IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Mª Carmen Obrero Tapias
August 21, 2025

Outline

- Executive Summary
- Introduction
- Methodology
- Insights drawn from EDA
- Launch Sites Proximities Analysis
- Dashboard with Plotly Dash
- Predictive Analysis (Classification)
- Conclusions
- Appendix

Executive Summary

- **Brief context**:
  - We analyzed SpaceX launches to identify factors associated with successful first-stage landings and to build a classification model that predicts them.
- **Summary of methodologies**:
  - **Data acquisition**: SpaceX API + web scraping (Wikipedia)
  - **Data cleaning and preparation (wrangling)**: category standardization, target variable creation, and one-hot encoding.
  - **EDA**: tables and charts
  - **Interactive analytics**: map with Folium and dashboard with Plotly Dash
  - **Prediction**: multiple classifiers; comparison by accuracy and confusion matrix for the best model
- **Summary of all results**:
  - Overall success ≈ 67%
  - By site, KSC LC-39A ≈ 77% (best performance)
  - Upward success trend (≈ 90% in 2019)
  - Payloads 3-6 t show higher success than >6 t in this dataset
- **Implications**:
  - Launch site and payload range have a practical impact on recovery probability; orbit also provides signal (mind the sample size)

Introduction

- Project background and context:
  - SpaceX aims to maximize first-stage recovery to reduce costs. Using historical launch data, we investigate which variables (launch site, payload mass, orbit, booster version, flight number, etc.) influence success and how results have changed over time.
- Objective:
  - Identify the factors associated with a successful first-stage landing and evaluate a classification model to predict it
- Research questions:
  - Which **factors** are associated with a higher probability of a **successful first-stage landing**?
  - How do **payload mass (kg)** and **orbit** affect the outcome?
  - Are there differences by **launch site**?
  - How has the **success rate** evolved overtime?
  - Can we **predict** success with a **reliable classification model**?
- Target variable definition
  - Successful first-stage landing (**1 = success**, **0 = failure**).

Section 1

Methodology

Methodology

Methodology Overview

- Data collection methodology:
  - SpaceX REST API (v4) + Wikipedia web scraping (Python: `requests`, `pandas`, `BeautifulSoup`).
- Data wrangling & processing:
  - Standardized categories; engineered target `Class` (`1` = successful landing if Outcome starts with “True”, else `0`); one-hot encoding for categorical features; basic handling of missing values
- EDA (SQL & visualization):
  - SQLite queries for success rates by `site` / `orbit` and yearly trend. Charts for success by `launch site`, payload bands, and temporal trend.
- Interactive visual analytics:
  - **Folium** map of launch sites; **Plotly Dash** dashboard with filters (`site`, `payload`)
- Predictive analysis:
  - Built and compared classifiers (**SVM**, **Decision Tree**, **Logistic Regression**, **KNN**); **hyperparameter tuning** via **GridSearchCV**; evaluation with `accuracy`, confusion matrix, and classification report.

Data Collection

- **Source (modeling):** SpaceX REST API (v4)
- **Period & scope:** Falcon 9 (2010 – 2020); 90 records → dataset_part1/2/3.csv
- **Tools:** Python (*`requests`*, *`pandas`*)
- **Key fields:** *`Date`*, *`FlightNumber`*, *`LaunchSite`*, *`Orbit`*, *`PayloadMass`*, *`Outcome`*
- **Separate artifact:** Wikipedia scraped table (121 rows) – not merged (kept for exploration/demonstration)

Reproducibility: Code & data available – modeling dataset (SpaceX API, 90 records): *`dataset_part_1.csv`*, *`dataset_part_2.csv`*, *`dataset_part_3.csv`*; separate artifact (scraped, 121 rows): *`spacex_web_scraped.csv`*.
Notebooks/Dashboard: GitHub

Data Collection – SpaceX API

- **SpaceX REST calls**
  - *GET /v4/launches/past* → *response.json* → *pd.json_normalize()*
  - Fields kept: *rocket*, *payloads*, *launchpad*, *cores*, *flight_number*, *date_utc*
  - Filter: **1 core & 1 payload** per launch; parse *date_utc* → *date*
  - Time window: *date* ≤ *2020-11-13*
- **Per-record enrichment**
  - *GET /V4/rockets/{id}* → **BoosterVersion**
  - *GET /V4/launchpads/{id}* → **LaunchSite**, **Latitude**, **Longitude**
  - *GET /V4/payloads/{id}* → **PayloadMass**, **Orbit**
  - *GET /V4/cores/{id}* → **Block**, **ReusedCount**, **Serial**, **Flights**, **GridFins**, **Reused**, **Legs**, **LandingPad**, **Outcome**
- **Final steps**
  - Keep **Falcon 9** only; **fill** *PayloadMass* with **mean**
  - **Reset** *FlightNumber = 1..N*
  - **Export**: *dataset_part_1.csv* (**90 rows**)
- **Notebook (GitHub):** GitHub

Define helper functions

- *getBoosterVersion*
- *getLaunchSite*
- *getPayloadData*
- *getCoreData*

Request launch data from the SpaceX API

- Get the initial *response*
- Decode the response content as JSON
- Keep the columns we will use
- Filter to "simple" launches
- Normalize the date
- Restrict the time range

Create a new DataFrame

- Run the helper functions to enrich (*rocket*, *payload*, *launchpad*, *cores*)
- Filter the DataFrame to include only Falcon 9 launches
- Renumber FlightNumber as 1..N after filtering to Falcon 9
- Replace missing values in *PayloadMass* with the mean
- Export: *dataset_part_1.csv* (**90 rows**)

Data Collection - Scraping

- **Static URL (Wikipedia revision):**
  - `https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922`
- **Request & parse:**
  - `resp = requests.get(static_url, timeout=15); resp.raise_for_status()` → `response = resp.content`
  - `soup = BeautifulSoup(response, 'html.parser')`
- **Locate tables:**
  - `soup.find_all('table', 'wikitable plainrowheaders collapsible')`
- **Row parsing (loop):**
  - Use the header cell `th` for **Flight No.**; parse other cells with helpers
- **Build & export:**
  - Append values into `launch_dict` → `pd.DataFrame(launch_dict)` → `df.to_csv('spacex_web_scraped.csv', index=False)` **(121 rows)**

Define helper functions

- `date_time`
- `booster_version`
- `landing_status`
- `get_mass`
- `extract_column_from_header`

Fetch HTML from the given URL

- Use `requests.get(...)` to obtain a response
- Create a `BeautifulSoup` object
- Extract column names from the table header

Build a DataFrame from the parsed HTML

- Build a dictionary using the helper functions
- Create a DataFrame from that dictionary
- Export to CSV: `spacex_web_scraped.csv` **(121 rows)**
- **Notebook (GitHub):** GitHub

Data Wrangling

- **Input**: *dataset_part_1.csv* **(90 rows)**
- **Missing values:** *df.isnull().sum() / len(df) * 100*
- **Types:** *df.types()* (identify numerical vs categorical)
- **Launch counts (by site):** *df['LaunchSite'].value_counts()*
- **Orbit x Outcome (counts & %):** *df.loc[df['Orbit'] != 'GTO', 'Orbit'].value_counts()* → *df.loc[df['Orbit'] != 'GTO', 'Orbit'].value_counts(normalize=True)*
- **Enumerate landing outcomes:** *df['Outcome'].value_counts()* → define *bad_outcomes*
- **Create target:** build *landing_class* (*0* if outcome = *bad_outcomes* else *1*) *df['Class'] = landing_class*
- **Export:** *df.to_csv('dataset_part_2.csv', index=False)* **(90 rows)**
- **Notebook (GitHub URL):** GitHub

Data analysis

- Identify and compute % of missing values
- Identify numeric vs categorical columns

Calculations

- Launches by site
- Orbit counts & percentages (excluding GTO for these tables)
- Mission outcomes by orbit: counts & percentages

Create new column

- From *Outcome*, define *bad_outcomes*
- Build target *Class*
- Compute success rate using *Class*
- Export: *dataset_part_2.csv*

EDA with Data Visualization 1

- In this section we explore the correlations between these pairs:
  - **Flight Number vs Payload Mass** (*sns.catplot*, *hue=Class*)
    - **Why: quick view of how payload mass and launch sequence relate to landing outcome**
  - **Flight Number vs Launch Site** (*sns.catplot* / *strip*, *hue=Class*)
    - **Why: compare success patterns across sites along the flight sequence**
  - **Payload Mass vs Launch Site** (*sns.scatterplot*, *hue=Class*)
    - **Why: check whether heavier payloads behave differently by launch site**
  - **Success rate by Orbit** (*sns.barplot* **of** *mean(Class) * 100* **by** *Orbit*)
    - **Why: categories bar chart is appropriate to compare success proportions**
  - **Flight Number vs Orbit** (*sns.scatterplot*, *hue=Class*)
    - **Why: inspect outcome variation across orbits over time (proxied by flight number)**

EDA with Data Visualization 2

- **Payload Mass vs Orbit (*sns.scatterplot*, *hue=Class*)**
  - **Why: visualize the interplay between mass and orbit type on success**
- **Yearly success trend (*sns.lineplot* of *mean(Class) * 100* by year from *Date*)**
  - **Why: reveal temporal trend in landing success**

- Also we did feature engineering in this lab
  - Input: *dataset_part_2.csv*
  - Categorical encoding: for categorical columns → *features_one_hot = pd.get_dummies(features, columns=['Orbit', 'LaunchSite', 'LandingPad', 'Serial'], dtype=int)*
  - Cast to *float64* DataFrame *features_one_hot*
  - Output: *dataset_part_3.csv* (90 rows, 80 columns)
- **Notebook (GitHub URL)**: GitHub

EDA with SQL - Summary

- **Setup**: `%sql sqlite:///my_data1.db`; `DROP TABLE IF EXISTS SPACEXTABLE;` → `CREATE TABLE SPACEXTABLE AS SELECT * FROM SPACEXTBL WHERE Date IS NOT NULL;`
- **Inventory & Sample**: `DISTINCT Launch_Site;` preview rows for `Launch_Site LIKE 'CCA%'` (`LIMIT 5`)
- **Aggregations**: `SUM(PAYLOAD_MASS__KG_)` for `'NASA (CRS)'`; `AVG(PAYLOAD_MASS__KG_)` for `'F9 v1.1%'`
- **Milestone**: earliest ground-pad success `MIN(date(Date))` where `Landing_Outcome = 'Success (ground pad)'`
- **Conditional filter**: booster versions with **drone-ship success** and **4-6 t** payload
- **Outcomes**:
  - overall **success vs failure** counts (`CASE WHEN...`);
  - **landing outcome** distribution ranked by count
- **Extremes**: booster version at **maximum payload mass** (subquery on `MAX`)
- **Time slice: 2015** failed **drone-ship** landings by **month** (`substr(Date, 6, 2)`)
- **Notebook (GitHub URL)**: GitHub

Build an Interactive Map with Folium

- **Base map**:
  - `folium.Map(location=[29.55968, -95.08310])` – centered initially at **NASA JSC.**
- **Objects added & why**:
  - **Circle** (`folium.Circle`) at NASA JSC + **Popup** – highlight the reference location and label it
  - **Text marker** (`folium.Marker` with `DivIcon`) at NASA JSC – readable site name on the map
  - **MarkerCluster** (`folium.plugins.MarkerCluster`) – group many launch markers to keep the map clear
  - **Launch markers** (`folium.Marker` with `folium.Icon`) – one per launch row; **icon color from** `class` (`green` success / `red` failure) for instant outcome reading; popup shows *Launch Site – Success/Failure*
  - **Mouse position** (`folium.plugins.MousePosition`) – live `lat`/`long` under the cursor for inspection
  - **Small site dots** (`folium.CircleMarker`) – precise location of each launch site
  - **Polylines** (`folium.PolyLine`) from each **launch site** to nearby targets – **train, city, highway, coast** – to visualize proximity
  - **Distance labels** (`folium.Marker` with `DivIcon`) **at each target – show distance in** km; popup with target label and coordinates
- **Notebook (GitHub URL)**: GitHub

Build a Dashboard with Plotly Dash – Summary 1

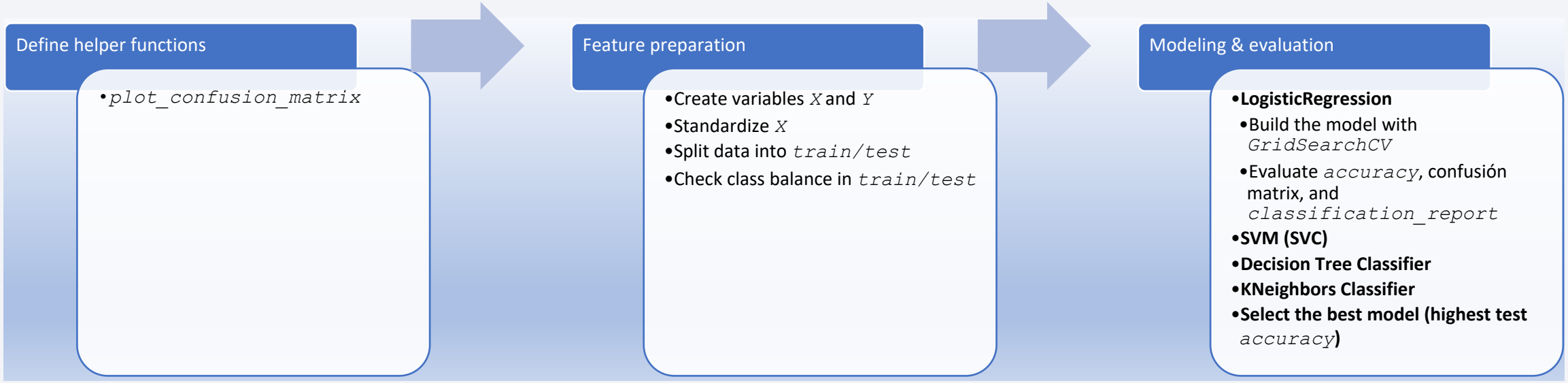
- **Plots**
  - **Success Pie** (`px.pie`)
    - **All sites**: total successful launches by launch site (`sum(class)` by `Launch Site`)
    - **Selected site**: success vs. failure counts for that site
  - **Payload vs. Success Scatter** (`px.scatter`)
    - **x:** `Payload Mass (kg)` / **y:** `class` **(0/1)**
    - **color:** `Booster Version Category`
    - **hover**: `Launch Site`, `Payload Mass (kg)`
- **Interactions**
  - **Dropdown** (`site-dropdown`): switches between **All Sites** (default) and a **specific site** for both charts
  - **RangeSlider** (`payload-slider`, 0-10,000 kg): filters the scatter plot by payload range

Build a Dashboard with Plotly Dash – Summary 2

- **Why these plots & interactions**
  - **Pie:** quickly compare success distribution across sites, or success vs. failure within one site
  - Scatter + slider: explore the relationship between payload mass and landing outcome, and see how it varies by booster version and site; the slider focuses the analysis on relevant mass ranges.
- **Dash app (GitHub URL)**: GitHub

Predictive Analysis (Classification) 1

- **Data & split**
  - Input features: *dataset_part_3.csv* (90x80); target *Y = Class*
  - Standardize features: *StandardScaler.fit_transform(X)* → *X_scaled*
  - Split: *train_test_split(X_scaled, Y, test_size=0.2, random_state=2, stratify=Y)*



Predictive Analysis (Classification) 2

- **Models trained (each with 10-fold CV via** *GridSearchCV*, *scoring='accuracy'*)
  - **Logistic Regression (***LogisticRegression***): grid** *C=[0.01, 0.1, 1]*, *penalty='l2'*, *solver='lbfgs'*
  - **SVM (***SVC***): grids over** *kernel = {linear, rbf, poly, sigmoid}*, *C* **&** *gamma* **on logspace**
  - **Decision Tree (***DecisionTreeClassifier***): grid on** *criterion*, *splitter*, *max_depth*, *max_features*, *min_samples-leaf*, *min_samples_split*
  - **KNN (***KNeighborsClassifier***): grid on** *n_neighbors*, *algorithm*, *p*
- **Evaluation & model selection**
  - For each best-CV model: compute **test accuracy** (*.score(X_test, Y_test)*)
  - Plot **confusion matrix**(*plot_confusion_matrix*) and print **classification report** (precision, recall, f1, balanced accuracy)
  - **Select best-performing model** = highest **test accuracy**
- **Notebook (GitHub URL)**: GitHub

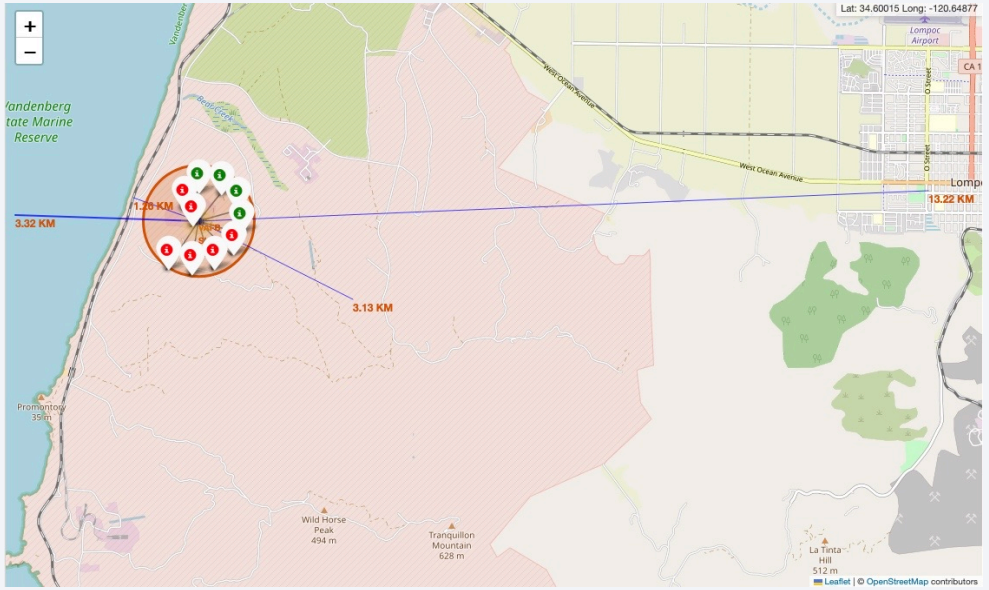
Results – Exploratory data analysis (EDA)

- **Overall landing success** (*Class=1*): **~66.7%** (from *dataset_part_2.csv*)
- **By launch site**: KSC LC-39A ~77%, VAFB SLC-4E ~77%, CCAFS SLC-40 ~60% (success rate)
- **Yearly trend**: upward improvement peaking at ~90% in 2019
- **By payload band**: ≤1 t ~75%, 1-3 t ~68%, 3-6 t ~62%, 6-10 t ~65% (success rate)
- **By orbit (note small-n effects):** several orbits show **100%** in this sample (e.g., **SSO**, **ES-L1**, **GEO**, **HEO**), while **GTO** ~52%, **ISS** ~62%, **LEO** ~71%

> These figures come from the Lab 3 / 5 datasets (*dataset_part_2.csv*) and match the visuals summarized in Slide 11.

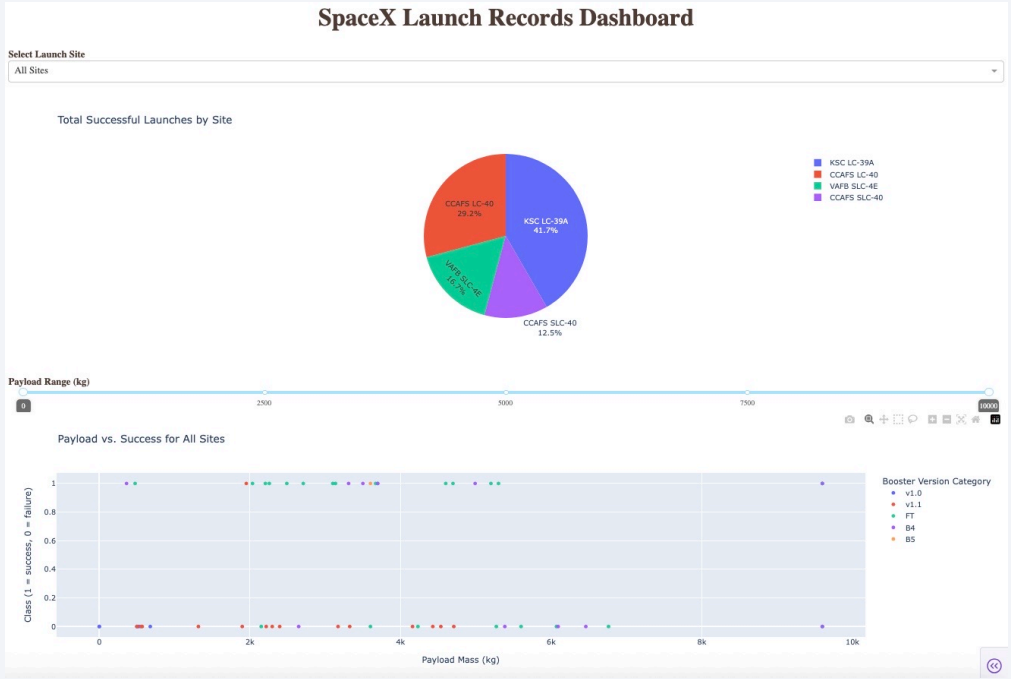
Results – Interactive analytics 1

- **Folium map (Lab 6)**
  - Map centered near NASA JSC with **site markers** (*`Marker`*, *`CircleMarker`*).
  - **Polylines** to nearest **city/train/highway/coast**, and **distance labels**

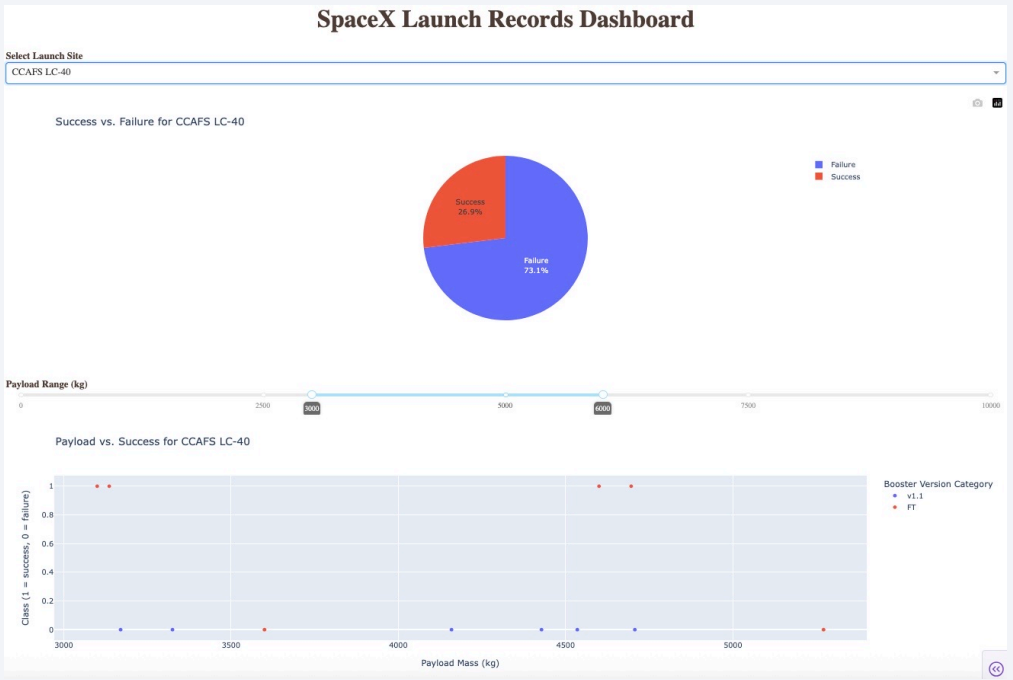


Results – Interactive analytics 2

- **Dash dashboard (Dash app (Lab))**
  - **Success pie (All Sites)** and **success vs. failure (Selected Site)**
  - **Payload vs. Success scatter** filtered with the **payload slider** and **site dropdown**



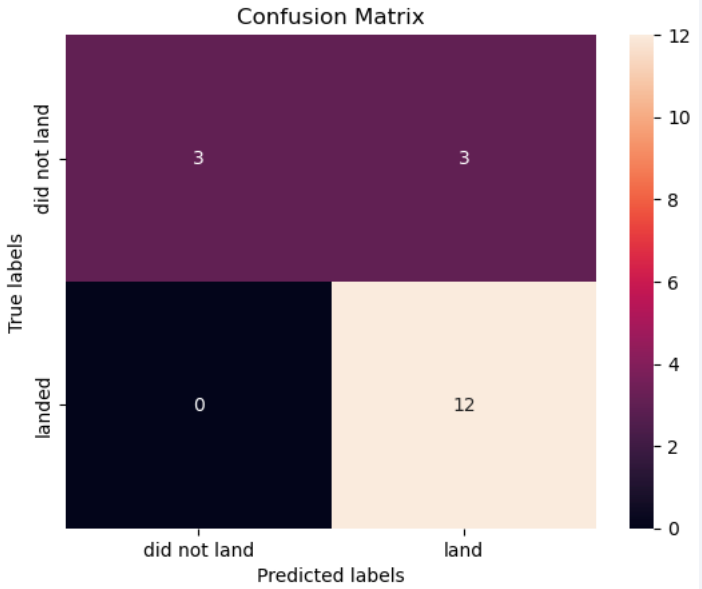
Launch Site: *All Sites* Payload Range (kg): *0 – 10 t*



Launch Site: *CCAFS LC-40* Payload Range (kg): *3 – 6 t*

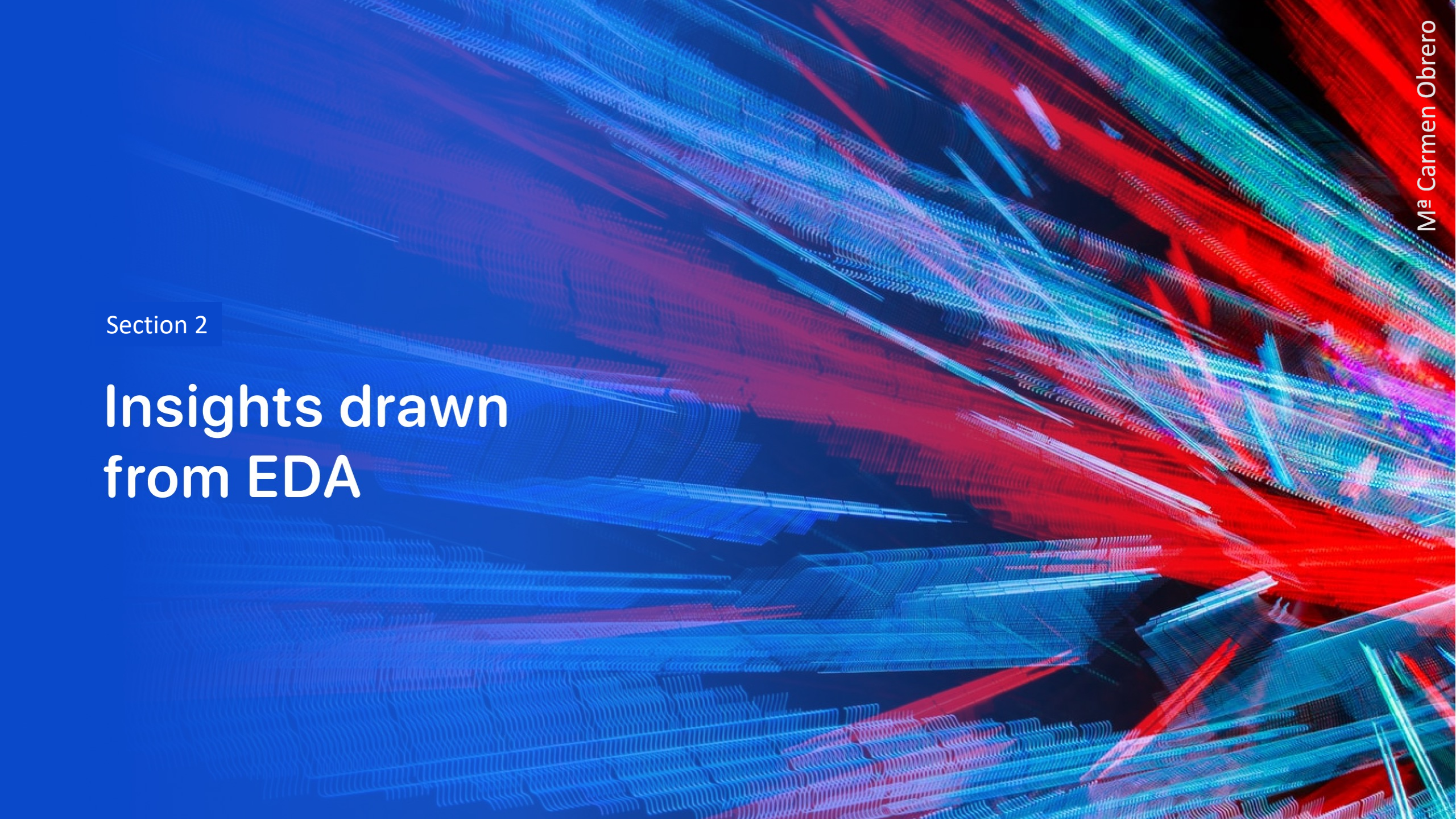
Results – Predictive analysis (Classification)

- **Process:** standardized features → **train/test split (80/20 = 72/18, stratified)** → **GridSearchCV** (`cv=10`, `scoring='accuracy'`) on **Logistic Regression, SVM (SVC), Decision Tree, KNN** → evaluate on **test**
- **What to report:**
  - **Best model(s): Logistic Regression and SVM (sigmoid)** – both Test Accuracy = 0.833 and with same confusion matrix (see images)
- **Takeaway:** LR and SVM (sigmoid) provide the **highest test accuracy** among the four; performance aligns with patterns observed in EDA (`site`, `payload`, `orbit`).



- **Notebooks and Dash app (GitHub URL):** [GitHub]

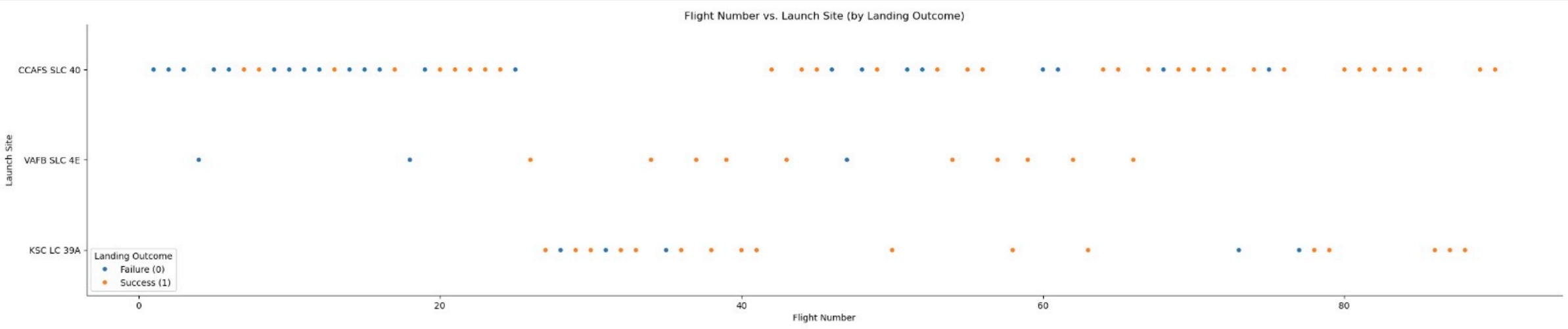
Model	Train Accuracy (CV-mean, 10-fold)	Test Accuracy	Precision (0)	Recall (0)	F1 (0)	Precision (1)	Recall (1)	F1 (1)	Macro F1	Weighted F1
Logistic Regression	0.850	**0.833**	1.000	0.500	0.667	0.800	1.000	0.889	0.778	0.815
SVM (sigmoid)	0.864	**0.833**	1.000	0.500	0.667	0.800	1.000	0.889	0.778	0.815
Decision Tree	0.932	0.778	0.667	0.667	0.667	0.833	0.833	0.833	0.750	0.778
KNN (k=4, p=1)	0.864	0.778	0.667	0.667	0.667	0.833	0.833	0.833	0.750	0.778



Section 2

Insights drawn from EDA

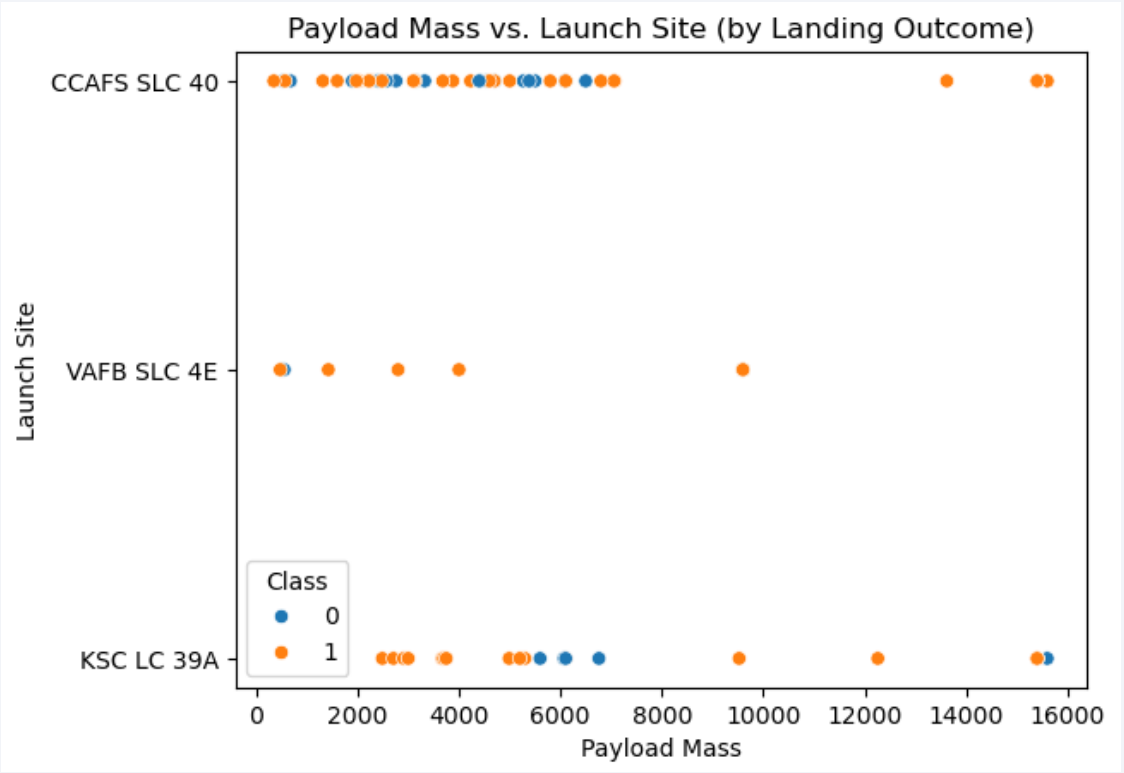
Flight Number vs. Launch Site



- **Chart spec**
  - Catplot/strip plot: x = *FlightNumber*, y = *LaunchSite*, hue = *Class* (1 = successful landing, 0 = failed landing)
- **How to read it**: each dot = a launch; color encodes the landing outcome
- **Key patterns**:
  - **Later flights skew to success** (more *Class=1* at higher *FlightNumber*)
  - **LC-39A** concentrates successes; **SLC-40** is mixed; **SLC-4E** is sparse (small-n)
  - **Context**: sites were activated at different times, which partly explains the "later = better" pattern.

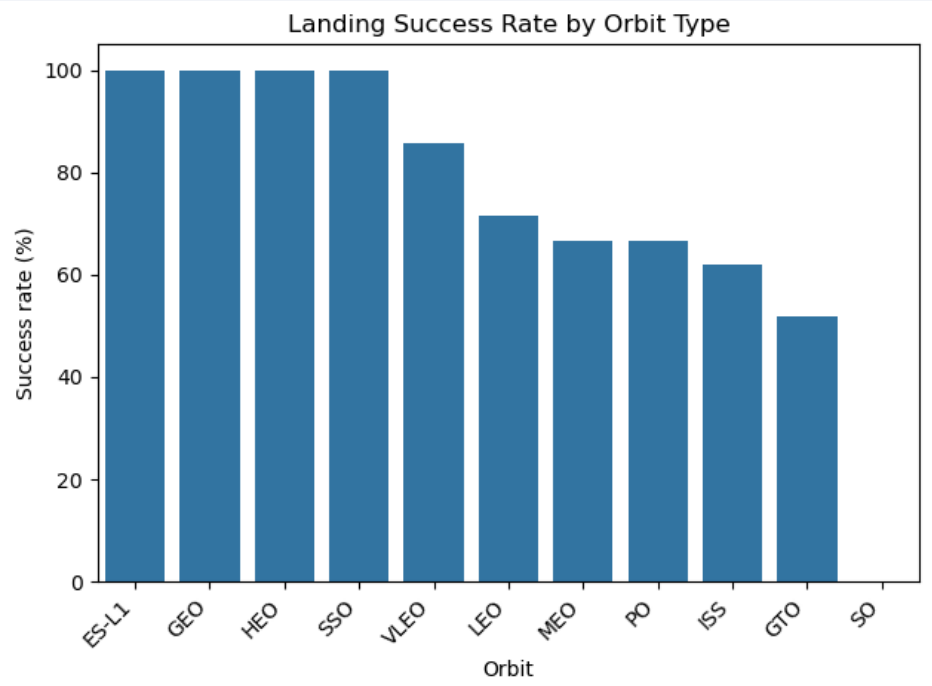
Payload vs. Launch Site

- **Chart spec**
  - Scatter/strip plot: x = *`Payload Mass (kg)`*, y = *`LaunchSite`*, hue = *`Class`* (*1* = successful landing, *0* = failed landing)
- **How to read it:**
  - Each dot = a launch; color encodes the **landing outcome**
- **Key patterns:**
  - **No strict monotonic effect**: successes and failures appear across the payload range (mid-range masses look more favorable ≈3-6 t)



  - **By site:** **LC-39A** clusters many successes; **SLC-40** shows a more **mixed** pattern; **SLC-4E** is **sparse** (small-n)
  - A few **high-payload outliers** do **not** systematically change the outcome

Success Rate vs. Orbit Type



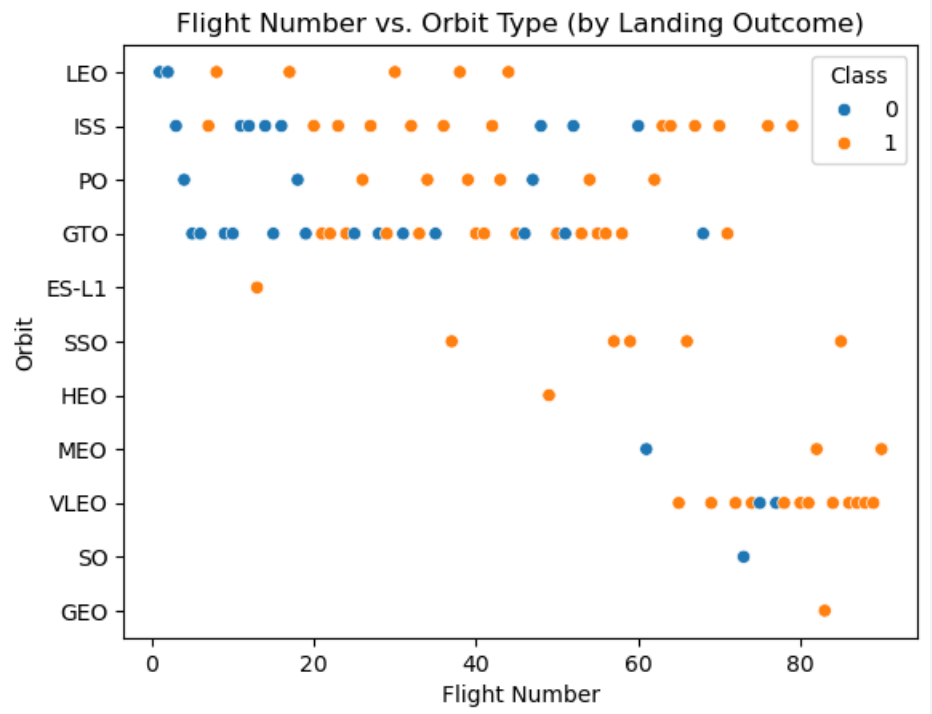
- **Chart spec:**
  - **Bar chart of mean (`Class`) * 100 by `Orbit` (success rate in %)**
  - **Order bars descending**
- **How to read it:**
  - Each bar height = **success rate** (%) for that orbit; label shows the %
- **Key patterns**:
  - **Highest**: **SSO 100%** (count=5); **VLEO 85.7%** (count=14); **LEO 71.4%** (count=7)

Orbit	count	success_rate
ES-L1	1	1.000
GEO	1	1.000
HEO	1	1.000
SSO	5	1.000
VLEO	14	0.857
LEO	7	0.714
MEO	3	0.667
PO	9	0.667
ISS	21	0.619
GTO	27	0.519
SO	1	0.000

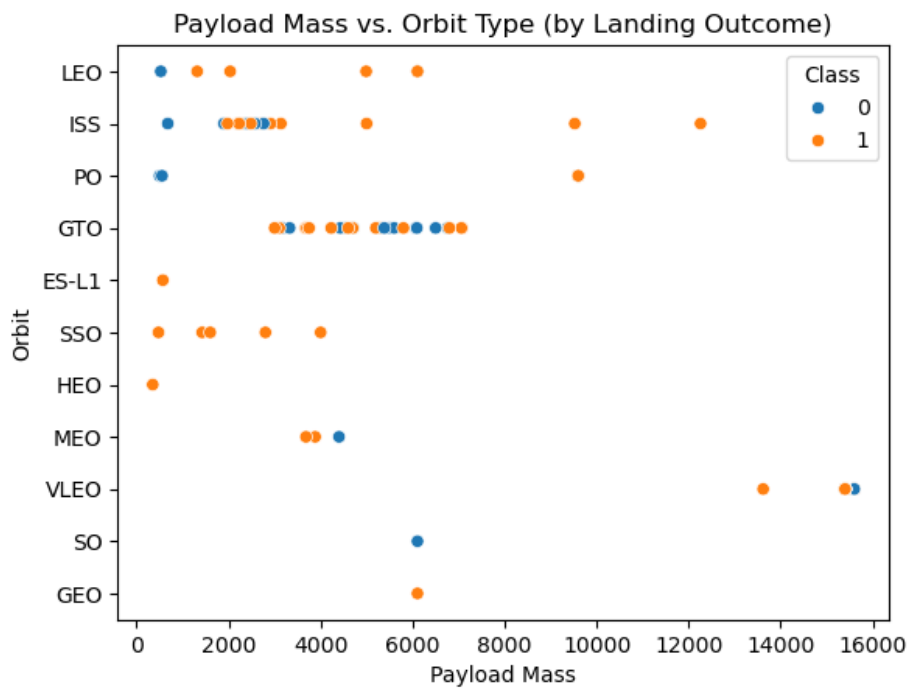
- **Mid**: **MEO 66.7%** (count=3); **PO 66.7%** (count=9); **ISS 61.9%** (count=21)
- **Lower**: **GTO 51.9%** (count=27) — largest sample
- **Edge cases (small-n)**: **ES-L1 100%** (count=1), **GEO 100%** (count=1), **HEO 100%** (count=1), **SO 0%** (count=1) → interpret with caution.

Flight Number vs. Orbit Type

- **Chart spec:**
  - Catplot/strip: x = *`FlightNumber`*, y = *`Orbit`*, hue = *`Class`* (*`1`* = successful landing, *`0`* = failed landing)
- **How to read it:**
  - Each dot = a launch; color encodes the **landing outcome**
- **Key patterns:**
  - **Later flights skew to success** across orbits
  - **GTO**: success improves from **early** → **late** flights
  - **ISS**: **modest** improvement
  - PO: **roughly stable** across periods
  - VLEO: appears only in later flights; **LEO** appears only in earlier flights – reflects when those orbits were flown
  - **Small-n orbits** (e.g., **SSO, GEO, HEO, ES-L1, SO)** show isolated points → interpret cautiously



Payload vs. Orbit Type

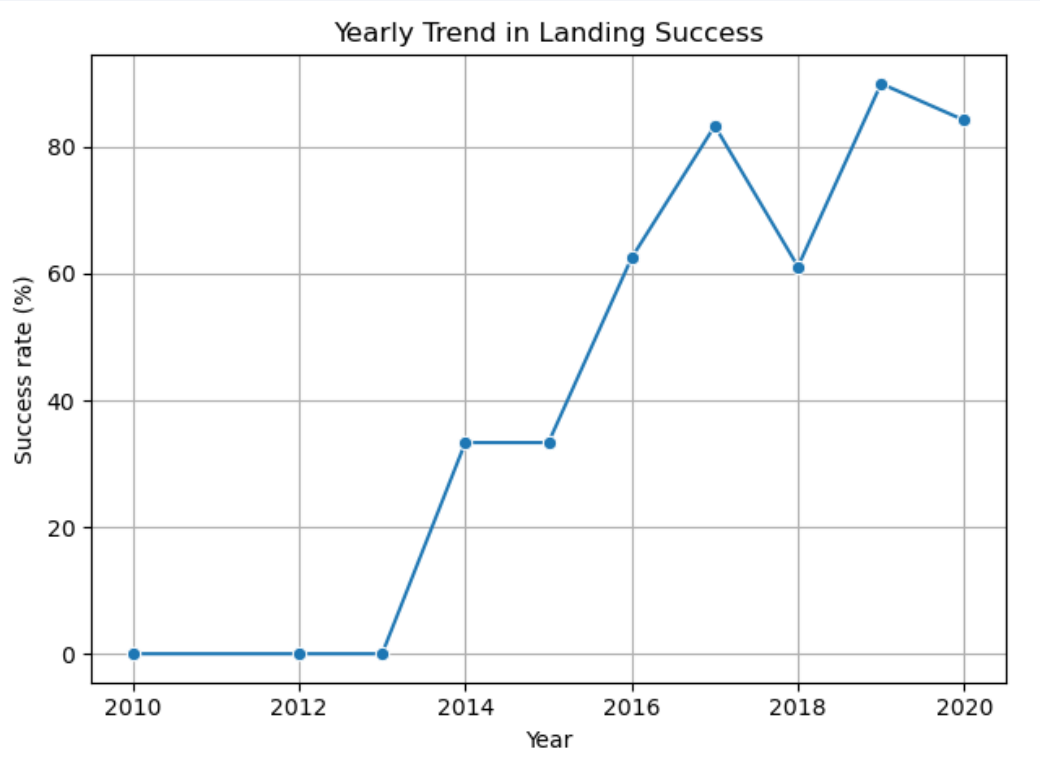


- **Chart spec:**
  - Catplot/strip: x = *Payload Mass (kg)*, y = *Orbit*, hue = *Class* (*1* = successful landing, *0* = failed landing)
- **How to read it:**
  - Each dot = one launch; color encodes the **landing outcome**
- **Key patterns:**
  - **GTO**: success declines with heavier payloads

- **ISS / LEO**: **mid range payloads** (≈ 1-6 t) show **moderate hight** success
- **VLEO**: only **> 10 t** payloads in this sample, with **high** success
- **Small-n orbits** (e.g., **SSO, GEO, HEO, ES-L1, SO**) → interpret cautiously
- **Overall:** GTO shows more failures at higher masses, while mid-range payloads are comparatively favorable in several orbits

Launch Success Yearly Trend

- **Chart spec**:
  - Line chart of **mean(**`Class`**) * 100 by** `Year` (success rate, %), with point markers.
- **How to read it:**
  - The line shows the **annual landing success rate.**



- **Key patterns**
  - **2010-2013: 0%** success
  - **2014-2016:** steady improvement to ≈ 62% in **2016**
  - **2017: ≈ 82%**
  - **2018:** dip to **≈ 60%**
  - **2019:** peak **≈ 90%**
  - **2020:** remains high at **≈ 84%**
- **Overall:** the yearly landing success shows a clear upward trend (peaking in 2019 and staying high in 2020)

All Launch Site Names (SQL)

- **Query:**

```
%%sql
SELECT DISTINCT Launch_Site AS launch_site
    FROM SPACEXTABLE
    ORDER BY launch_site;
```

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

- **Result (as shown in the image):**
  - **CCAFS LC-40 / CCAFS SLC-40** (Cape Canaveral LC-40 (Florida))
  - **KSC LC-39A** (Kennedy LC-39A (Florida))
  - **VAFB SLC-4E** (Vandenberg SLC-4E (California))
- **Brief explanation:**
  - The query returns **four distinct labels** from *Launch_Site*
  - ***CCAFS LC-40*** and ***CCAFS SLC-40*** are **two label variants for the same Cape Canaveral pad (SLC-40).** In this project we **did not normalize** them, so the dashboard shows **all four** labels
  - **Implication:** site-level counts/plots may **split Cape Canaveral launches** across the two labels

Launch Site Names Begin with 'CCA' (SQL)

- **Query**

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

- **Result:**
  - Rows where *Launch_Site* **starts with** *'CCA'* (Cape Canaveral labels)
  - In this sample, the **first 5 matches** are all *CCAFS LC-40* (2010-2013), with early missions whose *Landing_Outcome* is *Failure (parachute)* or *No attempt*

- **Brief explanation:**
  - *LIKE 'CCA%'* applies a **prefix filter** (*%* = any trailing characters)
  - *LIMIT 5* returns **only five** matching rows
  - We use this filter to **quickly subset Cape Canaveral launches** for inspection

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

Total Payload Mass (SQL)

- **Query:**

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

total_payload_mass
45596

- **Result:**
  - *total_payload_mass* = 45,596 kg
- **Brief explanation:**
  - We **aggregate** *PAYLOAD_MASS__KG_* for all rows where *Customer = 'NASA (CRS)'*
  - *SUM* returns the **cumulative payload mass** (in kg) for CRS missions **present in this table** (project time window)
  - Note: *SUM* **ignores NULLs**; rows with **0 kg** are included as zero

Average Payload Mass by F9 v1.1 (SQL)

- **Query**:

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

avg_payload_mass
2534.6666666666665

- **Result**:
  - *avg_payload_mass* = 2,534.67 kg (shown as *2534.6666…*)
- **Brief explanation**:
  - *AVG* computes the **mean** of *PAYLOAD_MASS__KG_* for rows where *Booster_Version* **starts with** *'F9 v1.1'* (the *%* is the **wildcard** for any suffix/sub-variant)
  - The average is in **kilograms**; *AVG* **ignores NULLs**, while **0 kg** rows (if any) are included as zero
  - The displayed decimals come from the exact arithmetic; for reporting we **round to 2,534.67 kg**

First Successful Ground Landing Date (SQL)

- **Query**:

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

first_ground_success
2015-12-22

- **Result:**
  - *First_ground_success* = *2015-12-22*
- **Brief explanation**:
  - Filters rows to **ground-pad landing successes** and returns the **earliest calendar date**
  - **date(Date)** normalizes the string to **YYYY-MM-DD** before applying *MIN*
  - The value reflects the **first such event recorded in this table** (i.e., within the dataset's time window)

Successful Drone Ship Landing with Payload between 4000 and 6000 (SQL)

- **Query**:

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

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

- **Result**:
  - **F9 FT B1022** — Success (drone ship) — 4696 kg
  - **F9 FT B1026** — Success (drone ship) — 4600 kg
  - **F9 FT B1021.2** — Success (drone ship) — 5300 kg
  - **F9 FT B1031.2** — Success (drone ship) — 5200 kg
- **Brief explanation**:
  - Filters **successful drone-ship landings** with payload mass **strictly between 4,000** and **6,000** kg (> and < are **exclusive**)
  - Returns **4 rows** in this dataset; all are **Falcon 9 FT** boosters in that mass band.
  - If you needed an **inclusive** range, use *BETWEEN 4000 AND 6000* or *>= 4000 AND <= 6000*.

Total Number of Successful and Failure Mission Outcomes (SQL)

- **Query**:

```
%%sql
SELECT
  SUM(CASE WHEN Mission_Outcome LIKE 'Success%'   THEN 1 ELSE 0 END) AS success_missions,
  SUM(CASE WHEN Mission_Outcome LIKE 'Failure%'  THEN 1 ELSE 0 END) AS failure_missions
FROM SPACEXTABLE;
```

success_missions	failure_missions
100	1

- **Results:**
  - *success_missions* = 100
  - *failure_missions* = 1
- **Brief explanation**:
  - We count rows where *Mission_Outcome* **starts with** *'Success'* or *'Failure'* (*%* is the wildcard), so labels like *Success (drone ship)* are included
  - **Other outcome categories** (e.g., *Partial Failure*, *Prelaunch Failure*, *No attempt*) are **excluded** by design – this slide reports **only** explicit successes and failures.
  - These are **counts** (not percentages)

Boosters Carried Maximum Payload (SQL)

- **Query:**

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

- **Results**:
  - All returned rows have *PAYLOAD_MASS__KG_* = **15,600 kg**
  - Boosters include several **Falcon 9 Block 5** units (see image)
- **Brief explanation**:
  - The **subquery** gets the **global maximum** payload mass in the table
  - The **outer query** returns **every booster** whose recorded payload mass equals that maximum; *DISTINCT* prevents duplicate booster/payload pairs
  - Several boosters appear because **multiple missions** share the **same top mass (15,600 kg)** in this dataset

2015 Launch Records (SQL)

- **Query:**

```
%%sql
SELECT
    substr(Date, 6, 2) AS Month,
    Landing_Outcome,
    Booster_Version,
    Launch_Site
    FROM SPACEXTABLE
    WHERE substr(Date, 0, 5) = '2015'
    AND Landing_Outcome = 'Failure (drone ship)'
    ORDER BY substr(Date, 6, 2), Date;
```

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

- **Results:**
  - See image
- **Brief explanation:**
  - Filters to **year 2015** (*substr(Date, 0, 5) = '2015'*) and to **drone-ship landing failures only**
  - Extracts the **month** as *substr(Date, 6, 2)* (MM) and **orders chronologically** by month (then date)
  - In this dataset, there are **two matching records**, both at **Cape Canaveral LC-40** with **Falcon 9 v1.1** boosters

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

- **Query**:

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

Landing_Outcome	number_outcome
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

- **Results**:
  - See ranked table (right)
- **Brief explanation:**
  - *date(Date)* casts timestamps to a **date** (YYYY-MM-DD) and *BETWEEN* is **inclusive** of both endpoints
  - We **aggregate** counts by *Landing_Outcome* and **rank** them with *ORDER BY number_outcome DESC*
  - Counts reflect **only this time window** (early Falcon 9 era), hence the high number of **No attempt** outcomes

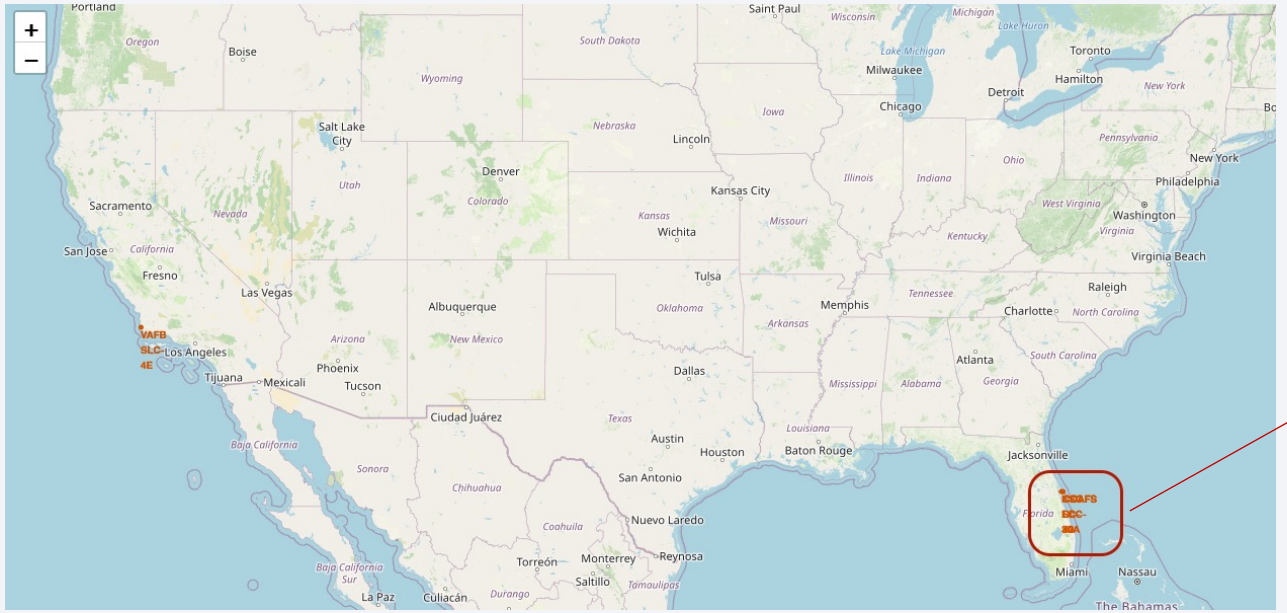
Section 3

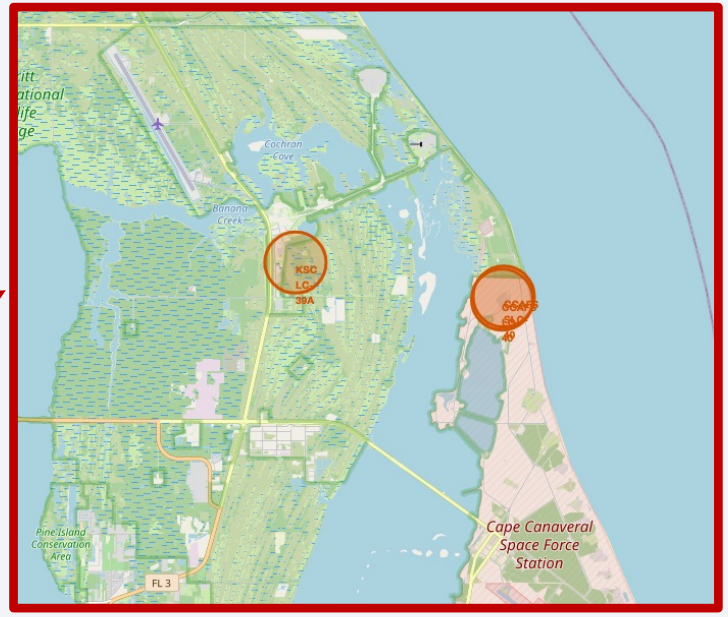
Launch Sites Proximities Analysis

Mª Carmen Obrero

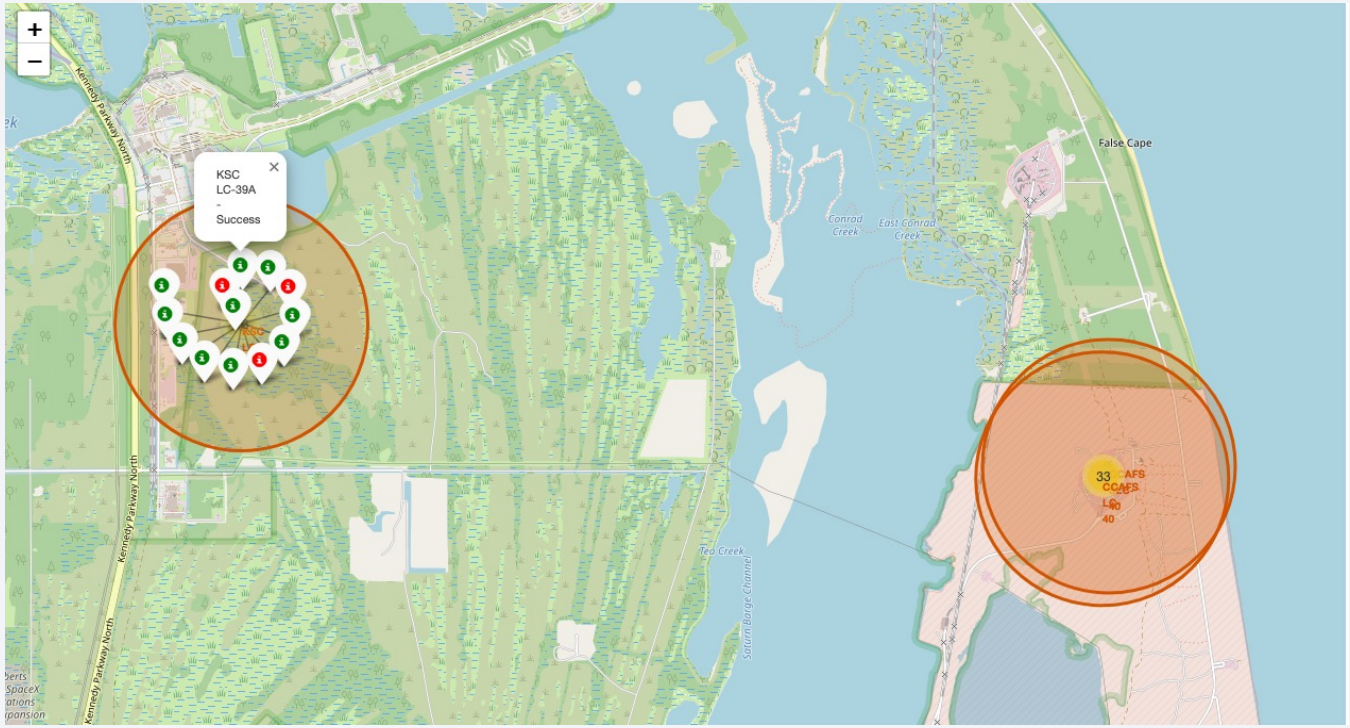
SpaceX Launch Sites – Global Map (Folium)

- **Markers** show the **four launch sites** used in this project:
  - **KSC LC-39A, CCAFS LC-40, CCAFS SLC-40 (Cape Canaveral, Florida)**
  - **VAFB SLC-4E (Vandenberg, California)**
- At global zoom the **three Florida pads are nearly co-located and overlap** (task1 uses **no MarkerCluster**). The **inset** zoom separates them and shows their exact placement on Cape Canaveral.
- All sites are **coastal**, consistent with range-safety corridors and downrange recovery operations (drone ship / ground pad)
- In the interactive map, **clicking a marker** shows the site name.





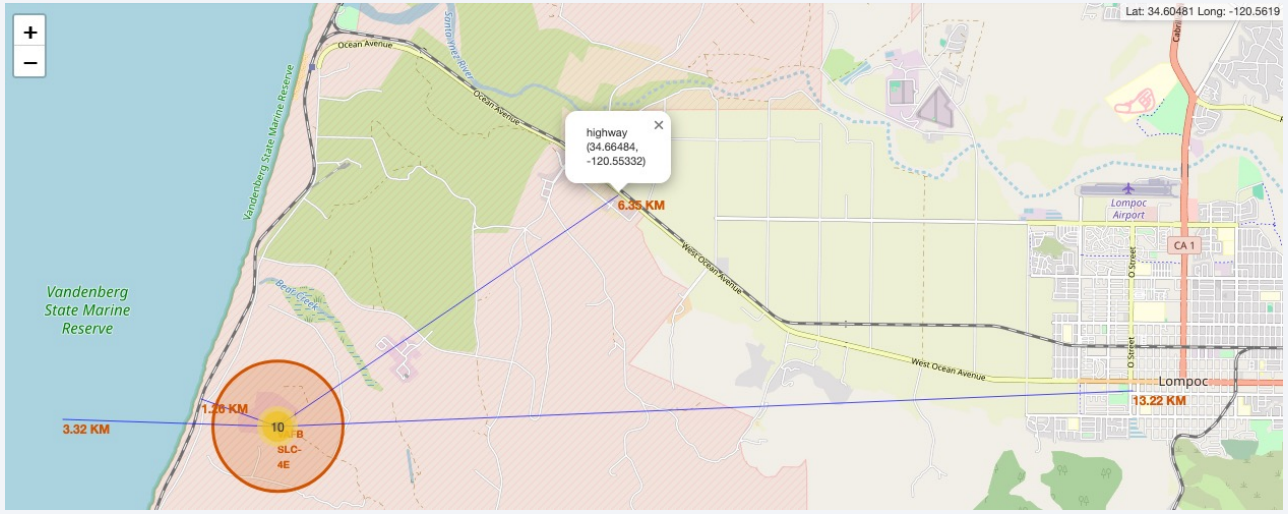
Launch Outcomes at Cape Canaveral– Color-coded Map (Folium)

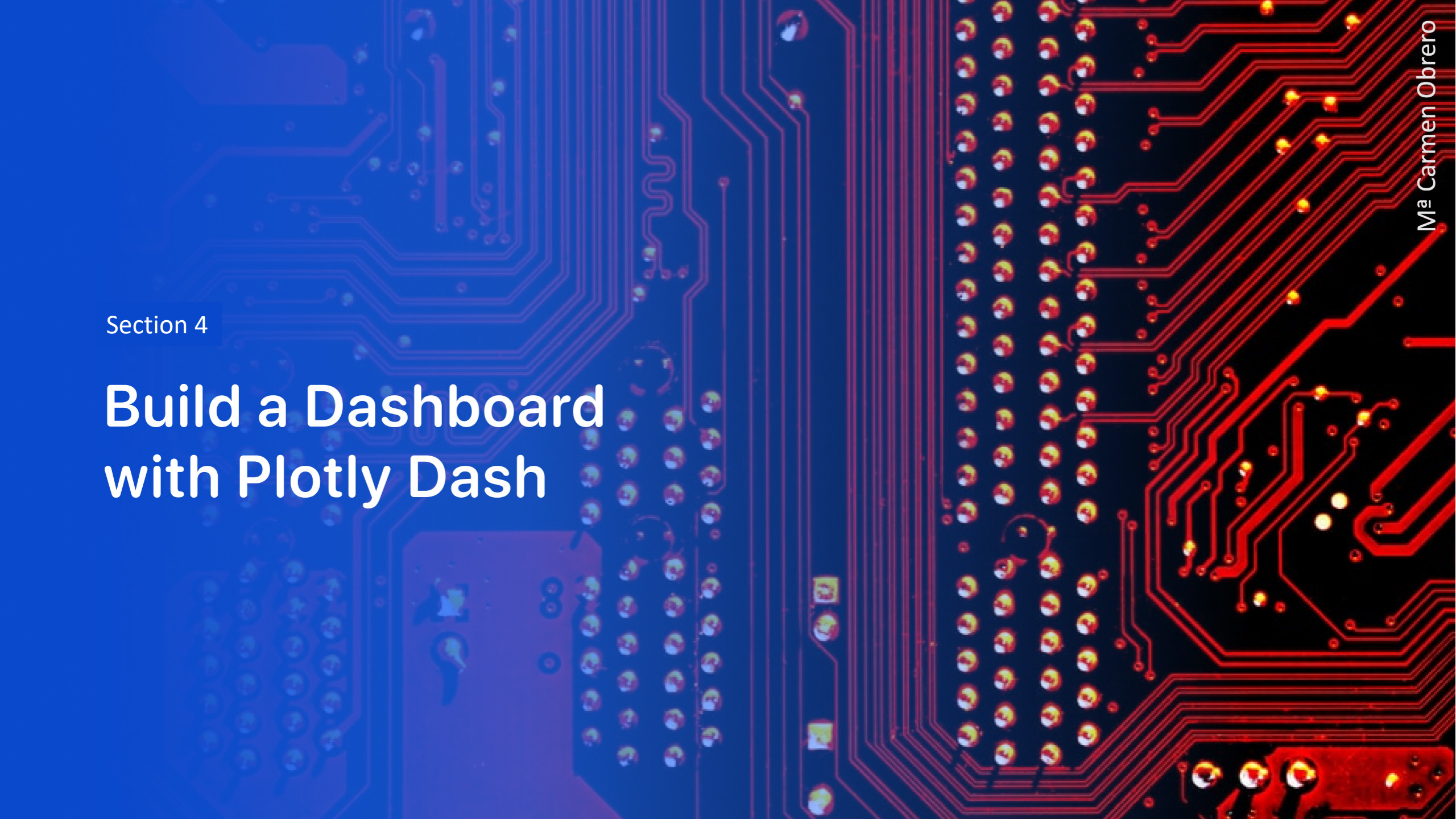


- **What the screenshot shows**:
  - Points = launches; color = outcome (green = success, Class=1; red = failure, Class=0). Clicking a marker shows site & outcome
  - **Left:** launches around **KSC LC-39A** (mostly green)
  - **Right:** a **cluster** over CCAFS LC-40 / CCAFS SLC-40; zooming reveals the individual colored pins
  - Orange site circles mark the pad areas: all pads are **coastal**, consistent with range-safety corridors and downrange recovery
- **Findings in this view**:
  - **Success dominates** at Cape Canaveral (green clearly outnumbers red)
  - **LC-40** appears **most active** in this sample; **LC-39A** shows a high success rate with occasional failures
- **Scope note**:
  - This crop focuses on **Florida**; **VAFB SLC-4E** (not shown here) has fewer launches in this period and also shows a majority of successes
- **Overall:**
  - Greens dominate at **LC-39A**; the Cape pads show mixed early results; **Vandenberg** has fewer, mixed records

Vandenberg SLC-4E – Distances to Coast, Train, Highway & city (km) (Folium)

- **What the screenshot shows:**
  - Orange circle = **SLC-4E** launch pad
  - Blue lines with **km labels** (straight-line from the pad) to:
    - **Rail line:** 1.26 km (west)
    - **Coastline:** ≈ 3.32 km (west)
    - **Highway (CA-1 / West Ocean Ave corridor):** ≈ 6.35 km (NE)
    - **Lompoc city center (≈ 13.22 km)** (east)
- **Key findings**:
  - Very close to **rail (≈1.3 km)** and **coast (≈3.3 km)** → good logistics and over-ocean trajectories/downrange recovery.
  - **CA-1 access (≈6.3 km)** provides reliable ground transport
  - The **~13 km standoff** to Lompoc provides a reasonable **safety buffer** from populated areas



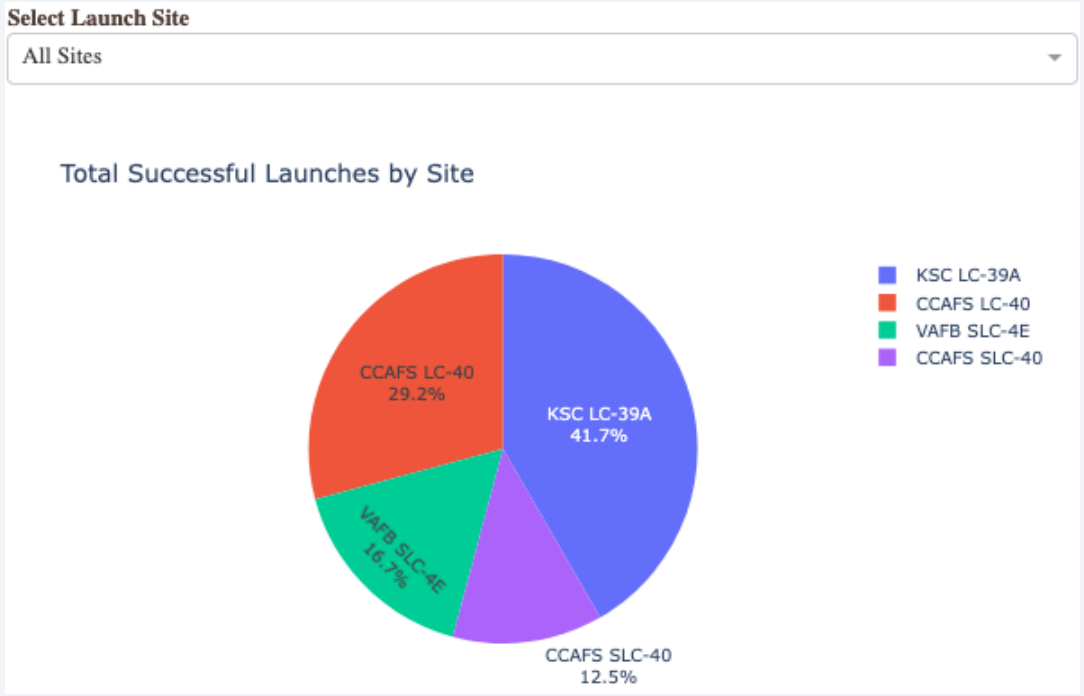


Section 4

Build a Dashboard with Plotly Dash

Mª Carmen Obrero

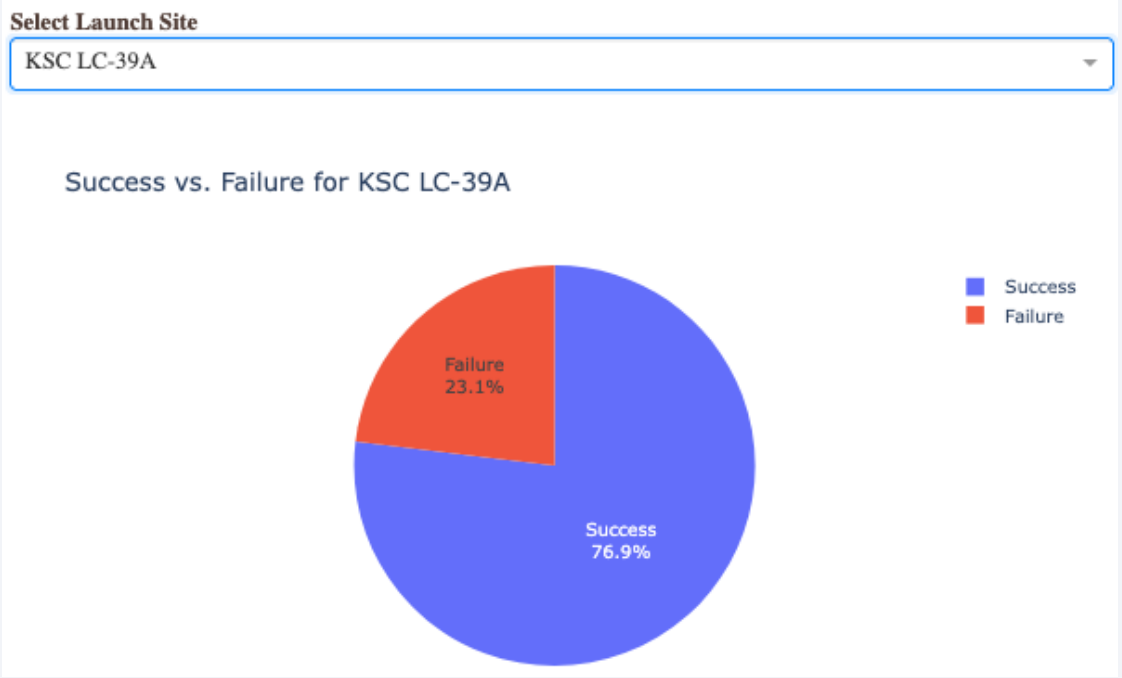
Launch Success Count by Site (All Sites – Dash Pie Chart)



- **What this shows**: Distribution of **successful launches** (sum of `Class == 1`) **grouped by launch site**
- **How to read it**: Slice size = **number of successes** for that site (not a success rate); colors match the legend
- **Key takeaways**:
  - **KSC LC-39A: ~41.7% — largest slice, CCAFS LC-40: ~29.2% — second largest, VAFB SLC-4E ~16.7%, CCAFS SLC-40 ~12.5% — smallest slice**
  - Overall, **Cape Canaveral dominates**: LC-39A + LC-40 + SLC-40 ≈ **83%** of all successes in this dataset (LC-40 and SLC-40 are shown separately because that's how the dataset labels them)
- **Note:**
  - This chart shows **count, not success rates**; proportions reflect how many successful missions each site has, not the percentage success at that site
  - Dropdown = **All Sites**; the pie summarizes successes across all sites

Site with Highest Success Ratio – KSC LC-39A (Dash Pie Chart)

- **What this shows**:
  - Pie chart of **Success vs. Failure** at the **selected site** (dropdown set to **KSC LC-39A**)
  - Values are **ratios within this site** (not counts across sites)
- **How to read it:**
  - Blue = **Success (Class=1)**, Red = **Failure (Class=0)**
  - Slice labels show each class's **percentage of LC-39A launches**
- **Key finding**
  - **KSC LC-39A ≈ 76.9% success** vs. **23.1% failure** — the **highest success ratio** among sites in this dataset (VAFB SLC-4E is similar but with fewer launches, so interpret ties cautiously)
- **Note:**
  - Changing the dropdown recomputes the ratios; keep it on KSC LC-39A to match this slide



Payload vs. Landing Outcome by Payload Range (Dash Scatter)

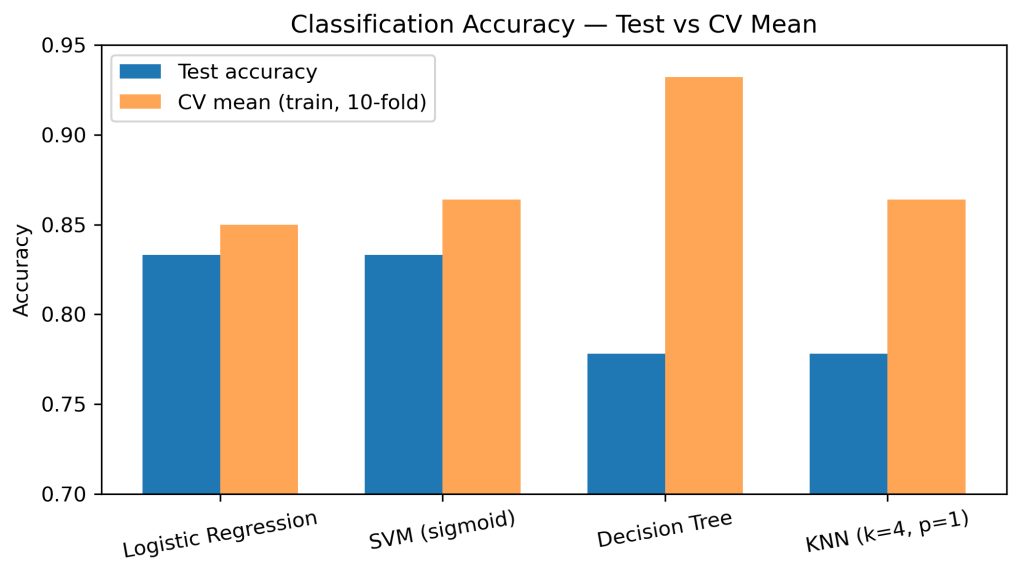
- **What this shows:** Scatter of **all sites** — x = *`Payload (kg)`*, y = *`Class`* (*1*=success, *0*=failure), color = *`Booster version Category`*. The **range slider** filters the payload band.
- **Screenshots:** Left **0–10,000 kg** (full range). Right = **3,000–6,000 kg** (focused mid-range)
- **Insights**: Success points cluster in the **~2–6 t** band; very **low** and **very high** payloads show a more mixed outcome. Filtering to **3–6 t** reveals a clear majority of successes.
- **Booster note**: Later boosters (**FT, B4, B5**) dominate among successes; earlier **v1.0/v1.1** show more failures at lower masses
- **Caution:** This is **outcome distribution**, not a normalized success rate by payload or by site

Section 5

Predictive Analysis (Classification)

Mª Carmen Obrero

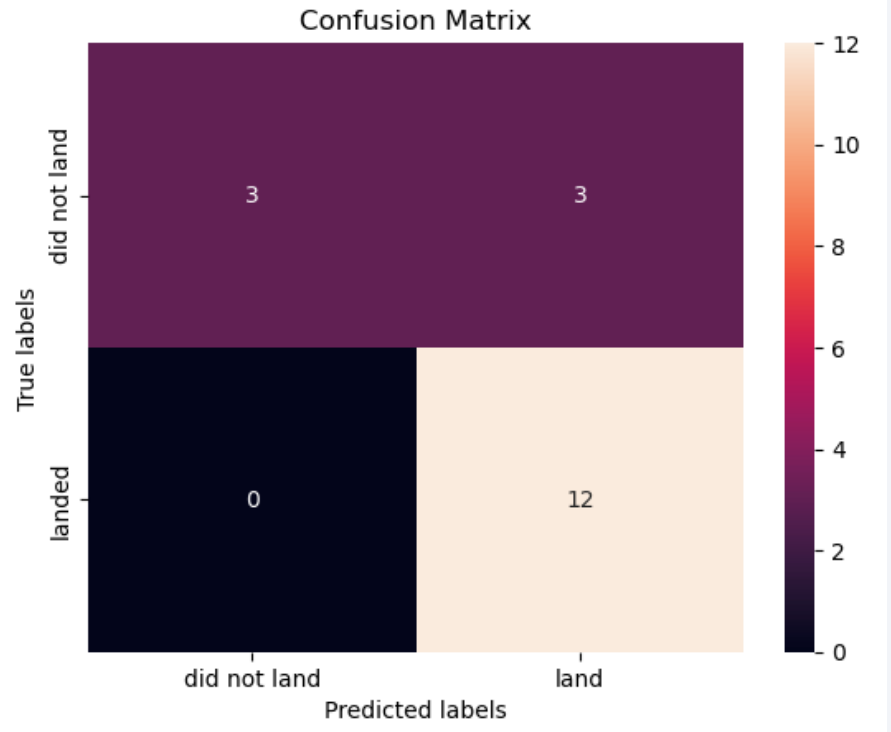
Classification Accuracy (Test vs. CV Mean)



- **What the chart shows:**
  - Bar chart comparing **test accuracy** (blue) vs. **10-fold CV mean** on training (orange) for four models: **Logistic Regression, SVM (sigmoid), Decision Tree, KNN (k=4, p=1)**
- **Numbers (test accuracy)**
  - **Logistic Regression: 0.833**
  - **SVM (sigmoid): 0.833**
  - Decision Tree: 0.778; KNN (k=4, p=1): 0.778
- **Key takeaways:**
  - **Best test accuracy (tie): Logistic Regression and SVM (sigmoid), both 0.833**
  - **Decision Tree shows the highest CV mean (0.932) but drops to 0.778 on test → overfitting**
  - **KNN** is the lowest performer **(0.778)**
- **Conclusion**:
  - Select **Logistic Regression** or **SVM (sigmoid)** as the final model; both are the most reliable and consistent with EDA patterns

Confusion Matrix – Best Models (LR & SVM-sigmoid)

- **What this shows**:
  - Rows = **true labels**; columns = **predicted labels**
  - Positive class = **land (Class=1)**; negative = **did not land (Class=0)**
- **Counts:**
  - **TP = 12** (land → land)
  - **TN = 3** (did not land → did not land)
  - **FP = 3** (did not land → land)
  - **FN = 0** (land → did not land)
- **Takeaways**:
  - The models **do not miss any landings** (FN=0) — very conservative for the positive class
  - Errors are **false alarms** (3 cases predicted "land" when it didn't), which lowers precision
  - Overall test **accuracy ≈ 0.833**, consistent with the report shown earlier
  - **Note**: **small test set (n=18)** — interpret with caution



Identical confusion matrix for Logistic Regression and SVM (sigmoid); both models produce the same test predictions

Conclusions 1

- **What we answered.**
  - Using historical Falcon 9 launches (2010–2020), we identified factors associated with first-stage landing success and built a classifier to predict it
- **Key EDA findings.**
  - **Trend**: success rates improved markedly over time.
  - **By launch site**: LC-39A led the success rate (VAFB SLC-4E close, with fewer launches)
  - **By payload mass**: the 3–6 t range showed higher success; very low/very high payloads were more mixed
  - **By orbit**: orbit type correlates with outcomes, but some categories have small sample sizes
- Best models.
  - **Logistic Regression** and **SVM (sigmoid)** tied as top performers on the hold-out test set (**accuracy = 0.833**) and produced the **same confusion matrix: TP = 12, TN = 3, FP = 3, FN = 0**
    - **Interpretation:** the model did **not miss any actual successful landings** (FN = 0) at the cost of a **few false alarms** (FP)

Conclusions 2

- **Practical implications**.
  - **Launch site** and **payload mass range** materially influence recovery odds; orbit helps, but be cautious with small-n categories
- **Limitations**.
  - Falcon 9 only, data up to 2020
  - Modest sample size (~90 records; test n ≈ 18)
  - PayloadMass imputed with the mean
  - Potential inconsistencies in site labels
  - Missing context variables (e.g., weather/sea state)
- **Next steps**.
  - Extend the time window and features (weather, sea state, mission profile); normalize site labels; calibrate probabilities and tune decision thresholds with cost-sensitive analysis; retrain and monitor with more data

Appendix 1

- **Code & artifacts**.
  - GitHub repository with notebooks, preprocessing pipeline, Folium map, and Plotly Dash app (project repository)
- **Data used**.
  - **Primary sources**: SpaceX REST API (v4) + Wikipedia web scraping
  - **Modeling dataset**: *dataset_part_1.csv*, *dataset_part_2.csv*, *dataset_part_3.csv* (≈90 rows; Falcon 9 up to 2020-11-13)
  - **Exploration table**: *spacex_web_scraped.csv* (web-scraped for EDA)
  - **SQL EDA database**: *my_data1.db*

Appendix 2

- **Notebooks/Labs (overview).**
  - Data collection (API + scraping) and export to CSV
  - Data wrangling (target creation from landing outcome, one-hot encoding, imputations)
  - EDA & visualization (by site, payload range, orbit, annual trend)
  - EDA in SQL (aggregations, filters, temporal slices)
  - Geospatial analysis with Folium (launch sites and distances)
  - Interactive dashboard with Plotly Dash (success ratios and payload-vs-class scatter)
  - Supervised classification (LR, SVM, Decision Tree, KNN) with GridSearchCV and standardized features
- **Representative SQL queries shown**.
  - Unique sites and CCAFS filters; payload-mass sums/means by customer/booster; first ground-pad success date; drone-ship success for 4–6 t; success vs failure counts; max payload by booster; monthly failures in 2015; ranking of landing outcomes (2010–2017 window)

Appendix 3

- **Reproducibility notes.**
  - Pipeline: API → enrichment/filtering (Falcon 9) → payload mass imputation (mean) → one-hot encoding → standardization (for linear models) → stratified 80/20 train/test split → GridSearchCV → evaluation (accuracy, confusion matrix, classification report)
  - Environment: Python 3.x; pandas, scikit-learn, plotly, folium
- **Acknowledgments**.
  - SpaceX REST API, Wikipedia contributors, IBM Applied Data Science Capstone template

Thank you!

Mª Carmen Obrero