IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Yury Cavalcante
4th - January - 2024

# Outline

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

# Executive Summary

- Summary of methodologies
- Summary of all results

# Introduction

- Project background and context
  - **Cost Advantage**: Falcon 9's launch cost is $62 million, significantly lower than competitors' average of $165 million.
  - **Innovation in Reusability**: Key savings due to Space X's ability to reuse the rocket's first stage.
  - **Project Goal**: Develop a machine learning model to predict the success of Falcon 9 first stage landings.
  - **Competitive Analysis**: Insightful for competitors aiming to bid against Space X.

- Problems you want to find answers
  - **Determinants of Success**: What factors most influence the success of Falcon 9's first stage landing?
  - **Feature Interactions**: How do different operational parameters interact to affect landing outcomes?
  - **Optimal Conditions**: Identifying conditions that maximize successful landing probabilities.

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
    - Data sourced from SpaceX API and Wikipedia (via web scraping).
  - Perform data wrangling
  - Cleaning, structuring, and preparing data for analysis and applied **one-hot encoding** to transform categorical features for model compatibility.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Construct classification models to predict launch outcomes based on historical data with techniques like grid search and cross-validation to fine-tune model parameters for optimal performance and Utilize metrics such as accuracy, precision, recall, and the ROC curve to thoroughly evaluate model effectiveness.

# Data Collection

- **SpaceX API Usage:**
  - Executed GET requests to retrieve launch data.
- **Data Decoding and Structuring:**
  - Transformed API response to JSON format.
  - Used .json_normalize() to convert JSON to a pandas DataFrame.
- **Data Cleaning and Preprocessing:**
  - Performed thorough data cleaning.
  - Identified and filled missing values.
- **Web Scraping for Additional Records:**
  - Utilized BeautifulSoup for scraping Falcon 9 launch records from Wikipedia.
  - Extracted and parsed HTML tables to form structured DataFrame.

# Data Collection – SpaceX API

- We utilized the GET request method to retrieve data from the SpaceX API, followed by data cleansing and fundamental data wrangling and formatting tasks.

- You can access the notebook detailing this process at this GitHub link: https://github.com/YUCAVALCANTE/Applied-Data-Science-Capstone---ibm/blob/main/001%20-%20Data%20Collection%20API.ipynb

```python
import requests
import pandas as pd
from pandas import json_normalize

# Step 1: Get request for rocket launch data using API
spacex_url = "https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)

# Step 2: Use json_normalize method to convert json result to dataframe
# Decode response content as json
static_json_df = response.json()

# Apply json_normalize
data = json_normalize(static_json_df)

# Step 3: We then performed data cleaning and filling in the missing values
# Here you would clean the data. The specific code for cleaning is not shown in the screenshot.
# An example could be replacing NaN values with a specified value or using interpolation, depending on context.

# For example, let's say we want to fill NaN values in 'PayloadMass' with the average:
if 'PayloadMass' in data.columns:
    average_payload = data['PayloadMass'].mean()
    data['PayloadMass'] = data['PayloadMass'].fillna(average_payload)

# Continue with the data cleaning process as required...
```

# Data Collection - Scraping

- We utilized web scraping to extract Falcon 9 launch records using BeautifulSoup.
- The data was then parsed into a structured table and transformed into a pandas DataFrame for analysis.
- You can find the notebook detailing this process at the following GitHub link: https://github.com/YUCAVALCANTE/Applied-Data-Science-Capstone---ibm/blob/main/002%20-%20Data%20Collection%20with%20Web%20Scraping.ipynb

```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Step 1: Apply HTTP GET method to request the Falcon 9 rocket launch page
static_url = "https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches"
response = requests.get(static_url)
print(response.status_code)  # Should print 200 if successful

# Step 2: Create a BeautifulSoup object from the HTML response
soup = BeautifulSoup(response.text, 'html.parser')

# Step 3: Verify the BeautifulSoup object by printing the page title
print(soup.title.text)

# Step 4: Extract all column names from the HTML table header
column_names = []
# Assuming 'first_launch_table' is the id or relevant identifier of the table containing launch records
launch_table = soup.find('table', {'id': 'first_launch_table'})
headers = launch_table.find_all('th')

for header in headers:
    column_name = header.text.strip()
    if column_name:  # Ensure column name is not empty
        column_names.append(column_name)

print(column_names)

# Steps 5 and 6: Parse the HTML table into a pandas DataFrame and export to CSV
# The code for these steps would depend on the structure of the HTML table
# As a placeholder, it might look something like this:

rows = launch_table.find_all('tr')
data = []
for row in rows:
    cols = row.find_all('td')
    cols = [ele.text.strip() for ele in cols]
    data.append([ele for ele in cols if ele])  # Get rid of empty values

df = pd.DataFrame(data, columns=column_names)
df.to_csv('spacex_launches.csv', index=False)
```

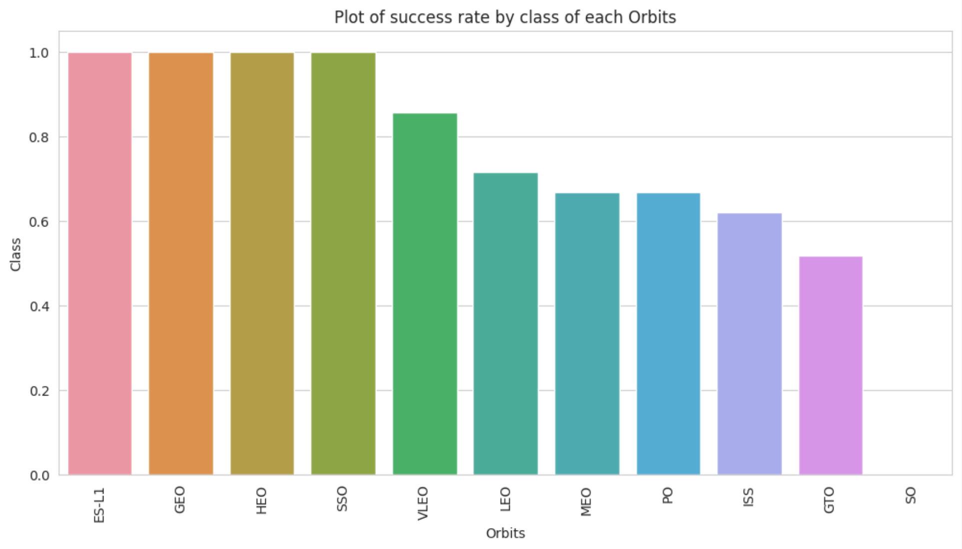
# Data Wrangling

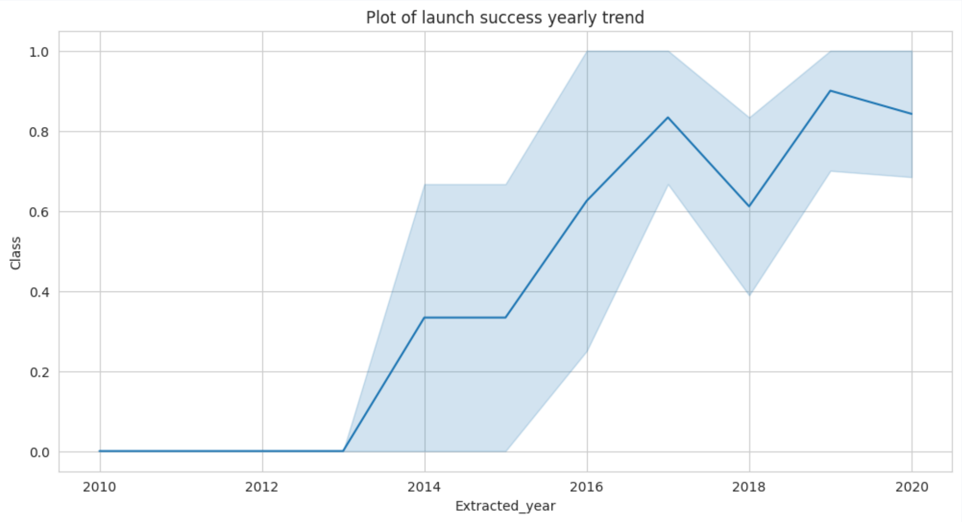
Our data wrangling process involved several key steps to prepare the dataset for exploratory data analysis and model training:

- **Data Acquisition**: We initiated the process by collecting data through API calls and web scraping methods, employing libraries like requests for API interactions and BeautifulSoup for parsing HTML content.
- **Data Transformation:** We used json_normalize to convert JSON responses into a structured pandas DataFrame. This facilitated the subsequent handling and manipulation of the data.
- **Exploratory Data Analysis (EDA):** We conducted an exploratory analysis to understand the dataset's characteristics, including the distribution and relationship of various features.
- **Feature Engineering:** We derived new informative features from existing data, such as calculating the total number of launches per site and categorizing orbits.
- **Label Determination:** The 'Outcome' column was processed to create a binary 'Landing Outcome' label, which served as our target variable for subsequent predictive modeling tasks.
- **Data Cleaning:** We addressed missing values, outliers, and anomalies to improve data quality, ensuring that the dataset was suitable for training machine learning models.
- **Exporting Data:** After preprocessing and cleaning, we exported the processed data to a CSV file for easy access and reproducibility of results.
- **Documentation and Sharing:** We documented the entire data wrangling process in Jupyter notebooks and shared them on GitHub for peer review and external reference.

- For a visual representation of our data wrangling process, a flowchart was created and included in the project documentation.
- You can find the Jupyter notebooks detailing each of these steps in our data wrangling process at the following GitHub link: https://github.com/YUCAVALCANTE/Applied-Data-Science-Capstone---ibm/blob/main/003%20-%20Data%20Wrangling.ipynb

# EDA with Data Visualization




## SPACEX LAUNCH DATA: INSIGHTS & TRENDS

- **Flight Number vs. Launch Site:** Scatter plots show the influence of experience on launch success.
- **Payload vs. Launch Site:** Scatter plots analyze the impact of payload mass on the success rate at various sites.
- **Success Rate by Orbit Type:** Bar charts reveal success rates and challenges associated with different orbits.
- **Flight Number vs. Orbit Type:** Scatter plots examine the relationship between mission count and orbit types, hinting at technological advancements.
- **Yearly Trend of Launch Success:** Line charts depict the progression and consistency of successful launches over time.
- **Visualization Choices:** Each chart was carefully selected to clearly convey key data relationships and to inform strategic planning for future missions.
- **Detailed Analysis**: Explore our full EDA and visualization work on GitHub: https://github.com/YUCAVALCANTE/Applied-Data-Science-Capstone---ibm/blob/main/005%20-%20EDA%20with%20Visualization%20Lab.ipynb

# EDA with SQL

- **Unique Launch Sites**: Retrieved distinct names of launch sites to understand the variety of launch locations.
- **Launch Sites with 'CCA'**: Selected records where launch site names begin with 'CCA' to filter specific geographic data.
- **Total Payload by NASA (CRS)**: Calculated the cumulative payload mass for missions contracted by NASA (CRS).
- **Average Payload for F9 v1.1**: Computed the average payload mass carried by the Falcon 9 version 1.1 booster.
- **First Successful Ground Pad Landing**: Identified the date of the first successful landing on a ground pad.
- **Successful Boosters for Drone Ship Landings**: Listed booster names with successful drone ship landings and specific payload mass range.
- **Mission Outcomes**: Counted the total number of successful and failed mission outcomes.
- **Heaviest Payload Missions**: Found booster versions that carried the maximum payload mass.
- **Failure Details in 2015**: Detailed failed landing outcomes, booster versions, and launch sites for missions in 2015.
- **Landing Outcomes Ranking**: Ranked landing outcomes by frequency between specified dates.

- For peer review and to access the complete set of SQL queries performed for the EDA, please refer to the Jupyter notebook available at the following GitHub link: https://github.com/YUCAVALCANTE/Applied-Data-Science-Capstone---ibm/blob/main/004%20-%20EDA%20with%20SQL.ipynb

# Build an Interactive Map with Folium

## Interactive Map Visualization with Folium

- **Launch Site Markers**: Marked all launch sites on the map to provide geographical context.
- **Outcome Indicators**: Added markers and circles to represent the success (Class 1) or failure (Class 0) of each launch, enhancing visual analytics.
- **Success Rate Visualization**: Utilized color-coded clusters to distinguish between launch sites based on their success rates, allowing for quick assessment of performance.
- **Proximity Analysis**: Calculated and displayed the distances from each launch site to nearby railways, highways, and coastlines to understand logistical advantages.
- **City Distances**: Evaluated the remoteness of launch sites from populated areas for safety and security considerations.
- ***Map Objects Rationale:*** Each object was integrated to facilitate a comprehensive understanding of launch site selection, operational success, and strategic positioning in relation to essential infrastructure and urban areas.
- *Further Exploration:* For an in-depth review of the interactive mapping analysis, visit the GitHub repository: https://github.com/YUCAVALCANTE/Applied-Data-Science-Capstone---ibm/blob/main/006%20-%20Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

# Build a Dashboard with Plotly Dash

- **Launch Site Distribution**: Incorporated pie charts to depict the distribution of total launches from various sites, enabling a clear visual comparison of launch site activity.
- **Booster Performance Analysis**: Integrated scatter plots to illustrate the relationship between launch outcomes and payload mass for different booster versions. This interaction helps to discern patterns and trends in booster performance relative to the mass they carry.
- **User Interactivity**: Added dropdowns, sliders, and other interactive components to allow users to filter and customize the data views. These interactions enhance user engagement and enable personalized data exploration.
- ***Rationale for Dashboard Elements:*** The chosen visualizations and interactive features are designed to provide intuitive insights into launch frequencies and efficiencies, as well as to facilitate a deeper understanding of the factors contributing to mission success.

# Predictive Analysis (Classification)

## MODEL DEVELOPMENT PROCESS

- **Data Preparation**: We began by loading the dataset into a NumPy array and a pandas DataFrame, followed by data transformation and normalization to prepare for machine learning.
- **Training and Testing**: We split the dataset into training and testing sets to ensure a robust evaluation of model performance.
- **Model Building**: Various machine learning models were constructed, including logistic regression, SVM, decision trees, and KNN, to identify baseline accuracies.
- **Hyperparameter Tuning**: Utilized GridSearchCV to systematically work through multiple combinations of parameter tunes, cross-validating as we went to determine the most effective settings.
- **Performance Evaluation**: We employed accuracy as the primary metric to assess each model, using the testing set to gauge generalization capabilities.
- **Feature Engineering and Tuning**: Enhanced model performance by engineering new features from existing data and refining algorithms.
- **Best Model Selection**: Through iterative tuning and evaluation, we identified the classification model that yielded the highest accuracy, signifying the best predictive performance.

- ***Key Phrases:*** Data Loading, Model Training, Hyperparameter Optimization, Accuracy Measurement, Feature Engineering, Best Model Identification.
- ***Flowchart:*** A flowchart would be created to visually represent the step-by-step process from data loading to final model selection.

- *Comprehensive Analysis and Code:* Explore our model development journey in detail on GitHub: https://github.com/YUCAVALCANTE/Applied-Data-Science-Capstone---ibm/blob/main/007%20-%20Machine%20Learning%20Prediction%20lab.ipynb

# Results

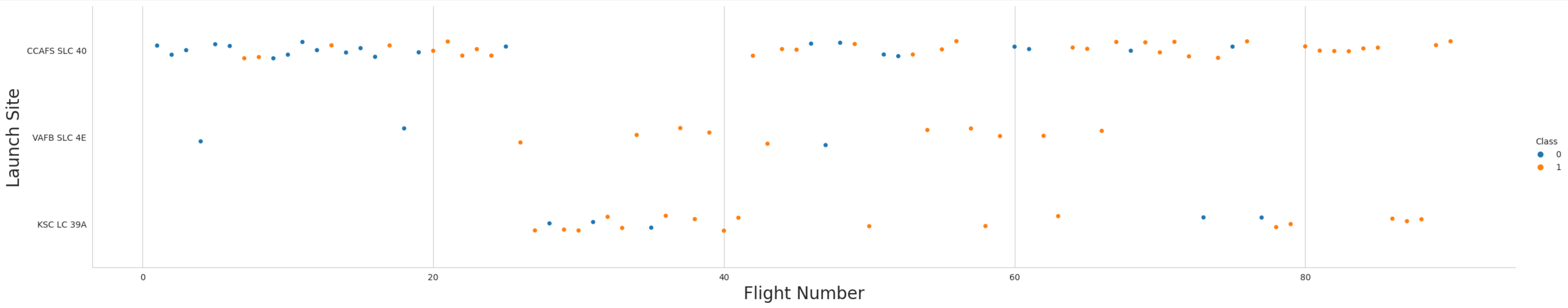
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

# Insights drawn from EDA

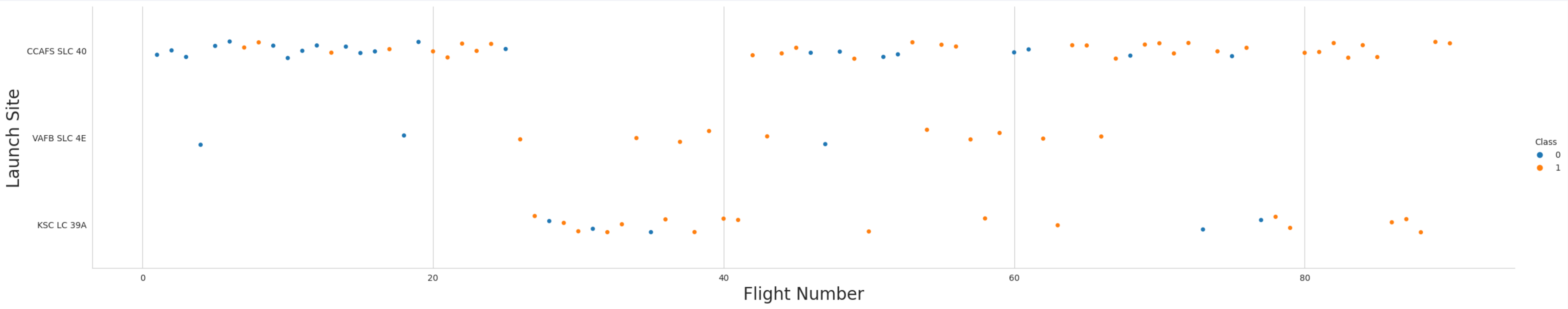
# Flight Number vs. Launch Site

- **Visualization**: Scatter plot with Flight Number (X-axis) and Launch Site (Y-axis).
- **Color Coding**:
  - Blue dots represent failed launches (Class 0).
  - Orange dots indicate successful launches (Class 1).
- **Insights**:
  - Increased flight numbers correlate with a higher success rate.
  - Suggests operational learning enhances success over time.
- **Launch Site Performance**:
  - Allows comparison of launch success across different sites.
  - Can identify sites with higher efficiency and reliability.



# Payload vs. Launch Site

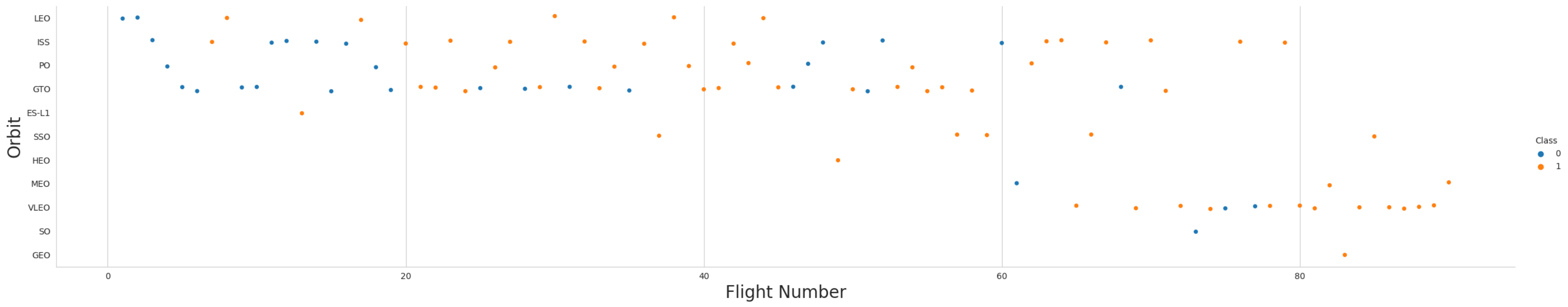
- **Visualization**: Scatter plot with Payload Mass (X-axis) against Launch Site (Y-axis).
- **Color Coding**:
  - Blue dots indicate failed launches (Class 0).
  - Orange dots indicate successful launches (Class 1).
- **Insights**:
  - Larger payload masses at CCAFS SLC 40 are associated with a higher success rate.
  - Demonstrates the site's capability to handle significant payloads effectively.
- **Impact**:
  - Highlights the importance of payload capacity in mission planning and success.
  - Suggests infrastructure and experience at CCAFS SLC 40 are conducive to successful launches.



# Success Rate vs. Orbit Type

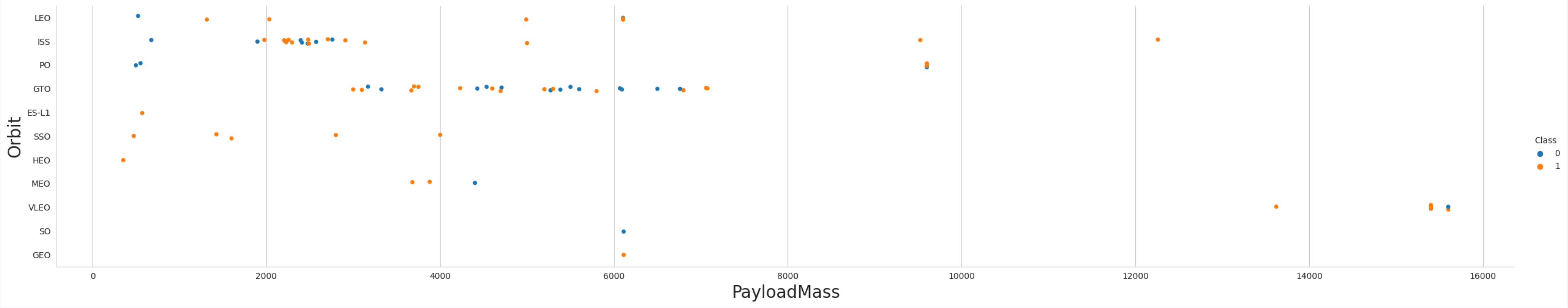
- **Visualization**: Bar chart comparing the success rates across different orbits.
- **High Success Rates**:
  - ES-L1, GEO, HEO, SSO, VLEO orbits display the highest success rates.
- **Orbit Insights**:
  - These orbits are indicative of reliable mission profiles within SpaceX's operations.
- **Strategic Implications**:
  - Understanding orbit-specific success rates can inform future mission planning and risk assessment.

# Flight Number vs. Orbit Type



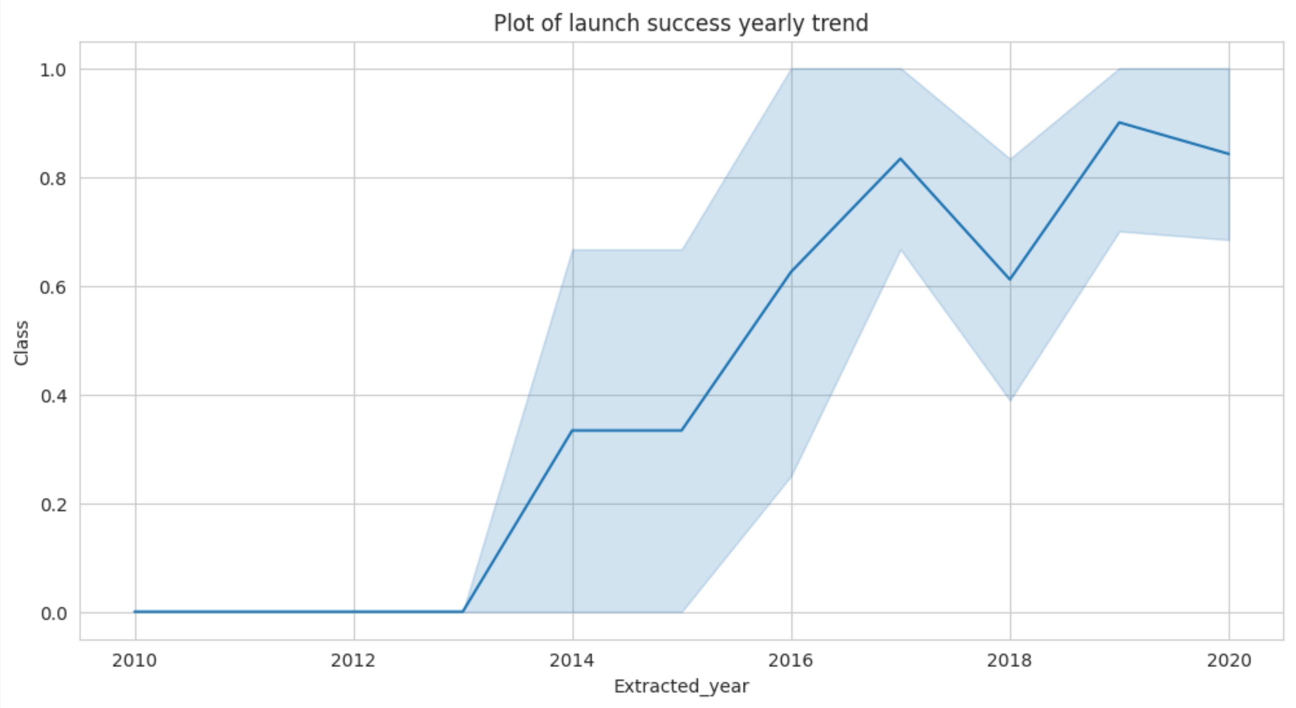

- **Visualization**: Scatter plot displaying the relationship between Flight Number and Orbit Type.
- **Orbit Insights**:
  - In Low Earth Orbit (LEO), a higher flight number correlates with increased success.
  - In Geostationary Transfer Orbit (GTO), success seems independent of flight number.
- **Implications**:
  - Suggests accumulated experience enhances success rates in certain orbits like LEO.
  - Indicates that factors other than experience may influence success in GTO orbits.
- **Operational Strategy**:
  - Insights can guide targeted improvements and resource allocation for specific orbit types.

# Payload vs. Orbit Type




- **Visualization**: Scatter plot with Payload Mass (X-axis) versus Orbit Type (Y-axis).
- **Observations**:
  - Heavier payloads are more likely to result in successful landings, particularly in Polar Orbit (PO), Low Earth Orbit (LEO), and International Space Station (ISS) missions.
- **Implications**:
  - Indicates the capability of SpaceX's rockets to deliver heavy payloads to specific orbits with high success.
  - May influence mission planning regarding payload allocations for different orbit types.

# Launch Success Yearly Trend




- **Visualization**: Line chart with Year (X-axis) and Average Success Rate (Y-axis).
- **Trend Observation**:
  - Steady increase in success rate from 2013 through 2020.
- **Performance Highlight**:
  - Demonstrates significant improvement in launch reliability over time.
- **Strategic Insights**:
  - The trend underscores the effectiveness of SpaceX's iterative learning and advancements in technology.

# All Launch Site Names

**IDENTIFYING UNIQUE SPACEX LAUNCH SITES**

- **SQL Query:** Utilized SELECT DISTINCT to extract unique launch site names.
- **Purpose:** To identify all different launch locations used by SpaceX.
- **Database Connection**: Connected to SQLite database my_data.db.
- **Query Execution:** Retrieved distinct names from SPACEXTBL.
- **Insight:** The distinct keyword effectively filters out duplicate entries, ensuring each launch site is listed only once.

```
con = sqlite3.connect("my_data1.db")  # Ensure this is the correct path to your database file

query = "SELECT DISTINCT Launch_Site FROM SPACEXTBL;"  # Replace 'SPACEXTBL' with your actual table name
result = pd.read_sql_query(query, con)
con.close()

print(result)
```

```
    Launch_Site
0   CCAFS LC-40
1   VAFB SLC-4E
2    KSC LC-39A
3  CCAFS SLC-40
```

# Launch Site Names Begin with 'CCA'

- **SQL Query:** Implemented a query with a LIKE clause to find records starting with 'CCA'.
- **Data Retrieval:** Fetched 5 records from the database where launch site names start with 'CCA'.
- **Purpose:** To isolate and analyze launches from CCAFS sites.
- **Query Result:**
    - Displayed a snippet of launch records including Date, Payload, and Outcome.
    - Confirms all selected records are from the CCAFS LC-40 launch site.

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

```
[ ] con = sqlite3.connect("my_data1.db")  # Ensure this is the correct path to your database file

    query = "SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5;"
    result = pd.read_sql_query(query, con)
    con.close()

    print(result)
```

```
         Date Time (UTC) Booster_Version  Launch_Site  \
0  2010-06-04   18:45:00  F9 v1.0  B0003  CCAFS LC-40
1  2010-12-08   15:43:00  F9 v1.0  B0004  CCAFS LC-40
2  2012-05-22    7:44:00  F9 v1.0  B0005  CCAFS LC-40
3  2012-10-08    0:35:00  F9 v1.0  B0006  CCAFS LC-40
4  2013-03-01   15:10:00  F9 v1.0  B0007  CCAFS LC-40

                                               Payload  PAYLOAD_MASS__KG_  \
0              Dragon Spacecraft Qualification Unit                  0
1  Dragon demo flight C1, two CubeSats, barrel of...                  0
2                                 Dragon demo flight C2                525
3                                          SpaceX CRS-1                500
4                                          SpaceX CRS-2                677

      Orbit         Customer Mission_Outcome      Landing_Outcome
0       LEO           SpaceX         Success  Failure (parachute)
1  LEO (ISS)  NASA (COTS) NRO         Success  Failure (parachute)
2  LEO (ISS)      NASA (COTS)         Success           No attempt
3  LEO (ISS)       NASA (CRS)         Success           No attempt
4  LEO (ISS)       NASA (CRS)         Success           No attempt
```

# Total Payload Mass

- **Objective:** Calculate the cumulative payload mass for NASA's Commercial Resupply Services.
- **SQL Query:** Executed an aggregate SUM function on the Payload_Mass_kg column.
- **Criteria:** Focused on missions where the customer is 'NASA (CRS)'.
- **Result:** The total payload mass delivered by SpaceX for NASA (CRS) is 45,596 kg.
- **Significance:** Highlights the substantial contribution of SpaceX to NASA's resupply missions.

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

```
[ ] con = sqlite3.connect("my_data1.db")  # Ensure this is the correct path to your database file

    query = "SELECT SUM(Payload_Mass__kg_) AS Total_Payload_Mass FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';"
    result = pd.read_sql_query(query, con)
    con.close()

    # Display the result
    print(result)
```

```
   Total_Payload_Mass
0               45596
```

# Average Payload Mass by F9 v1.1

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

```
[ ] con = sqlite3.connect("my_data1.db")
    query = "SELECT AVG(Payload_Mass__kg_) AS Average_Payload_Mass FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1';"
    result = pd.read_sql_query(query, con)
    con.close()
    print(result)
```

```
   Average_Payload_Mass
0                2928.4
```

- **Objective:** Determine the average payload mass delivered by the F9 v1.1 booster.
- **SQL Query:** Used the AVG function to calculate the mean payload mass.
- **Filter Applied:** Specific to missions with booster version 'F9 v1.1'.
- **Result:** The F9 v1.1 booster's average payload mass is 2,928.4 kg.
- **Significance:** Reflects the payload capacity and performance of the F9 v1.1 model.

# First Successful Ground Landing Date

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

*Hint:Use min function*

```
[ ]  con = sqlite3.connect("my_data1.db")
     query = "SELECT MIN(Date) AS First_Successful_Landing FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)';"
     result = pd.read_sql_query(query, con)
     con.close()
     print(result)
```

```
  First_Successful_Landing
0               2015-12-22
```

## FIRST SUCCESSFUL GROUND PAD LANDING

- **Goal:** Identify the date of the inaugural successful landing on a ground pad.
- **SQL Query:** Employed the MIN function to find the earliest date of success.
- **Criteria:** Filtered for landings with the outcome 'Success (ground pad)'.
- **Discovery:** The first successful ground pad landing occurred on December 22, 2015.
- **Historical Milestone:** Marks a significant achievement in SpaceX's reusable launch system development.

# Successful Drone Ship Landing with Payload between 4000 and 6000

### BOOSTERS WITH SUCCESSFUL DRONE SHIP LANDINGS AND SPECIFIC PAYLOAD MASS

- **Criteria:** Selected boosters that landed on a drone ship with payloads between 4000 and 6000 kg.
- **SQL Query:**
  - Applied a WHERE clause to filter for successful drone ship landings.
  - Used AND to specify the payload mass range criteria.
- **Query Result:** Identified boosters including F9 FT B1022, F9 FT B1026, and F9 FT B1021.2.
- **Implications:** Indicates the capability of these boosters to deliver significant payloads while achieving landing success.

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

```
con = sqlite3.connect("my_data1.db")
query = "SELECT Booster_Version FROM SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' AND Payload_Mass__kg_ > 4000 AND Payload_Mass__kg_ < 6000;"
result = pd.read_sql_query(query, con)
con.close()
print(result)
```

```
  Booster_Version
0     F9 FT B1022
1     F9 FT B1026
2  F9 FT  B1021.2
3  F9 FT  B1031.2
```

# Total Number of Successful and Failure Mission Outcomes

## MISSION OUTCOMES: SUCCESS VS. FAILURE

- **Objective:** Calculate the count of mission outcomes, distinguishing between successes and failures.
- **SQL Query:**
  - Used GROUP BY to aggregate results based on Mission_Outcome.
  - Employed the COUNT function to tally the number of occurrences.
- **Methodology:** No wildcard % was actually used in the provided query; the query simply groups and counts the different outcomes.

List the total number of successful and failure mission outcomes

```
[ ] con = sqlite3.connect("my_data1.db")
    query = "SELECT Mission_Outcome, COUNT(*) AS Count FROM SPACEXTBL GROUP BY Mission_Outcome;"
    result = pd.read_sql_query(query, con)
    con.close()
    print(result)
```

```
                Mission_Outcome  Count
0             Failure (in flight)      1
1                         Success     98
2                        Success       1
3  Success (payload status unclear)      1
```

# Boosters Carried Maximum Payload

- **Objective:** Identify booster versions that have transported the heaviest payloads.
- **SQL Query Strategy:**
  - Implemented a subquery to determine the maximum payload mass.
  - Selected booster versions equal to this maximum mass.
- **Query Result:** Listed multiple iterations of Falcon 9 boosters, such as B1048.4 and B1051.3.
- **Analysis:** Indicates these specific boosters are capable of carrying the most substantial payloads.

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
[ ] con = sqlite3.connect("my_data1.db")
    query = "SELECT Booster_Version FROM SPACEXTBL WHERE Payload_Mass__kg_ = (SELECT MAX(Payload_Mass__kg_) FROM SPACEXTBL);"
    result = pd.read_sql_query(query, con)
    con.close()
    print(result)
```

```
     Booster_Version
0      F9 B5 B1048.4
1      F9 B5 B1049.4
2      F9 B5 B1051.3
3      F9 B5 B1056.4
4      F9 B5 B1048.5
5      F9 B5 B1051.4
6      F9 B5 B1049.5
7     F9 B5 B1060.2
8     F9 B5 B1058.3
9      F9 B5 B1051.6
10     F9 B5 B1060.3
11    F9 B5 B1049.7
```

# 2015 Launch Records

## 2015 FAILED DRONE SHIP LANDING OUTCOMES

- **Query Focus:** Isolate drone ship landing failures with corresponding booster versions and launch sites from 2015.
- SQL Techniques:
  - Utilized WHERE clause with LIKE to specify 'Failure' outcomes.
  - Applied SUBSTR function to filter records by the year '2015'.
- **Query Result:** Identified two failures: one in January (01) and one in April (04), both at CCAFS LC-40.
- **Methodology:** The query was carefully crafted to extract precise information within a specific time frame and conditions.

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.

**Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.**

```
[ ] con = sqlite3.connect("my_data1.db")
    query = "SELECT SUBSTR(Date, 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure%' AND SUBSTR(Date, 1, 4) = '2015';"
    result = pd.read_sql_query(query, con)
    con.close()
    print(result)
```

```
  Month       Landing_Outcome Booster_Version  Launch_Site
0    01  Failure (drone ship)   F9 v1.1 B1012  CCAFS LC-40
1    04  Failure (drone ship)   F9 v1.1 B1015  CCAFS LC-40
```

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

- **Query Objective:** Quantify and rank the frequency of different landing outcomes within the specified period.
- **SQL Strategy:**
  - Filtered outcomes using WHERE clause for dates between 2010-06-04 and 2017-03-20.
  - Grouped the data by Landing_Outcome using GROUP BY.
  - Ordered the counts in descending order with ORDER BY.
- **Query Result:**
  - Most common outcome was 'No attempt' followed by successful drone ship and ground pad landings.
  - Less frequent were various forms of unsuccessful landings including controlled ocean landings and drone ship failures.
- **Insight:** Provides a clear understanding of landing success rates and the evolution of landing techniques over time.

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.

```
[ ] con = sqlite3.connect("my_data1.db")
    query = "SELECT Landing_Outcome, COUNT(*) AS Count FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY Count DESC;"
    result = pd.read_sql_query(query, con)
    con.close()
    print(result)
```

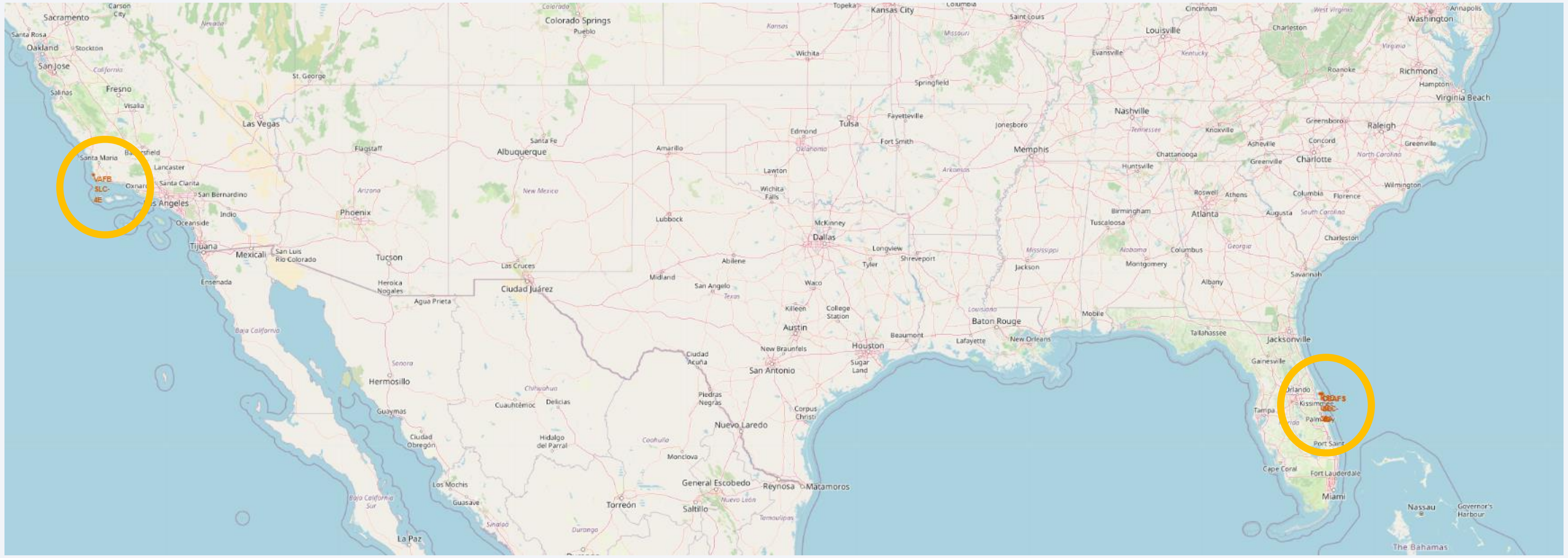
```
         Landing_Outcome  Count
0             No attempt     10
1     Success (drone ship)    5
2     Failure (drone ship)    5
3     Success (ground pad)    3
4       Controlled (ocean)    3
5     Uncontrolled (ocean)    2
6      Failure (parachute)    2
7   Precluded (drone ship)    1
```

Section 3

# Launch Sites Proximities Analysis

# Global Distribution of SpaceX Launch Sites

- **Location Markers:**
  - CCAFS SLC 40 at Cape Canaveral, Florida.
  - VAFB SLC 4E at Vandenberg Air Force Base, California.
- **Key Observations:**
  - Launch sites are strategically located on coastlines, facilitating trajectory planning and reducing risk to populated areas.
  - Proximity to the equator at Florida's site maximizes payload efficiency by leveraging the Earth's rotation.



# Launch Outcomes at SpaceX Sites

- **Visualization**: Folium maps with color-coded markers for each launch outcome.
- **Green Markers**: Represent successful launches. / **Red Markers**: Indicate failed launches.
- **Insights**:
  - The proximity of green markers to launch pads reflects SpaceX's increasing landing success rate over time.
  - Red markers highlight the challenges and learning opportunities from early failures.
- **Strategic Implications**:
- Marker colors on the map provide immediate visual feedback on the historical performance of each site.
- Analysis of marker distribution can inform future improvements in launch and landing operations.

**FLORIDA LAUNCH SITES**

- Dense clustering of markers around CCAFS SLC 40.
- A mix of successful and failed launches, with successes predominating in recent years.

**CALIFORNIA LAUNCH SITE**

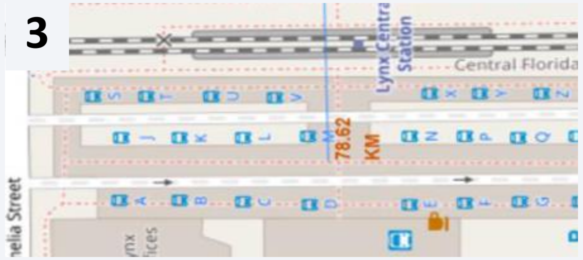
- VAFB SLC 4E shows fewer launches but a high success rate.

# Proximity Analysis for Launch Sites

- **Focus**: Assessing the strategic placement of launch sites concerning railways, highways, coastlines, and cities.
- **Visual Aid**: Folium map displaying calculated distances to various landmarks.
- **Key Distances**:
  - **To City (1)**: Launch sites maintain a significant distance from populated areas for safety. Example: 78.45 km from the nearest city.
  - **To Highway (2)**: Accessibility via highways is ensured, with the closest being 29.21 km away.
  - **To Railway (3)**: Sites are typically not in close proximity to railways, supporting operational security. Example: 78 km from the nearest railway station.
  - **To Coastline (4)**: Proximity to the coastline is crucial for trajectory planning and potential splashdowns. Example: 0.90 km from the coast.
- **Safety & Logistics**:
  - The locations suggest a balance between safety requirements and logistical needs for transport and recovery operations.
- **Strategic Placement**:
  - Sites are strategically chosen to optimize launch conditions and ensure public safety, while maintaining accessibility for logistical support.

4


3




2


1

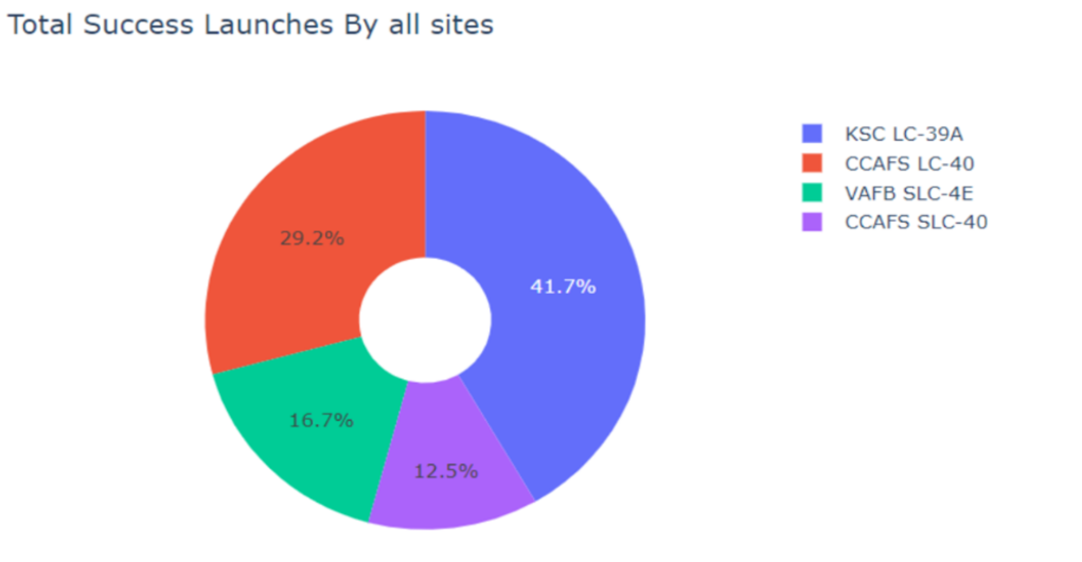
Section 4

# Build a Dashboard with Plotly Dash

# Launch Success Distribution Across SpaceX Sites

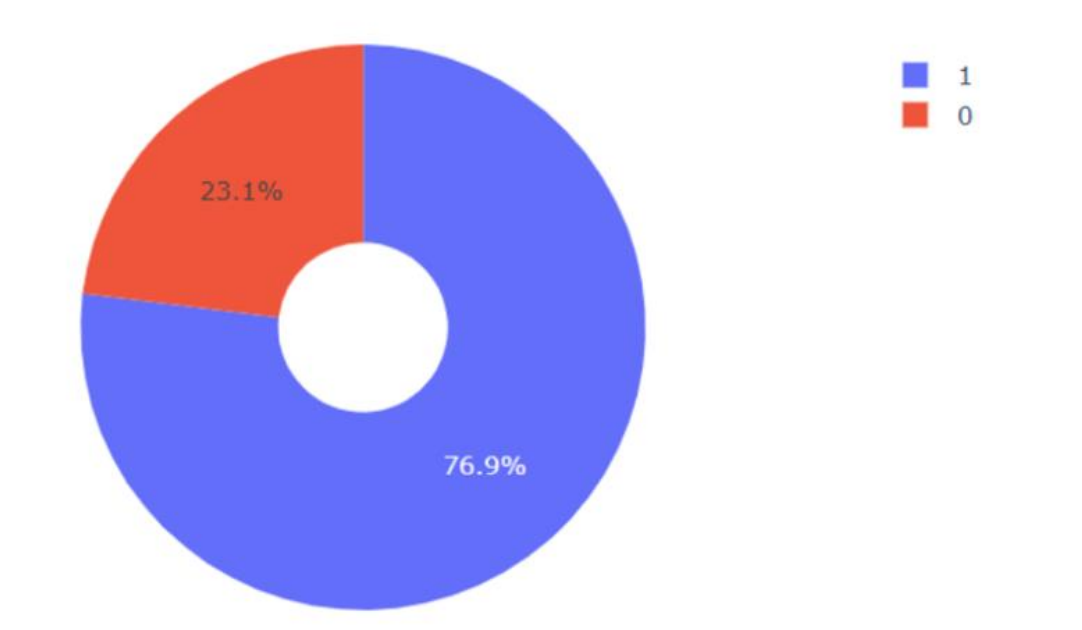
- **Visualization**: Pie chart illustrating the proportion of successful launches from each SpaceX site.
- **Key Data Points**:
  - KSC LC-39A has the highest share of successful launches, representing 41.7% of the total.
  - CCAFS LC-40 is also a significant contributor to success, split between two entries in the chart due to data representation.
  - VAFB SLC-4E contributes 16.7%, showcasing its role in SpaceX's operations.

- **Insights**:
  - The data underscores KSC LC-39A's prominent role in SpaceX's successful missions.
  - The pie chart's color coding makes it easy to distinguish between the sites.
- **Analysis**:
  - The success distribution reflects both the operational history and the evolution of launch site usage over time.



# Launch Success Ratio for KSC LC-39A

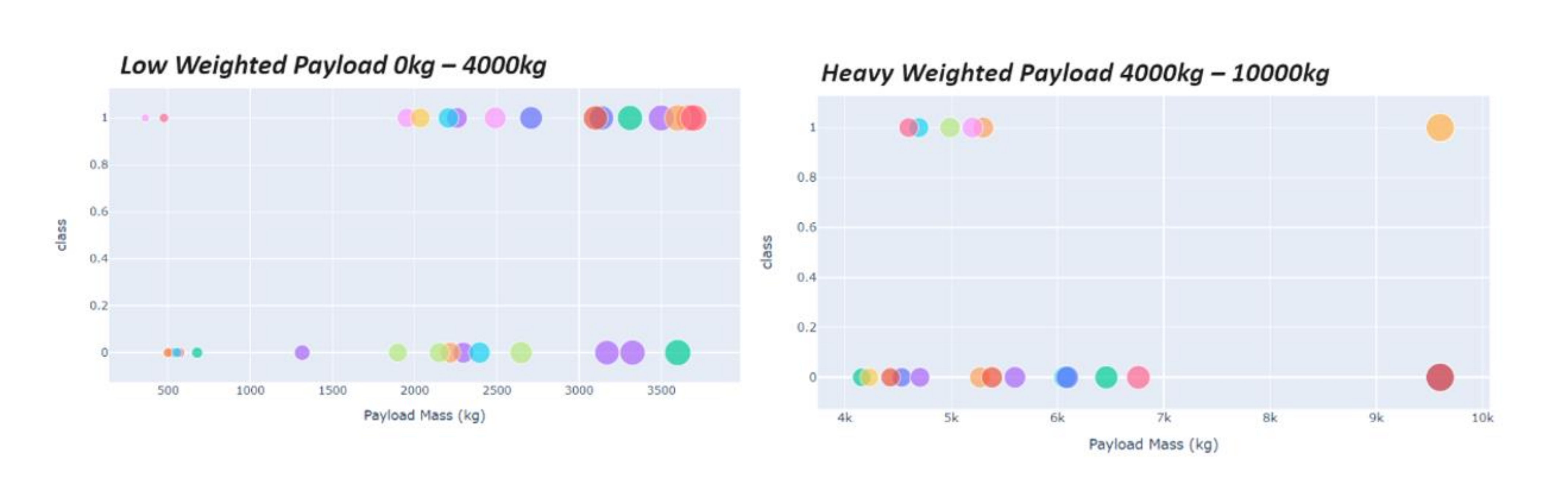
- **Visualization**: Pie chart displaying the success-to-failure ratio of launches.
- **Color Coding**:
  - Blue represents successful launches (76.9%).
  - Red indicates failed launches (23.1%).
- **Key Takeaway**:
  - KSC LC-39A has a high success rate, demonstrating its reliability as a launch site.




- **Implications**:
  - The substantial success rate underscores the effectiveness of launch operations and technology at KSC LC-39A.
- **Analysis**:
  - The pie chart provides a clear visual representation of launch outcomes, facilitating quick assessment of site performance.

# Scatter Plot of Payload vs. Launch Outcome for All Sites

- Using a range slider, we explored the relationship between payload weight and launch outcomes.
- Notably, the success rates for low-weighted payloads are higher compared to heavy payloads.
- Further analysis can help identify specific payload ranges or booster versions associated with the largest success rates.



Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

The decision tree classifier achieves the highest level of classification accuracy among the models.

```
[ ] models = {'KNeighbors':knn_cv.best_score_,
                  'DecisionTree':tree_cv.best_score_,
                  'LogisticRegression':logreg_cv.best_score_,
                  'SupportVector': svm_cv.best_score_}

    bestalgorithm = max(models, key=models.get)
    print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
    if bestalgorithm == 'DecisionTree':
        print('Best params is :', tree_cv.best_params_)
    if bestalgorithm == 'KNeighbors':
        print('Best params is :', knn_cv.best_params_)
    if bestalgorithm == 'LogisticRegression':
        print('Best params is :', logreg_cv.best_params_)
    if bestalgorithm == 'SupportVector':
        print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8767857142857143
Best params is : {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'random'}
```

# Confusion Matrix

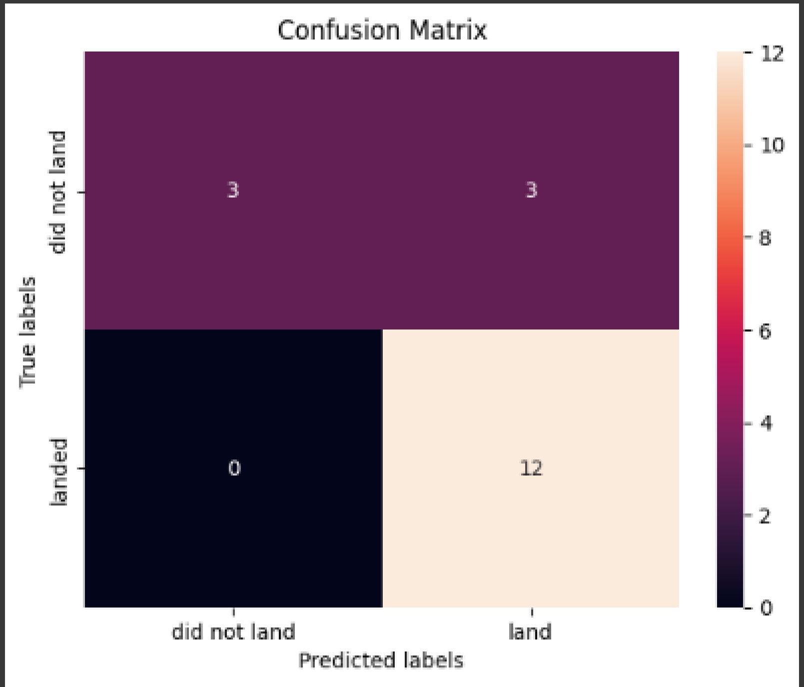
The confusion matrix of the decision tree classifier reveals its ability to differentiate between various classes, with a noteworthy concern being the occurrence of false positives. These false positives indicate instances where the classifier incorrectly identifies unsuccessful landings as successful ones.

```
[ ] tree_accuracy = tree_cv.score(X_test, Y_test)
    print("Decision Tree Test Accuracy:", tree_accuracy)
```

```
Decision Tree Test Accuracy: 0.8333333333333334
```

We can plot the confusion matrix

```
[ ] yhat = tree_cv.predict(X_test)
    plot_confusion_matrix(Y_test,yhat)
```



# Conclusions

- **Launch Site Performance:** The analysis of launch success rates across different sites revealed that certain launch sites, such as KSC LC-39A, consistently achieve high success rates, highlighting their reliability and effectiveness.
- **Payload Mass Impact:** The investigation into the relationship between payload mass and launch success demonstrated that for specific orbit types, heavier payloads tend to result in more successful missions, suggesting the importance of payload considerations in mission planning.
- **Orbit Type Significance:** The success rates associated with different orbit types indicated variations in mission complexity. This insight can inform mission planning and resource allocation for specific orbits.
- **Positive Launch Trend:** Over the years, there has been a positive trend in launch success rates, showcasing advancements in space technology and mission execution. This trend bodes well for future space exploration endeavors.
- **Booster Version Impact:** The analysis of different booster versions revealed variations in their performance. Some booster versions exhibited higher success rates, while others faced challenges. Understanding the strengths and weaknesses of each version is crucial for future mission planning.
- **Geographical Insights:** The exploration of launch site locations and their proximity to transportation infrastructure and coastlines provided valuable insights into site selection strategies. These findings can guide decisions on future launch site locations for optimal mission success.
- **Yearly Improvement:** Over the years, there has been a noticeable improvement in the success rate of SpaceX missions. This upward trend suggests that SpaceX's commitment to innovation and continuous improvement has resulted in more reliable and successful launches.

# Appendix

## 1. Python Code Snippets

1. Data Preprocessing Code: Link to Code
2. Machine Learning Model Development Code: Link to Code

## 2. SQL Queries

1. SQL Queries for Data Analysis: Link to Queries

## 3. Exploratory Data Analysis Charts

1. Scatter Plot: Flight Number vs. Launch Site: Link to Chart
2. Scatter Plot: Payload vs. Launch Site: Link to Chart
3. Bar Chart: Success Rate by Orbit Type: Link to Chart
4. Scatter Plot: Flight Number vs. Orbit Type: Link to Chart
5. Scatter Plot: Payload vs. Orbit Type: Link to Chart
6. Line Chart: Yearly Average Success Rate: Link to Chart

## 4. Interactive Analytics Dashboard

1. Pie Chart: Launch Success Count for All Sites: Link to Chart
2. Pie Chart: Launch Site with Highest Success Ratio: Link to Chart

## 5. Predictive Analysis Results

1. Model Evaluation Metrics: Link to Results

# Thank you!