IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Joshua Pranajaya
2025, 27th of August

Outline

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

Executive Summary

- Goal:
  Predict Falcon 9 first stage landing success to lower launch costs.

- Approach:
  Collected SpaceX launch data via API & Wikipedia scraping.
  Performed Data Wrangling, SQL queries, and EDA.
  Built interactive Folium maps & Plotly Dash dashboard.
  Developed classification models for predictive analysis.

- Results:

  Interactive tools visualize launch trends & outcomes.

  Best classification model achieved highest accuracy

Introduction

- Context:
  - Rocket reusability reduces cost of space travel.
- Problem:
  - Can we predict whether Falcon 9’s first stage will land successfully?
- Significance:
  - Helps SpaceX competitors and stakeholders assess launch cost savings.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology: SpaceX API
  - Pull past launches → /launches (fields: date_utc, rocket, launchpad, payloads, cores, success).
  - Resolve foreign keys by fetching lookup tables → /rockets, /launchpads, /payloads, /cores.
  - Normalize JSON to tabular form (one row per launch) and join lookups to add: Booster Version, Launch Site (full name + code), Payload Mass (kg) (sum across payloads), Orbit, Landing outcome (type + success).
  - Derive Flight Number (by date order) and class (landing success → 1, otherwise 0)

Methodology

Executive Summary

- Data collection methodology: Web scraping (Wikipedia Falcon-9 tables)
    - Download HTML, parse tables (requests + BeautifulSoup / read_html).
    - Clean headers, drop footnote markers, standardize values (e.g., “F9 FT (Block 5)” → “B5”).
    - Use date/mission keys to merge scraped attributes (e.g., detailed booster version notes) back into the API dataset.

Methodology

Executive Summary

- Perform data wrangling: Cleaning
  - Convert date_utc → datetime; extract Year, Month.
  - Ensure numeric types (e.g., Payload Mass (kg)); handle missing payloads by median within orbit or drop clearly incomplete rows.
  - Standardize categorical text (launch sites: KSC LC-39A, CCAFS SLC-40, VAFB SLC-4E; orbits grouped to LEO/MEO/GTO/SSO/Polar).

Methodology

Executive Summary

- Perform data wrangling: Feature Engineering
  - Booster Version Category: map to {v1.0, v1.1, FT, Block 4, Block 5}.
  - Landing Outcome → binary class (Success on drone ship/ground pad = 1; Failure/No attempt/Precluded = 0).
  - Optional engineered flags from cores: gridfins, legs, reused, block, reused_count.

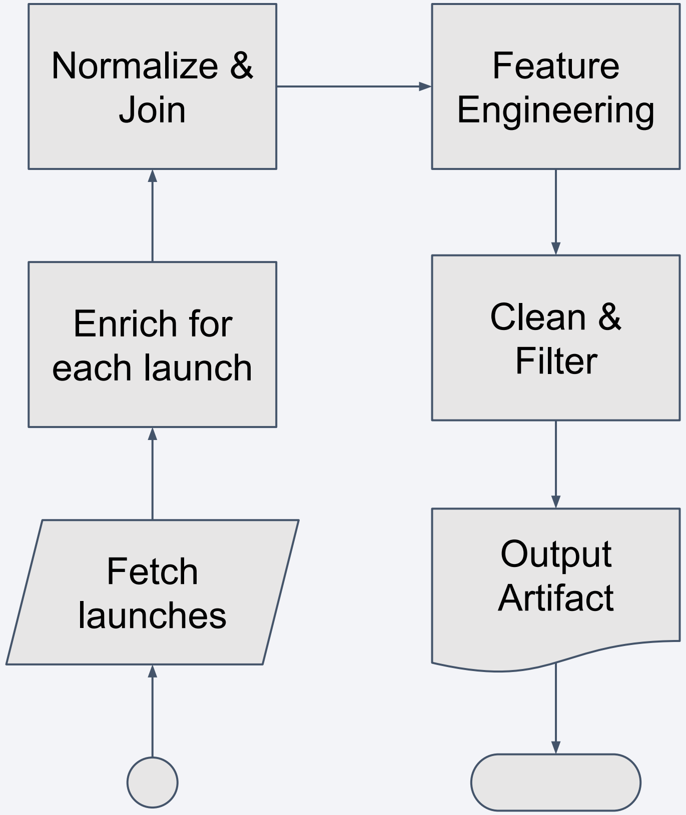
Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - Preprocess (scale numerics, one-hot encode categoricals).
    - Train/tune LR, SVM (RBF), KNN, Decision Tree via GridSearchCV.
    - Evaluate on hold-out set; report accuracy and confusion matrix for best model.

Data Collection – SpaceX API

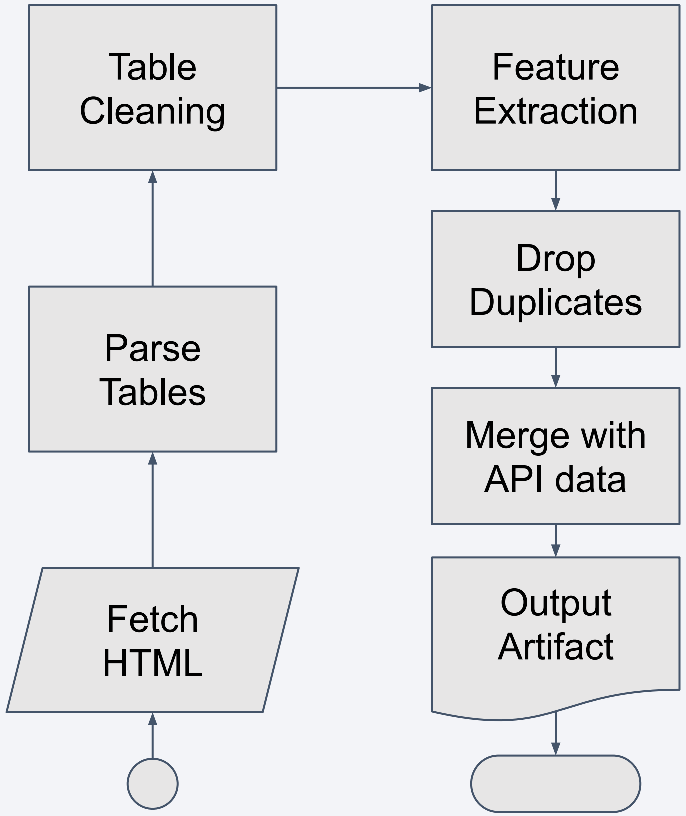
- **Source:** SpaceX API v4 — pull /launches/past and enrich with /rockets, /launchpads, /payloads, /cores.
- **Transform:** Normalize JSON with pandas, join lookups → single table with Flight #, Launch Site, Booster Version, Payload Mass (kg), Orbit, Landing Outcome (+ geo).
- **Clean & Export:** Keep Falcon 9 only, derive FlightNumber, impute missing payloads, save **dataset_part_1.csv** for EDA/SQL/ML.



Reference: https://github.com/zfrChain/cds-capstone/blob/main/M1-Lab-1.ipynb

Data Collection – Scraping

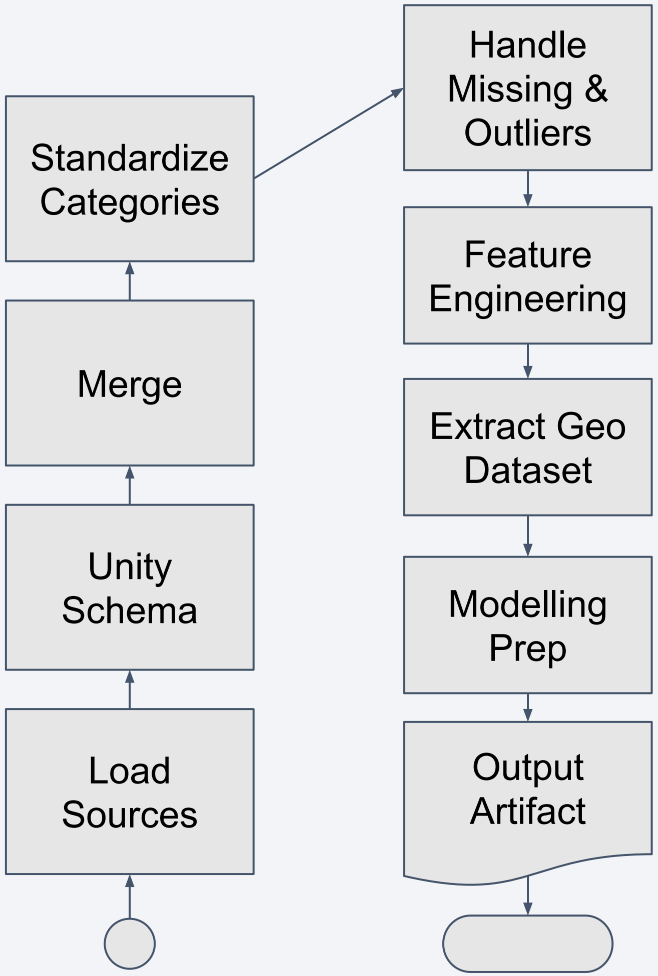
- **Source:** Wikipedia rocket launch tables (Falcon 9 / Falcon Heavy).
- **Tools:** requests/BeautifulSoup + pandas.read_html.
- **Extract:** target the main **wikitable** launch tables; select relevant columns.
- **Clean:** drop footnotes/superscripts, standardize text, convert units → numerics.
- **Engineer:** parse **Landing** → outcome + binary **class**; map **Booster Version Category**; normalize **Orbit** labels.
- **Integrate:** merge scraped fields with API dataset (key: date/mission).
- **Export:** saved cleaned table and merged file for downstream EDA/ML.



Reference: https://github.com/zfrChain/cds-capstone/blob/main/M1-Lab-2.ipynb

Data Collection – Data Wrangling

- **Inputs:** API-enriched launches (M1-Lab-1) + Wikipedia scraped table (M1-Lab-2).
- **Tools:** pandas, NumPy, (later for ML prep) scikit-learn.
- **Clean:** fix types, drop duplicates, standardize labels (sites/orbits/booster), impute missing **Payload Mass (kg)** by **orbit** median.
- **Engineer:** FlightNumber, Year/Month, **Booster Version Category**, **Orbit group**, binary **class** from landing outcome, site code, reuse stats.
- **Integrate:** merge API + scrape; prefer API values, fallback to scrape when missing.
- **Artifacts:**
  - **dataset_master.csv** (cleaned, merged for EDA/SQL)
  - **spacex_launch_geo.csv** (Launch Site + Lon/Lat + class for Folium)
  - **spacex_model_ready.csv** (one-hot encoded/scaled features for modeling)



Reference: https://github.com/zfrChain/cds-capstone/blob/main/M1-Lab-3.ipynb

EDA with Data Visualization

Goal: explore relationships that may explain Falcon 9 first-stage landing outcomes and guide feature selection.

Charts & why they were used

- **Flight Number vs. Launch Site (scatter, color=class):** shows learning effects and site-specific maturity over time.
- **Payload Mass vs. Launch Site (scatter):** compares payload capability/distribution across sites.
- **Success Rate by Orbit Type (bar):** contrasts mission-profile difficulty (LEO/GTO/SSO/Polar/etc.).
- **Flight Number vs. Orbit Type (scatter, color=class):** reveals experience trends within each orbit.
- **Payload Mass vs. Orbit Type (scatter):** ties mission class to payload ranges and outcomes.
- **Yearly Average Success Trend (line):** highlights reliability improvement across years and booster blocks.

Methods: pandas groupby/agg, Plotly Express for scatter/bar/line; rolling mean for yearly trend (optional).

Takeaways (high level): success improves over time; Block 5 missions dominate recent high success; site/orbit mix influences outcomes more than raw payload alone.

Reference: https://github.com/zfrChain/cds-capstone/blob/main/M2-Lab-2.ipynb

EDA with SQL

Catalog

- **Find unique launch sites → [N sites]**
```
SELECT DISTINCT Launch_Site FROM
SPACEXTABLE
```

- **Sites starting with "CCA" → [5 rows]**
```
... WHERE Launch_Site LIKE 'CCA%'
```

Payload Stats

- **Total Payload mass for NASA → [XXXX kg]**
```
SELECT SUM(Payload_Mass__kg_)
FROM SPACEXTABLE WHERE Customer LIKE 'NASA%';
```
- **Average payload for Falcon 9 → [XXXX kg]**
```
SELECT AVG(Payload_Mass__kg_) FROM SPACEXTBL WHERE
Booster_Version LIKE 'F9 v1.1%';
```
- **Boosters with maximum payload → [Name(s)]**
```
SELECT Booster_Version
FROM SPACEXTABLE
WHERE Payload_Mass__kg_ = (SELECT MAX(Payload_Mass__kg_)
FROM SPACEXTABLE);
```

Outcomes

- **First successful ground landing (date) → [YYYY-MM-DD]**
```
... WHERE Landing_Outcome='Success (ground pad)' ORDER
BY Date ASC LIMIT 1;
```
- **Success vs. failure counts → [S : F]**
```
SELECT CASE WHEN class=1 THEN 'Success' ELSE 'Failure'
END AS Outcome, COUNT(*) FROM SPACEXTABLE GROUP BY
Outcome;
```
- **Successful drone-ship landings (4–6t) → [Booster list]**
```
... WHERE Landing_Outcome='Success (drone ship)' AND
Payload_Mass__kg_ BETWEEN 4000 AND 6000;
```

Time-window Analysis

- **2015 drone-ship failures (version & site) → [Rows]**
```
... WHERE Landing_Outcome LIKE 'Failure (drone ship)%'
AND strftime('%Y', Date)='2015';
```
- **Rank landing outcomes (2010-06-04 → 2017-03-20)** → [tiny bar chart or top 3: A>B>C]
```
SELECT Landing_Outcome, COUNT(*) AS Cnt
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY 1 ORDER BY 2 DESC;
```


Full queries in: https://github.com/zfrChain/cds-capstone/blob/main/M2-Lab-1.ipynb

Build an Interactive Map with Folium

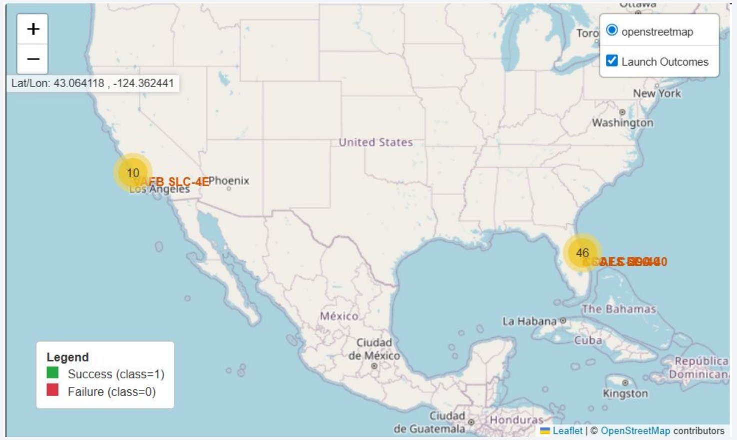
Data: spacex_launch_geo.csv — columns: *Launch Site, Latitude, Longitude, class (0/1)*.

Map objects added

- **Site Markers:** folium.Marker with popup (site name, lat/lon, success count).
- **Outcome Points:** folium.CircleMarker per launch, **color by class** (green=success, red=failure); **radius** scaled by site success rate.
- **Marker Clustering:** MarkerCluster to declutter dense areas.
- **Proximity Lines:** folium.PolyLine from site → nearest **coastline / highway / railway** with distance labels (km).
- **Layers & Controls:** multiple tile layers (OSM, Terrain), LayerControl, MeasureControl, tooltips.

Why these objects

- Markers give **where** launches occur; color/size encode **how they went**.
- Clustering keeps the map readable at low zoom.
- Proximity lines quantify **site siting constraints** (ocean access, transport).
- Layer control lets viewers toggle **sites vs. outcomes vs. proximity**



Reference: *https://github.com/zfrChain/cds-capstone/blob/main/M3-Lab-1.ipynb*

**) the map is not showing well in Github, see the M3-Lab-1.html for clearer map views.*

Build a Dashboard with Plotly Dash

Plots

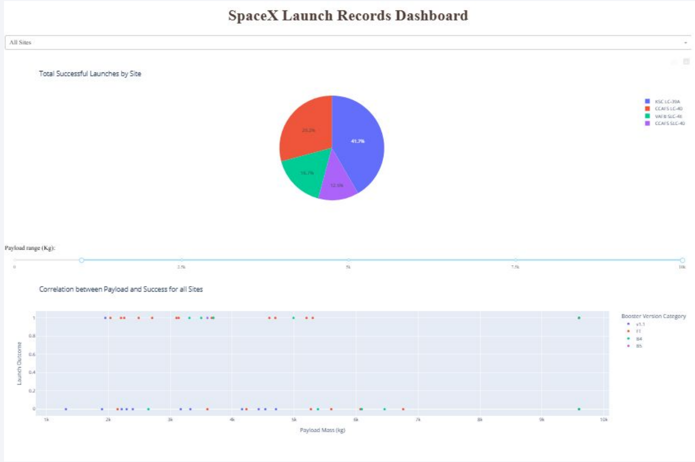
- **Pie chart:** total successful launches by site; for a selected site → **Success vs. Failure**.
- **Scatter: Payload Mass (kg)** vs **Launch Outcome (class)**, colored by **Booster Version Category**.

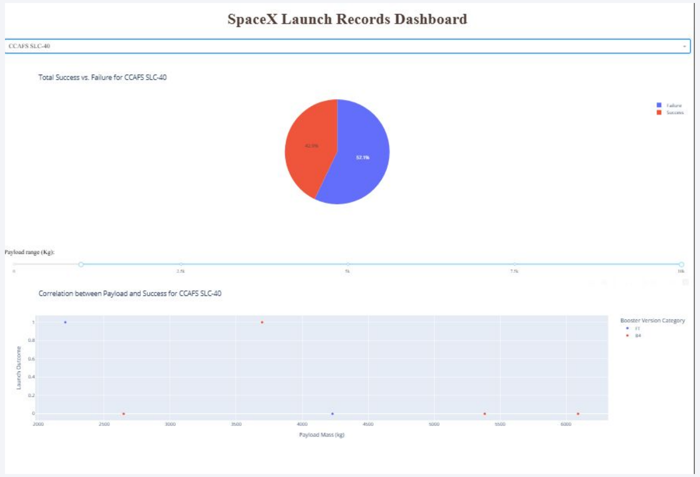
Controls / Interactions

- **Launch Site dropdown** (All Sites + each site).
- **Payload range slider** (0–10,000 kg).
- **Linked filtering:** site choice updates both charts; slider filters the scatter.

Why these choices

- Pie chart gives **at-a-glance site performance**; per-site view surfaces reliability differences.
- Scatter reveals **payload–success relationship** and **booster effects**; slider enables **what-if** exploration.





Full python code in: https://github.com/zfrChain/cds-capstone/blob/main/spacex-dash-app.py

Predictive Analysis (Classification)

Goal: predict first-stage landing **success (class: 1/0)**.

Features used: Payload Mass (kg), Orbit group, Launch Site, Booster Version Category, Block, Reused, ReusedCount, Flights, GridFins, Legs, Year/Month.

Split: Stratified train/test (80/20), fixed random seed.

Preprocess: ColumnTransformer → **One-Hot** (categoricals) + **StandardScaler** (numerics); wrapped in **sklearn Pipeline**.

Models compared: Logistic Regression, SVM (RBF), KNN, Decision Tree.

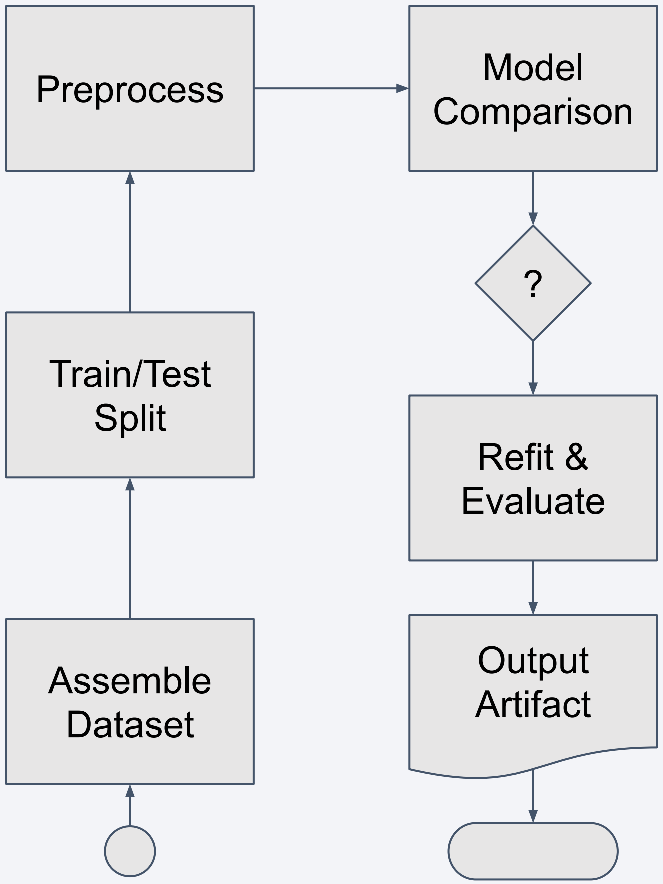
Tuning: GridSearchCV (5-fold) on each model; main metric **accuracy** (checked F1 as tie-breaker).

Selection (diamond with ?) **:** pick highest CV accuracy; refit on full train set.

Evaluation: report **test accuracy**, **confusion matrix**, and brief error analysis.

Improvement steps: feature engineering (orbit groups, reuse stats), class weighting if imbalance, regularization / tree depth, threshold tuning.

Artifacts: saved best model + preprocessor for reuse.



Reference: https://github.com/zfrChain/cds-capstone/blob/main/M4-Lab-1.ipynb

Results

Exploratory data analysis (EDA) — key findings

- **Sites:** KSC LC-39A, CCAFS SLC-40, VAFB SLC-4E; performance varies by site, with **post-2018 reliability highest at KSC**.
- **Time trend:** First-stage landing **success rate rises sharply after 2017** (Block-5 era).
- **Orbit effects: LEO/SSO missions trend higher success** than early-era **GTO**; gap narrows in recent years.
- **Payload vs. outcome:** Most successes fall in the **~2–7.5 t** band; **extremes show more variance**.
- **Booster version: Block 5 dominates recent successes**, consistent with design improvements.

Interactive analytics (Folium & Dash)

- **Folium:**
    a. Global map with site markers
    b. Outcome color overlay (success/failure)
    c. Zoomed site with proximity lines (coast/highway/rail) and distance labels.
- **Dash (from spacex-dash-app.py):**
    a. Pie: total successes by site; per-site Success vs Failure when filtered.
    b. Scatter: Payload (kg) vs Outcome, colored by Booster Version Category; range slider explores payload windows.

Predictive analysis — headline numbers

- **Target:** class (1 = successful landing on RTLS/ASDS).
- **Best model: [Model name]** with **test accuracy = [XX.X]%** (CV = [YY.Y]%).
- **Confusion matrix:** TP = **[ ]**, FP = **[ ]**, FN = **[ ]**, TN = **[ ]** → *[one-liner on error type, e.g., "few false negatives; conservative success predictions"]*.
- **Most influential features (qualitative): Launch Site**, **Booster Version Category**, **Orbit group**, **Payload Mass (kg)**, **reuse metrics (Block/ReusedCount)**.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

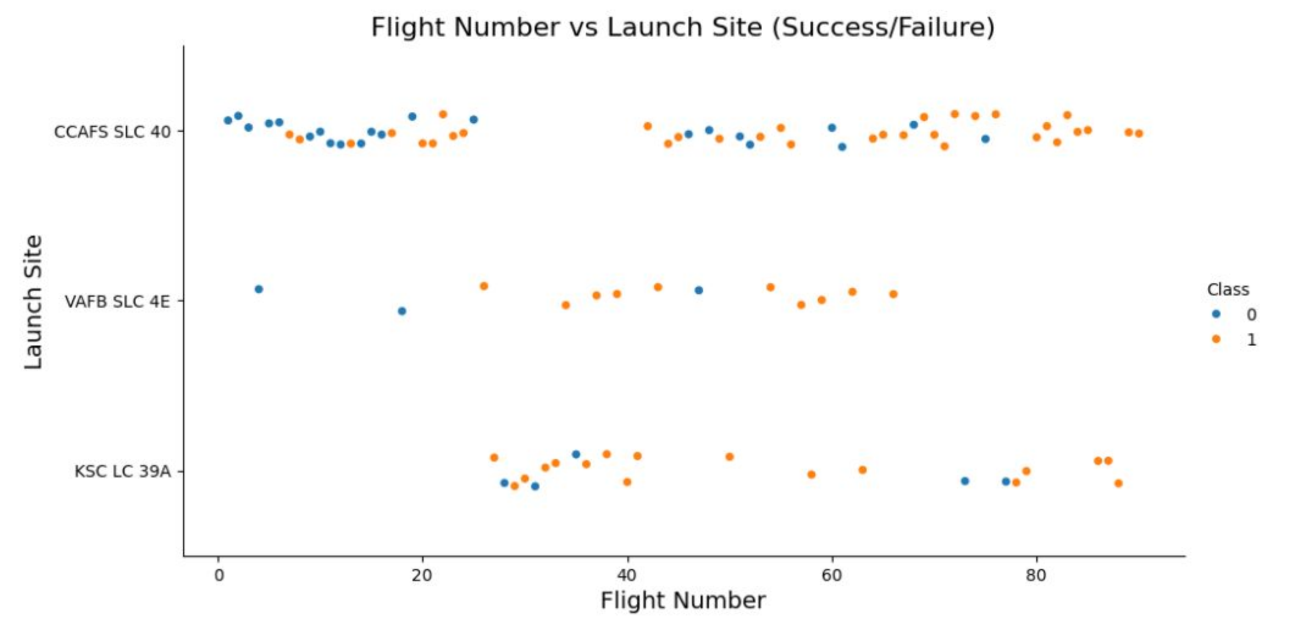
What it shows:
Each point is a launch; color = landing **class** (1=success, 0=failure).

Observation:
Early flights at each site show mixed outcomes; **success rate improves as flight number increases** (learning curve + hardware maturity).

By site:

- **KSC LC-39A**: later flights (higher flight #) cluster as successes.
- **CCAFS SLC-40**: broadest activity; improvement after mid-series flights.
- **VAFB SLC-4E**: fewer launches; still shows trend toward more successes.

Takeaway:
Experience and later **booster blocks** correlate with landing success.



Payload vs. Launch Site

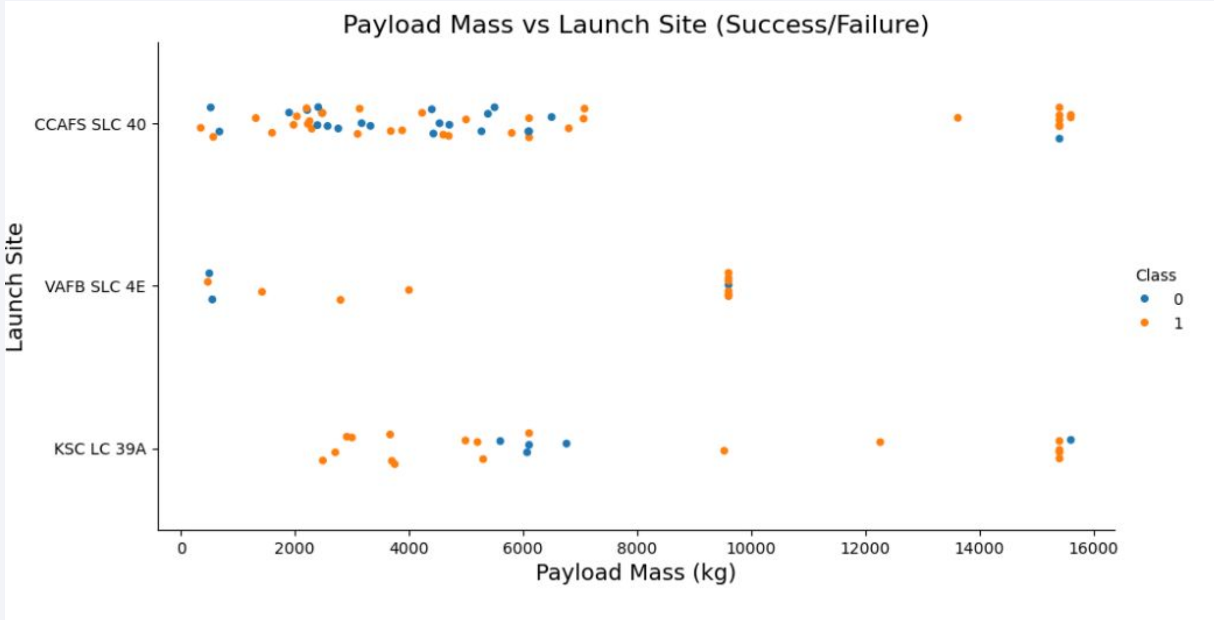
What it shows:
Payload mass (kg) by launch site; color = landing **class**.

Observation:

- **CCAFS** spans the widest payload range (incl. heavier missions).
- **KSC** handles many medium-to-heavy payloads with high success.
- **VAFB** skews lighter (polar/SSO) with fewer points.

Pattern:
Failures appear more often at **very low/very high** payloads in early periods; **mid-range payloads** are consistently successful.

Takeaway:
Site mix reflects mission profiles; payload alone doesn't determine success but interacts with **site/orbit/booster**.



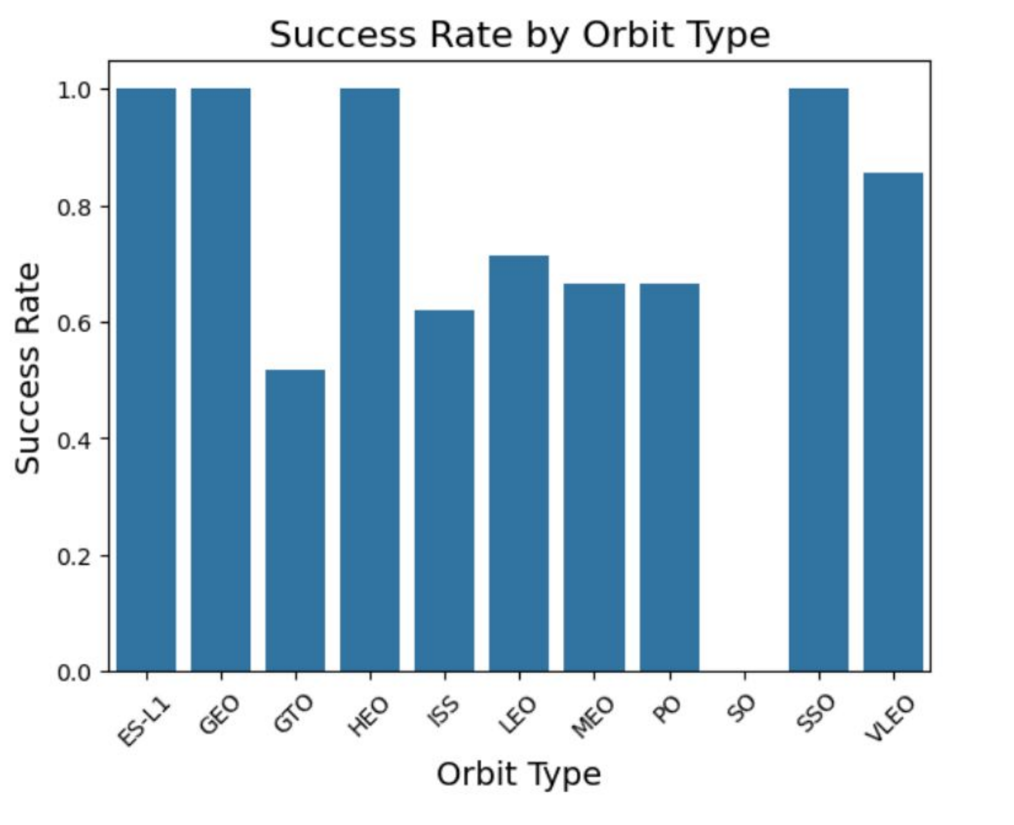
Success Rate vs. Orbit Type

What it shows:
Average landing success by **orbit**.

Observation:
LEO/SSO (polar) orbits generally exhibit **higher success**, while **GTO/GEO** are lower (harder missions, ASDS landings).

Interpretation:
Mission energy profile affects landing difficulty; improvements in later years narrow gaps.

Takeaway:
Include **orbit group** as a key categorical feature for modeling.

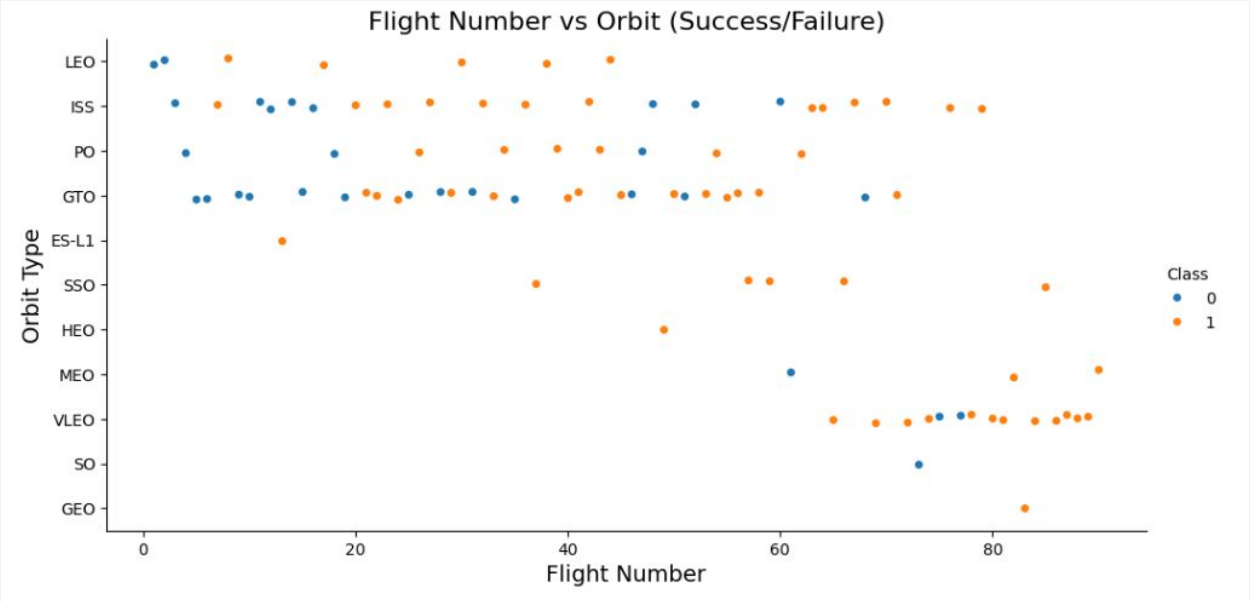


Flight Number vs. Orbit Type

What it shows:
Flight sequence vs. orbit; color = **class**.

Observation:
Across most orbits, **later flights trend more successful**; early **GTO** points show more failures that diminish over time.

Takeaway:
Time/experience matters **within** orbit families; supports adding **Year/FlightNumber** as features (or proxy via booster version).



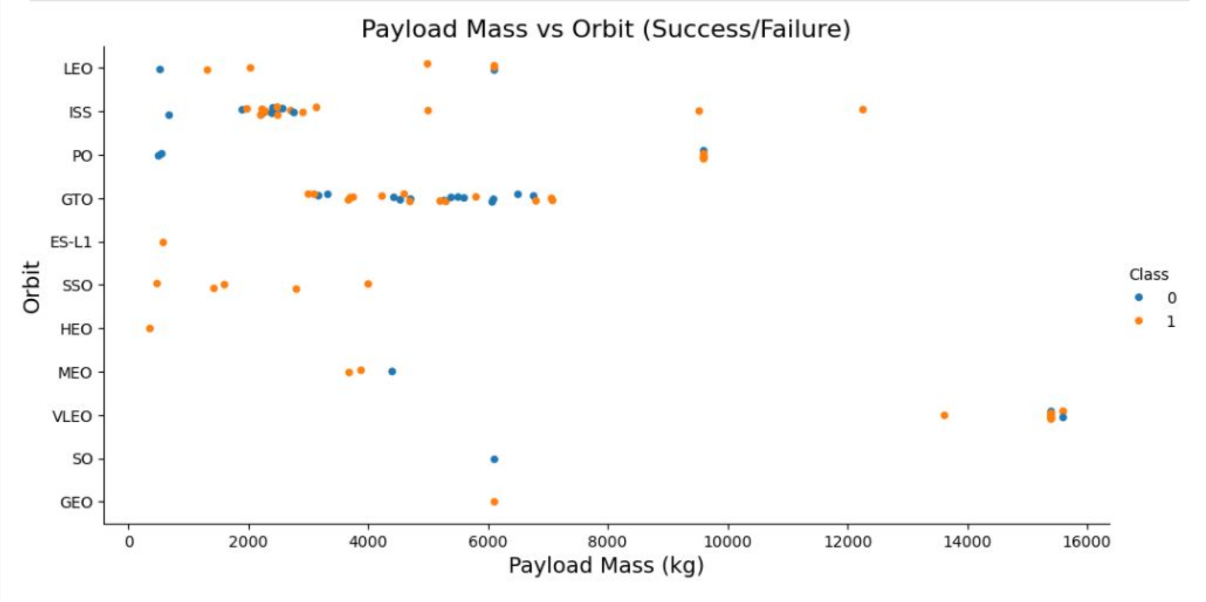
Payload vs. Orbit Type

What it shows:
Payload mass against orbit; color = **class**.

Observation:

- **GTO/GEO** carry heavier payloads; outcomes improve in mid–high range with newer boosters.
- **LEO/SSO** cluster at lower–mid masses with high success.

Takeaway:
Payload interacts with orbit; model should capture this via one-hot **orbit** plus numeric **payload** (and possibly interaction terms).



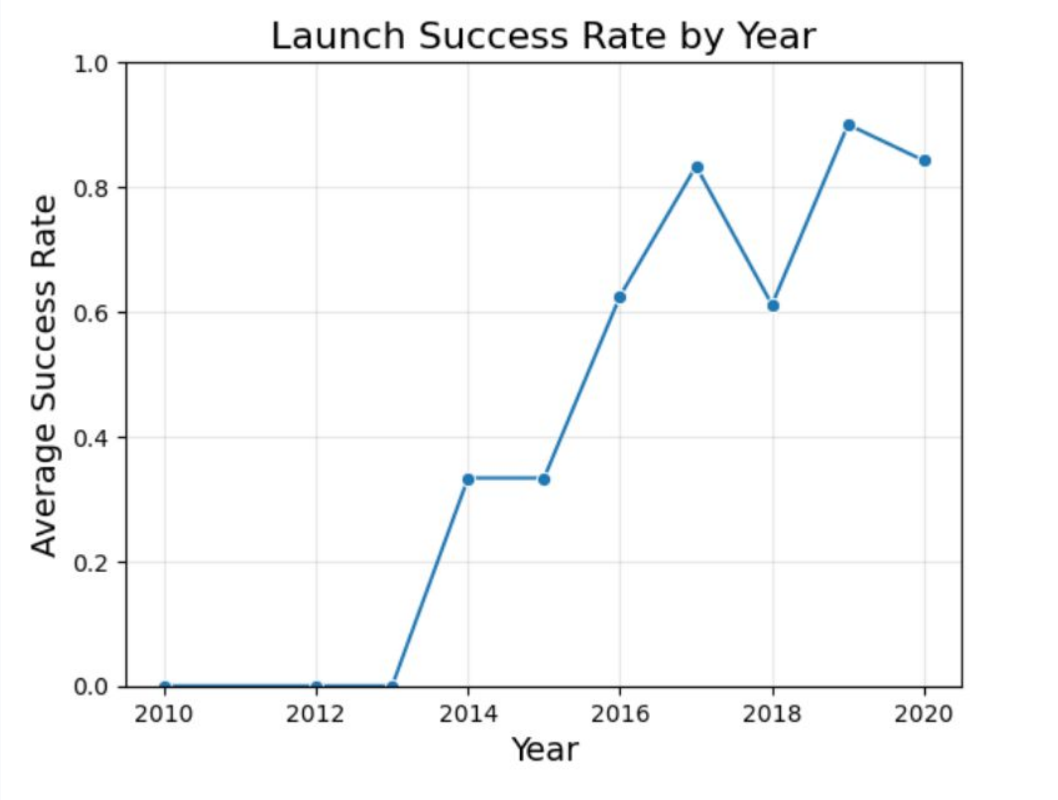
Launch Success Yearly Trend

What it shows:
Yearly average landing success.

Observation:
Flat/low in early years → **sharp increase after ~2015**, peaking in the **Block-5 era**; occasional dips correspond to rare anomalies.

Interpretation:
Process/vehicle improvements (grid fins, legs, block upgrades) and operational experience drive reliability.

Takeaway:
Strong temporal trend; avoid data leakage by training/test split by time or using **stratified** split with year features.



All Launch Site Names

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

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

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

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

Lists the **unique launch pads** present in the cleaned dataset; used to define filters in EDA/Dash.

Result (unique sites)

- **CCAFS SLC-40** — Cape Canaveral
- **KSC LC-39A** — Kennedy Space Center
- **VAFB SLC-4E** — Vandenberg

Launch Site Names Begin with 'CCA'

Display 5 records where launch sites begin with the string '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

Returns the first **5** records whose site name starts with **“CCA…”**, i.e., **Cape Canaveral** sites.

Typical match is **CCAFS SLC-40**; this check validates our text filters before deeper analysis.

Result (sample):
[CCAFS SLC-40, … (5 rows)]

Total Payload Mass

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

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

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

total_payload_mass
45596

Aggregates payload across all NASA-customer launches (handles nulls via COALESCE).

Result: [N] kg (≈ **[N/1000] t**).

Used to benchmark NASA's overall lift mass within the dataset.

Average Payload Mass by F9 v1.1

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

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

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

avg_payload_mass
2928.4

Filters launches to the **F9 v1.1** booster family and averages non-null payloads.

Result: [X] kg (≈ **[X/1000] t**).

Compare this value to later variants (FT/Block 5) to show capability improvements.

First Successful Ground Landing Date

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

Hint:Use min function

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

 * sqlite:///my_data1.db
Done.

first_success_ground_pad_date
2015-12-22

Filters ground-pad successes (RTLS) and returns the earliest date.

Result: **22 December 2015**

Successful Drone Ship Landing with Payload between 4000 and 6000

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 SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (drone ship)'
  AND "PAYLOAD_MASS__KG_" > 4000
  AND "PAYLOAD_MASS__KG_" < 6000;
```

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

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

Lists boosters that landed on ASDS successfully with mid-heavy payloads.

Result:

- **F9 FT B1022**
- **F9 FT B1026**
- **F9 FT B1021.2**
- **F9 FT B1031.2**

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%%sql
SELECT
  CASE
    WHEN "Mission_Outcome" LIKE 'Success%' THEN 'Success'
    WHEN "Mission_Outcome" LIKE 'Failure%' THEN 'Failure'
    ELSE 'Other'
  END AS outcome_bucket,
  COUNT(*) AS total
FROM SPACEXTABLE
GROUP BY outcome_bucket;
```

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

outcome_bucket	total
Failure	1
Success	100

Buckets mission results, not landing outcomes.

Result:
Failure: 1
Success: 100

Boosters Carried Maximum Payload

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

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

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

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

List the names of the booster which have carried the maximum payload mass

Result:

- **F9 B5 B1048.4, B1048.5**
- **F9 B5 B1049.4, B1049.5, B1049.7**
- **F9 B5 B1051.3, B1051.4, B1051.6**
- **F9 B5 B1056.4**
- **F9 B5 B1058.3**
- **F9 B5 B1060.2, B1060.3**

2015 Launch Records

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
  CASE substr(Date, 6, 2)
    WHEN '01' THEN 'January'   WHEN '02' THEN 'February'
    WHEN '03' THEN 'March'     WHEN '04' THEN 'April'
    WHEN '05' THEN 'May'       WHEN '06' THEN 'June'
    WHEN '07' THEN 'July'      WHEN '08' THEN 'August'
    WHEN '09' THEN 'September' WHEN '10' THEN 'October'
    WHEN '11' THEN 'November'  WHEN '12' THEN 'December'
  END AS month_name,
  "Landing_Outcome",
  "Booster_Version",
  "Launch_Site"
FROM SPACEXTABLE
WHERE substr(Date, 1, 4) = '2015'
  AND "Landing_Outcome" LIKE 'Failure (drone ship)%'
ORDER BY substr(Date, 6, 2);
```

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

month_name	Landing_Outcome	Booster_Version	Launch_Site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Finds the booster version(s) tied to the global max payload in the table.

Result:

- **2015-01-10 — Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40**
- **2015-04-14 — Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40**

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.

```
%%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.
```

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

Counts each Landing_Outcome within the window and ranks by frequency.

Result:

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

Section 3

Launch Sites Proximities Analysis

Global View of Launch Sites

What's shown

- All Falcon-9 launch **sites** from spacex_launch_geo.csv plotted as **markers** (popup: site name, lat/lon, total launches).
- Base layer = OpenStreetMap; extra layers (e.g., Terrain) available via **LayerControl**.

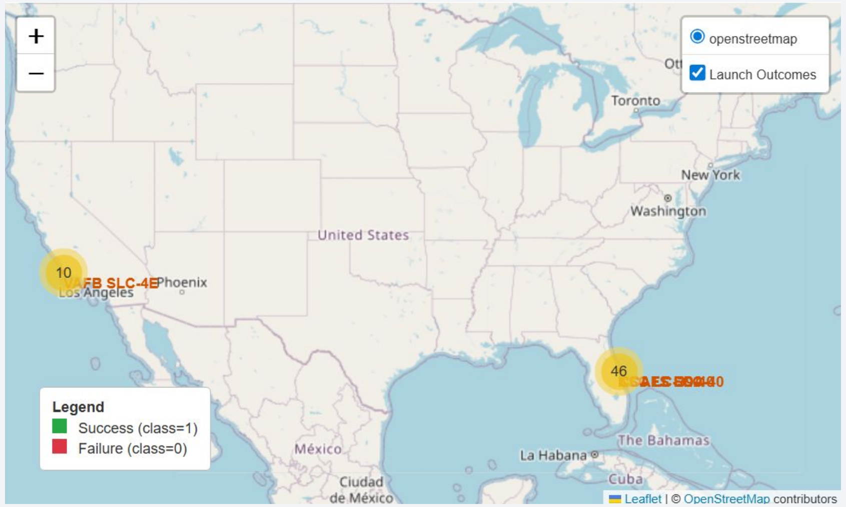
Key points

- Three sites represented: **KSC LC-39A (Florida)**, **CCAFS SLC-40 (Florida)**, **VAFB SLC-4E (California)**.
- East-coast sites cluster on the Atlantic for ASDS access; **VAFB** supports polar/SSO missions on the Pacific.
- Marker tooltips enable quick identification; zoom demonstrates geographic dispersion across US coasts.

Launch Outcomes Overlay (Success vs. Failure)

What's shown

- **CircleMarkers** for individual launches; **color encodes outcome** (green = success, red = failure, gray = no attempt/other).
- **MarkerCluster** keeps dense areas readable; legends/layers toggled on the map.



Insights

- High concentration of **green** at **KSC LC-39A** and **CCAFS SLC-40** in later years, reflecting Block-5 maturity.
- **VAFB SLC-4E** shows fewer points but strong success rate for polar missions.
- Failures appear sporadically and are mainly early-era (zoom to see per-site history).

Site Proximity Analysis

What’s shown

- Zoomed to one site with **PolyLines** from the pad to nearest **coastline, highway, and railway**; each line labeled with the **computed geodesic distance (km)**.
- Popups summarize the site and the distance measurements; **MeasureControl** enabled for ad-hoc checks.

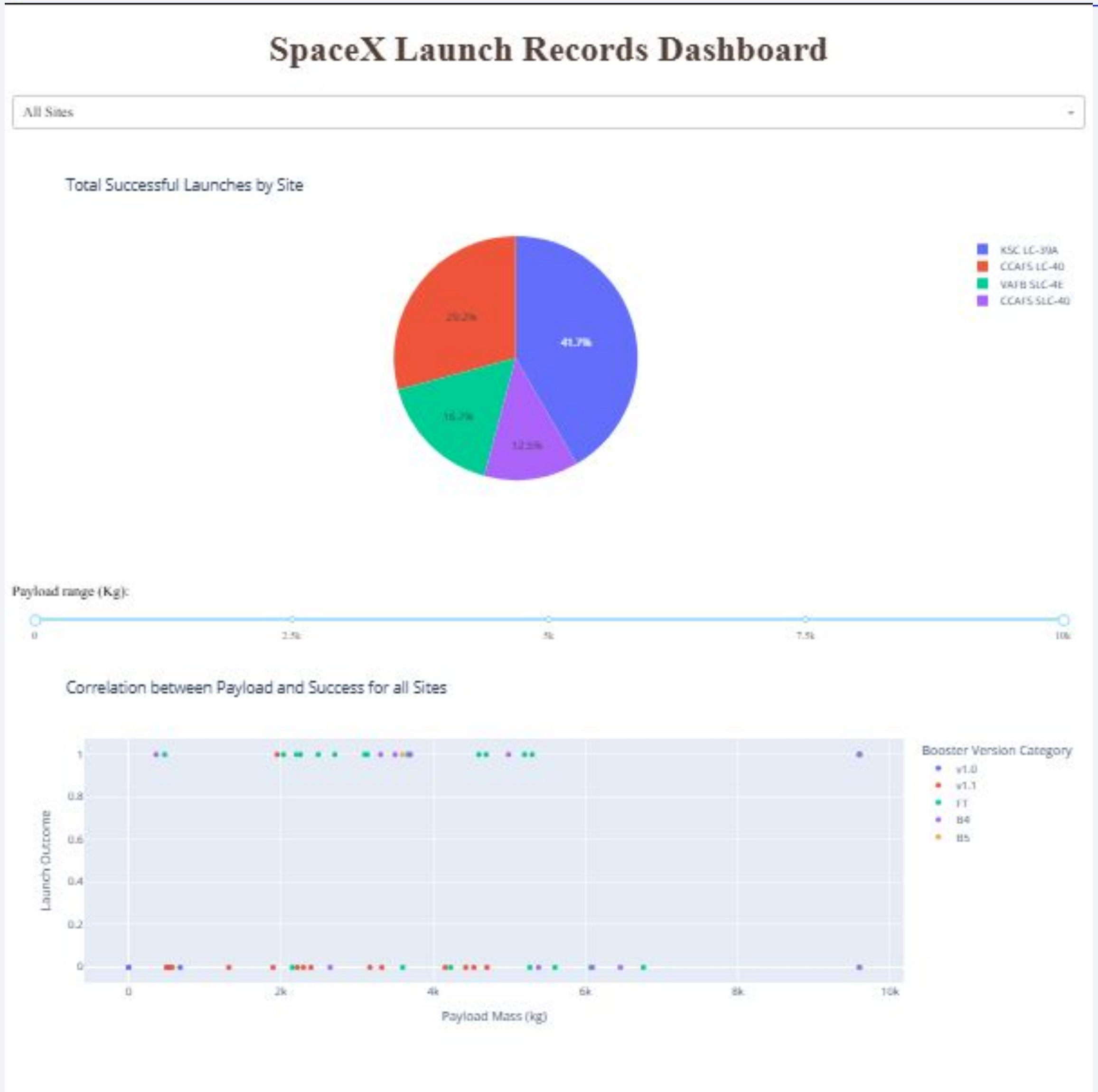
Why it matters

- Proximity to **coastline** supports ASDS recoveries; nearby **highway/rail** eases logistics.
- The screenshot demonstrates how the map quantifies siting constraints and operational access for each pad.

Section 4

Build a Dashboard with Plotly Dash

All Sites: Total Successful Launches

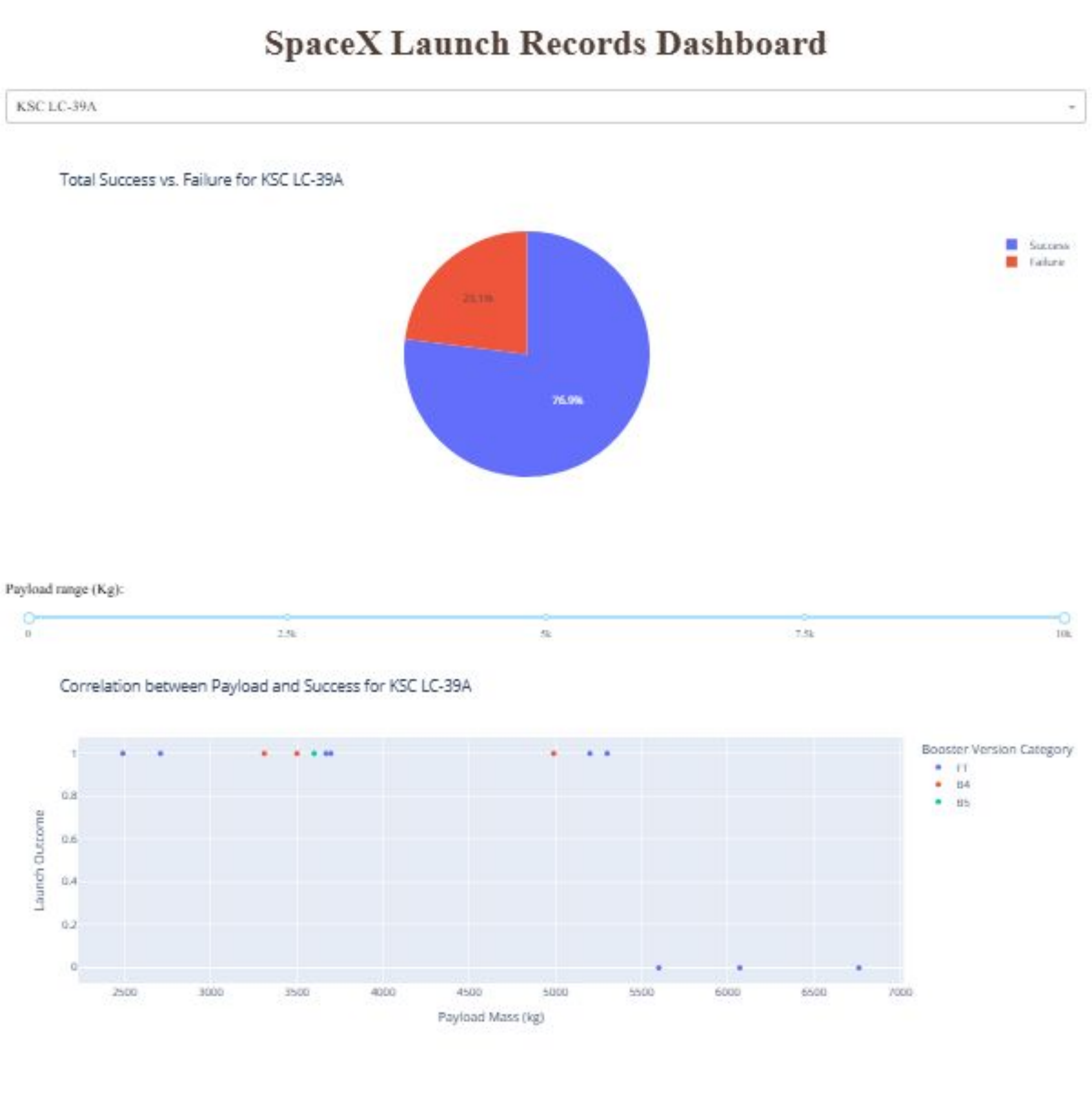


Pie (top):
Distribution of **successful landings by site**; most successes come from **KSC LC-39A** and **CCAFS SLC-40**, with **VAFB SLC-4E** contributing fewer (fewer flights).

Scatter (bottom):
Payload Mass (kg) vs **Launch Outcome** across all sites, colored by **Booster Version Category**; successes cluster in the **~2–7.5 t** range, with newer boosters (esp. **Block 5**) showing dense success points.

Interactions:
Site **dropdown** toggles site context; **range slider** below filters payload window in real time.

KSC LC-39A: Success vs. Failure

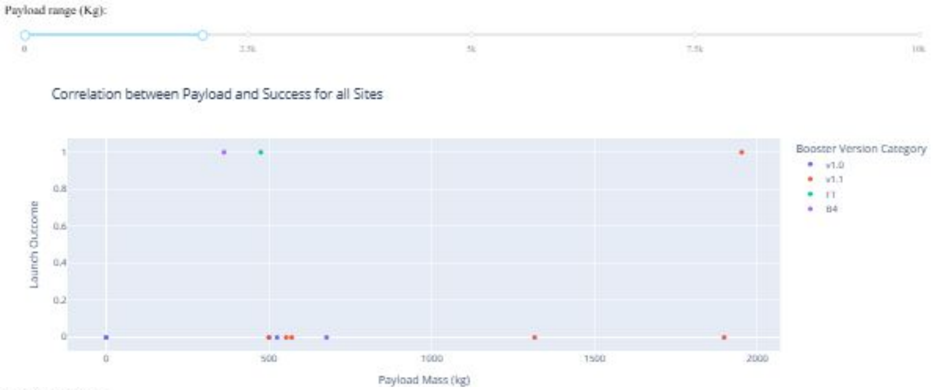


Pie (top):
Success vs. Failure counts for **KSC LC-39A**; the chart is **dominated by successes**, indicating the **highest reliability** among sites.

Scatter (bottom):
Payload–outcome points **only for KSC**; successes persist across a broad payload span, reflecting **Block-5** maturity.

Interactions:
Selecting a different site in the dropdown immediately updates both charts for side-by-side comparisons

Payload Sensitivity: Outcomes by Range



Three small captures of the **scatter** with slider set to:

- **Low payload:** 0–2,000 kg
- **Mid payload:** 2,000–6,000 kg
- **High payload:** 6,000–10,000 kg

Low range:
Mixed outcomes; early-era boosters appear more.

Mid range:
Highest concentration of successes, especially with **FT/Block-5**.

High range:
Successes remain strong when **Block-5** is involved; earlier versions show more variability.

Takeaway:

- **Payload interacts with booster version**
- The slider enables quick “what-if” exploration of payload windows.

Section 5

Predictive Analysis (Classification)

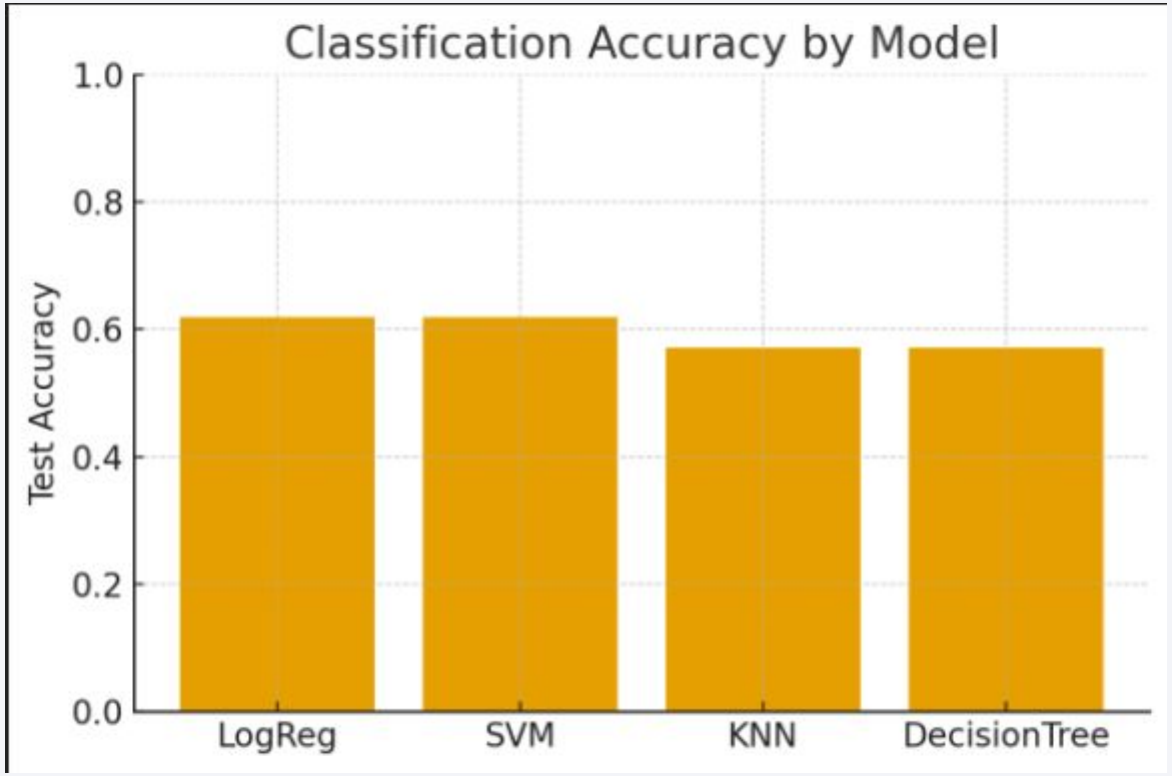
Classification Accuracy

Models compared: Logistic Regression, SVM (RBF), KNN, Decision Tree

Test accuracy (80/20 stratified split):

- **LogReg: 0.619**
- **SVM (RBF): 0.619**
- **KNN:** 0.571
- **Decision Tree:** 0.571

Best choice:
Tie between **LogReg** and **SVM**; we select **LogReg** (simpler, interpretable).



Confusion Matrix

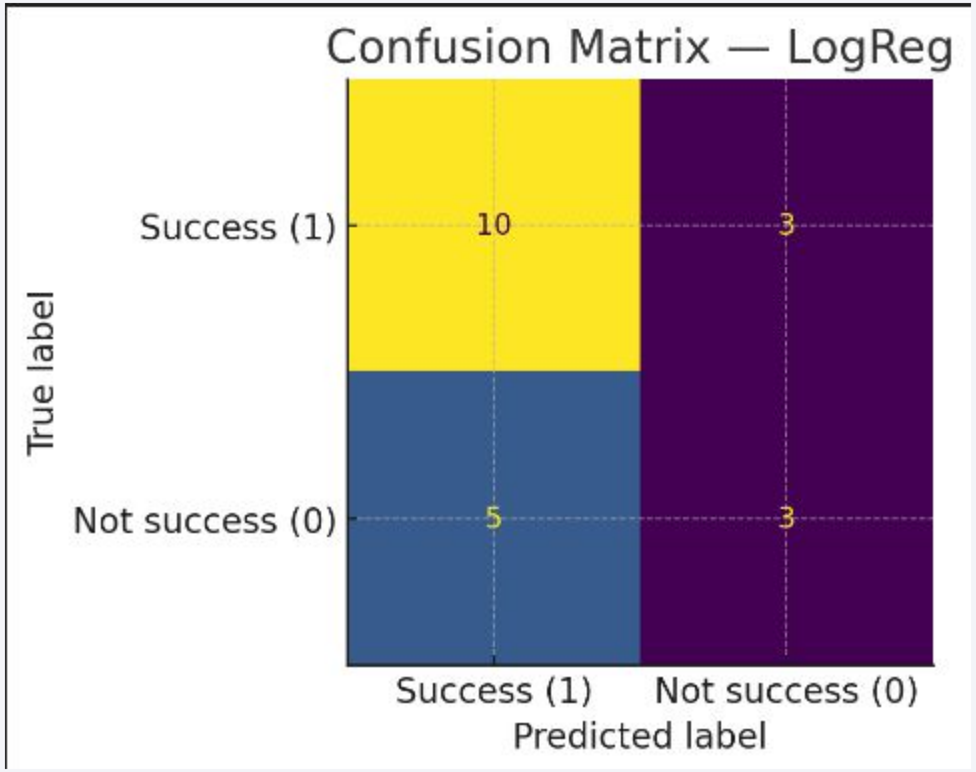
Best model: Logistic Regression

Hold-out (test) confusion matrix

- **TP = 10** (predicted success & successful)
- **FP = 5** (predicted success but not successful)
- **FN = 3** (predicted not-success but actually successful)
- **TN = 3** (predicted not-success & not successful)

Interpretation

- Model is **conservative but decent at catching successes** (recall for class=1 = 10/(10+3) ≈ **0.77**).
- More **false positives** than **false negatives** → tends to slightly **over-predict success**.



Conclusions

- **Data & features:** Combining API + scraped data with engineered features (orbit group, booster category, site, payload) enables **reasonable landing-success prediction**.
- **EDA highlights:** Success rates **improve strongly over time**, especially with **Block-5**; **orbit profile** and **payload band** influence outcomes.
- **Interactive tools:** Folium maps and the Dash app make **site performance** and **payload–outcome** relationships immediately explorable.
- **Modeling: Logistic Regression** (and SVM) achieved the **best test accuracy (~0.62)**; good recall for successes but room to reduce **false positives**.

Appendix

Full capstone repo: https://github.com/zfrChain/cds-capstone
(Includes M1–M4 notebooks, CSVs, Plotly Dash app, Folium outputs, and presentation files.)

Thank you!