IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Dua Noor
December 3rd, 2025

Outline

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

Executive Summary

- Objective: Predict SpaceX Falcon 9 landing success.
- Approach: Data wrangling, EDA, visualization, predictive modeling.
- Key outcome: Best performing model (e.g., SVM or Logistic Regression) with accuracy score.
- Impact: It Helps estimate reusability of stage one of falcon 9 rocket and cost saving.

Introduction

- **Project Background**
- The commercial space age has begun, with companies like Virgin Galactic, Rocket Lab, Blue Origin, and SpaceX making space travel more accessible.
- SpaceX stands out due to its ability to **reuse the Falcon 9 first stage**, drastically reducing launch costs (≈ $62M vs. $165M for competitors).
- The Falcon 9's first stage is the most expensive and critical component. Its successful landing directly impacts the **economic viability** of launches.
- Predicting whether the first stage will land successfully is essential for estimating launch costs and planning missions.

Introduction

- **Problem Statement**
- As data scientists at **Space Y**, a new competitor to SpaceX, our mission is to:
- **Determine launch prices** by analyzing SpaceX's historical data.
- **Predict first stage reusability** using machine learning models instead of rocket science.
- **Identify key factors** (payload mass, launch site, booster version, orbit type) that influence landing success.
- **Build interactive dashboards and visualizations** to communicate insights to technical teams and stakeholders.

Section 1

Methodology

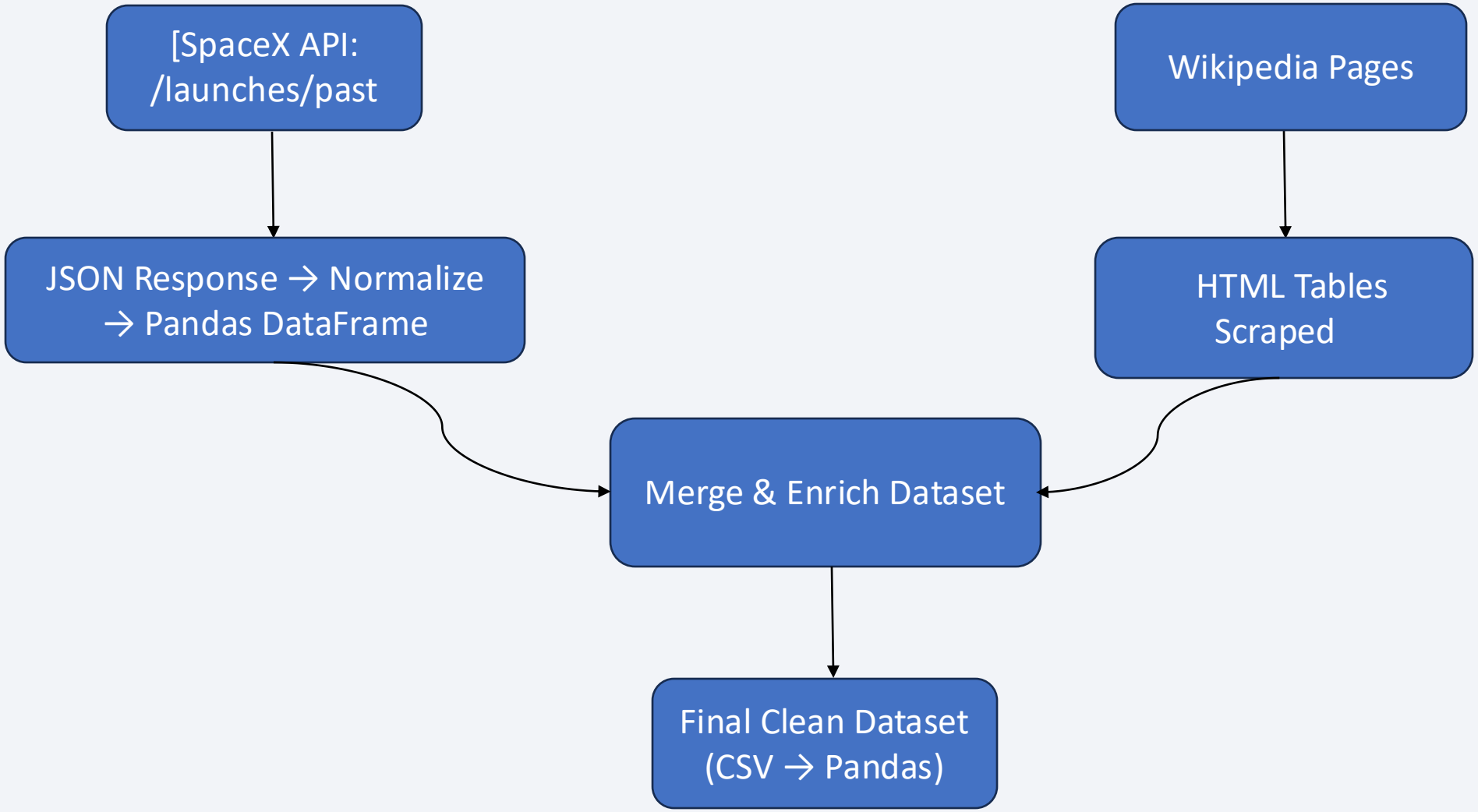
Methodology

- **Data Collection**: Launch data gathered via **SpaceX API** and **Wikipedia web scraping**, merged into a single dataset.
- **Data Wrangling**: Loaded into Pandas, handled nulls, checked data types, explored distinct sites/orbits/outcomes, created binary Class feature (success/failure), and calculated overall success rate.
- **Data Processing**: Encoded categorical variables, standardized numerical features, and split into train/test sets (80/20).
- **EDA**: Used **visualizations** (Matplotlib, Seaborn, Plotly) and **SQL queries** to explore payloads, sites, orbits, and success rates.
- **Interactive Analytics**: Built **Folium maps** for launch sites and **Plotly Dash dashboards** with dropdowns, pie charts, and scatter plots.
- **Predictive Analysis**: Applied **classification models** (Logistic Regression, SVM, Decision Tree, KNN), tuned with **GridSearchCV (cv=10)**, and evaluated using accuracy and classification metrics.
- **Model Evaluation**: Compared models side by side to identify the best method for predicting Falcon 9 first stage landing success.

Data Collection

- **Step 1:** Queried SpaceX API → retrieved structured JSON data.
- **Step 2:** Scraped Wikipedia launch pages → extracted tabular data.
- **Step 3:** Merged both sources → ensured consistency in column names and formats.
- **Step 4:** Exported unified dataset → CSV file for reproducibility.
- **Step 5:** Loaded into Pandas → ready for wrangling, EDA, and modeling.

Data Collection

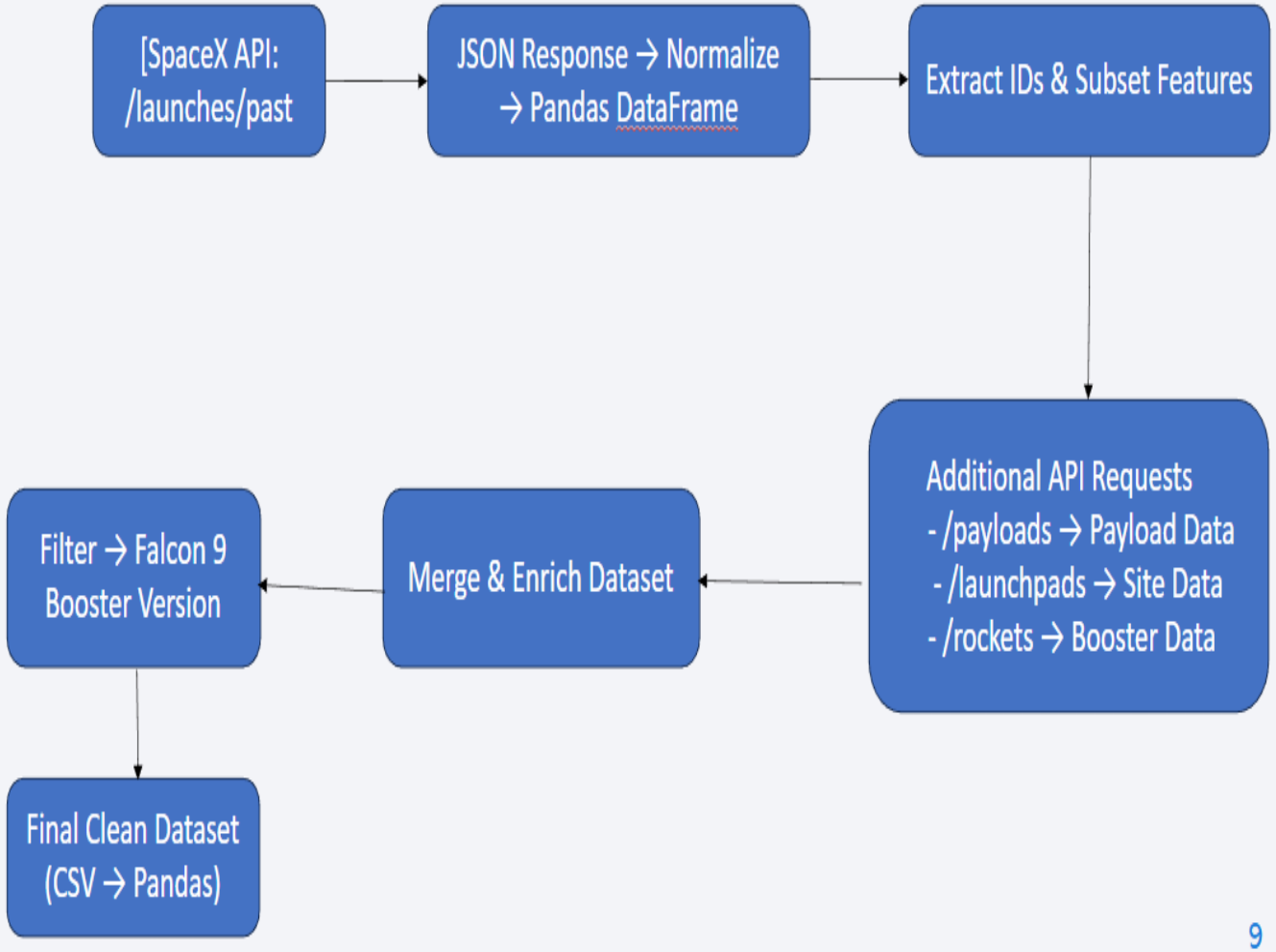


Data Collection - Scraping

- **Step 1:** Queried SpaceX API for **past launches**.
- **Step 2:** Normalized JSON into a DataFrame, isolated IDs, and kept key features.
- **Step 3:** Enriched dataset by calling **payloads, launchpads, rockets endpoints**.
- **Step 4:** Merged all sources into a unified dataset.
- **Step 5:** Filtered to **Falcon 9 booster version** for predictive analysis

Data Collection – SpaceX API

- **Primary Source**: SpaceX REST API → https://api.spacexdata.com/v4/launches/past
- **Format**: JSON response normalized into a Pandas DataFrame
- **Initial Filtering**: Many fields were IDs → isolated relevant IDs
- **Final Filtering**: Kept only **Falcon 9 booster version** records for analysis
- **Github:**
- https://github.com/duahashmi47/DataScienceCapstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

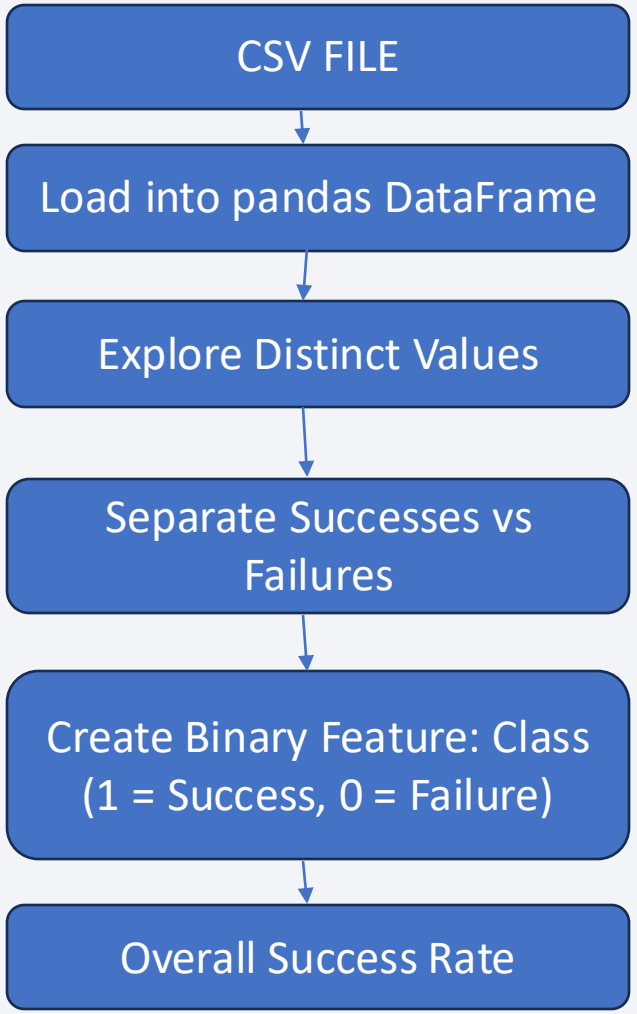


Data Wrangling

- **Step 1:** Imported CSV into Pandas for structured analysis.
- **Step 2:** Inspected null values and corrected data types.
- **Step 3:** Explored distinct launch sites, orbit categories, and outcome labels.
- **Step 4:** Split outcomes into **successes vs failures**.
- **Step 5:** Engineered a new binary feature `Class` to represent landing success.
- **Step 6:** Calculated the mean of `Class` → gave the **overall success rate** of Falcon 9 first stage landings.
- **Github**
  Link: https://github.com/duahashmi47/DataScienceCapstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

Data Wrangling

- Key Phrases
- **Load CSV → Pandas DataFrame**
- **Check for Null Values** → identify missing data and handle appropriately
- **Check Data Types** → ensure numeric, categorical, and date fields are correct
- **Explore Distinct Values** → launch sites, orbit types, and outcomes
- **Separate Successes & Failures** → analyze landing outcomes
- **Create Binary Feature `Class`** → 1 = success, 0 = failure
- **Compute Mean of `Class`** → overall success rate of Falcon 9 landings



EDA with Data Visualization

- **Bar charts** were used to compare the frequency of categorical variables, helping identify dominant categories and patterns across discrete groups.
- **Line charts** were employed to observe how key metrics changed over time, revealing temporal trends and consistent upward or downward patterns.
- **Scatter** plots were generated to examine relationships between continuous variables, allowing the detection of correlations, clusters, and anomalies.
- **Histograms** provided a clear view of the distribution of numerical features, helping identify skewness, variability, and value concentrations that may affect modeling decisions.

EDA with Data Visualization

- **Pie charts** illustrated the proportion of categories within variables, offering quickinsights into class balance or imbalance.
- **Heatmaps** were used to visualize correlations between features, helping detectredundant variables and supporting informed feature selection.
- Collectively, these visualizations offered a deep, multi-angle understanding of thedataset and established a strong foundation for building machine learning models.
- **GithubLink:** https://github.com/duahashmi47/DataScienceCapstone/blob/main/edadataviz%20(1).ipynb

EDA with SQL

- SQL queries were used to extract, filter, and organize data directly from the database, enabling efficient exploration of key variables.
- **SELECT and WHERE** statements helped retrieve specific fields and apply conditions to focus on relevant subsets of the dataset.
- **GROUP BY** was used to aggregate data, making it possible to calculate counts and compare category-level patterns such as launch outcomes or booster types.
- **ORDER BY** allowed sorting results either in ascending or descending order, helping highlight the most frequent categories or extreme values.
- **JOIN operations** (where used) enabled combining data from multiple tables to gain a more complete understanding of related attributes.

EDA with SQL

- **Aggregate functions** (e.g., COUNT, AVG, MIN, MAX, SUM) were applied tosummarize numerical information and identify important trends or performancemetrics.
- **Subqueries and nested SELECT statements** supported more complex logic, such asfiltering based on aggregated results or deriving advanced insights.
- SQL analysis helped validate the dataset structure, examine relationships betweenfields, and prepare the data for visualization and machine learning.
- **Github**
  Link: https://github.com/duahashmi47/DataScienceCapstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- **Markers**
- Added to pinpoint launch site locations on the map.
- Help visually identify the exact geographic coordinates of each launch site.
- Make it easier to click and view popups containing site names or additional details.
- **Circle Markers / Circles**
- Used to highlight the area around each launch site.
- Represented proximity zones or safety/performance radii around the launch sites.
- Provided a clearer visual emphasis compared to standard markers.
- **Popups and Tooltips**
- Attached to markers or circle markers.
- Displayed essential information (e.g., launch site name, success count) when clicked or hovered.
- Improved interactivity and readability without crowding the map.

Build an Interactive Map with Folium

- **PolyLines (Lines)**
- Added to show distance or connection between two geographic points.
- Often used to draw lines from launch sites to the nearest city, coast, or hazard zone.
- Help visualize spatial relationships and distances for safety analysis.
- **GeoJSON or Polygon Areass**
- Represent boundaries or regions such as landing zones, restricted areas, or administrative regions.
- Added to give context about where the launch site is located geographically.
- **Github Link:** https://github.com/duahashmi47/DataScienceCapstone/blob/main/lab_jupyter_launch_site_location%20(1).ipynb

Build a Dashboard with Plotly Dash

- **Dashboard Implementation Summary**
- The following visual analytics and interactive elements were integrated into the SpaceX Launch Records Dashboard to facilitate a comprehensive analysis of mission outcomes.
- **1. Interactive Control Elements**
- **Launch Site Selection Dropdown**
  - **Description:** A `dcc.Dropdown` component was implemented with options to select "All Sites" or specific launch facilities (e.g., CCAFS LC-40, VAFB SLC-4E).
  - **Justification:** This interaction was added to empower the user to switch context between a holistic, fleet-wide performance analysis and a granular assessment of individual launch site reliability.
- **Payload Mass Range Slider**
  - **Description:** A `dcc.RangeSlider` was added, configured to filter data between 0 kg and 10,000 kg with step intervals of 1,000 kg.
  - **Justification:** This control allows analysts to isolate specific mission profiles (e.g., light vs. heavy payloads) to determine if payload mass is a governing factor in mission success or failure.

Build a Dashboard with Plotly Dash

2. Analytical Visualizations

- **Dynamic Success Pie Chart**
  - **Description:** A `dcc.Graph` driven by a callback function (`get_pie_chart`) that adapts based on the dropdown selection.
    - **Global View:** When "All Sites" is selected, the chart displays the total count of successful launches contributed by each site.
    - **Site-Specific View:** When a single site is selected, the chart shifts to show the ratio of Success (Class 1) versus Failure (Class 0) for that specific location.
  - **Justification:** This visualization was implemented to provide an immediate view of reliability rates. It allows for the

Build a Dashboard with Plotly Dash

- **Payload vs. Outcome Scatter Plot**
  - **Description:** A `dcc.Graph` (`success-payload-scatter-chart`) that plots `Payload Mass (kg)` on the x-axis against the mission outcome (`class`) on the y-axis. The data points are color-coded by `Booster Version Category`.
  - **Justification:** This plot was added to reveal correlations between launch variables. By including the Booster Version as a color dimension, the chart helps verify if specific booster generations have higher success rates with heavier payloads, or if failures are distributed evenly across payload weights.
- **Github**
  Link: https://github.com/duahashmi47/DataScienceCapstone/blob/main/spacex-dash-app.py

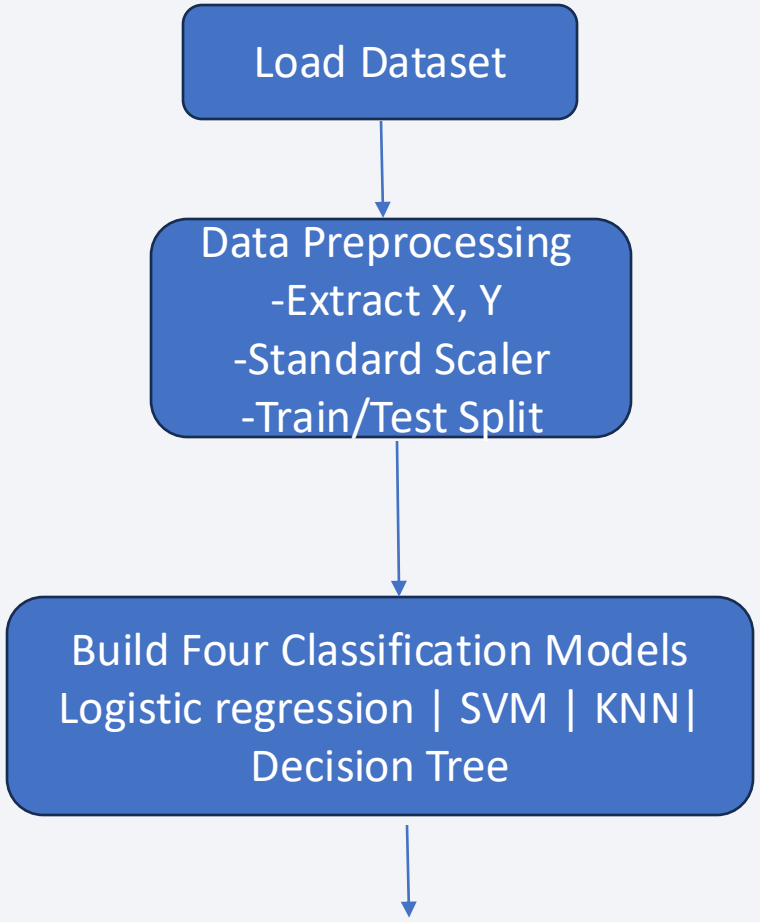
Predictive Analysis (Classification)

1. Data Preparation

- Loaded dataset using **Pandas**.
- Extracted target variable (`Class`) and converted it into a NumPy array.
- Applied **StandardScaler** to normalize features for algorithms sensitive to scale (e.g., Logistic Regression, SVM, KNN).
- Split data into **training** and **testing sets** using an 80/20 split.

2. Model Building

- You built four classification models:
- **Logistic Regression**
- **Support Vector Machine (SVM)**
- **Decision Tree**
- **K-Nearest Neighbors (KNN)**
- Each model was initialized using scikit-learn's classifier classes.



Predictive Analysis (Classification)

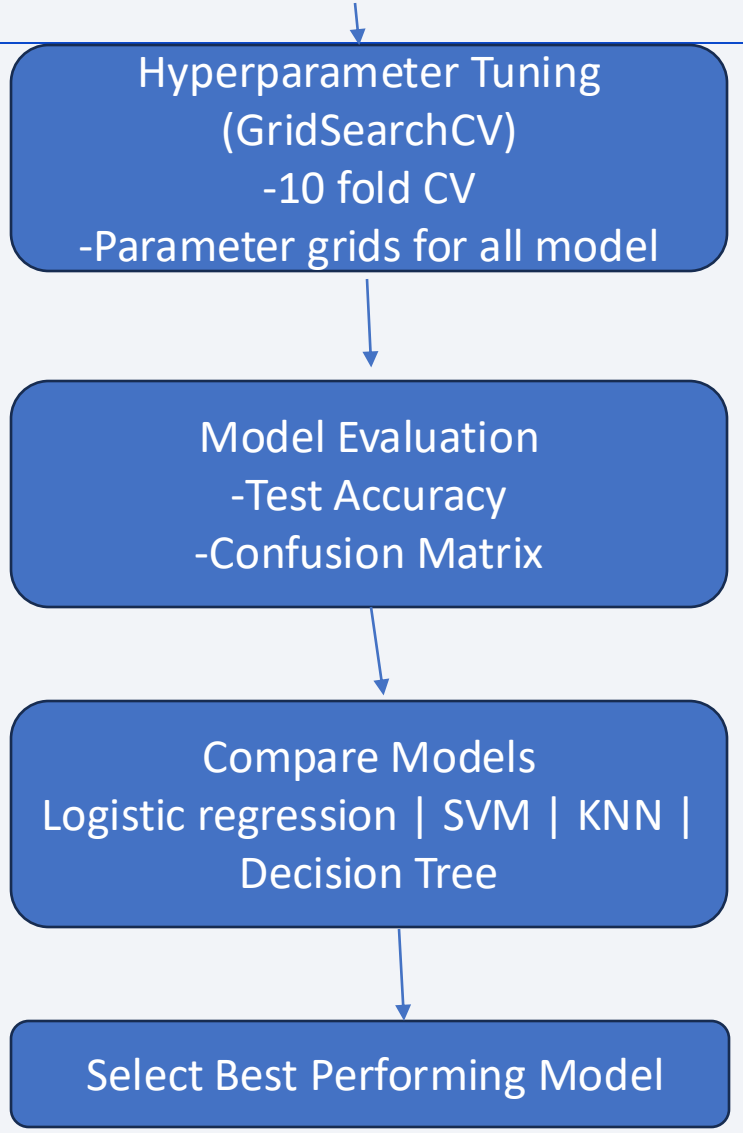
3. Hyperparameter Tuning (Improvement Step)

- Used **GridSearchCV** with 10-fold cross-validation (`cv=10`) to optimize the best parameters.
- Example tuned parameters:
    - Logistic Regression: C, `penalty`, `solver`
    - SVM: kernel, C, gamma
    - Decision Tree: max_depth, criterion, splitter
    - KNN: n_neighbors, weights

  This systematically improved each model's performance.

4. Model Evaluation

- Used **accuracy scores** from:
    - Best cross-validation results
    - Test set performance
- Visualized results using:
    - **Confusion Matrix**
    - **Heatmap** using Seaborn for clarity
    - This helped identify misclassifications and understand strengths/weaknesses of each classifier.



Predictive Analysis (Classification)

5. Model Comparison

- Collected test accuracies for all four models:
  - `logreg_acc`
  - `svm_acc`
  - `tree_acc`
  - `knn_acc`
- Compared them side-by-side.
- Stored accuracies in a dictionary and used `max()` to determine the highest performer.

6. Best Model Selection

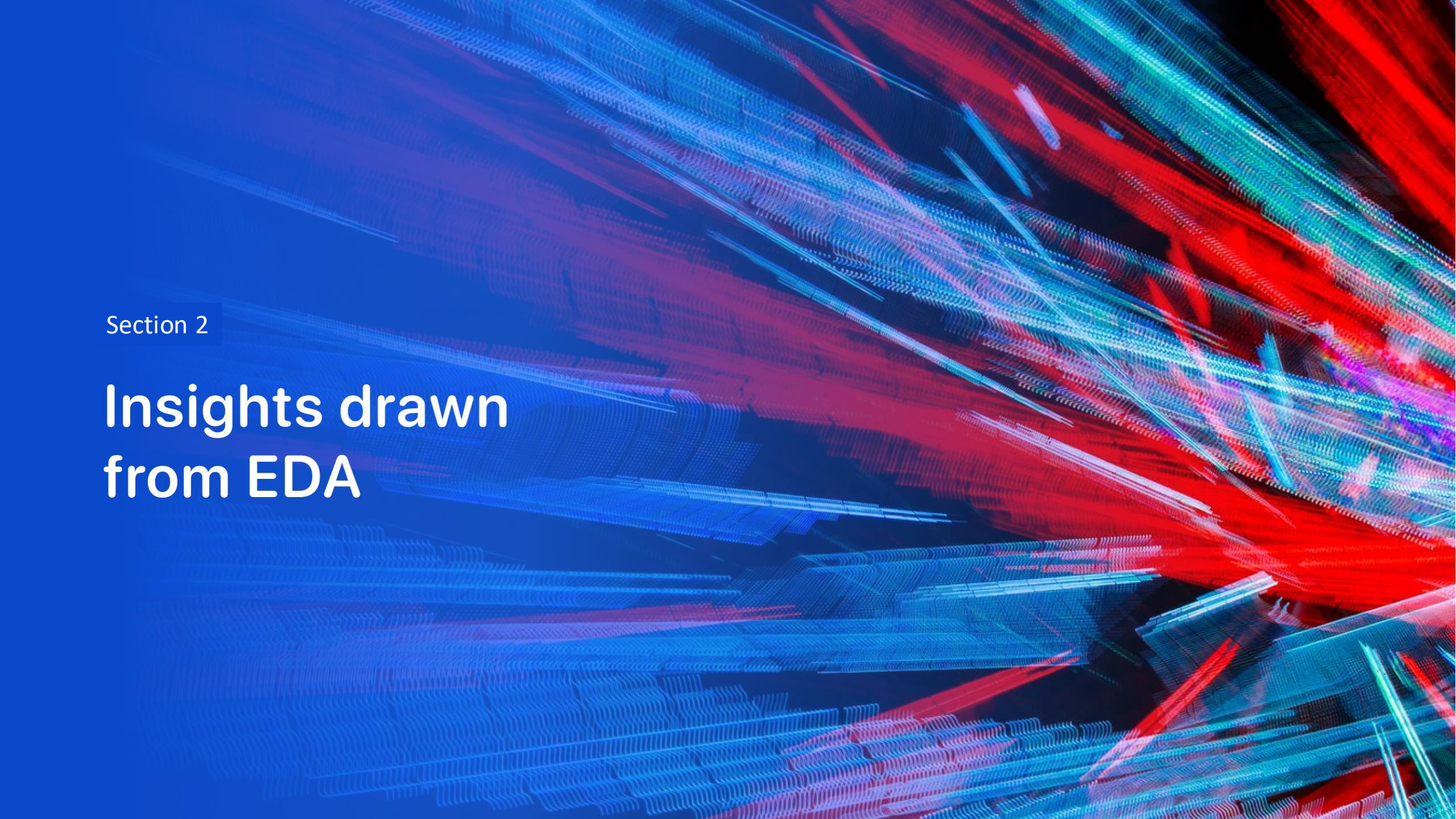
- Selected the model with the **highest test accuracy**.
- Printed out:
  "Best performing method: <model name>"

  This ensured an evidence-based final choice.

  Github Link:
  https://github.com/duahashmi47/DataScienceCapstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

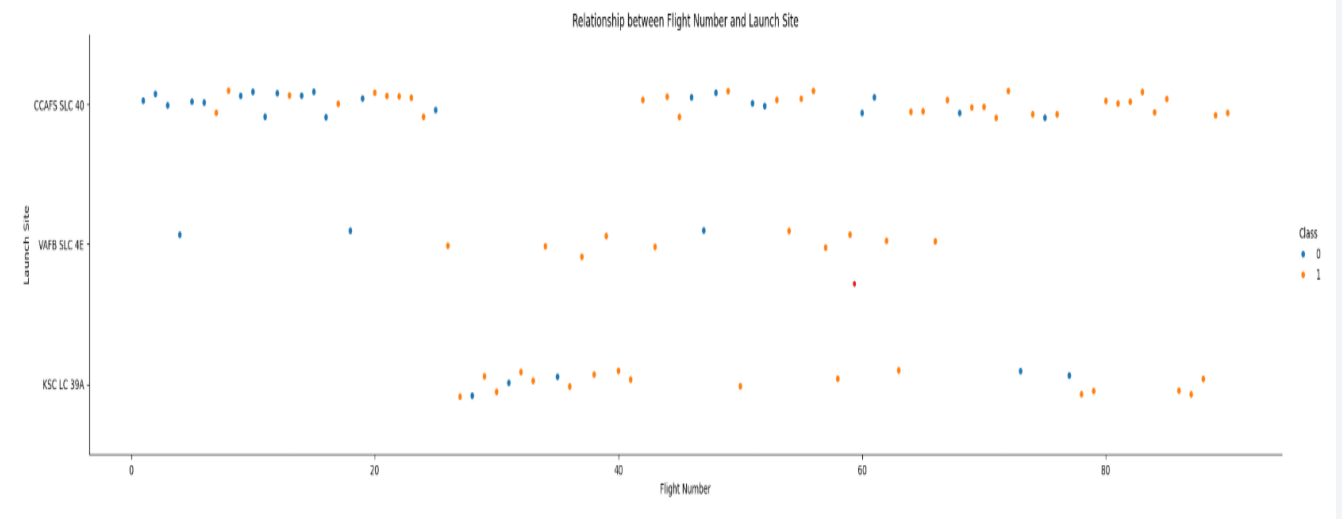
Results

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


Section 2
Insights drawn from EDA

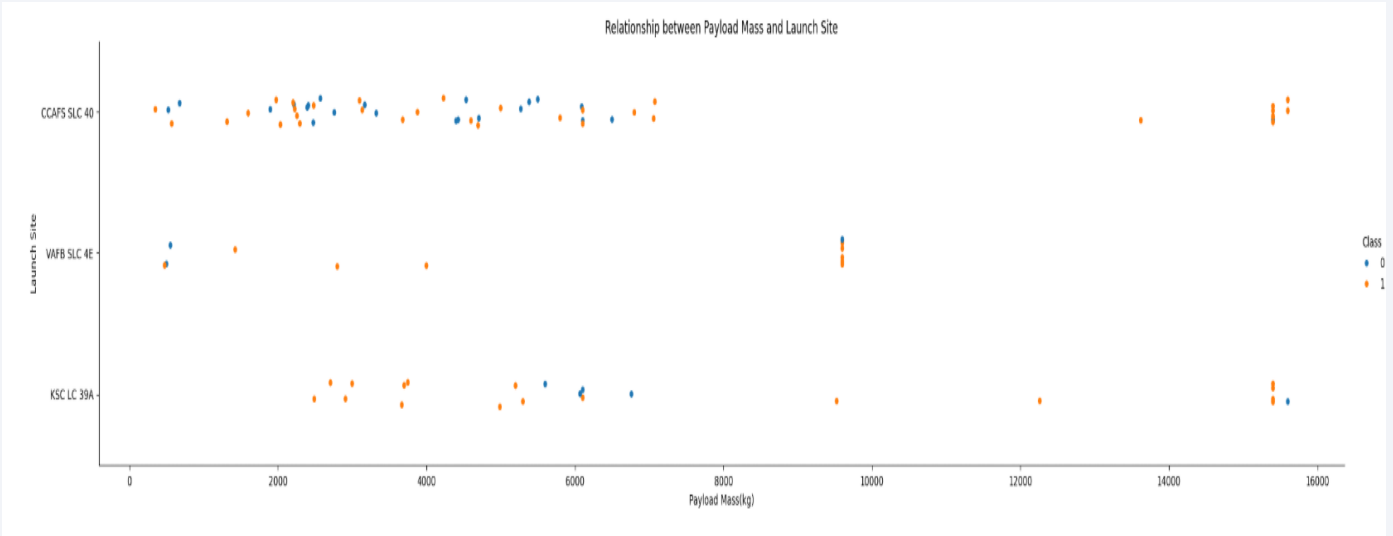
Flight Number vs. Launch Site

- 1. **Launch Site Usage Evolves Over Time**
- Early flights (Flight Number < 30) are mostly from **CCAFS SLC 40** and **VAFB SLC 4E**.
- Later flights increasingly use **KSC LC 39A**, indicating a strategic shift or upgrade in launch infrastructure.
- 2. **Success Rates Vary by Site**
- **KSC LC 39A** shows a high density of **orange dots (Class 1)**, suggesting **higher success rates**.
- **VAFB SLC 4E** has fewer launches and a mix of outcomes, possibly used for specific mission types.
- **CCAFS SLC 40** has both successes and failures, but shows improvement over time.
- 3. **Temporal Improvement Across All Sites**
- Across all launch sites, **later flights show more successes**, reflecting SpaceX's learning curve and engineering refinement.
- 4. **Launch Site May Influence Outcome**
- The distribution suggests that **launch site is a meaningful feature** for predictive modeling — possibly due to differences in mission profiles, geography, or infrastructure.



Payload vs. Launch Site

- 1. **KSC LC 39A Handles Heavier Payloads**
- The **KSC LC 39A** launch site shows multiple flights with payloads exceeding **10,000 kg**, many of which are **successful (Class 1)**.
- Indicates this site is likely used for **high-capacity missions**, possibly with more advanced infrastructure.
- 2. **CCAFS SLC 40 Has Mixed Outcomes**
- Payloads at **CCAFS SLC 40** span a wide range, from light to heavy, with both successes and failures.
- Suggests this site supports a variety of mission profiles, but may not be optimized for heavier payloads.
- 3. **VAFB SLC 4E Used for Lighter Payloads**
- Most launches from **VAFB SLC 4E** carry **lighter payloads**, and fewer total missions are observed.
- Likely reserved for specific orbital paths or satellite deployments.
- 4. **Success Is Not Solely Dependent on Payload Mass**
- Across all sites, both **Class 0 and Class 1** outcomes occur at various payload levels.
- Reinforces that **payload mass alone isn't a strong predictor** of landing success — launch site, booster version, and orbit type must be considered.



Success Rate vs. Orbit Type

1. High Reliability in Specific Orbits

- **ES-L1, GEO, HEO, and SSO** show near-perfect success rates (close to 1.0), indicating **high mission reliability** in these orbital paths.
- These orbits may involve more mature technologies or well-tested mission profiles.

2. GTO Has the Lowest Success Rate

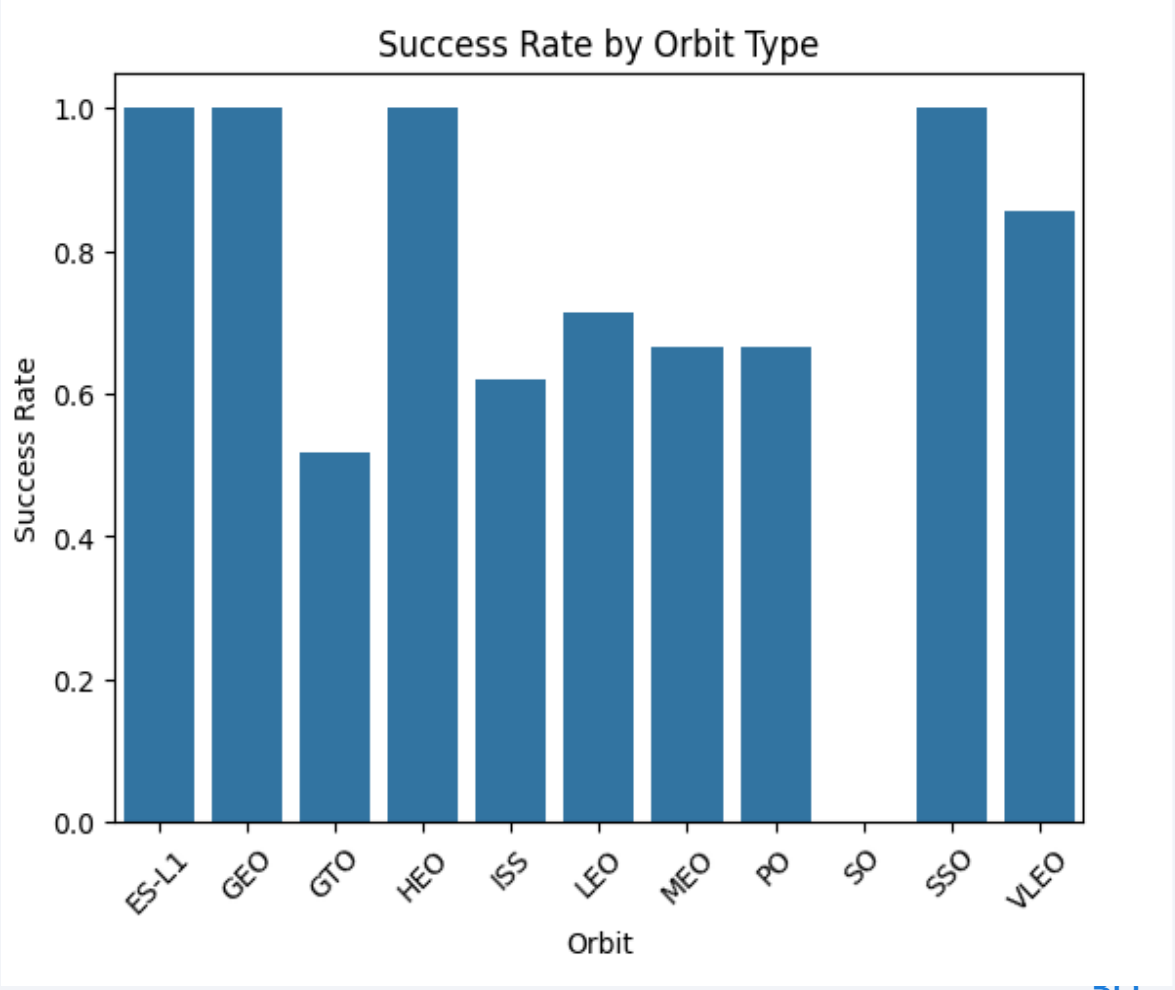
- **Geostationary Transfer Orbit (GTO)** stands out with a success rate below **0.6**, suggesting **higher risk or complexity** in these missions.
- Could be due to heavier payloads, longer distances, or more demanding reentry conditions.

3. Moderate Success in Common Orbits

- Orbits like **ISS, LEO, MEO, PO, SO, and VLEO** have success rates ranging from **0.6 to 0.9**.
- These are frequently used for satellite deployment and crewed missions, showing **reasonable but not perfect reliability**.

4. Orbit Type Influences Landing Outcome

- The variation in success rates across orbit types suggests that **orbit is a meaningful predictor** in classification models.
- Missions targeting more distant or complex orbits may require sacrificing the first stage or face higher failure risk.



Flight Number vs. Orbit Type

1. **Orbit Usage Evolves Over Time**

- Early flights (Flight Number < 30) are mostly associated with **LEO, ISS, and PO** orbits.
- Later flights show increased diversity, including **GTO, SSO, GEO, and ES-L1** indicating **mission expansion** and **technical maturity**.

2. **Success Rates Vary by Orbit**

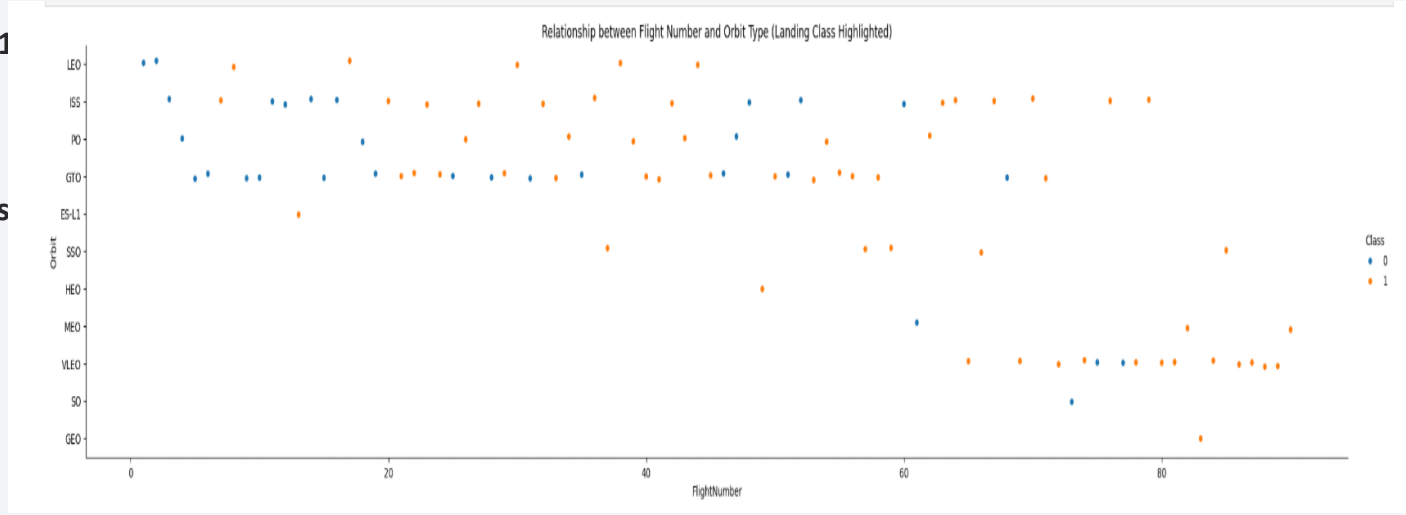
- Orbits like **SSO, GEO, ES-L1, and HEO** show a higher concentration of **Class 1 (orange dots)**, suggesting **greater reliability**.
- **GTO** shows a mix of outcomes, including several **Class 0 (blue dots)**, reinforcing its **lower success rate** seen in earlier charts.

3. **Later Flights Are More Successful**

- Across most orbit types, **higher flight numbers correlate with more successful landings**, reflecting **SpaceX's learning curve** and **engineering improvements**.

4. **Orbit Type Is a Key Predictor**

- The distribution of success and failure across orbit types suggests that **orbit type significantly influences landing outcome**.
- Likely due to differences in mission complexity, altitude, and reentry dynamics.



Payload vs. Orbit Type

1. **Certain Orbits Handle Heavier Payloads**

- Orbits like **GTO, GEO, and ES-L1** show flights with **payloads exceeding 10,000 kg**, indicating these missions are designed for **high-capacity deployments**.
- These orbits are typically more demanding, often requiring more powerful boosters and precise navigation.

2. **Success Rates Vary Across Orbits**

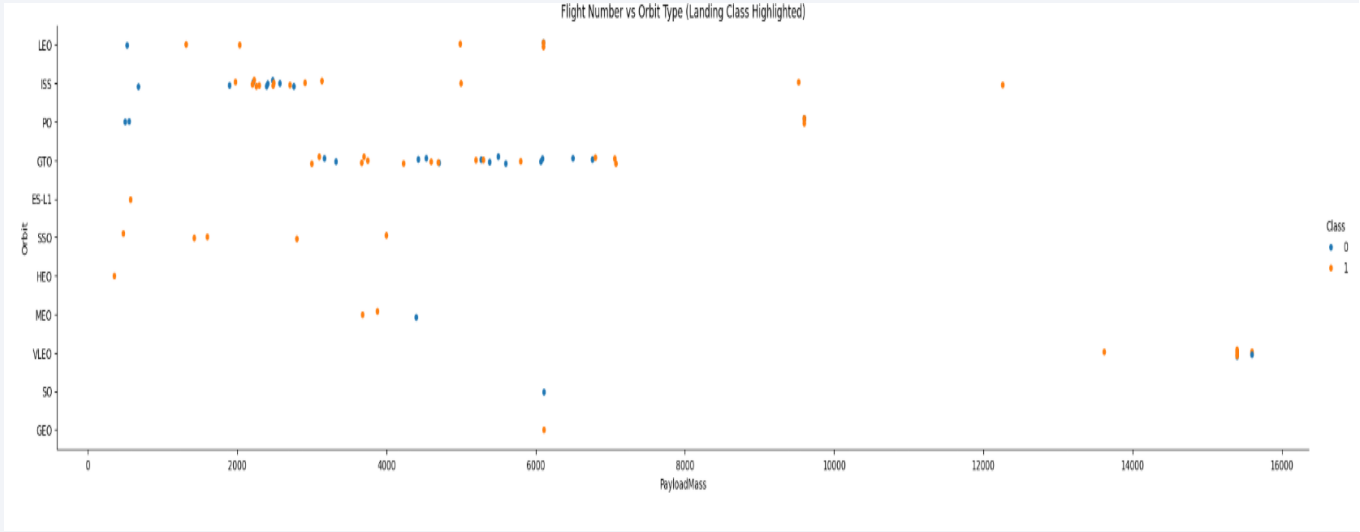
- **SSO, ES-L1, and GEO** show a higher concentration of **Class 1 (orange dots)**, suggesting **greater reliability** in these orbital missions.
- **GTO** again shows a mix of outcomes, reinforcing its **lower success rate** seen in previous charts.

3. **Payload Mass Alone Doesn't Predict Success**

- Across all orbit types, both **Class 0 and Class 1** outcomes occur at various payload levels.
- Indicates that **orbit type and mission profile** are more influential than payload mass alone in determining landing success.

4. **Orbit Type Influences Mission Complexity**

- Orbits like **LEO, ISS, and PO** tend to carry lighter payloads and show mixed success, possibly due to frequent use and varied mission goals.
- **VLEO and SO** have fewer data points, suggesting niche or experimental missions.



Launch Success Yearly Trend

1. **Consistent Improvement Over Time**

- The success rate shows a **clear upward trend**, especially after **2014**, indicating **technological refinement** and **operational maturity**.
- By **2019**, SpaceX achieved its **highest average success rate**, approaching **90%**, reflecting strong reliability.

2. **Early Years Were Experimental**

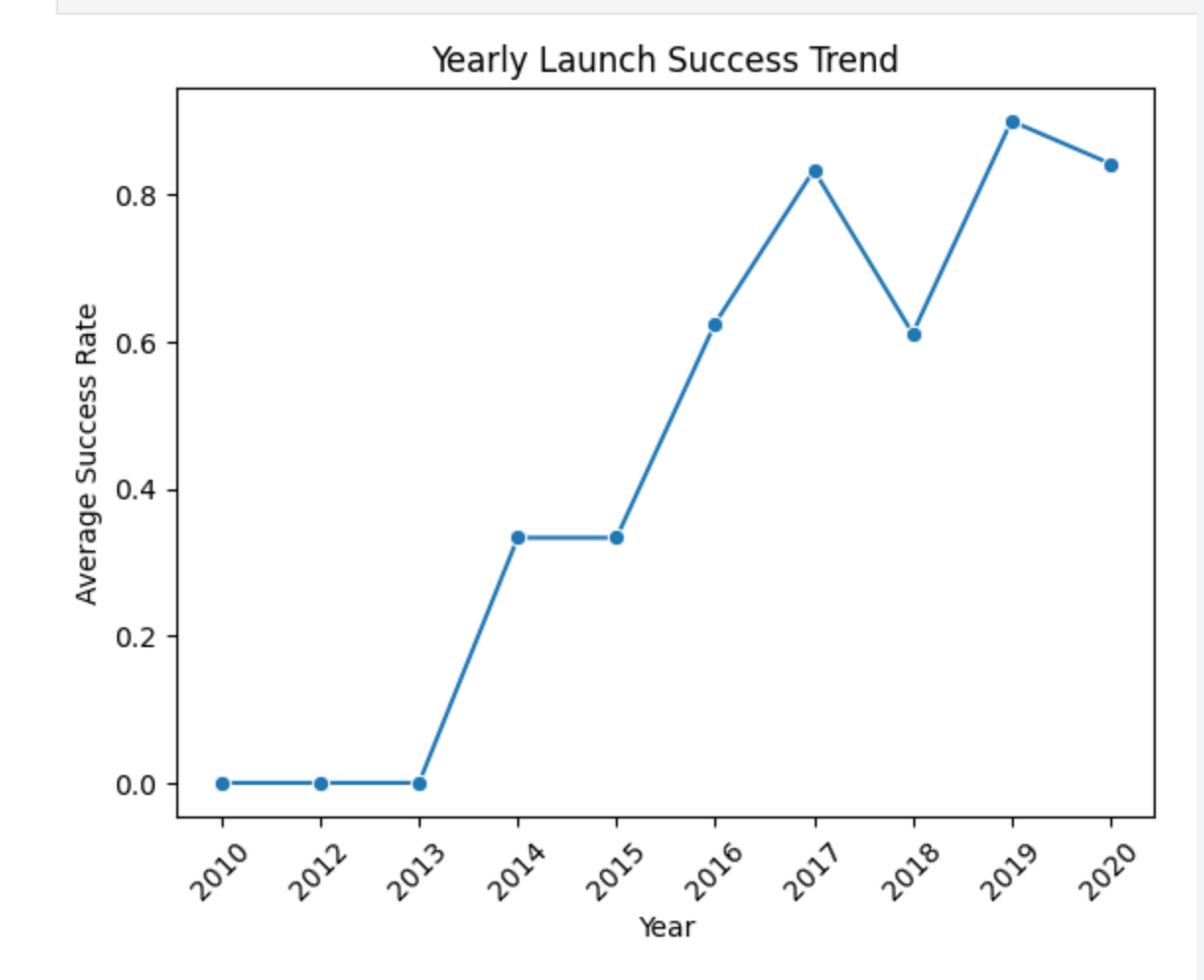
- Between **2010 and 2013**, success rates were lower and more volatile, suggesting a **trial-and-error phase** in SpaceX's development cycle.

3. **Learning Curve Evident**

- The steady rise in success rates supports the idea of a **learning curve**, where each launch contributes to better engineering, testing, and recovery strategies.

4. **Modeling Implication**

- **Flight Year or Flight Number** can be a meaningful feature in predictive modeling, as it correlates with **experience and reliability**.



All Launch Site Names

```
%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

- The query lists all **distinct launch sites** in the dataset, helping us analyze site-specific success rates and patterns

Launch Site Names Begin with 'CCA'

```
%sql 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
2010-06-04	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

- This query filters launches from **Cape Canaveral** using a wildcard match (LIKE "CCA%") and shows early Falcon 9 missions. It helps analyze site-specific performance and early-stage reliability.

Total Payload Mass

```
%sql select sum("PAYLOAD_MASS__KG_") as totalpayloadmass from SPACEXTABLE where "Customer" = 'NASA (CRS)'
```

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

totalpayloadmass
45596

- This query calculates the **total payload mass** launched by SpaceX for **NASA (CRS)** missions. It filters rows where the customer is 'NASA (CRS)' and sums the "PAYLOAD_MASS__KG_" column to get the cumulative payload carried by Falcon 9 boosters for NASA’s Commercial Resupply Services.

Average Payload Mass by F9 v1.1

```
%sql select avg("PAYLOAD_MASS__KG_") as avgpayloadmass from SPACEXTABLE where "Booster_Version" = 'F9 v1.1'
```

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

avgpayloadmass
2928.4

- This query calculates the **average payload mass** carried by the **F9 v1.1 booster version** by filtering relevant rows and applying the AVG() function on the "PAYLOAD_MASS__KG_" column. It helps assess the typical mission load for this booster type.

First Successful Ground Landing Date

```
%sql select min("Date") as FirstSuccessfulGroundPadLanding from SPACEXTABLE where "Landing_Outcome" = 'Success (ground pad)';
```

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

FirstSuccessfulGroundPadLanding
2015-12-22

- This query finds the **earliest date** when a Falcon 9 booster successfully landed on a **ground pad**, using the `MIN()` function to extract the first such event from the `"Landing_Outcome"` column.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select "Booster_Version" from SPACEXTABLE where "Landing_Outcome" = 'Success (drone ship)' and "PAYLOAD_MASS__KG_" between 4000 and 6000;
```

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

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

- This query filters launches where the **booster landed successfully on a drone ship** and carried a **payload between 4000 and 6000 kg**. It returns the names of boosters that achieved this — useful for analyzing performance under medium payload conditions.

Total Number of Successful and Failure Mission Outcomes

```
%sql select "Mission_Outcome", count(*) as totalmissions from SPACEXTABLE group by "Mission_Outcome";
```

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

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

- This query groups all missions by their outcome and counts how many fall into each category. It shows that **SpaceX has achieved a high success rate**, with **99 out of 101 missions** marked as successful or partially successful.

Boosters Carried Maximum Payload

- This query identifies all boosters that launched the **heaviest payload** recorded in the dataset by matching the maximum value in "PAYLOAD_MASS__KG_". All belong to the **F9 Block 5** series, known for high performance and reusability.

```
%sql select "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

```
%sql select substr("Date", 6, 2) as month, "Landing_Outcomes", "Booster_Version", "Launch_Site" from SPACEXTABLE where substr("Date", 0,5) and "Landing_Outcome" = 'Failure (drone ship)'
```

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

month	"Landing_Outcomes"	Booster_Version	Launch_Site
01	Landing_Outcomes	F9 v1.1 B1012	CCAFS LC-40
04	Landing_Outcomes	F9 v1.1 B1015	CCAFS LC-40
01	Landing_Outcomes	F9 v1.1 B1017	VAFB SLC-4E
03	Landing_Outcomes	F9 FT B1020	CCAFS LC-40
06	Landing_Outcomes	F9 FT B1024	CCAFS LC-40

- This query filters for **2015 launches** with **failed drone ship landings**, showing the **booster versions** and **launch sites** involved. It helps analyze early-stage recovery challenges and site-specific performance

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

```
%sql select "Landing_Outcome", count(*) as OutcomeCount from SPACEXTABLE where "Date" between '2010-06-04' and '2017-03-20' group by "Landing_Outcome" order by OutcomeCount desc;
```

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

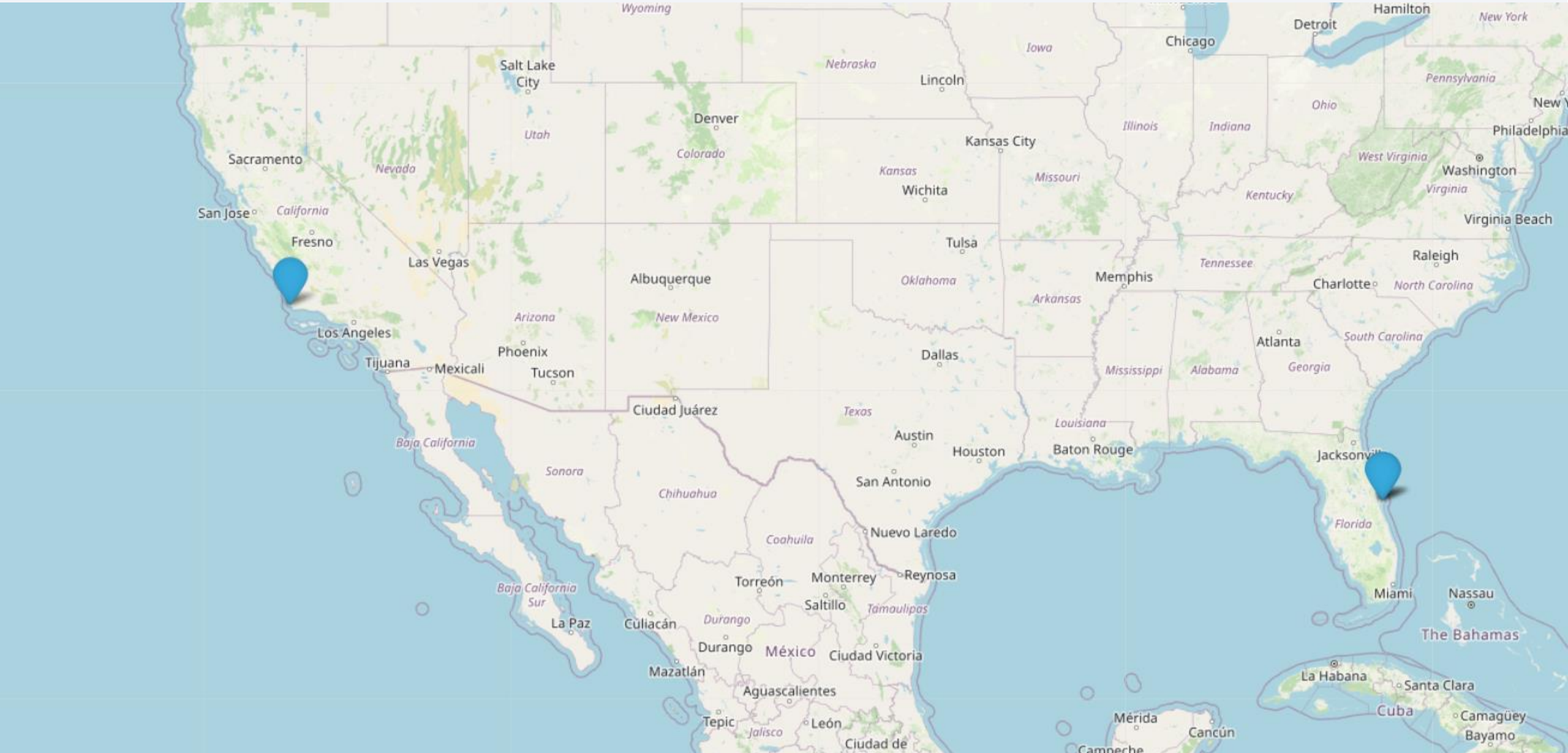
Landing_Outcome	OutcomeCount
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

- This query ranks all **landing outcomes** between June 2010 and March 2017 by their **frequency**, helping identify the most common recovery results during SpaceX's early development phase. "No attempt" was the most frequent, reflecting limited recovery capability in early missions.

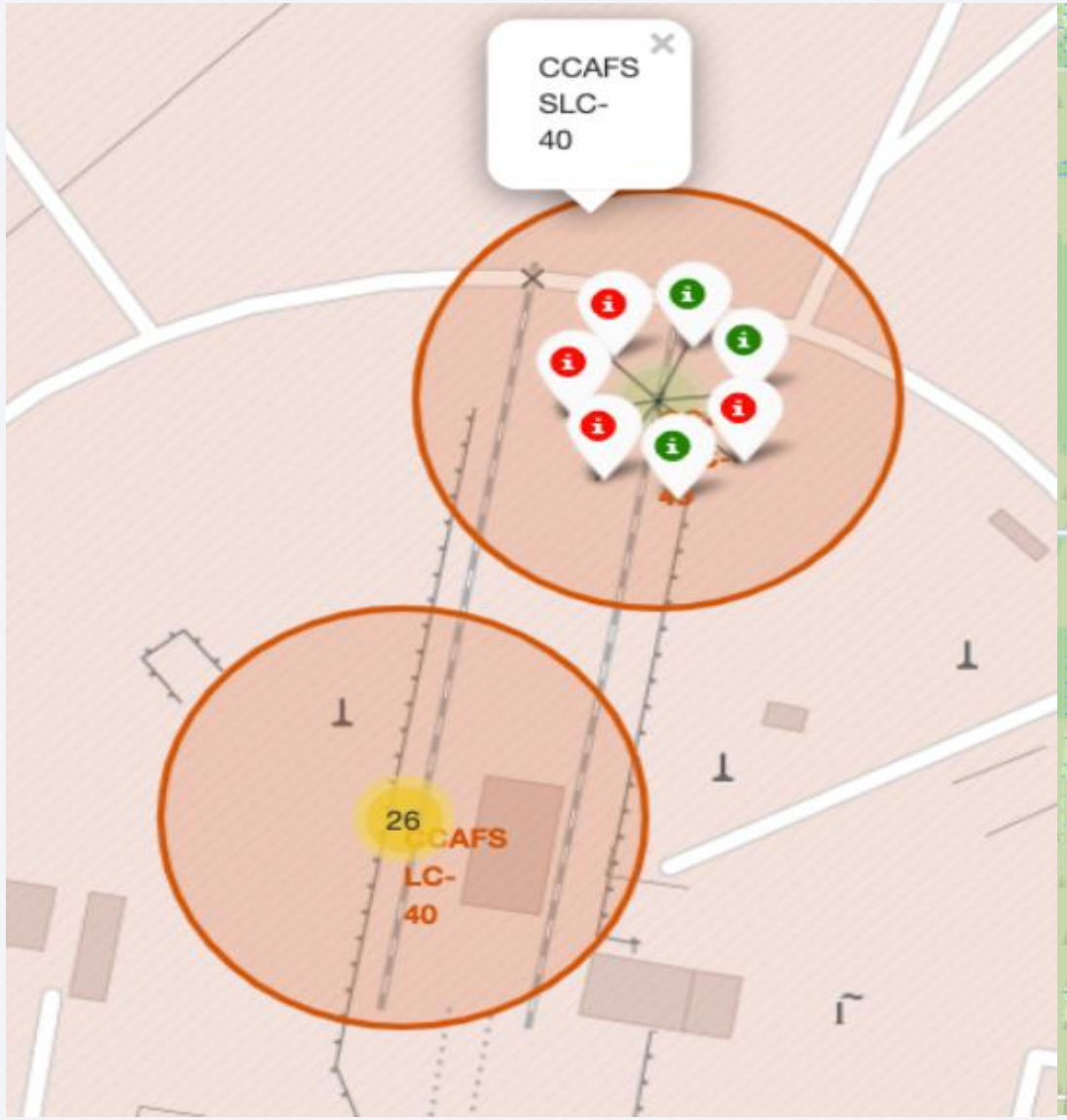
Section 3

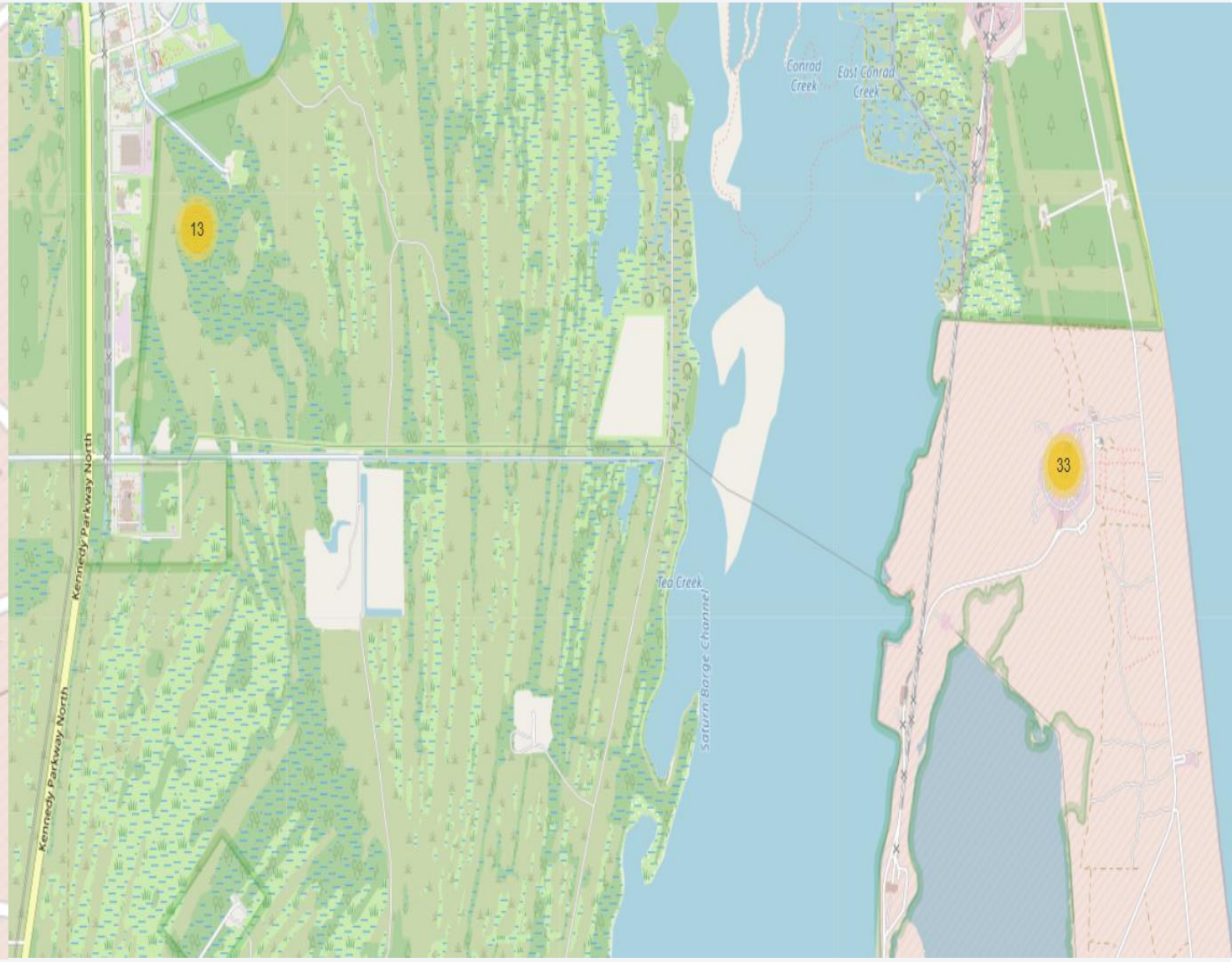
Launch Sites Proximities Analysis

All Launch Sites on Map

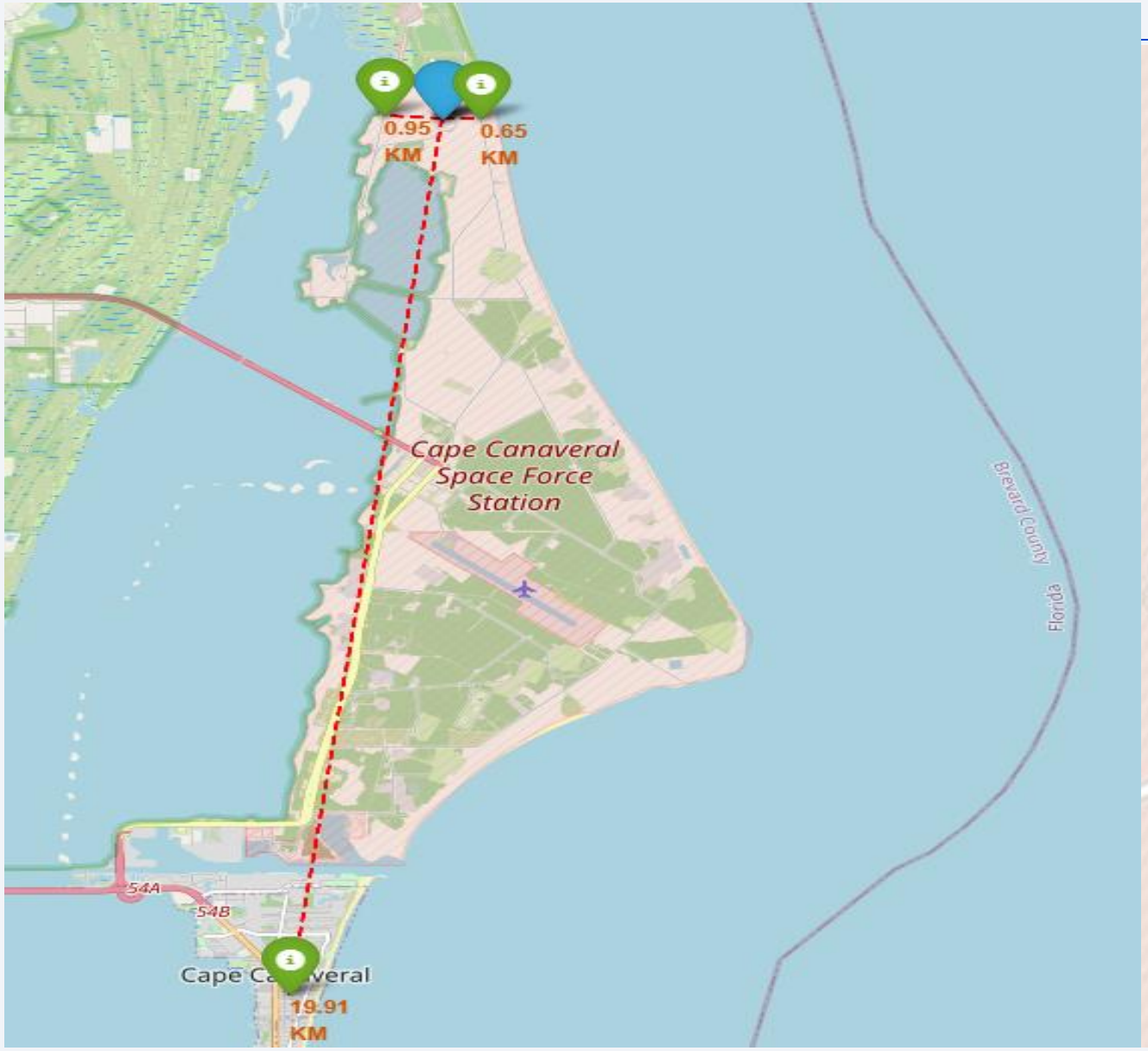


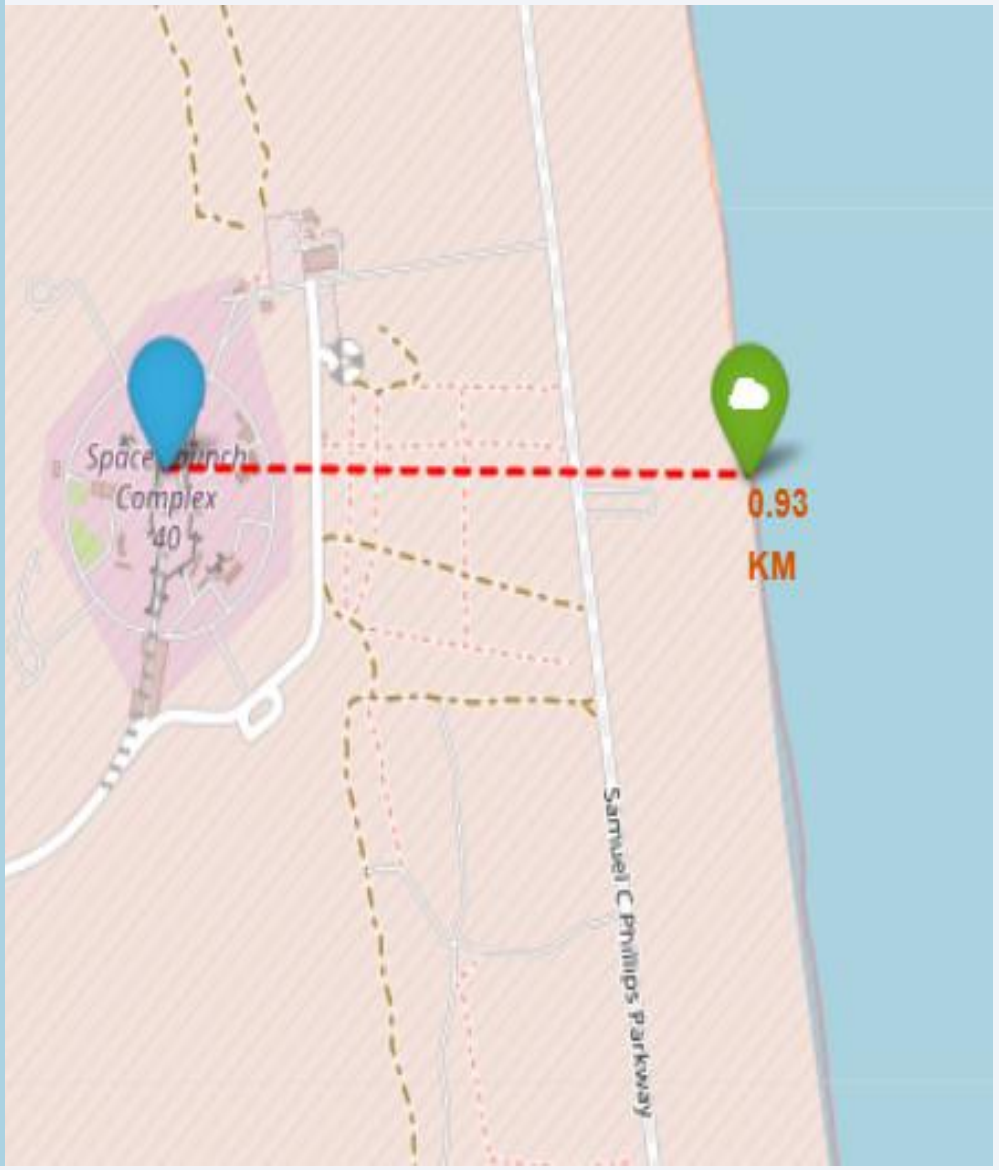
Finding the Launch Success Record

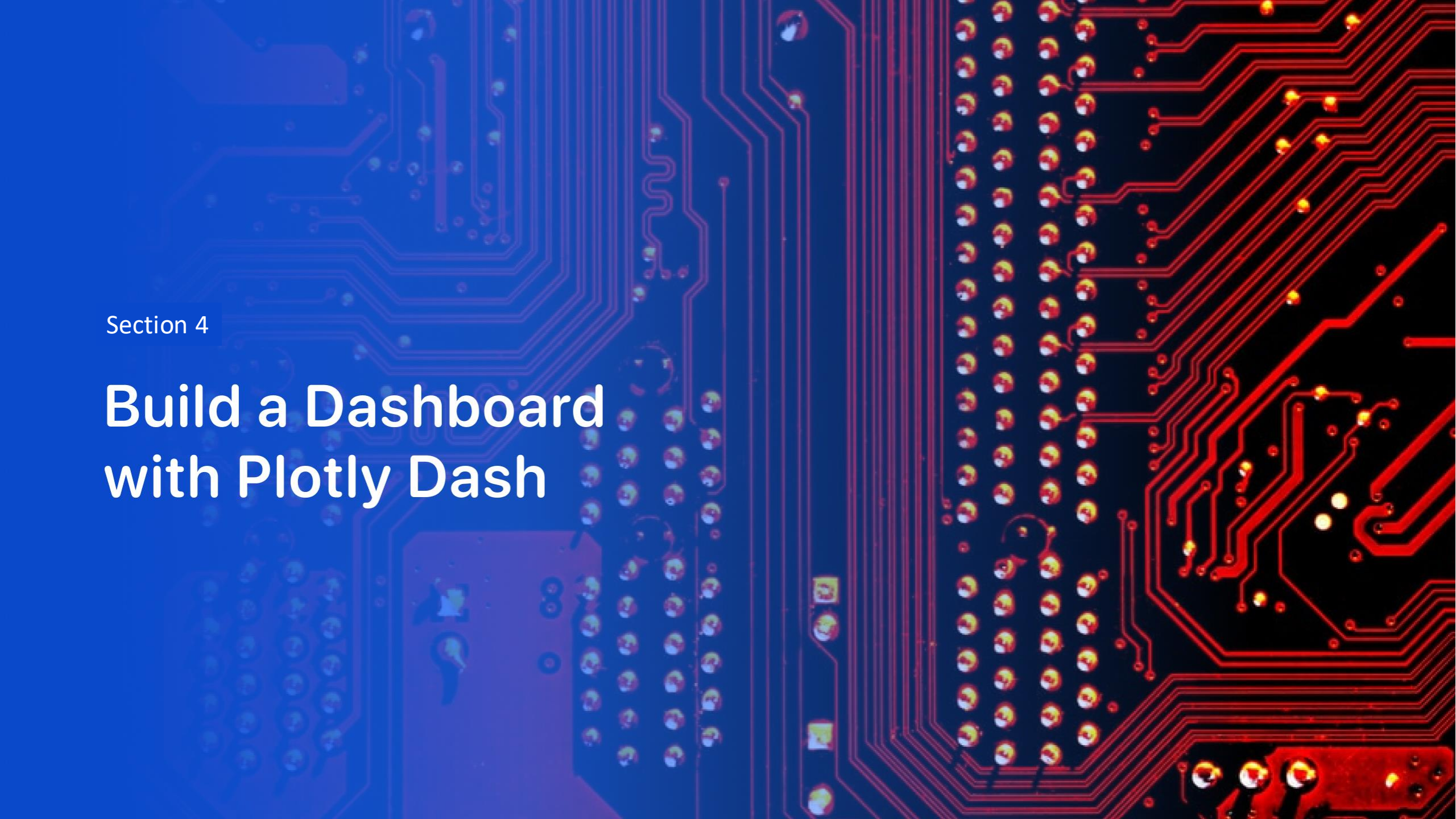




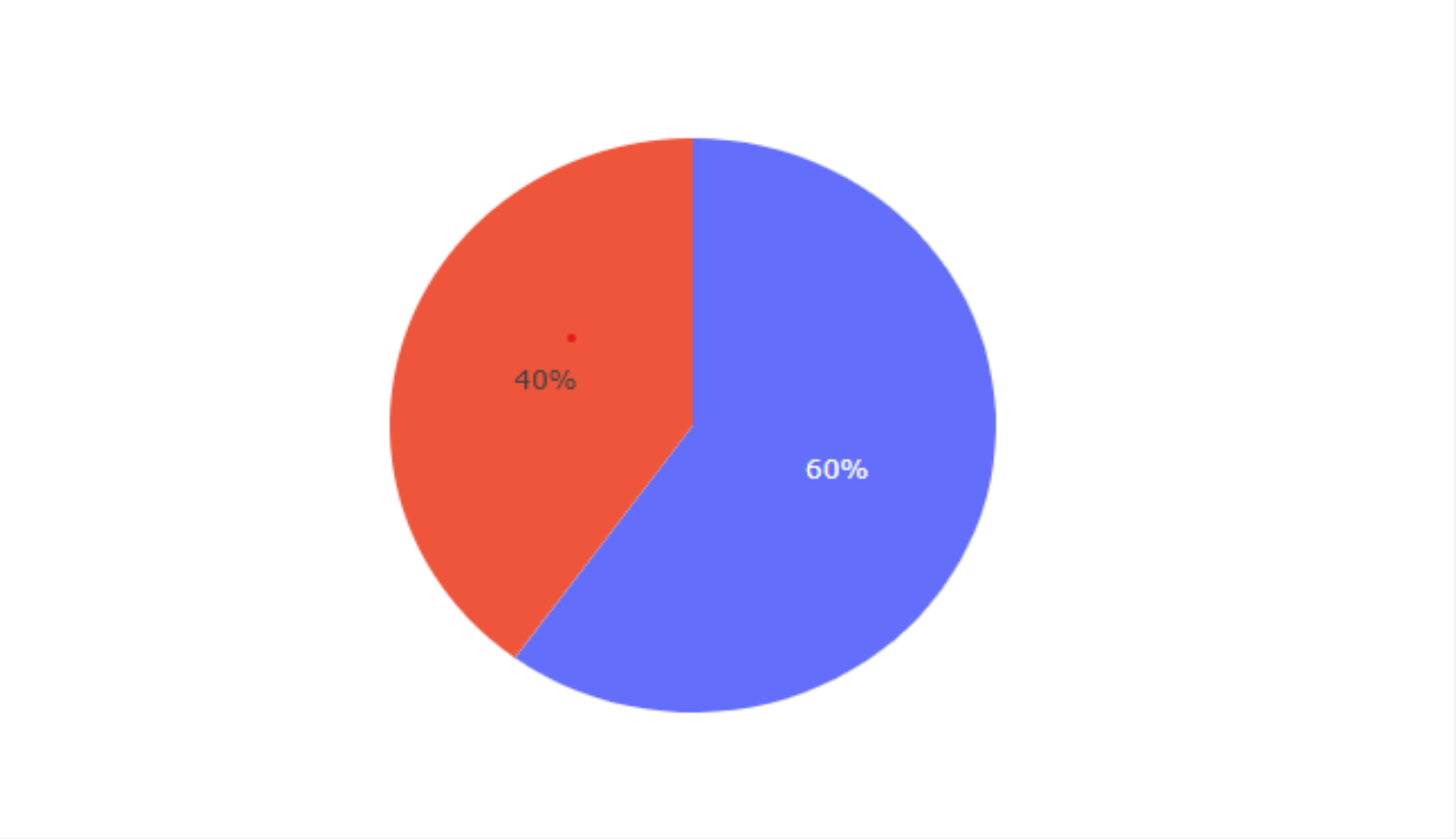
Distance between launch site and proximities





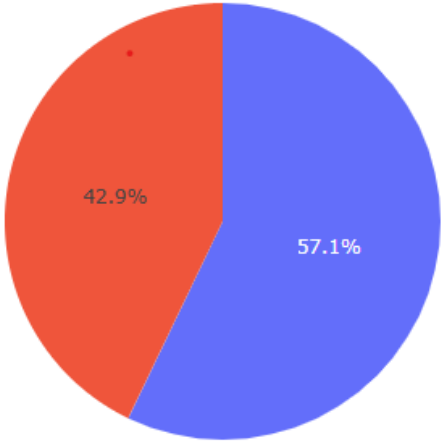

Section 4
Build a Dashboard with Plotly Dash

Pie Chart of All Sites

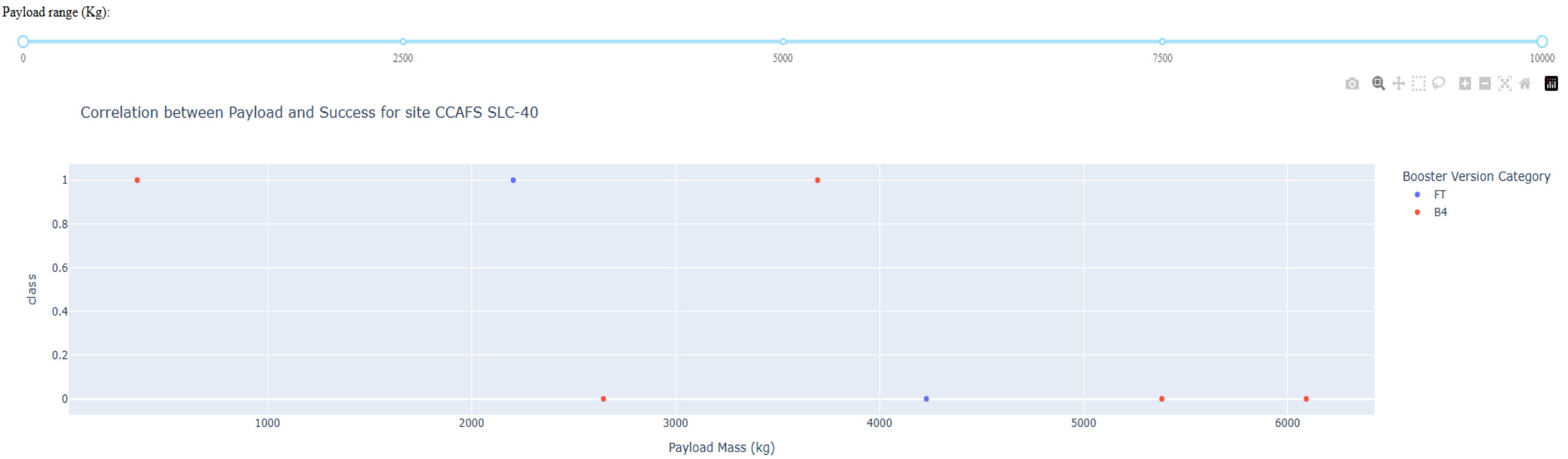


Pie Chart for specific site

Success vs Failure for CCAFS SLC-40



Coorelation between Payload and Success

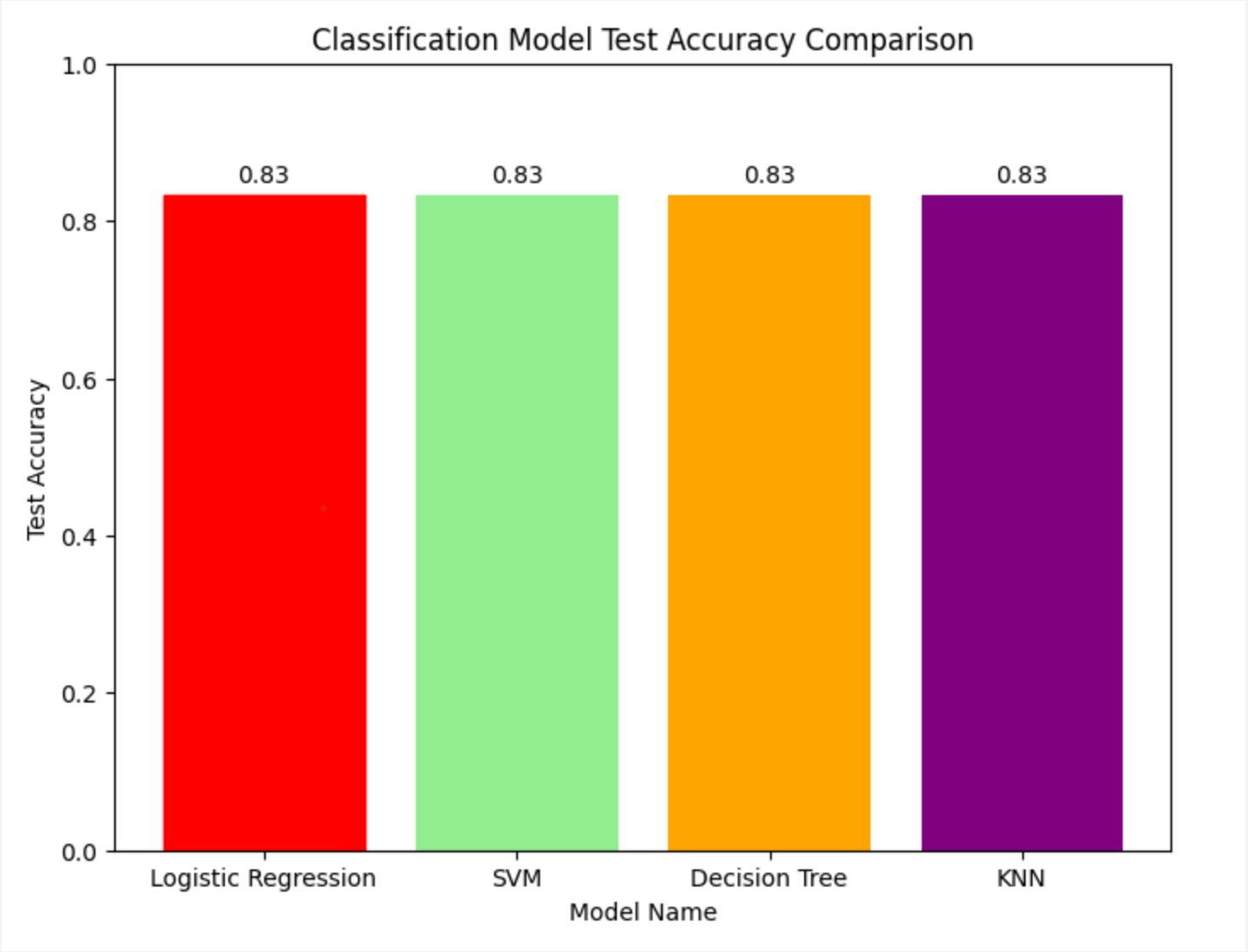


Section 5

Predictive Analysis (Classification)

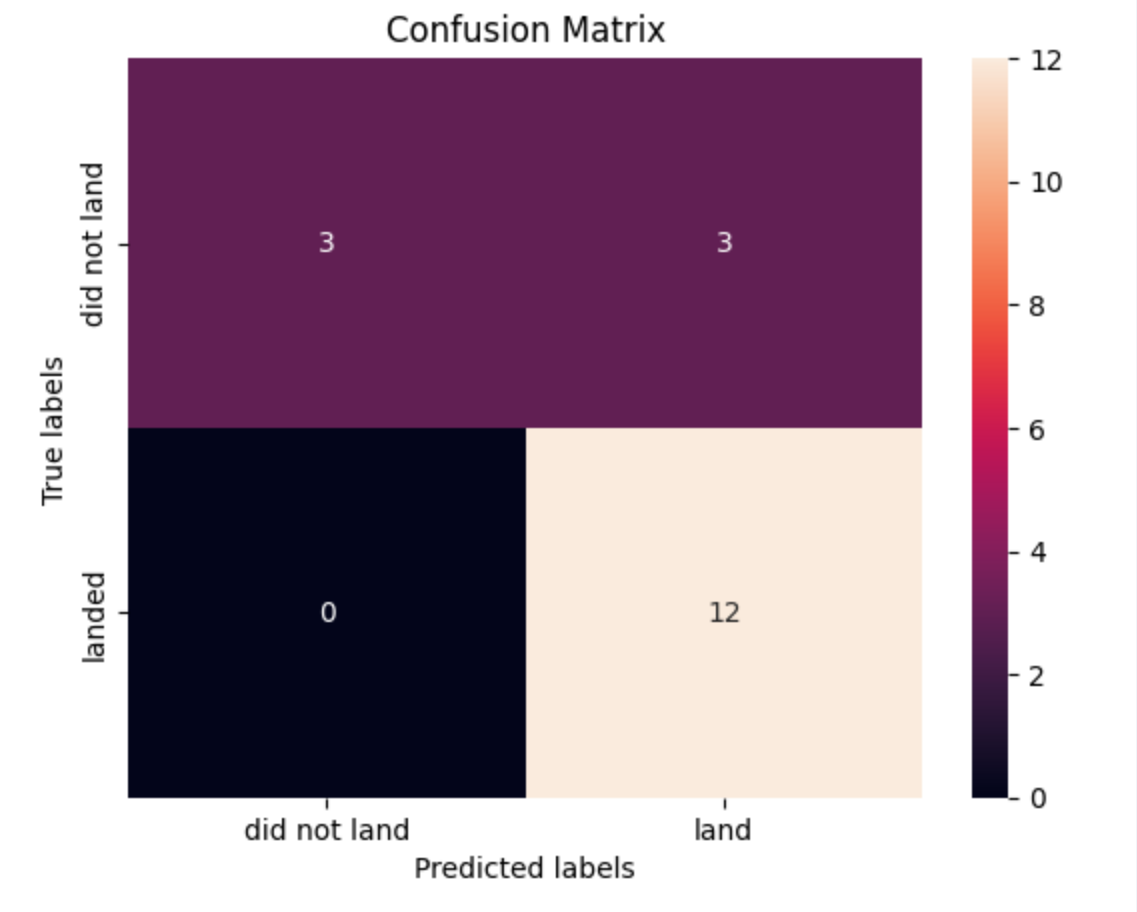
Classification Accuracy

- All seems equal but **Logistic Regression** has the highest accuracy by 0.01%



Confusion Matrix

- Confusion Matrix of Logistic Regression:



Conclusions

1. **All Models Perform Equally**

- Logistic Regression, SVM, Decision Tree, and KNN all achieved the **same test accuracy (~83.3%)**.
- This suggests that the dataset may not be complex enough to differentiate performance among these algorithms, or the features used provide similar predictive power across models.

2. **No Clear Superiority**

- Since all models reached identical accuracy, **no single algorithm clearly outperforms the others**.
- Logistic Regression is listed as the "best performing method," but in reality, it ties with the others.

3. **Dataset Characteristics**

- The uniform accuracy implies the dataset might be **balanced and relatively simple**, where linear and non-linear models converge to similar results.
- It may also suggest **limited feature diversity** or that the classification boundary is straightforward.

Appendix

- **SQL Queries:**
- %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
- %sql SELECT SUM("PAYLOAD_MASS__KG_") AS totalpayloadmass FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)';
- %sql SELECT AVG("PAYLOAD_MASS__KG_") AS avgpayloadmass FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1';
- %sql SELECT MIN("Date") AS FirstSuccessfulGroundPadLanding FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)';
- %sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" BETWEEN 4000 AND 6000;
- %sql SELECT "Mission_Outcome", COUNT(*) AS totalmissions FROM SPACEXTABLE GROUP BY "Mission_Outcome";
- **Dataset:**
- **SPACEXTABLE** (SQLite database `my_data1.db`) containing launch records with columns:
- Date, Time, Booster_Version, Launch_Site, Payload, Payload_Mass, Orbit, Customer, Mission_Outcome, Landing_Outcome.

Thank you!