IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Janani Pradeep

27 March, 2025

Outline

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

Executive Summary

SpaceX has transformed satellite launches through its innovative use of reusable launch systems—namely the Falcon 9 and Falcon Heavy. This reduces launch costs by reusing the Falcon 9 first stage, bringing launch prices down to ~$62M compared to ~$165M from other providers.The success of first-stage recovery is a critical factor in maintaining this cost advantage.

Booster recovery is influenced by several key factors, including:

- Orbit type
- Payload mass
- Booster version
- Launch site location

This capstone project leverages real-world launch data to predict first-stage landing outcomes using data science and machine learning techniques. I built interactive dashboards, maps, and trained ML models to support insights and predict recovery with high accuracy with the best one with booster recovery prediction accuracy of nearly 87%. This supports SpaceY with a analysis and risk assessment for satellite launches.

Introduction

- SpaceX’s innovation in reusable rockets, particularly the Falcon 9 first stage, has disrupted the aerospace market by lowering launch costs to ~$62 million. Successful recovery of the first stage is critical to cost-effectiveness and mission planning.
- This capstone project builds a data-driven pipeline to:
    - Collect and process SpaceX launch data.
    - Perform exploratory and interactive visual analytics.
    - Build predictive machine learning models to estimate landing success.
- The outcome aids the competing launch providers; SpaceY in estimating potential cost savings and optimizing decision-making.

Section
1
Methodology

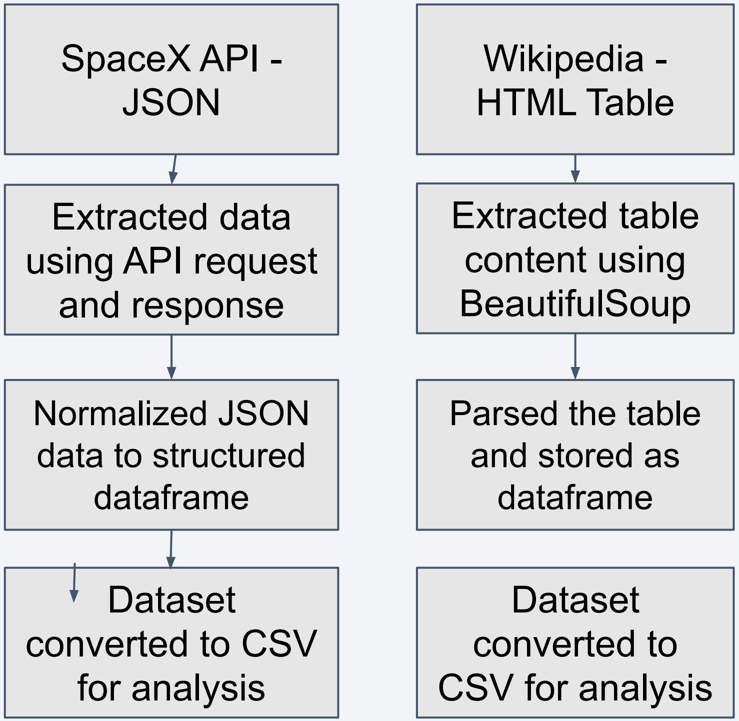
Methodology

- Collected data from open API
- Performed data wrangling
- Performed exploratory data analysis (EDA) using visualization and SQL
- Performed interactive visual analytics using Folium and Plotly Dash
- Performed predictive analysis using classification models

Data Collection

Collected SpaceX launch data from two sources:

- structured data in the form of JSON from the SpaceX REST API
- unstructured data in the form of HTML table from the Wikipedia Page



Data Collection – SpaceX API

SpaceX REST API → Extract nested attributes → Flatten JSON and Merge All Extracted Data → Unified Launch Data for Analysis

Used SpaceX REST API (endpoint: `https://api.spacexdata.com/v4/launches/past`) to retrieve structured JSON data on Falcon 9 launches.

Data Retrieved Includes:

- Booster version (from rockets)
- Launch site details: name, latitude, longitude (from launchpad)
- Payload mass and orbit (from payloads)
- Landing outcome, reuse status, block version, grid fins, legs, etc. (from cores)

`pandas.json_normalize()` to convert nested JSON to flat DataFrames

Used a fallback URL for consistent results

Unified DataFrame with all mission metadata and converted to csv for analysis.

Github: https://github.com/Janani241/falcon9/blob/main/jupyter-labs-spacex-data-collection-api-v2.ipynb

Data Collection - Scraping

HTTP Request and Data Parsing → **Table Extraction** → **Data Restructuring**

HTTP Request and Data Parsing

Extracted the HTML content from Wikipedia link of Falcon 9 and Falcon Heavy launches using `requests.get().text`

Parsed the extracted content by creating BeautifulSoup object with `html.parser`.

Table Extraction

Located the <table> of interest from the parsed content and used `find_all('th')` to get column headers and iterated over rows using `find_all('tr')` to extract data.

Data Restructuring

Stored the extracted table as a Pandas dataframe and exported to csv for analysis.

Github: https://github.com/Janani241/falcon9/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

- Data was processed by:
  - Preprocessing Step:
    - Checking for missing values using df.isnull().
    - Identifying column data types (df.dtypes).
    - Cleaning missing values if any (e.g., ~29% in LandingPad).
  - Target Variables labeling for classification and feature engineering:
    - Extracting outcome labels like "True ASDS", "False Ocean", etc.
    - Defining a bad_outcomes set indicating failed landings.
    - Creating a new column Class:
      - 1 if launch was successful.
      - 0 if launch failed.

Preprocessing by handling missing values → Target variable classification and feature engineering → Visualization and SQL Querying for initial analysis

Github: https://github.com/Janani241/falcon9/blob/main/labs-jupyter-spacex-Data%20wrangling-v2.ipynb

EDA with Data Visualization

- Used scatter, strip, bar and line charts to visualize and perform exploratory data analysis.
  - Visualized the impact of flight number on payload and launch success.
  - Analyzed launch site activity over time to identify high-traffic locations.
  - Compared payload mass across different launch sites.
  - Evaluated success rate by orbit type using bar plots.
  - Explored how flight number and orbit type relate to mission outcomes.
  - Observed payload mass distribution across orbit types.
  - Tracked launch success rate trends over the years with a line chart.
- The plots helped ese visualizations helped us:
  - Detect outliers in payload and success
  - Correlate launch sites and orbits with outcomes
  - Confirm SpaceX's performance has improved significantly over the years

Github: https://github.com/Janani241/falcon9/blob/main/jupyter-labs-eda-dataviz-v2.ipynb

EDA with SQL

- Retrieved unique launch site names to understand site distribution.
- Filtered and displayed specific launch records using partial matches.
- Calculated total and average payloads for selected boosters and customers.
- Identified the first successful landing date on a ground pad.
- Filtered boosters with specific payload and landing outcomes.
- Aggregated mission outcomes (success vs failure counts).
- Found the booster version with the highest payload using a subquery.
- Extracted monthly launch trends for a specific year.
- Ranked landing outcomes by frequency within a defined date range.

Github: https://github.com/Janani241/falcon9/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Summary of Map Objects:
  - Circle Markers: Helps identify spatial clustering of launch activity.
  - Text Markers (Labels): Enhances clarity while analyzing locations.
  - Colored Launch Markers: Used red for failed launches and green for successful ones based on the class column. Aids in visually assessing the success rate by location.
  - Marker Clusters: Grouped multiple launch markers in the same location to simplify crowded visuals and improve readability on zoom.
  - Mouse Position Tool: Enabled to dynamically read coordinates by hovering over the map. Useful for manual proximity analysis to coastlines and infrastructure.
  - Polyline Lines: This supports the investigation of geographical impact on launch success.
- These elements helped explore the spatial distribution of launch sites, assess proximity to human-made infrastructure, and examine correlations with launch success visually.
- Github: https://github.com/Janani241/falcon9/blob/main/lab-jupyter-launch-site-location-v2.ipynb

Build a Dashboard with Plotly Dash

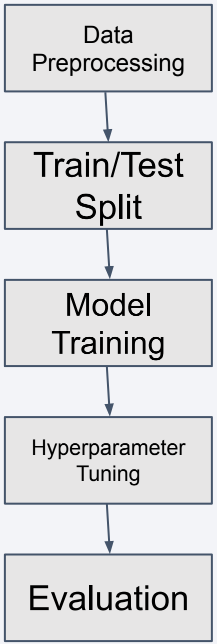
- To gain dynamic insights into the SpaceX launch data, we developed interactive visualizations using Plotly Dash. This allowed to explore launch outcomes, payload impact, and booster performance across various launch sites in an intuitive and engaging way
- Key Components Implemented:
  - Dropdown Menu to filter visualizations by launch site.
  - Pie Chart to show the distribution of successful vs. failed launches across all or specific sites.
  - Range Slider to filter records based on payload mass.

- Github: https://github.com/Janani241/falcon9/blob/main/Build_a_Dashboard_Application_with_Plotly_Dash_v10.ipynb

Predictive Analysis (Classification)

- Data Preparation
    - Extracted Class column as label Y
    - Standardized features X using StandardScaler
    - Split dataset (80% train / 20% test) using train_test_split
- Model Building & Tuning
    - Applied GridSearchCV (cv=10) to tune hyperparameters for:
    - Logistic Regression
    - Support Vector Machine (SVM)
    - Decision Tree
    - K-Nearest Neighbors (KNN)
- Model Evaluation
    - Validation Accuracy
    - Test Accuracy
    - Confusion Matrix (focus on True Positives & False Positives)

- Github: https://github.com/Janani241/falcon9/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb



Results

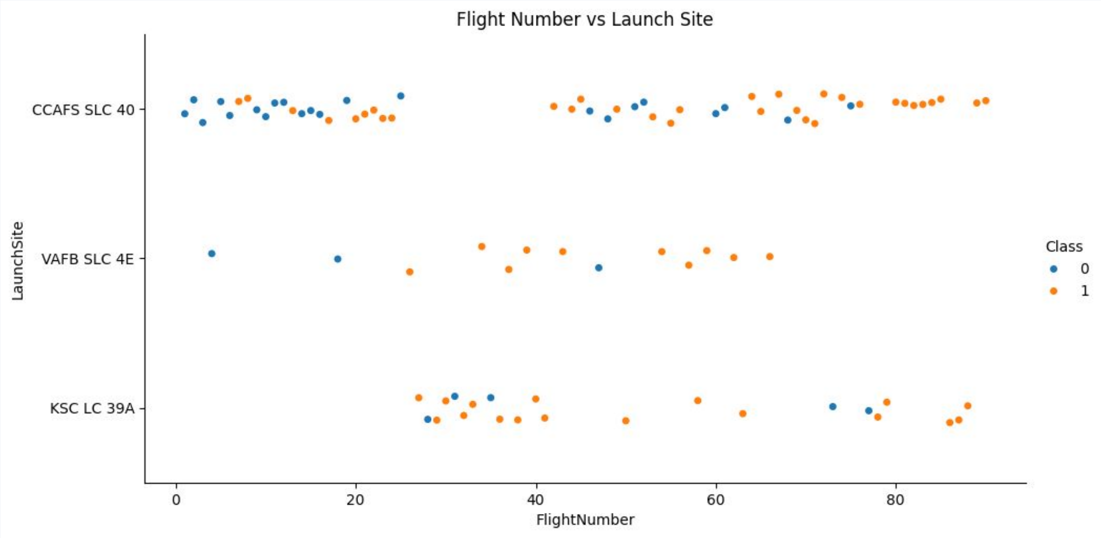
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

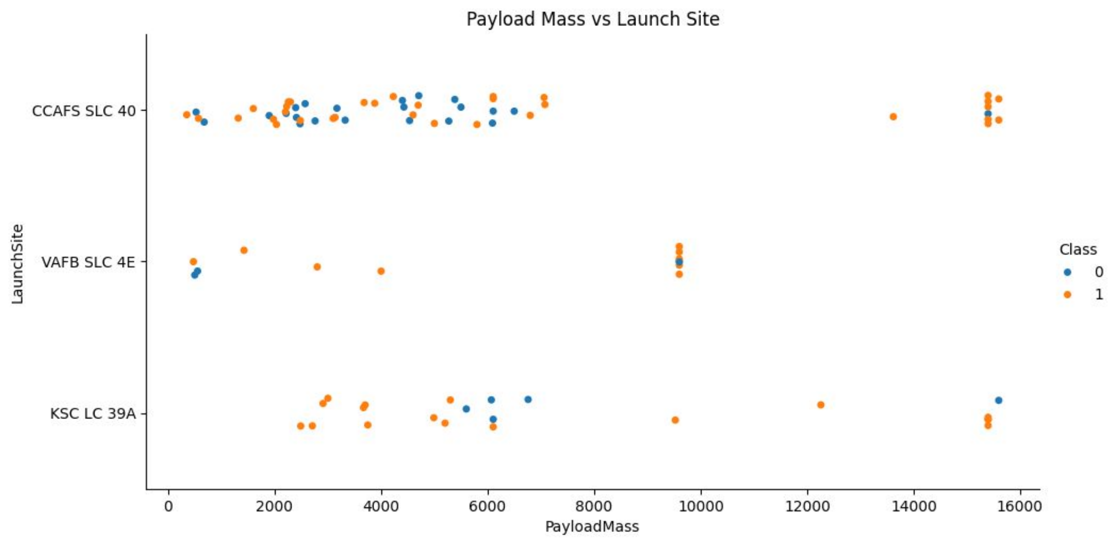
Flight Number vs. Launch Site

- The purpose of this was to examine the distribution of launches across different sites over time.
- It shows that CCAFS SLC 40 was used heavily early on; later launches are more evenly distributed.



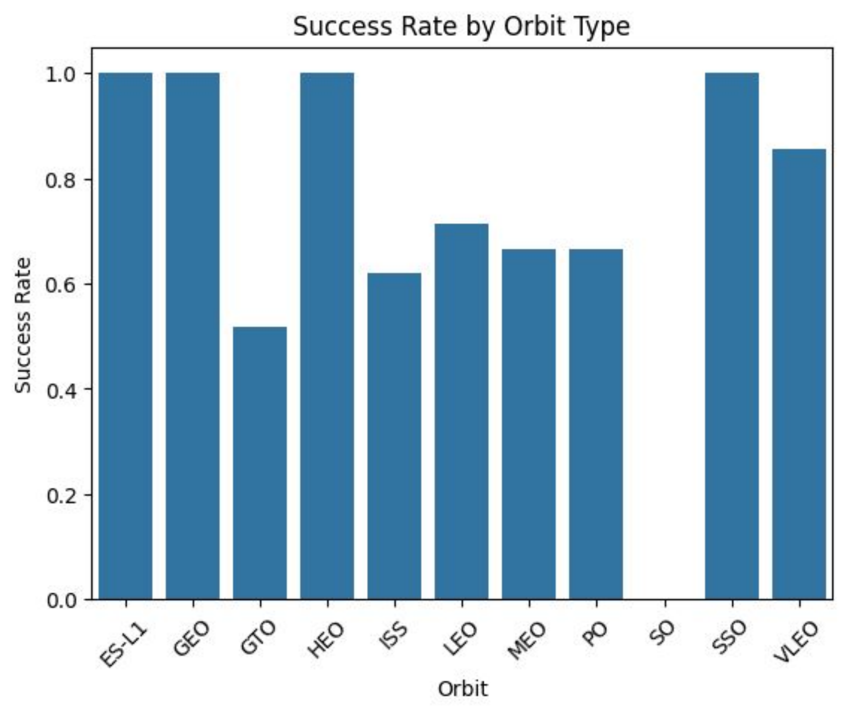
Payload vs. Launch Site

- The purpose of this graph was to assess whether payload mass varies significantly by launch site
- It shows that all sites handled a wide payload range, but CCAFS SLC 40 handled the heaviest and for the VAFB-SLC launchsite there are no rockets launched for heavy payload mass



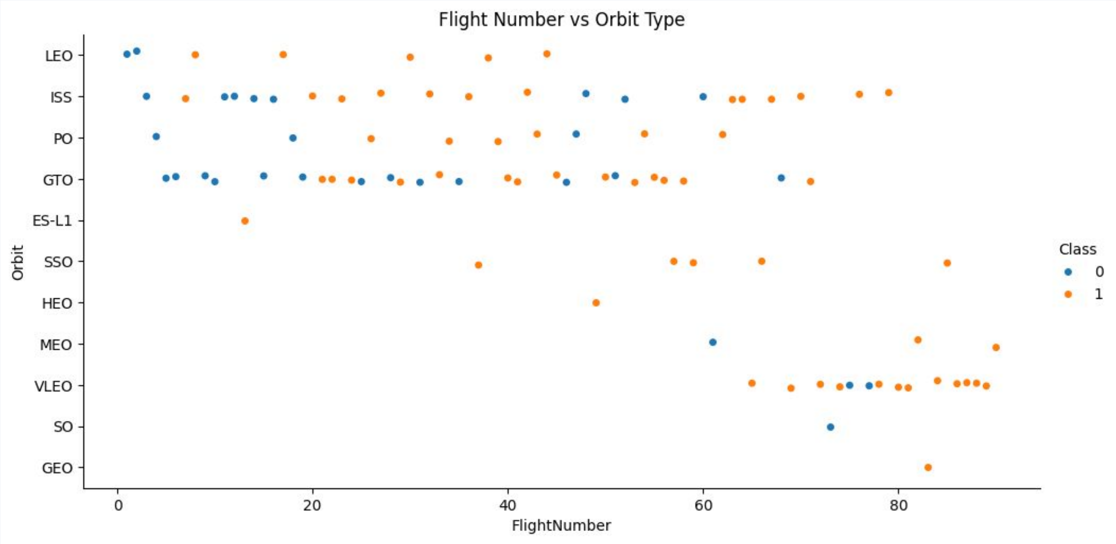
Success Rate vs. Orbit Type

- This graph shows how orbit type affects launch success probability.
- We can see that ES-L1, GEO, SSO, and SO orbits have the highest success rates. HEO and ISS are riskier



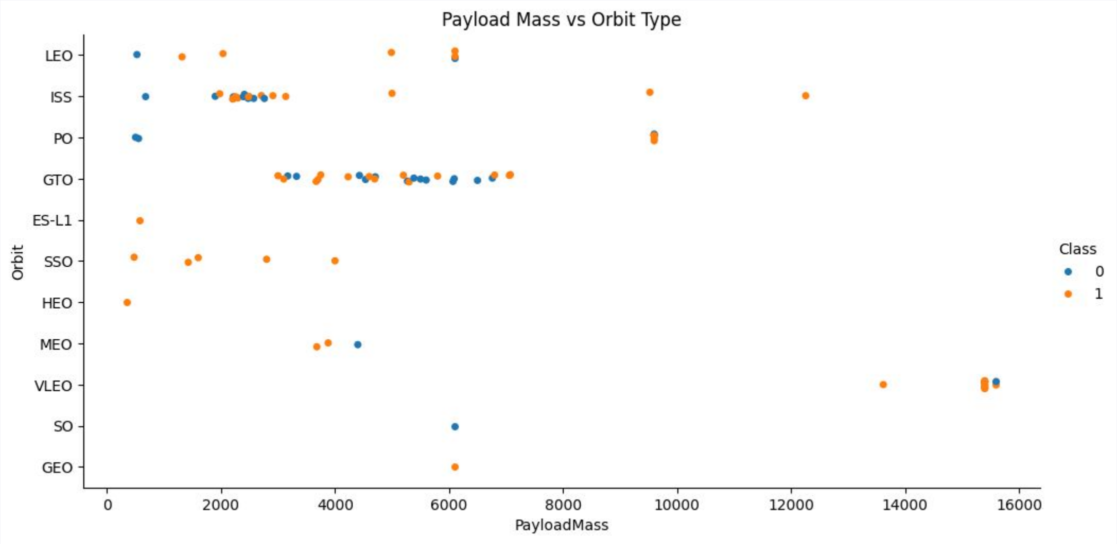
Flight Number vs. Orbit Type

- This plot is to visualize if specific orbits were targeted more frequently in later missions.
- We can see that as mission count increased, more variety in orbit targeting emerged, with consistent success improvement
- Also in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit



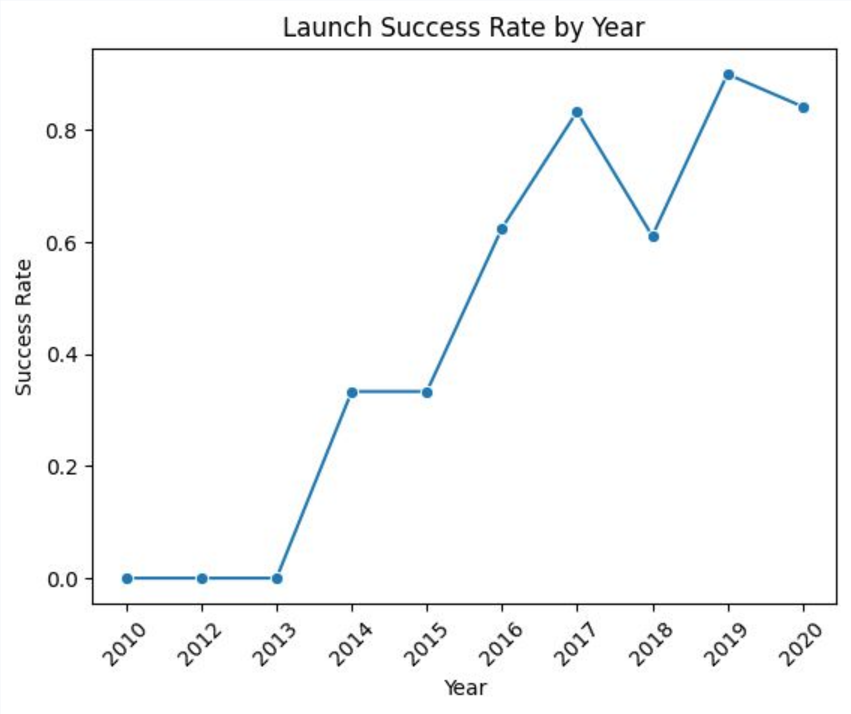
Payload vs. Orbit Type

- This plot is to understand how payload mass correlates with different orbits.
- It shows Polar, LEO, and ISS orbits tend to carry heavier payloads with higher success.



Launch Success Yearly Trend

- This line chart is to track improvements in SpaceX launch reliability over time
- It shows a strong upward trend in success rate from 2013 to 2017, showing learning and tech improvements



All Launch Site Names

There are 4 distinct launch site names. This query identified all unique launch locations in the dataset

```
In [10]: %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE

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

Out[10]:

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

Launch Site Names Begin with 'CCA'

- Top 5 records with Launch Site Names beginning with ‘CCA’. LIMIT keyword Limited to first 5 matching records.

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

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

Out[11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (p
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 (p
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	N
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	N
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	N

Total Payload Mass

- Use SUM to calculate the total payload mass which is 45596 kg

```
In [12]: %sql SELECT SUM("Payload_Mass__kg_") AS Total_Payload FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)'

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

Out[12]:

Total_Payload
45596

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is 2928.4 kg which was calculated using AVG.

```
In [13]: %sql SELECT AVG("Payload_Mass__kg_") AS Average_Payload FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1'

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

Out[13]:

Average_Payload
2928.4

First Successful Ground Landing Date

- Query: %sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "Payload_Mass__kg_" > 4000 AND "Payload_Mass__kg_" < 6000

```
In [15]: %sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "Payload_Mass

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

Out[15]:

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

Successful Drone Ship Landing with Payload between 4000 and 6000

- The names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 are:

Out[16]:

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

- Query: %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") AS Total_Number FROM SPACEXTABLE GROUP BY "Mission_Outcome"

Total Number of Successful and Failure Mission Outcomes

- Query: %sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE "Payload_Mass__kg_" = (SELECT MAX("Payload_Mass__kg_") FROM SPACEXTABLE)
- Done.

  Out[18]:

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

Boosters Carried Maximum Payload

- The names of the booster which have carried the maximum payload mass

```
* sqlite:///my_data1.db
Done.
Out[20]:
```

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

- %sql SELECT SUBSTR("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE WHERE "Landing_Outcome" LIKE 'Failure (drone ship)%' AND SUBSTR("Date", 1, 4) = '2015'

- Used a subquery to find the booster that carried the maximum payload.

2015 Launch Records

- Query: %sql SELECT SUBSTR("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE WHERE "Landing_Outcome" LIKE 'Failure (drone ship)%' AND SUBSTR("Date", 1, 4) = '2015'

Out[20]:

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

- Used SUBSTR(Date, 6, 2) and LIKE '2015%' to filter missions from 2015.
- Displayed landing outcomes, boosters, and launch sites.

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

- %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
- Out[21]:

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

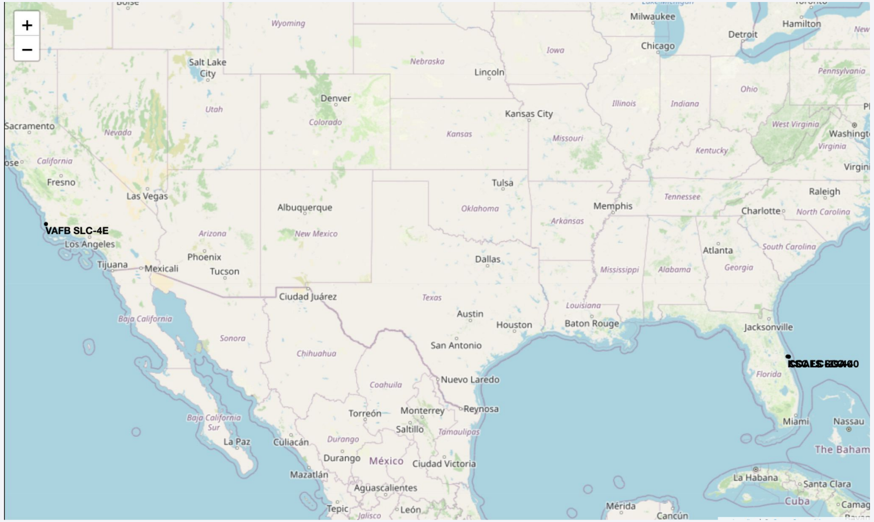
- Counted all landing outcomes between 2010-06-04 and 2017-03-20.
- Ranked them in descending order based on frequency.

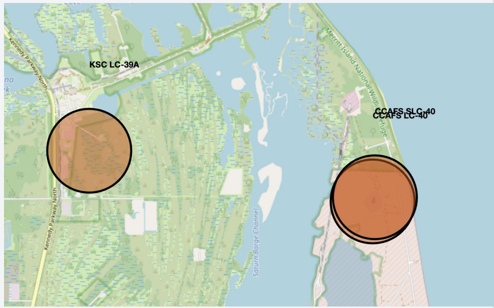
Section
3

Launch Sites Proximities Analysis

Mapped Launch Sites Using Folium

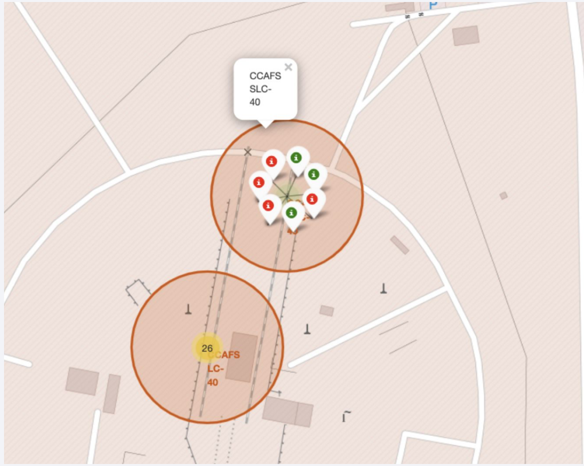
- The folium map marks the geographical locations of all major SpaceX launch sites in the United States. The markers are labeled for clear identification.
- Markers indicate the exact coordinates of each launch site.
- Launch sites shown:
    - VAFB SLC-4E – California (West Coast)
    - CCAFS LC-40, CCAFS SLC-40, and KSC LC-39A – Florida (East Coast)

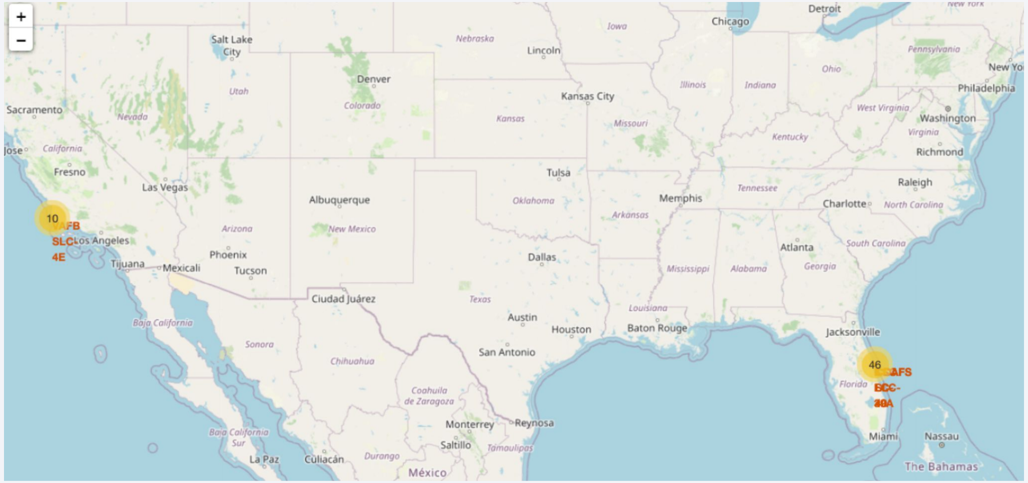




Visualizing Launch Sites and Outcomes with Folium

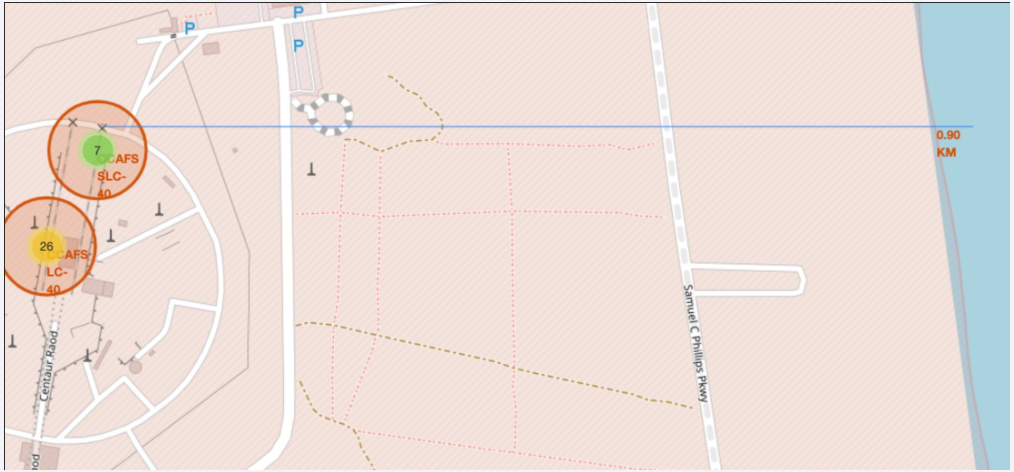
- Markers show all unique launch site locations.
- Circles highlight activity zones around sites.
- Color-coded markers:
- Success and Failure color coded — quickly visualize launch outcomes.
- Marker Clusters prevent overlap for dense launch locations.
- Key Insight: Launch sites are coastal, likely for safety and recovery. CCAFS and KSC have high activity with varied outcomes.





<Folium Map Screenshot 3>

- Blue line shows 0.90 KM from launch site to coastline
- Circle Markers: Indicate launch success (7) and total launches (26)
- Proximity to the coast (<1 KM) supports safe rocket trajectory and recovery.
- Launch sites are strategically placed near accessible infrastructure and open space.

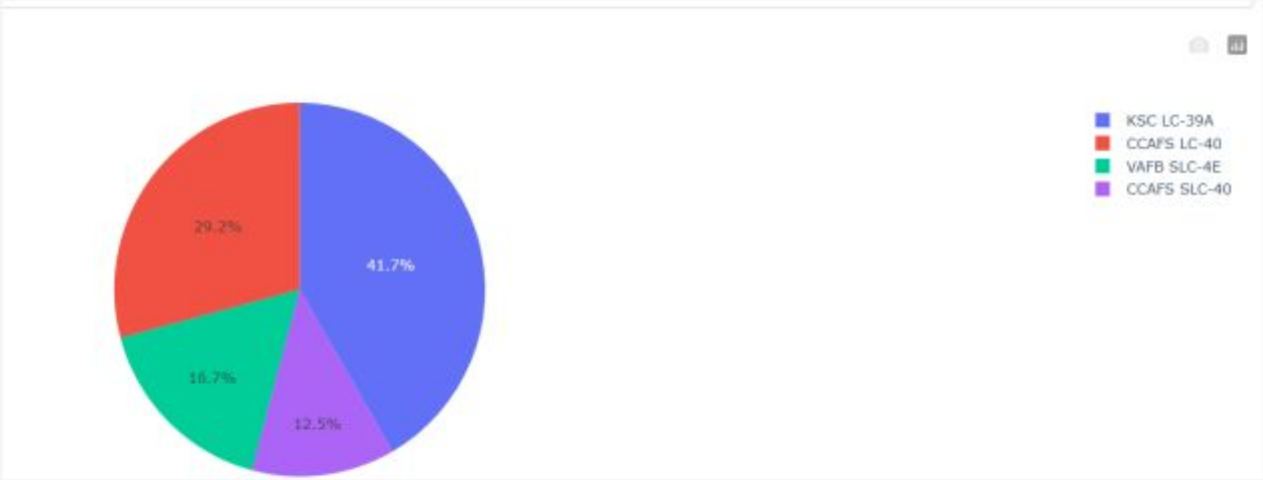


Section
4

Build a Dashboard with Plotly Dash

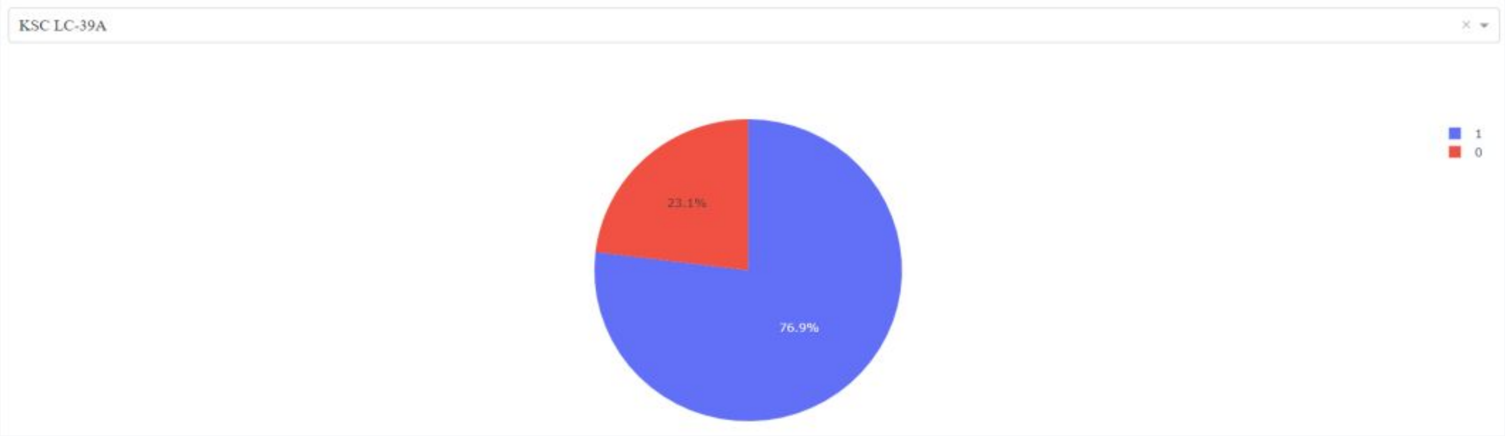
Launch Site Pie Chart

- The chart displays the distribution of successful launches across all launch sites.
- KSC LC-39A leads with 41.7%, showing the highest success count.
- Followed by CCAFS LC-40 (29.2%), VAFB SLC-4E (16.7%), and CCAFS SLC-40 (12.5%).



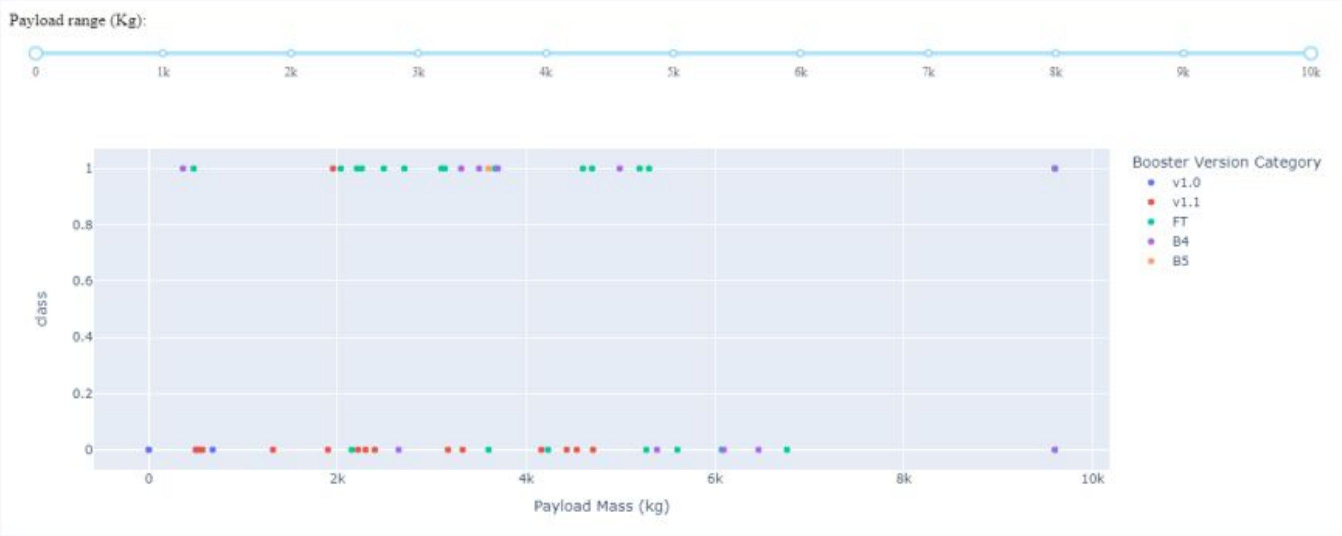
Launch Success Rate at KSC LC-39A

- The pie chart shows the success vs failure ratio at KSC LC-39A.
- With 76.9% of launches being successful (blue) and only 23.1% failures (red), this site has the highest success rate among all launch sites.



Payload vs. Launch Outcome Scatter Plot

- This scatter plot visualizes payload mass (kg) vs launch outcome (1 = success, 0 = failure) for all sites.
- The payload range [3600–5300 kg] shows the highest cluster of successful launches.
- FT booster version has the highest success rate, followed by B4.
- Very few successful launches occur above 5400 kg payload, with a single success at 9600 kg.

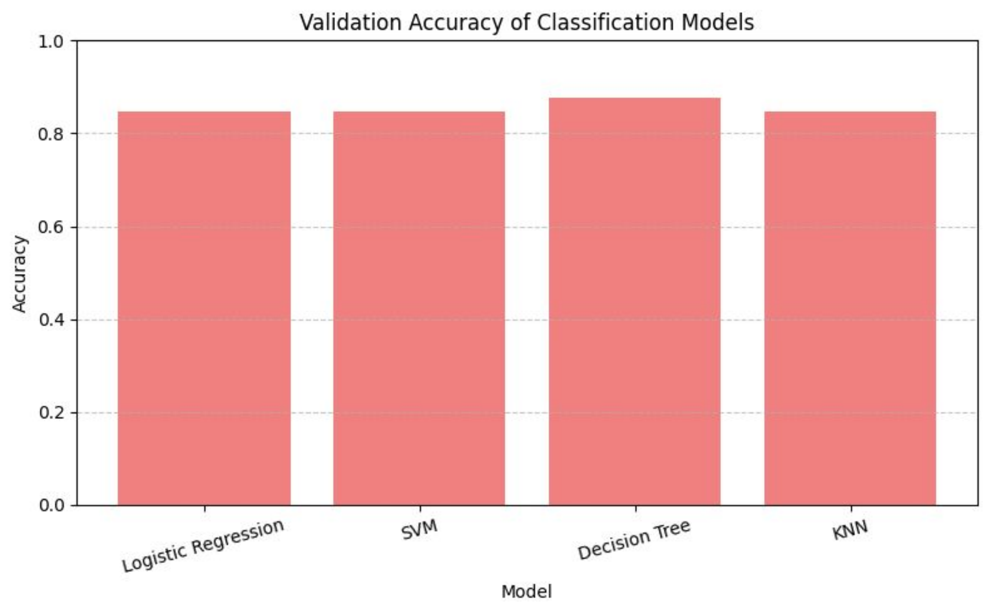


Section
5

Predictive Analysis (Classification)

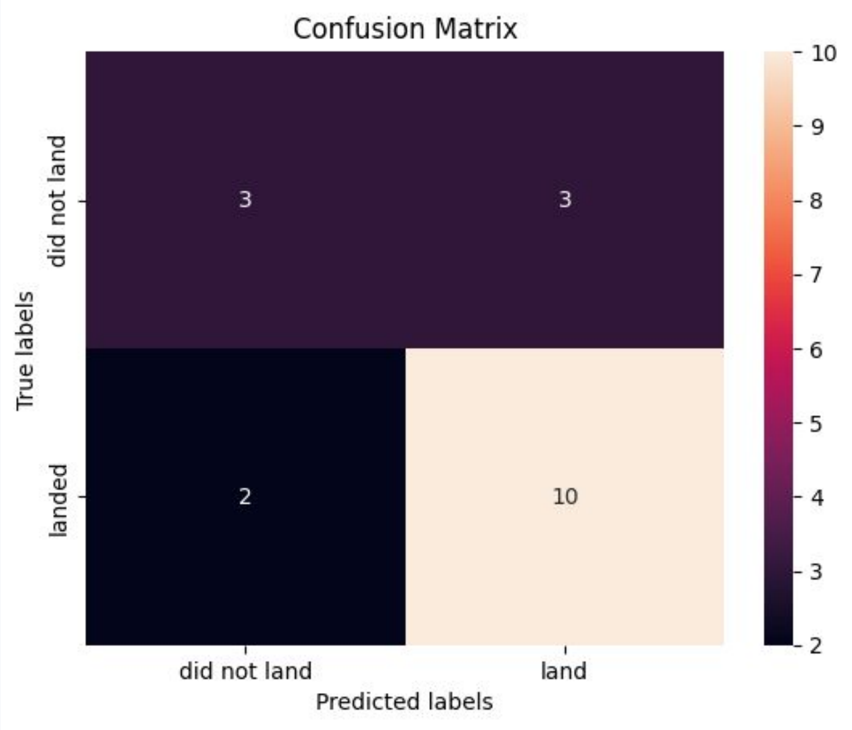
Classification Accuracy

Decision Tree achieved the highest validation accuracy (~0.88) among all the models tested



Confusion Matrix

- The model is slightly more likely to predict landings than failures.
- Despite the few misclassifications, the Decision Tree model still has the highest validation accuracy (~88%), making it the best performer among the evaluated models



Conclusions

- Launch Site Insights: KSC LC-39A had the most launches and the highest success rate (~77%).
- Payload Analysis: Payloads between 3600–5300 kg had the highest success rate.
- Payloads above 5400 kg had very low success.
- Booster Version: FT was the most reliable booster with the highest launch success
- Model Performance: The Decision Tree Classifier performed best with 87.5% validation accuracy. It correctly predicted 10 landings and 3 non-landings with few errors.

Appendix

- All relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets are in this Github

Thank you!