Winning Space Race with Data Science

Son Ho
July-2025

Project Outline

- ❑ Executive Summary
- ❑ Introduction
- ❑ Section 1: Methodology
- ❑ Section 2: Insights drawn from EDA
- ❑ Section 3: Launch Sites Proximities Analysis
- ❑ Section 4: Build a Dashboard with Plotly Dash
- ❑ Section 5: Predictive Analysis (Classification)
- ❑ Section 6: Conclusion & Personal Reflection
- ❑ Appendix

Executive Summary

The project **explores SpaceX Falcon 9 first-stage landing** outcomes to determine which factors influence landing success and divided into **4 modules**:

- **Module 1**: Collected data via SpaceX REST API & web scraping from Wikipedia.
- **Module 2**: Data wrangling, exploratory data analysis with visualizations & SQL queries to clean & uncover trends and patterns.
- **Module 3**: Built interactive maps using Folium & dashboards with Plotly Dash provided spatial and dynamic insights.
- **Module 4**: Developed and tuned classification models (Logistic Regression, SVM, Decision Tree, KNN) to predict landing success.

➢ The models was evaluated for accuracy and interpretability, enabling informed insights for SpaceX landing missions.

➢ This report functions as a stand-alone summary of all analysis and results.

Introduction

- This analysis focuses on predicting successful landings of Falcon 9 first-stage boosters, which is critical for SpaceX's reusability strategy.
- The primary problem addressed: *What factors most influence Falcon 9 landing outcomes?*
- Key analysis questions:
  - ✓How do launch site, payload mass, orbit type, and booster version affect landing success?
  - ✓What patterns exist in successful versus failed landings?
  - ✓Which machine learning model best predicts landing outcomes?
- Understanding these questions helps support decision-making in launch planning and risk assessment.

Section 1

Methodology

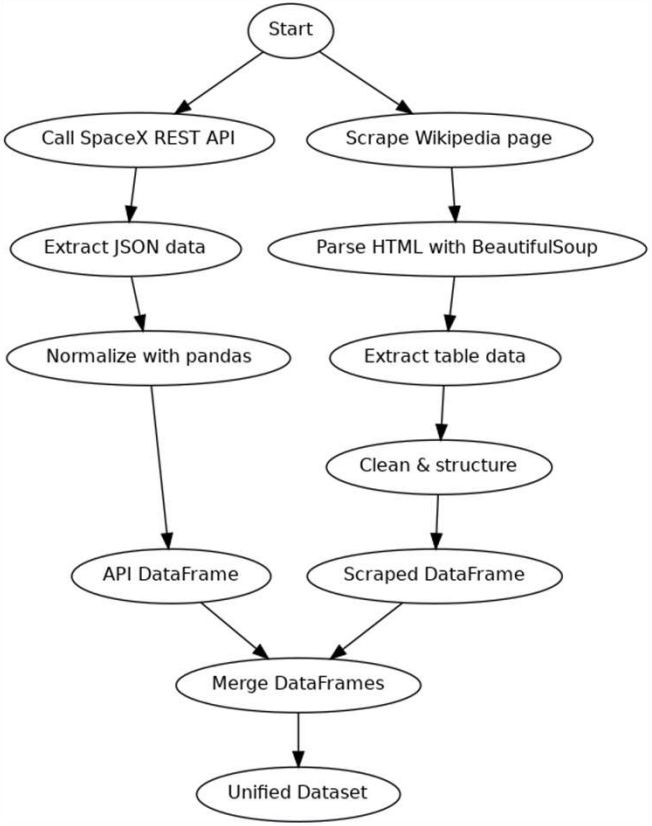
Data Collection Overview

➢ Key Activities:

- ✓Retrieved launch records using the SpaceX REST API.
- ✓Supplemented with Wikipedia web scraping for missing/extended metadata.

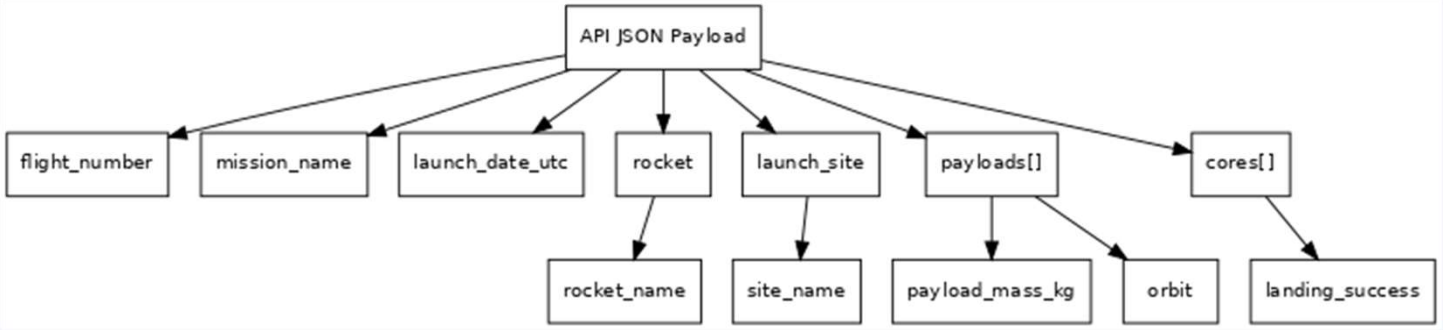
➢ Tools Used:

- ✓Python libraries: requests, BeautifulSoup, pandas
- ✓Data retrieved in JSON and HTML formats



Data Collection – SpaceX API

Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module1



➢ Details:

- ✓ Collected launch data using SpaceX REST API endpoints.
- ✓ Extracted fields: launch_site, payload_mass_kg, orbit, landing_success, etc.

➢ Key Steps:

- ✓ Used requests.get() to call the API.
- ✓ Normalized JSON data with pd.json_normalize()

Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

Check the content of the response

```
print(response.content)
```

```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,
s2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"launch":nu
utube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https
g/wiki/DemoSat"},"static_fire_date_utc":"2006-03-17T00:00:00.000Z","static
es":[{"time":33,"altitude":null,"reason":"merlin engine failure"}],"detail
b6c3bb0006eeb1e1"],"launchpad":"5e9e4502f5090995de566f86","flight_number":
30:00+12:00","date_precision":"hour","upcoming":false,"cores":[{"core":"5e
ng_success":null,"landing_type":null,"landpad":null}],"auto_update":true,"
empt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https
t":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{
d":"Lk4zQ2wP-Nc","article":"https://www.space.com/3590-spacex-falcon-1-roc
l,"static_fire_date_unix":null,"net":false,"window":0,"rocket":"5e9d0d95ed
to premature engine shutdown"}],"details":"Successful first stage burn and
ch orbit, Failed to recover first stage","crew":[],"ships":[],"capsules":[
oSat","date_utc":"2007-03-21T01:10:00.000Z","date_unix":1174439400,"date_l
```

Data Collection – Web Scraping

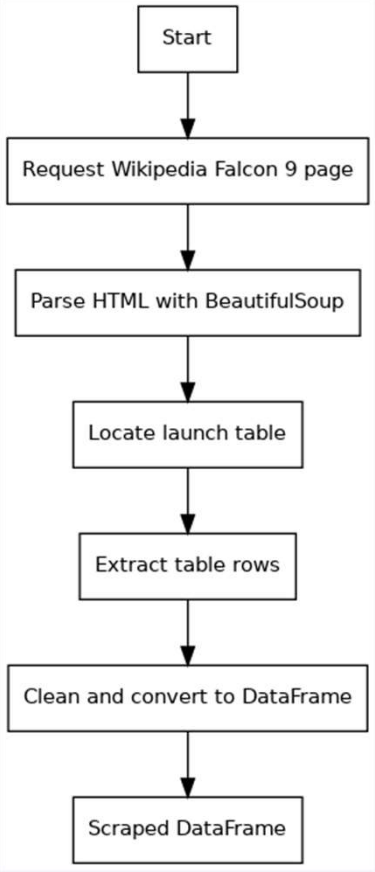
Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module1

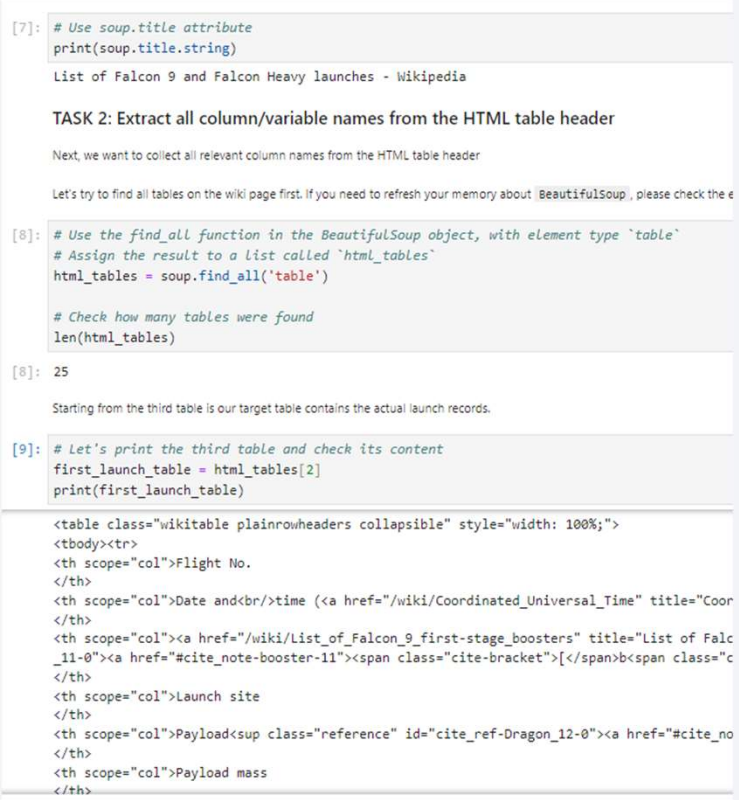
➢ Details:

- ✓Scraped Falcon 9 launch data from Wikipedia.
- ✓Augmented features like Booster Version, Landing Outcome, and Landing Pad.

➢ Key Steps:

- ✓Used requests + BeautifulSoup to extract HTML tables.
- ✓Cleaned and converted to DataFrames.





```
[7]: # Use soup.title attribute
print(soup.title.string)
```

```
List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the e

```
[8]: # Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')

# Check how many tables were found
len(html_tables)
```

```
[8]: 25
```

Starting from the third table is our target table contains the actual launch records.

```
[9]: # Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coor
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falc
_11-0"><a href="#cite_note-booster-11"><span class="cite-bracket">[</span>b<span class="c
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_no
</th>
<th scope="col">Payload mass
</th>
```

Data Wrangling

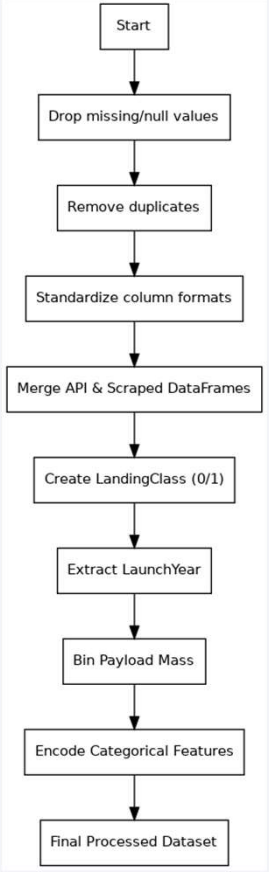
Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module1

➢ Steps Taken:

- ✓Merged API and scraped data on flight identifiers.
- ✓Removed missing values and duplicates.
- ✓Engineered features such as:
  - LandingClass (binary target)
  - PayloadMassBin, LaunchYear.

➢ Tools Used:

- ✓pandas, numpy, sklearn.preprocessing.



TASK 2: Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

```
[6]: # Apply value_counts on Orbit column
     # Calculate the number and occurrence of each orbit type
     df['Orbit'].value_counts()
```

```
[6]: Orbit
     GTO      27
     ISS      21
     VLEO     14
     PO        9
     LEO       7
     SSO       5
     MEO       3
     ES-L1     1
     HEO       1
     SO        1
     GEO       1
     Name: count, dtype: int64
```

TASK 3: Calculate the number and occurence of mission outcome of the orbits

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then ass

```
[7]: # landing_outcomes = values on Outcome column
     landing_outcomes = df['Outcome'].value_counts()
     landing_outcomes
```

```
[7]: Outcome
     True ASDS      41
     None None      19
     True RTLS      14
     False ASDS      6
     True Ocean      5
     False Ocean     2
     None ASDS       2
     False RTLS      1
     Name: count, dtype: int64
```

Exploratory Data Analysis (EDA) – Visual

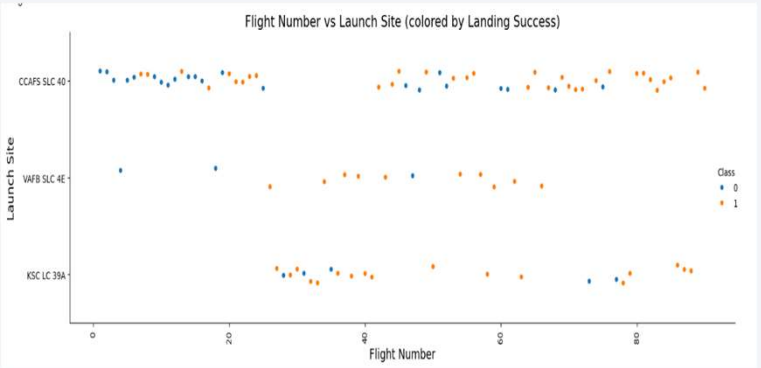
Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module2

➢ Charts Created:

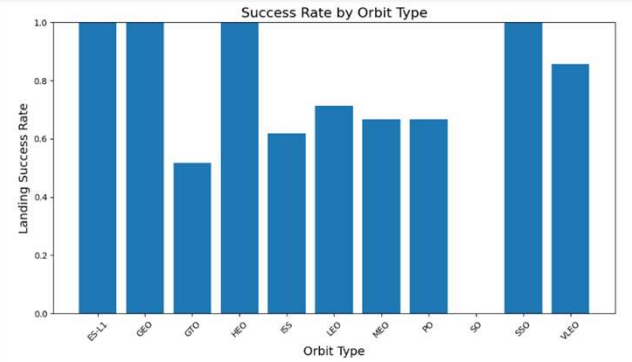
- ✓ Flight Number vs Launch Site
- ✓ Payload vs Launch Site
- ✓ Success Rate vs Orbit Type
- ✓ Launch Success Yearly Trend.

➢ Purpose:

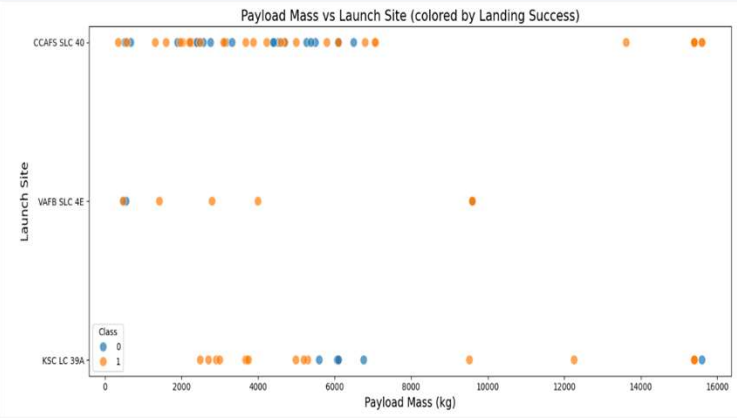
- ✓ Understand relationships between variables and landing outcomes.



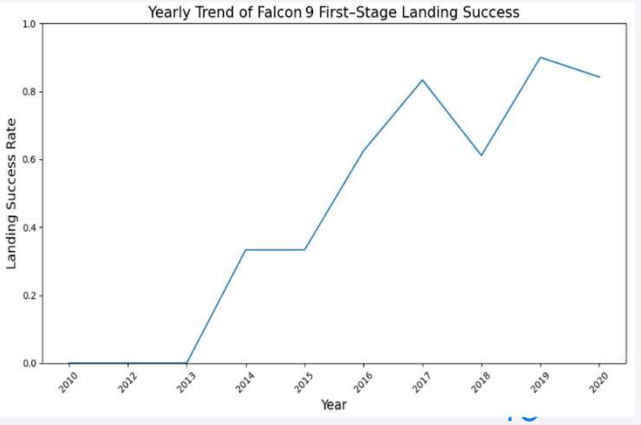
Launch frequency increased over time at key sites like CCAFS LC-40 & KSC LC-39A.



LEO & GTO orbits show highest success rates among frequently used types..



KSC LC-39A handled heavier payloads, showing strategic mission planning.



Overall landing success improved year over year, indicating operational maturity.

Exploratory Data Analysis (EDA) – SQL Analysis

Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module2

➢ Key SQL Queries:

- ✓Total payload by NASA boosters
- ✓First successful ground landing date
- ✓Average payload for F9 v1.1
- ✓Number of successful vs failed landings
- ✓Booster with max payload
- ✓Failed drone ship landings in 2015.

➢ Tool Used:

- ✓SQL via Jupyter Notebook with sqlite3 and pandas.read_sql_query.

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

```
%%sql
-- Task 3: Display the total payload mass carried by boosters launched by NASA (CRS)
SELECT SUM("Payload_Mass__kg_") AS total_payload_kg
FROM SPACEXTABLE
WHERE "Customer" = 'NASA (CRS)';
```

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

total_payload_kg
45596

Total payload

Some queries + sample output

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

```
%%sql
-- Task 4: Display average payload mass carried by booster version F9 v1.1
SELECT AVG("Payload_Mass__kg_") AS avg_payload_kg
FROM SPACEXTABLE
WHERE "Booster_Version" = 'F9 v1.1';
```

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

avg_payload_kg
2928.4

Average payload

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

Hint:Use min function

```
%%sql
-- Task 5: List the date when the first successful landing outcome in ground pad was achieved
SELECT MIN("Date") AS first_success_ground_pad
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)';
```

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

first_success_ground_pad
2015-12-22

First successful landing.

Interactive Map with Folium

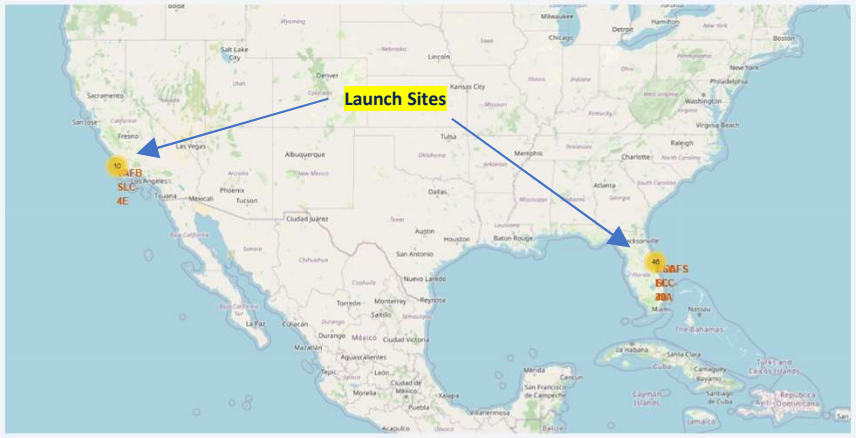
Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module3

➢ Features Visualized:

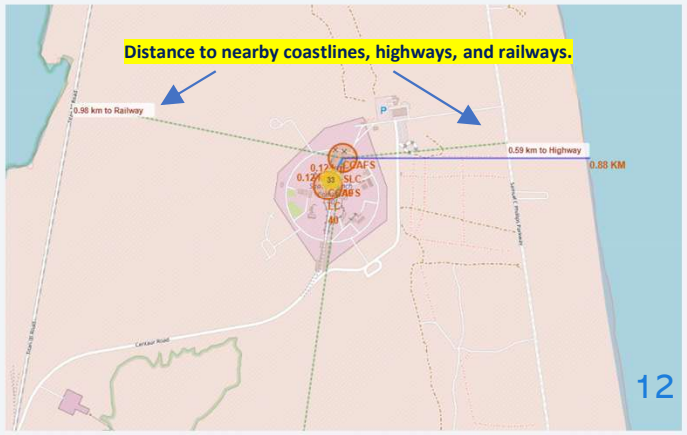
- ✓ Global launch site markers
- ✓ Launch outcomes (color-coded)
- ✓ Distance to nearby coastlines, highways, and railways.

➢ Purpose:

- ✓ Geographic pattern and accessibility of launch sites.



Some samples of interactive map

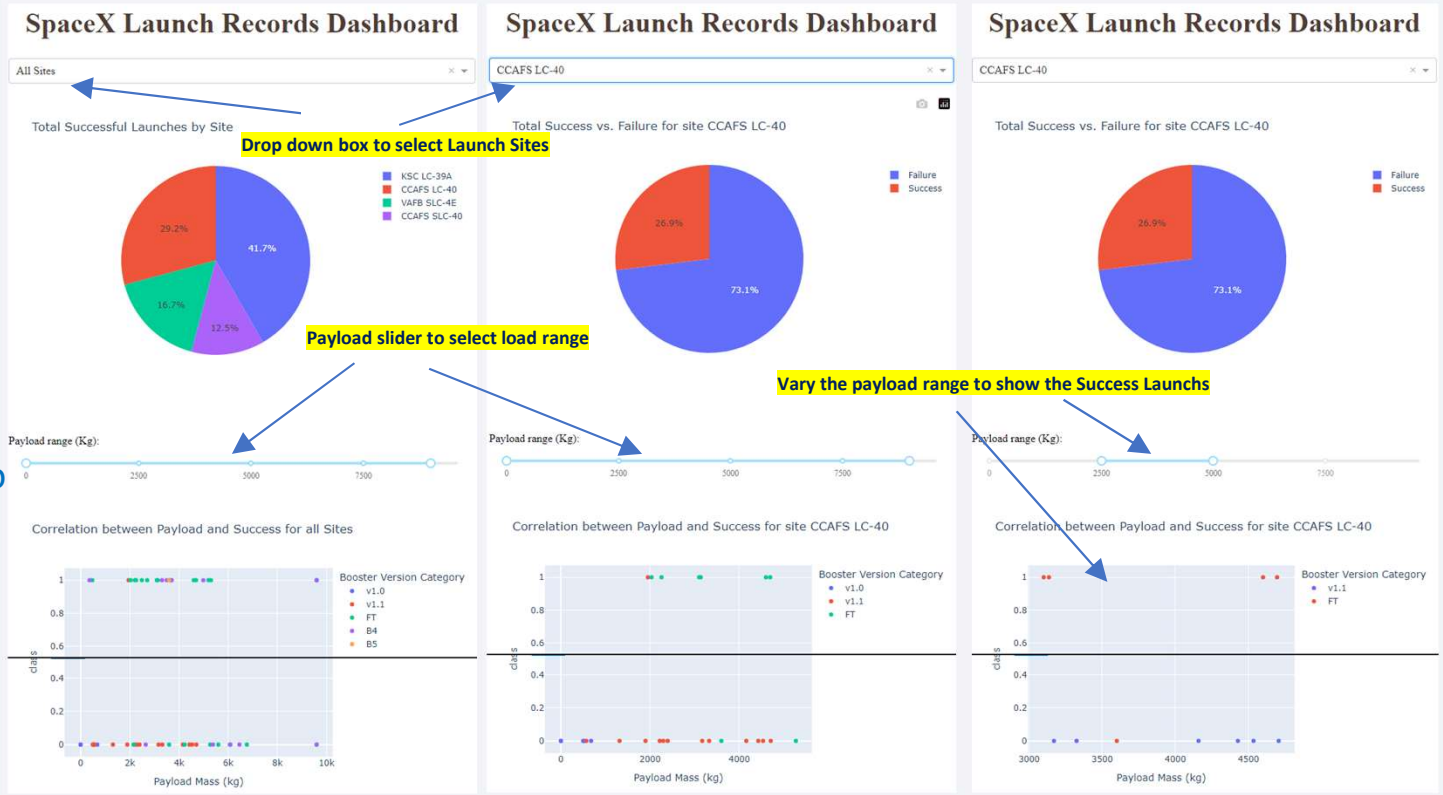


Interactive Dashboard – Plotly Dash

Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module3

Some samples of interactive Dashboard

- Dashboard Highlights:
  - ✓ Pie charts of launch success counts by site
  - ✓ Payload vs Outcome scatter plots with filter sliders
  - ✓ Interactive components to explore correlations.
- Tool Used:
  - ✓ plotly, dash, dash_core_components.



Predictive Modeling – Overview

Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module4

➢ Goal:

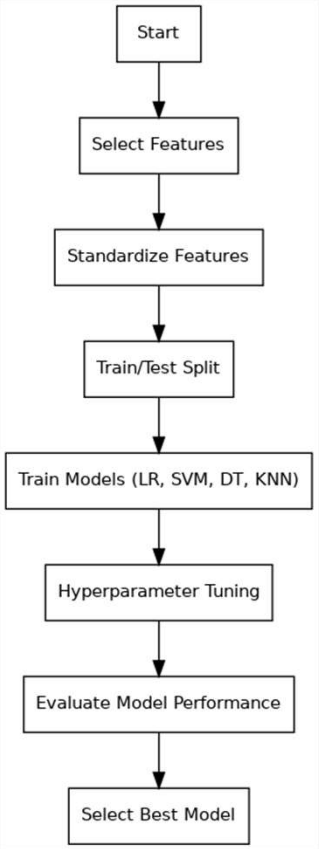
- ✓ Predict Falcon 9 landing success using classification models.

➢ Steps:

1. Feature selection and scaling
2. Model training and evaluation
3. Hyperparameter tuning.

➢ Models Used:

- ✓ Logistic Regression
- ✓ Support Vector Machine (SVM)
- ✓ Decision Tree
- ✓ K-Nearest Neighbors (KNN).



Samples Code block for Logistic Regression Model

```
print("X_test: ", X_test.shape)
print("Y_train:", Y_train.shape)
print("Y_test: ", Y_test.shape)
```

```
X_train: (72, 83)
X_test:  (18, 83)
Y_train: (72,)
Y_test:  (18,)
```

TASK 4

Create a logistic regression object then create a GridSearchCV object `logreg_cv` with cv = 10. Fit the object to find the best parameters from the

[14]:
```
parameters ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}
```

[15]:
```
# parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
# lr=LogisticRegression()
# 1. Create a base Logistic Regression estimator
lr = LogisticRegression(max_iter=1000)

# 2. Wrap it in a GridSearchCV with 10-fold CV
logreg_cv = GridSearchCV(
    estimator=lr,
    param_grid=parameters,
    cv=10,
    n_jobs=-1       # parallelize if you like
)

# 3. Fit to the training data
logreg_cv.fit(X_train, Y_train)
```

[15]:
```
GridSearchCV
best_estimator_: LogisticRegression
LogisticRegression
```

We output the `GridSearchCV` object for logistic regression. We display the best parameters using the data attribute `best_params_` and the acc

[16]:
```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

Predictive Modeling – Evaluation

Github: https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module4

➢ Evaluation Metrics:

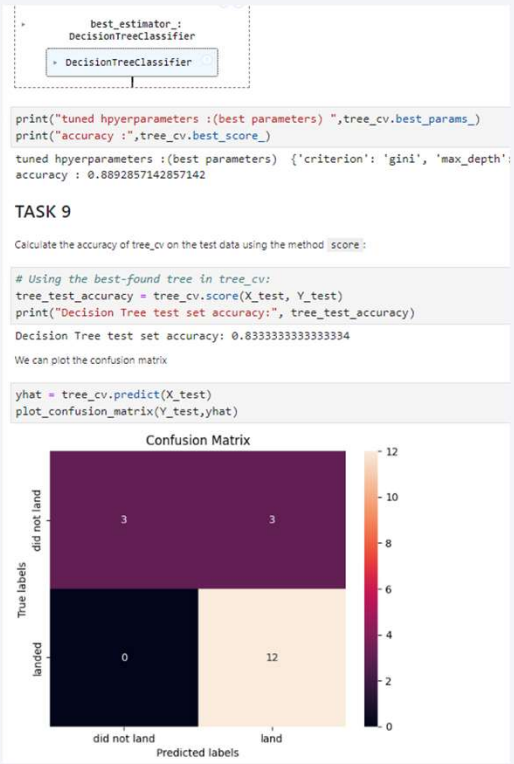
- ✓Accuracy scores
- ✓Confusion Matrix
- ✓Best Model: Decision Tree.

```
Logistic Regression : 0.8333
SVM                 : 0.8333
Decision Tree       : 0.8333
KNN                 : 0.8333

Best on test set: Logistic Regression (accuracy = 0.8333)
Best by CV score: Decision Tree (CV accuracy = 0.8625)
```

All four models tie at 83.33% accuracy on the hold-out test set, but the decision tree had the highest mean cross-validation score (≈ 0.8625), making it the strongest performer in CV.

Best Model: Decision Tree



➢ Insights:

1. Identified key features impacting success
2. High-performing model provided reliable prediction.

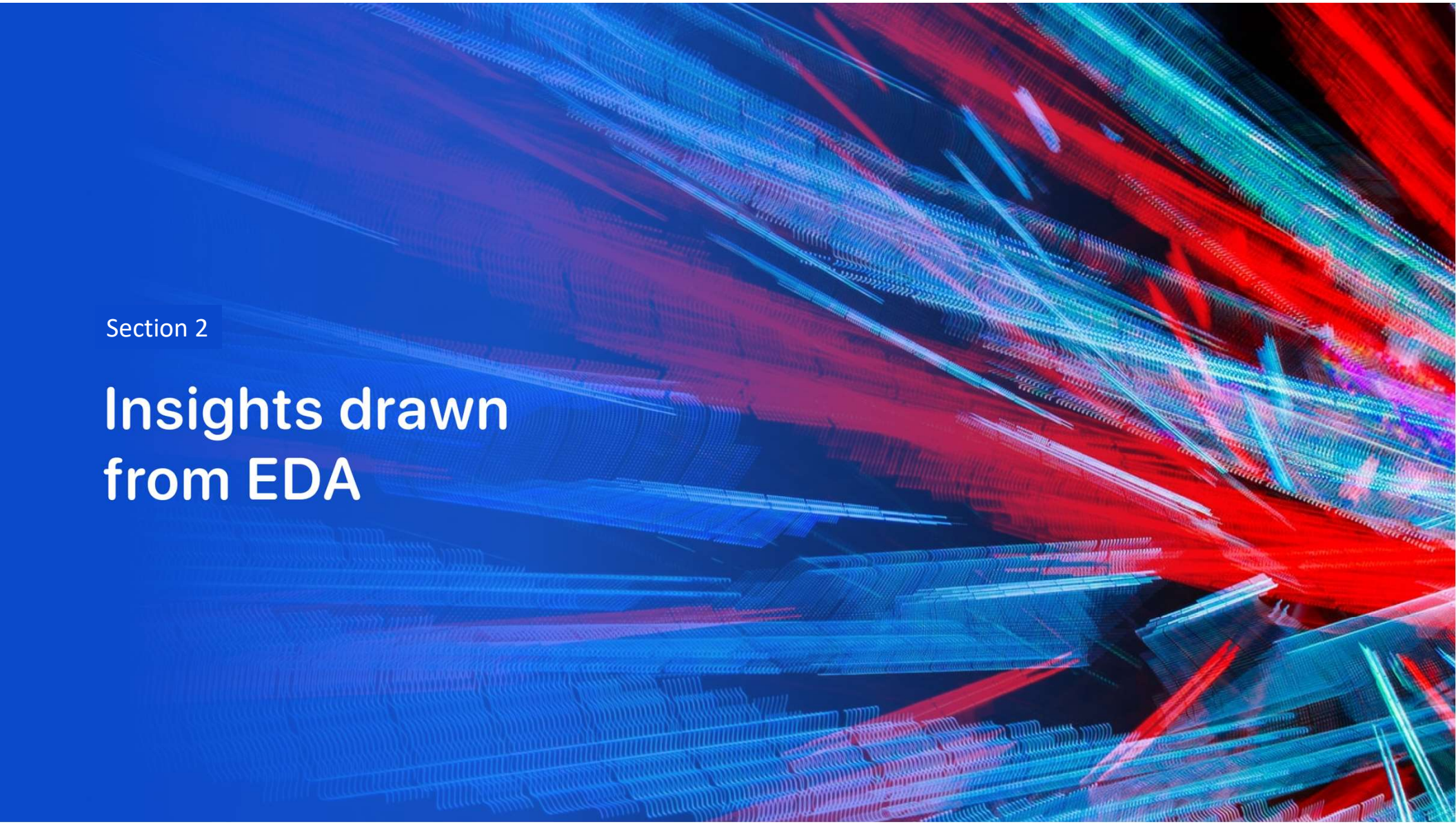
Summary of Methodology

➢ Summary:

- ✓Collected, cleaned, and merged SpaceX data via API and scraping.
- ✓Performed SQL and visual EDA to identify key patterns.
- ✓Developed interactive maps and dashboards for exploratory insights.
- ✓Built, tuned, and evaluated classification models to predict landing success.

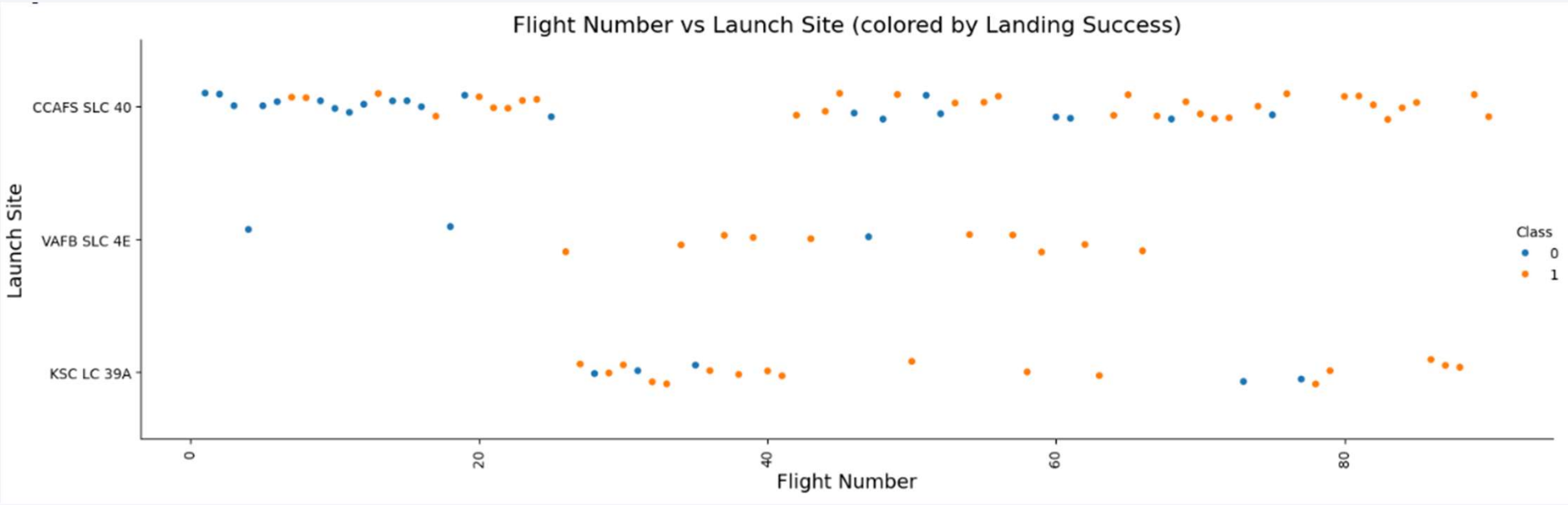
➢ Key Deliverables: https://github.com/sonhtgit/SpaceX-Landing-Prediction

- ✓ GitHub notebooks
- ✓ Reproducible dashboards
- ✓ Predictive model for future launch missions.


Section 2
Insights drawn
from EDA

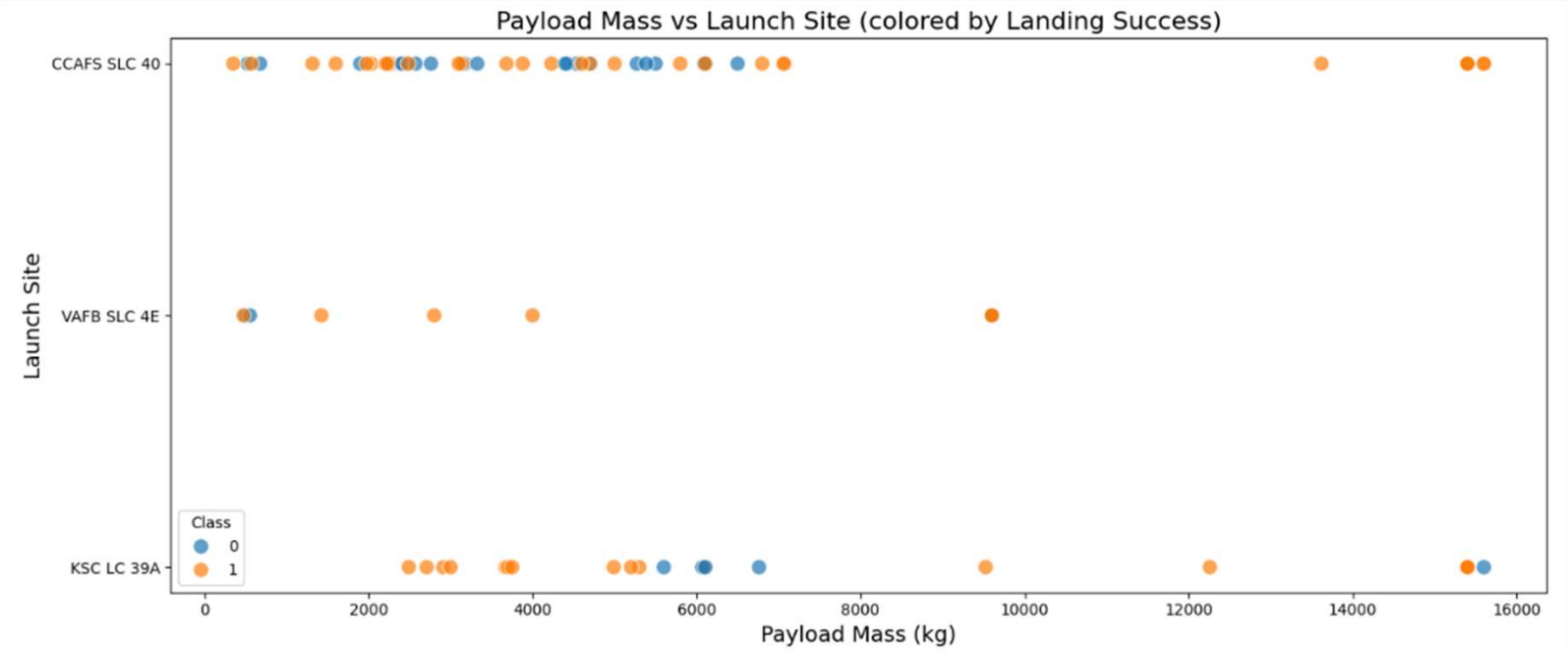
Insight: Flight Number vs. Launch Site

- Insight:
  - ✓ Certain launch sites like CCAFS SLC 40 and KSC LC 39A had more launches over time.
  - ✓ A trend suggests an increasing number of launches from specific sites.



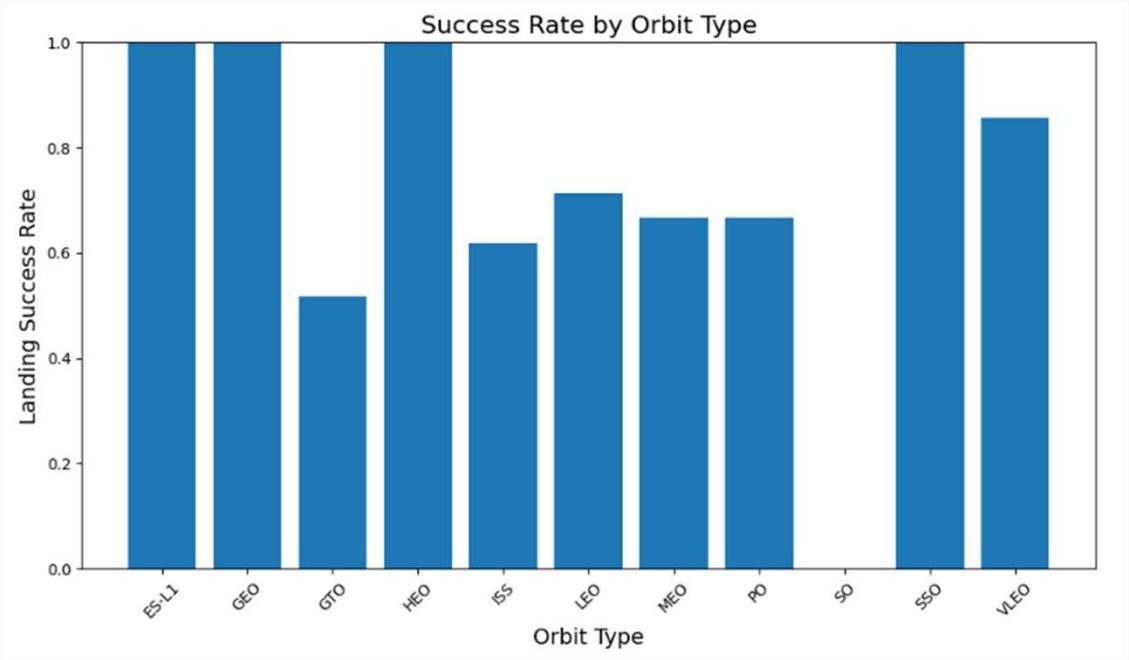
Insight: Payload Mass vs. Launch Site

- Insight:
  - ✓ Some sites are used for heavier payloads (e.g., KSC LC 39A).
  - ✓ The payload range varies by launch site.



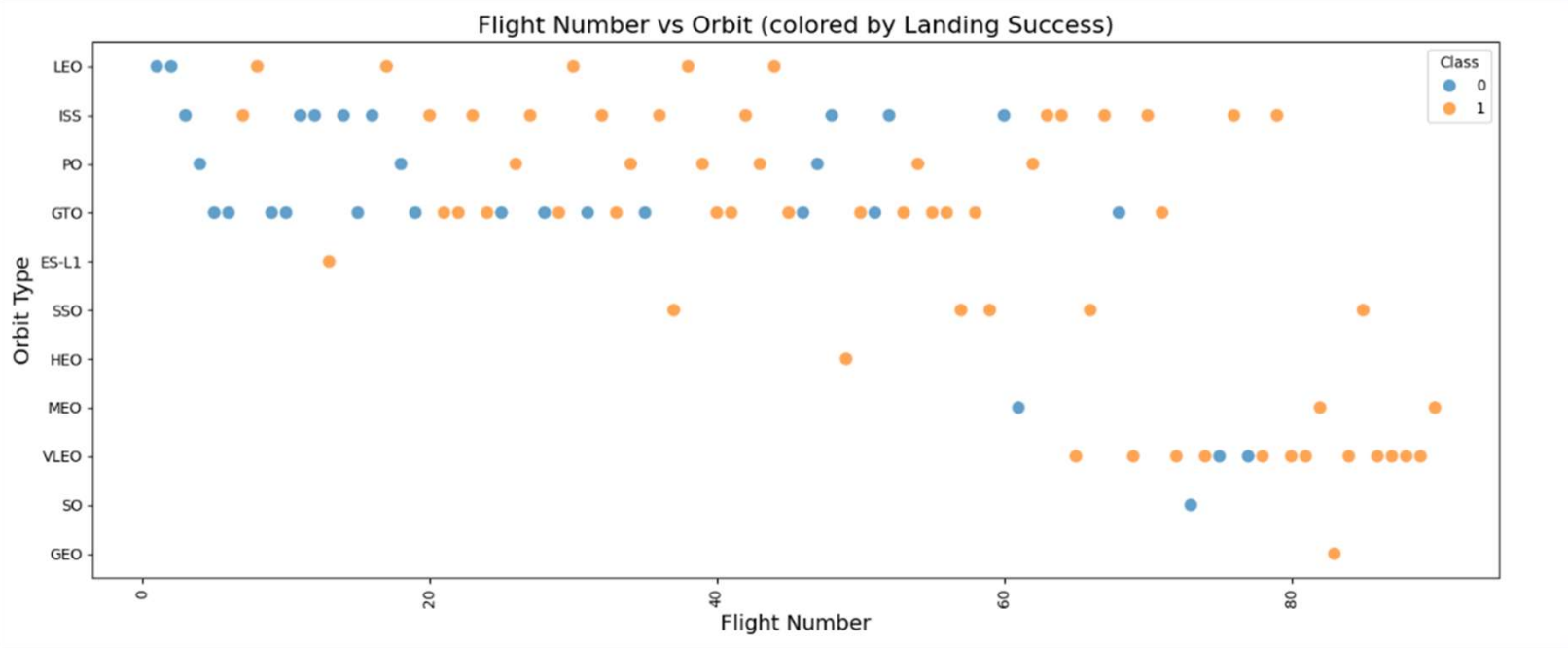
Insight: Success Rate vs. Orbit Type

- Insight:
  - ✓ LEO and GTO had the highest number of launches.
  - ✓ Success rates varied; LEO had higher success compared to GEO and others.



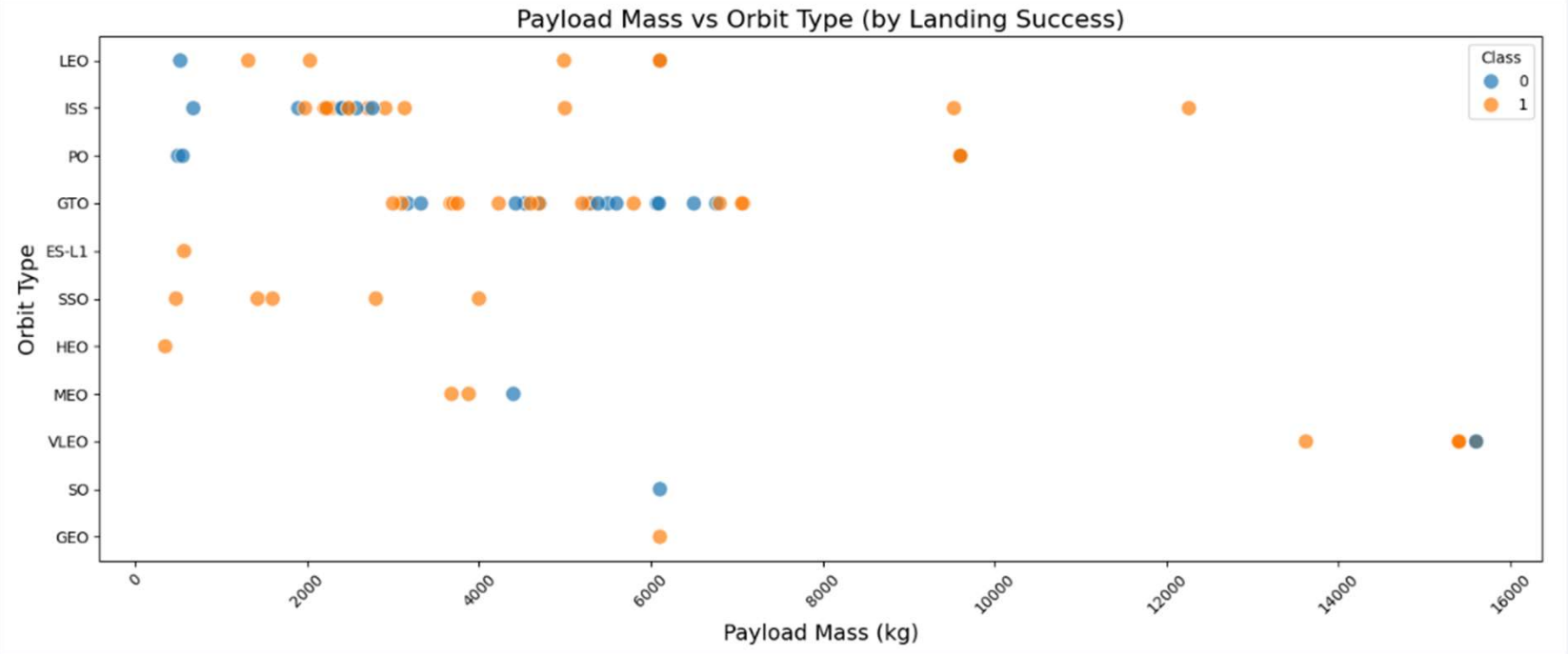
Insight: Flight Number vs. Orbit Type

- Insight:
  - ✓ Over time, a greater diversity of orbits were attempted.
  - ✓ LEO and GTO remained consistent across missions.



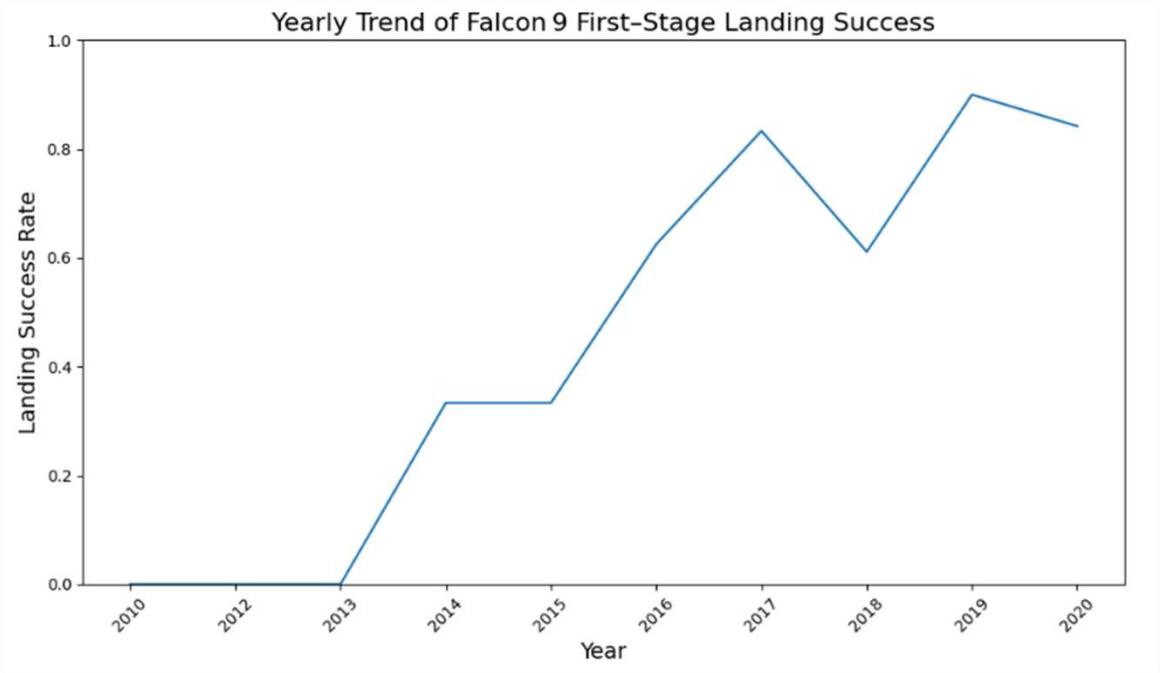
Insight: Payload Mass vs. Orbit Type

- Insight:
  - ✓ Payloads destined for VLEO tend to be heavier.
  - ✓ ISS had a wide range of payloads.



Insight: Launch Success Yearly Trend

- Insight:
  - ✓ Overall launch success improved over the years.
  - ✓ SpaceX appears to have optimized their systems post-2016.



Insight: All Launch Site Names

- SQL Insight:
  - ✓Unique launch sites used: CCAFS LC-40, KSC LC-39A, VAFB SLC-4E..

```
%%sql
-- Task 1: Display the names of the unique launch sites in the space mission
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

Insight: Launch Sites Starting with 'CCA'

- SQL Insight:
  - ✓ Queried sites beginning with 'CCA' – primarily CCAFS LC-40.
  - ✓ Useful for location-based filtering.

```
%%sql
-- Task 2: Display 5 records where launch sites begin with the string 'CCA'
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

Insight: Total Payload by NASA Boosters

- SQL Insight:
  - ✓Total payload mass carried by NASA-tagged missions summed from the dataset.

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

```
%%sql
-- Task 3: Display the total payload mass carried by boosters launched by NASA (CRS)
SELECT SUM("Payload_Mass__kg_") AS total_payload_kg
FROM SPACEXTABLE
WHERE "Customer" = 'NASA (CRS)';
```

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

total_payload_kg
45596

Insight: Average Payload Mass for F9 v1.1

- SQL Insight:
  - ✓ F9 v1.1 booster version carried a specific average payload mass, indicating its operational range.

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

```
%%sql
-- Task 4: Display average payload mass carried by booster version F9 v1.1
SELECT AVG("Payload_Mass__kg_") AS avg_payload_kg
FROM SPACEXTABLE
WHERE "Booster_Version" = 'F9 v1.1';
```

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

avg_payload_kg
2928.4

Insight: First Successful Ground Landing Date

- SQL Insight:
  - ✓ Identified earliest successful landing on a ground pad.
  - ✓ Helps analyze SpaceX's timeline for achieving ground-based recoveries.

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

Hint:Use min function

```
%%sql
-- Task 5: List the date when the first successful landing outcome in ground pad was achieved
SELECT MIN("Date") AS first_success_ground_pad
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)';
```

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

first_success_ground_pad
2015-12-22

Insight: Successful Drone Ship Landings with Payload 4000–6000 kg

- SQL Insight:
  - ✓ Found booster names for successful drone ship landings in specified payload range.
  - ✓ Critical for analyzing success under moderate-heavy load.

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

```
%%sql
-- Task 6: List the names of the boosters which have success in drone ship
--         and have payload mass >4000 but <6000
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

Insight: Total Successful and Failed Missions

- SQL Insight:
  - ✓ Summarized counts of success vs. failure.
  - ✓ Essential to evaluate overall performance.

List the total number of successful and failure mission outcomes

```
%%sql
-- Task 7: List the total number of successful and failure mission outcomes
SELECT
  "Mission_Outcome",
  COUNT(*) AS outcome_count
FROM SPACEXTABLE
GROUP BY "Mission_Outcome";
```

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

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

Insight: Booster with Max Payload

- SQL Insight:
  - ✓ Identified booster version carrying the heaviest payload.
  - ✓ Could inform booster selection strategy for future missions.

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

```
%%sql
-- Task 8: List all booster_versions that have carried the maximum payload mass
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

Insight: 2015 Failed Drone Ship Landings

- SQL Insight:
  - ✓ Filtered 2015 failures on drone ships with launch site and booster version info.
  - ✓ Revealed early operational difficulties.

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
-- Task 9: For year 2015, show month name, failure (drone ship) landing outcomes,
--         booster version and launch site
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 "Landing_Outcome" = 'Failure (drone ship)'
  AND substr("Date",1,4) = '2015';
```

```
 * 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

Insight: Rank of Landing Outcomes (2010–2017)

- SQL Insight:
  - ✓ Ranked landing outcomes between 2010-06-04 and 2017-03-20.
  - ✓ Showed which landing types were most common.

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
-- Task 10: Count of each landing outcome between 2010-06-04 and 2017-03-20, ranked descending
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

Section 3

Launch Sites Proximities Analysis

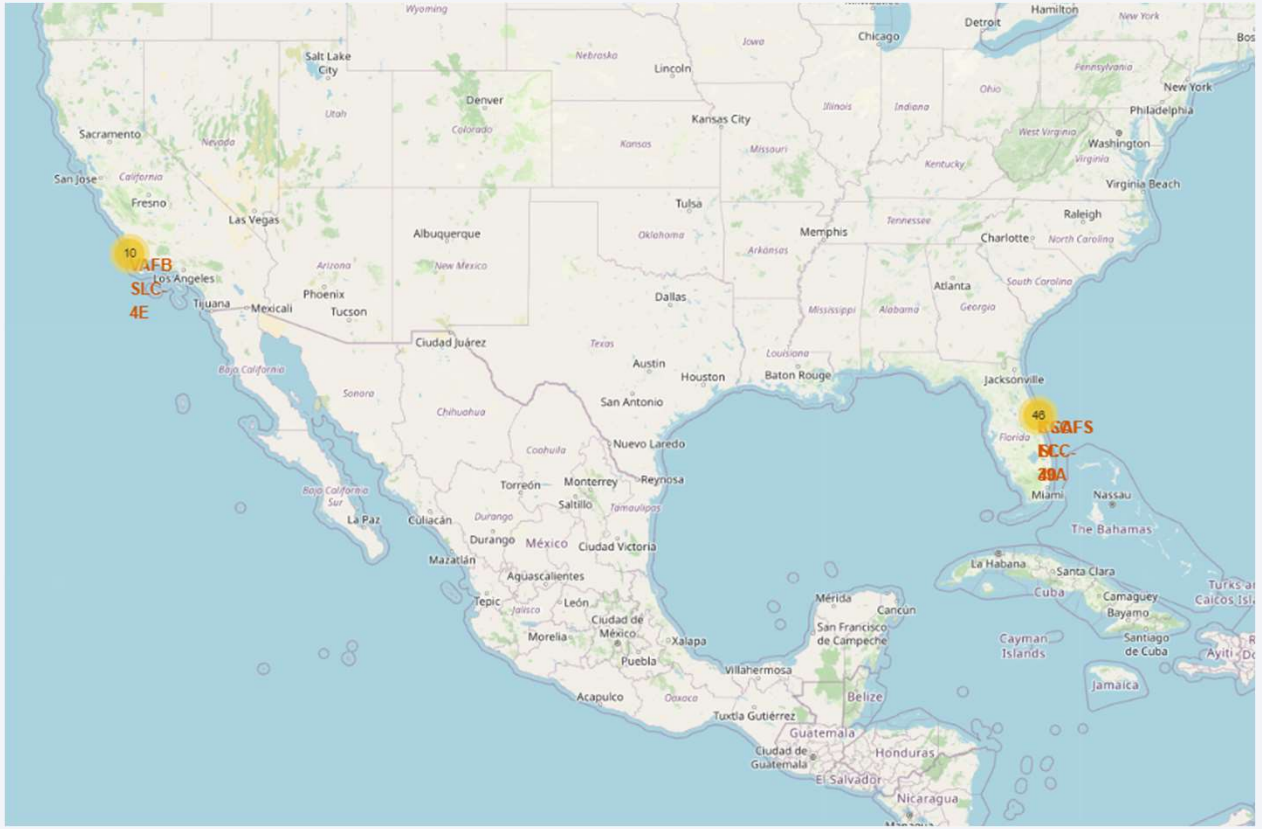
Folium Map: Global Launch Sites Map

➢ Description:

- ✓ This Folium map visualizes all known SpaceX launch sites.
- ✓ Each launch site is marked with a circle marker and labeled with its site name.
- ✓ Helps understand geographical spread and global positioning of operations.

➢ Insights:

- ✓ Major sites include:
  - **CCAFS SLC-40 (Florida)**
  - **KSC LC-39A (Florida)**
  - **VAFB SLC-4E (California)**



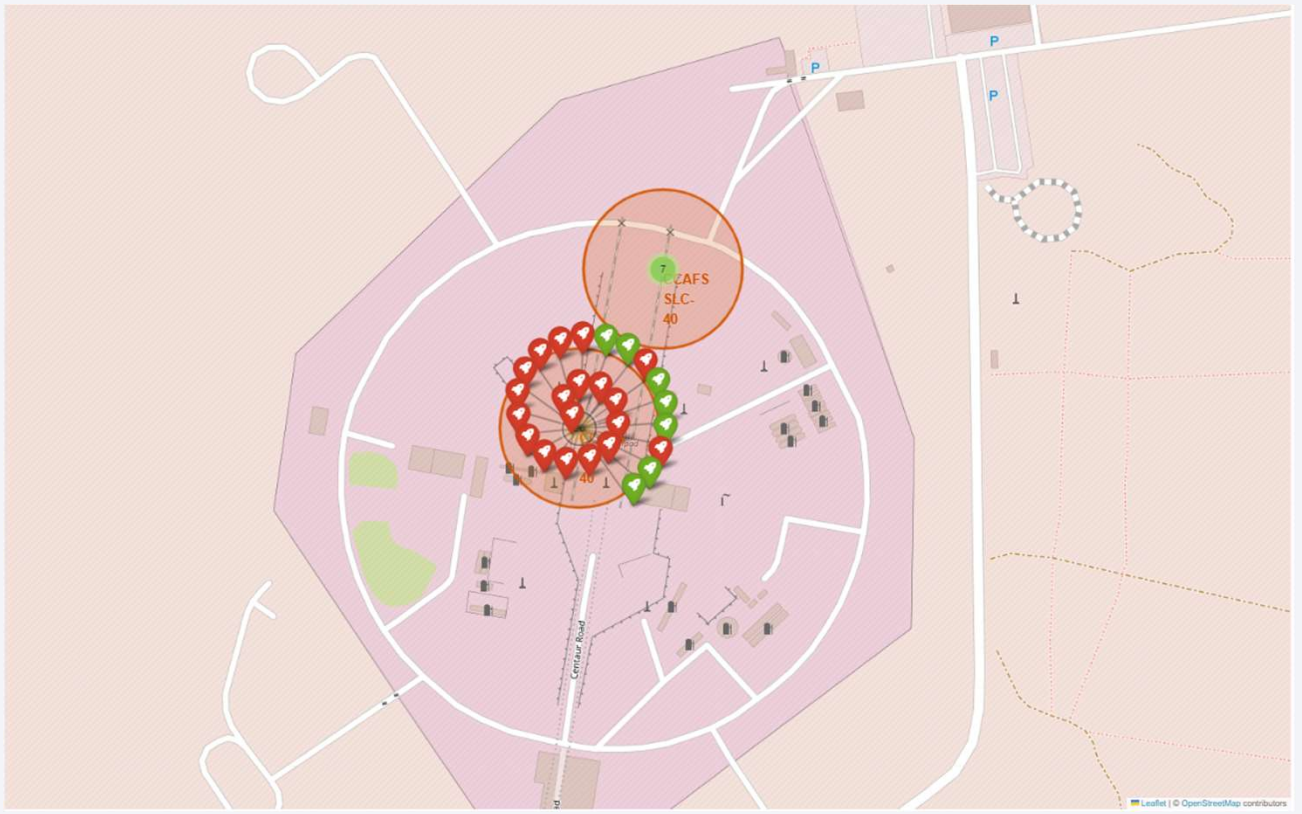
Folium Map: Launch Outcome Visualization on Map

➢ Description:

- ✓ Folium map enhanced with color-coded markers representing launch success or failure:
  - **Green**: Successful landings
  - **Red**: Failed landings
  - **Blue**: No landing attempt.

➢ Insights:

- ✓ Successful landings were more frequent at specific locations.
- ✓ Some sites had higher failure rates or unattempted landings.



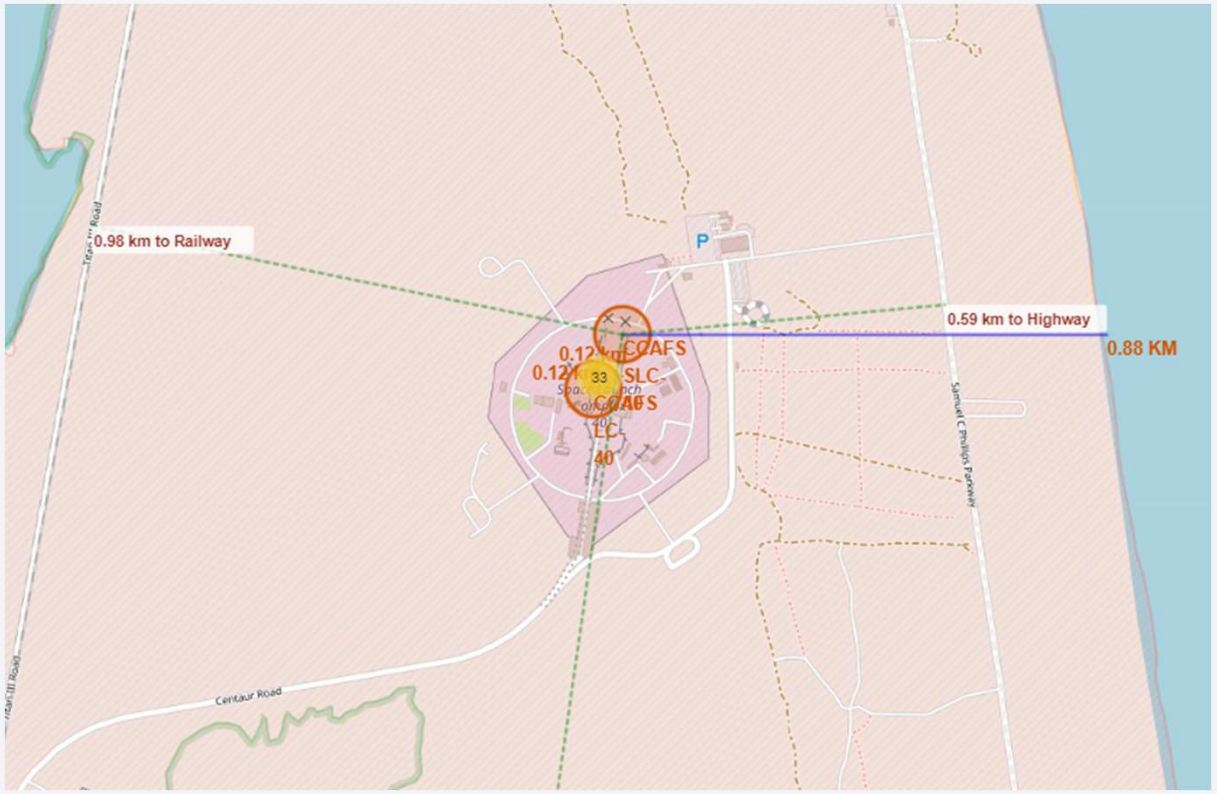
Folium Map: Site Proximity to Key Features

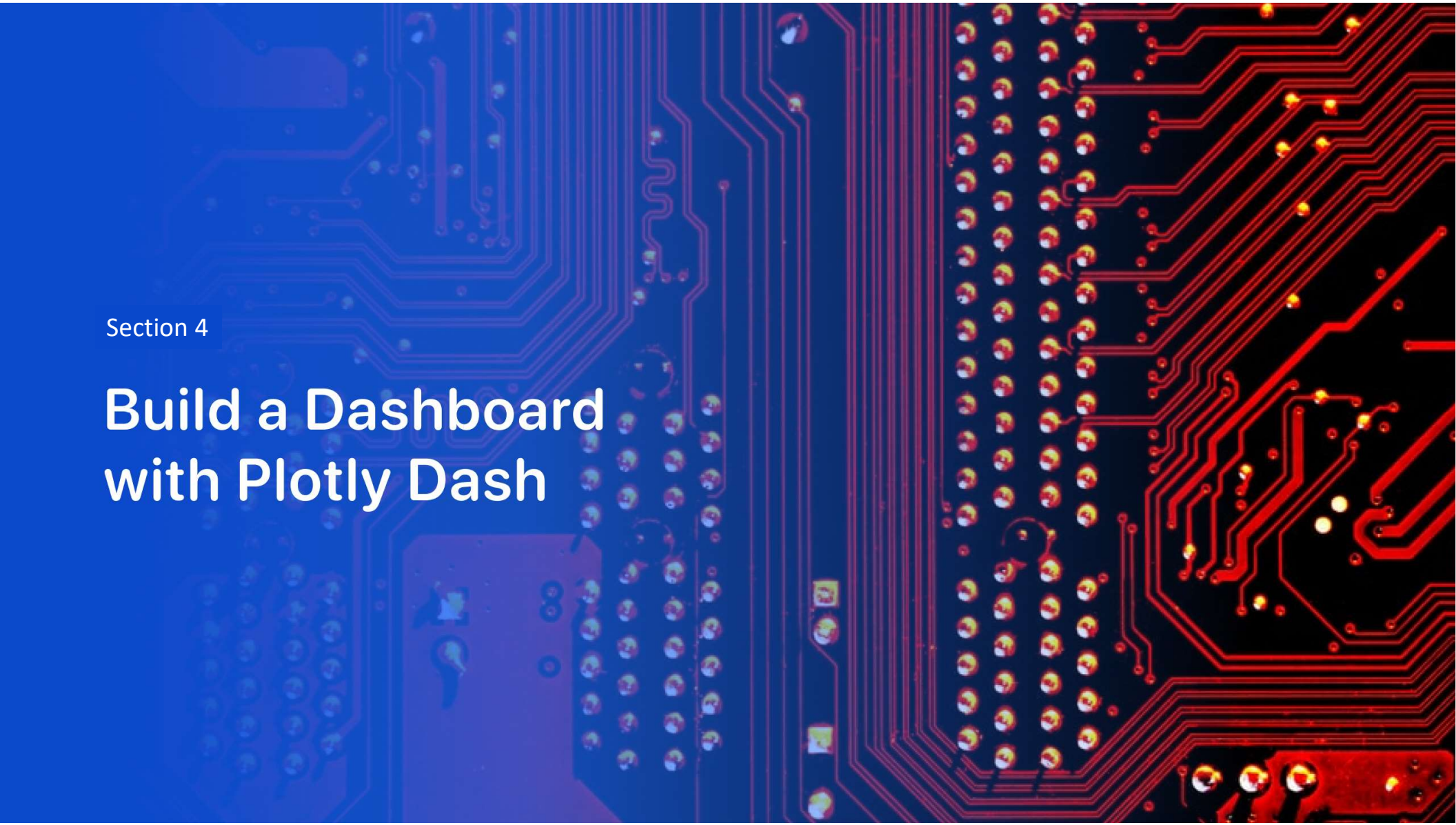
➢ Description:

- ✓ Analyzed the distance between launch sites and key geographic features using Folium + geopy:
  - **Coastlines** – strategic for recovery operations.
  - **Highways and Railways** – relevant for transport logistics.
- ✓ Example: Measured distance from CCAFS SLC-40 to nearest coast, highway and city.

➢ Insights:

- ✓ Most SpaceX sites are located close to coastlines, indicating sea-based recovery strategy.
- ✓ Proximity to infrastructure supports logistics and turnaround efficiency.





Section 4

Build a Dashboard with Plotly Dash

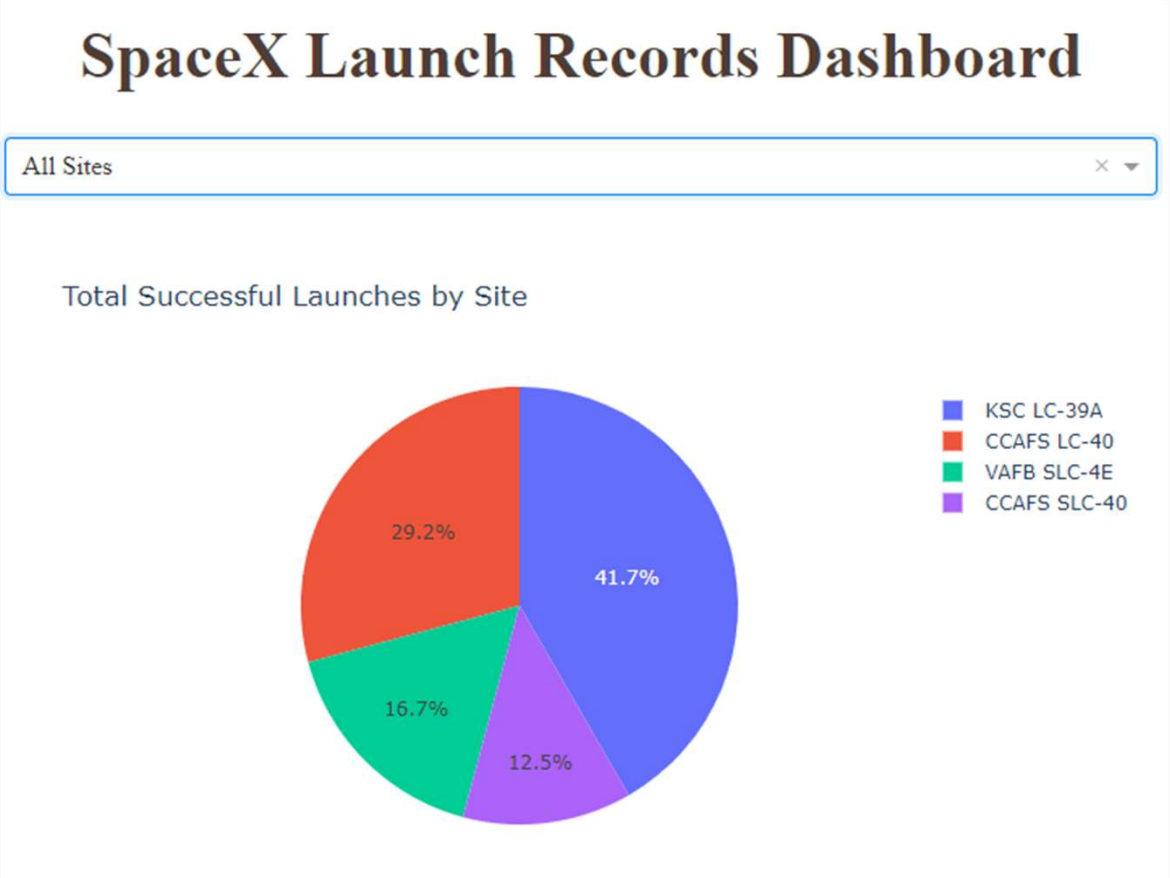
Dashboard: Launch Success Counts for All Sites

➢ Description:

- ✓ A pie chart displays total successful launches from each launch site.
- ✓ Hover-over tooltips show both absolute numbers and percentages.

➢ Insights:

- ✓ **CCAFS SLC-40** and **KSC LC-39A** had the most successful launches.
- ✓ Visualizes SpaceX's reliance on specific sites for repeat success.



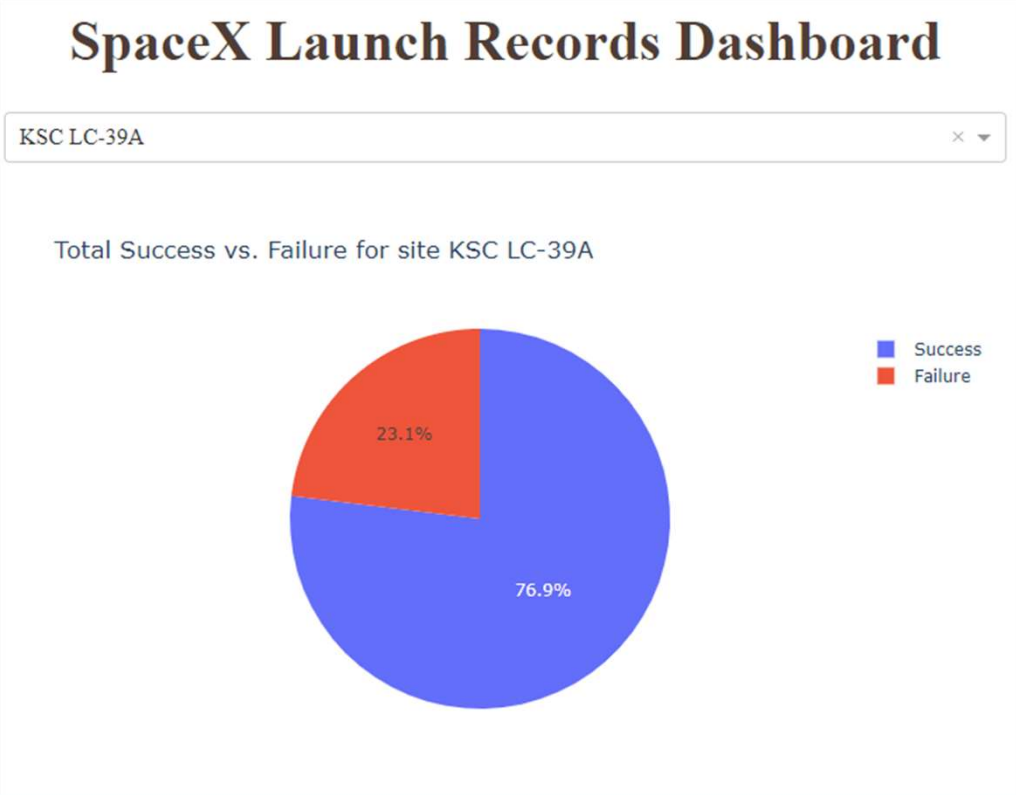
Dashboard: Highest Success Ratio Launch Site

➢ Description:

- ✓ Filtered pie chart displays outcomes (success/failure) for the site with the highest success ratio.
- ✓ Provides insight into reliability of a specific location.

➢ Insights:

- ✓ **KSC LC-39A** had the highest successful launches at 76.9%.
- ✓ Useful for evaluating site performance consistency.



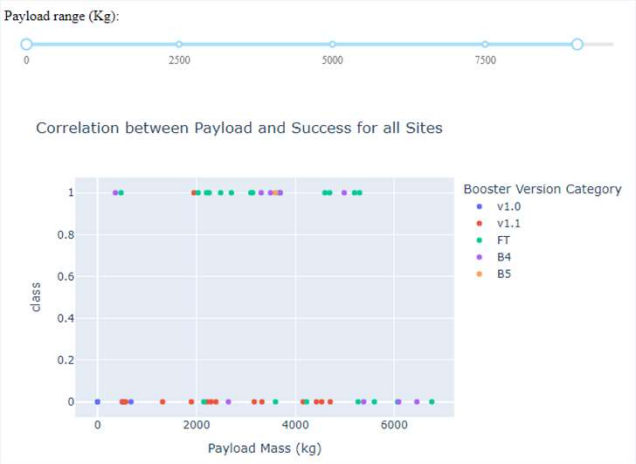
Dashboard: Payload vs. Launch Outcome with Slider

➢ Description:

- ✓ Scatter plots shows correlation between payload mass (kg) and launch outcome.
- ✓ Interactive range slider allows users to filter by payload weight.
- ✓ Hover data includes booster version, orbit, and success status..

➢ Insights:

- ✓ Medium-range payloads (2,000–4,000 kg) had higher success rates.
- ✓ Some booster versions consistently performed better within this range.



Section 5

Predictive Analysis (Classification)

Analysis: Model Comparison – Classification Accuracy

➢ Description:

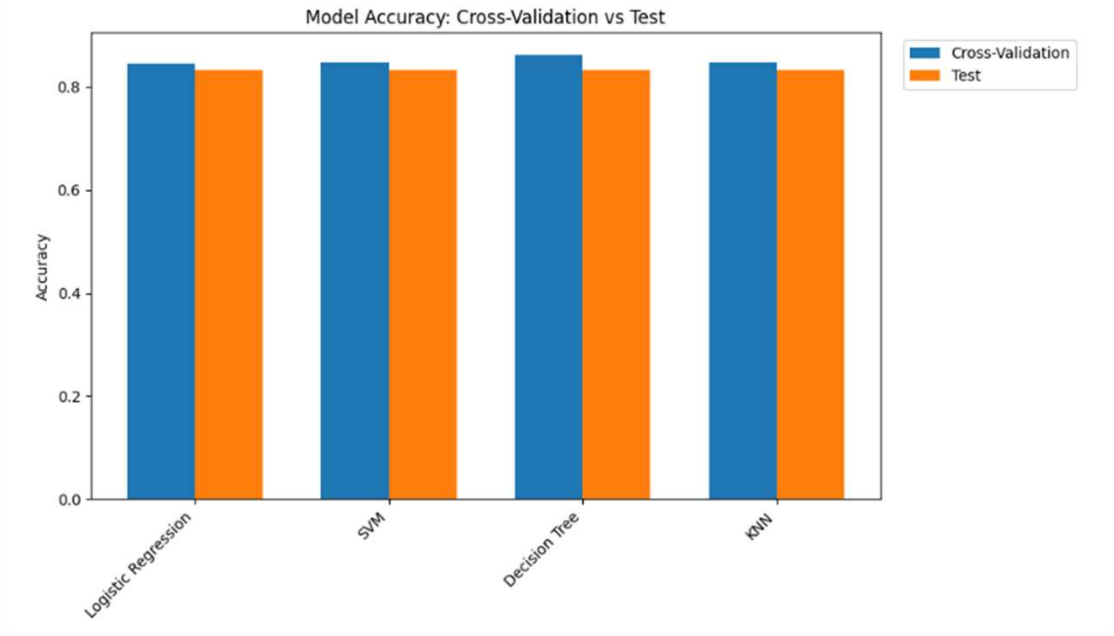
- ✓ Four models were trained and evaluated:
  - **Logistic Regression**
  - **Support Vector Machine (SVM)**
  - **Decision Tree**
  - **K-Nearest Neighbors (KNN)**
- ✓ Accuracy scores were compared after hyperparameter tuning and train/test evaluation.

➢ Insights:

- ✓ All four models tie at 83.33% accuracy on the hold-out test set.
- ✓ But the decision tree had the highest mean cross-validation score (≈ 0.8625), making it the strongest performer in CV.

```
Logistic Regression : 0.8333
SVM                 : 0.8333
Decision Tree       : 0.8333
KNN                 : 0.8333

Best on test set: Logistic Regression (accuracy = 0.8333)
Best by CV score: Decision Tree (CV accuracy = 0.8625)
```



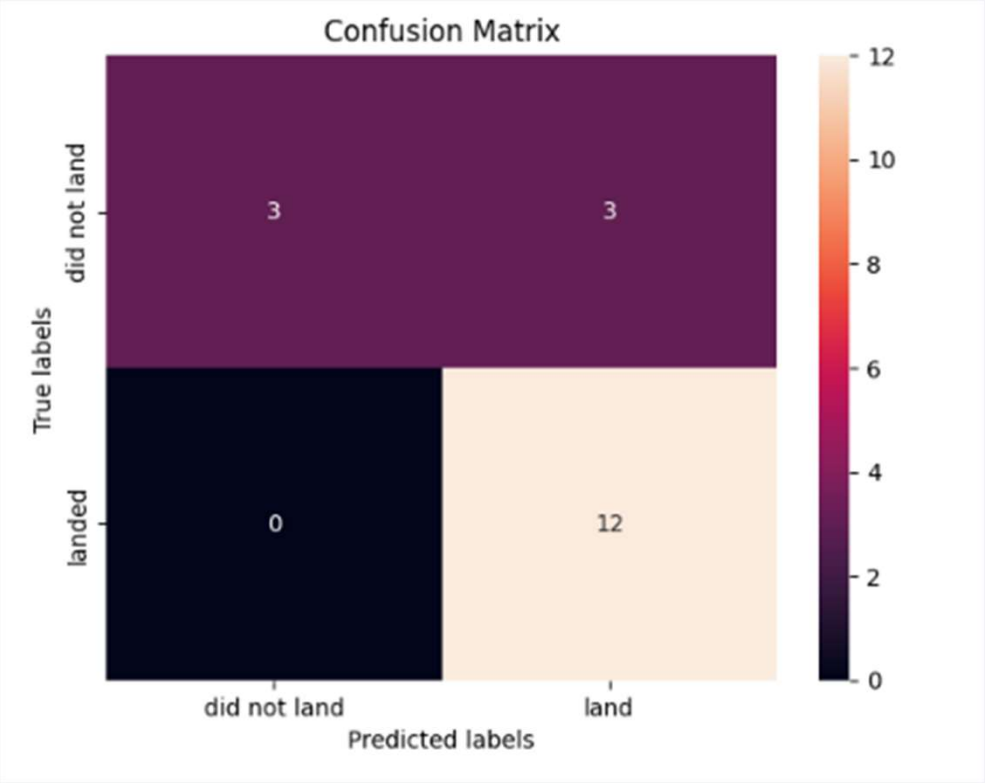
Analysis: Confusion Matrix – Best Performing Model

➢ Description:

- ✓ The confusion matrix breaks down:
  - **True Positives** (correctly predicted successes)
  - **True Negatives** (correctly predicted failures)
  - **False Positives** (incorrectly predicted successes)
  - **False Negatives** (missed successes).

➢ Insights:

- ✓ High number of true positives → good success prediction.
- ✓ Relatively low false negatives → robust model for minimizing launch risk.



Analysis: Key Features and Tuning Strategy

➢Feature Engineering:

- ✓ Selected key features based on domain knowledge and EDA:
  - Payload Mass (kg)
  - Booster Version
  - Launch Site
  - Orbit Type
- ✓ Converted categorical variables into dummy/one-hot encoded columns

➢ Data Preparation:

- ✓ Applied StandardScaler to normalize features.
- ✓ Split data into training (80%) and test (20%) sets using train_test_split.

➢Model Tuning Strategies:

- ✓ Logistic Regression: Adjusted regularization strength (C).
- ✓ SVM: Tuned kernel type (rbf, linear), C, and gamma.
- ✓ Decision Tree: Tuned max_depth, min_samples_split.
- ✓ KNN: Tuned n_neighbors and distance metrics..

➢ Insights:

- ✓ The model's performance heavily depended on correct feature scaling and categorical encoding.
- ✓ SVM and Decision Tree yielded the best results after tuning..

Analysis: Summary of Predictive Modeling

➢ Key Takeaways:

- ✓ Trained multiple classification models using engineered features.
- ✓ Identified the best-performing model for predicting landing outcomes.
- ✓ Model demonstrated strong predictive power, especially for mid-range payloads.
- ✓ Approach is extensible to future missions and SpaceX data pipelines.

Section 6

Conclusion and Reflection

Project Conclusion

➢ Key Takeaways:

- ✓ Developed a robust data pipeline from API scraping to model deployment.
- ✓ Gained deep insights into SpaceX launch patterns through EDA and dashboards.
- ✓ Built and evaluated several machine learning models; SVM/Decision Tree performed best.
- ✓ Successfully predicted Falcon 9 landing success with high accuracy.

Personal Reflection

➢Skills Acquired:

- ✓ API data extraction, web scraping
- ✓ Data wrangling and cleaning
- ✓ SQL-based and visual EDA
- ✓ Interactive dashboards with Folium and Dash
- ✓ Model training, tuning, and evaluation.

➢ Challenges Overcome:

- ✓ Merging multiple datasets with missing values
- ✓ Optimizing model performance with limited features

➢Future Improvements:

- ✓ Explore ensemble models (Random Forest, Gradient Boosting)
- ✓ Include more time-series or mission metadata

Appendix

Appendix

➢ Included Artifacts & References:

- ✓ Python Code (API calls, Web Scraping, Feature Engineering)
- ✓ SQL Queries (EDA & Data Filtering)
- ✓ Charts and Visualizations (used in EDA section)
- ✓ Folium Map Screenshots (Global Sites, Outcomes, Proximities)
- ✓ Plotly Dash Dashboard Screenshots
- ✓ Model Evaluation Outputs (Accuracy, Confusion Matrix)
- ✓ GitHub Respo Link: https://github.com/sonhtgit/SpaceX-Landing-Prediction
  - Data Collection → https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module1
  - EDA & Visualization → https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module2
  - SQL Analysis → https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module2
  - Folium Map → https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module3
  - Dash App → https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module3
  - Classification Models → https://github.com/sonhtgit/SpaceX-Landing-Prediction/tree/main/module4

Thank you!