IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Augusto Gontijo
May 2022

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- With the success of SpaceX, a new player appeared in the market (SpaceY) seeking to compete for market share.
- After a complete data extraction using API and web scraping, the final models were able to predict the first stage rocket booster landing successfully with an accuracy level of **83.3%**

Introduction: Context

- SpaceX has gained worldwide attention for a series of historic milestones. It is the only private company ever to return a spacecraft from low-earth orbit, which it first accomplished in December 2010. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars wheras other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage.

- Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Introduction: The Problem

If we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

This project aims to accurately predict if the first stage rocket will successfully land, as a proxy for the cost of a launch

Section 1

Methodology

Methodology

The methodology adopted for this project can be outlined as such:

- 1. Data collection
- 2. Data wrangling
- 3. Exploratory data analysis
- 4. Data visualization
- 5. Model development
- 6. Reporting results to stakeholders

Data Collection

For data collection, two different methods were used:

API

Acquired historical launch data from Open Source REST API for SpaceX

- Requested and parsed the SpaceX launch data using the GET request
- Filtered the dataframe to only include Falcon 9 launches
- Replaced missing payload mass values from classified missions with mean

Web Scraping

Acquired historical launch data from Wikipedia page 'List of Falcon 9 and Falcon

Heavy Launches'

- Requested the Falcon9 Launch Wiki page from its Wikipedia URL
- Extracted all column/variable names from the HTML table header
- Parsed the table and converted it into a Pandas data frame

Data Wrangling

Explored data to determine the label for training supervised models.

Calculations:

- number of launches on each site
- number and occurrence of each orbit
- number and occurrence of mission outcome per orbit type

Created a landing outcome training label from 'Outcome' column

Training label: 'Class'

Class = 0; first stage booster did not land successfully

- None None; not attempted
- None ASDS; unable to be attempted due to launch failure
- False ASDS; drone ship landing failed
- False Ocean; ocean landing failed
- False RTLS; ground pad landing failed

Class = 1; first stage booster landed successfully

- True ASDS; drone ship landing succeeded
- True RTLS; ground pad landing succeeded
- True Ocean; ocean landing succeeded

EDA with SQL

Task 3

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

In [11]:
```
ps.sqldf("SELECT SUM(PAYLOAD_MASS__KG_) AS 'total_payload_Nasa' \
FROM SPACEXTBL \
WHERE Customer = 'NASA (CRS)';")
```

Out[11]:

	total_payload_Nasa
0	45596

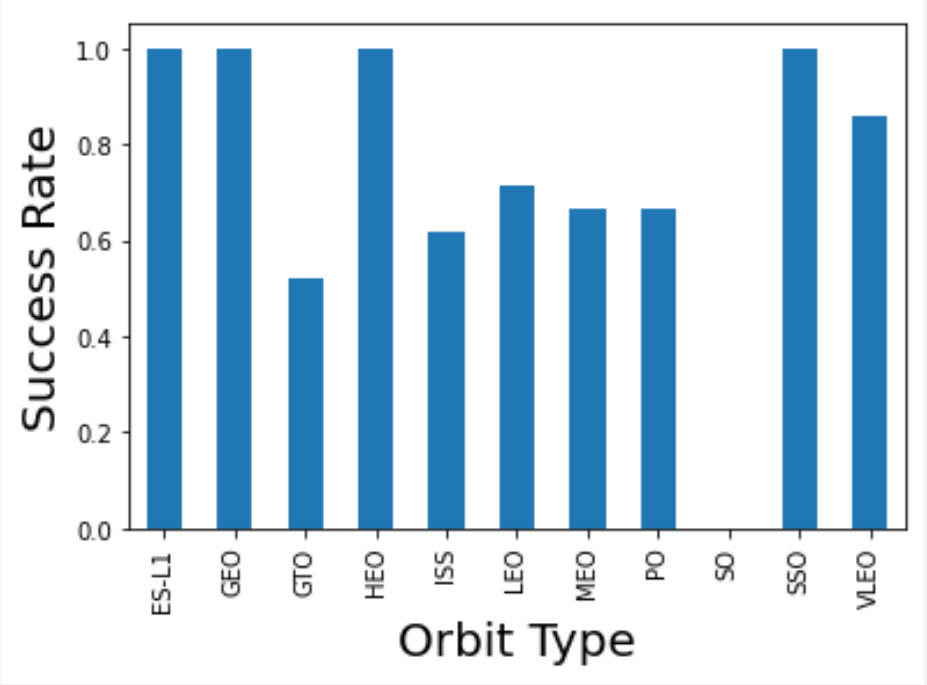
EDA performed using SQL:

- Ran SQL queries to display and list information about
- Launch sites
- Payload masses
- Booster versions
- Mission outcomes
- Booster landings

EDA with Data Visualization

EDA performed using Data Visualization:

- Read the dataset into a Pandas dataframe
- Used Matplotlib and Seaborn visualization libraries to plot
- FlightNumber x PayloadMass
- FlightNumber x LaunchSite
- Payload x LaunchSite
- Orbit type x Success rate
- FlightNumber x Orbit type
- Payload x Orbit type
- Year x Success rate



Build an Interactive Map with Folium

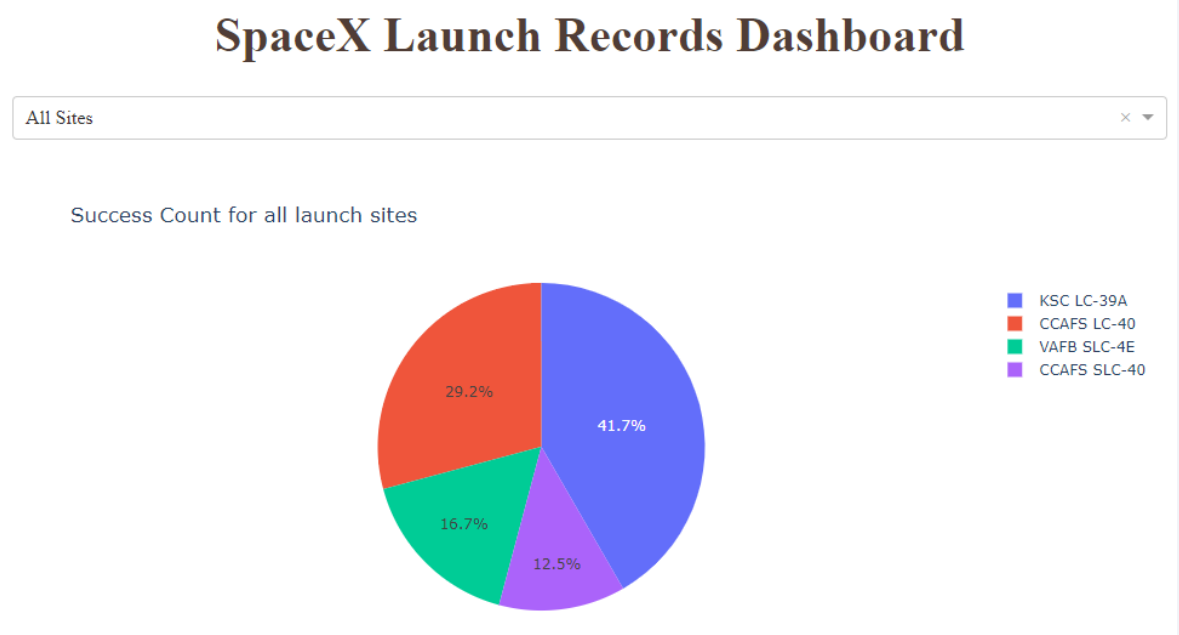
Launch Sites Location Analysis

- Used Python interactive mapping library called Folium
- Marked all launch sites on a map
- Marked the successful/failed launches for each site on map
- Calculated the distances between a launch site to its proximities
- Railways
- Highways
- Coastlines
- Cities

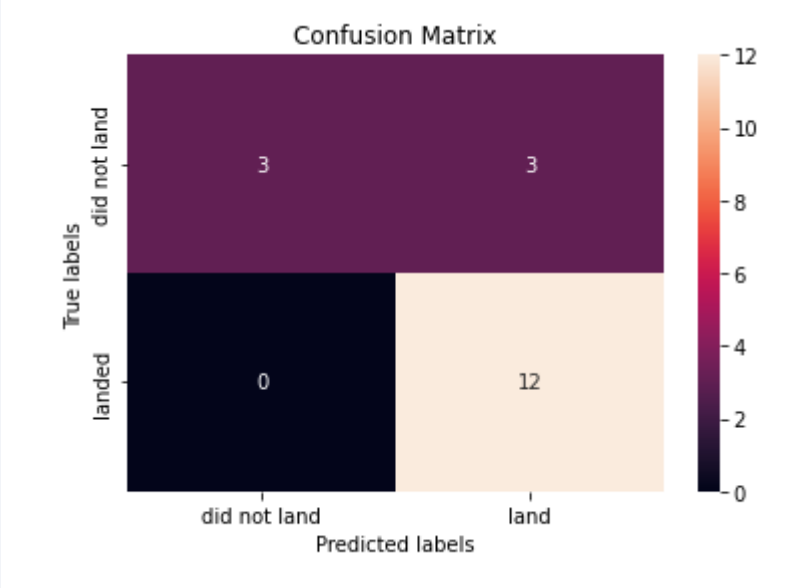
Build a Dashboard with Plotly Dash

Launch Records Dashboard

- Used Python interactive dashboarding library called Plotly Dash to enable stakeholders to explore and manipulate data in real-time
- Pie chart showing success rate
- Color coded by launch site
- Scatter chart showing payload mass vs. landing outcome
- Color coded by booster version
- With range slider for limiting payload amount
- Drop-down menu to choose between all sites and individual launch sites



Predictive Analysis (Classification)

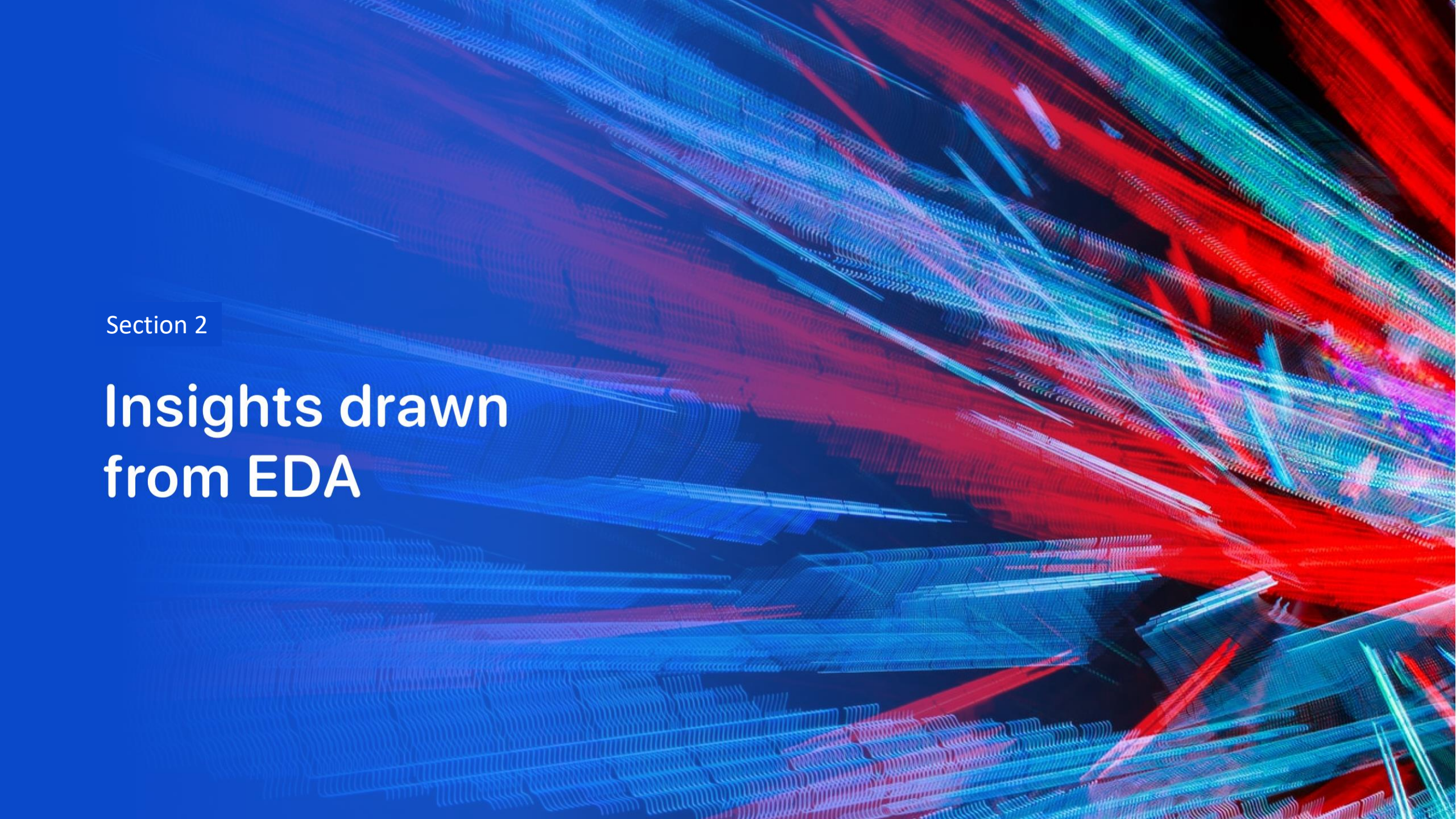


Model development

- Imported libraries and defined function to create confusion matrix (Pandas, Numpy, Matplotlib, Seaborn, Sklearn)
- Loaded the dataframe created during data collection
- Created a column for our training label 'Class' created during data wrangling
- Standardized the data
- Split the data into training data and test data
- Fit the training data to various model types (Logistic Regression, Support Vector Machine, Decision Tree Classifier, K Nearest Neighbors Classifier)
- Used a cross-validated grid-search over a variety of hyperparameters to select the best ones for each model
- Enabled by Scikit-learn library function GridSearchCV
- Evaluated accuracy of each model using test data to select the best model

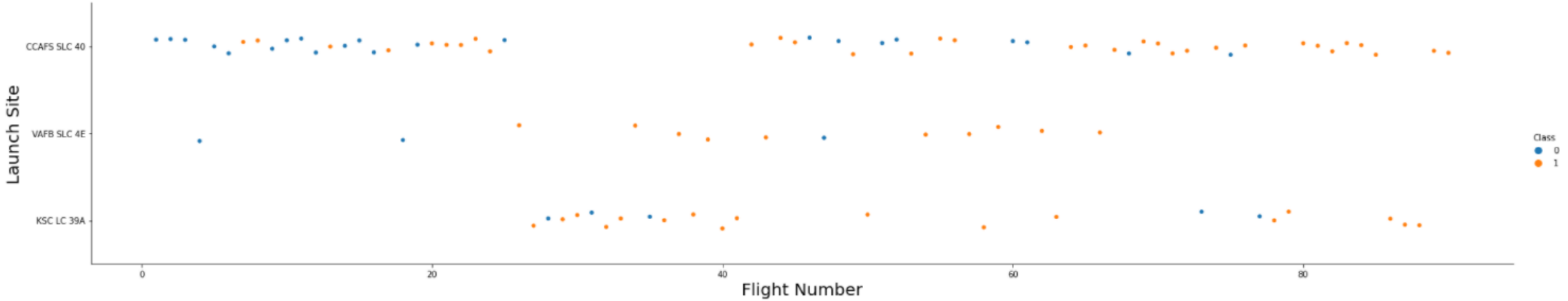
Results

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


Section 2
Insights drawn from EDA

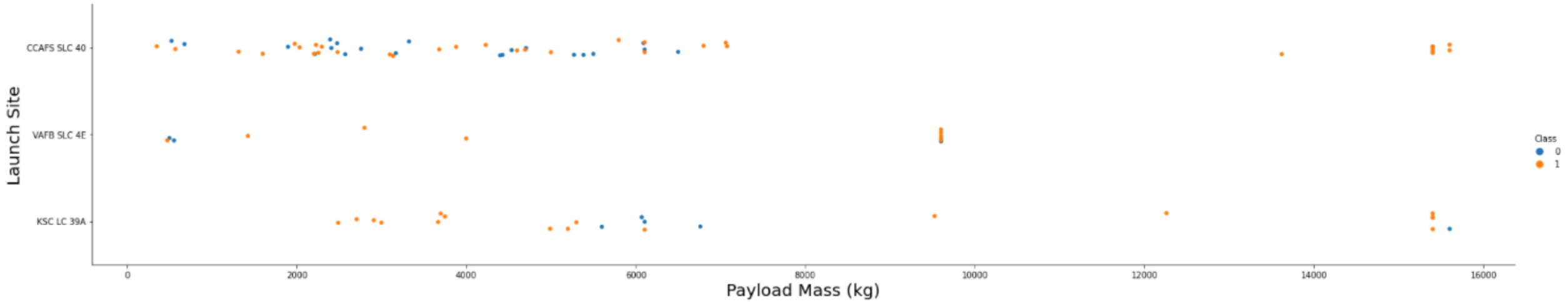
Flight Number vs. Launch Site

In [4]:
```
# 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 value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

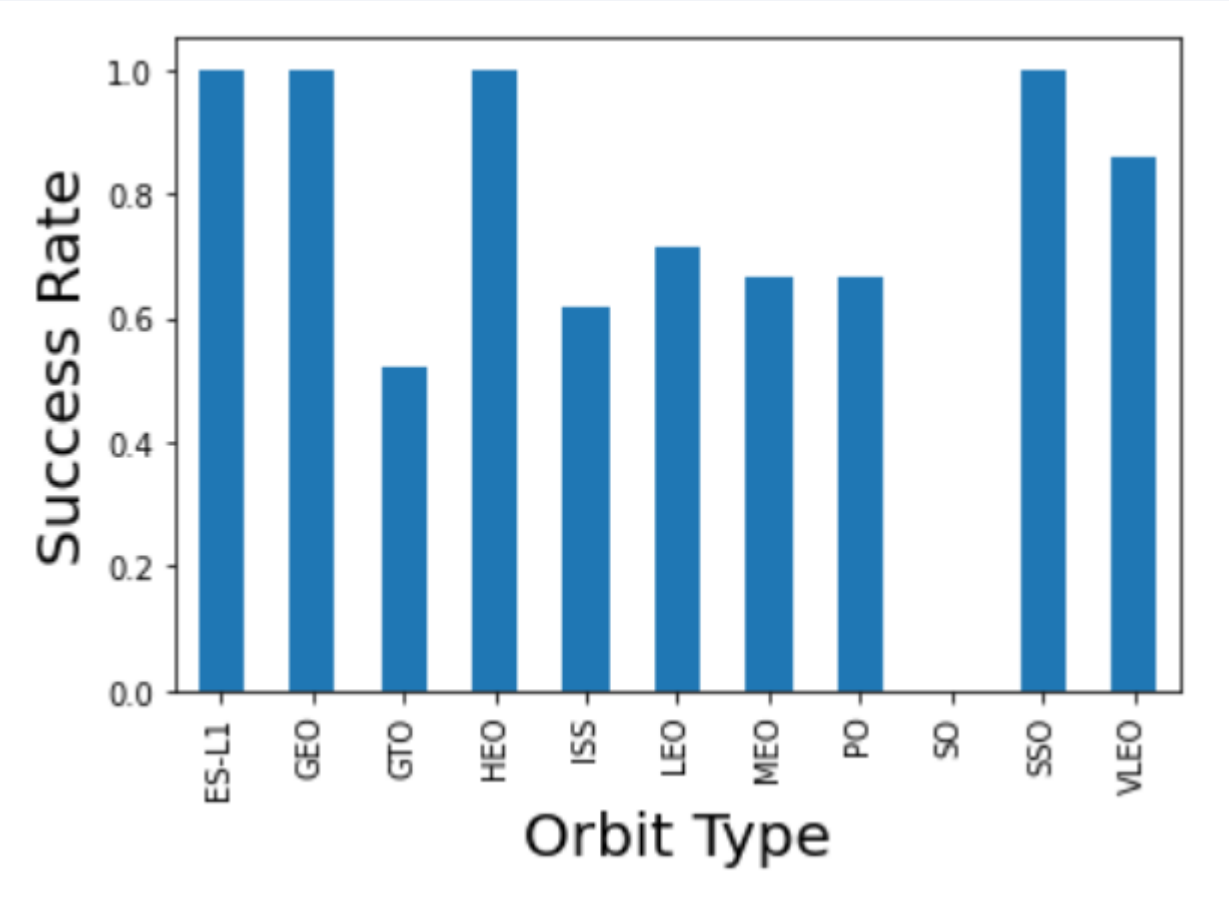


Payload vs. Launch Site

```
# 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 class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

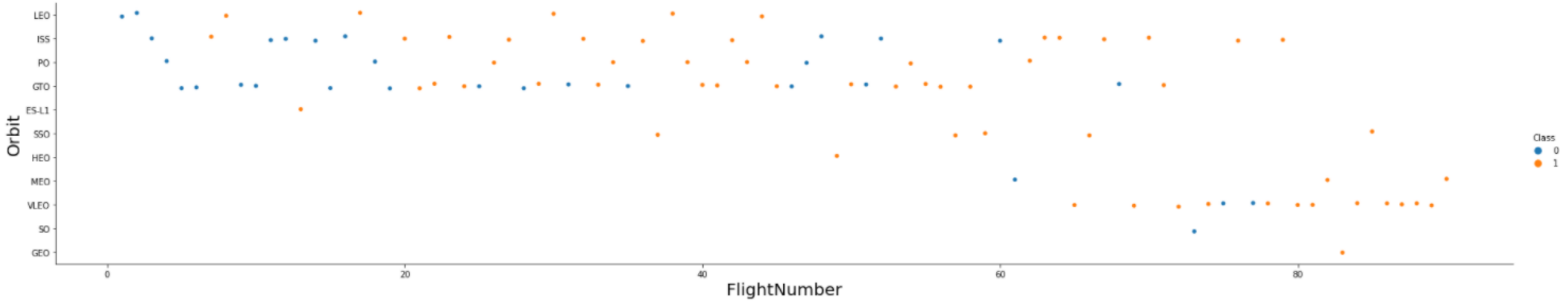


Success Rate vs. Orbit Type



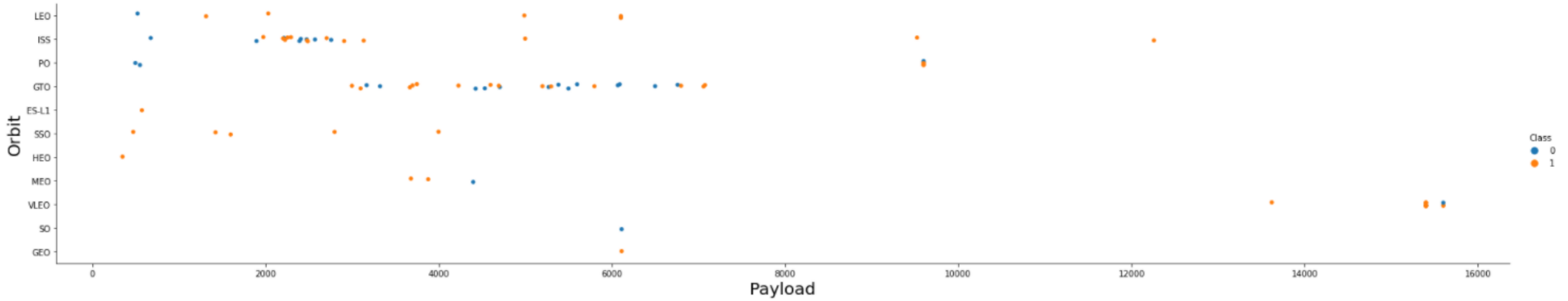
Flight Number vs. Orbit Type

```
# 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.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Payload vs. Orbit Type

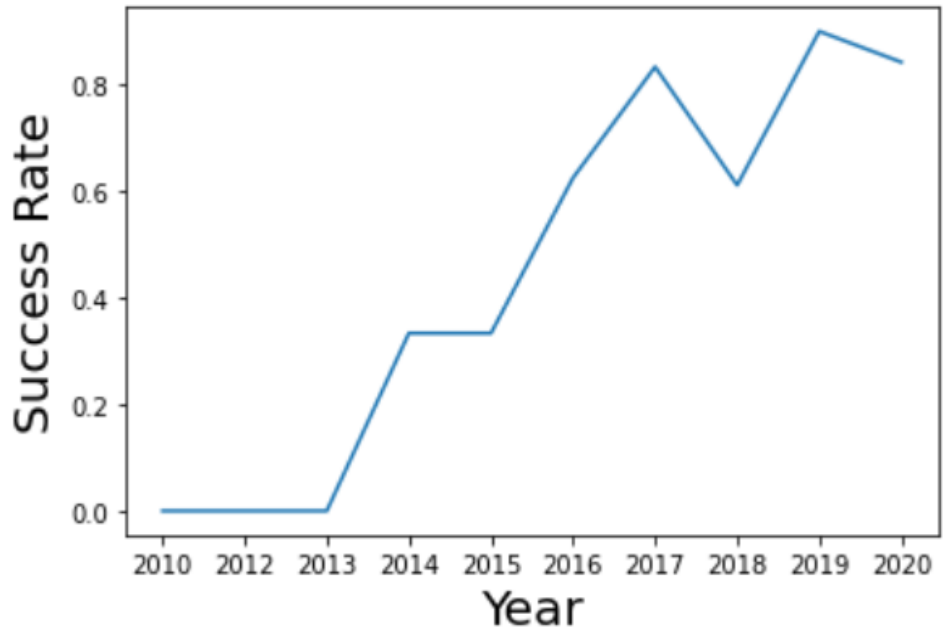
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df1=pd.DataFrame(Extract_year(df['Date']),columns =['year'])
df1['Class']=df['Class']
df1 = df1.groupby(by = 'year', as_index= False).mean()

sns.lineplot(data=df1, x='year', y='Class')
plt.xlabel("Year", fontsize=20)
plt.ylabel("Success Rate", fontsize=20)
plt.show()
```



All Launch Site Names

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

```
ps.sqldf("SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL")
```

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

Launch Site Names Begin with 'CCA'

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

```
ps.sqldf("SELECT LAUNCH_SITE \
FROM SPACEXTBL \
WHERE LAUNCH_SITE LIKE 'CCA%' \
LIMIT 5;")
```

	Launch_Site
0	CCAFS LC-40
1	CCAFS LC-40
2	CCAFS LC-40
3	CCAFS LC-40
4	CCAFS LC-40

Total Payload Mass

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

```
ps.sqldf("SELECT SUM(PAYLOAD_MASS__KG_) AS 'total_payload_Nasa' \
FROM SPACEXTBL \
WHERE Customer = 'NASA (CRS)';")
```

	total_payload_Nasa
0	45596

Average Payload Mass by F9 v1.1

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

```
ps.sqldf("SELECT AVG(PAYLOAD_MASS__KG_) AS 'avg_payload_f9v1.1' \
FROM SPACEXTBL \
WHERE Booster_Version LIKE 'F9 v1.1';")
```

	avg_payload_f9v1.1
0	2928.4

First Successful Ground Landing Date

List the date when the first successful landing outcome in ground pad was acheived.

Hint:Use min function

```
ps.sqldf("SELECT MIN(Date) FROM SPACEXTBL WHERE [Landing _Outcome] = 'Success (ground pad)';")
```

	MIN(Date)
0	01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
ps.sqldf("SELECT BOOSTER_VERSION \
FROM SPACEXTBL \
WHERE [LANDING _OUTCOME] = 'Success (drone ship)' \
AND 4000 < PAYLOAD_MASS__KG_ < 6000;")
```

	Booster_Version
0	F9 FT B1021.1
1	F9 FT B1022
2	F9 FT B1023.1
3	F9 FT B1026
4	F9 FT B1029.1
5	F9 FT B1021.2
6	F9 FT B1029.2
7	F9 FT B1036.1
8	F9 FT B1038.1
9	F9 B4 B1041.1
10	F9 FT B1031.2
11	F9 B4 B1042.1
12	F9 B4 B1045.1
13	F9 B5 B1046.1

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
ps.sqldf("SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS 'Total' \
FROM SPACEXTBL \
GROUP BY MISSION_OUTCOME;")
```

	Mission_Outcome	Total
0	Failure (in flight)	1
1	Success	98
2	Success	1
3	Success (payload status unclear)	1

Boosters Carried Maximum Payload

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

```
ps.sqldf("SELECT DISTINCT BOOSTER_VERSION \
FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ = ( \
    SELECT MAX(PAYLOAD_MASS__KG_) \
    FROM SPACEXTBL);")
```

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

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

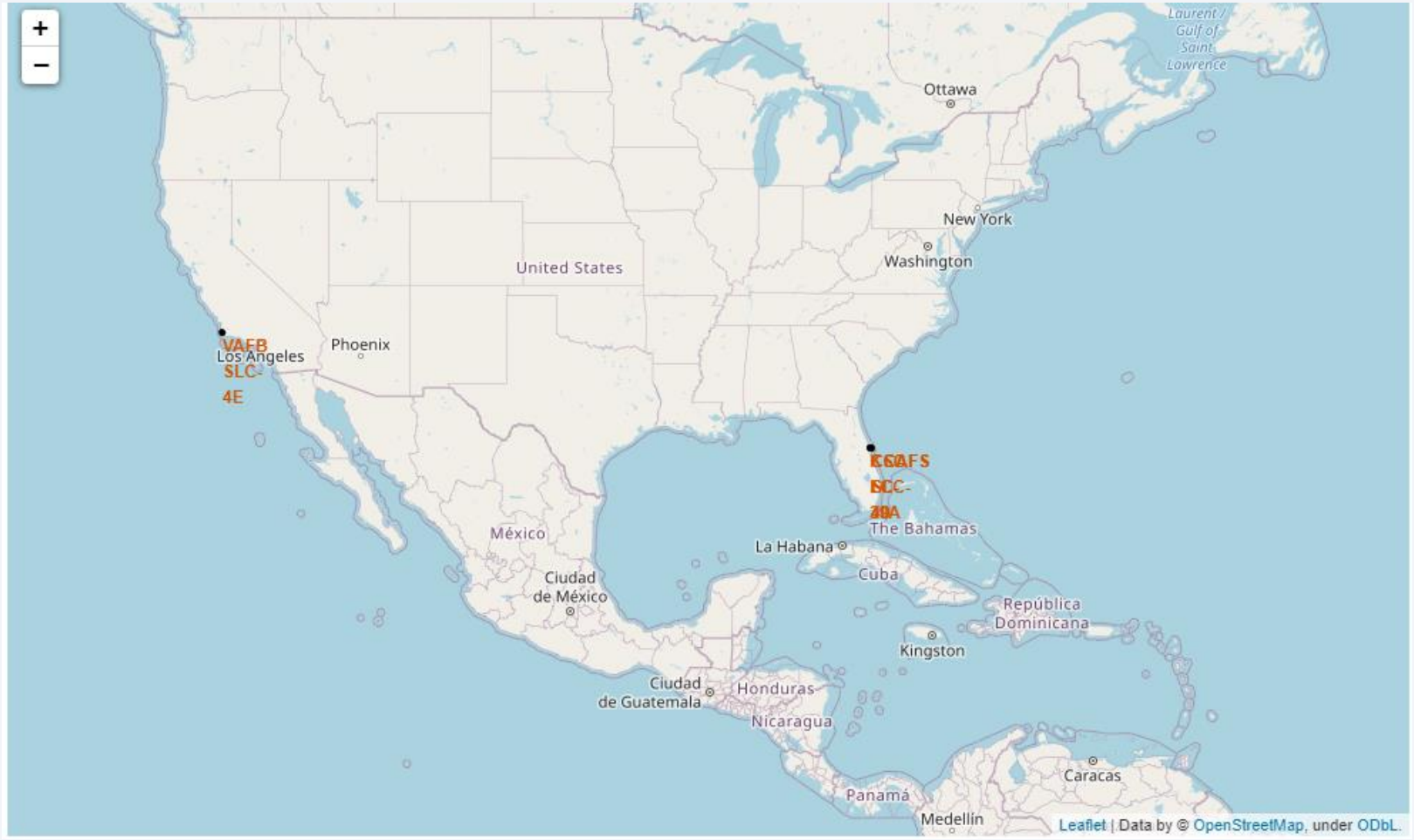
```
ps.sqldf("SELECT [LANDING _OUTCOME], COUNT([LANDING _OUTCOME]) AS TOTAL_NUMBER \
FROM SPACEXTBL \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY [LANDING _OUTCOME] \
ORDER BY TOTAL_NUMBER DESC;")
```

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

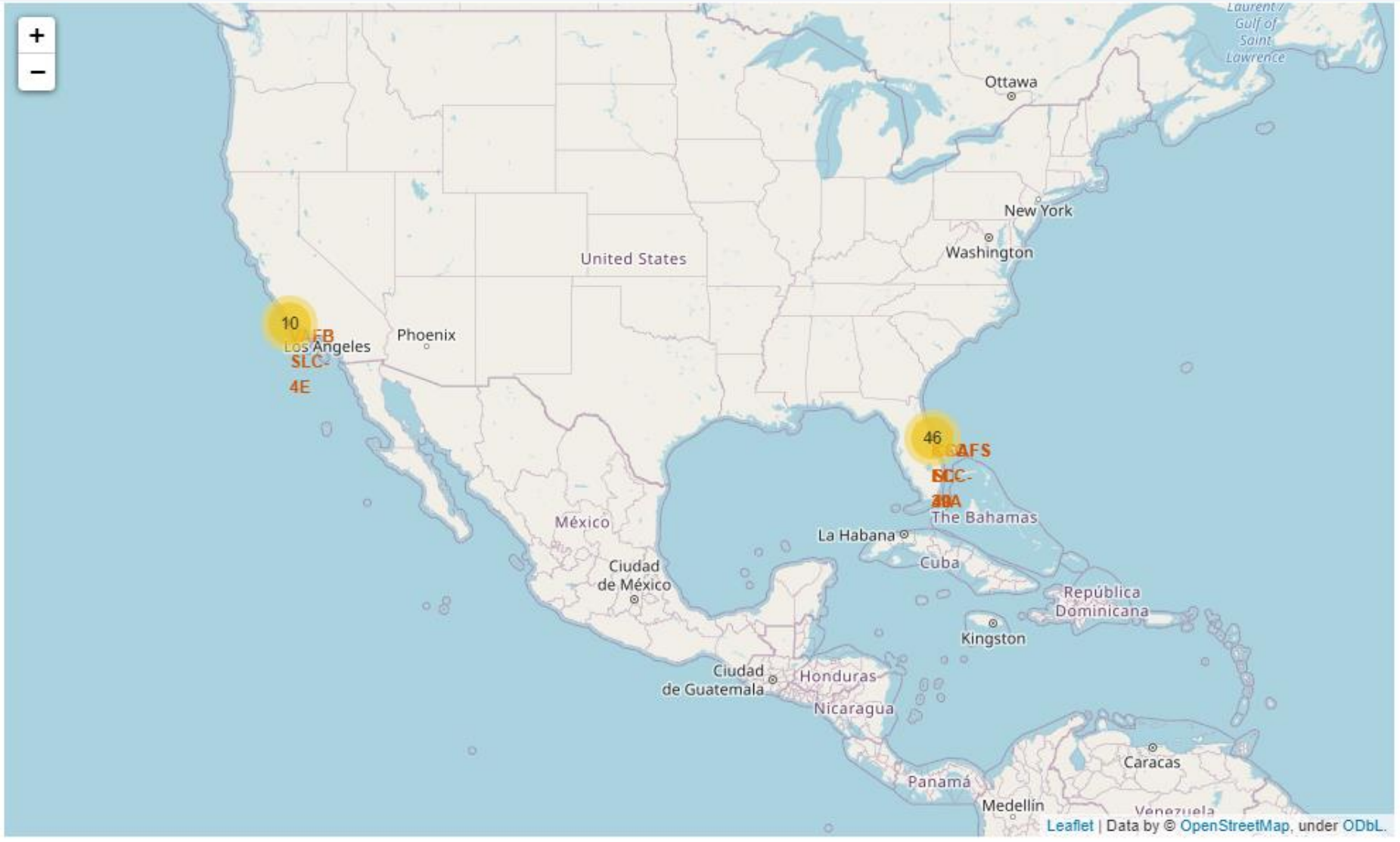
Section 3

Launch Sites Proximities Analysis

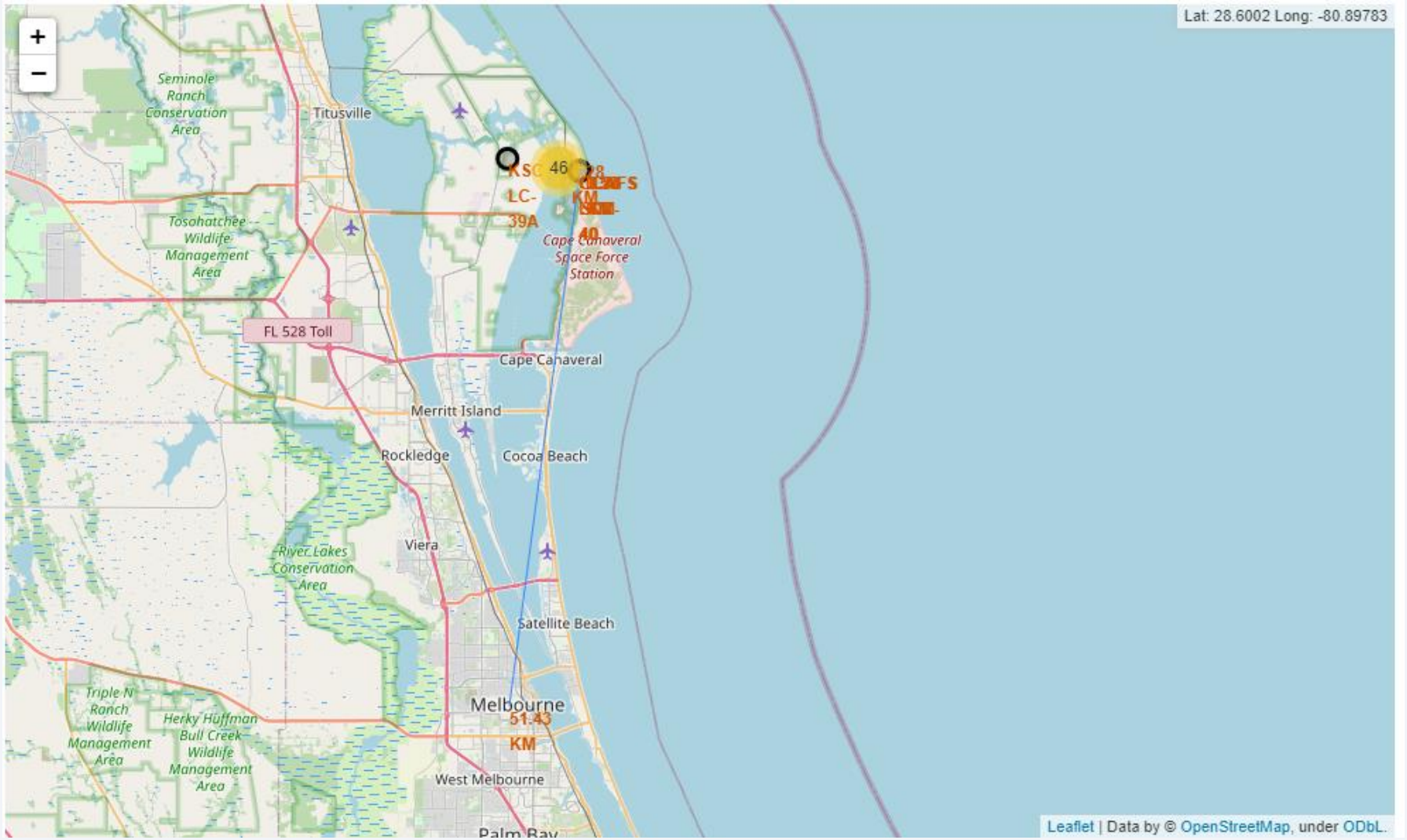
Launch site locations

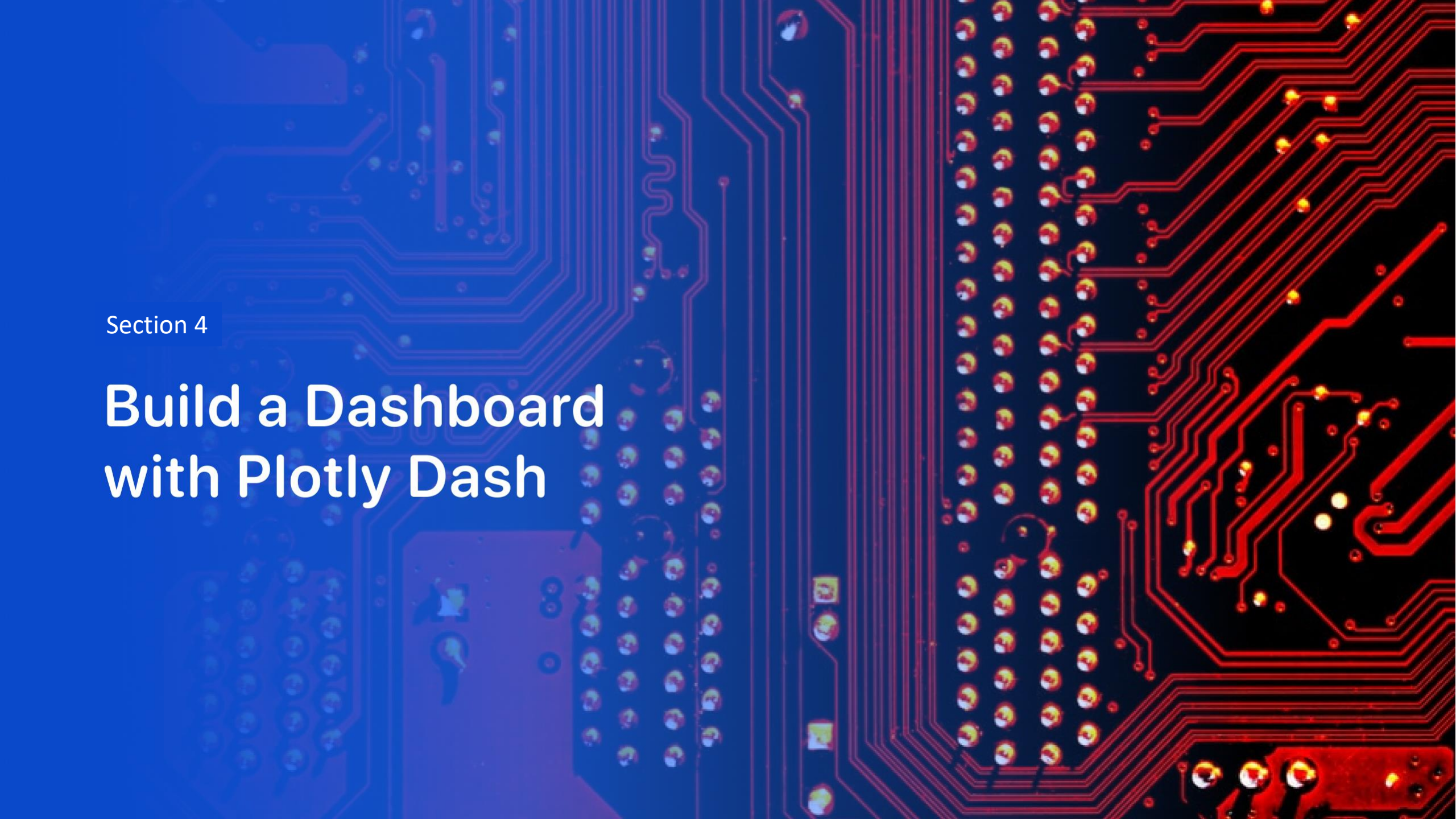


Launch Outcomes

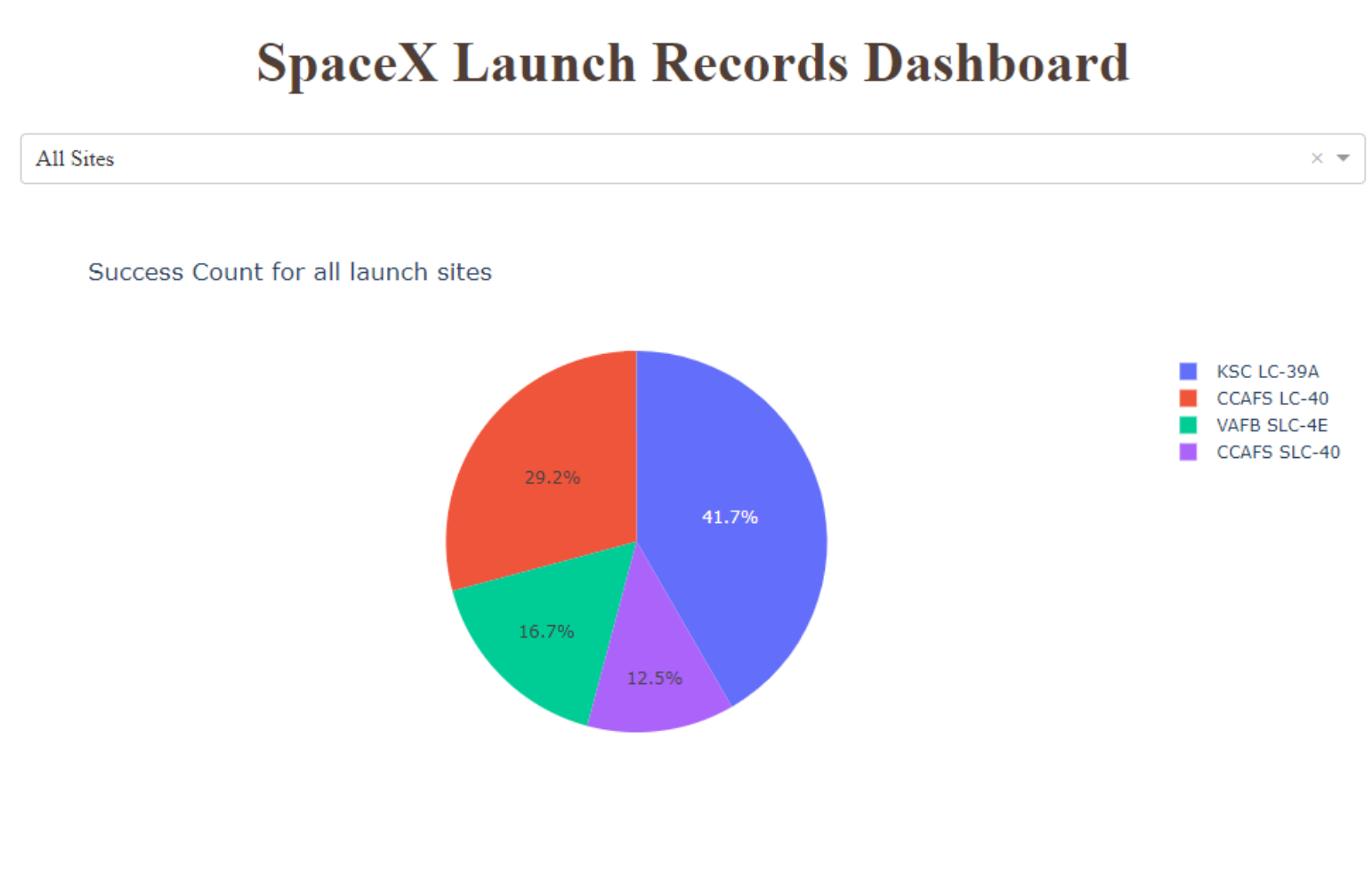


Launch site proximities

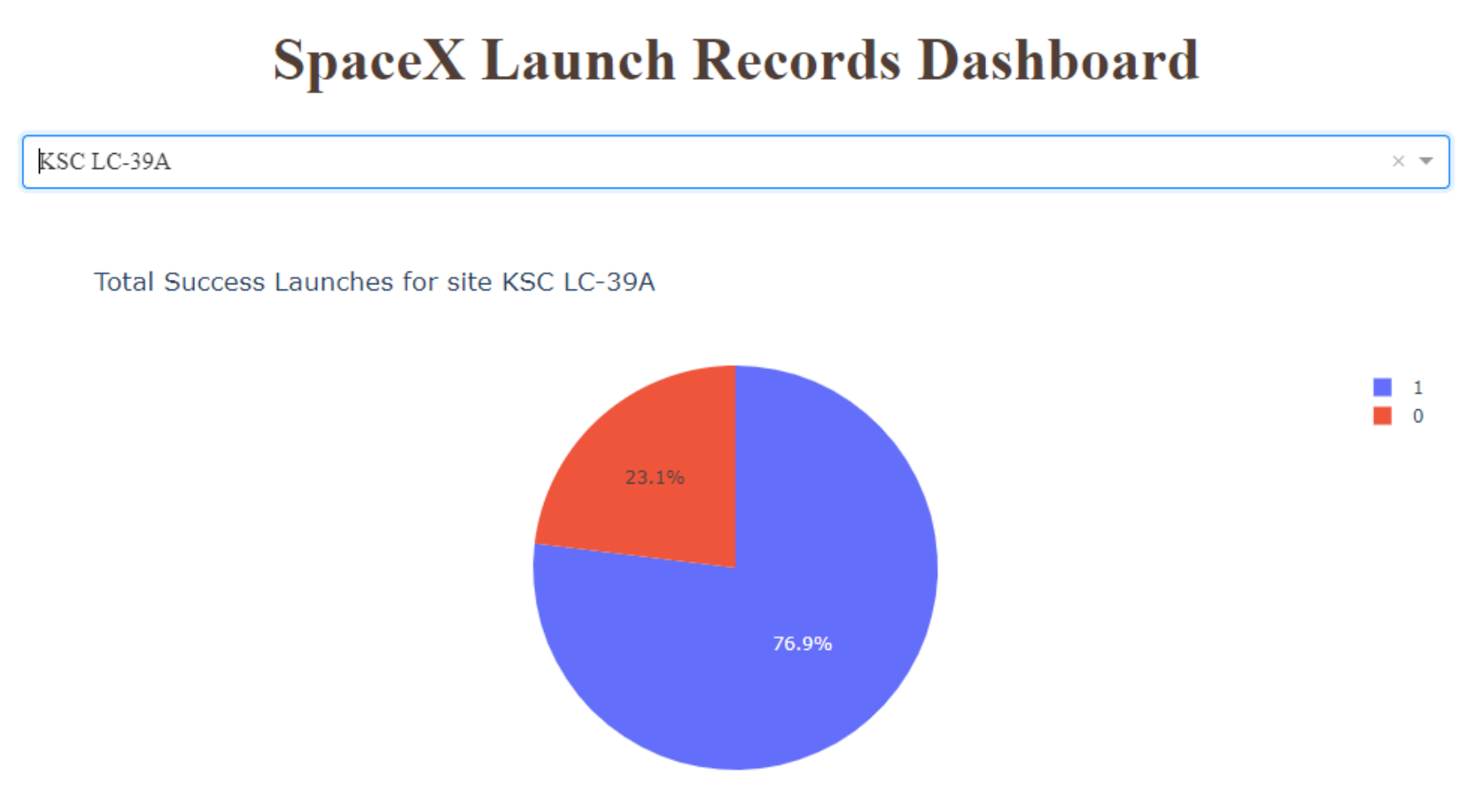



Section 4
Build a Dashboard with Plotly Dash

Launch success rate per site

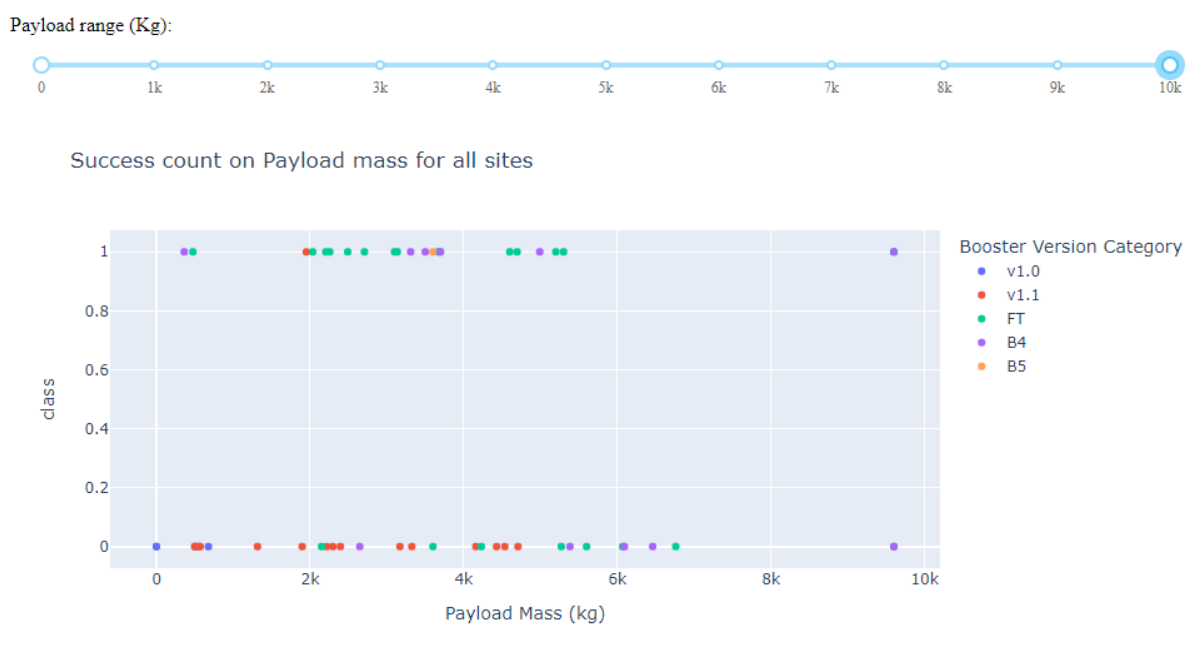


Success ratio of KSC LC-39A (highest success ratio)

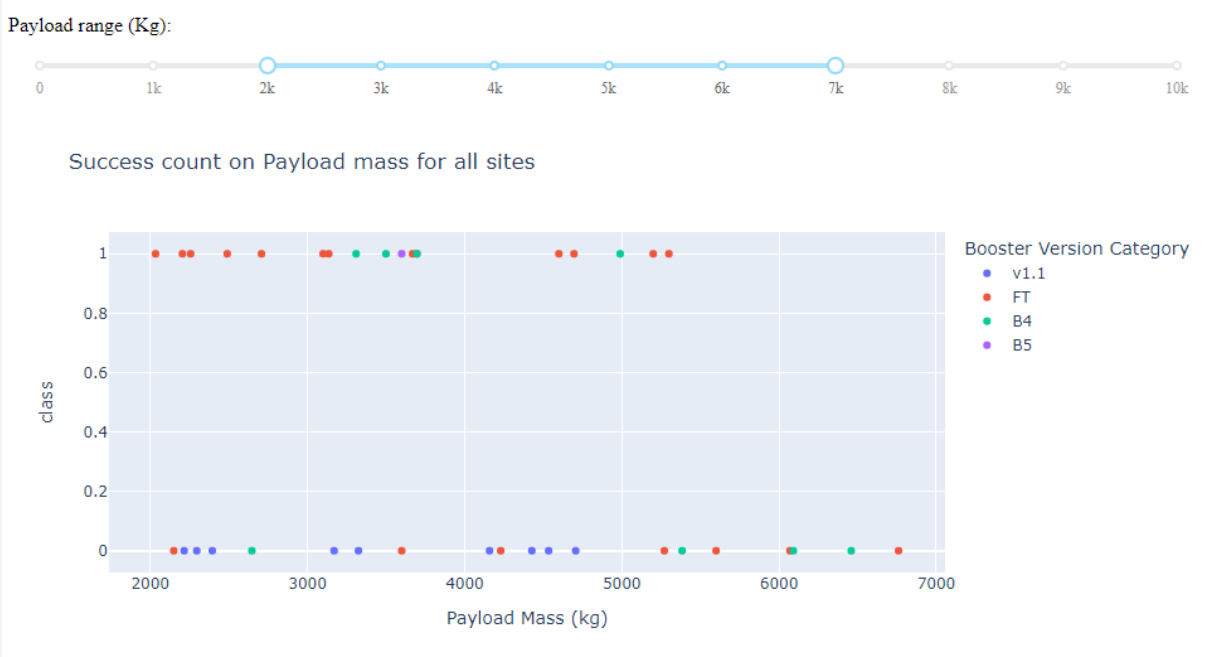


Payload vs. Launch Outcome

Full payload range



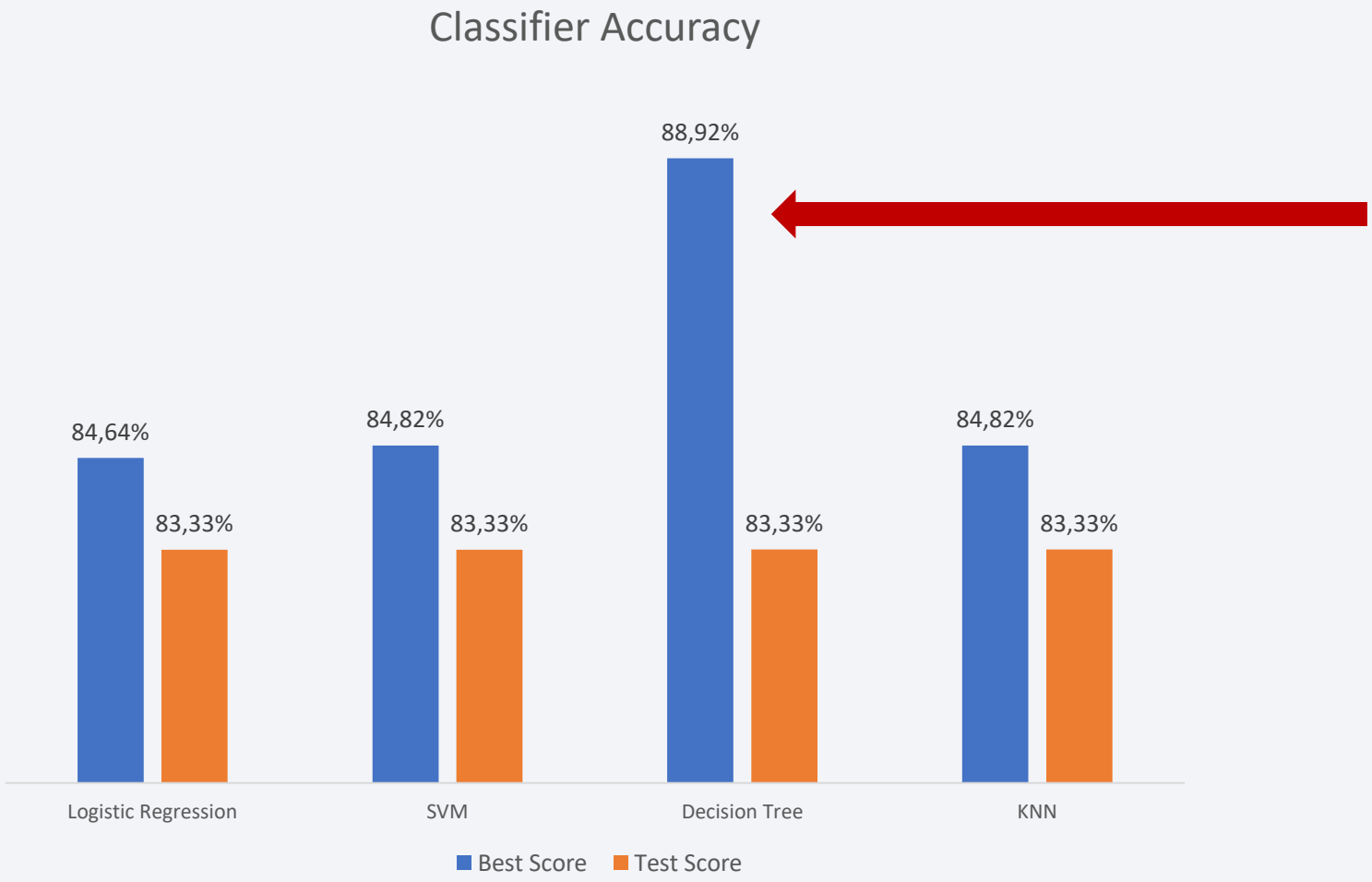
Payload range between 2k and 7k



Section 5

Predictive Analysis (Classification)

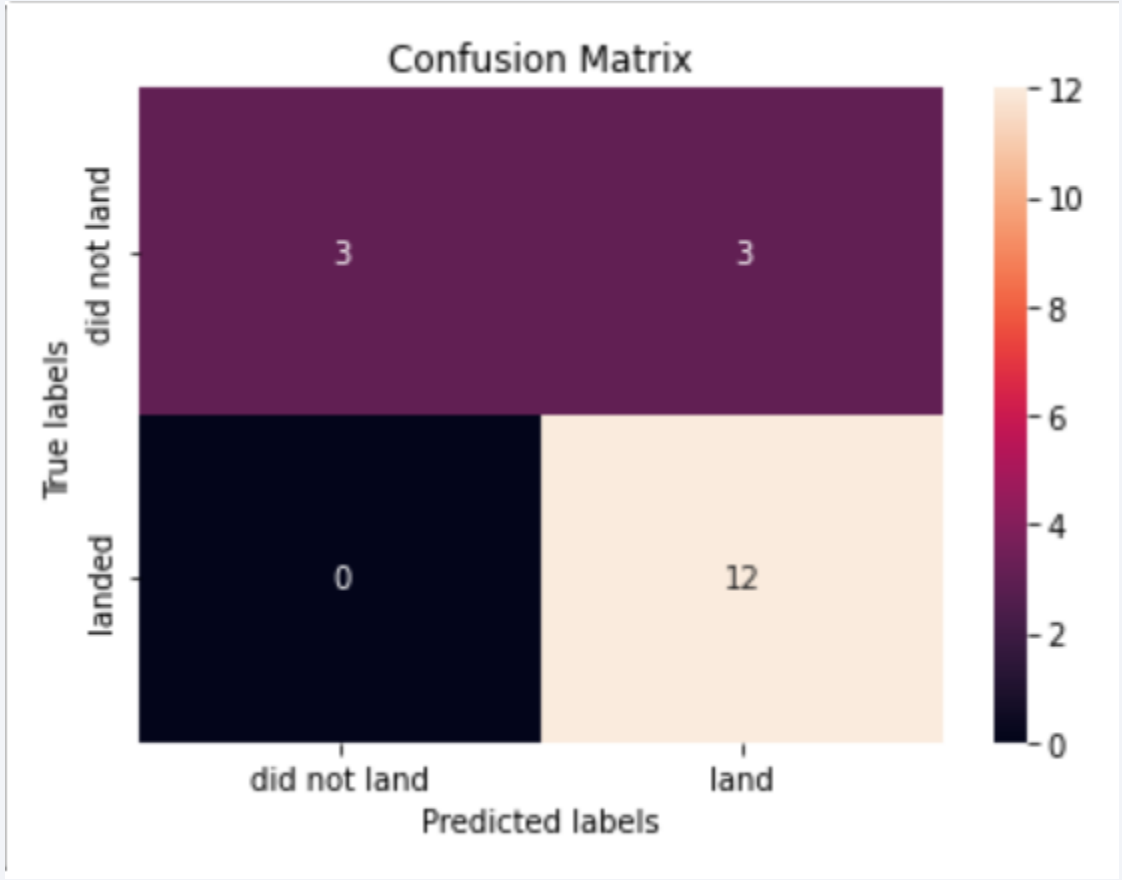
Classification Accuracy



Decision Tree delivered the best score.

However, on the test dataset, all classifiers delivered the same accuracy, of 83,33%

Confusion Matrix



When the outcome is positive (“landed”) the model can precisely predict it: 12/12 (100%).

However, when the outcome is negative (“did not land”), the model don’t perform so well: 3/6 (50%).

Final Thoughts

Some insights from this project:

- Using the models from this project, the company can predict with a booster will successfully land with 83,3% accuracy.
- The CCAFS-SLC40 launch site has the lowest success rate, should the company keep using it?
- There are no successful record of launches with payload between 5500k and 9000k

Thank you!