Winning Space Race with Data Science

Edward Richard Airs
December 2021

https://github.com/EdwardAirs/CapStoneMaster.git

Outline

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

Executive Summary

- Summary of methodologies: I used the Space X API to retrieve the data. Using data wrangling, including sorting and filtering, I performed exploratory data analysis to understand the data, visualize the correlations with Matplotlib, as well as Folium and building a Plotly Dashboard. To then use machine learning to find the best model fit on our data set with the highest outcome.
- Summary of all results: after constructing various graphs, interactive maps and applying several machine learning models we found the best model to fit our data set was a tree classifier giving us a precision of circa 89%.

Introduction

- Project background and context: We aim to seek a Learning Machine model to predict the success fullness of the landing outcome of the Space X rocket, thus to determine the cost a single launch.
- Problems you want to find answers: if a Space X rocket, with certain Pay Load, for a certain orbit (intended mission), landing in a determined landing pad, will be successful or not? What will be the percentage of correctness?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - The data set was collected using the public Space X API and the Coursera prepared API
- Perform data wrangling
  - The data set was created using pandas Data Frame methods, filtering and
- Perform exploratory data analysis (EDA) using visualization and SQL
  - The Explanatory Data Analysis was executed to discover patterns and correlations between the data.
- Perform interactive visual analytics using Folium and Plotly Dash
  - Using at first Folium and the Plotly Dash to build interactive visualizations of the data set.
- Perform predictive analysis using classification models
  - It was used different machine learning techniques to discover the highest percentage of good outcome given a test set from the original data set tested on undiscovered part of the data set.

Data Collection

At first we discovered the json data set given by the Space X API.

Then it was given a pre-prepared data set to have consistency of results across the students.

Following the steps followed:

1) The process to collect the data will use a request to download the data from an online api directly from coursera.
2) The data set then was normalized from a json file to a pandas Data Frame
3) The data frame was then filtered based on wanted columns of information's using declared functions, using global variable to then build a dictionary.
4) The dictionary then was filtered and wrangled for missing information, the payload mass information, which were filled with the mean of the payloads.
5) In the end the process was saved as a CSV file to be used later in other applications.

Data Collection – SpaceX API

- Importing the request library to dialog with the API request
- Set a variable “spacex_url” to store the URL of the Space X API
- Used the variable response with the method request.get() to download and store the json data from the API.
- To visualize the content used the print function of the method .content()

```
import requests
```

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

```
response = requests.get(spacex_url)
```

```
print(response.content)
```

Git Hub link for Space X data collection: [001-Data-Collection.ipynb]

Data Collection - Scraping

The same method was used to collect the data set provided by Coursera to make the results more consistent:

- Using the same library “requests”.
- Set a variable “static_json_url” to store the URL of the Coursera Space X API
- Verified response with the method .status_code to verify the good connection
- Importing from the pandas library the function json_normalize()
- Then used the pandas normalizing function for transforming it in a data frame set

Git Hub link for Space X data collection: 001-Data-Collection.ipynb

```
import requests
```

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

```
response.status_code
```

```
from pandas import json_normalize
```

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

Data Wrangling

In the Data Wrangling process we discover more insights of the data set provided.

At first I calculated the percentage of the missing values in each attribute with the functions of isnull(), .sum() and .count() as following:

```
df.isnull().sum()/df.count()*100
```

Revealing a 40.53% of data missing from the “LandingPad” column:

To understand the data types at hand with the function .dtypes:

```
df.dtypes
```

Using the method .value_counts() to determine the number of each orbit:

```
df['Orbit'].value_counts()
```

Then transforming the column “Outcome” to a binary value in the new column ‘Class’:

```
 for i in df['Outcome']:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
df['Class']=landing_class
```

To calculate the success rate resulting in a 0.66 mean value:

```
 df["Class"].mean()
```

Git Hub link: [Data Wrangling.ipynb]

EDA with Data Visualization

Flight Number vs Payload Mass

To view the correlation between the Flight Number and the Payload Mass success rate

Flight Number vs Launch Site

To view is the success rate of a particular Launch Site

Payload Mass vs Launch Site

To view if a certain Launch pad is directly effected by the Payload

Orbit vs Success Rate

To view if a particular Orbit would be of impact on the success rate

Flight Number vs Orbit

To view if a flight number would be influenced by a certain Orbit

Payload Mass vs Orbit

To view the correlation between Payload Mass and the specific Orbit

Year vs Success

To view the advancement of success rate during the technological advancement of Space X

GitHub Link: EDA-Pandas-Matplotlib.ipynb

EDA with SQL

- “SELECT DISTINCT Launch_Site FROM SXDB;” to list the unique launch sites.
- “SELECT * FROM SXDB WHERE Launch_Site > “CCA” LIMIT 5;” to display 5 records where launch sites begin with CCA.
- “SELECT SUM(PAYLOAD_MASS__KG_) FROM SXDB WHERE Customer = "NASA (CRS)";” to display the total mass payload in KG carried by boosters launched by NASA (CRS).
- “SELECT AVG(PAYLOAD_MASS__KG_) FROM SXDB WHERE Booster_Version = "F9 v1.1";” to display the average payload mass carried by booster version F9 V1.1.
- “SELECT MIN(Date) FROM SXDB WHERE Mission_Outcome ="Success";” to list the date when the first successful landing outcome in ground pad was achieved.
- “SELECT Booster_Version FROM SXDB WHERE Landing_Outcome = "Success (drone ship)" AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000”;” to list the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
- “SELECT COUNT(*), Mission_Outcome FROM SXDB GROUP BY Mission_Outcome”;” to list the total number of successful and failure mission outcomes.
- “SELECT Booster_Version FROM SXDB WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SXDB)”;” to

Build an Interactive Map with Folium

We created an interactive map using the library Folium to observe the geographical locations fo the launch site.

In the map we added as a reference the latitude and longitude of the head quarter for the NASA company.

We added a circle for each launch site, for each of these launch sites we added Markers (I personally choose little rockets colored Blue for successful return landing and red for failed ones) to demonstrate the possibility to list and visualize on the map the landings.

Also we observed the geographical vicinity of land marks as coastline, roads, railways, and cities to see the the distance using the Polyline function of folium from the land mark to the launch sites.

We noticed the majority of launches have a great distance from cities but a great vicinity with transportation facilities, also to notice the vicinity with the coast line. Adding these objects on a map and giving a physical visualization we can observe the need to keep a greater distance for possible human harmfulness in the likely even of a disaster and the very good connection with transportation means to facilitate transportation.

GitHub Link: Visual Analytics Folium.ipynb

Build a Dashboard with Plotly Dash

Within the Plotly Dash dashboard we have added a pie chart to render more intuitive the comparison of Failed and Successful landings determined by the selection in the drop down menus.

After we added a slide bar to select the desired Payload to be visualized in the below Scatter Plot showing correlation between Payload Mass, Class of success and color coded by the Booster Version.

This dashboard give the user the opportunity to have an interactive visualization of the exploratory data analysis at a glance to explore further possible correlations and explanations of possible failures of landings.

GitHub Link: SpaceX_Plotly_Dashboard.py

Predictive Analysis (Classification)

To implement any machine learning model at first we must standardize our data set (method .StandardScaler()).

We will split in an equivalent 80 / 20 split for training and testing (method .train_test_split()).

We will apply the model wanted with the relevant parameters with a preset grid search parameters (method .GridSearchCV()) and fit it to our train set (method .fit()).

After training the model we test it against the test set to verify the accuracy (method .best_score()) and visualize the result with a confusion matrix (method confusion_matrix()).

GitHub Link: SpaceX_Machine Learning Prediction_Part_5.ipynb

.StandardScaler()

.transform()

.train_test_split()

.fit()

.best_score()

Results

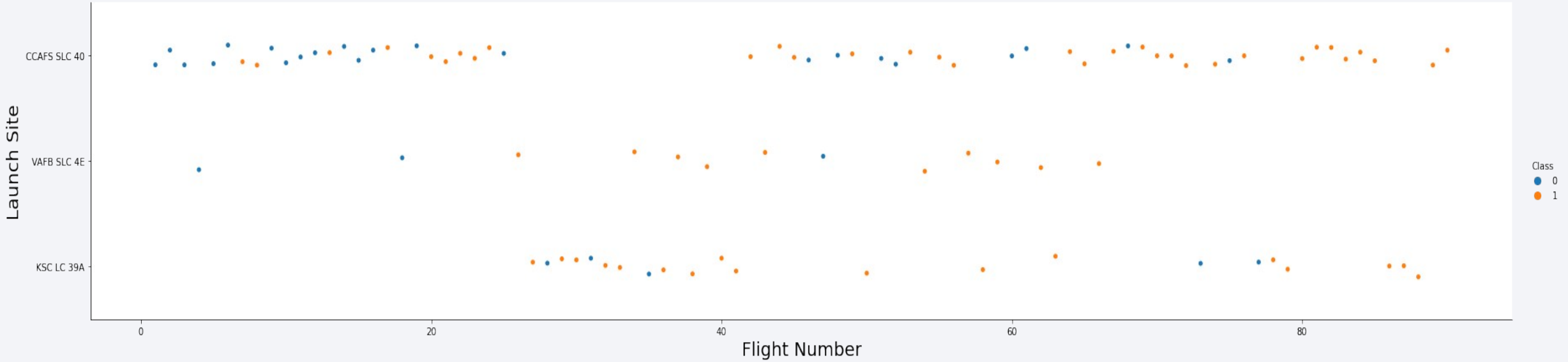
- Exploratory data analysis results:
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

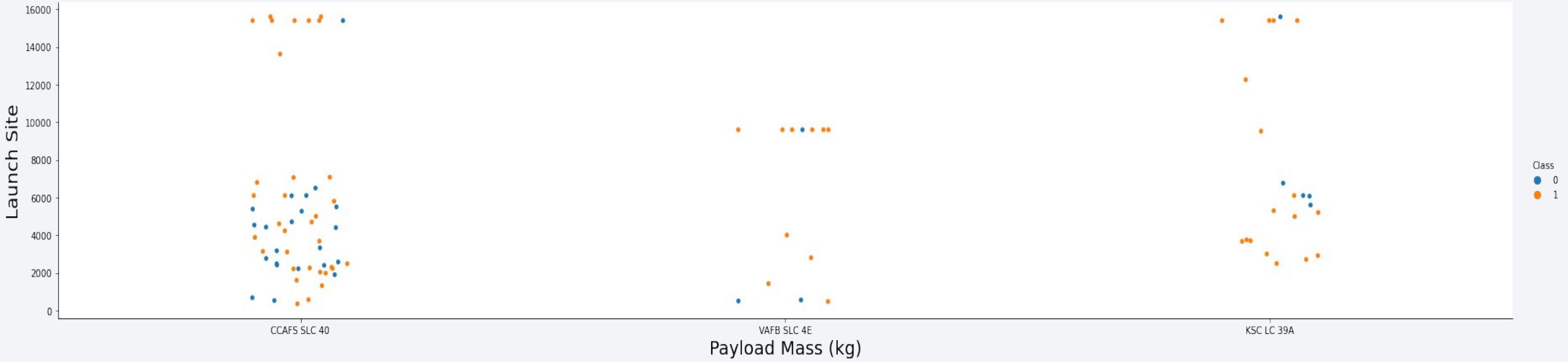
Flight Number vs. Launch Site

- Scatter Plot of Flight Number vs. Launch Site, indexed 0 = Fail and 1 = Success
- We see over time the test flights have more success rate, site CCAFS SLC 40 was the most tested and KSC LC 39A was the most successful.



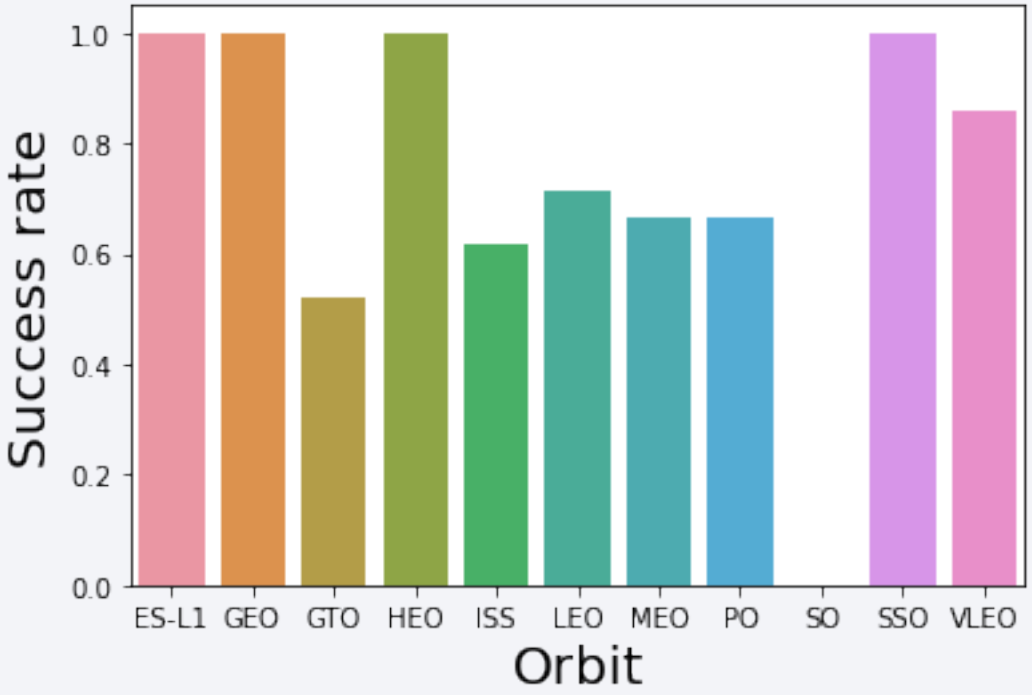
Payload vs. Launch Site

- Scatter Plot of Payload vs. Launch Site, indexed 0 = Fail and 1 = Success
- We see Site KSC LC 39A has a higher success rate across the payload spectrum, CCAFS SLC 40 had the most number of tests and highest results with the highest payloads, site VAFB SLC 4E had the highest result with payload between 8,000 to 12,000 KG.



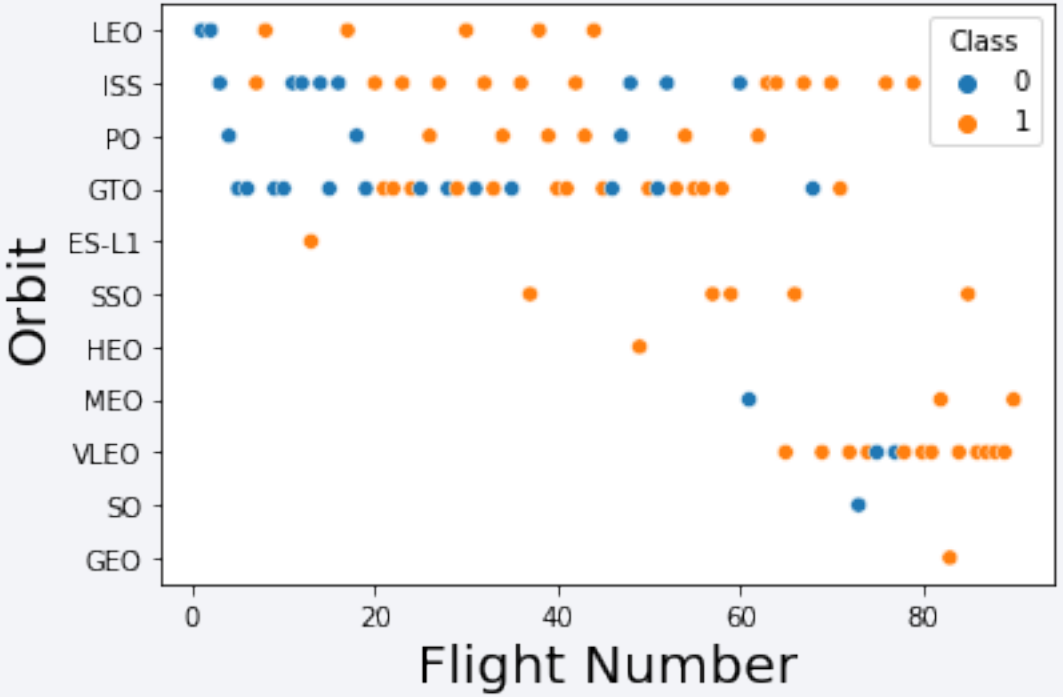
Success Rate vs. Orbit Type

- Bar Chart of success rate vs Orbit
- We see orbit SO has success rate 0, all others are as minimum of 0.5 success on average like orbit GTO, wile ISS, LEO, MEO, PO and VLEO have a success rate between 0.6 to 0.8.
- Orbit ES-L1, GEO, HEO and SSO, have almost perfect success rate of 1.



Flight Number vs. Orbit Type

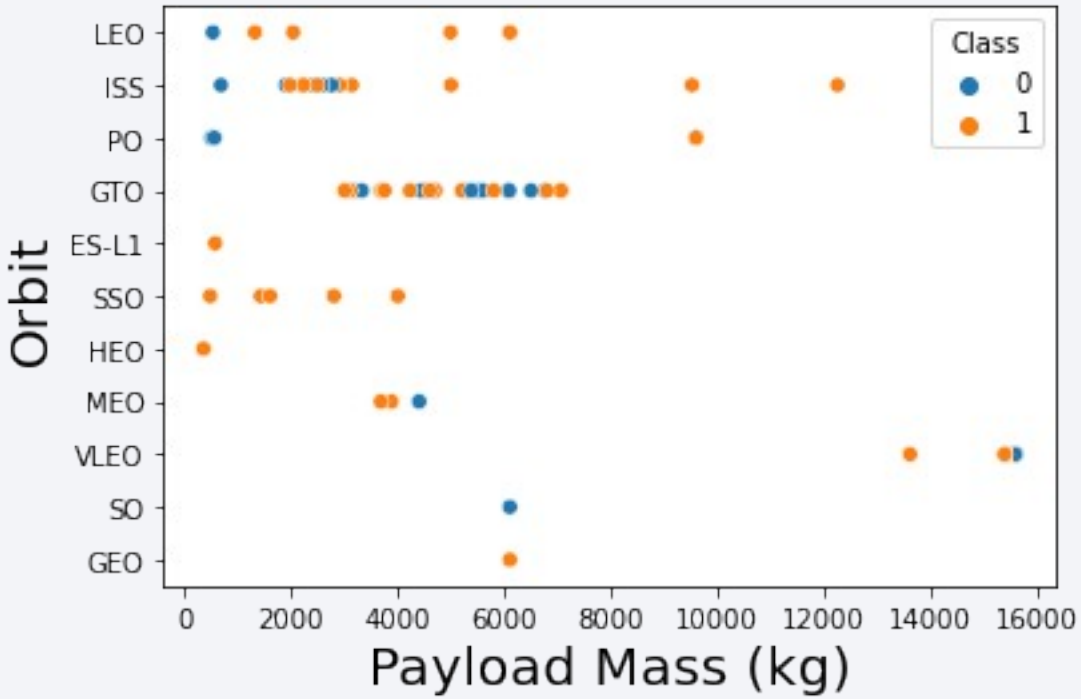
- Scatter plot of Flight Number vs Orbit, with classifier 0=Failed and 1=Success
- We can see the orbit ISS is the most frequently used, the earliest VLEO is the most successful.
- The orbit GTO seems to have the highest fail rate especially at the beginning of the flights.



Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

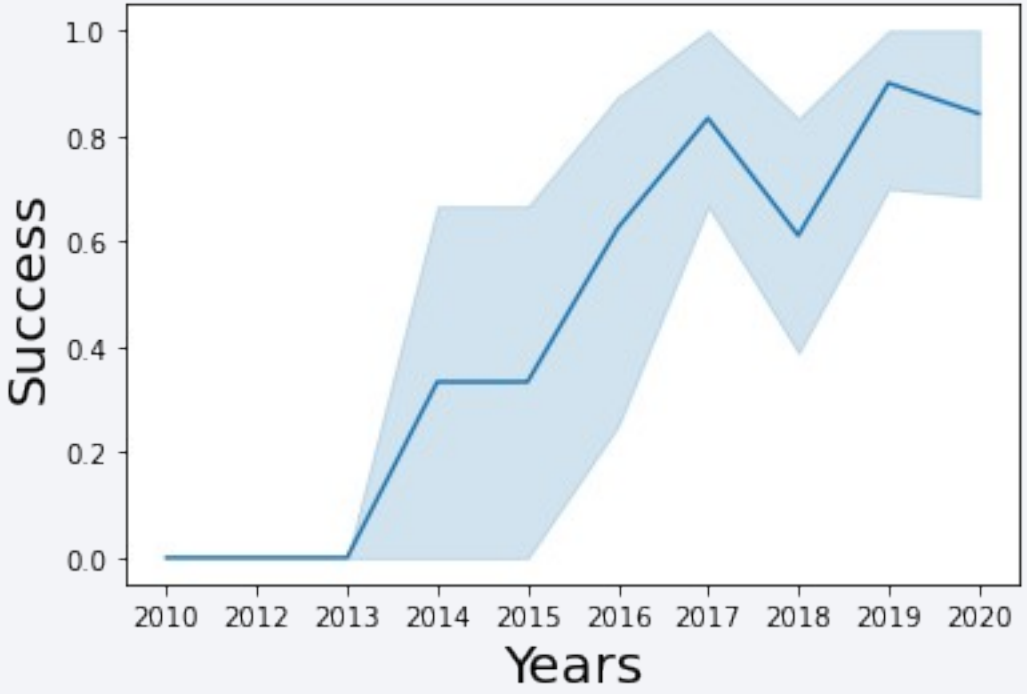
- If comparing Orbit with Payload Mass we notice I high concentration between 0 to 8,000 KG, with the highest prevalence for the Orbit ISS for payloads between 2,000 to 4,000 and Orbit GTO between 3,500 to 8,000 KG.



Launch Success Yearly Trend

- Line chart of yearly average success rate

- From year 2013 the success rate has been increasing constantly with some minor stalls between year 2014 to 2015 and set backs between 2017 to 2018.
- Overall a constant increase in reliability and successes.



Launch Site Names Begin with 'CCA'

Find 5 records where launch sites begin with `CCA`

Present your query result with a short explanation here:

Selecting only distinct results from the column Launch Site with string “CCA”.

```
result2=pd.DataFrame(c.execute('''SELECT DISTINCT Launch_Site FROM SXDB WHERE Launch_Site like "%CCA%" LIMIT 5''').fetchall())
print(result2)
```

```
              0
0   CCAFS LC-40
1  CCAFS SLC-40
```

Total Payload Mass

Calculate the total payload carried by boosters from NASA

Present your query result with a short explanation here: summing all payloads in KG from the Customer column with specific string NASA (CRS), in total 45,496 KG.

```
result3=pd.DataFrame(c.execute('''SELECT SUM(PAYLOAD_MASS__KG_) FROM SXDB WHERE Customer = "NASA (CRS)"''').fetchall())
print(result3)
```

```
       0
0  45596
```

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1

Present your query result with a short explanation here: average Payload in KG carried by the Booster F6 V 1.1 is 2,928 KG.

```
result4=pd.DataFrame(c.execute('''SELECT AVG(PAYLOAD_MASS__KG_) FROM SXDB WHERE Booster_Version = "F9 v1.1"''').fetchall())
print(result4)
```

```
        0
0  2928.4
```

First Successful Ground Landing Date

Find the dates of the first successful landing outcome on ground pad

Present your query result with a short explanation here: the first successful date a landing on a PAD was the 1st March 2013.

```
result5=pd.DataFrame(c.execute(''' SELECT MIN(Date) FROM SXDB WHERE Mission_Outcome ="Success" ''').fetchall())
print(result5)
```

```
            0
0  01-03-2013
```

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

Present your query result with a short explanation here: there are 4 booster versions that had a successful landing on a drone ship with payload between 4,000 and 6,000 KG

```
result6=pd.DataFrame(c.execute(''' SELECT Booster_Version FROM SXDB WHERE Landing_Outcome = "Success (drone ship)" AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 ''').fetchall())
print(result6)
```

```
                0
0     F9 FT B1022
1     F9 FT B1026
2  F9 FT  B1021.2
3  F9 FT  B1031.2
```

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

Present your query result with a short explanation here: there have been in total N´ 02 Failures in flight, N´ 198 Successful landings and N´ 02 successful landing with payload status unclear.

```
result7=pd.DataFrame(c.execute('''SELECT COUNT(*), Mission_Outcome FROM SXDB GROUP BY Mission_Outcome''').fetchall())
print(result7)
```

```
     0                                1
0    2              Failure (in flight)
1  196                          Success
2    2                          Success
3    2  Success (payload status unclear)
```

Boosters Carried Maximum Payload

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

Present your query result with a short explanation here: there have been 24 different boosters that had a successful landing with a max payload.

```
result8=pd.DataFrame(c.execute('''SELECT Booster_Version FROM SXDB WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SXDB)''').fetchall())
print(result8)
```

```
                0
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
12   F9 B5 B1048.4
13   F9 B5 B1049.4
14   F9 B5 B1051.3
15   F9 B5 B1056.4
16   F9 B5 B1048.5
17   F9 B5 B1051.4
18   F9 B5 B1049.5
19  F9 B5 B1060.2
20  F9 B5 B1058.3
21   F9 B5 B1051.6
22   F9 B5 B1060.3
23  F9 B5 B1049.7
```

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

Present your query result with a short explanation here: there have been N´ 05 different booster versions that had a failure to land on a pad in the year 2015.

```
result9=pd.DataFrame(c.execute('''SELECT Booster_Version, Launch_Site FROM SXDB WHERE Landing_Outcome = "Failure (drone ship)" AND Date BETWEEN "01-01-2015" AND "31-12-2015"''').fetchall())
print(result9)
```

```
              0           1
0  F9 v1.1 B1012  CCAFS LC-40
1  F9 v1.1 B1015  CCAFS LC-40
2  F9 v1.1 B1017  VAFB SLC-4E
3    F9 FT B1020  CCAFS LC-40
4    F9 FT B1024  CCAFS LC-40
```

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

Present your query result with a short explanation here: there are 9 different outcomes with the majority of being Successful, quantity 40, and adding the N´16 for success landing on a drone ship as well as N´12 on ground pads, in total we have N´68 successful landings in total between 04.06.2010 and 20.03.2017, circa 3 years..

```
result10=pd.DataFrame(c.execute(''' SELECT COUNT(*) AS count, Landing_Outcome FROM SXDB WHERE Date BETWEEN "04-06-2010" AND "20-03-2017" GROUP BY Landing_Outcome ORDER BY count DESC''').fetchall())
print(result10)
```

```
    0                     1
0  40               Success
1  20            No attempt
2  16  Success (drone ship)
3  12  Success (ground pad)
4   8  Failure (drone ship)
5   6               Failure
6   6    Controlled (ocean)
7   4   Failure (parachute)
8   2            No attempt
```

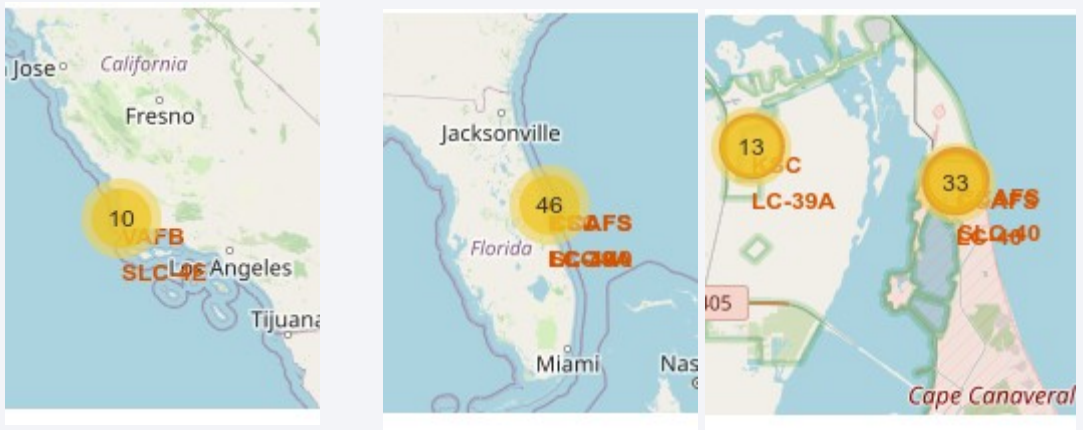
Section 4

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

Explore the generated folium map and make a proper screenshot to include all launch sites’ location markers on a global map

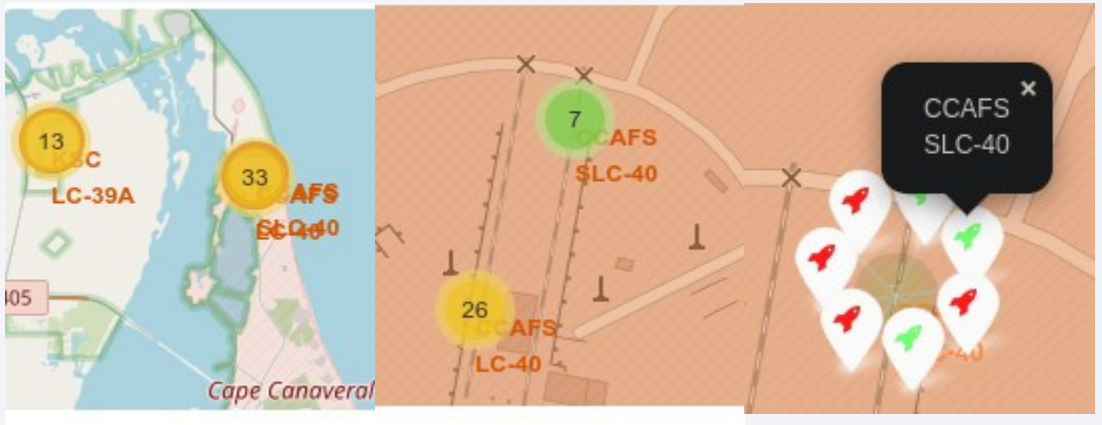
Explain the important elements and findings on the screenshot: we can see how the launches are distributed globally.



<Folium Map Screenshot 2>

Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map

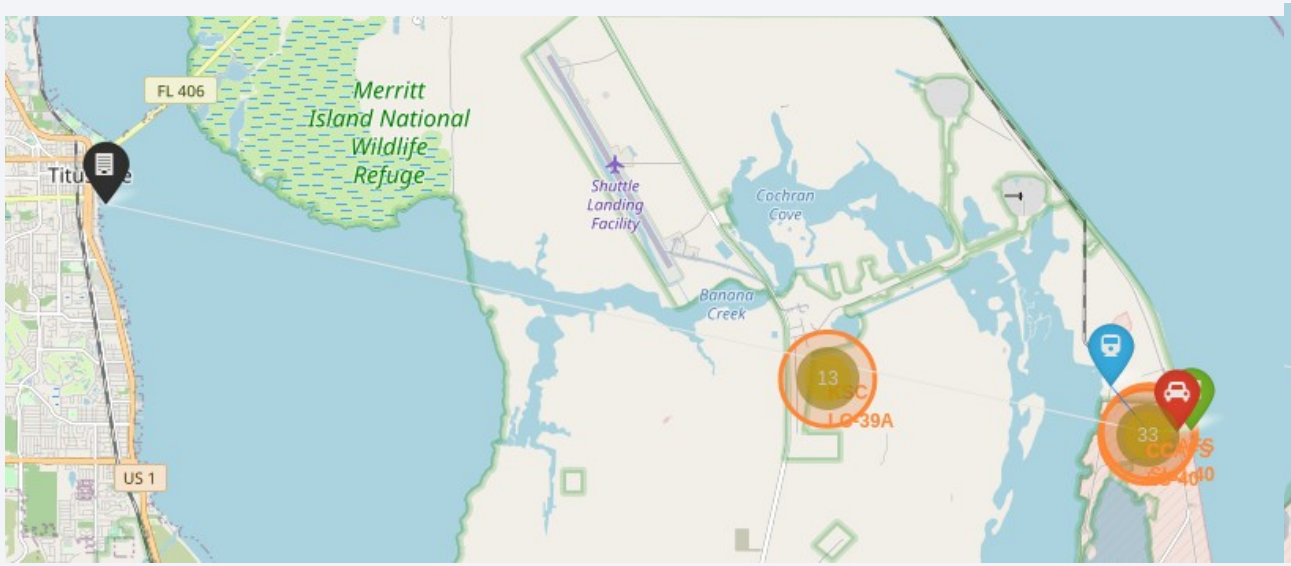
Explain the important elements and findings on the screenshot: we can visualize from each site the out comes of each launch site (Green rockets for Success and Red rockets for Failed).

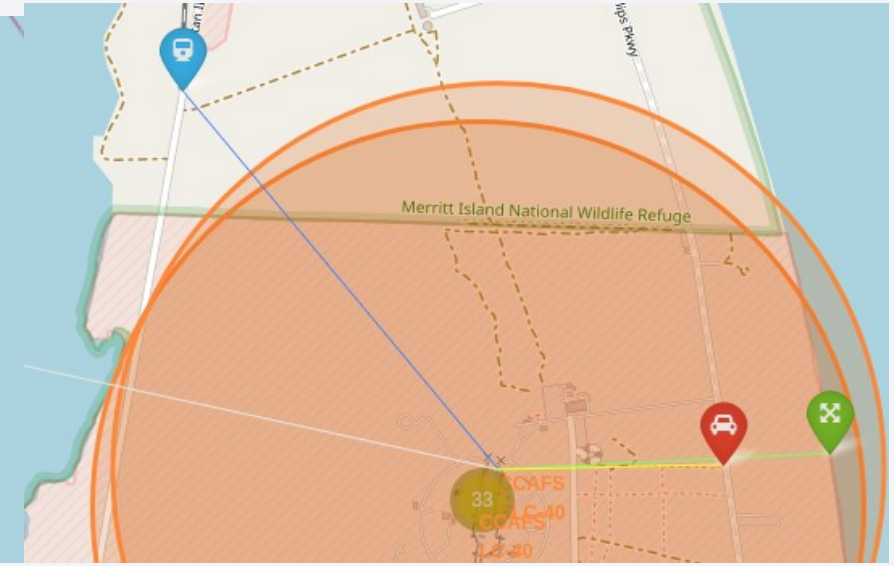


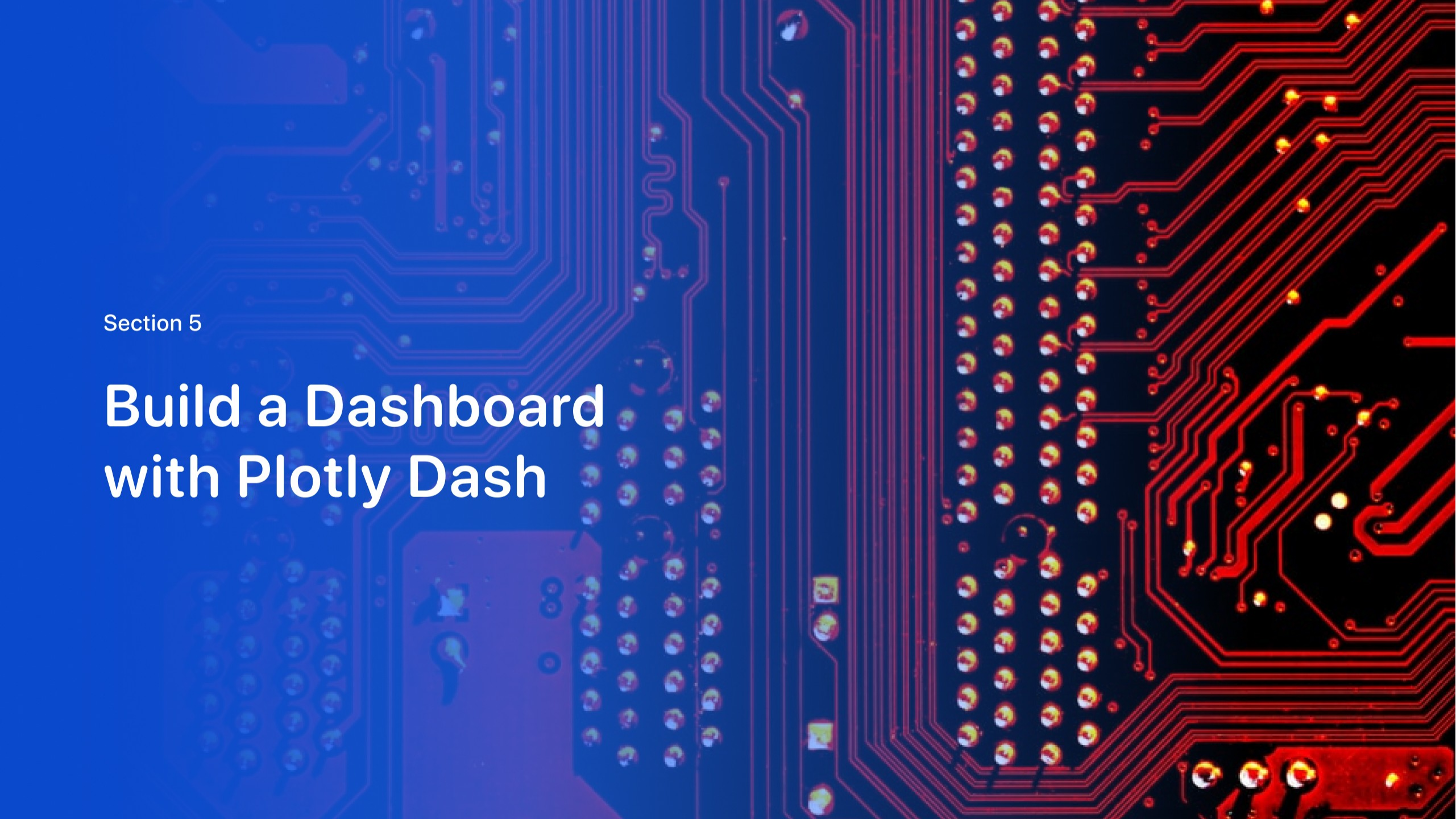
<Folium Map Screenshot 3>

Explore the generated folium map and show the screenshot of a selected launch site to its proximity such as railway, highway, coastline, with distance calculated and displayed

Explain the important elements and findings on the screenshot: we can see the distance of the nearest city and the vicinity of the rail way, road and coastline.





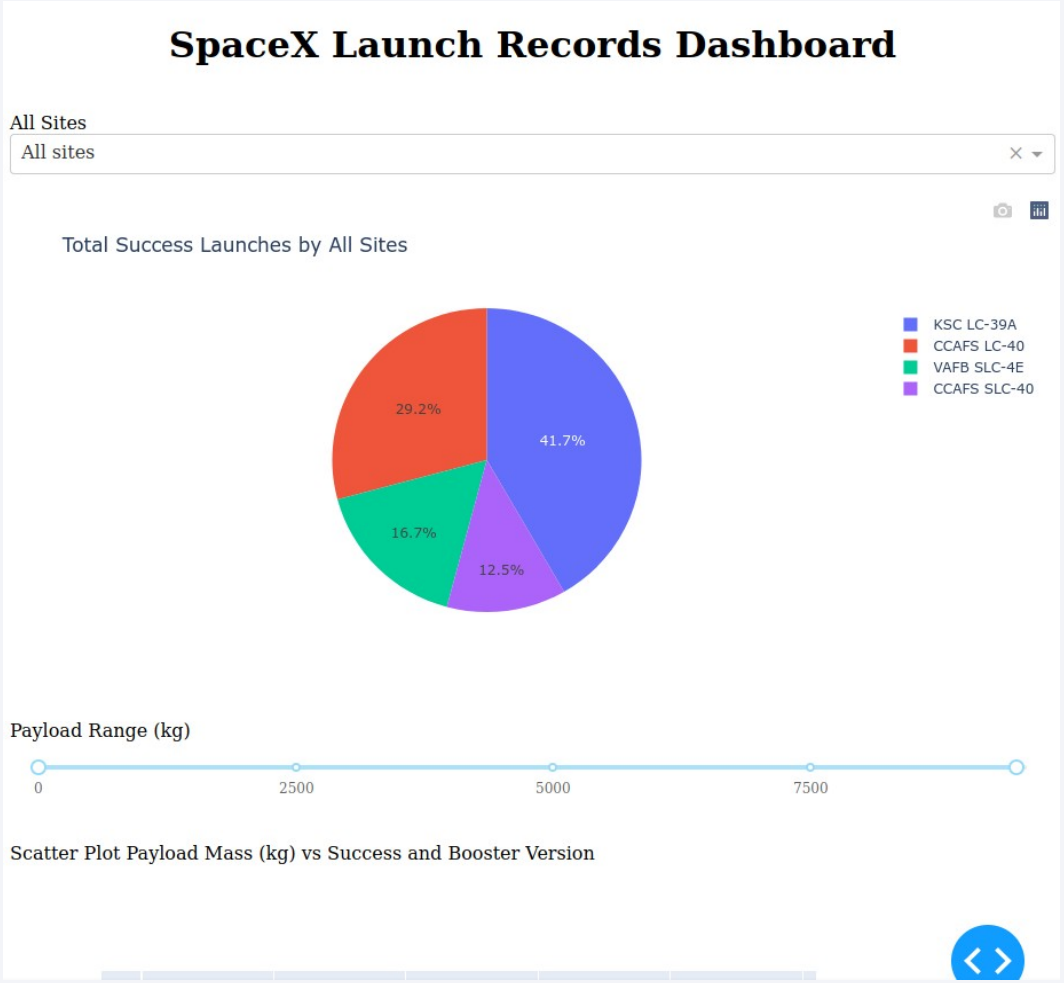

Section 5
Build a Dashboard with Plotly Dash

Interactive dashboard - Initial

Explain the important elements and findings on the screenshot:

The initial screen starts with all sites selected, the pie chart showing the percentage of the total results per site.

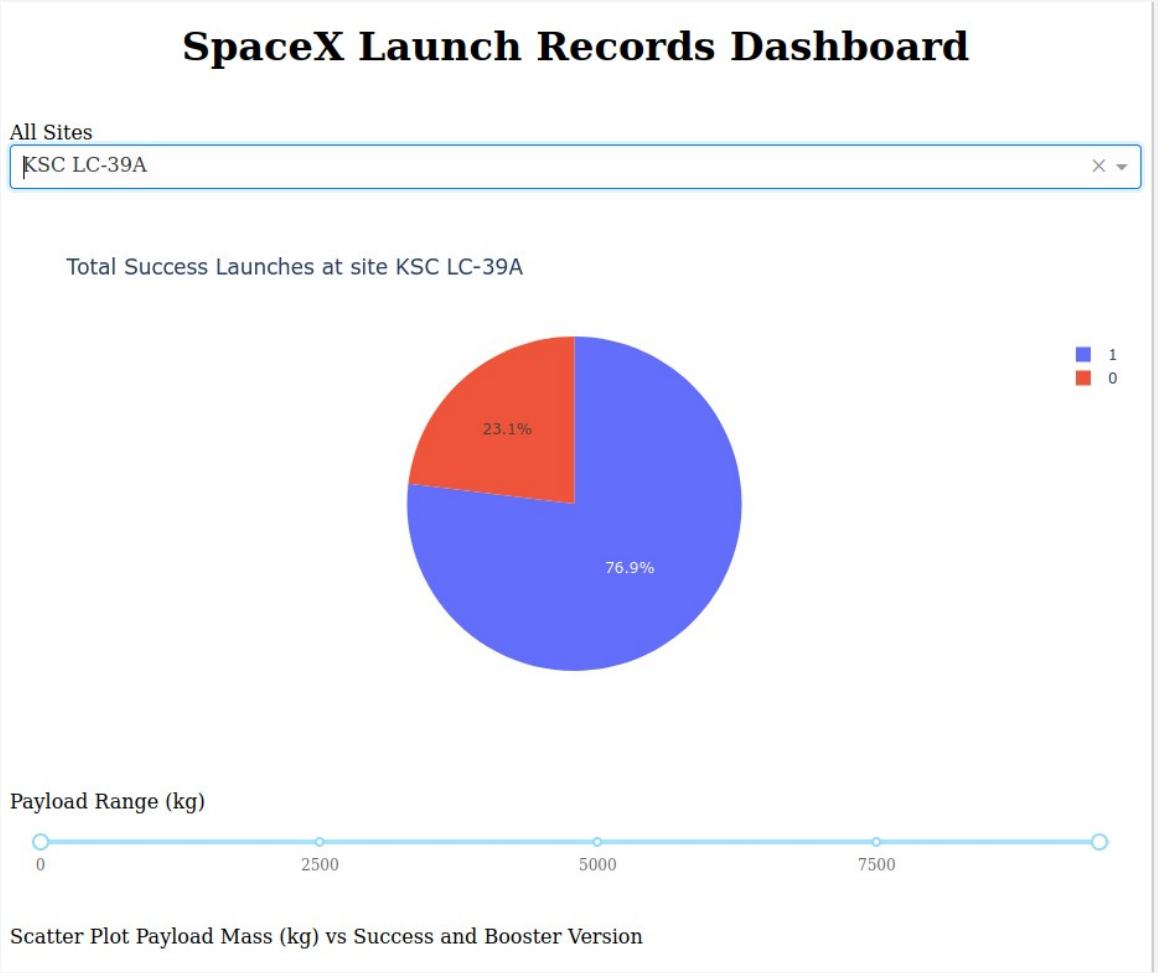
We can select from the drop down menu a specific site of use the pie chart capabilities to select and visualize more in details certain comparisons.



Interactive Dashboard - Selection

Explain the important elements and findings on the screenshot:

Selecting the highest rating site, KSC LC 39A, we can see the pie chart change to percentages of Success rates and Fail rates, indexed as 0 = Fail (23.1%) and 1 = Success (76.9%).



Interactive Dashboard – other options

At the bottom of the dashboard we can find the secondary options we can use to discover more insights.

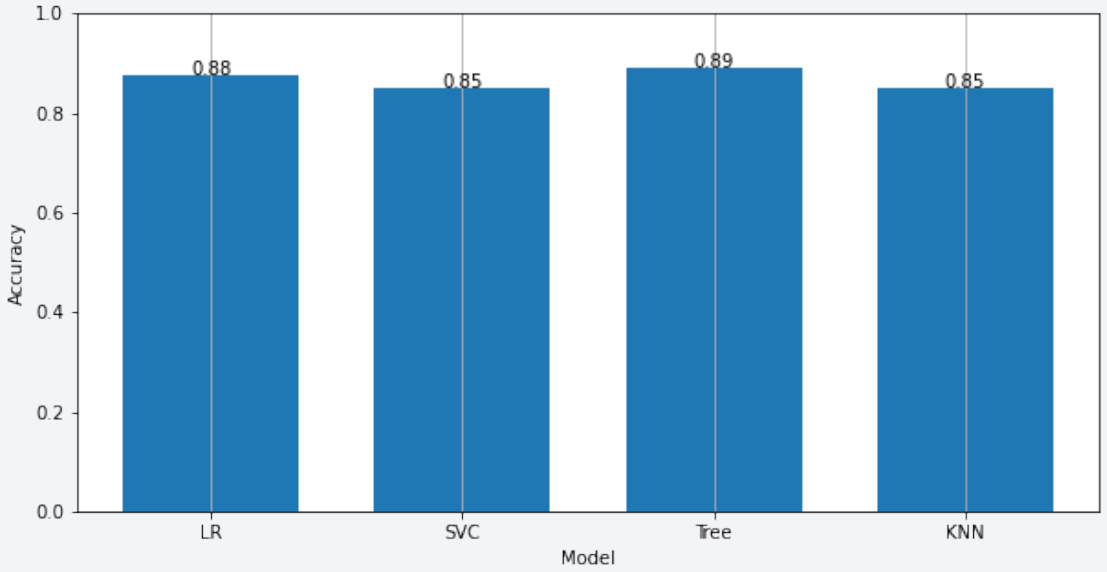
We can select a payload range with a slide bar which in turn will update the last Scatter Plot indexed as Payload vs. Class (0 = Success and 1 = Fail), also indexed and color coded as per booster version. Also hovering above each point will give us further information as per Orbit, detail weight, and more.

Section 6

Predictive Analysis (Classification)

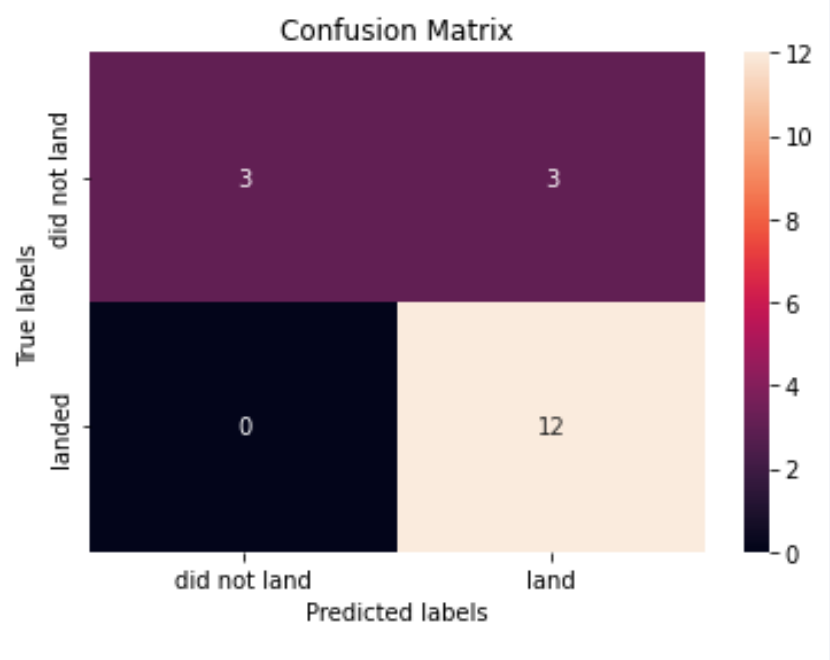
Classification Accuracy

- With a bar chart to visualize the different accuracies of eahc model used we can easily identify the Tree Model is the highest if 0.89 accuracy overall.
- The others are not too far off as well.



Confusion Matrix

- Here I am showing the confusion matrix of the Tree model as it is the highest in accuracy.
- This shows how many predictions have been accurate as per the test table.
- In this case the model has predicted exactly 12 landed outcomes but predicted 3 out of 6 in total did not land.



Conclusions

- Using Data Wrangling we can discover insights of patterns and correlations that we will use later. The Data Wrangling discloses, especially graphically, that at the beginning there were more unsuccessful landing, Space X studying the reasons, as well as voluntarily creating a unsuccessful landing, would enhance the performance of the reliability. We can see this with the graph “Launch Success Yearly Trend” that shows us how exponentially the rate has grown during the years becoming increasingly efficient.
- With an interactive Dashboard, built with Dash, we can visually see there is little correlation between failure and payload.
- Comparing Machine Learning techniques we can see there are very good correlations between information's used that give us fairly high accuracy responses, in general we could use any of these techniques to obtain an accurate response.
- As of this point there is much further ground to discover and test to build an even more accurate model for predicting the successful rate. Must be said this is a highly technological driven product / service, thus it would be sensible to track also the technological advancement of other elements including the Boosters and comparing them. I think the first step would be to compare on a year base the booster version vs the success rate with the weight ration. This could give a multi dimensional correlation visualization.

Appendix

I only took the liberty to add images from free repositories to enhance and play around with the map visualization in Folium.

Thank you!