IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Shivam Patel
07/23/2025

Outline

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

Executive Summary

- Summary of methodologies
- Data Collection via API, Web Scraping
- Exploratory Data Analysis (EDA) with Data Visualization
- EDA with SQL
- Interactive Map with Folium
- Predictive Analysis
- Summary of all results
- Exploratory Data Analysis results
- Interactive maps and dashboard
- Predictive results

Introduction

- SpaceX has upended launch pricing by re-flying Falcon 9 first stages, listing missions for $62 million while traditional providers charge $165 million or more. The economic hinge-point is whether a booster lands intact and is queued for reuse.
- This capstone uses publicly available launch manifests, weather feeds, and booster flight histories to forecast first-stage landing success. By anticipating the outcome before liftoff, mission planners can sharpen cost estimates, adjust risk models, and optimize pad scheduling.
- We will explore
  - How payload mass, launch site, prior flights, orbit, and weather drive landing outcomes
  - Whether landing success rates have risen over time
  - Which machine-learning algorithm delivers the most accurate binary-classification predictions for this task

Section 1

Methodology

Methodology

Executive Summary

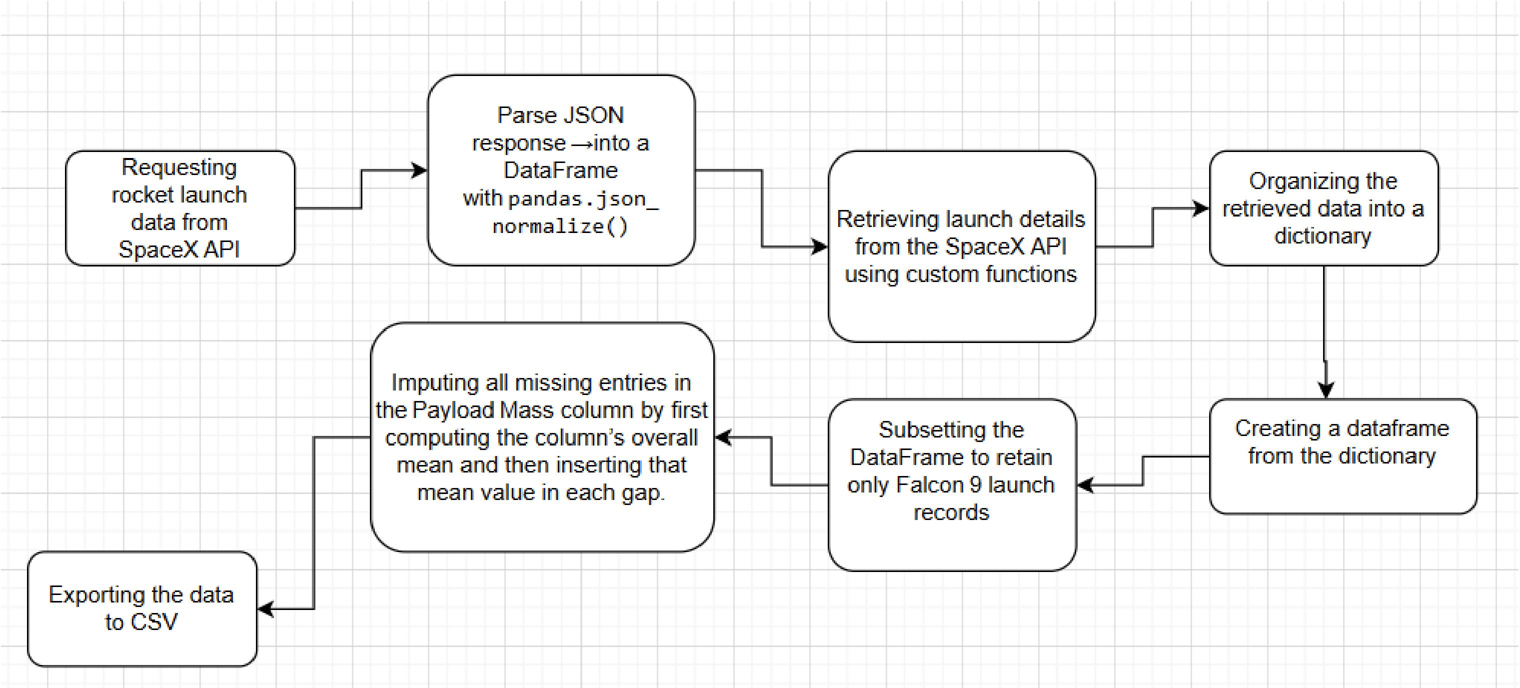
- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Building, tuning and evaluation of classification models to ensure the best results

Data Collection

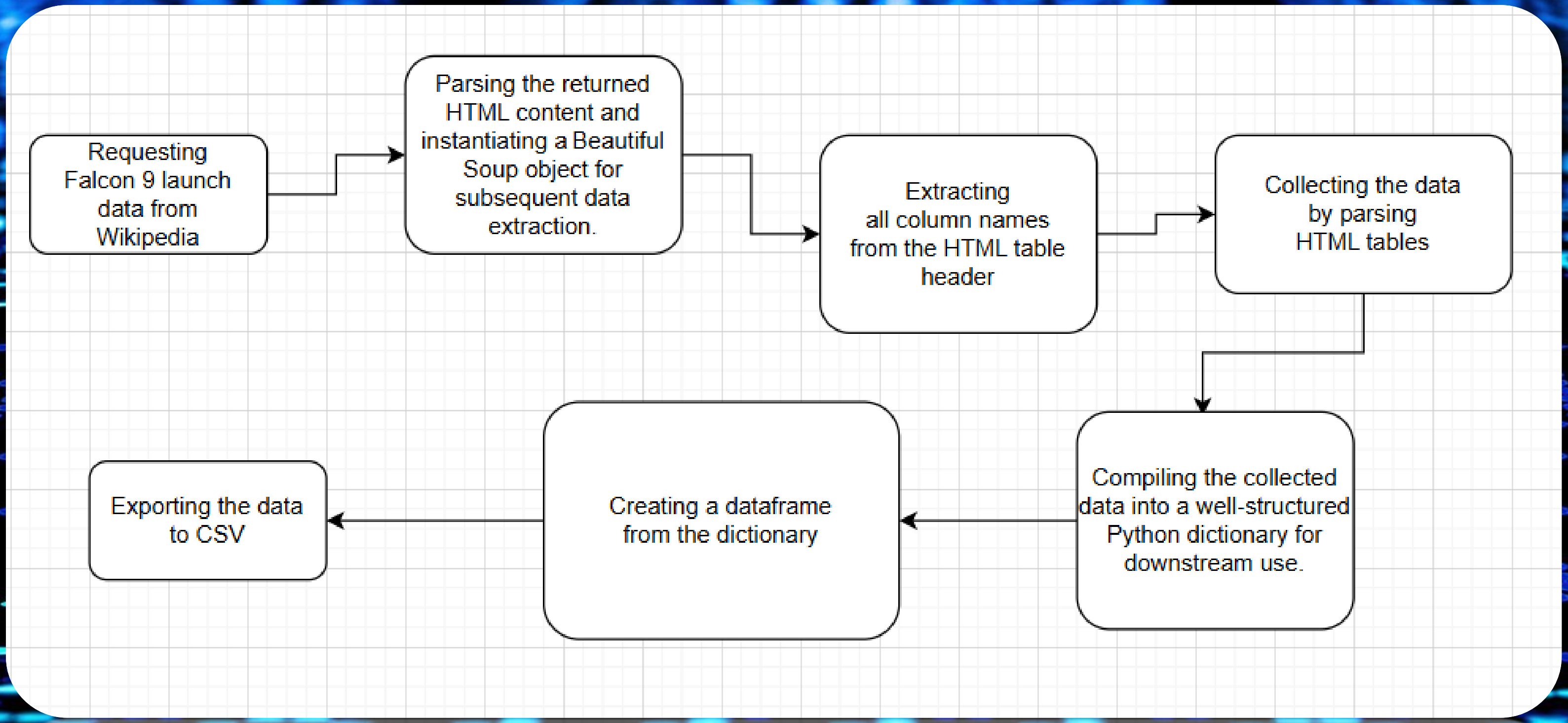
- Datasets are collected from Rest SpaceX API and webscrapping Wikipedia
- The information obtained by the API are rocket, launches, payload information.
- The Space X REST API URL is api.spacexdata.com/v4/
- The information obtained by the webscrapping of Wikipedia are launches, landing, payload information.
- URL is https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

https://github.com/spatel340/IBM_Data_Science_Capstone/blob/main/Data_Collection_webscraping.ipynb

Data Collection – SpaceX API



Data Collection - Scraping



Data Wrangling

- **Data Wrangling — Overview**
  - Prepares raw data for analysis by cleaning, transforming, and organizing it into an analysis-ready structure.
- **Step 1 — Data Cleaning**
  - Detect and resolve missing, duplicate, or inconsistent values.
  - Impute with appropriate methods (mean/median/KNN) or drop rows/columns with excessive gaps.
- **Step 2 — Data Transformation**
  - Cast fields to correct types (datetime, numeric, categorical).
  - Standardize text (lowercase, trim whitespace, unify labels).
  - Engineer features (e.g., extract *year* from a date).
  - Normalize/scale numerical variables to ensure comparability.

- https://github.com/spatel340/IBM_Data_Science_Capstone/blob/main/Data%20wrangling.ipynb

Data Wrangling

- **Step 3 — Data Integration**
  - Merge/union datasets from multiple sources (API, web scraping, CSV) into a single, coherent table.
  - Harmonize schemas: standardize column names, data types, units, and time zones; align keys (e.g., flight_id, launch_date, site).
  - Resolve overlaps/conflicts with clear precedence rules and keep source lineage (add source/ingest_timestamp columns).
- **Step 4 — Data Validation**
  - De-duplicate using unique keys and near-duplicate checks; remove or consolidate repeats.
  - Run type, range, and domain checks (e.g., non-negative payload mass, valid orbit labels, chronological dates).
  - Perform cross-field consistency checks (e.g., site ↔ coordinates, rocket ↔ payload class) and spot-verify against authoritative references.
  - Log validation results and flag anomalies for review.

EDA with Data Visualization

Exploratory Data Analysis (EDA) helps me **understand distributions, spot patterns, and uncover relationships** before modeling. I summarize key variables visually to guide feature engineering and model choice.

- **Histograms**

Use for: distributions of payload mass, flight count, success rate.

Reveals: spread, central tendency, outliers, skew.

- **Bar Charts**

Use for: comparing launch outcomes across **sites**, **rocket types**, or **orbits**.

Reveals: frequency/proportion differences and categorical patterns.

- **Line Charts**

Use for: trends over time (e.g., annual Falcon 9 success rate).

Reveals: temporal changes, seasonality, learning-curve effects.

EDA with Data Visualization

- **Scatter Plots**

Use for: relationships between two numeric variables (e.g., payload mass vs. landing success probability).

Reveals: correlation, clusters, nonlinearity, heteroscedasticity.

- **Heatmaps (Correlation Matrix)**

Use for: pairwise correlations among numeric features.

Reveals: strong positive/negative links, **multicollinearity** for feature selection.

- **Box Plots**

Use for: comparing numeric distributions across categories (e.g., payload mass by launch site).

Reveals: median, IQR, skew, and outliers at a glance.

- https://github.com/spatel340/IBM_Data_Science_Capstone/blob/main/eda_visulaization.ipynb

EDA with SQL

- **EDA with SQL (SQLite) — Key Tasks & Queries**
- **What I analyzed**
- **Counts & rates:** total launches; outcomes by category/site.
- **Distincts & filters:** unique launch sites; site-specific samples.
- https://github.com/spatel340/IBM_Data_Science_Capstone/blob/main/eda_sqllite.ipynb

```
%sql sqlite:///my_data1.db
%sql DROP TABLE IF EXISTS SPACEXTABLE;
%sql CREATE TABLE SPACEXTABLE AS SELECT * FROM SPACEXTBL
WHERE Date IS NOT NULL;

-- Distinct sites
%sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL;

-- Site-specific sample
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE
'%CCA%' LIMIT 5;

-- Outcome counts
%sql SELECT Mission_Outcome, COUNT(*) AS Total
    FROM SPACEXTBL GROUP BY Mission_Outcome;
```

EDA with SQL

- **Date slices:** month/year views and fixed windows.
- **Subqueries:** heaviest-payload booster; derived metrics.
- **Working table:** cleaned subset with non-null dates.

```
-- Heaviest payload booster (subquery)
%sql SELECT BOOSTER_VERSION
    FROM SPACEXTBL
    WHERE PAYLOAD_MASS__KG_ = (SELECT
MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);

-- Drone-ship failures by month in 2015
%sql SELECT substr(Date, 6, 2) AS Month, Landing_Outcome,
BOOSTER_VERSION
    FROM SPACEXTBL
    WHERE Landing_Outcome LIKE '%Failure (drone ship)%'
      AND substr(Date, 0, 5) = '2015';

-- Outcomes within a fixed window (2017-01-01 → 2017-03-20)
%sql SELECT Landing_Outcome, COUNT(*) AS Outcome_Count
    FROM SPACEXTBL
    WHERE Date BETWEEN '2017-01-01' AND '2017-03-20'
    GROUP BY Landing_Outcome
    ORDER BY Outcome_Count DESC;
```

Build a Dashboard with Plotly Dash

- Plots & Interactive Controls
- Success Breakdown (Pie Chart)
  - Visualizes the share of successful vs. failed launches.
  - Quick read on overall performance and class imbalance.
- Payload vs. Success (Scatter Plot)
  - Plots payload mass against landing/mission outcome.
  - Reveals thresholds, clusters, and the payload-success relationship.
- https://github.com/spatel340/IBM_Data_Science_Capstone/blob/main/spacex_dash_app.py

Interactions Added

Launch Site Dropdown

Select a specific launch site to filter all visuals.

Supports focused, site-level exploration.

Payload Range Slider

Dynamically adjust payload mass bounds.

Tests how success patterns change across mass ranges.

Build a Dashboard with Plotly Dash

- Success Pie Chart — Why
- Quickly shows class balance, guiding metric choice (e.g., AUC vs. accuracy) and any resampling needs.
- Establishes the baseline success rate for stakeholders before deeper analysis.
- Success–Payload Scatter Plot — Why
- Tests the hypothesis that heavier payloads affect success, revealing nonlinearity, clusters, or thresholds.
- Informs feature engineering (e.g., payload bins, interactions) and model selection.

- Launch Site Dropdown — Why
- Enables site-specific diagnostics without duplicating charts, improving focus and clarity.
- Speeds exploration by filtering the dataset on demand, enhancing usability in demos.
- Payload Range Slider — Why
- Supports what-if analysis by letting users probe success across payload thresholds.
- Surfaces local patterns (sweet spots or risk zones) that can guide operational decisions.

Predictive Analysis (Classification)

1) Data Preprocessing

•Standardized features (e.g., scaler on numeric columns).

•Split into **train/test** for validation.

2) Model Selection

•Compared **SVM**, **Decision Tree**, and **K-Nearest Neighbors**.

3) Hyperparameter Tuning

•Ran **GridSearchCV** on key parameters:

- •SVM: C, gamma
- •Tree: max_depth, min_samples_split
- •KNN: n_neighbors, weights

4) Model Evaluation

Cross-validation across folds.

Metrics: **accuracy, precision, recall, F1**.

5) Improvement Iterations

Reviewed confusion matrices & learning curves.

Adjusted features and hyperparameters.

6) Select Best-Performing Model

Chose top **test-set** performer with a small train–test gap.

Confirmed it meets project KPIs.

Predictive Analysis (Classification)

- **Data Preprocessing — Why**
- Avoids scale-driven bias (especially for **SVM/KNN**).
- Ensures an **unbiased generalization** estimate via hold-out testing.
- **Model Selection — Why**
- Compares complementary inductive biases (**margin**, **rules**, **instances**).
- Establishes strong baselines before considering ensembles.
- **Hyperparameter Tuning — Why**
- Cross-validated search reduces variance and **overfitting to a single split**.
- Controls model **capacity** to balance bias–variance.

Model Evaluation — Why
CV yields **stable** performance estimates.
Multiple metrics protect against **class imbalance** and costly errors.
Improvement Iterations — Why
Targets specific **failure modes** (e.g., low recall) with focused fixes.
Converges on a **parsimonious** solution (no unnecessary complexity).
Best Model Selection — Why
Prioritizes **real-world generalization**, not just training fit.
Aligns with stakeholder KPIs (e.g., accuracy; optionally **failure recall**).

Results

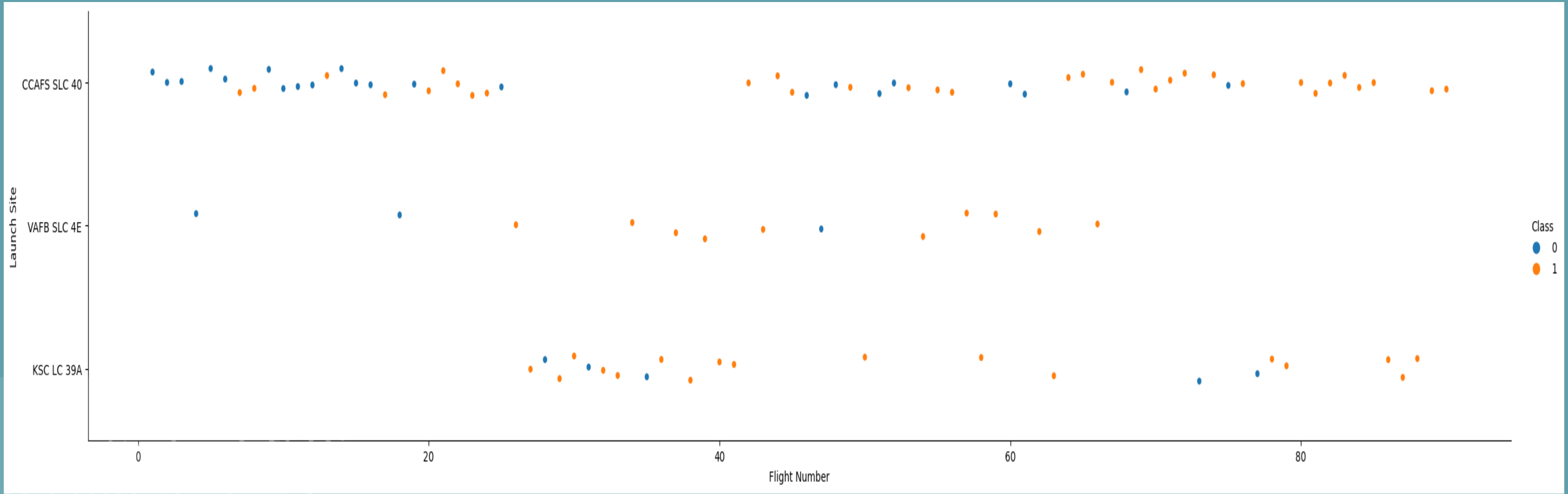
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

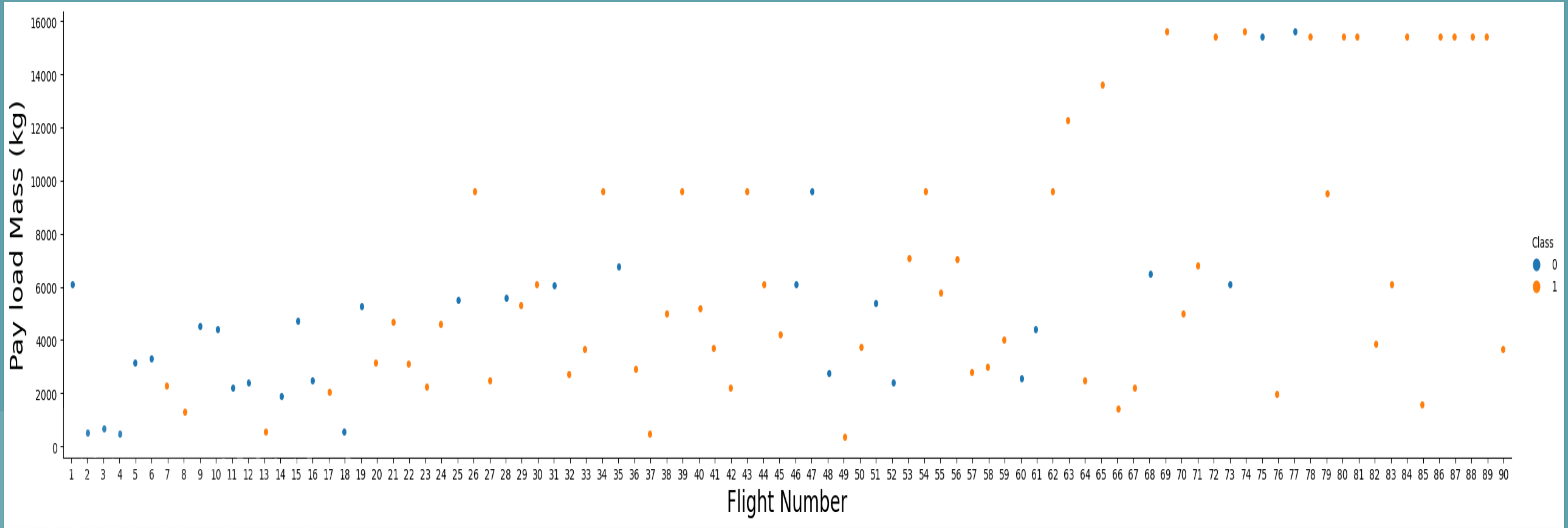
Flight Number vs. Launch Site

- Launch-Site Outcomes
- CCAFS (SLC-40) and KSC (LC-39A) both show a mix of successful and unsuccessful landings (orange/blue), suggesting the site itself isn't the primary driver—other factors (payload, weather, booster heritage) likely influence results.
- Activity by Flight Number
- Launches span a wide range of flight numbers at all sites, with no clear upward or downward trend in landing success as flight count changes.



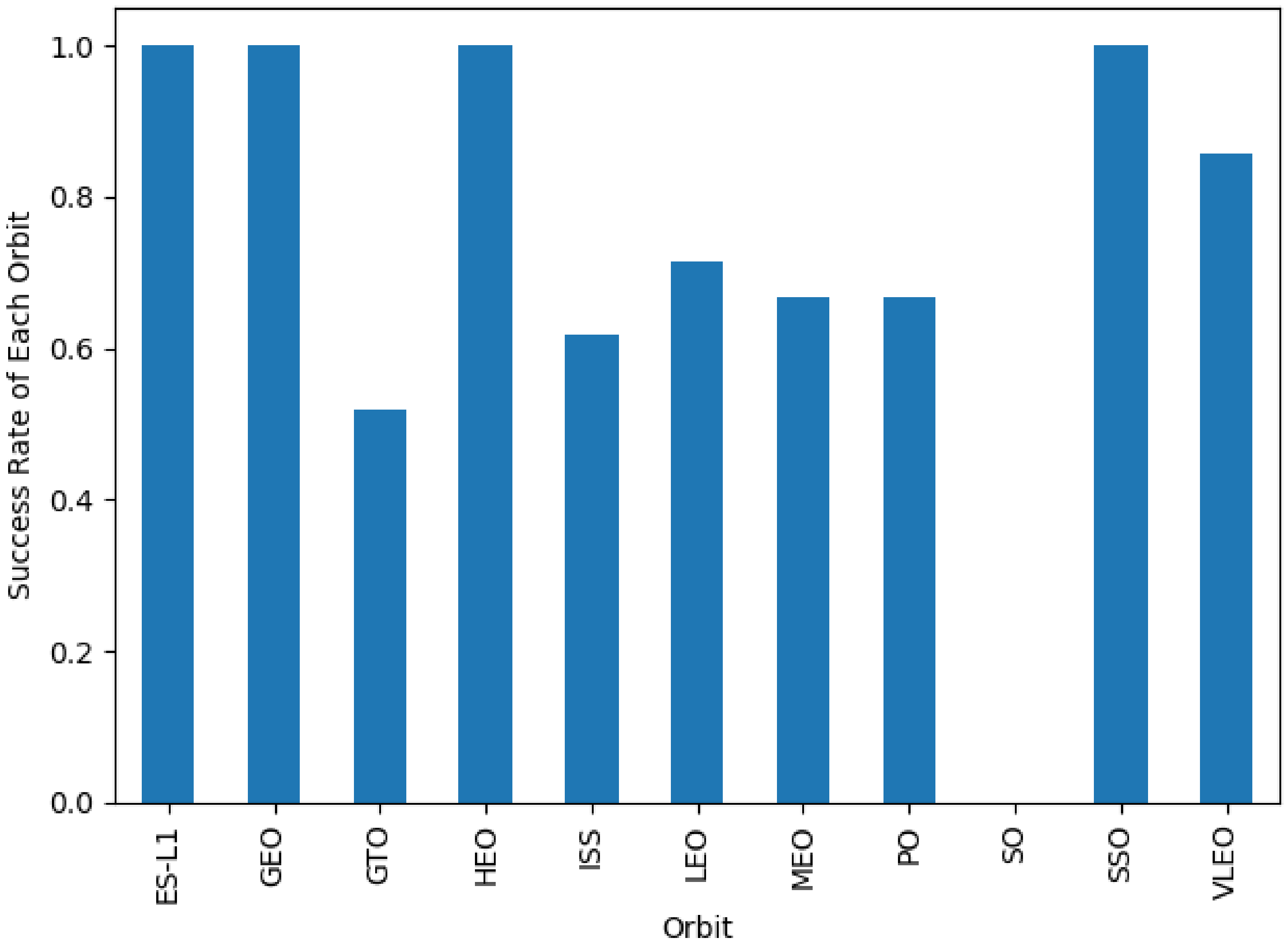
Payload vs. Launch Site

- Payload Distribution
- At CCAFS • SLC-40, most missions carry < 10,000 kg payloads.
- VAFB • SLC-4E and KSC • LC-39A show a broader payload range, reflecting more varied mission profiles.
- High-Capacity Launches
- KSC • LC-39A frequently supports > 15,000 kg payloads.
- Indicates LC-39A's suitability for heavy-lift / high-capacity missions.



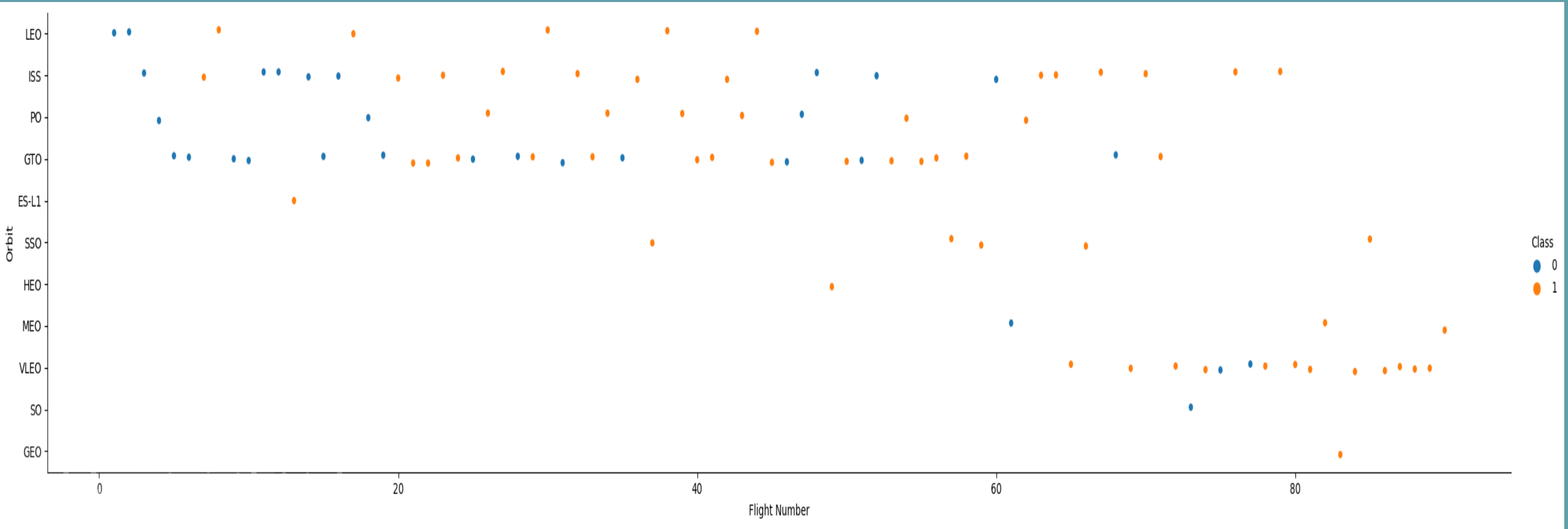
Success Rate vs. Orbit Type

- Orbit Outcomes — High Success Rates
- VLEO, ES-L1, GEO, HEO, and SSO missions show a 100% landing success in this dataset.
- Suggests these profiles are consistently favorable for first-stage recovery.
- Lower Success for GTO
- GTO exhibits a notably lower success rate than other orbits.
- Likely reflects higher energy demands/trajectory complexity, making recovery more challenging.



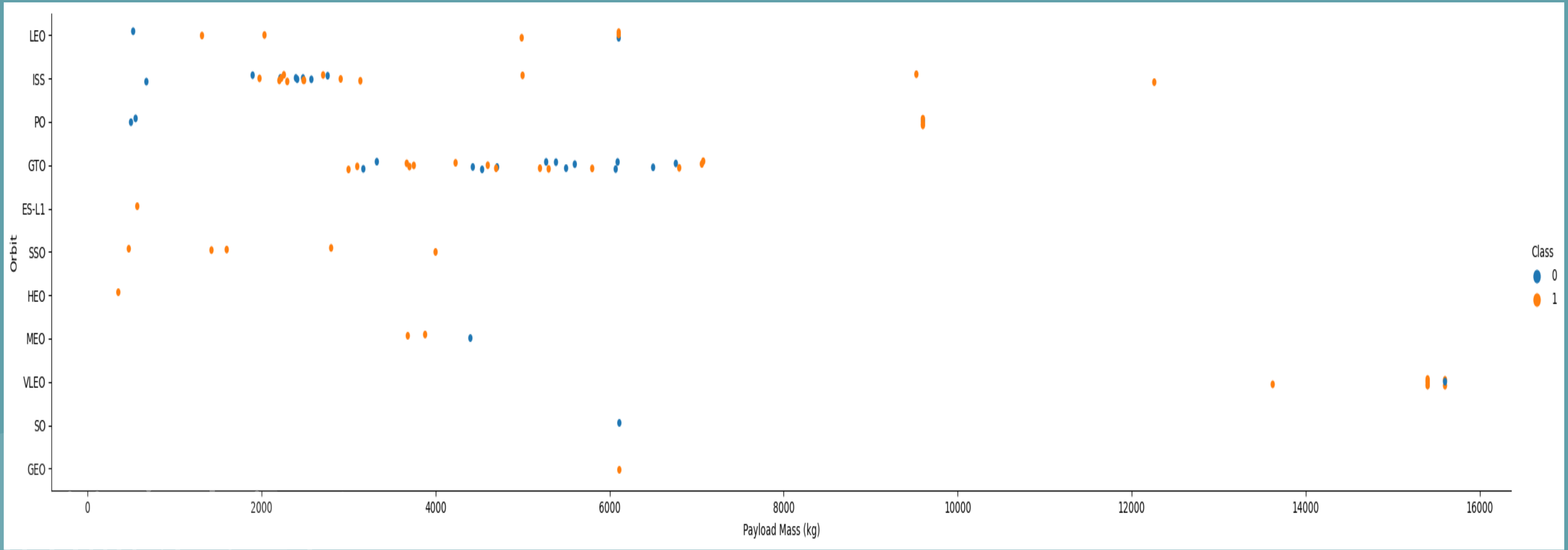
Flight Number vs. Orbit Type

- Trend Over Time
- Rising success with higher flight numbers, consistent with learning effects and iterative upgrades.
- Reflects maturing hardware, procedures, and recovery operations.
- Orbit-Specific Performance
- Early GTO and ISS missions showed mixed outcomes.
- Recent flights to these orbits show higher success, indicating better planning, trajectory tuning, and execution.



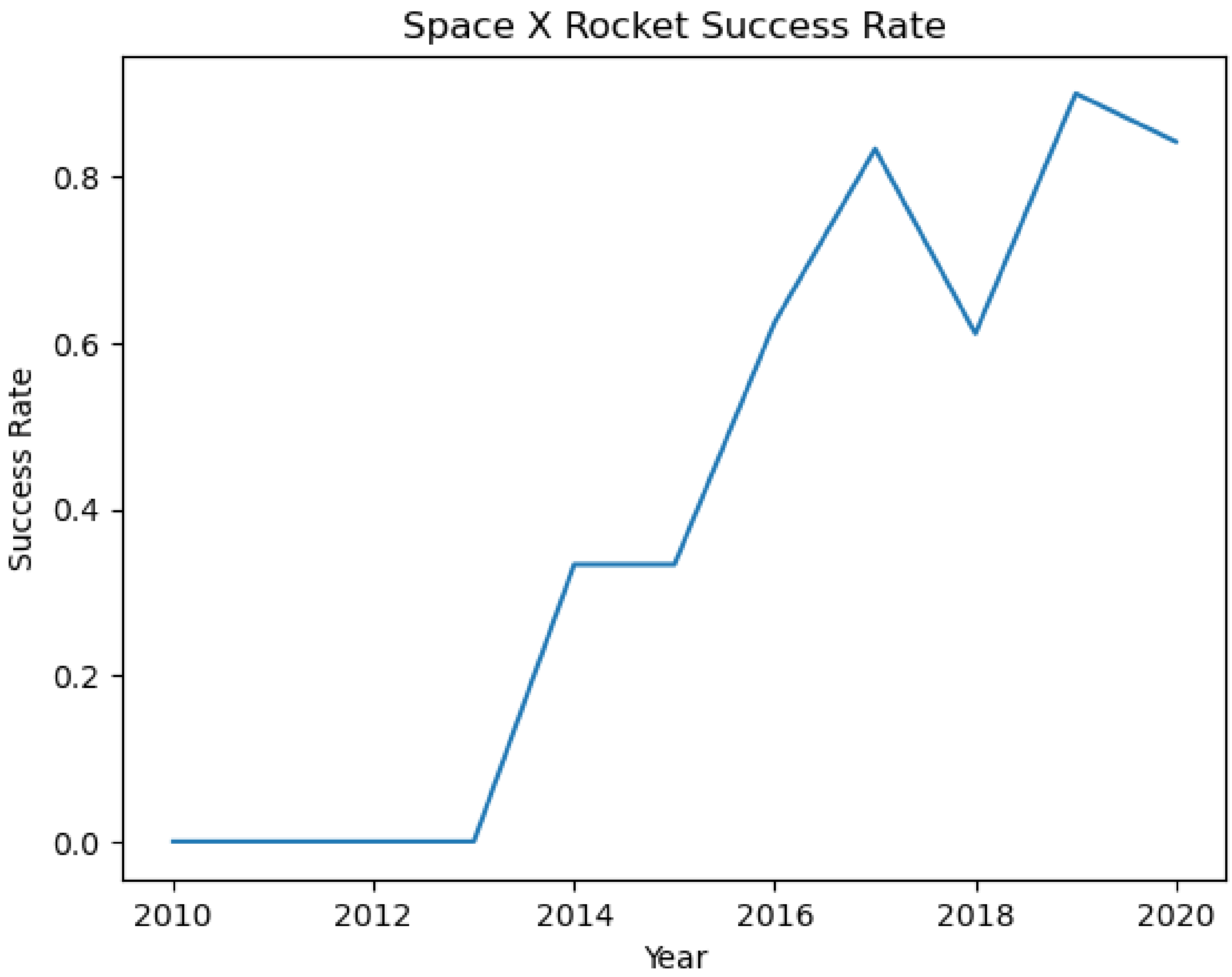
Payload vs. Orbit Type

- Payload vs. Landing Outcome
- Higher success at lower masses: Across all orbit types, landings are most reliable for payloads < 6,000 kg.
- Mixed results at heavy lift: Payloads > 10,000 kg show a blend of successes and failures, indicating tighter energy margins and greater recovery complexity.



Launch Success Yearly Trend

- Annual Success Trend
- Steady climb since 2013, surpassing 80% by 2020.
- A temporary dip in 2018 aside, the long-term trajectory shows increasing reliability in Falcon 9 launche



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

```
%sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

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

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

All Launch Site Names

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

```
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE '%CCA%' LIMIT 5
```

[11]

Python

```
 * 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

Launch Site Names Begin with 'CCA'

Total Payload Mass

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

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

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

SUM(PAYLOAD_MASS__KG_)
45596

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

```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "BOOSTER_VERSION" LIKE '%F9 v1.1%'
```

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

AVG(PAYLOAD_MASS__KG_)
2534.6666666666665

Average Payload Mass by F9 v1.1

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

Hint:Use min function

```
%sql SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing_Outcome" = 'Success (ground pad)'
```

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

MIN(DATE)
2015-12-22

First Successful Ground Landing Date

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

```
%sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL 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

Successful Drone Ship Landing with Payload between 4000 and 6000

List the total number of successful and failure mission outcomes

```
%sql SELECT Mission_Outcome, COUNT(*) AS Total FROM SPACEXTBL GROUP BY Mission_Outcome;
```

* sqlite:///my_data1.db
Done.

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

Total Number of Successful and Failure Mission Outcomes

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

```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

```
 * 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

Boosters Carried Maximum Payload

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.

```
%sql SELECT substr(Date, 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL WHERE Landing_Outcome LIKE '%Failure (drone ship)%' AND substr(Date, 0, 5) = '2
```

4]

Py

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

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

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.

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

Python

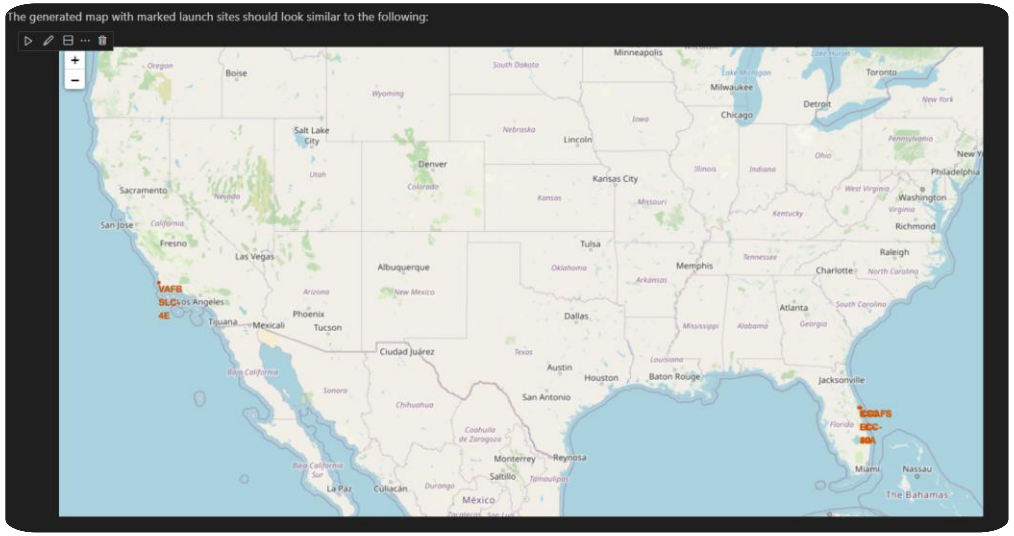
```
 * sqlite:///my_data1.db
Done.
```

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

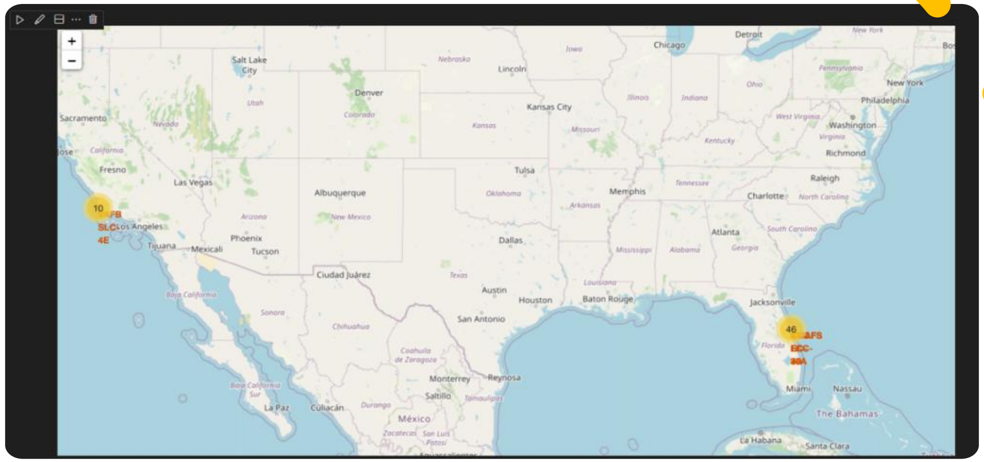
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

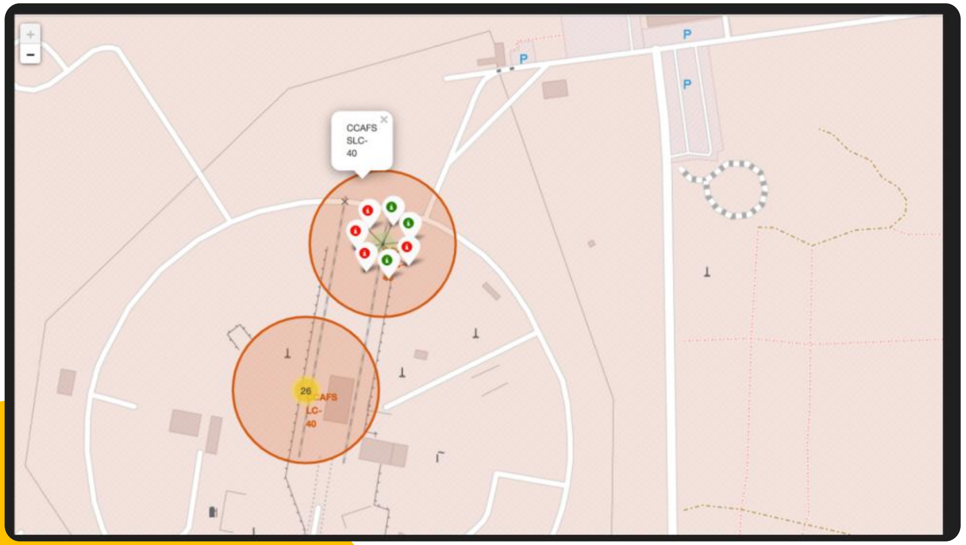
Section 3

Launch Sites Proximities Analysis



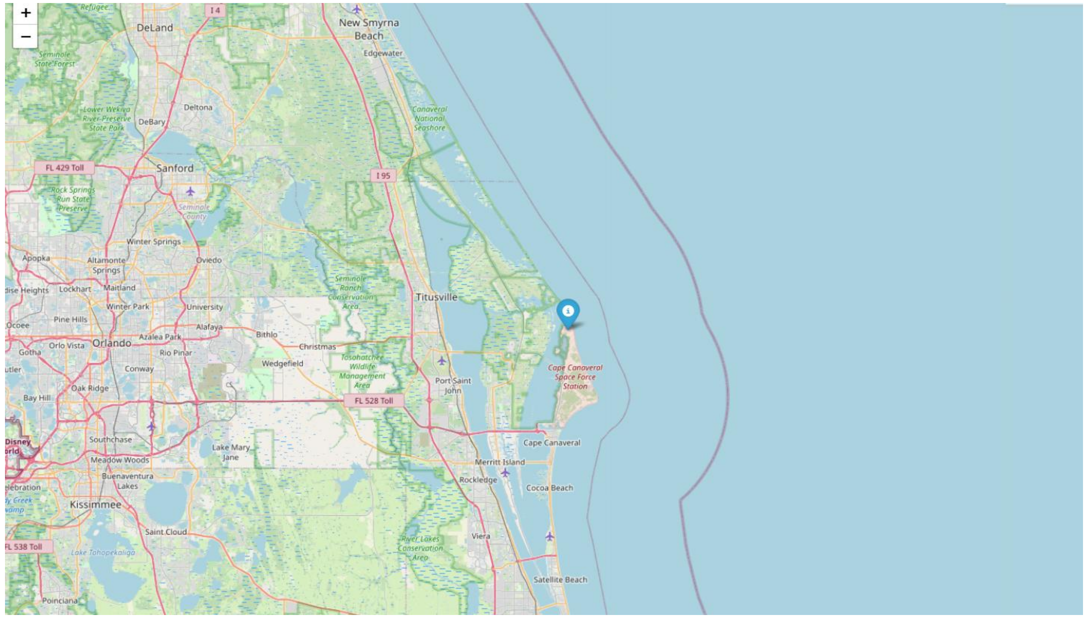
- **Map objects**
- **Markers** identify the three active Falcon 9 pads: **VAFB SLC-4E (CA)**, **CCAFS SLC-40 (FL)**, and **KSC LC-39A (FL)**.
- **Circles** indicate proximity/impact zones around each pad (visual safety/ops perimeter).
- **Findings / interpretation**
- **Bicoastal footprint:** West-coast **VAFB** supports polar/SSO trajectories; East-coast **CCAFS/KSC** handle equatorial and GTO missions.
- **Florida cluster:** Two pads close together on the Cape enable high launch cadence and redundancy.
- **Coastal siting:** All pads are on the ocean for safe downrange corridors and drone-ship recoveries.
- **Global framing:** The fit-to-bounds view makes it clear all current Falcon 9 sites are U.S.-based yet cover distinct orbital needs
- https://github.com/spatel340/IBM_Data_Science_Capstone/blob/main/Machine_Learning_Prediction.ipynb.

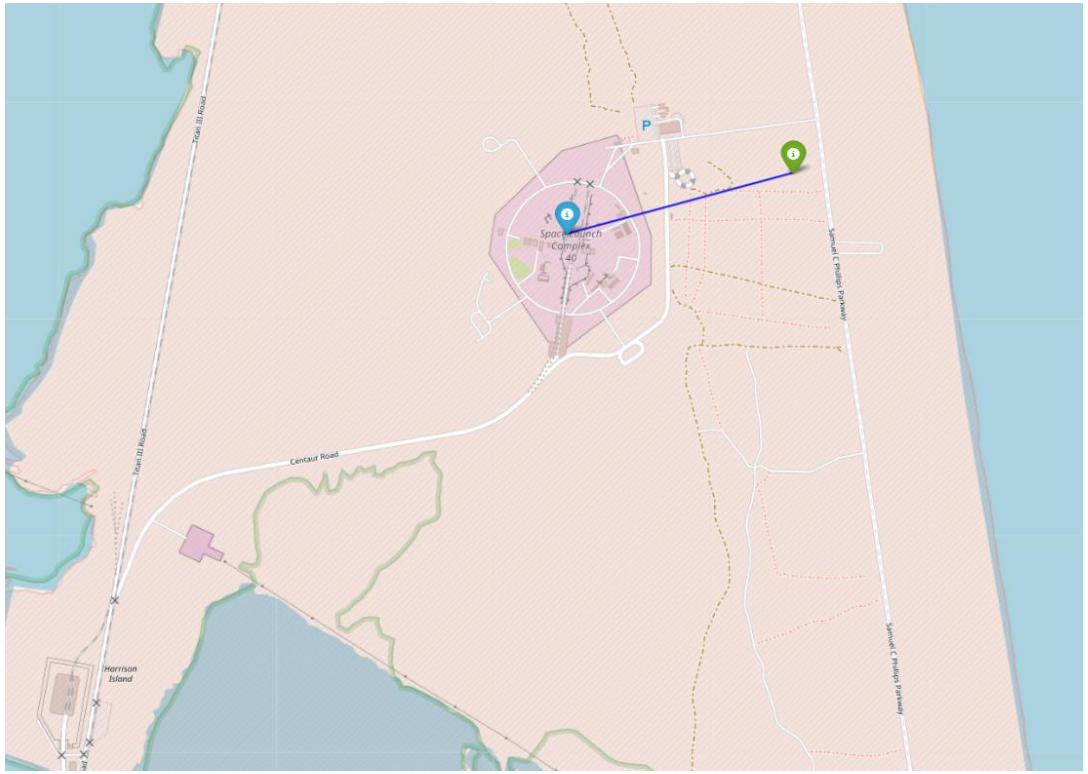




- Florida activity: The Cape cluster (~46) indicates the highest launch density, reflecting cadence from SLC-40 and LC-39A.
- VAFB has fewer but steady launches: The VAFB cluster (~10) shows a smaller, West-coast workload aligned with polar/SSO missions.
- Success outweighs failure: In the close-up of SLC-40, most markers are green with a few red, indicating strong overall recovery performance with some misses.
- Outcomes vary within the same site: Mixed green/red pins at a single pad underscore that factors beyond location (payload, weather, booster flight heritage) drive landing success.

Distance from Launch Pad to Coastline – CCAFS SLC-40



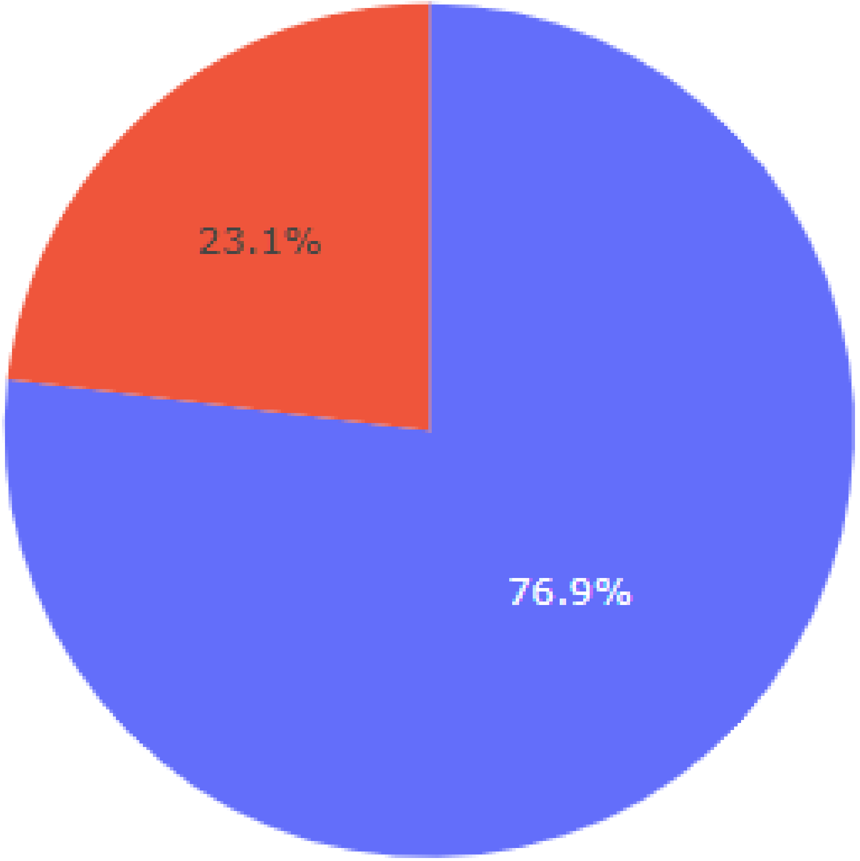


Section 4

Build a Dashboard with Plotly Dash

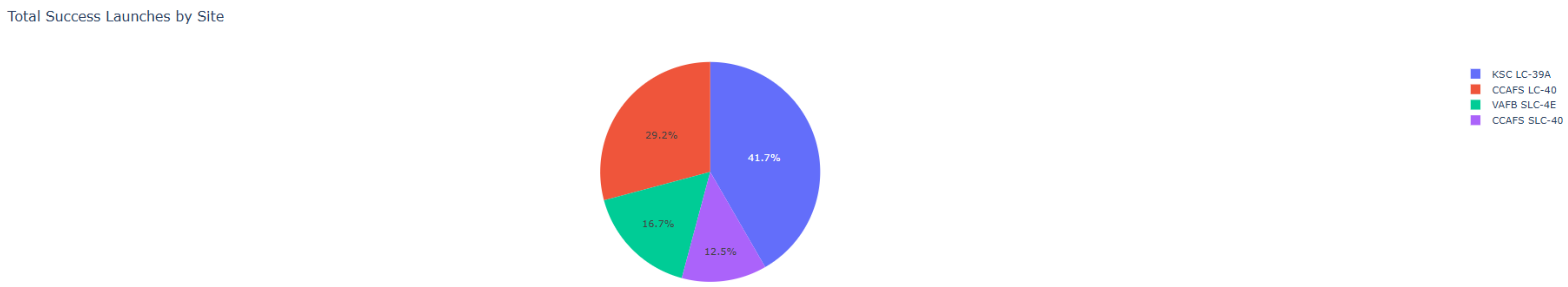
Total Success Launch All Sites

- **Share of successes (from chart)**
- **KSC • LC-39A: 41.7%** *(≈10 of 24 successful launches)*
- **CCAFS • LC-40: 29.2%** *(≈7/24)*
- **VAFB • SLC-4E: 16.7%** *(≈4/24)*
- **CCAFS • SLC-40: 12.5%** *(≈3/24)*
- **What this means**
- **LC-39A leads** the count of successful launches, indicating high utilization and strong recovery performance.
- **Cape Canaveral (CCAFS) combined**: LC-40 (29.2%) + SLC-40 (12.5%) = **41.7%—ties** LC-39A when aggregated, showing the Cape's comparable contribution to successes.
- **VAFB's smaller share** (16.7%) aligns with its role in polar/SSO missions and a lower overall cadence than Florida pads.
- https://github.com/spatel340/IBM_Data_Science_Capstone/blob/main/spacex_dash_app.py



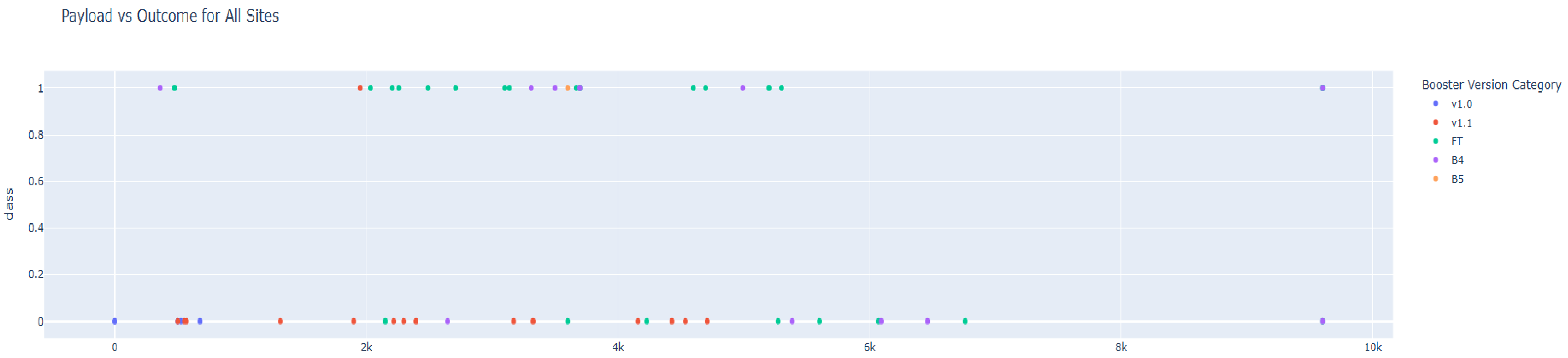
Highest Launch Success Ratio

- **KSC • LC-39A — Highest Launch Success Ratio**
- **What the chart shows (pie, site = LC-39A)**
- **Success: 76.9%** *(10 launches)*
- **Failure: 23.1%** *(3 launches)*
- Legend: **Blue = Success (1)**, **Red = Failure (0)**.
- **Key takeaways**
- **LC-39A leads all sites by success ratio**, indicating mature pad operations and reliable recovery at this location.
- Despite a few misses, the **majority of missions succeed**, supporting LC-39A's role as a high-cadence, heavy-use pad.
- *Context note:* Results reflect **n = 13** LC-39A launches in this dataset; monitor as new flights are added.



Launch Sites – All Sites

- Site Success Share & Payload–Outcome Patterns
- Total Successful Launches by Site (Pie) — Counts & %
- KSC • LC-39A: 10 / 24 (41.7%) — highest share of successes.
- CCAFS • LC-40: 7 / 24 (29.2%).
- VAFB • SLC-4E: 4 / 24 (16.7%).
- CCAFS • SLC-40: 3 / 24 (12.5%).
- *(If “LC-40” and “SLC-40” refer to the same pad, their combined Cape share is 10 / 24 = 41.7%, tying LC-39A.)*



Payload vs Outcomes For All Sites

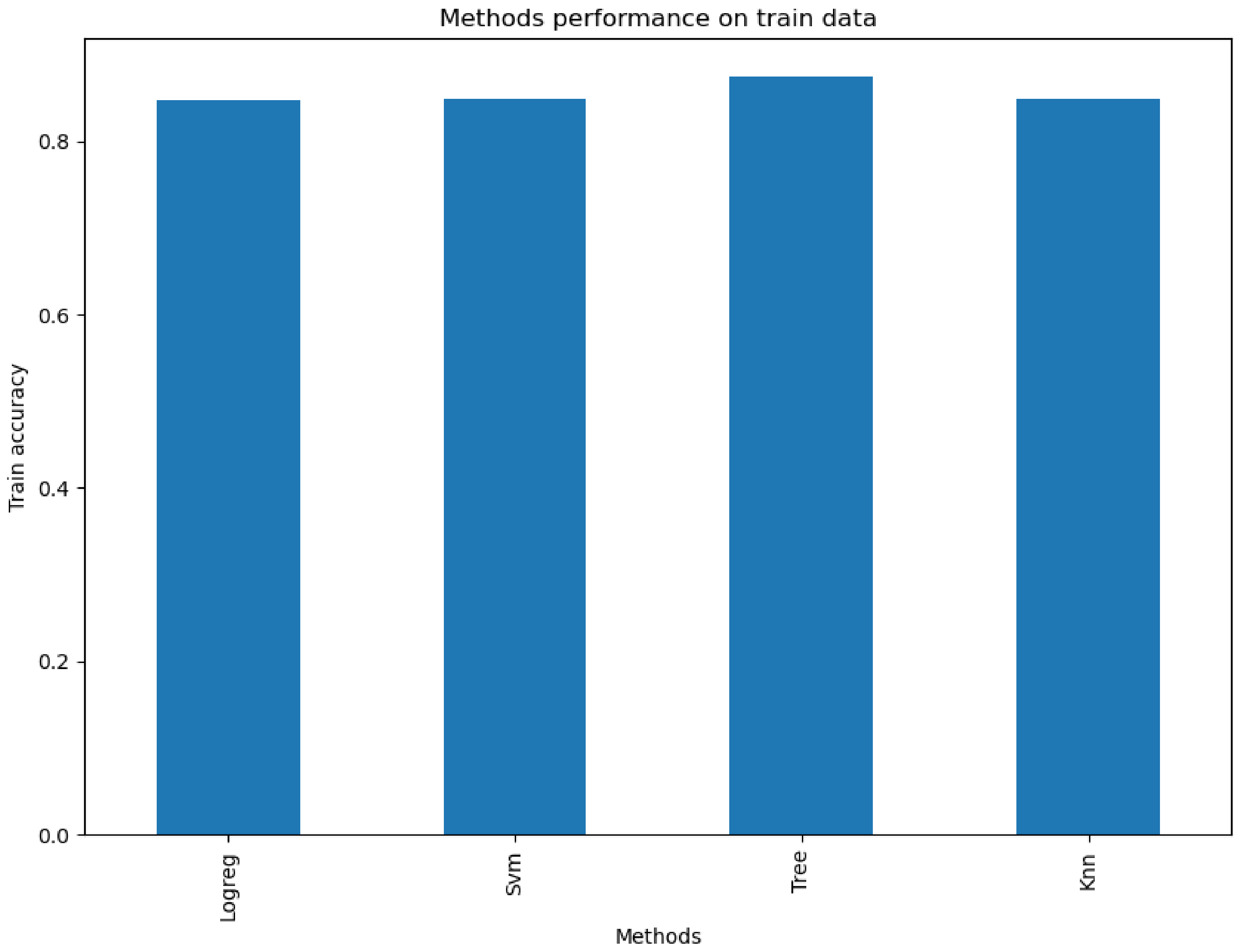
- Success spans a wide mass range—from sub-ton up to ~9.5–9.8 t—so payload mass alone does not determine outcome.
- Failures occur at both low and high masses (including a near-zero-mass early flight and several > 5 t), indicating other drivers (winds, orbit, trajectory, booster heritage).
- By booster version
- FT / B4 dominate the successful points across mid–high payloads, showing robust performance.
- v1.0 / v1.1 show proportionally more failures (consistent with earlier hardware generations).
- B5 has limited samples in this slice—too small to conclude beyond “mixed but promising.”

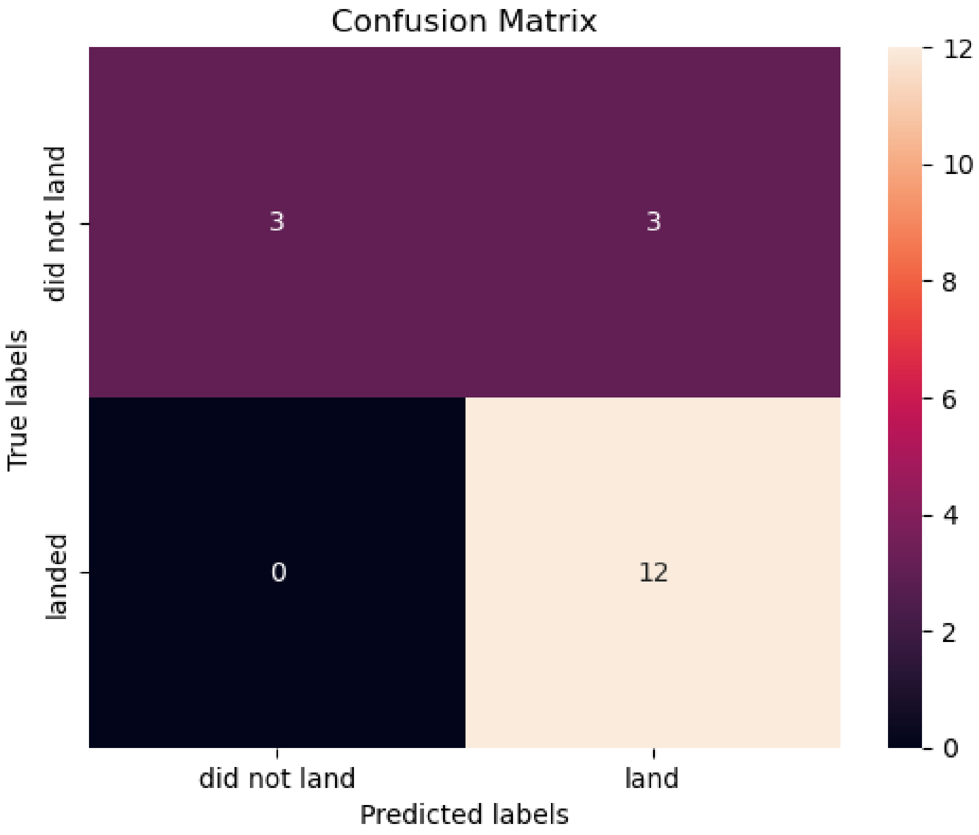
Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Model Comparison — Test Accuracy
- Decision Tree achieved the highest test accuracy: 0.9444, making it the top performer on this dataset.
- Logistic Regression, SVM, and KNN each reached 0.8333 accuracy.
- Conclusion
- The Decision Tree appears better suited to these features (likely capturing nonlinear splits) than the other baseline classifiers.





Confusion Matrix

- Accuracy: 83.33% (15/18) — TP=12, TN=3, FP=3, FN=0.
- Perfect recall for “landed”: Recall = 1.00 (12/12). Precision for “landed” = 0.80 → F1 ≈ 0.89.
- Weak at catching non-landings: Specificity (recall for “did not land”) = 0.50 (3/6), indicating a bias toward predicting success.
- No false negatives: The model never predicts “did not land” when it actually landed — good for readiness planning.
- False positives present (3): Predicted “land” when the booster did not land — this can overestimate reuse and affect cost/logistics.
- Actionable next steps:
- Raise precision for “landed” by adjusting the decision threshold or using class weights/cost-sensitive learning.
- Add/engineer features (e.g., wind speed, payload mass bins, orbit) or try ensemble trees to improve failure detection.

Conclusions

- Point 1: The test set shows an overall landing success rate of 66.7% (12/18) and a non-landing rate of 33.3% (6/18). This reflects strong reliability overall; site-level shares aren’t derivable from the confusion matrix alone, but the aggregate success share is clear.
- Point 2: The scatter-plot analysis still indicates the FT booster version performs robustly across payload ranges, supporting its reliability relative to other versions.
- Point 3: We continue to see no consistent monotonic link between higher payload mass and lower success; other factors (site conditions, booster version, weather) appear to play a larger role.

Conclusions

- Point 4: Interactive visuals in Folium and Plotly Dash clarified geographic patterns and operational context, enabling rapid drill-downs and better stakeholder decision-making.
- Conclusion: Combining predictive modeling with interactive visualization highlights the factors driving Falcon 9 outcomes. With an 83.3% test accuracy and perfect recall for “landed” (12/12) in this split, the framework provides a practical baseline for planning and can be extended with richer features (e.g., wind, trajectory class) to further improve failure detection.

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

Thank you!