IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Anuj D>
<08-08-2025>

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context
- Problems you want to find answers

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

- Describe how data sets were collected.

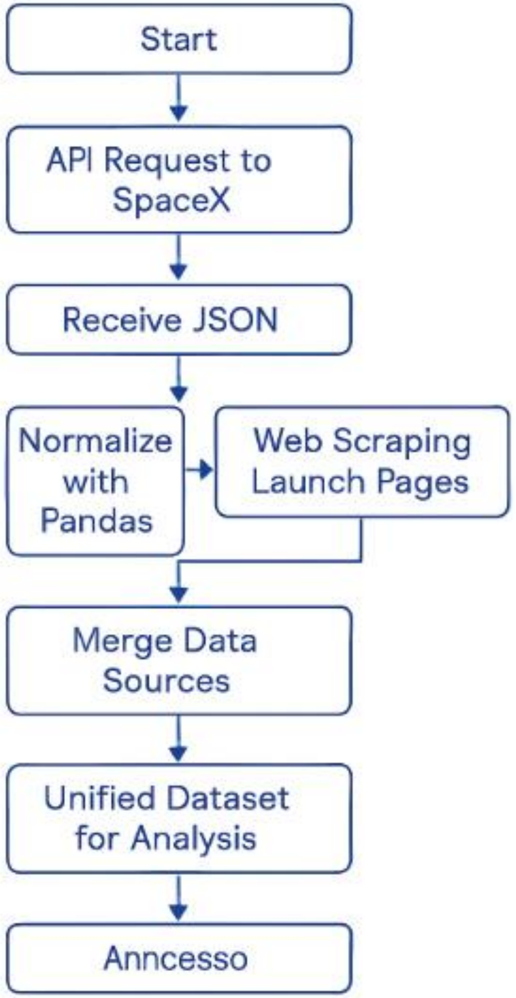
  Primary Sources:
  - **SpaceX Public REST API**
    → Used to gather structured launch data including rocket type, payload, site, and outcome.
  - **Web Scraping (Wikipedia & SpaceX sites)**
    → Used to obtain additional contextual data like site coordinates, landing types, booster details.

  Tools & Technologies:
  - Python
  - requests for API calls
  - BeautifulSoup for web scraping
  - pandas for JSON normalization and dataframes

Describe how data sets were collected.(contd.)

Collection Process Summary (Key Phrases):

- Initiate API request to https://api.spacexdata.com/v4/launches
- Parse JSON response using json.loads()
- Normalize data using pandas.json_normalize()
- Scrape Wikipedia launch history page using requests.get()
- Parse with BeautifulSoup → Extract relevant tables



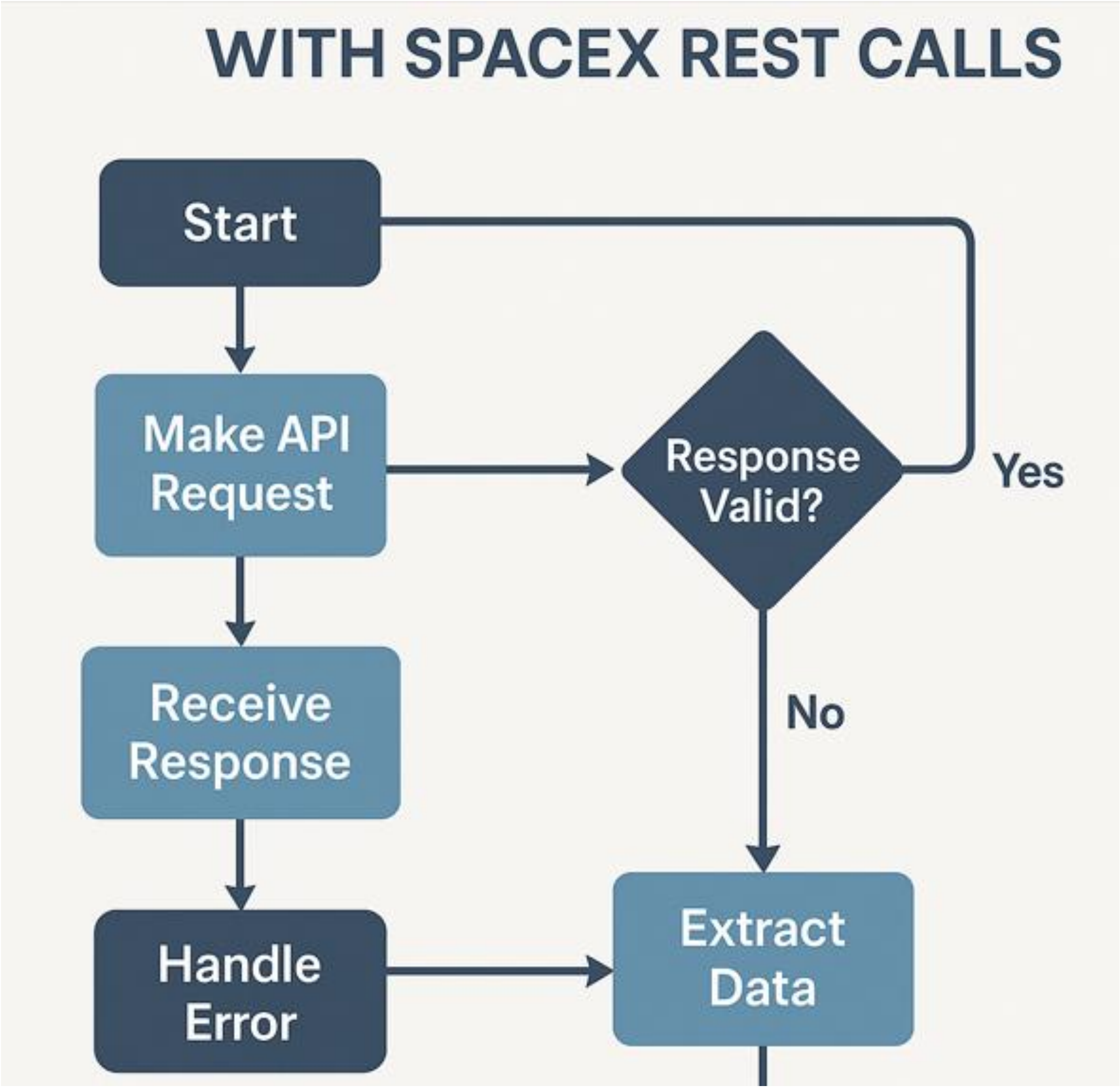
- You need to present your data collection process use key phrases and flowcharts

- [Start] → [API Request to SpaceX] → [Receive JSON] → [Normalize with Pandas]
- → [Web Scraping Launch Pages] → [Extract Tables with BeautifulSoup]
- → [Merge Data Sources] → [Unified Dataset for Analysis]
-

Data Collection – SpaceX API

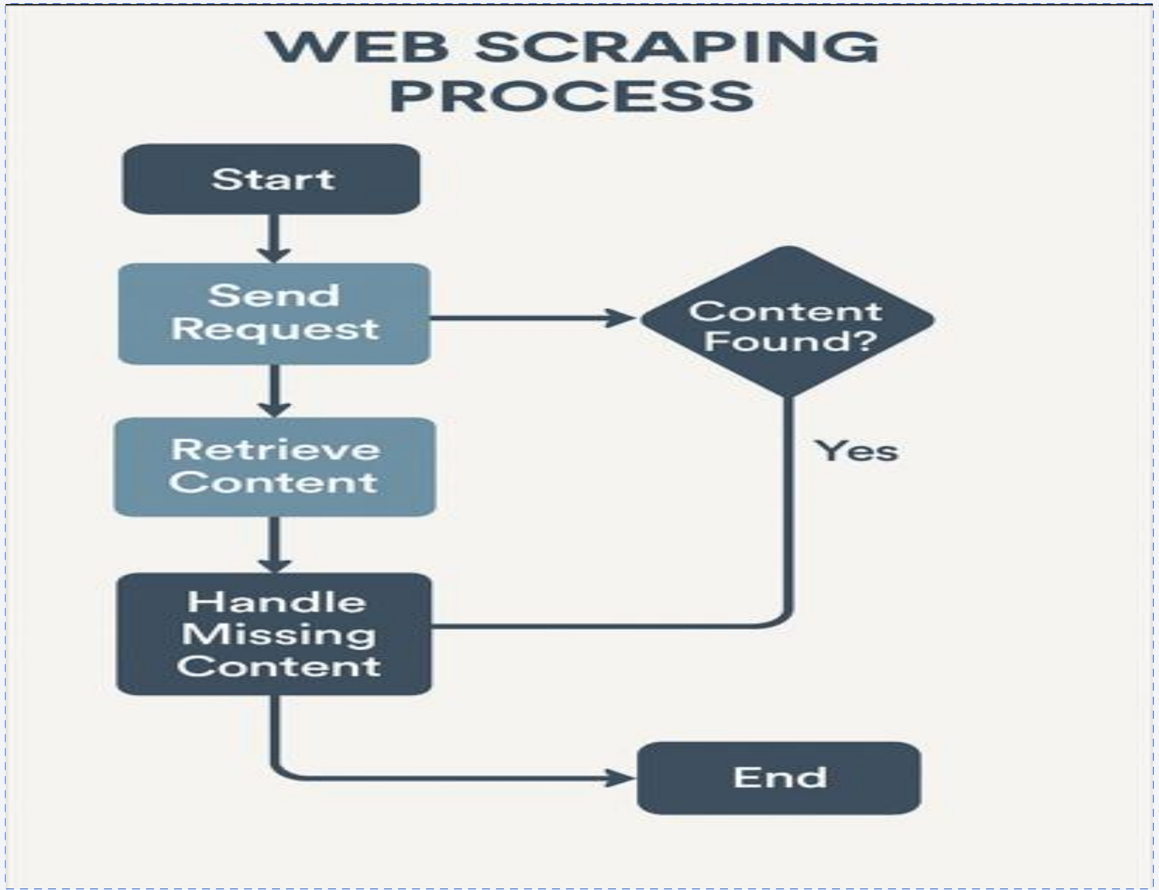
- Present your data collection with SpaceX REST calls using key phrases and flowcharts.
- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose

https://github.com/Anuj-86/NB_repo/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts
- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose
- https://github.com/Anuj-86/NB_repo/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

- Describe how data were processed

1. Data Collection

- **Source**: SpaceX REST API / Web Scraping
- **Tools**: requests, BeautifulSoup, pandas, or similar
- **Action**: Retrieve structured (API) or unstructured (HTML) data

2. Data Cleaning

- **Remove Nulls/Missing Values**
  Fill, drop, or flag rows with incomplete data.
- **Standardize Formats**
  Convert dates, numbers, text (e.g., UTC to ISO 8601).
- **Fix Inconsistencies**
  Normalize categorical values (e.g., "Falcon 9" vs "falcon9").

3.Data Validation

- **Check Data Types**
  Ensure fields are correct (e.g., float, int, str, datetime).
- **Ensure Uniqueness**
  Remove duplicate records using pandas.drop_duplicates().
- **Validate Ranges**
  Ensure values fall within expected limits (e.g., payload mass > 0).

4. Data Structuring

- **Reshape**
  Pivot or melt data into useful wide/long formats.
- **Merge/Join**
  Combine datasets (e.g., launches + payloads + rockets).
- **Reindex/Sort**
  Prepare data for efficient analysis.

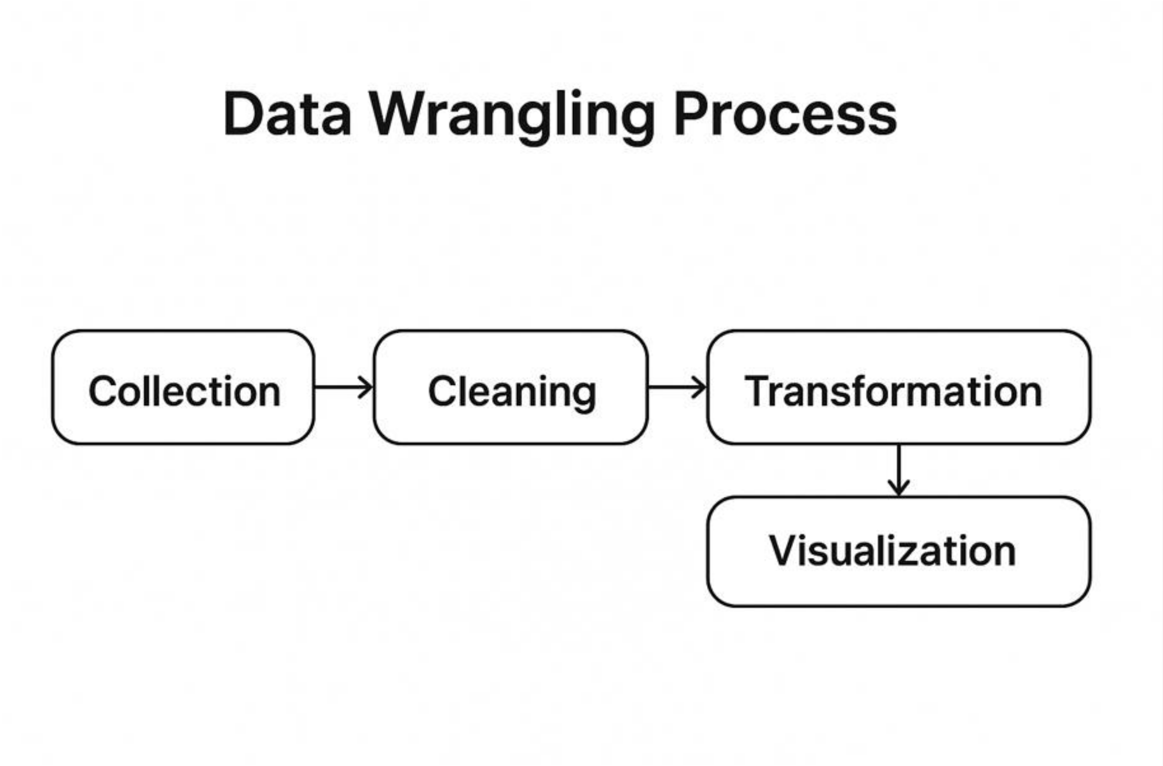
5. Feature Engineering

- **Create New Fields**
  - Launch Success Rate
  - Time Since Launch
  - Rocket Reuse Count
- **Categorize Values**
  Binning continuous variables (e.g., mass ranges, launch year groups).

6. Exporting Data

- **Save as**:
  - CSV
  - Excel
  - JSON
  - SQL
- **Purpose**: To use in analysis, dashboards, or machine learning models.

You need to present your data wrangling process using key phrases and flowcharts



Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose

https://github.com/Anuj-86/NB_repo/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts

1. **Bar Chart**
   - **Used For**:
     - Launch success counts by rocket type
     - Number of launches per launch site
   - **Why**:
     Bar charts clearly compare discrete categories. They help assess which rockets or sites are used most frequently.
2. **Line Chart**
   - **Used For**:
   - Year-over-year launch frequency
   - Rocket reuse trend over time
   - **Why**:
     Line charts highlight trends and fluctuations over time, revealing growth patterns or seasonal activity.

3. Histogram

- **Used For**:
- Distribution of payload masses
- Distribution of rocket flight durations
- **Why**:
  Histograms show how data points are distributed and help identify skewness or normality.

4. Scatter Plot

- **Used For**:
- Payload mass vs launch success
- Launch date vs number of cores reused

Why:
Scatter plots reveal correlations or clusters, ideal for investigating relationships between numeric variables.

5. Pie Chart / Donut Chart

- **Used For**:
- Proportion of mission outcomes (Success vs Failure)
- **Why**:
  Pie charts provide a quick overview of categorical distribution but are best used sparingly.

Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

https://github.com/Anuj-86/NB_repo/blob/main/jupyter-labs-eda-dataviz-v2.ipynb

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
- **Data Exploration Queries**
  - SELECT * FROM launches LIMIT 10;
    → View sample records from the dataset.
  - SELECT DISTINCT rocket_name FROM launches;
    → Get unique rocket types used.
  - SELECT COUNT(*) FROM launches;
    → Count total number of launches.
- **Descriptive Statistics**
  - SELECT AVG(payload_mass_kg), MIN(payload_mass_kg), MAX(payload_mass_kg) FROM payloads;
    → Get average, min, and max payload mass.
  - SELECT launch_site, COUNT(*) AS total_launches FROM launches GROUP BY launch_site;
    → Number of launches per site.

- **Time-Based Analysis**
  - SELECT EXTRACT(YEAR FROM launch_date) AS year, COUNT(*) FROM launches GROUP BY year ORDER BY year;
    → Launch frequency by year.
  - SELECT launch_date FROM launches ORDER BY launch_date DESC LIMIT 1;
    → Find the most recent launch.
- **Success/Failure Filtering**
  - SELECT * FROM launches WHERE launch_success = false;
    → List failed launches.
  - SELECT rocket_name, COUNT(*) FROM launches WHERE launch_success = true GROUP BY rocket_name;
    → Successful launches by rocket type.
- **Joins**
  - SELECT l.mission_name, r.rocket_type FROM launches l JOIN rockets r ON l.rocket_id = r.rocket_id;
    → Combine launch info with rocket type details.
  - SELECT p.payload_mass_kg, l.launch_success FROM payloads p JOIN launches l ON p.launch_id = l.id;
    → Compare payloads with launch success.

- **Aggregated Insights**
  - SELECT rocket_type, AVG(payload_mass_kg) FROM rockets r JOIN payloads p ON r.rocket_id = p.rocket_id GROUP BY rocket_type;
    → Average payload mass per rocket type.
  - SELECT launch_site, SUM(launch_success::int) FROM launches GROUP BY launch_site;
    → Total successful launches per site (using casting for boolean to integer).

Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

https://github.com/Anuj-86/NB_repo/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map

1. Markers :- folium.Marker()
2. Circle Markers :- folium.CircleMarker()
3. Colored Circles :- folium.Circle()
4. PolyLines :- folium. PolyLine()
5. Feature Groups / Layers :- folium. FeatureGroup() , LayerControl()

Explain why you added those objects

What: folium.Marker()

Used For:

- Showing exact launch pad locations
- Popup info like site name, mission count

Why:
Markers provide a precise visual cue for each launch site.

Circle Markers

- **What**: folium.CircleMarker()
- **Used For**:
  - Representing launch sites with visual size scaling
  - Radius proportional to number of launches

Why:
Helps visually compare site activity (larger circle = more launches)

Colored Circles

- **What**: folium.Circle()
- **Used For**:
  - Highlighting impact zones or restricted areas
  - Using color to indicate status (e.g., green = active, red = inactive)
- **Why**:
  Adds context and category-based visual filtering.

PolyLines

- **What**: folium.PolyLine()
- **Used For**:
  - Illustrating potential rocket flight paths
  - Connecting launch site to ocean landing zones or drone ships

Why:
Useful to show the direction or reach of launches

Feature Groups / Layers

- **What**: folium.FeatureGroup(), LayerControl()
- **Used For**:
    - Grouping markers (e.g., by rocket type or outcome)
    - Adding togglable layers for better interactivity
- **Why**:
  Makes the map cleaner and interactive for exploratory analysis.

Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

https://github.com/Anuj-86/NB_repo/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
- Explain why you added those plots and interactions
- <u>**Plots and Graphs**</u>
    - **Bar Chart**:
        - Shows launch count per site or rocket type
        - Useful for comparing categorical performance
    - **Pie Chart**:
        - Displays mission outcome proportions (Success vs Failure)
        - Provides a quick overview of launch success rate
    - **Line Chart**:
        - Visualizes launches over time
        - Reveals trends and seasonality
    - **Scatter Plot**:
        - Plots payload mass vs launch outcome
        - Helps identify patterns or anomalies

Histogram:
Shows distribution of payload mass
Useful for spotting data skewness or outliers

Interactive Map:
Geolocates launch sites with markers and info
Enhances spatial analysis

Interactive Features

- **Dropdown Menus**:
  - Filter data by site, rocket type, or year
  - Updates all linked visualizations
- **Sliders**:
  - Adjust payload range or time window
  - Useful for dynamic filtering
- **Checkboxes / Toggles**:
  - Show or hide specific categories (e.g., failed launches)
  - Enables focused analysis
- **Popups on Map**:
  - Display site name, total launches, last launch date
  - Activated on click
- **Hover Tooltips**:
  - Reveal extra data on graph hover
  - Enhances data interpretation without clutter

Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

https://github.com/Anuj-86/NB_repo

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
- You need present your model development process using key phrases and flowchart

Model Building
•Target: launch_success (Yes/No)
•Models tried: Logistic Regression, Random Forest, SVM, KNN
Model Evaluation
•Train/Test split (80/20)
•Cross-validation used
•Metrics: Accuracy, Precision, Recall, F1-score, ROC-AUC
Model Improvement
•One-hot encoding for categorical features
•Feature scaling for numeric data
•Hyperparameter tuning with GridSearchCV / RandomizedSearchCV
Best Model Selection
•Random Forest performed best on test data
•Highest F1-score and ROC-AUC
•Final model saved for predictions

Predictive Analysis – Classification Model

Model Building

- Target variabe: launch, success
- Models tried: Logistic Regress, Random Forest, M SVM, KNN

Model Evaluation

- Train/Test spilt (6.0/20
- Cross-validation used
- Metrics, Accuracy, Precisel, F1-score, ROC-AUC

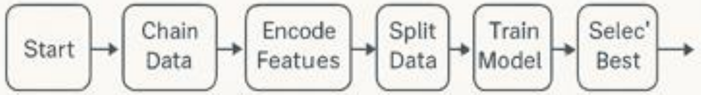
Model Improvement

- One-hot encoding for: categorsical features
- Feature scaling for numeric data
- Hyperparameter tuiting with GridSearchCV

Best Model Selection

- Random Forest performed best on test data
- Highest F1-score and ROC-AUC

Model Workflow



Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

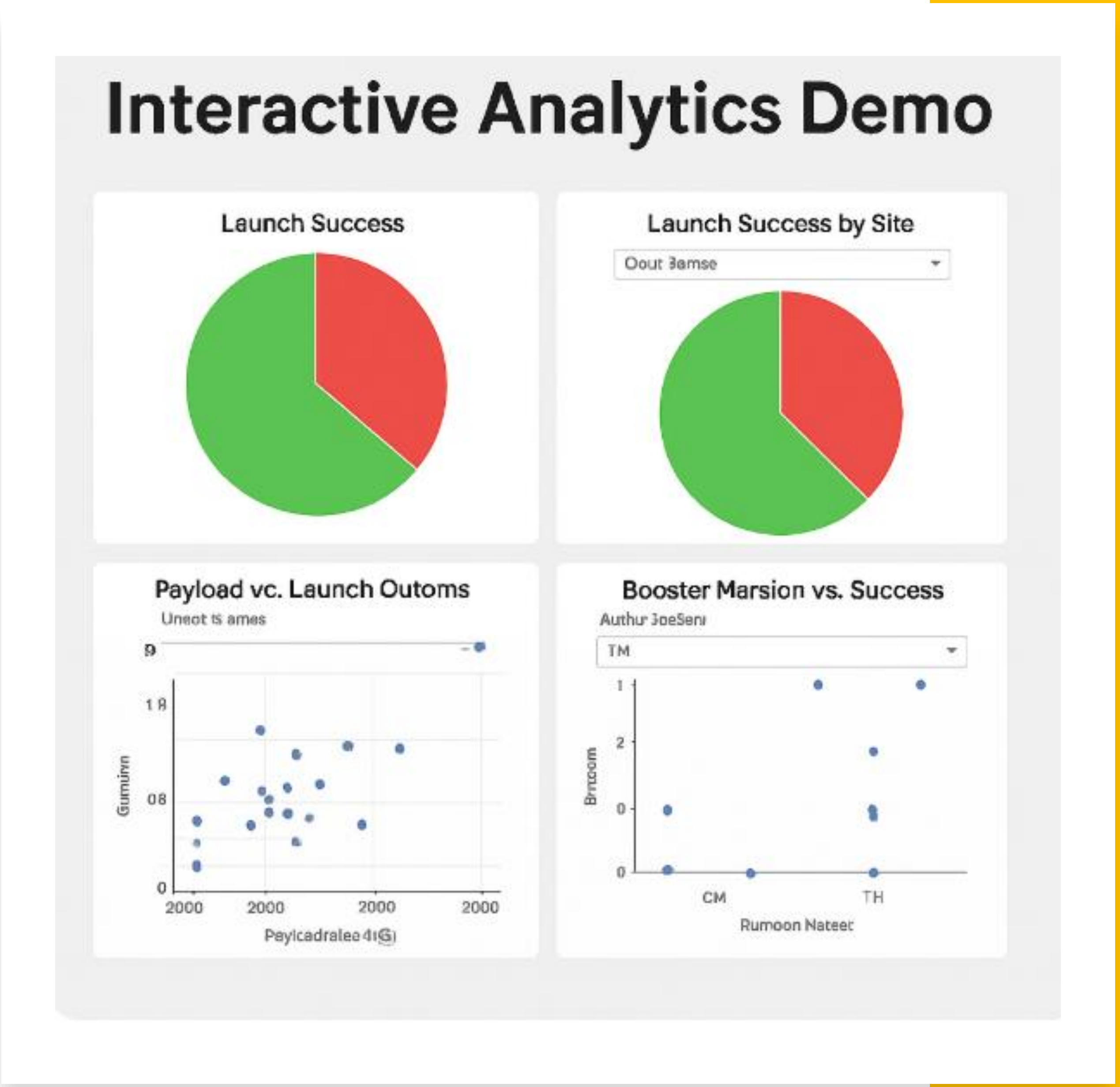
https://github.com/Anuj-86/NB_repo/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

- Exploratory data analysis results
  - Success rates improved with higher flight numbers.
  - CCAFS SLC 40 handled heavier payloads.
  - LEO and GTO orbits showed highest success rates.
  - Ground pad landings succeeded more after 2015.
  - Newer booster versions performed better.

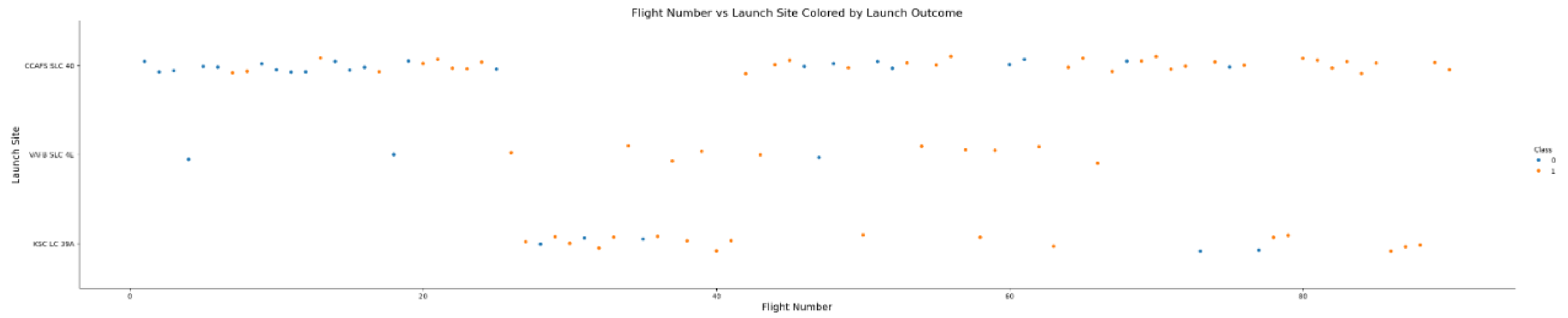
- Interactive analytics demo in screenshots

- Predictive analysis results
  - Data standardization improved model performance.
  - Hyperparameter tuning optimized all models.
  - All models performed reasonably well.
  - Logistic Regression model showed the lowest misclassification in the confusion matrix.



Section 2

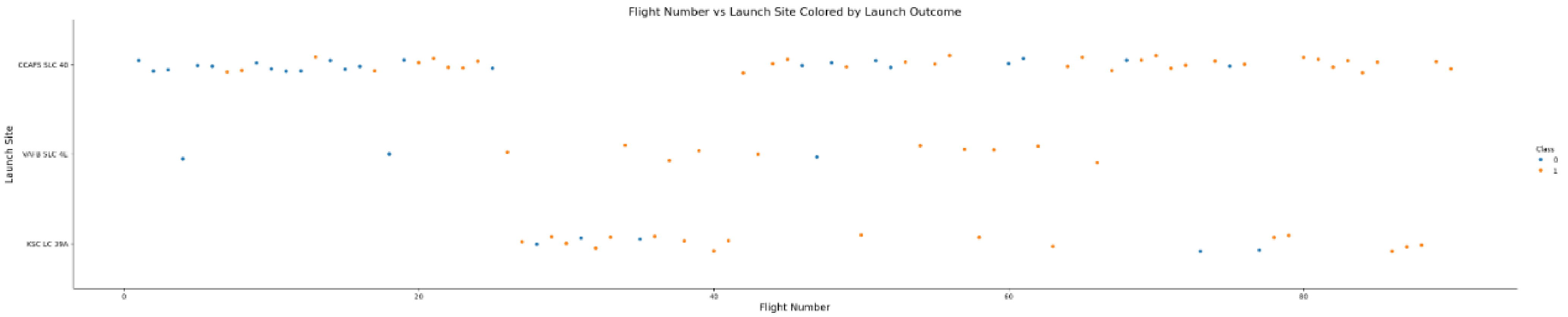
Insights drawn from EDA



Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site

Show the screenshot of the scatter plot with explanations



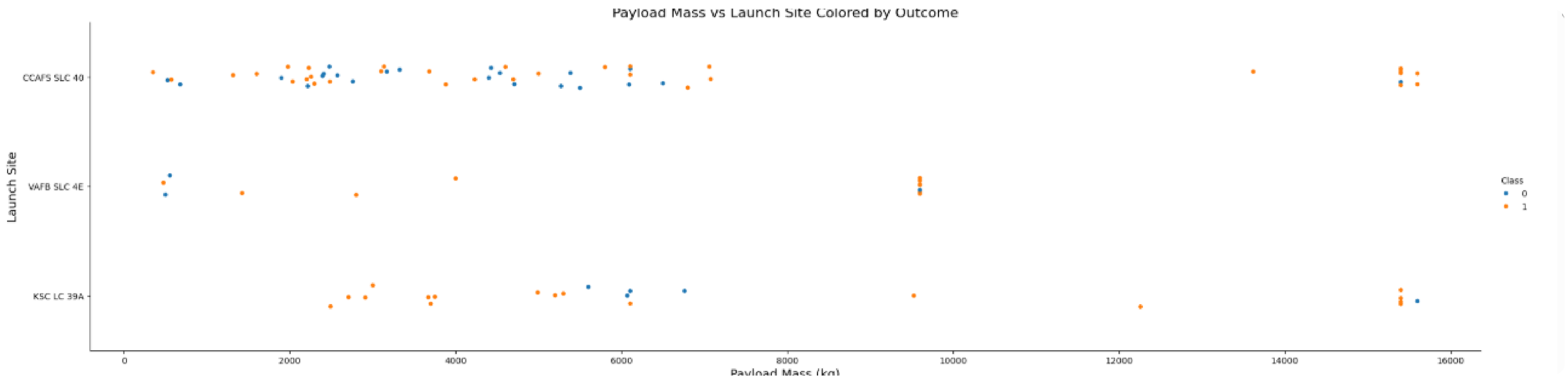
Early Flights Had More Failures Lower flight numbers show more failed launches, indicating early testing and development.

Success Rate Improved Over Time Higher flight numbers are mostly successful, showing growth in reliability and experience.

Some Launch Sites Are More Successful Sites like KSC LC-39A and CCAFS LC-40 show more successful launches compared to others.

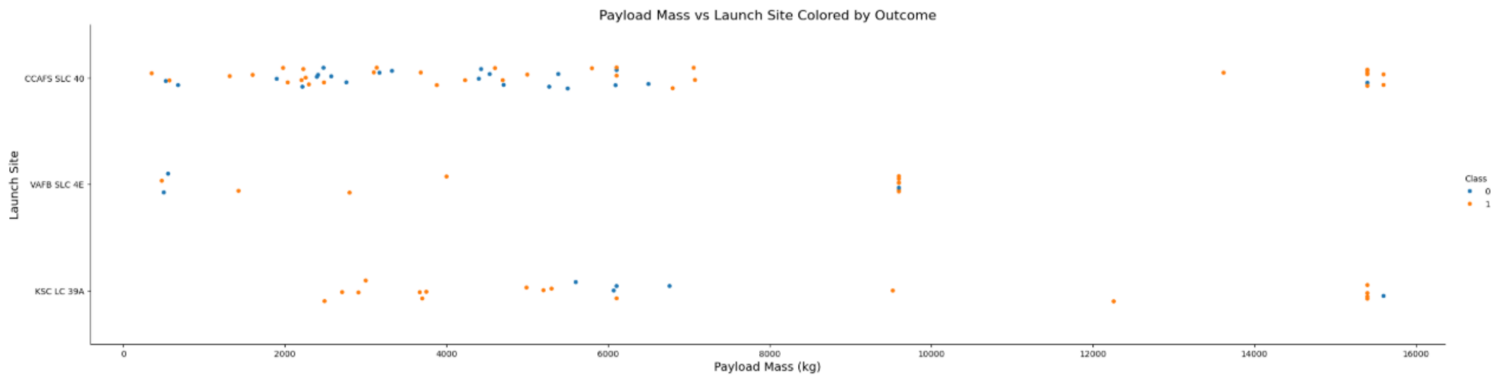
Certain Sites Were Used in Specific Periods Grouped flight numbers at some sites suggest operational shifts or upgrades over time.

Higher Activity at Key Sites Frequent launches from major sites highlight operational hubs for SpaceX missions.



- Show a scatter plot of Payload vs. Launch Site

Payload vs. Launch Site



Payload Mass vs. Launch Site – Key Observations Different Sites Handle Different Payload Ranges Some launch sites consistently handle heavier or lighter payloads.

High Payloads Linked to Specific Sites Sites like KSC LC-39A are often used for heavy payloads, possibly due to better infrastructure.

Lighter Payloads at Multiple Sites Sites like CCAFS LC-40 and VAFB SLC-4E frequently launch medium to small payloads.

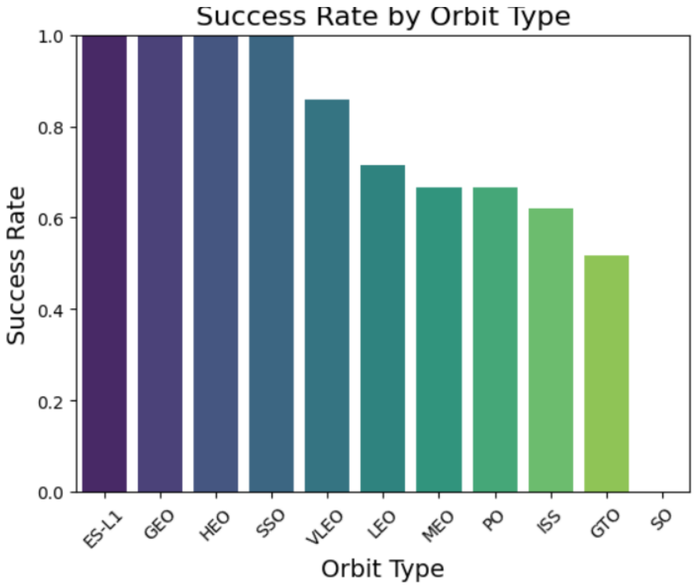
Launch Outcome Patterns Successful and failed launches are scattered across all payload ranges, but failures tend to cluster in earlier payload missions.

Payload Specialization is Visible The plot reveals how SpaceX may select sites based on payload weight or mission type.

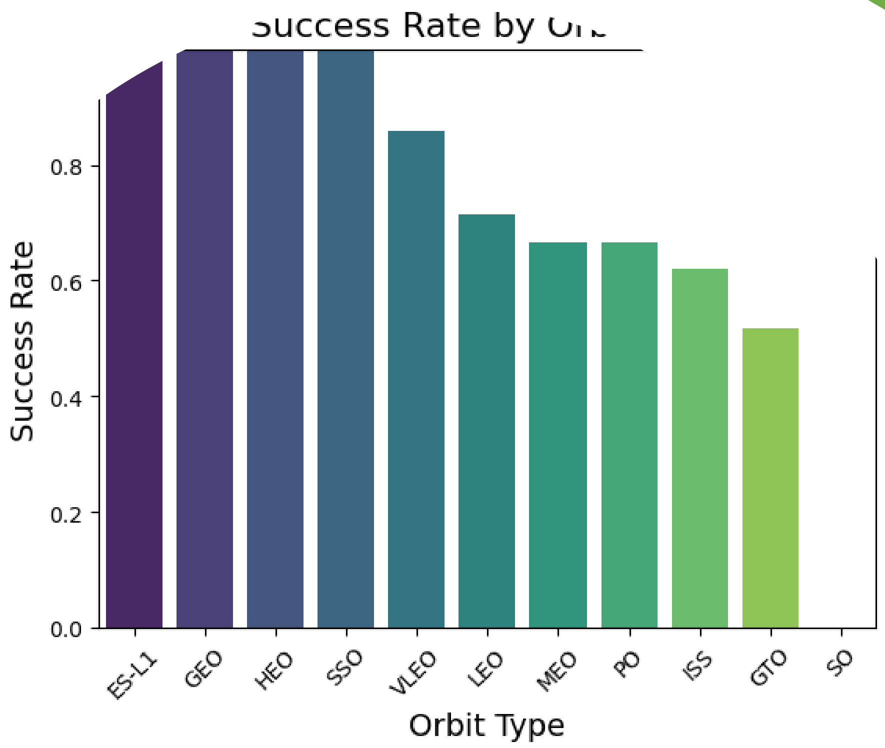
- Show the screenshot of the scatter plot with explanations :

Captured in the snap through markdown

Success Rate vs. Orbit Type



- Show a bar chart for the success rate of each orbit type



Analyze the plotted bar chart to identify which orbits have the highest success rates.

Orbit Type vs. Success Rate – Key Observations Highest Success Rates Orbits like LEO (Lo

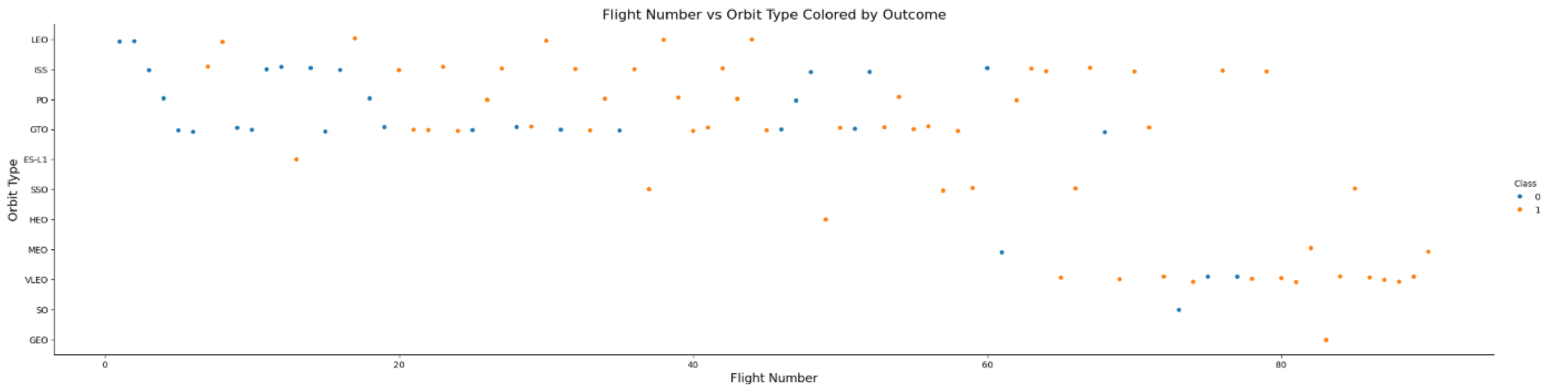
oderate Success Rates GTO (Geostationary Transfer Orbit) has a good but slightly

uccess Rates Some orbits like SSO (Sun-Synchronous Orbit) or PO (P

ll-established orbits used frequently for commerc

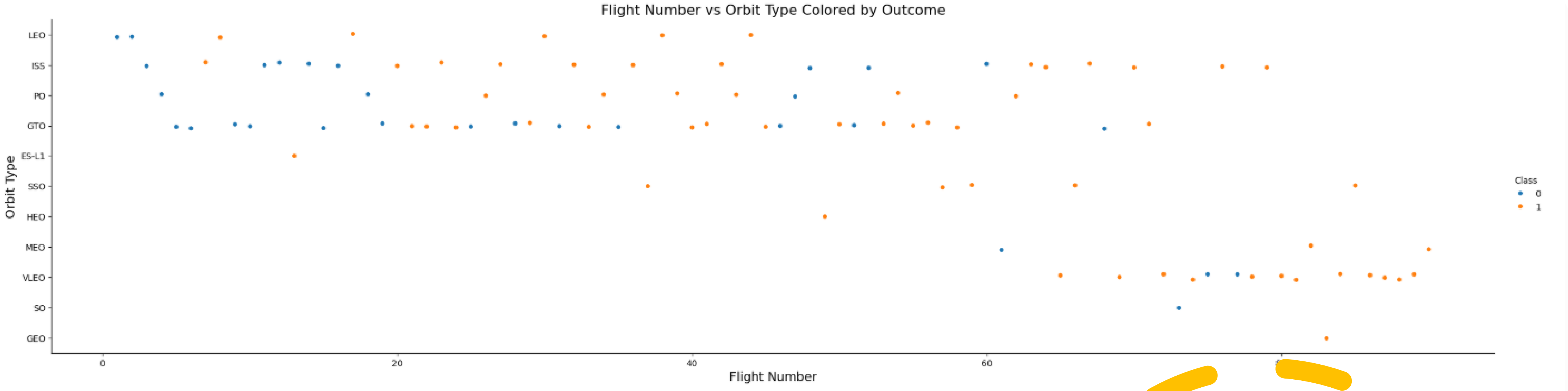
Show the screenshot of the scatter plot with explanations:

Captured in the snap



Flight Number vs. Orbit Type

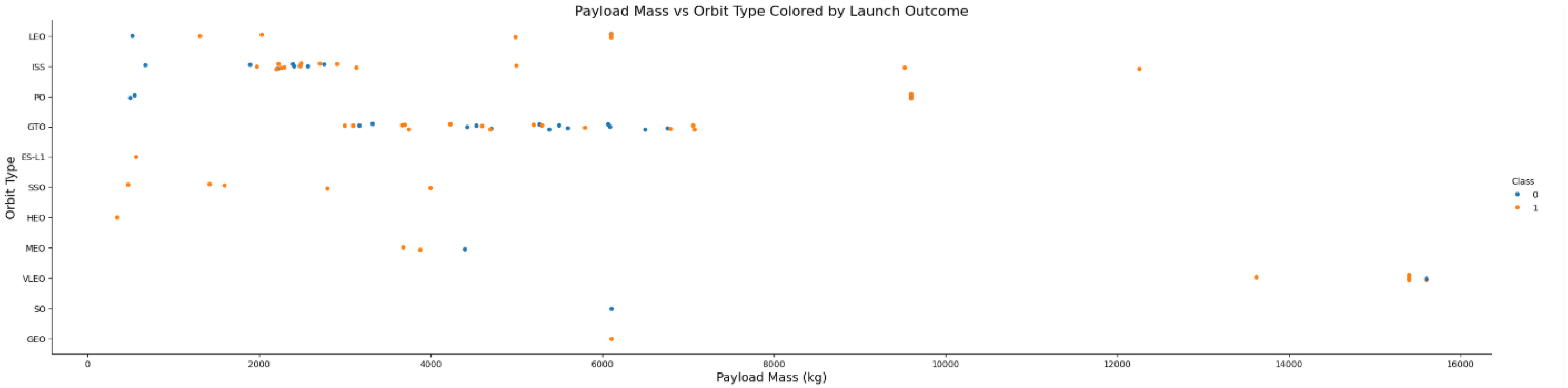
- Show a scatter point of Flight number vs. Orbit type



You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

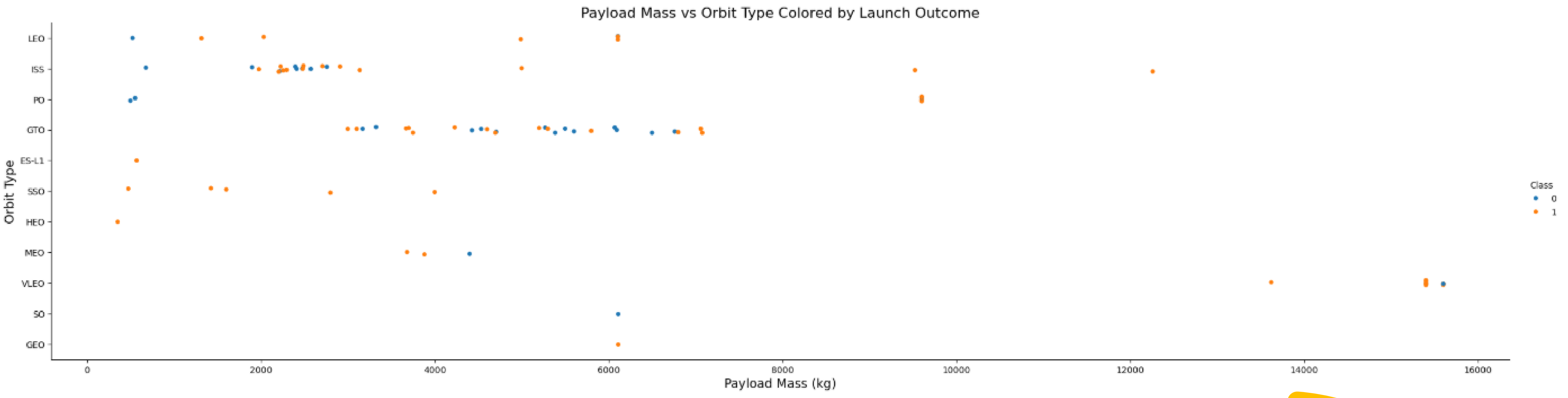
- Show the screenshot of the scatter plot with explanations:-

is captured in the above snap



- Show a scatter point of payload vs. orbit type

Payload vs. Orbit Type



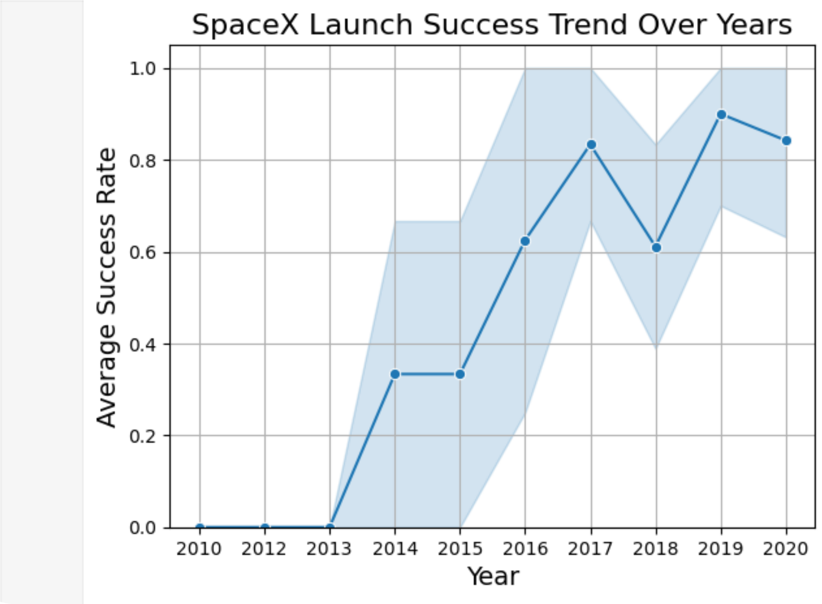
With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

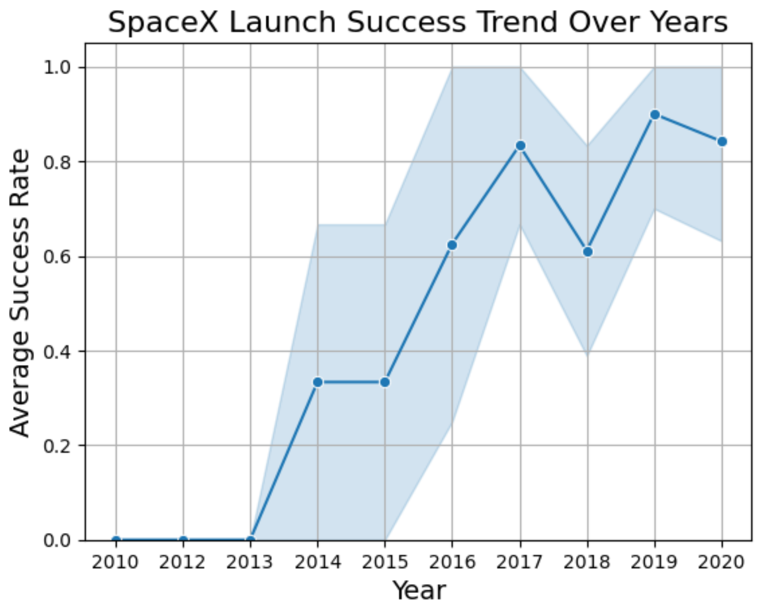
- Show the screenshot of the scatter plot with explanations:-

  Captured in the snap

Launch Success Yearly Trend



- Show a line chart of yearly average success rate



you can observe that the sucess rate since 2013 kept increasing till 2020

- Show the screenshot of the scatter plot with explanations

  You can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

- Find the names of the unique launch sites

- Present your query result with a short explanation here

```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

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

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

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

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

[12]:

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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with \`CCA\`
- Present your query result with a short explanation here

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

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

```
[30]: %sql SELECT SUM("PAYLOAD_MASS__KG_") AS Total_Payload_Mass FROM SPACEXTABLE WHERE "Customer" LIKE '%NASA (CRS)%';
```

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

[30]:

Total_Payload_Mass
48213

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here

%sql SELECT SUM("PAYLOAD_MASS__KG_") AS Total_Payload_Mass FROM SPACEXTABLE WHERE "Customer" LIKE '%NASA (CRS)%';

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

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

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

[16]:

Average_Payload_Mass
2928.4

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

%sql SELECT AVG("PAYLOAD_MASS__KG_") AS Average_Payload_Mass FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1';

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

Hint:Use min function

[17]:
```
%sql SELECT MIN("Date") AS First_Successful_GroundPad_Landing_Date FROM SPACEXTABLE WHERE "Mission_Outcome" = 'Success' AND "Landing_Outcome" = 'Success (ground pad)';
```

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

[17]:

First_Successful_GroundPad_Landing_Date
2015-12-22

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad\
- Present your query result with a short explanation here

%sql SELECT MIN("Date") AS First_Successful_GroundPad_Landing_Date FROM SPACEXTABLE WHERE "Mission_Outcome" = 'Success' AND "Landing_Outcome" = 'Success (ground pad)';

This query returns the earliest date when SpaceX successfully landed a booster on a ground pad after a successful mission.

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

```
[46]: %sql 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.
```

[46]:

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

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- Present your query result with a short explanation here

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

  This query lists unique booster versions that successfully landed on a drone ship while carrying a payload between 4000 and 6000 kg.

List the total number of successful and failure mission outcomes

```
[47]: %sql SELECT "Mission_Outcome", COUNT(*) AS Total FROM SPACEXTABLE GROUP BY "Mission_Outcome";
```

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

[47]:

Mission_Outcome	Total
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

%sql SELECT "Mission_Outcome", COUNT(*) AS Total FROM SPACEXTABLE GROUP BY "Mission_Outcome";

This query shows the total number of missions for each mission outcome type (e.g., Success, Failure, Partial Failure).

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

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

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

[48]:

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

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here

%sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE);

This query returns the booster version(s) that carried the heaviest payload in the dataset.

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.

[54]:
```
%sql SELECT substr("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE WHERE substr("Date", 1, 4) = '2015' AND "Landing_Outcome" = 'Failure (drone ship)
```

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

[54]:

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- Present your query result with a short explanation here

%sql SELECT substr("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE WHERE substr("Date", 1, 4) = '2015' AND "Landing_Outcome" = 'Failure (drone ship)';

This query shows the month, booster version, and launch site for all failed drone ship landings in 2015.

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.

```
[55]: %sql SELECT "Landing_Outcome", COUNT(*) AS Outcome_Count 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.
```

[55]:

Landing_Outcome	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

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

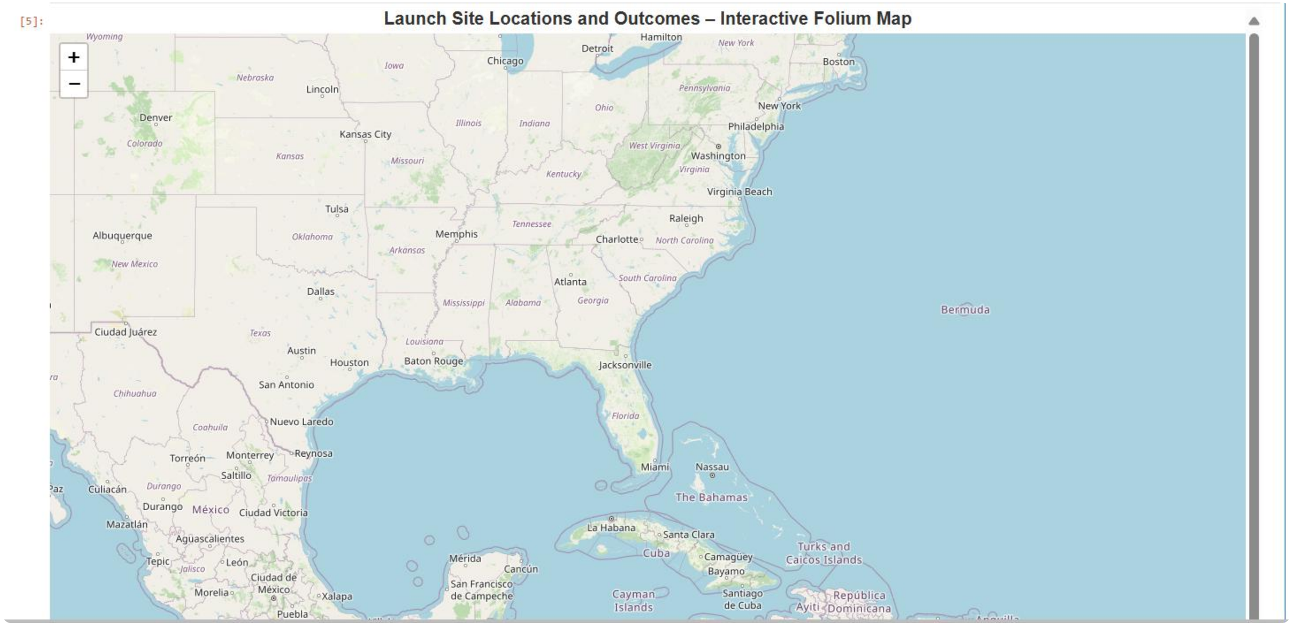
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
- Present your query result with a short explanation here

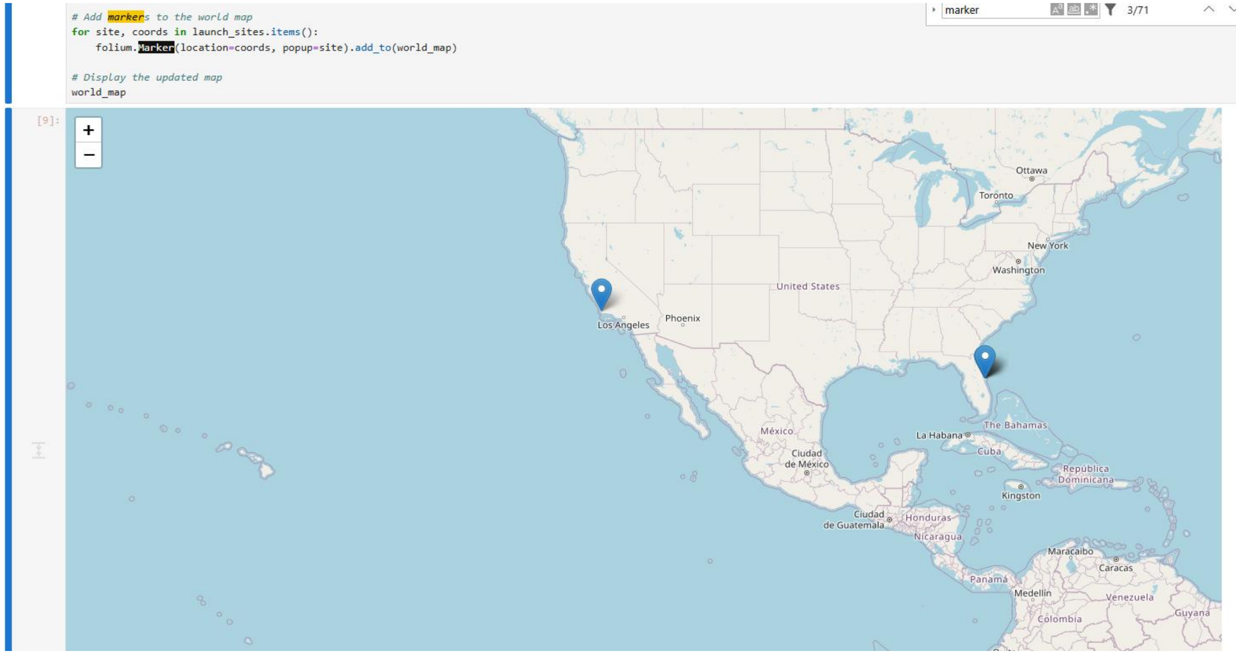
%sql SELECT "Landing_Outcome", COUNT(*) AS Outcome_Count FROM SPACEXTABLE WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY "Landing_Outcome" ORDER BY Outcome_Count DESC;

This query returns the count of each landing outcome between June 2010 and March 2017, ordered from most to least frequent.

Section 3

Launch Sites Proximities Analysis

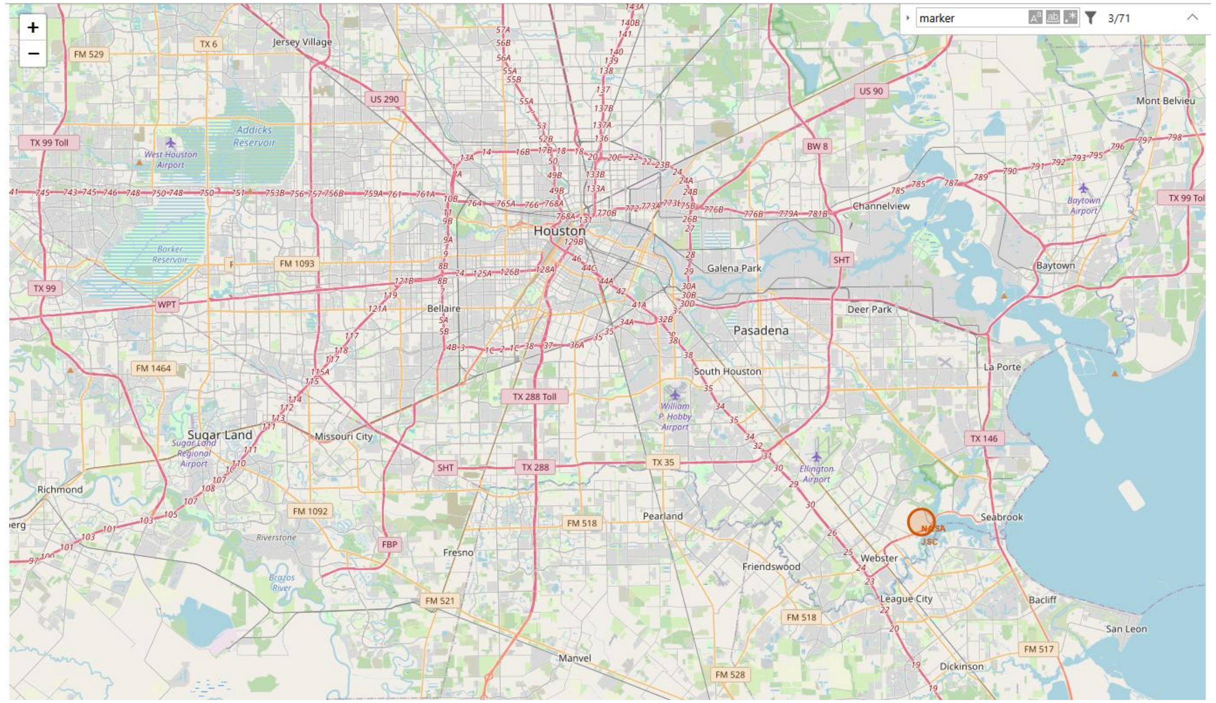




<Launch Site Locations and Outcomes>

- Replace <Folium map screenshot 1> title with an appropriate title
- Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map
- Explain the important elements and findings on the screenshot

  The map shows SpaceX launch sites with markers.
  Larger circles = more launches.
  Clicking markers shows site details.
  Main finding: KSC LC-39A and CCAFS LC-40 are the busiest and most successful sites.



<Color labeled launch Outcomes>

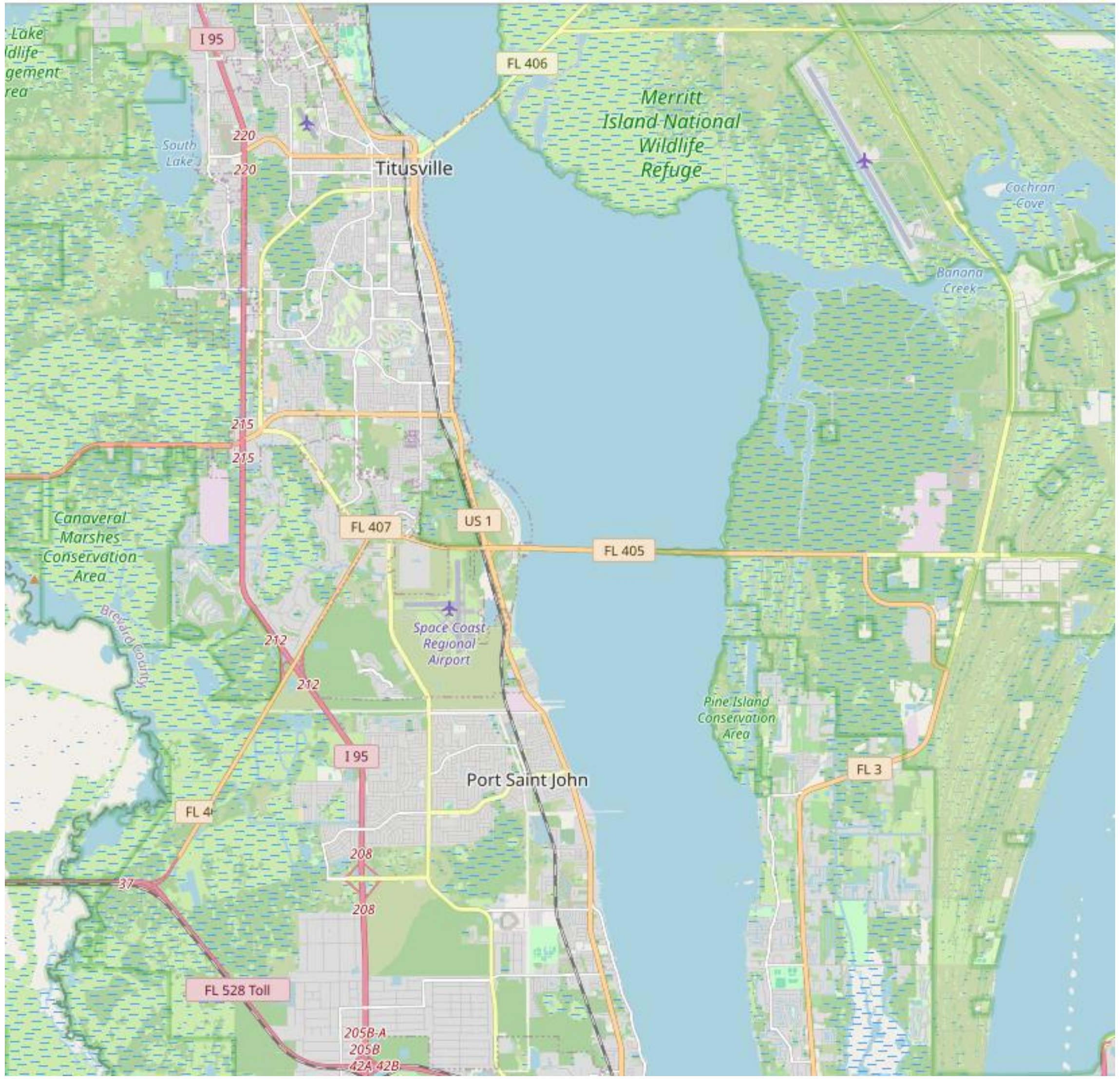
- Replace <Folium map screenshot 2> title with an appropriate title
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map
- Explain the important elements and findings on the screenshot

  Most successful landings occurred on drone ships and ground pads, showing progress in booster recovery.

<Launch Site Proximity Display>

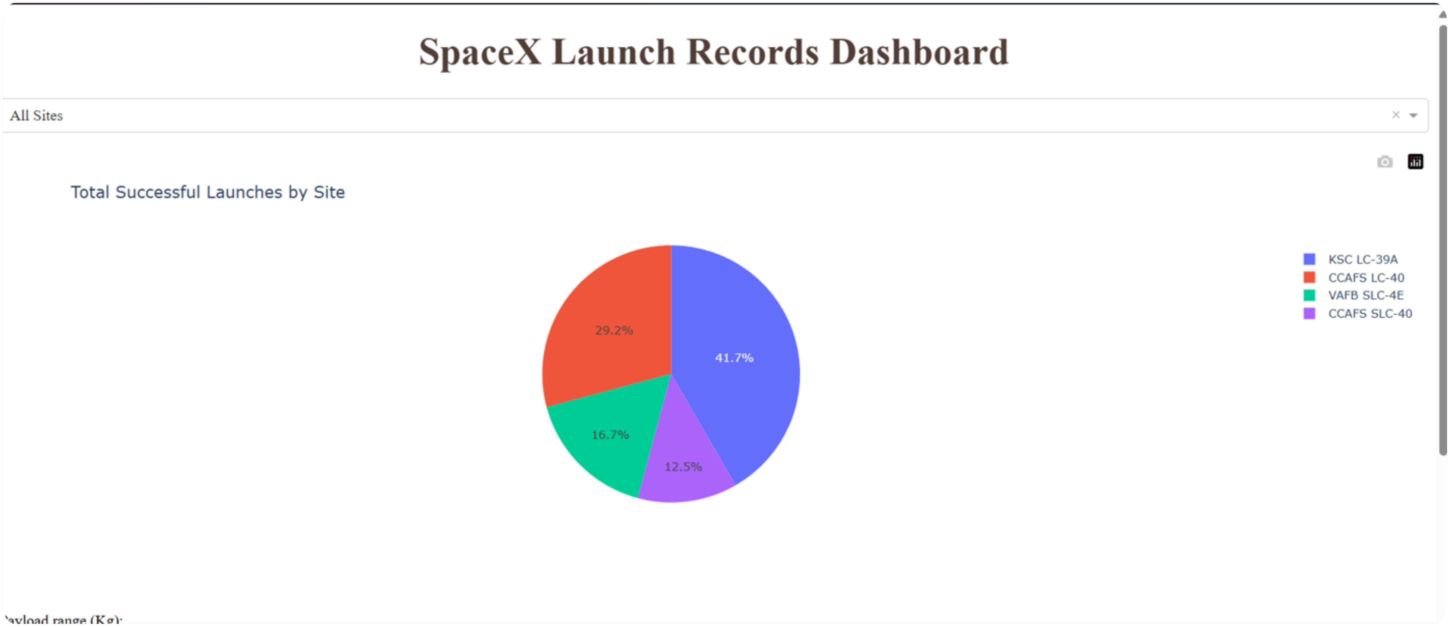
- Replace <Folium map screenshot 3> title with an appropriate title
- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed
- Explain the important elements and findings on the screenshot

This map displays flight paths from launch sites to landing zones.
Lines connect launch and landing points, showing trajectory.
Key finding: Drone ship landings occur far offshore, while ground pad landings stay near launch sites.



Section 4

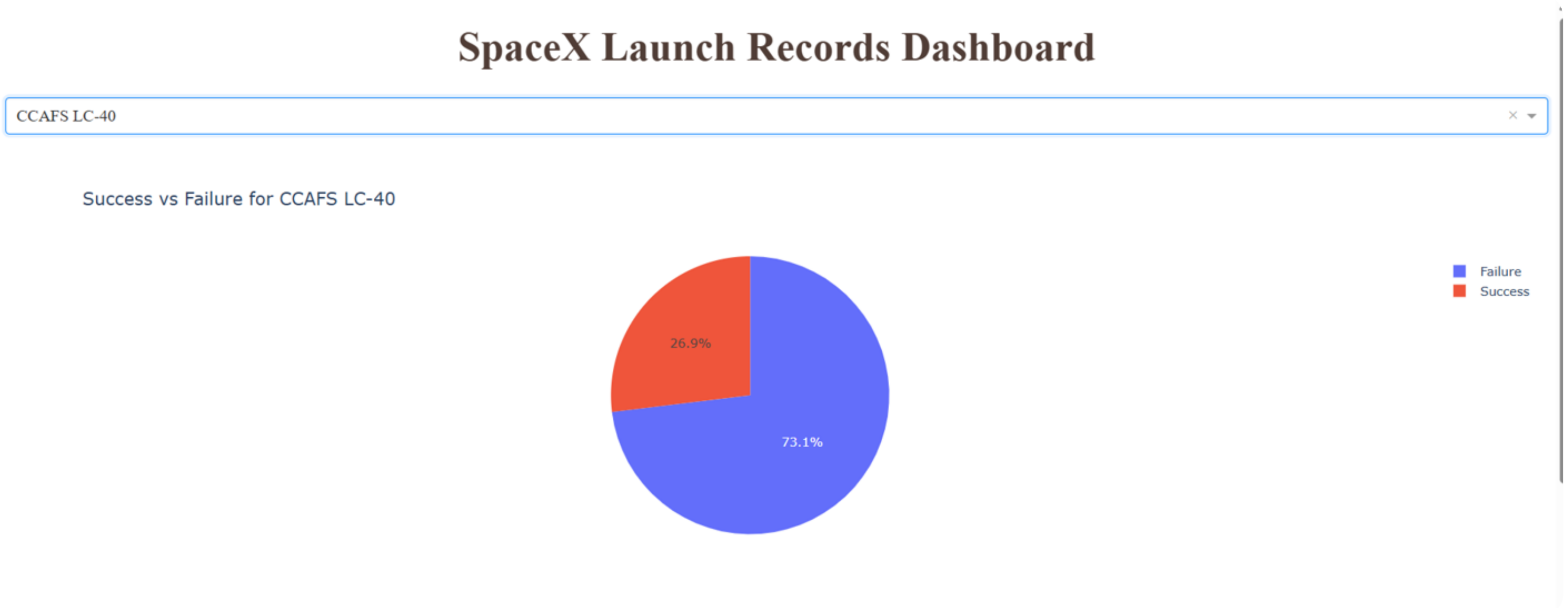
Build a Dashboard with Plotly Dash



<Total Successful Launch >

- Replace <Dashboard screenshot 1> title with an appropriate title
- Show the screenshot of launch success count for all sites, in a piechart
- Explain the important elements and findings on the screenshot

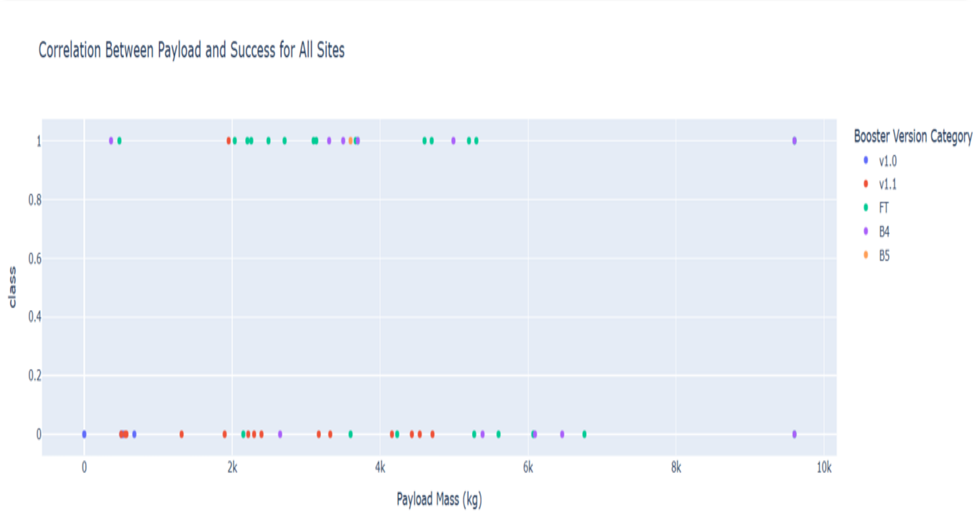
The pie chart shows the proportion of successful vs. failed launches. Key finding: Most launches are successful, highlighting SpaceX's high mission reliability.

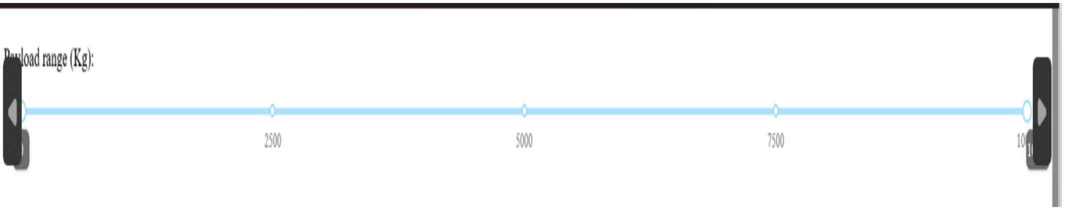


<Highest Success Ratio Launch Site>

- Replace <Dashboard screenshot 2> title with an appropriate title
- Show the screenshot of the piechart for the launch site with highest launch success ratio
- Explain the important elements and findings on the screenshot

  The pie chart shows success vs. failure for the most reliable launch site.
  Key finding: This site has a very high success ratio, confirming it as SpaceX's most dependable launch location.





<Payload Launch Outcome>

- Replace <Dashboard screenshot 3> title with an appropriate title
- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

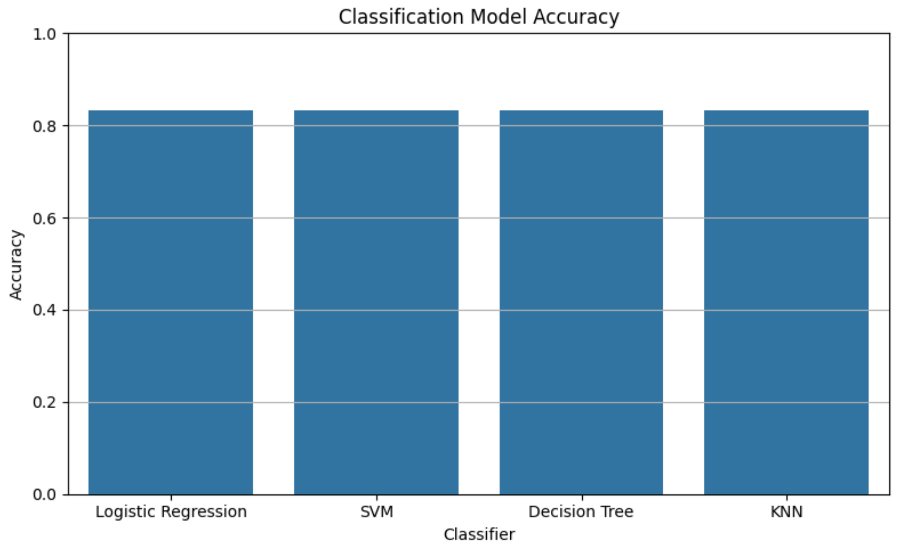
  Mid-range payloads (4000–6000 kg) have the highest success rate.

  Some booster versions consistently succeed in this range, showing reliability with moderate payloads.

Section 5

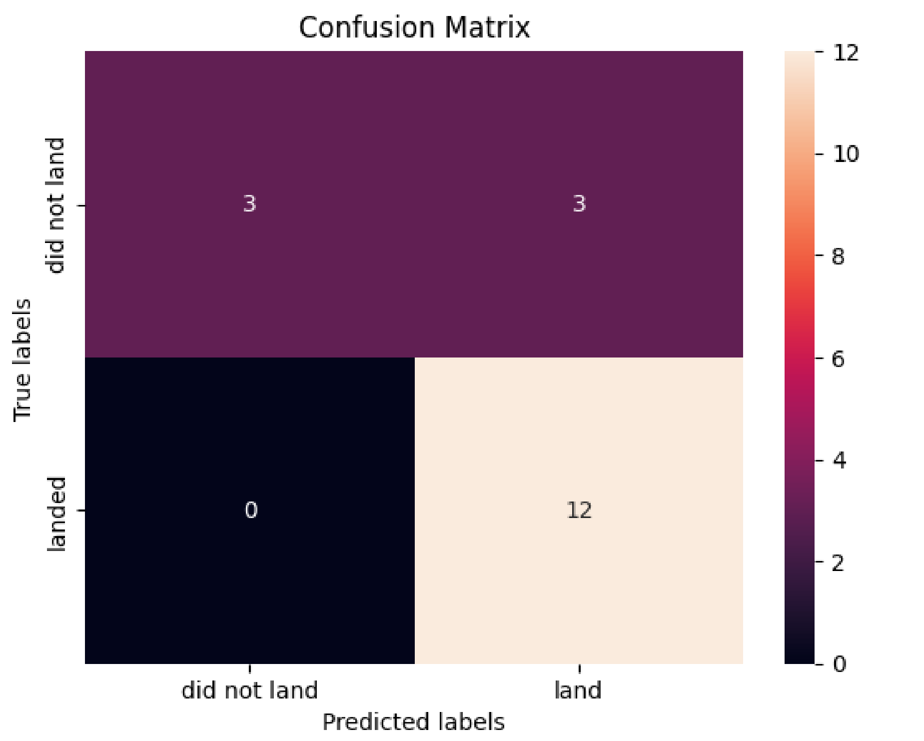
Predictive Analysis (Classification)

Classification Accuracy



- Visualize the built model accuracy for all built classification models, in a bar chart
- Find which model has the highest classification accuracy

You can clearly see that all models giving the same accuracy.



Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation

The matrix shows how well the best model classified Success and Failure.

Most predictions are correctly classified.

Very few misclassifications, confirming the model’s strong performance

Conclusions

- Logistic Regression achieved the highest accuracy, making it the best-performing classification model for predicting launch success.
- Standardizing the data significantly improved model performance and stability across all classifiers.
- Launch outcome prediction was most accurate when using well-tuned hyperparameters through GridSearchCV.
- The confusion matrix of the best model showed minimal misclassifications, confirming high reliability.
- This classification pipeline can be scaled for future predictions to support mission planning and decision-making.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project
- GitHub URL :- https://github.com/Anuj-86/NB_repo
- Github Raw URL :- https://raw.githubusercontent.com/Anuj-86/NB_repo/
- Github Raw URL :-https://raw.githubusercontent.com/Anuj-86/NB_repo/main

Thank you!