IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<IHTESHAM USMAN>

23 MAY 2025

Outline

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

Executive Summary

- **Summary of methodologies**

1. Data Collection:

Accessed SpaceX launch data via API and web scraped records from Wikipedia.

1. Data Cleaning & Preparation

1. Cleaned and formatted the data.
2. Stored data in Db2 database and performed SQL queries.
3. Conducted exploratory data analysis.

2. Feature Engineering:

1. Created new features and standardized the data.

3. Interactive Visualizations

1. Mapped launch sites and success rates using Folium.
2. Built an interactive dashboard with Plotly Dash.

4. Model Building & Evaluation

1. Implemented SVM, Decision Trees, and K-Nearest Neighbors.
2. Tuned hyperparameters with GridSearchCV
3. Evaluated models using test data accuracy.

Summary of Results

1.Data Insights

1. Identified factors influencing Falcon 9 first stage landings.
2. Visualized geographical patterns and success rates.

2.Model Performance

1. Logistic Regression, SVM and K-Nearest Neighbors: 83.33%
2. Decision Tree: 94.44% accuracy.

3.Key Findings

1. Launch site and payload mass impact landing success.
2. Decision Tree model is the most effective predictor.

Introduction

- **Project Overview and Framework:**
- For this culminating endeavor, our objective is to forecast the triumphant touchdown of the Falcon 9's initial booster. SpaceX promotes its rocket launches at a considerably reduced expense compared to other service providers, primarily owing to their capacity to repurpose the rocket's primary stage. Through precise forecasting of landing triumph, we can gauge launch expenditures and offer significant foresight for enterprises competing with SpaceX.

- **Questions We Seek to Address:**
- What elements contribute to the victorious touchdown of the Falcon 9's first stage?
- How can we precisely foresee the landing result utilizing artificial intelligence algorithms?
- Which artificial intelligence algorithm demonstrates optimal performance in forecasting landing prosperity?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - This Project Employs a comprehensive approach to predict the succesfull landing of the falcon 9 first stage, incorporating data collection, exploratory analysis, interactive visualizations, and predictive modeling.
- Perform data wrangling
  - Data cleaning involved handlimg missing values, standardizing, and ensuring consistency.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Visualized launch success rates, payloads, and launch sites using Matplotlib and Seaborn.
  - Executed SQL queries to derive insights and answer specific questions regarding the dataset.

Methodology

- Perform interactive visual analytics using Folium and Plotly Dash
  - Used Folium to create interactive maps displaying launch sites and outcomes.
  - Developed a Plotly Dash application with interactive components like dropdowns and sliders to analyse launch success rates and payload ranges.
- Perform predictive analysis using classification models
  - Built and evaluated various classification models including Logistic Regression, SVM, KNN, and Decision Trees.
  - Employed GridSearchCV for hyperparameter tuning.
  - Evaluated models based on accuracy, and identified the best performing model for predicting landing success.

  - https://github.com/ihteshamusman/DataScience-Capstone-IBM/tree/main

Data Collection

- Describe how data sets were collected.
  - SpaceX API Request
    - Initiate API Request
    - Fetch launch Data
    - Stored Locally Data
  - Web Scraping Wikipedia
    - Extract HTML Table
    - Parse with beautiful soap
    - Convert to DataFrame
  - Data Integration
    - SpaceX APIs
    - Merge Datastes
    - Final Integration

Data Collection – SpaceX API

Step 1: Initiate API Request

Use Python's `requests` library to connect to the SpaceX API.

static_json_url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'

Step 2: Parse API Response

Convert API response from JSON to a Python

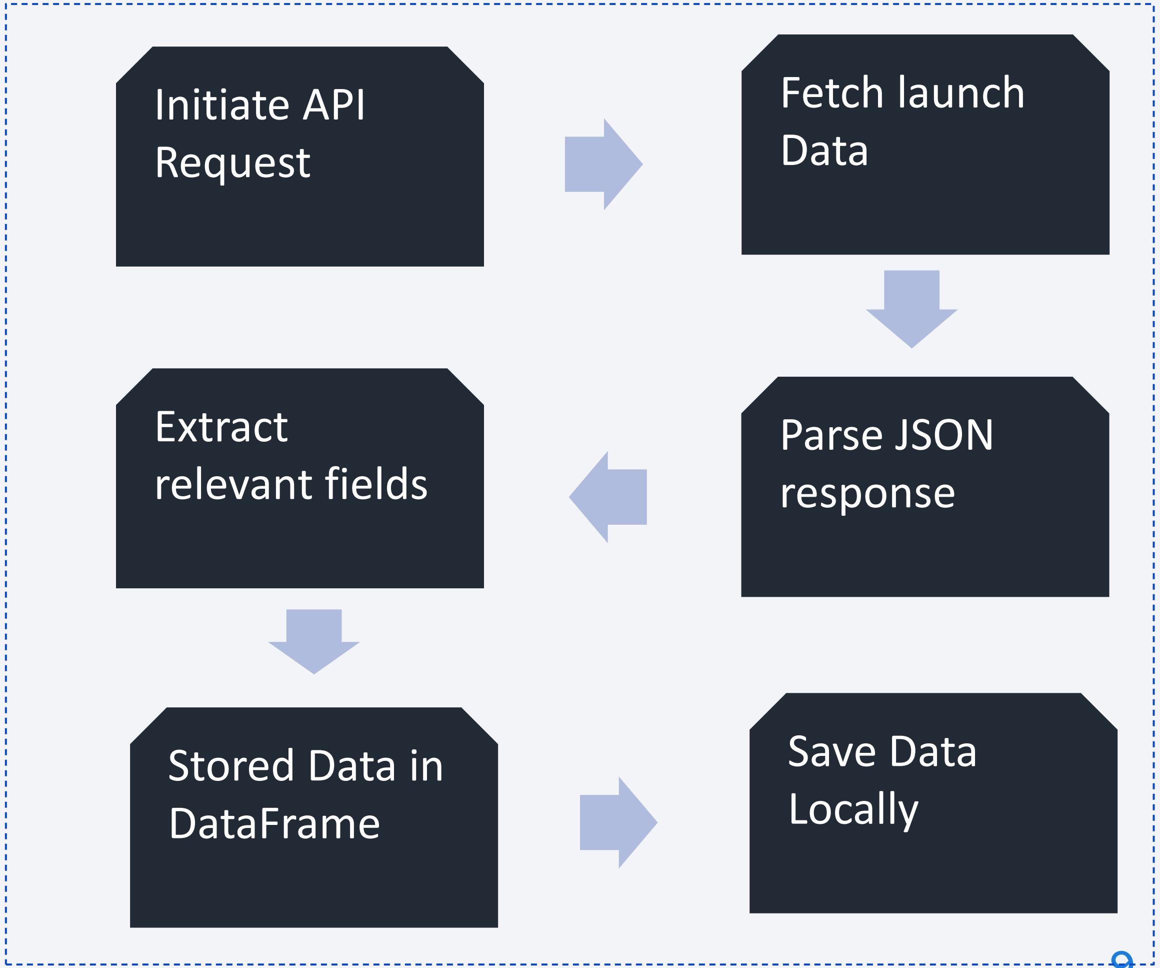
Extract relevant fields: launch date, launch site,

payload mass, rocket type, outcome.

Step 3: Store Data Locally

Save extracted data into a pandas

Stored the DataFrame locally for further processing



https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/1-jupyter-labs-spacex-data-collection-api-v2%20(1).ipynb

Data Collection - Scraping

Step 1: Initiate Web Scraping

- Use Python's `requests` library to fetch the
- HTML content of the Wikipedia page.

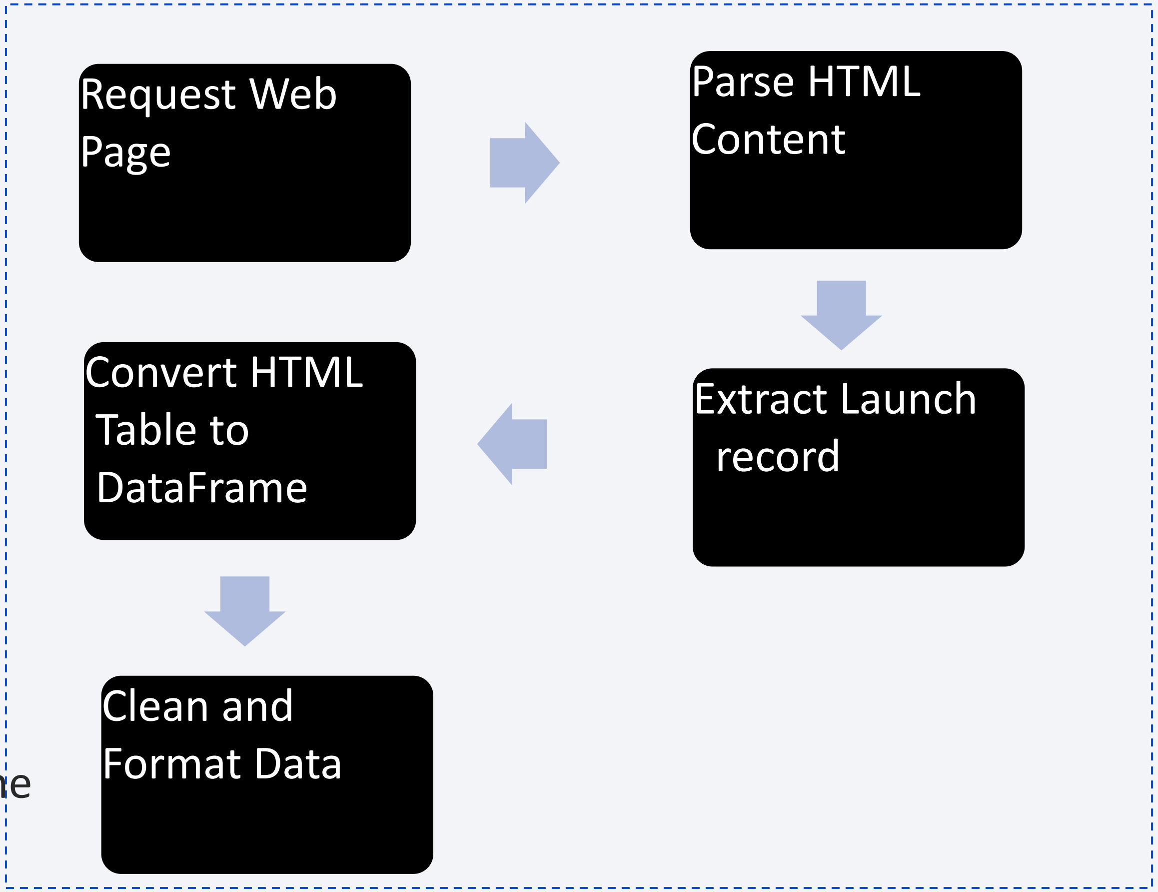
- static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

Step 2: Parse HTML Content

- Use beautiful Soup ` to parse the HTML content.
- Extract the HTML table containing Falcon 9 launch records.

Step 3: Convert to DataFrame

Convert the extracted HTML table into a pandas DataFrame



https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/2-jupyter-labs-webscraping.ipynb

Data Wrangling

Overview: Data wrangling involves cleaning, transforming, and organizing raw data into a structured format suitable for analysis.

Step 1: Data Cleaning
Fix or remove incorrect, missing, or duplicate data to make it accurate and consistent.

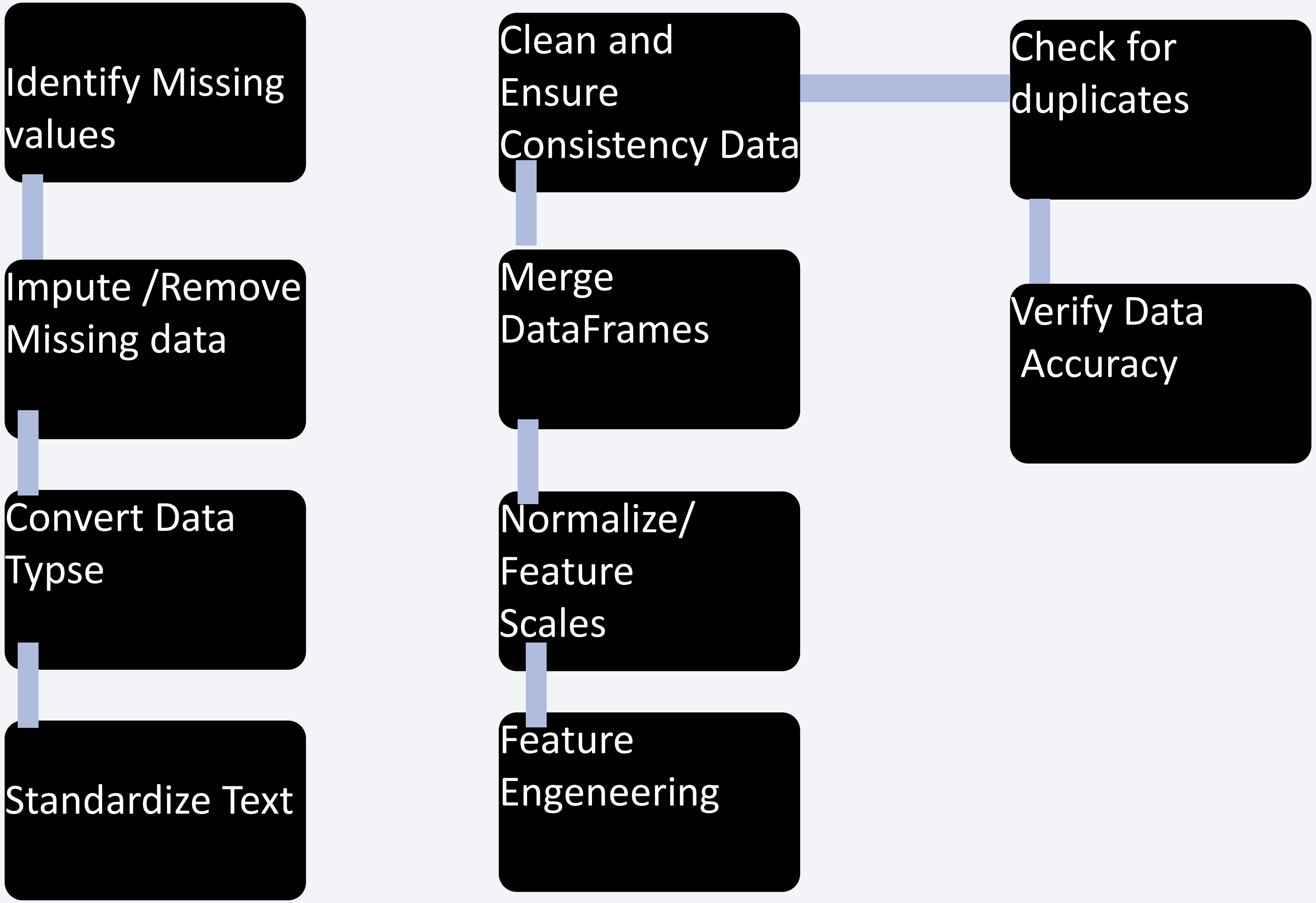
Step 2: Data Transformation
Convert data into a usable format, like changing data types or encoding categories.

Step 3: Data Integration
Combine data from different sources or tables into a single, unified dataset.

Step 4: Data Validation
Ensure the data is correct, complete, and follows the required rules or formats.

https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/3-labs-jupyter-spacex-Data%20wrangling-v2%20(1).ipynb

Data Wrangling



EDA with Data Visualization

•**Histogram**

Purpose: To show the distribution of a numeric variable.
Why: Helps identify skewness, central tendency, and spread.

•**Box Plot**

Purpose: To visualize the spread and detect outliers.
Why: Useful for comparing distributions and spotting extreme values.

•**Scatter Plot**

Purpose: To observe relationships between two numeric variables.
Why: Reveals trends, correlations, or clusters.

•**Bar Chart**

Purpose: To compare values across categories.
Why: Clearly shows differences in category counts or metrics.

EDA with Data Visualization

•Pie Chart:

Purpose: To show proportions of a whole.
Why: Useful for simple visual representation of part-to-whole ratios.

•Correlation Heatmap

Purpose: To display correlations between variables.
Why: Quickly highlights strong or weak relationships.

•Line Chart

Purpose: To show trends over time.
Why: Ideal for identifying increases, drops, or cycles in time-series data.

https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/5-jupyter-labs-eda-dataviz-v2.ipynb

EDA with SQL

- **Aggregate Queries:**
- Calculated total number of launches.
- Counted successful and failed launches.
- Calculated success rates by launch site and rocket type.
- **Join Queries:**
- Joined tables to link launch records with additional data (e.g., rocket details).
- Combined datasets for comprehensive analysis.
- **Filtering Queries:**
- Filtered data to focus on specific launch outcomes (success/failure).
- Applied conditions to extract launches based on criteria like launch date or rocket configuration.
- **Sorting Queries:**
- Sorted data to identify trends or outliers.
- Ordered launches by date or success rate for analysis.
- **Subqueries:**
- Nested queries to calculate derived metrics (e.g., average payload mass per launch site).
- Subqueries used to perform detailed analysis within larger datasets.

https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/4-jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- **Markers:**
- Placed markers to indicate launch sites on the map.
- Each marker represents a specific geographical location
- **Circles:**
- Added circles around launch sites to visually represent proximity zones.
- Circles help visualize the areas around launch sites that might influence operational decisions.
- **Lines:**
- Drew lines to connect launch sites with their proximities or other relevant locations.
- Lines provide spatial context and connections between different points of interest related to launches.

- **Reasons for Adding Objects**
- **Markers:**
- To pinpoint exact launch locations for spatial reference.
- Helps users identify where SpaceX has conducted launches geographically.
- **Circles:**
- Illustrates the potential impact zones around launch sites.
- Provides a visual representation of safety perimeters or operational boundaries.
- **Lines:**
- Shows connections or relationships between launch sites and relevant features.
- Enhances understanding of spatial relationships and dependencies.

https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/6-lab-jupyter-launch-site-location-v2.ipynb

Build a Dashboard with Plotly Dash

- **Plots/Graphs Added**
- **Success Pie Chart:**
- Displays the distribution of successful and failed launches.
- Helps visualize the overall success rate and performance trends.

- **Success-Payload Scatter Plot:**
- Shows the relationship between payload mass and launch success.
- Allows users to explore how payload mass influences mission outcomes.

- **Interactions Added**
- **Launch Site Dropdown:**
- Enables users to select specific launch sites for analysis.
- Facilitates filtering and focused exploration based on geographical locations.

- **Range Slider for Payload:**
- Allows users to adjust payload mass ranges dynamically.
- Offers flexibility in examining launch success concerning payload mass variations.
-

https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/Dashboard.ipynb

Reasons for Adding Plots and Interactions

- **Success Pie Chart:**
- This visualization provides a quick overview of mission success rates, which is essential for stakeholders to understand overall performance metrics at a glance.
- **Launch Site Dropdown:**
- This feature enhances user experience by allowing users to focus their analysis on specific launch locations, facilitating regional insights and comparisons across different launch sites.
- **Success-Payload Scatter Plot:**
- This plot helps identify correlations between payload characteristics and launch outcomes, supporting decision-making processes related to payload planning and operational strategies.
- **Range Slider for Payload:**
- This interactive tool offers an exploration of how payload mass affects mission success, enabling detailed analysis and insights into payload-related performance factors.

Predictive Analysis (Classification)

•**Data Pre-processing**

•Standardized features to ensure all variables contribute equally.
•Split data into training and test sets for model validation.

•**Model Selection**

•Explored multiple classification algorithms: SVM, Decision Trees, and K-Nearest Neighbours (KNN).
•Chose algorithms suitable for binary classification tasks based on project requirements.

•**Hyperparameter Tuning**

•Used GridSearchCV to systematically search for optimal hyperparameters.
•Tuned parameters such as C (SVM), max_depth (Decision Trees), and neighbours (KNN).

•**Model Evaluation**

•Evaluated models using cross-validation techniques to ensure robustness and generalizability.
•Utilized metrics like accuracy, precision, recall, and F1-score to assess model performance.

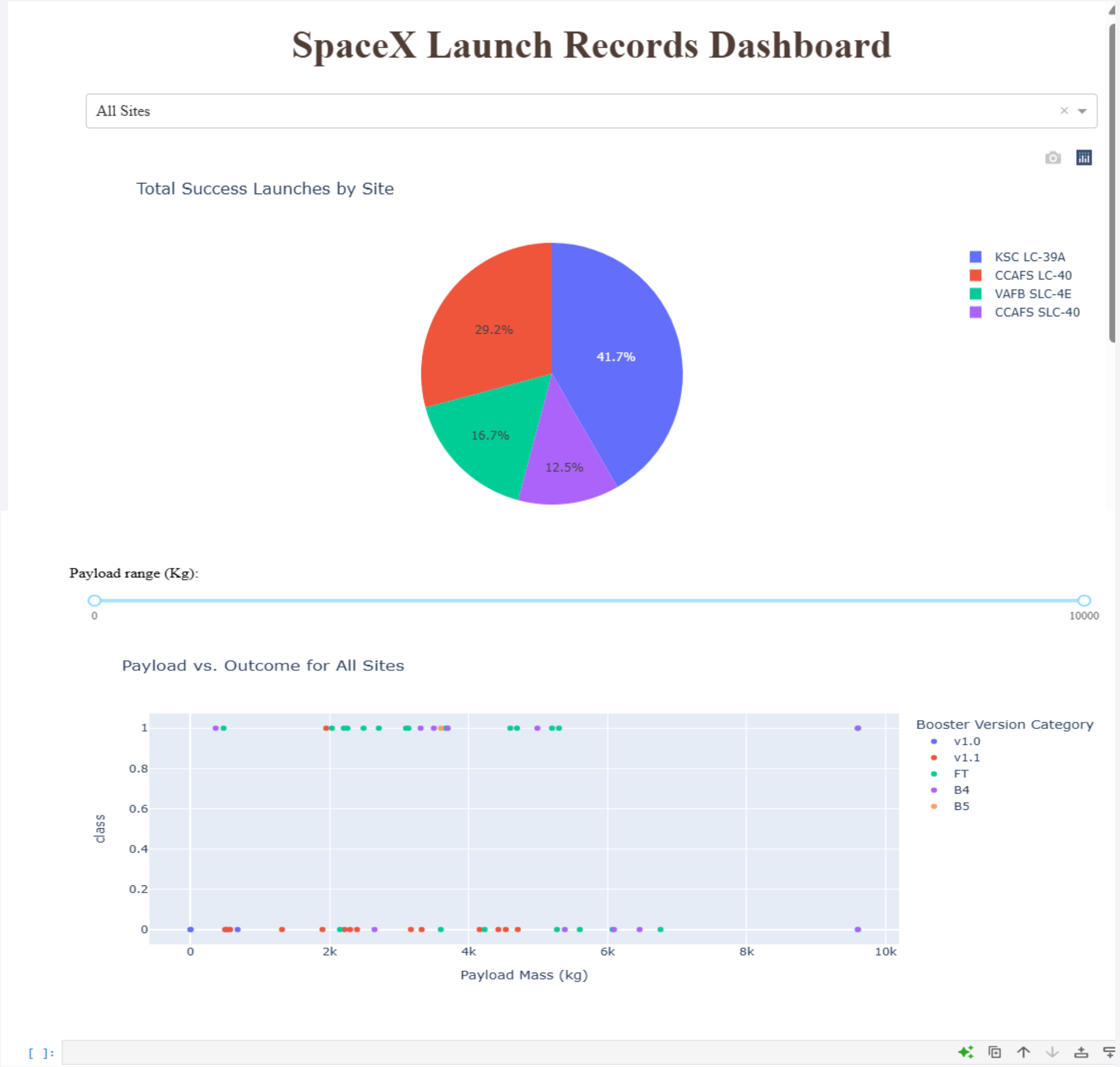
•**Improvement Iterations**
•Iteratively adjusted models based on insights from validation results.
•Fine-tuned hyperparameters to maximize predictive accuracy and reliability.

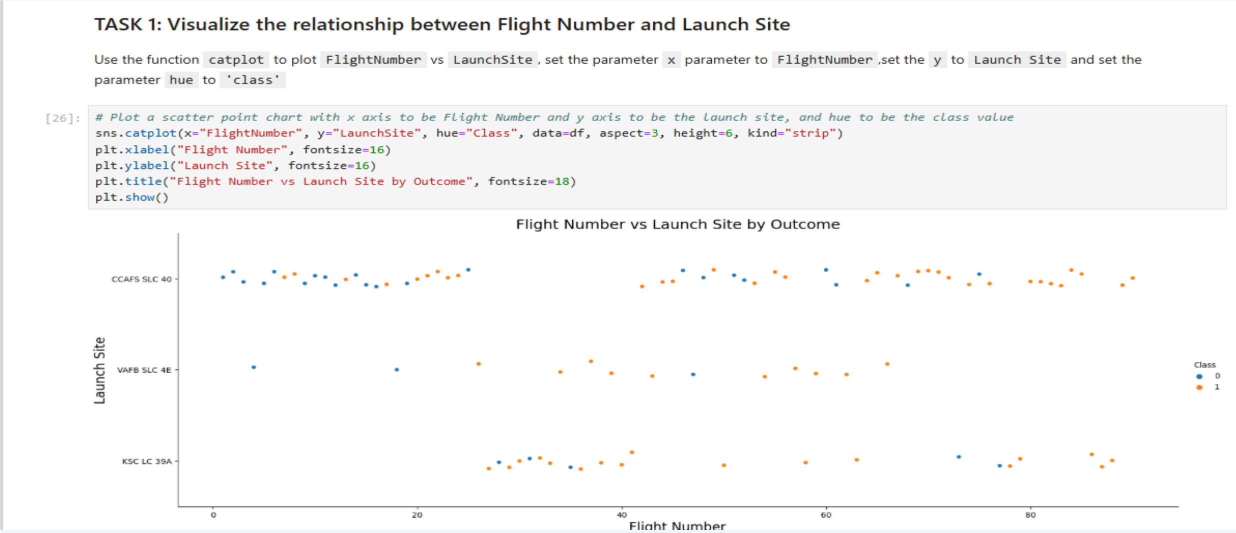
•**Selection of Best Performing Model**

•Identified the model with the highest accuracy on the test set as the best performer.
•Considered both training and test set performance to avoid overfitting and ensure real-world applicability.

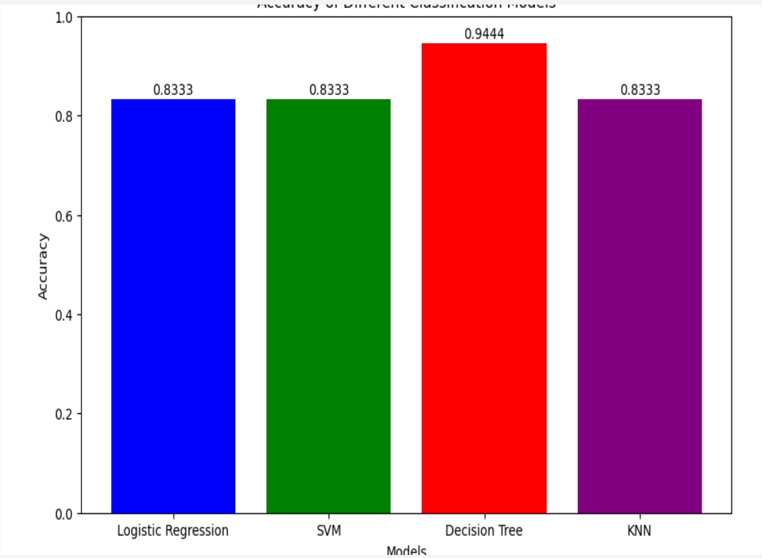
https://github.com/ihteshamusman/DataScience-Capstone-IBM/blob/main/7-SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb

Results





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



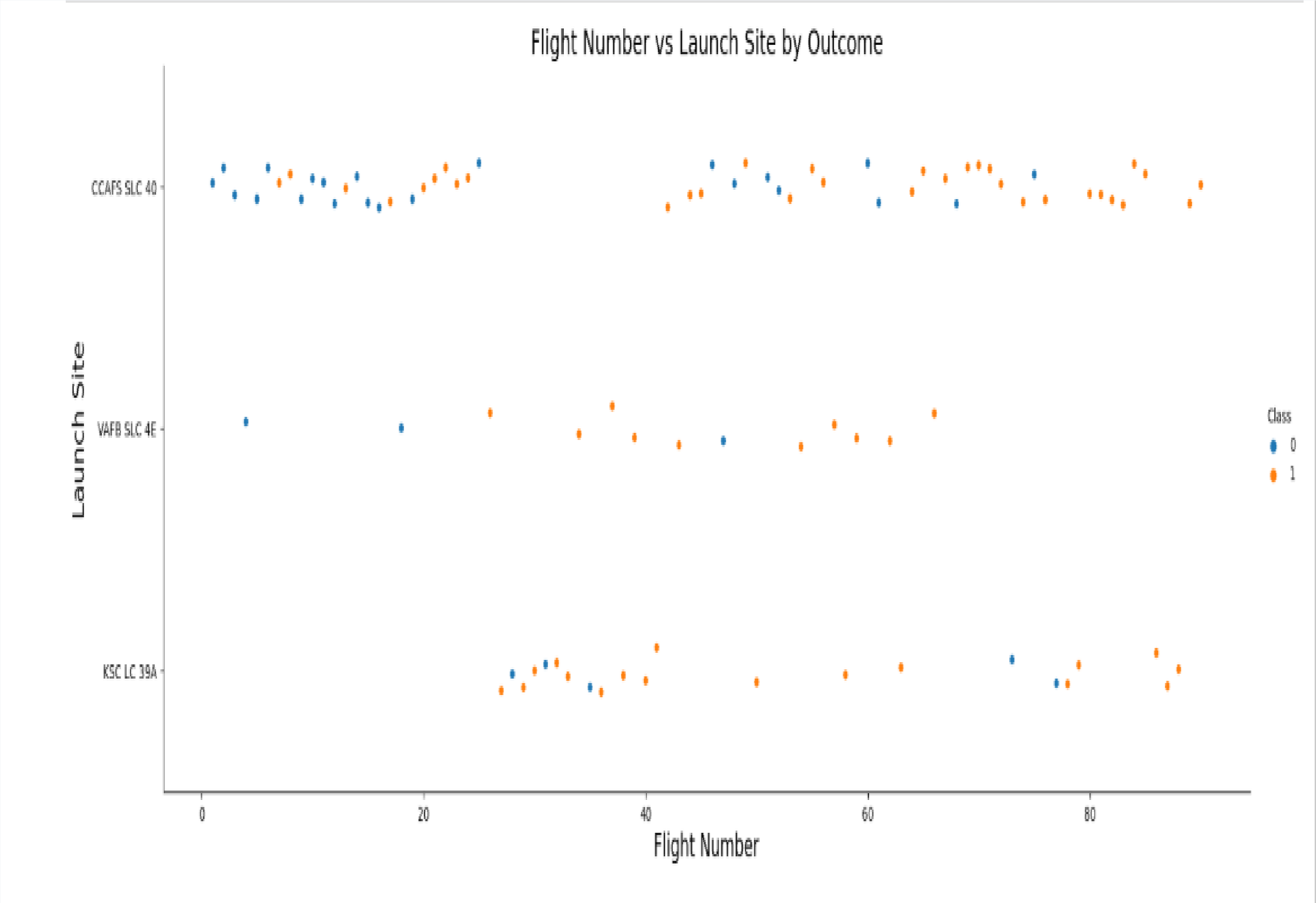
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

Mixed Outcomes at Major Launch Sites: Both CCAFS SLC 40 and KSC LC 39A
have a mix of successful (orange) and unsuccessful (blue) landings, indicating that factors other than the launch site itself may influence the landing success.

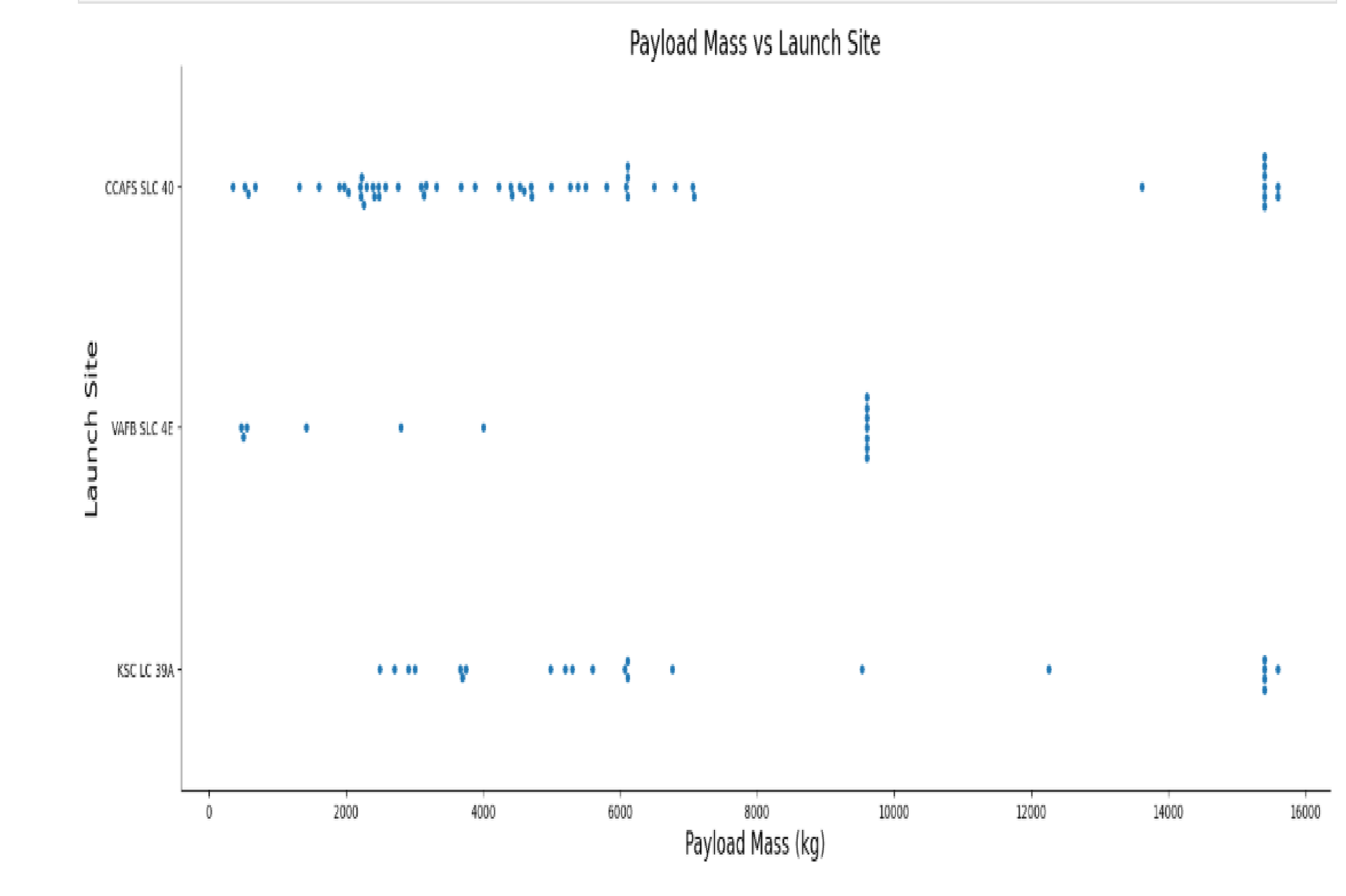
Consistent Activity Across Flight Numbers: Launches are spread across a wide
range of flight numbers at all sites, suggesting consistent activity over time without a clear trend of increasing or decreasing landing success.



Payload vs. Launch Site

- Payload:payloads below 10,000 kg, while the VAFB SLC 4E and KSC LC 39A sites have a wider range of payload masses, indicating varied mission profiles.

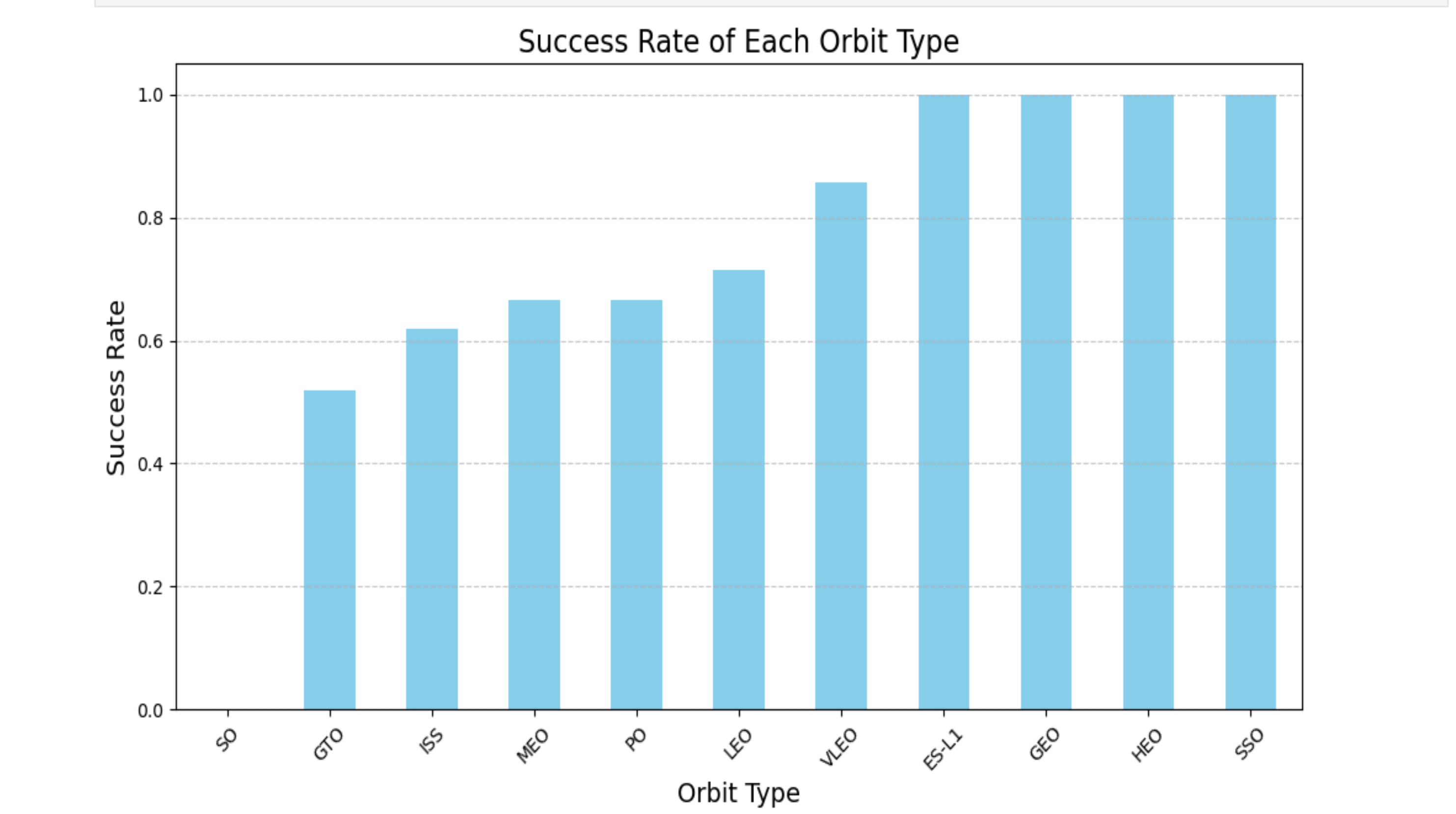
- Launch Site•:**High-Capacity Launches:** The KSC LC 39A site is frequently used for launching
- heavier payloads, with multiple launches carrying over 15,000 kg, suggesting its suitability for high-capacity missions.



Success Rate vs. Orbit Type

- **High Success Rates:** Missions to VLEO, ES-L1, GEO, HEO, and SSO orbits have achieved a perfect success rate, indicating these orbits are highly reliable for successful first stage landings.

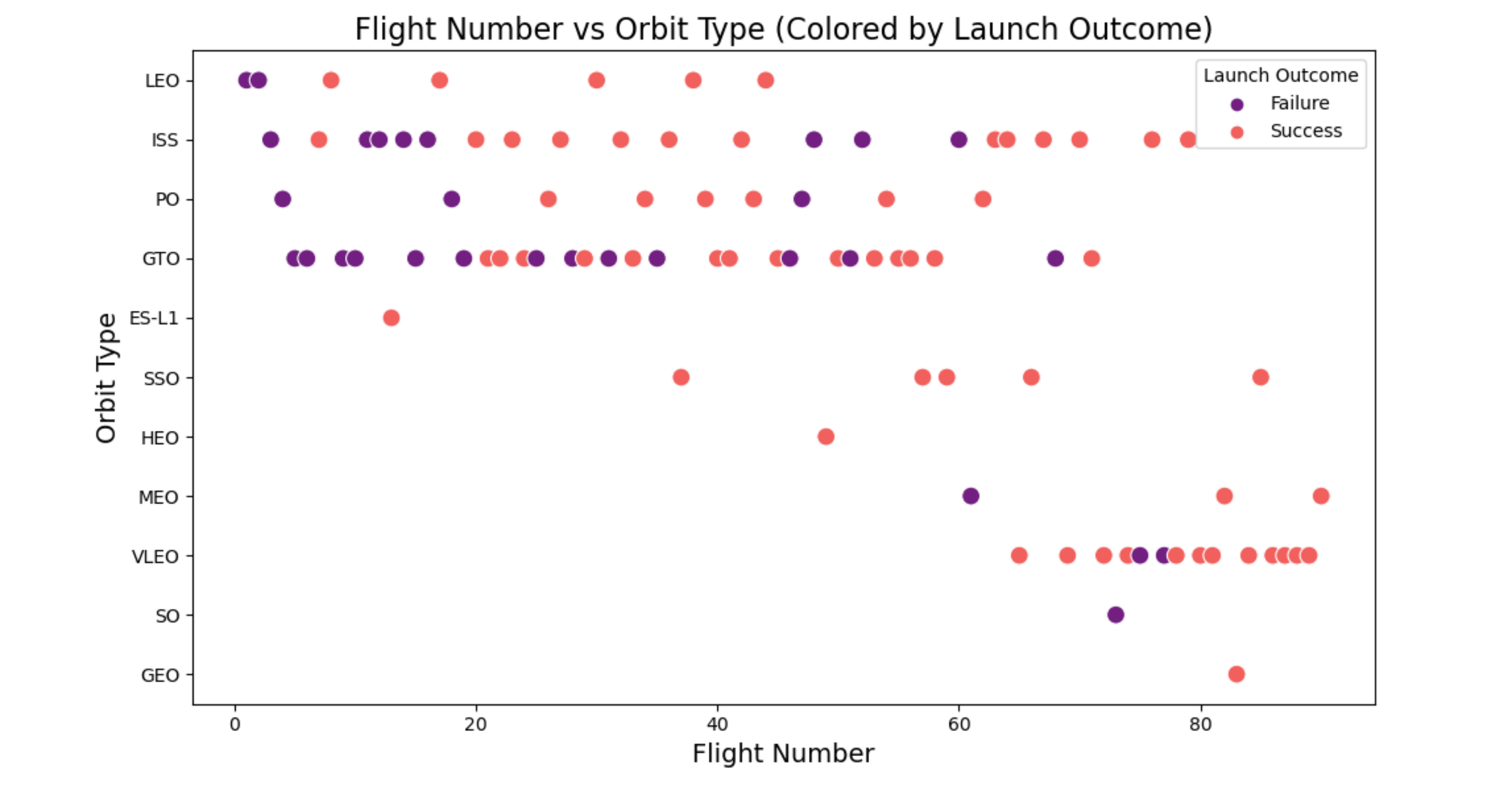
- **Lower Success Rate for GTO:** The GTO orbit type shows a significantly lower success rate compared to other orbit types, suggesting that missions to this orbit may involve greater challenges or complexities.



Flight Number vs. Orbit Type

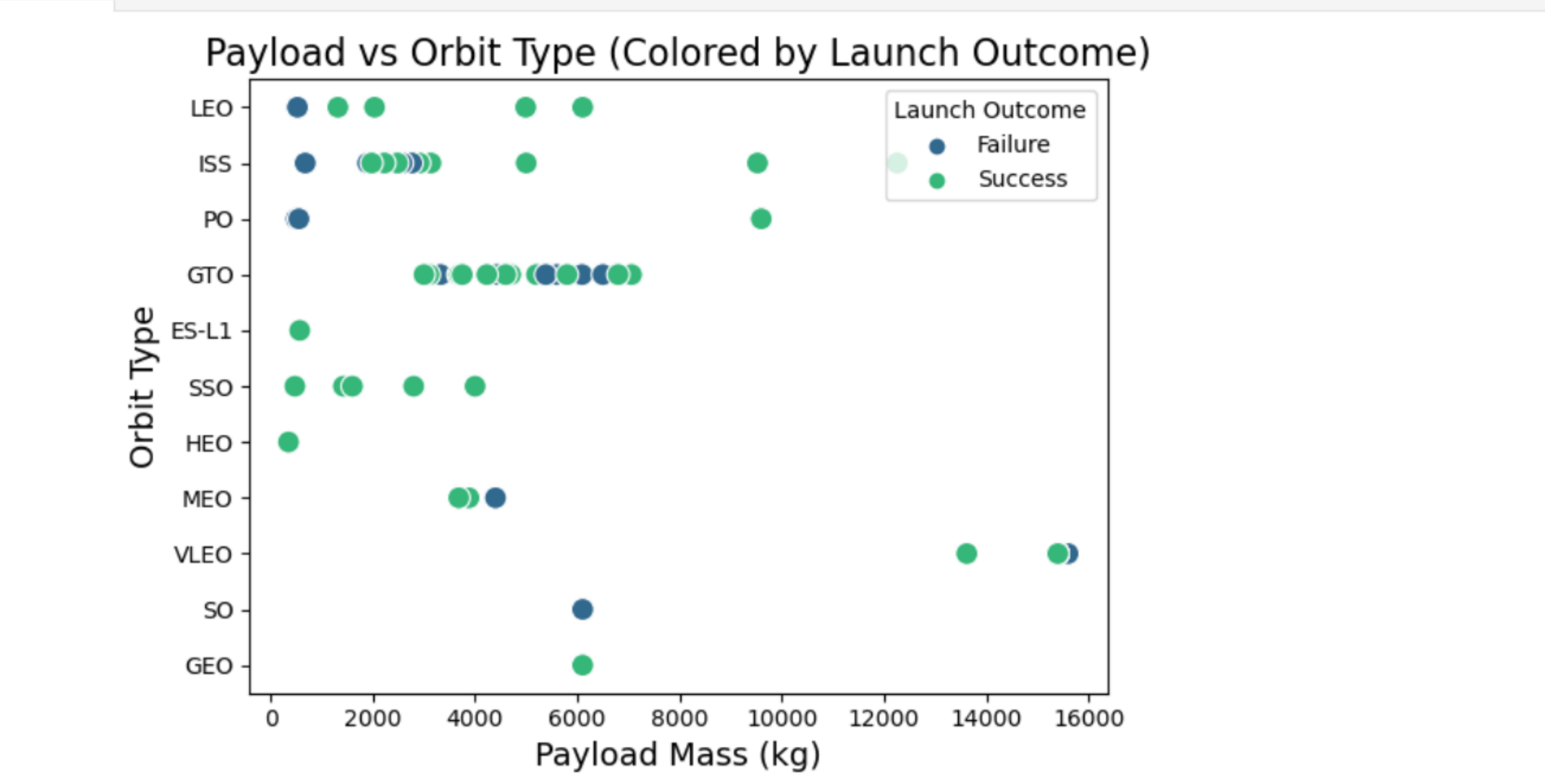
- **Increased Success Over Time:** The success rate of Falcon 9 launches improves significantly with higher flight numbers, indicating that experience and iterative improvements contribute to better outcomes.

- **Orbit-Specific Performance:** Early flights to GTO and ISS orbits had mixed outcomes, but recent missions to these orbits show a higher success rate, reflecting advancements in mission planning and execution.



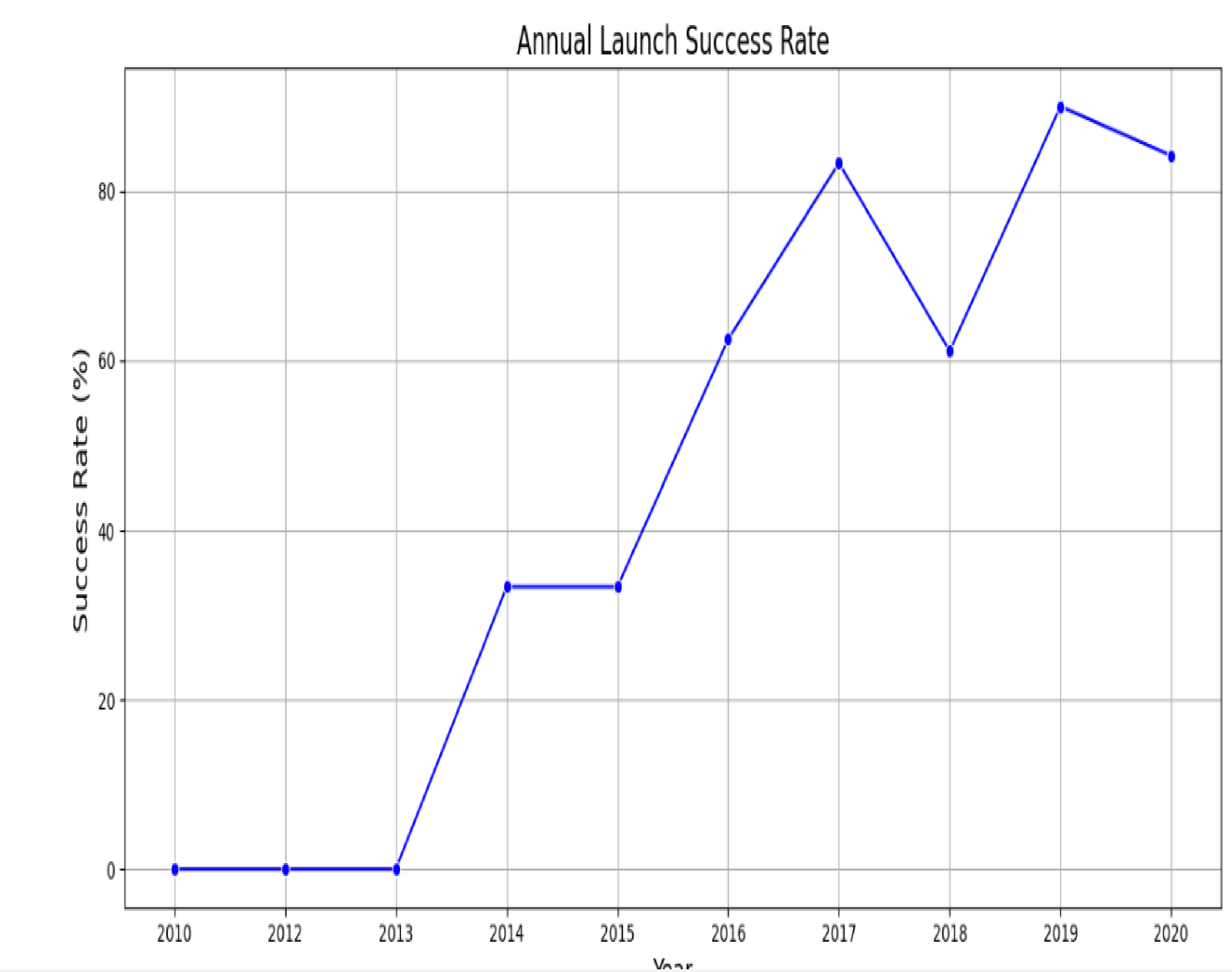
Payload vs. Orbit Type

- Successful landings are more frequent across all orbit types, especially for payloads less than 6000 kg.
- Higher payload masses (above 10,000 kg) show a mix of successes and failures, indicating increased difficulty with heavier payloads.



Launch Success Yearly Trend

- The annual launch success rate has shown a significant improvement from 2013 onwards, reaching over 80% by 2020.
- Despite a dip in 2018, the overall trend indicates increasing reliability and success in Falcon 9 launches over the years.



All Launch Site Names

Task 1

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

```
[16]: %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

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

[16]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

[18]:
```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

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

[18]:

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

Task 3

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

[20]:
```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)';
```

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

[20]:

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

Task 4

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

[22]:
```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1';
```

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

[22]:

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

[24]:
```
%sql SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)';
```

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

[24]:

MIN(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

[26]:
```
%sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS
```

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

[26]:

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

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

[28]:

```
%sql SELECT "Mission_Outcome", COUNT(*) AS "Total" FROM SPACEXTABLE WHERE "Mission_Outcome" IN ('Success', 'Failure') GROUP BY "Mission_Outcome";
```

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

[28]:

Mission_Outcome	Total
Success	98

Boosters Carried Maximum Payload

Task 8

List all the booster_versions that have carried the maximum payload mass. Use a subquery.

[30]:
```
%sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE);
```

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

[30]:

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

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.

[32]:

```
%%sql

SELECT
    CASE
        WHEN substr("Date", 6, 2) = '01' THEN 'January'
        WHEN substr("Date", 6, 2) = '02' THEN 'February'
        WHEN substr("Date", 6, 2) = '03' THEN 'March'
        WHEN substr("Date", 6, 2) = '04' THEN 'April'
        WHEN substr("Date", 6, 2) = '05' THEN 'May'
        WHEN substr("Date", 6, 2) = '06' THEN 'June'
        WHEN substr("Date", 6, 2) = '07' THEN 'July'
        WHEN substr("Date", 6, 2) = '08' THEN 'August'
        WHEN substr("Date", 6, 2) = '09' THEN 'September'
        WHEN substr("Date", 6, 2) = '10' THEN 'October'
        WHEN substr("Date", 6, 2) = '11' THEN 'November'
        WHEN substr("Date", 6, 2) = '12' THEN 'December'
        ELSE 'Unknown'
    END AS "Month_Name",
    "Mission_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM
    SPACEXTABLE
WHERE
    substr("Date", 0, 5) = '2015';
```

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

[32]:

Month_Name	Mission_Outcome	Booster_Version	Launch_Site
January	Success	F9 v1.1 B1012	CCAFS LC-40
February	Success	F9 v1.1 B1013	CCAFS LC-40
March	Success	F9 v1.1 B1014	CCAFS LC-40
April	Success	F9 v1.1 B1015	CCAFS LC-40
April	Success	F9 v1.1 B1016	CCAFS LC-40
June	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
December	Success	F9 FT B1019	CCAFS LC-40

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

Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

[34]:

```
%%sql

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

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

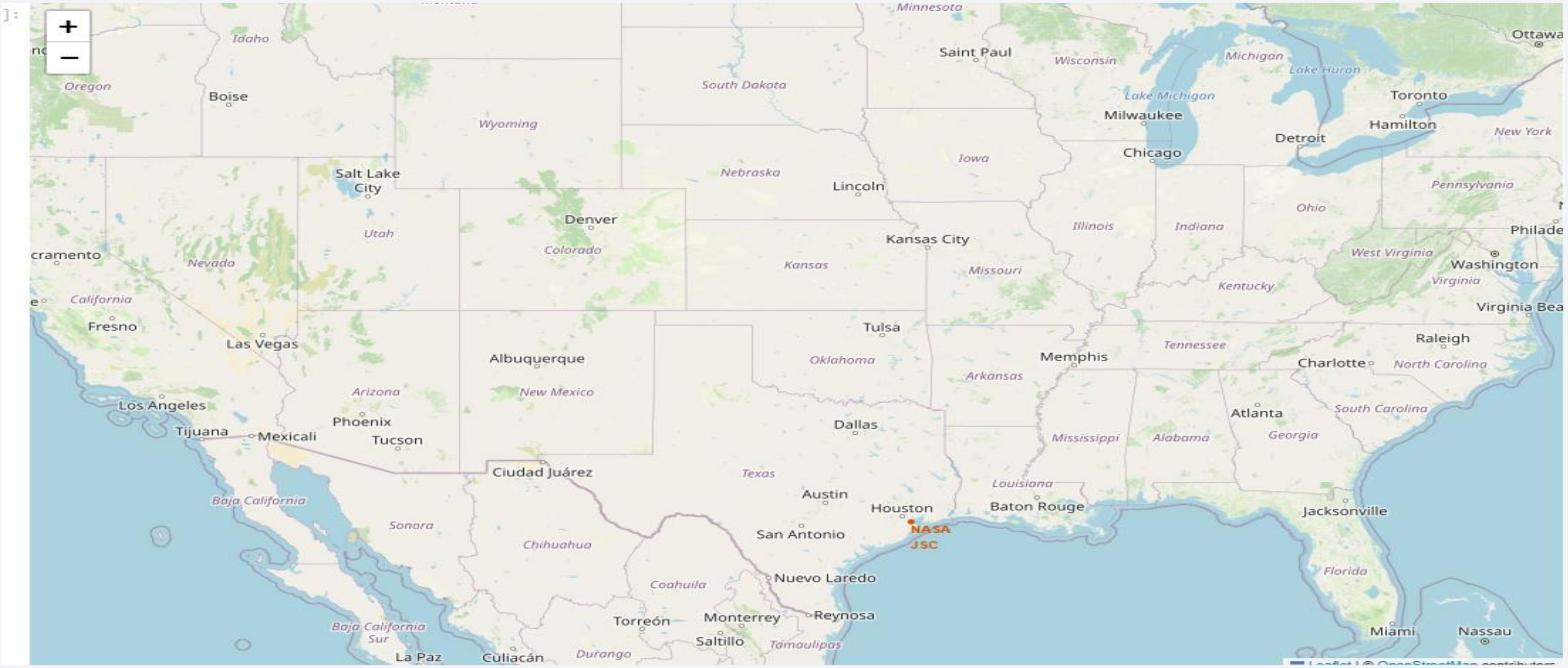
[34]:

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

Section 3

Launch Sites Proximities Analysis

Map with Folium



Distance between launch site and its proximities

TASK 3: Calculate the distances between a launch site to its proximities

Next, we need to explore and analyze the proximities of launch sites.

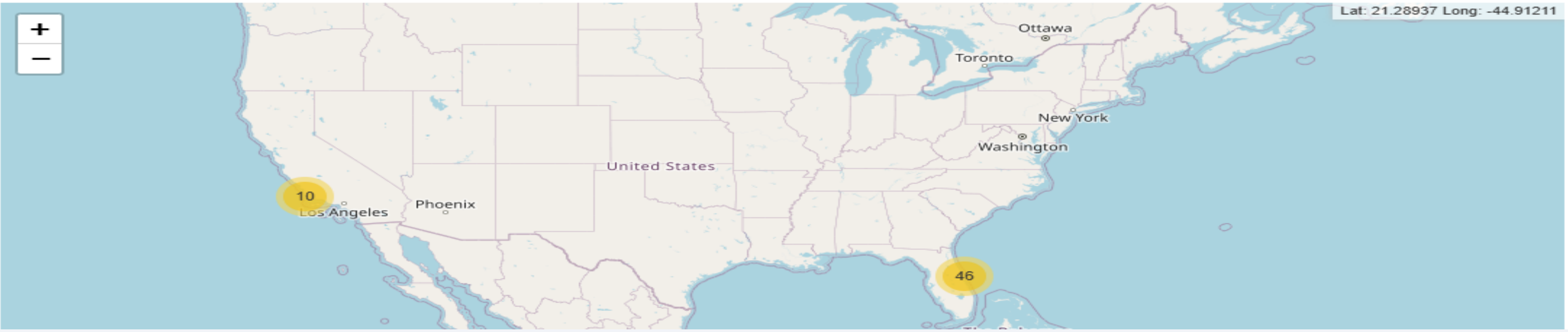
Let's first add a `MousePosition` on the map to get coordinate for a mouse over a point on the map. As such, while you are exploring the map, you can easily find the coordinates of any points of interests (such as railway)

[58]:

```
# Add Mouse Position to get the coordinate (Lat, Long) for a mouse over on the map
formatter = "function(num) {return L.Util.formatNum(num, 5);};"
mouse_position = MousePosition(
    position='topright',
    separator=' Long: ',
    empty_string='NaN',
    lng_first=False,
    num_digits=20,
    prefix='Lat:',
    lat_formatter=formatter,
    lng_formatter=formatter,
)

site_map.add_child(mouse_position)
site_map
```

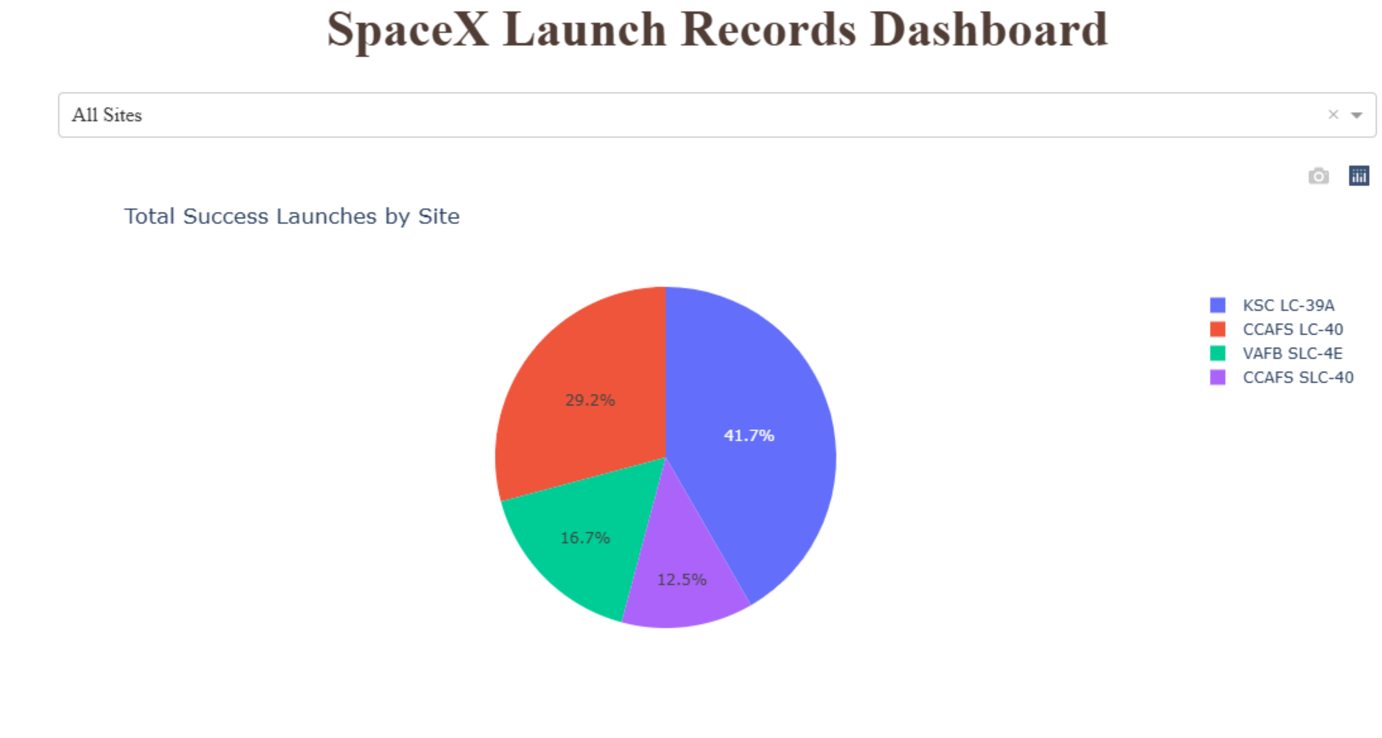
[58]:



Section 4

Build a Dashboard with Plotly Dash

SpaceX Pie chart Dashboard



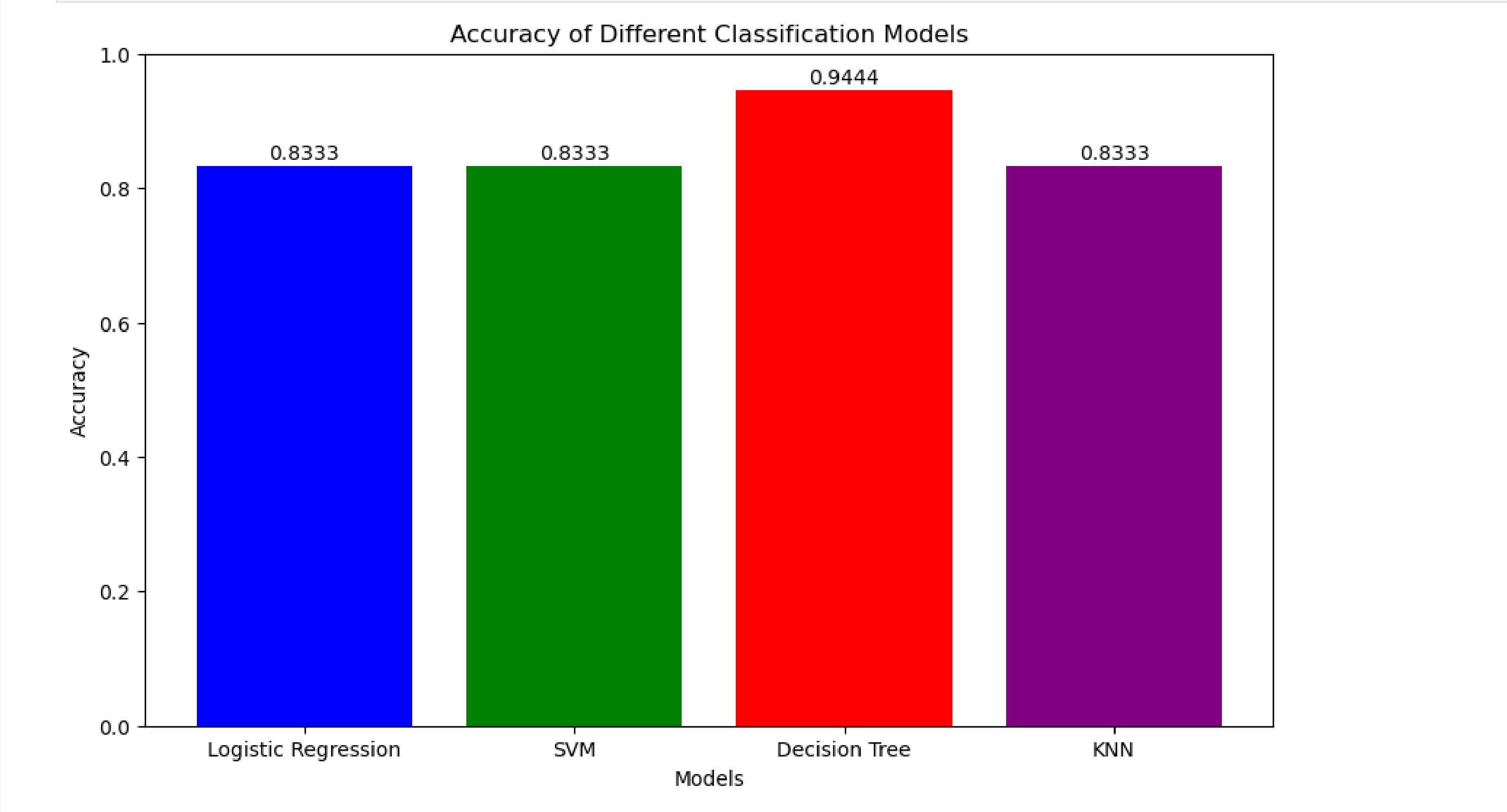
- Percentage of KSC LC is 41.7% leading. While CCAFS LC 29.25%, VAFB SLC is 16.7% and last one is CCAFS SLC with 12.5%.

SpaceX Scatter chart Dashboard

Section 5

Predictive Analysis (Classification)

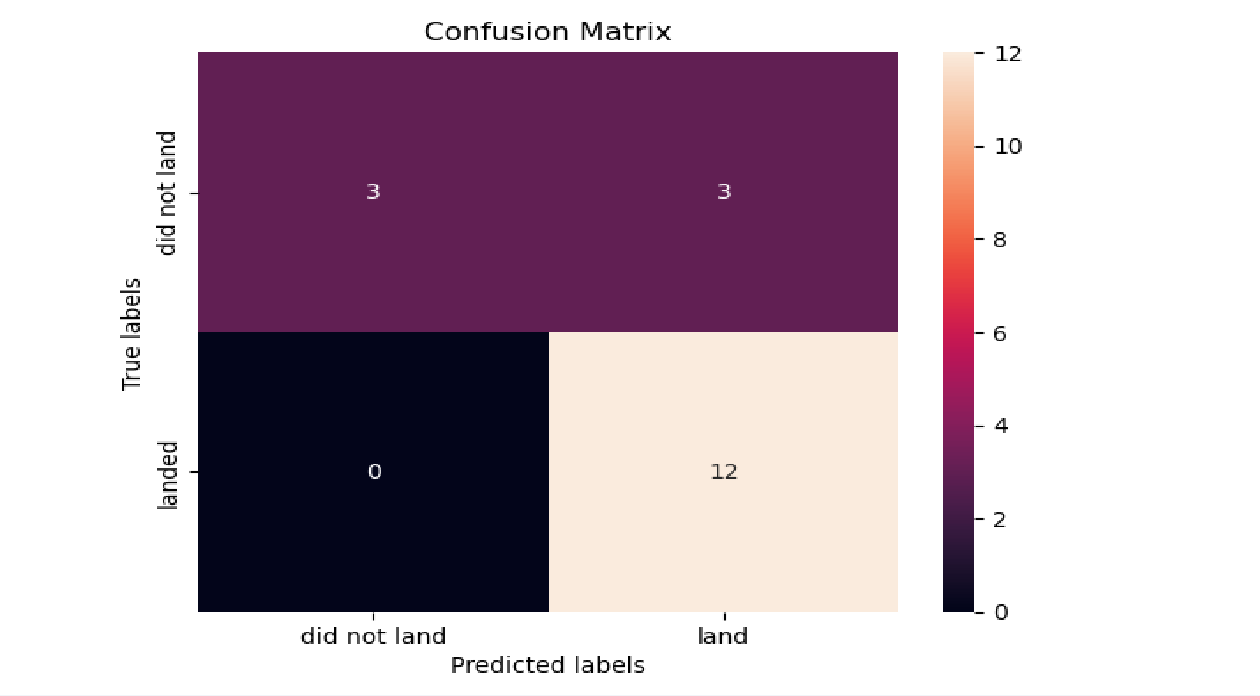
Classification Accuracy



- Decision Tree Model have the better accuracy.

Confusion Matrix

- **Confusion Matrix of the Best Performing Model (Decision Tree with 94.94% Accuracy):**
- The best performing model in our project is the **Decision Tree classifier**, which achieved an impressive **accuracy of 94.94%** on the test dataset. To better understand the performance of this model beyond just accuracy, we present its confusion matrix below:



Conclusions

Point 1:
Our analysis revealed that the "CCAFS LE40" launch site has the highest success rate among all sites, accounting for 43.7% of successful launches. This indicates that this site might have optimal conditions or processes that contribute to a higher success rate.

Point 2:
The scatter plot analysis showed that the "FT" booster version has a high success rate across various payload masses, demonstrating its reliability and robustness compared to other booster versions. This suggests that future missions might benefit from utilizing this booster version for improved success rates.

Point 3:
No clear pattern was observed linking higher payload masses to lower success rates, indicating that factors other than payload mass, such as launch site conditions and booster versions, play a more significant role in determining the outcome of a launch.

Conclusions

Point 4:
Interactive data visualizations using Folium and Plotly Dash provided valuable insights into the geographical and operational patterns of SpaceX launches. These tools allowed for a deeper understanding of the data, enabling stakeholders to make informed decisions based on comprehensive visual analytics.

- In conclusion, our predictive analysis and interactive visualizations have not only shed light on key factors influencing SpaceX's launch success but also provided a robust framework for future assessments and decision-making in the aerospace industry. The insights gathered can help improve launch strategies and contribute to the ongoing success of reusable rocket technology.

Appendix

jupyter 1-jupyter-labs-spacex-data-collection-api-v2 (1) Last Checkpoint: 2 days ago

File Edit View Run Kernel Settings Help Trusted

Markdown Open in... Python [conda env:base] *

[17]:
```python
# Requests allows us to make HTTP requests which we will use to get data from an API
import requests
# Pandas is a software library written for the Python programming language for data manipulation and analysis.
import pandas as pd
# NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection
import numpy as np
# Datetime is a library that allows us to represent dates
import datetime

# Setting this option will print all collumns of a dataframe
pd.set_option('display.max_columns', None)
# Setting this option will print all of the data in a feature
pd.set_option('display.max_colwidth', None)
```

Below we will define a series of helper functions that will help us use the API to extract information using identification numbers in the launch data.

From the `rocket` column we would like to learn the booster name.

[19]:
```python
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
       if x:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])
```

From the `launchpad` we would like to know the name of the launch site being used, the logitude, and the latitude.

[21]:
```python
# Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
       if x:
         response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
         Longitude.append(response['longitude'])
         Latitude.append(response['latitude'])
         LaunchSite.append(response['name'])
```

From the `payload` we would like to learn the mass of the payload and the orbit that it is going to.

Activ

[23]:
```python
# Takes the dataset and uses the payloads column to call the API and append the data to the lists
def getPayloadData(data):
    for load in data['payloads']:
       if load:
        response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
        PayloadMass.append(response['mass_kg'])
        Orbit.append(response['orbit'])
```

From `cores` we would like to learn the outcome of the landing, the type of the landing, number of flights with that core, whether gridfins were used, wheter the core is reused, wheter legs were used, the landing pad used, the block of the core which is a number used to seperate version of cores, the number of times this specific core has been reused, and the serial of the core.

[25]:
```python
# Takes the dataset and uses the cores column to call the API and append the data to the lists
def getCoreData(data):
    for core in data['cores']:
            if core['core'] != None:
                response = requests.get("https://api.spacexdata.com/v4/cores/"+core['core']).json()
                Block.append(response['block'])
                ReusedCount.append(response['reuse_count'])
                Serial.append(response['serial'])
            else:
                Block.append(None)
                ReusedCount.append(None)
                Serial.append(None)
            Outcome.append(str(core['landing_success'])+' '+str(core['landing_type']))
            Flights.append(core['flight'])
            GridFins.append(core['gridfins'])
            Reused.append(core['reused'])
            Legs.append(core['legs'])
            LandingPad.append(core['landpad'])
```

Now let's start requesting rocket launch data from SpaceX API with the following URL:

[27]:
```python
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

[28]:
```python
response = requests.get(spacex_url)
```

Check the content of the response

Act

[18]:
```sql
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

* sqlite:///my_data1.db
Done.

[18]:

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

Thank you!