IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Marie-Caroline Bouteloup
May 29th, 2023

Outline

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

Executive Summary

- Throughout the project, I will diligently follow the Data Science methodology, which involves various stages such as data collection, data wrangling, exploratory data analysis, data visualization, model development, model evaluation, and reporting the outcomes to relevant stakeholders.
- By leveraging the insights and models derived from my Data Science work, the startup I am working for will be equipped to make more informed and competitive bids against SpaceX for rocket launches.
- We will see at the end that the different algorithmic methods show the same results of 0.83 on the test data

Introduction

Falcon 9 Flight 20 OG2 first stage post-landing (23273082823).jpg

- The capstone project aims to predict the successful landing of the Falcon 9 first stage. SpaceX, the company behind Falcon 9, promotes its rocket launches on its website, offering a price of $62 million per launch. In comparison, other providers charge over $165 million for each launch due to their inability to reuse the first stage.
- Therefore, accurately predicting the first stage landing outcome would enable the estimation of launch costs. This information becomes valuable for competing companies interested in bidding against SpaceX for rocket launch contracts.

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

- Description how the SpaceX Falcon9 datasets were collected

  This project involves collecting data from the SpaceX REST API for a capstone assignment, focusing on rocket details, payload information, launch and landing specifications, and outcomes. The goal is to predict if SpaceX will attempt rocket landings. The specific endpoint used is "api.spacexdata.com/v4/launches/past." A GET request is made to obtain launch data in JSON format.

  JSON data is converted into a structured table using the "json_normalize" function. Falcon 9 launch data can also be acquired through web scraping using BeautifulSoup on Wiki pages. The scraped data is parsed and transformed into a Pandas dataframe for analysis.

  Challenges include dealing with identification numbers in some columns, requiring additional API endpoints for specific data. Falcon 1 launches are filtered out to focus on Falcon 9 launches. Null values in the PayloadMass column are replaced with the mean of available data. NULL values in the LandingPad column indicate no landing pad, which will be addressed using one-hot encoding later on.

Data Collection – SpaceX API

1. Request and parse the SpaceX launch data by making a GET request.

2. Decode the response content into JSON and normalize it to a Pandas dataframe.

3. Extract useful columns using provided launch ids.

4. Construct the dataset by combining the obtained columns into a dictionary.

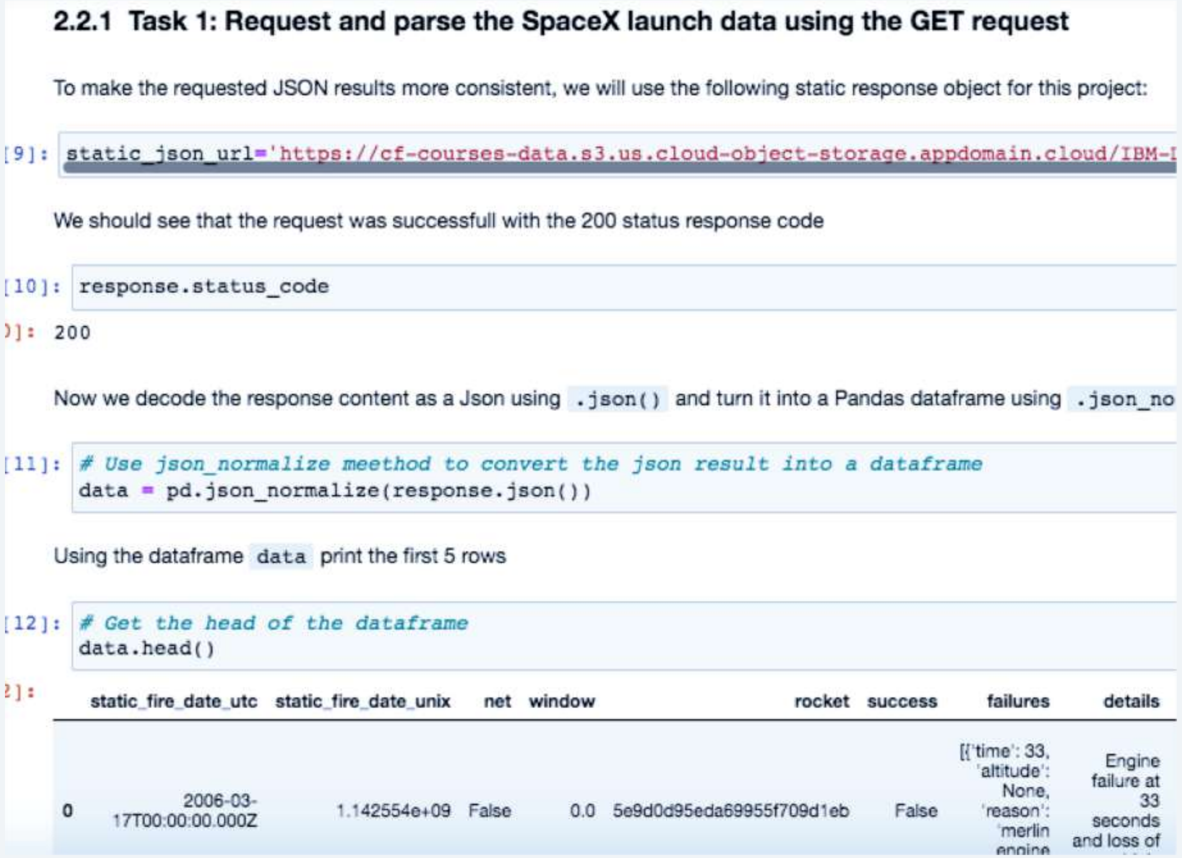
5. Create a Pandas dataframe from the dictionary.

6. Filter the dataframe to include only Falcon 9 launches.

7. Handle missing values in the dataset.

8. Export the processed data to a CSV file.

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/1-Space-X%20Data%20Collection%20API.ipynb



2.2.1 Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
[9]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-I
```

We should see that the request was successfull with the 200 status response code

```
[10]: response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_no`

```
[11]: # Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
[12]: # Get the head of the dataframe
data.head()
```

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine	Engine failure at 33 seconds and loss of

Data Collection - Scraping

1. Request the Falcon9 Launch Wiki page from its URL

2. Extract all column/variable names from the HTML table header

3. Create a data frame by parsing the launch HTML tables

4. Export to CSV file

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/2-Data%20Collection%20with%20Web%20Scraping.ipynb

1.2.1 TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTT

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response
soup = BeautifulSoup(response.content, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
# Use soup.title attribute
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Data Wrangling

1. Calculate the number of launches on each site

2. Calculate the number and occurrence of each orbit

3. Calculate the number and occurence of mission outcome per orbit type

4. Create a landing outcome label from Outcome column

5. Export to CSV file

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/3-Data%20Wrangling.ipynb

1.3.2 TASK 1: Calculate the number of launches on each site

The data contains several Space X launch facilities: [Cape Canaveral Space]() Launch C
Complex 4E **(SLC-4E)**, Kennedy Space Center Launch Complex 39A **KSC LC 39A** .T

Next, let's see the number of launches for each site.

Use the method `value_counts()` on the column `LaunchSite` to determine the

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

EDA with Data Visualization

1. Perform exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib

    Exploratory Data Analysis

    Preparing Data Feature Engineering

2. Using scatter plots to visualize the relationship between FlightNumber vs. PayloadMassand, Flight Number vs. Launch Site and Payload vs. Launch Site, FlightNumber vs. Orbit type

3. a bar chart for the sucess rate of each orbit (comparison of values between several groups at the same time)

4. a line plot to observe that the sucess rate since 2013 kept increasing till 2020 (view data trends over time)

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/5-EDA%20with%20Visualization.ipynb

EDA with SQL

- Summary of SQL queries used

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/4-EDA%20with%20SQL.ipynb

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

```
%sql SELECT DISTINCT(launch_site) FROM SPACEXTBL
```

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

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

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS 'Total_Payload_Mass(Kg)', Customer FROM SPACEXTBL WHERE Customer
```

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS 'Average_Payload_Mass(Kg)', Customer, Booster_Version FROM SPACEX
```

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

```
%sql SELECT "Date", Landing_Outcome, MIN(substr('Date',7,4)) AS '' FROM SPACEXTBL WHERE Landing_Outcom
```

EDA with SQL

- 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 Booster_Version, Landing_Outcome, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE Landing_Outc
```

- List the total number of successful and failure mission outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) AS 'Total_Number' FROM SPACEXTBL GROUP BY Mis
```

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

```
%sql SELECT Booster_Version, Payload, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ =
```

- 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.

```
%sql SELECT substr(Date,4,2) AS 'Month', substr(Date,7,4) AS 'Year', Booster_Version, Launch_Site
```

- Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT Landing_Outcome, COUNT(*) AS 'Count' from SPACEXTBL WHERE Landing_Outcome LIKE 'S
```

Build an Interactive Map with Folium

- The success rate of rocket launches can be influenced by various factors, including payload mass, orbit type, and launch site location. The location of a launch site plays a crucial role in determining the initial trajectories of rockets. Identifying an optimal location for constructing a launch site involves considering multiple factors. By analyzing existing launch site locations, it is possible to gain insights into some of these factors.
- Interactive visual analytics using Folium will be performed to further explore and analyze the data. Folium is a Python library used for creating interactive maps and visualizations. This tool will enable a more interactive and dynamic approach to visualizing and understanding the launch site locations and their associated factors.
- After drawing the distance lines to the proximities, we can easily answer the following questions:
  - Are launch sites in close proximity to railways? Yes
  - Are launch sites in close proximity to highways? Yes
  - Are launch sites in close proximity to coastline? Yes
  - Do launch sites keep certain distance away from cities? Yes

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/6-Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

- The Plotly Dash app allowing users to perform interactive visual analysis on SpaceX launch data in real time.
- This dashboard application contains input components such as a dropdown list and a range slider to interact with a pie chart and a scatter point chart:
  - A drop-down entry component from the launch site
  - A callback function to display the success pie chart based on the selected site dropdown
  - A range slider to select the payload
  - A callback function to display the success scatter plot payload-scatter-chart

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/7-Build%20an%20Interactive%20Dashboard%20with%20Ploty%20Dash.py

Predictive Analysis (Classification)

- Summary of how I built, evaluated, improved and found the best performing classification model after load the data
- Perform exploratory Data Analysis and determine Training Labels:
  - create a column for the class, by applying the method to_numpy() then assign it to the variable Y, make sure the output is a Pandas series.
  - Standardize the data in X then reassign it to the variable X using the transform provided below.
  - Split into training data and test data. Set the parameter test_size to 0.2 and random_state to 2. The training data and test data should be assigned to the following labels.
  - Find best Hyperparameter for SVM, Classification Trees and Logistic Regression
  - Find the method performs best using test data based on their accuracy scores and confusion matrix

GitHub URL: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone/blob/main/8-Space-X%20Machine%20Learning%20Prediction.ipynb

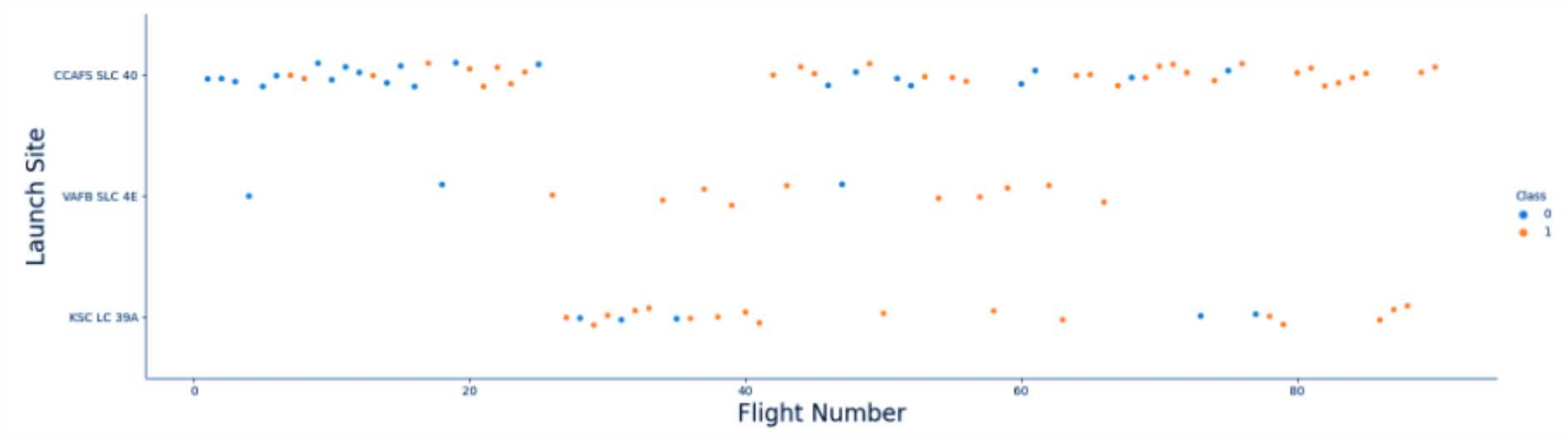
Results

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

Section 2

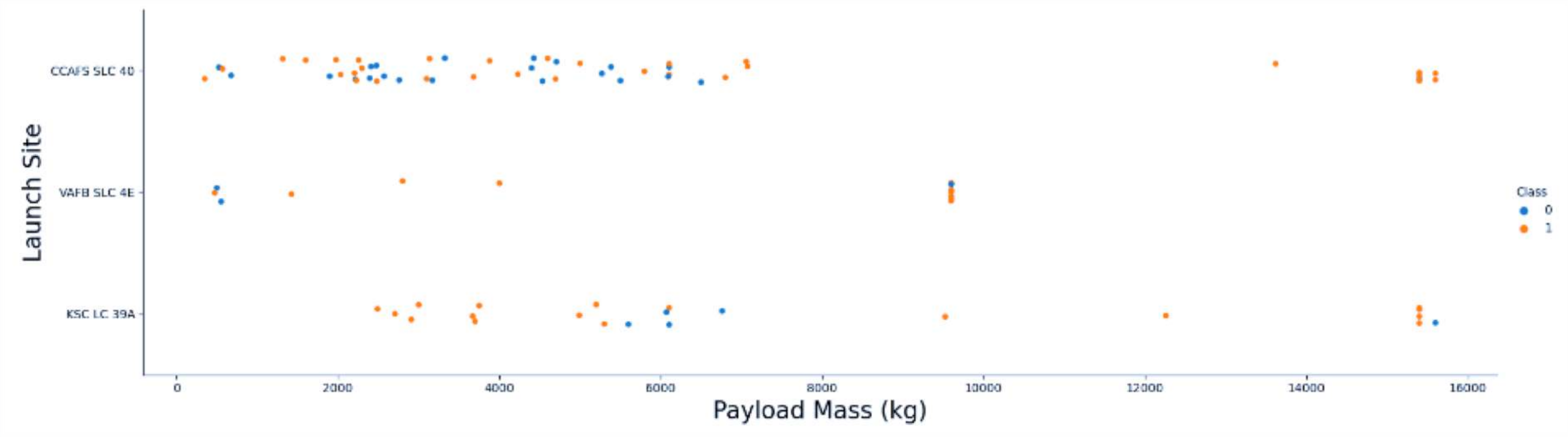
Insights drawn from EDA

Flight Number vs. Launch Site



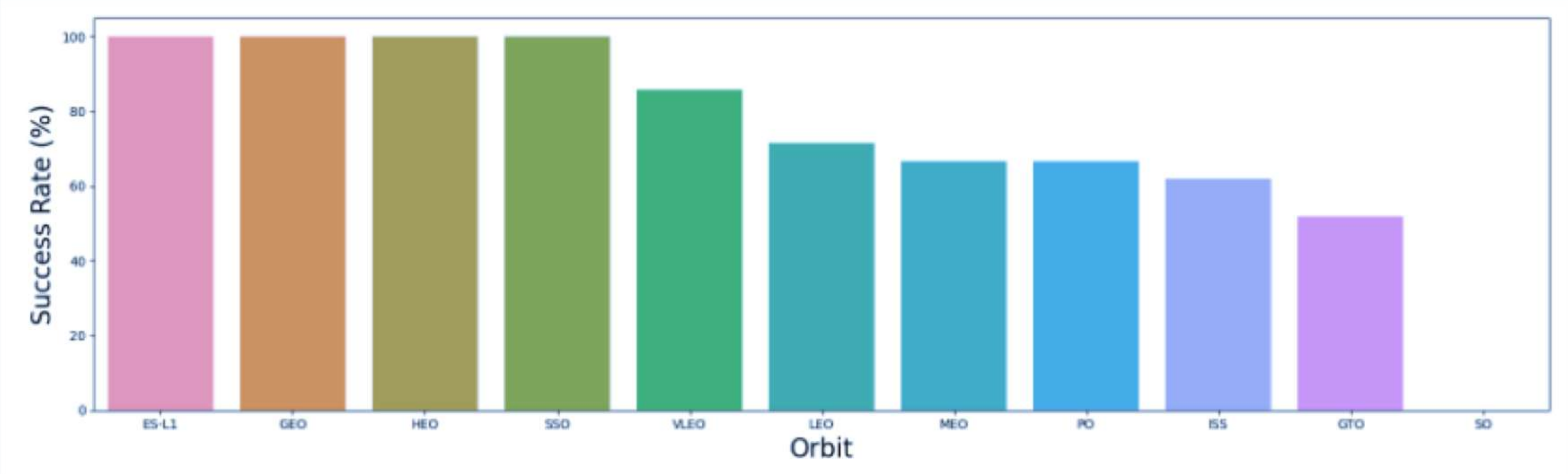
- We see that different launch sites have different success rates. VAFB SLC 4E, has a success rate of 50% , while CCAFS LC-40 and KSC LC-39A has a success rate of 80%.

Payload vs. Launch Site



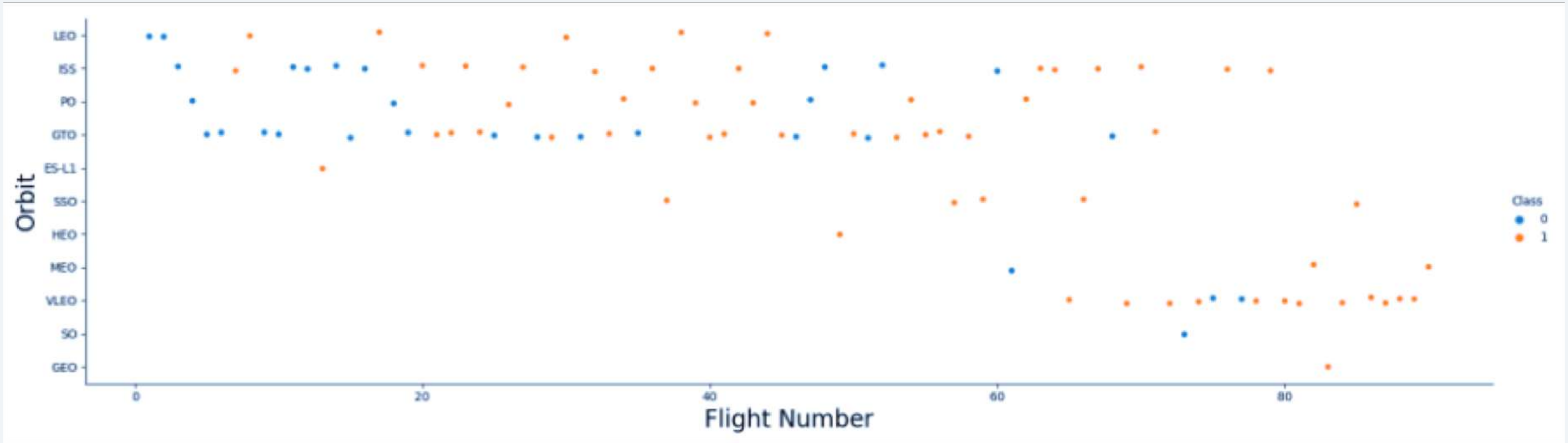
- Now if we observe Payload Vs. Launch Site scatter point chart we will find for the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).

Success Rate vs. Orbit Type



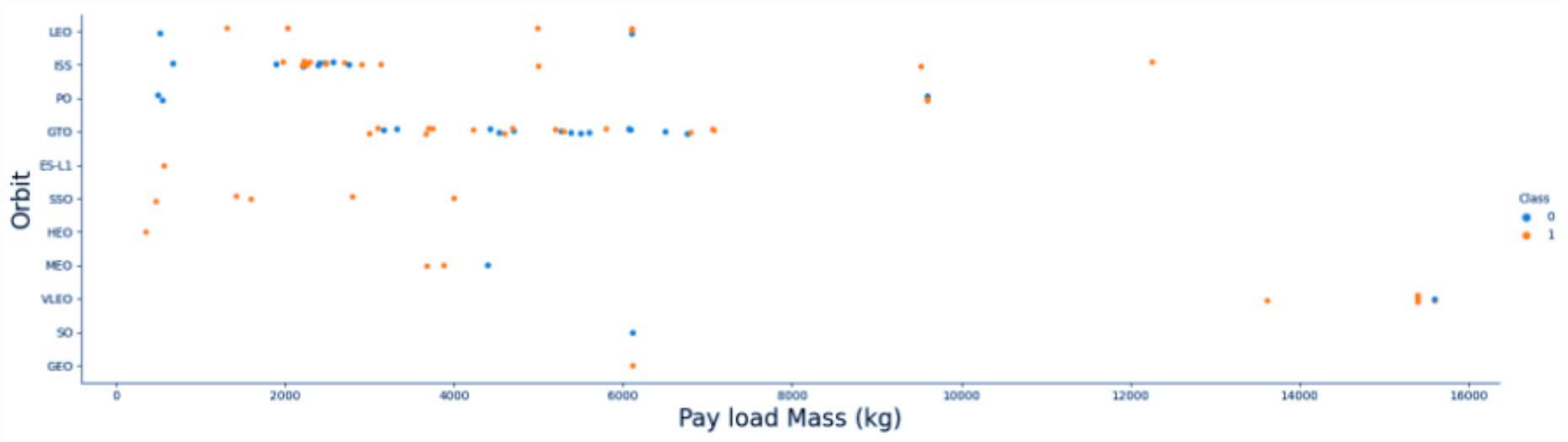
- The success rate of each orbit is 100% for ES-L1, GEO, HEO, SSO, an average of 60% for others but 0% for SO.

Flight Number vs. Orbit Type



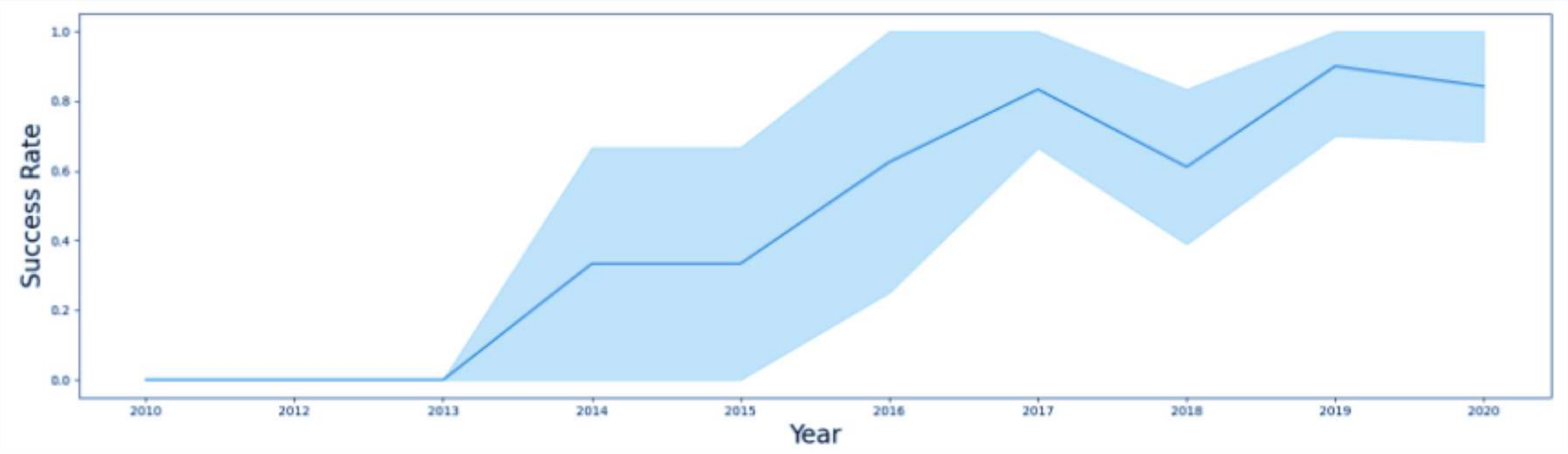
- We should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for PO, SSO, LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here.

Launch Success Yearly Trend



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

All Launch Site Names

We used *DISTINCT(launch_site)* to find the names of the unique launch sites in the space mission

```
%sql SELECT DISTINCT(launch_site) FROM SPACEXTBL
```

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

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Launch Site Names Begin with 'CCA'

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

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt

- We used the *WHERE* clauses of the *Lunch_Site* column to select the launch sites, *LIKE* begin with the string *CCA* with the *%* and *LIMIT* limited to 5 lines.

Total Payload Mass

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS 'Total_Payload_Mass(Kg)', Customer FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';
```

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

Total_Payload_Mass(Kg)	Customer
45596.0	NASA (CRS)

- We used the *SUM()* function to calculate the total sum of the *PAYLOAD_MASS_KG* column for the customer *NASA (CRS)* from *SPACEXTBL* table.

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS 'Average_Payload_Mass(Kg)', Customer, Booster_Version
FROM SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.1';
```

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

Average_Payload_Mass(Kg)	Customer	Booster_Version
2928.4	SES	F9 v1.1

- We calculated the average of the *PAYLOAD_MASS__KG_* column and displayed the *Average_Payload_Mass(Kg), Customer, Booster_Version* columns then selected F9 v1.1 with *LIKE* of the *Booster_Version* column from *SPACEXTBL* table.

First Successful Ground Landing Date

```
%sql SELECT "Date", Landing_Outcome, MIN(substr('Date',7,4)) AS '' FROM SPACEXTBL
WHERE Landing_Outcome = 'Success (ground pad)';
```

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

Date	Landing_Outcome
22/12/2015	Success (ground pad)

- We selected the *Date, Landing_Outcome* columns where the first date of the *Date* column with the *MIN()* function, is *Success (ground pad)* of the *Landing_Outcome* column.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT Booster_Version, Landing_Outcome, PAYLOAD_MASS__KG_ FROM SPACEXTBL
WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN '4001' AND '5999';

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

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696.0
F9 FT B1026	Success (drone ship)	4600.0
F9 FT B1021.2	Success (drone ship)	5300.0
F9 FT B1031.2	Success (drone ship)	5200.0

- We selected *Booster_Version, Landing_Outcome, PAYLOAD_MASS__KG_* columns where the *Success (drone ship)* is in the *Landing_Outcome* column and the search result *PAYLOAD_MASS__KG_* must between greater than 4000 and less than 6000.

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) AS 'Total_Number' FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

Mission_Outcome	Total_Number
None	0
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- To know the total number of successful and failed mission results, we selected the *Mission_Outcome* column, used the *COUNT()* function of the *Mission_Outcome* column naming it *Total_Number*, grouped with the *GROUP BY* clause of the *Mission_Outcome* column.

Boosters Carried Maximum Payload

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

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

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600.0
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600.0
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600.0
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600.0
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600.0
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600.0
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600.0
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600.0
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600.0
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600.0
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600.0
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600.0

- To list the names of the booster_versions that carried the maximum payload mass we selected the columns *Booster_Version, Payload, PAYLOAD_MASS__KG_* and displayed the maximum mass of the column with the *MAX()* function of the *PAYLOAD_MASS__KG_* column.

2015 Launch Records

```
: %sql SELECT substr(Date,4,2) AS 'Month', substr(Date,7,4) AS 'Year', Booster_Version, Launch_Site,
  Payload, Customer, Landing_Outcome FROM SPACEXTBL WHERE substr(Date,7,4)='2015'
  AND Landing_Outcome = 'Failure (drone ship)';

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

Month	Year	Booster_Version	Launch_Site	Payload	Customer	Landing_Outcome
10	2015	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	NASA (CRS)	Failure (drone ship)
04	2015	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	NASA (CRS)	Failure (drone ship)

- To list the records that will show month names, landing_outcomes failures in the drone, booster, launch_site versions for the months of the year 2015, we selected *Booster_Version, Launch_Site, Payload, Customer, Landing_Outcome* columns subtracted the year with the *substr()* clause of the *Date* column to create a *Year* column and display the year 2015, then select Failure (drone ship) of the *Landing_Outcome* column.

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

```
%sql SELECT Landing_Outcome, COUNT(*) AS 'Count' from SPACEXTBL WHERE Landing_Outcome LIKE 'Success%'
AND Date BETWEEN '04/06/2010' AND '20/03/2017' GROUP BY Landing_Outcome ORDER BY 2 DESC;

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

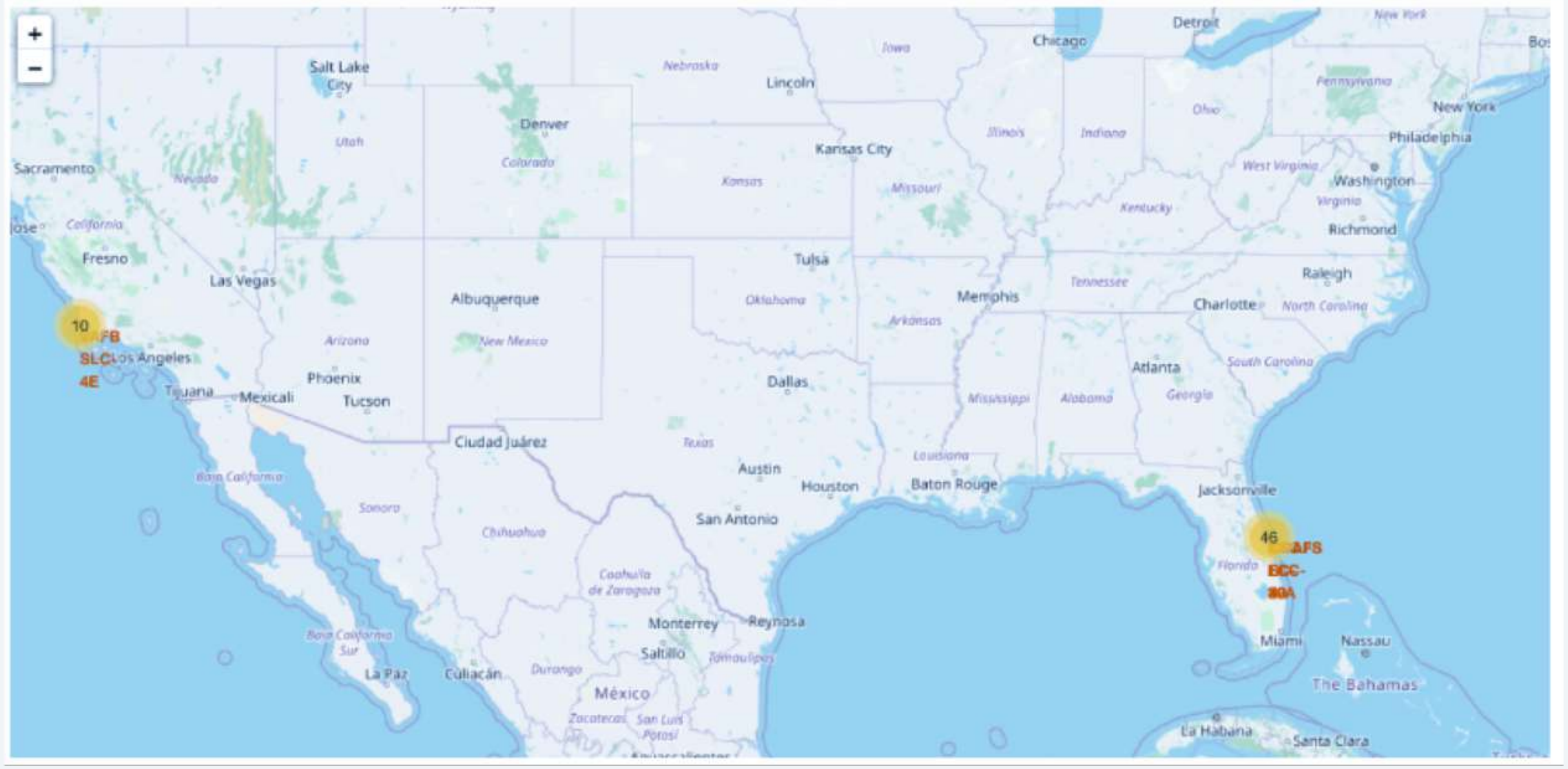
Landing_Outcome	Count
Success	20
Success (drone ship)	8
Success (ground pad)	7

- To rank the number of successful landing_outcomes, we counted everything with the *COUNT(*)* function and created a column *Count* the successes of the *Landing_Outcome* column grouped with *GROUP BY* where the date must be between 04/06/2010 AND 20/03/2017 in descending order *DESC*.

Section 3

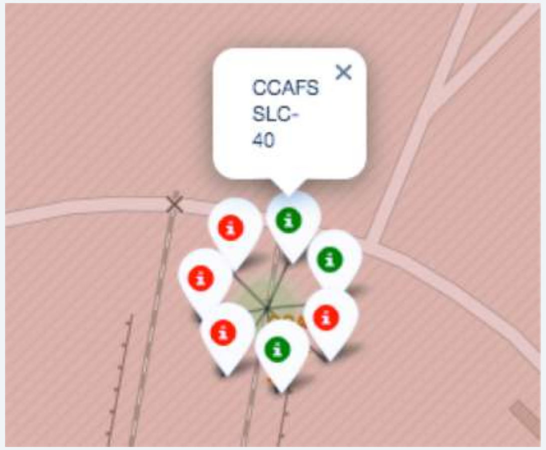
Launch Sites Proximities Analysis

Location of all SpaceX launch sites



- It is in the interest of rocket operators to use as little fuel as possible to reach orbit, hence the idea of getting as close as possible to the Equator line and coasts hence the yellow markers Spacex launch sites in the US.

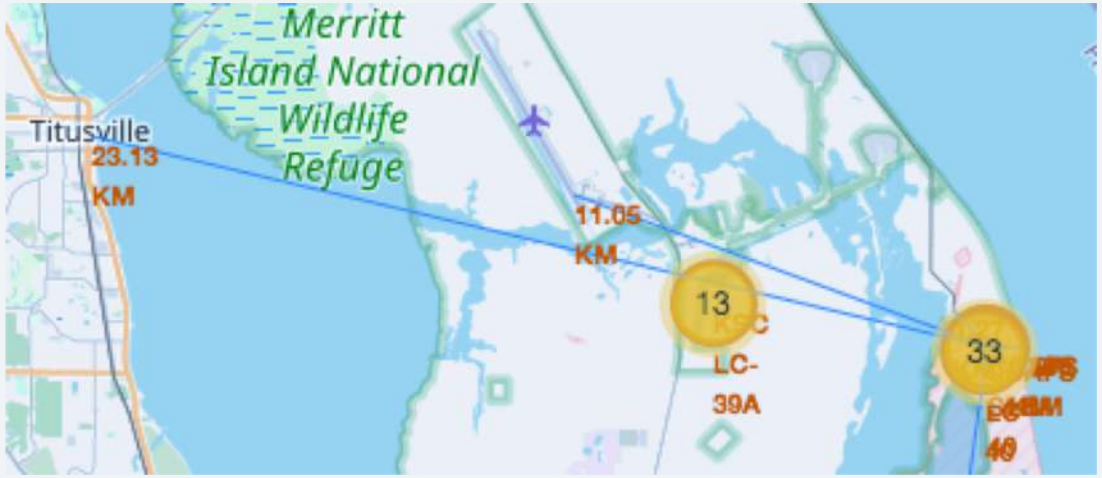
Show the color-labeled launch outcomes on the map



- *California sites*
- *Florida sites*

- We visualize the launch sites in California and Florida.
- It is noticed that the KSC LC-39A launch site in Florida has higher success rates compared to CCAFS LC-40 and CCAFS SLC-40 .

Screenshot of a selected launch site to its proximities



- The proximities of the CCAFS SLC-40 launch site are:

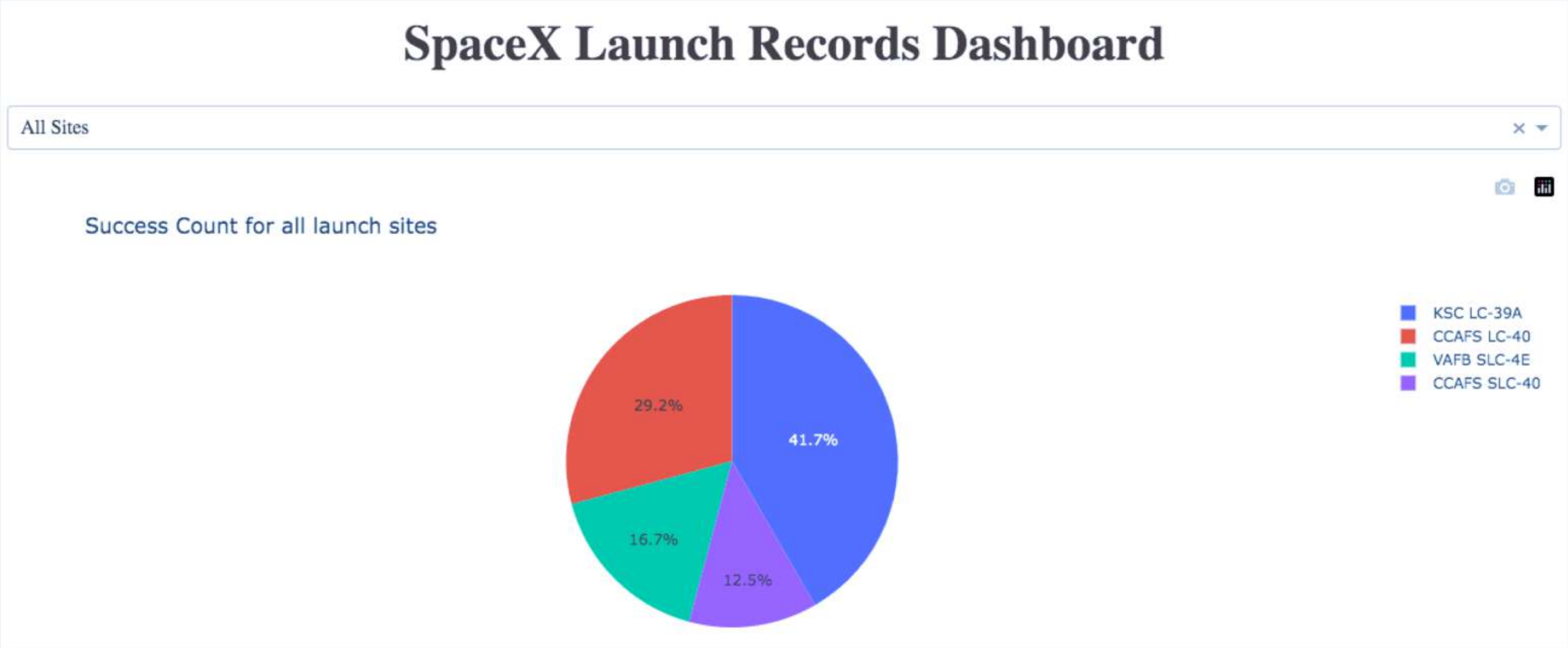
```
distance_coastline = 0.86 km
```

```
Distance railway = 1.27 km
Distance highway = 0.57 km
Distance closest_city (Titusville) = 23.13 km
Distance airport = 11.05 km
Distance city 2 (Melbourne) = 51.78 km
```

Section 4

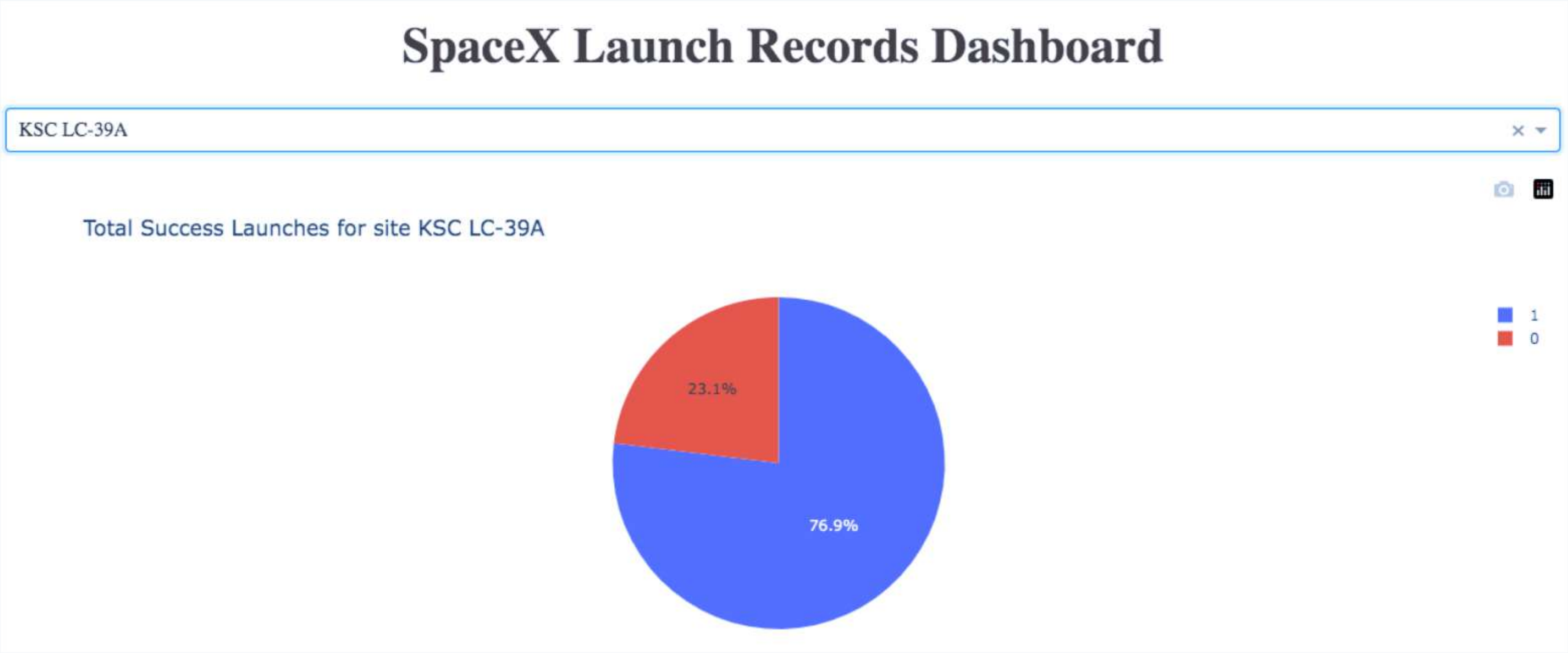
Build a Dashboard with Plotly Dash

Screenshot of launch success count for all sites



- The launch success results counted for all sites, in a pie chart are:
  - 41.7% KSC LC-39A, 29.2% CCAFS LC-40, 16.7% VAFB SLC-4E and 12.5% CCAFS SLC-40

Launch site with highest launch success ratio



- The launch site with highest launch success ratio is the KSLC-39A with 76.9%.

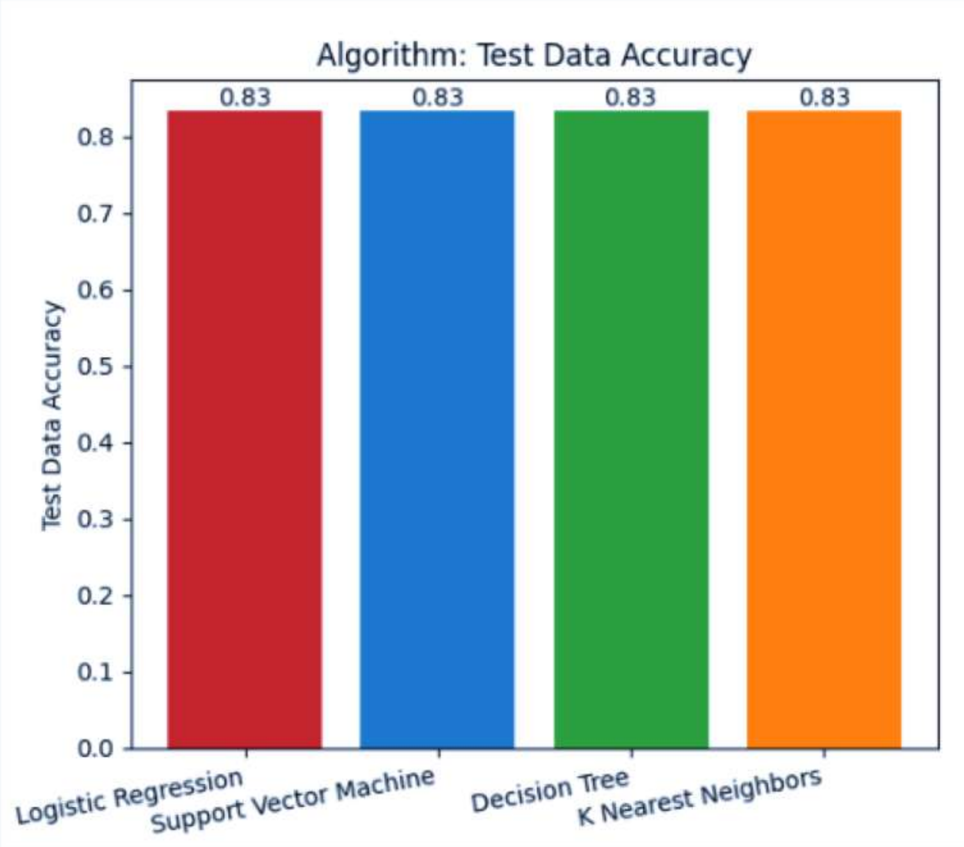
Payload vs launch result for all sites

- The highest number of successful launches is the CCAFS LC-40 launch site with the FT booster version from a payload mass above 2000 kg.

Section 5

Predictive Analysis (Classification)

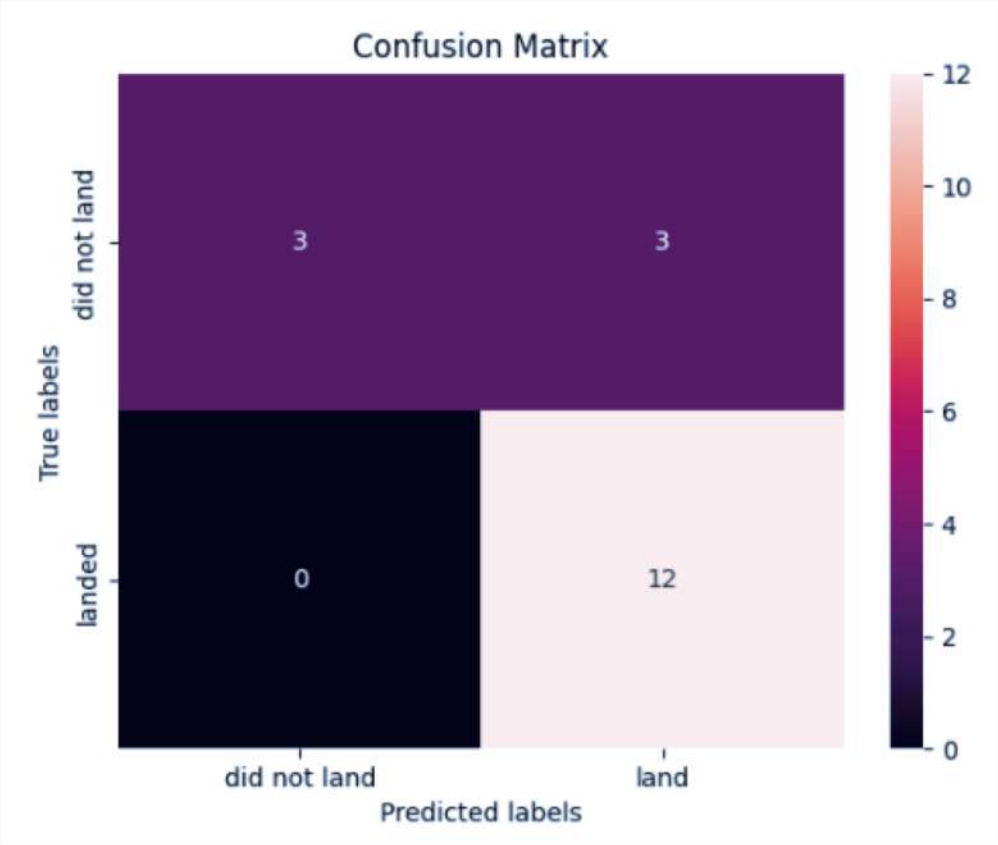
Classification Accuracy



- The different algorithm methods show the same results of 0.83 on the test data.

Algorithm	Test Data Accuracy
Logistic Regression	0.833333
Support Vector Machine	0.833333
Decision Tree	0.833333
K Nearest Neighbors	0.833333

Confusion Matrix



- The model predicted:
    - 12 successful landings (True Positive)
    - 3 failed landings (True Negative)
- But,
    - 3 successful landings (false positive)
- False positives are problematic.

Conclusions

- We see that different launch sites have different success rates. VAFB SLC 4E, has a success rate of 50% , while CCAFS LC-40 and KSC LC-39A has a success rate of 80%.
- The success rate of each orbit is 100% for ES-L1, GEO, HEO, SSO, an average of 60% for others but 0% for SO
- We can observe that the success rate since 2013 kept increasing till 2020.
- We should see that in the LEO orbit the success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.
- Lighter payloads have a higher success rate.
- The predictive models used for this dataset have a result of 83%.

Appendix

- IBM Applied Data Science Capstone: https://www.coursera.org/learn/applied-data-science-capstone/home/week/1

- GitHub repository: https://github.com/mariek22/IBM-Applied-Data-Science-Capstone

Thank you!