
# Winning Space Race with Data Science

Vicente Franco
January, 2025

# Outline

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

# Executive Summary

This project leverages **machine learning** to predict the success of SpaceX missions using historical launch data sourced from SpaceX's API.

- **Goal**: Predict mission success to optimize launch planning and reduce costs.
- **Methodology**: Data cleaning, exploratory analysis with SQL and visuals, feature engineering, and machine learning (LR, SVR, Decision Tree and K-nearest).
- **Results**: Identified critical factors influencing success, and a model that predicts outcomes of **81% prediction accuracy**.
  - Models performed better predicting Successful landings. However, 44% of the times predicted a success when it was really a failure.
  - The best Model were LR and SVM. Being LR faster to compute.
- **Impact**: Provides actionable insights for mission planning and risk assessment.

# Introduction

- SpaceX has revolutionized space exploration with its reusable rockets and ambitious missions. However, not all launches are successful.
- This project aims to **predict the success of SpaceX missions** using historical data sourced directly from SpaceX's API.
- By analyzing real mission features provided by the company, we can build a model to assess the likelihood of mission success.

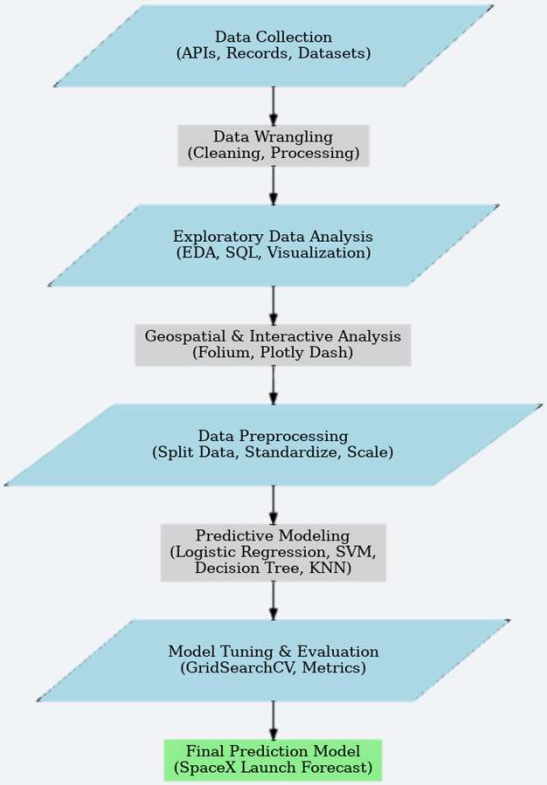
This work is part of the **IBM Data Science Professional Certificate**, demonstrating the application of data science techniques to real-world problems.

Section 1

# Methodology

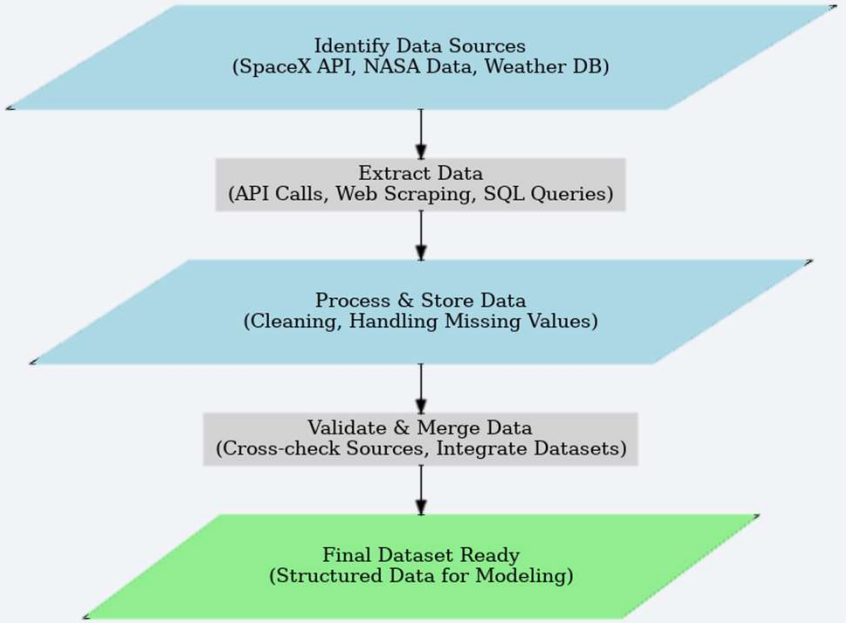
# Methodology

- **Data Collection & Preparation:**
  - Collected data from **public APIs, SpaceX records, and Wikipedia.**
  - Cleaned, transformed, and standardized data using **Pandas and SQL.**
- **Exploratory Data Analysis (EDA):**
  - Identified key patterns using **visualizations (Matplotlib, Seaborn)**.
  - Performed **SQL queries** for data insights.
  - Conducted **geospatial analysis (Folium)** and built **interactive dashboards (Plotly Dash)**.
- **Predictive Modeling:**
  - **Data Preprocessing:**
    - Split data into **training and test sets** (e.g., 80/20 split).
    - Standardized and scaled features to improve model performance.
  - **Applied Classification Models:**
    - **Logistic Regression** - Baseline probability estimation**.**
    - **Support Vector Machine (SVM)** - Complex decision boundaries.
    - **Decision Tree** - Feature-based classification.
    - **K-Nearest Neighbors (KNN)** - Similarity-based classification
- **Model Tuning & Evaluation:**
  - Optimized models using **GridSearchCV & hyperparameter tuning**.
  - Evaluated performance with **accuracy, precision, recall, F1-score,** and **confusion matrix.**



# Data Collection

- **Identifying Data Sources:**
  - **Primary Sources**: SpaceX API.
  - **Secondary Sources**: Wikipedia Launch Information.
- **Data Extraction Methods:**
  - API Calls: Used Python libraries (requests, JSON) to extract real-time launch data.
  - Web Scraping: Collected additional details using BeautifulSoup.
- **Data Processing & Storage:**
  - **Cleaning & Transformation**: Removed duplicates, handled missing values
  - **Structured Storage**: Stored in **SQL databases, Pandas DataFrames, and CSV formats**
- **Data Validation & Integration:**
  - Cross-referenced multiple sources to ensure accuracy
  - Merged different datasources into a final dataset for comprehensive insights.

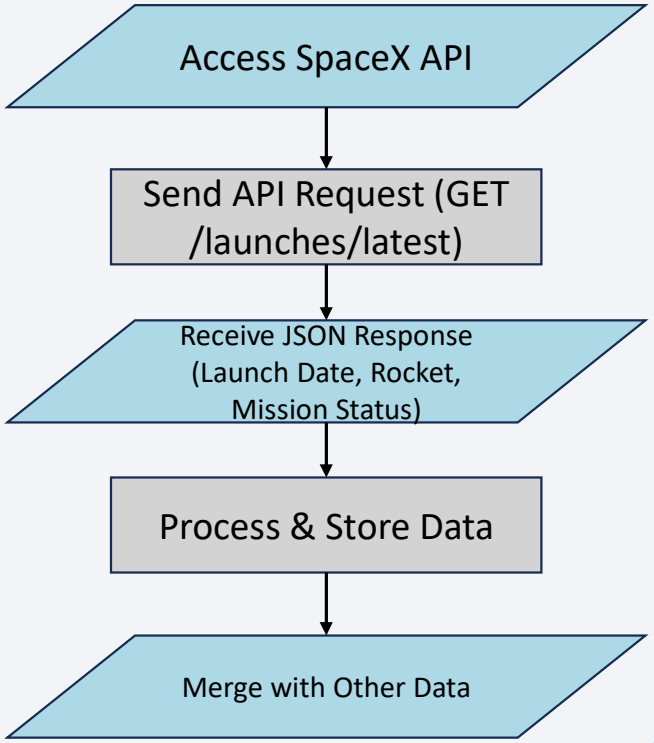



- **Outcome**: Built a **high-quality, structured dataset** for predictive modeling

# Data Collection – SpaceX API

- **Understanding the SpaceX API:**
  - Provides real-time launch data, rocket details, mission history, and launch success rates.
  - RESTful API with JSON responses, accessible via HTTP requests.
- **Making API Requests:**
  - Used Python's requests library to send GET requests.
  - Extracted key launch details like date, rocket type, mission status.
- **API Data Integration:**
  - Merged data from several API's to complete a single robust dataset
- **Data Processing & Storage:**
  - Converted JSON data into structured Pandas DataFrames.
  - Stored in SQL databases for further analysis.

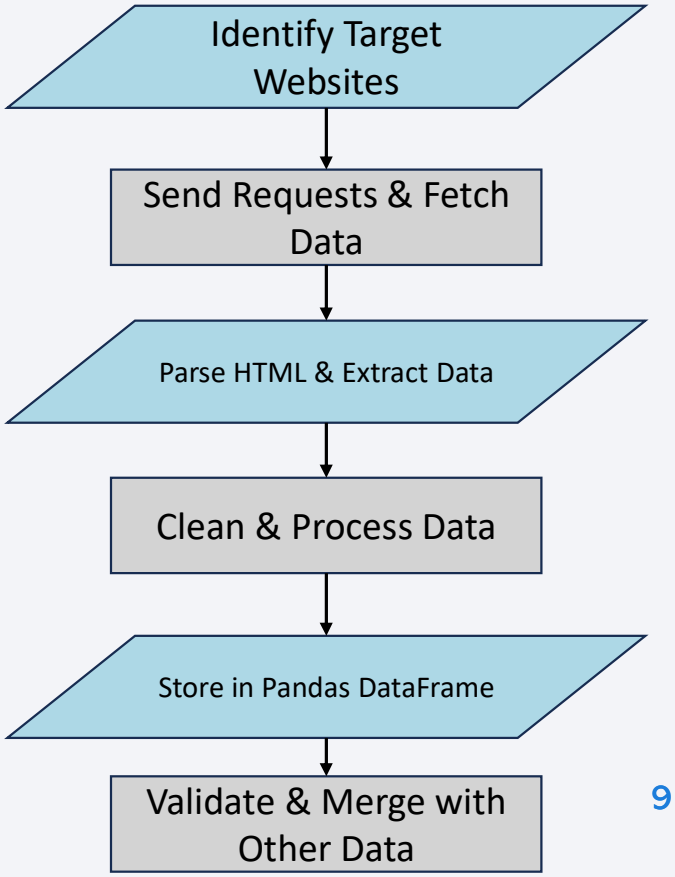
https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb



# Data Collection - Scraping

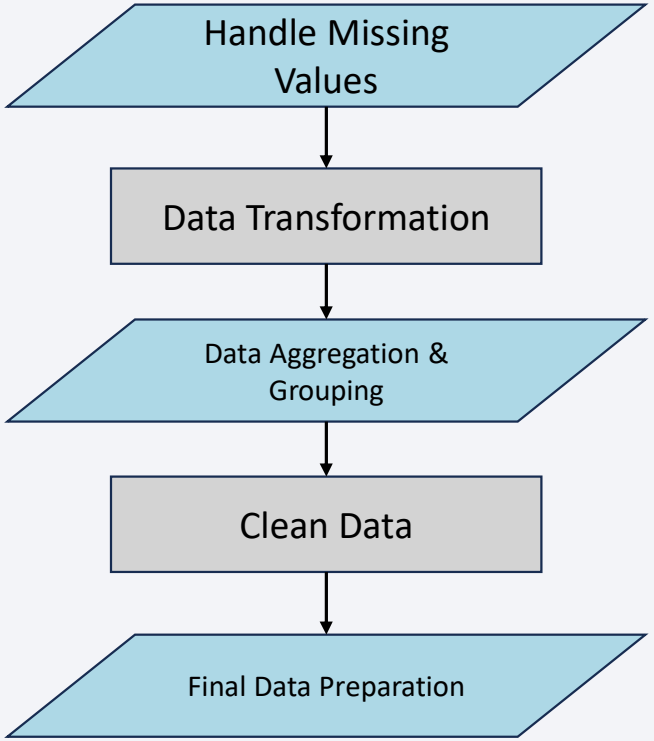
- **Identifying Target Websites:**
  - Scraped data from **Wikipedia**
- **Web Scraping Tools & Libraries**
  - Used **BeautifulSoup** to parse HTML.
- **Sending Requests & Parsing HTML**
  - **Requests Library**: Used to fetch web page content.
  - **BeautifulSoup**: Parsed the HTML structure to extract key data points like **launch dates, Payload, and mission descriptions**.
- **Data Extraction & Processing**
  - Cleaned scraped data by removing unnecessary HTML tags and formatting errors.
  - Stored the data in **Pandas DataFrames** and in SQL for further analysis.
- **Data Validation & Integration:**
  - Cross-referenced scraped data with **official SpaceX records** to ensure accuracy.

https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb



# Data Wrangling

- **Handling Missing Values:**
  - **Imputation**: Filled missing values using **mean for numerical data.**
  - **Removal**: Dropped rows/columns with excessive missing data that could skew analysis.
- **Data Transformation**
  - **Normalization & Standardization**: Applied techniques like **Min-Max Scaling** to scale data between specific ranges.
- **Data Aggregation & Grouping**
  - **Grouping Data**: Grouped launch data by **rocket type**, **mission success**, etc., to derive insights.
  - **Aggregation**: Calculated statistical metrics like **mean launch success rate** for each group.
- **Data Cleaning**
  - **Feature Encoding**: Converted categorical data into numerical values using **One-Hot Encoding**.
- **Final Data Preparation**
  - Cleaned, transformed, and structured the dataset into **Pandas DataFrames** and **SQL tables** ready for modeling.




https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb

# EDA with Data Visualization

- **Payload Mass vs. Flight Number**
  - **Importance**: Helps identify whether heavier payloads influenced launch success or failure trends.
- **Launch Site vs. Flight Number**
  - **Importance**: Reveals patterns and potential biases related to specific sites, like **more frequent launches** at certain locations.
- **Payload Mass vs. Launch Site**
  - **Importance**: Helps identify if certain sites are better suited for heavier payloads, affecting success rates.
- **Success Rate by Orbit Type**
  - **Purpose**: Plotted the **success rate per orbit type** (LEO, GTO, etc.).
  - **Importance**: Identifies if certain orbit types are more challenging or associated with lower success rates.
- **Flight Number vs. Orbit with Class**
  - **Purpose**: Analyzed how **orbit type** (LEO, GTO, etc.) changed with flight number and **mission class**.
  - **Importance**: Provides insight into whether SpaceX's mission complexity evolved over time.
- **Payload Mass vs. Orbit with Launch Outcome**
  - **Purpose**: Explored the relationship between **payload mass, orbit type**, and **launch success**.

  •**Importance**: Indicates if specific payloads and orbits have higher failure rates, which can inform predictions.
- **Success Rate by Launch Site**
  - **Importance**: Identifies if certain sites are more reliable, aiding in future launch site selection.
- **Number of Successful and Failed Launches Over Time**
  - **Purpose**: Tracked the trend of **successful vs. failed launches** over time.
  - **Importance**: Demonstrates the progress in SpaceX's reliability and pinpointed periods with high failure rates.
- **Yearly Success Rates**
  - **Purpose**: Analyzed **yearly success rates** to see improvements over time.
  - **Importance**: Helps gauge SpaceX's **growth in performance**, showing technological advancements.

https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb

# EDA with SQL

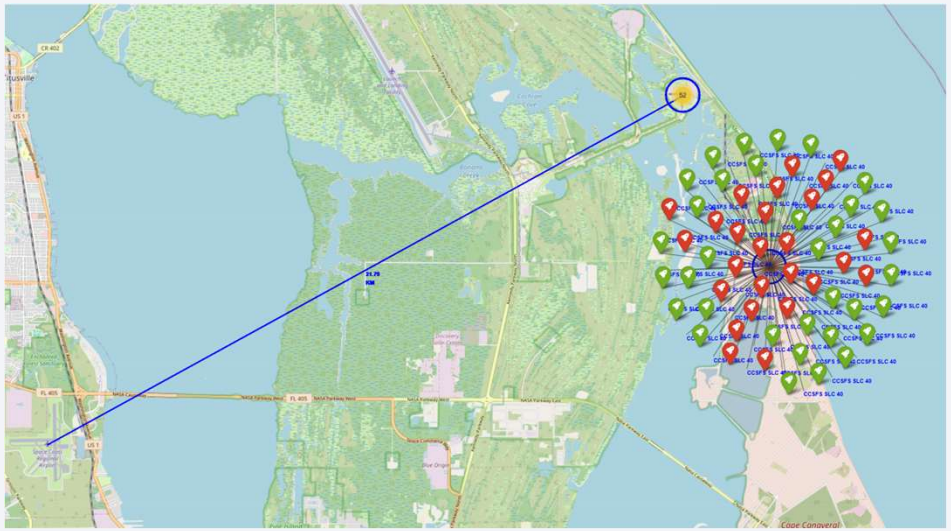
These SQL queries were essential for cleaning, filtering, and analyzing SpaceX's **launch performance**, identifying patterns, and drawing actionable conclusions to improve future predictions.

- **Retrieve all unique LaunchSite from the SPACEXTBL table:**
  - Provided an overview of all the launch sites used by SpaceX, essential for understanding site performance trends and distribution.
- **Retrieve up to 5 rows where the LaunchSite contains "CCS":**
  - Focused on specific launch sites (e.g., "CCS") to analyze how certain sites may influence launch success and failure patterns.
- **Calculate the average PayloadMass for missions using the Falcon 9 booster version:**
  - Offered insights into the typical payload capacity handled by Falcon 9, helping to assess performance and success rates based on payload mass.
- **Identify the first successful launch date:**
  - Pinpointed SpaceX's earliest successful mission, marking a milestone in the company's development and performance evolution.
- **Find the BoosterVersion for payloads between 4000 and 6000 kg with successful outcomes:**
  - Identified booster versions most effective for handling mid-range payloads, providing insights into booster efficiency for specific payload types.
- **Count the number of occurrences for each Outcome:**
  - Helped understand the distribution of launch outcomes, allowing for a high-level view of overall success/failure trends.
- **Calculate the percentage of successful and failed launches:**
  - Calculated the success rate of launches, revealing the reliability and trends of SpaceX's missions over time.
- **Identify the BoosterVersion that carried the maximum payload:**
  - Highlighted which booster versions were most capable in terms of payload capacity, offering insight into SpaceX's most powerful models.
- **Retrieve launch details from 2015 with the Outcome "False ASDS":**
  - Focused on launches with specific failures, like "False ASDS", to identify potential reasons for failure and areas for improvement in future missions.
- **Count occurrences of each outcome between two dates:**
  - Analyzed the frequency of each outcome between two time periods, helping to observe changes in performance over time and identifying potential periods of improvement or concern.

https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb

# Build an Interactive Map with Folium

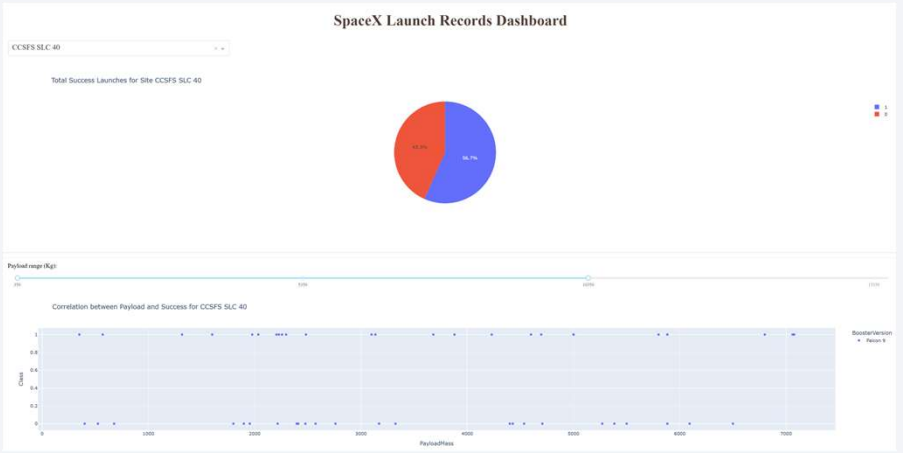
- **Launch Site Markers:**
  - Display the location of each launch site with its name in a custom icon. These markers help identify and highlight each site on the map.
- **Rocket Icon Markers:**
  - Represent the launch site with a rocket icon, color-coded based on the success or failure of the launch. This visually distinguishes sites based on performance.
- **Blue Circle:**
  - Represents an area of influence around each launch site (500m radius). It provides context to the geographical spread around the site.
- **Polyline (Path):**
  - Connects the start and end points (e.g., launch site to city/target) with a blue line. It visually shows the route or trajectory between two locations.
- **Distance Marker:**
  - Displays the calculated distance between the start and end points on the map. This provides a quick view of the distance between the launch site and the target.
- **Mouse Position Tracker:**
  - Tracks and displays the user's cursor position (latitude and longitude) at the top-right of the map for reference while exploring the map.



https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb

# Build a Dashboard with Plotly Dash

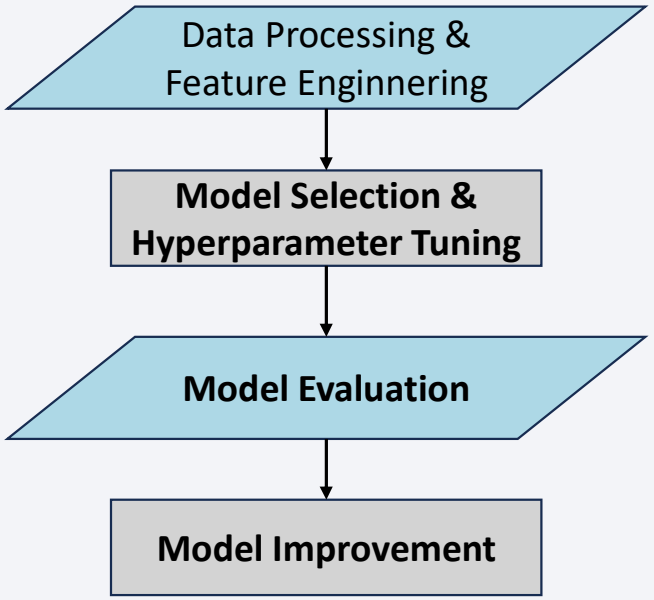
- **Pie Chart (Success Launches by Site)**
  - **Purpose**: Shows the distribution of successful launches for each site or a selected site.
  - **Interaction**: Dropdown to select a specific launch site or view data for all sites.
  - **Why Added**: Provides a quick overview of success rates across different launch sites and allows comparison between sites.
- **Scatter Plot (Payload vs Success)**
  - **Purpose**: Displays the correlation between the payload mass and launch success for each site or globally.
  - **Interaction**: Dropdown for site selection and a slider to filter payload range.
  - **Why Added**: Helps to analyze if larger payloads are linked to higher or lower success rates. Interactive filters provide deeper insights into specific payload ranges and sites.
- **Dropdown for Launch Site Selection**
  - **Purpose**: Allows the user to select specific launch sites or view data for all sites.
  - **Why Added**: Provides flexibility to focus on individual sites and compare them, or view a global summary.
- **Payload Range Slider**
  - **Purpose**: Filters the scatter plot to display payload ranges.
  - **Why Added**: Offers users the ability to explore how payload mass affects success, making the dashboard more interactive and customizable.




https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb

# Predictive Analysis (Classification)

- **Data Preprocessing & Feature Engineering:**
  - **Data Cleaning:** Missing values handled, categorical features encoded.
  - **Feature Selection:** Selected features based on correlation and domain relevance.
  - **Scaling:** Standardized numerical features for improved model performance.
- **Model Selection & Hyperparameter Tuning:**
  - **Model Selection:** Tried multiple models: Logistic Regression, SVM, KNN, Decision Trees.
  - **Hyperparameter Tuning:** Used **GridSearchCV** for optimal model parameters.
  - **Cross-Validation:** Ensured robustness with **10-fold cross-validation**.
- **Model Evaluation:**
  - **Evaluation Metrics:** Used **accuracy**, **precision**, **recall**, and **F1-score** for model comparison.
  - **Confusion Matrix:** Analyzed performance using raw and normalized confusion matrix.
- **Model Improvement:**
  - **Fine-Tuning:** Optimized hyperparameters for better accuracy (e.g., n_neighbors, kernel, max_depth).
  - **Model Comparison:** Compared all models and focused on the one with the best balance of precision and recall.

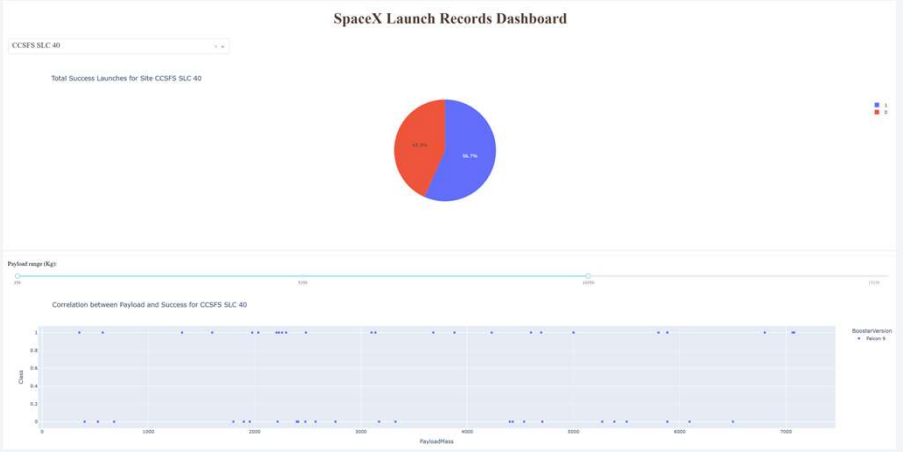


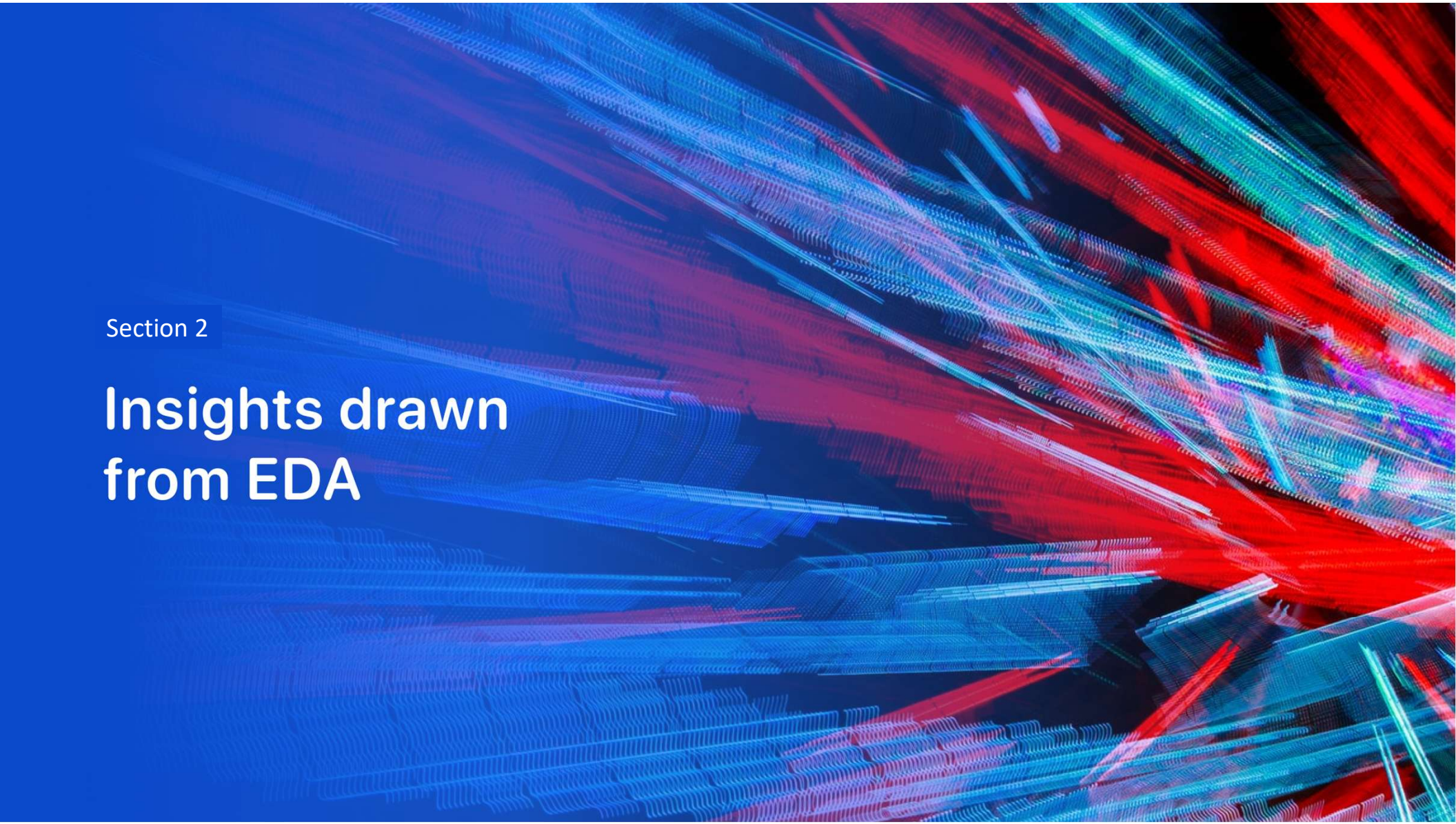

https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb

# Results

- **Exploratory Data Analysis:**
  - **Overall success rates** have improved since 2013, with declines in 2015 and 2018 due to specific challenges.
  - **GTO missions** exhibit no clear relationship between payload mass and success, though heavier payloads might indicate technological maturity.
  - **LEO missions** show better success rates with higher flight numbers, suggesting improved technology and experience.
  - **GEO missions** were 100% successful
  - **Higher payload mass** is identified as an outlier by the BoxPlot chart
  - **Higher mass payloads** tend to have lower failure rates, likely due to more mature technology, with SLC4E being an exception due to payload mass limits.
- **Predictive Analysis:**
  - **Logistic Regression (LR):** Better at predicting successful landings (Class 1) than failed landings (Class 0), with an overall accuracy of 81%. It identifies 56% of true negatives, missing several instances.
  - **Support Vector Machine (SVM):** Similar performance to LR, excelling in predicting successful landings (Class 1), with 81% accuracy and 56% true negative identification.
  - **Decision Tree (DT):** Lowest accuracy at 71%, with better performance on successful landings (83%) compared to failed landings (56%).
  - **K-Nearest Neighbors (KNN):** Matches LR and SVM with 81% accuracy but also struggles with true negatives.

**Conclusion:** LR, SVM, and KNN models show similar accuracy, with Decision Tree performing the weakest.





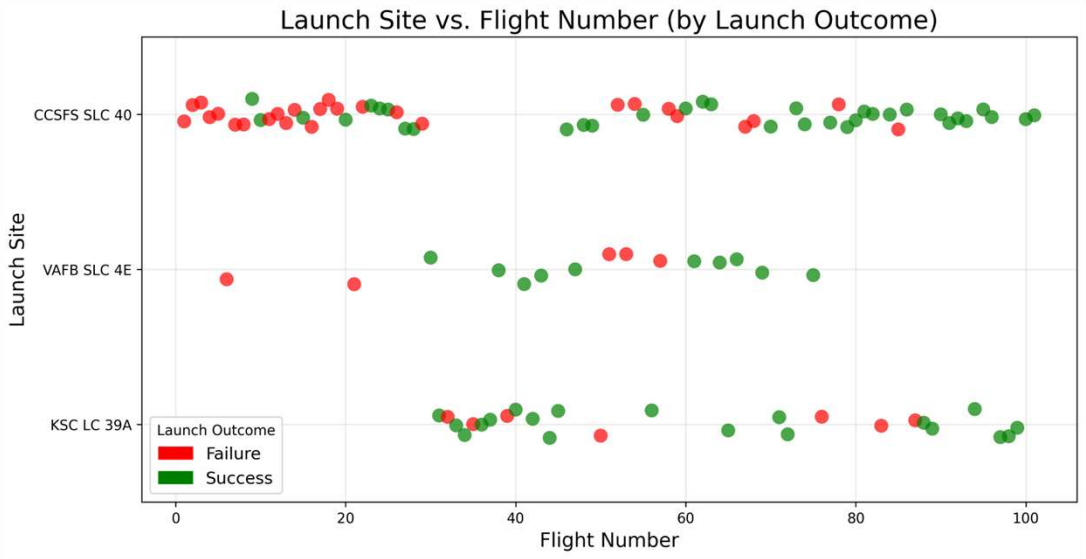
Section 2

# Insights drawn from EDA

# Flight Number vs. Launch Site

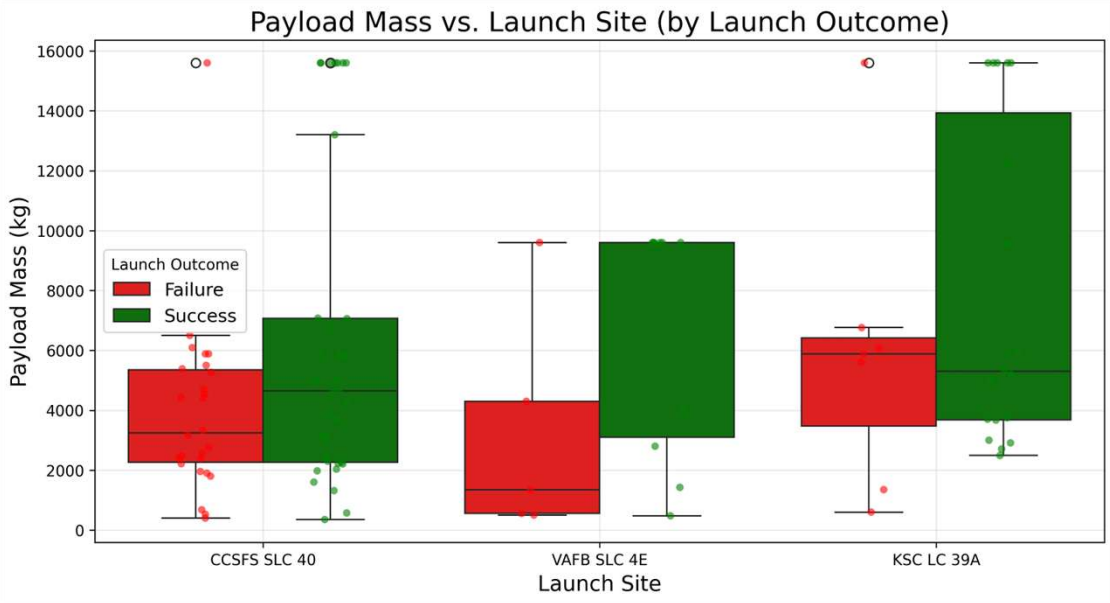
- **KSC LC 39A:** As the most recently utilized launch site, KSC LC 39A has demonstrated a high success rate for missions. This could be attributed to its modern infrastructure and advanced operational processes, positioning it as one of the most reliable sites.
- **CCSFS SLC 40:** Despite being the oldest operational site, CCSFS SLC 40 has experienced the highest number of failed missions compared to others. However, there have been improvements over time, suggesting a trajectory towards enhanced reliability.
- **VAFB SLC 4E:** This site has hosted fewer missions overall, possibly due to lower utilization or specific mission requirements that limit its use frequency.

**Summary:** KSC LC 39A leads in mission success rates, CCSFS SLC 40 shows potential improvement despite historical challenges, and VAFB SLC 4E operates with less activity. Each site's performance is influenced by factors such as age, infrastructure, and specific operational needs.



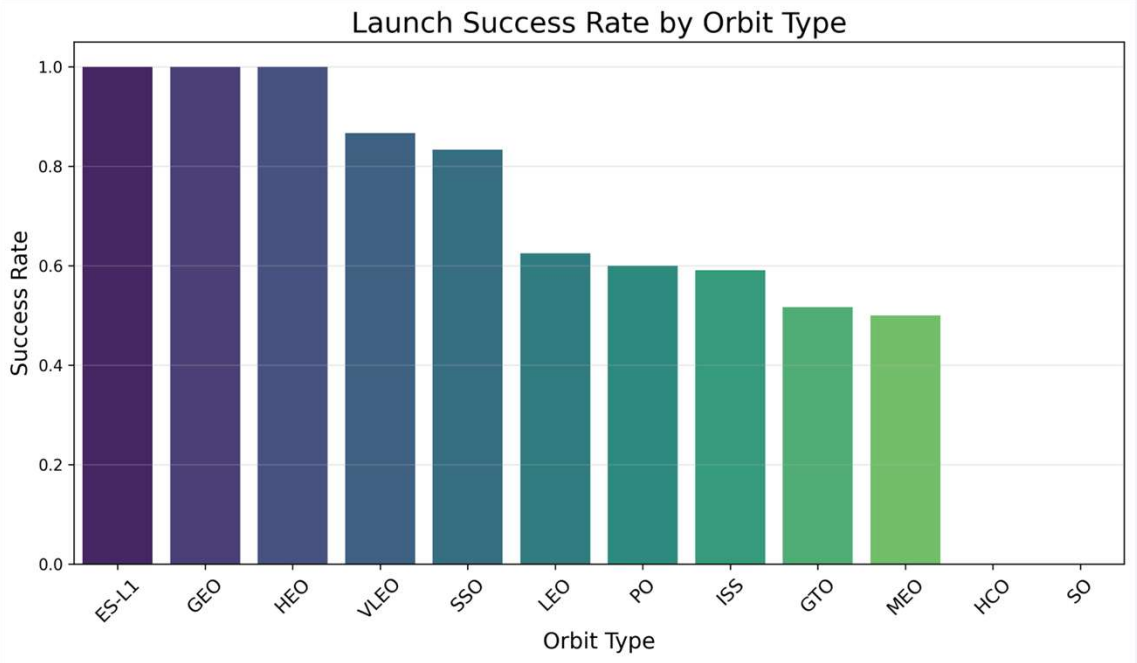
# Payload vs. Launch Site

- SLC4E does not conduct launches with payloads exceeding 10,000 kg, unlike other launch sites which do.
- Heavier payloads experience fewer failures, likely due to technological advancements enabling more efficient handling of larger masses.
- t CCSFS SLC 40 and KSC LC 39A, higher payload masses are treated as outliers, impacting both successful and failed landing missions.



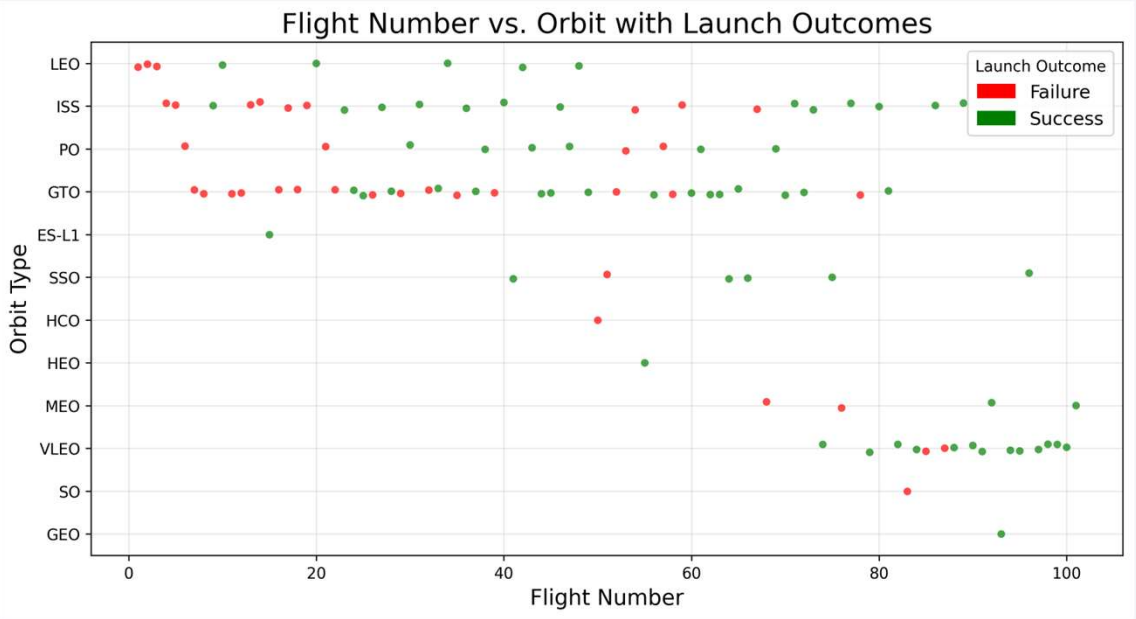
# Success Rate vs. Orbit Type

- Missions destined for ES-L1, GEO, and HEO orbits exhibit higher success rates.
- In contrast, missions targeting HCO (Heliocentric Orbit) and SO (Sun-Oriented) destinations show lower success rates.



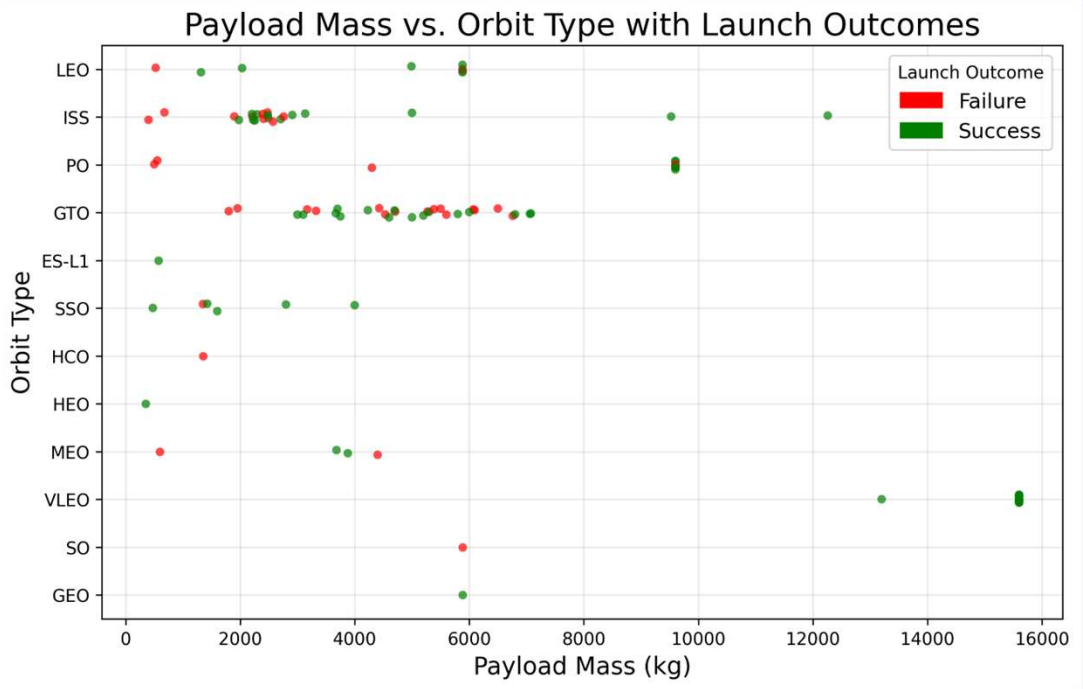
# Flight Number vs. Orbit Type

- Analysis reveals that missions to LEO (Low Earth Orbit) demonstrate higher success rates in achieving successful landings as flight numbers increase. These LEO missions also exhibit a favorable success-to-failure ratio.
- In contrast, missions to GTO (Geostationary Transfer Orbit) do not show any discernible pattern linking mission outcomes to flight number or payload mass



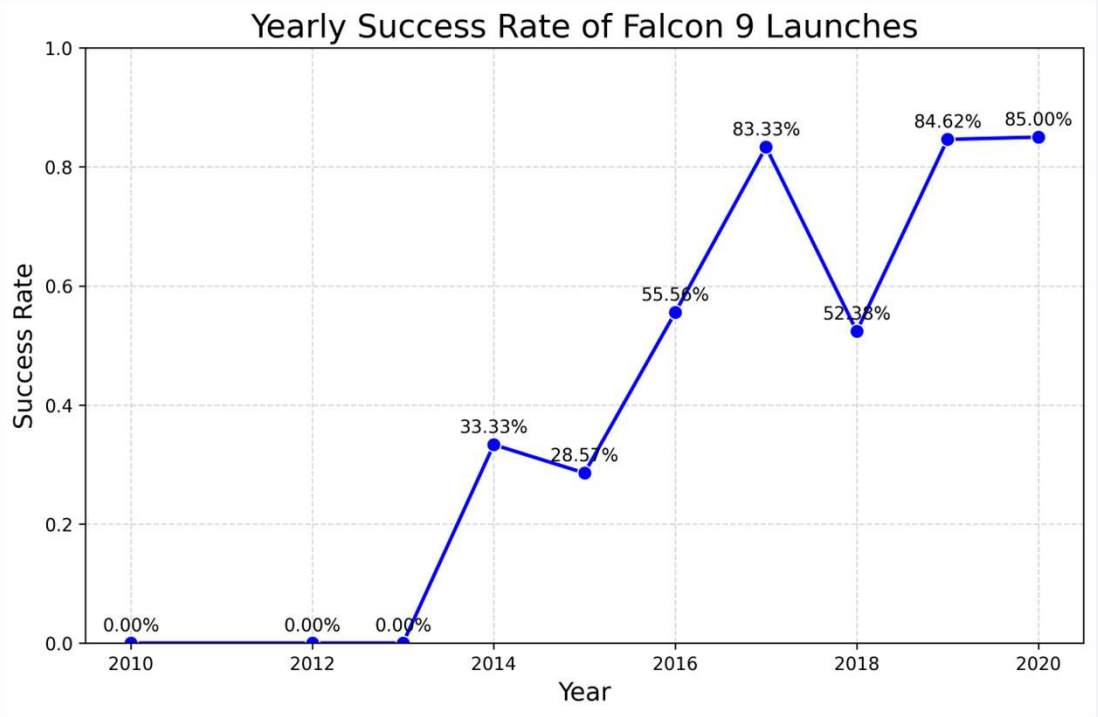
# Payload vs. Orbit Type

- Heavier payloads exhibit better success rates, potentially reflecting technological advancements or maturity in handling larger missions.
- However, GTO (Geostationary Transfer Orbit) missions do not show a consistent relationship between payload mass and mission outcomes.



# Launch Success Yearly Trend

- The chart on SpaceX mission success rates reveals an upward trend from 2013 onwards, with notable dips in 2015 and 2018. These decreases likely reflect specific challenges or incidents during those years, such as rocket explosions or technical difficulties. Despite these setbacks, the overall improvement suggests that SpaceX is enhancing its technology and operational reliability over time.



# All Launch Site Names

- **SELECT** Clause:
  - `DISTINCT(LaunchSite)` This part selects unique instances of the 'LaunchSite' column, eliminating duplicates.

- **FROM** Clause:
  - 'SPACEXTBL' Indicates that data is being retrieved from the SPACEXTBL table.

**Purpose**: The primary goal is to identify all unique 'LaunchSite locations present in the SPACEXTBL table. This could be useful for understanding operational spread or specific mission characteristics.

```
%%sql
  SELECT DISTINCT(LaunchSite)
  FROM SPACEXTBL;
```

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

| LaunchSite |
| --- |
| CCSFS SLC 40 |
| VAFB SLC 4E |
| KSC LC 39A |

# Launch Site Names Begin with ‘CCS'

- **SELECT** Clause:
  - '*' This wildcard selects all columns from the SPACEXTBL table.
- **FROM** Clause:
  - SPACEXTBL Indicates that the query is accessing data from the SPACEXTBL table.
- **WHERE** Clause:
  - 'CCS' within a LIKE clause:
    - LIKE '%CCS%' This filters records to include only those where the 'LaunchSite' field contains 'CCS'.
- **LIMIT** Clause:
  - 5 Limits the number of returned records to five, ensuring only top results are displayed.

Purpose: The primary goal is to retrieve a limited set of records from the SPACEXTBL table where the 'LaunchSite' is associated with 'CCS'. This could be useful for identifying specific missions or operations conducted at 'CCS'.

```
[ ] %%sql
    SELECT *
    FROM SPACEXTBL
    WHERE LaunchSite LIKE '%CCS%'
    LIMIT 5;
```

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

| Date | FlightNumber | BoosterVersion | PayloadMass | Orbit | LaunchSite | Longitude | Latitude | Outcome |
|---|---|---|---|---|---|---|---|---|
| 2010-06-04 | 1 | Falcon 9 | 5882.6900000000005 | LEO | CCSFS SLC 40 | -80.577366 | 28.5618571 | None None |
| 2010-12-08 | 2 | Falcon 9 | 5882.6900000000005 | LEO | CCSFS SLC 40 | -80.577366 | 28.5618571 | None None |
| 2012-05-22 | 3 | Falcon 9 | 525.0 | LEO | CCSFS SLC 40 | -80.577366 | 28.5618571 | None None |
| 2012-10-08 | 4 | Falcon 9 | 400.0 | ISS | CCSFS SLC 40 | -80.577366 | 28.5618571 | None None |
| 2013-03-01 | 5 | Falcon 9 | 677.0 | ISS | CCSFS SLC 40 | -80.577366 | 28.5618571 | None None |

# Total Payload Mass

- **SELECT** Clause:
  - SUM(PayloadMass) AS 'Total Payload Mass' computes the aggregated sum of the 'PayloadMass' column for all records in the SPACEXTBL table.
- **FROM** Clause:
  - SPACEXTBL Indicates that data is being retrieved from the SPACEXTBL table.

**Purpose**: The primary goal of this query is to determine the total 'PayloadMass' across all records in the database, providing a measure of overall payload capacity.

```
query = """
SELECT SUM(PayloadMass) AS 'Total Payload Mass'
FROM SPACEXTBL
"""
print(pd.read_sql(query, conn))
```

```
   Total Payload Mass
0           594151.69
```

# Average Payload Mass by F9 v1.1

- **SELECT** Clause:
  - 'AVG(PayloadMass) AS 'Average Payload Mass' calculates the average value of the PayloadMass column for records where the 'BoosterVersion' is 'Falcon 9'.
- **FROM** Clause:
  - 'SPACEXTBL' Indicates that the query is accessing data from the SPACEXTBL table.
- **WHERE** Clause:
  - 'Falcon 9' filters results to include only records where the 'BoosterVersion' matches 'Falcon 9'.

**Purpose:** The primary goal of this query is to compute an average 'PayloadMass' for the Falcon 9 booster, potentially aiding in comparing different boosters based on their average payload capacities.

```
%%sql
  SELECT AVG(PayloadMass) AS 'Average Payload Mass'
  FROM SPACEXTBL
  WHERE BoosterVersion = 'Falcon 9';
```

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

| Average Payload Mass |
| --- |
| 5981.649897959183 |

# First Successful Ground Landing Date

- **SELECT** Clause:
  - Renames `Date` to `'First Success Date'`.
- **FROM** Clause:
  - Pulls data from the database table SPACEXTBL.
- **WHERE** Clause:
  - Filters records where `Class = 1`, indicating successful outcomes.
- **ORDER BY** Clause:
  - Sorts results chronologically by `Date` in ascending order.
- **LIMIT 1**:
  - Retrieves only one record, presumably the earliest date of a successful outcome (`'First Success Date'`).

**Purpose:** To identify the earliest date when a successful outcome (Class = 1) occurred. This provides insights into when successful operations began within the dataset.

```
[ ] query = """
    SELECT Date AS 'First Success Date'
      FROM SPACEXTBL
      WHERE Class = 1
      ORDER BY Date ASC
      LIMIT 1;"""

    print(pd.read_sql(query, conn))
```

```
  First Success Date
0         2014-04-18
```

# Successful Drone Ship Landing with Payload between 4000 and 6000

- **SELECT** Clause: Extracts `BoosterVersion` from `SPACEXTBL`.
- **FROM** Clause: Pulls data from the database table SPACEXTBL.
- **WHERE** Clause:
  - Filters records where `PayloadMass` is equal to the maximum `PayloadMass` within the same table.
  - Further filters to include only records where `PayloadMass` is between 4000 and 6000 AND `Class = 1`.

**Purpose:** To identify booster versions associated with specific payload masses (`PayloadMass` values between 4000 and 6000 where `Class = 1`). This helps in analyzing which booster configurations were used for successful outcomes (when `Class = 1`) within the defined mass range.

```
[ ]  query = """
     SELECT BoosterVersion
     FROM SPACEXTBL
     WHERE PayloadMass = (
       SELECT PayloadMass
       FROM SPACEXTBL
       WHERE PayloadMass > 4000 AND PayloadMass < 6000 AND Class = 1
     );"""

     print(pd.read_sql(query, conn))
```

```
  BoosterVersion
0       Falcon 9
```

# Total Number of Successful and Failure Mission Outcomes

- **Categorize Outcomes**:
  - The CASE Class statement categorizes records into 'Failure' (when `Class = 0`) and 'Success' (when `Class = 1`).
- **Calculate Percentage:**
  - Using ROUND(), the query calculates the percentage of each outcome relative to the total number of records in the table.
- **Count Occurrences:**
  - The COUNT(*) AS Occurrences adds a column showing how many times each outcome occurred.
- **Group by Class:**
  - Results are grouped by Class, ensuring we get one row per class value (0 or 1).
- **Sort by Count:** Implicit sorting orders the results from highest to lowest count, highlighting which outcome is more frequent.

**Purpose:** To provide a clear overview of how often each outcome (Success or Failure) occurred, offering insights into their relative frequencies and absolute counts within the dataset.

```
[30] query = """
     SELECT CASE Class
         WHEN 0 THEN 'Failure'
         WHEN 1 THEN 'Success'
       END AS Outcome,
       ROUND(CAST(COUNT(*) AS REAL) * 100 / (SELECT COUNT(*) FROM SPACEXTBL), 2) AS Percentage,
       COUNT(*) AS Occurrences
     FROM SPACEXTBL
     GROUP BY Class;
     """
     print(pd.read_sql(query, conn))
```

```
    Outcome  Percentage  Occurrences
0  Failure       37.62           38
1  Success       62.38           63
```

# Boosters Carried Maximum Payload

- **SELECT** Clause:
  - DISTINCT('BoosterVersion') AS Maximum Load Booster: Retrieves unique booster versions and renames them for clarity.
- **FROM** Clause:
  - Pulls data from the database table SPACEXTBL.
- **WHERE** Clause:
  - Filters records where 'PayloadMass' equals the maximum value of 'PayloadMass' in the table.

Purpose:

- To identify booster configurations that achieved the highest payload mass, useful for performance analysis or optimization.
- This query efficiently pinpoints which booster versions are linked to peak performance, aiding in understanding the maximum capacities and configurations within the dataset.

```
[ ]  %%sql
     SELECT DISTINCT(BoosterVersion) AS 'Maximum Load Booster'
     FROM SPACEXTBL
     WHERE PayloadMass = (
       SELECT MAX(PayloadMass)
       FROM SPACEXTBL
     );
```

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

| Maximum Load Booster |
| --- |
| Falcon 9 |

# 2015 Launch Records

- **SELECT** Clause:
  - Extracts 'Month Name' by converting date substring into month number and mapping it to corresponding month name (e.g., 01='Jan', ..., 12='Dec'; else='Other').
  - Includes 'Date', 'Outcome', 'BoosterVersion', and 'LaunchSite'.

- **FROM** Clause:
  - Sources data from database table SPACEXTBL.

- **WHERE** Clause:
  - Filters records where 'Outcome' is specifically 'False ASDS'.
  - Further filters to include only dates in the year 2015.

- **ORDER BY** Clause:
  - Sorts results chronologically by 'Date'.

**Purpose**:
- Analyzes failures ('False ASDS') occurring in 2015.
- Provides insights into specific incidents and patterns during that timeframe.
- Useful for trend analysis or detailed incident reporting.

```
[ ] query = """
    SELECT CASE CAST(SUBSTR(Date, 6, 7) AS INTEGER)
        WHEN 01 THEN 'Jan'
        WHEN 02 THEN 'Feb'
        WHEN 03 THEN 'Mar'
        WHEN 04 THEN 'Apr'
        WHEN 05 THEN 'May'
        WHEN 06 THEN 'Jun'
        WHEN 07 THEN 'Jul'
        WHEN 08 THEN 'Aug'
        WHEN 09 THEN 'Sep'
        WHEN 10 THEN 'Oct'
        WHEN 11 THEN 'Nov'
        WHEN 12 THEN 'Dec'
        ELSE 'Other'
      END AS 'Month Name',
      Date, Outcome, BoosterVersion, LaunchSite
    FROM SPACEXTBL
    WHERE Outcome = 'False ASDS'
      AND CAST(SUBSTR(Date, 0, 5) AS INTEGER) = 2015
    ORDER BY Date ASC;
    """
    print(pd.read_sql(query, conn))
```

```
  Month Name        Date     Outcome BoosterVersion     LaunchSite
0        Jan  2015-01-10  False ASDS       Falcon 9  CCSFS SLC 40
1        Apr  2015-04-14  False ASDS       Falcon 9  CCSFS SLC 40
```

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

- **SELECT** Clause:
  - Date: Extracts unique dates within the specified range.
  - Outcome: Categorizes each record as either 'Success' or 'Failure'.
  - COUNT(Outcome) AS Occurrences: Calculates how many times each outcome occurred.
- **FROM** Clause:
  - SPACEXTBL: The database table containing date and outcome records.
- **WHERE** Clause:
  - Filters records to include only those within the specified date range (`2010-06-04` to `2017-03-20`).
- **GROUP BY** Clause:
  - Groups the results by each unique outcome (either 'Success' or 'Failure').
- **ORDER BY** Clause:
  - Sorts the grouped outcomes by their count in descending order, showing which outcome occurred more frequently.

**Purpose:** To provide a clear overview of how often each outcome (success or failure) occurred between June 4, 2010, and March 20, 2017. This helps in analyzing trends and performance over the specified period.

```
[32] %%sql
     SELECT Date, Outcome, COUNT(Outcome) AS Occurrences
     FROM SPACEXTBL
     WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
     GROUP BY Outcome
     ORDER BY COUNT(Outcome) DESC;
```

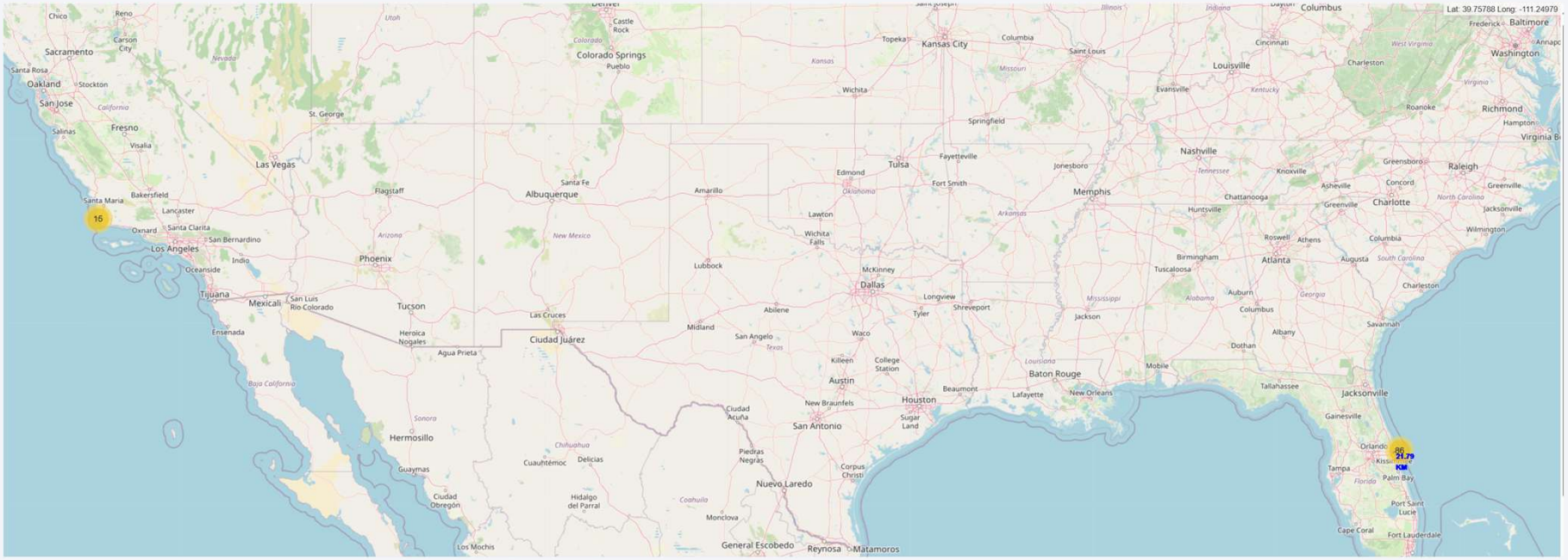
```
 * sqlite:///my_data1.db
Done.
```

| Date | Outcome | Occurrences |
|---|---|---|
| 2010-06-04 | None None | 12 |
| 2016-04-08 | True ASDS | 5 |
| 2015-01-10 | False ASDS | 5 |
| 2015-12-22 | True RTLS | 3 |
| 2014-04-18 | True Ocean | 3 |
| 2015-06-28 | None ASDS | 2 |
| 2013-09-29 | False Ocean | 2 |

Section 3

# Launch Sites Proximities Analysis

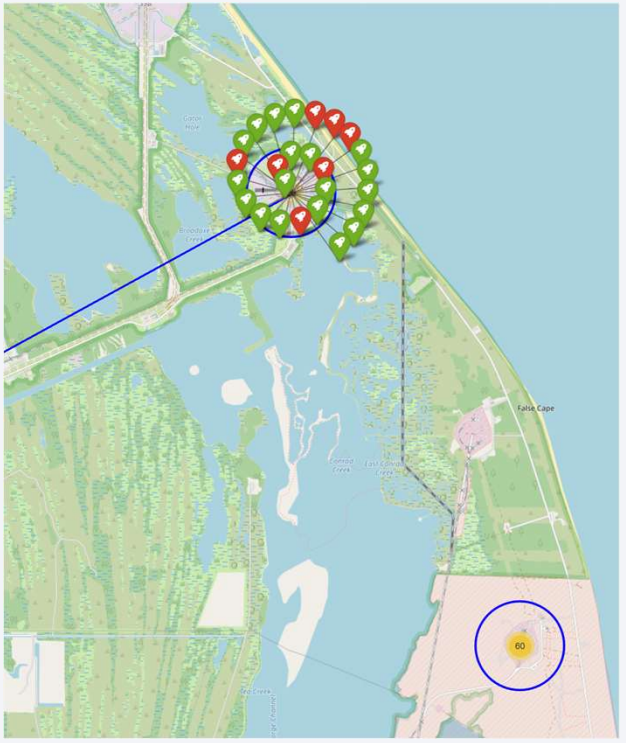
# Folium Map – Launch Sites




- The map displays three launch sites, with two located on the east coast and one on the west coast
- When you move the mouse over any point, the top-right corner of the screen shows the latitude and longitude coordinates for that location.
- You can also view the total number of launches that have taken place from each coastal site at the current zoom level

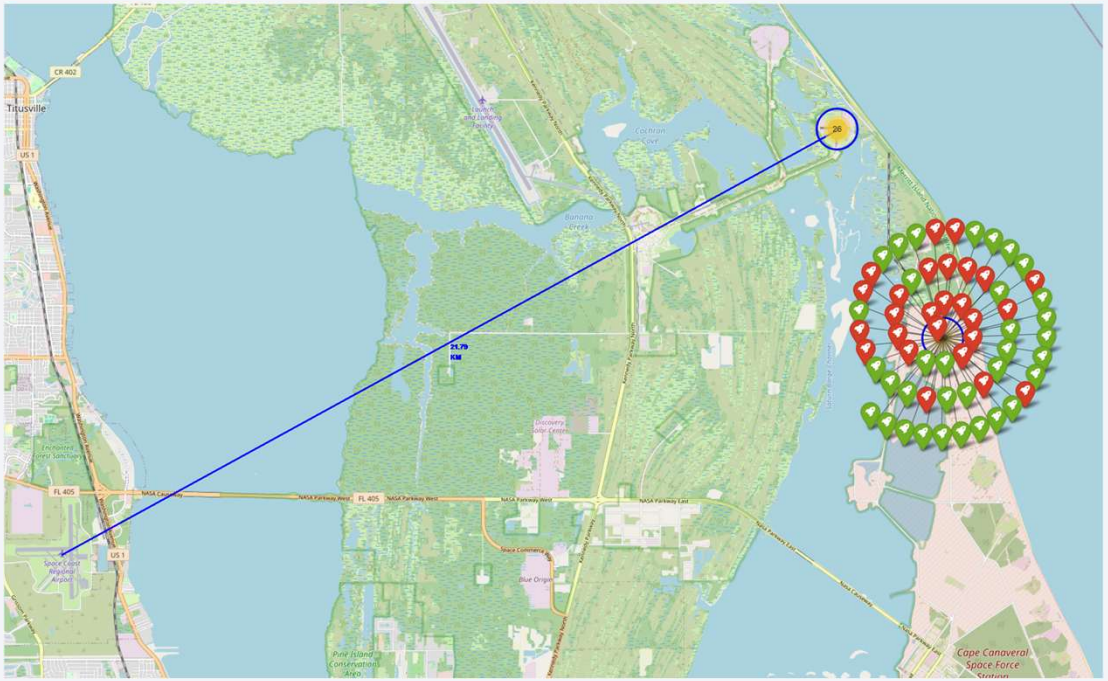
# Folium Map – Launch Outcomes

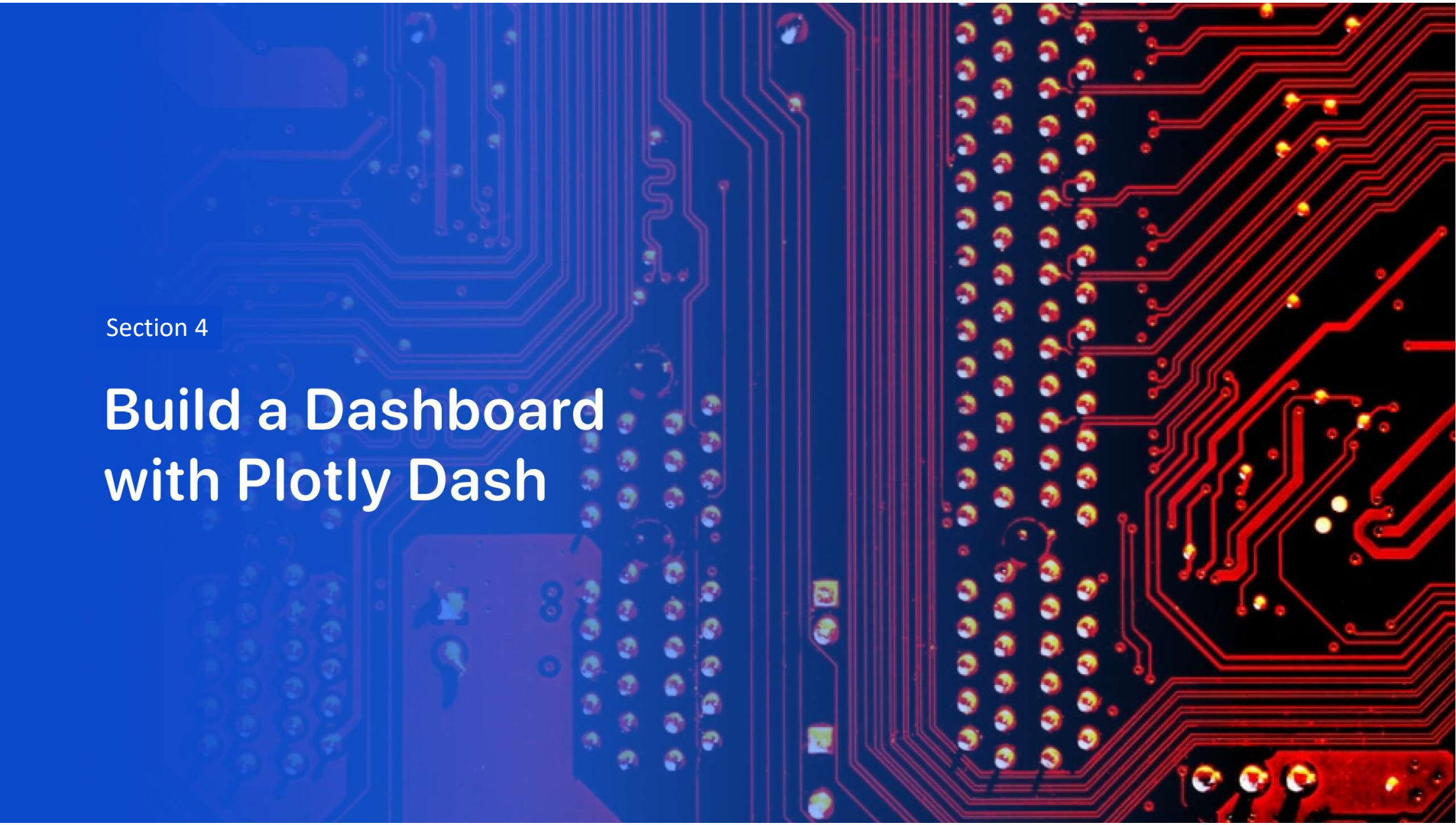
- The map shows two closely located launch sites on the east coast.
- Mission outcomes are indicated by colored icons: green signifies success, while red indicates failure.
- Each launch site is highlighted with a blue circle on the map.



# Folium Map – Distance Tool

- A straight line on the map connects one launch site to the nearby Space Coast Regional Airport.
- The distance between these two points is 21.79 kilometers.

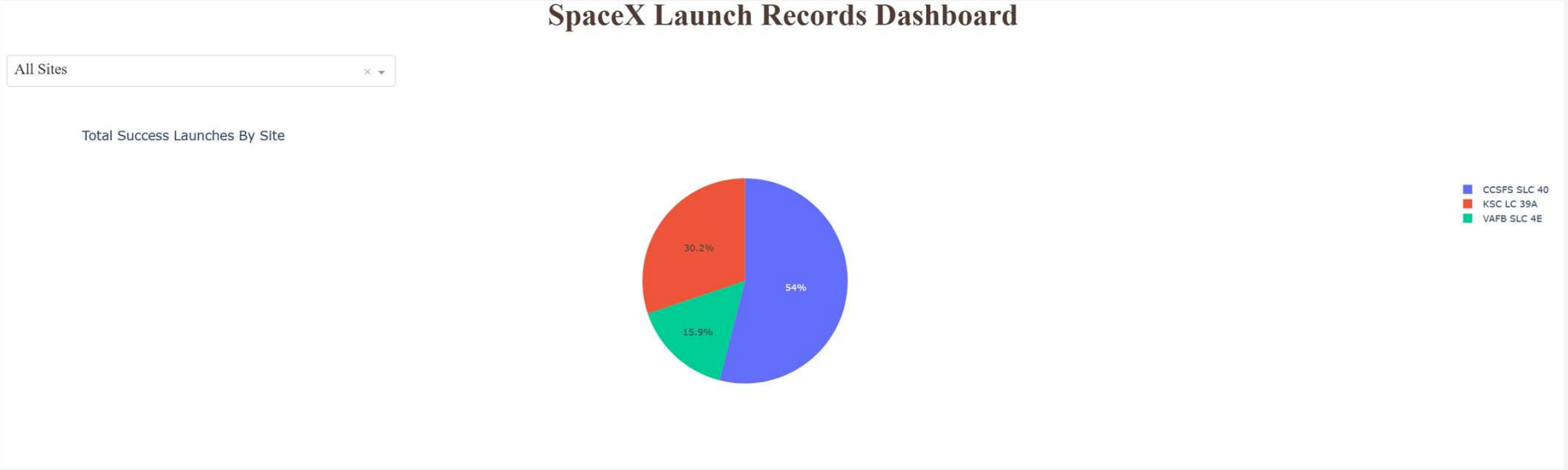




Section 4

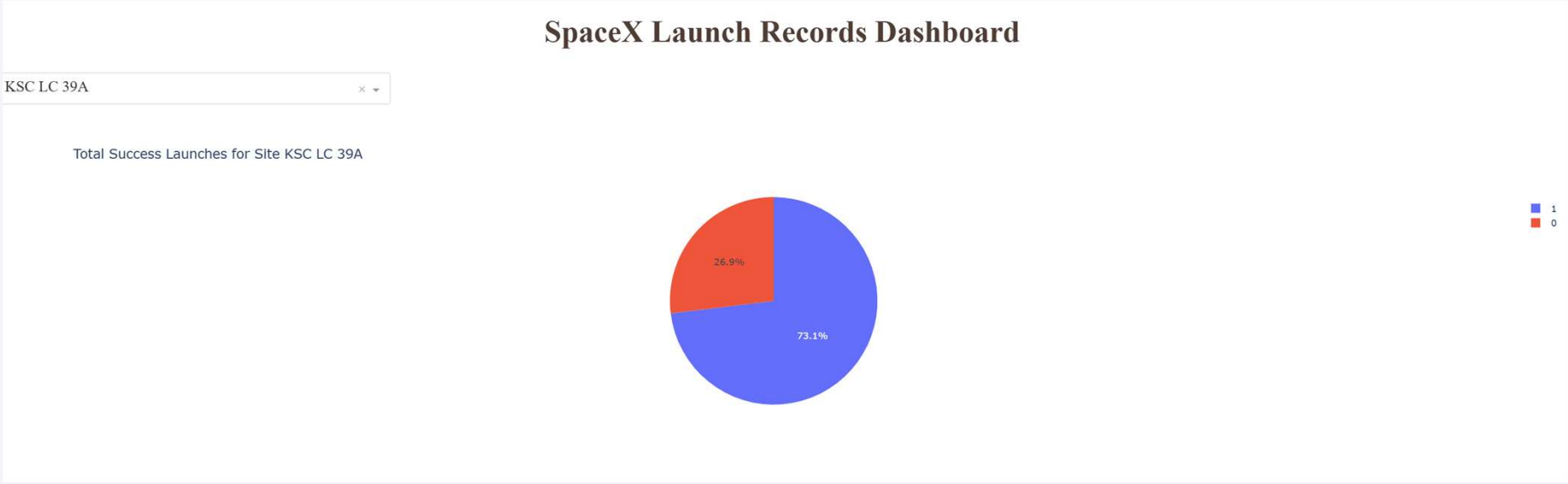
# Build a Dashboard with Plotly Dash

# Dash Intecractive – Launches Proportion



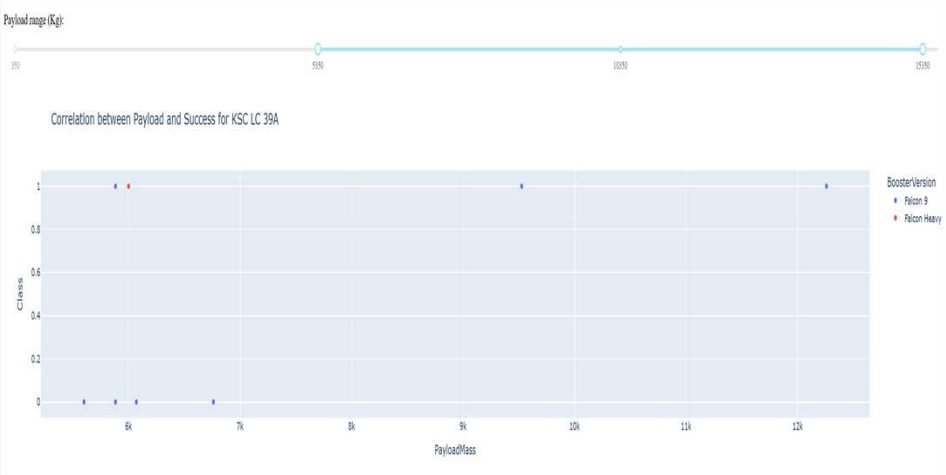

- The pie chart displays the percentage distribution of launches conducted at each site.
- To delve deeper into the details, there's a dropdown menu located at the top left corner that enables users to select individual launch sites. The legend on the right side provides a clear indication of which segments correspond to each site.
- Furthermore, it is possible to analyze the success versus failure proportion for each specific site. Additionally, selecting all sites allows users to view the overall distribution of launches across multiple locations. This interactive functionality enhances the ability to explore and understand launch data effectively.

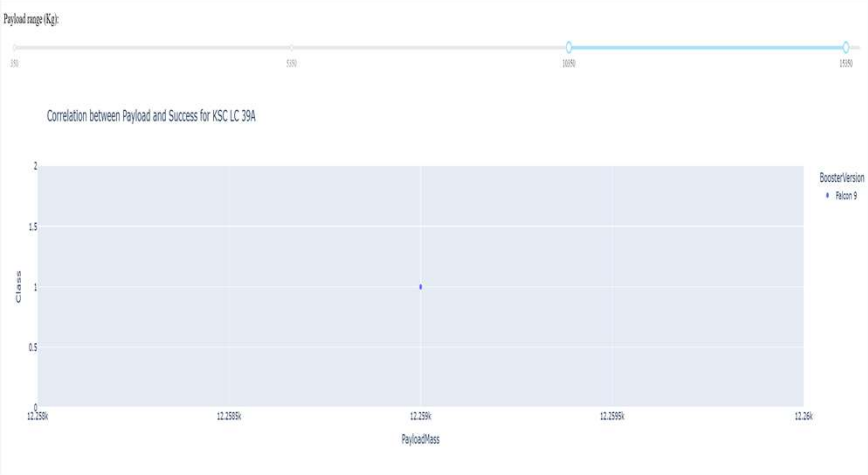
# Launch Site with higuest Successful Missions




- The dropdown menu on the top left allows you to choose between the three launch sites displayed in the legend on the right side of the chart.
- By selecting a specific site, you can view detailed insights into its performance. The pie chart illustrates the proportion of successful versus failed launches for each selected site. For instance, KSC LC 39A boasts a success rate of 73.1%. This interactive feature enhances your ability to explore and understand the launch data more effectively.

# Payload Mass vs Mission Outcome by Booster Version




- **Interactive Slider with Scatterplot:**
  - A slider allows users to adjust the payload mass range, filtering which data points appear on the scatterplot. This enables focused analysis on specific payload sizes.
- **Rocket Type and Payload Mass:**
  - Missions with higher payloads are predominantly executed by the Falcon 9 booster.
  - The Falcon Heavy booster is utilized for missions carrying payloads around 6000 kg.
- **Data Visualization Benefits:**
  - This approach offers an engaging way to identify trends and patterns in the data, such as correlations between rocket type and payload capacity.
- **Key Points:**
  - The slider filters the scatterplot based on payload mass.
  - Falcon 9 is associated with higher payload missions.
  - Falcon Heavy is used for payloads around 6000 kg.
  - This interactive method aids in understanding mission patterns and success rates.

Section 5

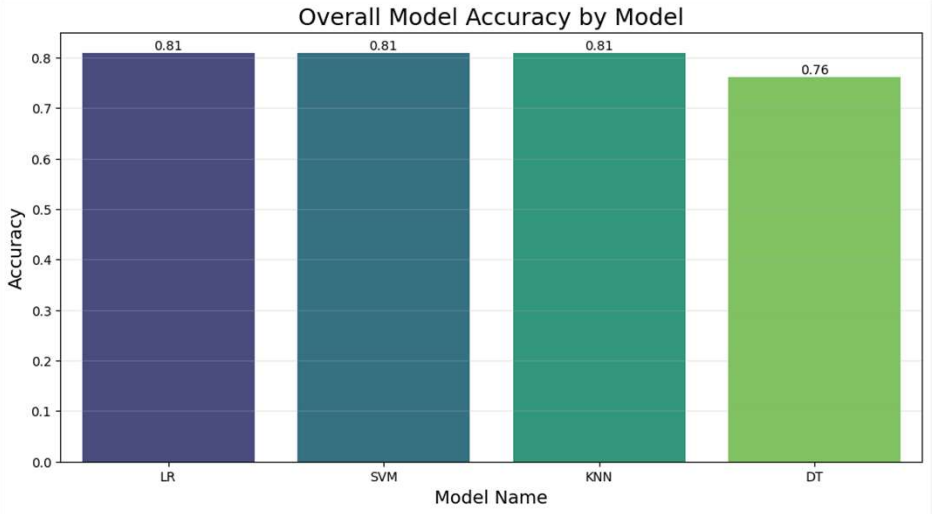
# Predictive Analysis (Classification)

# Classification Accuracy

- The highest test accuracy is achieved by three models: Logistic Regression, Support Vector Machine (SVM), and k-Nearest Neighbors (KNN). Among these, Logistic Regression stands out for its superior training efficiency. However, it is crucial to evaluate additional metrics such as recall and precision to comprehensively assess model performance.

- Notably, the Decision Tree model demonstrates the lowest accuracy in this comparison. This underperformance may be attributed to several factors, including overfitting or insufficient complexity in capturing the underlying patterns of the data. Exploring hyperparameter tuning or more sophisticated algorithms could potentially enhance its predictive capabilities.

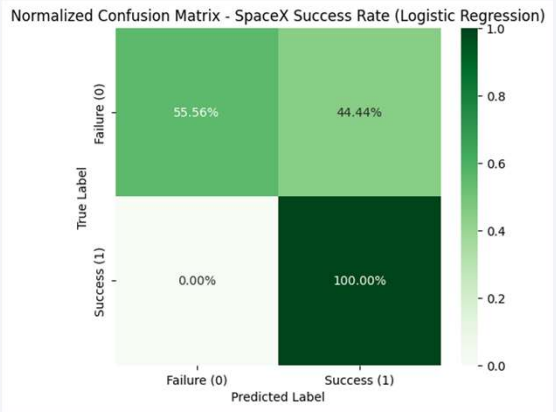
In summary, while Logistic Regression, SVM, and KNN show comparable accuracy, their differing strengths in training speed and performance metrics highlight the importance of selecting models based on specific operational needs and desired outcomes.

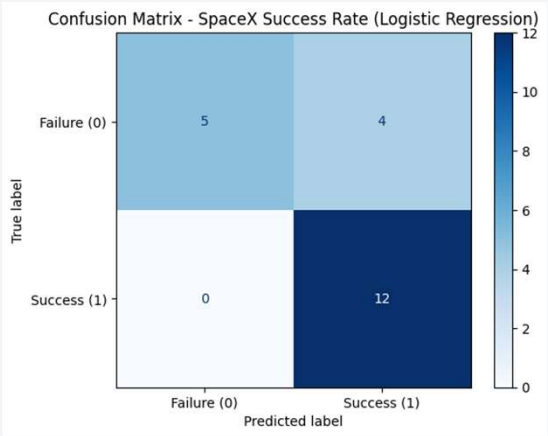


# Confusion Matrix

- **Precision:**
  - For "Landing Failure," the precision is 1, meaning that when the model predicts a failure, it is always correct.
  - For "Successful Landing," the precision is 0.75, suggesting that the model correctly identifies successful landings 75% of the time.
- **Recall:**
  - The recall for "Landing Failure" is 0.56, meaning that the model only detects about half of actual failures.
  - For "Successful Landing," the recall is 1, indicating that the model perfectly identifies all successful landings.
- **F1 Scores:**
  - The F1 score for "Successful Landings" is 0.86, reflecting a strong balance between precision and recall.
  - The F1 score for "Failure Landings" is 0.71, which is lower than the other class, indicating less effectiveness in identifying failures.

These metrics highlight that while the model performs well in detecting successful landings, there is room for improvement in accurately predicting failures. This imbalance could be due to an imbalanced dataset or inherent complexities in distinguishing between failure cases.



# Conclusions

- **High Accuracy Across Models:** Logistic Regression (LR), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN) achieve a high accuracy of approximately 81%, indicating strong performance in predicting mission outcomes.
- **Precision in Identifying Failures:** All models exhibit perfect precision for class 0 ("failure"), demonstrating confidence in detecting real failures with no false positives.
- **Challenges in Detecting Successes:** Precision and recall for class 1 ("success") are around 75%, suggesting some uncertainty or overlap between success and failure cases, indicating room for improvement in accurately predicting successful landings.
- **Recall Disparities:** Models show a significant drop in recall for failures (around 56%), highlighting the need to enhance failure detection capabilities. Successes, however, are nearly perfectly recalled across all models except Decision Tree.
- **Model Training Efficiency:** Logistic Regression stands out for its training efficiency, compared to the other models, specially SVM.
- EDA, data cleaning, and model fine-tuning collectively ensure that our analyses are robust, reliable, and capable of delivering actionable insights into SpaceX launch outcomes.

# Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

**https://github.com/vjfrancob/DataScience/blob/main/Tech/SpaceX_Launch/SpaceX_Prediction.ipynb**

# Thank you!