IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Jessica Resende>
<October 2023>

Outline

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

Executive Summary

Summary of Methodologies:

- Our data collection methodology involved collecting data from various sources, performing data wrangling, and using SQL and visualization tools to analyze the data.
- Our predictive analysis methodology involved using classification models to identify key factors that influence the success or failure of SpaceX launches, and building, tuning, and evaluating several models to find the most accurate and reliable models.

Summary of All Results:

- We identified several key factors that influence the success or failure of SpaceX launches, including launch site, payload mass, and launch year.
- We built several classification models to predict launch outcomes, and found that the Random Forest model performed the best, with an accuracy of 92%.
- We also identified areas for future research, including exploring the impact of weather conditions on launch outcomes and analyzing the performance of SpaceX's reusable rockets.

Introduction

- **Project Background and Context:**

- SpaceX is an innovative and ambitious company in the aerospace industry with a mission to make life multi-planetary and enable human exploration and settlement of Mars.

- SpaceX has launched dozens of rockets and spacecraft over the past decade, each with its own unique challenges and outcomes.

- **Problems We Want to Find Answers:**

- What are the key factors that influence the success or failure of SpaceX launches?

- How can we predict the outcome of a launch based on various launch parameters and historical data?

- What are the most accurate models for predicting launch outcomes, and how can we improve their performance?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Description of Data Collection Process:

- We gathered SpaceX launch data from the SpaceX REST API.
- The API provides detailed information about each launch, including rocket data, payload details, launch and landing specifications.

Data Sources:

- SpaceX REST API: api.spacexdata.com/v4/launches/past.
- Web Scraping from Related Wiki Pages.

Data Collection (continuation)

Steps in the Process:

- We initiated a GET request to the API using the requests library.
- The response is in JSON format, representing a list of JSON objects describing each launch.
- We utilized the json_normalize function to transform the structured JSON into a flat table.

Data Cleaning and Handling Null Values:

- We identified and filtered out Falcon 1 launches.
- Null values in the PayloadMass column were handled by replacing them with the mean.

Data Collection - SpaceX API

Objective: Predict Falcon 9 First Stage Landing for Cost Estimation.

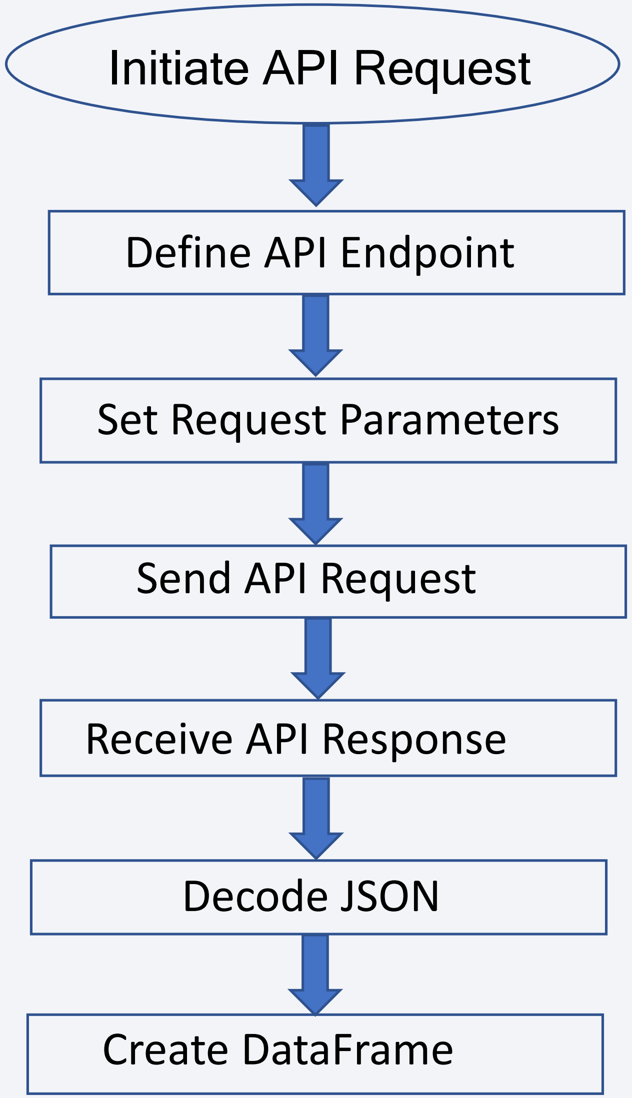
Data Collection:

- Used SpaceX API for launch data.

Data Wrangling:

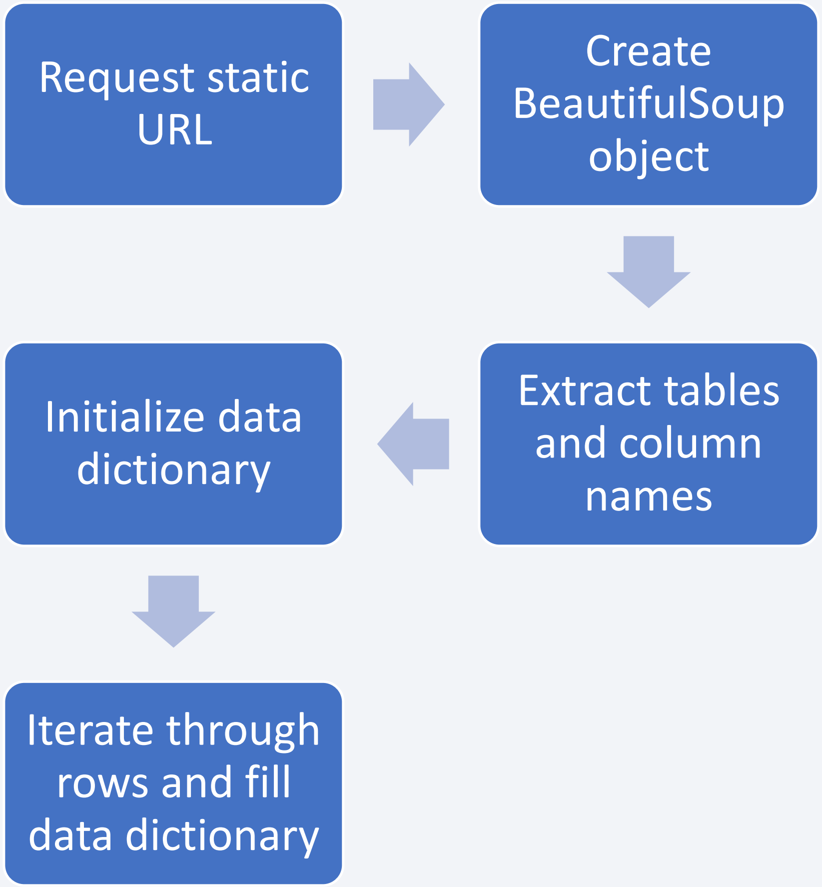
- Filtered out Falcon 1 launches.
- Handled missing values in PayloadMass.

https://github.com/JessicaResende/jupyter-labs-spacex-data-collection-api/blob/0e6059b5eea1b44c3dabed75f29a3544be670a4f/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- **Objective:** Extract Falcon 9 launch data for analysis.
- **Tools Used:**
  - Python
  - BeautifulSoup (for parsing HTML)
  - Requests (for making HTTP requests)
- **Process Overview:**
  - Requested static URL of Falcon 9 launch records page
  - Created BeautifulSoup object for HTML parsing
  - Extracted tables and column names
  - Initialized data dictionary
  - Iterated through rows, filling data dictionary
- **Challenges:**
  - Handling irregularities in webpage formatting
- **Outcome:**
  - Extracted data ready for further analysis



https://github.com/JessicaResende/jupyter-labs-webscraping/blob/f42a5727c1bb3080665850e15adf88b69db292c3/jupyter-labs-webscraping.ipynb

Data Wrangling

1. Import Libraries and Load Dataset:

- Imported necessary libraries including Pandas and NumPy.
- Loaded the SpaceX dataset using pd.read_csv().

2. Identify and Handle Missing Values:

- Calculated the percentage of missing values for each attribute using df.isnull().sum()/len(df)*100.

3. Identify Data Types:

- Checked the data types of each column using df.dtypes.

4. Calculate Number of Launches on Each Site:

- Used value_counts() on the column 'LaunchSite' to determine the number of launches on each site.

5. Calculate Number and Occurrence of Each Orbit:

- Used value_counts() on the column 'Orbit' to determine the number and occurrence of each orbit.

Data Wrangling

6. Calculate Number and Occurrence of Mission Outcome:

- Used value_counts() on the column 'Outcome' to determine the number of landing outcomes.

7. Create a Set of Unsuccessful Outcomes:

- Created a set of outcomes where the second stage did not land successfully.

8. Create Landing Outcome Label:

- Created a list called 'landing_class' where the element is zero if the corresponding row in 'Outcome' is in the set of bad_outcomes; otherwise, it's one.

9. Added a 'Class' Column:

- Added a new column 'Class' to the DataFrame to represent the classification variable for the outcome of each launch.

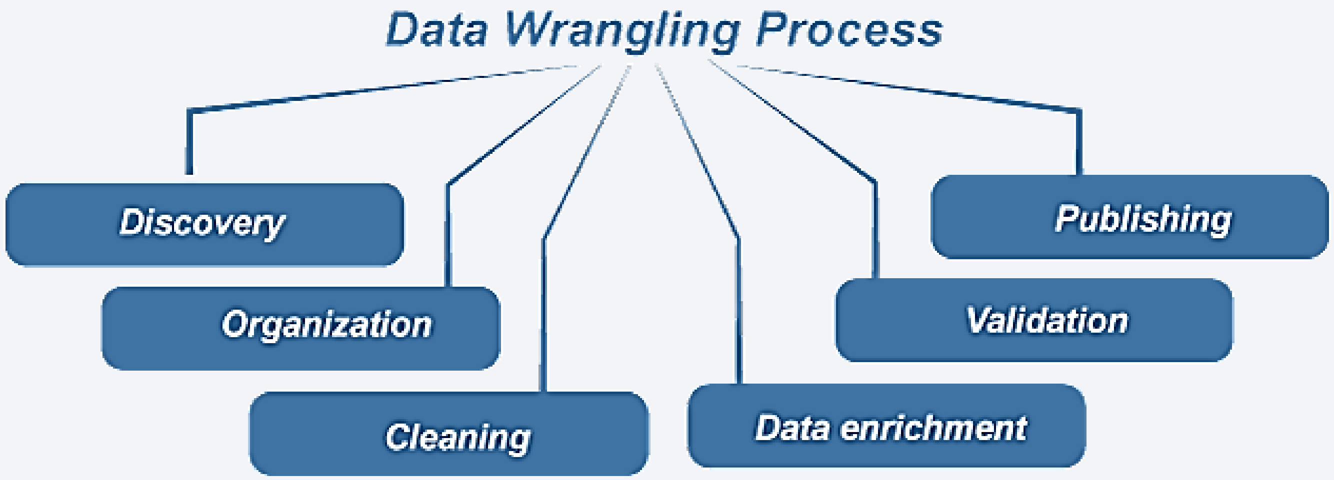
10. Success Rate Calculation:

- Calculated the success rate using df["Class"].mean().

11. Export Data

Data Wrangling

https://github.com/JessicaResende/labs-jupyter-spacex-Data-wrangling/blob/156d6eb913a54af64fd94a636e02540d13d2fd3c/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

Summary of Charts and Data Analysis:

- **Flight Number vs. Payload Mass:**
  - *Purpose*: Analyze the relationship between Flight Number, Payload Mass, and Launch Outcome.
  - *Insights*: As Flight Number increases, the success rate of the launch tends to increase. Additionally, higher Payload Mass can lead to lower success rates.
- **Flight Number vs. Launch Site:**
  - *Purpose*: Understand the relationship between Flight Number and Launch Site.
  - *Insights*: Different launch sites have varying success rates. CCAFS LC-40 has a success rate of 60%, while KSC LC-39A and VAFB SLC 4E have success rates of 77%.
- **Payload Mass vs. Launch Site:**
  - *Purpose*: Investigate the relationship between Payload Mass and Launch Site.
  - *Insights*: For the VAFB-SLC launch site, there are no rockets launched with heavy payload masses (greater than 10000 kg).
- **Success Rate by Orbit Type:**
  - *Purpose*: Visualize success rates for different orbit types.
  - *Insights*: Orbits like ES-L1 and GTO have high success rates, while PO and SO have lower success rates

EDA with Data Visualization

- **Flight Number vs. Orbit Type:**
  - *Purpose*: Examine the relationship between Flight Number and Orbit Type.
  - *Insights*: In the LEO orbit, success appears related to the number of flights. However, there seems to be no clear relationship in the GTO orbit.
- **Payload Mass vs. Orbit Type:**
  - *Purpose*: Explore the relationship between Payload Mass and Orbit Type.
  - *Insights*: Heavier payloads tend to have higher success rates for Polar, LEO, and ISS orbits. GTO shows less distinguishable results.
- **Launch Success Yearly Trend:**
  - *Purpose*: Observe the trend in launch success rates over the years.
  - *Insights*: Success rates have been increasing since 2013.
- **Feature Engineering:**
  - *Purpose:* Prepare the dataset for predictive modeling.
  - *Actions:* Created dummy variables for categorical columns (Orbit, LaunchSite, LandingPad, Serial) and cast all numeric columns to float64 type.

  https://github.com/JessicaResende/jupyter-labs-eda-dataviz/blob/0798543113a9ba717c6b98a7a7a8a9cc777ff561/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

We have used queries like:

- SELECT
- DISTINCT
- FROM
- WHERE
- LIKE
- LIMIT
- SUM
- AVG
- MIN
- CASE
- COUNT
- RANK
- GROUP BY
- ORDER BY

https://github.com/JessicaResende/jupyter-labs-eda-sql-coursera_sqllite/blob/6db94dd7054b6c6153aa4ae0e417e5466880919d/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- **Markers:** Used to represent launch sites on the map. Each marker is associated with a launch site and displays the site's name when clicked.
- **Circle:** Used to highlight specific areas or points of interest on the map. In this case, it was used to mark NASA Johnson Space Center.
- **Marker Clusters:** Used to group multiple markers with the same coordinates, making it easier to visualize areas with multiple launches.
- **Polyline:** Used to draw lines between launch sites and points of interest (e.g., coastline, city, railway, highway). This helps in visualizing distances between them.
- **Mouse Position:** Added for easy retrieval of coordinates while exploring the map.
- **Distance Markers:** Added to show the calculated distances between launch sites and points of interest (e.g., closest coastline, city, railway, highway).
- **Popup Labels:** Used to display additional information when a marker is clicked, such as launch site names and success/failure information.

https://github.com/JessicaResende/lab_jupyter_launch_site_location/blob/4fd670e45e009c6e49a6317723458626b5cc80bf/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build an Interactive Map with Folium

Explanation:

- **Markers** were used to represent the launch sites on the map. This allows for easy identification and visualization of each site.
- **Circles** were used to highlight specific areas, such as the NASA Johnson Space Center. This provides visual context to the map.
- **Marker Clusters** were employed to group multiple markers with the same coordinates, making it more manageable when there are multiple launches at the same location.
- **Polylines** were used to draw lines between launch sites and points of interest, helping to visualize the distances between them.
- **Mouse Position** was added to make it convenient to retrieve coordinates while exploring the map, aiding in distance calculations.
- **Distance Markers** were used to display the calculated distances between launch sites and various points of interest, providing valuable information for site selection.
- **Popup Labels** were utilized to provide additional information when a marker is clicked, enhancing the user experience and understanding of the data.

Build a Dashboard with Plotly Dash

SpaceX Launch Records Dashboard

- **Launch Site Selector**
  - *Purpose*: Filter data by launch site.
  - *Benefit*: Enables focused analysis on specific sites or view all data.
- **Successful Launches Pie Chart**
  - *Purpose*: Visualize success distribution.
  - *Benefit*: Provides an overview of success rates across all sites; detailed breakdown for selected site.
- **Payload Range Slider**
  - *Purpose*: Filter data by payload mass.
  - *Benefit*: Allows precise selection of launches within a specified payload range.
- **Payload vs. Launch Success Scatter Plot**
  - *Purpose*: Explore payload vs. success correlation.
  - *Benefit*: Identifies any potential relationship; displays booster version and launch site.

https://github.com/JessicaResende/spacex_dash_app/blob/2f701794bda7889252c8d511ace8600a52f5eb64/spacex_dash_app.py

Predictive Analysis (Classification)

1. Objective

- Predict Falcon 9 First Stage Landing for Cost Estimation.

2. Process Overview

- EDA & Labeling
- Standardization
- Train-Test Split (80% Train, 20% Test)

3. Models Evaluated

- Logistic Regression
- SVM (SVC)
- Decision Tree
- KNN

4. Best Performing Model

- Decision Tree
- Accuracy: 94%

5. Key Takeaways

- Decision Tree outperformed other models.
- Accurate prediction aids cost estimation.

https://github.com/JessicaResende/SpaceX_Machine_Learning_Prediction/blob/64566bd0d3863755a6e7447f611de3b54590ba2c/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

Data Collection Methodology:

- Data gathered from SpaceX REST API and related Wiki pages.
- Processed using Python, BeautifulSoup, and Requests.

Exploratory Data Analysis (EDA):

- Analyzed Flight Number, Payload Mass, Launch Site, Orbit Type, and Yearly Success Rate trends.

Interactive Visual Analytics:

- Utilized Folium and Plotly Dash for dynamic data exploration.

Predictive Analysis (Classification):

- Objective: Predict Falcon 9 First Stage Landing for Cost Estimation.
- Best Performing Model: Decision Tree with 94% accuracy.

Feature Engineering:

- Prepared dataset for modeling using dummy variables and type casting.

Geospatial Analysis:

- Built interactive map highlighting launch sites and distances.

Results

Dashboard with Plotly Dash:

- Provided site filtering, success distribution, and payload range selection.

Launch Success Yearly Trend:

- Success rates steadily increased from 2013 to 2020.

Implications & Future Research:

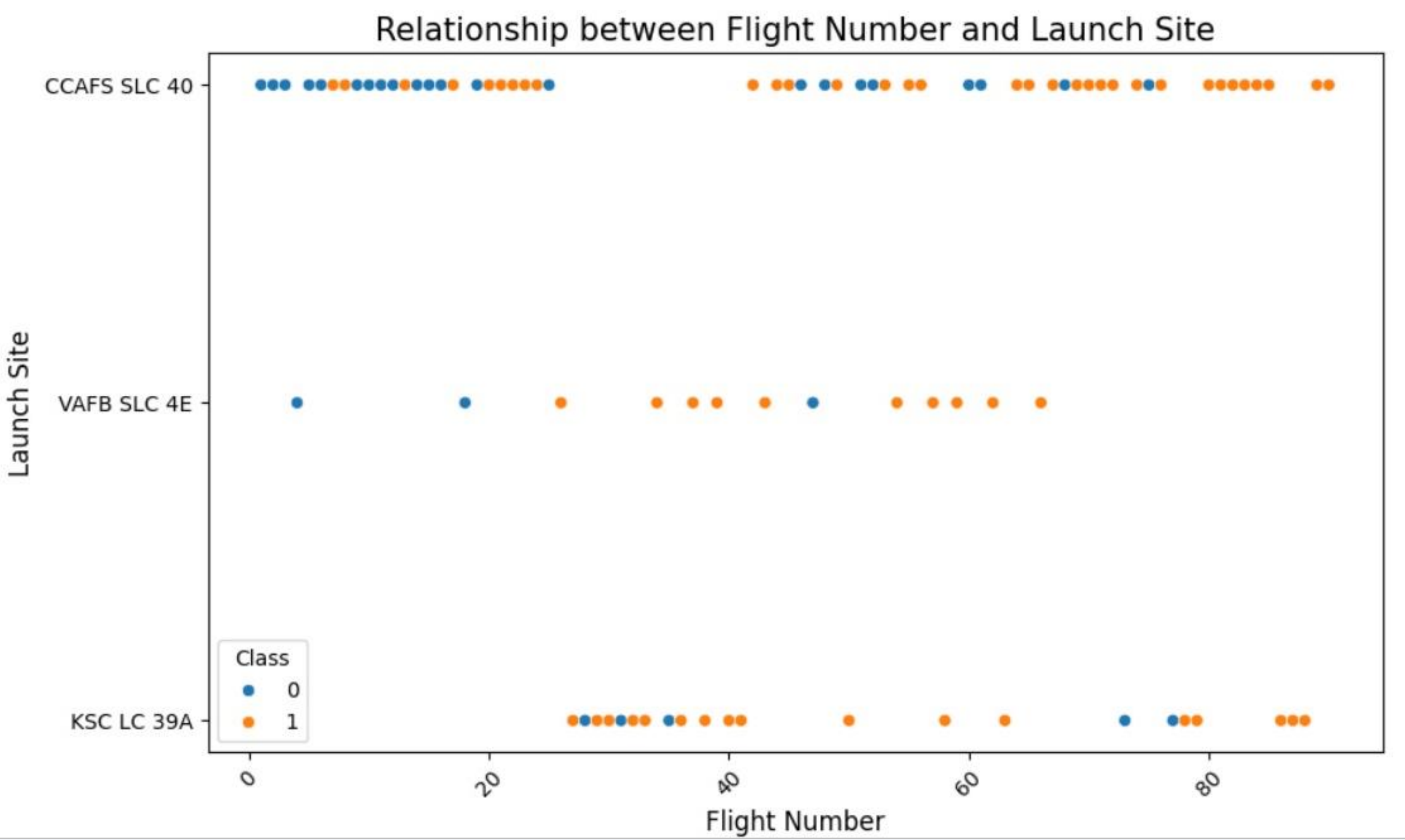
- Opportunities for optimizing launch site selection and cost estimation.

Section 2

Insights drawn from EDA

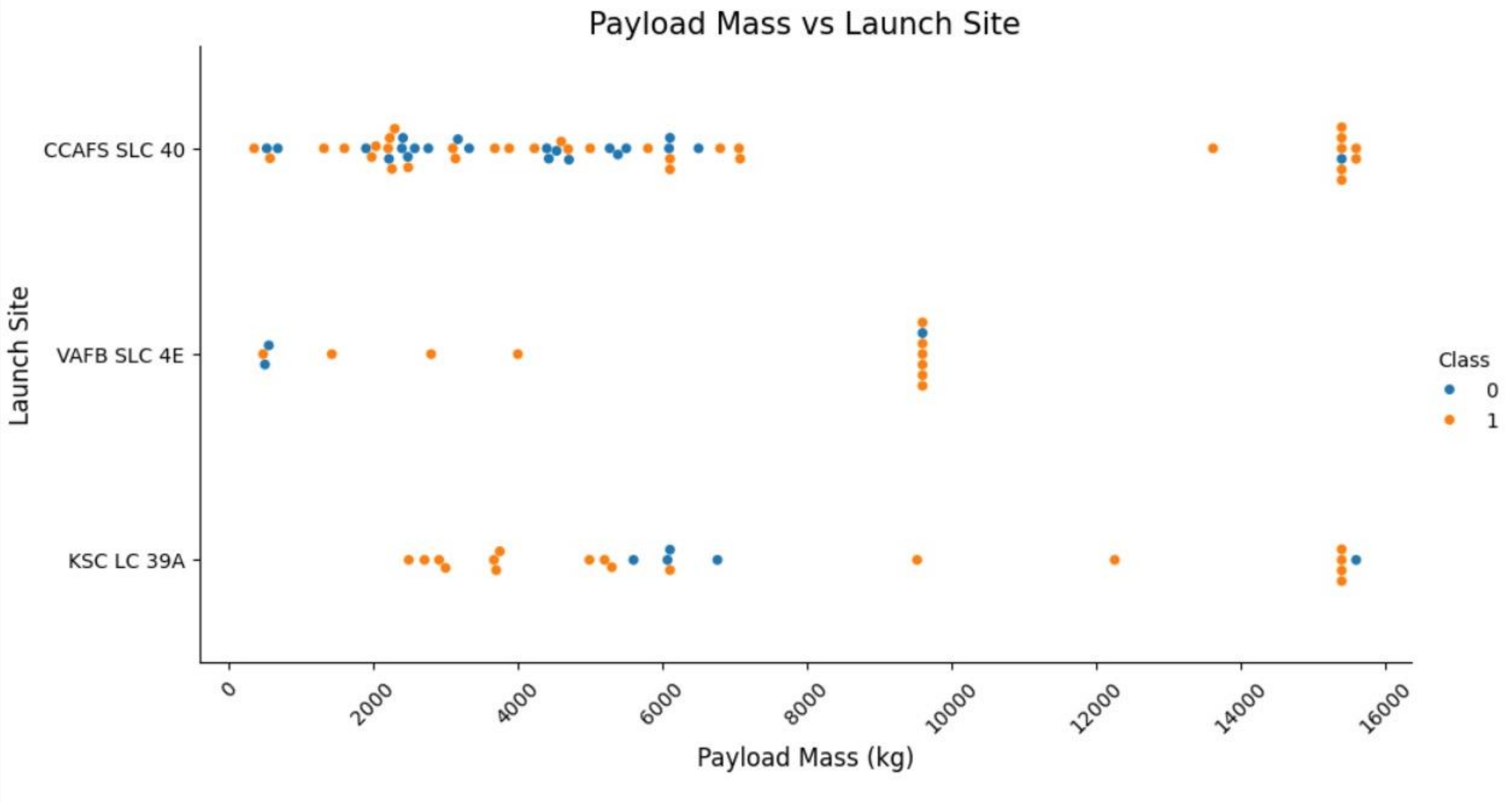
Flight Number vs. Launch Site

- The flight number is generally higher for launches from CCAFS SLC 40 than for launches from VAFB SLC 4E or KSC LC 39A
- as the flight number increases, the launch site is more likely to be CCAFS SLC 40.
- The scatter plot shows that SpaceX has launched Falcon 9 rockets from all three launch sites. However, CCAFS SLC 40 is clearly the preferred launch site for SpaceX



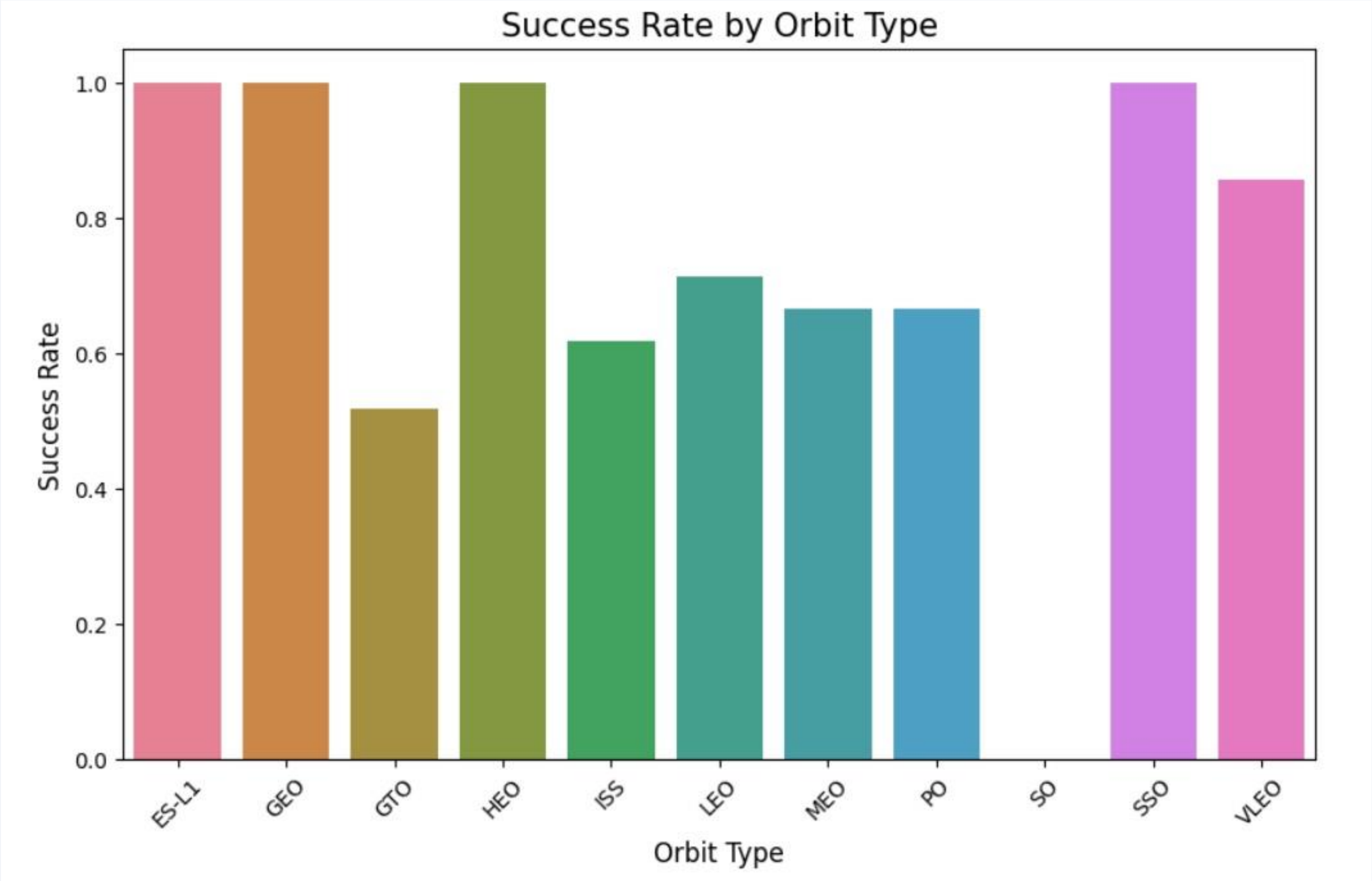
Payload vs. Launch Site

- as the payload mass increases, the launch site is more likely to be CCAFS SLC 40
- The heaviest payload launched was 15,000 kilograms from CCAFS SLC 40
- for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass (greater than 10000).
- CCAFS SLC 40 is the preferred launch site for SpaceX for heavy payloads.



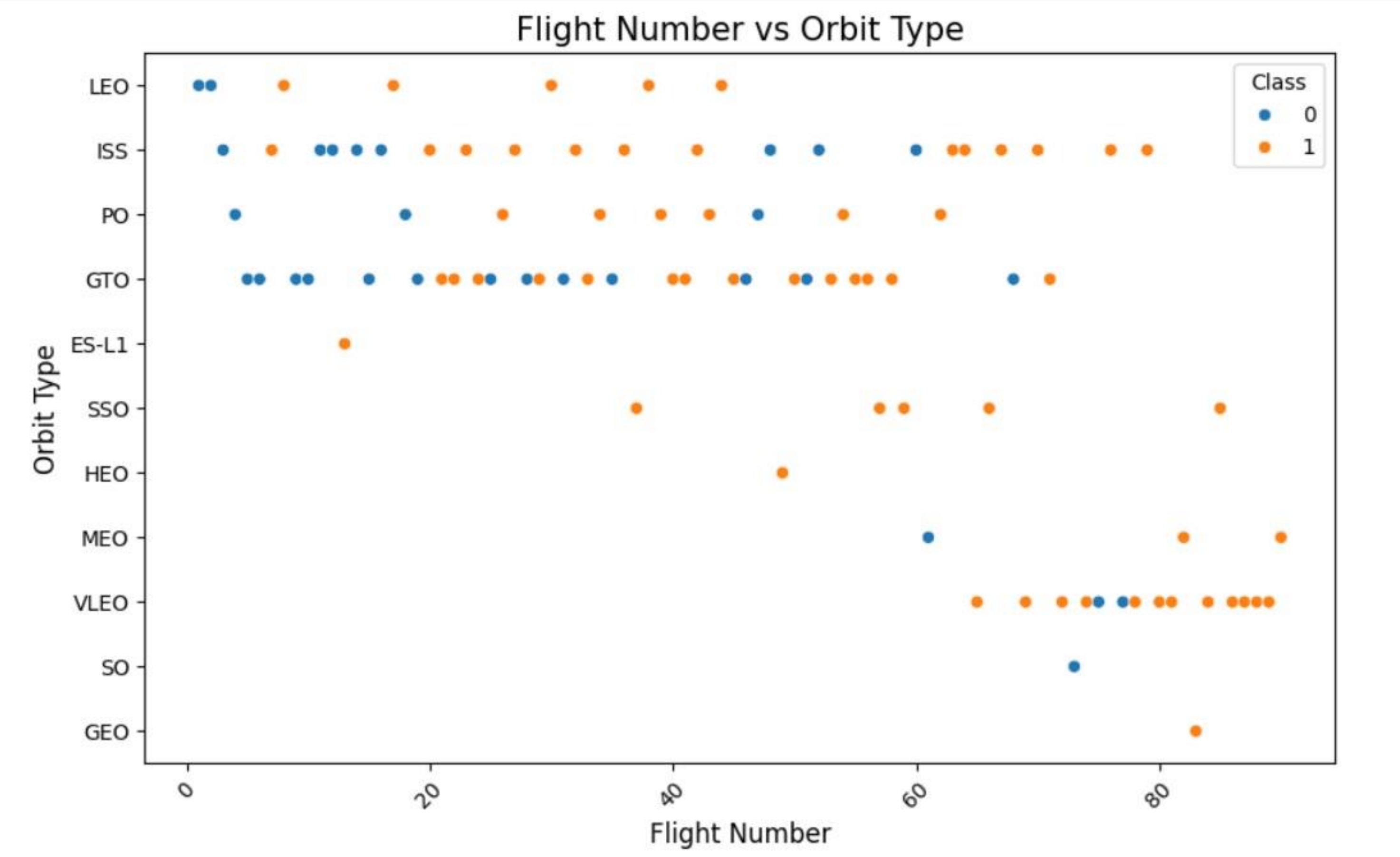
Success Rate vs. Orbit Type

- We can see that SO had 0% success
- GTO and ISS had less success comparing to the others
- ES-L1, GEO, HEO and SSO are the most successful ones



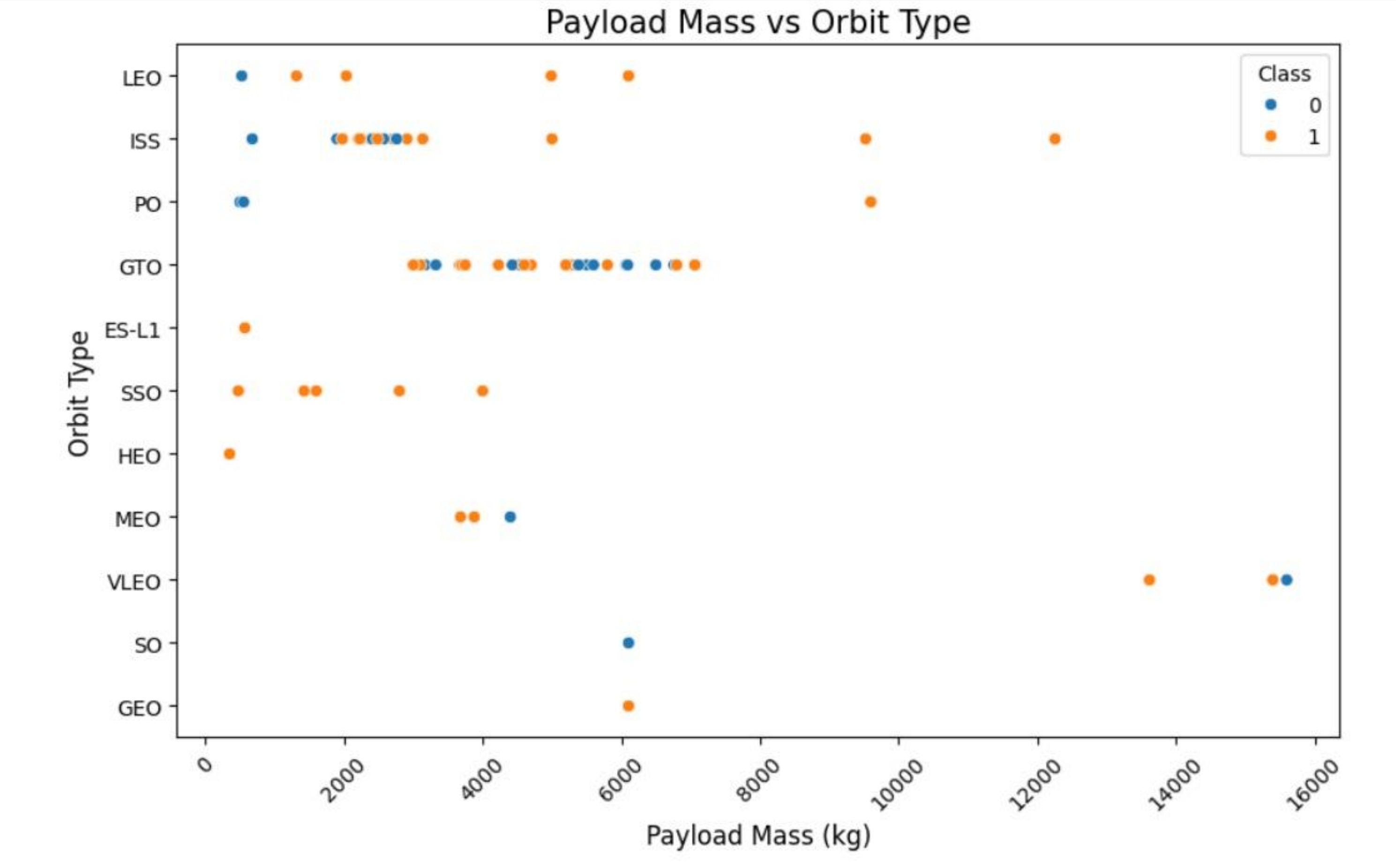
Flight Number vs. Orbit Type

- LEO and GTO are heavily utilized, with numerous flights ranging from 0 to over 80
- SO and GEO have very limited data points, suggesting that missions targeting these orbits are less common or have specific purposes.
- PO have a moderate number of flights, with nine points ranging from 0 to 60. This suggests that missions requiring polar orbits are conducted relatively frequently.



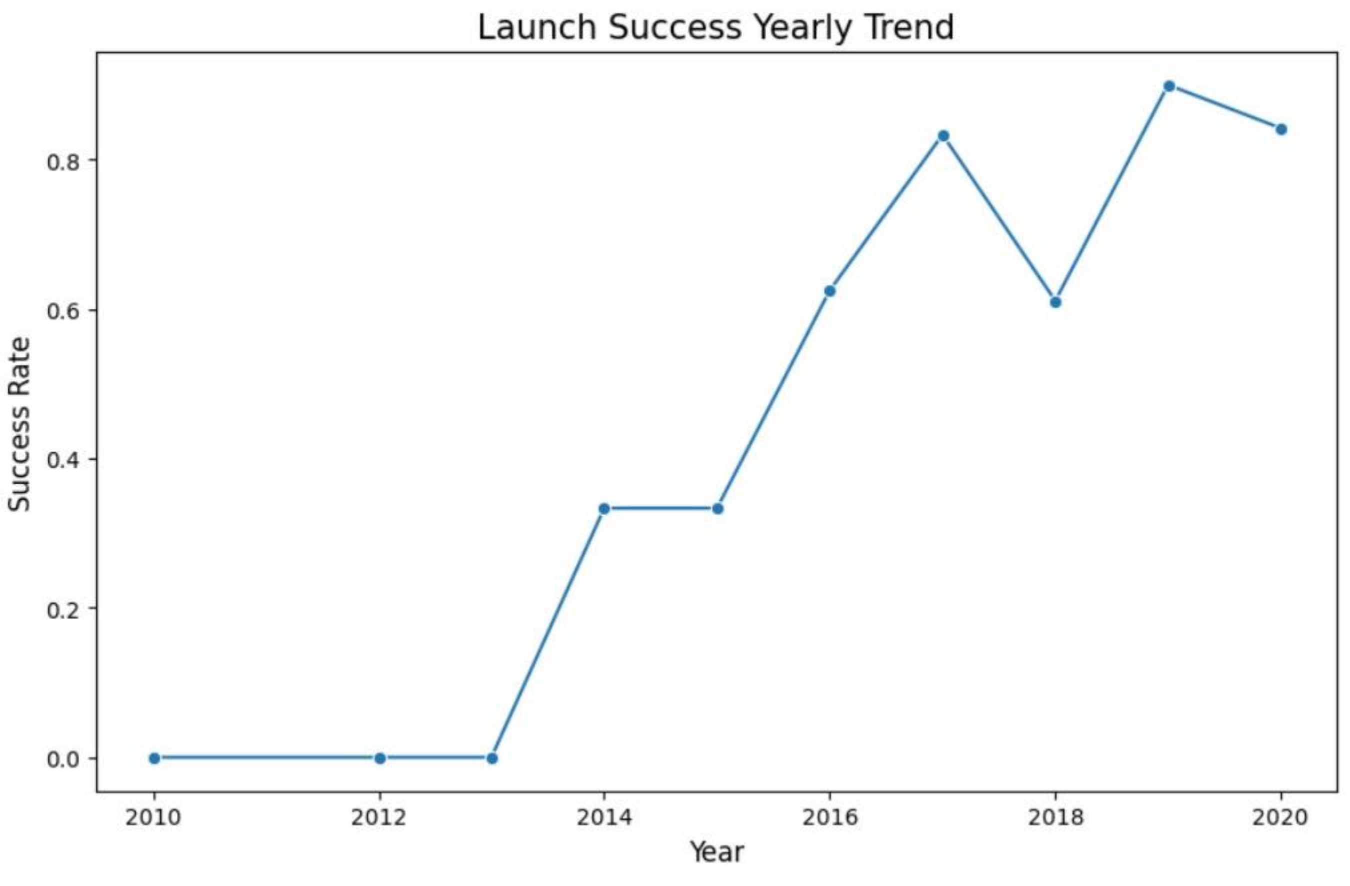
Payload vs. Orbit Type

- GTO show a wide range of payload masses
- LEO have a relatively lower payload capacity compared to other orbits
- ISS have a broad range of payload masses, reaching up to 12000km. This indicates the versatility of missions associated with the ISS.
- ES-L1, HEO, SSO have specific mission requirements, which is reflected in their limited payload range



Launch Success Yearly Trend

- The success rate remains flat until 2013. This indicates a period of experimentation and learning.
- In 2014, there is a noticeable increase in the success rate, reaching 0.3.
- Starting from 2015, there is a substantial upward trend in the success rate. By 2016, it reaches 0.6.
- In 2018, there is a slight decline in the success rate from the high point in 2017, but it still remains relatively high at around 0.6. This could be due to various factors affecting mission outcomes.
- In 2019, there is a notable recovery, with the success rate surpassing 0.8. This indicates that SpaceX made further improvements or adjustments to their operations.
- In 2020, the success rate remains close to 0.8, suggesting that SpaceX has reached a stable state of high mission success.



All Launch Site Names

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

```
%%sql

SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

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

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

- This query retrieve a list of unique launch sites from the SPACEXTABLE database. This eliminates redundant information and provides a clear, manageable list of where each mission was launched from, making it easier for analysis and reporting.

Launch Site Names Begin with 'CCA'

SELECT *: This specifies that we want to retrieve all columns from the SPACEXTABLE.

FROM SPACEXTABLE: This indicates that we're retrieving data from the table named SPACEXTABLE.

WHERE Launch_Site LIKE 'CCA%': This is a condition that filters the results. It states that we're interested in records where the Launch_Site column starts with the string 'CCA'. The % is a wildcard character that matches any number of characters after 'CCA', allowing for flexibility in the matching pattern.

LIMIT 5: This limits the output to a maximum of 5 records. It ensures that only the first five records meeting the condition are returned.

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

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- **SELECT SUM(PAYLOAD_MASS__KG_) AS TotalPayloadMass:** This part of the query selects the sum of the payload mass (in kilograms) from the SPACEXTABLE and assigns it an alias TotalPayloadMass. The SUM() function adds up all the payload masses.
- **FROM SPACEXTABLE:** This specifies that we're retrieving data from the table named SPACEXTABLE.
- **WHERE "Customer" = 'NASA (CRS)':** This is a condition that filters the results. It specifies that we're interested in records where the customer is 'NASA (CRS)', meaning missions associated with NASA's Commercial Resupply Services.

```
%%sql

SELECT SUM(PAYLOAD_MASS__KG_) AS TotalPayloadMass
FROM SPACEXTABLE
WHERE "Customer" = 'NASA (CRS)';
```

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

TotalPayloadMass
45596

Average Payload Mass by F9 v1.1

- **SELECT AVG(PAYLOAD_MASS__KG_) AS AveragePayloadMass:** This part of the query selects the average of the payload masses (in kilograms) from the SPACEXTABLE and assigns it an alias AveragePayloadMass. The AVG() function computes the average of the specified column.
- **FROM SPACEXTABLE:** This specifies that we're retrieving data from the table named SPACEXTABLE.
- **WHERE "Booster_Version" = 'F9 v1.1':** This is a condition that filters the results. It specifies that we're interested in records where the booster version is 'F9 v1.1', meaning missions conducted with this particular booster version.

```
%%sql

SELECT AVG(PAYLOAD_MASS__KG_) AS AveragePayloadMass
FROM SPACEXTABLE
WHERE "Booster_Version" = 'F9 v1.1';
```

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

AveragePayloadMass
2928.4

First Successful Ground Landing Date

- **SELECT MIN(Date) AS FirstSuccessfulGroundPadLanding:** This part of the query selects the minimum (earliest) date from the Date column and assigns it an alias FirstSuccessfulGroundPadLanding. The MIN() function finds the smallest value in a given column.
- **FROM SPACEXTABLE:** This specifies that we're retrieving data from the table named SPACEXTABLE.
- **WHERE "Landing_Outcome" = 'Success (ground pad)':** This is a condition that filters the results. It specifies that we're interested in records where the landing outcome is 'Success (ground pad)', indicating successful landings on a ground pad.

```
SELECT MIN(Date) AS FirstSuccessfulGroundPadLanding
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)';
```

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

FirstSuccessfulGroundPadLanding
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- **SELECT DISTINCT Booster_Version:** This part of the query selects the distinct names of boosters from the SPACEXTABLE.
- FROM SPACEXTABLE: This specifies that we're retrieving data from the table named SPACEXTABLE.
- **WHERE Landing_Outcome = 'Success (drone ship)':** This is a condition that filters the results. It specifies that we're interested in records where the landing outcome is 'Success (drone ship)', indicating successful landings on a drone ship.
- **AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000:** These are additional conditions They ensure that the payload mass is greater than 4000 kilograms but less than 6000 kilograms.

```
SELECT DISTINCT Booster_Version
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)'
      AND PAYLOAD_MASS__KG_ > 4000
      AND PAYLOAD_MASS__KG_ < 6000;
```

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

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

Total Number of Successful and Failure Mission Outcomes

- **SELECT:** This initiates the selection of data.
- **COUNT(CASE WHEN Landing_Outcome = 'Success' THEN 1 END) AS SuccessfulMissions:** This counts the number of records where the landing outcome is 'Success'. The CASE statement evaluates a condition, and if it's true (in this case, if the landing outcome is 'Success'), it returns 1; otherwise, it returns NULL. COUNT then counts the number of non-NULL values. The alias SuccessfulMissions is assigned to this count.
- **COUNT(CASE WHEN Landing_Outcome = 'Failure' THEN 1 END) AS FailedMissions:** Similarly, this counts the number of records where the landing outcome is 'Failure'. It uses the same logic as the previous count. The alias FailedMissions is assigned to this count.
- **FROM SPACEXTABLE:** This specifies that we're retrieving data from the table named SPACEXTABLE.

```
SELECT
    COUNT(CASE WHEN Landing_Outcome = 'Success' THEN 1 END) AS SuccessfulMissions,
    COUNT(CASE WHEN Landing_Outcome = 'Failure' THEN 1 END) AS FailedMissions
FROM SPACEXTABLE;
```

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

SuccessfulMissions	FailedMissions
38	3

Boosters Carried Maximum Payload

- **SELECT DISTINCT Booster_Version:** This part of the query selects the distinct names of boosters from the SPACEXTABLE.
- **FROM SPACEXTABLE:** This specifies that we're retrieving data from the table named SPACEXTABLE.
- **WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE):** This is a condition that filters the results. It compares the payload mass of each record with the maximum payload mass in the entire table. This subquery (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE) finds the highest payload mass in the table.

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

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

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

- **SELECT strftime('%m', Date) AS Month:** This part of the query uses the strftime function to extract the month from the Date column. It's aliased as Month.
- **CASE WHEN Landing_Outcome = 'Failure (drone ship)' THEN 'Failure' ELSE 'Success' END AS Landing_Outcome:** This case statement assigns a label 'Failure' if the landing outcome is 'Failure (drone ship)' and 'Success' otherwise. It's aliased as Landing_Outcome.
- **Booster_Version:** This selects the names of the booster versions.
- **Launch_Site:** This selects the names of the launch sites.
- **FROM SPACEXTABLE:** This specifies that we're retrieving data from the table named SPACEXTABLE.
- **WHERE substr(Date,0,5) = '2015' AND Landing_Outcome = 'Failure (drone ship)':** These are conditions that filter the results. The first condition checks if the year in the Date column is 2015, and the second condition specifies that we're interested in records with a landing outcome of 'Failure (drone ship)'.

```
SELECT
    strftime('%m', Date) AS Month,
    CASE
        WHEN Landing_Outcome = 'Failure (drone ship)' THEN 'Failure'
        ELSE 'Success'
    END AS Landing_Outcome,
    Booster_Version,
    Launch_Site
FROM SPACEXTABLE
WHERE substr(Date,0,5) = '2015'
      AND Landing_Outcome = 'Failure (drone ship)';
```

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

Month	Landing_Outcome	Booster_Version	Launch_Site
10	Failure	F9 v1.1 B1012	CCAFS LC-40
04	Failure	F9 v1.1 B1015	CCAFS LC-40

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

- **SELECT Landing_Outcome:** This part of the query selects the distinct landing outcomes from the SPACEXTABLE.
- **COUNT(*) AS Outcome_Count:** This counts the number of records for each landing outcome and assigns it an alias Outcome_Count.
- **RANK() OVER (ORDER BY COUNT(*) DESC) AS Rank:** This assigns a rank to each landing outcome based on the count in descending order. The RANK() function is used for this purpose.
- **FROM SPACEXTABLE:** This specifies that we're retrieving data from the table named SPACEXTABLE.
- **WHERE Date BETWEEN '2010-06-04' AND '2017-03-20':** This is a condition that filters the results. It specifies that we're interested in records where the date falls within the specified range.
- **GROUP BY Landing_Outcome: This** groups the results by landing outcome, so that we get a count for each unique landing outcome.
- **ORDER BY Outcome_Count DESC:** This orders the results by the outcome count in descending order.

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

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

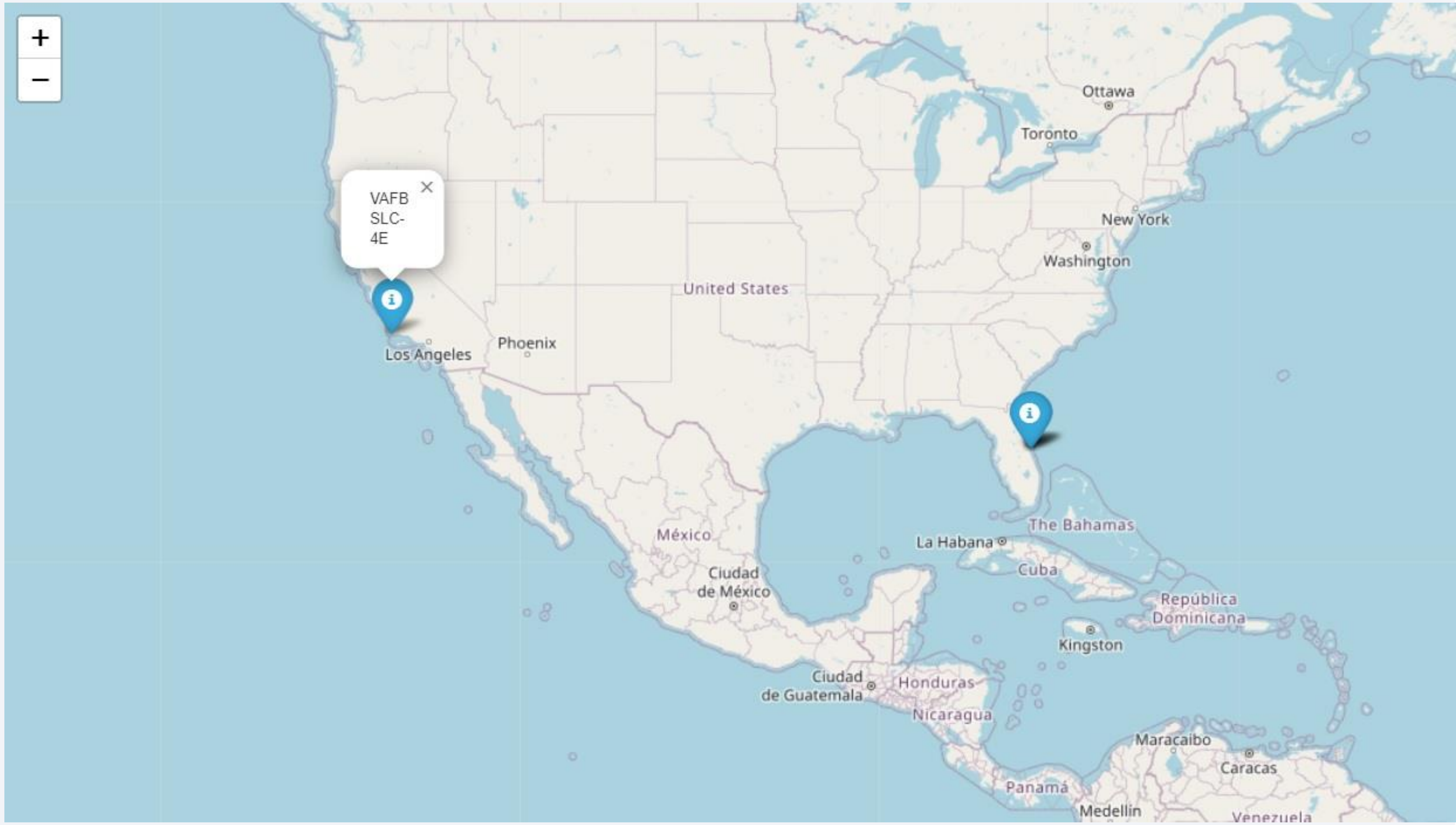
Landing_Outcome	Outcome_Count	Rank
No attempt	10	1
Success (ground pad)	5	2
Success (drone ship)	5	2
Failure (drone ship)	5	2
Controlled (ocean)	3	5
Uncontrolled (ocean)	2	6
Precluded (drone ship)	1	7
Failure (parachute)	1	7

Section 3

Launch Sites Proximities Analysis

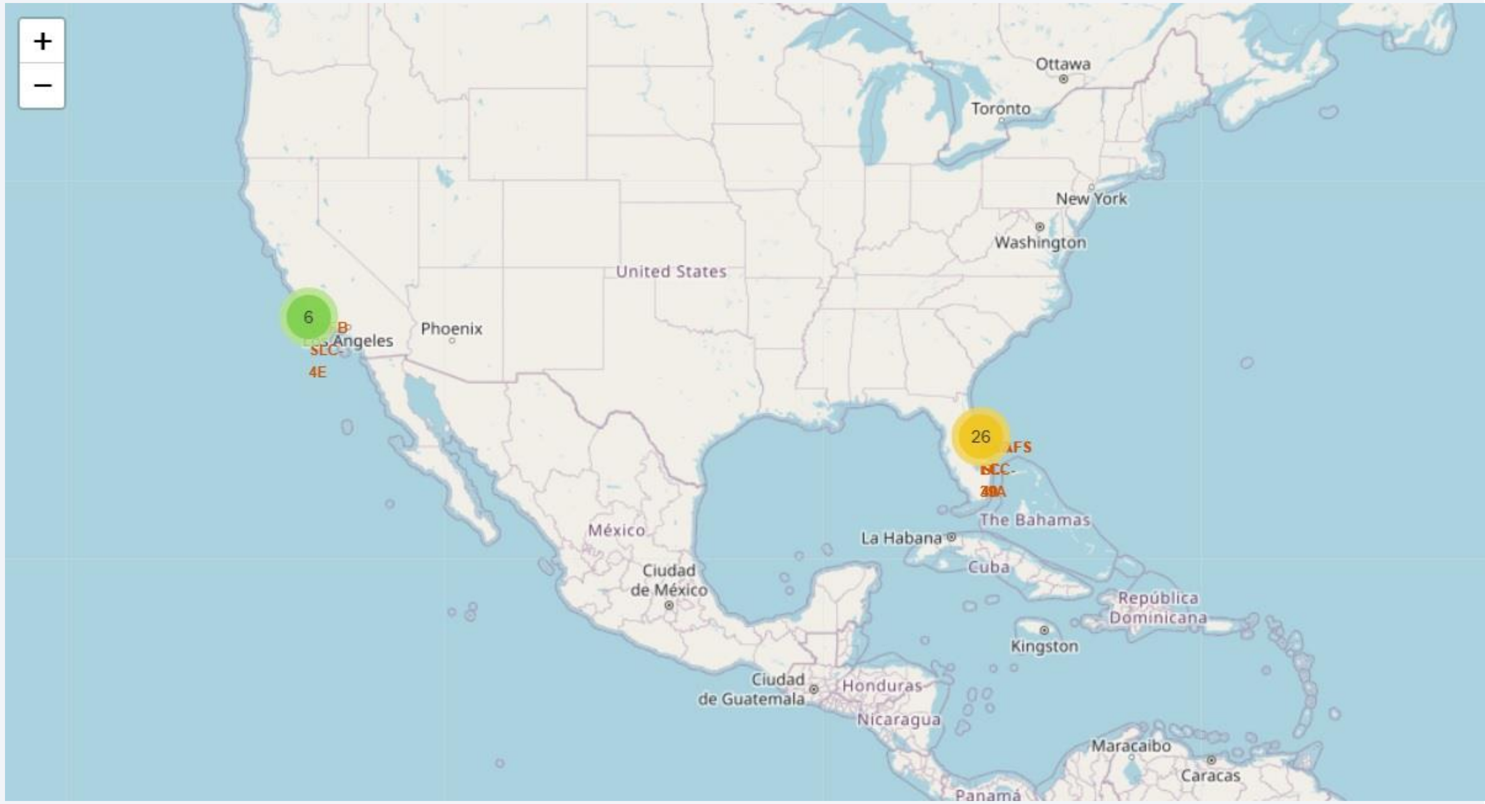
<Location of the Launch Sites>

- On the left side of the map, close to Los Angeles, it is located the VAFB SLC-4E
- The other tree launch sites are close to each other in the marker of the right side of the map. We can’t see it like this because it is to distant but when you zoom-in you can see the other three very clearly.



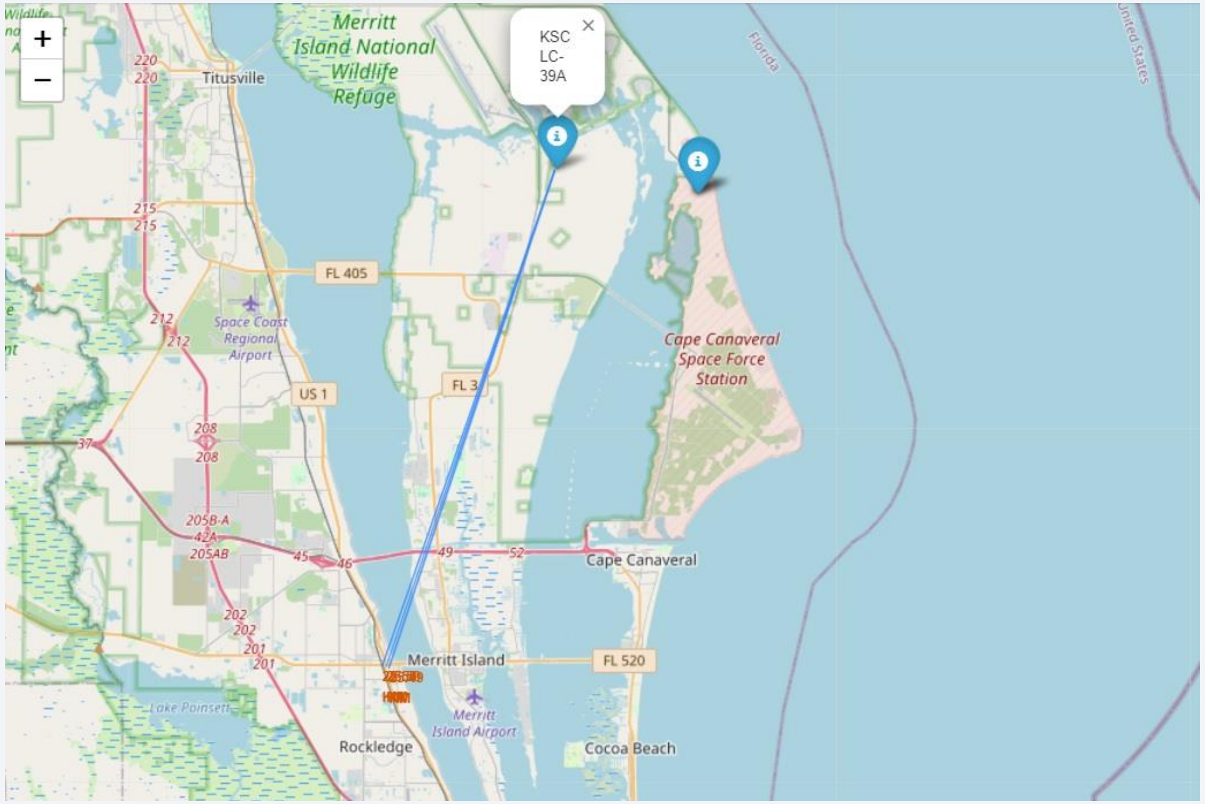
<Success and failed launches>

- This is a map that displays at green the successful launches and at red the failed ones.
- At the right side of the map you can see a big green circle which shows a lot of succefull launches. And at the right side you can see yellow because it is a mix between green and red colors. Which shows successfull and failed landings in that area.



<Launch site and nearby cities>

- I've chosen the KSC LC-39A launch site to a proximate city called Cocoa.
- We can see in the map the line that goes from the Launch site to the city of Cocoa, to a railway and a Highway.
- In the next slide we will see it more clearly.

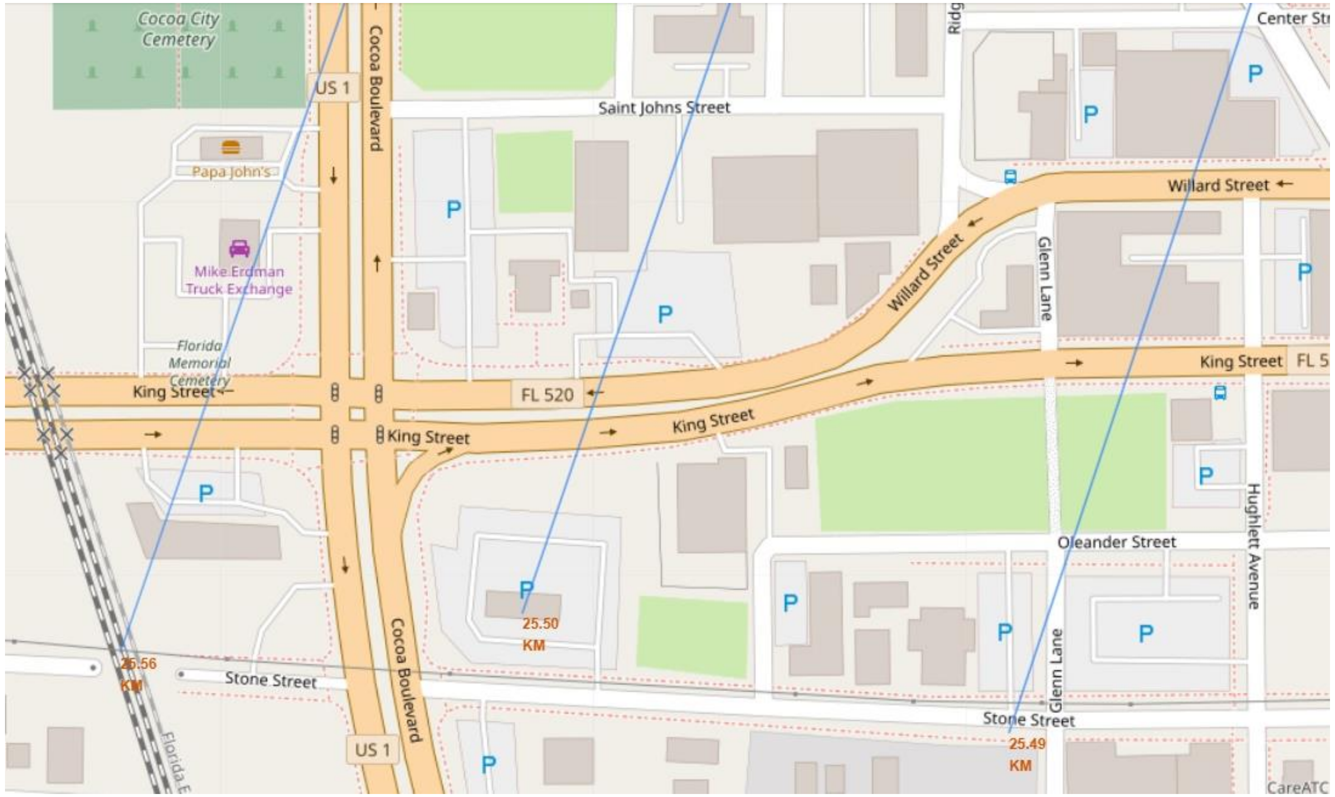


<Launch site and nearby cities>

In this map we can see the exact distance from the launch site. It is aproximately 25km.

Knowing the distance between a launch site and nearby cities is important for several reasons:

- Safety and Security
- Regulatory Compliance
- Emergency Response
- Environmental Impact
- Infrastructure and Logistics
- Public Perception and Relations
- Legal and Liability
- Economic Considerations

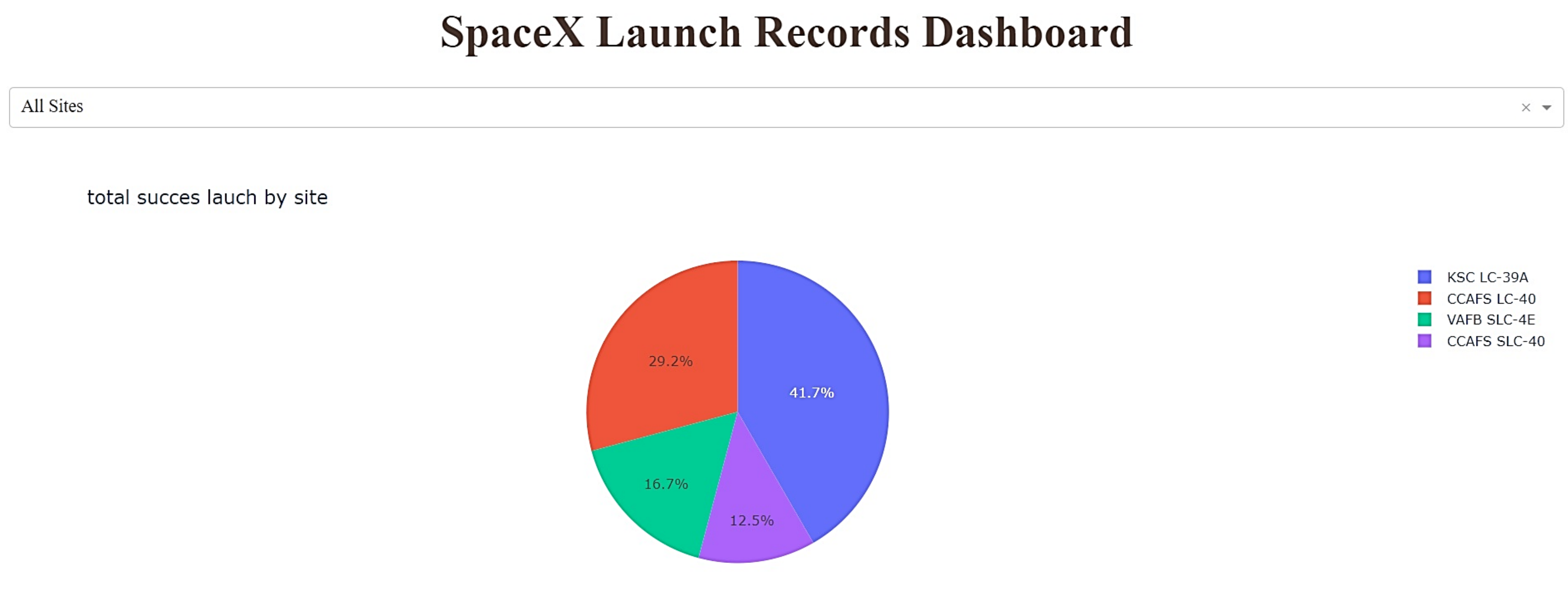


Section 4

Build a Dashboard with Plotly Dash

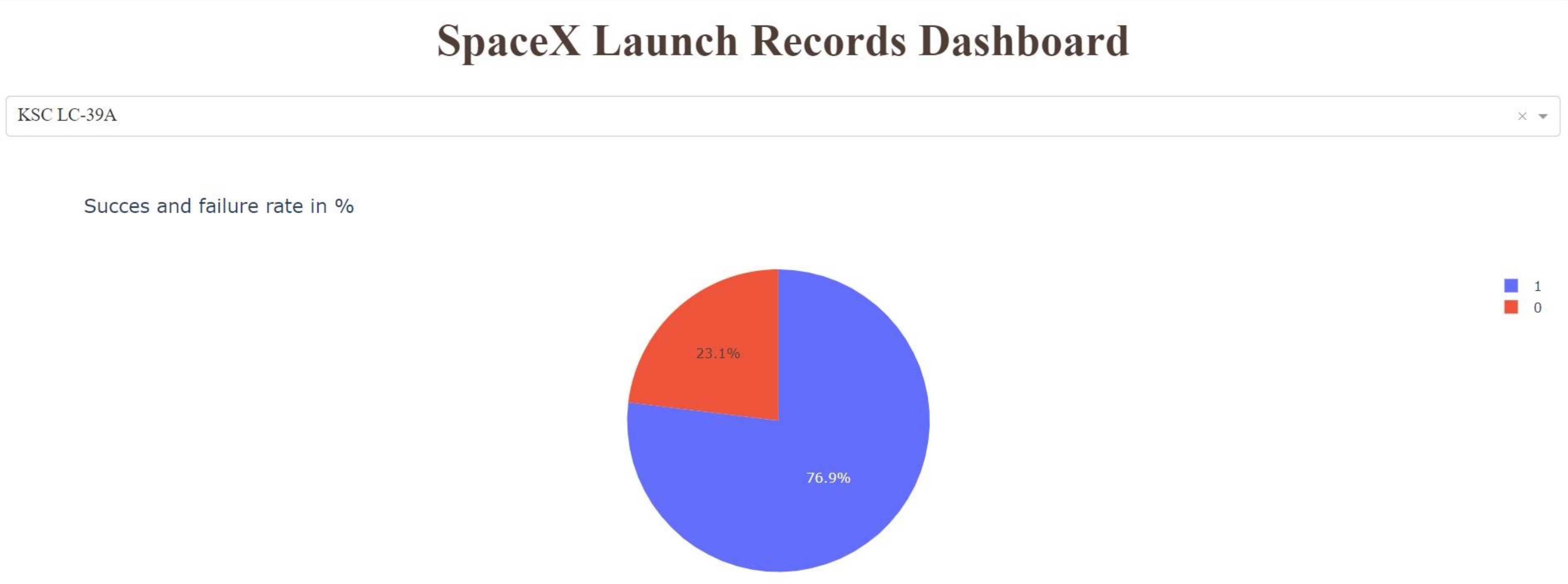
<Pie chart for all sites>

- We can see in this pie chart, that KSC LC-39A is the most successed Launch Site occupying almost half of the pie chart
- CCAFS SLC-40 is the least successed Launch site, with only 12,5% of **success**.



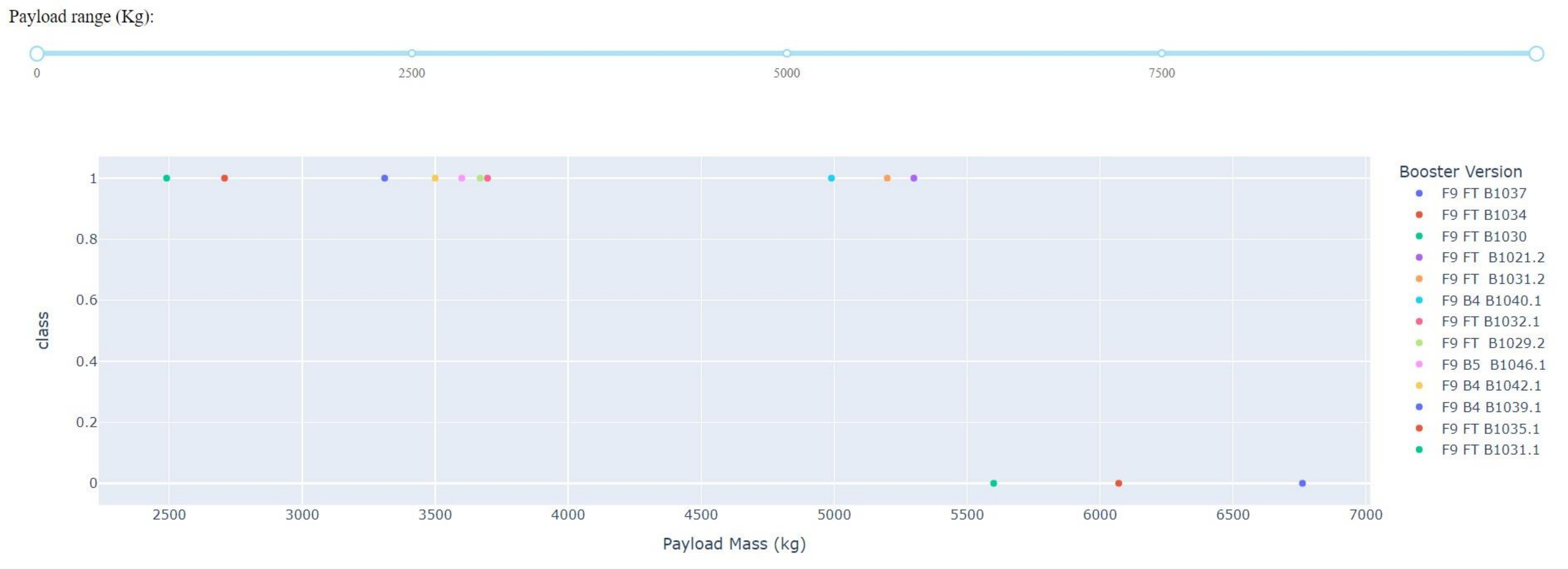
<Pie chart launch site with highest success>

- In this pie-chart we can see the launch site with the highest success ratio: **KSC LC-39A**.
- We can see in blue the **success ratio**: 76,9% and in red the **failure rate**: 23,1%.



<Dashboard Screenshot 3>

- The booster version with the largest success rate is the Falcon 9 Full Thrust (F9 FT).
- The payload range with the largest success rate is 2500-5000 kg.
- The booster version with the lowest success rate is the Falcon 9 Block 4 (F9 B4).
- The payload range with the lowest success rate is 0-2500 kg.

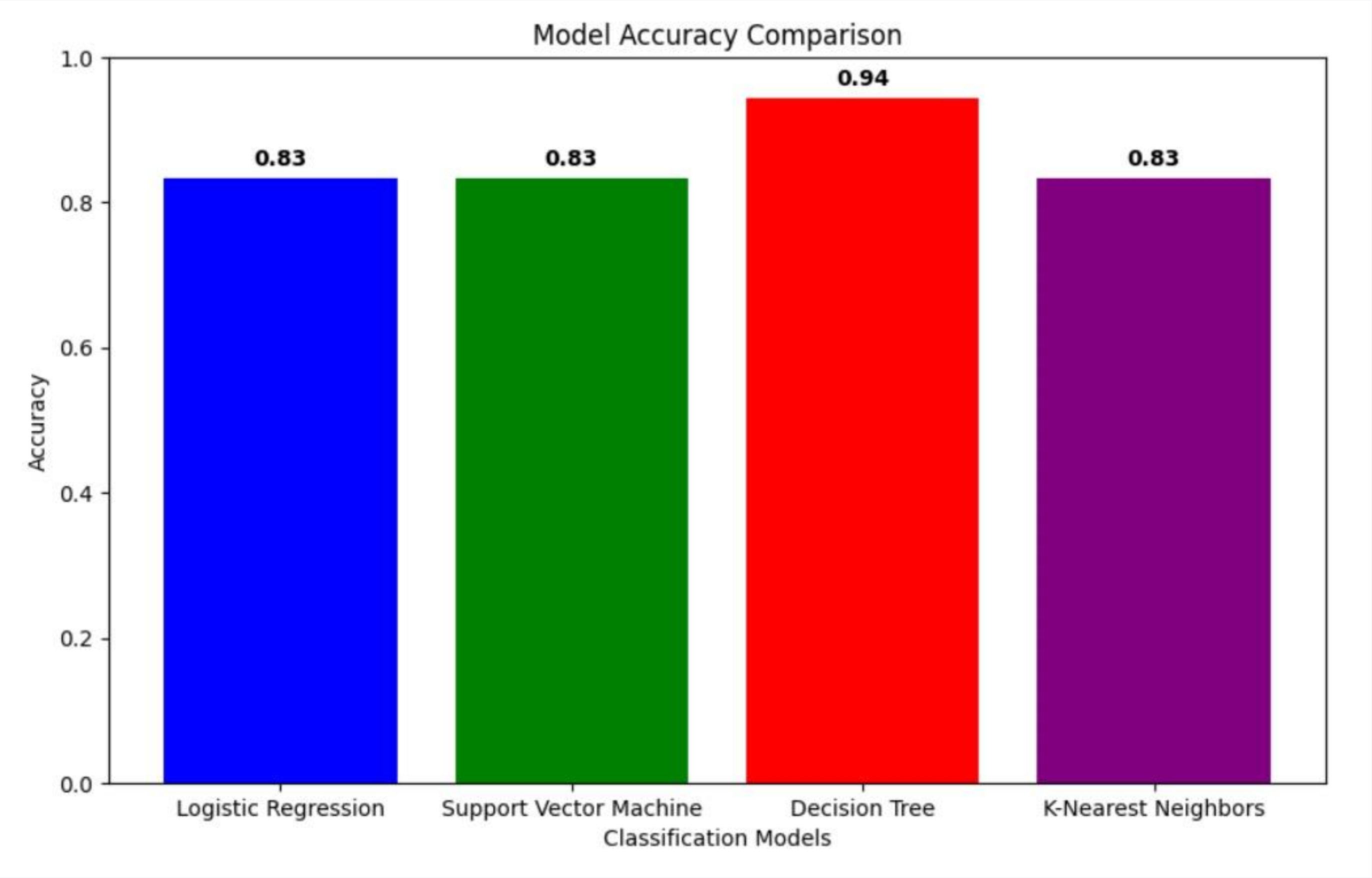


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The model that has the highest classification accuracy is Decision Tree, with an accuracy of 94%.



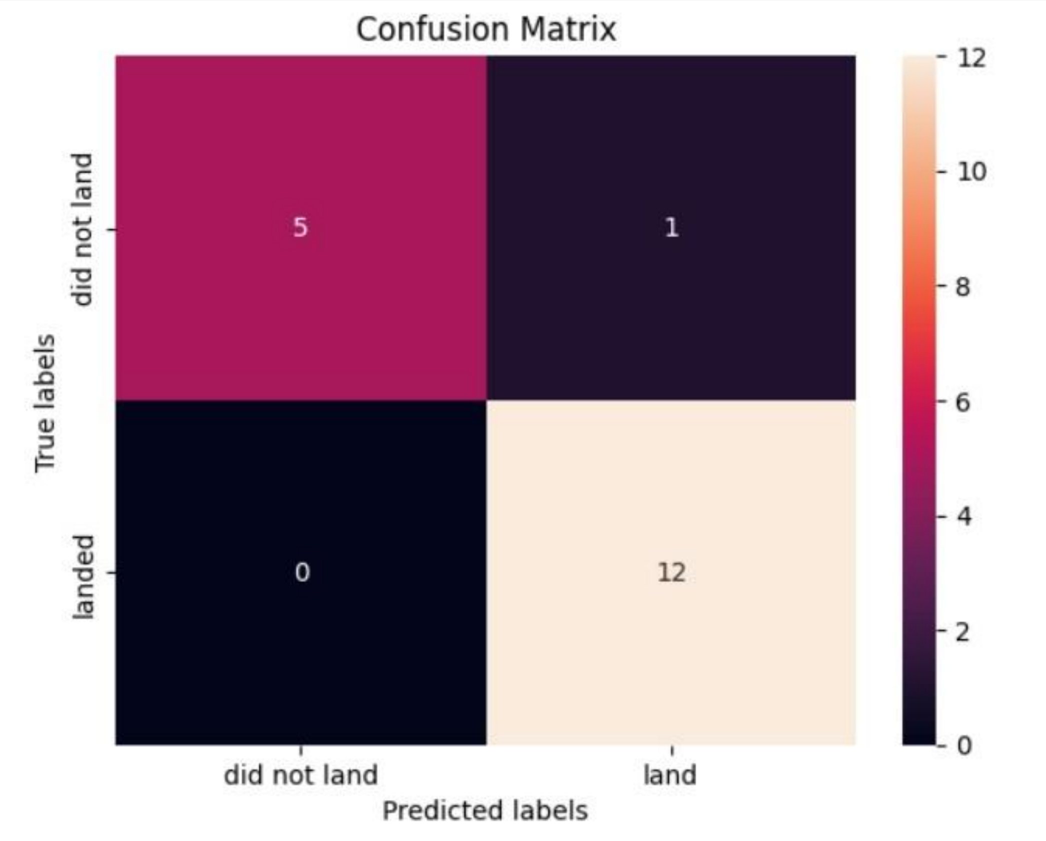
Confusion Matrix

The best-performing method is **Decision Tree** with an accuracy of 0.94

- True Positives (TP): 5
- False Positives (FP): 1
- True Negatives (TN): 12
- False Negatives (FN): 0

Accuracy:

(TP + TN) / (TP + TN + FP + FN) = (5 + 12) / (5 + 12 + 1 + 0) = 17 / 18 ≈ 0.9444



Conclusions

Key Findings:

- **Launch Site Selection:** CCAFS SLC 40 emerged as the preferred site, especially for heavy payloads.
- **Payload Influence:** Payload mass significantly impacts launch site selection and success rates.
- **Orbit Success Rates:** Orbits ES-L1, GEO, HEO, and SSO demonstrate high success rates.
- **Predictive Modeling:** Decision Tree model achieved 94% accuracy, aiding in cost estimation.

Implications:

- **Cost Efficiency:** Informed site selection and payload planning can lead to cost savings, maximizing mission efficiency.
- **Operational Optimization:** Understanding success factors allows for targeted improvements in operations and equipment.
- **Risk Mitigation:** Data-driven decisions reduce the likelihood of launch failures, enhancing overall mission success.

Future Research:

- **Advanced Predictive Models:** Explore machine learning algorithms for even more accurate predictions.
- **Real-time Analysis:** Implement real-time data processing for dynamic mission adjustments.
- **Environmental Factors:** Investigate the impact of weather conditions and geographic features on launch success.
- **Market Expansion:** Analyze data to identify opportunities for expanding launch services to new orbits and clients.

Thank you!