IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Rousseur G. Timbreza
04-05-24

Outline

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

Executive Summary

The goal of this research is to analyze SpaceX Falcon 9 data collected through various sources and employ Machine Learning models to predict the success of first stage landing that provides other space agencies the ability to decide if they bid against SpaceX.

- Summary of methodologies
  - Following concepts and methods were used to collect and analyze data, build and evaluate machine learning models, and make predictions:
  - Collect data through API and Web scraping
  - Transform data through data wrangling
  - Conduct exploratory data analysis with SQL and data visuals
  - Build an interactive map with folium to analyze launch site proximity
  - Build a dashboard to analyze launch records interactively with Plotly Dash
  - Finally, build a predictive model to predict if the first stage of Falcon 9 will land successfully

- • Summary of all results
  - This report will share results in various formats such as:
  - Data analysis results
  - Data visuals, interactive dashboards
  - Predictive model analysis results

Introduction

Project background and context:

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward 0f 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 Space X for a rocket launch.

Problems to find answers to:

- What are the variables that most influence the outcome of a landing?
- How do these variables influence the outcome of a landing?
- We want to predict if the Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

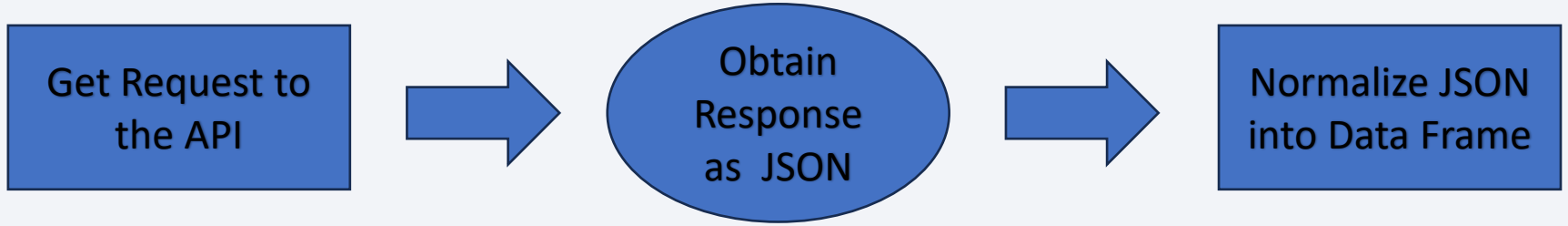
Executive Summary

- Data collection methodology:
  - • SpaceX API
  - • Web scrap Falcon 9 and Falcon Heavy launch records from Wikipedia (link)
- Perform data wrangling
  - One Hot Encoding and selection of relevant columns
- 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

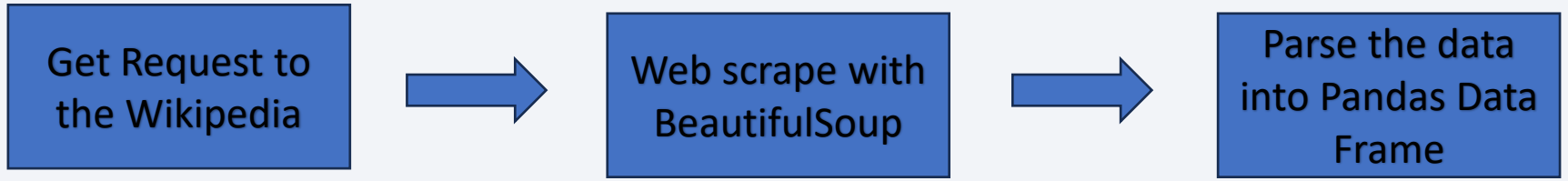
Data Collection with Space X Rest API:

A URL has been used to target a specific end point of the API to get past launch data. We used a get request to obtain the data and a response was obtained in the form of a list of JSON objects. The Json data have been normalized into a data frame.



Data Collection by Web Scraping Wikipedia with BeautifulSoup:

We used BeautifulSoup to web scrape HTML tables from the Wikipedia page. We then parsed the data and converted them into a Pandas data frame. Some columns had identifications number instead of actual data. We therefore needed to use the API again, , targeting another end point to gather specific data in order to build the dataset. Raw data was transformed into a clean dataset by wrangling, sampling and dealing with nulls.



Data Collection – SpaceX API

1.)Request data from SpaceX API with the URL:

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

2.)Use a get-request:

response = requests.get(spacex_url)

3.)Decode the response content as a Json and normalize it into a Pandas data frame:

data = pd.json_normalize(response.json())

4.)Some of the columns have IDs, use the API again to get data and store it in lists.

Construct our dataset using the data we have obtained. Combine the columns into a dictionary.

Create a Pandas data frame from the dictionary launch_dict:

data = pd.DataFrame.from_dict(launch_dict)

Guide: Github Data Collection API

Data Collection - Scraping

1.) Perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response:

```
r = requests.get(static_url)
```

2.) Create a BeautifulSoup object from the HTML response:

```
soup = BeautifulSoup(r.text, 'html.parser’)
```

3.) Collect all relevant column names from the HTML table header, iterate through the <th> elements and extract column names.

4.) Create an empty dictionary with keys from the extracted column names:

```
launch_dict= dict.fromkeys(column_names)
```

5.)Fill up the launch_dict with launch records extracted from table rows.

Create a dataframe:

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

Guide: Github Collection Web Scraping

Data Wrangling

- **Conducted Exploratory Data Analysis (EDA) to find patters in data and define labels for training supervised models**

- **The data set contained various mission outcomes that were converted into Training Labels with 1 meaning the booster successfully landed and 0 meaning booster was unsuccessful in landing. Following landing scenarios were considered to create labels:**

    - True Ocean means the mission outcome was successfully landed to a specific region of the ocean
    - False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean
    - RTLS means the mission outcome was successfully landed to a ground pad
    - False RTLS means the mission outcome was unsuccessfully landed to a ground pad
    - True ASDS means the mission outcome was successfully landed on a drone ship
    - False ASDS means the mission outcome was unsuccessfully landed on a drone ship

Guide: Github Data Wrangling

EDA with Data Visualization

As part of the Exploratory Data Analysis (EDA), following charts were plotted to gain further insights into the dataset:

1.) Scatter plot:

- Shows relationship or correlation between two variables making patterns easy to observe
- Plotted following charts to visualize:
  - Relationship between Flight Number and Launch Site
  - Relationship between Payload and Launch Site
  - Relationship between Flight Number and Orbit Type
  - Relationship between Payload and Orbit Type

2.) Bar Chart:

- Commonly used to compare the values of a variable at a given point in time. Bar charts makes it easy to see which groups are highest/common and how other groups compare against each other. Length of each bar is proportional to the value of the items that it represents
- Plotted following Bar chart to visualize:
- Relationship between success rate of each orbit type

3.) Line Chart:

- Commonly used to track changes over a period of time. It helps depict trends over time.
- Plotted following Line chart to observe: Average launch success yearly trend

Guide: Github EDA with Data Visualization

EDA with SQL

Performed SQL queries for EDA:

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA’
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first successful landing outcome in ground pad was achieved
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster versions which have carried the maximum payload mass
- 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
- 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

Guide: Github EDA with SQL

Build an Interactive Map with Folium

Interactive Visual Analytics with Folium:

- Mark all launch sites on the map: **folium.Circle** and **folium.Marker** using site's latitude and longitude coordinates
- Mark the success/failed launches for each site on the map: folium.Marker to **marker.cluster** with green markers if the launch was successful and red markers if it was not successful
- Calculate the distances between a launch site to its proximities: points on the closest coastline, city, railway, highway ecc. using **MousePosition** to calculate the distance between the coastline point and the launch site drawing a **PolyLine**

Guide: Github Visual Analytics with Folium

Build a Dashboard with Plotly Dash

- **Built a Plotly Dash web application to perform interactive visual analytics on SpaceX launch data in real-time. Added Launch Site Drop-down, Pie Chart, Payload range slide, and a Scatter chart to the Dashboard**.
    1) Added a Launch Site Drop-down Input component to the dashboard to provide an ability to filter Dashboard visual by all launch sites or a particular launch site
    2) Added a Pie Chart to the Dashboard to show total success launches when 'All Sites' is selected and show success and failed counts when a particular site is selected
    3) Added a Payload range slider to the Dashboard to easily select different payload ranges to identify visual patterns
    4) Added a Scatter chart to observe how payload may be correlated with mission outcomes for selected site(s). The color-label Booster version on each scatter point provided missions outcomes with different boosters

- **Dashboard helped answer following questions:**
    1) Which site has the largest successful launches? KSC LC-39A with 10
    2) Which site has the highest launch success rate? KSC LC-39A with 76.9% success
    3) Which payload range(s) has the highest launch success rate? 2000 – 5000 kg
    4) Which payload range(s) has the lowest launch success rate? 0-2000 and 5500 - 7000
    5) Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate? FT

Guide: Github Dashboard with Plotly Dash

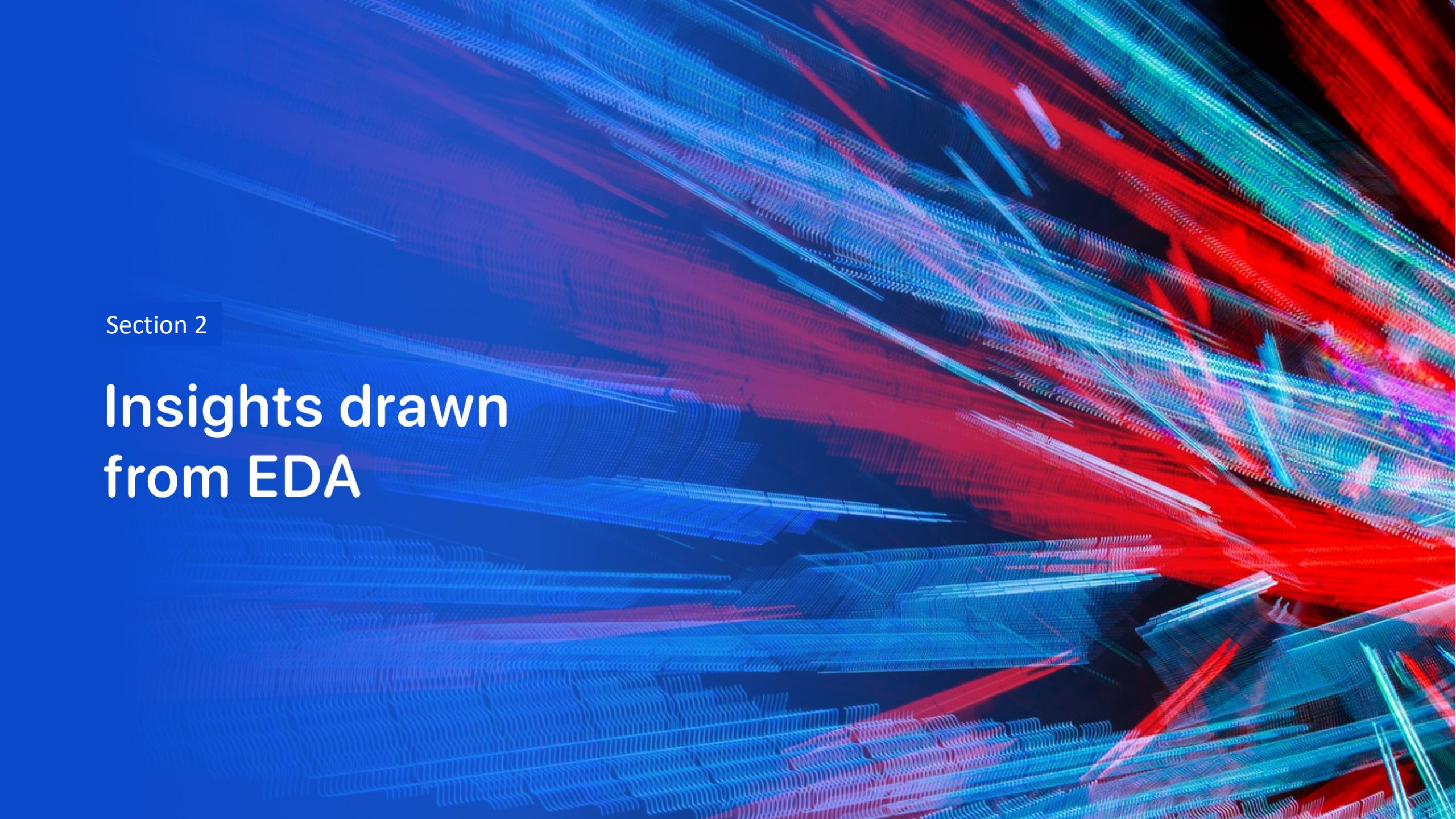
Predictive Analysis (Classification)

- **Building the Model:**
  - Standardize the data
  - Split into training and test data
  - Create Logistic Regression/Support Vector Machine/Decision Tree Classifier/K Nearest Neighbors objects
- **Evaluation:**
  - Calculate the accuracy on the test data using the method score
  - Create and examine the Confusion matrix
- **Improvement:**
  - Create a GridSearchCV object to find best parameters
- **The best performing Classification Model:**
  - Find the model with the best accuracy score

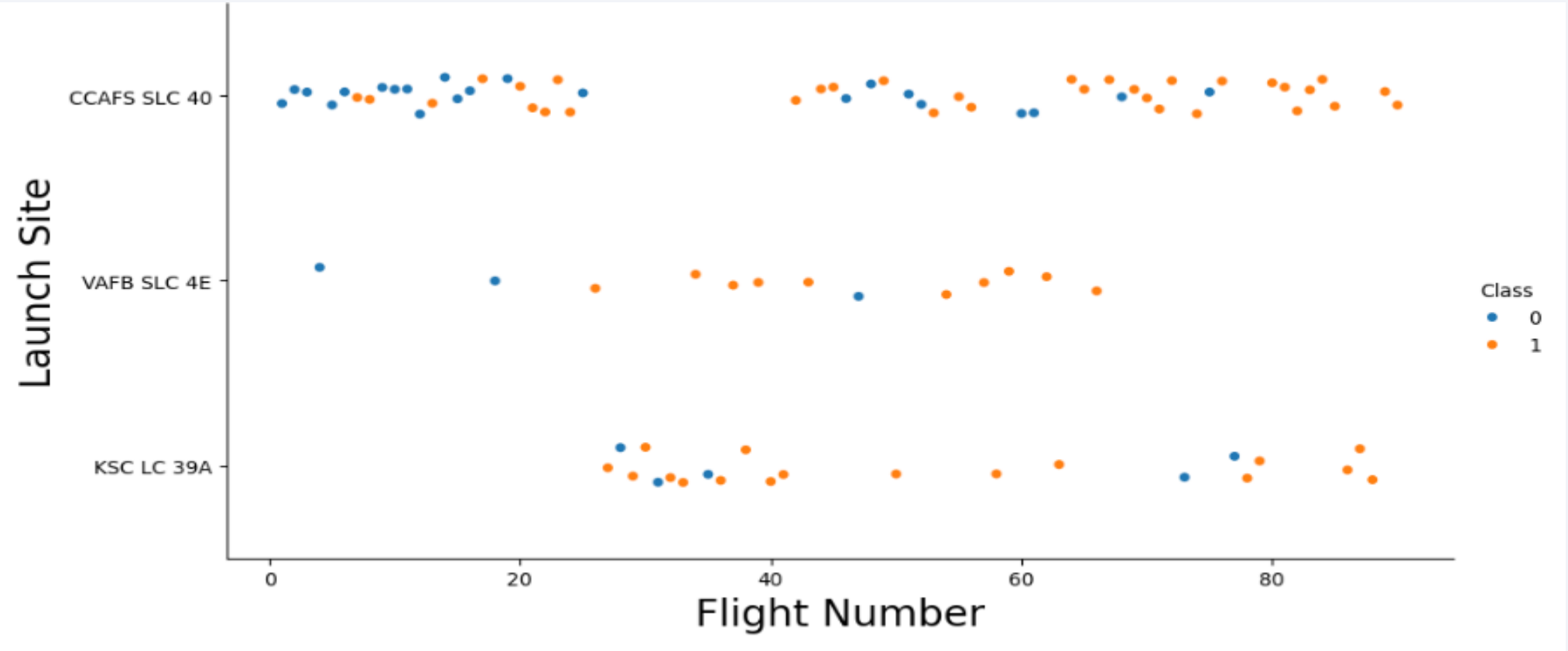
Guide: Github Machine Learning Prediction

Results

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

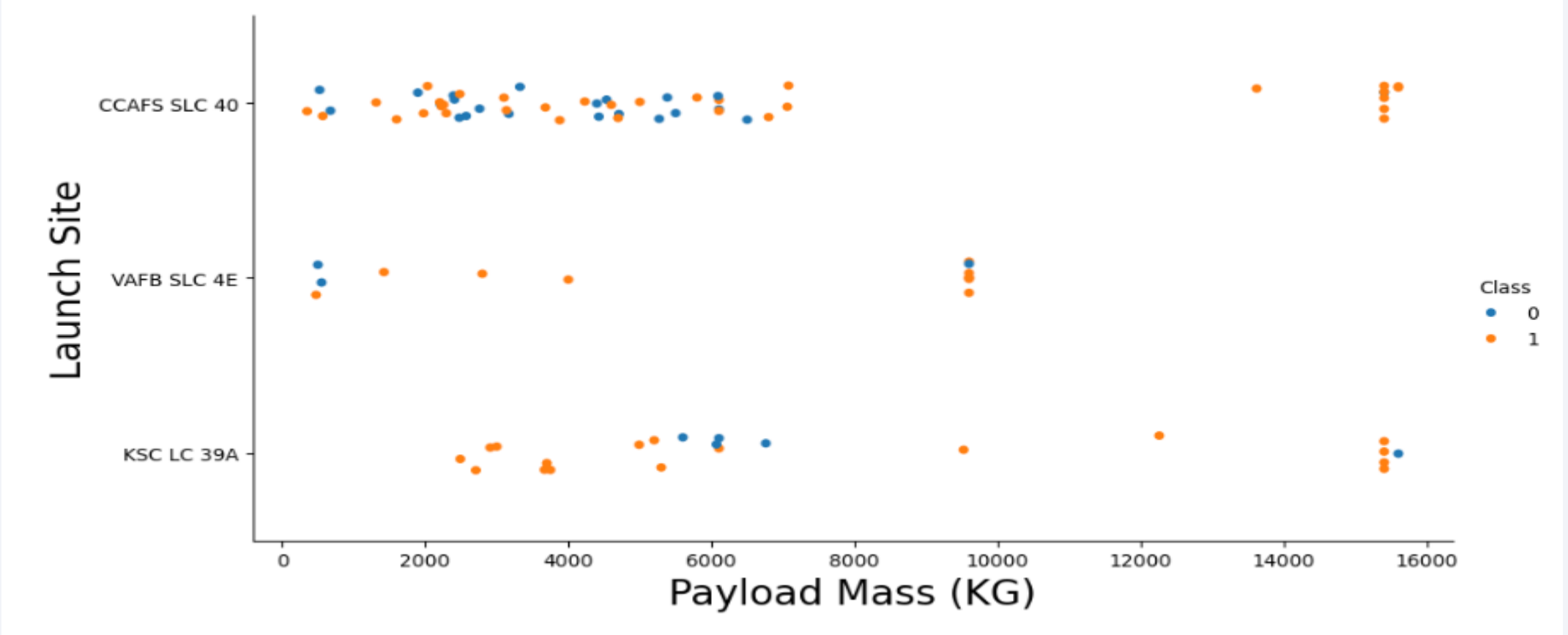

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



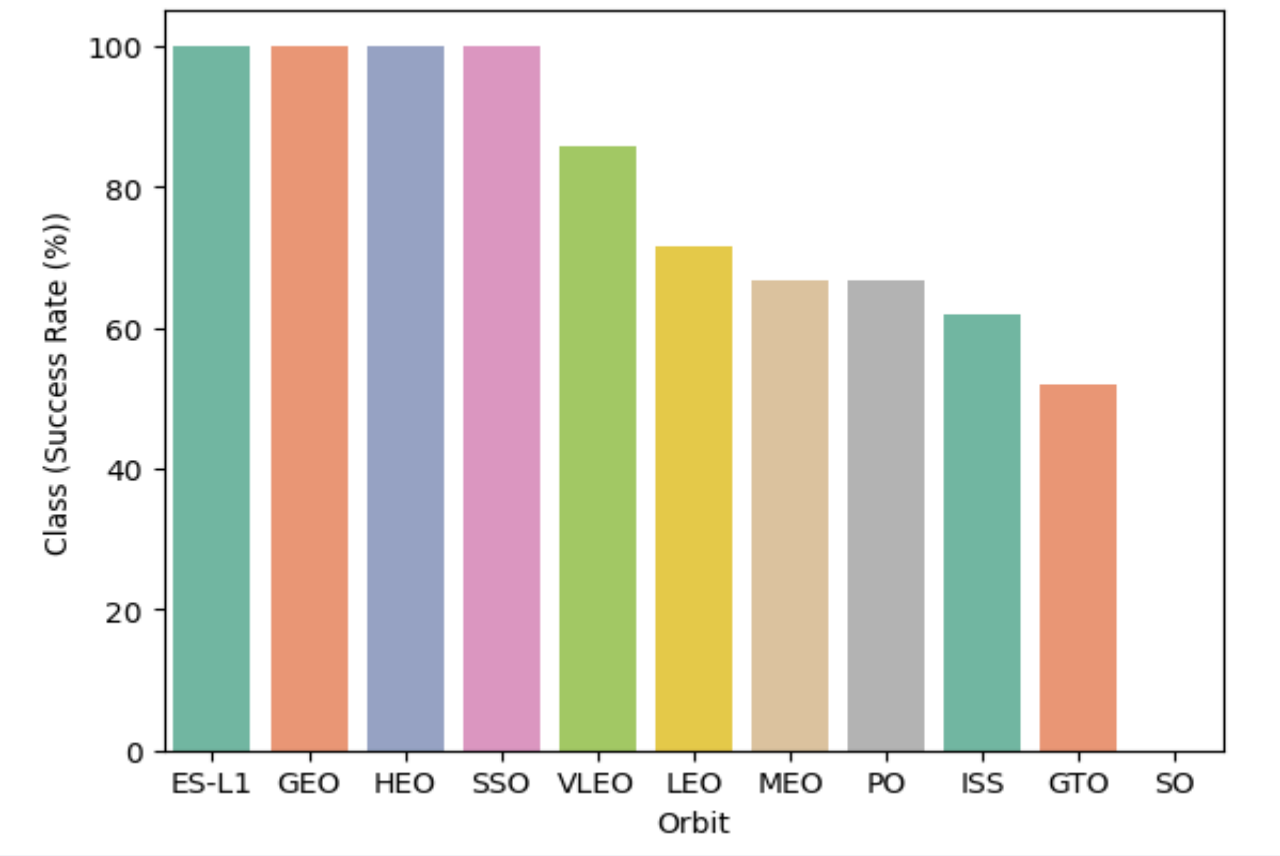
Flight Number vs Launch Site: As the flight number increases in each of the 3 launch sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after the Flight number 50. Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80th flight.

Payload vs. Launch Site



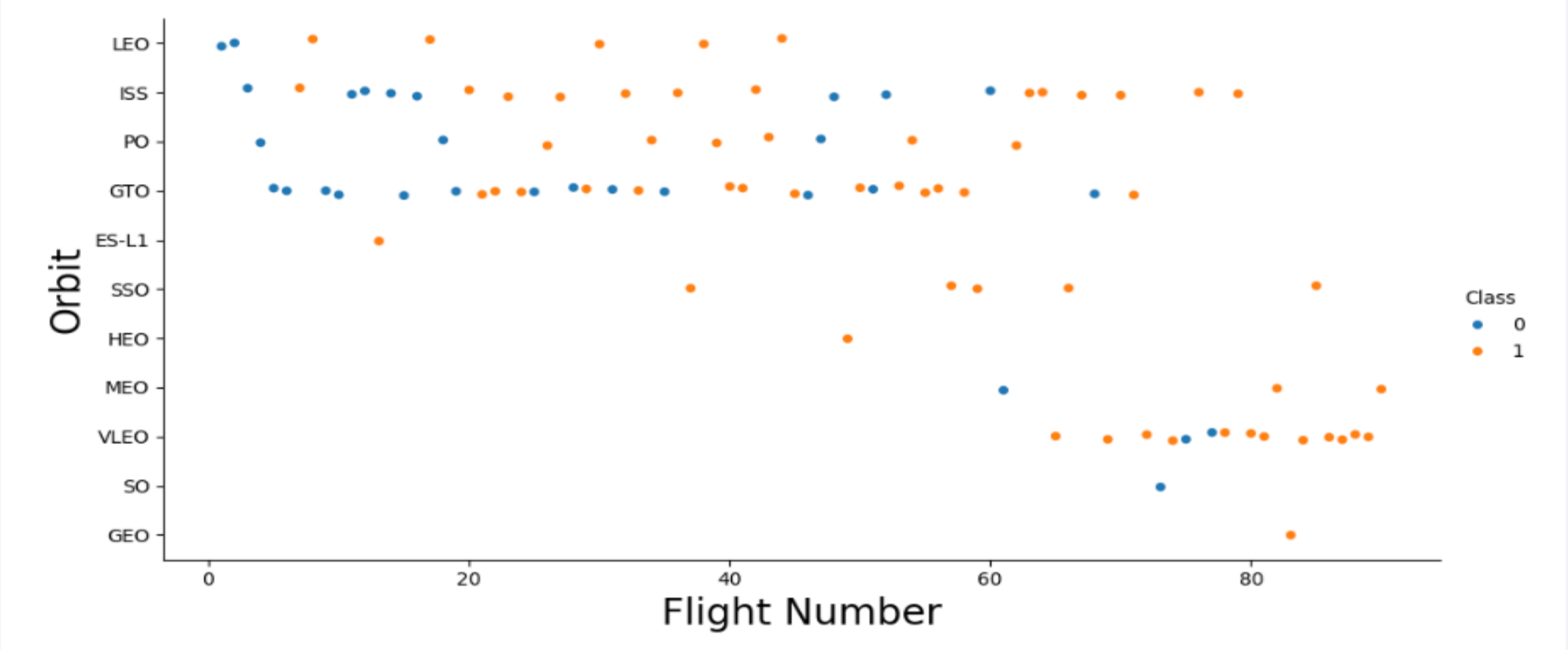
Payload Mass vs Launch Site: for the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).

Success Rate vs. Orbit Type



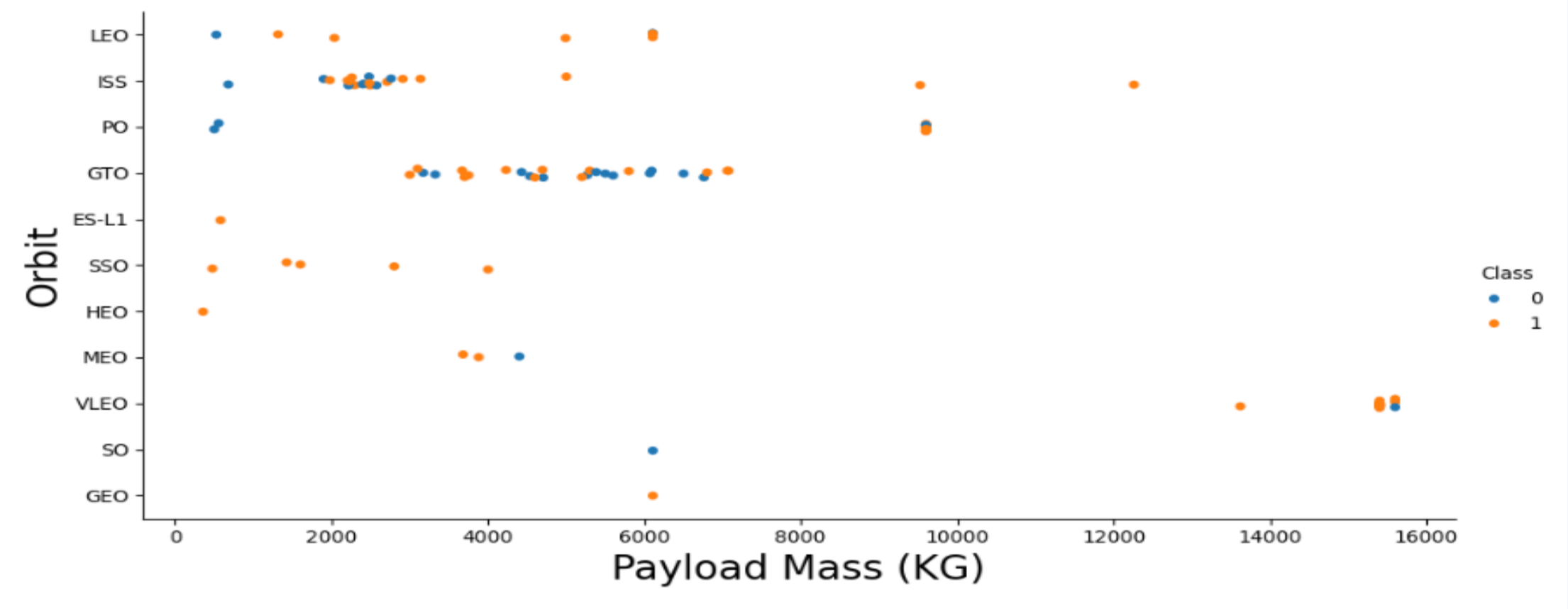
Orbit vs Success Rate: Orbits ES-L1, GEO, HEO and SSO have the highest success rates at 100%, GTO orbit has the lowest success rate at ~50%. Orbit SO has 0% success rate.

Flight Number vs. Orbit Type



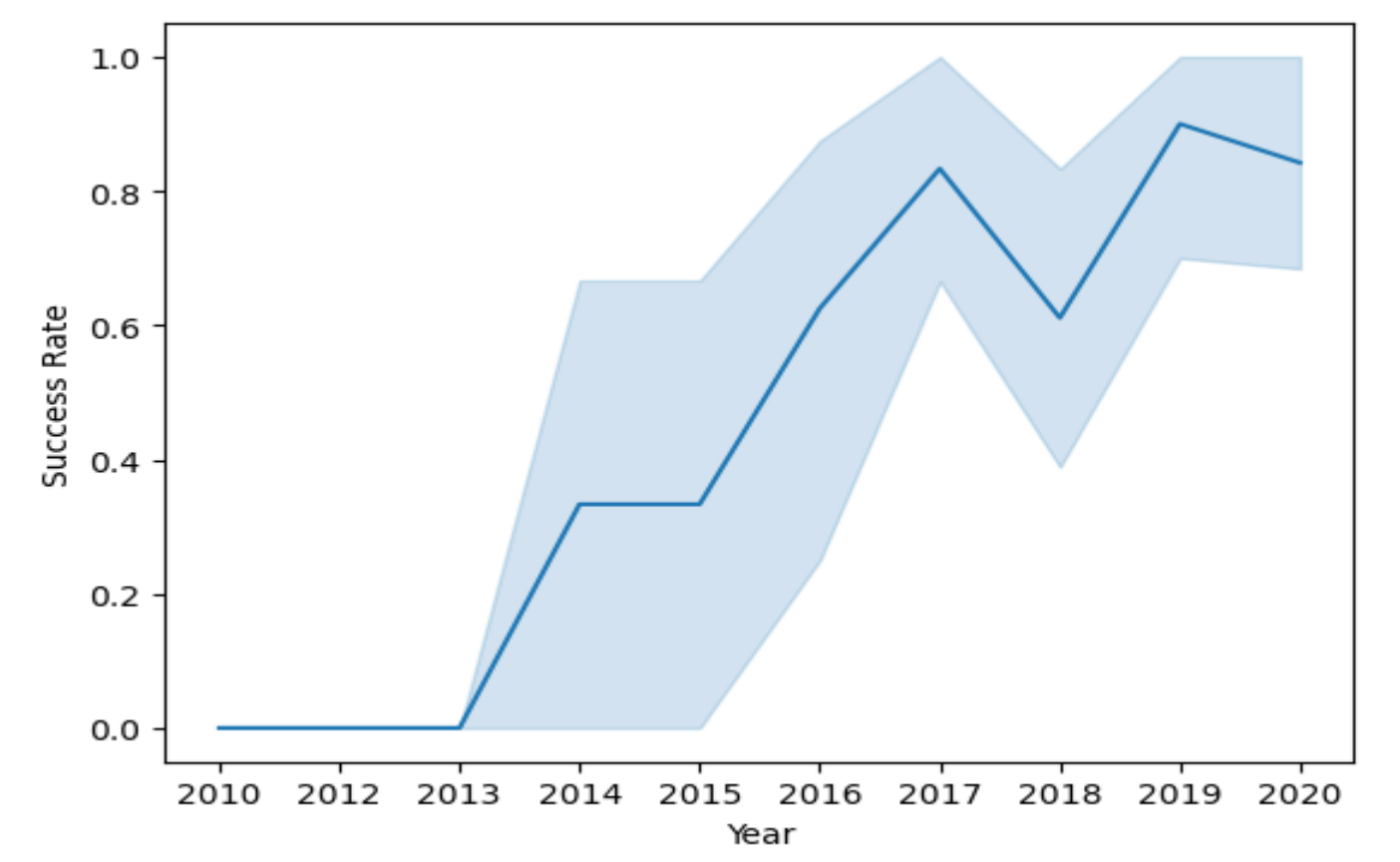
Flight Number vs Orbit: in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

Payload vs. Orbit Type



Payload Mass vs Orbit: 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



Year vs Success Rate: success rate since 2013 kept increasing till 2020, with a drop in 2018.

All Launch Site Names

- **Unique Launch Sites:**

CCAFS LC-40

VAFB SLC-4E

KSC LC-39A

CCAFS SLC-40

- **Query:**

```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTBL;
```

Only the distinct values of the Launch Site columns are displayed.

Launch Site Names Begin with 'CCA'

5 records where launch sites begin with the string ‘CCA’:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Query:

```
%sql SELECT * FROM SPACEXTBL WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

Only 5 values are shown, whose Launch Site column begins with 'CCA'.

Total Payload Mass

- **Total payload mass carried by boosters launched by NASA (CRS):**

NASA (CRS): 45596 KG

- **Query:**

%sql SELECT sum(PAYLOAD_MASS__KG_) AS "Total_Payload_Mass", Customer FROM SPACEXTBL WHERE Customer='NASA (CRS)';

The dataset is filtered in order to sum the payload mass data related to the Customer Nasa.

- Result

```
In [13]: %sql SELECT sum(PAYLOAD_MASS__KG_) AS "Total_Payload_Mass", Customer FROM SPACEXTBL WHERE Customer='NASA (CRS)';
 * sqlite:///my_data1.db
Done.
```

Out[13]:

Total_Payload_Mass	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

- **Average payload mass carried by booster version F9 v1.1:**

F9 v1.1: 2928.4 KG

- **Query:**

%sql SELECT avg(PAYLOAD_MASS__KG_) AS "Avg_Payload_Mass", "Booster_Version" FROM SPACEXTBL WHERE "Booster_Version"='F9 v1.1';

The dataset is filtered in order to sum the payload mass data related to the Booster Version F9.

- **Result**

```
In [15]: %sql SELECT avg(PAYLOAD_MASS__KG_) AS "Avg_Payload_Mass", "Booster_Version" FROM SPACEXTBL WHERE "Booster_Version"='F9 v1.1
 * sqlite:///my_data1.db
Done.
```

Out[15]:

Avg_Payload_Mass	Booster_Version
2928.4	F9 v1.1

First Successful Ground Landing Date

- **The date when the first successful landing outcome in ground pad was achieved:**

First Success (ground pad): 2015-12-22

- **Query:**

%sql SELECT MIN(DATE), "Landing_Outcome" FROM SPACEXTBL WHERE "Landing_Outcome"='Success (ground pad)';

The dataset is filtered in order to find the first successful landing outcome on ground pad .

- **Result:**

```
In [17]: %sql SELECT MIN(DATE), "Landing_Outcome" FROM SPACEXTBL WHERE "Landing_Outcome"='Success (ground pad)';
 * sqlite:///my_data1.db
Done.
```

Out[17]:

MIN(DATE)	Landing_Outcome
2015-12-22	Success (ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000

Names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000:

- **Query:**

```
%sql SELECT DISTINCT "Booster_Version", PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE
"Landing_Outcome"='Success (drone ship)' AND PAYLOAD_MASS__KG_>4000 AND PAYLOAD_MASS__KG_<6000;
```

The dataset is filtered to select the Boosters with payload mass between 4000 and 6000 Kg and that landed successfully on a drone ship.

- **Result:**

```
In [18]: %sql SELECT DISTINCT "Booster_Version", PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE "Landing_Outcome"='Success (drone ship)' AND
 * sqlite:///my_data1.db
Done.
```

Out[18]:

Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

Total number of successful and failure mission outcomes

Successful Mission Outcomes: 100

Failure Mission Outcomes: 1

- **Query:**

```
%sql SELECT(SELECT COUNT("Mission_Outcome") FROM SPACEXTBL WHERE "Mission_Outcome" LIKE 'Success%')
AS "Successful_Mission_Outcomes", (SELECT COUNT("Mission_Outcome") FROM SPACEXTBL WHERE
"Mission_Outcome" LIKE 'Failure%') AS "Failure_Mission_Outcomes";
```

Successful and Failure mission outcomes are counted from the column Mission Outcome

- **Result:**

```
In [19]: %sql SELECT(SELECT COUNT("Mission_Outcome") FROM SPACEXTBL WHERE "Mission_Outcome" LIKE 'Success%') AS "Successful_Mission_(
 * sqlite:///my_data1.db
Done.
```

Out[19]:

Successful_Mission_Outcomes	Failure_Mission_Outcomes
100	1

Boosters Carried Maximum Payload

The names of the booster which have carried the maximum payload mass

- **Query:**

```
%sql SELECT "Booster_Version", "Payload" FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_=(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

The subquery finds the maximum payload mass, the query filters the names of the Booster Versions and Payload corresponding to the max mass.

- **Result:**

```
In [20]: %sql SELECT "Booster_Version", "Payload" FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_=(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXT
 * sqlite:///my_data1.db
Done.
```

Out[20]:

Booster_Version	Payload
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21

2015 Launch Records

The records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015:

- **Query:**

%sql SELECT SUBSTR(Date,6,2), "Booster_Version", "Launch_Site", "Landing_Outcome" FROM SPACEXTBL WHERE "Landing_Outcome"='Failure (drone ship)' AND SUBSTR(Date,0,5)='2015';

The dataset is filtered to find the month, booster version and launch site for the failure landing outcomes in drone ship for the year 2015.

- **Result:**

```
In [21]: %sql SELECT SUBSTR(Date,6,2), "Booster_Version", "Launch_Site", "Landing_Outcome" FROM SPACEXTBL WHERE "Landing_Outcome"='Fa
 * sqlite:///my_data1.db
Done.
```

Out[21]:

SUBSTR(Date,6,2)	Booster_Version	Launch_Site	Landing_Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- **Query:**

```
%sql SELECT "Landing_Outcome", COUNT("Landing_Outcome") AS "Landing_Outcomes_Count" FROM SPACEXTBL WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY "Landing_Outcome" ORDER BY "Landing_Outcomes_Count" DESC;
```

Each 'typology' of landing outcome is listed and ranked in descending order based on the landing outcomes count. Only landing that have taken place between two specific date have been taken into account.

- **Result:**

```
In [22]: %sql SELECT "Landing_Outcome", COUNT("Landing_Outcome") AS "Landing_Outcomes_Count" FROM SPACEXTBL WHERE "Date" BETWEEN '20:
 * sqlite:///my_data1.db
Done.
```

Out[22]:

Landing_Outcome	Landing_Outcomes_Count
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

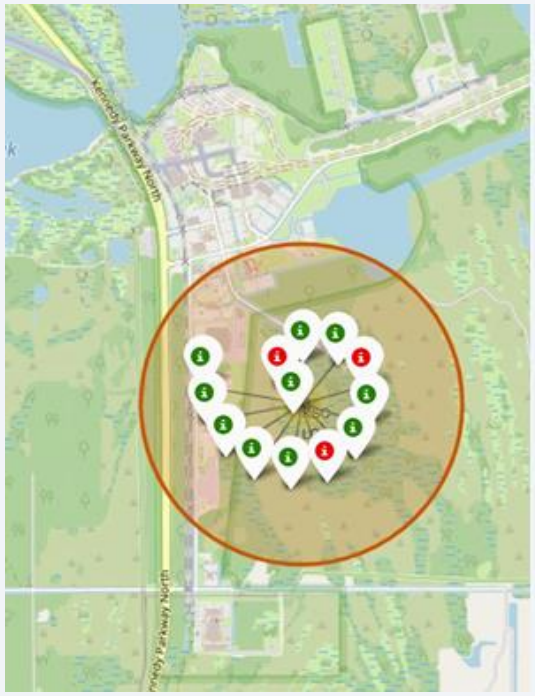
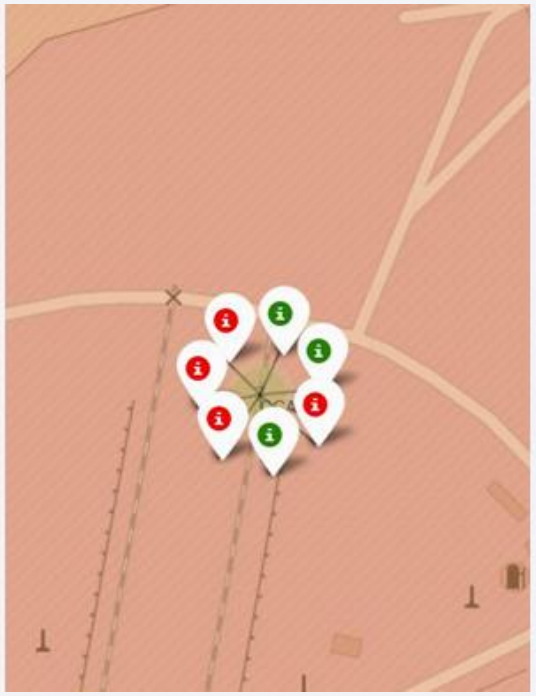
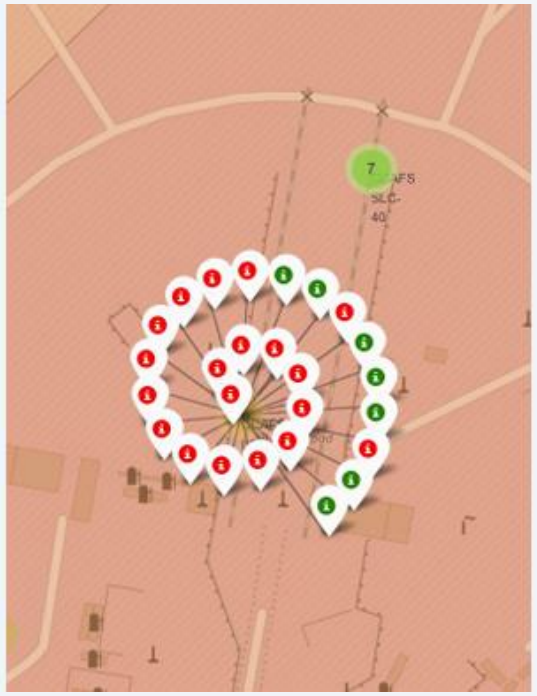
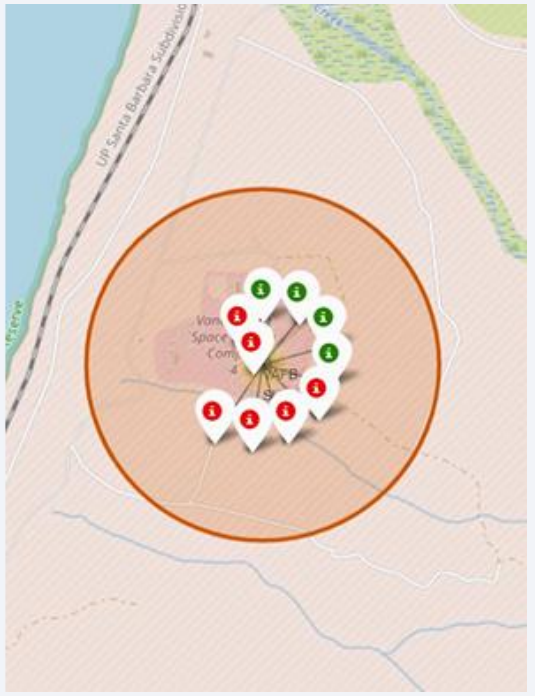
Launch Sites Proximities Analysis

All launch sites on global map

- • All launch sites are in proximity to the Equator
- • All launch sites are in very close proximity to the coast

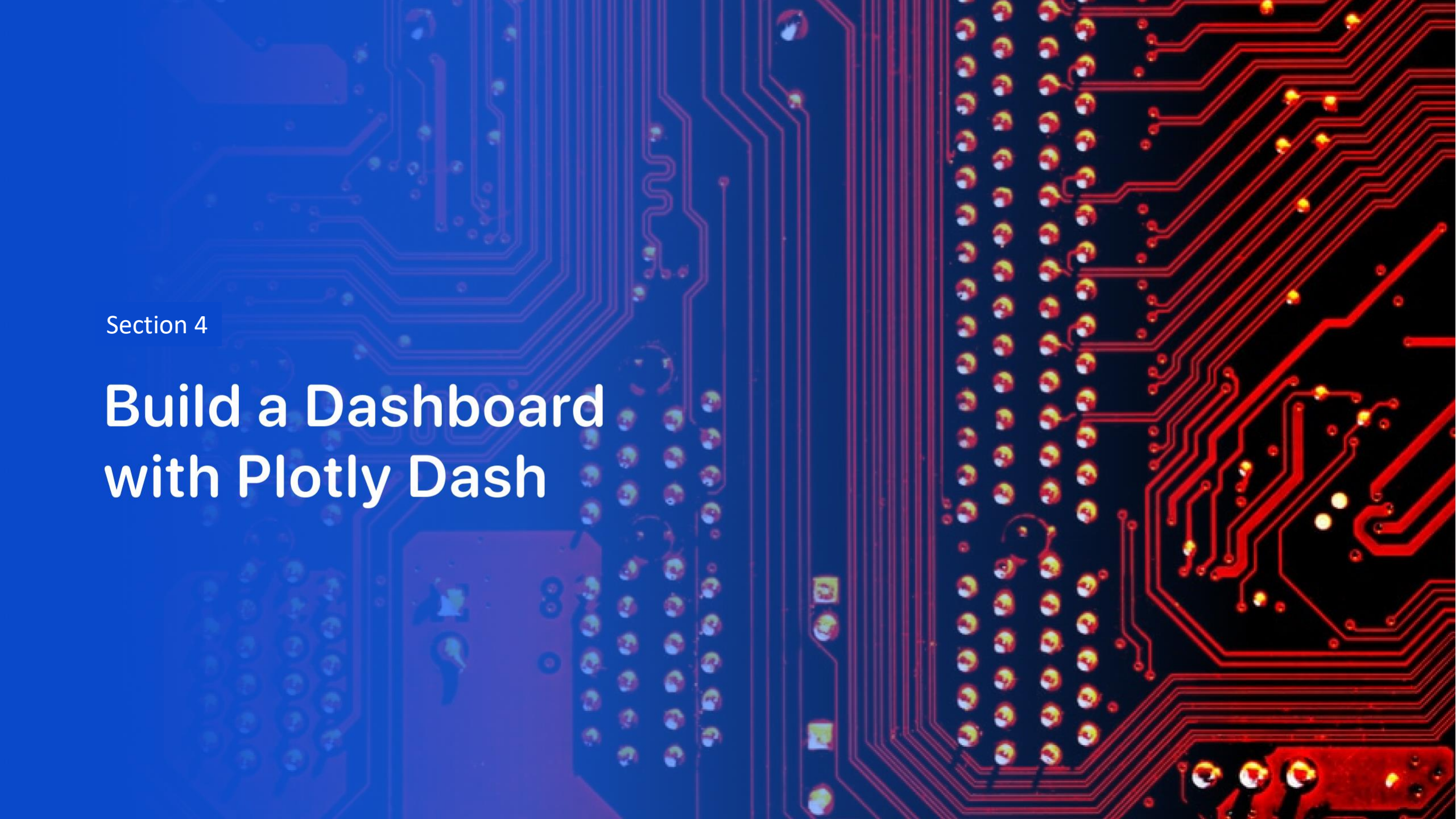
Success/Failed Launches Map

- If a launch was successful (class=1), then we used a green marker and if a launch was failed, we used a red marker (class=0), in this way it is easy to understand the success rate of a specific site.

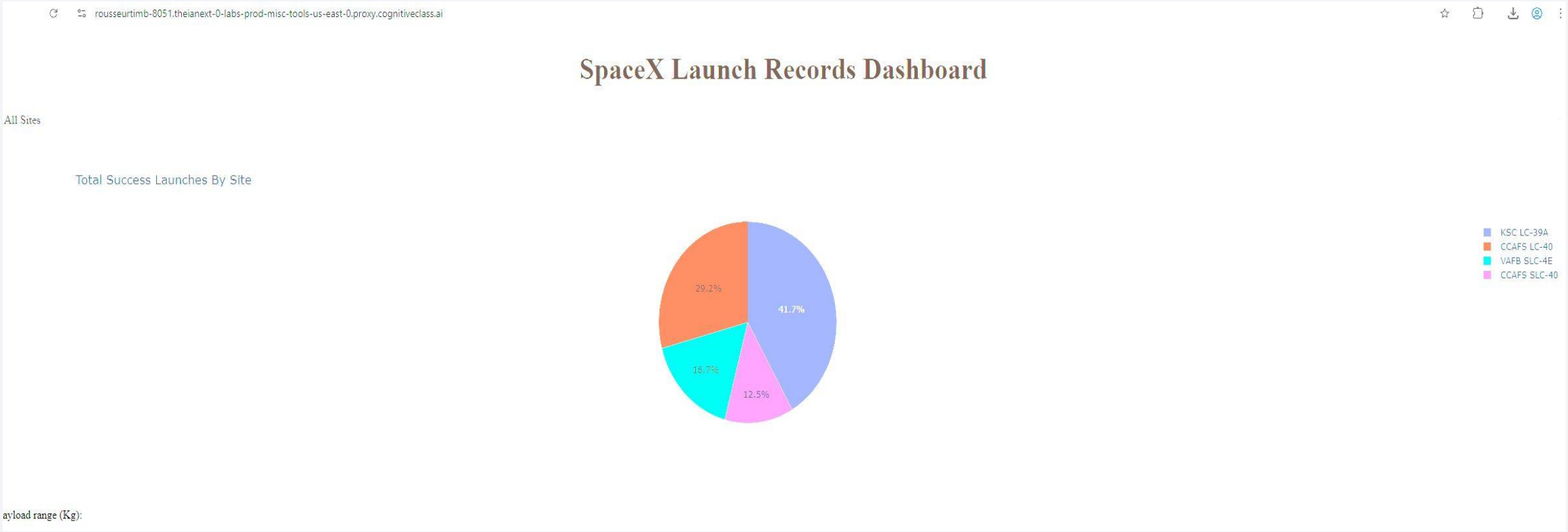
Distance between a launch site and its proximities

- 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

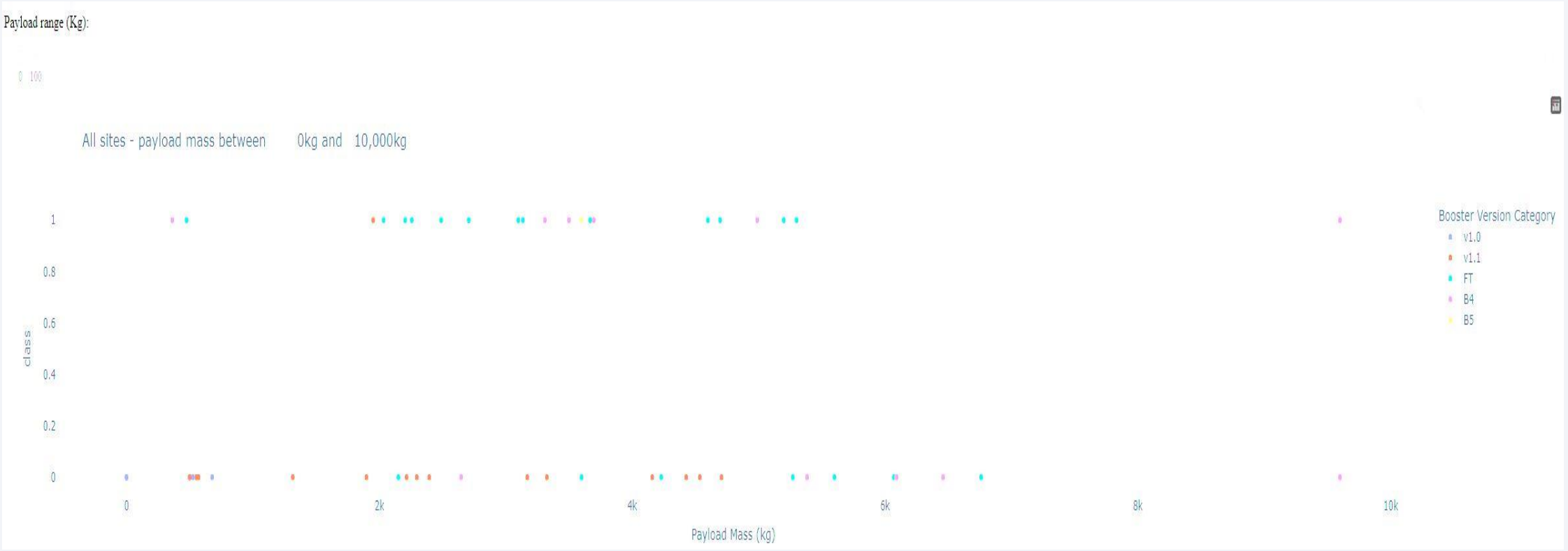
Dashboard Graph: launch success rate for all sites

- The site KSC LC-39A has the best success rate, followed by CCAFS LC-40, VAFB SLC-4E and CCAFS SLC-40.



Payload vs Launch Outcome, all sites, 0-4k KG

Lighter payload launches are more frequent and more successful, especially with the FT booster version.



Payload vs Launch Outcome, all sites, 2k-10k KG

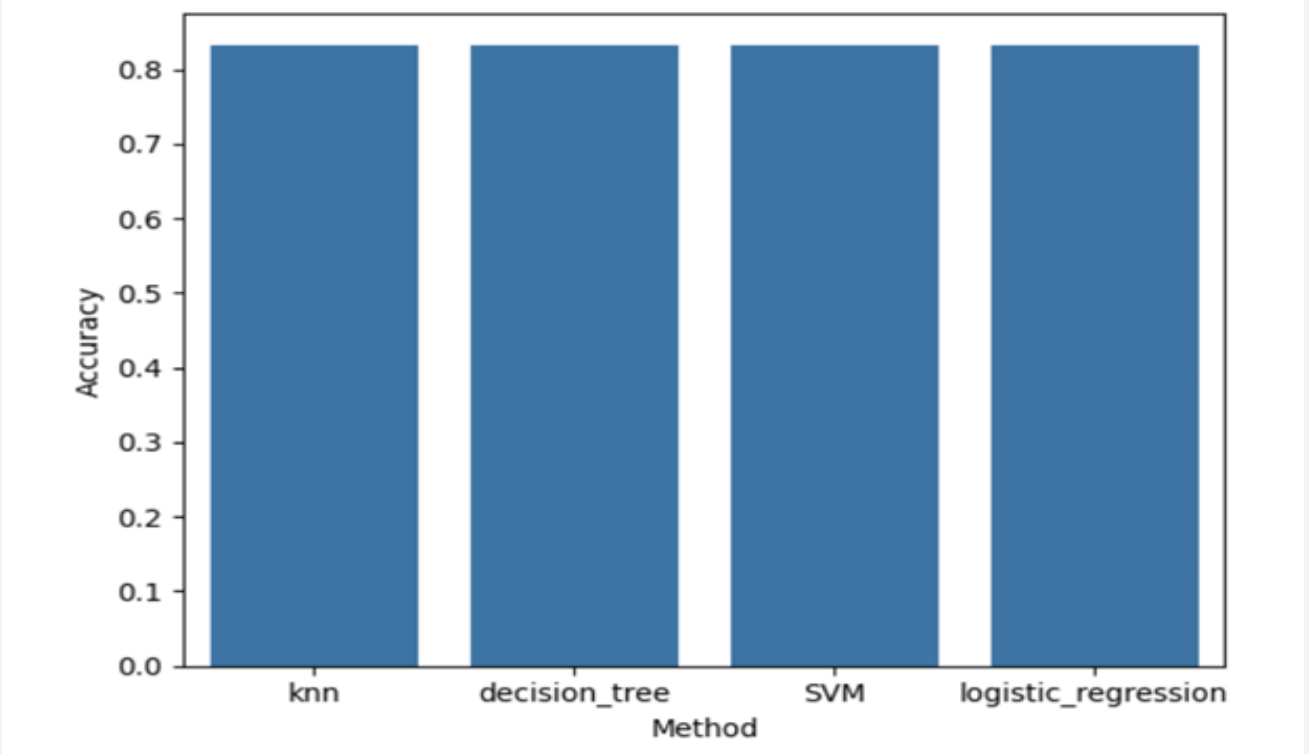
- Heavier payload launches are less frequent and less successful, and are made with fewer booster versions. Version FT is the most successful with heavier payload mass.

Section 5

Predictive Analysis (Classification)

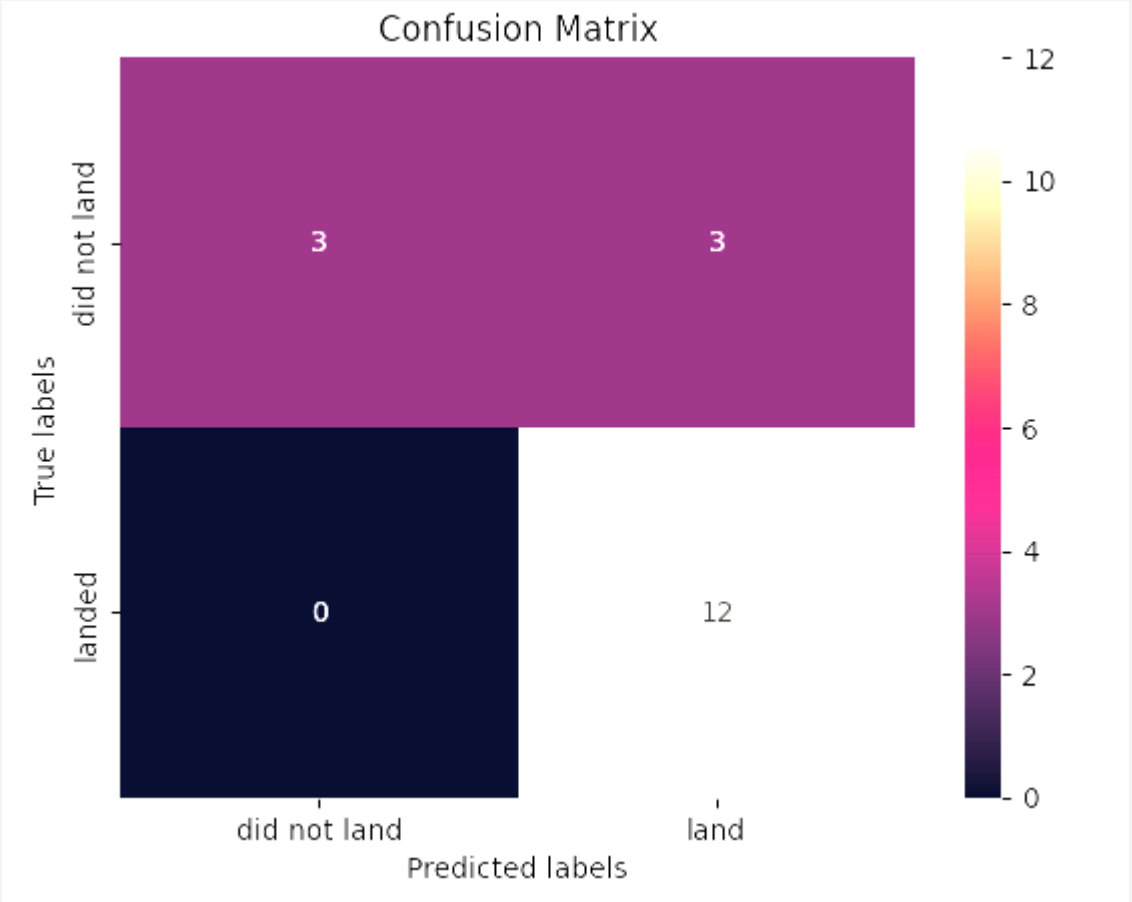
Classification Accuracy

- Accuracy is the same for all the models.
- Accuracy Score on the test data is the same for all the classification algorithms based on the data set with a value of .8333
- Given that the Accuracy scores for Classification algorithms are very close and the test scores are the same, we may need a broader data set to further tune the models



Confusion Matrix

- True Positive True Negative- 12 (True label is landed, Predicted label is also landed)
- False Negative False Positive- 3 (True label is did not land, Predicted label is also did not land)
- False Negative– 0 (True label is landed, Predicted label is did not land)
- False Positive- 3 (True label is not landed, Predicted label is landed)
- Overall, the classifier is correct about 83% of the time ((TP + TN) / Total) with a misclassification or error rate ((FP + FN) / Total) of about 16.5%
    - Accuracy= (12+3)/18= 0.83



Conclusions

- **The accuracy is 83% that means that the models are correct 83 times out of 100.**
- **Success rates appear go up as Payload increases but there is no clear correlation between Payload mass and success rates**
- **Precision is equal to 80% that means that the positive predictions are correct 80 times out of 100.**
- **The best performing Machine Learning Classification Model is the Decision Tree with an accuracy of about 87.5%. When the models were scored on the test data, the accuracy score was about 83% for all models. More data may be needed to further tune the models and find a potential better fit.**

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!