IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Hsu Jih-Hung

2023/07/09

# Outline

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

# Executive Summary

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

# Introduction

- Project background and context

  SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- Problems you want to find answers

  In this capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

-

# Data Collection

- Describe how data sets were collected.

1. Calling SpaceX API with Python request library, the response includes rocket launch data in JSON format. We then converted JSON into a Pandas data frame for subsequent data analysis process, this was done by using Python Pandas API.

2. Web scraping to collect Falcon 9 historical launch records from a Wikipedia page, with Python BeautifulSoup and request libraries, we can extract the Falcon 9 launch data in HTML format, then parsing HTML table with and extract data, then convert extracted data into Pandas data frame for data analysis process.

# Data Collection – SpaceX API

- Make request to SpaceX API, when retrieved response as json format then convert to data frame and normalize data.

- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

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:

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

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

```
[12]  response.status_code
                                                                                                                                                    Python
...   200
```

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

```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data=pd.json_normalize(response)
data.head()
```

# Data Collection - Scraping

- Make request to wiki page, then using BeautifulSoup to parse HTML table data and convert to data frame.
- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

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

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```python
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables=soup.find_all("table")
html_tables
```

Python

Starting from the third table is our target table contains the actual launch records.

```python
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

Python

# Data Wrangling

- Describe how data were processed
  - Filtered BoosterVersion column with only Falcon 9 data.
  - Replaced null PayloadMass column data with mean value.
- You need to present your data wrangling process using key phrases and flowcharts
- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

# EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
  - Plot scatter chart to Visualize the relationship between below 2 columns:

    1. Flight Number and Launch Site
    2. Payload and Launch Site
    3. FlightNumber and Orbit type
    4. Payload and Orbit type.

  - Plot Bar chart to Visualize the success rate of every orbit type
  - Pot Line chart to Visualize the launch success trend for each year
- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

# EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
- Display the names of the unique launch sites in the space mission

```
%sql select distinct Launch_Site from SPACEXTBL
```

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

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where Booster_Version = 'F9 v1.1'
```

- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

# Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
  - Marked all the launch sites on Folium map
  - Create Marker Cluster to mark success or failure of launches for each launch site
  - Draw a line between nearest railway, highway and each launch site
- Explain why you added those objects
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

# Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
  - A Launch Site Drop-down Component
  - A callback function to plot pie chart
  - A Range Slider to Select Payload
  - A callback function to plot the scatter chart
- Explain why you added those plots and interactions
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

# Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
- You need present your model development process using key phrases and flowchart
    - After retrieving csv data, convert it into Pandas Dataframe format, and split them into train and test set using train_test_split from sklearn
    - Determine which is the best machine learning model that have the highest score: SVM, Classification Trees, k nearest neighbors and Logistic Regression

- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose
- GitHub URL:
- https://github.com/HsuJihHung/Jupyter-Notebooks

# Results

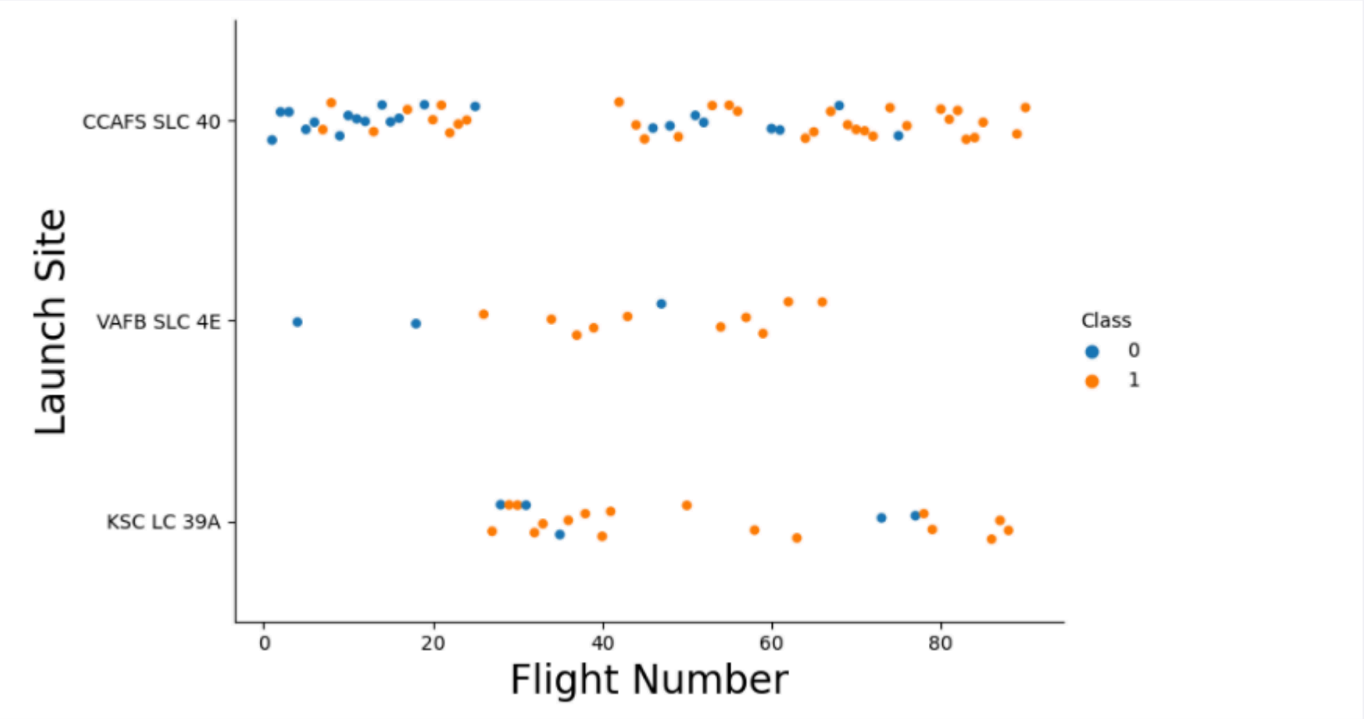
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

# Insights drawn from EDA

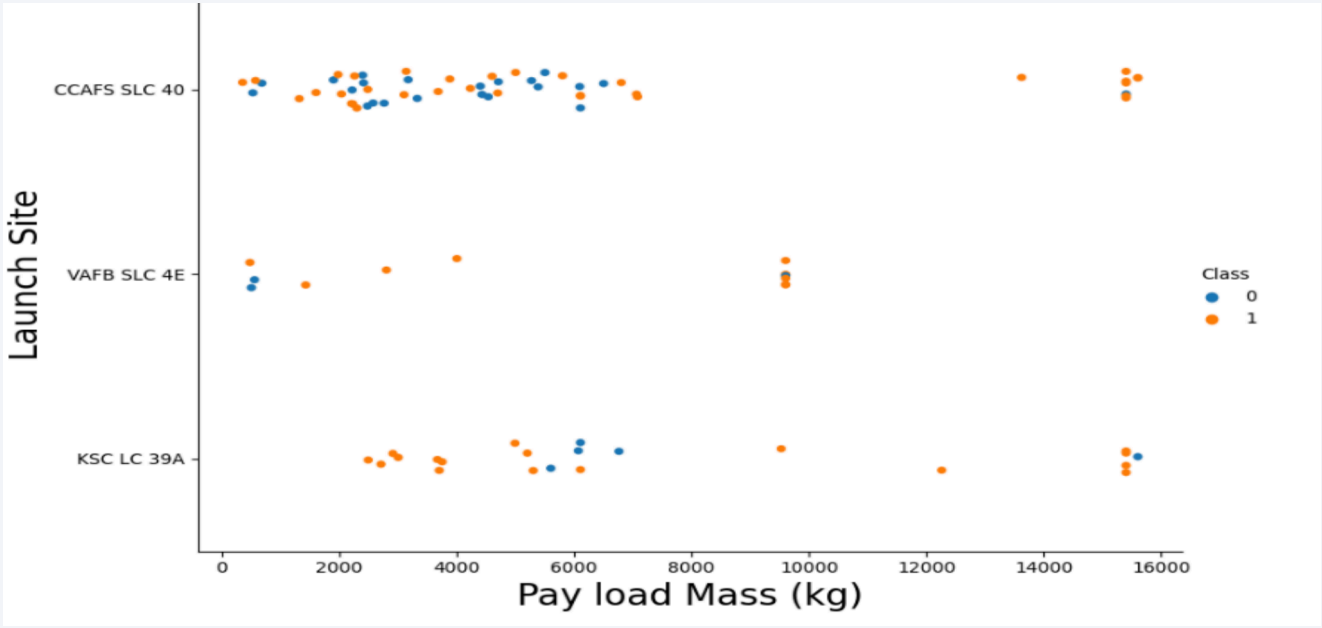
# Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site
- Show the screenshot of the scatter plot with explanations



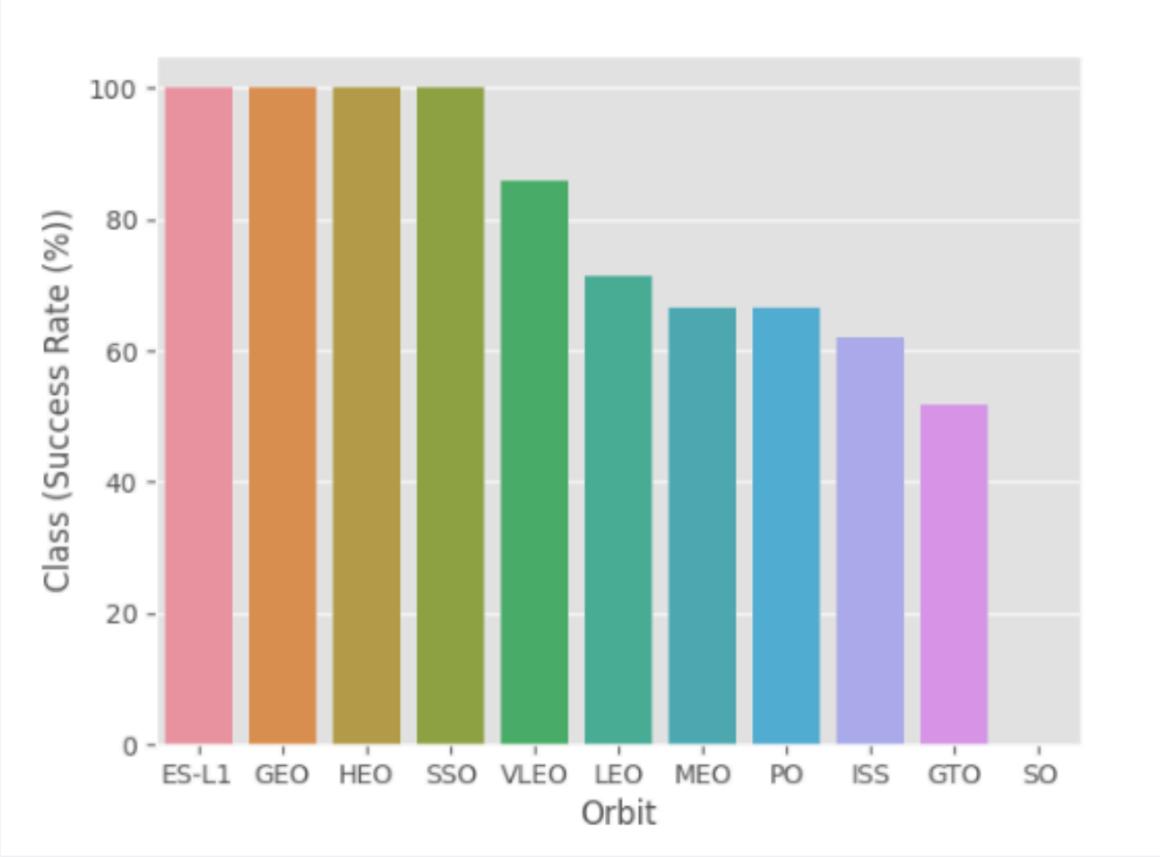
# Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site
- Show the screenshot of the scatter plot with explanations



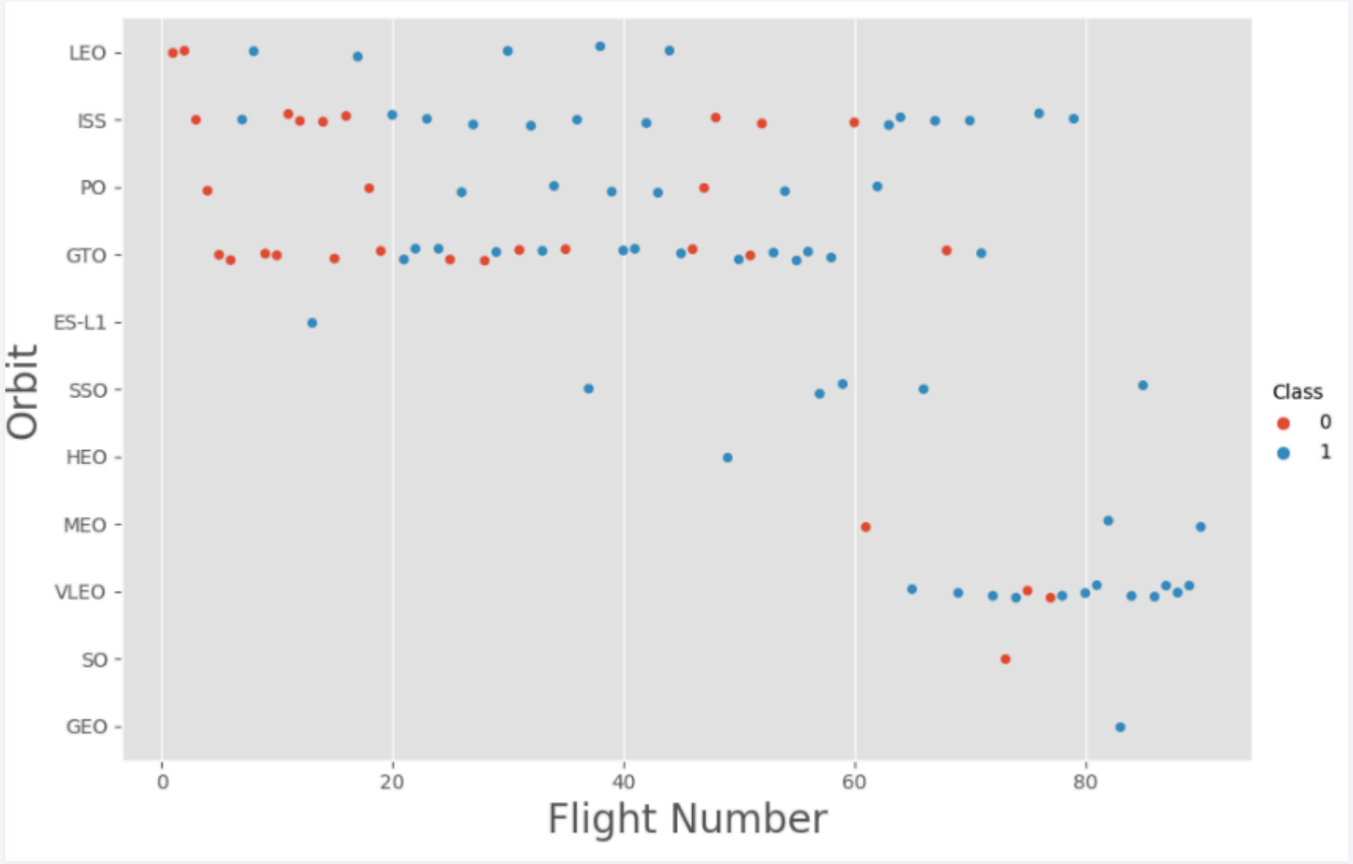
# Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type
- Show the screenshot of the scatter plot with explanations



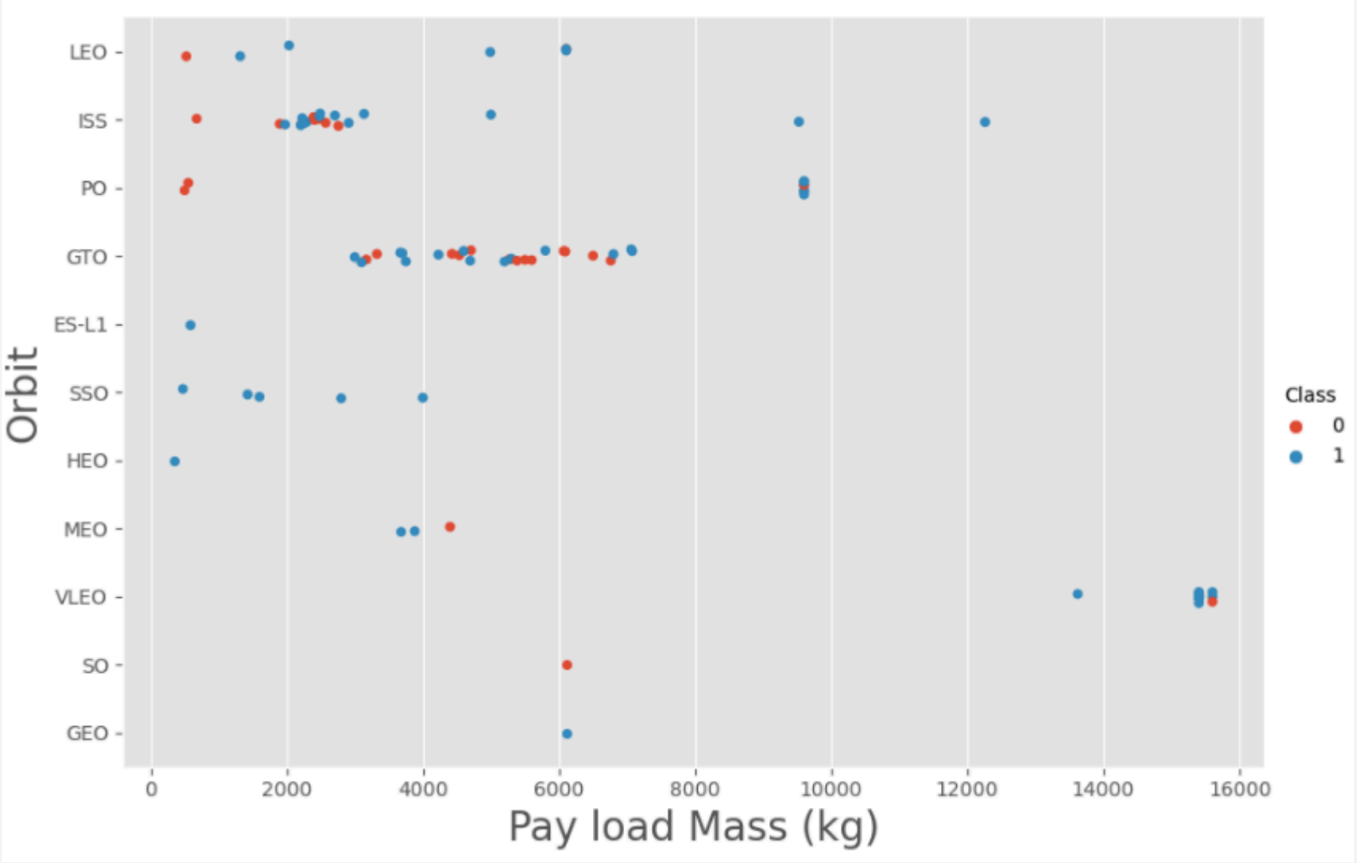
# Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type
- Show the screenshot of the scatter plot with explanations



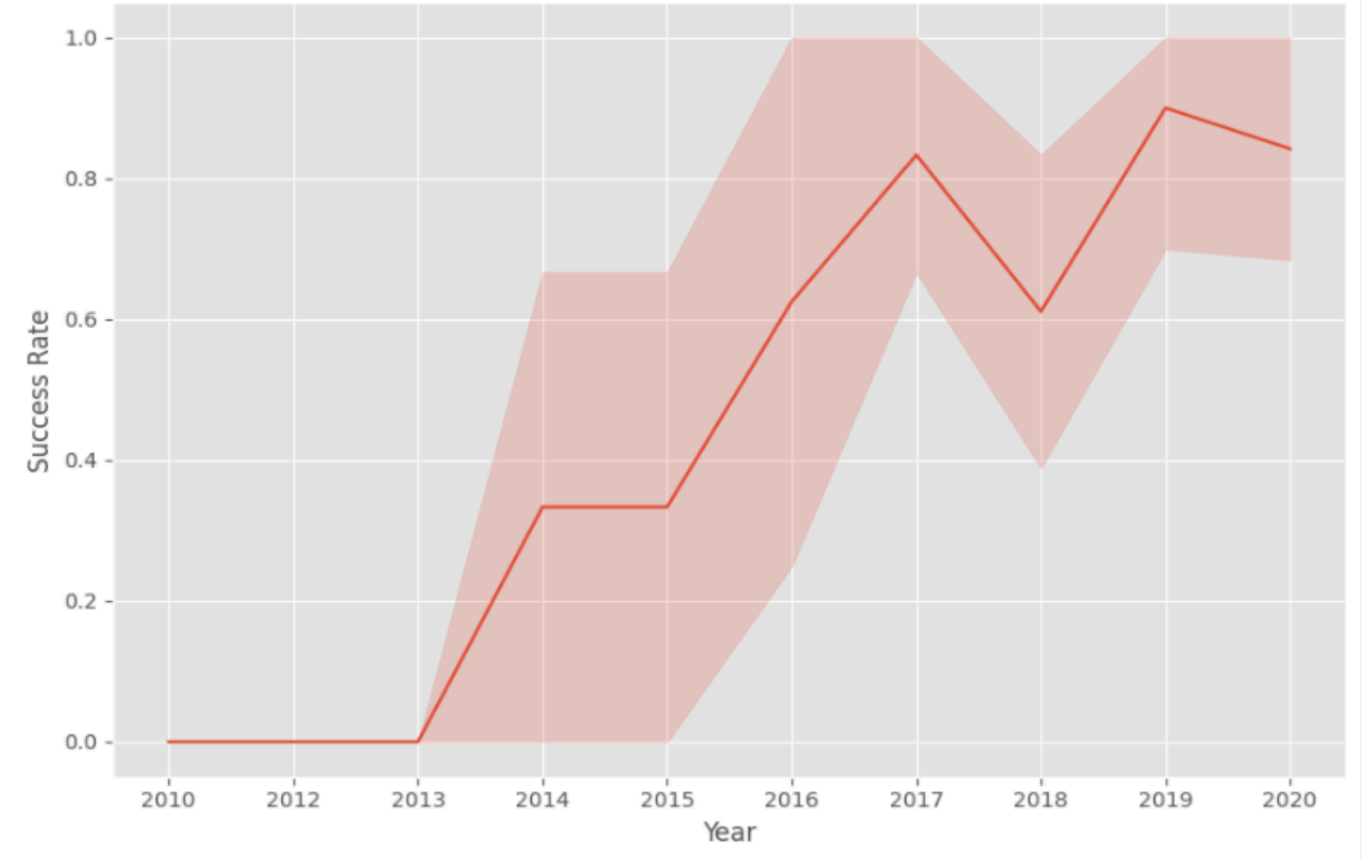
# Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type
- Show the screenshot of the scatter plot with explanations



# Launch Success Yearly Trend

- Show a line chart of yearly average success rate
- Show the screenshot of the scatter plot with explanations



# All Launch Site Names

- Find the names of the unique launch sites
- Present your query result with a short explanation here

Task 1

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

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

| Launch_Sites |
| --- |
| CCAFS LC-40 |
| VAFB SLC-4E |
| KSC LC-39A |
| CCAFS SLC-40 |

# Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Present your query result with a short explanation here

**Task 2**

Display 5 records where launch sites begin with the string '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 |
|---|---|---|---|---|---|---|---|---|---|
| 04-06-2010 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success | Failure (parachute) |
| 08-12-2010 | 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) |
| 22-05-2012 | 07:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attempt |
| 08-10-2012 | 00:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success | No attempt |
| 01-03-2013 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677 | LEO (ISS) | NASA (CRS) | Success | No attempt |

# Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here

Task 3

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

| Total Payload Mass(Kgs) | Customer |
|---|---|
| 45596 | NASA (CRS) |

# 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

Task 4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

| Payload Mass Kgs | Customer | Booster_Version |
|---|---|---|
| 2534.6666666666665 | MDA | F9 v1.1 B1003 |

# 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

**Task 5**

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

*Hint:Use min function*

```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
```

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

| MIN(DATE) |
| --- |
| 01-05-2017 |

# 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

Task 6

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

```
# %sql SELECT * FROM 'SPACEXTBL'
```

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOA
```

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

| Booster_Version | Payload |
|---|---|
| F9 FT B1022 | JCSAT-14 |
| F9 FT B1026 | JCSAT-16 |
| F9 FT B1021.2 | SES-10 |
| F9 FT B1031.2 | SES-11 / EchoStar 105 |

# 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

**Task 7**

List the total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

| Mission_Outcome | Total |
|---|---|
| Failure (in flight) | 1 |
| Success | 98 |
| Success | 1 |
| Success (payload status unclear) | 1 |

# 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

```
%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 |
| F9 B5 B1049.4 | Starlink 2 v1.0, Crew Dragon in-flight abort test | 15600 |
| F9 B5 B1051.3 | Starlink 3 v1.0, Starlink 4 v1.0 | 15600 |
| F9 B5 B1056.4 | Starlink 4 v1.0, SpaceX CRS-20 | 15600 |
| F9 B5 B1048.5 | Starlink 5 v1.0, Starlink 6 v1.0 | 15600 |
| F9 B5 B1051.4 | Starlink 6 v1.0, Crew Dragon Demo-2 | 15600 |
| F9 B5 B1049.5 | Starlink 7 v1.0, Starlink 8 v1.0 | 15600 |
| F9 B5 B1060.2 | Starlink 11 v1.0, Starlink 12 v1.0 | 15600 |
| F9 B5 B1058.3 | Starlink 12 v1.0, Starlink 13 v1.0 | 15600 |
| F9 B5 B1051.6 | Starlink 13 v1.0, Starlink 14 v1.0 | 15600 |
| F9 B5 B1060.3 | Starlink 14 v1.0, GPS III-04 | 15600 |
| F9 B5 B1049.7 | Starlink 15 v1.0, SpaceX CRS-21 | 15600 |

# 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

Task 9

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

```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome"
```

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

| substr(Date,7,4) | substr(Date, 4, 2) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Mission_Outcome | Landing _Outcome |
|---|---|---|---|---|---|---|---|
| 2015 | 01 | F9 v1.1 B1012 | CCAFS LC-40 | SpaceX CRS-5 | 2395 | Success | Failure (drone ship) |
| 2015 | 04 | F9 v1.1 B1015 | CCAFS LC-40 | SpaceX CRS-6 | 1898 | Success | 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

**Task 10**

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

```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;
```

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

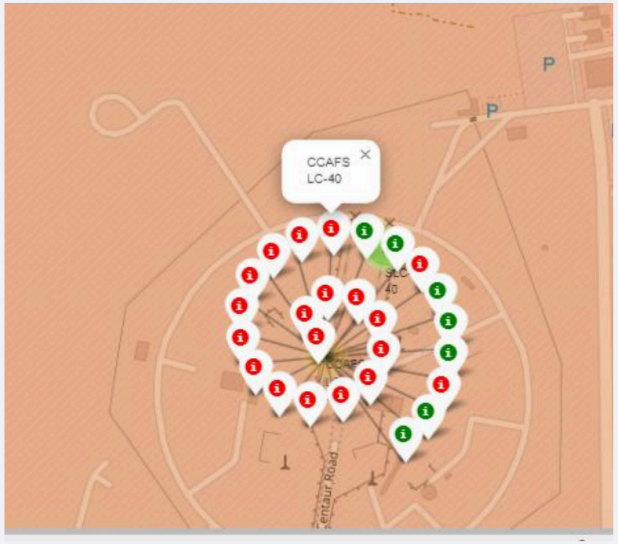
| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing _Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 19-02-2017 | 14:39:00 | F9 FT B1031.1 | KSC LC-39A | SpaceX CRS-10 | 2490 | LEO (ISS) | NASA (CRS) | Success | Success (ground pad) |
| 18-10-2020 | 12:25:57 | F9 B5 B1051.6 | KSC LC-39A | Starlink 13 v1.0, Starlink 14 v1.0 | 15600 | LEO | SpaceX | Success | Success |
| 18-08-2020 | 14:31:00 | F9 B5 B1049.6 | CCAFS SLC-40 | Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 1B | 15440 | LEO | SpaceX, Planet Labs, PlanetIQ | Success | Success |
| 18-07-2016 | 04:45:00 | F9 FT B1025.1 | CCAFS LC-40 | SpaceX CRS-9 | 2257 | LEO (ISS) | NASA (CRS) | Success | Success (ground pad) |
| 18-04-2018 | 22:51:00 | F9 B4 B1045.1 | CCAFS SLC-40 | Transiting Exoplanet Survey Satellite (TESS) | 362 | HEO | NASA (LSP) | Success | Success (drone ship) |

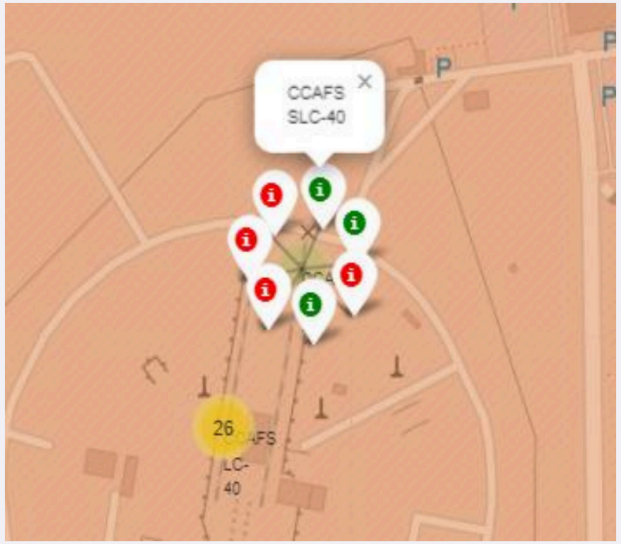
Section 3

# Launch Sites Proximities Analysis

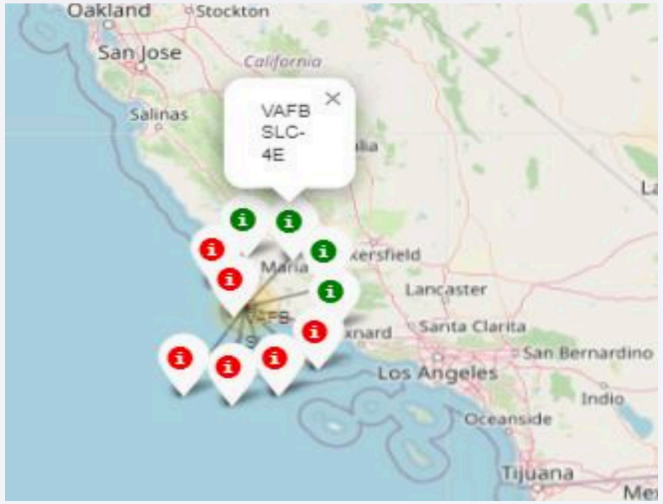
# All SpaceX Falcon 9 launch sites' location

# Launch outcomes for each launch site

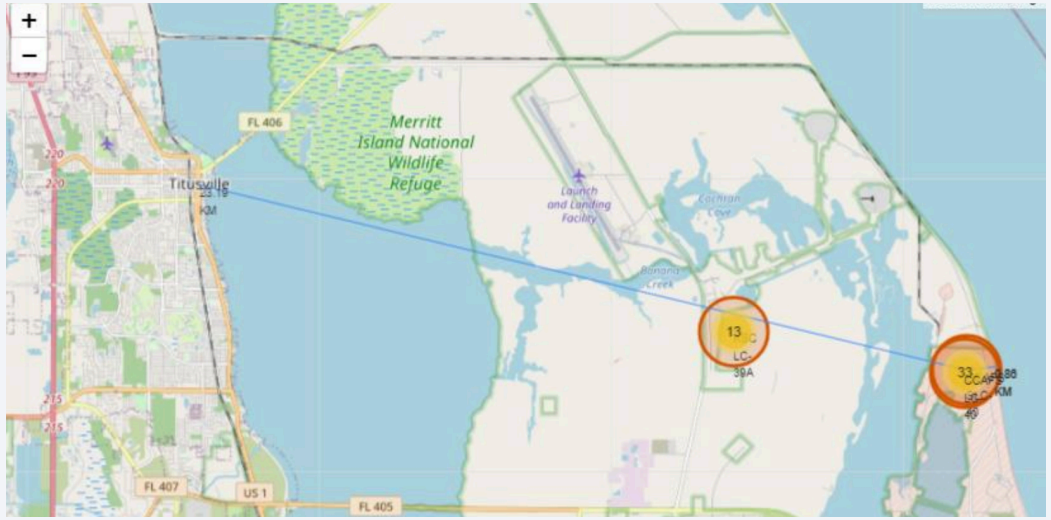


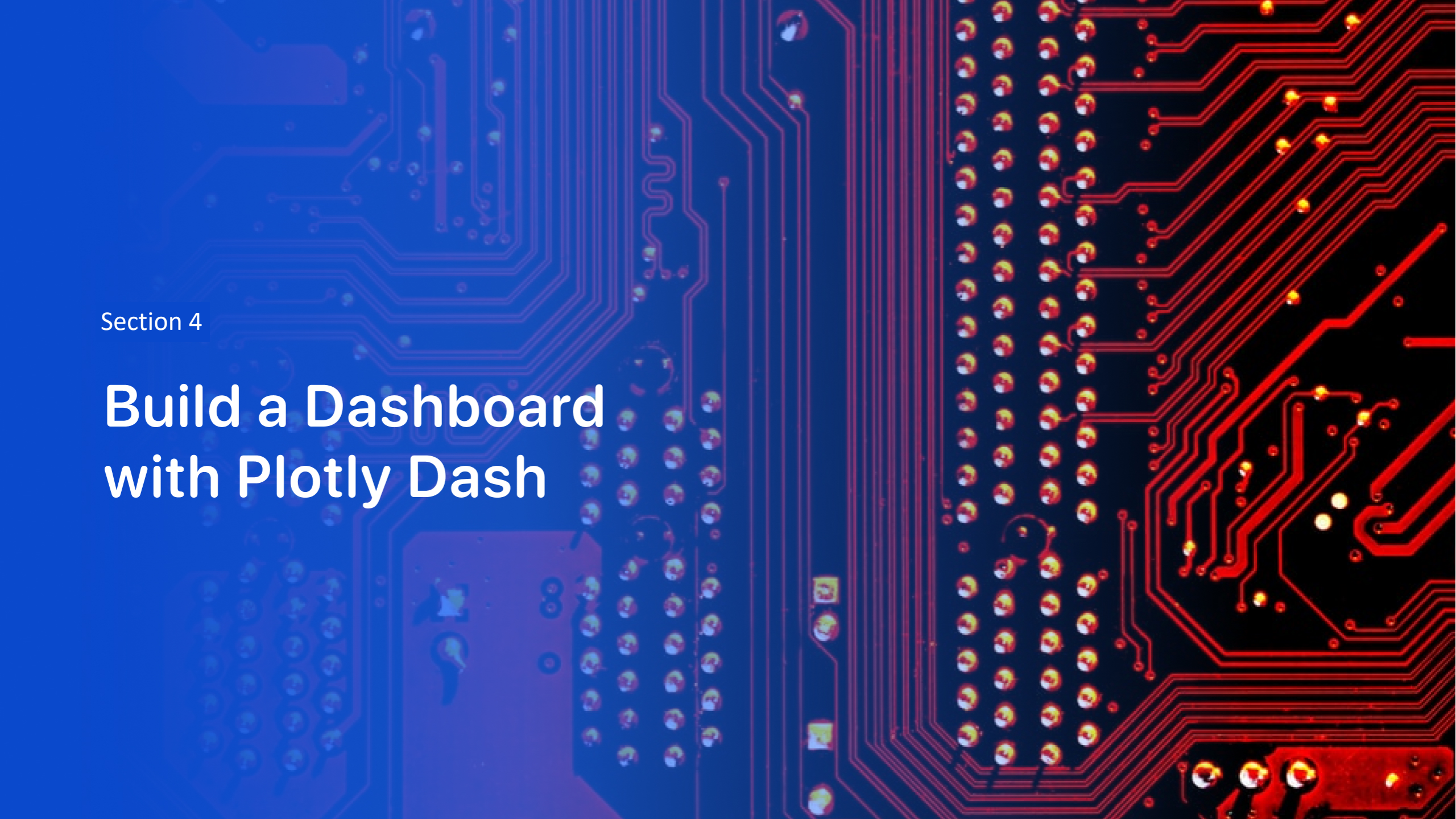





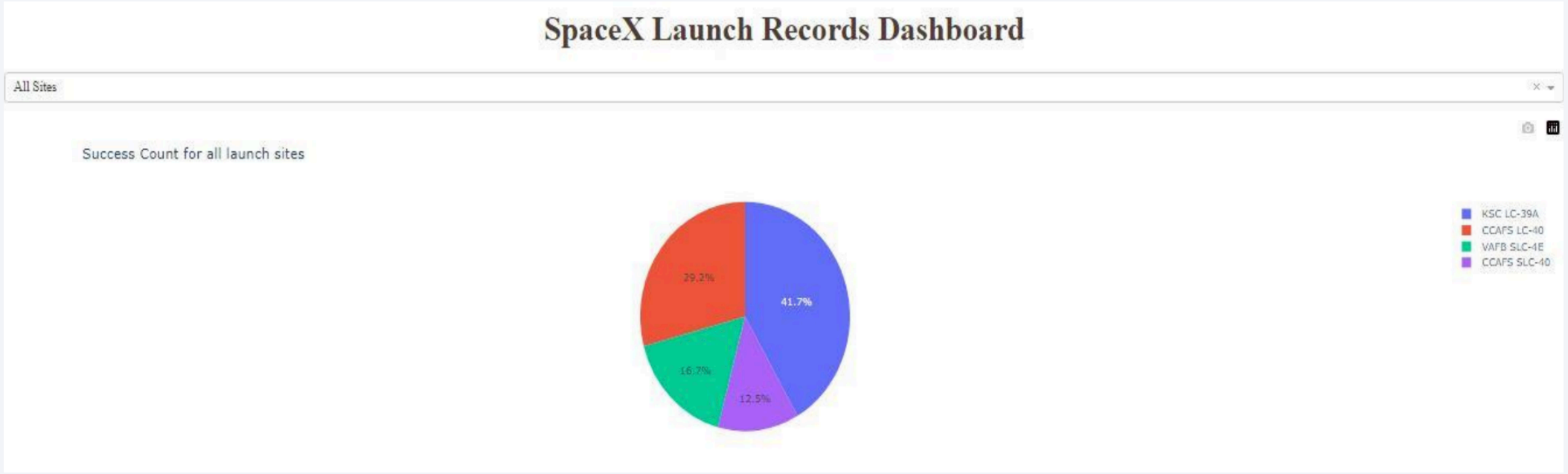


# Distances between a launch site to its proximities

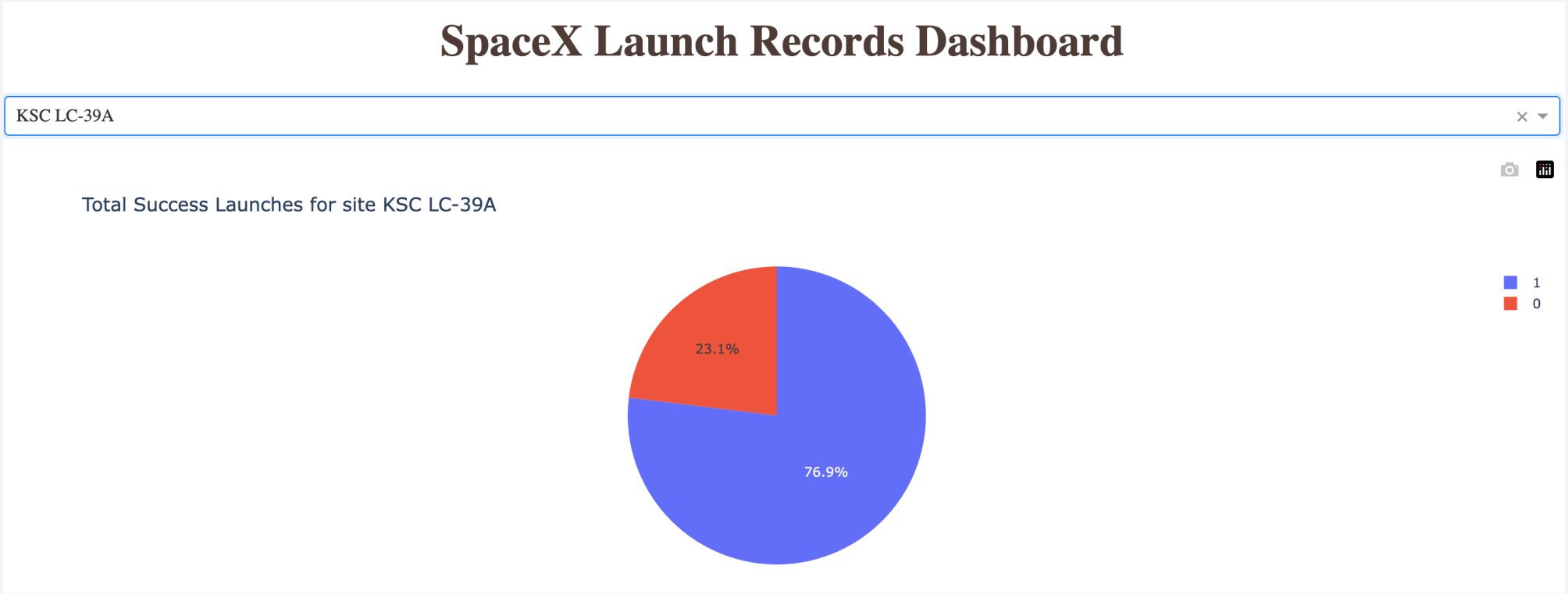




Section 4
Build a Dashboard with Plotly Dash

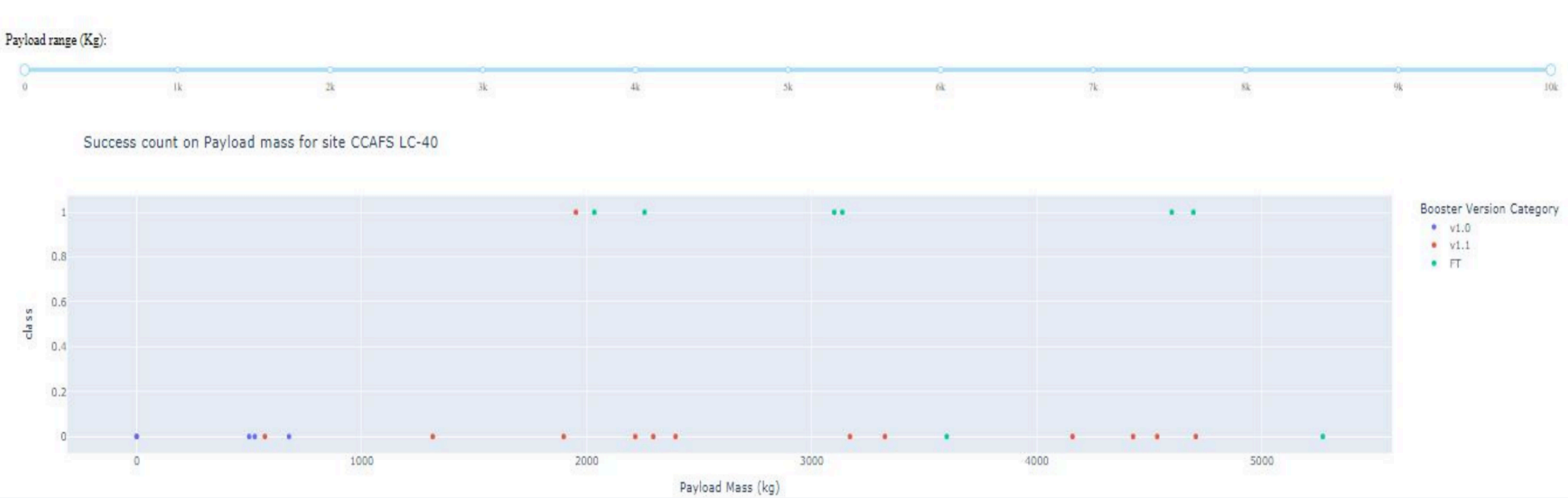
# Launch success count for all sites



# Launch site with highest launch success ratio



# Payload vs. Launch Outcome scatter plot for all sites



Section 5

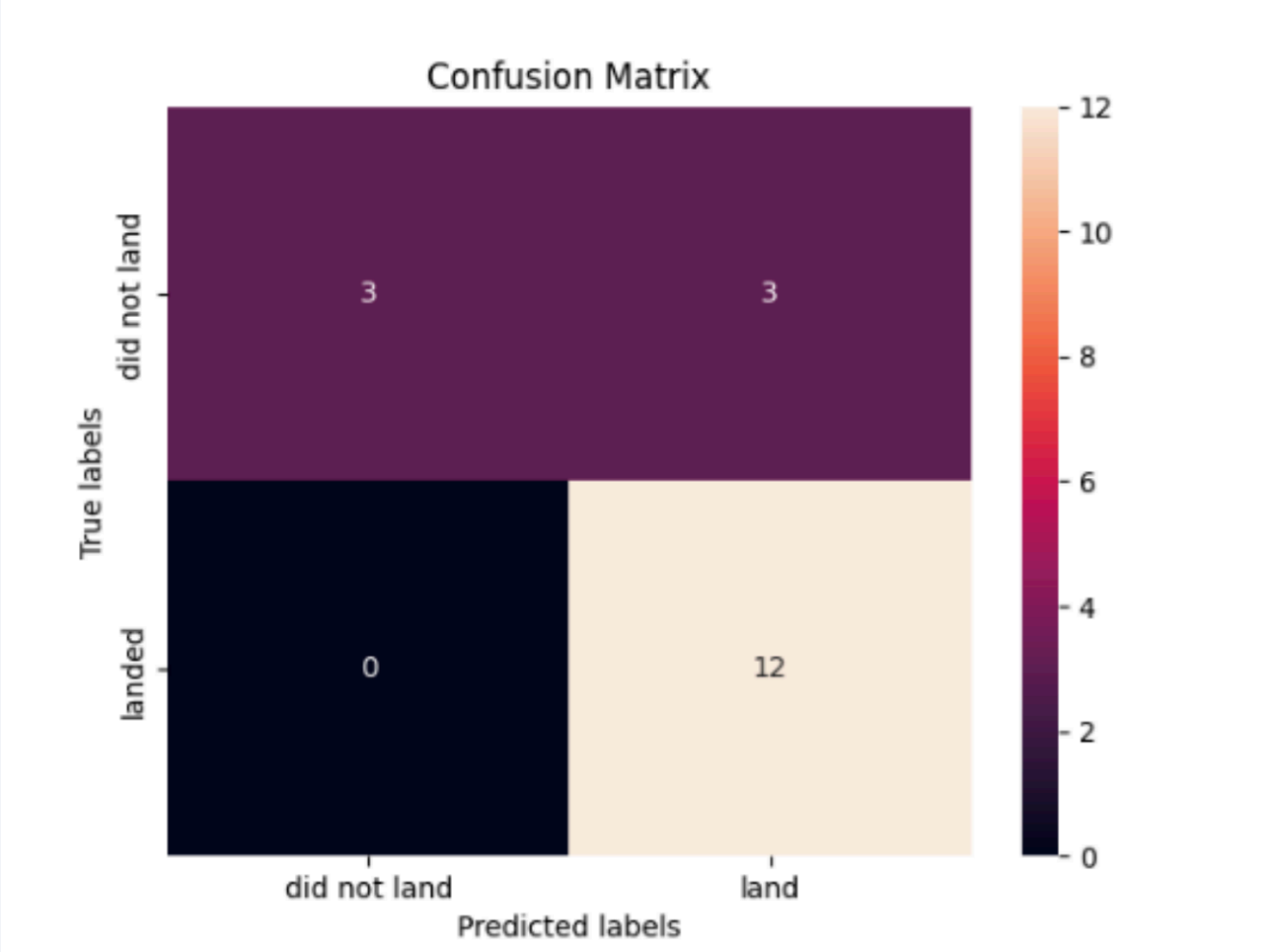
# Predictive Analysis (Classification)

# Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart
- Find which model has the highest classification accuracy

| | 0 |
|---|---|
| Method | Test Data Accuracy |
| Logistic_Reg | 0.833333 |
| SVM | 0.833333 |
| Decision Tree | 0.833333 |
| KNN | 0.833333 |

# Confusion Matrix



# Conclusions

- Launch site VAFB SLC 4E has a success rate of proximately around 77%.
- As the flight number increases, the success rate also increases.
- When observing Payload Vs. Launch Site scatter point chart, we can find that launch site # VAFB-SLC have no rockets launched for heavy payload mass(payload greater than 10000).
- Orbits type ES-L1, GEO, HEO & SSO have the highest success rates at 100%, while type SO having the lowest success rate at 50%. Orbit type SO has 0% success rate.

# 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!