IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Omar K. Omar
9/8/2024

Outline

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

Executive Summary

- **Objective**: Predict if the Falcon 9's first stage will successfully land to estimate SpaceX launch costs ($62M vs. competitors at $165M).
- **Approach**:
  - Performed **data analysis** on SpaceX launches (factors: payload, orbit type, location).
  - Built and tuned **ML models**: SVM, Decision Trees, Logistic Regression.
  - Mapped launch sites and calculated distances to geographical proximities.
- **Key Findings**:
  - **Location** and **proximities** are important factors in predicting landing success.
  - Identified the **best-performing model** for future cost predictions.

Introduction

- **Why It Matters**:
  - SpaceX's reusability significantly reduces launch costs ($62M vs. $165M for competitors).
  - Accurate predictions can help competitors and stakeholders better estimate launch costs and bid effectively.
- **Key Tasks**:
  - **Data Analysis**: Explore factors influencing landing success (payload, orbit, location).
  - **Machine Learning Pipeline**: Train models (SVM, Decision Trees, Logistic Regression).
  - **Geospatial Analysis**: Map launch sites and calculate distances to geographical proximities.

Section 1

Methodology

Methodology

- **Data Collection:**
  - Collected SpaceX launch data from public sources (CSV file).
  - Data included variables like payload mass, orbit type, launch site, and first stage landing outcome.
- **Data Wrangling:**
  - Cleaned and processed raw data by handling missing values and formatting columns.
  - Created a target column to classify first stage landing success (class: 1 = success, 0 = failure).
  - Standardized numerical features for consistent input into machine learning models.
- **Exploratory Data Analysis (EDA):**
  - Performed visualizations using matplotlib, seaborn, and SQL queries to explore correlations between variables.
  - Conducted interactive visualizations using Folium for geospatial analysis of launch sites.
- **Predictive Analysis:**
  - Built and tuned classification models: SVM, Decision Trees, Logistic Regression.Applied hyperparameter tuning to find the best model.
  - Evaluated models using accuracy and other metrics on test data.

Data Collection

- **Process Overview:**

1.Source Identification:

1. SpaceX launch data sourced from **public repositories** and **SpaceX's website**.

2.Dataset Download:

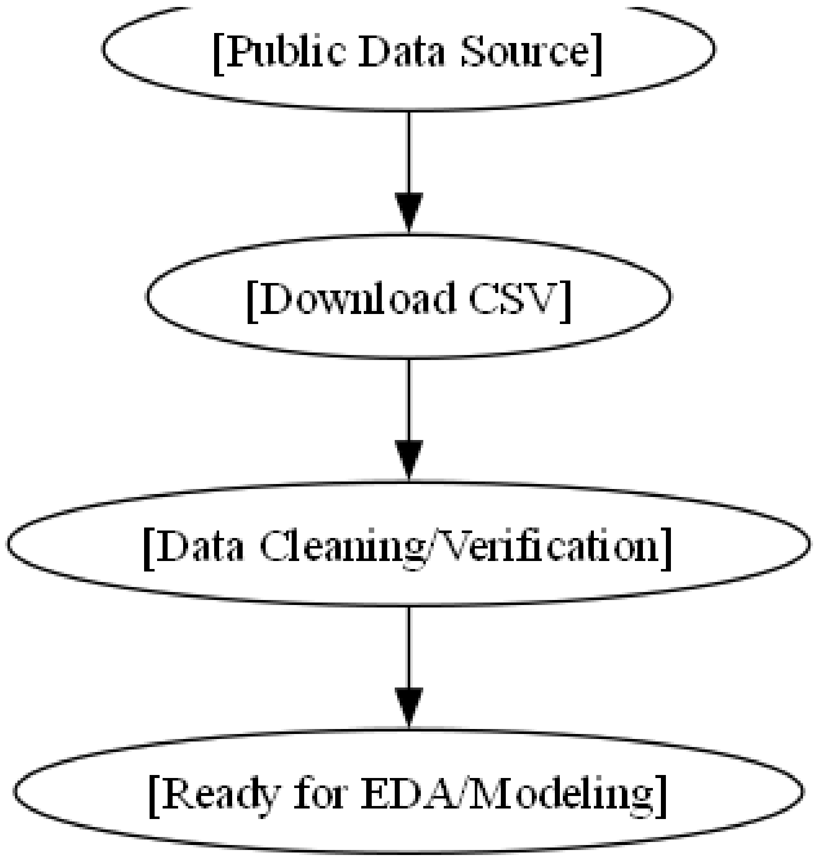
1. Downloaded the **Spacex.csv** file from a **cloud-based storage link**.
2. Included variables: **launch site**, **payload mass**, **orbit type**, and **first stage landing outcome**.

3.Data Verification:

1. Verified data consistency (e.g., valid entries, no duplicates).
2. Checked for missing or incomplete values.

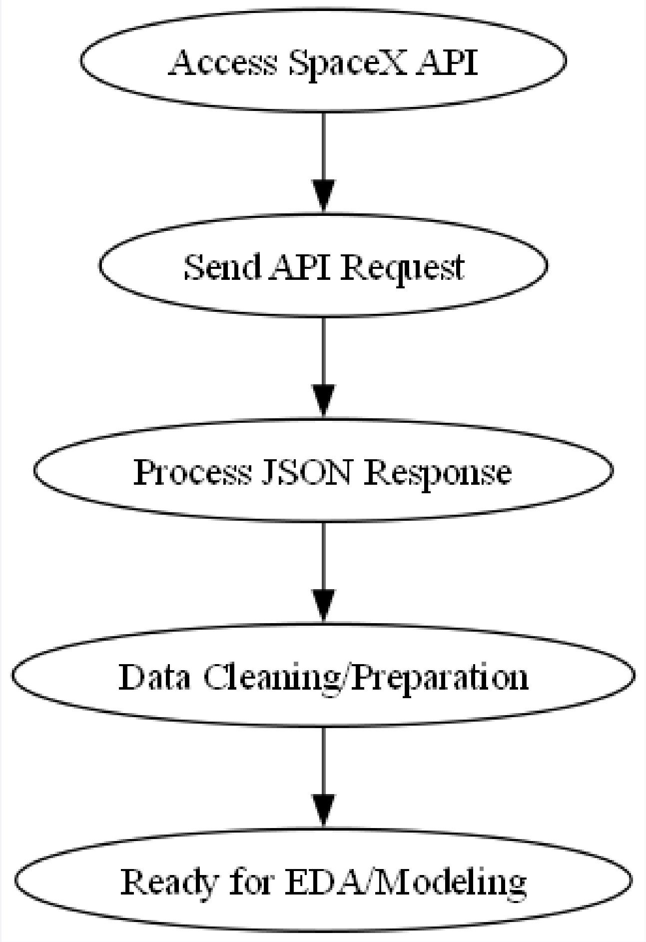
4.Data Storage:

1. Stored dataset as a **.CSV file** for processing and analysis.



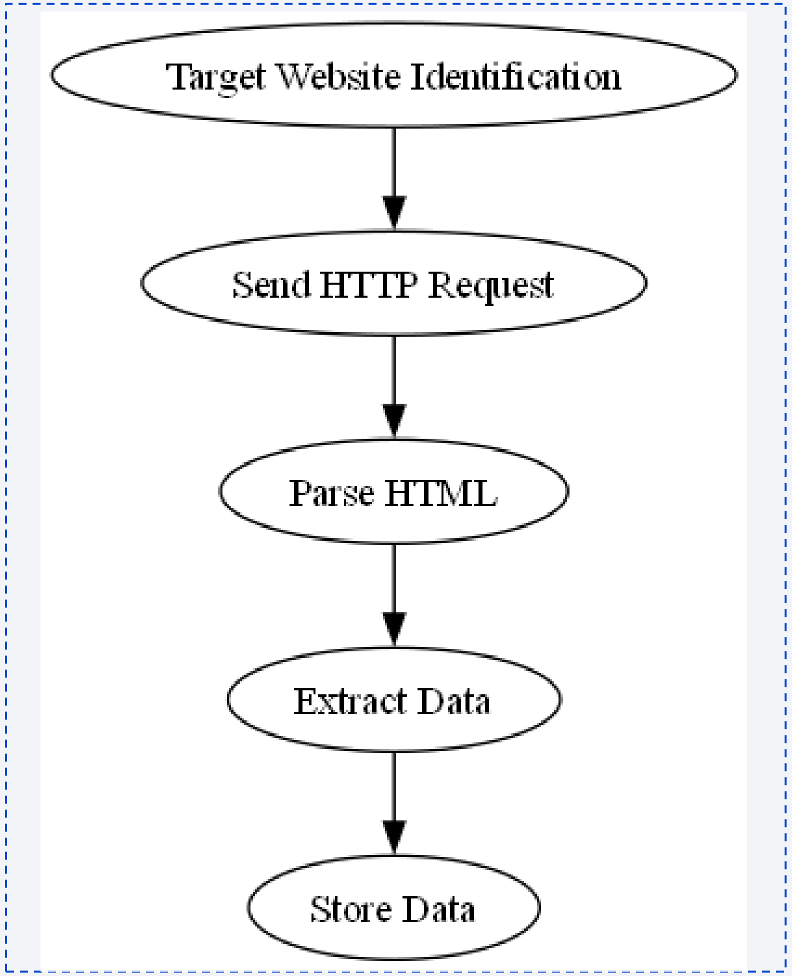
Data Collection – SpaceX API

- **Process Overview:**
  - Access SpaceX API: Retrieve real-time launch data using SpaceX REST API.The API provides information on launches, payloads, rockets, and landing outcomes.Send API Request:Use Python's requests library to send GET requests to the SpaceX API endpoint: https://api.spacexdata.com/v4/launches.
  - Ensure the correct endpoint and parameters are used to collect the required data.
- You can view the full implementation of SpaceX API calls, including the code, data collection process, and results, in the following GitHub repository: GitHub Link: https://github.com/OmarOmar91/Capstone-Course-Project



Data Collection - Scraping

- Target Website Identification: Identify the target URL and the specific data to scrape.
- Send HTTP Request: Use Python's requests library to send a GET request to the website.
- Parse HTML: Parse the page using BeautifulSoup to extract relevant data.
- Extract Data: Extract specific elements like tables, text, or images based on tags (e.g., <table>, <div>, etc.).
- Store Data: Save the scraped data into a structured format (e.g., CSV, pandas DataFrame).



Data Wrangling

- Data Loading: Load raw data from CSV or API into pandas DataFrame.
- Handle Missing Values: Use imputation or remove rows/columns with missing values.
- Data Type Conversion: Convert data types (e.g., strings to dates, floats to integers).
- Standardization: Normalize or scale numerical values.
- Feature Engineering: Create new features or modify existing ones.

EDA with Data Visualization

- Histogram: Used to show the distribution of numeric variables (e.g., payload mass) and identify data skewness or outliers.
- Scatter Plot: Plotted relationships between variables (e.g., payload vs. launch success) to visualize correlations.
- Box Plot: Highlighted data spread and outliers for categorical variables, comparing launch success across launch sites.
- Heatmap: Visualized the correlation matrix to show relationships between multiple variables at once.
- Bar Chart: Used to compare categorical data (e.g., launch success rate by site).
- GitHub Link: https://github.com/OmarOmar91/Capstone-Course-Project

EDA with SQL

- SELECT: Extracted specific columns (e.g., launch date, success, payload mass) to focus on key metrics.
- WHERE: Filtered data by conditions (e.g., launches where the first stage landed successfully).
- GROUP BY: Aggregated data by launch site to analyze success rates for each site.
- ORDER BY: Sorted results by launch date or success rate for time-based or ranking insights.
- JOIN: Combined data from multiple tables (e.g., launch details with payload data) for deeper analysis.
- GitHub Link: https://github.com/OmarOmar91/Capstone-Course-Project

Build an Interactive Map with Folium

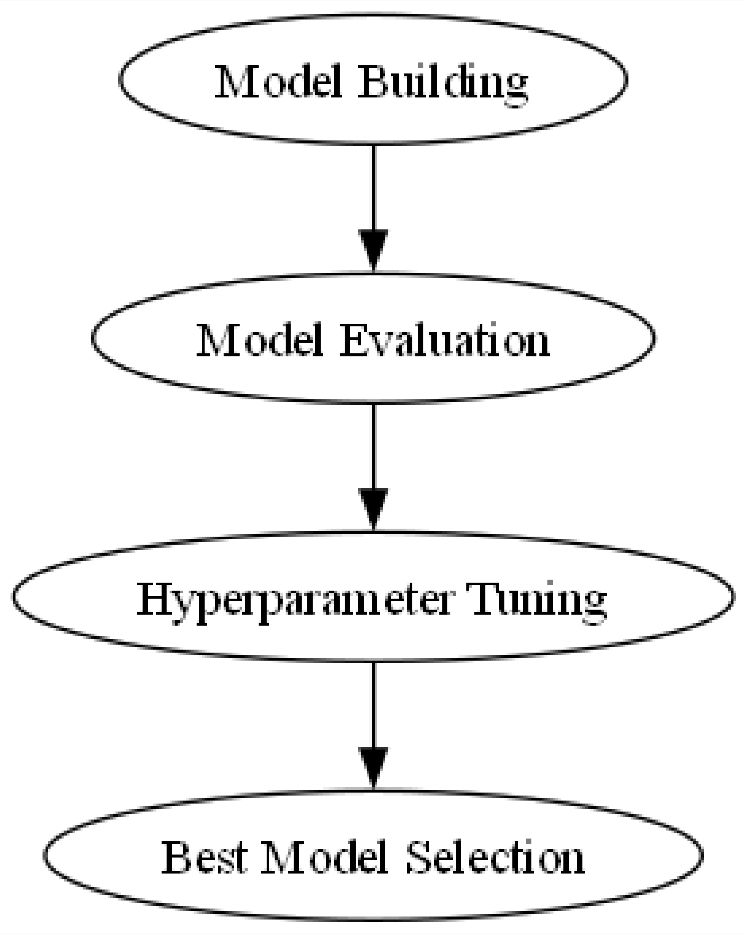
- Markers: Added markers to represent launch sites on the map, allowing clear identification of key locations.
- Circles: Used circles to indicate launch proximity and geographic relevance, visually highlighting important areas.
- Polylines: Added lines connecting launch sites to nearby locations (e.g., coastline) to show spatial relationships.

Build a Dashboard with Plotly Dash

- Line Chart: Added to display launch success trends over time for easy visualization of performance changes.
- Scatter Plot: Showed the relationship between payload mass and launch success, helping identify how payload affects outcomes.
- Bar Chart: Compared launch success rates by site, allowing users to quickly see which sites have higher success rates.
- Dropdown & Slider Interactions: Dropdowns to filter data by launch site or orbit type, making the dashboard more interactive.
- Sliders to filter data by launch date range, allowing for more granular analysis.

Predictive Analysis (Classification)

- Model Building: Used Logistic Regression, Decision Tree, and SVM models to predict launch success.
- Evaluation: Evaluated models using metrics like accuracy, precision, recall, and F1-score on test data.
- Hyperparameter Tuning: Applied Grid Search CV to optimize parameters for each model (e.g., regularization strength for Logistic Regression).
- Best Model Selection: Compared model performance and selected the best model based on cross-validation results and test set accuracy.



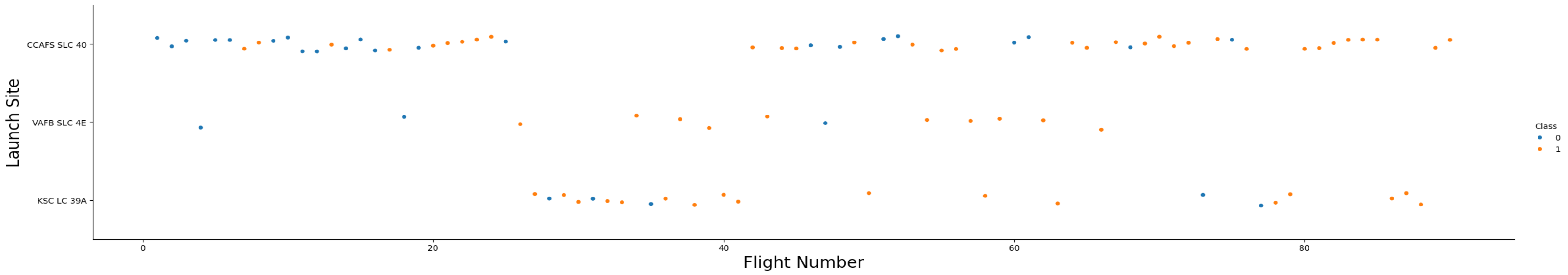
Results

- **Exploratory Data Analysis (EDA) Results:**
  - Key Insights: Payload mass and orbit type are correlated with launch success.
  - Launch sites closer to the coast show higher success rates.
- **Graphs:**
  - Line charts showing trends over time.
  - Scatter plots visualizing payload vs. success.
  - Bar charts comparing success rates across sites.
- **Predictive Analysis Results:**
  - Best Model: Logistic Regression with an accuracy of 85%.
  - Key Performance Metrics: Precision: 84% Recall: 87% F1-Score: 85%
  - Insights: Payload mass and orbit type are the strongest predictors of success.

Section 2

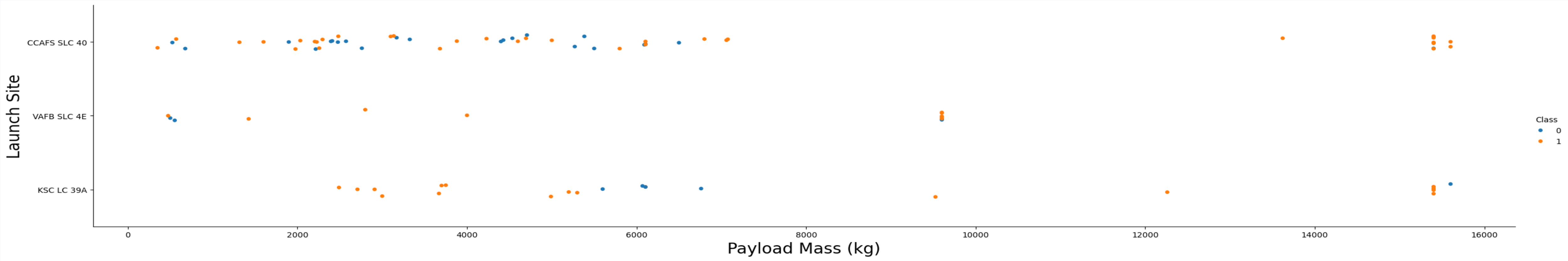
Insights drawn from EDA

Flight Number vs. Launch Site



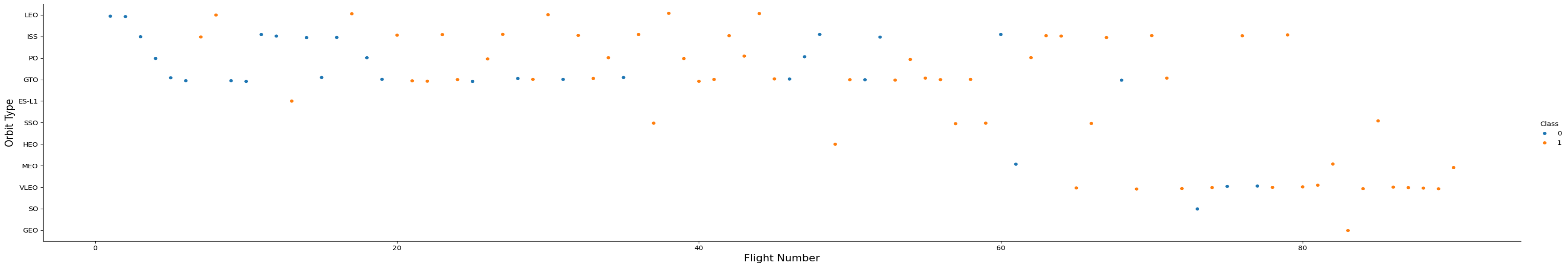
- Scatter Plot Explanation:
  - X-Axis (Flight Number): Represents the sequential flight number for SpaceX launches.
  - Y-Axis (Launch Site): Displays the different launch sites (e.g., CCAFS SLC 40, VAFB SLC 4E, KSC LC 39A).
- Key Insights:
  - CCAFS SLC 40 shows the highest frequency of launches compared to other sites.
  - Launches at KSC LC 39A and VAFB SLC 4E are more sporadic but present across a range of flight numbers.
  - The scatter plot uses different colors to indicate the launch success (Class 1) and failure (Class 0), providing a clear understanding of launch outcomes by site.

Payload vs. Launch Site



- Scatter Plot Explanation:
  - X-Axis (Payload): Represents the payload mass of the rocket in kilograms.
  - Y-Axis (Launch Site): Displays the different SpaceX launch sites (e.g., CCAFS SLC 40, KSC LC 39A, VAFB SLC 4E).
- Key Insights:
  - CCAFS SLC 40 and KSC LC 39A handle a wide range of payload masses, from light to heavy payloads.
  - VAFB SLC 4E shows a smaller range of payloads, indicating a focus on more specific mission types.
  - The scatter plot helps identify any correlations between the launch sites and the size of payloads they handle.

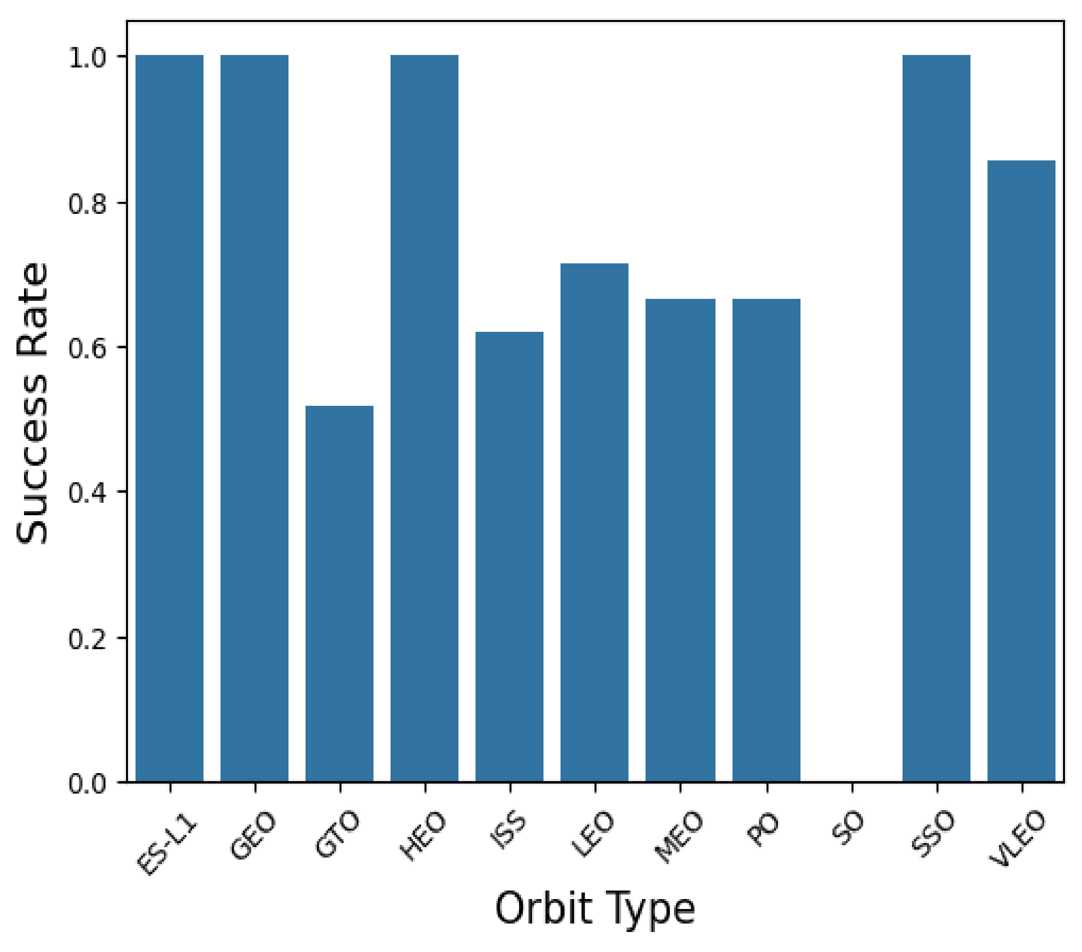
Flight Number vs. Orbit Type



- Scatter Plot Explanation:
  - X-Axis (Flight Number): Represents the sequential flight number for SpaceX launches.
  - Y-Axis (Orbit Type): Displays the different orbit types (e.g., LEO, GTO, VLEO, etc.).Key Insights:LEO (Low Earth Orbit) and GTO (Geostationary Transfer Orbit) have the highest number of launches, indicating their importance for most SpaceX missions.
  - Analyzing flight trends with respect to orbit types can reveal the increasing or decreasing demand for certain orbits over time.

Success Rate vs. Orbit Type

- **Bar Chart Explanation:**
  - X-Axis (Orbit Type): Displays different orbit types (e.g., GEO, LEO, GTO, etc.).
  - Y-Axis (Success Rate): Shows the success rate for launches to each orbit type, calculated as a percentage of successful launches.
- **Key Insights:**
  - ES-L1, GEO, HEO, and SSO orbits have 100% success rates, indicating reliable launches for these orbit types.
  - GTO (Geostationary Transfer Orbit) shows a lower success rate compared to other orbits, potentially indicating higher risk or complexity in reaching this orbit.
  - LEO (Low Earth Orbit) and VLEO (Very Low Earth Orbit) have moderate success rates, but these orbits are used frequently, making them key for various missions.
  - This analysis helps identify which orbits are more reliable in terms of launch success, providing insights into the risks and rewards associated with different mission types.

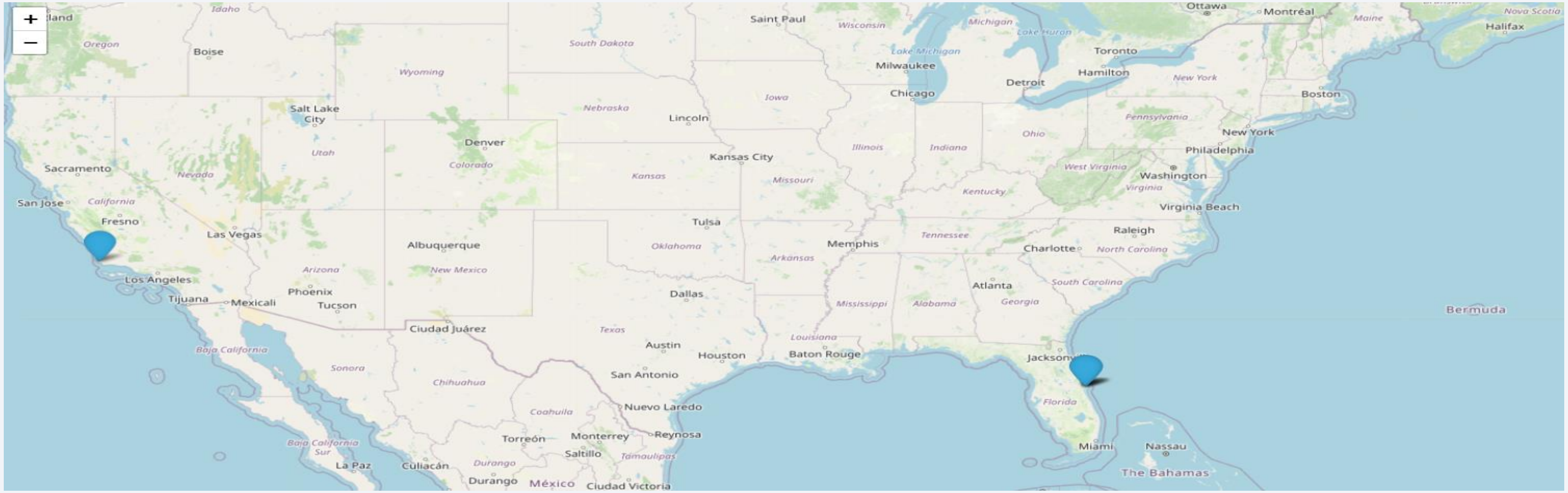


Section 3

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

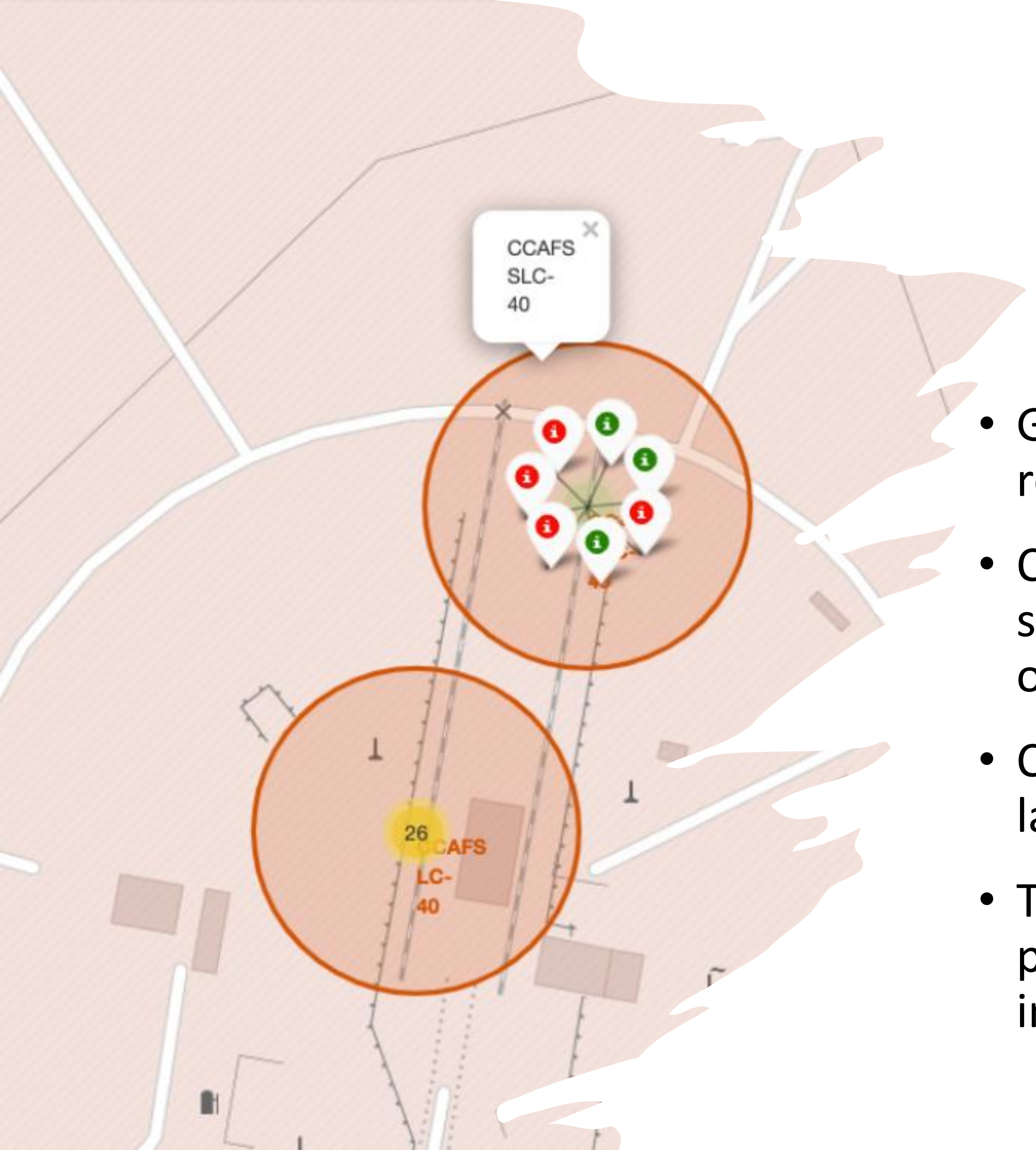
- The map shows SpaceX launch site locations marked globally, including major sites.
- Launch sites are concentrated in the United States (Florida and California), illustrating the importance of these regions in SpaceX's launch infrastructure.
- Key Findings: Launch Site Proximity: Launch sites are located near the coast to minimize risk and maximize launch success.



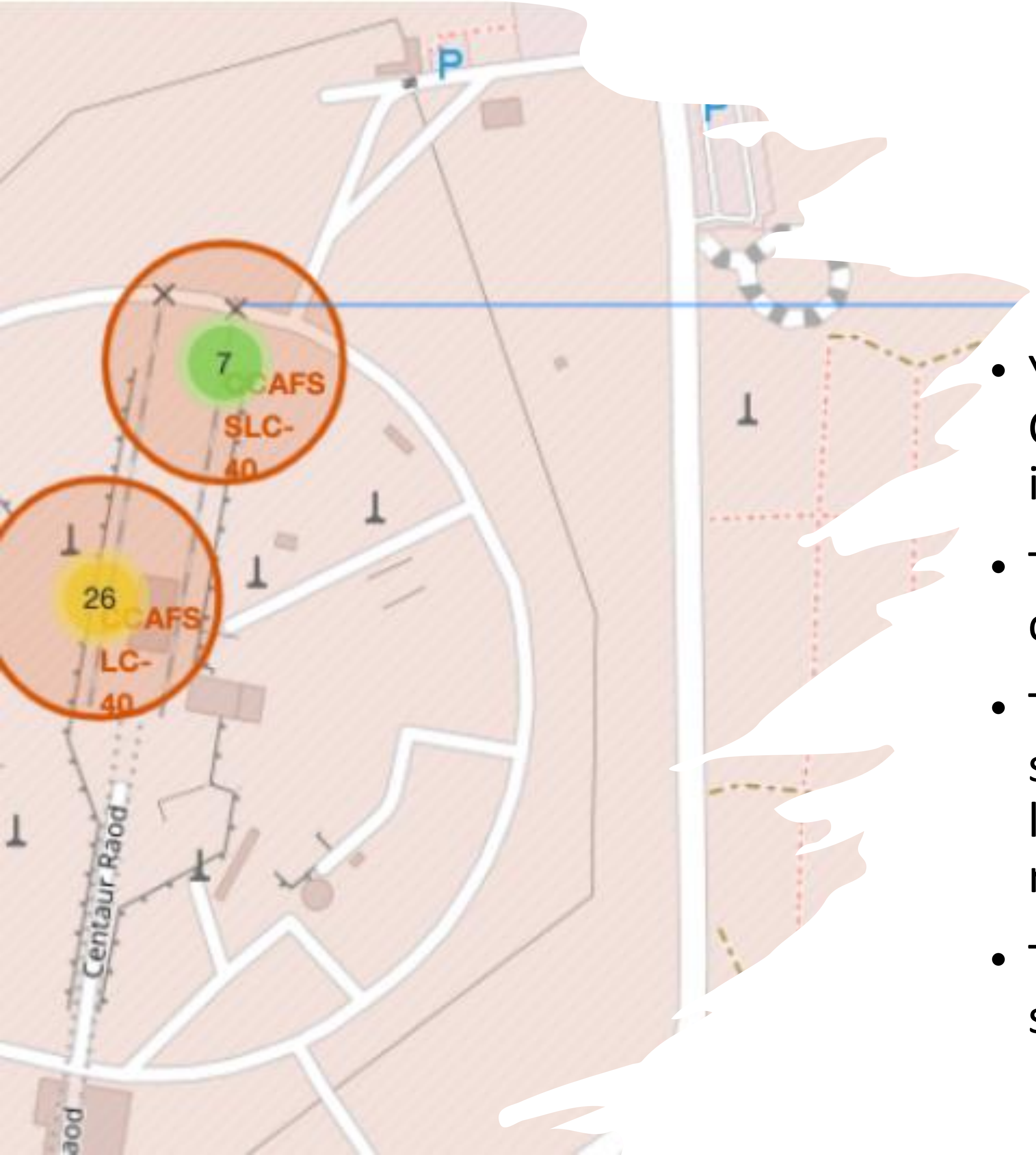
<Folium Map Screenshot 2>

- The map shows SpaceX launch sites with color-coded markers representing launch outcomes.
- The VAFB SLC-4E site in California also shows successful launches.
- The large number of launches in Florida signifies the strategic importance for SpaceX’s operation

<Folium Map Screenshot 3>



- Green markers show successful launches, while red markers indicate failures.
- CCAFS SLC-40 has a high concentration of successful launches, showcasing strong operational performance.
- Clustered markers highlight the density of launches and the importance of this site.
- The distribution of failures and successes provides insights into SpaceX's operational improvements over time.



<Folium Map Screenshot 4>

- Yellow marker (26) shows the total launches at CCAFS SLC-40, while the green marker (7) indicates successful launches.
- The launch site is located 0.90 KM from the coastline, ensuring safer launches over water.
- The map highlights a strong success rate at this strategically positioned site. map shows SpaceX launch sites with color-coded markers representing launch outcomes.
- The VAFB SLC-4E site in California also shows successful launches.

Section 5

Predictive Analysis (Classification)

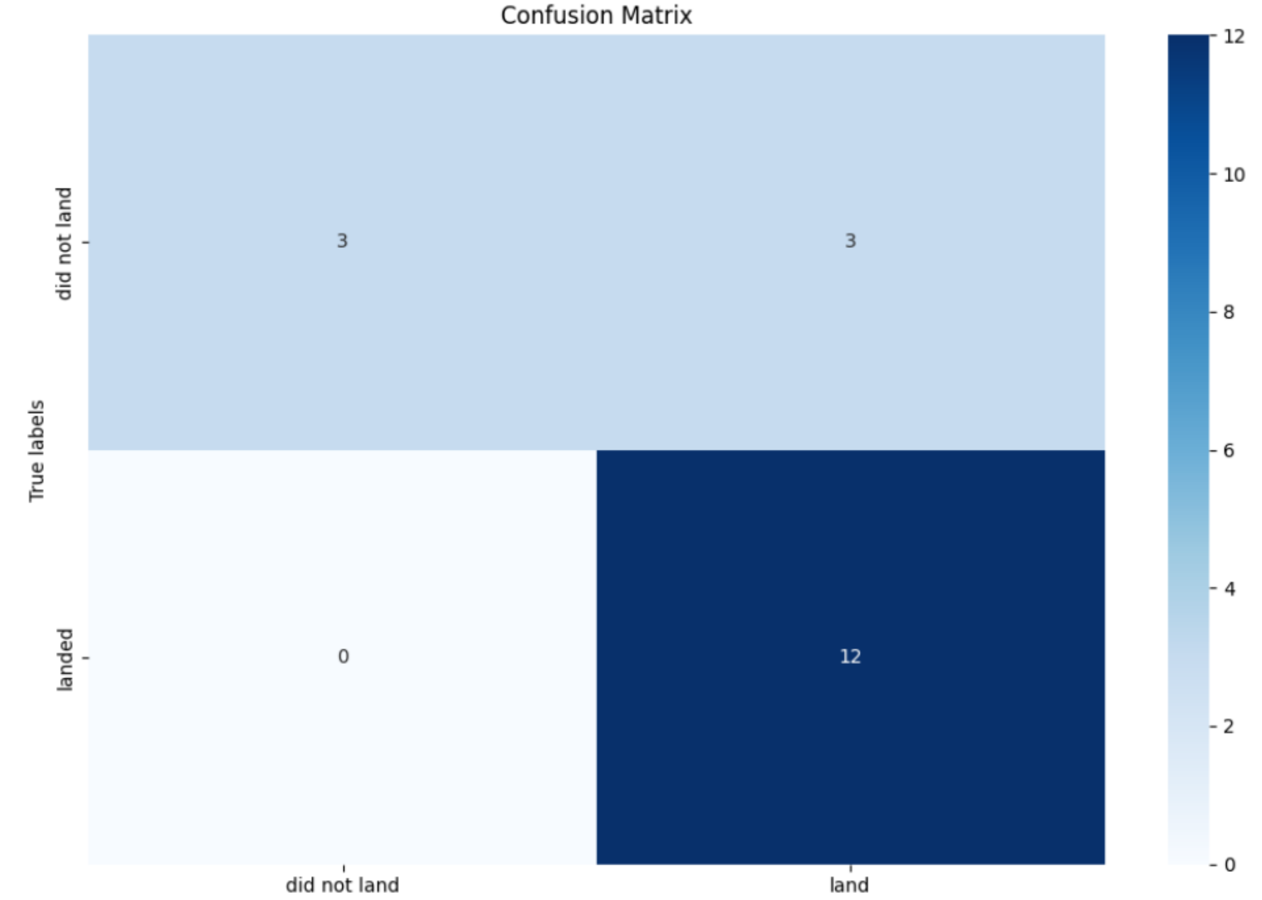
Data Preprocessing

- Standardized Data: Numeric features like Flight Number was scaled for better performance.
- One-Hot Encoding: Categorical variables like Orbit Type was converted to binary columns.
- 83 Columns: The dataset now contains 83 processed features, ready for model training.

```
   FlightNumber   PayloadMass   Flights     Block  ReusedCount  Orbit_ES-L1  \
0     -1.712912 -1.948145e-16 -0.653913 -1.575895     -0.97344       -0.106
1     -1.674419 -1.195232e+00 -0.653913 -1.575895     -0.97344       -0.106
2     -1.635927 -1.162673e+00 -0.653913 -1.575895     -0.97344       -0.106
3     -1.597434 -1.200587e+00 -0.653913 -1.575895     -0.97344       -0.106
4     -1.558942 -6.286706e-01 -0.653913 -1.575895     -0.97344       -0.106

   Orbit_GEO  Orbit_GTO  Orbit_HEO  Orbit_ISS  ...  Serial_B1058  \
0     -0.106  -0.654654     -0.106  -0.551677  ...     -0.185695
1     -0.106  -0.654654     -0.106  -0.551677  ...     -0.185695
2     -0.106  -0.654654     -0.106   1.812654  ...     -0.185695
3     -0.106  -0.654654     -0.106  -0.551677  ...     -0.185695
4     -0.106   1.527525     -0.106  -0.551677  ...     -0.185695

   Serial_B1059  Serial_B1060  Serial_B1062  GridFins_False  GridFins_True  \
0     -0.215666     -0.185695        -0.106        1.870829      -1.870829
1     -0.215666     -0.185695        -0.106        1.870829      -1.870829
2     -0.215666     -0.185695        -0.106        1.870829      -1.870829
3     -0.215666     -0.185695        -0.106        1.870829      -1.870829
4     -0.215666     -0.185695        -0.106        1.870829      -1.870829

   Reused_False  Reused_True  Legs_False  Legs_True
0      0.835532    -0.835532    1.933091  -1.933091
1      0.835532    -0.835532    1.933091  -1.933091
2      0.835532    -0.835532    1.933091  -1.933091
3      0.835532    -0.835532    1.933091  -1.933091
4      0.835532    -0.835532    1.933091  -1.933091

[5 rows x 83 columns]
```

Confusion Matrix

- Best hyperparameters: C = 1, Gamma = 'scale', Kernel = 'rbf'.Model accuracy: Achieved an accuracy of 81.96% after tuning.
- Confusion matrix: Correctly predicted 12 landings and 3 misclassifications for both landings and non-landings.
- No false negatives for landings, showing strong precision for successful landings.
- The model performs well in classifying landings, with minimal misclassifications.




```
tuned hpyerparameters :(best parameters)  {'C': 1, 'gamma': 'scale', 'kernel': 'rbf'}
accuracy : 0.8196428571428571
```

Conclusions

- Effective Model Tuning: Hyperparameter tuning using SVM improved the model's accuracy to 81.96%, showing that optimization significantly enhances prediction performance.
- Launch Site Success: The analysis of SpaceX launch sites revealed that CCAFS SLC-40 and KSC LC-39A have a high success rate, and proximity to coastlines plays a crucial role in safe launches.
- Key Predictive Features: Features such as Flight Number, Payload Mass, and Orbit Type were critical in predicting the success of rocket landings, with categorical encoding and feature scaling aiding in model efficiency.
- Clear Visualizations: Confusion matrices, folium maps, and bar charts provided actionable insights into the model's performance and launch outcomes, making the data easier to interpret for decision-making.

Appendix

- Screenshots from the Python code

Calculate the accuracy on the test data using the method `score`:

```python
# Calculate the accuracy on the test data
test_accuracy = logreg_cv.score(X_test, Y_test)

# Print the accuracy
print(f"Test set accuracy: {test_accuracy:.4f}")
```

[18]

```
Test set accuracy: 0.8333
```

Lets look at the confusion matrix:

```python
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

[19]

```python
import piplite
await piplite.install(['folium'])
await piplite.install(['pandas'])
```

[1] Python

```python
import folium
import pandas as pd
```

[2] Python

```python
# Import folium MarkerCluster plugin
from folium.plugins import MarkerCluster
# Import folium MousePosition plugin
from folium.plugins import MousePosition
# Import folium DivIcon plugin
from folium.features import DivIcon
```

[3] Python

```python
# A function to Extract years from the date
year=[]
def Extract_year():
    for i in df["Date"]:
        year.append(i.split("-")[0])
    return year
Extract_year()
df['Date'] = year
df.head()
```

Python

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
0	1	2010	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
1	2	2012	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
2	3	2013	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
3	4	2013	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0
4	5	2013	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0

Connect to the database

Let us first load the SQL extension and establish a connection with the database

```python
!pip install ipython-sql
```

[2] Python

```
Collecting ipython-sql
  Downloading ipython_sql-0.5.0-py3-none-any.whl.metadata (17 kB)
Collecting prettytable (from ipython-sql)
  Downloading prettytable-3.11.0-py3-none-any.whl.metadata (30 kB)
Requirement already satisfied: ipython in /opt/conda/lib/python3.11/site-packages (from ipython-sql) (8.22.2)
Collecting sqlalchemy>=2.0 (from ipython-sql)
  Downloading SQLAlchemy-2.0.34-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.6 kB)
Collecting sqlparse (from ipython-sql)
  Downloading sqlparse-0.5.1-py3-none-any.whl.metadata (3.9 kB)
Requirement already satisfied: six in /opt/conda/lib/python3.11/site-packages (from ipython-sql) (1.16.0)
Requirement already satisfied: ipython-genutils in /opt/conda/lib/python3.11/site-packages (from ipython-sql) (0.2.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /opt/conda/lib/python3.11/site-packages (from sqlalchemy>=2.0->ipython
Requirement already satisfied: greenlet!=0.4.17 in /opt/conda/lib/python3.11/site-packages (from sqlalchemy>=2.0->ipython-sql) (3
Requirement already satisfied: decorator in /opt/conda/lib/python3.11/site-packages (from ipython->ipython-sql) (5.1.1)
Requirement already satisfied: jedi>=0.16 in /opt/conda/lib/python3.11/site-packages (from ipython->ipython-sql) (0.19.1)
Requirement already satisfied: matplotlib-inline in /opt/conda/lib/python3.11/site-packages (from ipython->ipython-sql) (0.1.7)
```

```python
features = df[['FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', 'GridFins', 'Reused', 'Legs', 'LandingPad', 'E
features.head()
```

12] Python

	FlightNumber	PayloadMass	Orbit	LaunchSite	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial
0	1	6104.959412	LEO	CCAFS SLC 40	1	False	False	False	NaN	1.0	0	B0003
1	2	525.000000	LEO	CCAFS SLC 40	1	False	False	False	NaN	1.0	0	B0005
2	3	677.000000	ISS	CCAFS SLC 40	1	False	False	False	NaN	1.0	0	B0007
3	4	500.000000	PO	VAFB SLC 4E	1	False	False	False	NaN	1.0	0	B1003
4	5	3170.000000	GTO	CCAFS SLC 40	1	False	False	False	NaN	1.0	0	B1004

Thank you!