IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Bongani Mbambiso
06 April 2025

Outline

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

Executive Summary

- Summary of methodologies
  - *Data Collection*
  - *Data Wrangling*
  - *EDA with Data Visualization*
  - *EDA with SQL (connected to Python)*
  - *Interactive map (Folium)*
  - *Dashboard (Plotly Dash)*
  - *Predictive Analysis (Classification)*
- Summary of all results
  - *Data Collection*
  - *Data Wrangling*
  - *EDA with Data Visualization*

Introduction

In this Applied Data Science Capstone, we will predict if the **Falcon 9 first stage** will land successfully.

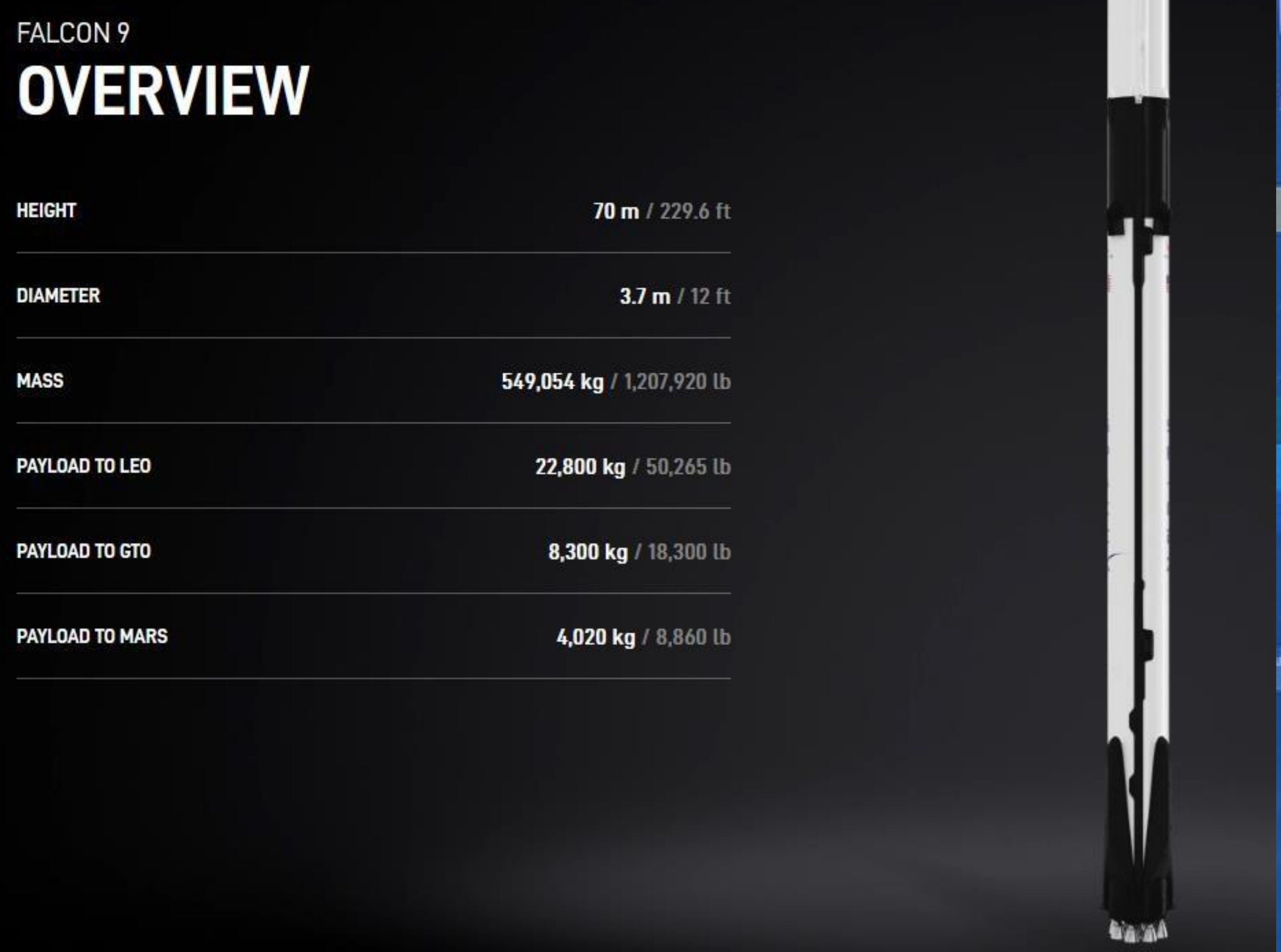
SpaceX advertises *Falcon 9 rocket launches* on its website, with ***a cost of 62 million dollars***; other providers cost upward of *165 million dollars* each, much of the savings is because SpaceX can reuse the first stage.

The key issue is, 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.

In this module, we will be provided with an overview of the problem and we will need to complete the course.

METHODOLOGY



FALCON 9

OVERVIEW

HEIGHT	70 m / 229.6 ft
DIAMETER	3.7 m / 12 ft
MASS	549,054 kg / 1,207,920 lb
PAYLOAD TO LEO	22,800 kg / 50,265 lb
PAYLOAD TO GTO	8,300 kg / 18,300 lb
PAYLOAD TO MARS	4,020 kg / 8,860 lb

Methodology

- Data collection methodology:
  - *Via SpaceX Rest API to obtain SpaceX launch data*
  - *Webscraping Wikipedia to obtain Falcon 9 launch data using BeautifulSoup*
- Perform data wrangling
  - *One hot encoding data fields for machine learning and dropping irrelevant columns*
- Perform exploratory data analysis (EDA) using visualization and SQL
  - *Bar charts and scatter plots used for pattern visual analysis.*
- Perform interactive visual analytics using Folium and Plotly Dash
  - *Used to perform interactive visual analysis*
- Perform predictive analysis using classification models
  - *How to build, tune, evaluate classification models*

Data Collection: Overview

	Step # 1:		Step# 2:		Step 3:
	Collect Data from SpaceX API and Convert data to .jason file	→	Scrap and filter data to include Falcon 9 data, assign data to dataframe and dictionary, and export data to a csv file	→	Plot and visualize the data

Dataset github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/Spacex.csv

Load Space X dataset.

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_1.csv")
df.head(10)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005	-80.577366	28.561857
6	7	2014-04-18	Falcon 9	2296.000000	ISS	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1006	-80.577366	28.561857
7	8	2014-07-14	Falcon 9	1316.000000	LEO	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1007	-80.577366	28.561857
8	9	2014-08-05	Falcon 9	4535.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1008	-80.577366	28.561857
9	10	2014-09-07	Falcon 9	4428.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1011	-80.577366	28.561857

Data Collection – SpaceX API

```
[ ]  1 data_falcon9.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.632093
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004	-80.577366	28.561857

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]
#
# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multiple payloads in a single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Request response from SpaceX API using get request and covert data to .Jason file

Use custom functions to clean data

Clean data and assign data to dictionary and data frame

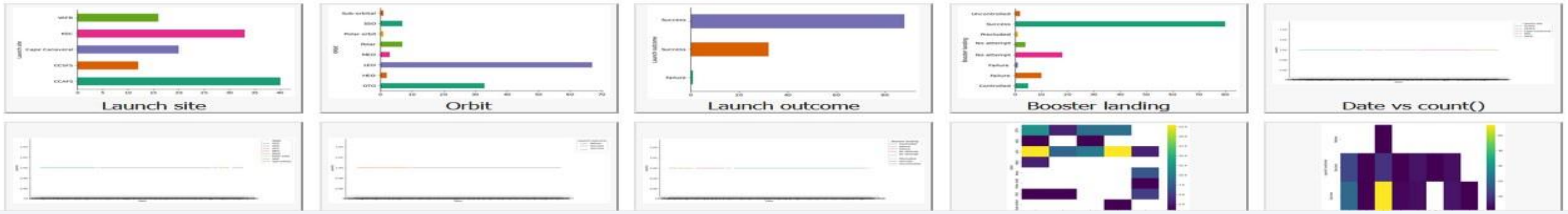
Filter data to include only Falcon 9 launches and export data to a csv file: dataset_part1

github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	[[SpaceX], \n]	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	[[.mw-parser-output .plainlist ol,.mw-parser-o...	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	[[NASA], (, [COTS],)\n]	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	[[NASA], (, [CRS],)\n]	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	[[NASA], (, [CRS],)\n]	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

Next steps: Generate code with df View recommended plots




```
[5]  # 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`

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

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

```
[7]  # Use soup.title attribute
     soup.title

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

Step 1: Perform HTTP get to request Falcon 9 HTML page and create Beautiful Soup object from HTML

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

Step3: Create a data frame by parsing the launch HTML tables

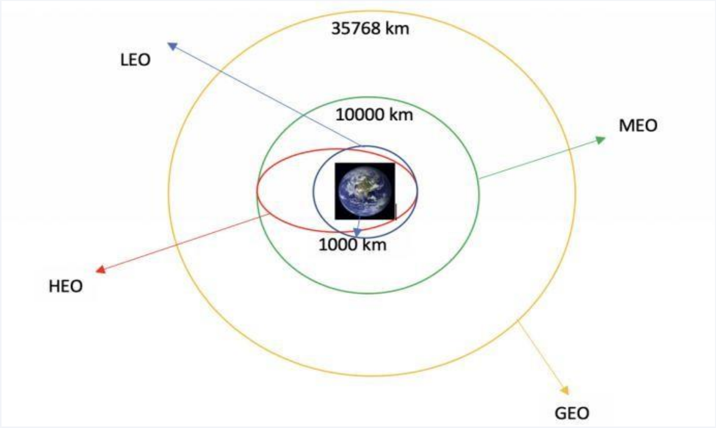
Step 4: export data into CSV file (spacex_web_scraped.csv)

github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

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

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



Step 1: Load data from dataset_part1.csv file and calculate the number of launches on each site

Step 2: Calculate the number and the occurrence of each orbit

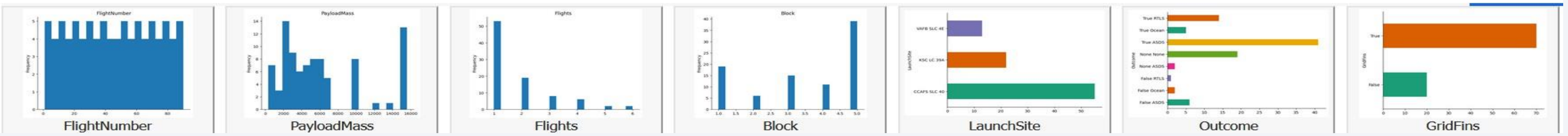
Step3: Calculate the number and occurrence of mission outcome of the orbits

Step 4: Create a landing outcome label from outcome column and export data into dataset_part2.csv file

```
[12] df.head(5)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857

Next steps: Generate code with df | View recommended plots



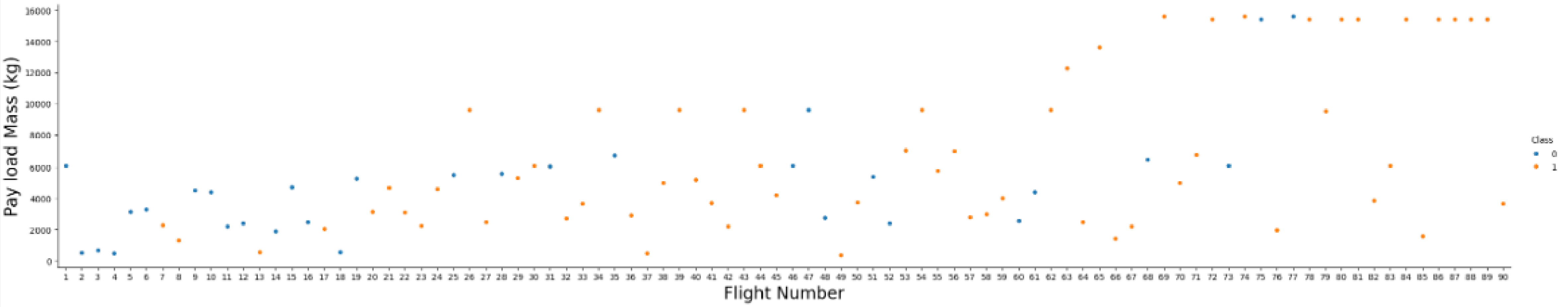
github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/jupyter-labs-webscraping.ipynb

EDA with Data Visualization (1)

- **Flight Number vs. Payload Mass Scatter Plot**

  - *I tried to plot the <u>Flight Number vs. Payload Mass</u> and overlay the outcome of the launch. We see that as the flight number increases, the first stage is more likely to land successfully. The payload mass also appears to be a factor; even with more massive payloads, the first stage often returns successfully.*

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

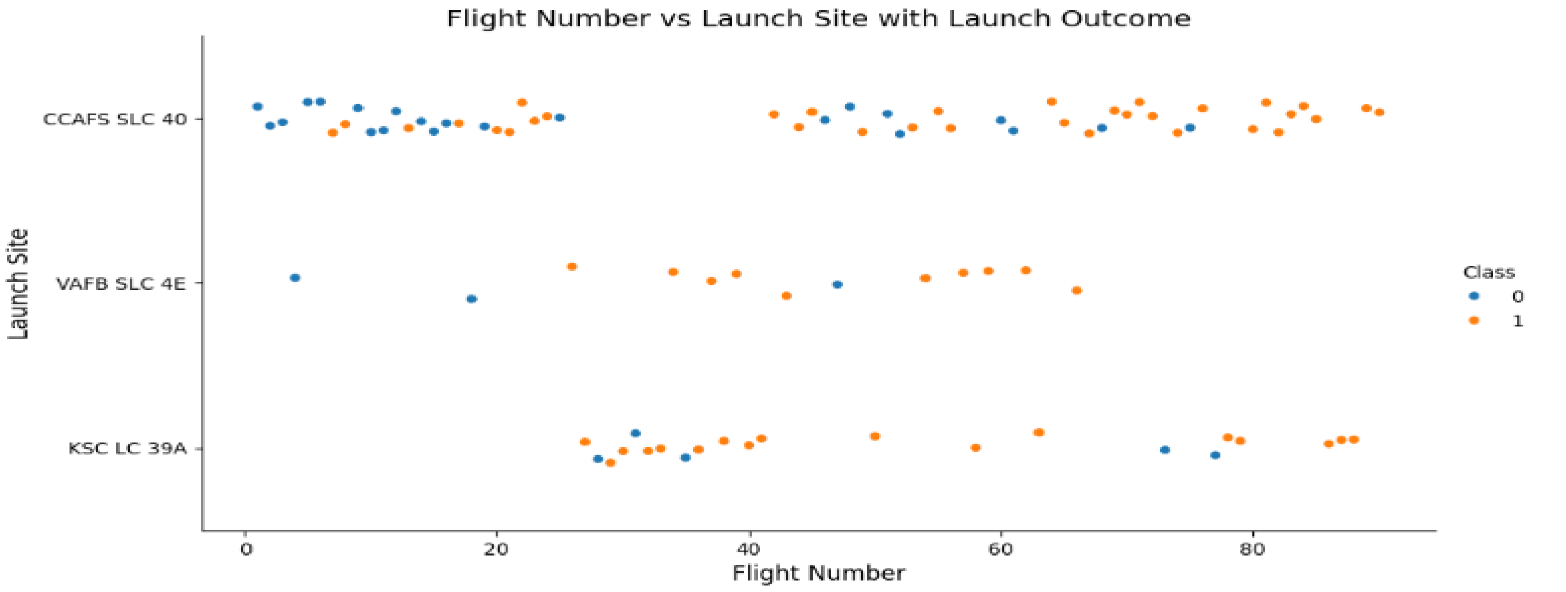


github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/edadataviz.ipynb

EDA with Data Visualization (2)

- **Flight Number vs. Launch Site Scatter Plot**
  - *Use the function catplot to plot Flight Number vs Launch Site, set the parameter x parameter to Flight Number set they to Launch Site and set the parameter hue to 'class'. '*

```
# Plot the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect=2)
plt.xlabel("Flight Number", fontsize=12)
plt.ylabel("Launch Site", fontsize=12)
plt.title("Flight Number vs Launch Site with Launch Outcome", fontsize=14)
plt.show()
```

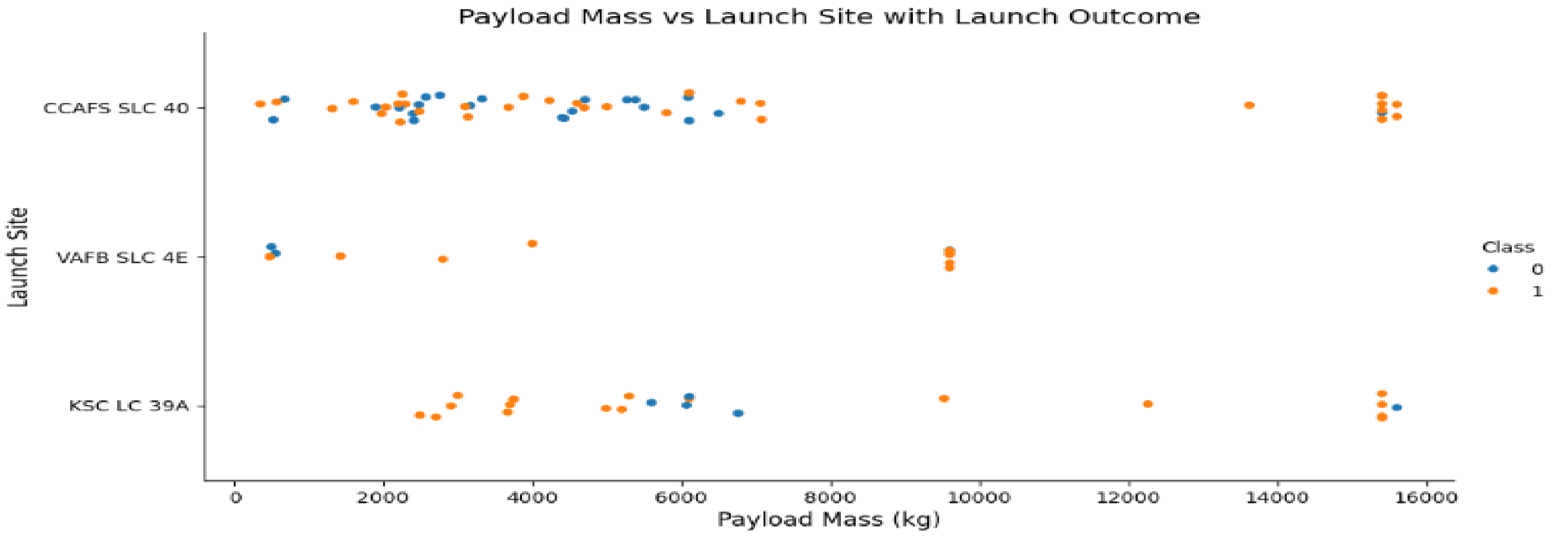


github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/edadataviz.ipynb

EDA with Data Visualization (3)

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

```
# Plot the relationship between Payload Mass and Launch Site
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect=2)
plt.xlabel("Payload Mass (kg)", fontsize=12)
plt.ylabel("Launch Site", fontsize=12)
plt.title("Payload Mass vs Launch Site with Launch Outcome", fontsize=14)
plt.show()
```

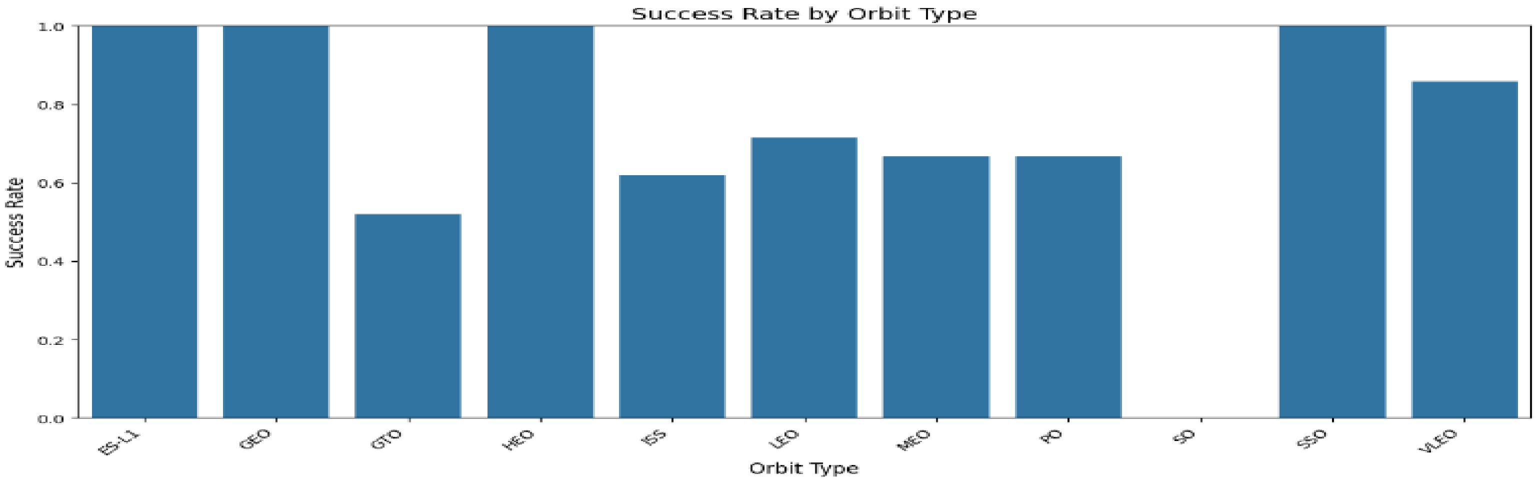


github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/edadataviz.ipynb

EDA with Data Visualization (4)

- **Success Rate vs. Orbit Bar Chart**
  - *We want to visually check if there are any relationship between success rate and orbit type. We created a bar chart for the success rate of each orbit. Analyse the plotted bar chart to identify which orbits have the highest success rates.*

```python
# Calculate success rate by orbit type
orbit_success = df.groupby('Orbit')['Class'].mean().reset_index()

# Create the bar plot
plt.figure(figsize=(12, 6))
sns.barplot(x='Orbit', y='Class', data=orbit_success)

# Customize the plot
plt.title('Success Rate by Orbit Type', fontsize=14)
plt.xlabel('Orbit Type', fontsize=12)
plt.ylabel('Success Rate', fontsize=12)
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for better readability
plt.ylim(0, 1)  # Set y-axis limits from 0 to 1 since success rate is a probability

# Display the plot
plt.tight_layout()
plt.show()
```

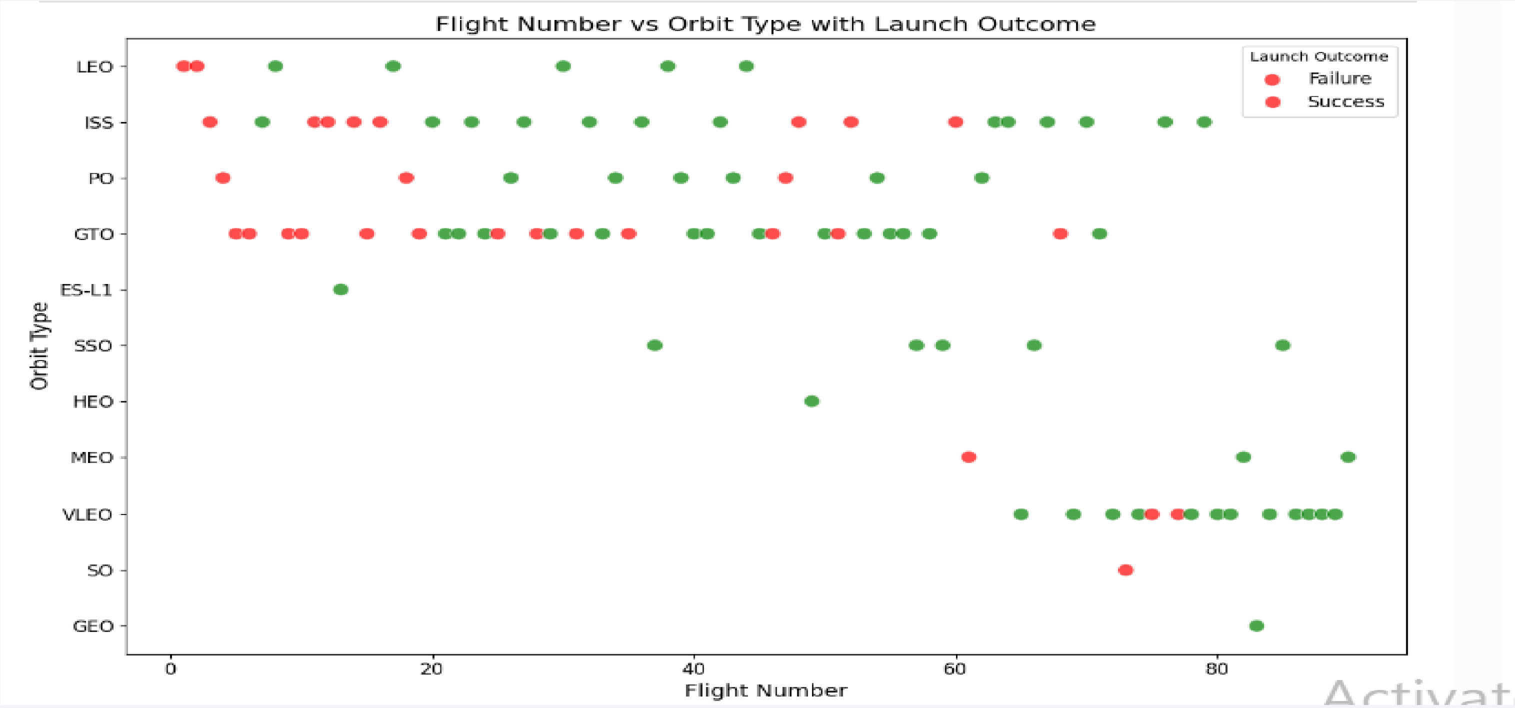


github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/edadataviz.ipynb

EDA with Data Visualization (5)

- **Flight Number and Orbit type**

- *For each orbit, we wanted to see if there is any relationship between Flight Number and Orbit type. You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.*

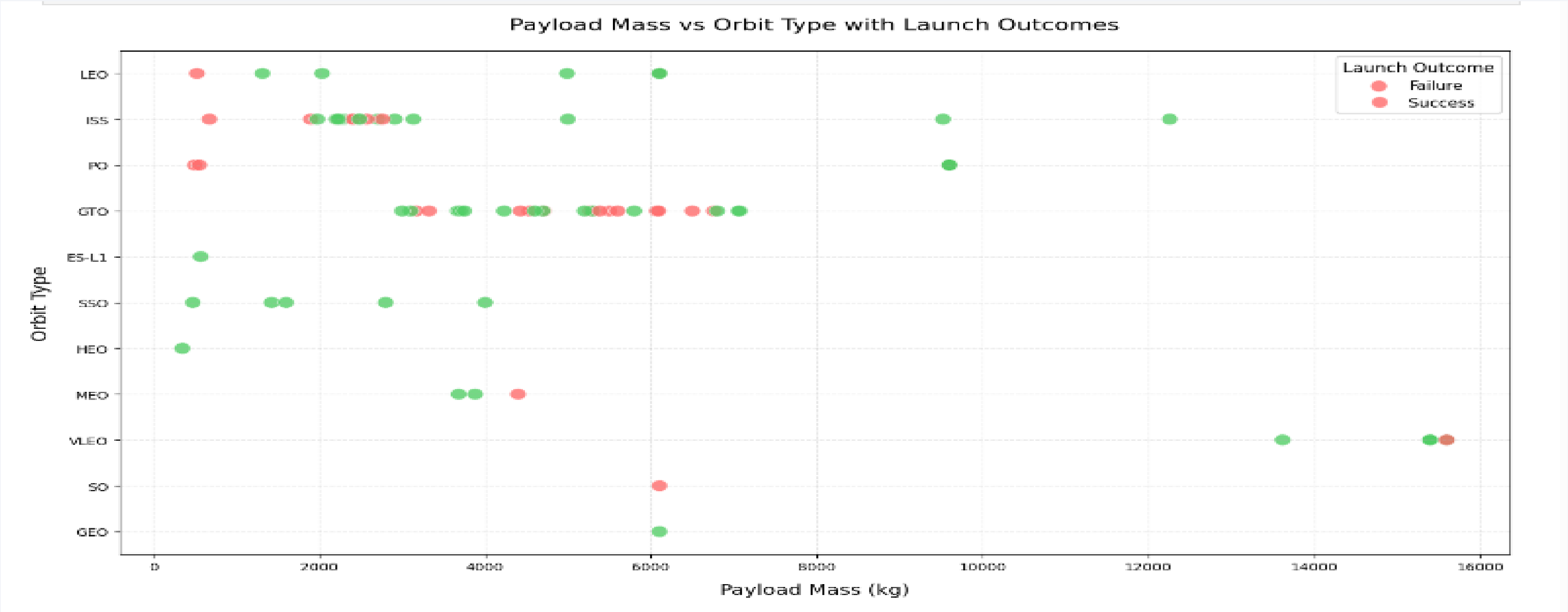


github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/edadataviz.ipynb

EDA with Data Visualization (6)

- **Payload Mass vs. Orbit scatter point chart**

- Similarly, we can plot the Payload Mass vs. Orbit scatter point charts to reveal the relationship between Payload Mass and Orbit type. 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.

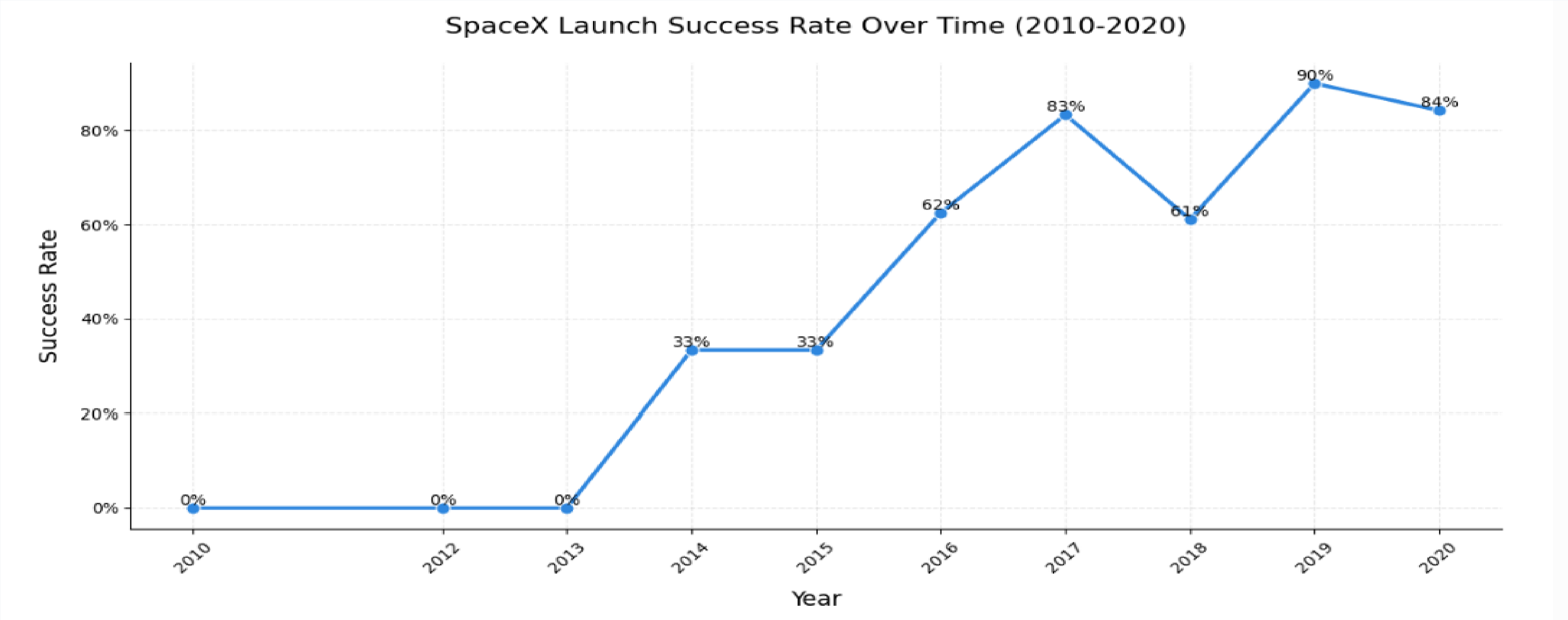


github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/edadataviz.ipynb

EDA with Data Visualization (7)

- **Success Rate Line Graph**

- Wed did plot a line chart with x axis to be Year and y axis to be average success rate, to get the average launch success trend. We observed that the success rate since 2013 kept increasing till 2020



github url: https://github.com/Dibongaz/Data-Science-Notebook-/blob/main/edadataviz.ipynb

EDA with SQL (1)

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

```
# Correct way to execute SQL query in Jupyter notebook
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

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

```
%sql SELECT * FROM SPACEXTABLE 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_NASA_CRS  FROM SPACEXTABLE WHERE "Customer" LIKE '%NASA%CRS%' OR "Customer" LIKE '%NASA (CRS)%';
```

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

```
%sql SELECT AVG("Payload_Mass__kg_") AS Average_Payload_Mass_F9_v1_1  FROM SPACEXTABLE  WHERE "Booster_Version" = 'F9 v1.1';
```

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

```
%sql SELECT MIN("Date") AS First_Successful_Ground_Landing_Date FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)';

%sql SELECT "Flight_Number", "Date", "Booster_Version", "Mission_Outcome", "Landing_Outcome"
%sql FROM SPACEXTABLE
%sql WHERE "Landing_Outcome" = 'Success (ground pad)'
%sql ORDER BY "Date" ASC LIMIT 1;
```


github url: https://github.com/Dibongaz/Data-Science-Notebook-/jupyter-labs-eda-sql-coursera_sqllite.ipynb

EDA with SQL (2)

- 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 "Landing_Outcome" = 'Success (drone ship)' AND "Payload_Mass__kg_" > 4000 AND "Payload_Mass__kg_" < 6000;
```

- List the total number of successful and failure mission outcomes

```
# Correct way to count successful and failed missions

%sql success_failure_counts = %sql SELECT \     CASE \        WHEN "Mission_Outcome" LIKE '%Success%' THEN 'Success' \        ELSE 'Failure' \     END AS Outcome_Type, \     COUNT(*) AS Count \  FROM SPACEXTABLE \  GROUP BY Outcome_Type;
success_failure_counts
```

-

github url: https://github.com/Dibongaz/Data-Science-Notebook-/jupyter-labs-eda-sql-coursera_sqllite.ipynb

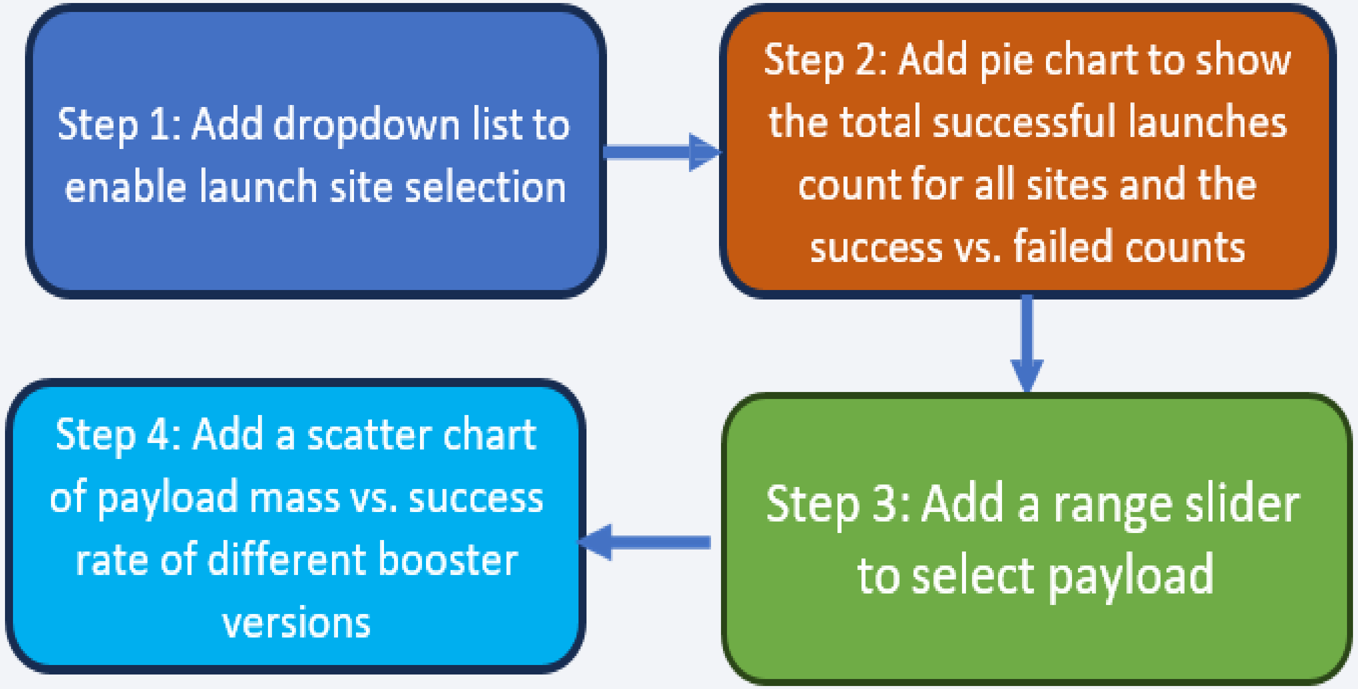
Build an Interactive Map with Folium

Folium makes it easy to visualize data that's been manipulated in Python on an interactive leaflet. We use the latitude and longitude coordinates for each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.

MAP OBJECTS	CODE	RESULT
Map Marker	folium.Marker	Map object to make a mark on a map
Icon Marker	folium.Icon	Creating an icon on a map
Circle Marker	folium.Circle	Creating a circle where Marker is being placed
PolyLine	Ffolium.PolyLine	Creating a line between points
Marker Cluster Object	MarkerCluster	Simplifying a map containing many markers with the same cordinate
AntPath	folium.plugina.AntPath	Creating an animated line between points

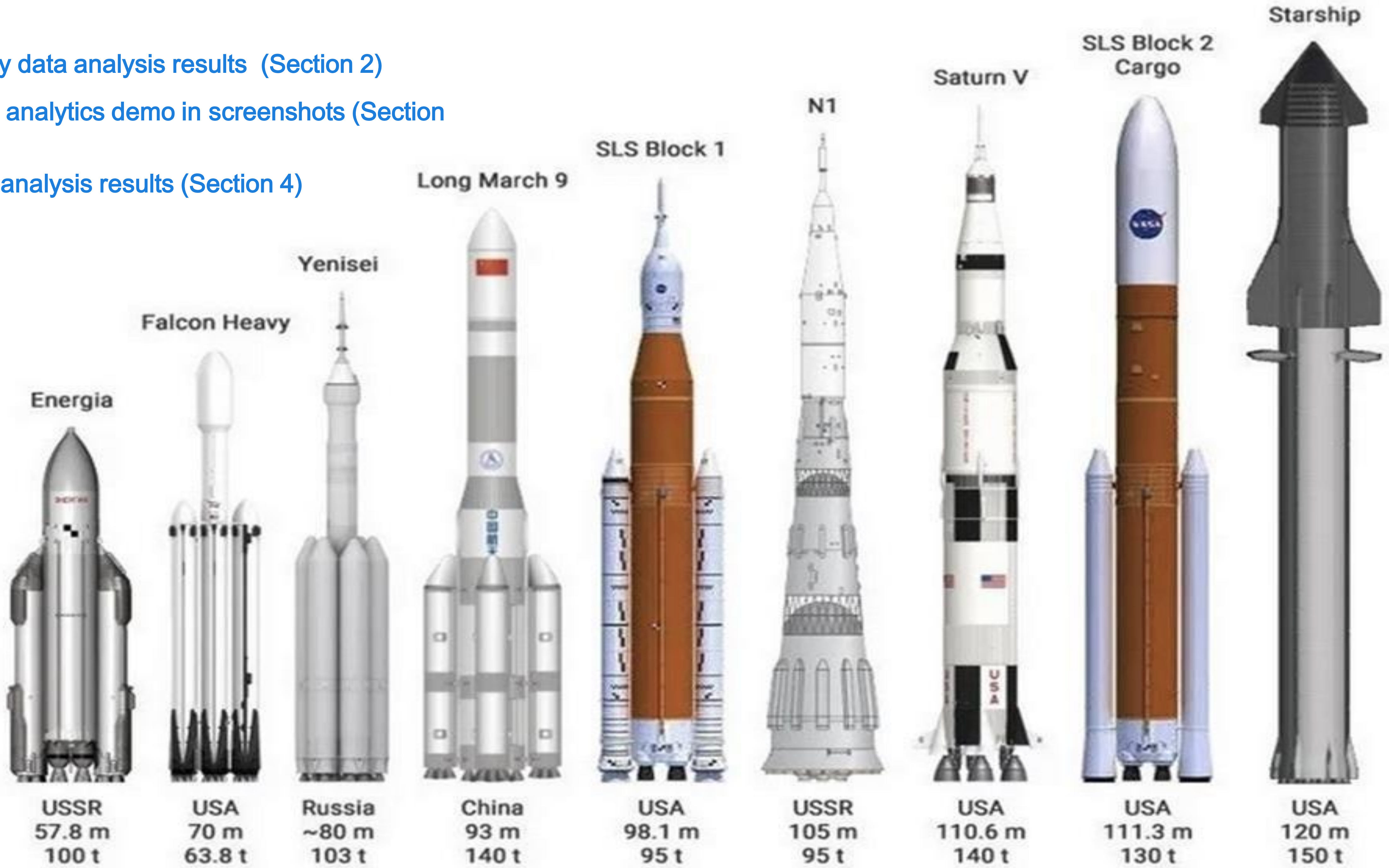
Build a Dashboard with Plotly Dash

- Pie Charts to show the total success for all sites or by certain launch site.
- Scatter plots to show the correlation between payloads and success for all sites.
- The dashboard is built using Dash web



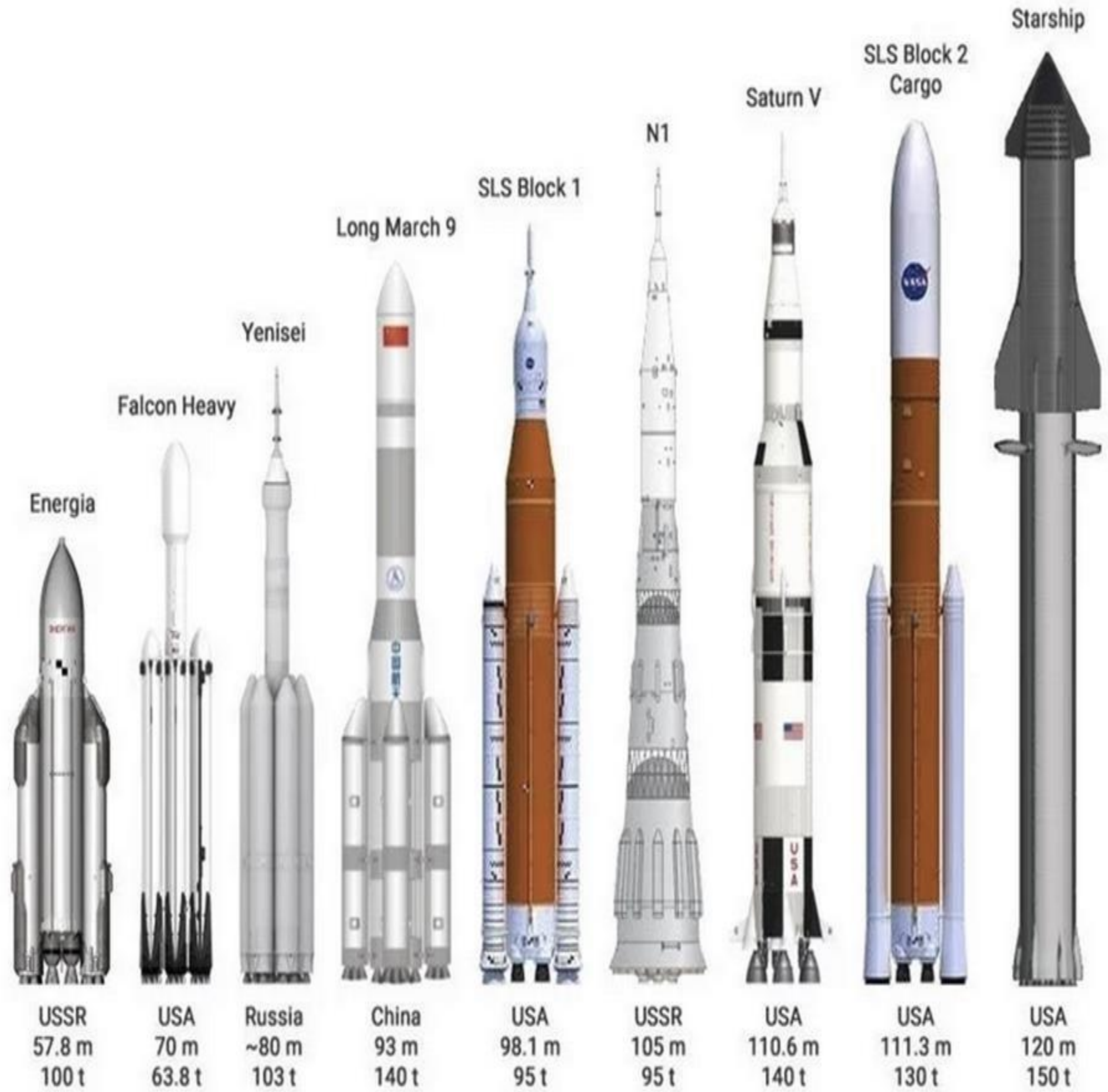
RESULTS

- Exploratory data analysis results (Section 2)
- Interactive analytics demo in screenshots (Section 3)
- Predictive analysis results (Section 4)



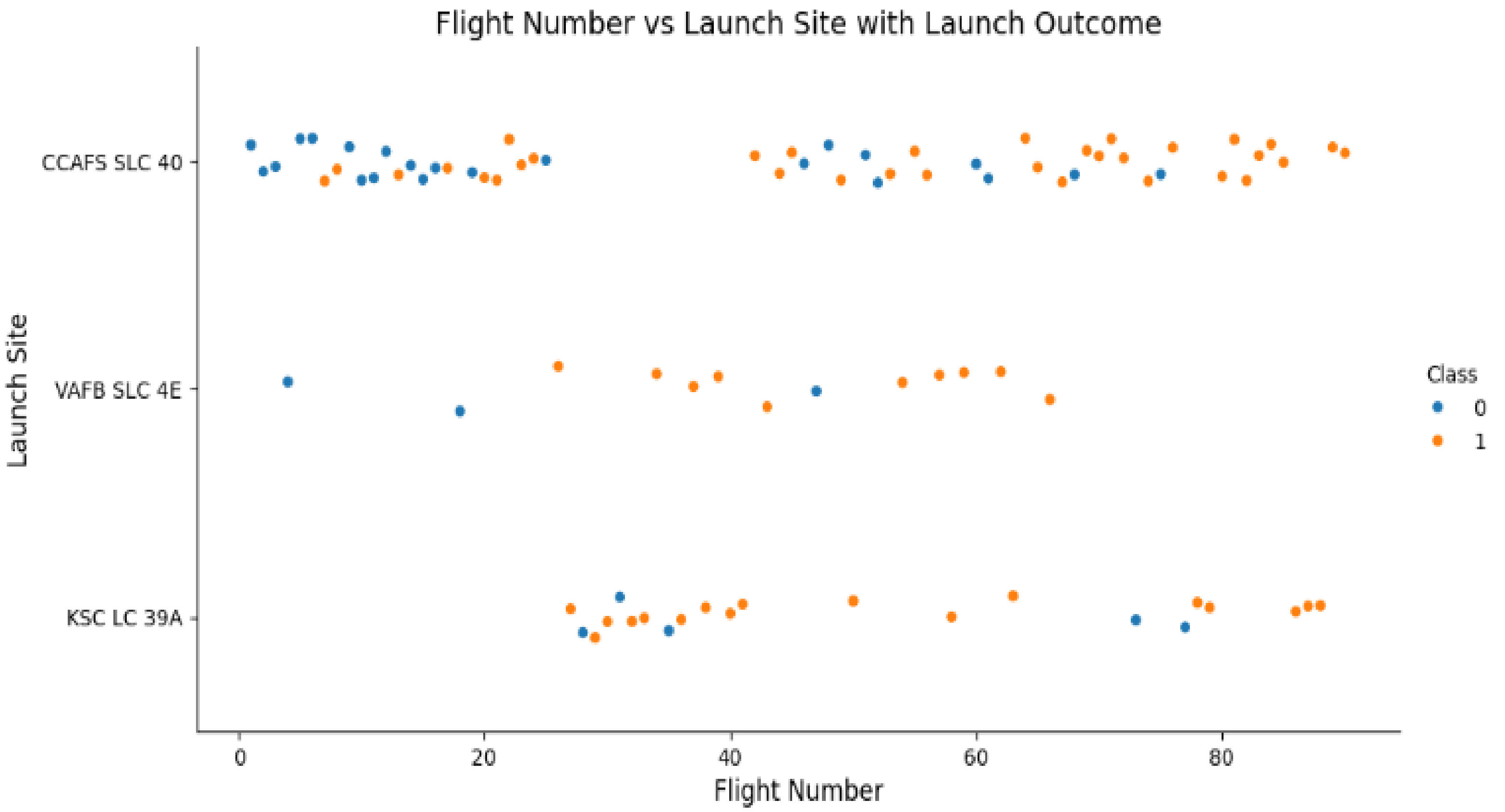
Section 2

Insights drawn from EDA



Flight Number vs. Launch Site

```
# Plot the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect=2)
plt.xlabel("Flight Number", fontsize=12)
plt.ylabel("Launch Site", fontsize=12)
plt.title("Flight Number vs Launch Site with Launch Outcome", fontsize=14)
plt.show()
```

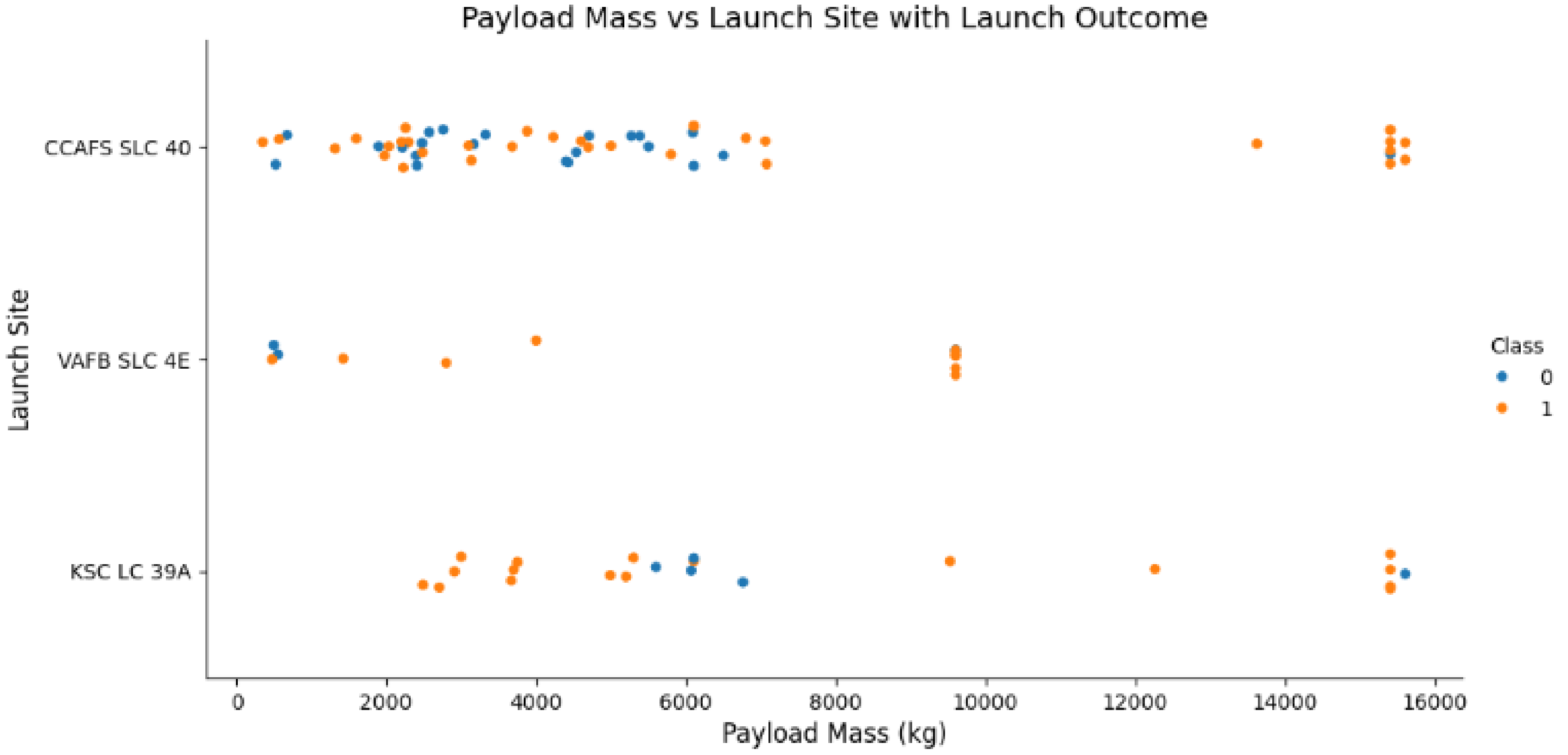


Flight Number vs. Launch Site Scatter Plot

Use the function catplot to plot Flight Number vs Launch Site, set the parameter x parameter to Flight Number set they to Launch Site and set the parameter hue to 'class'. '

Payload vs. Launch Site

```
# Plot the relationship between Payload Mass and Launch Site
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect=2)
plt.xlabel("Payload Mass (kg)", fontsize=12)
plt.ylabel("Launch Site", fontsize=12)
plt.title("Payload Mass vs Launch Site with Launch Outcome", fontsize=14)
plt.show()
```

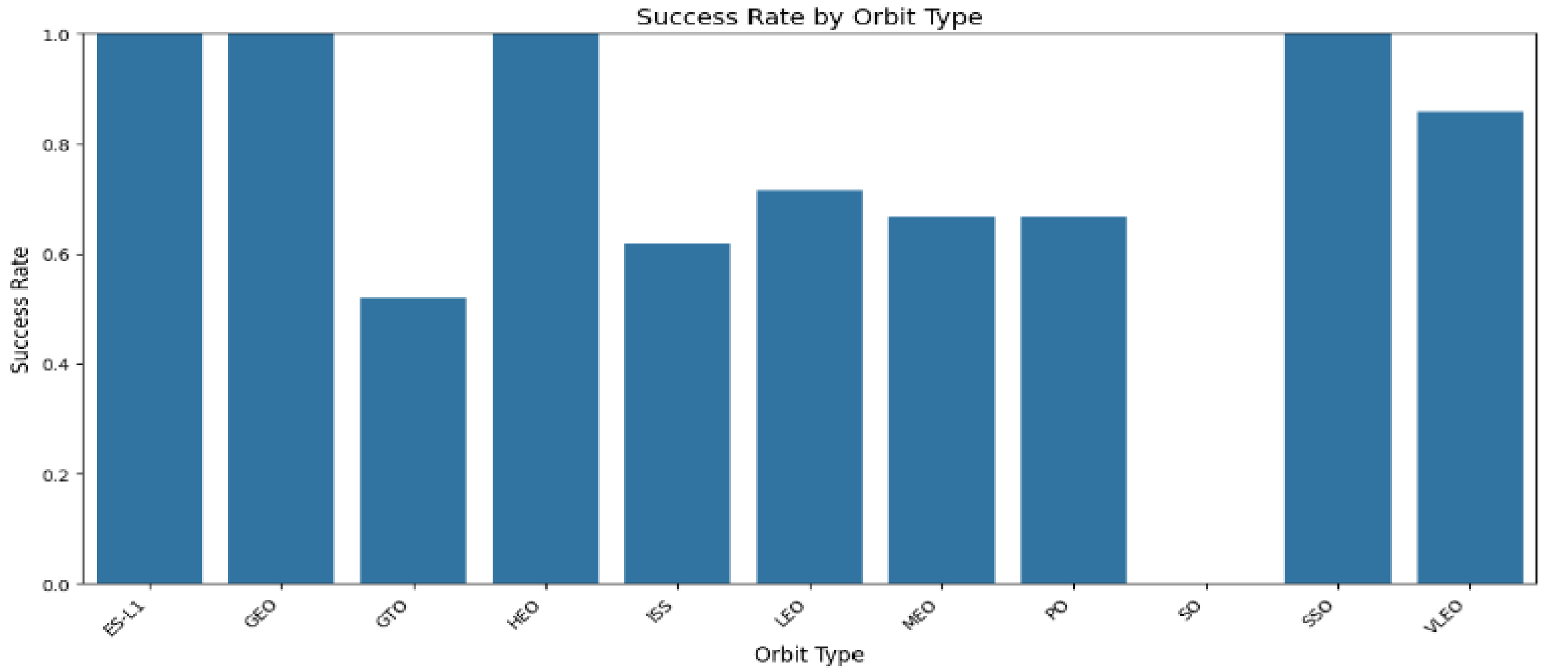


Launch Site vs. Payload Mass Scatter Plot

If you observe Payload Mass Vs. Launch Site scatter point chart you 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

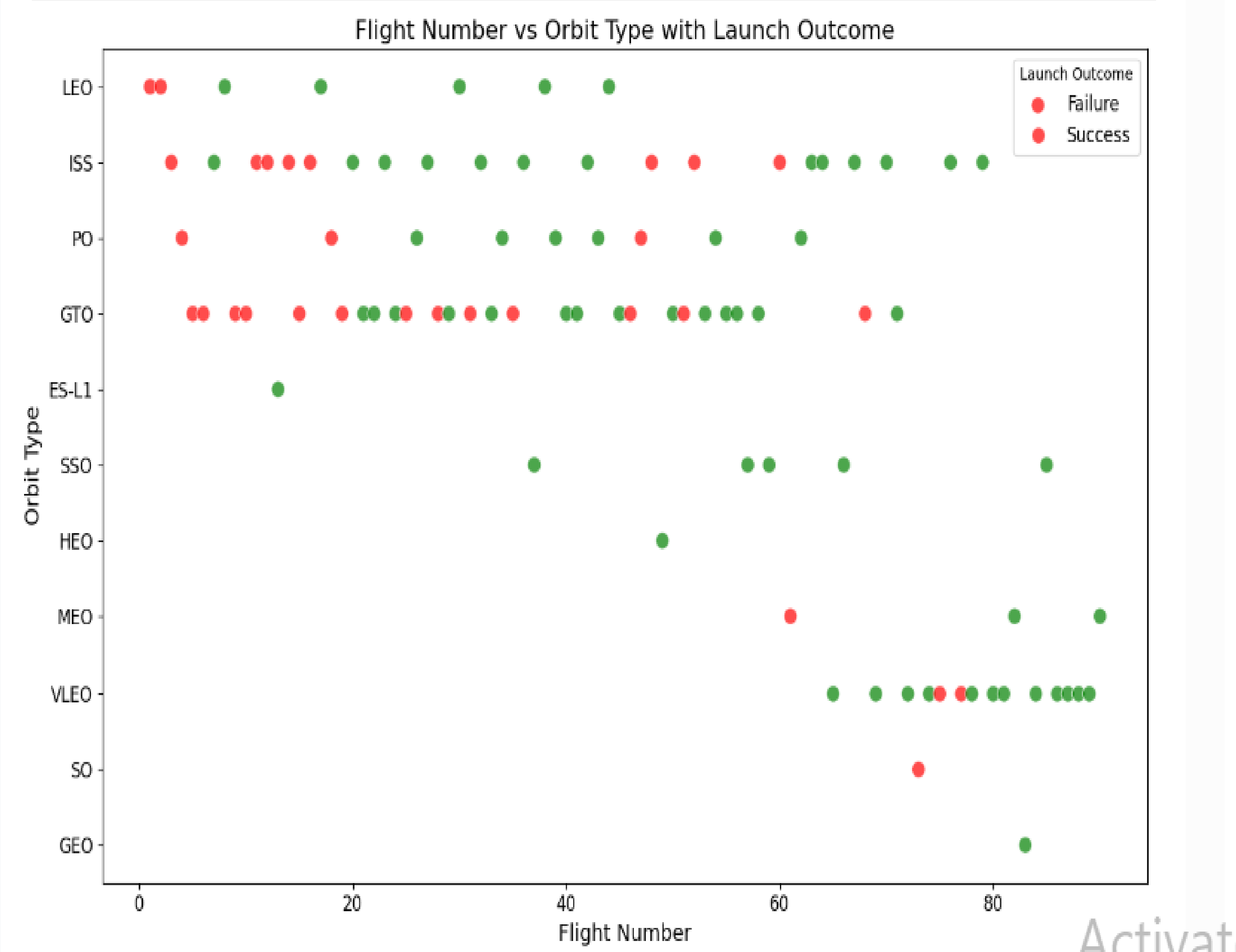
```
# Calculate success rate by orbit type
orbit_success = df.groupby('Orbit')['Class'].mean().reset_index()

# Create the bar plot
plt.figure(figsize=(12, 6))
sns.barplot(x='Orbit', y='Class', data=orbit_success)

# Customize the plot
plt.title('Success Rate by Orbit Type', fontsize=14)
plt.xlabel('Orbit Type', fontsize=12)
plt.ylabel('Success Rate', fontsize=12)
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for better readability
plt.ylim(0, 1)  # Set y-axis limits from 0 to 1 since success rate is a probability

# Display the plot
plt.tight_layout()
plt.show()
```



Success Rate vs. Orbit Bar Chart

We want to visually check if there are any relationship between success rate and orbit type. We created a bar chart for the success rate of each orbit. Analyse the plotted bar chart to identify which orbits have the highest success rates.

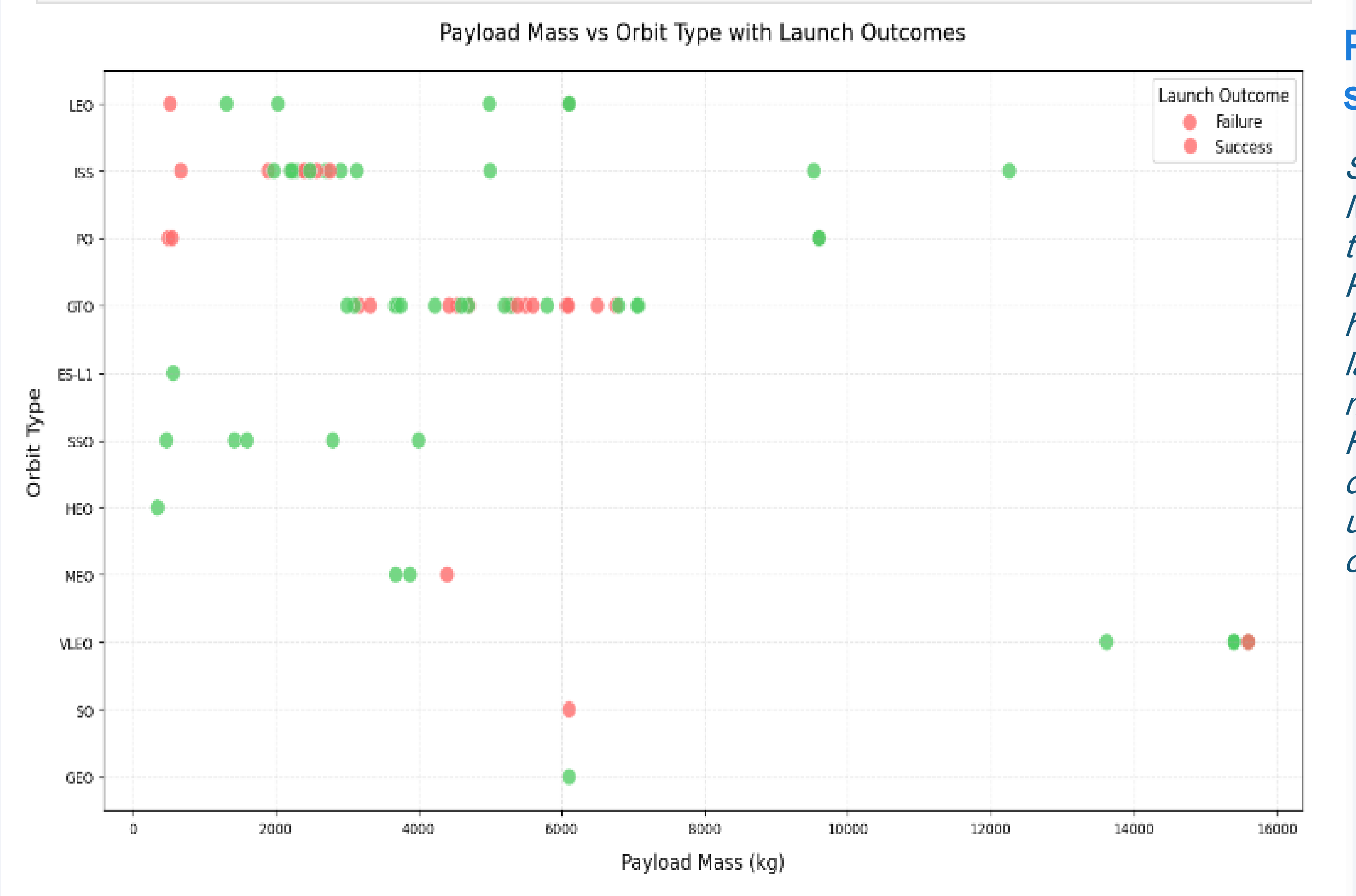
Flight Number vs. Orbit Type



Flight Number and Orbit type

For each orbit, we wanted to see if there is any relationship between Flight Number and Orbit type. You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

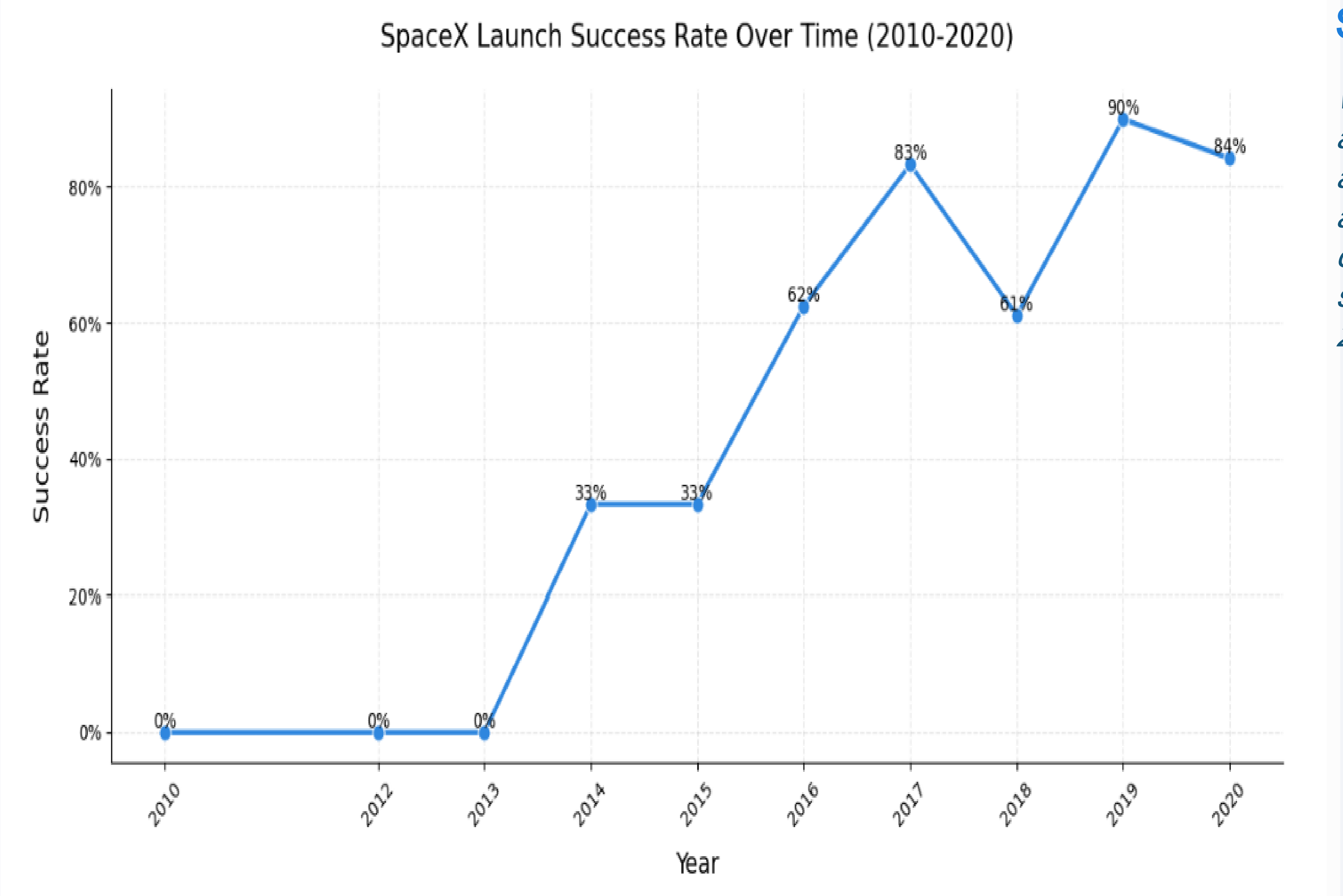
Payload vs. Orbit Type



Payload Mass vs. Orbit scatter point chart

Similarly, we can plot the Payload Mass vs. Orbit scatter point charts to reveal the relationship between Payload Mass and Orbit type. 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



Success Rate Line Graph

Wed did plot a line chart with x axis to be Year and y axis to be average success rate, to get the average launch success trend. We observed that the success rate since 2013 kept increasing till 2020

All Launch Site Names

```python
features= df[['LaunchSite']]
features.head()
```

	LaunchSite
0	CCAFS SLC 40
1	CCAFS SLC 40
2	CCAFS SLC 40
3	VAFB SLC 4E
4	CCAFS SLC 40

- Here are the names of the unique launch sites and the query structure for obtaining these sites is shown.

Launch Site Names Begin with 'CCA'

- 5 records for launch sites begin with the string 'CCA’ and the simple query used for obtaining the information is shown below.

```
[ ]  1 %sql SELECT * FROM SPACEXTABLE 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
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

Total Payload Mass

The query for obtaining the total payload mass is shown below

```
%sql select SUM(PAYLOAD_MASS__KG_) from SPACEXTABLE where "Customer" like 'NASA (CRS)%'
```

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

SUM(PAYLOAD_MASS__KG_)
48213

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 = 2534.7 Kg.
- Furthermore, the query used to calculate the average payload mass carried by booster F9 v1.1 is shown below.

```
%sql select AVG(PAYLOAD_MASS__KG_) from SPACEXTABLE where "Booster_Version" LIKE 'F9 v1.1%'
```

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

AVG(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

- The first successful landing outcome on a ground pad was in 2015-12-22.
- The query for obtaining this result is shown below.

```
[ ]  1 %sql SELECT MIN(Date) AS first_successful_landing_date FROM SPACEXTABLE  WHERE landing_outcome = 'Success (ground pad)';
```

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

first_successful_landing_date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 is shown below.
- The query used in obtaining this information is shown below.

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 FROM SPACEXTABLE WHERE landing_outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_
```

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

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failed missions is as follows:

  Failure (in flight)= 1

  Successful number of flights= 98

- The query result is shown below.

```
[ ]  1 %sql SELECT mission_outcome, COUNT(*) AS total_count FROM SPACEXTABLE WHERE  mission_outcome IN ('Success', 'Failure (in flight)') GROUP BY mis
     2
```

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

Mission_Outcome	total_count
Failure (in flight)	1
Success	98

Boosters Carried Maximum Payload

- List of the boosters which have carried the maximum payload mass are shown below.
- The query used in obtaining the booster names is shown below.

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_ = (SELECT MAX (PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
```

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

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

2015 Launch Records

- List of the failed ”landing_outcomes” in drone ship, their booster version, and the launch site name during year 2015 is shown below.
- The query used in obtaining the information is shown below

```
[ ]  1 %sql SELECT CASE WHEN substr(Date, 6, 2) = '01' THEN 'January' WHEN substr(Date, 6, 2) = '02' THEN 'February' WHEN substr(Date, 6, 2) = '03' TH
     2
```

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

month_name	Booster_Version	Launch_Site	Landing_Outcome
January	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
April	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- A rank of the count of landing outcomes (such as Failure (drone ship) or success (ground pad)) between the dates 2010-06-04 and 2017-03-20, in descending order is shown below.
- The query used to obtain the results is shown below

```
[ ]  %sql SELECT landing_outcome, COUNT(*) AS outcome_count FROM SPACEXTABLE WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY landing_outc
```

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

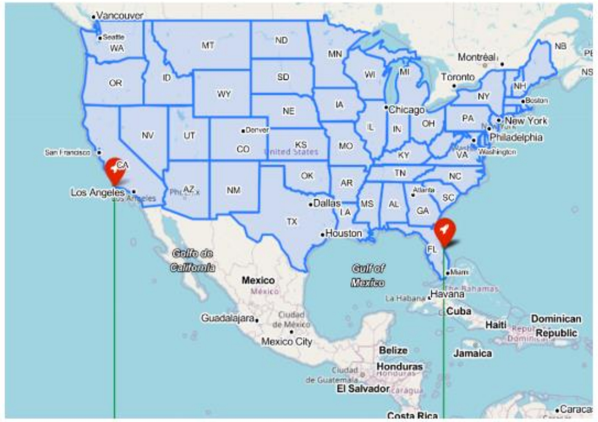
Landing_Outcome	outcome_count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

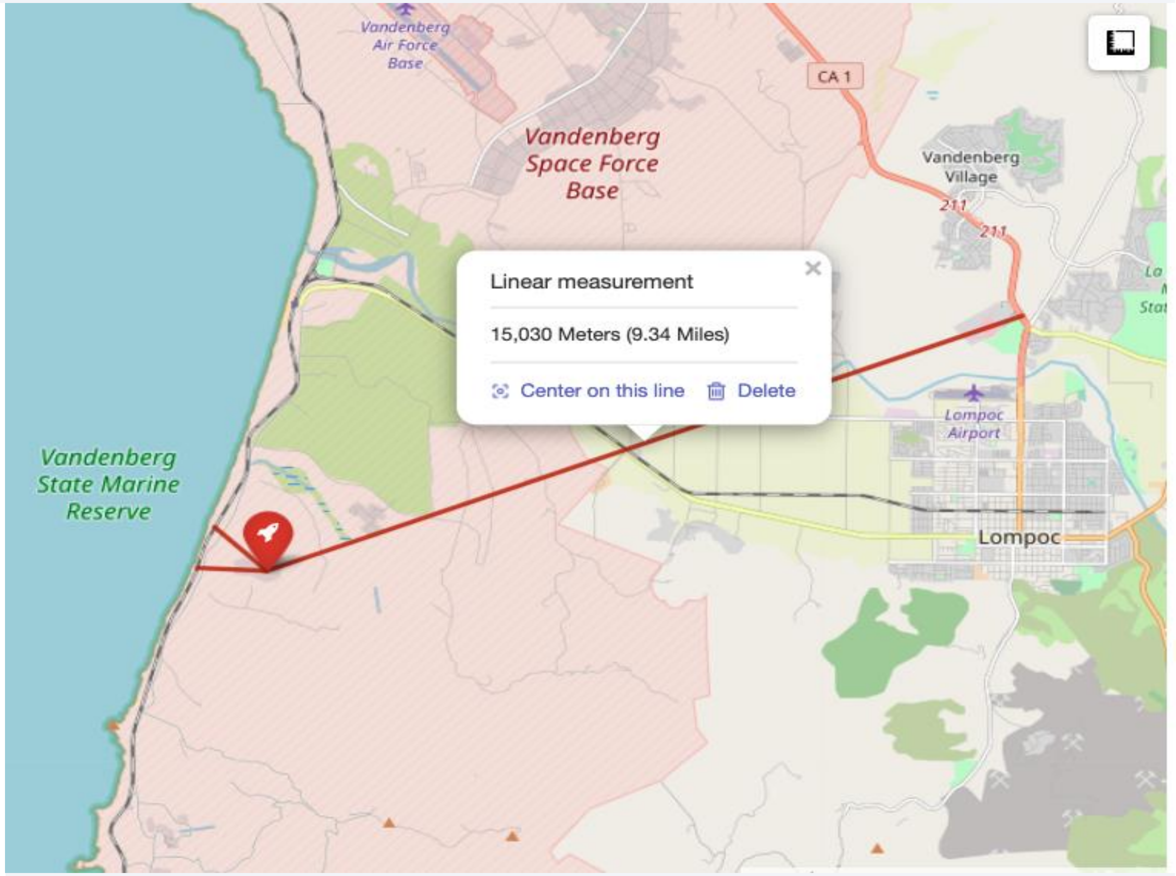
Section 3

Launch Sites Proximities Analysis

USA Launch Sites in California and Florida (1)

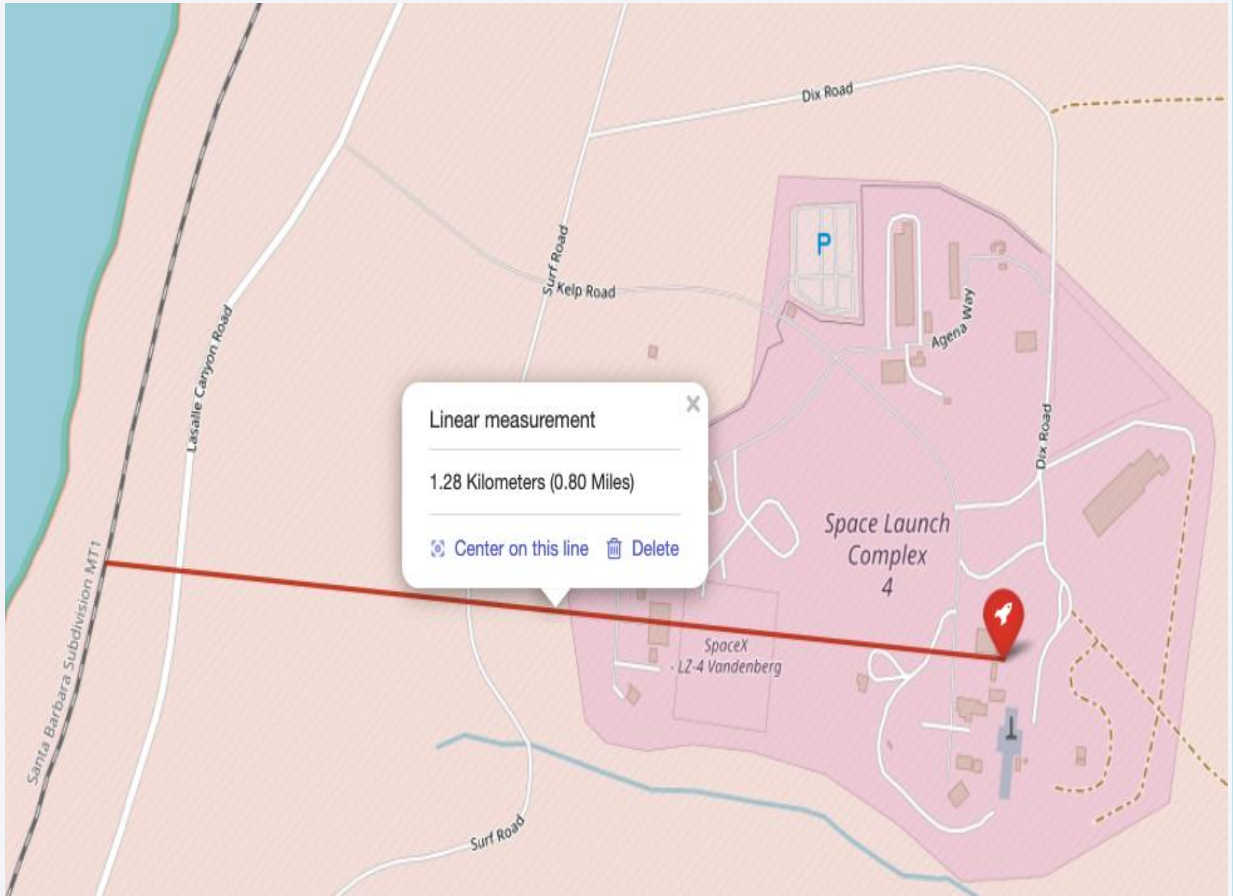
Most of Launch sites considered in this project are in proximity to the Equator line. Launch sites are made at the closest point possible to Equator line, because anything on the surface of the Earth at the equator is already moving at the maximum speed (1670 kilometers per hour). For example launching from the equator makes the spacecraft move almost 500 km/hour faster once it is launched compared half way to north pole.

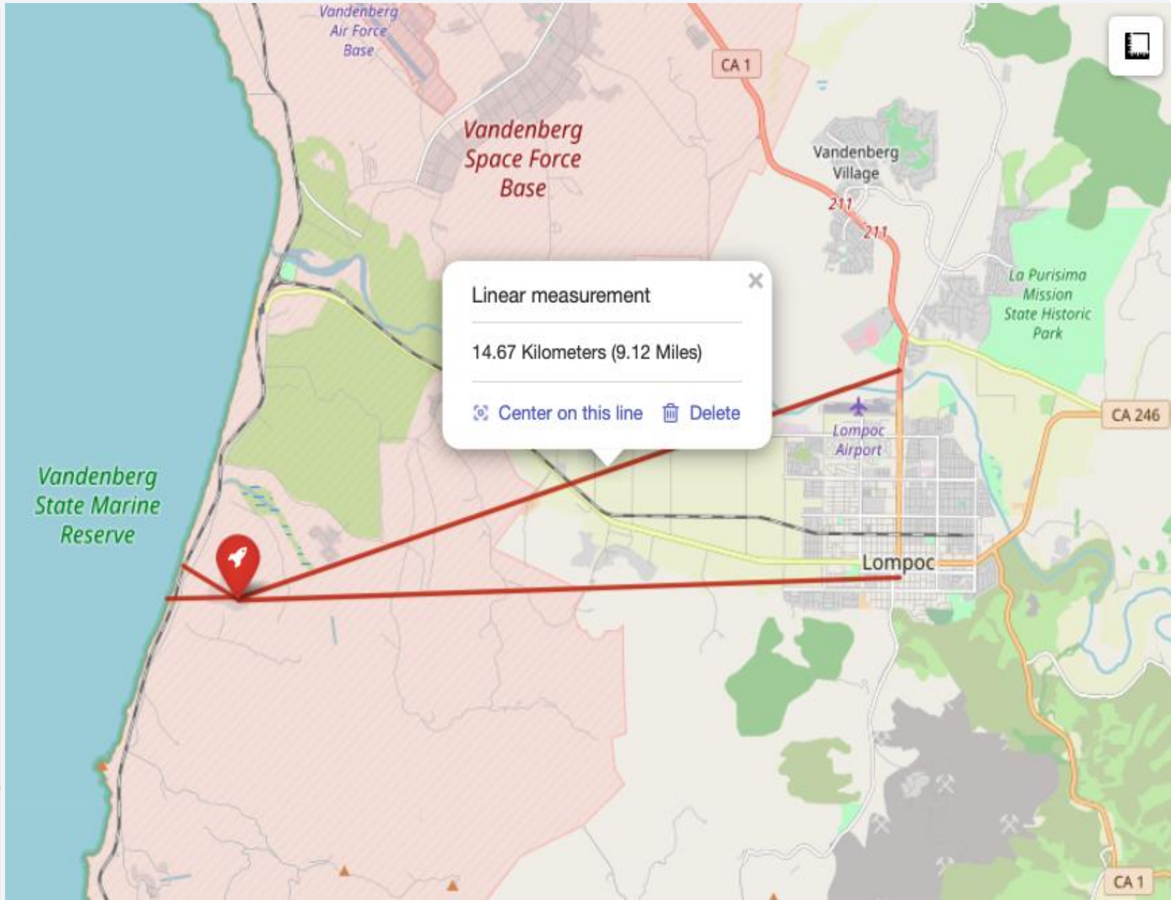




USA Launch Sites in California and Florida (2)

All launch sites considered in this project are in very close proximity to the coast While starting rockets towards the ocean we minimize the risk of having any debris dropping or exploding near people.

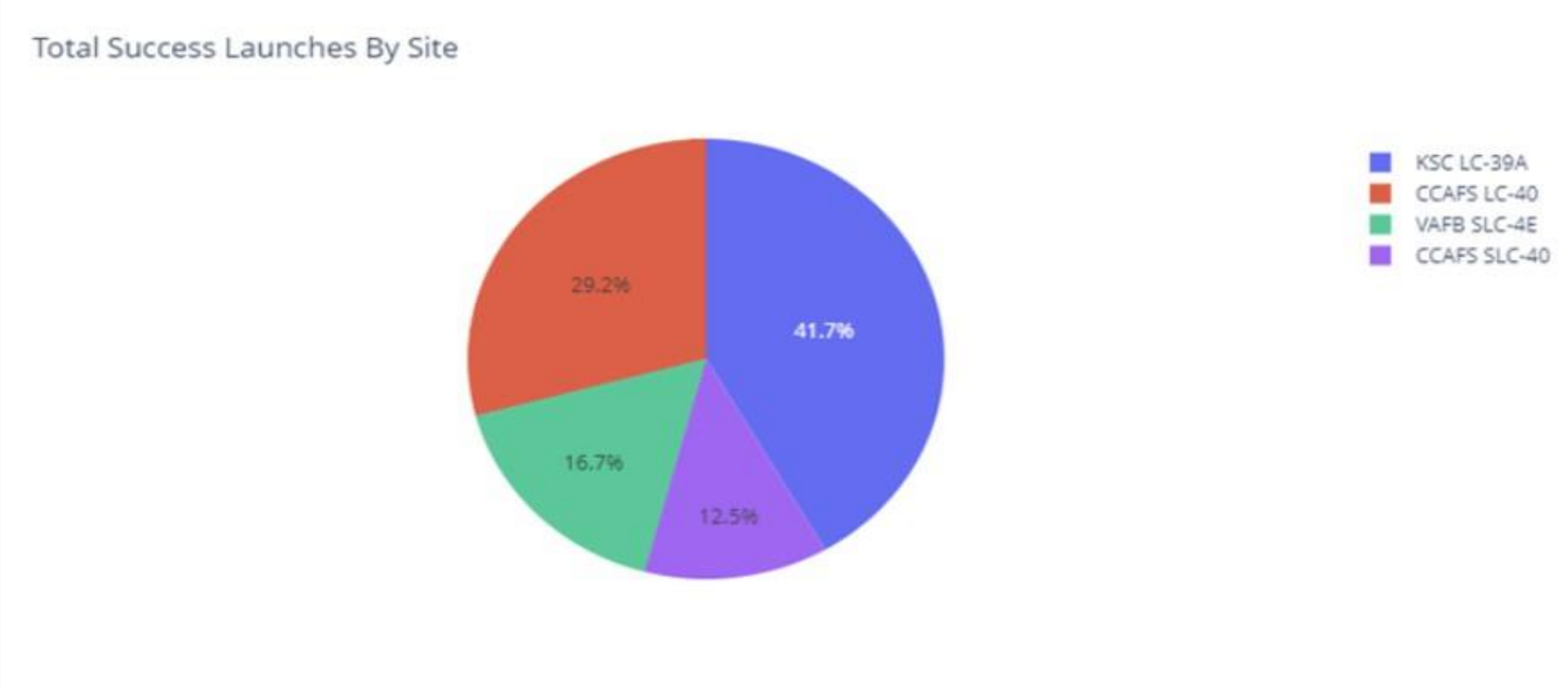




Section 4

Build a Dashboard with Plotly Dash

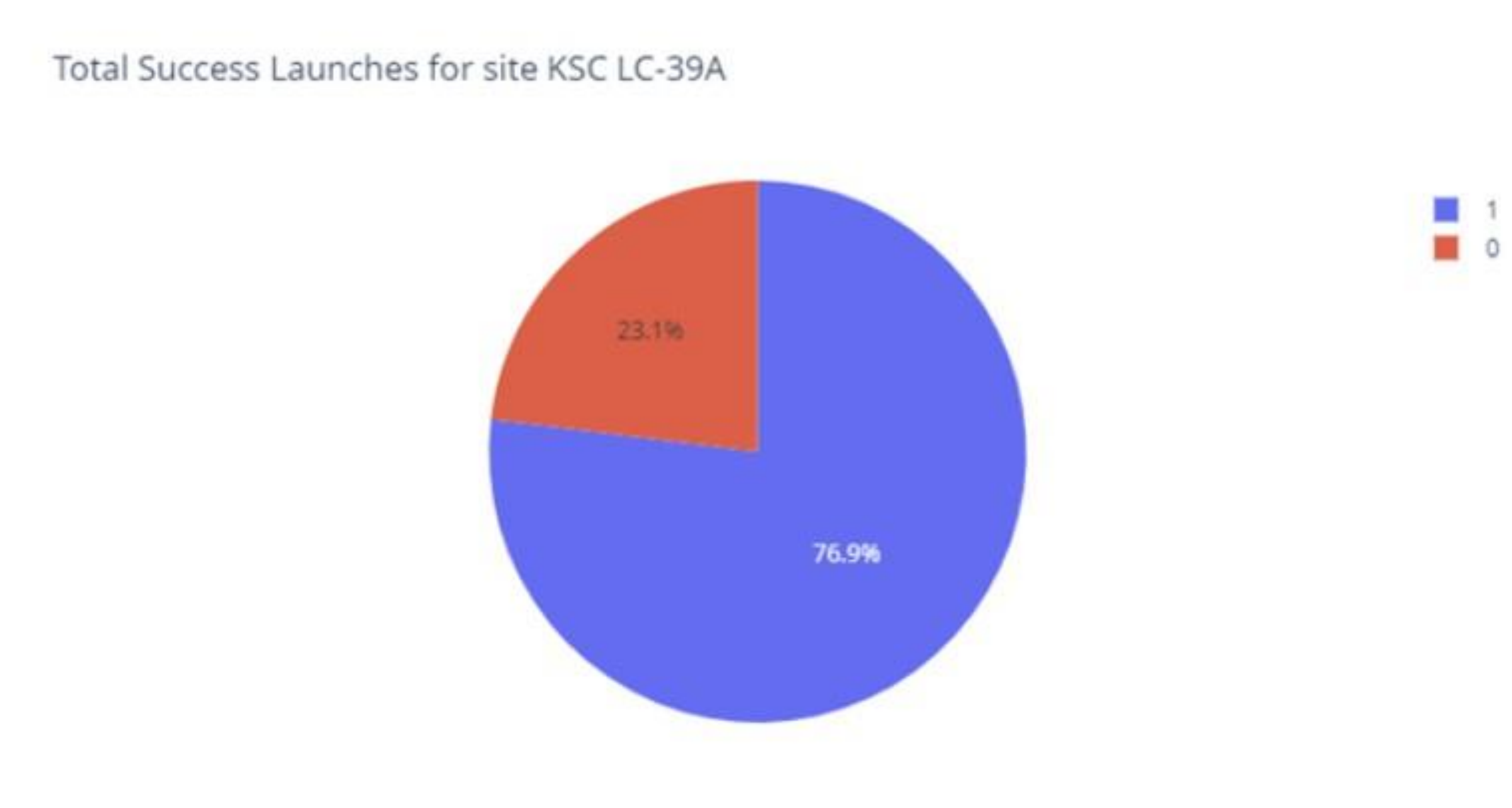
Total Launch Success for All Site



The highest success launch rates were recorded at these sites :

- 1. KSC LC-39A (41.7%)
- 2. CCAFS LC-40 (29.2%)

Launch Site Success Rate for KSC LC-39



KSC LC-39 success rate is 76.9%

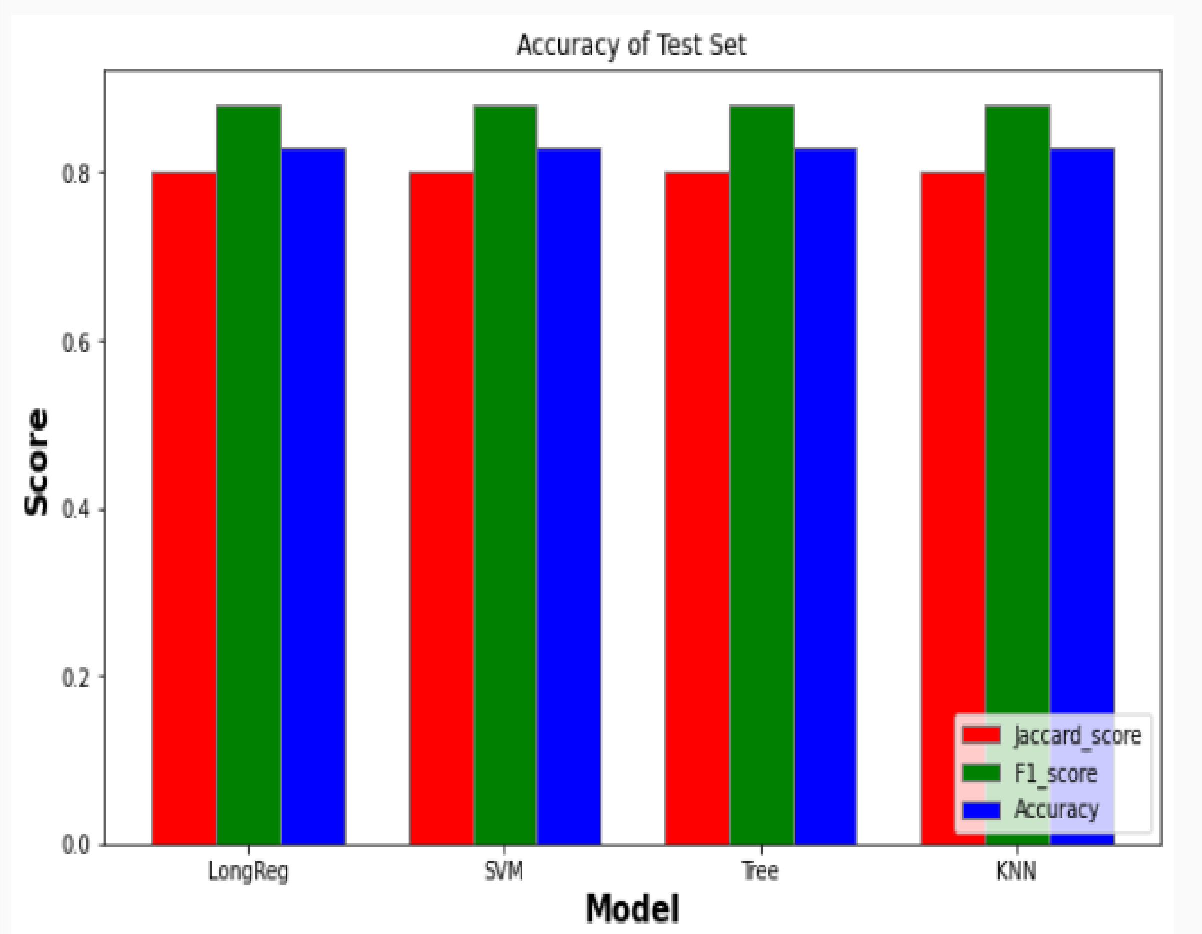
Payload vs. Launch Outcome for All Sites

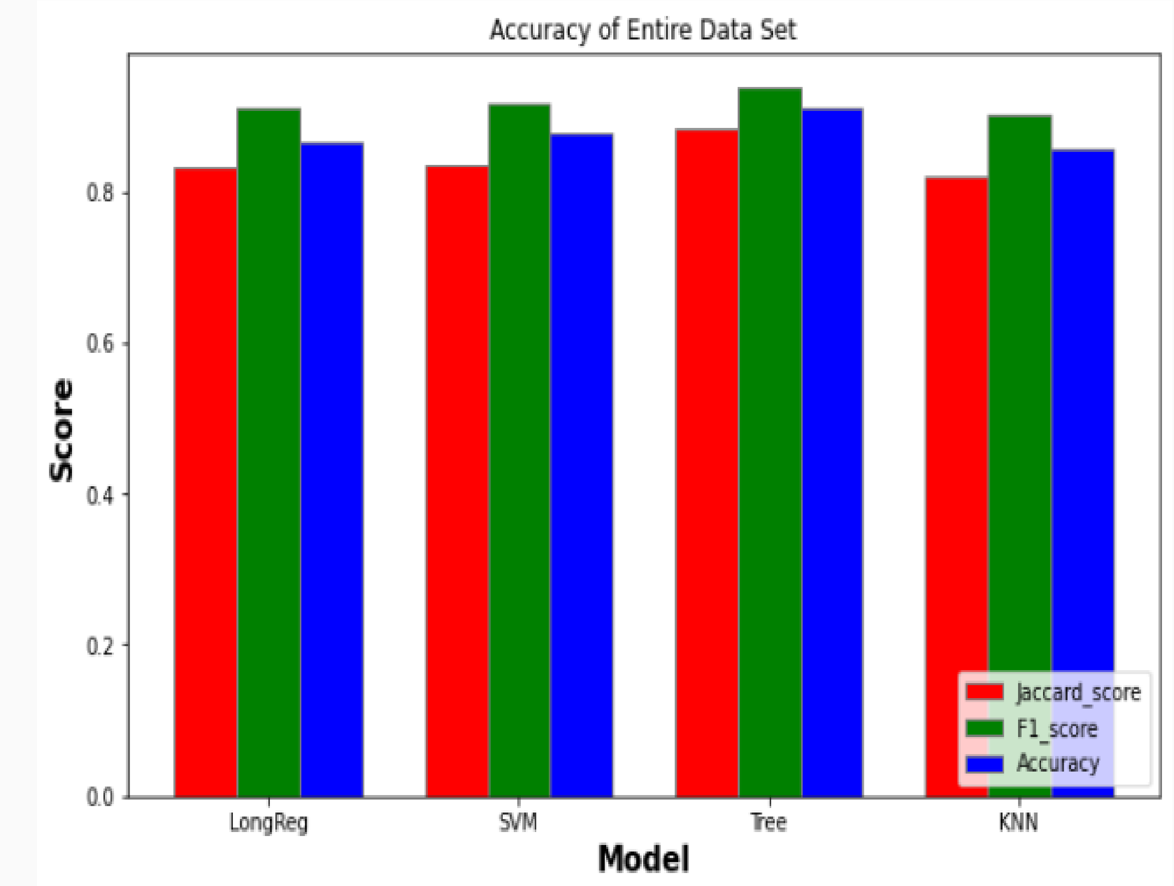
Highest success rate for payloads is between 2000 and 5500 Kgs

Section 5

Predictive Analysis (Classification)

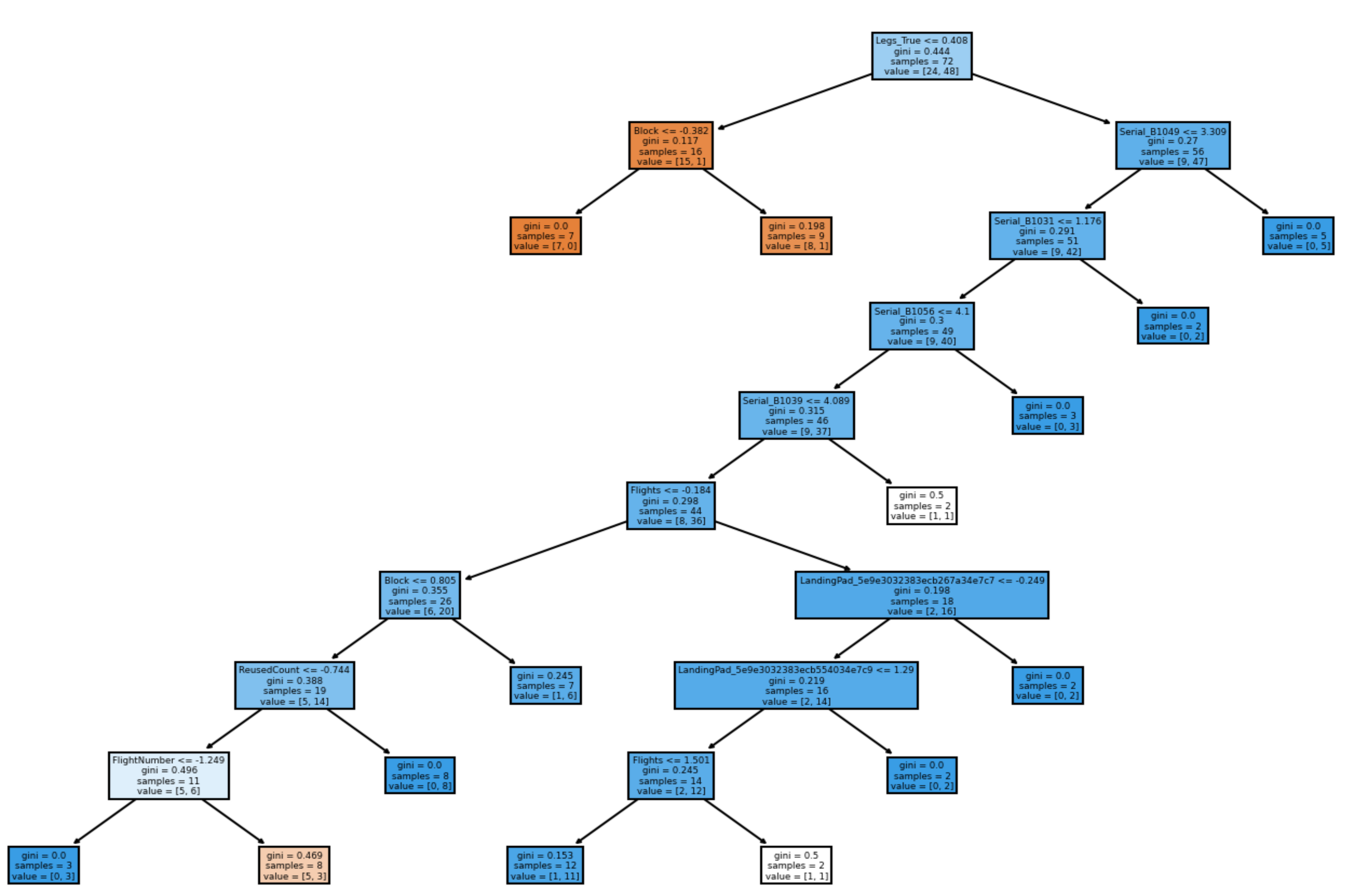
Classification Accuracy (1)





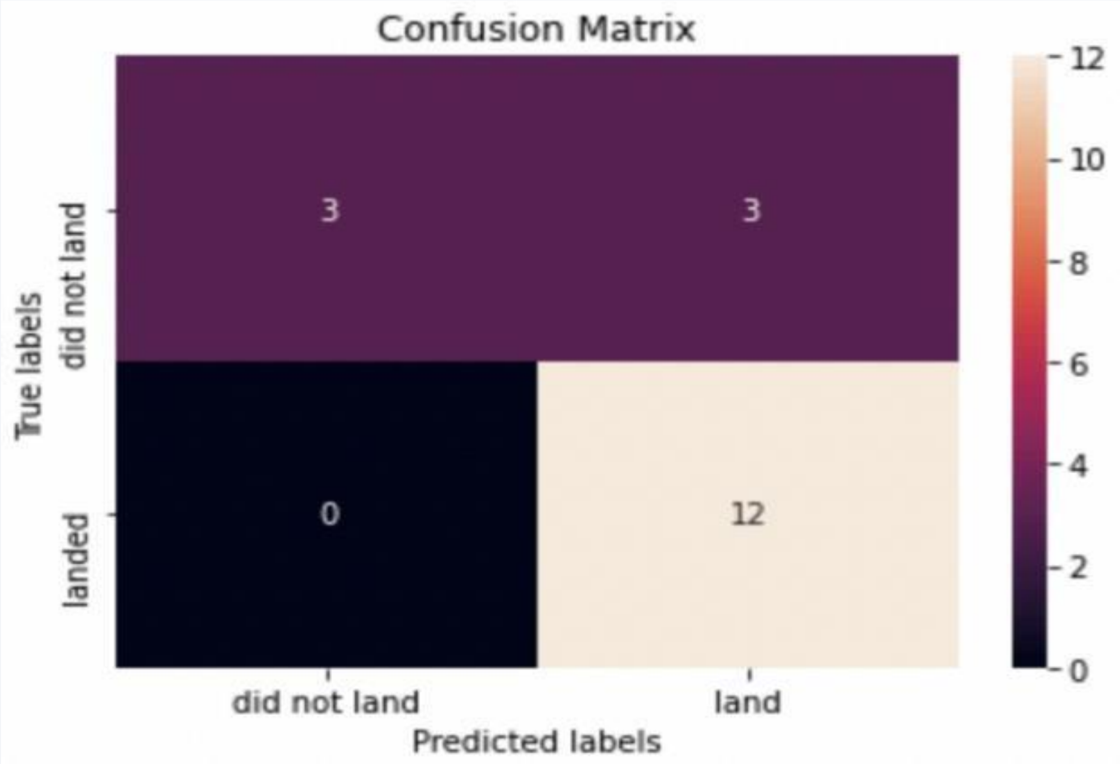
- Using the test set the same accuracy results were obtained from the four models.

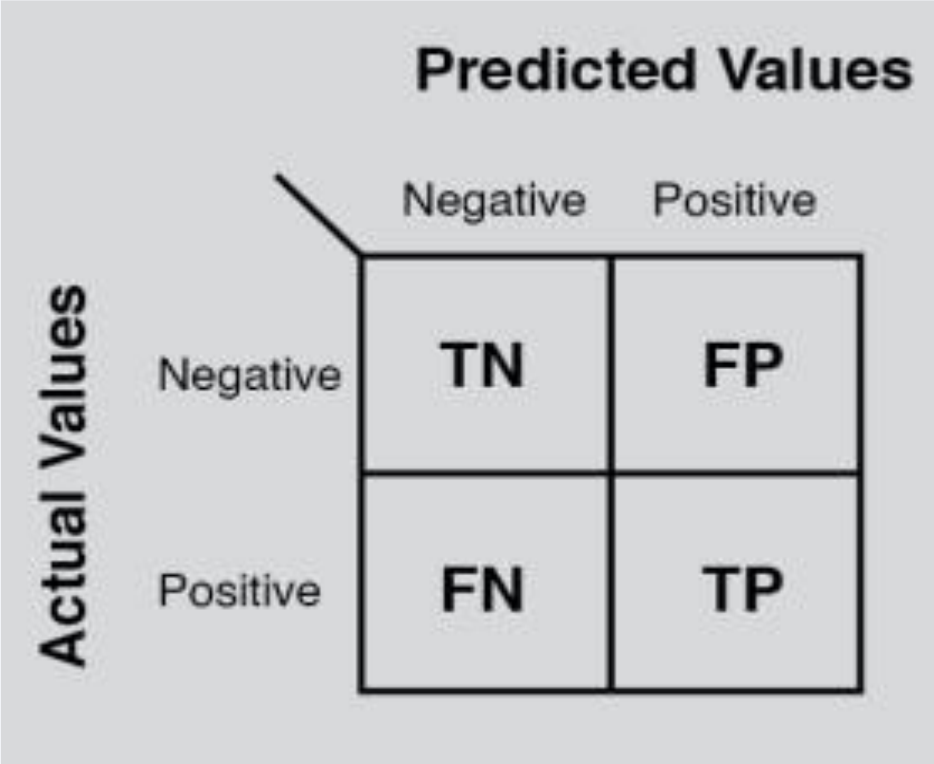
Classification Accuracy (2)



- The Decision Tree Model provided the best accuracy results for the entire data set.

Confusion Matrix

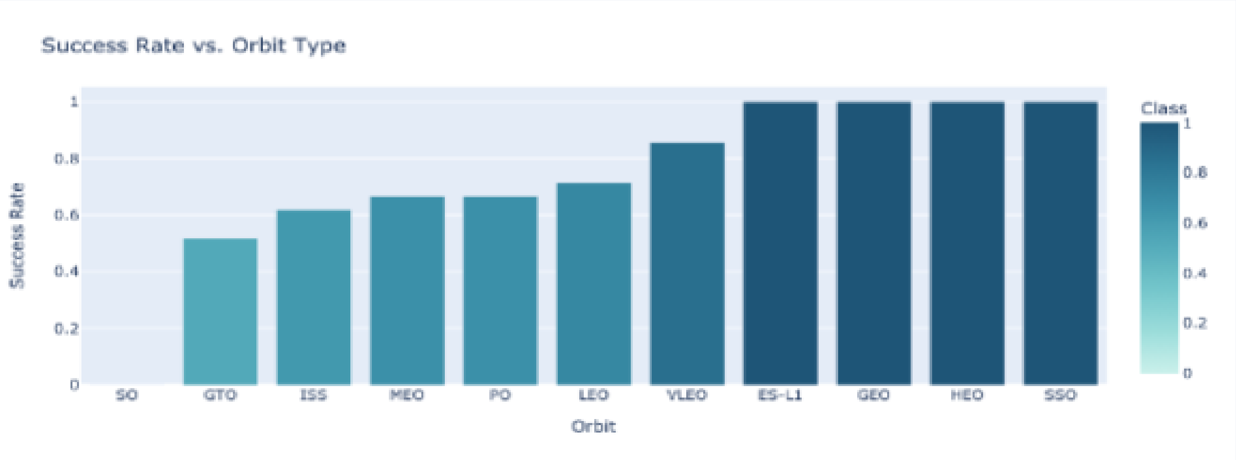


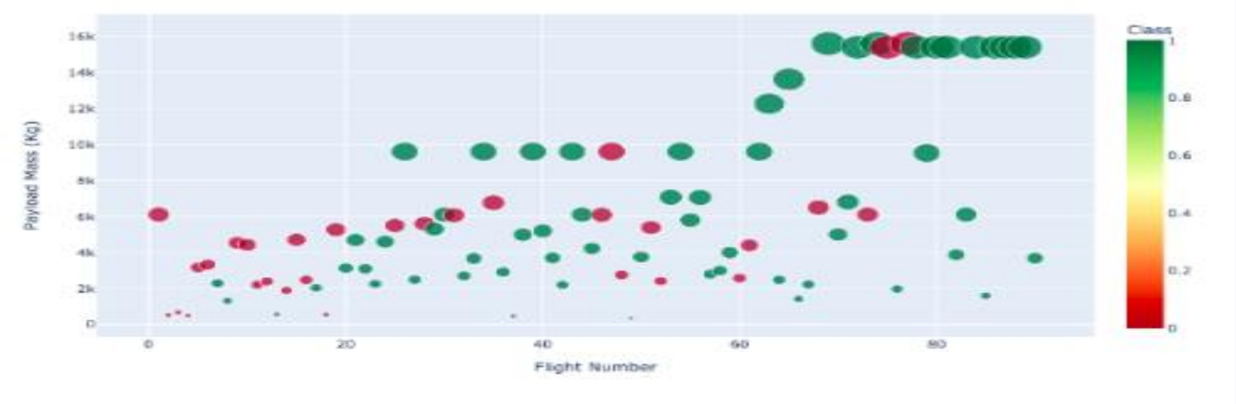


- The confusion matrix analysis suggests that the best performing model is the Logistic Regression model.
- The confusion matrix predicts 13 true positives, 3 false positives, 3 true positive, and 0 false negative.

Conclusions

- The success rate for the rocket launches increased after 2013.
- Orbits GEO, HEO, ES-L1 and SSO have 100% launch success rate.
- Launch site KSC LC-39A has the highest success rate.
- The Decision Tree model is the best ML algorithm for analyzing the SpaceX data set and provided the best accuracy results





Thank you!