IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ramón H
22/05/2025

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data collection
  - Data wrangling
  - Exploratory Data Analysis with Data Visualization
  - Exploratory Data Analysis with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- **Summary of all results**
  - Exploratory Data Analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

Project background and context

SpaceX is the most successful company of the commercial space age, making space travel affordable. The company advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the irst stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. Based on public information and machine learning models, we are going to predict if SpaceX will reuse the first stage.

Questions to be answered

- How do variables such as payload mass, launch site, number of lights, and orbits affect the success of the first stage landing?

- Does the rate of successful landings increase over the years?

- What is the best algorithm that can be used for binary classification in this case?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Using SpaceX Rest API
  - Using Web Scrapping from Wikipedia
- Perform data wrangling
  - Filtering the data
  - Dealing with missing values
  - Using One Hot Encoding to prepare the data to a binary classification
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Building, tuning and evaluation of classification models to ensure the best results

Data Collection

Data collection process involved a combination of API requests from SpaceX REST API and Web Scraping data from a table in SpaceX's Wikipedia entry.

We had to use both of these data collection methods in order to get complete information about the launches for a more detailed analysis.

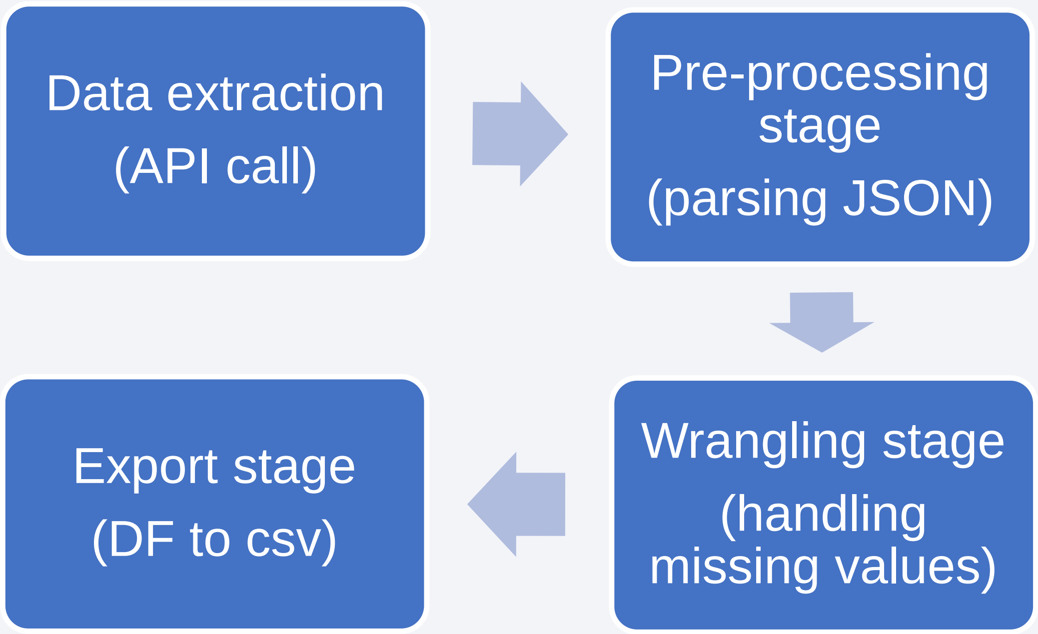
Data Columns are obtained by using SpaceX REST API:

FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude

Data Columns are obtained by using Wikipedia Web Scraping:

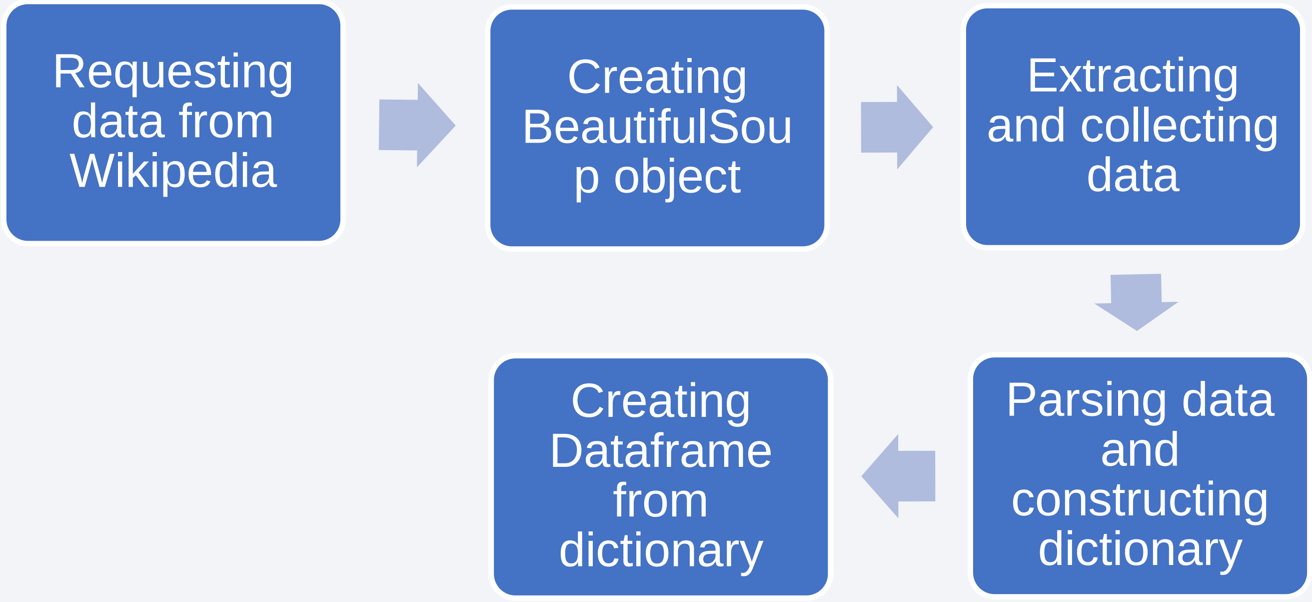
Flight No., Launch site, Payload, PayloadMass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time

Data Collection – SpaceX API



https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/01-jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping



https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/02-jupyter-labs-webscraping.ipynb

Data Wrangling

The dataset contains various cases where booster landings were unsuccessful. In some situations, a landing attempt occurred but failed due to an accident.

For instance, a "True Ocean" record indicates that the mission successfully landed in a designated ocean region, whereas a "False Ocean" record signifies an unsuccessful ocean landing.

Similarly, "True RTLS" denotes a successful landing on a ground pad, while "False RTLS" indicates a failed attempt.

The same applies to drone ship landings, where "True ASDS" represents a successful touchdown on the drone ship, and "False ASDS" indicates otherwise.

Ultimately, these outcomes are converted into training labels: a value of "1" means the booster landed successfully, and a "0" signifies an unsuccessful landing.

https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/03-labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

The plots and charts (using Matplotlib and Seaborn) are used to understand more about the relationships between several features, such as:

- The relationship between flight number and launch site
- The relationship between payload mass and launch site
- The relationship between success rate and orbit type

https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/05-edadataviz.ipynb

EDA with SQL

SQL queries are employed to extract key insights from the dataset, including:

- Identifying the unique names of the launch sites featured in the space mission
- Calculating the total payload mass carried by boosters launched by NASA (CRS)
- Determining the average payload mass for booster version F9 v1.1

https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/04-jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

The Folium library is used to:

- Mark all launch sites on a map
- Mark the succeeded launches and failed launches for each site on the map
- Mark the distances between a launch site to its proximities such as the nearest city and airport

https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/06-lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

Dash functions are employed to build an interactive website that lets users control inputs via a dropdown menu and a range slider. The site features both a pie chart and a scatterplot to illustrate:

- The total number of successful launches from each launch site
- The relationship between payload mass and mission outcome (success or failure) for every launch site

https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/spacex-dash-app.py

Predictive Analysis (Classification)

Functions from the Scikit-learn library form the backbone of our machine learning models. The prediction phase follows these key steps:

- Data Standardization: Normalize input features to maintain consistency.
- Data Splitting: Divide the dataset into training and test sets.
- Model Creation: Build various machine learning models, including:
  - Logistic Regression
  - Support Vector Machine (SVM)
  - Decision Tree
  - K-Nearest Neighbors (KNN)
- Model Fitting: Train each model using the training data.
- Hyperparameter Tuning: Determine the optimal hyperparameter combination for each model.
- Evaluation: Assess model performance through accuracy scores and confusion matrices.

https://github.com/rhp-arduino/IBM-DataScience-Capstone/blob/main/07-SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

Broadly speaking, the exploratory data analysis reveals several key insights:

- There is a consistent annual increase in successful landing outcomes from 2013.
- The overall success rate stands at approximately 66%.
- Success rates vary by launch site.
- The VAFB-SLC launch site has not conducted launches with heavy payloads.
- The orbits designated as ES-L1, GEO, HEO, and SSO have achieved a flawless 100% average success rate; when omitting the SO landing sites—which recorded a 0% success rate—all other landing sites maintain an average success rate above 50%.

Additionally, the top-performing classifier was decision tree.

SpaceX Launch Records Dashboard

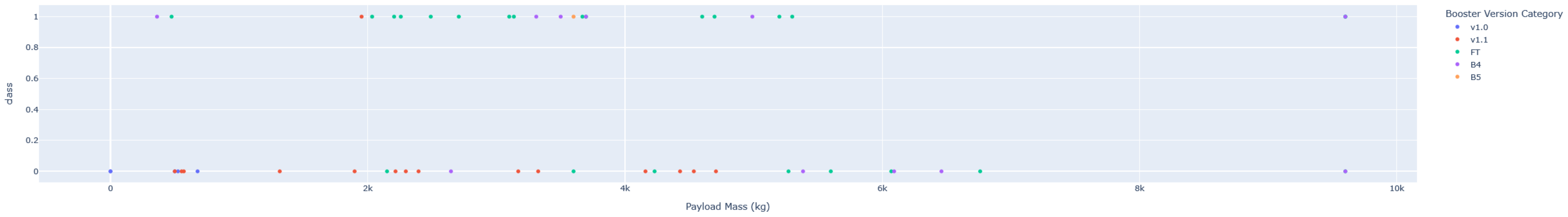
All Sites

Total Success Launches By Site

Payload range (Kg):

0 2500 5000 7500

Correlation between Payload and Success for all Sites



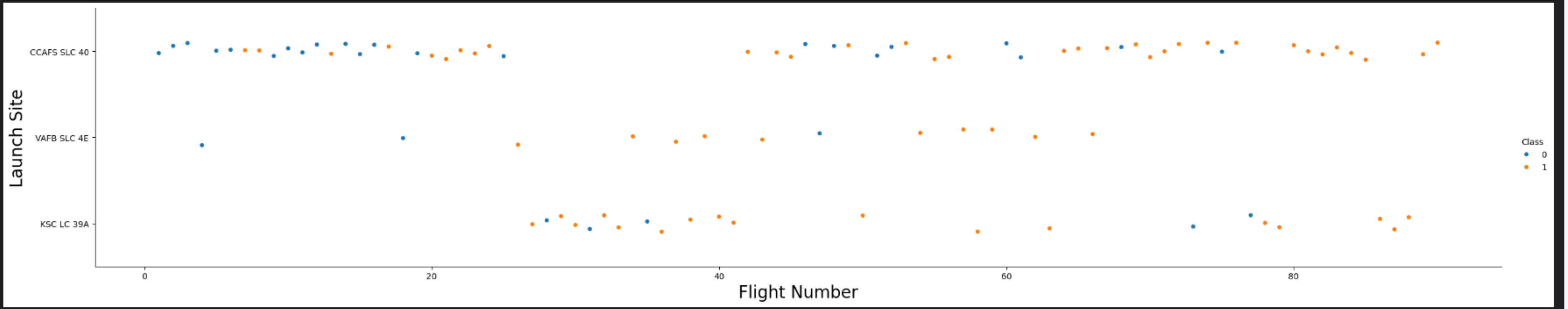
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

```python
# 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()
```

[5] Python



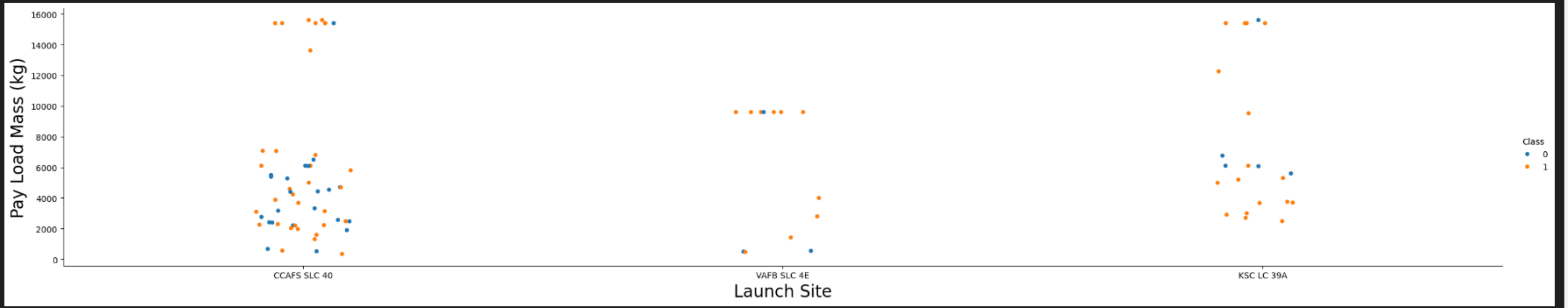
Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

In this graph, we can see how the number of successful flights evolves for each launch site, showing a trend of a higher number of successful launches as the total number of launches increases. This is particularly evident at "CCAFS SLC 40," where it can be seen that during the first 20 launches there was a similar rate of failed and successful missions, while from launch number 60 onward almost all have been successful.

Payload vs. Launch Site

```python
# 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="PayloadMass", x="LaunchSite", hue="Class", data=df, aspect = 5)
plt.xlabel("Launch Site",fontsize=20)
plt.ylabel("Pay Load Mass (kg)",fontsize=20)
plt.show()
```

[6] Python



Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

In this graph, it can be observed that there is a higher concentration of failed launches for payload values below 10,000 kg, while most launches with higher payloads are successful. This is especially evident at "CCAFS SLC 40."

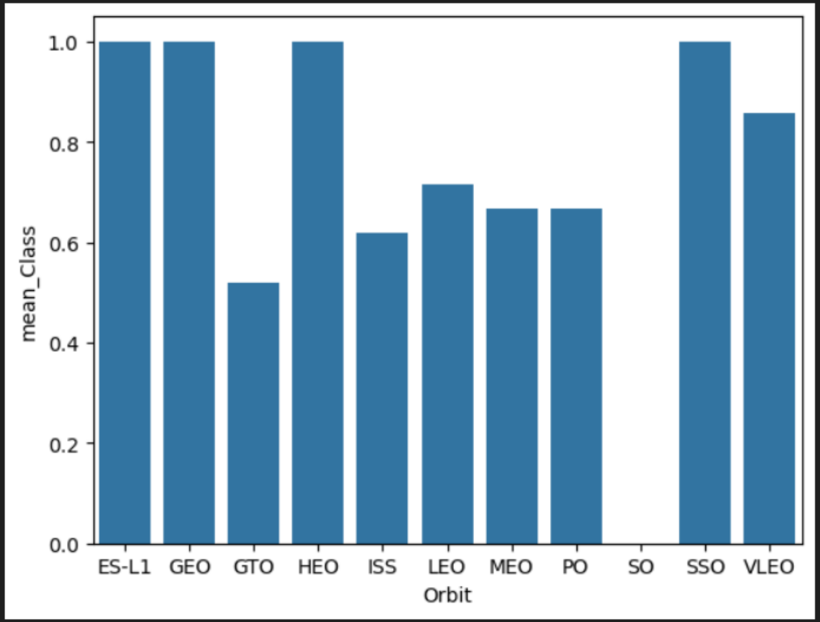
Success Rate vs. Orbit Type

```python
# HINT use groupby method on Orbit column and get the mean of Class column
df1 = df.groupby(['Orbit']).agg(mean_Class=('Class', 'mean'))
sns.barplot(x='Orbit', y='mean_Class',data=df1)
```

[7]

Python

<AxesSubplot:xlabel='Orbit', ylabel='mean_Class'>



Analyze the plotted bar chart to identify which orbits have the highest success rates.

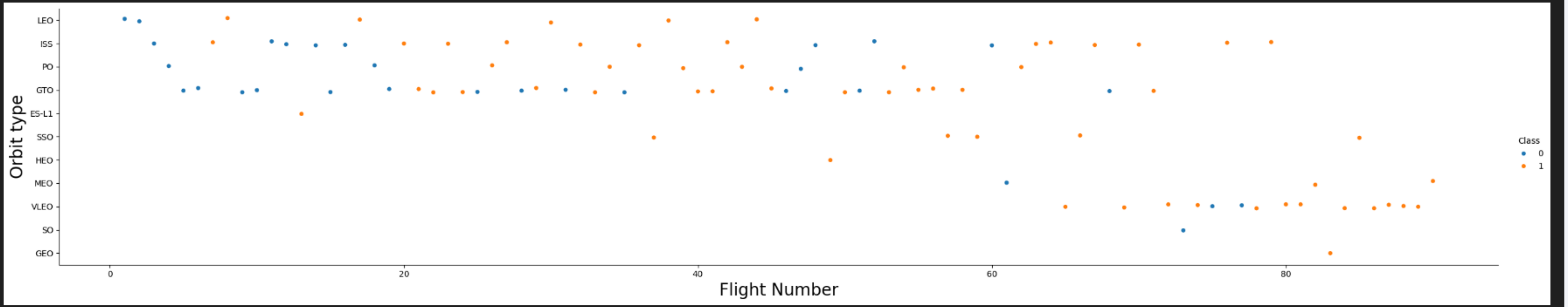
This bar chart easily shows that the ES-L1, GEO, HEO, and SSO orbits have the highest success rate, followed by VLEO.

Flight Number vs. Orbit Type

```python
# 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("Flight Number",fontsize=20)
plt.ylabel("Orbit type",fontsize=20)
plt.show()
```

[8] Python



You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

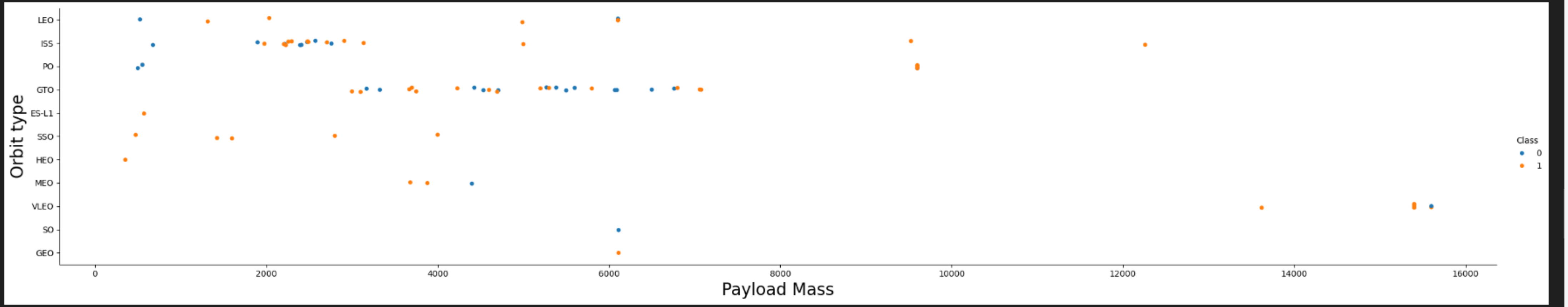
Payload vs. Orbit Type

```python
# Plot a scatter point chart with x axis to be Payload Mass 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 Mass",fontsize=20)
plt.ylabel("Orbit type",fontsize=20)
plt.show()
```

[9]

Python



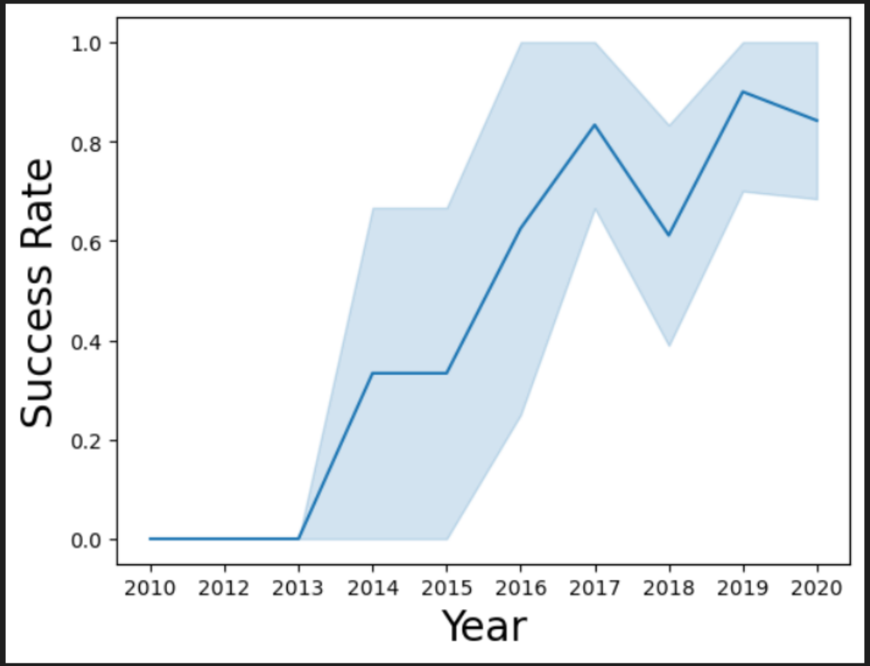
With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

Launch Success Yearly Trend

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

[33] Python



You can observe that the sucess rate since 2013 kept increasing till 2020.

All Launch Site Names

[11]

```
%sql SELECT DISTINCT Launch_Site from SPACEXTABLE
```

Python

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

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

Launch Site Names Begin with 'CCA'

```
[12]  %sql SELECT Launch_Site from SPACEXTABLE where Launch_Site LIKE 'CCA%' LIMIT 5
                                                                                    Python
 * sqlite:///my_data1.db
Done.
```

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

```
[13]  %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer Like 'NASA (CRS)'
                                                                                    Python
... * sqlite:///my_data1.db
    Done.
```

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

```
[15]
%sql select AVG(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version Like '%F9 v1.1%'
```

Python

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

AVG(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

```
[16] %sql select min(Date) from SPACEXTABLE where ("Landing_Outcome" Like '%Success%ground%')
```

Python

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

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
[17] %sql select Booster_Version from SPACEXTABLE where (PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000) and ("Landing_Outcome" Like '%Success%ship%')
```

Python

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

```
%sql select distinct(Mission_Outcome),count(*) from SPACEXTABLE group by Mission_Outcome
```

[18] Python

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

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

Boosters Carried Maximum Payload

[19]

```
%sql select "Booster_Version",(select max("PAYLOAD_MASS__KG_") from SPACEXTABLE) as max_payload from SPACEXTABLE where (PAYLOAD_MASS__KG_=max_payload)
```

Python

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

Booster_Version	max_payload
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

```
%sql select "Booster_Version",substr(Date, 6,2) as month from SPACEXTABLE where (substr(Date,0,5)='2015' and Landing_Outcome like '%Failure%ship%')
```

[20] Python

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

Booster_Version	month
F9 v1.1 B1012	01
F9 v1.1 B1015	04

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

```
[25] %sql select Landing_Outcome, count(*) as count_outcomes from SPACEXTABLE where date between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by count_outcomes desc;
```

Python

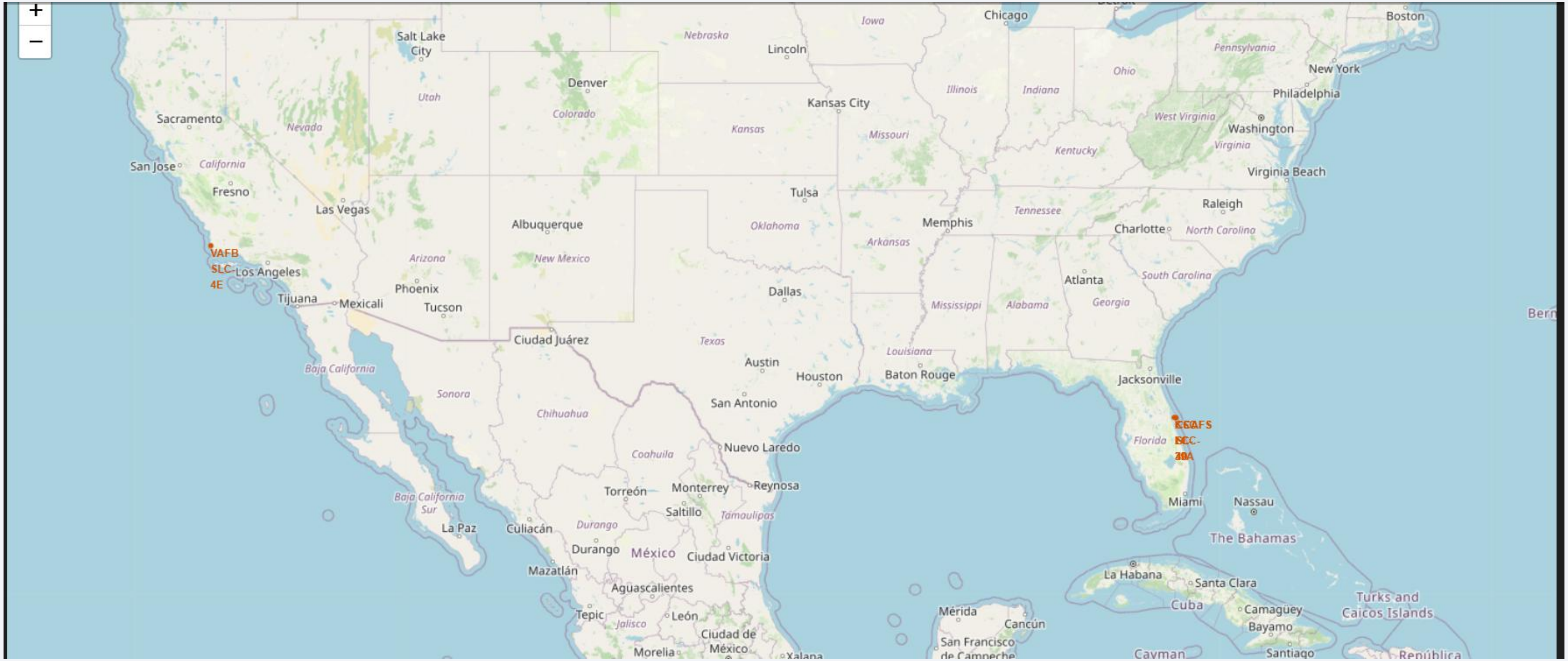
```
 * sqlite:///my_data1.db
Done.
```

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

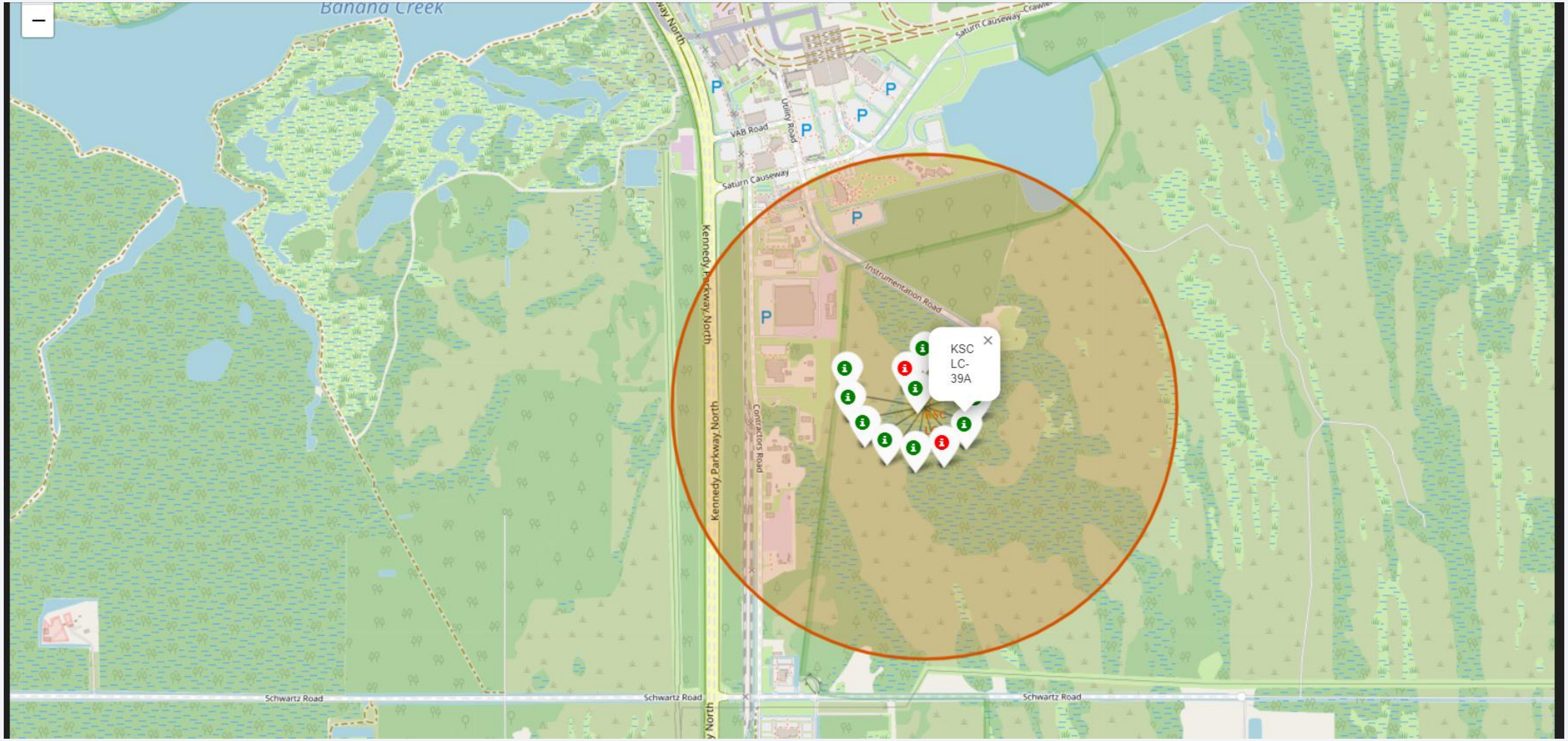
Section 3

Launch Sites Proximities Analysis

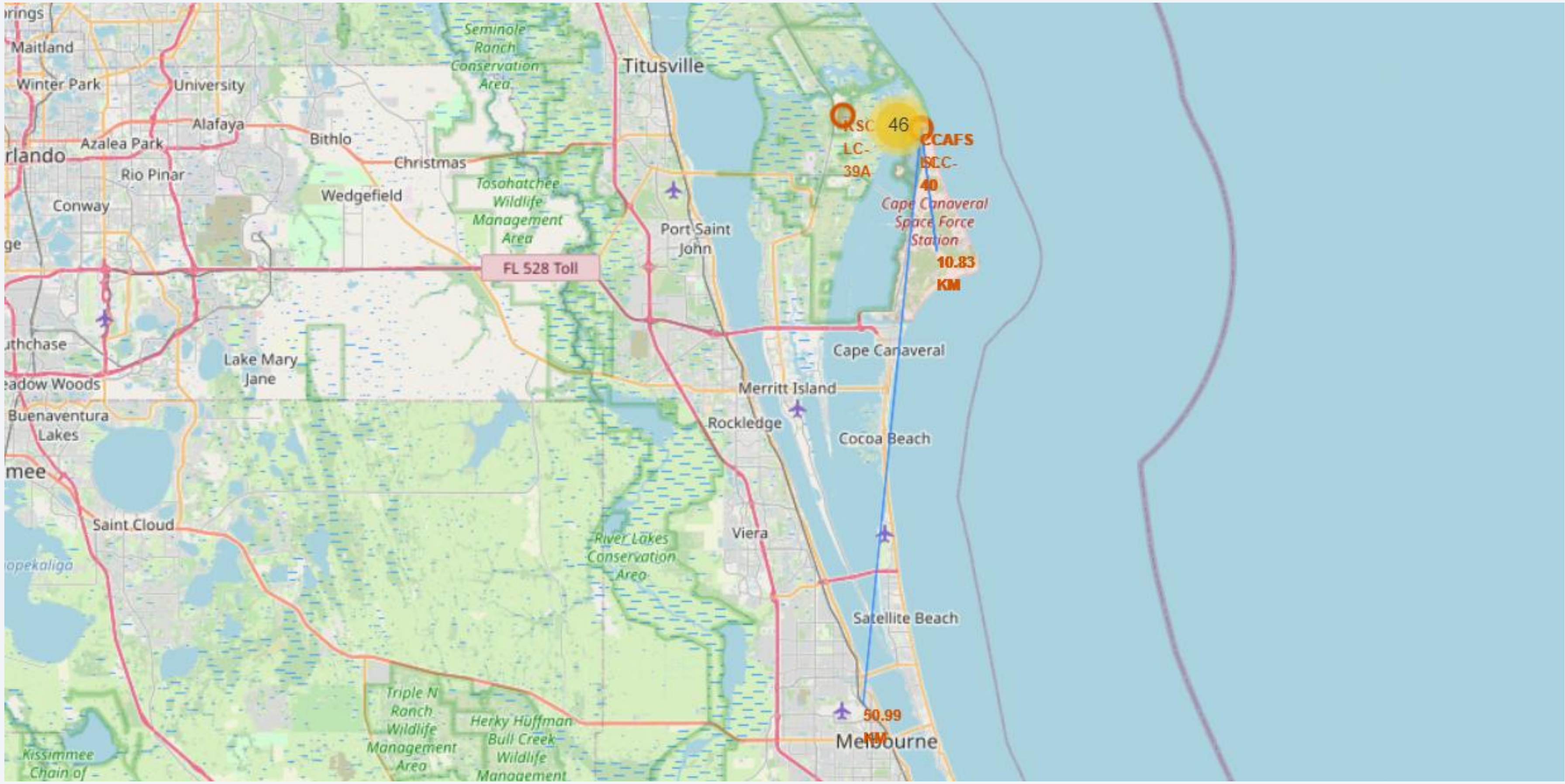
General view of launch sites



Success/failed launches location



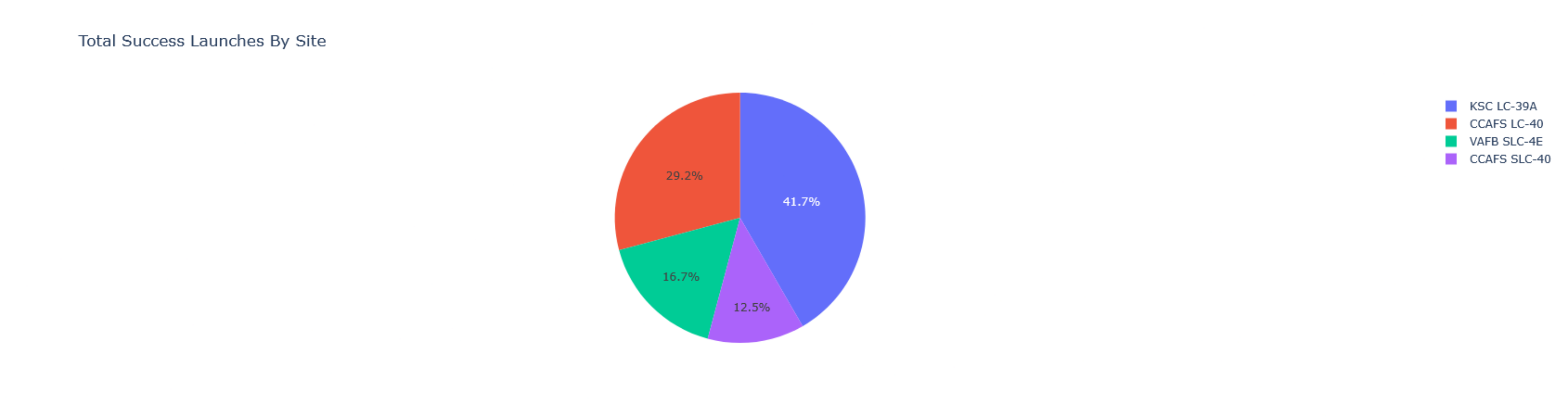
Distance between launch site and other infrastructures



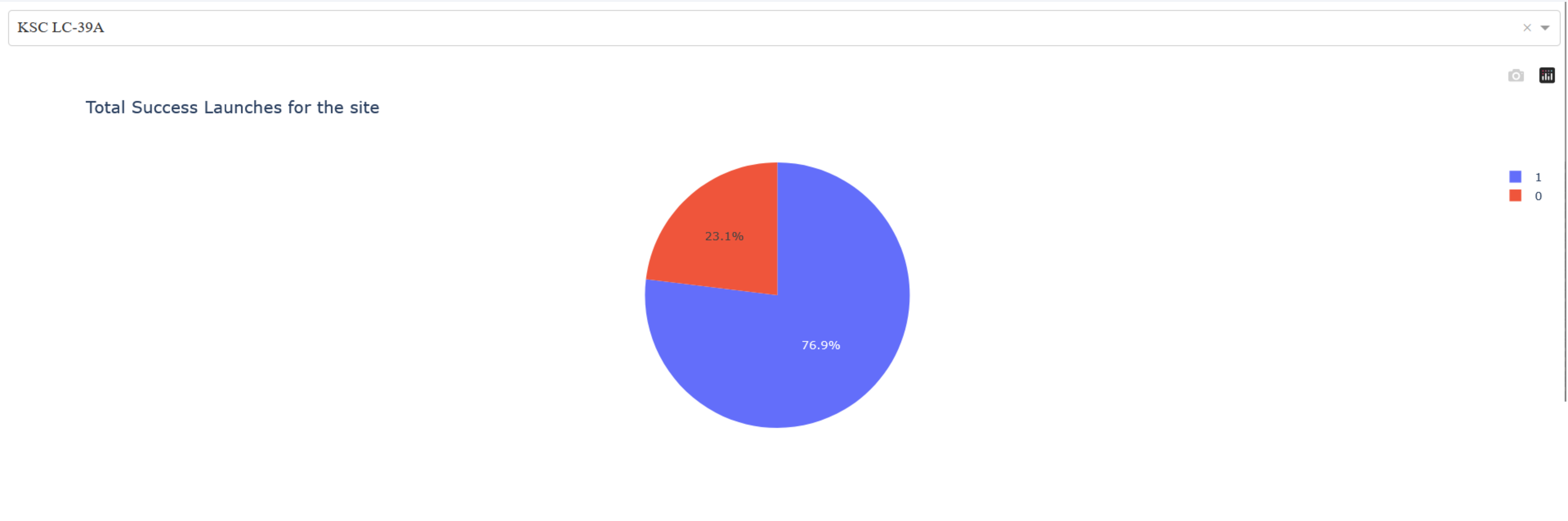
Section 4

Build a Dashboard with Plotly Dash

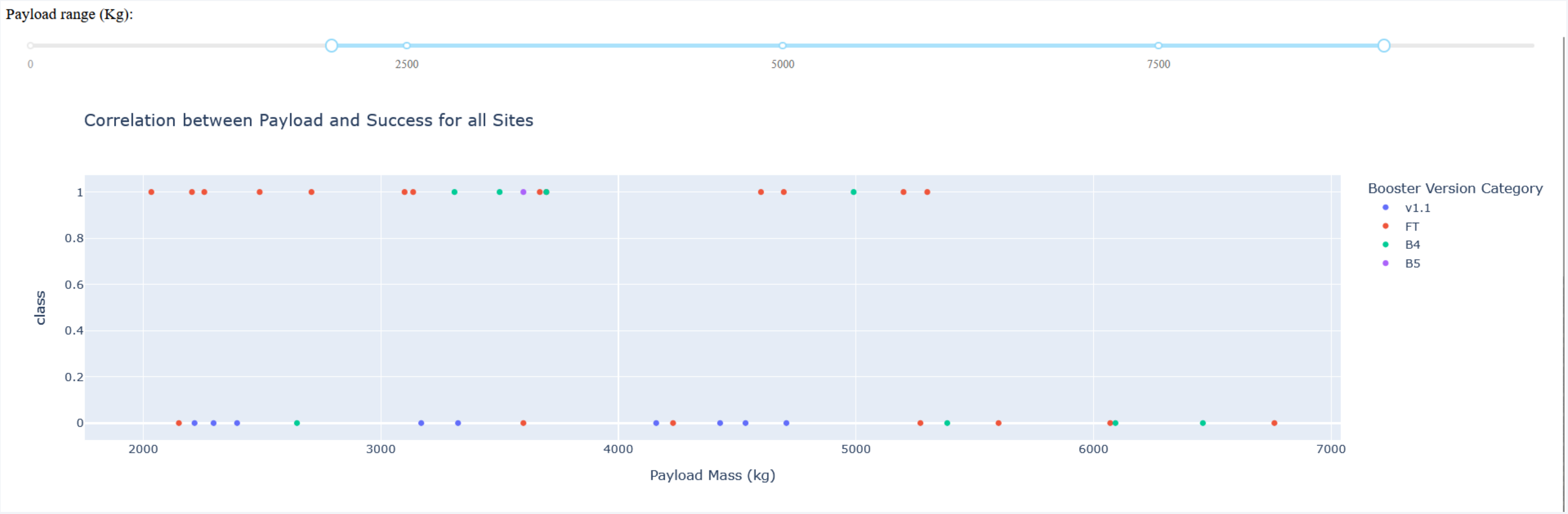
Total success launches by site



Total success rate of KSC LC-39A



Correlation between payload range and success for all sites



Section 5

Predictive Analysis (Classification)

Classification Accuracy

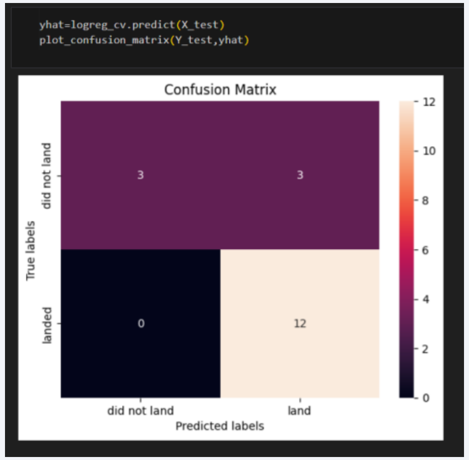
```python
best_score = {'Logistic regresssion': [logreg_cv.best_score_], 'SVM': [svm_cv.best_score_], 'Decision tree': [tree_cv.best_score_], 'KNN': [knn_cv.best_score_]}
df = pd.DataFrame.from_dict(best_score, orient='index', columns=['Best scores'])
df
```

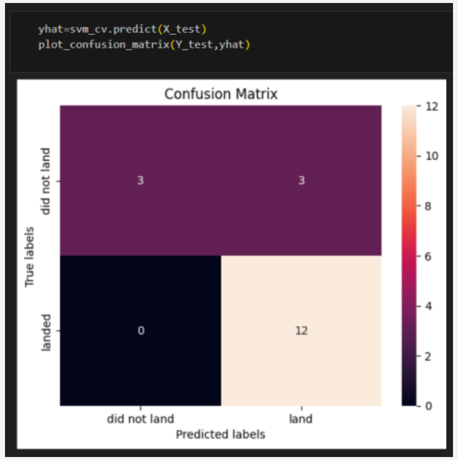
[45] Python

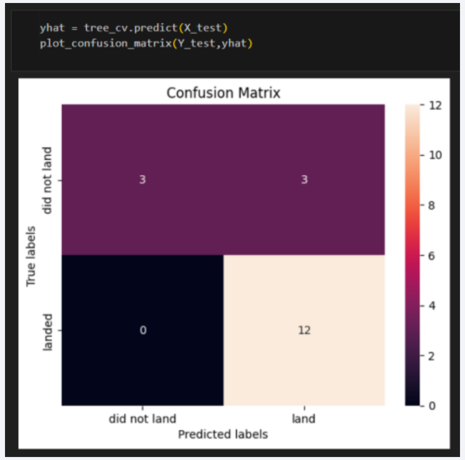
	Best scores
Logistic regresssion	0.846429
SVM	0.848214
Decision tree	0.889286
KNN	0.848214

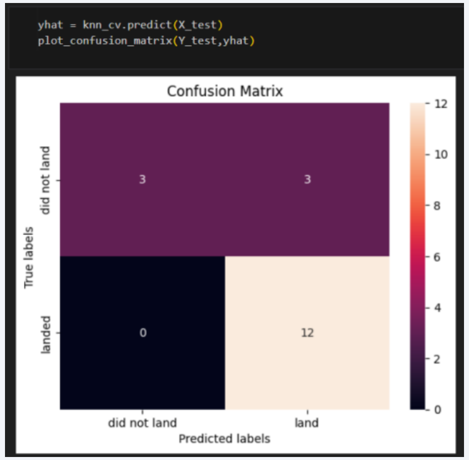
Decision tree has the best accuracy over 88.92%

Confusion Matrix









The result of the four confusion matrix of the models are the same. They perform the same with the given data.

Conclusions

- Launch success rates have shown a consistent increase over the years.
- Launches with higher payload masses tend to be more successful compared to those with heavier payloads.
- The orbits ES-L1, GEO, HEO, and SSO have achieved a flawless 100% success rate.
- The majority of launch sites are located near the Equator, with every site situated close to the coast.
- Among all the sites, KSC LC-39A boasts the highest success rate.
- The Decision Tree Model outperforms all other algorithms for this dataset.

Thank you!