Winning Space Race with Data Science

Shubham Rathore
05/09/2025

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection via API and Web Scraping
  - Data Wrangling
  - Exploratory data analysis with Data Visualizations
  - Exploratory data analysis with SQL in local sqlite database
  - Showing an intractive map with Folium
  - Building a dashboard with Plotly Dash
  - Predictive Analysis (Classification)
- Summary of all results
  - Exploratory data analysis (EDA) results
  - Interactive dashboard
  - Predictive analysis of Classification models

Introduction

- **Project Overview and Context**

SpaceX has emerged as a trailblazer in the commercial space industry, revolutionizing space travel by making it significantly more affordable. 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. Predicting whether the first stage will successfully land plays a crucial role in determining the overall launch cost. Leveraging publicly available data and machine learning models, this project aims to predict the likelihood of SpaceX reusing the first stage.

- **Key Question to Explore**

This project will ultimately predict if the SpaceX Falcon 9 stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

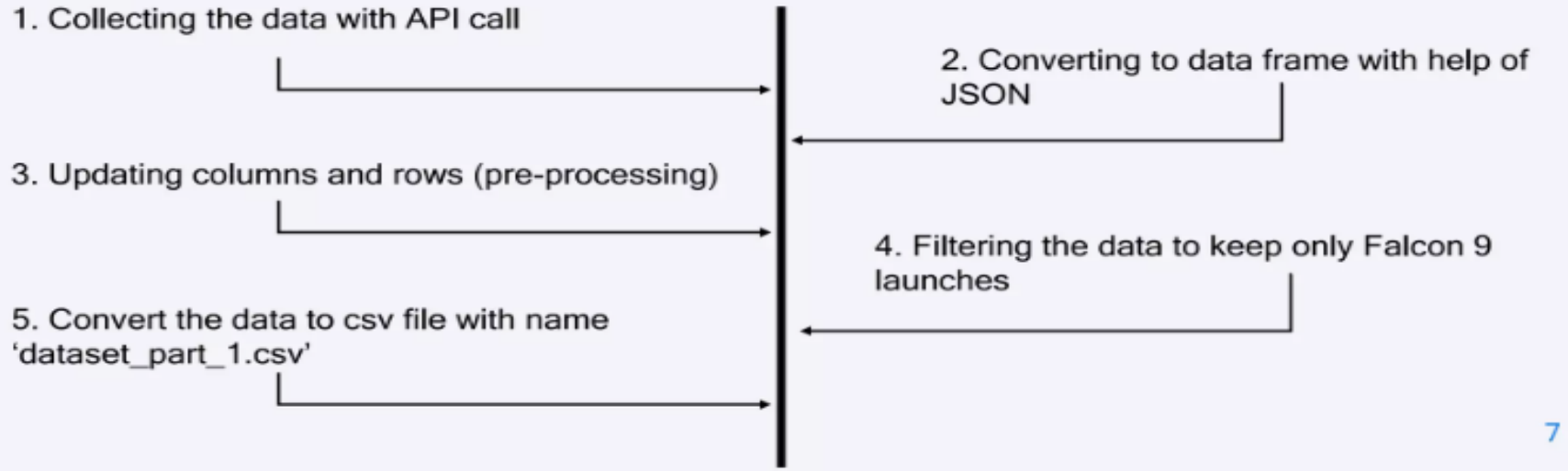
- Data collection methodology:
  - Making GET requests to the SpaceX REST API
  - Web Scraping from Wikipedia
- Perform data wrangling
  - Handling Missing Values
  - Features Transformation
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Constructing diverse classification models tailored to binary outcomes
  - Fine-tuning hyperparameters through Grid-search and Cross-validation
  - Models used are Logistic regression, Support vector machine(SVM), Decision tree, K-nearest neighbors(KNN)

Data Collection

- Data sets were collected using the API call and from Website

1. SpaceX REST API:

   Leveraged to perform programmatic API requests, obtaining structured and real-time data using https://api.spacexdata.com/v4



2. Wikipedia Web Scraping:

   Applied advanced web scraping techniques to extract tabular data from SpaceX's Wikipedia page, supplementing the dataset with historical context

Data Collection – SpaceX API

GitHub repo: https://github.com/Shubham-Rathore08/spacex_falcon9_first_stage_landing_prediction/blob/main/notebooks/jupyter-labs-spacex-data-collection-api.ipynb

1. Collecting the data with API call:

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

response = requests.get(spacex_url)
```

2. Converting to data frame with help of JSON:

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

3. Updating columns and rows (pre-processing)

```
# Lets take a subset of our dataframe keeping only the features we want and the flight
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extr
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 i
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
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)]
```

4. Filtering the data to keep only Falcon9 launches

```
data_falcon9 = df[df['BoosterVersion'] != 'Falcon 1']

data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9.reset_index(drop=True, inplace =True)
```

5. Convert the data to csv file

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

GitHub repo: https://github.com/Shubham-Rathore08/spacex_falcon9_first_stage_landing_prediction/blob/main/notebooks/jupyter-labs-webscraping.ipynb

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

1. Request the HTML page:

```
response= requests.get(static_url, headers=headers)
print(response.status_code)
```

2. Assign a beautiful soup object:

```
soup = BeautifulSoup(response.content, "html.parser")
```

3. Getting column names:

```
column_names = []

for th in first_launch_table.find_all("th"):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:    # only keep non-empty
        column_names.append(name)

print("Extracted column names:")
print(column_names)
```

4. Creating the launch_dict:

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []

# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

5. Convert to dataframe:

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

Data Wrangling

GitHub repo: https://github.com/Shubham-Rathore08/spacex_falcon9_first_stage_landing_prediction/blob/main/notebooks/jupyter-labs-spacex-data-wrangling.ipynb

Context:

- ❑ The landing outcome shown in the Outcome column:

1. **Ocean Landing:**
   - **True Ocean** - Successful landing in a designated ocean region
   - **False Ocean** - Unsuccessful attempt to land in the ocean
2. **RTLS(Return to launch site) :**
   - **True RTLS** - Successful landing on a ground pad
   - **False RTLS** - Unsuccessful attempt to land on a ground pad
3. **ASDS(Autonomous Spaceport Drone Ship)**
   - **True ASDS** - Successful landing on a drone ship
   - **False ASDS** - Unsuccessful attempt to land on a drone ship

- ❑ Defining a set of unsuccessful (bad) outcomes, bad_outcomes

- ❑ The binary labels for Class features represent:
  - **1** - Successful landing
  - **0** - Unsuccessful landing

1. Create landing_outcome variable :

```
landing_outcomes = data_falcon9['Outcome'].value_counts()
```

```
Outcome
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: count, dtype: int64
```

2. Define bad_outcomes:

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

3. Create a Class feature:

```
landing_class =  data_falcon9["Outcome"].apply(lambda x: 0 if x in bad_outcomes else 1)
```

```
data_falcon9['Class']=landing_class
data_falcon9[['Class']].head(10)
```

EDA with Data Visualization

❑ Various charts were created to explore relationships and trend in the data:

1. **Scatter Plots**: Show relationship between variables that may inform machine learning models.
   - Flight Number vs. Payload Mass
   - Flight Number vs. Launch Site
   - Payload Mass vs. Launch Site
   - Flight Number vs. Orbit Type
   - Payload Mass vs. Orbit Type

2. **Bar Charts**: Highlights comparison between specific categories and their measured values.
   - Success Rate vs. Orbit Type

3. **Line Charts**: Generally used to show change of a variable over time.
   - Success Rate vs. Yearly Trend

❑ This visualization strategy provides insights into patterns and relationships within the dataset.

GitHub repo: https://github.com/Shubham-Rathore08/spacex_falcon9_first_stage_landing_prediction/blob/main/notebooks/EDA-data-visualization.ipynb

EDA with SQL

GitHub repo: https://github.com/Shubham-Rathore08/spacex_falcon9_first_stage_landing_prediction/blob/main/notebooks/jupyter-labs-eda-sql.ipynb

To gather some information about the dataset, some SQL queries were performed that were:

1. Display the names of the unique launch sites in the space mission.
2. Display 5 records where launch sites begin with the string 'CCA'
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display average payload mass carried by booster version F9 v1.1
5. List the date when the first successful landing outcome in ground pad was acheived.
6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. List the total number of successful and failure mission outcomes
8. List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.
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.
10. 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.

Interactive Map with Folium

GitHub repo: https://github.com/Shubham-Rathore08/spacex_falcon9_first_stage_landing_prediction/blob/main/notebooks/folium_viz.ipynb

The following steps were taken to visualize the launch data on an interactive map:

1. **Launch Site Markers:**
   - **NASA Johnson Space Center**: A marker with a circle, popup, and text label was added, using its latitude and longitude as the starting locations.
   - **All Launches Sites**: Markers were added for each site to display their geographical locations, proximity to the equator, and coastal areas.

2. **Mark the success/failed launches for each site on map:**
   - **Color-Coded Markers:**
     - Green for successful launches
     - Red for failed launches
   - **Marker Clustering:** Used to highlight sites with higher success rates visually.

3. **Calculate the distances between a launch site to its proximities:**
   - After making a point using the Lat and Long values, create a folium.Marker object to show the distance.
   - To display the distane line between two points, draw a folium.Polyline and add this to map.

Build a Dashboard with Plotly Dash

- The following plots were added to a plotly Dash Dashboard to have an interactive visualization of the data:

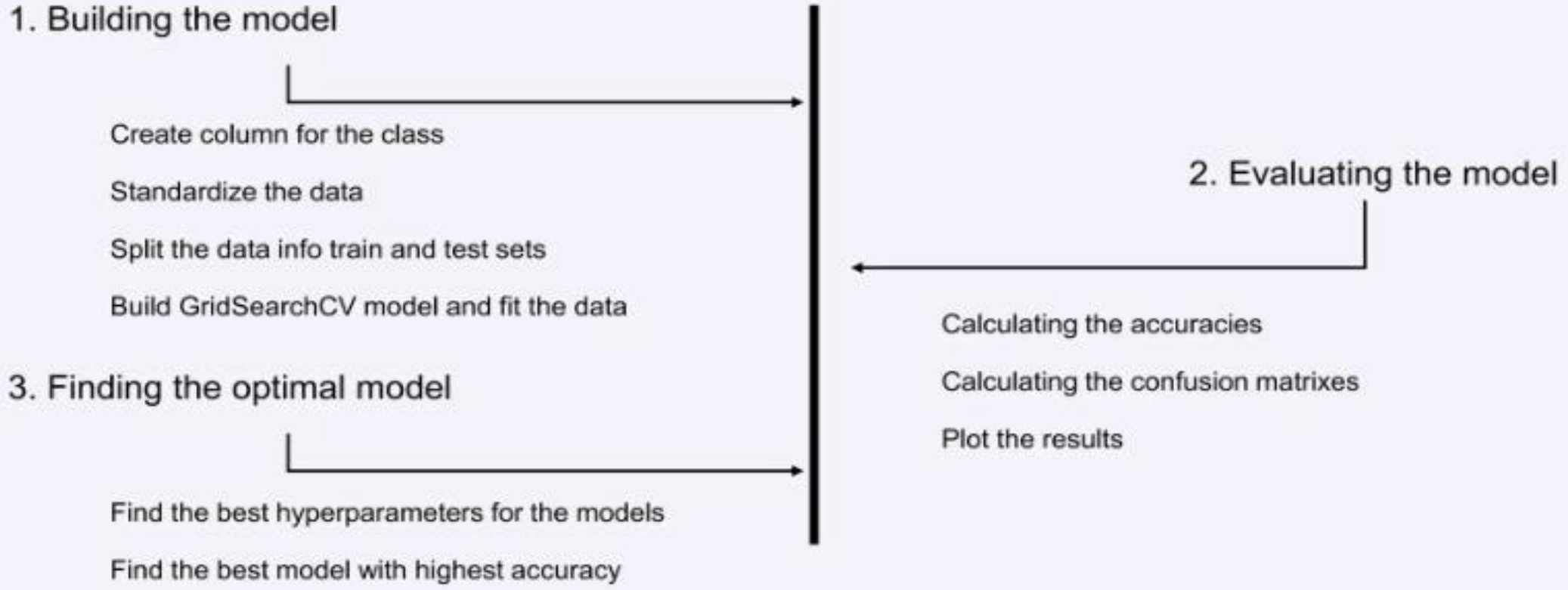
1. **Pie chart:**
   - **Overall View:** Displays total successful launches across all sites.
   - **Site-Specific View:** Shows the ratio of success vs. failed launches for a selected launch site.
   - **Dropdown Menu:** Enables users to select a specific launch site or view data for all sites

2. **Scatter Chart:**
   - **Overall View:** Shows the correlation between outcome (success or not) and payload mass.
   - **Booster Versions**: Highlights variations in performance across different booster version
   - **Range Slider**: Allows users to filter data based on a specified payload mass range

- This interactive dashboard provides dynamic insights, enabling tailored exploration of SpaceX launch data.

GitHub repo: https://github.com/Shubham-Rathore08/IBM_SpaceX_capstone_project/blob/main/Interactive_Visual_Analytics_and_Dashboards/spacex_dash_app_code.ipynb

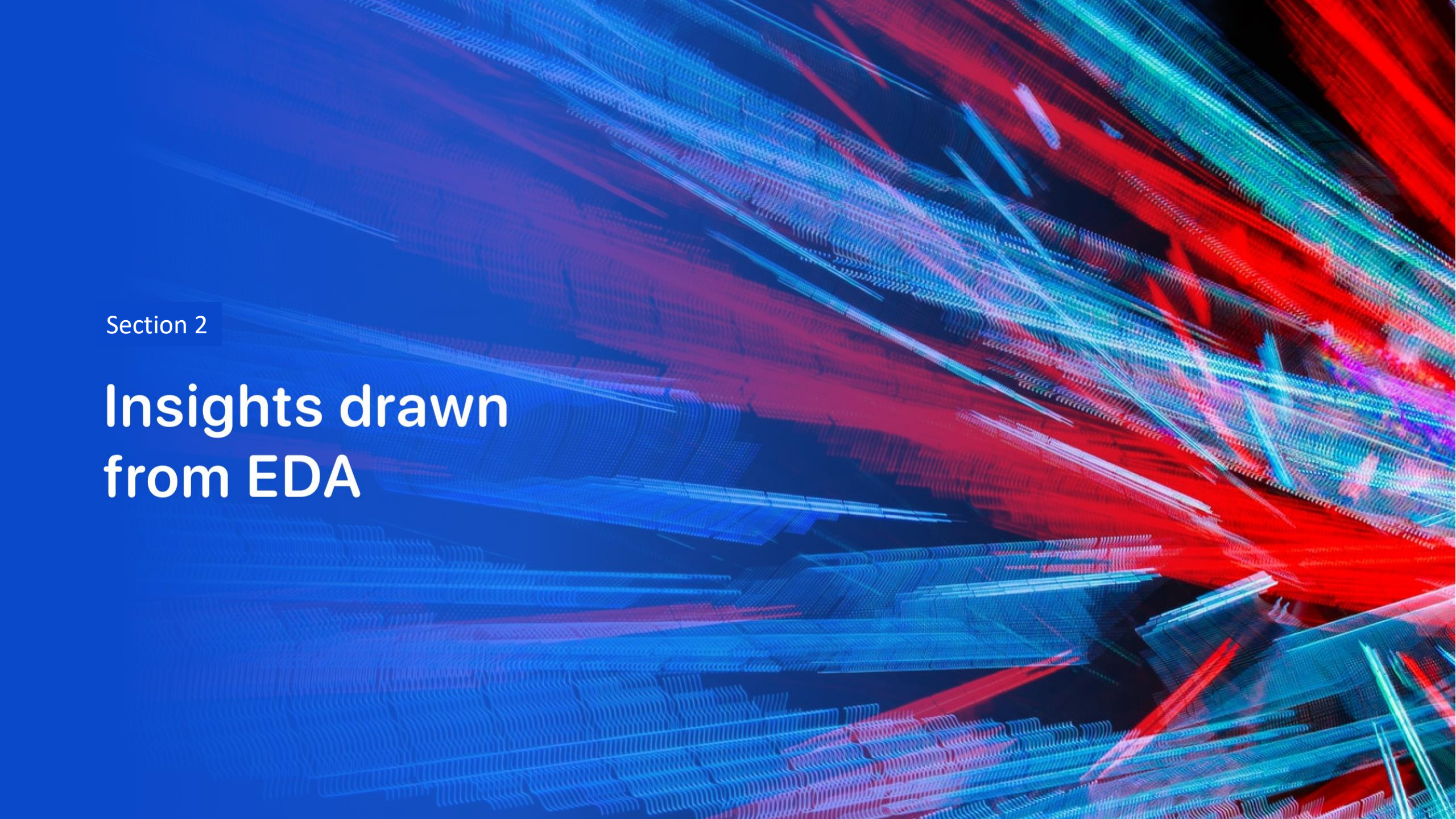
Predictive Analysis (Classification)



GitHub repo: https://github.com/Shubham-Rathore08/spacex_falcon9_first_stage_landing_prediction/blob/main/notebooks/spacex_ML_prediction.ipynb

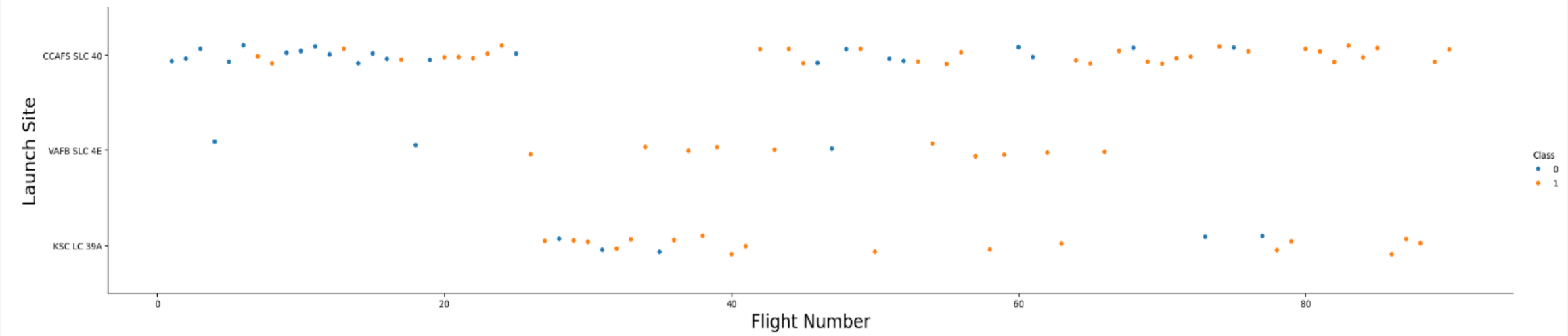
Results

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


Section 2
Insights drawn from EDA

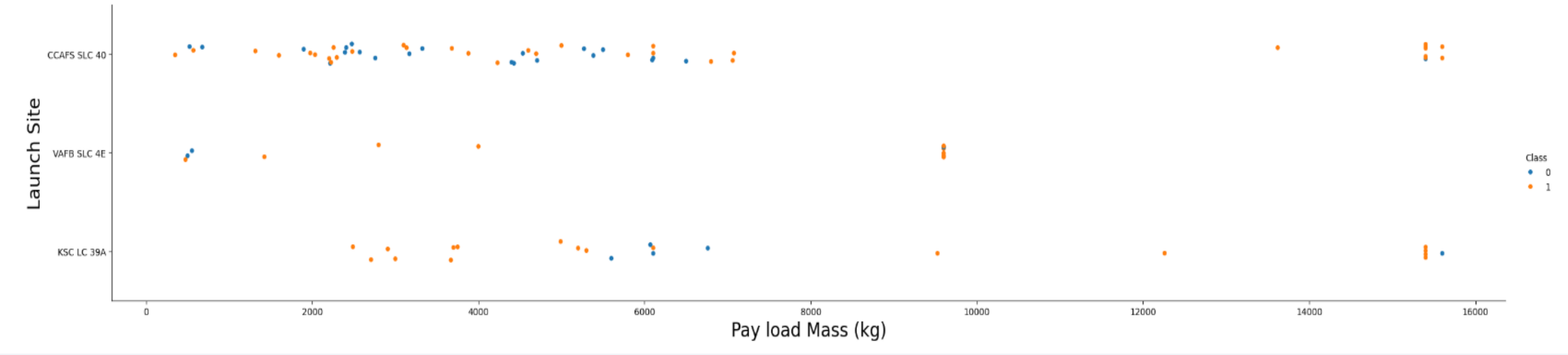
Flight Number vs. Launch Site

- This scatter plot of Flight Number vs. Launch Site shows that:
  - As the number of flights increases, the rate of success at a launch site increases.
  - **Launch Site Success Rates:**
    - **CCAFS LC-40**: Most of the early flights (flight numbers < 30) were generally unsuccessful.
    - **VAFB SLC 4E**: This trend shows that earlier flights were less unsuccessful.
    - **KSC LC-39A** : No early flights were launched, so the launches from this site are more successful.



Payload vs. Launch Site

- This scatter plot of Payload vs. Launch Site shows that:
  - Above a payload mass around 7000 kg there are very few unsuccessful landings, but there is also far less data for these heavier launches
  - All sites launched a variety of payload masses, with most of the launches from CCAFS SLC 40 being comparatively lighter payloads (with some outliers)
  - There is no clear correlation between payload mass and success rate



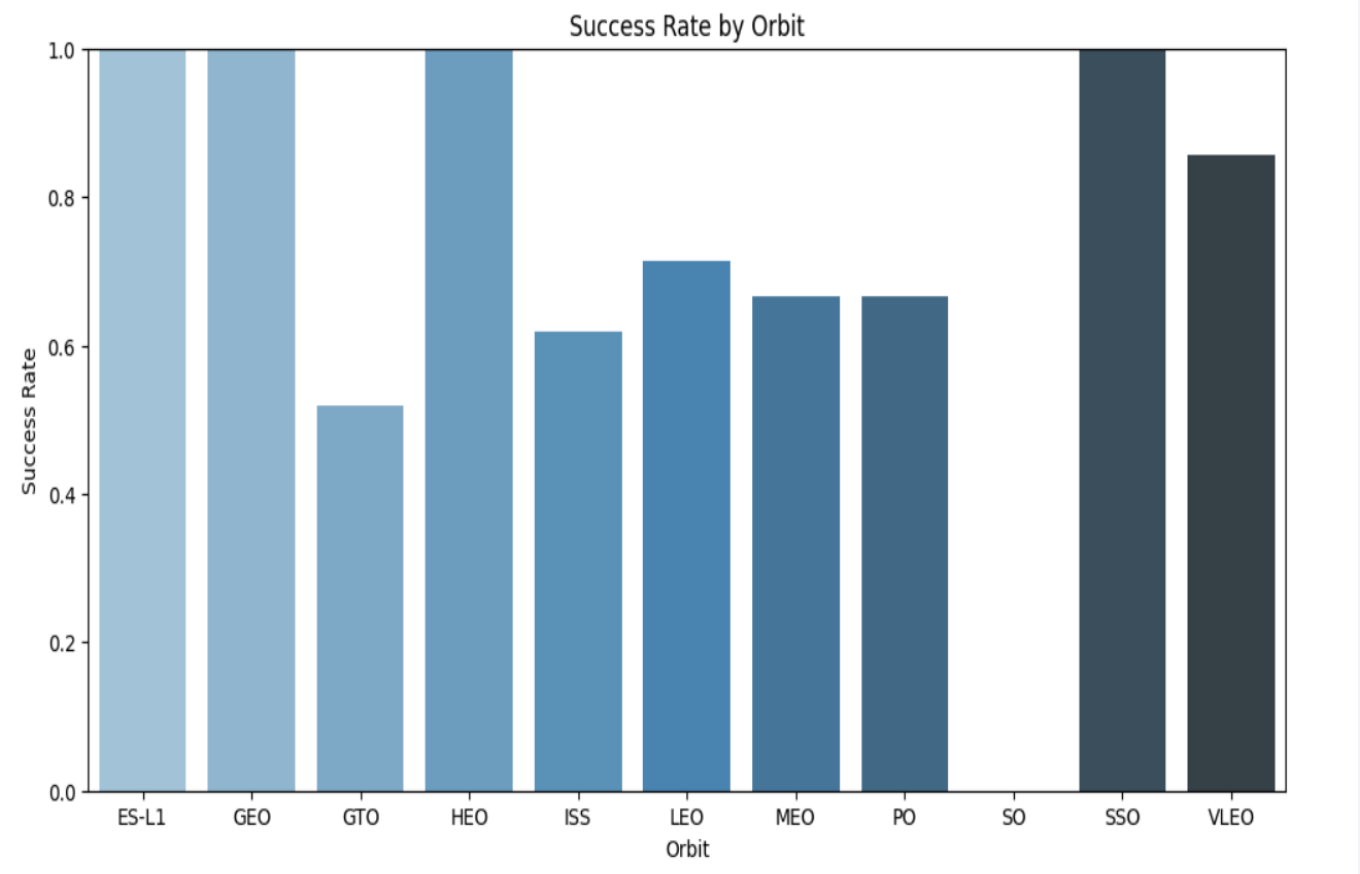
Success Rate vs. Orbit Type

- This Bar plot shows the comparison between success rates at different orbits:

1. **Orbits with 100% Success Rate:**
   - ES-L1 (Earth–Sun Lagrange Point 1)
   - GEO (Geostationary Earth Orbit)
   - HEO (Highly Elliptical Orbit)
   - SSO (Sun-Synchronous Orbit)

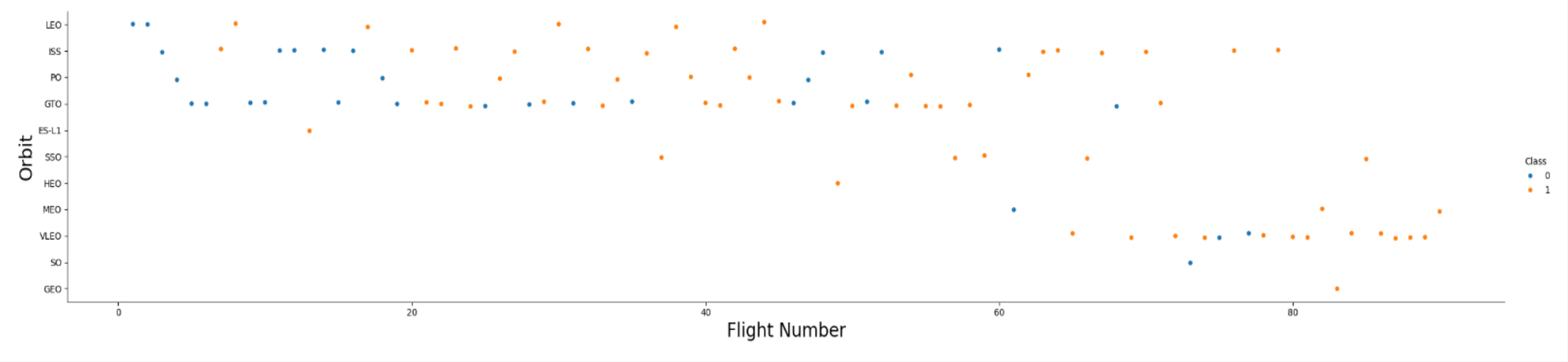
2. **Orbits with 0% Success Rate:**
   - SO (Solar Orbit)

3. **Intermediate Success Rates (50%-85%):**
   - GTO (Geostationary Transfer Orbit)
   - ISS (International Space Station Orbit)
   - LEO (Low Earth Orbit)
   - MEO (Medium Earth Orbit)
   - PO (Polar Orbit)



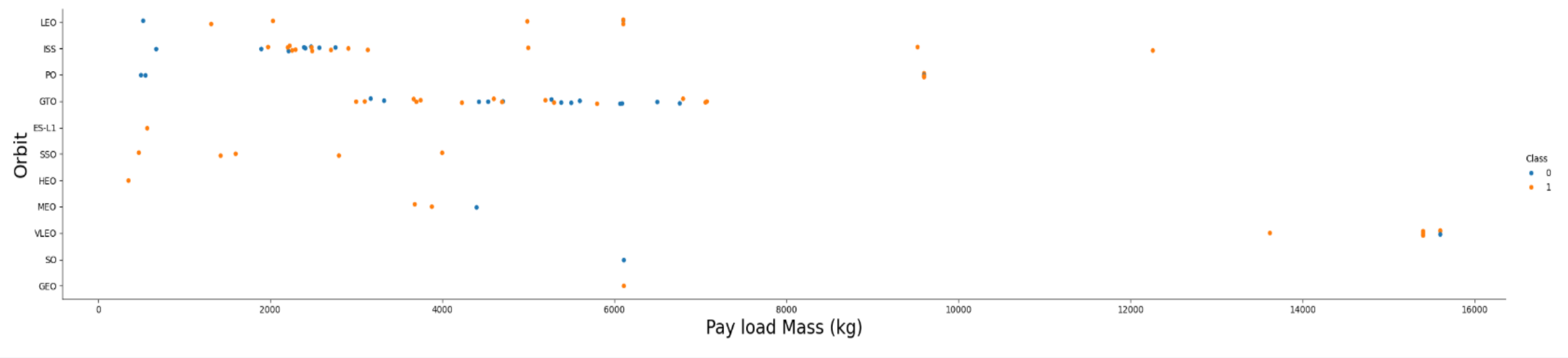
Flight Number vs. Orbit Type

- This scatter plot tells a few key points about the previous plot:
  - The 100% success rate of **GEO, HEO, and ES_L1** orbits can be explained by only having 1 flight into their respective orbits.
  - The 100% success rate of **SSO** orbit is more impressive with 5 successful flights.
  - There is no evident relationship flight number and success rate for **GTO** orbit.
  - For **LEO** orbit, A clear positive correlation is observed as Flight Number increases, the success rate also increases.



Payload vs. Orbit Type

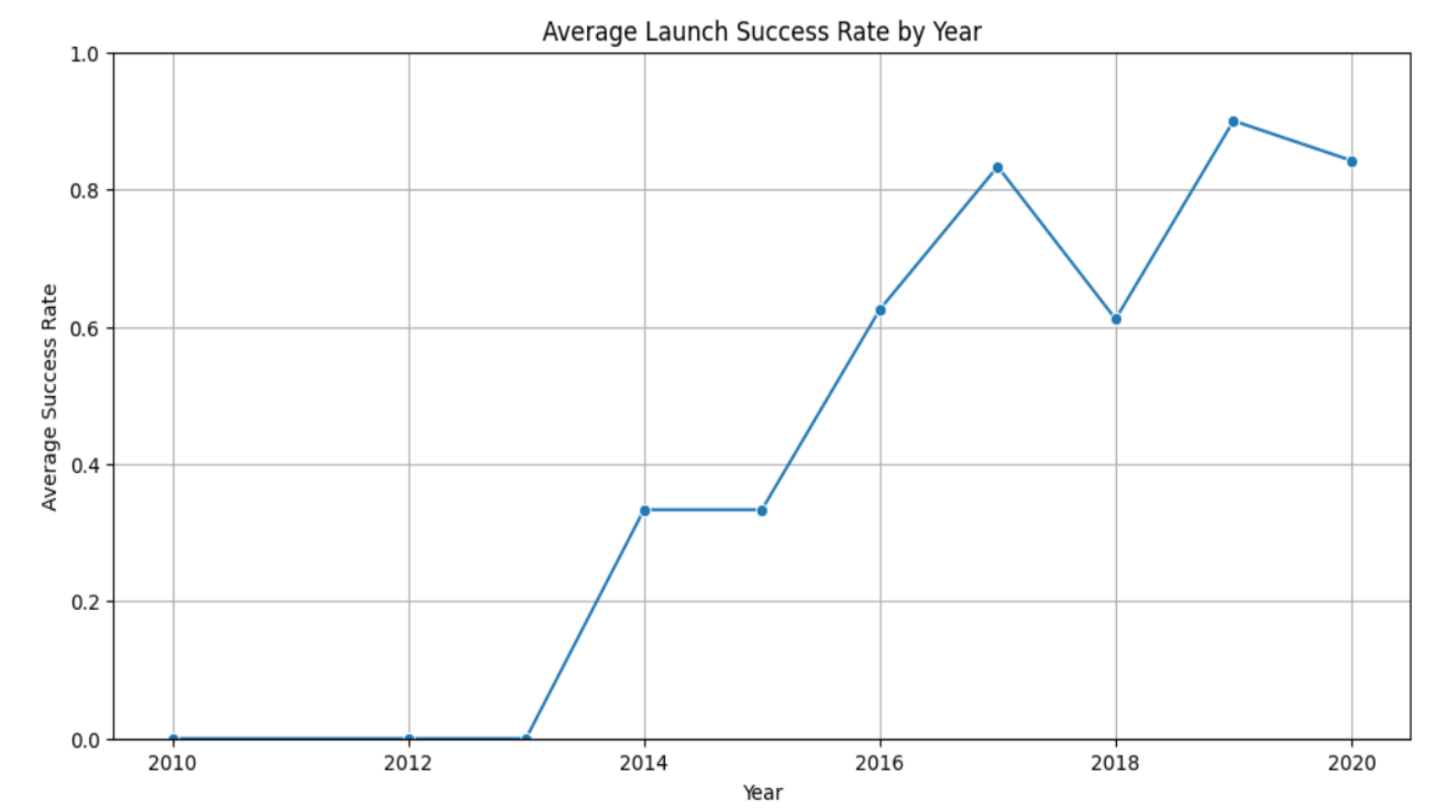
- This scatter plot shows how payload mass impacts success rates across different orbit types:
  - The following orbit types have more success with heavy payloads:
    - PO
    - ISS
    - LEO
  - For GTO, the relationship between payload mass and success rate is unclear.



Launch Success Yearly Trend

❑ The line chart reveals a clear trend in **success rates** over time:

- Between 2010 and 2013, all landings were unsuccessful (as success rate is 0)
- After 2013, the success rate generally increased.
- After 2016, there was always a greater than 60% chance of success.



All Launch Site Names

The query identifies all distinct launch sites used in the SpaceX missions.

- **Command Highlights:**

The **DISTINCT** keyword retrieves unique entries from the **launch_site** column in the dataset.

```
%%sql

SELECT DISTINCT Launch_Site
FROM SPACEXTABLE;
```

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

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Launch Site Names Begin with 'CCA'

The query retrieves the first five records where launch sites begin with "CCA.“

- **Command Highlights:**
  - LIKE 'CCA%’ : Filters launch sites starting with "CCA."
  - LIMIT 5 : Restricts the output to five entries.

```
%%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

This query calculates the **total payload mass** launched by SpaceX for **NASA (CRS)** missions:

- **Command Highlights:**
  - SUM(payload_mass__kg_): Computes the sum of payload masses
  - WHERE customer = 'NASA (CRS)': Filters launches associated with NASA's Commercial Resupply Services.

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

```
%%sql

SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass
FROM SPACEXTABLE
WHERE Customer = 'NASA (CRS)';
```

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

Total_Payload_Mass
45596

Average Payload Mass by F9 v1.1

This query calculates the **average payload mass** carried by the booster version **F9 v1.1.**

- **Command Highlights:**
  - AVG(payload_mass__kg_): Computes the average payload mass.
  - WHERE booster_version LIKE '%F9 v1.1%': Filters for launches using the F9 v1.1 booster.

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

```
%%sql

SELECT AVG(PAYLOAD_MASS__KG_) AS Avg_Payload_Mass
FROM SPACEXTABLE
WHERE Booster_Version = 'F9 v1.1';
```

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

Avg_Payload_Mass
2928.4

First Successful Ground Landing Date

This query identifies the **earliest date** when a booster successfully landed on a ground pad.

- **Command Highlights:**
  - MIN(date): Finds the earliest date in the dataset.
  - WHERE landing_outcome = 'Success (ground pad)': Filters for successful ground pad landings

```
%%sql

SELECT MIN(Date) AS First_Successful_GroundPad_Landing
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (ground pad)';
```

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

First_Successful_GroundPad_Landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

This query lists the **boosters** that Successfully landed on Drone Ship with **Payload** between **4000 and 6000.**

- **Command Highlights:**
  - landing_outcome = 'Success (drone ship)': Filters Successfully landing on a drone ship
  - payload_mass__kg_ BETWEEN 4000 AND 6000: Filters Had a payload mass between 4000 kg and 6000 kg

```
%%sql

SELECT Booster_Version
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)'
  AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

```
 * 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

This query calculates the **total number** of **successful** and **failue mission outcomes.**

- **Command Highlights:**
  - COUNT(*) as total_number: Counting the number of occurrences for each unique mission outcome.
  - Group by mission_outcome: to aggregate similar outcomes

```
%%sql

SELECT Booster_Version
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)'
  AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

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

Boosters Carried Maximum Payload

This query list the names of the **booster** which have carried the **maximum payload mass.**

- **Command Highlights:**
  - payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) ...): Ensures only records with the maximum payload mass are included

```
%%sql

SELECT Booster_Version, PAYLOAD_MASS__KG_
FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTABLE
);
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

This query retrieves **failed landings on drone ships** for the year **2015,** focusing on key details such as **month, date, booster version, launch site,** and **landing outcome.**

- **Command Highlights:**
  - strftime('%m', date): Extracts the month from the launch date.
  - strftime('%Y', date) = '2015': Filters records for the year 2015.
  - landing_outcome = 'Failure (drone ship)': Ensures only failed landings on drone ships are included.

```
%%sql

SELECT
    substr(Date, 6, 2) AS Month,
    Landing_Outcome,
    Booster_Version,
    Launch_Site
FROM SPACEXTABLE
WHERE substr(Date, 1, 4) = '2015'
  AND Landing_Outcome LIKE 'Failure (drone ship)%';
```

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

Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

This query ranks the count of **landing outcomes** based on their frequency during the date **2010-06-04 and 2017-03-20**, in descending order.

- **Command Highlights:**
  - date BETWEEN '2010-06-04' AND '2017-03-20': ensures only launches within this timeframe are analyzed.
  - GROUP BY landing_outcome: aggregates records by distinct landing outcomes.
  - ORDER BY count_outcomes DESC: ranks outcomes by their frequency in descending order.

```
%%sql

SELECT
    Landing_Outcome,
    COUNT(*) AS Outcome_Count
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY Outcome_Count DESC;
```

```
 * 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

All Launch Sites on a Map

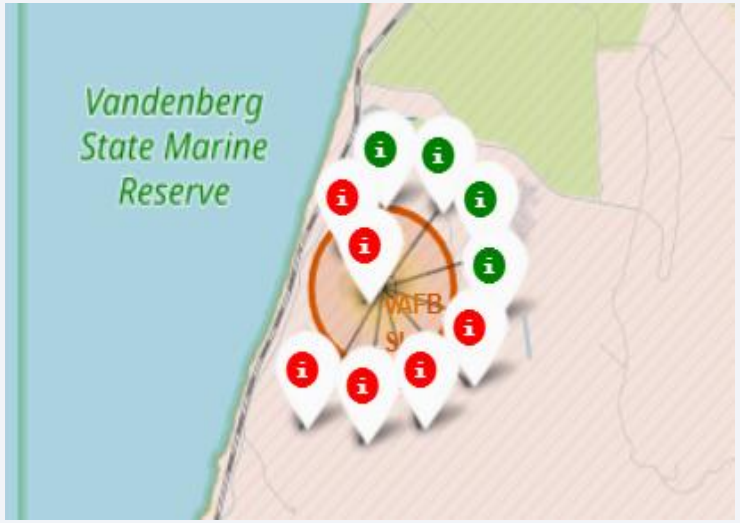
This map, created with **Folium**, uses **markers** and **circles** to visualize the precise geographical locations of SpaceX's launch sites.

Key findings:

- Launch sites are situated close to coastlines to minimize risks to populated areas.
- Rockets are launched over the ocean to ensure debris or failures occur far from human activity, enhancing safety

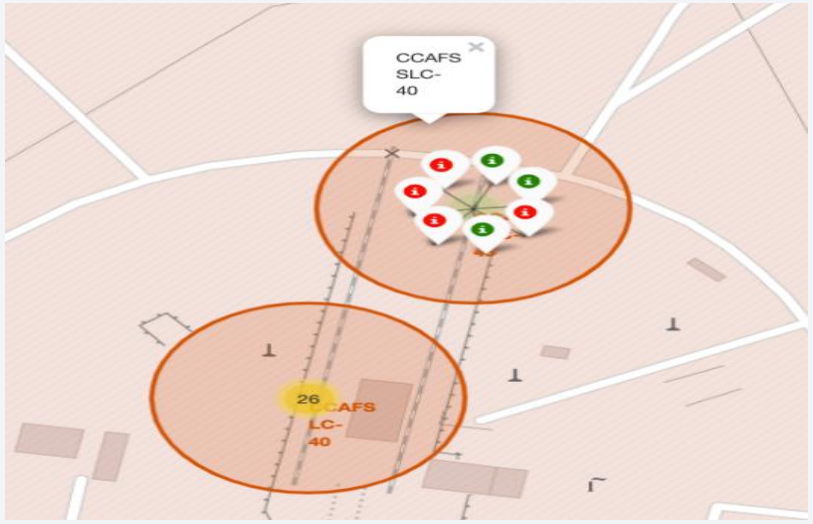
Visualization of Launch Site Success Rates

This **Folium-based map** provides an intuitive way to assess and compare success rates across different launch sites, leveraging **color-coded markers** for quick and clear insights



VAFB SLC-4E

KSC LC-39A



CCAFS LC-40 & CCAFS SLC-40

- **Color Representation:**
  - Green Marker: Indicates a successful launch.
  - Red Marker: Denotes a failed launch

- **Insights:**
  - Launch Site **KSC LC-39A** stands out with a **high success rate**, evident from the predominance of green markers

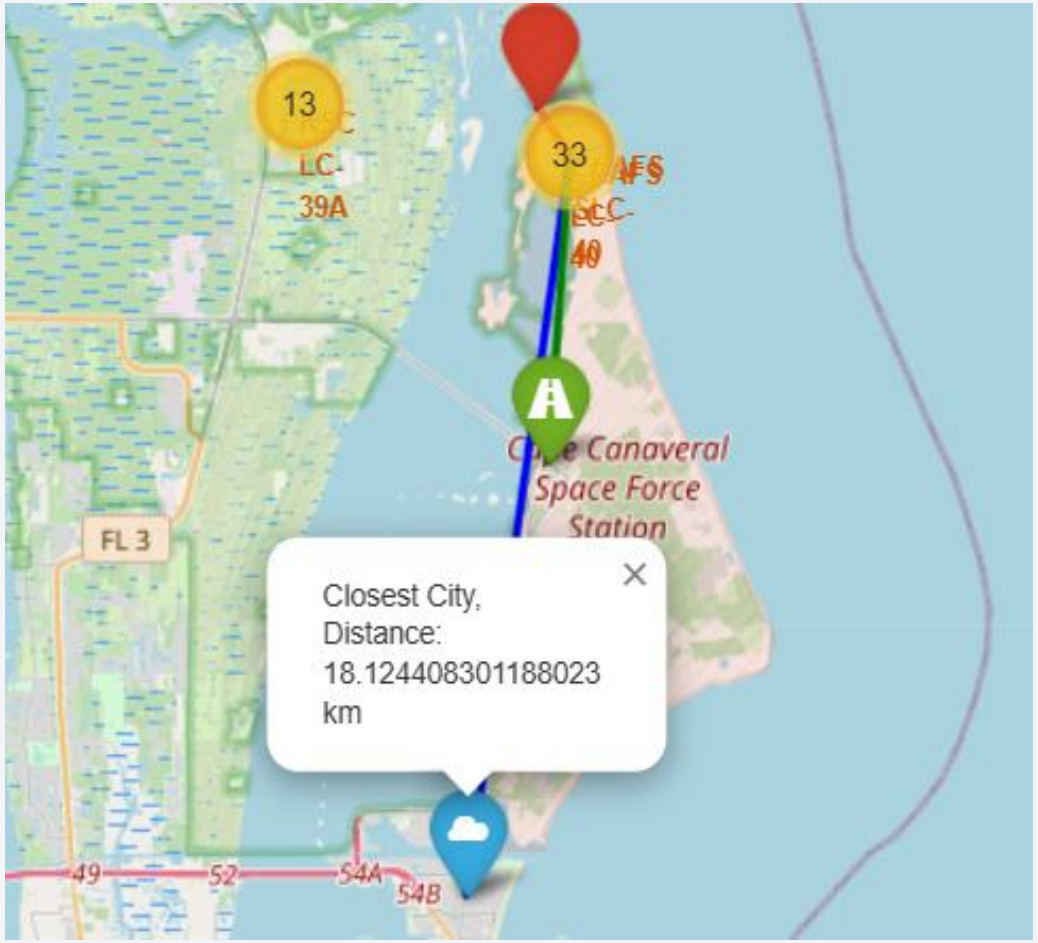
Analysis of Proximity for Launch Site CCAFS LC-40

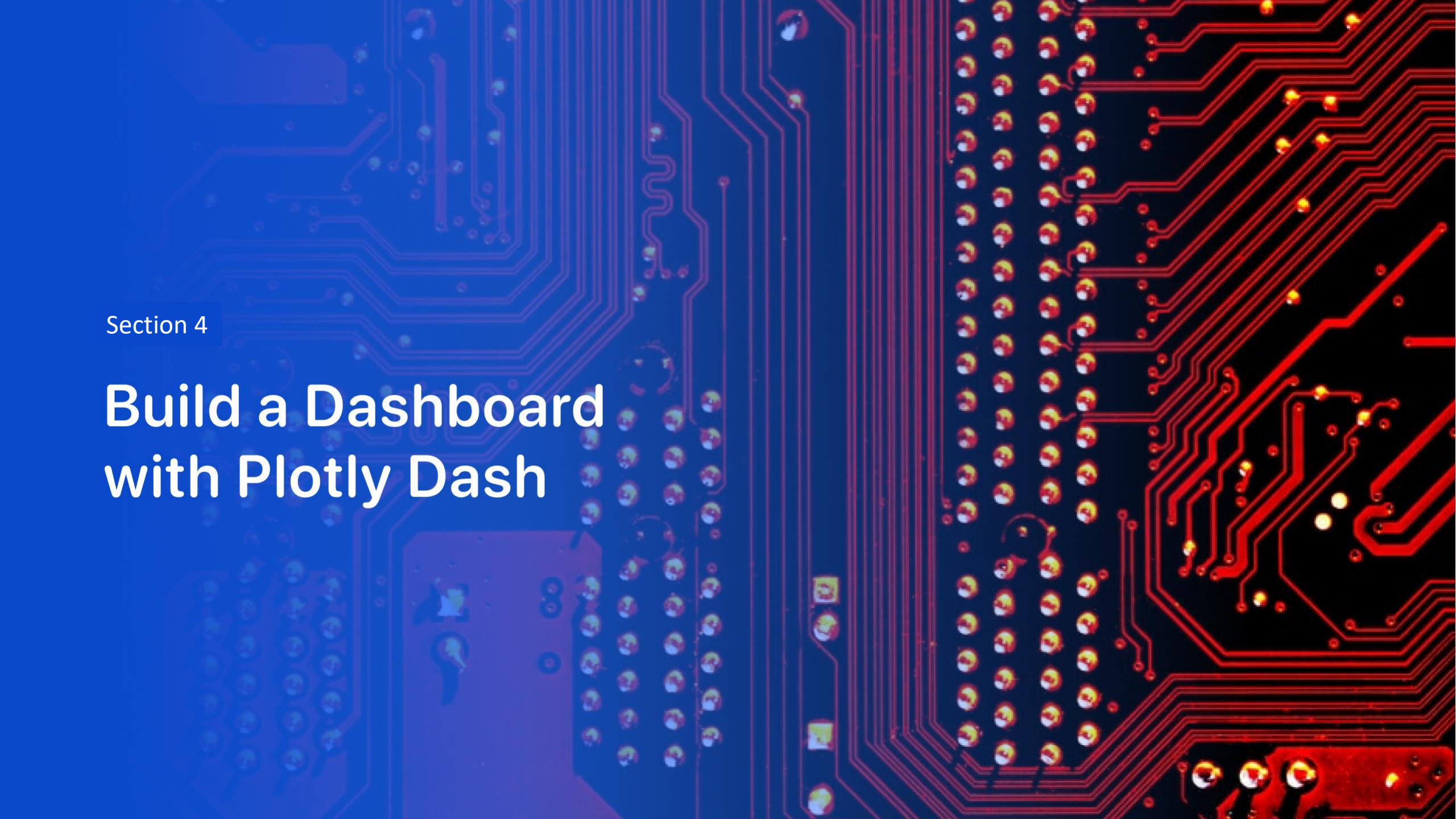
The geographic analysis of the **CCAFS LC-40** launch site reveals important proximity details to infrastructure and population centers:

1. **Marker used:**
   - Red represents Railway
   - Green represents Highway
   - Blue represents Closest City

2. **Proximity to Key Infrastructure:**
   - **Railway**: Located approximately 1.35 km away.
   - **Highway**: Found at a distance of 7.43 km

3. **Closest City**:
   - The closest city is 18.12 km far from launch site.



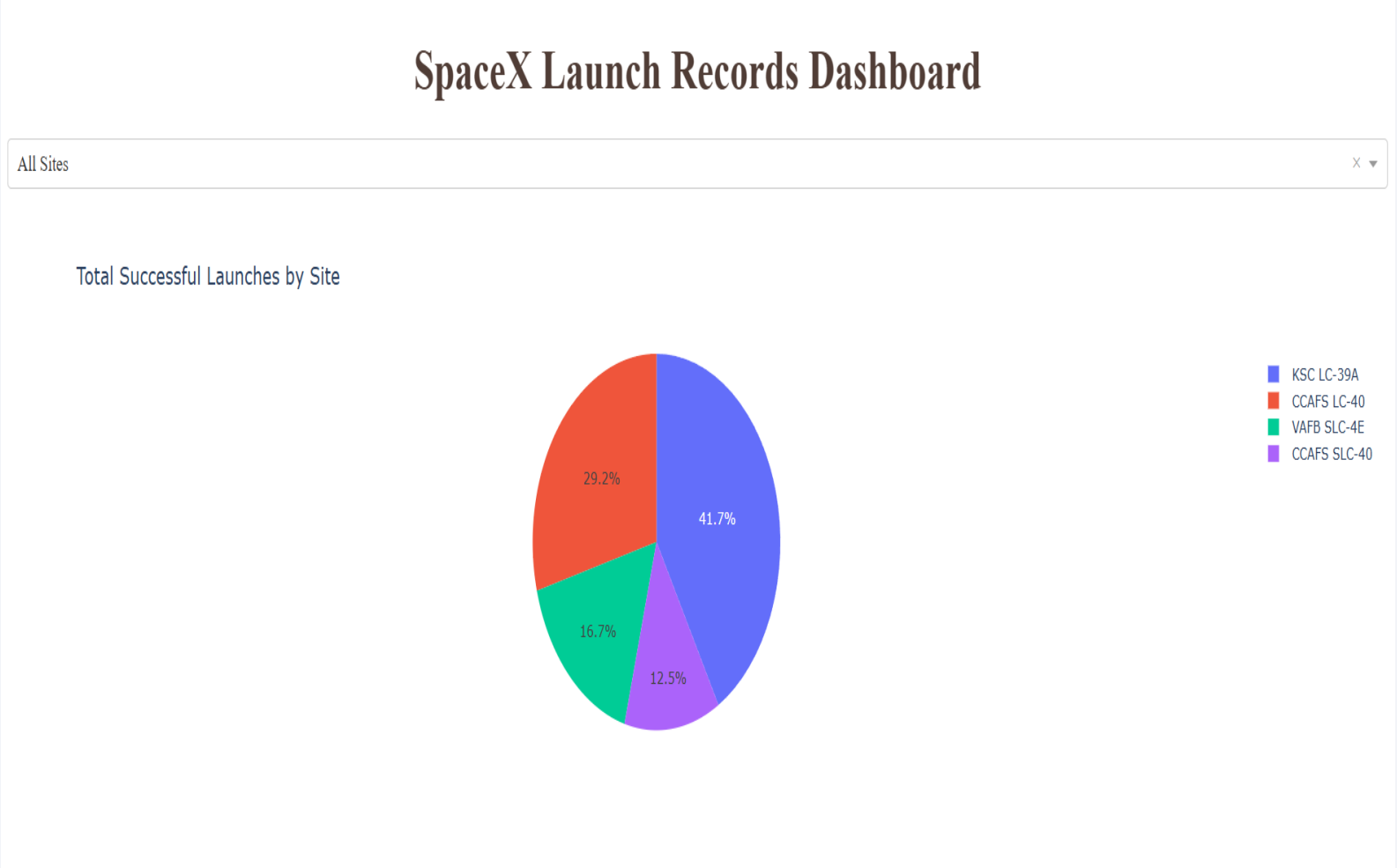

Section 4
Build a Dashboard with Plotly Dash

Successful Launches by Site

- The pie chart clearly shows the proportion of successful launches from each SpaceX launch site. Among the analyzed sites, **KSC LC-39A** stands out with the highest percentage of successful launches

- **Key Insights (% successful launches):**

- **KSC LC-39A**: 41.7%
- **CCAFS LC-40**: 29.2%.
- **VAFB SLC-4E**: 16.7%.
- **CCAFS SLC-40**: 12.5%

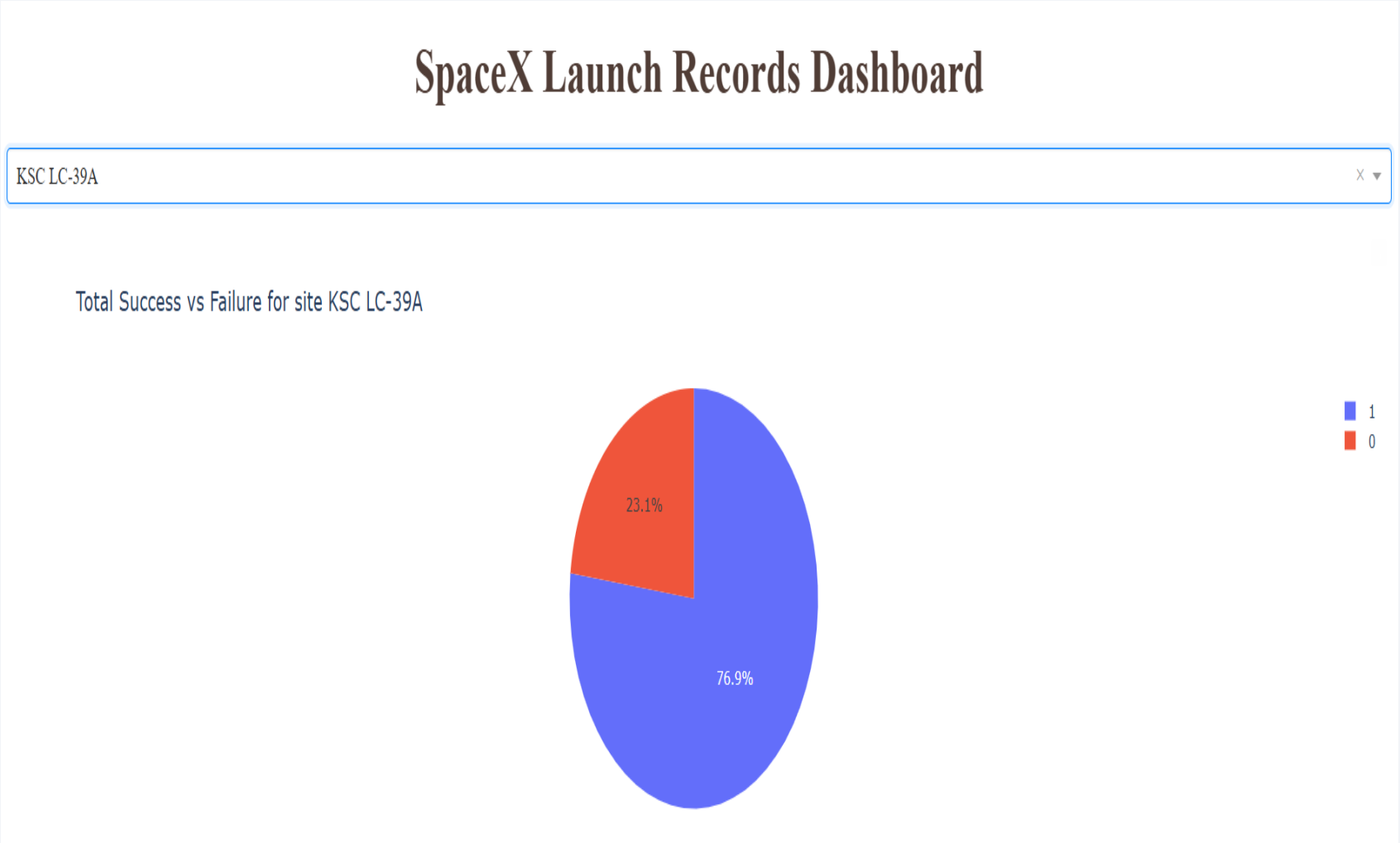


Launch Success Rate at KSC LC-39A

- **KSC LC-39A** demonstrates the highest success rate among all SpaceX launch sites, with an impressive **76.9%** success rate

- **Key Metrics:**

- Total launches: 13
- Successful landings: 10
- Failed landings: 3

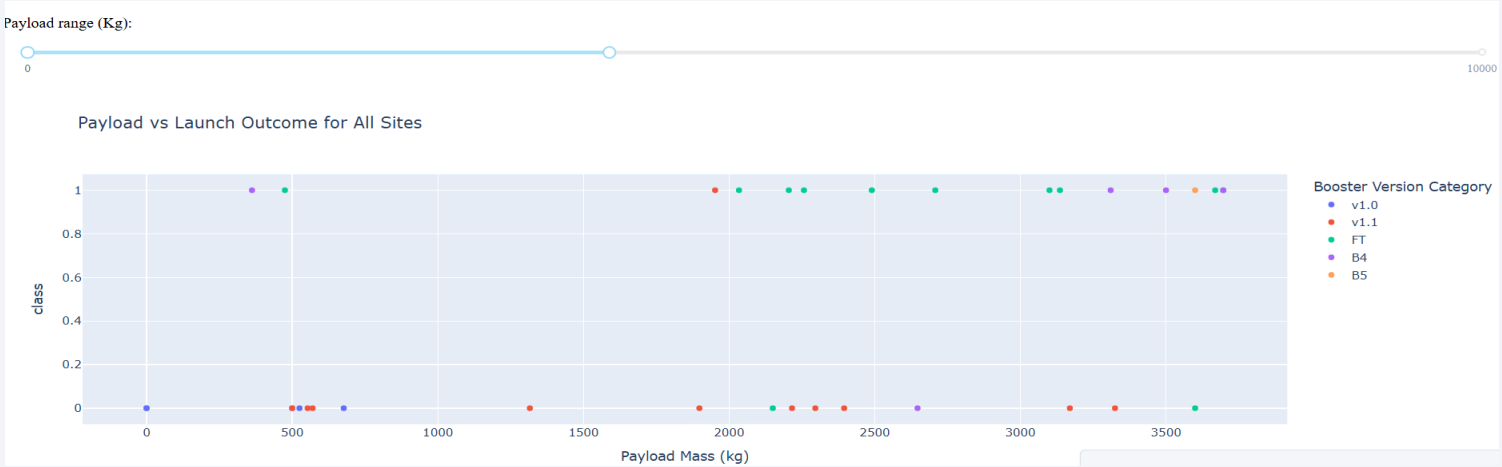


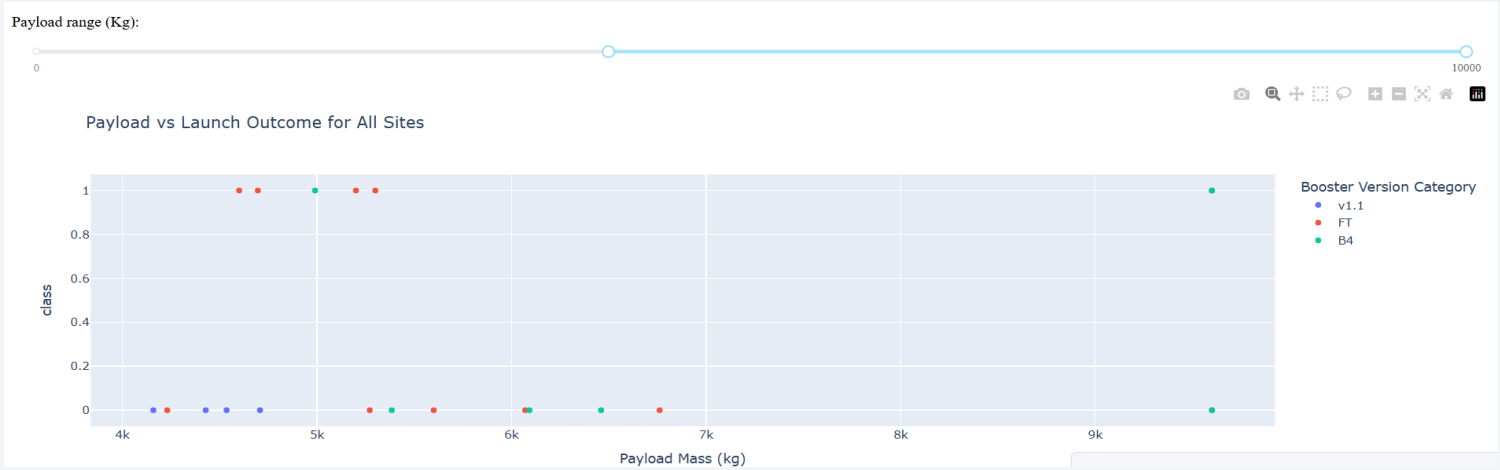
Payload vs. Launch Outcome scatter plot for all sites

- We will split the the data into 2 ranges:
  - 0-4000 kg (low payloads)
  - 4000-10000 kg (massive payloads)

- From these two plots, it can be shown that the success for massive payloads is lower than that for low payloads.

- Booster Version **FT** and **B4** stands out with relatively better success rates in both range.





Section 5

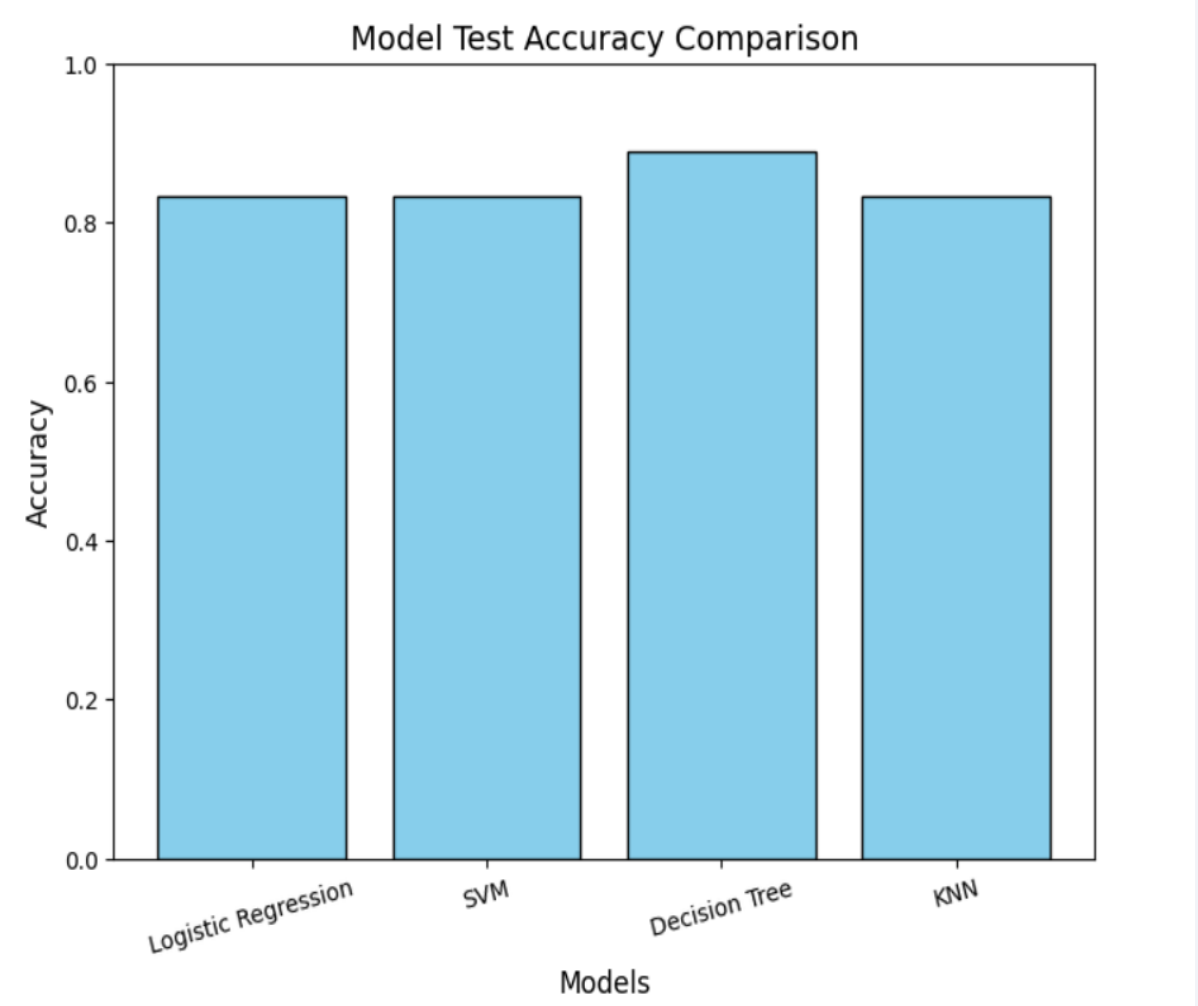
Predictive Analysis (Classification)

Classification Accuracy

- As we can see that the model accuracy for three types are same that is **83.33%.**
  - **Why this happened?**
  - Bcz **test set is small** (only 18 samples: 6 of class 0, 12 of class 1).

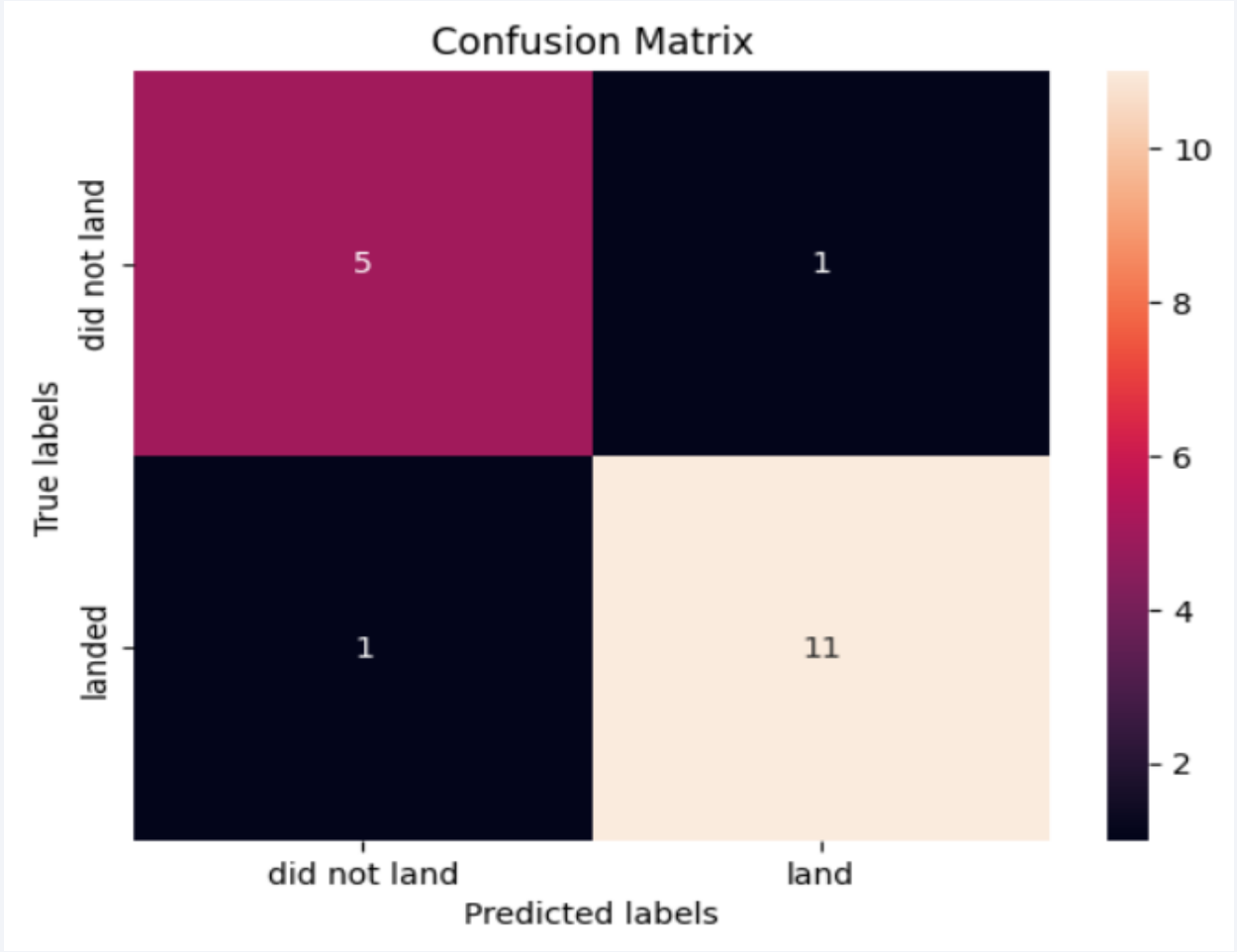
- But, the model accuracy for **Decision Tree** is **88.8%.**

- **Overall Performance:** The model demonstrates robust classification ability for both successful and unsuccessful landings, making it the most effective among the tested models



Confusion Matrix

- The confusion matrix for all classification models are also same.
- This confusion matrix shows that:
  - **True Postive** - 11 (True label is landed, Predicted label is also landed)
  - **False Postive** - 1 (True label is not landed, Predicted label is landed)
  - **True Negative** - 5 (True label is not landed, Predicted label is not landed)
  - **False Negative** - 1 (True label landed, Predicted label is not landed)



Conclusions

- As the number of flights increases, the rate of success at a launch site increases, with most of the early flights being unsuccessful, but with more experience, the success rate increases.
  - Between 2010 and 2013, all landings were unsuccessful **(as success rate is 0).**
  - After 2013, the success rate generally **increased.**
  - After 2016, there was always a greater than 60% chance of success.
- Orbit types ES-L1, GEO, HEO, and SSO have highest (100%) success rate.
  - The 100% success rate of **GEO, HEO, and ES_L1** orbits can be explained by only having 1 flight into their respective orbits.
  - The 100% success rate of **SSO** orbit is more impressive with **5** successful flights..
  - For **LEO** orbit, A clear positive correlation is observed as Flight Number increases, the success rate also increases
- The launch site **KSC LC-39 A** had the most successful launches, with **41.7%** of the total successful launches, and also the highest rate of successful launches, with a **76.9%** success rate.
- The success for massive payloads (over 4000 kg) is lower than that for low payloads.
- The best performing classification model is Decision tree Model, as it has a best Accuracy of **88.8%.**

Appendix

❑Project Repository

The complete codebase, data, and visualizations for this project can be found in the GitHub repository:

GitHub Repository: https://github.com/Shubham-Rathore08/IBM_SpaceX_capstone_project/tree/main

This repository contains:

- All Python scripts used for data processing, visualization, and modeling.
- Dash application implementation for interactive visual analytics.
- Jupyter Notebooks documenting the analysis, SQL queries, and machine learning pipelines.
- CSV datasets used throughout the project.

❑Acknowledgments

Special thanks to the instructors of the IBM Data Science Professional Certificate program on Coursera for their guidance and expertise

Thank you!