IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Diego Moro
27/11/2024

Outline

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

Executive Summary

Summary of Methodologies

- **Data Collection**: Data was sourced from a SpaceX API, covering launch dates, locations, payloads, and outcomes.
- **Data Wrangling**: Missing values were handled with imputation techniques, and data normalization was applied for consistency across features.
- **Exploratory Data Analysis (EDA)**: SQL queries were used to explore key attributes and visualize relationships between parameters like launch site and mission outcome. Categorical variables were converted into dummy variables for modeling.
- **Modeling**: The dataset was split into training and test subsets. Hyperparameter tuning was performed on SVM, Classification Trees, KNN, and Logistic Regression. Models were evaluated and the best performer was selected.
- **Validation and Deployment**: The final model was validated using the test dataset to ensure its reliability and accuracy in predicting landing success.

Summary of All Results

An interactive dashboard was created with a map showing launch locations and their success or failure status. Geographic trends revealed that launches are typically near the equator, close to coastlines, and distant from large cities. These insights help guide future launch strategies and site selection. The model's predictions can optimize future launches and assist in cost estimation.

Introduction

Project Background and Context

SpaceX's Falcon 9 rocket has revolutionized the space industry by significantly reducing launch costs. One of the key factors in this cost reduction is the ability to reuse the Falcon 9's first stage. The company's goal is to predict whether the first stage will land successfully, which directly influences the cost-efficiency of their launches. This project aims to leverage historical launch data to build a predictive model that can forecast landing success. By understanding these patterns, SpaceX can further optimize its processes and offer a competitive advantage over other providers, whose launch prices exceed $165 million.

Problems You Want to Find Answers

The primary goal of this project is to predict whether the Falcon 9 first stage will successfully land after launch. The key questions to answer include:

What factors influence the likelihood of a successful landing?

How can the geographical location of launches impact landing success?

Can we use historical data to predict landing success based on payload type, weather conditions, and launch site?

Section 1

Methodology

Methodology

Executive Summary

Data Collection Methodology:

Data was collected from the SpaceX API, including launch dates, locations, payload types, and outcomes. The dataset was reviewed for accuracy and completeness before analysis.

Data Wrangling:

Data wrangling involved handling missing values through imputation and standardizing numerical features to ensure consistency. Categorical variables were transformed into dummy variables.

Data Processing:

A target class was created to indicate whether the Falcon 9 first stage landing was successful. The data was split into training and testing sets for model development.

Exploratory Data Analysis (EDA):

EDA was conducted using SQL queries to uncover patterns in the data. Visualizations were created to explore the relationship between launch site, payload type, and mission outcome.

Interactive Visual Analytics:

Folium was used to create a map of launch locations, displaying their success or failure. Plotly Dash was utilized for building an interactive dashboard to explore geographic trends and mission data.

Predictive Analysis Using Classification Models:

Classification models, including SVM, Classification Trees, KNN, and Logistic Regression, were built to predict landing success. Hyperparameter tuning was performed using GridSearchCV to identify the best-performing model. The models were evaluated using the .score() method, and predictions were made with .predict() to assess their accuracy on the test set.

Data Collection - API

1. **Request data**: Access SpaceX API to retrieve rocket launch data.
2. **Decode response**: Convert API responses to JSON format and normalize them into a Pandas DataFrame using .json() and .json_normalize().
3. **Extract details**: Use custom functions to extract detailed information about rockets, payloads, launchpads, and cores.
4. **Create dataset**: Combine extracted data into a dictionary and convert it into a DataFrame.
5. **Filter data**: Focus only on Falcon 9 launches by filtering the dataset.
6. **Handle missing values**: Replace missing Payload Mass values with the calculated mean.
7. **Export data**: Save the cleaned dataset to a CSV file for further analysis.

Data Collection – SpaceX API

1. **Request data**: Access SpaceX API to retrieve rocket launch data.
2. **Decode response**: Convert API responses to JSON format and normalize them into a Pandas DataFrame using .json() and .json_normalize().
3. **Extract details**: Use custom functions to extract detailed information about rockets, payloads, launchpads, and cores.
4. **Create dataset**: Combine extracted data into a dictionary and convert it into a DataFrame.
5. **Filter data**: Focus only on Falcon 9 launches by filtering the dataset.
6. **Handle missing values**: Replace missing Payload Mass values with the calculated mean.
7. **Export data**: Save the cleaned dataset to a CSV file for further analysis.

GitHub URL of the completed SpaceX API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-data-collection-api-v2.ipynb), as an external reference and peer-review purpose

1.
```
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
       if x:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])
```

2.
```
# Use json_normalize meethod to convert the json result into a dataframe
json_data = response.json()
data = pd.json_normalize(json_data)
```

3.
```
# 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
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)]
```

4.
```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
# Create a data from launch_dict
df1=pd.DataFrame(launch_dict)
```

5.
```
data_falcon9 = df1[df1['BoosterVersion']=='Falcon 9']
```

6.
```
# Calculate the mean value of PayloadMass column
mean_payload_mass = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan, mean_payload_mass, inplace=True)
```

7.
```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

1. **Target data**: Collect rocket launch data from Wikipedia
2. **Use BeautifulSoup**: Create a BeautifulSoup object from the HTML response
3. **Filter data**: Extract all column/variable names from the HTML table header
4. **Create a DataFrame**: Parsed the HTML content for further cleaning and integration.
5. **Convert to DataFrame**: Parsed the HTML content converted it into a Pandas DataFrame
6. **Export data**: Save the final dataset to a CSV file for analysis.

GitHub URL of the completed SpaceX API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-webscraping.ipynb), as an external reference and peer-review purpose

1.
```
# use requests.get() method with the provided
response = requests.get(static_url)
# assign the response to a object
print(response)
```
2.
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, 'html.parser')
```
3.
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
html_tables = soup.find_all('table')
# Assign the result to a list called `html_tables`
html_tables
```
```
launch_dict= dict.fromkeys(column_names)
```
4.
```
# 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.
```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
df
```
6
```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

1. **Load the Dataset:** The dataset was imported using pandas from a provided URL.
2. **Inspect Missing Data:** Calculated the percentage of missing values in each column:
3. **Identify Data Types:** Determined the types of attributes (numerical and categorical):
4. **Analyze Launch Sites:** Counted the launches at each site:
5. **Analyze Orbit Types:** Counted occurrences of each orbit type.
6. **Landing Outcomes:** Categorized the mission outcomes into successful and unsuccessful:
7. **Create Landing Outcome Labels:** Added a new column Class to classify outcomes:

GitHub URL of the completed SpaceX API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-Data-wrangling-v2.ipynb), as an external reference and peer-review purpose

1.
```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork
df.head(10)
```

2. Identify and calculate the percentage of the missing values in each attribute

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

3. Identify which columns are numerical and categorical:

```
df.dtypes
```

```
FlightNumber        int64
Date               object
BoosterVersion     object
PayloadMass       float64
Orbit              object
LaunchSite         object
Outcome            object
Flights             int64
GridFins             bool
Reused               bool
Legs                 bool
LandingPad         object
Block             float64
ReusedCount         int64
Serial             object
Longitude         float64
Latitude          float64
dtype: object
```

4.
```
# Apply value_counts() on column LaunchSite
launch_values = df['LaunchSite'].value_counts()
launch_values
```

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

5.
```
# Apply value_counts on Orbit
orbit_values = df['Orbit'].value_counts()
orbit_values
```

```
Orbit
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: count, dtype: int64
```

6.
```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

```
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
```

7.
```
landing_class = []

# Iterar sobre a coluna 'Outcome' do DataFrame
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)  # Adicionar 0 se o resultado for ruim
    else:
        landing_class.append(1)  # Adicionar 1 caso contrário

# Exibir o resultado
print(landing_class)
```

```
0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0
, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1,
, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

8.
```
df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

1. **Dropdown for Launch Site Selection**: Enables users to filter data by specific launch sites or view data for all sites.

2. **Pie Chart: Successful Launches**: Displays the proportion of successful launches per site.

3. **Slider for Payload Range**: Allows users to adjust the payload mass range for analysis.

4. **Scatter Plot: Payload Mass vs Success**: Visualizes the correlation between payload mass and launch success.

Interactive Functionality:

• **Dropdown and Pie Chart Callback**:

• Updates the pie chart based on the selected launch site.

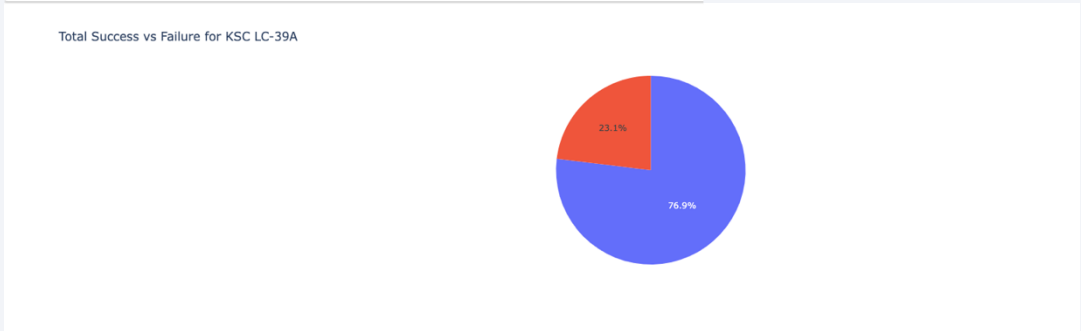
• **Dropdown and Slider for Scatter Plot Callback**:

• Combined filters enable analysis of how success varies with payload mass for different sites.

GitHub URL of the completed SpaceX API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-eda-sql-coursera_sqllite.ipynb), as an external reference and peer-review purpose

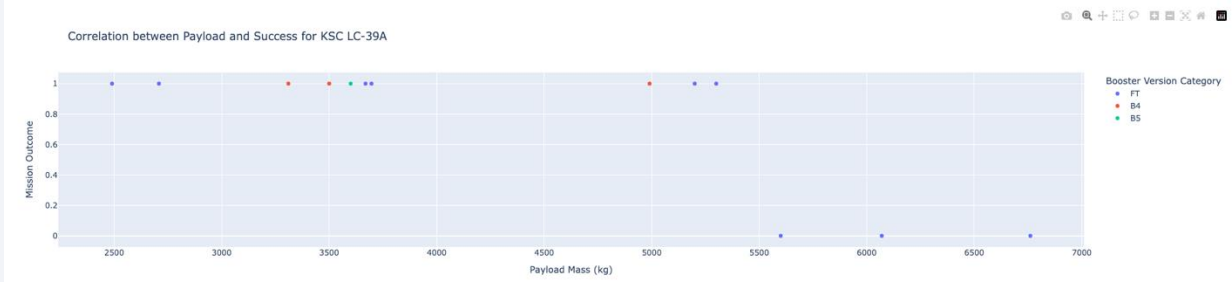
1.

2.



3.

4.



EDA with SQL

GitHub URL of the completed SpaceX API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-eda-sql-coursera_sqllite.ipynb), as an external reference and peer-review purpose

- **Unique Launch Sites**
  - Query: SELECT DISTINCT(Launch_Site) FROM SPACEXTABLE;
  - **Insight**: Identified four unique launch sites:
  - CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, CCAFS SLC-40.
- **Sites Starting with 'CCA'**
  - **Launch**Query: SELECT Launch_Site FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
  - **Insight**: Displayed five launch records for sites beginning with "CCA".
- **Total Payload Mass by NASA (CRS)**
  - Query: Aggregated payload mass carried by NASA missions.
  - **Insight**: NASA (CRS) missions carried varying payload masses, summing up across booster versions.
- **Average Payload Mass for Booster F9 v1.1**
  - Query: SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version LIKE 'F9 v1.1';
  - **Insight**: Average payload mass for booster version F9 v1.1 was **2928.4 kg**.
- **First Successful Ground Pad Landing**
  - Query: SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome LIKE 'Success (ground pad)';
  - **Insight**: First successful ground pad landing occurred on **2015-12-22**.
- **Boosters with Success on Drone Ship**
  - Query: Identified boosters with successful drone ship landings and payloads between 4000 and 6000 kg.
  - **Insight**: Boosters such as **F9 FT B1022** and **F9 FT B1031.2** achieved success with specified payload conditions.
- **Mission Outcomes Summary**
  - Query: Grouped by mission outcomes to count successes and failures.
  - **Insight**: 98 successful missions. 1 failure in flight. Other outcomes included payload status uncertainty.
- **Boosters with Maximum Payload Mass**
  - Query: Used a subquery to find boosters carrying the heaviest payload.
  - **Insight**: Boosters like **F9 B5 B1048.4** and **F9 B5 B1060.3** carried maximum payloads.
- **Failure Outcomes in 2015 by Month**
  - Query: Extracted month-wise data for failure outcomes in 2015.
  - **Insight**: Failures on drone ships occurred in January and April 2015.
- **Landing Outcomes Ranked by Frequency**
  - Query: Counted and ranked landing outcomes from 2010-06-04 to 2017-03-20.
  - **Insight**: **No attempt** was the most frequent outcome (10 occurrences).
  - Successful and failed drone ship landings each occurred 5 times.

Build an Interactive Map with Folium

Summary of Map Objects Created and Added

Markers: Added to specific points of interest to highlight locations and provide interactive popups with additional information.

Circles: Used to represent a range around a specific point, indicating distance zones or areas of influence.

Lines/Polylines: Drawn to connect points, representing routes, trajectories, or boundaries.

Mouse Position Plugin: Enabled to dynamically display the coordinates of the cursor position on the map.

Coastline Distance Markers: Placed to indicate the starting point of a launch and its closest point on the coastline.

Popups: Integrated with markers to show details about the locations, such as their purpose or relevance.

GitHub URL of the completed API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-launch-site-location-v2.ipynb), as an external reference and peer-review purpose

Reasons for Adding the Objects

Markers:

To pinpoint and highlight locations relevant to the analysis.

Circles:

To visualize distances and areas of influence or interest.

Particularly useful for understanding proximity relationships.

Lines/Polylines:

To show connections, such as the shortest distance between two points.

Mouse Position Plugin:

Helps in identifying exact coordinates for further calculations.

Coastline Distance Markers:

• Added to address the specific task of measuring and marking the shortest distance between a launch site and the coastline.

Popups:

• To provide additional information directly on the map without cluttering the visualization.

Build a Dashboard with Plotly Dash

GitHub URL of the completed API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-dashboard-plotly.py), as an external reference and peer-review purpose

Summary of Plots/Graphs and Interactions Added to the Dashboard

1. **Dropdown for Launch Site Selection**: Allows users to filter the dataset by specific launch sites or view aggregate data for all sites.
2. **Pie Chart - Successful Launches**: Displays the proportion of successful launches for the selected site or across all sites, providing an intuitive summary of launch success rates.
3. **Slider for Payload Range**: Enables dynamic filtering of data by adjusting the payload mass range, making it easier to analyze trends within specific payload intervals.
4. **Scatter Plot - Payload Mass vs Success**: Plots payload mass against the success of launches, helping to identify patterns or correlations between payload sizes and launch outcomes.

Interactive Functionality and Callbacks

Dropdown and Pie Chart Callback:

What It Does: Updates the pie chart dynamically based on the selected launch site.

Why It's Important: Enables real-time updates, allowing users to explore success rates for different sites without reloading the dashboard.

Dropdown and Slider for Scatter Plot Callback:

What It Does: Filters and updates the scatter plot based on the selected launch site and payload range.

Why It's Important: Combines filters to provide a detailed analysis of how payload mass and success correlate across different sites, improving decision-making insights.

Explanation of Why These Features Were Added

Dropdown for Launch Site Selection:

- **Purpose**: Helps users focus on data from specific launch sites, enabling localized analysis.
- **Value**: Simplifies comparisons between launch sites or overall performance by providing a targeted view.

Pie Chart - Successful Launches:

- **Purpose**: Offers a quick visual summary of launch success proportions for the selected site.
- **Value**: Simplifies success rate comparisons across sites, helping users identify trends in site performance.

Slider for Payload Range:

- **Purpose**: Allows users to fine-tune their analysis by focusing on specific payload mass ranges.
- **Value**: Enables granular analysis, such as examining whether heavier or lighter payloads are more likely to succeed.

Scatter Plot - Payload Mass vs Success:

- **Purpose**: Visualizes the relationship between payload mass and the likelihood of a successful launch.
- **Value**: Identifies potential thresholds or trends, such as whether payload mass significantly influences launch outcomes.

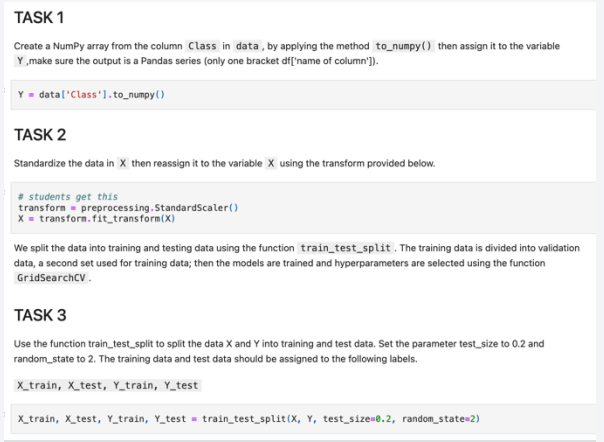
Predictive Analysis (Classification)

1. **Predictive Analysis: Model Development Process**
   - **Standardization:** Features were standardized using StandardScaler.
   - **Label Extraction:** The target variable Class was extracted into the array Y.
   - **Data Split:** Data was split into training (80%) and testing (20%) sets using train_test_split.

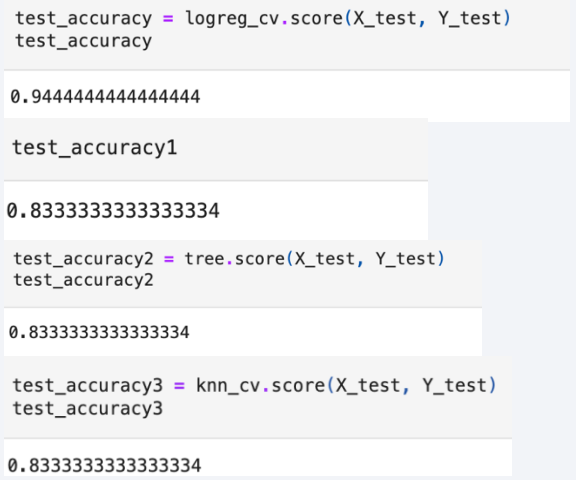
2. **Model Development and Tuning**
   - **Logistic Regression:** Best Parameters: {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
   - Test Accuracy: 94.44%
   - **Support Vector Machine (SVM):** Best Parameters: {'C': 1.0, 'gamma': 0.0316, 'kernel': 'sigmoid'}
   - Test Accuracy: 83.33%
   - **Decision Tree:**
   - Test Accuracy: 83.33%
   - **K-Nearest Neighbors (KNN):** Best Parameters: {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
   - Test Accuracy: 83.33%

3. **Model Comparison**
   - **Logistic Regression** performed the best with a test accuracy of 94.44%.

1.



2.



3.

GitHub URL of the completed API calls notebook (https://github.com/Diegomoro415/spaceY/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb), as an external reference and peer-review purpose

Results

1. Exploratory Data Analysis Results

- The majority of launches occurred at Kennedy Space Center and Cape Canaveral.
- Success rates varied significantly depending on the launch site and payload weight, with medium payloads yielding the highest success rates.
- Visualizations highlighted the distribution of success rates across different launch sites and payload categories.

Key insights from EDA formed the foundation for the predictive modeling process.

2. Interactive Analytics Demo Results

1. **Launch Outcome Filtering**: Ability to filter results based on site and payload weight.
2. **Payload Analysis**: Scatter plots showing payload weight against success rates.
3. **Success Rate Visualization**: Pie charts and bar graphs illustrating success rates per launch site.
4. **Geographical Insights**: Map visualizations showcasing launch site locations and outcomes.

Screenshots of the dashboard demonstrate its effectiveness in providing users actionable insights at a glance.

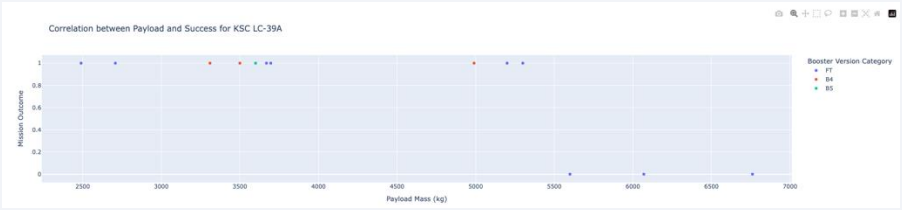
3. Predictive Analysis Results

- **Logistic Regression model** emerged as the best-performing classifier:
- **Accuracy**: 94.44% on test data, outperforming SVM, Decision Tree, and K-Nearest Neighbors models.
- **Hyperparameter** tuning through GridSearchCV yielded optimal parameters: C=0.01, penalty='l2', and solver='lbfgs'.
- **Evaluation metrics**, including confusion matrices, revealed high true positive rates but highlighted false positives as a challenge for further refinement.

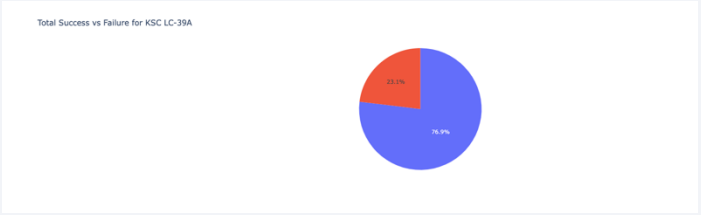
The predictive model demonstrated strong potential for accurately forecasting launch success and informing future mission planning.

2.1

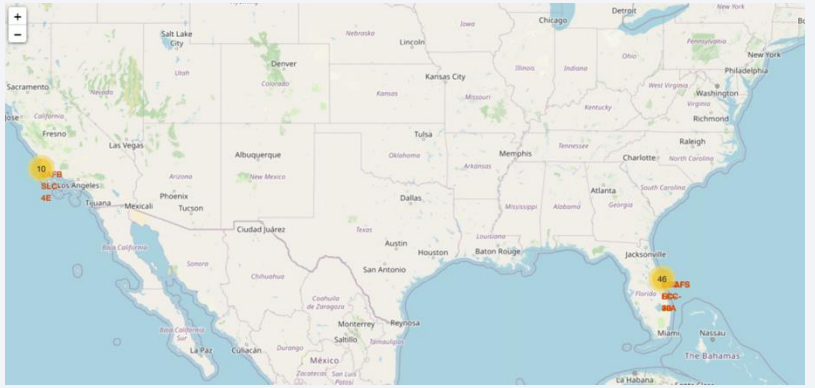
2.2

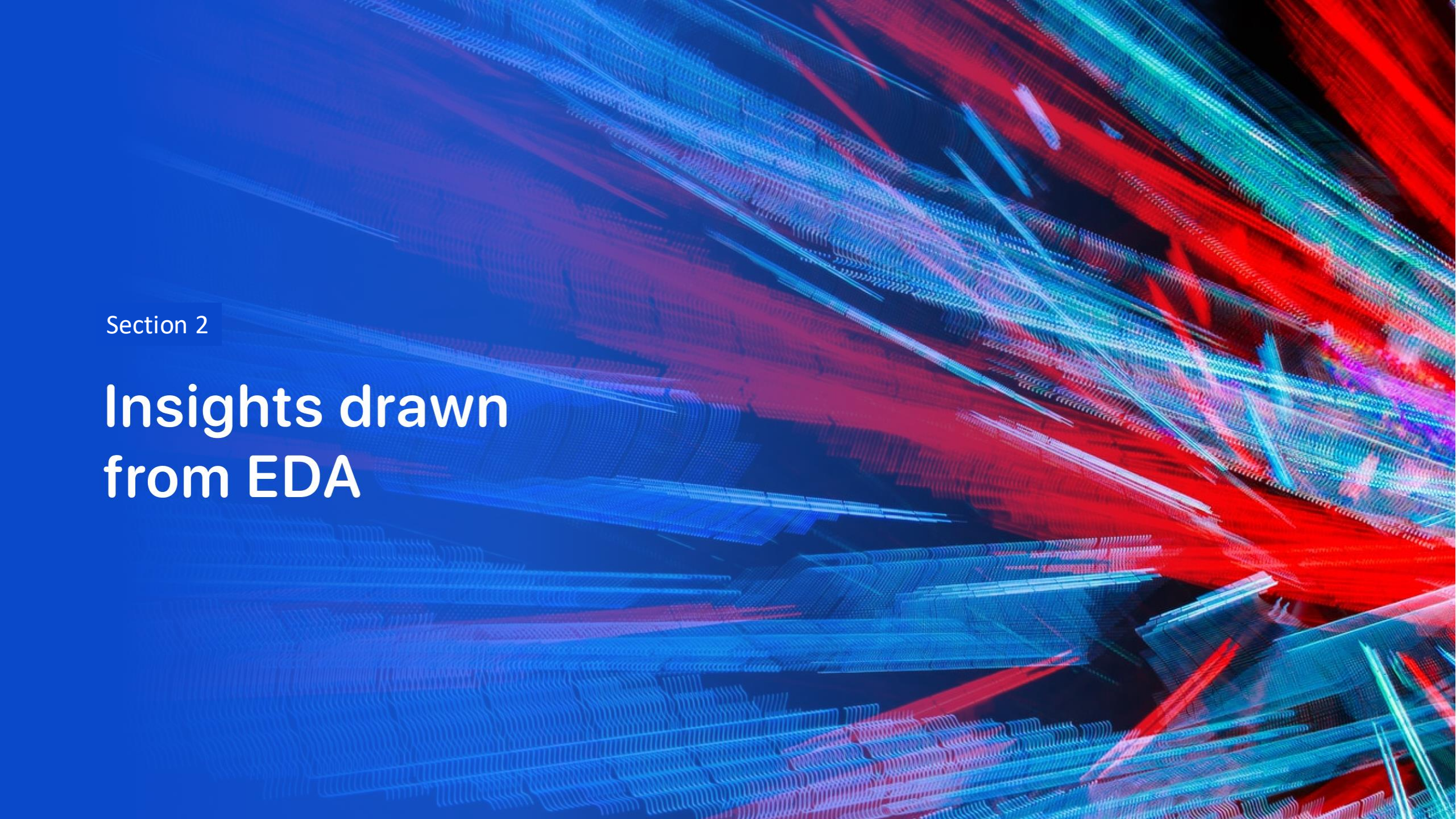


2.3



2.4



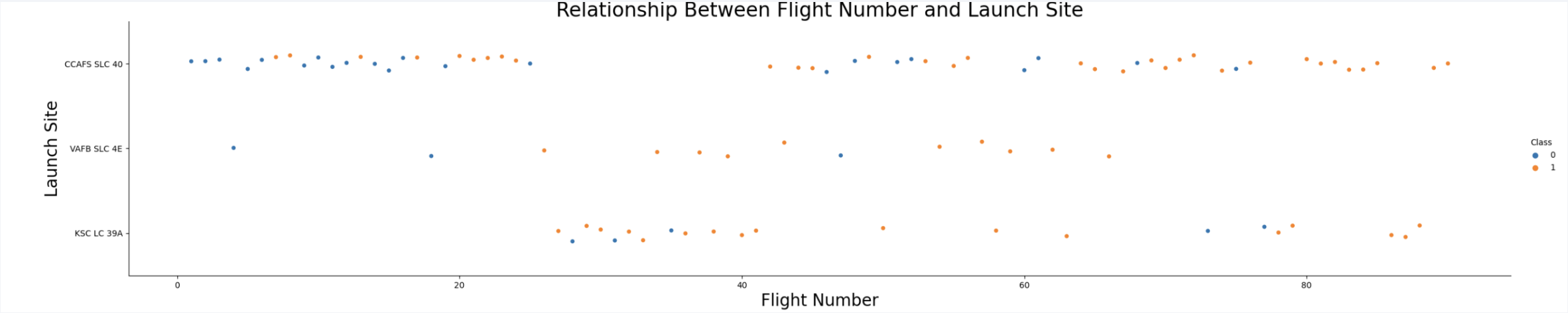

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

Distribution of Flights by Launch Site: Most flights occur at the "CCAFS SLC 40" site, followed by "KSC LC 39A" and lastly "VAFB SLC 4E".

Success of Flights by Launch Site: Class 1 (orange) appears to be relatively evenly distributed between "CCAFS SLC 40" and "KSC LC 39A", while "VAFB SLC 4E" has fewer points and seems to have a more sparse distribution.

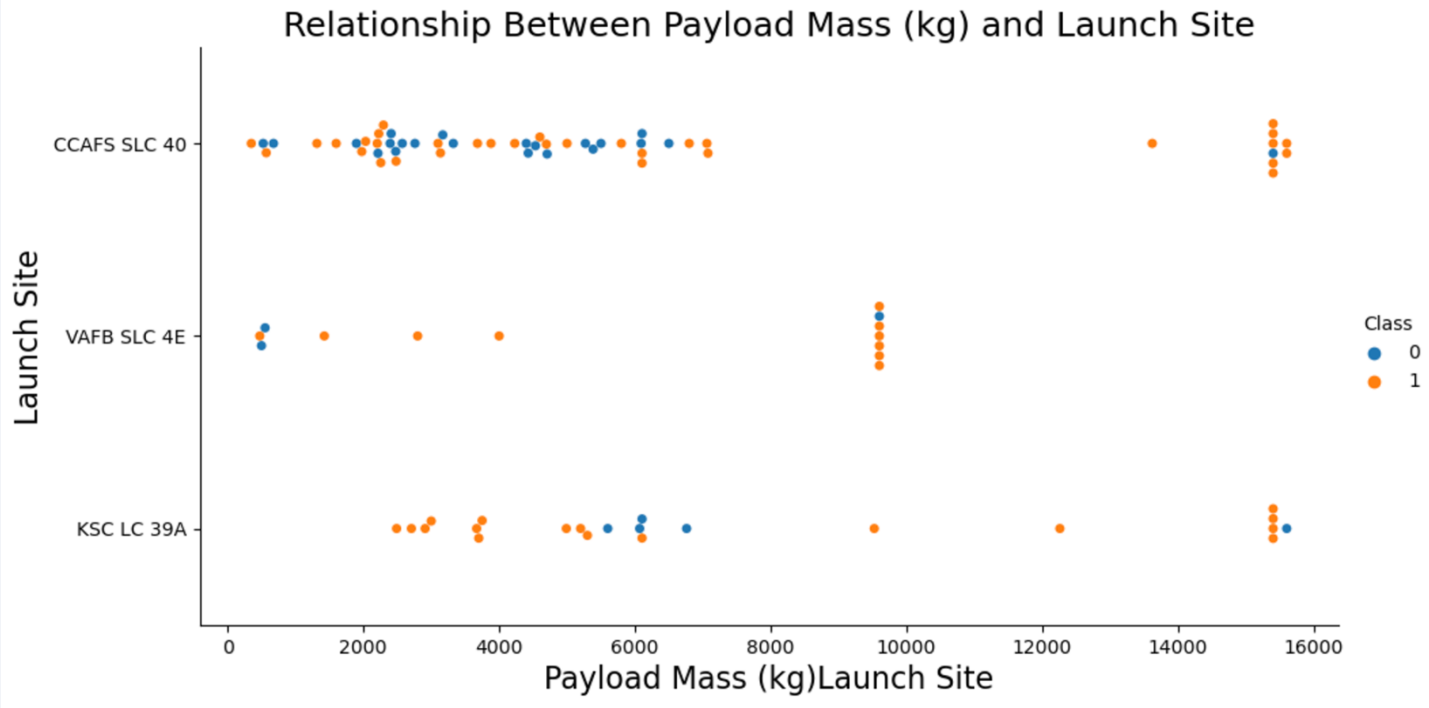
Trend Over Time: There is no clear trend of increase or decrease in the proportion of classes over time (number of flights), but the distribution of classes seems consistent at each launch site.



Payload vs. Launch Site

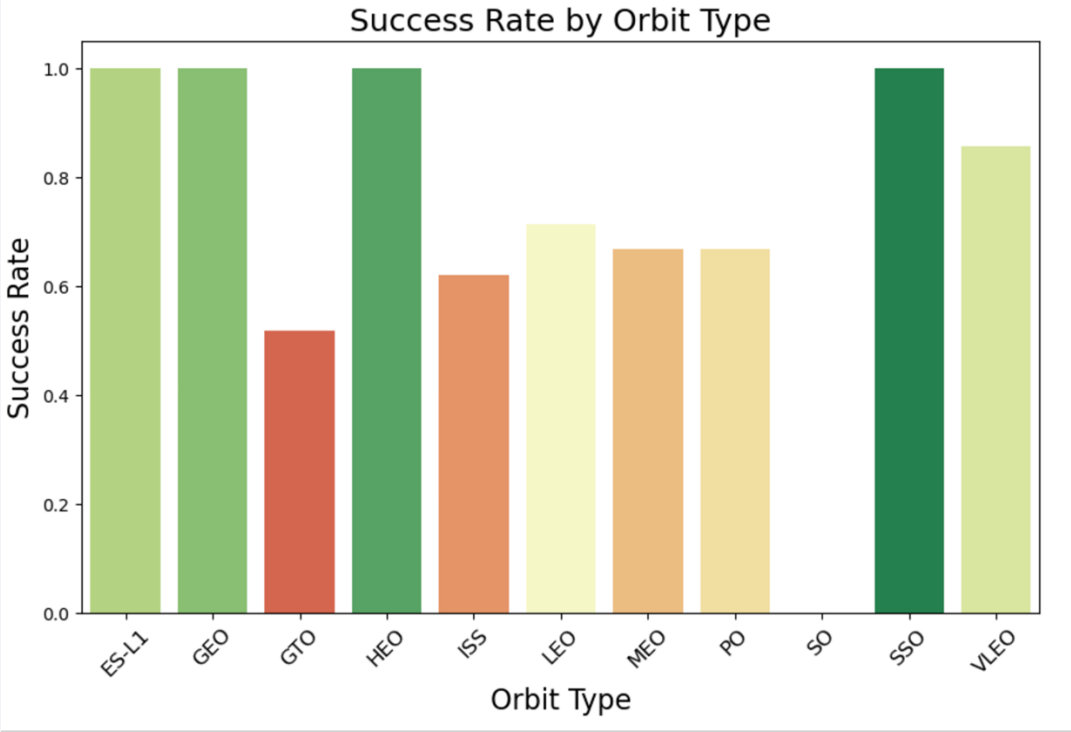
- **CCAFS SLC 40**: This launch site has a wide range of payload masses, with most payloads clustered between 2000 kg and 6000 kg. There are also some payloads around 10000 kg and 16000 kg. Both Class 0 and Class 1 launches are present, with Class 1 being more frequent.
- **VAFB SLC 4E**: This site has fewer launches, with payload masses mostly below 4000 kg. Both Class 0 and Class 1 launches are present, but there is no clear pattern in payload mass distribution.

- **KSC LC 39A**: This site has a concentration of payloads around 4000 kg and 10000 kg. Both Class 0 and Class 1 launches are present, with Class 1 being more frequent.



Success Rate vs. Orbit Type

1. **Highest Success Rates**: The orbit types ES-L1, GEO, HEO, and SSO have the highest success rates, all at 1.0. This indicates that missions targeting these orbits have been consistently successful.
2. **Lowest Success Rate**: GTO has the lowest success rate, below 0.5. This suggests that missions to this orbit type face more challenges or have higher failure rates.
3. **Moderate Success Rates**: The orbit types ISS, LEO, MEO, PO, SO, and VLEO have moderate success rates, ranging between approximately 0.6 and 0.8. This shows a varied level of success for missions targeting these orbits.



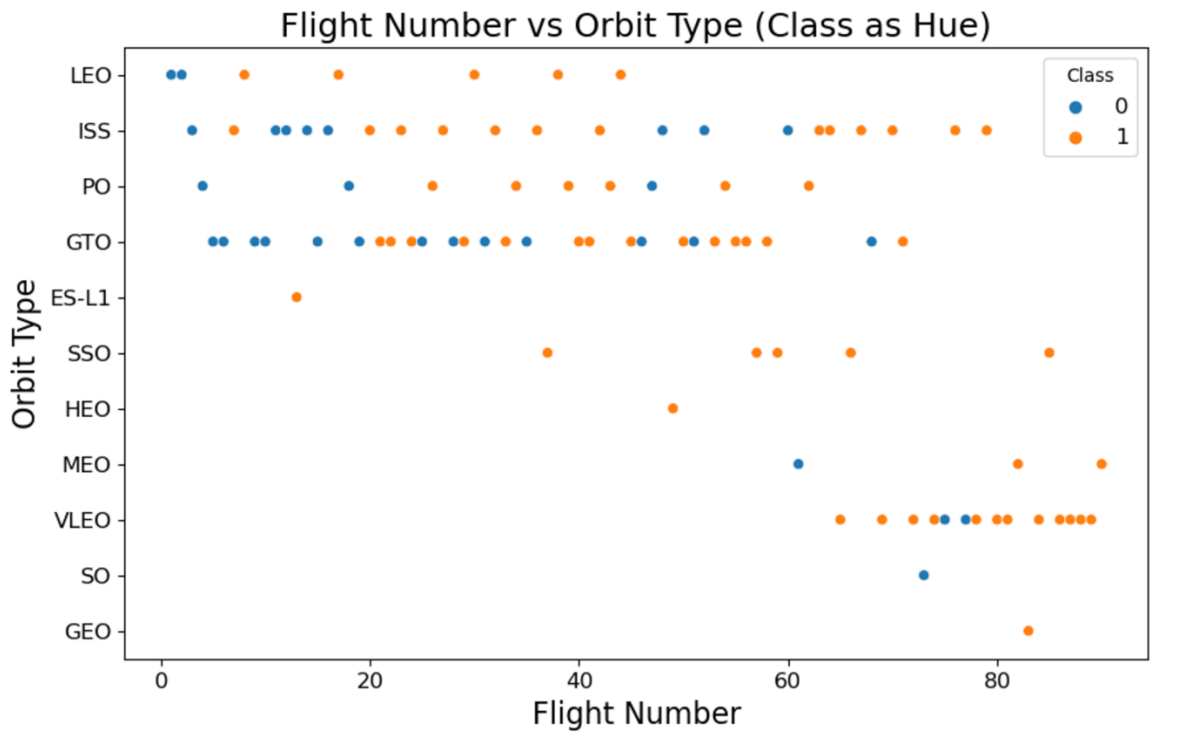
Flight Number vs. Orbit Type

Distribution of Flights by Orbit Type: Most flights are concentrated in LEO, ISS, PO, and GTO orbits.

Relationship Between Class and Orbit Type: Class 1 (orange) is more common in LEO, ISS, and GTO orbits, while Class 0 (blue) also appears in these orbits but in smaller quantities.

Trends Over Time: As the number of flights increases, the distribution between classes and orbit types may show specific trends or patterns.

Less Common Orbits: Orbits like ES-L1, SSO, HEO, MEO, VLEO, SO, and GEO have fewer flights, indicating they are less common or less utilized.



Payload vs. Orbit Type

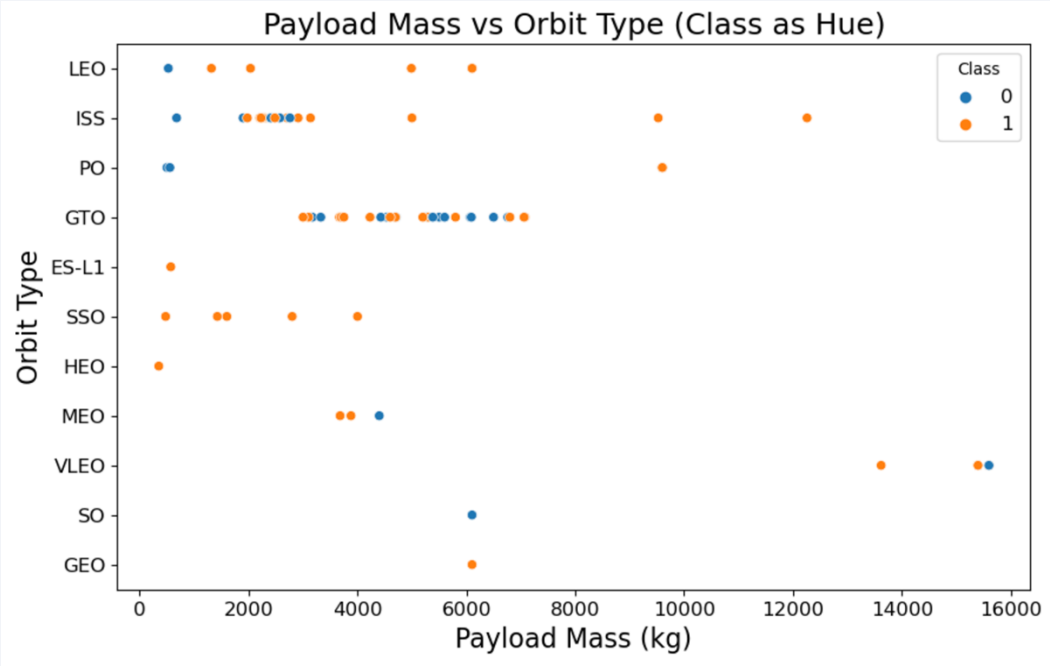
Payload Mass Distribution by Orbit Type: Most payloads for LEO and ISS orbits have smaller masses, while payloads for GTO and GEO orbits tend to have larger masses.

Payload Classes: Classes 0 and 1 are relatively evenly distributed across various orbit types, but there is a notable concentration of Class 1 payloads in orbits like ES-L1 and SSO.

Specific Orbits: Some orbits, such as VLEO and SO, do not have data represented in the graph, which may indicate a lower frequency of launches or a lack of data for these specific orbits.

Maximum Payload Mass: The highest payload mass observed is approximately 16000 kg, associated with GEO and GTO orbits.

Data Concentration: There is a high concentration of data around smaller payload masses (0 to 6000 kg) for most orbit types.



Launch Success Yearly Trend

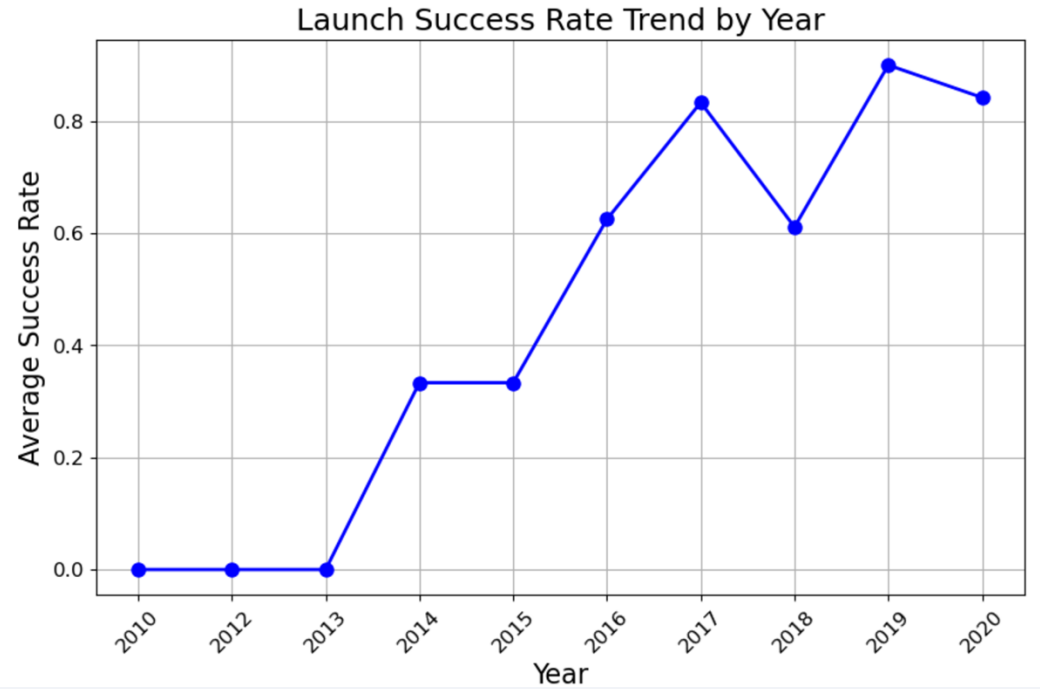
Slow Initial Growth (2010-2013): From 2010 to 2013, the average success rate remained at 0, indicating no successful launches or unrecorded data.

Significant Increase (2014-2015): In 2014, the average success rate rose to about 0.4 and continued to increase in 2015, reaching approximately 0.6.

Continuous Growth (2016-2017): The average success rate continued to rise, reaching around 0.8 in 2016 and staying high in 2017.

Drop and Recovery (2018-2019): In 2018, there was a significant drop in the average success rate, falling back to around 0.4. However, in 2019, the average success rate recovered to about 0.8.

High Stability (2020): In 2020, the average success rate remained high, slightly below 0.8.



All Launch Site Names

- This query selects all unique values from the Launch_Site column in the SPACEXTABLE table.
- Using the DISTINCT function is essential in data analysis to ensure that you are working with unique entries, which can help in identifying patterns and making accurate predictions.

```
%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'

- To find 5 records where launch sites begin with CCA, you can use the LIKE operator in SQL.
- Using the LIKE operator is essential in data analysis to filter records based on specific patterns, which can help in identifying relevant data points for further analysis.

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

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

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- This query selects the sum of the Payload_Mass__kg_ column from the SPACEXTABLE table where the Customer is 'NASA (CRS)'. The result of this query is the total payload mass carried by boosters from NASA.

- Using the SUM function is essential in data analysis to aggregate values, which can help in understanding the total impact or contribution of specific data points.

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

```
%sql select sum(PAYLOAD_MASS__KG_), Customer from SPACEXTABLE WHERE Customer LIKE "NASA (CRS)" GROUP BY Booster_Version
```

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

sum(PAYLOAD_MASS__KG_)	Customer
2647	NASA (CRS)
3310	NASA (CRS)
2697	NASA (CRS)
2268	NASA (CRS)
2972	NASA (CRS)
1977	NASA (CRS)
2500	NASA (CRS)
2495	NASA (CRS)
2205	NASA (CRS)
3136	NASA (CRS)
2257	NASA (CRS)

Average Payload Mass by F9 v1.1

- This function calculates the average value of a specified column.

- Using the AVG function to determine the mean value of a dataset, which can help in understanding the typical performance and capabilities of specific booster versions.

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

```
%sql select avg(PAYLOAD_MASS__KG_), Booster_Version from SPACEXTABLE WHERE Booster_Version LIKE 'F9 v1.1'
```

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

avg(PAYLOAD_MASS__KG_)	Booster_Version
2928.4	F9 v1.1

First Successful Ground Landing Date

- This query selects the minimum date (MIN(Date)) from the SPACEXTABLE table where the Landing_Outcome is 'Success (ground pad)'. The result of this query will be the date of the first successful landing on a ground pad.
- Using the MIN function to identify the earliest occurrence of a specific event, which can help in understanding historical trends and milestones.

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

```
%sql select min(date) from SPACEXTABLE where Landing_Outcome LIKE "Success (ground pad)"
```

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

min(date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- This query selects the Booster_Version column from the SPACEXTABLE table where the Landing_Outcome is 'Success (drone ship)' and the Payload_Mass__kg_ is between 4000 and 6000 kilograms.
- Using this query helps in identifying specific boosters that have met the criteria of successful drone ship landings with a payload mass within the specified range, which can be useful for analyzing the performance and reliability of these boosters.

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 SPACEXTABLE WHERE Landing_Outcome LIKE 'Success (drone ship)' AND PAYLOAD
```

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

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Total Number of Successful and Failure Mission Outcomes

This query counts the number of records for each unique value in the Mission_Outcome column from the SPACEXTABLE table and groups the results by the Mission_Outcome values.

To calculate the total number of successful and failure mission outcomes:

- Total number of successful mission outcomes: 98 (Success) + 1 (Success) + 1 (Success with payload status unclear) = 100
- Total number of failure mission outcomes: 1 (Failure in flight)

Therefore, the total number of successful mission outcomes is 100, and the total number of failure mission outcomes is 1.

List the total number of successful and failure mission outcomes

```
%sql select Count(Mission_Outcome), Mission_Outcome from SPACEXTABLE Group by Mission_Outcome
```

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

Count(Mission_Outcome)	Mission_Outcome
1	Failure (in flight)
98	Success
1	Success
1	Success (payload status unclear)

Boosters Carried Maximum Payload

- This query selects the Booster_Version column from the SPACEXTABLE table where the Payload_Mass__kg_ is equal to the maximum payload mass found in the table. The subquery (SELECT MAX(Payload_Mass__kg_) FROM SPACEXTABLE) finds the maximum payload mass, and the main query retrieves the booster versions that have carried this maximum payload mass.
- Using this query helps in identifying the boosters that have carried the heaviest payloads, which can be useful for analyzing the performance and capabilities of these boosters.

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

- This query retrieves records from the SPACEXTABLE table where the Landing_Outcome includes the word "failure" and the year is 2015. It selects the month, landing outcome, booster version, and launch site.

- This query helps in identifying the specific failed landing outcomes on drone ships, along with their booster versions and launch site names for the year 2015. This information can be useful for analyzing the performance and challenges faced during that period.

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 strftime('%m', Date) AS month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTABLE WHERE Landing_Outcome LIKE '%failure
```

```
 * 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 selects the Landing_Outcome column and counts the number of occurrences for each outcome within the specified date range. The results are then grouped by the Landing_Outcome and ordered by the count in descending order.
- This ranking shows the frequency of different landing outcomes for space missions within the specified date range, highlighting that "No attempt" was the most common outcome, followed by equal counts of "Success (drone ship)" and "Failure (drone ship)".

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.

```
%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

Global map All Launch Sites

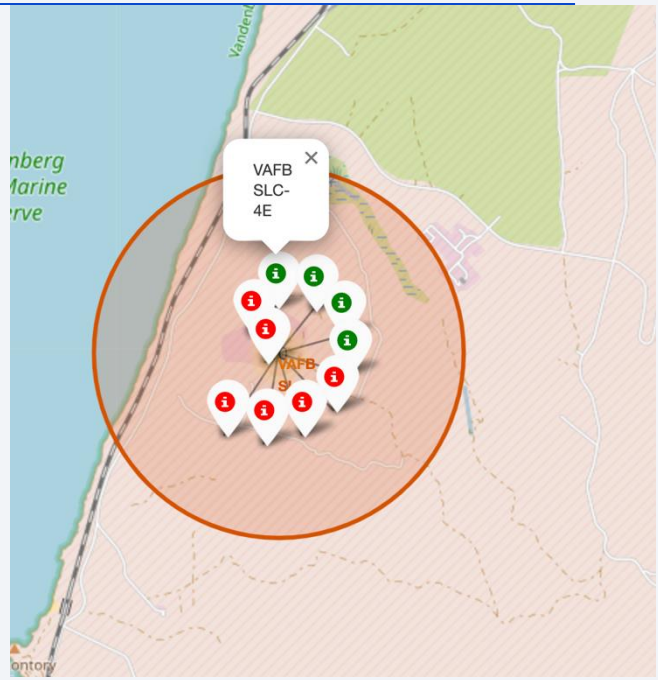
- All lauch sites are located in U.S.A specifically on the coast (California and Florida)

Color-Labeled Launch Outcomes

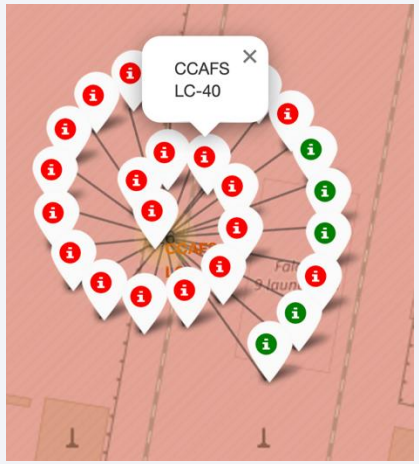
Marker's icon property to indicate if this launch was successed or failed

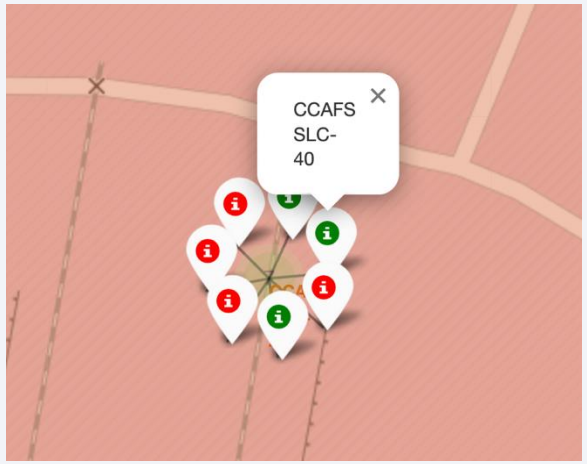
- Green indicates the launches are successed
- Red indicates the launches failed

California



Florida

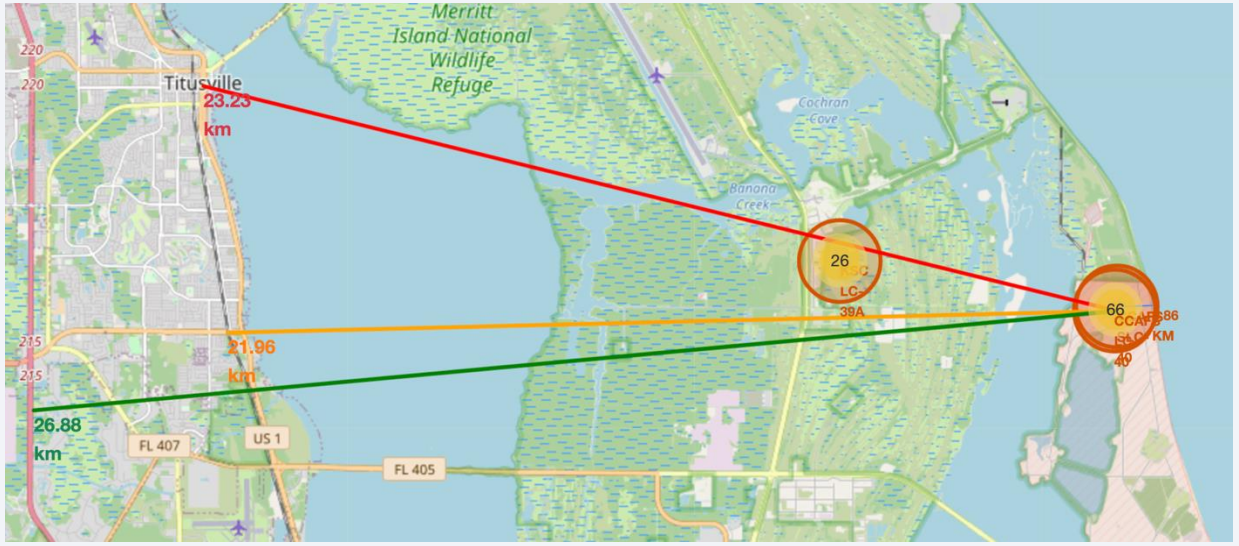


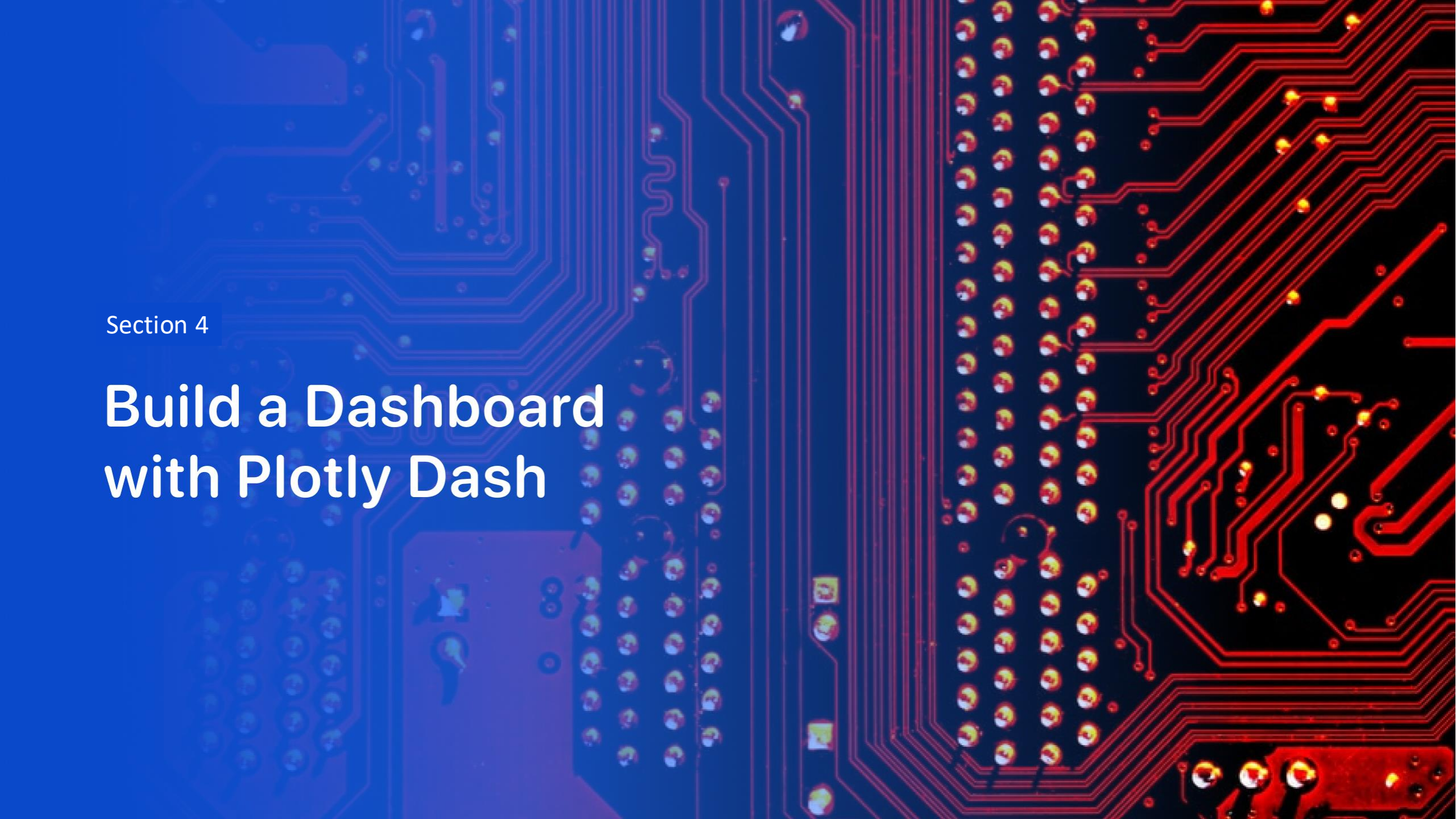


Launch Sites to Its Proximities

Map shows the distance to its proximities

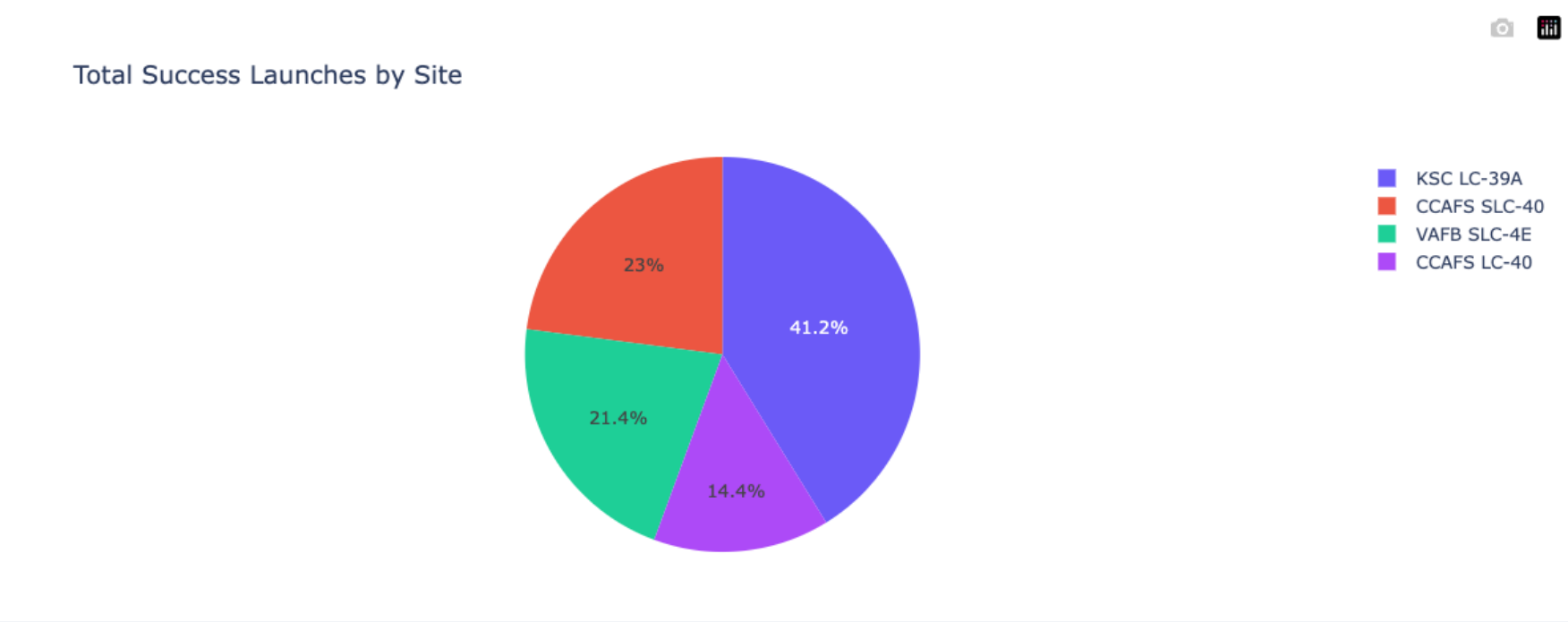
- Cities
- Railways
- Highways
- Coast




Section 4
Build a Dashboard with Plotly Dash

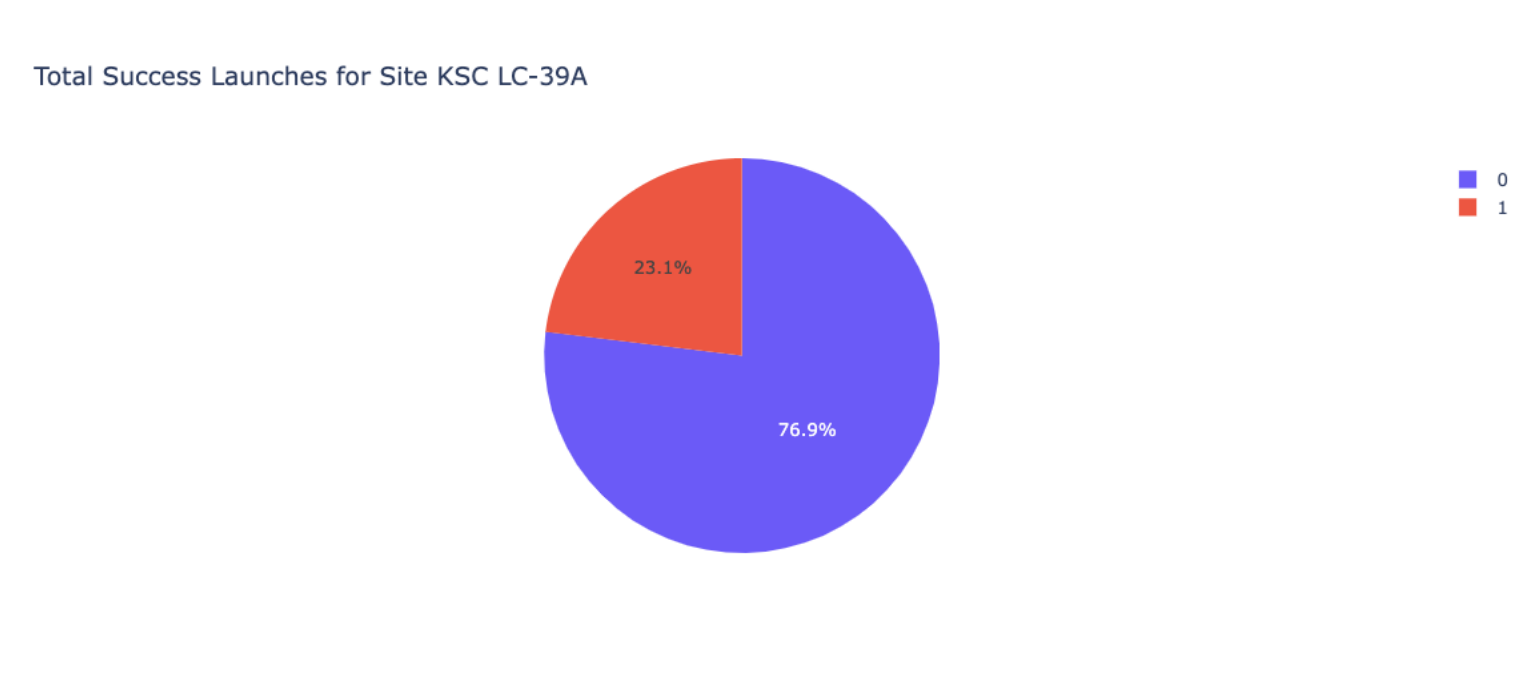
Total Success Launches by Site

- **KSC LC-39A**: This site accounts for the largest portion of successful launches, with **41.2%** of the total. This indicates that nearly half of all successful launches occurred at this site.
- **CCAFS SLC-40**: This site contributes **23%** of the total successful launches, making it the second most successful launch site.
- **VAFB SLC-4E**: This site has a share of **21.4%** of the total successful launches, closely following CCAFS SLC-40.
- **CCAFS LC-40**: This site accounts for **14.4%** of the total successful launches, making it the least successful among the four sites represented in the chart.



Highest Launch Success Ratio

- **Purple Segment (76.9%)**: This segment represents the majority of the successful launches at the KSC LC-39A site. It indicates that 76.9% of the total launches at this site were successful.
- **Red Segment (23.1%)**: This segment represents the remaining successful launches at the KSC LC-39A site. It indicates that 23.1% of the total launches at this site were successful.



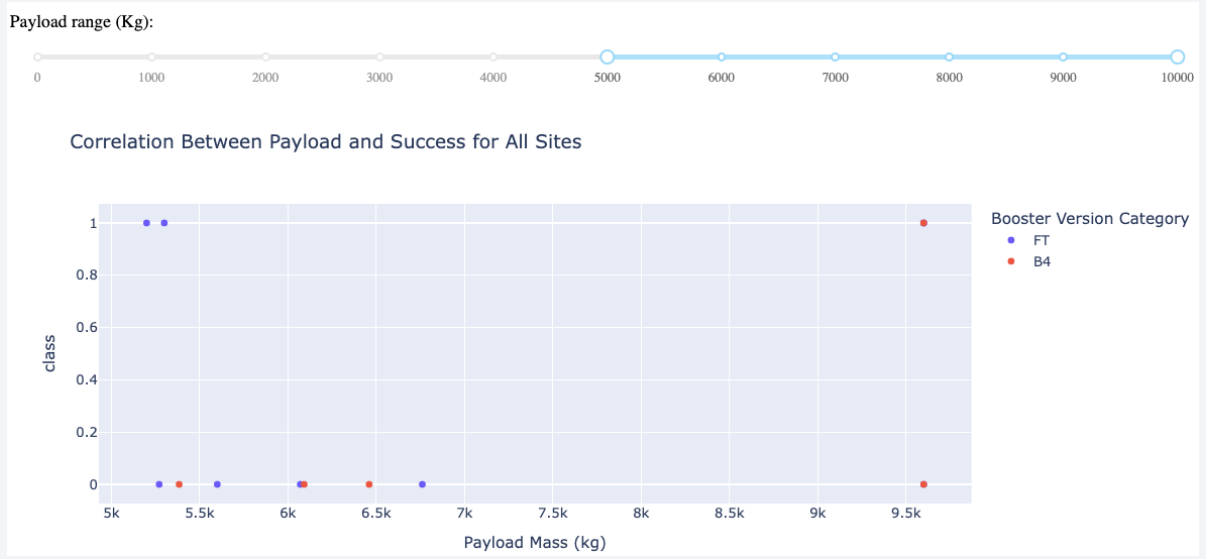
Payload vs. Launch Outcome

1. The x-axis represents the payload mass (kg), ranging from 0 to 5000 kg, while the y-axis represents the class, where 1 indicates a successful launch and 0 indicates a failure.

2. The x-axis represents the payload mass (kg), ranging from 5000 to 10000 kg, while the y-axis represents the class, where 1 indicates a successful launch and 0 indicates a failure.

1.

2.

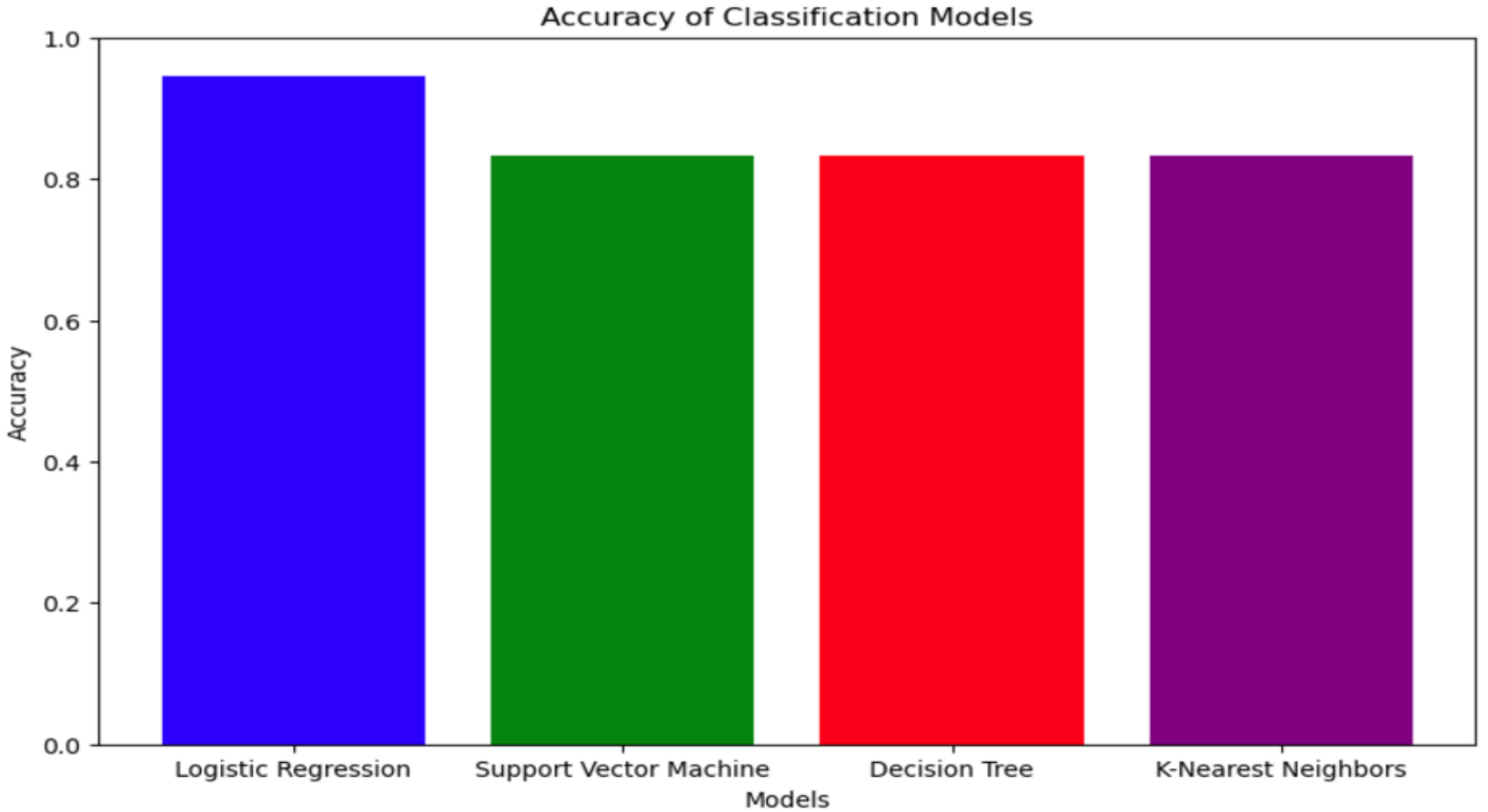


Section 5

Predictive Analysis (Classification)

Classification Accuracy

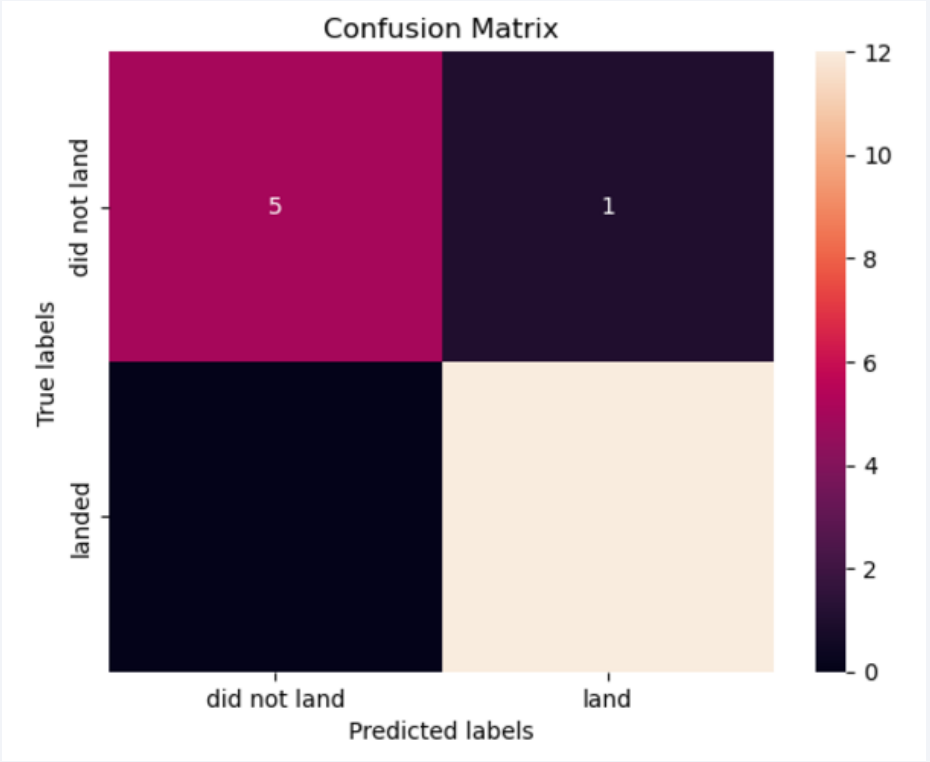
```
Logistic Regression Accuracy: 0.9444444444444444
Support Vector Machine Accuracy: 0.8333333333333334
Decision Tree Accuracy: 0.8333333333333334
K-Nearest Neighbors Accuracy: 0.8333333333333334
The best model is: Logistic Regression with an accuracy of 0.9444
```



Confusion Matrix

The confusion matrix is a 2x2 grid that compares the predicted labels against the true labels. The labels are "did not land" and "land."

- **True Negatives (Top-Left Cell: 5)**: Instances where the model correctly predicted "did not land."
- **False Positives (Top-Right Cell: 1)**: Instances where the model incorrectly predicted "land" when it should have predicted "did not land."
- **False Negatives (Bottom-Left Cell: 0)**: Instances where the model incorrectly predicted "did not land" when it should have predicted "land."
- **True Positives (Bottom-Right Cell: 6)**: Instances where the model correctly predicted "land."



Conclusions

1. KSC LC-39A has the highest Launch Success Ratio
2. Logestic Regression model has hightest accuracy comperting to others
3. Map show a safety distance from cities to launch sites
4. Launch sites are all located in USA and close to the coast
5. Less payload shows more booster version diversity

Appendix

- Python code, SQL queries, charts, Notebook outputs, or data sets can be found on project's page on my Github (https://github.com/Diegomoro415/spaceY/tree/main)

Thank you!