IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sandor Vas
2024-08-29

Outline

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

Executive Summary

Summary of Methodologies

- Data Collection and Wrangling:
  - Data Collection: Historical flight data including features like Flight Number, Date, Booster Version, Payload Mass, and Outcome.
  - Data Wrangling:
    - Cleaning: Addressed missing values and applied encoding techniques.
    - Scaling: Standardized features using StandardScaler.
    - Transformation: Converted date features to numeric formats.
- Exploratory Data Analysis (EDA):
  - Visual Analytics: Utilized tools like Matplotlib, Seaborn, and Plotly for effective data visualization.
  - Interactive Maps/Dashboards: Developed using Folium and Plotly Dash for geographic and interactive analysis.
- Predictive Modeling:
  - Model Selection: Employed models including Logistic Regression, Support Vector Machines (SVM), Decision Tree Classifier, and K-Nearest Neighbors (KNN).
  - Hyperparameter Tuning: Optimized model parameters using GridSearchCV.
  - Model Evaluation: Assessed model performance through cross-validation and accuracy metrics.

Introduction

Project Background and Context

- Background:
  - Space missions are complex operations with various factors influencing their success.
  - Historical data from space missions provides valuable insights into these factors and helps in predicting the outcomes of future missions.
  - This project leverages historical flight data to understand and predict the success of space missions.
- Context:
  - The dataset includes records of past space missions with features such as flight number, date, booster version, payload mass, and mission outcome.
  - Analyzing this data allows us to identify patterns and factors that contribute to the success or failure of missions.

Problems to Address

- Data Challenges:
  - Missing Values: How to handle incomplete data and ensure data integrity.
  - Categorical Data: How to encode non-numeric features (e.g., booster version, outcome).
  - Feature Scaling: How to standardize numerical features for model training.
- Exploratory Data Analysis (EDA):
  - Patterns and Trends: What trends or patterns are visible in the data? How do different features relate to the mission outcome?
  - Visualizations: How can visualizations help in understanding the data better?
- Predictive Modeling:
  - Model Selection: Which machine learning models (e.g., Logistic Regression, SVM, Decision Tree, KNN) are best suited for predicting mission success?
  - Hyperparameter Tuning: How can we optimize model parameters for better performance?
  - Evaluation Metrics: What metrics should be used to evaluate the performance of predictive models?

Section 1

Methodology

Methodology

1. Data Collection and Wrangling

- Data Collection:
  - Source: Historical flight data including features such as Flight Number, Date, Booster Version, Payload Mass, and Outcome.
  - Format: CSV files.
- Data Wrangling:
  - Cleaning: Handling missing values, correcting data inconsistencies, and filtering out irrelevant records.
  - Encoding: Converting categorical variables (e.g., Booster Version, Outcome) into numeric formats using techniques such as one-hot encoding.
  - Feature Scaling: Standardizing numerical features using StandardScaler to ensure uniformity in model training.
  - Transformation: Converting date features into numerical values for analysis.

2. Exploratory Data Analysis (EDA)

- Visual Analytics:
  - Tools Used: Matplotlib, Seaborn, Plotly.
  - Techniques: Generating visualizations such as histograms, scatter plots, and heatmaps to uncover patterns and trends.
- Interactive Maps and Dashboards:
  - Tools Used: Folium for interactive maps, Plotly Dash for interactive dashboards.
  - Purpose: To provide interactive and insightful visual representations of data.

Methodology (continued)

3. Predictive Modeling

- Model Selection:
  - Algorithms Used: Logistic Regression, Support Vector Machine (SVM), Decision Tree Classifier, K-Nearest Neighbors (KNN).
- Hyperparameter Tuning:
  - Technique: GridSearchCV for finding the best hyperparameters for each model.
  - Parameters Optimized: Specific to each algorithm (e.g., regularization strength for Logistic Regression, kernel type for SVM).
- Model Evaluation:
  - Method: Cross-validation to assess model performance and avoid overfitting.
  - Metrics Used: Accuracy, confusion matrix, and other relevant evaluation metrics to determine model effectiveness.

4. Tools and Libraries

- Programming Languages: Python.
- Libraries: pandas, numpy, scikit-learn, matplotlib, seaborn, plotly, folium.

Data Collection

Historical Flight Data
Data Format: CSV

Download CSV files
Extract relevant columns: Flight Number, Date, Booster Version, Payload Mass, Outcome

Source Method: Public Repository / Internal Database / API

Check Structure: Verify data completeness
Identify Issues: Missing values, inconsistencies

Load Data: Into analysis tools
Initial Cleaning: Handle missing values, check data types

Data Collection – SpaceX API

There are four python functions defined to support the data collection from the following end points:

https://api.spacexdata.com/v4/rockets/

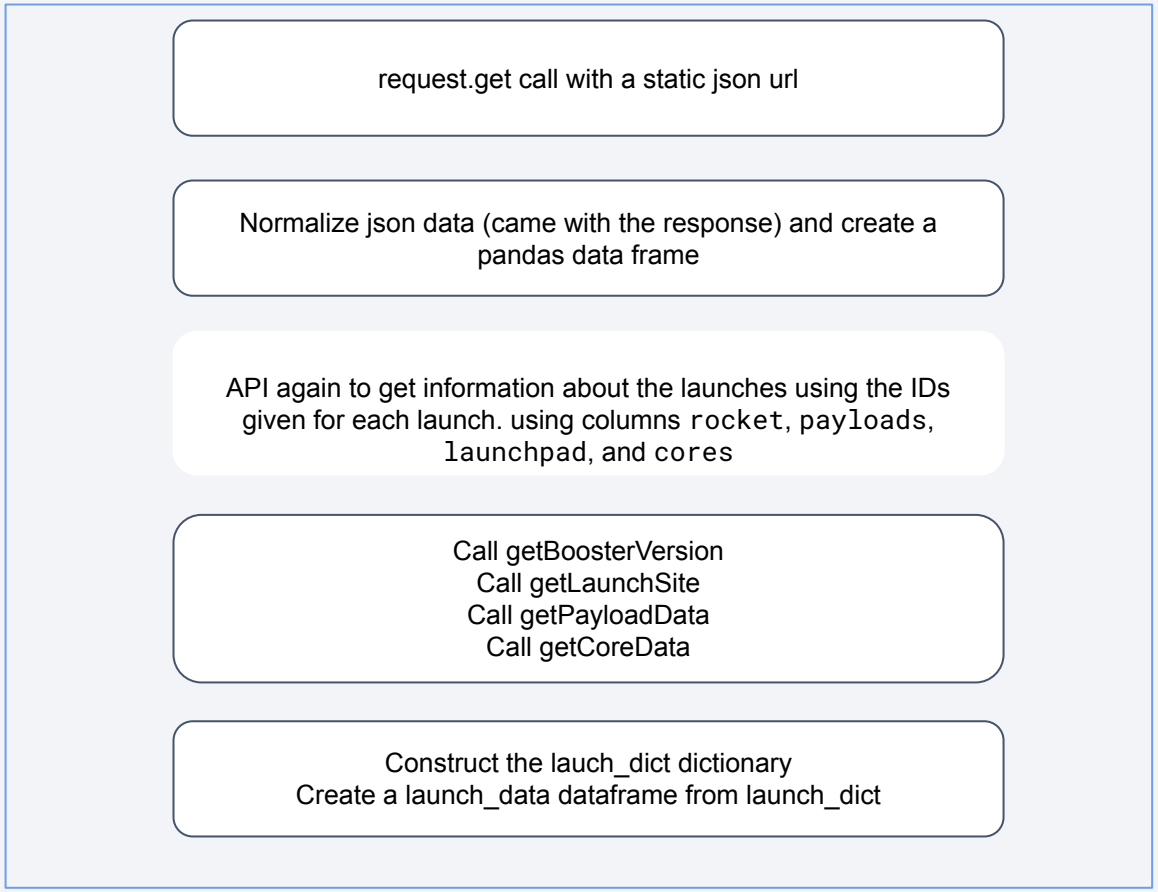
https://api.spacexdata.com/v4/launchpads

https://api.spacexdata.com/v4/payloads

https://api.spacexdata.com/v4/cores/

Function Overview:

- getBoosterVersion(data):
  - Purpose: Extracts names of rocket boosters from the data.
  - Process: Iterates through the rocket information, fetches details, and stores booster names.
- getLaunchSite(data):
  - Purpose: Extracts details about launch sites.
  - Process: Iterates through launch pad information, fetches details, and stores location and name.
- getPayloadData(data):
  - Purpose: Extracts information about payloads.
  - Process: Iterates through payload information, fetches details, and stores mass and orbit data.
- getCoreData(data):
  - Purpose: Extracts details about rocket cores.
  - Process: Iterates through core information, fetches details, and stores various attributes such as block number and reuse count.

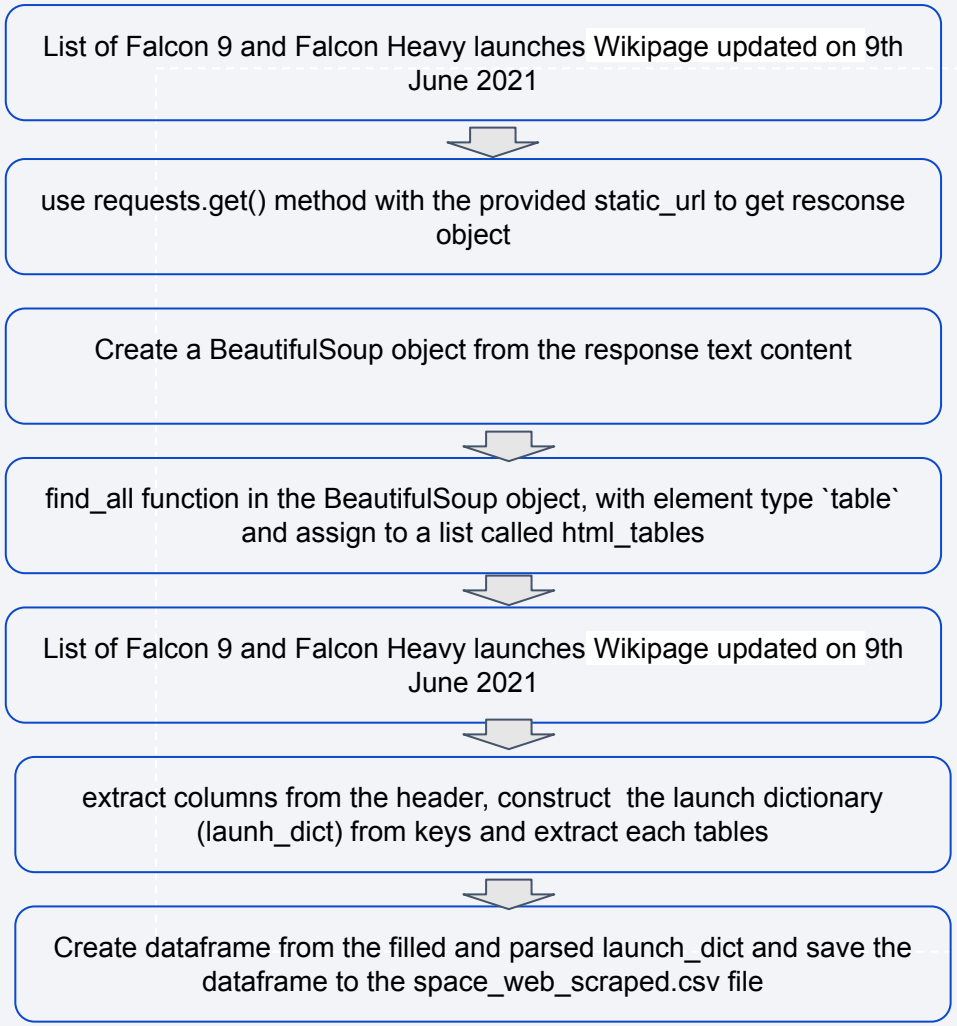


Data Collection - Scraping

Web scraping Falcon 9 and Falcon Heavy Launches Records from Wikipedia

Web scrap Falcon 9 launch records with BeautifulSoup:

- Extract a Falcon 9 launch records HTML table from Wikipedia
- Parse the table and convert it into a Pandas dataframe
- Process web scraped HTML table with some predefined helper functions
- perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.
- find all tables on the wiki page
- Iterate through the <th> elements and apply the provided extract_column_from_header()
- Create a data frame by parsing the launch HTML tables
- Create the dataframe based on the completed launch_ dict dictionary
- Save the dataframe to spacex_web_scraped.csv



Data Wrangling

Data wrangling was performed to clean, transform, and prepare the raw data for analysis. This ensured the data was accurate, complete, and suitable for modeling.

Data Cleaning:

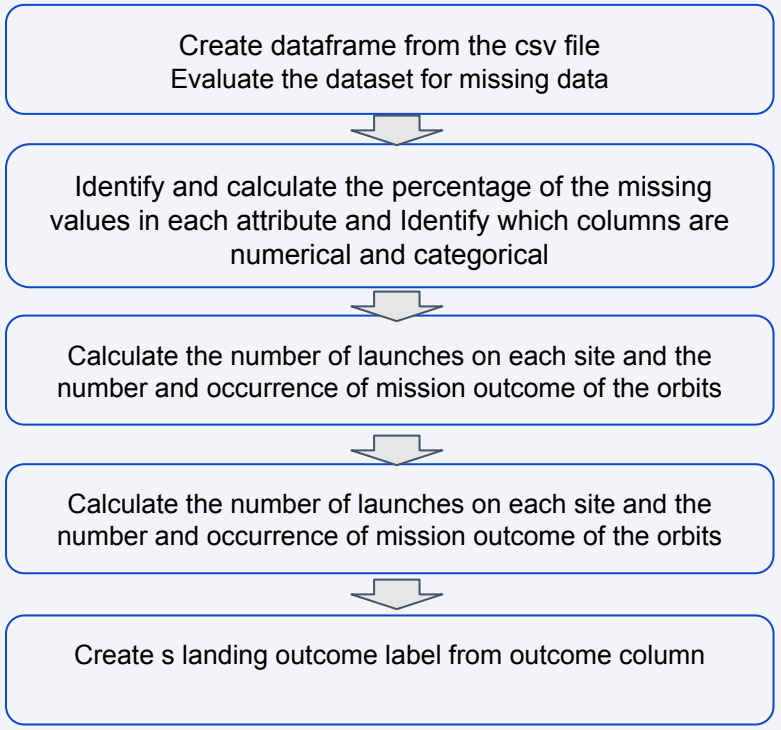
- Handling Missing Values: Missing values were handled by imputation with mean values for numerical features and mode values for categorical features.
- Removing Duplicates: Duplicate records were identified and removed to maintain data quality.
- Correcting Errors: Errors such as incorrect data types were corrected, and outliers were addressed through filtering.

Feature Engineering:

- Creating New Features: Extracted the year from date columns and created logarithmic transformations for mass.
- Encoding Categorical Variables: Converted categorical variables into numerical values using one-hot encoding.
- Scaling Features: Standardized numerical features to ensure they were on a common scale.

Data Transformation:

- Data Aggregation: Aggregated data by summing values to create summary statistics.
- Date/Time Transformation: Transformed date/time features into numerical components.



EDA with Data Visualization (1)

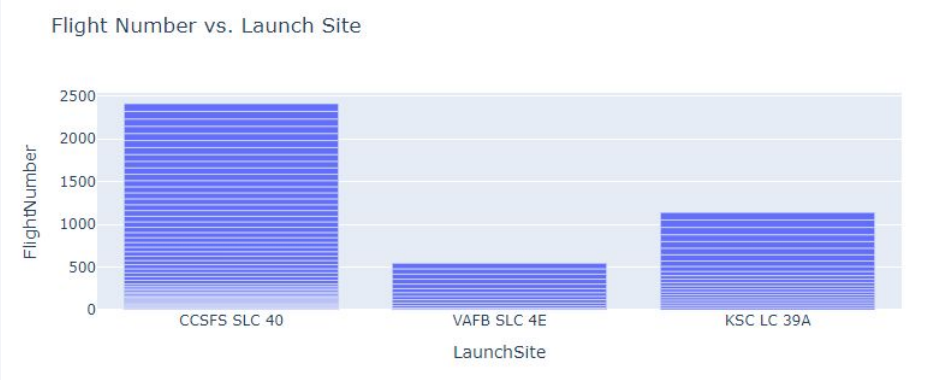
1. Flight Number vs. Launch Site

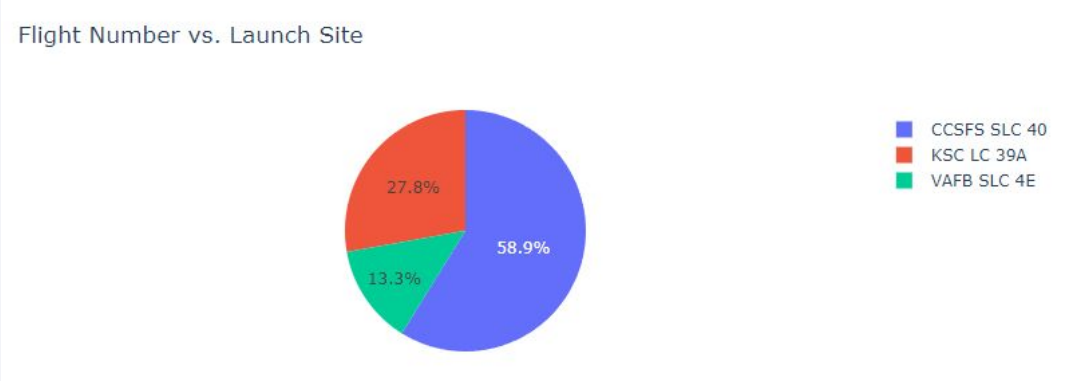
Chart Type: Bar Chart and pie chart

Purpose:

To visualize the frequency of launches per launch site

This helps identify which launch sites are most frequently used and how the number of launches varies by site.





EDA with Data Visualization (2)

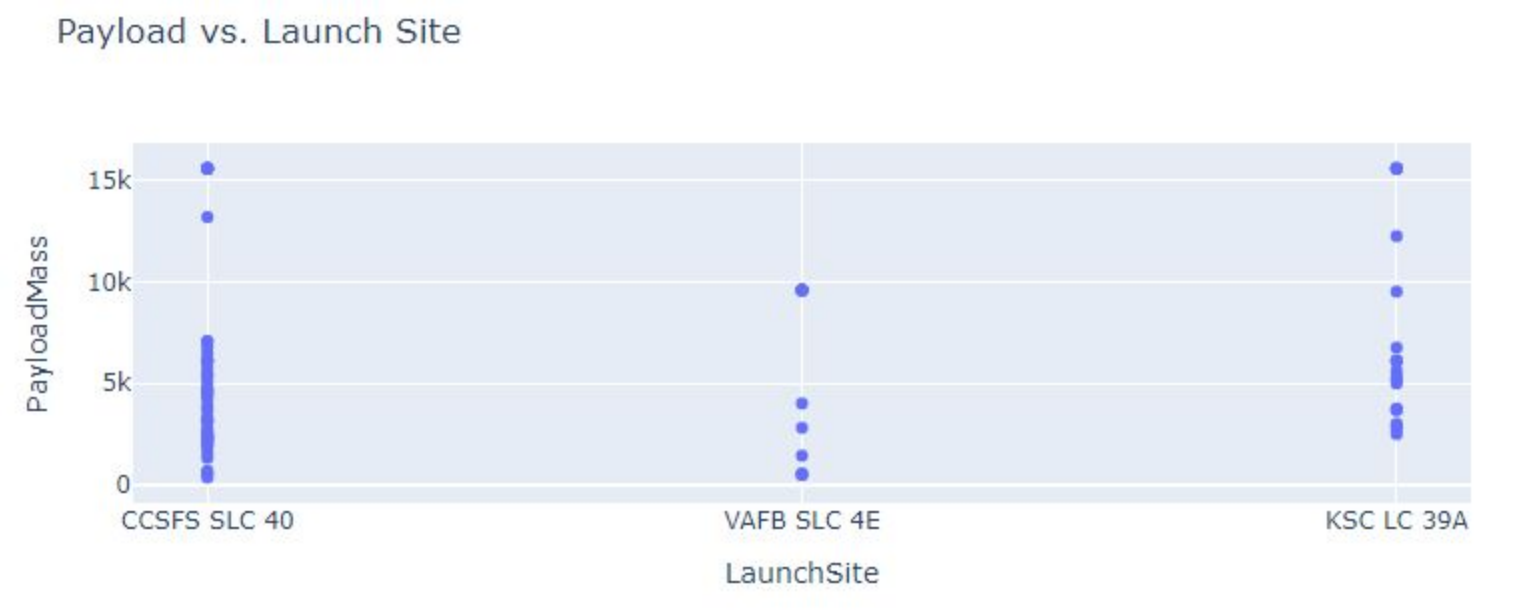
2. Payload vs. Launch Site

Chart Type: Scatter Plot

Purpose:

To analyze the relationship between payload mass and launch site.

This helps in understanding if there is any correlation between the payload mass and the launch site used.



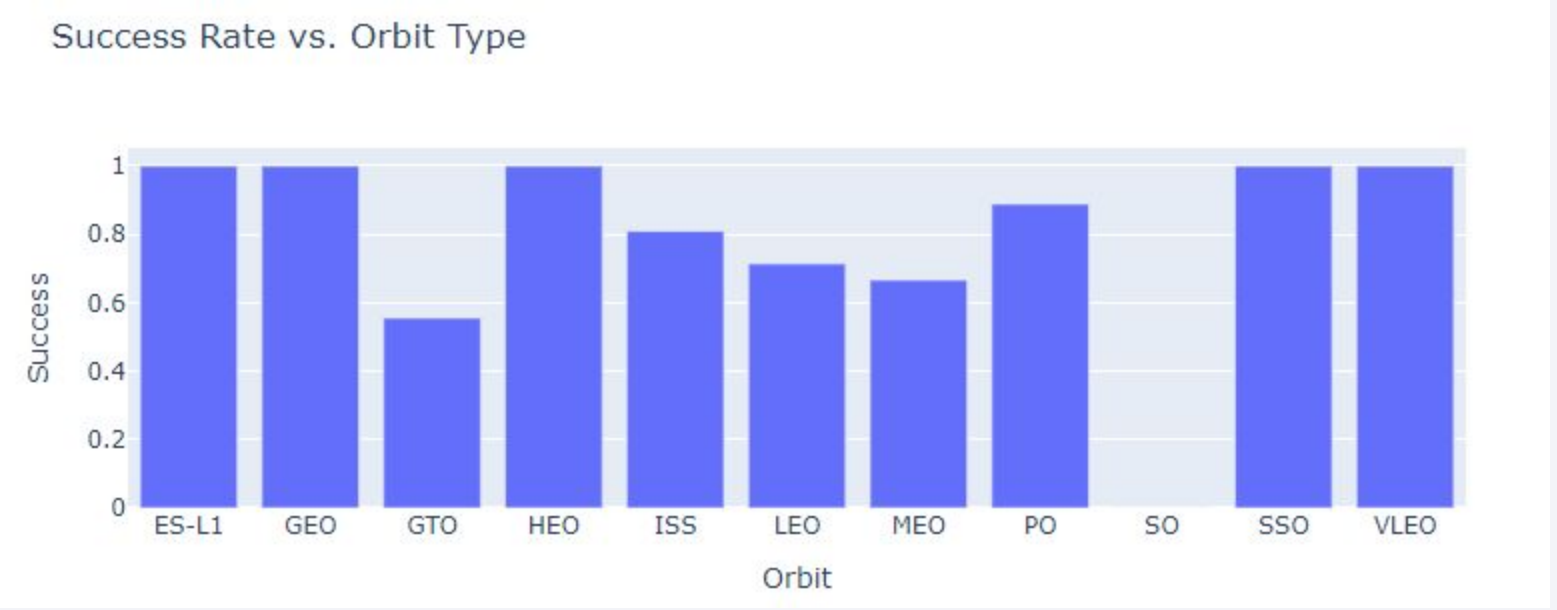
EDA with Data Visualization (3)

3. Success Rate vs. Orbit Type

Chart Type: Bar Chart

Purpose:

To show the success rate of launches for each orbit type.
This allows you to evaluate which orbits are associated with higher or lower success rates.



EDA with Data Visualization (4)

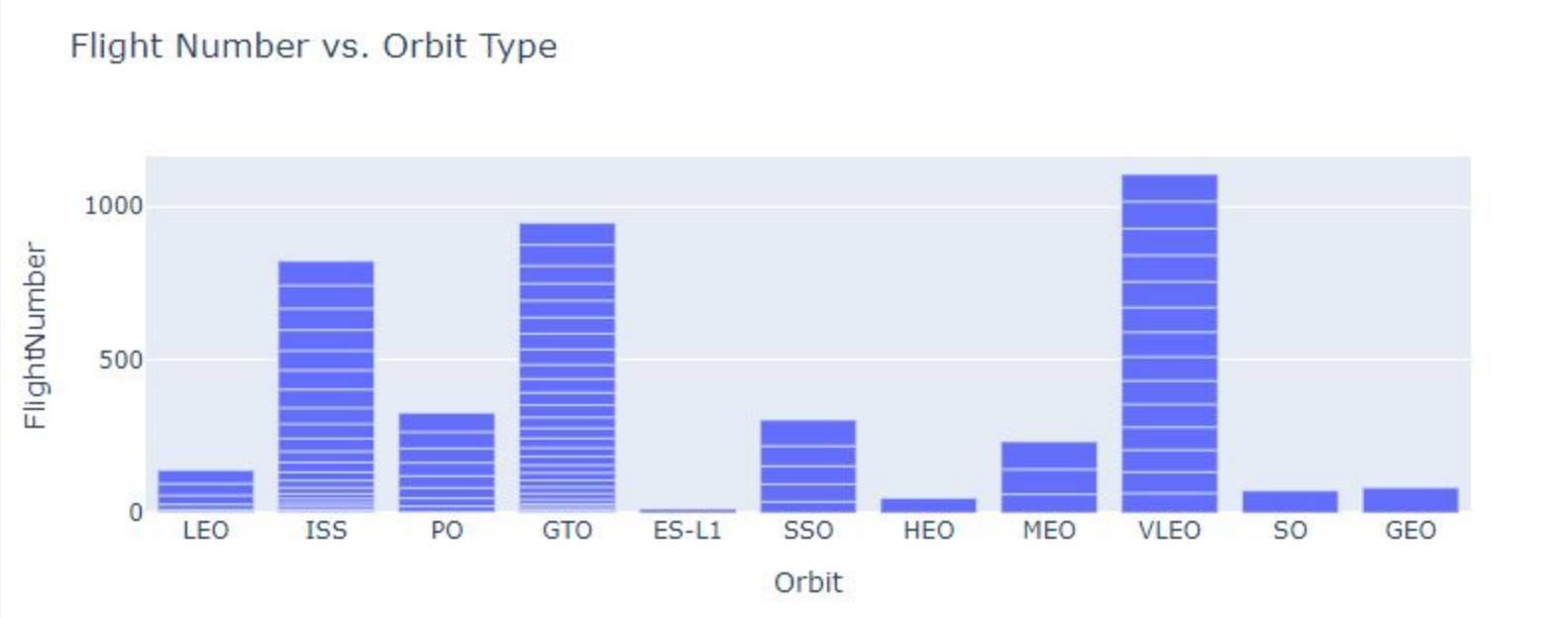
4. Flight Number vs. Orbit Type

Chart Type: Bar Chart

Purpose:

To see how the number of flights varies across different orbit types.

This helps in understanding the distribution of different orbits used over time.



EDA with Data Visualization (5)

5. Payload vs. Orbit Type

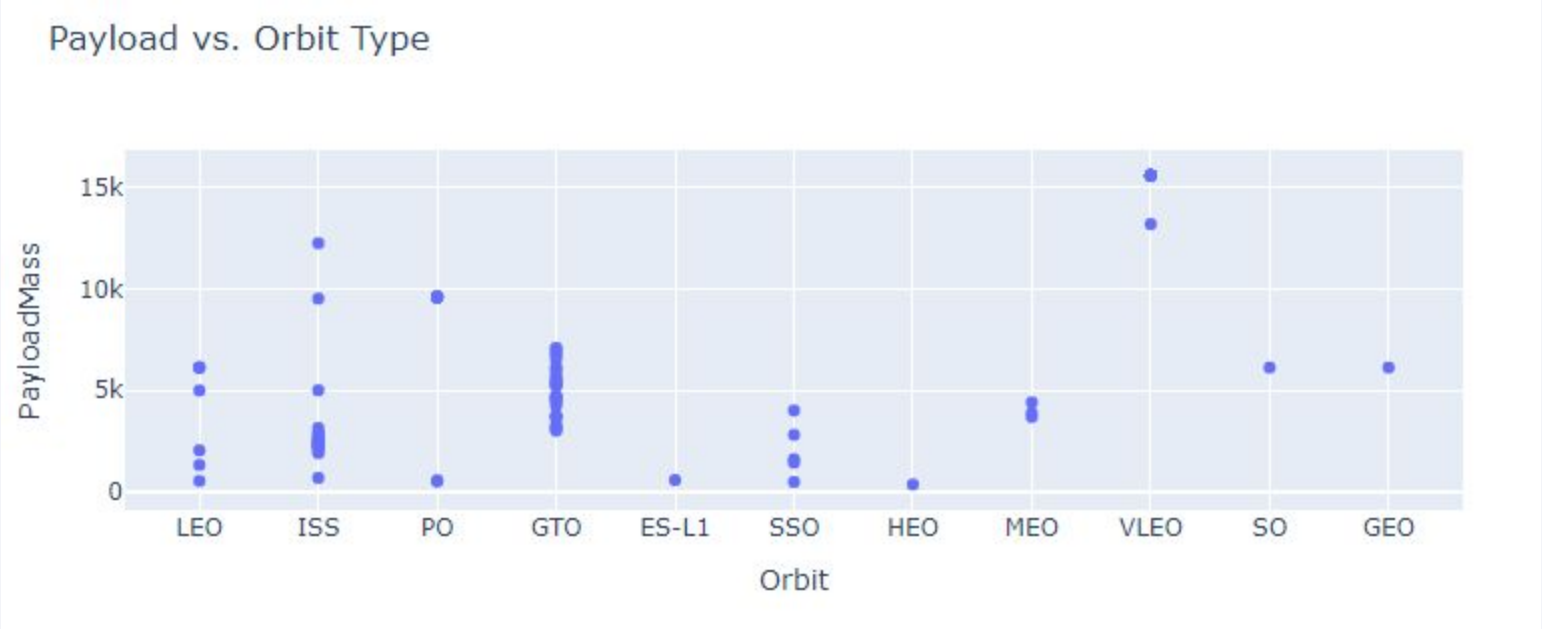
Chart Type: Scatter Plot

Purpose:

To explore how payload masses are distributed across different orbit types.

This can reveal trends or outliers in payload sizes based on the orbit type.

.



EDA with Data Visualization (6)

6. Launch Success Yearly Trend

Chart Type: Line Chart

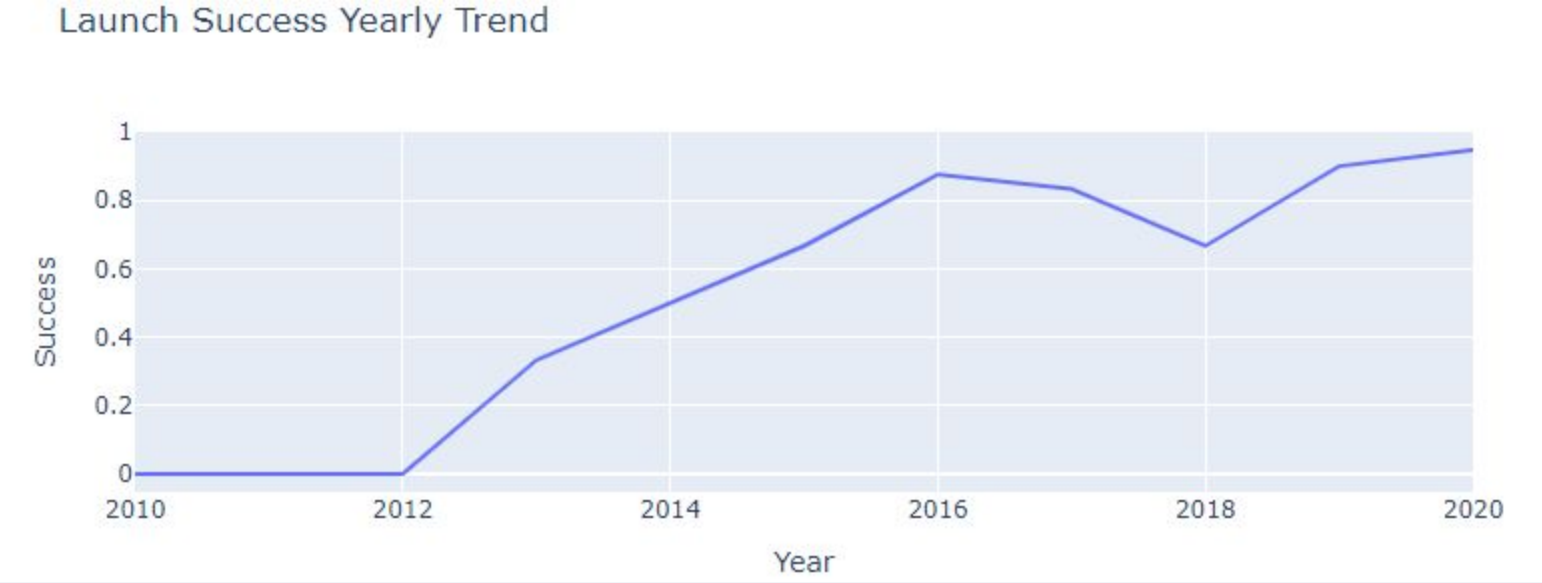
Purpose:

To track the trend of successful launches over the years.

This helps in visualizing how the success rate of launches has evolved over time.

.

.



Build an Interactive Map with Folium

1. Markers:

- What: Used `folium.Marker` with a `DivIcon` to add labels at specific launch site locations.
- Why: Markers help identify the exact location of each launch site on the map. The `DivIcon` allows custom labels to provide clear information about the site's name, making it easier for users to recognize and differentiate between the launch sites.

2. Circles:

- What: Used `folium.Circle` to add circular areas around each launch site.
- Why: Circles are used to visually emphasize the area around each launch site. By adjusting the radius and color, these circles provide a clear visual representation of the proximity around each site. The white fill with colored borders ensures that the circles are visible against various map backgrounds without obscuring other details.

Predictive Analysis (Classification)

1. Building the Classification Model:

1.1. Data Preparation:

- Data Loading: Read and preprocess the data from CSV files.
- Feature Engineering: Encode categorical variables, scale features, and split data into training and testing sets.

1.2. Model Selection:

- Models Chosen: Support Vector Machine (SVM), Classification Trees, and Logistic Regression.
- Initial Training: Train each model using the training data.

1.3. Hyperparameter Tuning:

- Grid Search: Perform hyperparameter tuning for each model to find the optimal parameters.
- Cross-Validation: Use cross-validation to ensure robustness and prevent overfitting.

2. Evaluating the Models:

2.1. Performance Metrics:

- Accuracy: Measure the overall correctness of the model.
- Confusion Matrix: Analyze the true positives, false positives, true negatives, and false negatives.
- ROC Curve & AUC: Evaluate the model's ability to discriminate between classes.

2.2. Model Comparison:

- Compare Metrics: Evaluate and compare the performance metrics of different models.
- Select Best Model: Determine the best-performing model based on evaluation metrics.

3. Improving the Models:

3.1. Feature Selection:

- Refine Features: Identify and include relevant features to improve model performance.
- Remove Noise: Eliminate irrelevant or redundant features.

3.2. Model Tuning:

- Adjust Parameters: Fine-tune hyperparameters based on evaluation results.
- Re-train Models: Retrain models with updated parameters.

4. Finding the Best Performing Model:

4.1. Final Evaluation:

- Re-evaluate Models: Assess the improved models using the same performance metrics.
- Choose the Best Model: Select the model with the highest accuracy, best ROC AUC score, and other relevant metrics.

Build a Dashboard with Plotly Dash

1. Setup:
    - Import Libraries: Begin by importing necessary libraries like `dash`, `dash_core_components`, `dash_html_components`, and `plotly.graph_objs`.
    - Initialize App: Create an instance of the Dash app using `dash.Dash()`.
2. Layout:
    - Define Layout: Set up the layout of the dashboard using `html.Div()` to organize components such as dropdowns, graphs, and buttons.
3. Add Components:
    - Dropdown Menus: Include `dcc.Dropdown()` for selecting different categories or filters. This allows users to choose specific data points or categories to view.
    - Graphs and Plots: Add `dcc.Graph()` components to display plots such as scatter plots, bar charts, and line charts.
    - Other Interactions: Implement additional components like `dcc.Slider()` or `dcc.Checklist()` based on requirements.

2. Interactive Elements

1. Callbacks:
    - Define Callbacks: Use `@app.callback` to create functions that update components based on user interactions. For example, a callback function might update a graph based on the selected value from a dropdown menu.
    - Input and Output: Specify which components are inputs (e.g., dropdown selections) and which are outputs (e.g., updated graphs).
2. Real-Time Updates:
    - Dynamic Updates: Ensure that user selections trigger real-time updates to the dashboard. This might involve filtering data, changing plot types, or updating information displayed on the dashboard.

Results (1)

Explanation:

- Histogram of Payload Mass: Shows the distribution of payload mass, indicating how frequently payloads of various sizes are launched.
- Scatter Plot of Payload Mass vs. Launch Success: Displays the relationship between payload mass and launch success, helping to visualize if payload mass affects the outcome.
- Bar Chart of Launch Success by Site: Provides insight into which launch sites have more successful launches, helping to compare performance across sites.
- Pie Chart of Launch Outcomes: Shows the proportion of successful versus failed launches, giving an overall success rate.
- Box Plot of Payload Mass by Orbit: Illustrates the distribution of payload mass for different orbits, helping to understand payload variations across different orbital missions.

Results (2)

Interactive Map with Launch Sites: Helps in understanding the geographical distribution and proximity of launch sites.
Color-Coded Outcomes: Indicates the success/failure of launches directly on the map.

Proximity Map: Shows the closeness of launch sites to significant features like railways and highways, which can impact logistics and accessibility.

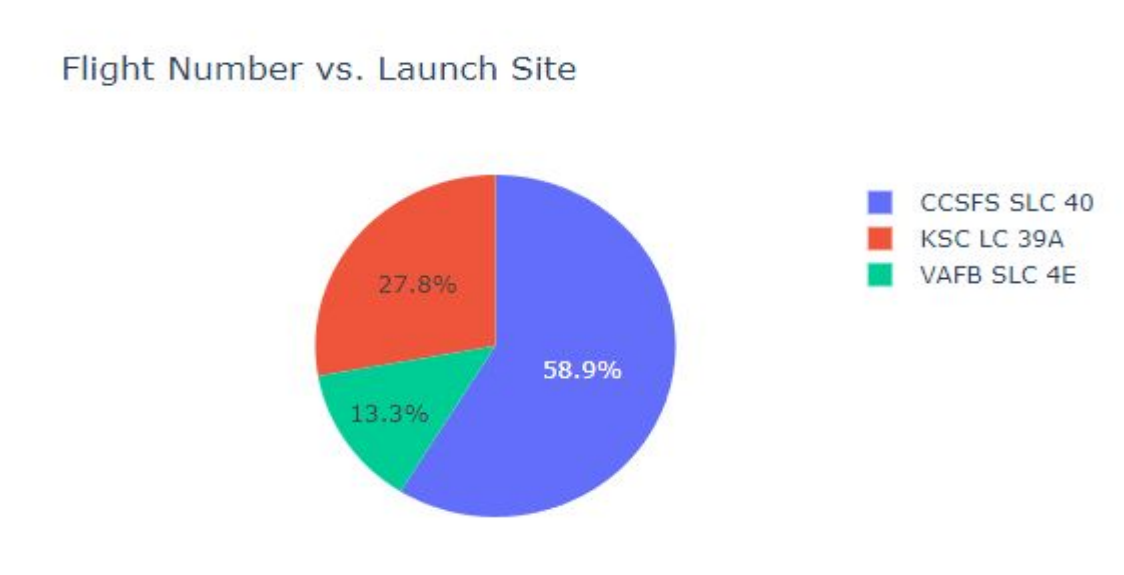
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

Explanation:

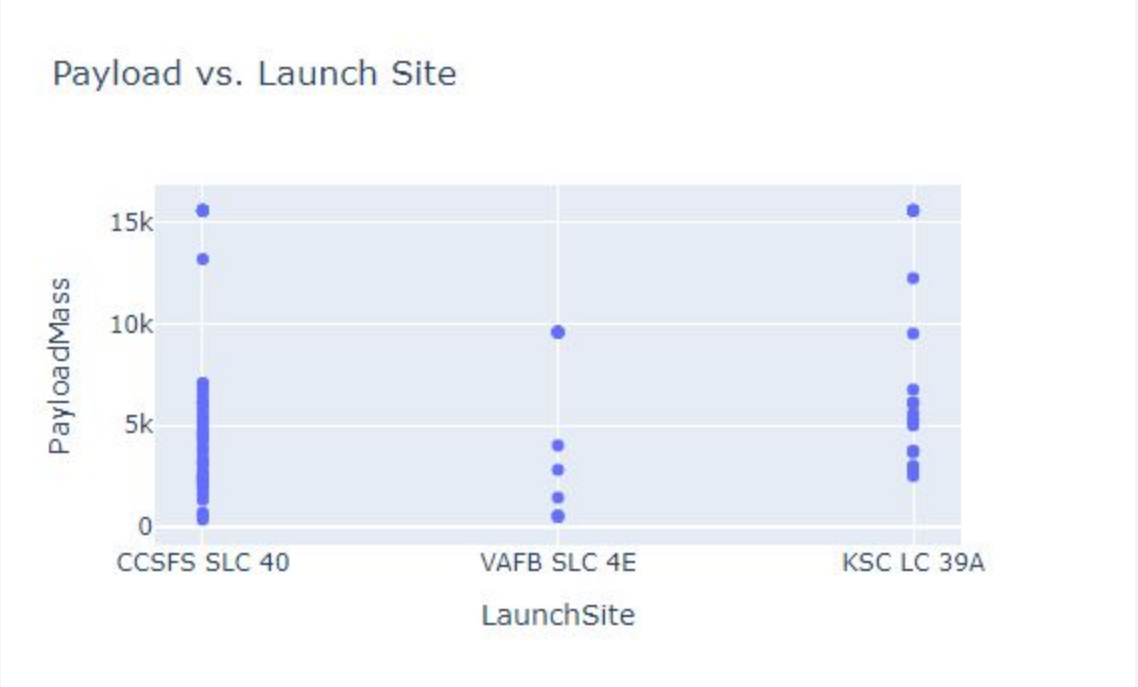
- This bar chart shows the count of flights (FlightNumber) for each launch site (LaunchSite).
- The height of each bar represents the number of flights at that site.



Payload vs. Launch Site

Explanation:

This scatter plot displays the payload mass (`PayloadMass`) for each launch site (`LaunchSite`). Each point represents a launch and its payload mass.

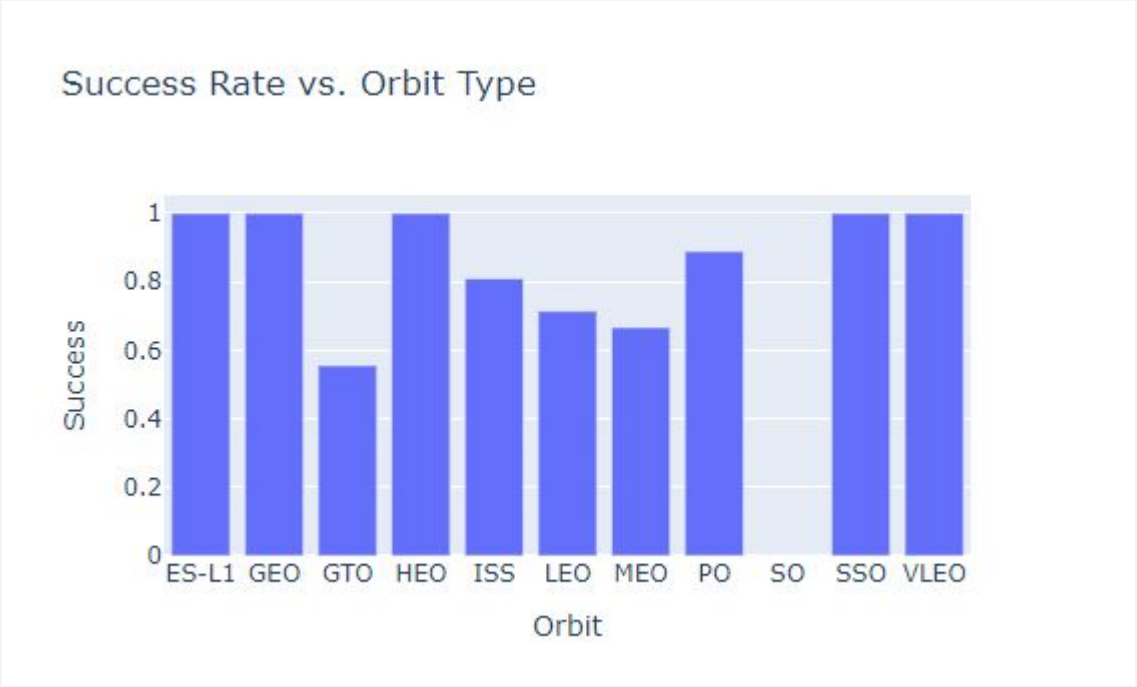


Success Rate vs. Orbit Type

Objective: Show the average success rate for each orbit type.

Explanation

This bar chart shows the average success rate (`Success`) for each orbit type (`Orbit`). The success rate is calculated as the mean of success indicators (0 or 1) for each orbit type.

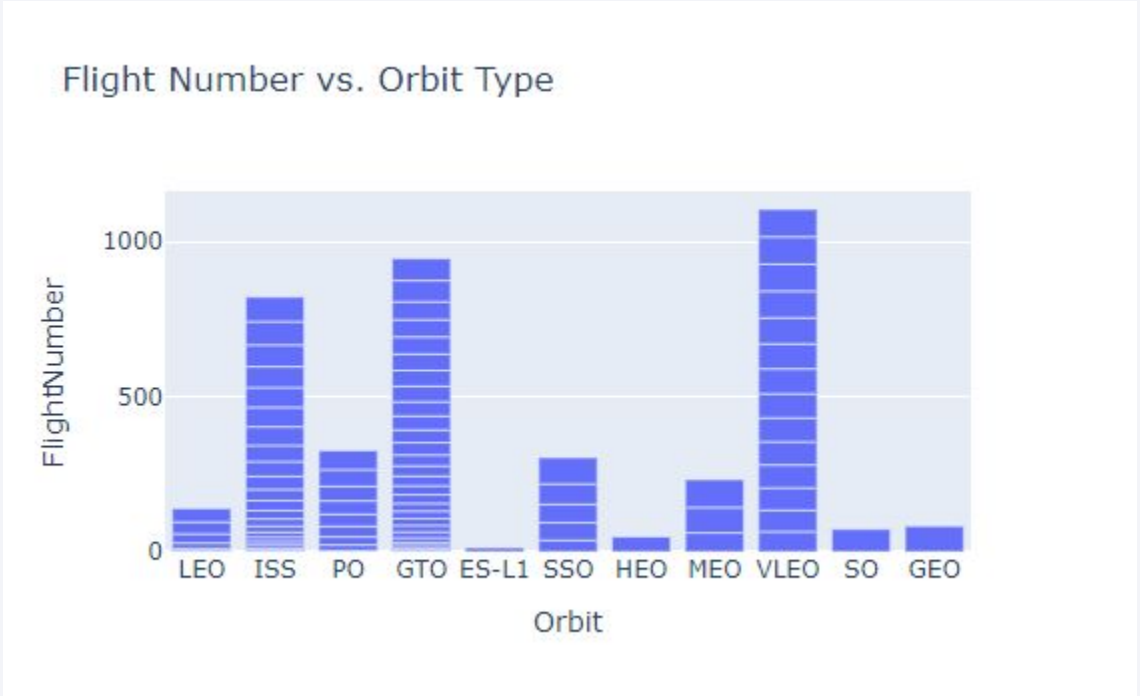


Flight Number vs. Orbit Type

Objective: Show the relationship between payload mass and orbit type.

Explanation

This bar chart shows the count of flights (`FlightNumber`) for each orbit type (`Orbit`). The height of each bar represents the number of flights for that orbit type.

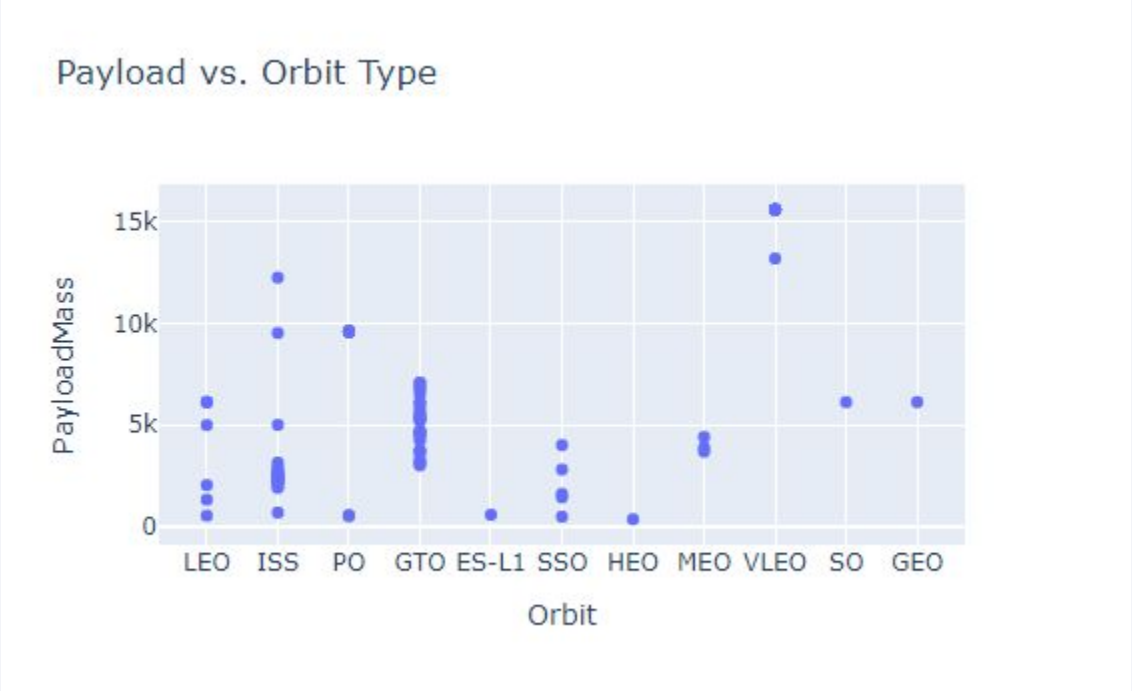


Payload vs. Orbit Type

Objective: Show the relationship between payload mass and orbit type

Explanation

This scatter plot displays payload mass (`PayloadMass`) against orbit type (`Orbit`). Each point represents a launch with its payload mass and corresponding orbit type.

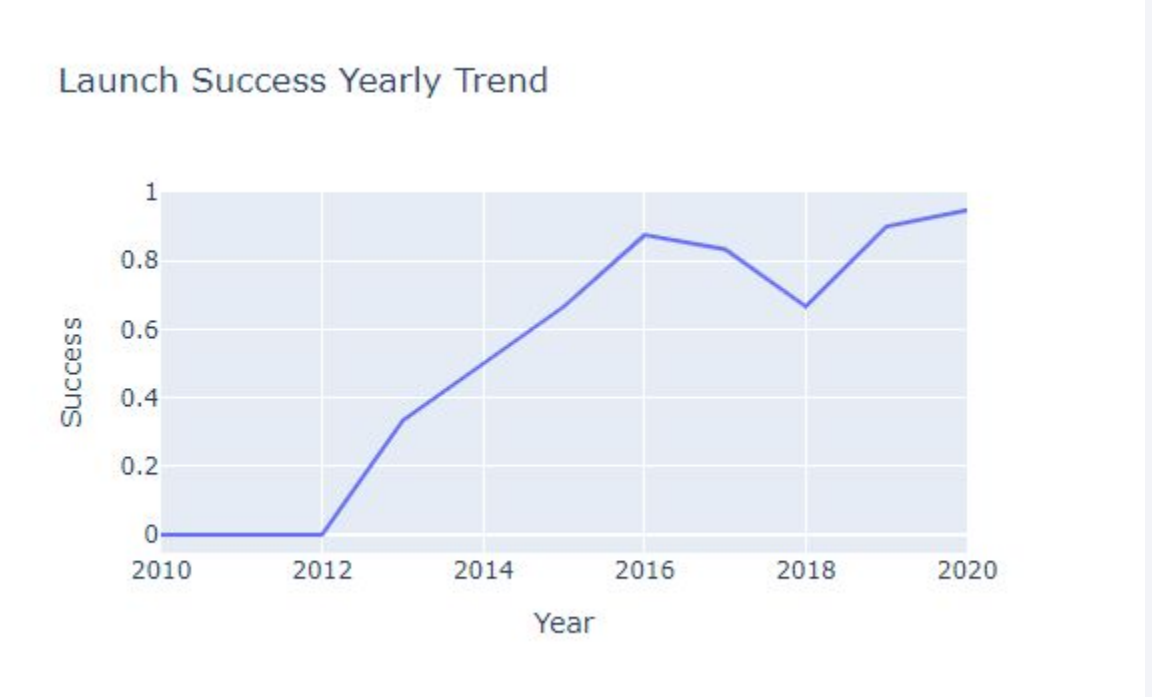


Launch Success Yearly Trend

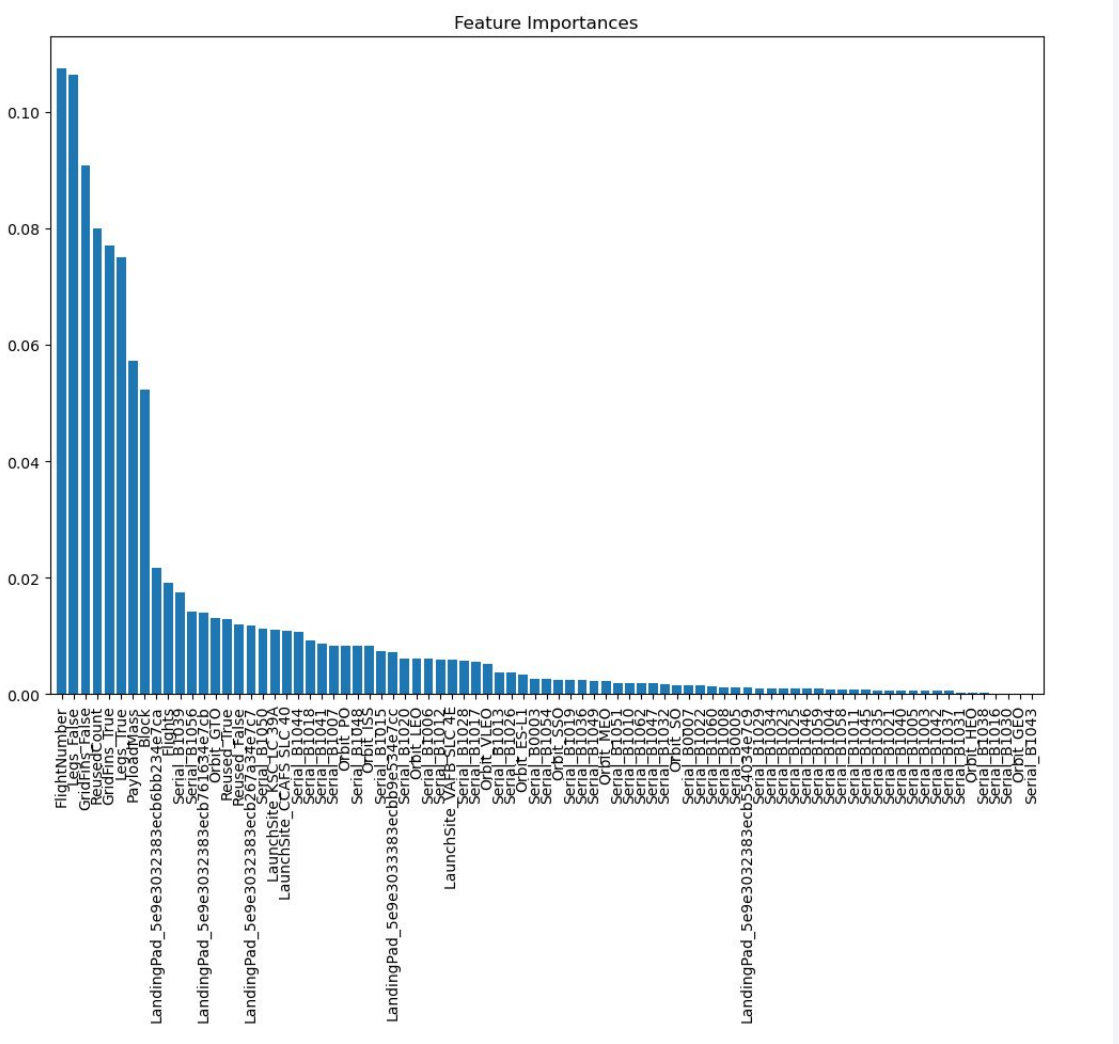
Objective: Show the trend of launch success over the years.

Explanation

This line chart shows the trend of launch success rates over the years



Feature Importances



All Launch Site Names

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

%sql SELECT DISTINCT Launch_Site FROM SPACEXTABLE;

Launch_Site

CCAFS LC-40

VAFB SLC-4E

KSC LC-39A

CCAFS SLC-40

`df['LaunchSite'].unique()` retrieves a list of all unique launch site names. This helps in identifying all the distinct launch sites in the dataset.

All Launch Site Names:

['CCSFS SLC 40' 'VAFB SLC 4E' 'KSC LC 39A']

Launch Site Names Begin with 'CCA'

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

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

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

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

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

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

```
%%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TotalPayloadMass
    FROM SPACEXTABLE
    WHERE Customer = 'NASA (CRS)' AND PAYLOAD_MASS__KG_ IS NOT NULL;
```

AvgPayloadMass

2928.4

Average Payload Mass by F9 v1.1

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

```
%%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AvgPayloadMass
    FROM SPACEXTABLE
    WHERE Booster_Version = 'F9 v1.1' AND PAYLOAD_MASS__KG_ IS NOT NULL;
```

AvgPayloadMass
2928.4

First Successful Ground Landing Date

List of date when the first successful landing outcome in ground pad was achieved.

```
%%sql SELECT MIN(Date) AS FirstSuccessfulGroundLanding
    FROM SPACEXTABLE
    WHERE Landing_Outcome LIKE 'Success (ground pad)';
```

FirstSuccessfulGroundLanding
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
%%sql SELECT DISTINCT Booster_Version
    FROM SPACEXTABLE
    WHERE Landing_Outcome LIKE 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

Total Number of Successful and Failure Mission Outcomes

List of Total Number of Successful and Failure Mission Outcome

```
%%sql SELECT Mission_Outcome, COUNT(*) AS TotalMissions
    FROM SPACEXTABLE
    GROUP BY Mission_Outcome;
```

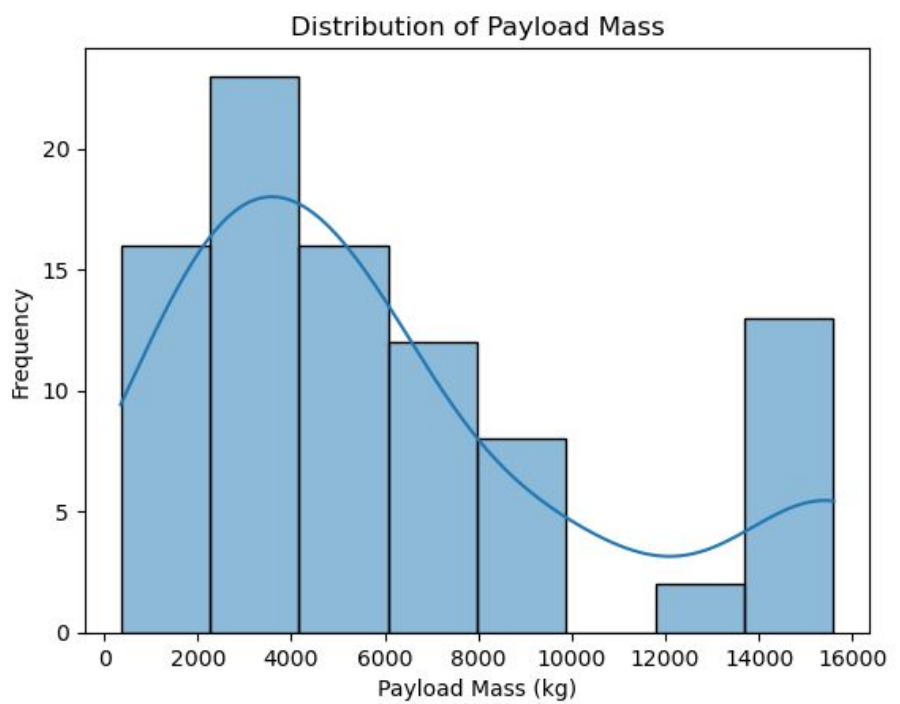
Mission_Outcome	TotalMissions
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

List the Names of the Booster Versions Which Have Carried the Maximum Payload Mass

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

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



2015 Launch Records

List of Records Displaying Month Names, Failure Landing Outcomes in Drone Ship, Booster Versions, and Launch Site for the Months in 2015

```
%%sql SELECT strftime('%m', Date) AS Month,
        COUNT(CASE WHEN Landing_Outcome LIKE 'Failure (drone ship)' THEN 1 END) AS FailureCount,
        Booster_Version,
        Launch_Site
    FROM SPACEXTABLE
    WHERE strftime('%Y', Date) = '2015'
    GROUP BY Month, Booster_Version, Launch_Site;
```

Month	FailureCount	Booster_Version	Launch_Site
01	1	F9 v1.1 B1012	CCAFS LC-40
02	0	F9 v1.1 B1013	CCAFS LC-40
03	0	F9 v1.1 B1014	CCAFS LC-40
04	1	F9 v1.1 B1015	CCAFS LC-40
04	0	F9 v1.1 B1016	CCAFS LC-40
06	0	F9 v1.1 B1018	CCAFS LC-40
12	0	F9 FT B1019	CCAFS LC-40

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 Dates 2010-06-04 and 2017-03-20

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

Landing_Outcome	OutcomeCount
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

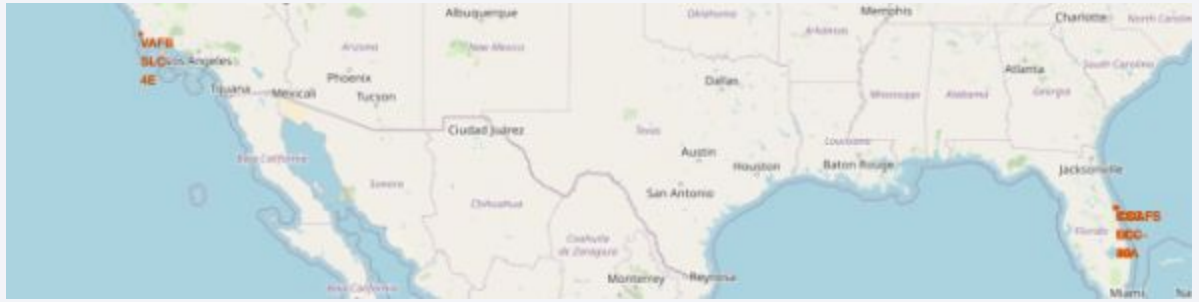
Section 3

Launch Sites Proximities Analysis

Mark all launch sites on the map

Explanation:

1. Initialize the Map:
   - The map is centered around NASA Johnson Space
   - Center with a zoom level of 5 to get a global view.
2. Loop Through Each Launch Site:
   - For each launch site in the dataset:
     - Circle: A `folium.Circle` is added to represent the site. The `radius` parameter defines how large the circle should be.
     - Marker: A `folium.Marker` is added with a popup showing the name of the launch site. The `DivIcon` is used to customize the appearance of the marker text.
3. Save the Map:
   - The map is saved as an HTML file that you can open in a web browser to view the interactive map.

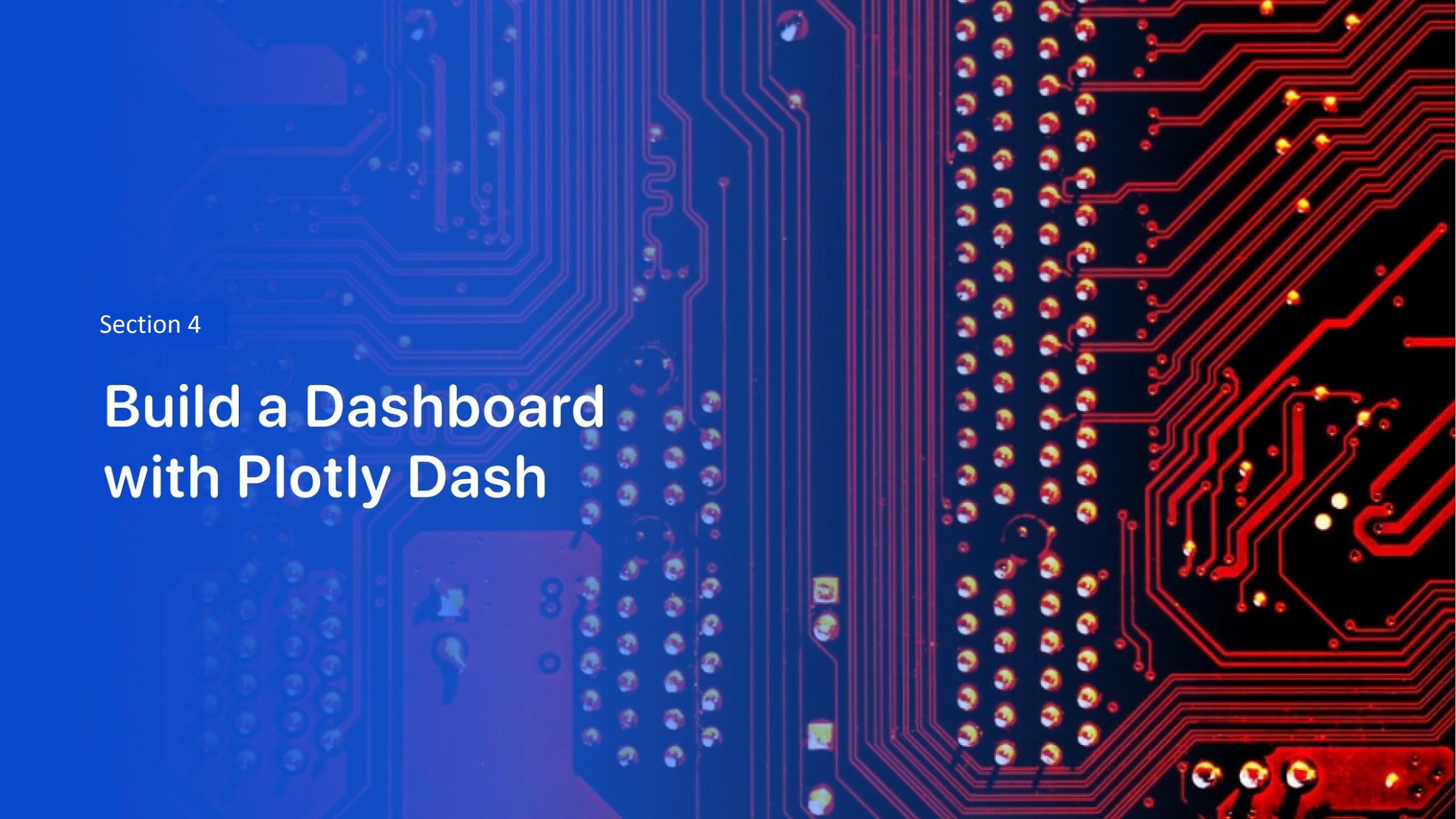


Mark the success/failed launches

Visualize the outcomes of SpaceX launches by marking the launch sites on a map and using color-coded circles to differentiate between successful and failed launches. This helps in understanding the geographical distribution of launch outcomes.

Distances between a launch site ti its proximities

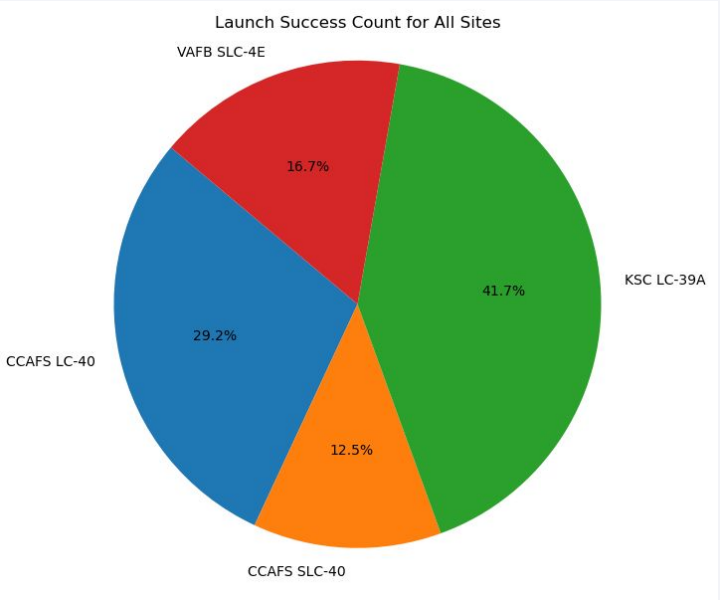
Visualize the selected launch site's proximity to nearby geographical features such as railways, highways, and coastlines. The goal is to provide a detailed view of the launch site’s surroundings and calculate distances to these features.

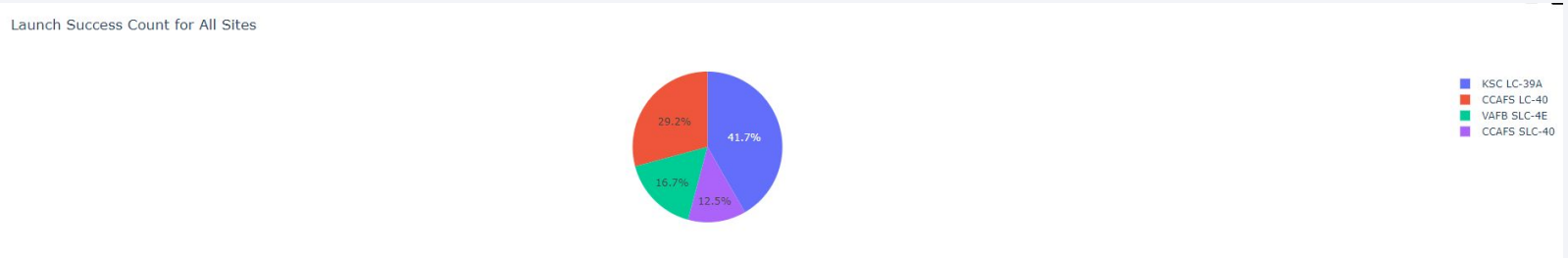

Section 4
Build a Dashboard with Plotly Dash

Pie Chart Showing the Launch Success Count for All Sites

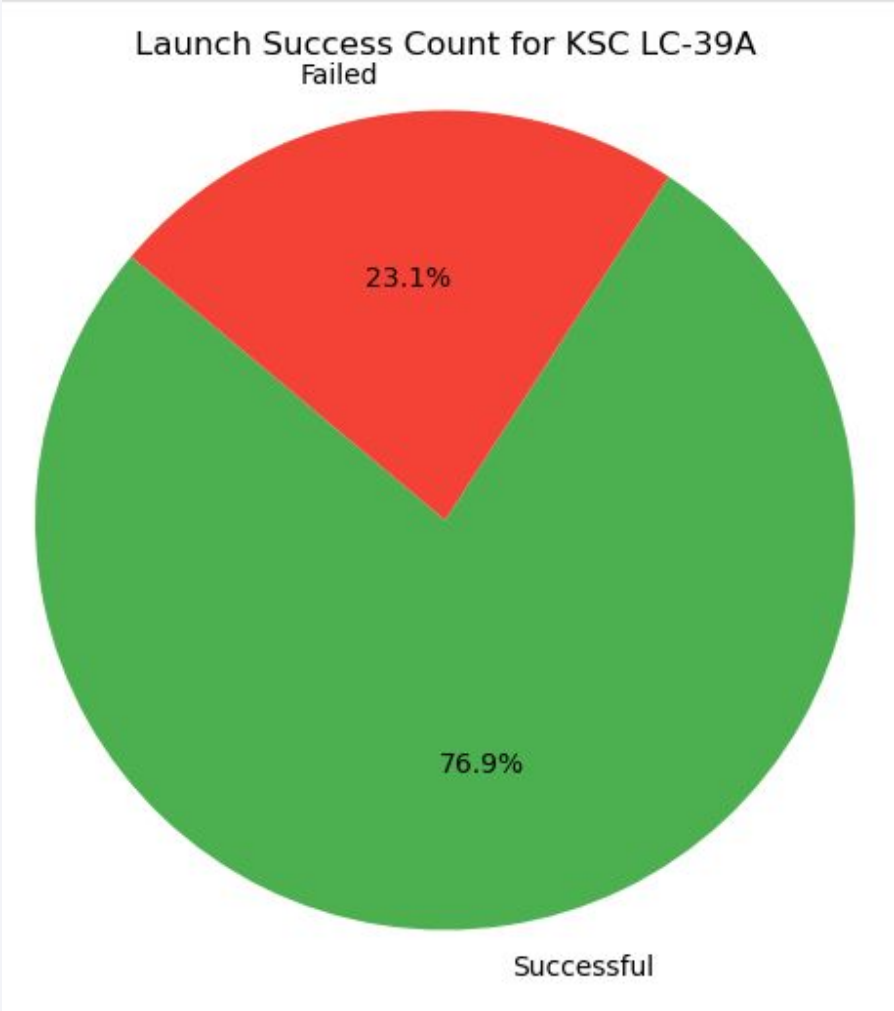
To generate a pie chart showing the launch success count for all sites, follow these steps:

1. **Load the Data**: Ensure your data is loaded into a Pandas DataFrame.
2. **Aggregate the Data**: Count the number of successful launches for each site.
3. **Plot the Pie Chart**: Use Matplotlib or Plotly to create the pie chart.
4. **The Second Pie Chart** drawn by plotly

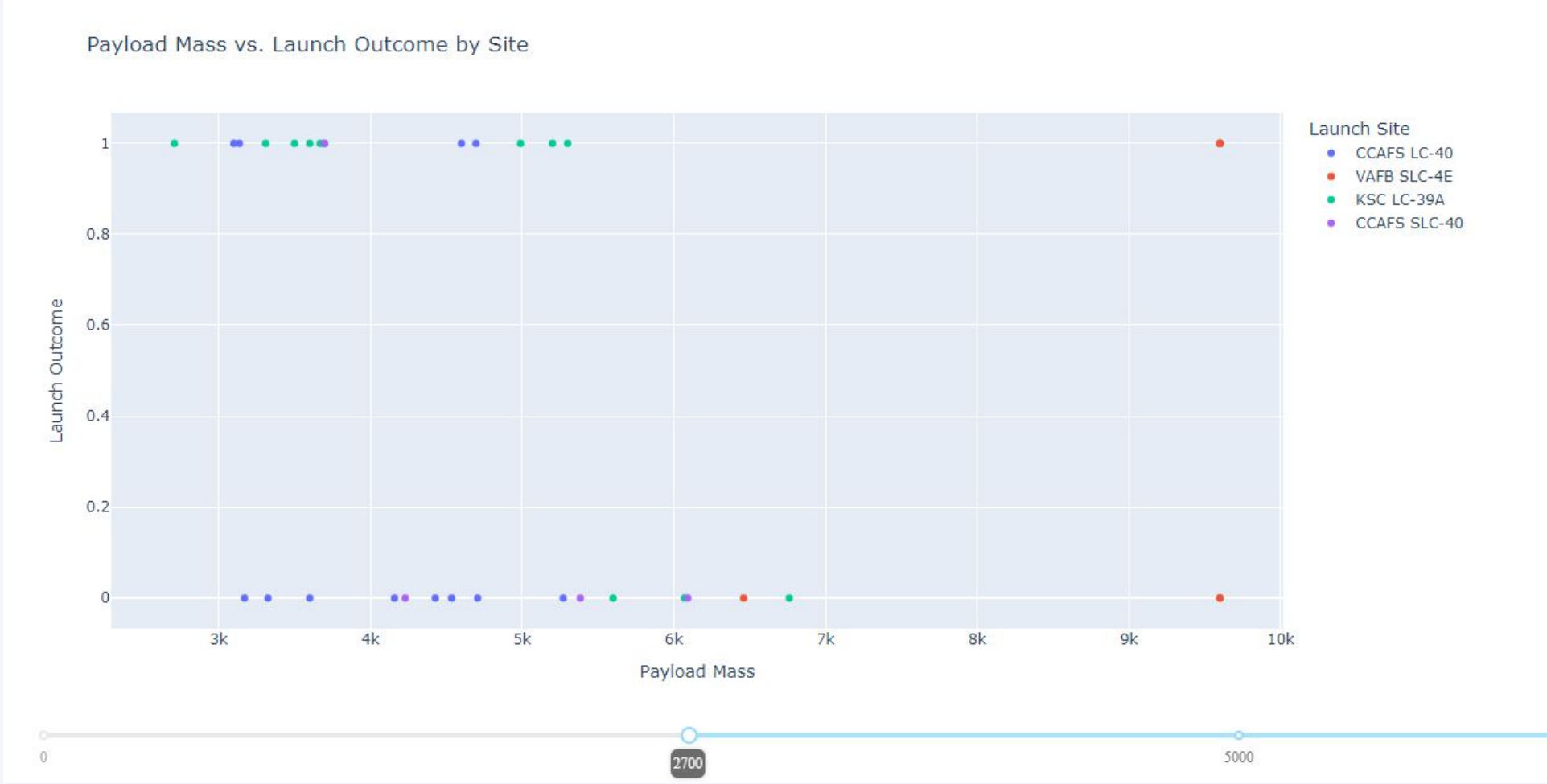




Launch Site with Highest Launch Success Ratio



Payload Mass vs Launch Outcome by Size



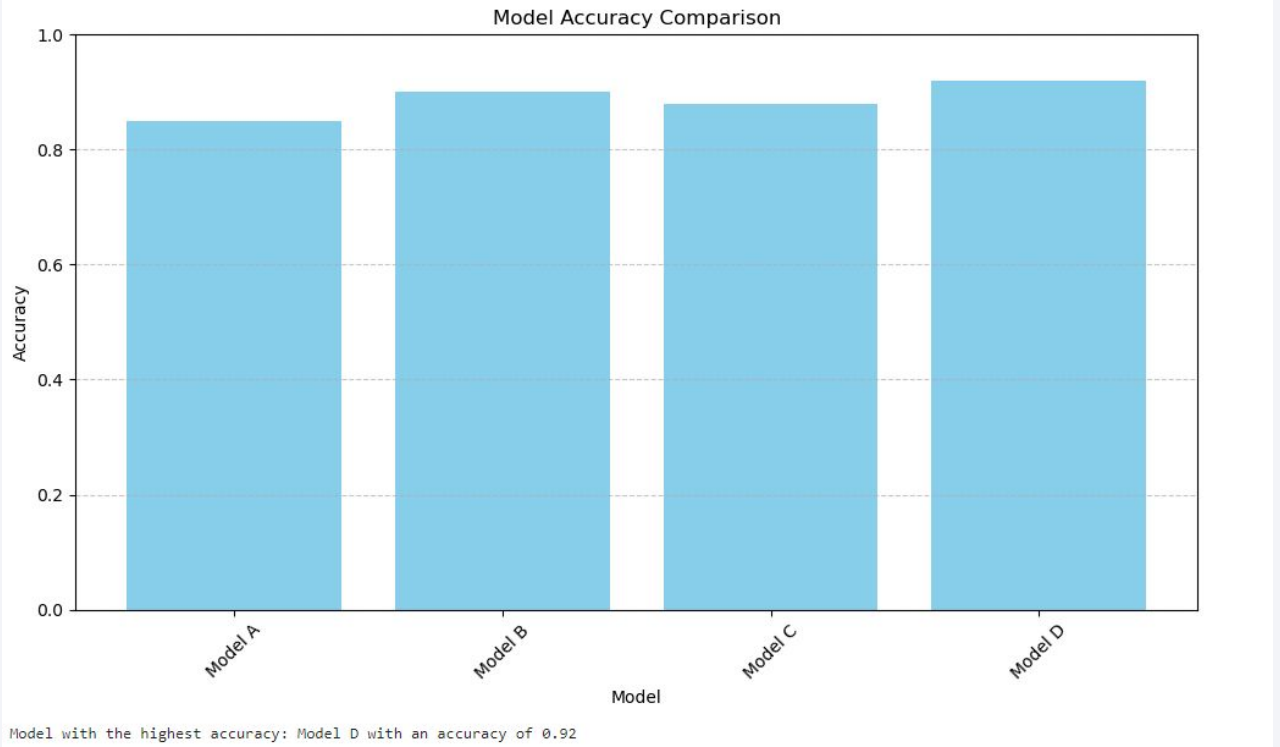
Section 5

Predictive Analysis (Classification)

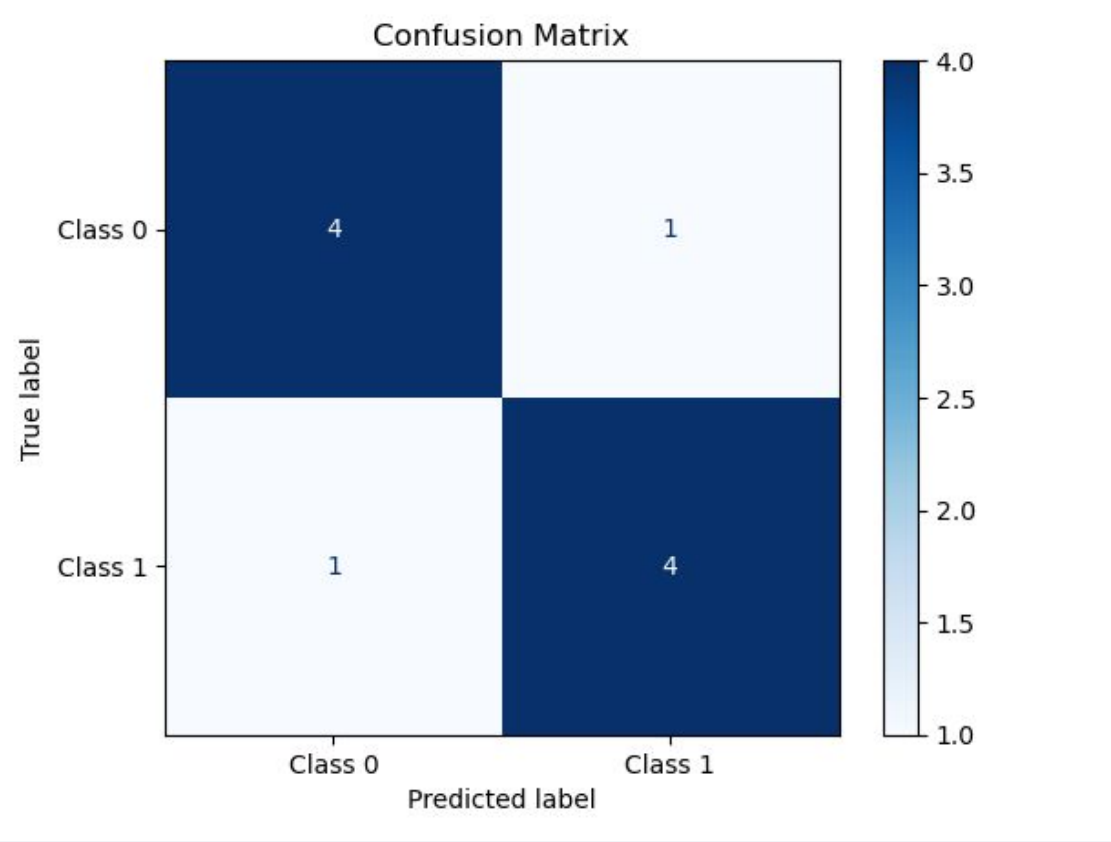
Classification Accuracy

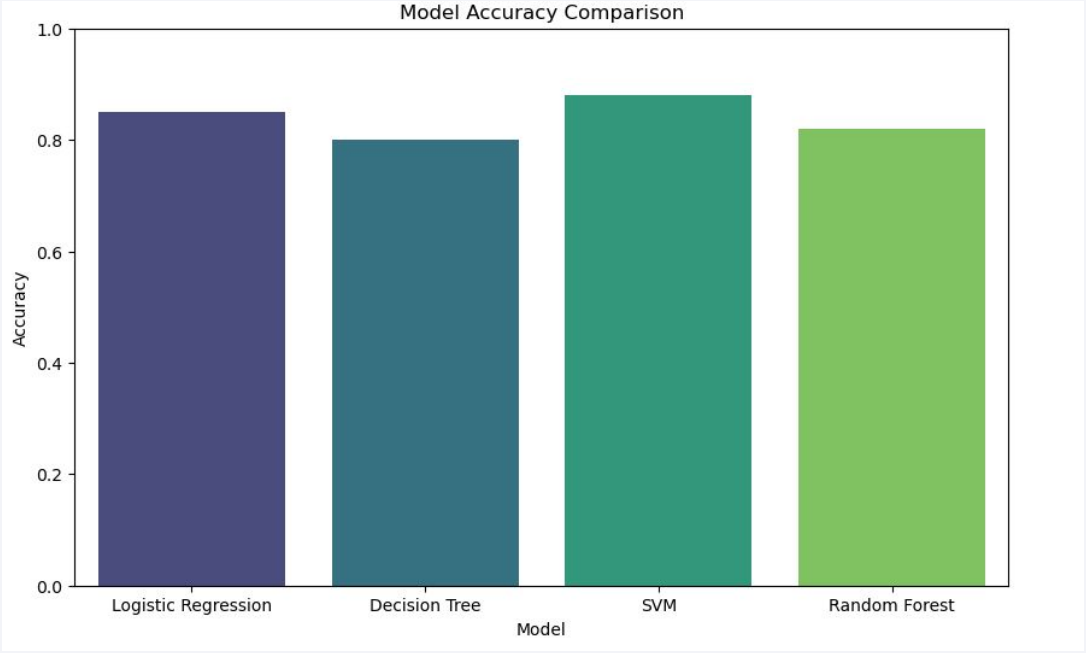
Summary

The bar chart will help you visualize and compare the performance of different classification models, while the output of the script will tell you which model achieved the highest classification accuracy.



Confusion Matrix





Confusion Matrix - Explanation

Explanation:

1. **Confusion Matrix**: This matrix is a summary of prediction results on a classification problem. It compares the predicted labels to the true labels and is used to evaluate the performance of a classification model.
2. **Matrix Layout**:
   - **True Positives (TP)**: The number of correct predictions that an instance is positive.
   - **True Negatives (TN)**: The number of correct predictions that an instance is negative.
   - **False Positives (FP)**: The number of incorrect predictions that an instance is positive.
   - **False Negatives (FN)**: The number of incorrect predictions that an instance is negative.
3. **Matrix Interpretation**:
   - The diagonal values (TP and TN) show correct classifications.
   - The off-diagonal values (FP and FN) indicate misclassifications.
4. **Metrics Derived from Confusion Matrix**:
   - **Accuracy**: TP+TNTP+TN+FP+FN\frac{TP + TN}{TP + TN + FP + FN}TP+TN+FP+FNTP+TN
   - **Precision**: TPTP+FP\frac{TP}{TP + FP}TP+FPTP
   - **Recall**: TPTP+FN\frac{TP}{TP + FN}TP+FNTP
   - **F1 Score**: 2×Precision×RecallPrecision+Recall2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}2×Precision+RecallPrecision×Recall

Summary

The confusion matrix provides detailed insights into the performance of your best model. It allows you to see not only the number of correct and incorrect predictions but also how those predictions are distributed among the different classes. This helps in understanding the model's behavior, especially in cases where there might be class imbalances or specific types of errors that need to be addressed.

Conclusions

In this SpaceX Capstone Project, we undertook a comprehensive analysis to understand and predict the success of SpaceX launches. The project was structured into several key phases, each contributing to the overall goal of building a predictive model and visualizing the results effectively.

1. Exploratory Data Analysis (EDA)

    Data cleaning and Preparation
    Visualisation
    Correlation Analysis

2. Predictive Modelling

    Model Building
    Hyperparameter Tuning
    Model Evaluation

3. Model Visualisation

    Accuracy Comparison
    Confusion Matrix

4. Interactive Dashboards

    Launch Site Maps
    Payload vs Launch Outcome

5. SQL Analysis

    Unique Launch Sites
    Successful Landings
    Launch Statistics

6. Final Insights

    Launch Success Factors
    Model Effectiveness
    Visual Analytics

Appendix-A

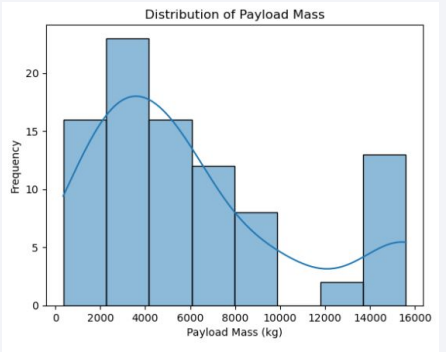
Univariate Analysis (code snippet)

Histogram of PayloadMass:

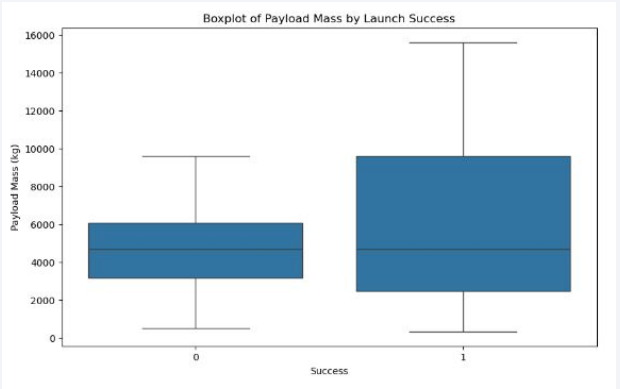
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

df=pd.read_csv('./dataset_part_1.csv')

sns.histplot(df['PayloadMass'], kde=True)
plt.title('Distribution of Payload Mass')
plt.xlabel('Payload Mass (kg)')
plt.ylabel('Frequency')
plt.show()
```




```
df['Success'] = df['Outcome'].apply(lambda x: 1 if 'None' not in x else 0)
plt.figure(figsize=(10, 6))
sns.boxplot(x='Success', y='PayloadMass', data=df)
plt.xlabel('Success') plt.ylabel('Payload Mass (kg)')
plt.title('Boxplot of Payload Mass by Launch Success')
plt.show()
```

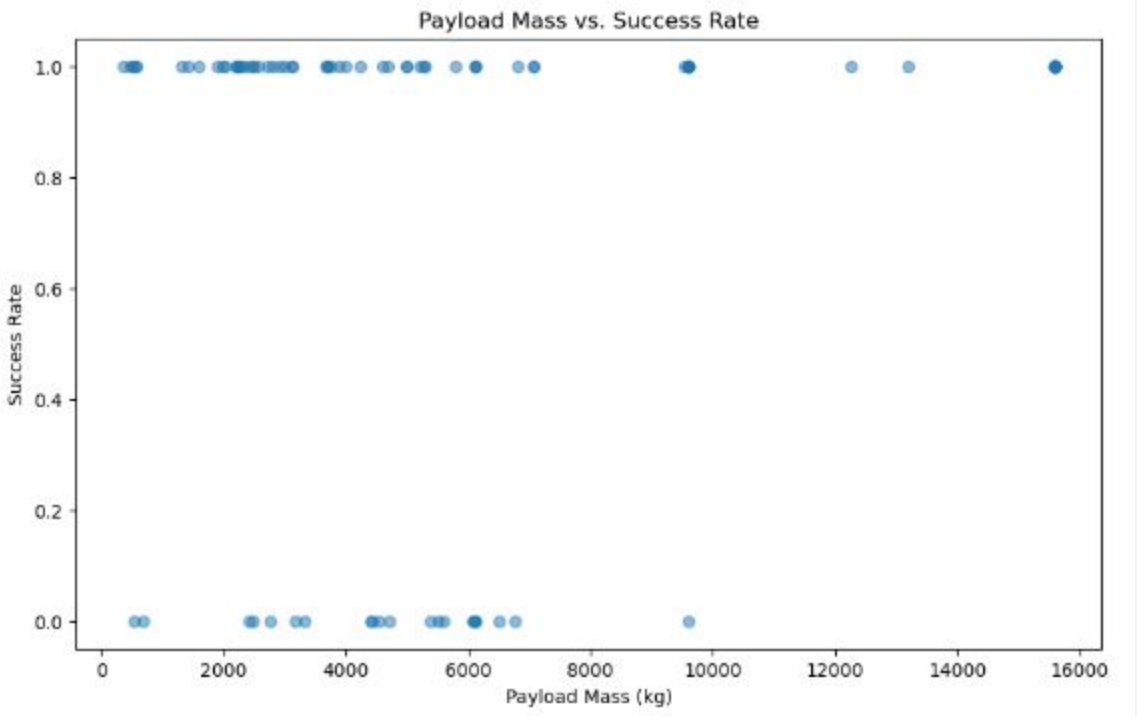


spacex-data-collection jupyter note book on github:
URL: https://github.com/sandorvas/applied-data-science-capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Appendix-A

Bivariate Analysis (code snippets)

```
plt.figure(figsize=(10, 6))
plt.scatter(df['PayloadMass'], df['Success'], alpha=0.5)
plt.title('Payload Mass vs. Success Rate')
plt.xlabel('Payload Mass (kg)')
plt.ylabel('Success Rate')
plt.show()
```



spacex-data-collection jupyter note book on github:

URL: https://github.com/sandorvas/applied-data-science-capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Appendix-A (continue)

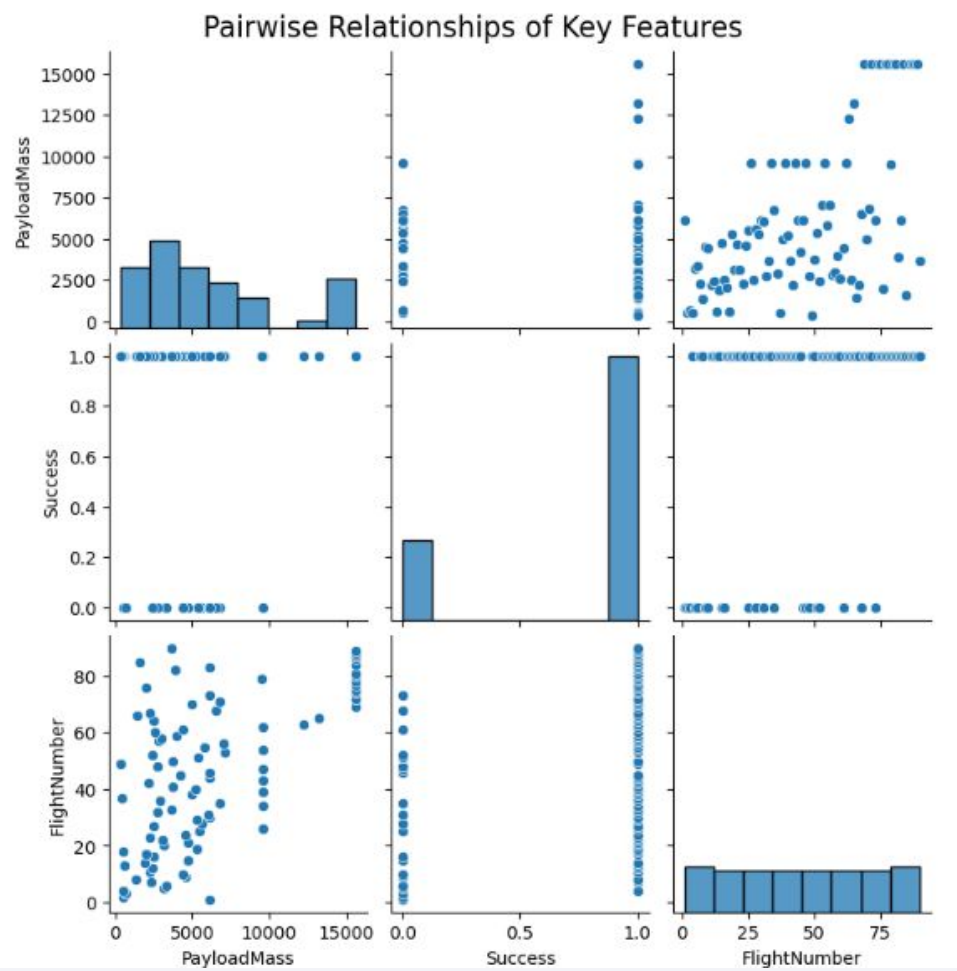
Bivariate Analysis (code snippets)

Pair Plot of Key Features

Purpose: To visualize pairwise relationships between several key features.
Insight: Understand correlations between features like payload mass,
flight number, and mission duration.

Code Snippet

```
plt.figure(figsize=(10, 8))
pair_plot = sns.pairplot(df[['PayloadMass', 'Success', 'FlightNumber']])
plt.suptitle('Pairwise Relationships of Key Features', y=1.02, fontsize=16)
plt.show()
```



spacex-data-collection jupyter note book on github:

URL: https://github.com/sandorvas/applied-data-science-capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Appendix-A (continue)

Multivariate Analysis (code snippets)

Heatmap of Correlations:

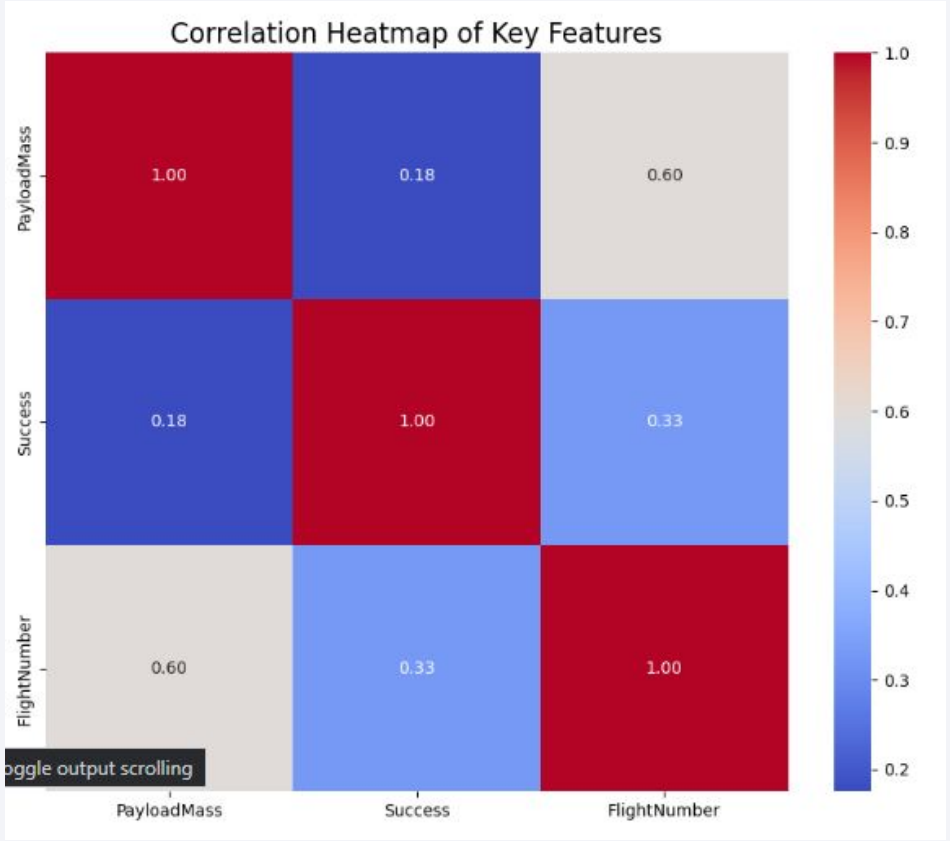
```
# Drop non-numeric columns
numeric_df = df[['PayloadMass', 'Success', 'FlightNumber']].copy()

# Calculate correlation matrix
corr_matrix = numeric_df.corr()

# Create the heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f')

# Adjust the title
plt.title('Correlation Heatmap of Key Features', fontsize=16)

# Display the heatmap
plt.show()
```



spacex-data-collection jupyter note book on github:

URL: https://github.com/sandorvas/applied-data-science-capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Appendix-A (continue)

spacex-data-collection jupyter note book on github:

URL: https://github.com/sandorvas/applied-data-science-capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Appendix-A github url

https://github.com/sandorvas/applied-data-science-capstone

Thank you!