IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Giang Le
30/4/2024

Outline

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

Executive Summary

- Summary of methodologies
  - SpaceX Data collection using SpaceX API
  - SpaceX Data Collection with Web Scraping
  - SpaceX Data Wrangling
  - SpaceX Exploratory Data Analysis using SQL
  - SpaceX EDA DataViz Using Python Pandas and Matplotlib
  - SpaceX Launch Sites Analysis with Folium-Interactive Visual Analytics and Ploty Dash
  - SpaceX Machine Learning Landing Prediction
- Summary of all results
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis(Classification)

Introduction

- Project background and context
  - SpaceX 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 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
- Problems you want to find answers
  - In this capstone, we will predict if the Falcon 9 first stage will land successfully

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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

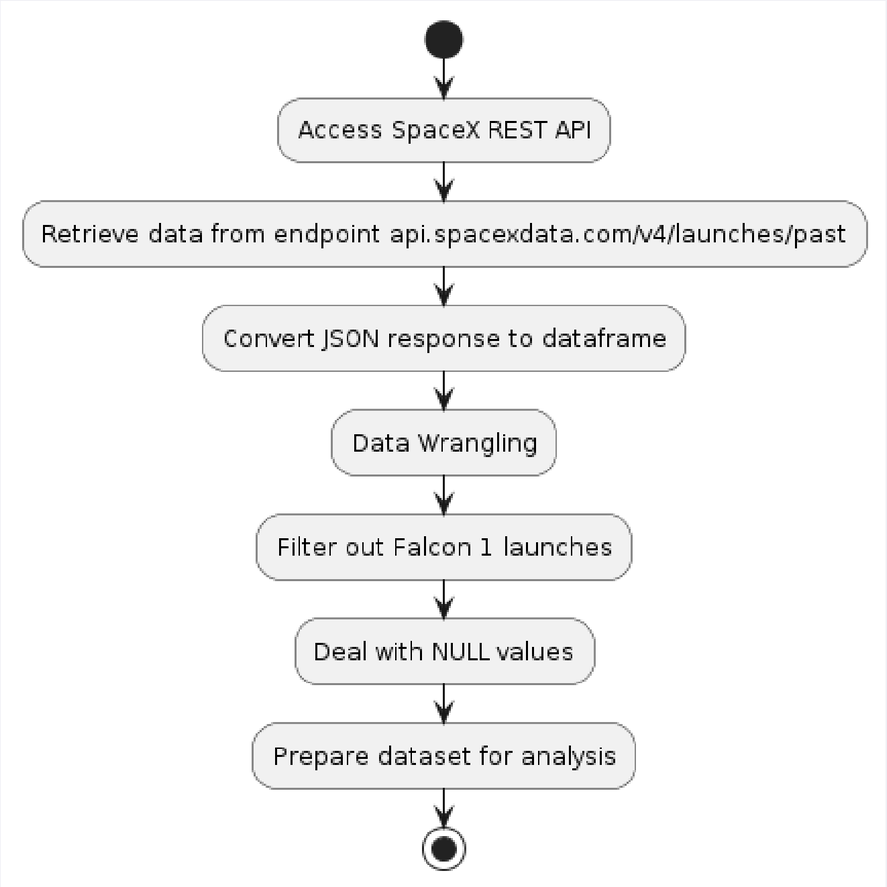
Data Collection

- Describe how data sets were collected.
  - **Using the SpaceX REST API**: Through API, various information about launches, including rocket details, payload information, launch and landing specifications, and landing outcomes, can be obtained.

  - **Web Scraping Wiki Pages**: Another method mentioned for obtaining Falcon 9 launch data is by web scraping Wiki pages. Python's BeautifulSoup package is used for this purpose to extract HTML tables containing Falcon 9 launch records

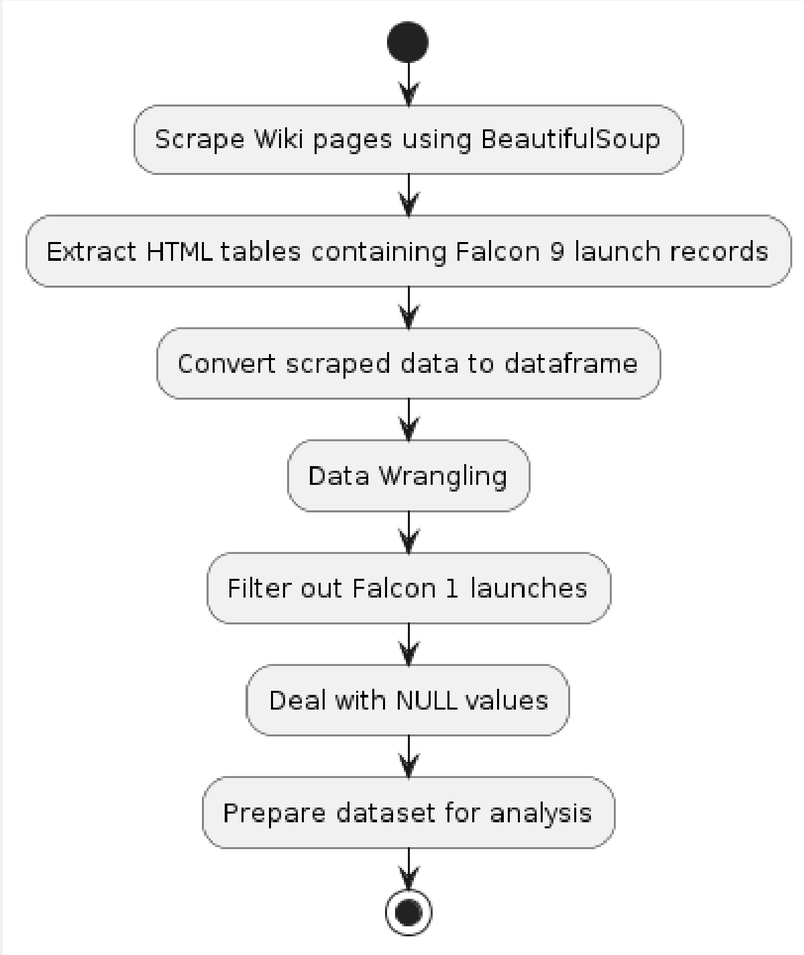
Data Collection – SpaceX API

- Access SpaceX REST API:
  - Using the SpaceX REST API to gather data related to Falcon 9 launches.
- Retrieve data from endpoint:
  - Data is retrieved from the specific endpoint api.spacexdata.com/v4/launches/past
- Convert JSON response to dataframe:
  - Retrieved data in JSON format, is converted into a dataframe for easier manipulation and analysis.
- Data Wrangling:
  - Filtering out Falcon 1 launches, dealing with NULL values, and preparing the dataset for analysis.
- Notebook URL: (must include completed code cell and outcome cell)



Data Collection - Scraping

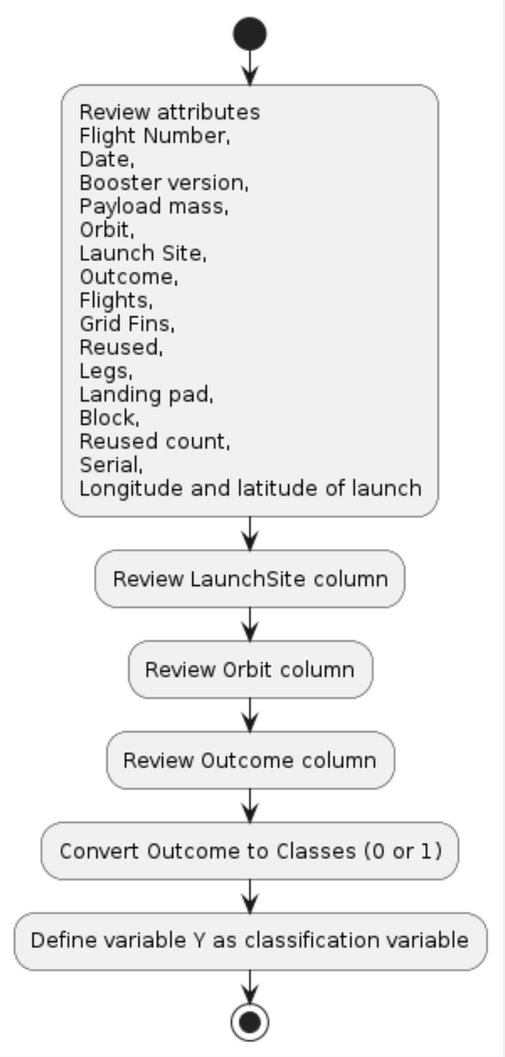
- Scrape Wiki pages using BeautifulSoup:
  - Ssing the BeautifulSoup to scrape Wiki pages that contain information about Falcon 9 launches.
- Extract HTML tables:
  - HTML tables containing Falcon 9 launch records are extracted from the scraped Wiki pages.
- Convert scraped data to dataframe:
  - The extracted data is then converted into a dataframe for further processing and analysis.
- Data Wrangling:
  - This phase includes steps such as filtering out Falcon 1 launches, handling NULL values, and preparing the dataset for analysis.
- Notebook URL: (must include completed code cell and outcome cell)



Data Wrangling

- Review attributes:
  - Review of several attributes associated with SpaceX launches
- Review LaunchSite column:
  - The flow moves to specifically review the LaunchSite column, which contains different launch sites where SpaceX missions have taken place.
- Review Orbit column:
  - Which lists the different orbits of the payload, such as Low Earth Orbit (LEO) and Geostationary Transfer Orbit (GTO).
- Review Outcome column:
  - Which indicates the success or failure of the first stage landing.
- Convert Outcome to Classes (0 or 1):
  - Outcome column is processed to convert the landing outcomes into classes represented by 0 (failed landing) or 1 (successfully land).
- Define variable Y as classification variable:
  - The flow concludes by defining the variable Y as the classification variable representing the outcome of each launch

- Notebook URL: (must include completed code cell and outcome cell)



EDA with Data Visualization

- Success Rate Over Time:
  - A line chart showing the success rate of Falcon 9 launches since 2013. This chart helps visualize the improvement in success rates over the years.
- Success Rate by Launch Site:
  - A bar chart comparing the success rates of different launch sites. This chart used to compare success rates between different launch sites, aiding in site selection for future launches.
- Success Rate by Payload Mass:
  - A scatter plot overlaying landing outcomes as colors, with payload mass on the x-axis. This chart illustrates the relationship between payload mass and success rates, highlighting patterns and thresholds.
- Correlation Analysis:
  - Charts showing correlations between different attributes and successful landings. These charts help identify which attributes are most influential in predicting successful landings, guiding feature selection for machine learning models.

- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

EDA with SQL (1)

- **Select distinct "Launch_Site" from SPACEXTABLE**
  - This SQL query retrieves all distinct values from the "Launch_Site" column in the SPACEXTABLE table. It provides a list of unique launch sites where SpaceX launches have taken place.
- **Select * from SPACEXTABLE where "Launch_Site" like "CCA%" limit 5**
  - This query selects all columns from the SPACEXTABLE where the "Launch_Site" column starts with "CCA" (using the % wildcard to match any characters after "CCA") and limits the results to 5 rows.
- **Select sum("PAYLOAD_MASS__KG_") from SPACEXTABLE where Customer = 'NASA (CRS)'**
  - This query calculates the sum of the "PAYLOAD_MASS__KG_" column for rows where the customer is 'NASA (CRS)'.
- **Select avg("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Booster_Version" like "F9 v1.1"**
  - This query calculates the average value of the "PAYLOAD_MASS__KG_" column for rows where the "Booster_Version" column matches 'F9 v1.1'.
- **Select * from SPACEXTABLE where "Landing_Outcome" like "Success (ground pad)" order by Date**
  - This query retrieves all columns from the SPACEXTABLE where the "Landing_Outcome" column matches 'Success (ground pad)' and orders the results by the "Date" column.

EDA with SQL (2)

- **select distinct "Booster_Version" from SPACEXTABLE where "PAYLOAD_MASS__KG_" between 4000 and 6000**
  - This query retrieves distinct values of the "Booster_Version" column from the SPACEXTABLE where the "PAYLOAD_MASS__KG_" column falls between 4000 and 6000.
- **select count(*), "Mission_Outcome" from SPACEXTABLE group by "Mission_Outcome"**
  - This query calculates the count of rows for each unique value in the "Mission_Outcome" column and groups the results accordingly.
- **select "Booster_Version" from SPACEXTABLE where "PAYLOAD_MASS__KG_" in (select max("PAYLOAD_MASS__KG_") from SPACEXTABLE)**
  - This query retrieves the "Booster_Version" column from the SPACEXTABLE for rows where the "PAYLOAD_MASS__KG_" column matches the maximum payload mass value in the table.
- **select * from SPACEXTABLE where "Landing_Outcome" like "Failure (drone ship)" and substr(Date, 0,5)='2015'**
  - This query retrieves all columns from the SPACEXTABLE where the "Landing_Outcome" column matches 'Failure (drone ship)' and the year portion of the "Date" column is '2015'.
- **select count(*) as count ,Landing_Outcome from SPACEXTABLE where Date between '2010-06-04' and '2017-03-20' group by "Landing_Outcome" order by count desc**
  - This query calculates the count of rows for each unique value in the "Landing_Outcome" column for rows where the "Date" column falls between '2010-06-04' and '2017-03-20', and orders the results by count in descending order.

Build an Interactive Map with Folium

- Two main types of map objects were created and added to a Folium map: markers and circles
  - Markers:
    - These markers provide visual representation of the exact locations of the launch sites.
    - By clicking on these markers, stakeholders can access additional information or perform further analysis related to each launch site.

  - Circles:
    - Circles were added to represent the close proximities of the launch sites on the interactive map.
    - These circles visually illustrate the areas surrounding each launch site where certain conditions or constraints may apply.

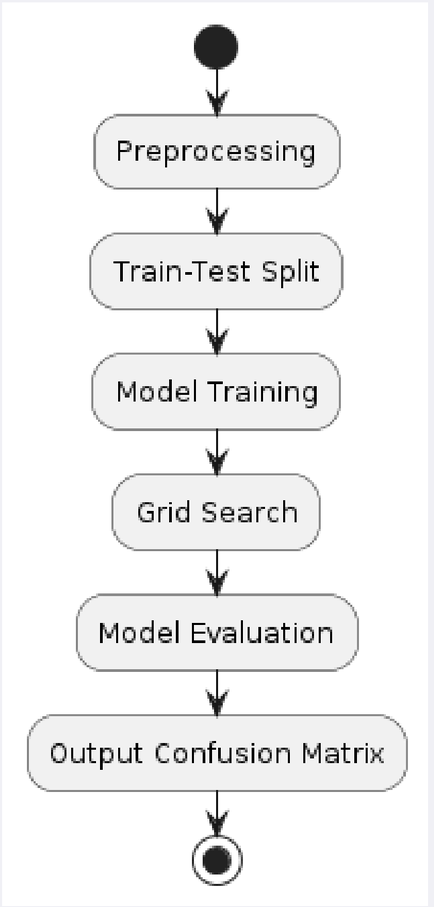
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build a Dashboard with Plotly Dash

- Two main types of plots/graphs and interactions were added to the dashboard:Pie Chart and Scatter Point Chart
  - Pie Chart:
    - The dashboard includes a pie chart that likely visualizes some categorical data related to the SpaceX dataset.
    - This chart allows stakeholders to quickly understand the distribution or composition of certain variables within the dataset.
  - Scatter Point Chart:
    - Which is commonly used to visualize relationships between two numerical variables.
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Predictive Analysis (Classification)

- **Preprocessing** the data, ensures that the data is in a suitable format for training the model.
- **Train-Test Split,** dataset is divided into two subsets: the training set used to train the model and the testing set to evaluate its performance.
- **Model Training** using various algorithms such as Logistic Regression, Support Vector Machines, Decision Tree Classifier, and K-nearest neighbors. Each algorithm is trained independently.
- **Grid Search** is performed to find the best hyperparameters for each model.
- **Model Evaluation**, after training and optimizing the models, they are evaluated using the training data.
- **Output Confusion Matrix** is outputted to visualize the performance of each model.
- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

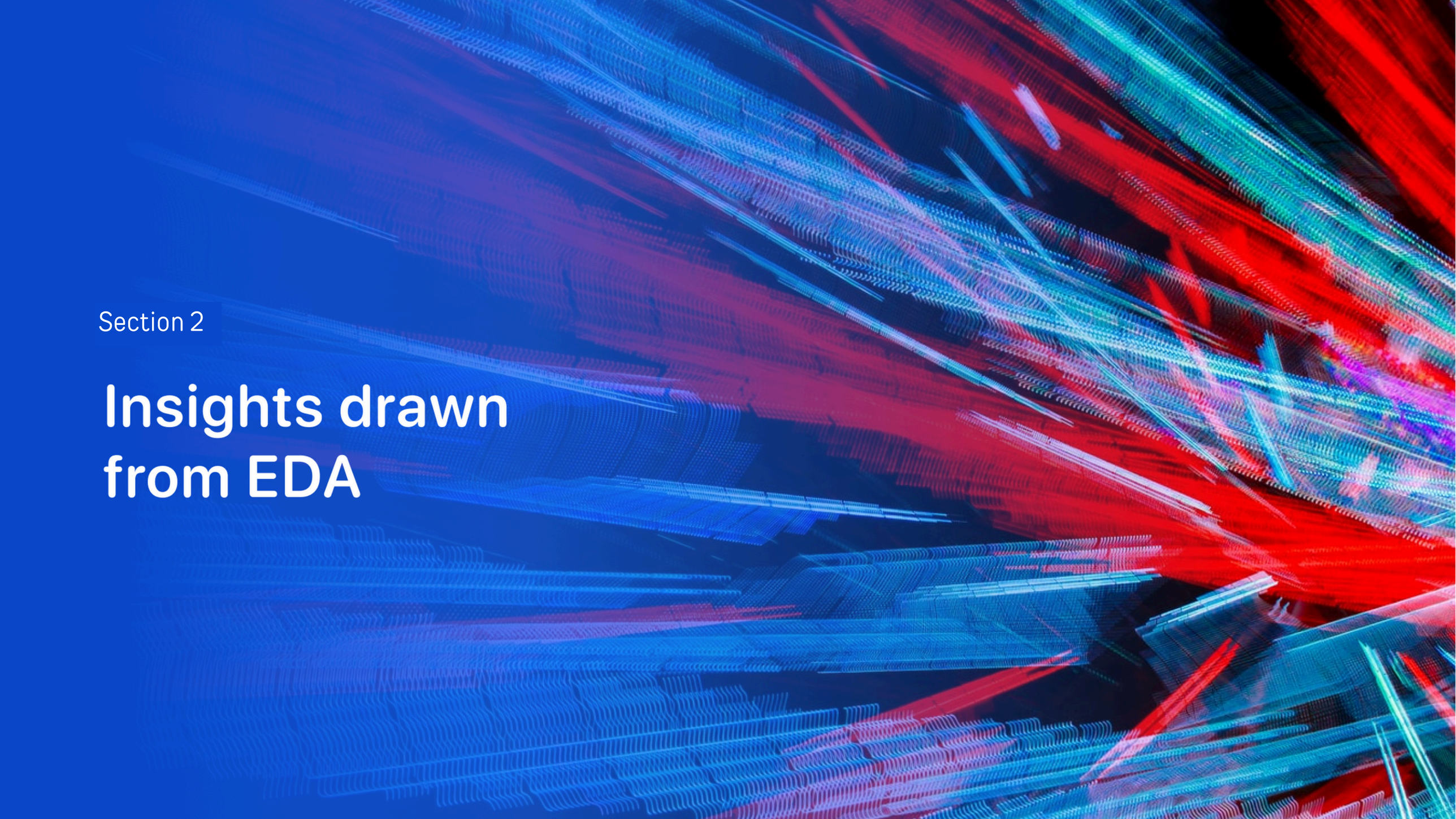


Results

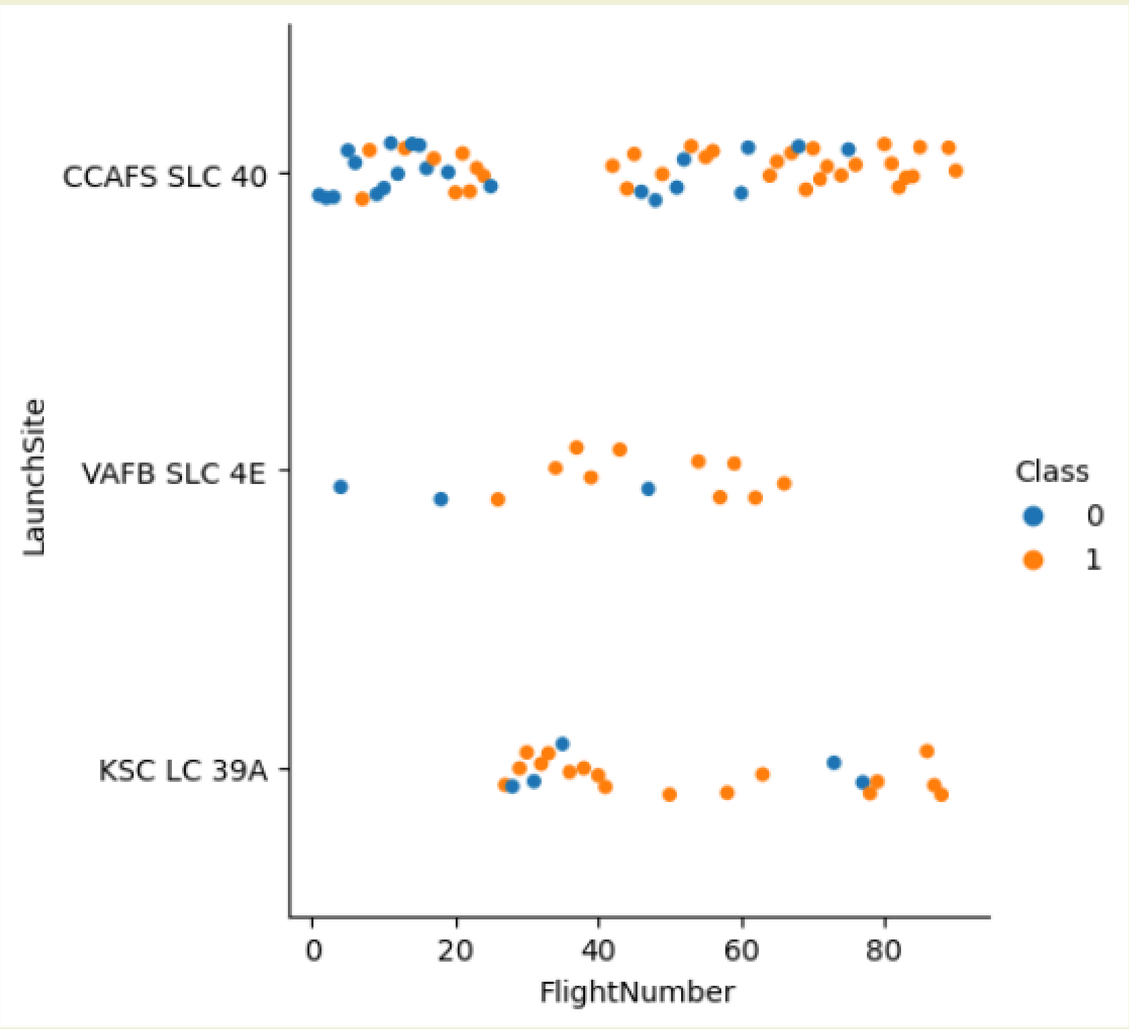
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

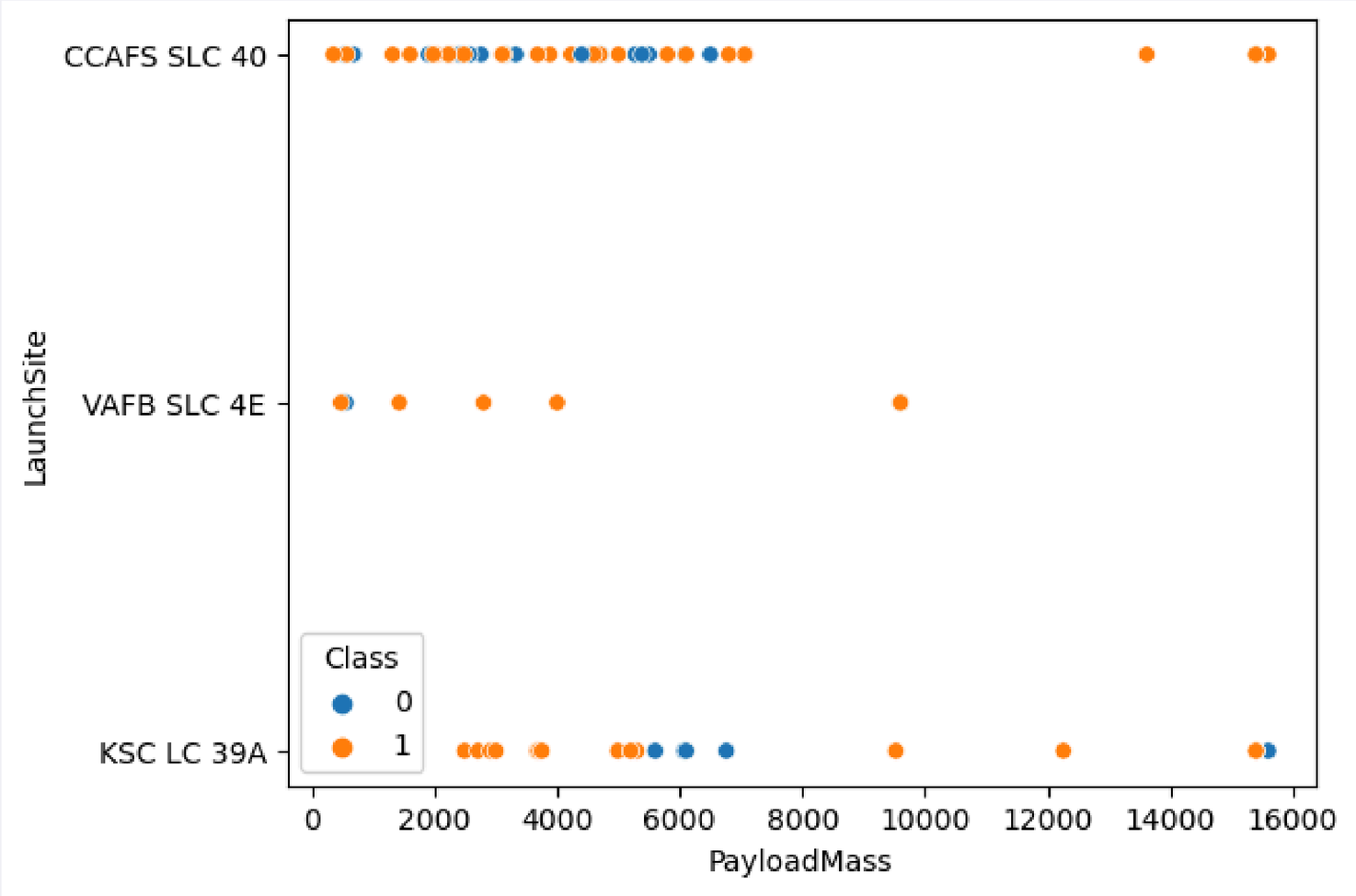
Insights drawn from EDA



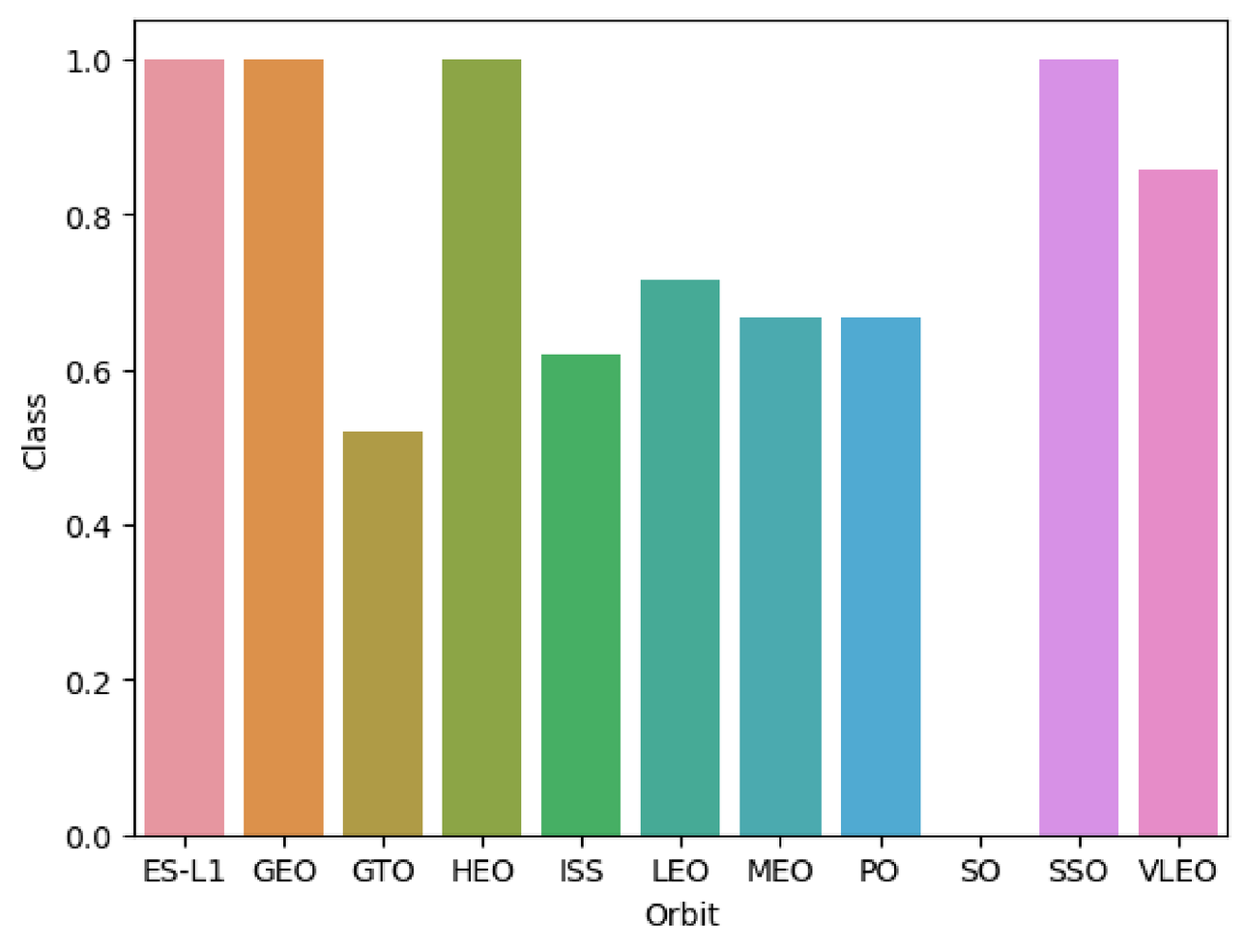
Flight Number vs. Launch Site



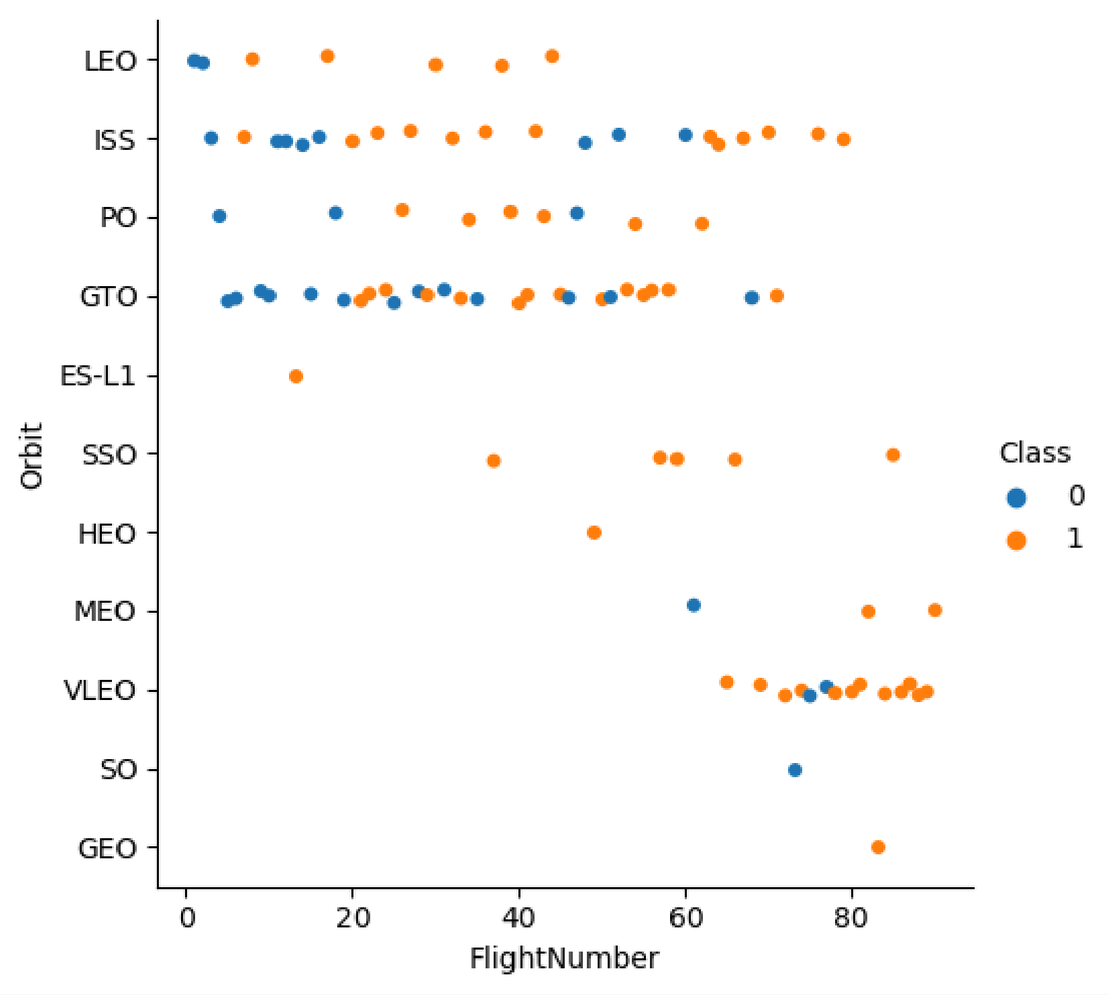
Payload vs. Launch Site



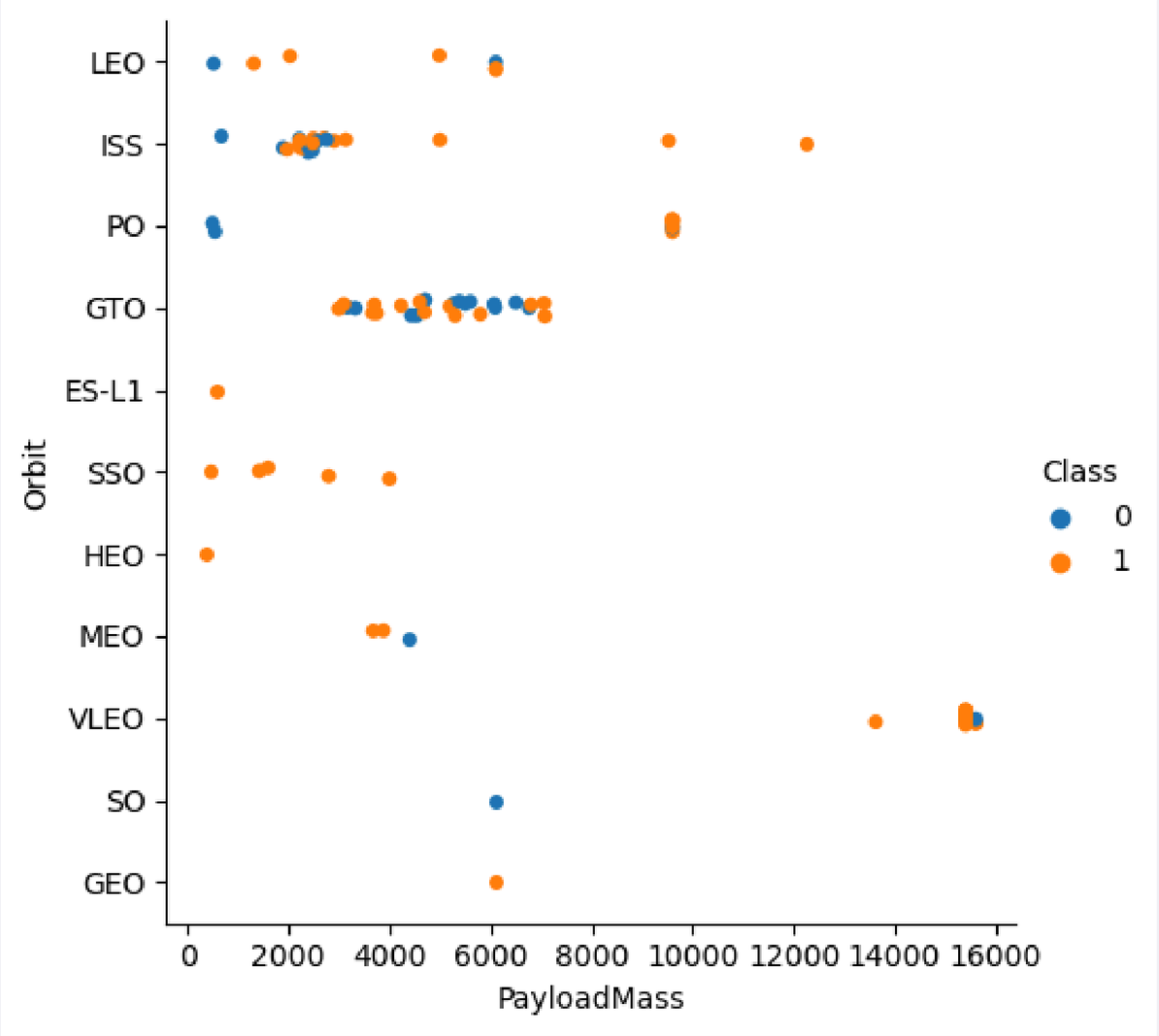
Success Rate vs. Orbit Type



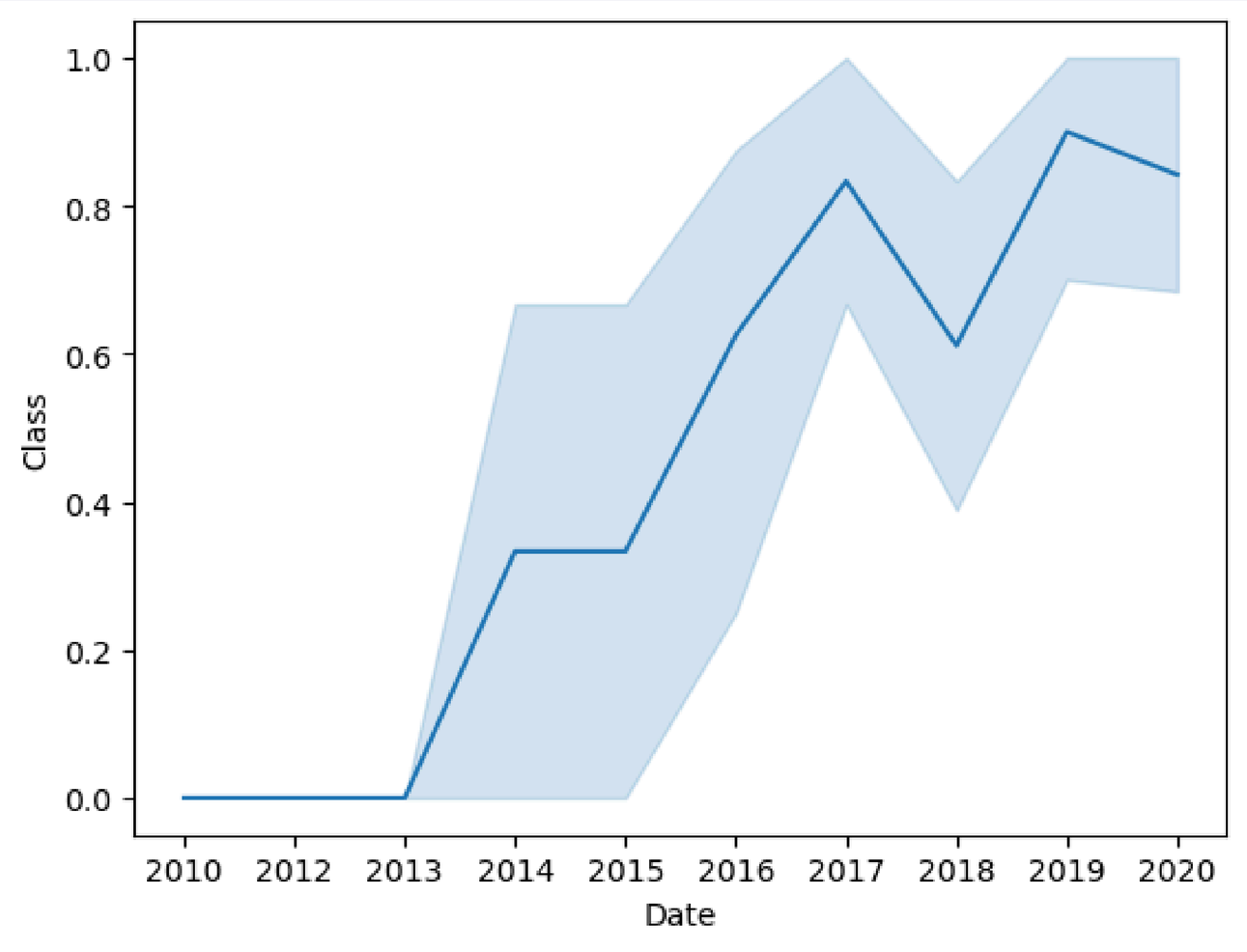
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

Task 1

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

```
%sql select distinct "Launch_Site" from SPACEXTABLE
```

[13] Python

Running query in 'sqlite:///my_data1.db'

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

- Find the names of the unique launch sites

- This SQL query retrieves all distinct values from the "Launch_Site" column in the SPACEXTABLE table. It provides a list of unique launch sites where SpaceX launches have taken place.

Launch Site Names Begin with 'CCA'

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

```
[21] %sql select * from SPACEXTABLE where "Launch_Site" like "CCA%" limit 5
                                                                        Python
```

Running query in 'sqlite:///my_data1.db'

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)

- Find 5 records where launch sites begin with `CCA`
- This query selects all columns from the SPACEXTABLE where the "Launch_Site" column starts with "CCA" (using the % wildcard to match any characters after "CCA") and limits the results to 5 rows.

Total Payload Mass

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

```
%sql select sum("PAYLOAD_MASS__KG_") from SPACEXTABLE where Customer = 'NASA (CRS)
'
```

Python

Running query in 'sqlite:///my_data1.db'

sum("PAYLOAD_MASS__KG_")
45596

- Calculate the total payload carried by boosters from NASA
- This query calculates the sum of the "PAYLOAD_MASS__KG_" column for rows where the customer is 'NASA (CRS)'.

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- This query calculates the average value of the "PAYLOAD_MASS__KG_" column for rows where the "Booster_Version" column matches 'F9 v1.1'.

Task 4

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

```
%sql select avg("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Booster_Version"
like "F9 v1.1"
```

Python

Running query in 'sqlite:///my_data1.db'

avg("PAYLOAD_MASS__KG_")
2928.4

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

```
%sql select * from SPACEXTABLE where "Landing_Outcome" like "Success (ground pad)
" order by Date
```

Python

Running query in 'sqlite:///my_data1.db'

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO

- Find the dates of the first successful landing outcome on ground pad
- This query retrieves all columns from the SPACEXTABLE where the "Landing_Outcome" column matches 'Success (ground pad)' and orders the results by the "Date" column.

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

```
%sql select distinct "Booster_Version" from SPACEXTABLE where "PAYLOAD_MASS__KG_"
between 4000 and 6000
```

Python

Running query in 'sqlite:///my_data1.db'

Booster_Version
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016
F9 FT B1020
F9 FT B1022

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- This query retrieves distinct values of the "Booster_Version" column from the SPACEXTABLE where the "PAYLOAD_MASS__KG_" column falls between 4000 and 6000.

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

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

Python

Running query in 'sqlite:///my_data1.db'

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

- Calculate the total number of successful and failure mission outcomes
- This query calculates the count of rows for each unique value in the "Mission_Outcome" column and groups the results accordingly.

Boosters Carried Maximum Payload

Task 8

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

```
%sql select "Booster_Version" from SPACEXTABLE where "PAYLOAD_MASS__KG_" in
(select max("PAYLOAD_MASS__KG_") from SPACEXTABLE)
```

Python

Running query in 'sqlite:///my_data1.db'

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5

- List the names of the booster which have carried the maximum payload mass
- This query retrieves the "Booster_Version" column from the SPACEXTABLE for rows where the "PAYLOAD_MASS__KG_" column matches the maximum payload mass value in the table.

2015 Launch Records

Task 9

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%sql select * from SPACEXTABLE where "Landing_Outcome" like "Failure (drone ship)
" and substr(Date,0,5)='2015'
```

[42] Python

Running query in 'sqlite:///my_data1.db'

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Cu
2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- This query retrieves all columns from the SPACEXTABLE where the "Landing_Outcome" column matches 'Failure (drone ship)' and the year portion of the "Date" column is '2015'.

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

Task 10

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.

```
%sql select count(*) as count ,Landing_Outcome from SPACEXTABLE where Date
between '2010-06-04' and '2017-03-20' group by "Landing_Outcome" order by count
desc
```

Python

Running query in 'sqlite:///my_data1.db'

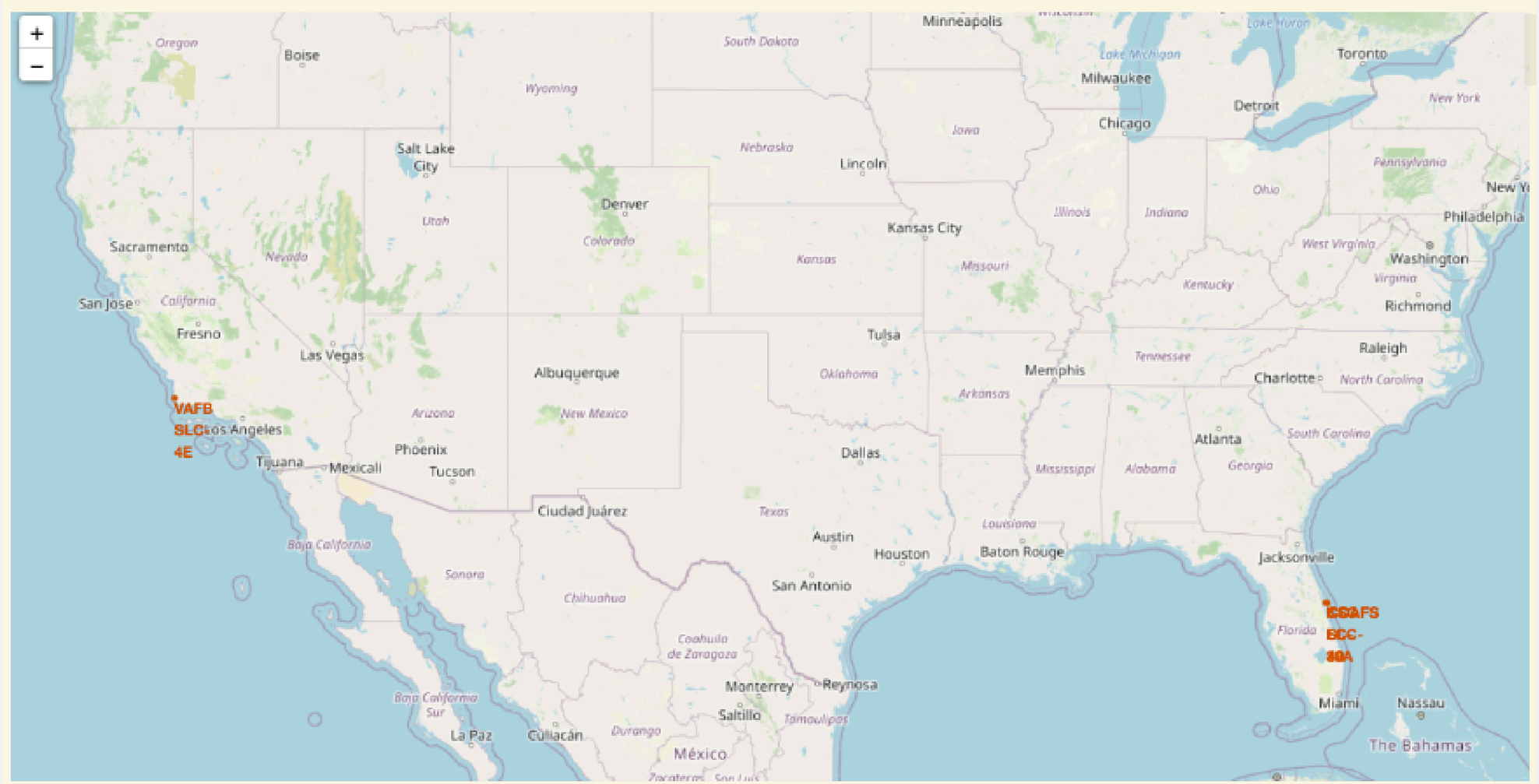
count	Landing_Outcome
10	No attempt
5	Success (drone ship)
5	Failure (drone ship)
3	Success (ground pad)
3	Controlled (ocean)
2	Uncontrolled (ocean)
2	Failure (parachute)
1	Precluded (drone ship)

- 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
- This query calculates the count of rows for each unique value in the "Landing_Outcome" column for rows where the "Date" column falls between '2010-06-04' and '2017-03-20', and orders the results by count in descending order.

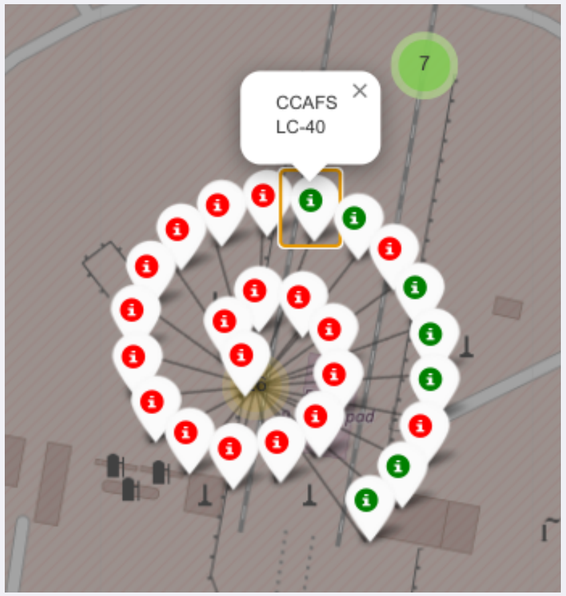
Section 3

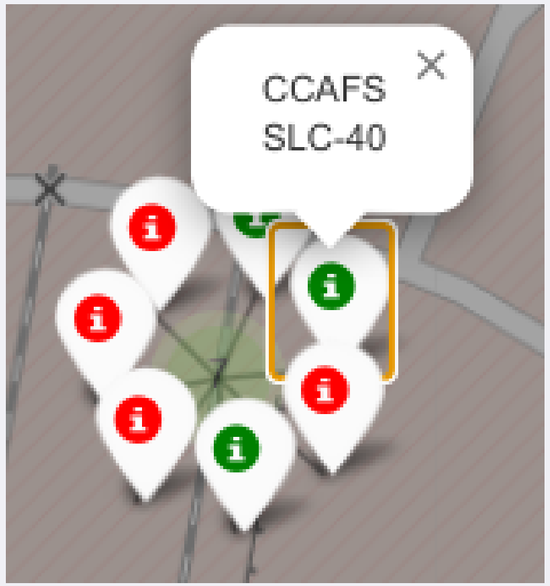
Launch Sites Proximities Analysis

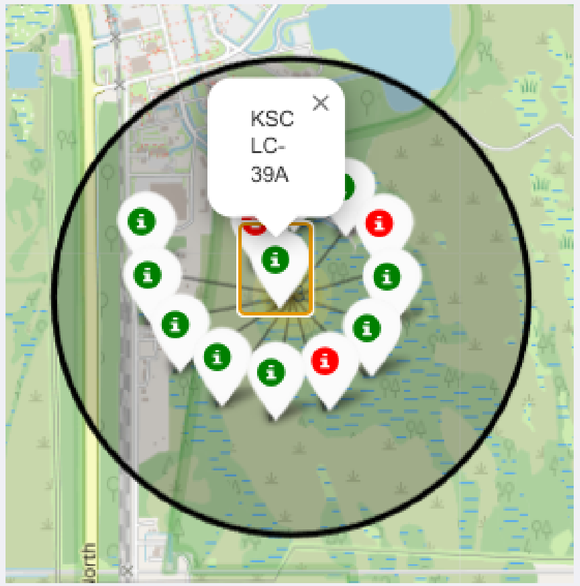
Mark all launch sites on a map

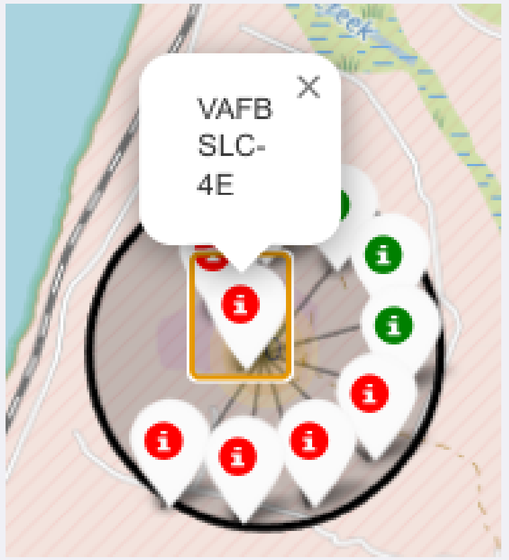


Mark the success/failed launches for each site on the map

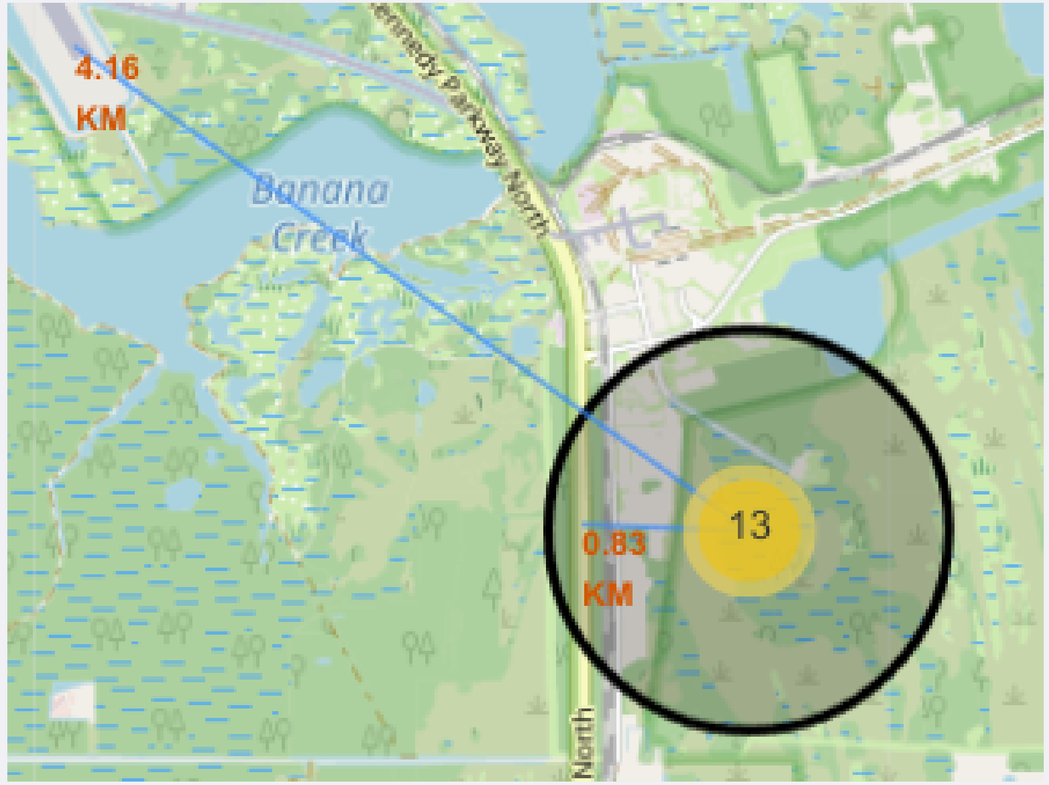


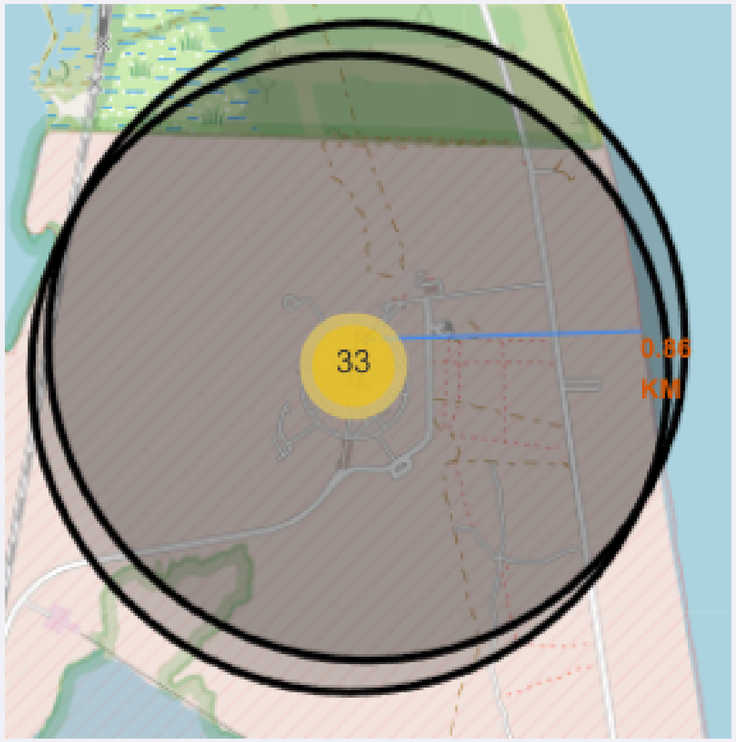


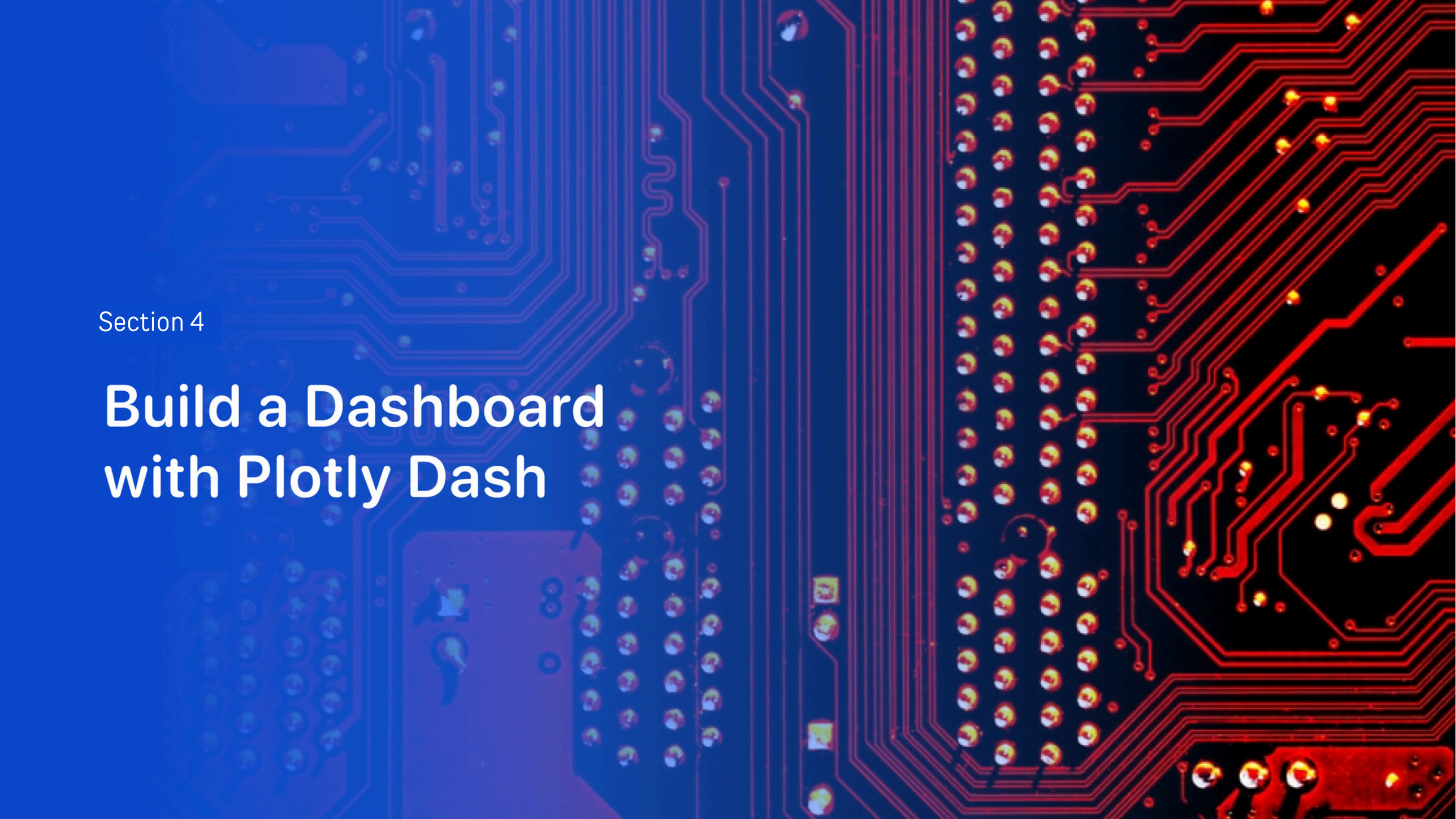




Calculate the distances between a launch site to its proximities



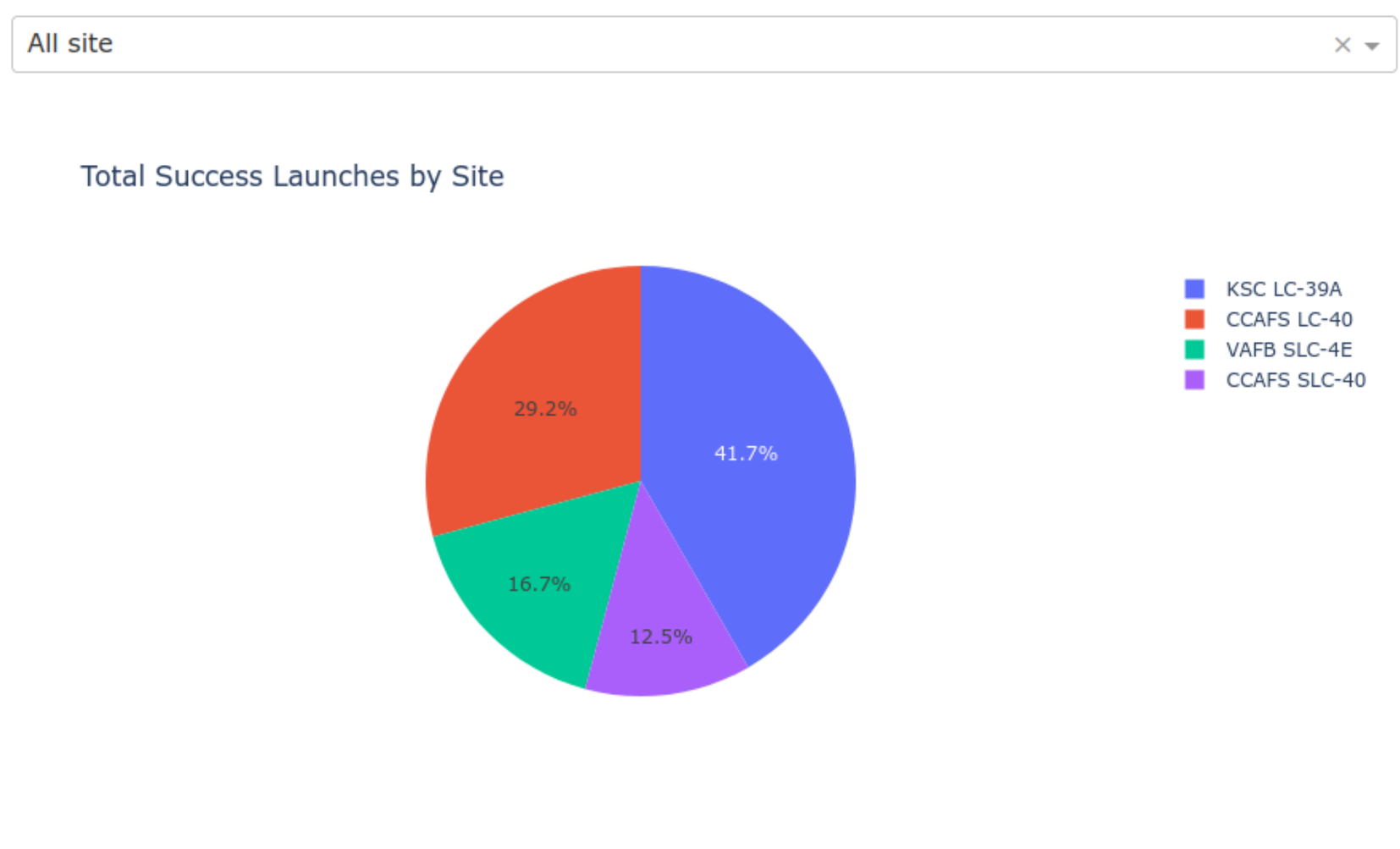




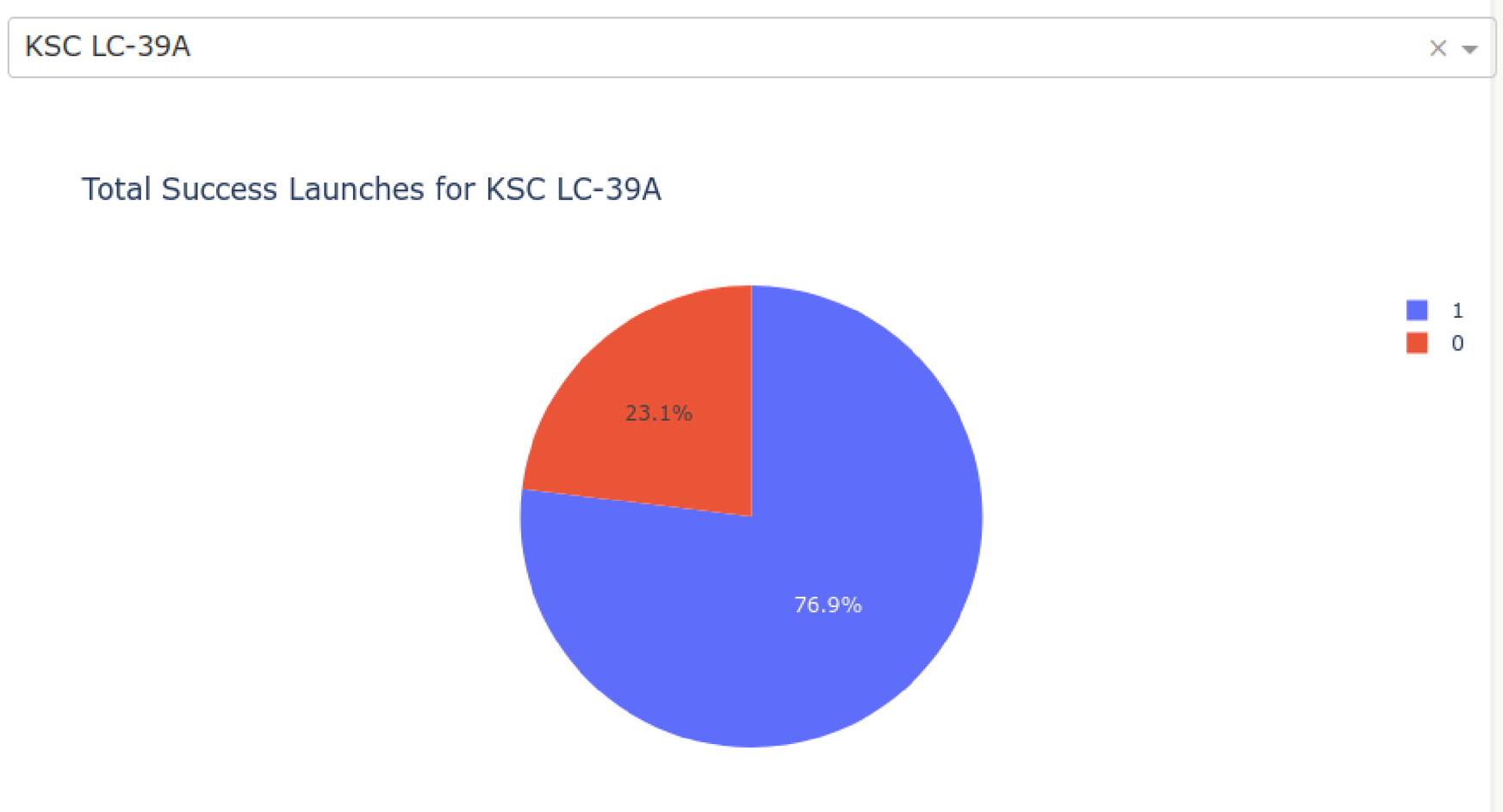
Section 4

Build a Dashboard with Plotly Dash

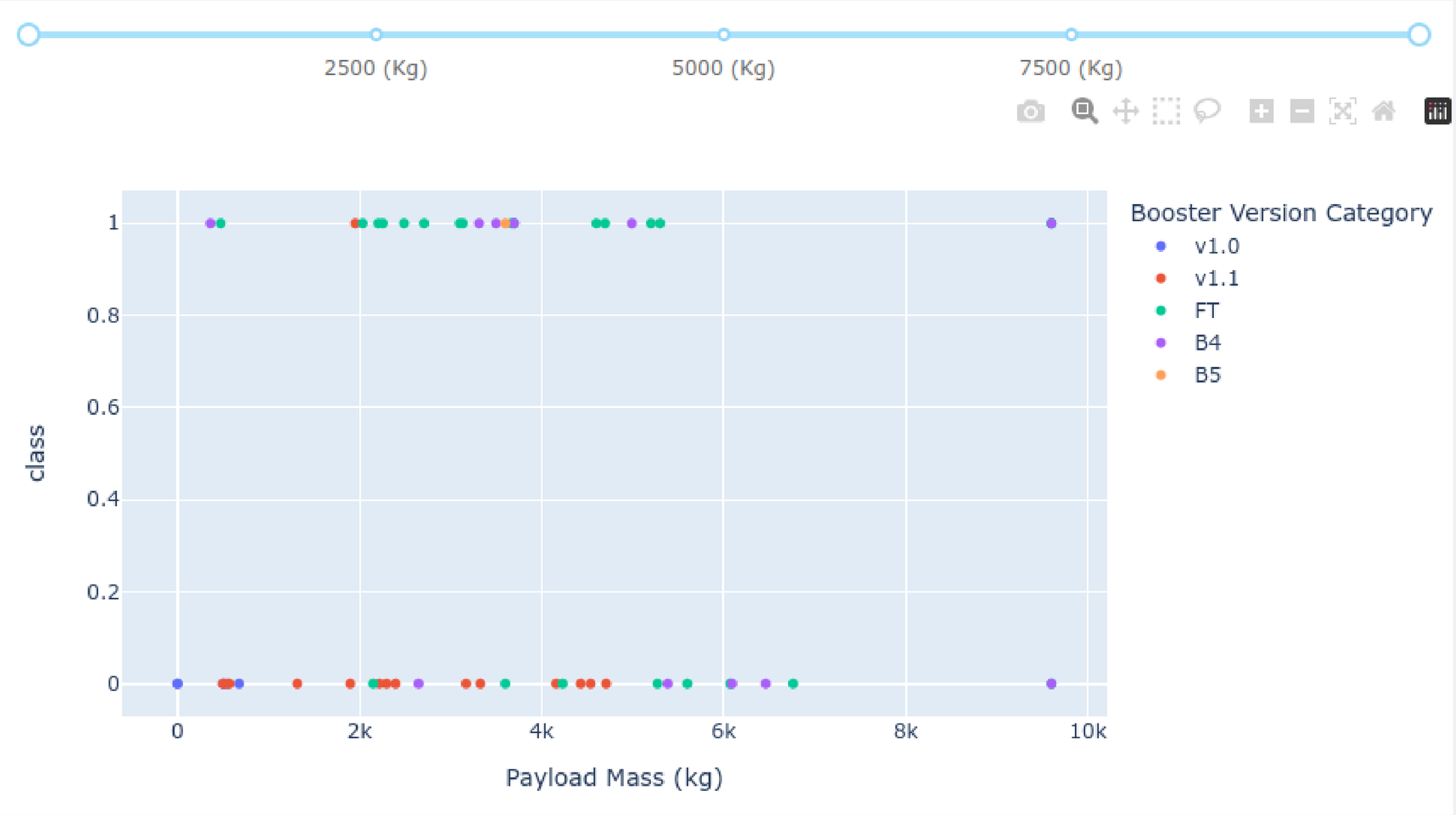
Launch success count for all sites (piechart)



Launch site with highest launch success ratio



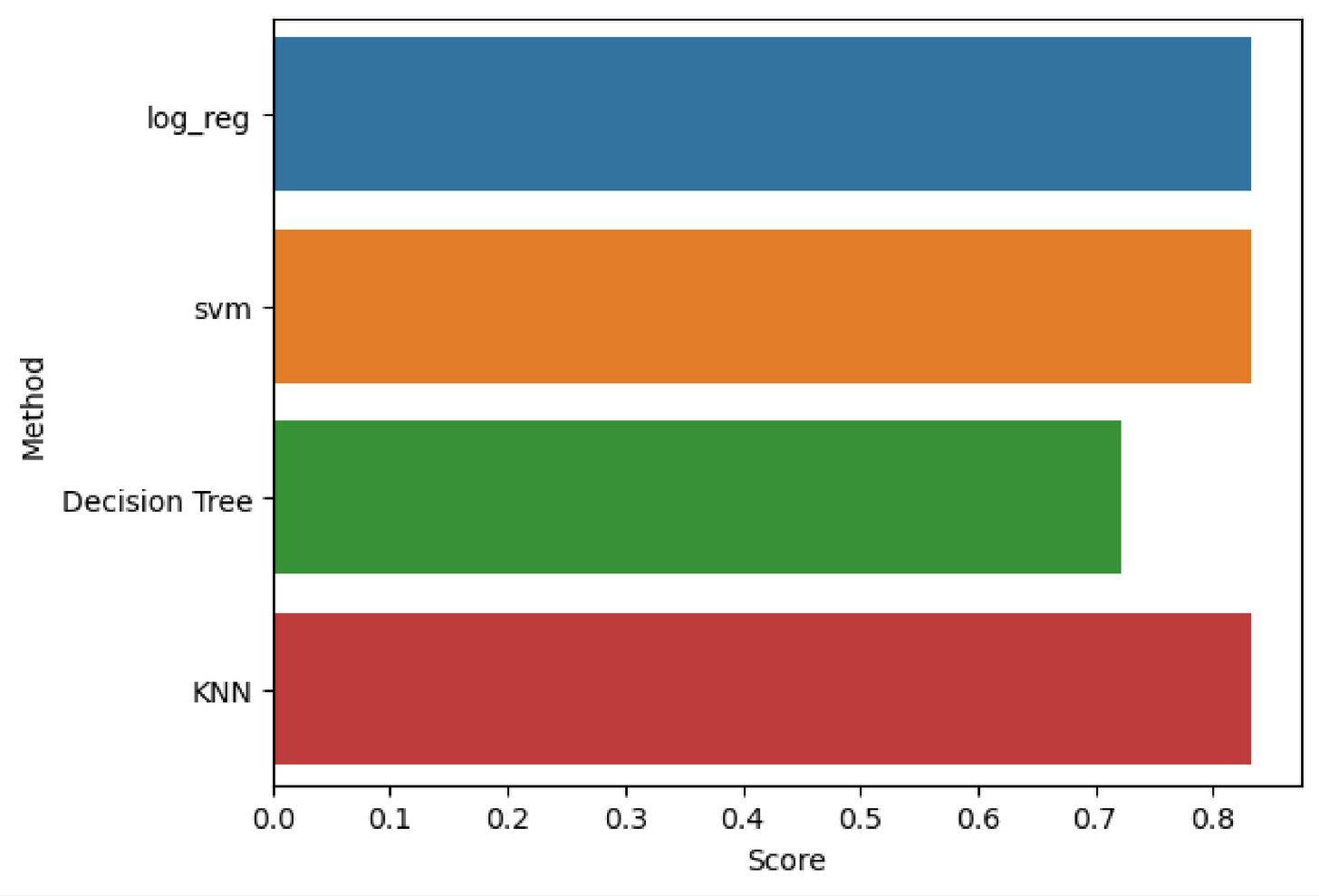
Payload vs. Launch Outcome



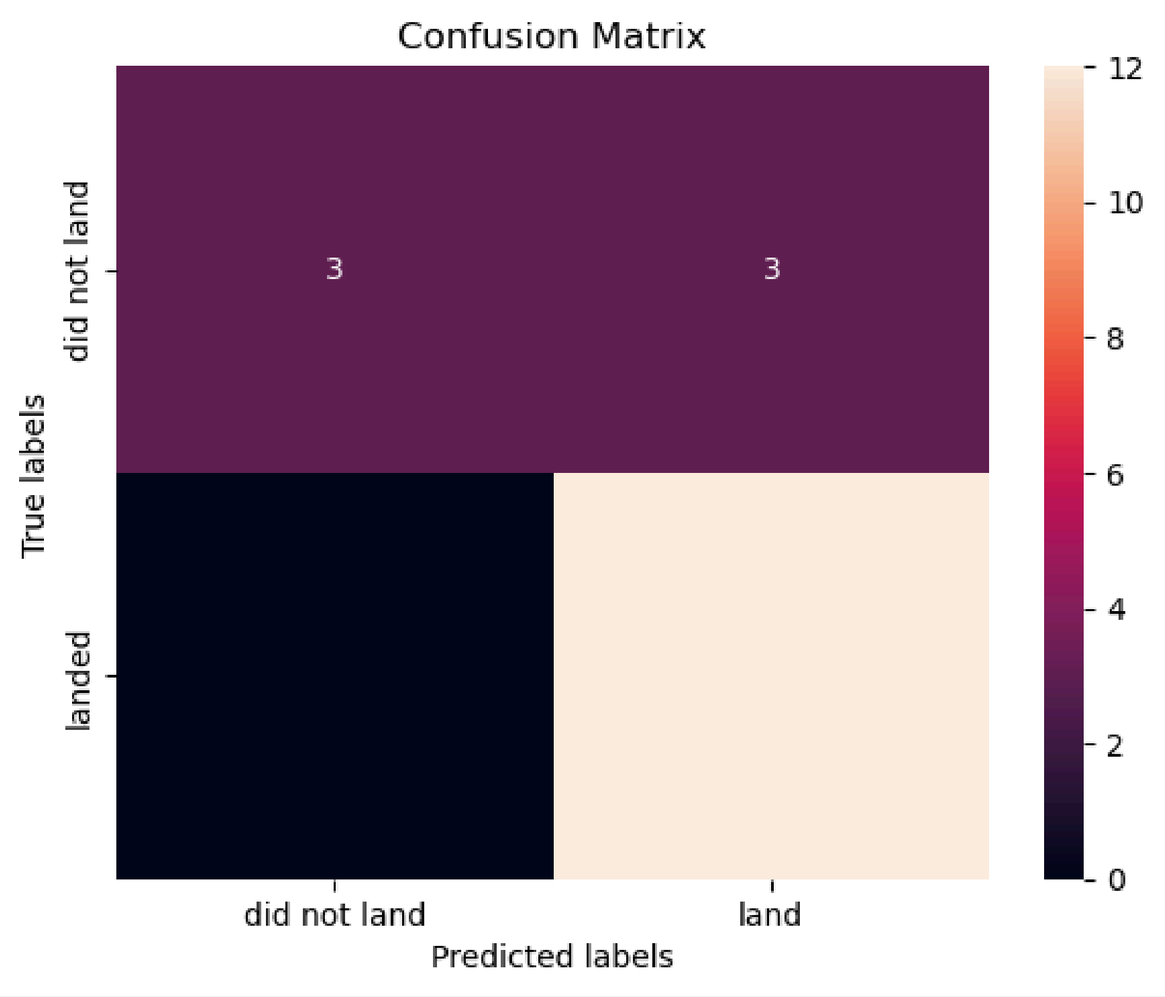
Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix



Thank you!