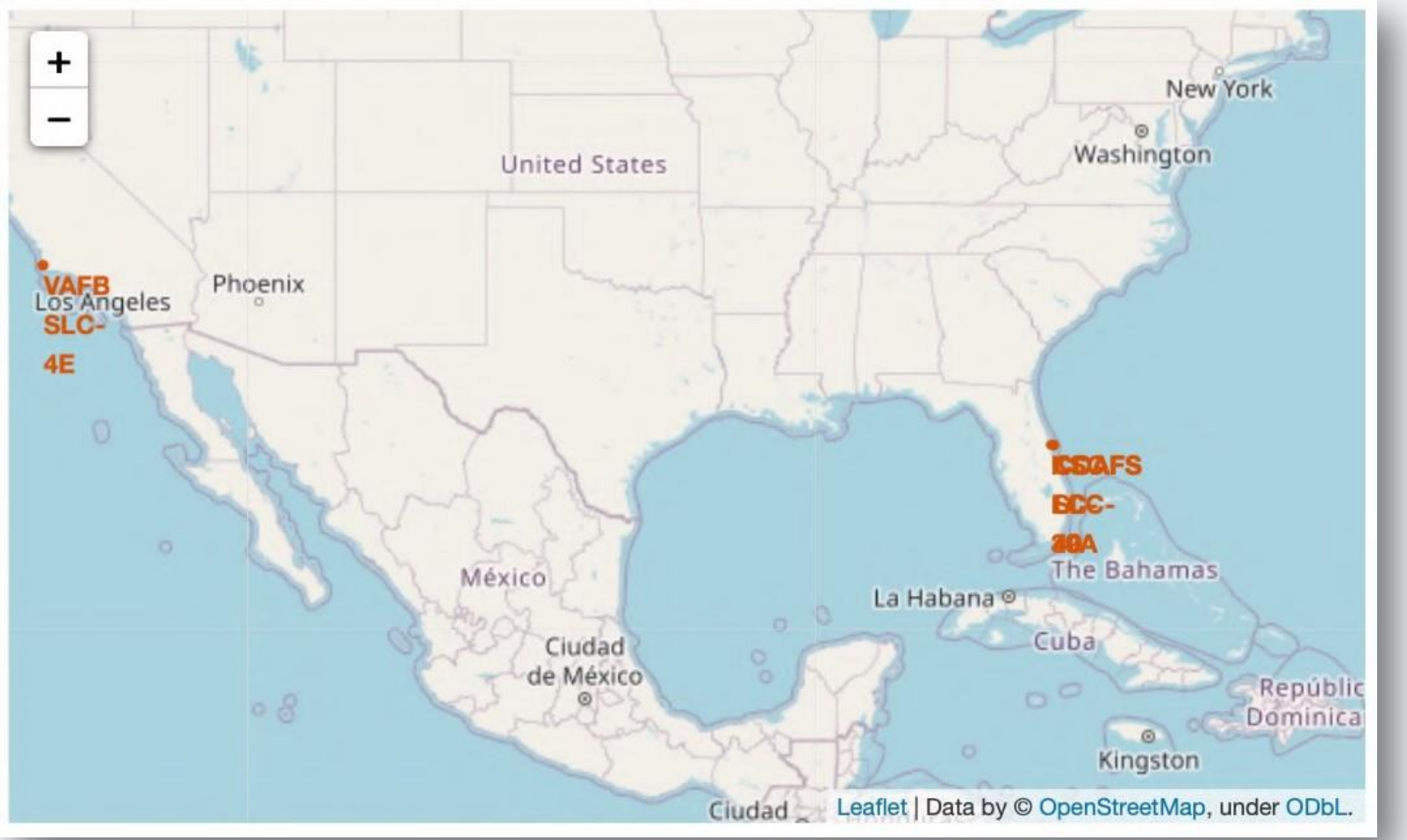
Landing Outcome	Total Count
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

We selected the landing results and COUNT landing results from the data and used the WHERE clause to filter the landing results between 2010-06-04 and 2010-03-20.

We applied a GROUP BY clause to group the landing results and an ORDER BY clause to sort the grouped landing results in descending order.


Section 3
Launch Sites
Proximities Analysis

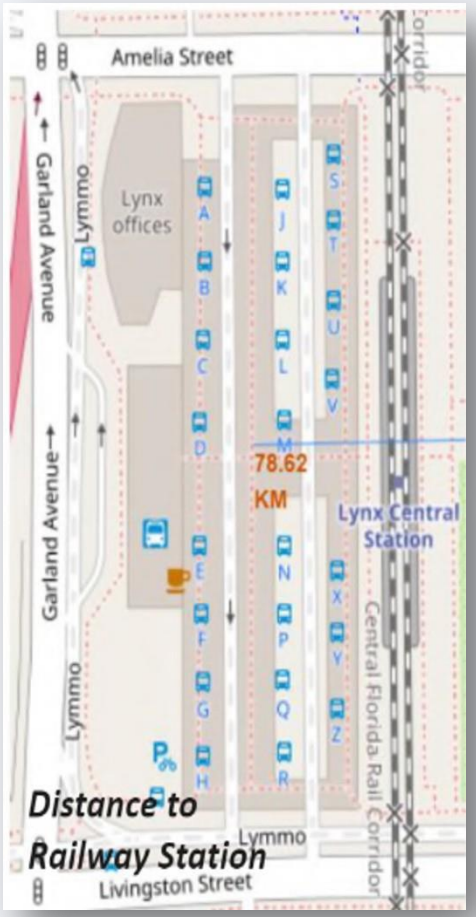
Location of all the Launch Sites

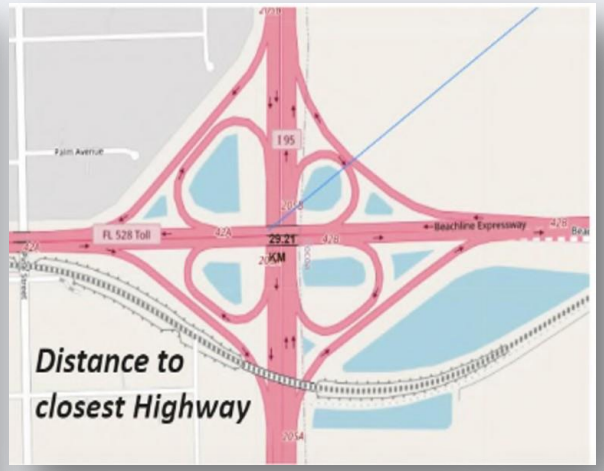


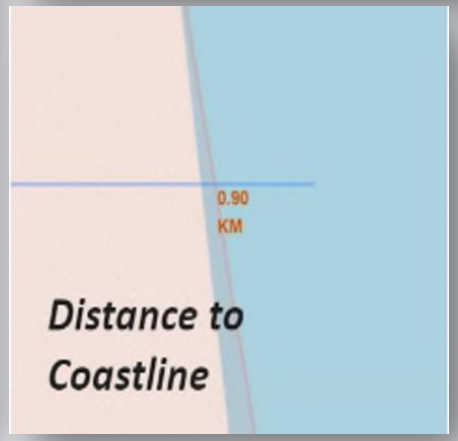
It turns out that all SpaceX launch sites are in the US

Markers showing launch sites with color labels

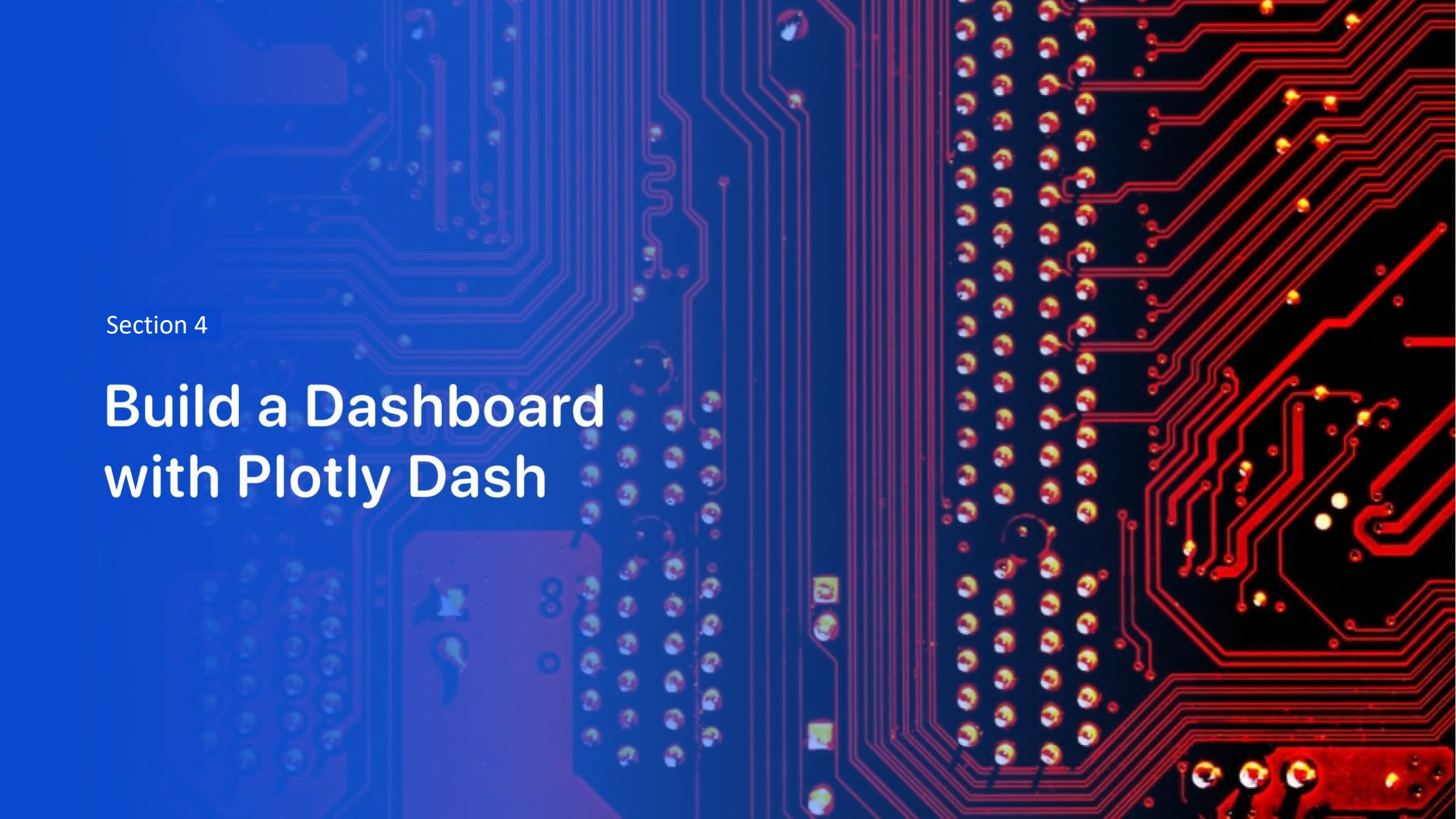
Launch Sites Distance to Landmarks



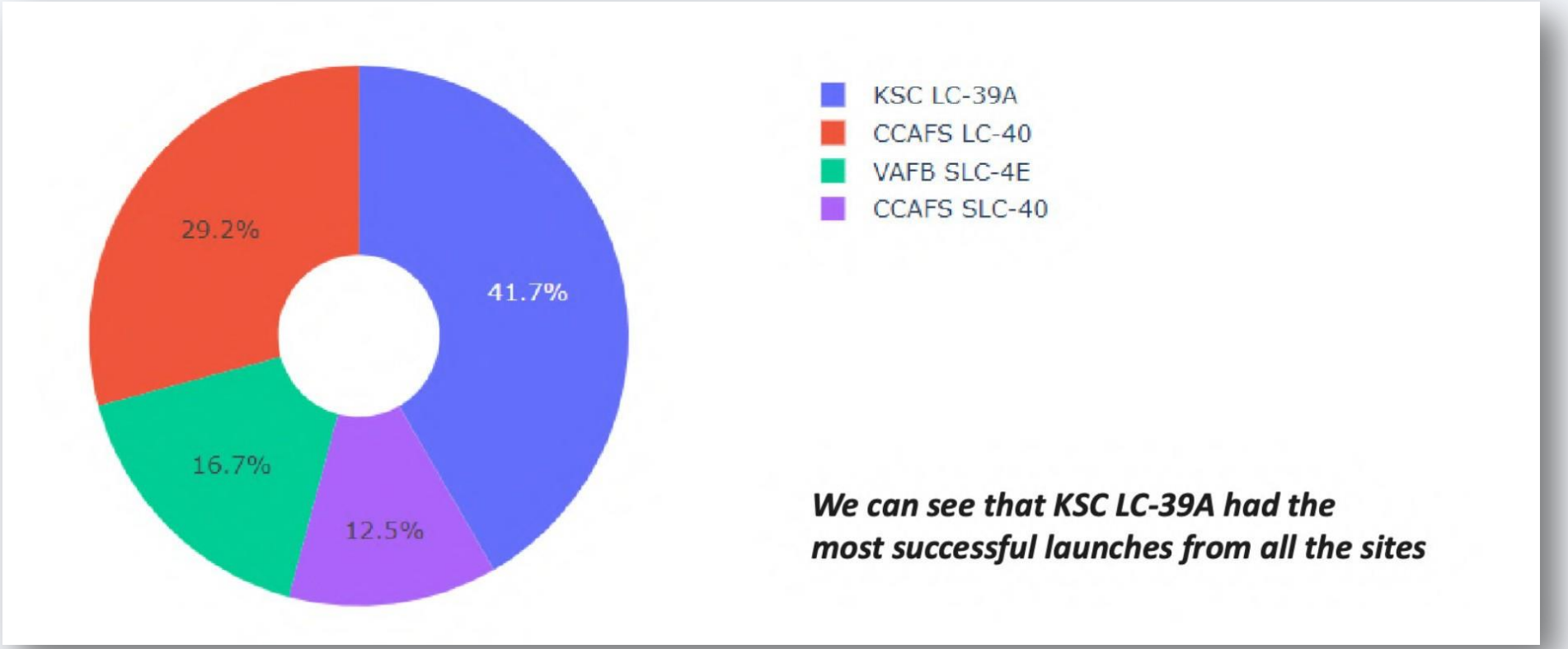




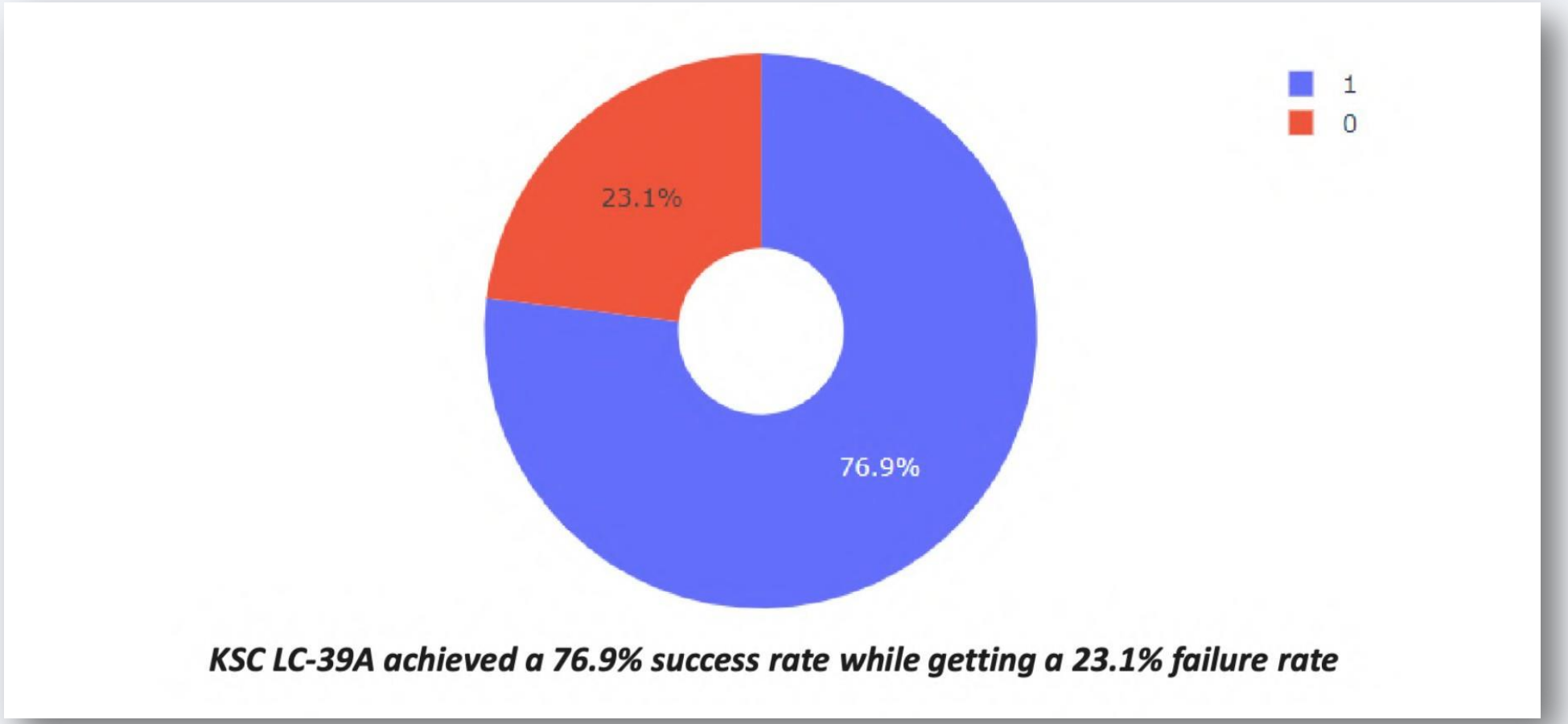
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

The success percentage by each sites.



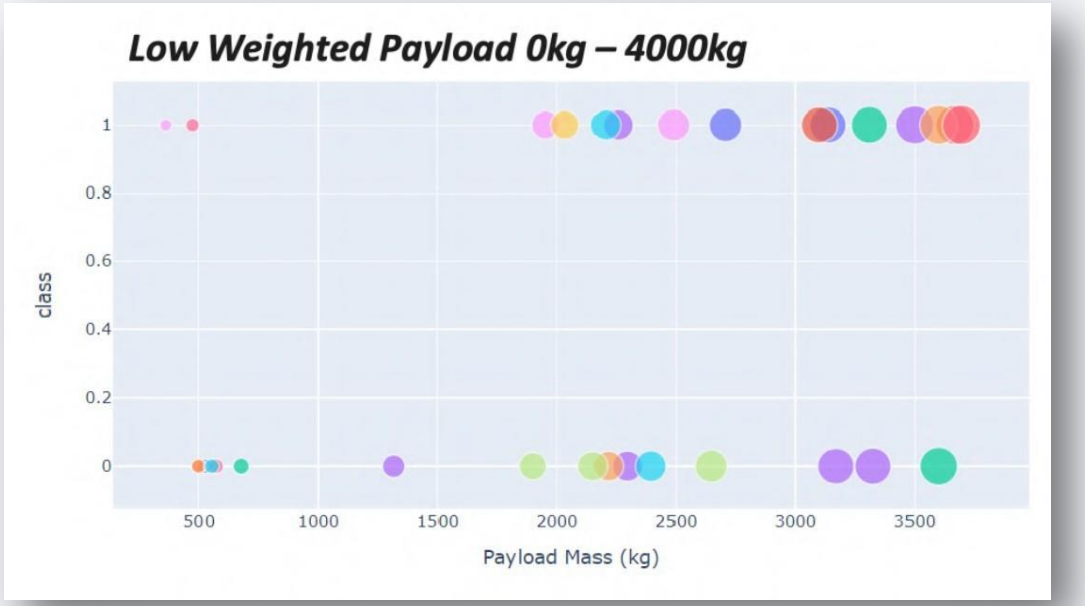
The highest launch-success ratio: KSC LC-39A

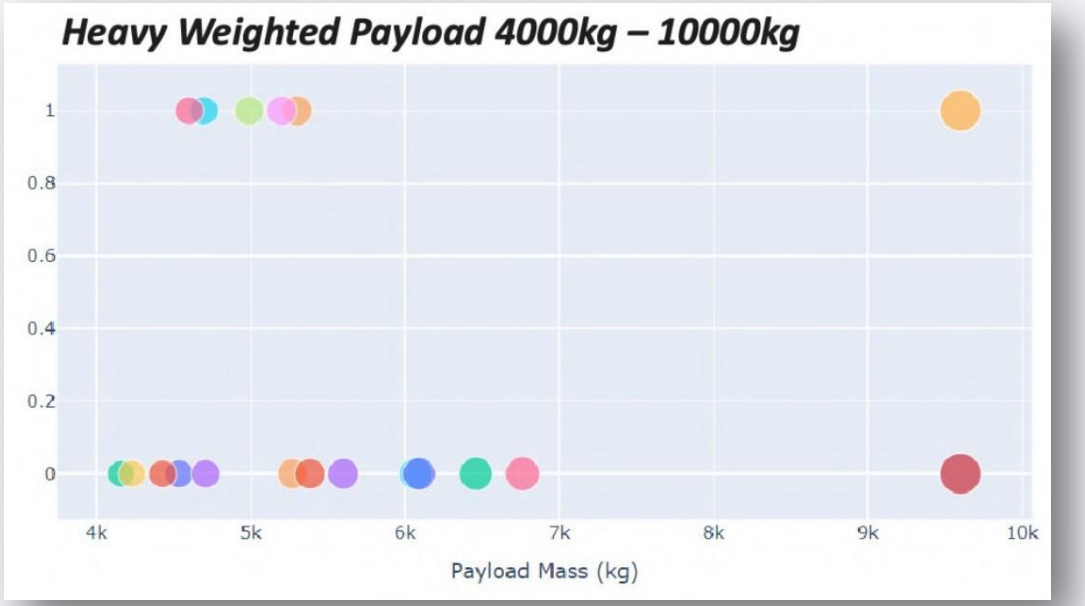


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch Outcome Scatter Plot

It can be seen that the success rate for lightweight payloads is higher than that for heavy payloads.





Section 5

Predictive Analysis (Classification)

Classification Accuracy

As you can see from the code below, we were able to identify the tree algorithm with the highest classification accuracy as the best algorithm.

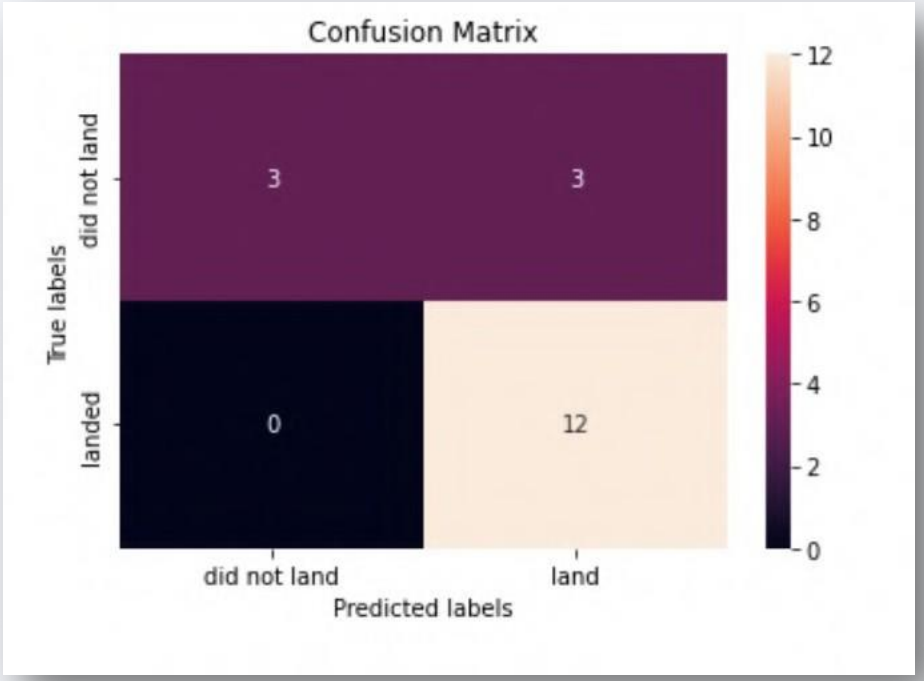
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params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

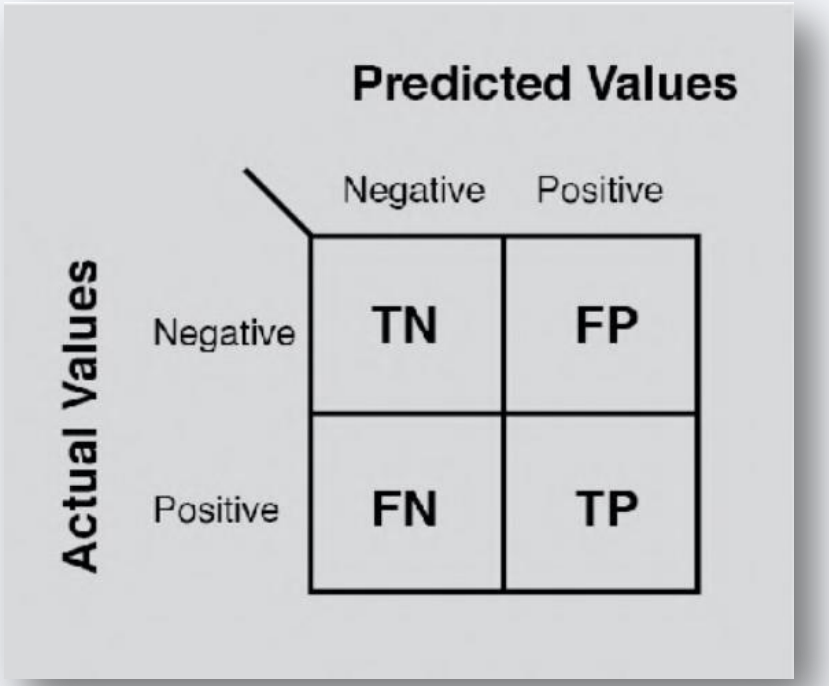
```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

Confusion Matrix

The confusion matrix of the decision tree classifier shows that the classifier can distinguish between different classes. The main problem is false alarms

i.e. unsuccessful landings marked as successful landings by the classifier.





Conclusions

Can we conclude that:

- A tree classification algorithm is the best machine learning approach for this dataset.
- Light payloads (defined below 4000 kg) performed better than heavy payloads.
- From 2013, SpaceX's launch success rate will increase in direct proportion until his 2020, when future launches are finally completed.
- KSC LC-39A has the highest launch success rate of any site. 76.9%
- SSO Orbit has the highest success rate. 100% and 1 or more occurrences.

Thank you!