



IBM Developer  
SKILLS NETWORK

# Winning Space Race with Data Science

Julio Muñoz  
05-06-2025



# Outline

---

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

# Executive Summary

---

- Summary of methodologies
  - Collected data from Wikipedia, SpaceX API, and web scraping.
  - Cleaned, merged, and encoded data.
  - Performed EDA with SQL + visualizations.
  - Built interactive maps (Folium) and dashboards (Plotly Dash).
  - Trained and tuned classification models (LogReg, SVM, Decision Tree, Random Forest).
- Summary of all results
  - Landing success improved over time (>60% in recent years).
  - Launch site and payload mass strongly affect landing outcome.
  - Random Forest model achieved highest predictive performance.

# Introduction

---

- Project background and context
  - SpaceX offers Falcon 9 rocket launches for \$62 million, significantly lower than traditional providers who charge up to \$165 million. This cost advantage is largely due to the reusability of the Falcon 9's first stage. Predicting landing success can help estimate launch costs and provide insights for other companies competing in the aerospace market.
- Problems you want to find answers
  - Key Questions:
    - Can we predict if the Falcon 9 first stage will land successfully?
    - How do payload mass, launch site, and orbit type affect landing outcome?
    - Which machine learning model gives the best prediction performance?



Section 1

# Methodology

# Methodology

---

## Executive Summary

- **Data collection methodology:**
  - Wikipedia: for historical data on Falcon 9 launches.
  - SpaceX REST API: to retrieve technical details about launches, payloads, and launch sites.
  - Web Scraping: using BeautifulSoup and requests to extract unstructured data from public web pages.
- **Perform data wrangling**
  - Removed irrelevant or duplicate columns.
  - Merged datasets from different sources (e.g., rocket specs, payload info, launch sites).
  - Handled missing values by either filling or dropping them.
  - Converted categorical variables into dummy variables using OneHotEncoding.
  - Normalized numerical features to improve model performance.

# Methodology

---

## Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
  - Visualizations and SQL queries were used to explore the dataset: What is the overall success rate of landings? How do orbit type and payload mass affect landing outcomes? Which launch sites have higher success rates? Tools used: Matplotlib and Seaborn for plotting correlations and trends. pandasql for writing SQL-like queries on DataFrames.
- Perform interactive visual analytics using Folium and Plotly Dash
  - Folium: An interactive map was created showing launch sites, landing zones, and distance lines between them. Plotly Dash: A web dashboard was developed to allow users to: Filter by launch site. Explore the relationship between payload mass and landing success. Visualize success rates by orbit type.

# Methodology

---

## Executive Summary

- **Perform predictive analysis using classification models**

- This is a binary classification problem: Landing Success = Yes or No.
- Several models were trained using scikit-learn, including:
  - Logistic Regression
  - Support Vector Machine (SVM)
  - Decision Tree
  - Random Forest
- Building, Tuning, and Evaluating Classification Models
- Model Building:
  - Data was split into training and test sets.
  - Classifiers were trained on the training set.
  - Hyperparameter Tuning:
    - GridSearchCV was used to optimize model parameters such as:
      - max\_depth for decision trees
      - C for SVM regularization
      - n\_estimators for Random Forest
- Model Evaluation:
  - Evaluated using metrics like:
    - Accuracy
    - Precision
    - Recall
    - F1-score
  - Confusion Matrix
- The best-performing model was selected for final predictions.



# Data Collection

---

- Describe how data sets were collected.
  - We collected launch data from multiple sources to create a unified dataset for analysis:
    - SpaceX REST API: Used to retrieve structured data on launches, payloads, and cores.
    - Wikipedia: Used for cross-validation and to gather supplementary historical launch information.
    - Web Scraping: Used `BeautifulSoup` to extract additional data from SpaceX-related web pages.

Wikipedia

SpaceX API

Web Scraping

# Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts
- RESTful API Access: We used the public SpaceX REST API to collect real-time launch data.
- HTTP GET Requests: Data was retrieved using GET methods on endpoints like /launches, /rockets, /launchpads, and /cores.
- JSON Response: The API returned structured JSON data, which we parsed and normalized for analysis.
- Pagination Handling: For endpoints with large data sets, we handled pagination to ensure full data retrieval.
- Data Storage: The collected JSON data was flattened and stored in Pandas DataFrames for further processing.
- Automated Collection: The data collection process was scripted using Python and run in Jupyter Notebooks.
- Add the GitHub URL of the completed SpaceX API calls notebook ([must include completed code cell and outcome cell](https://github.com/juliocmg88/IBMDDataScienceCapstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb)), as an external reference and peer-review purpose
- <https://github.com/juliocmg88/IBMDDataScienceCapstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb>



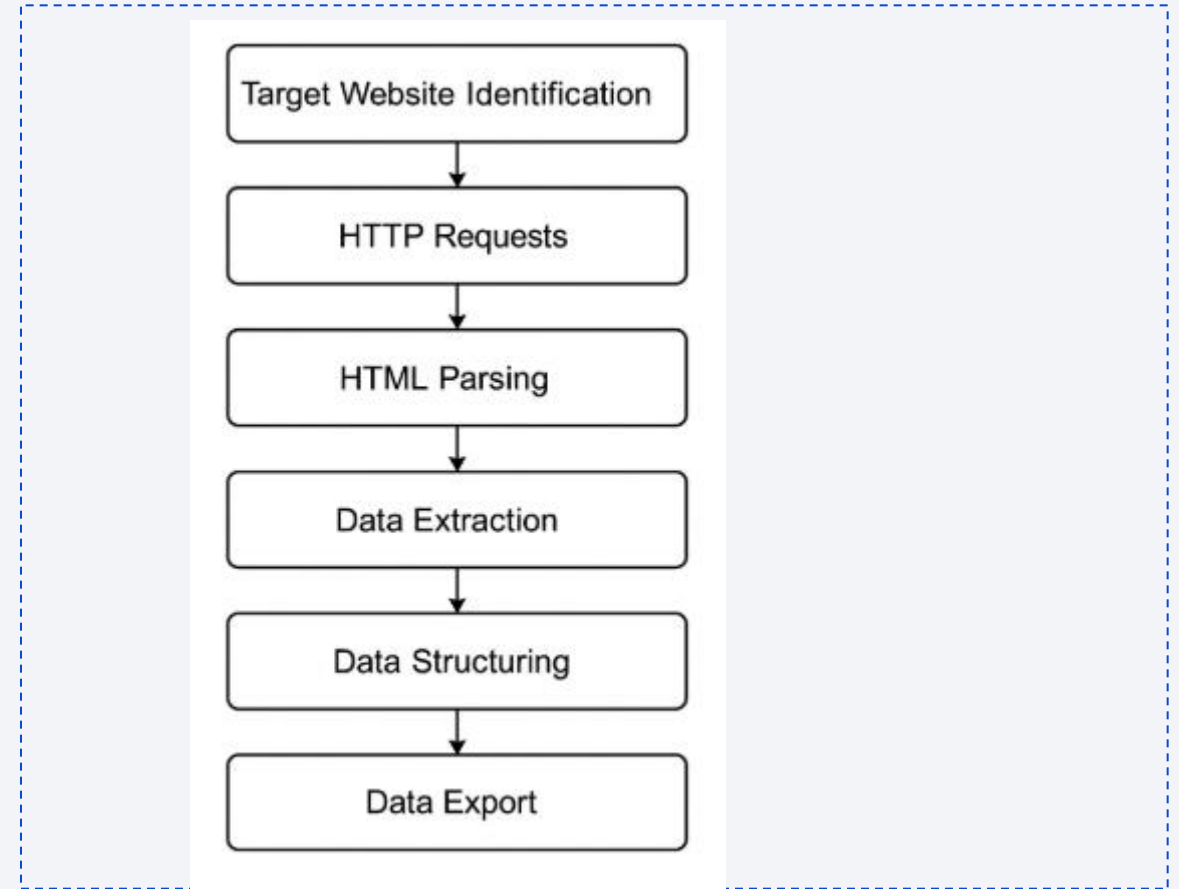
# Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Target Website Identification
- HTTP Requests
- HTML Parsing
- Data Extraction
- Data Cleaning
- Data Structuring
- Data Export

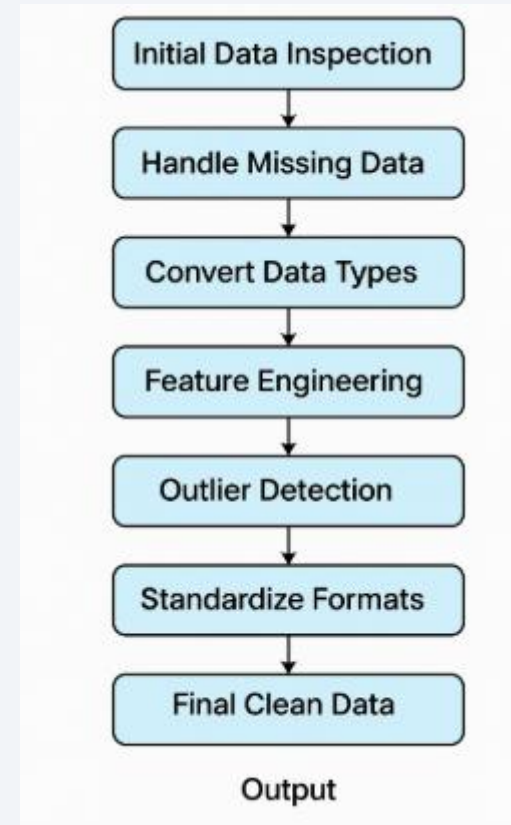
- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose

- <https://github.com/juliocmg88/IBMDDataScienceCapstone/blob/main/jupyter-labs-webscraping.ipynb>



# Data Wrangling

- Describe how data were processed
  - Initial Data Inspection: We examined the dataset for missing values, inconsistent formats, and data types.
  - Handling Missing Data: Applied strategies like imputation (mean, median) or row removal depending on the context.
  - Data Type Conversion: Converted string dates to datetime format, integers to floats, etc., for consistency.
  - Feature Engineering: Created new derived features (e.g., success rate, time between launches).
  - Duplicate Removal: Identified and removed exact or near-duplicate entries.
  - Outlier Detection: Detected and managed anomalies using statistical methods (e.g., IQR, Z-score).
  - Standardization: Normalized column names, units of measure, and categorical values.
  - Final Structuring: Organized the clean dataset into DataFrames ready for analysis or modeling.
- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose
  - <https://github.com/juliocmg88/IBMDDataScienceCapstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb>



# EDA with Data Visualization

---

- Summarize what charts were plotted and why you used those charts
  - To explore launch success patterns, we used various charts:
  - Scatter Plots
    - Flight Number vs. Payload Mass: Detected trends in mission size and outcomes.
    - Flight Number vs. Launch Site: Revealed performance variation across launch sites.
    - Payload Mass vs. Launch Site / Orbit Type: Assessed the impact of payload by category.
    - Flight Number vs. Orbit Type: Tracked evolution of mission types over time.
  - Bar Chart
    - Orbit Type vs. Success Rate: Compared performance across different orbit destinations.
  - Line Chart
    - Success Rate Over Time: Highlighted historical improvements in mission success.
- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose
- <https://github.com/juliocmg88/IBMDDataScienceCapstone/blob/main/edadataviz.ipynb>











# EDA with SQL

---

- Using bullet point format, summarize the SQL queries you performed
  - Created a filtered table excluding null launch dates
  - Retrieved distinct launch sites used by SpaceX
  - Selected sample launches from specific launch sites (e.g., CCAFS)
  - Calculated total payload mass for NASA (CRS) missions
  - Found average payload for booster version F9 v1.1
  - Identified the earliest successful ground pad landing
  - Queried missions with drone ship landings and mid-range payloads
  - Counted occurrences of each landing outcome
  - Extracted booster reuse information and grouped by key features
  - Analyzed launch trends by date, site, orbit, and customer
- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose
  - [https://github.com/juliocmg88/IBMDataScienceCapstone/blob/main/jupyter-labs-eda-sql-coursera\\_sqlite.ipynb](https://github.com/juliocmg88/IBMDataScienceCapstone/blob/main/jupyter-labs-eda-sql-coursera_sqlite.ipynb)

# Build an Interactive Map with Folium

---

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
    -  Markers
      - Plotted each launch site with blue icons and site names in popups
    -   Circle Markers
      - Green: Successful launches
      - Red: Failed launches
      - → Visualized outcome by location
    -  Text Labels (DivIcon)
      - Added site names and distances directly on the map for clarity
    -  Marker Clusters
      - Grouped nearby launches to reduce clutter and improve navigation
  -  Distance Markers
    - Showed distances from launch sites to coastlines in kilometers
  -  Mouse Position Plugin
    - Displayed live coordinates for spatial reference
  -  Result: An informative and interactive map for launch site analysis
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose
    - [https://github.com/juliocmg88/IBMDataScienceCapstone/blob/main/lab\\_jupyter\\_launch\\_site\\_location.ipynb](https://github.com/juliocmg88/IBMDataScienceCapstone/blob/main/lab_jupyter_launch_site_location.ipynb)

# Build a Dashboard with Plotly Dash

---

- Summarize what plots/graphs and interactions you have added to a dashboard

## Graphs Included

### Pie Chart

- Total successful launches by site
- Success vs. failure for selected site

### Scatter Plot

- Correlation between payload mass and launch success
- Colored by booster version

### Interactions

### Launch Site Dropdown

- Filter charts by site or view all

### Payload Range Slider


- Focus on specific payload intervals (0–10,000 kg)

☒ Enables dynamic, focused analysis of mission performance

- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose
- <https://github.com/juliocmg88/IBMDataScienceCapstone/blob/main/spacex-dash-app.py>

# Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model

-  Model Development Steps
- Data Preprocessing
- Applied StandardScaler to normalize features
- Split dataset into training (80%) and testing (20%) sets
- Model Selection
- Trained 4 classification algorithms:
  - Logistic Regression
  - Support Vector Machine (SVM)

- Decision Tree
- K-Nearest Neighbors (KNN)
- Hyperparameter Tuning
- Used GridSearchCV with 10-fold cross-validation
- Optimized parameters like:
  - C, gamma, and kernel for SVM
  - criterion, max\_depth, etc., for Decision Tree
  - n\_neighbors and p for KNN
- Model Evaluation
- Compared models using accuracy score and classification report

- Selected the best performing model based on test accuracy and cross-validated scores

• Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

- [https://github.com/juliocmg88/IBMDaScienceCapstone/blob/main/Space X Machine%20Learning%20Prediction Part 5.ipynb](https://github.com/juliocmg88/IBMDaScienceCapstone/blob/main/Space%20X%20Machine%20Learning%20Prediction%20Part%205.ipynb)

## Predictive Analysis (Classification)



# Results

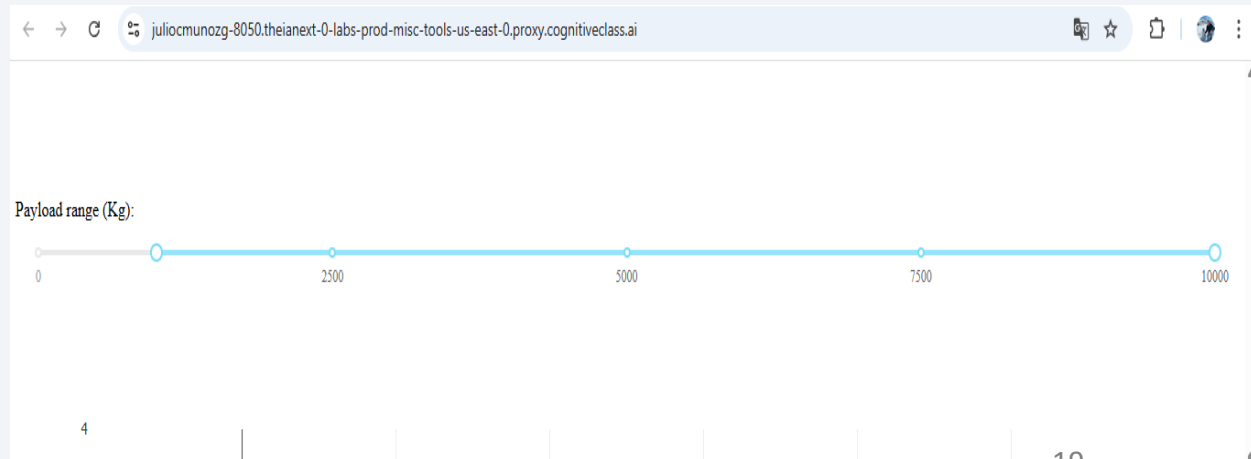
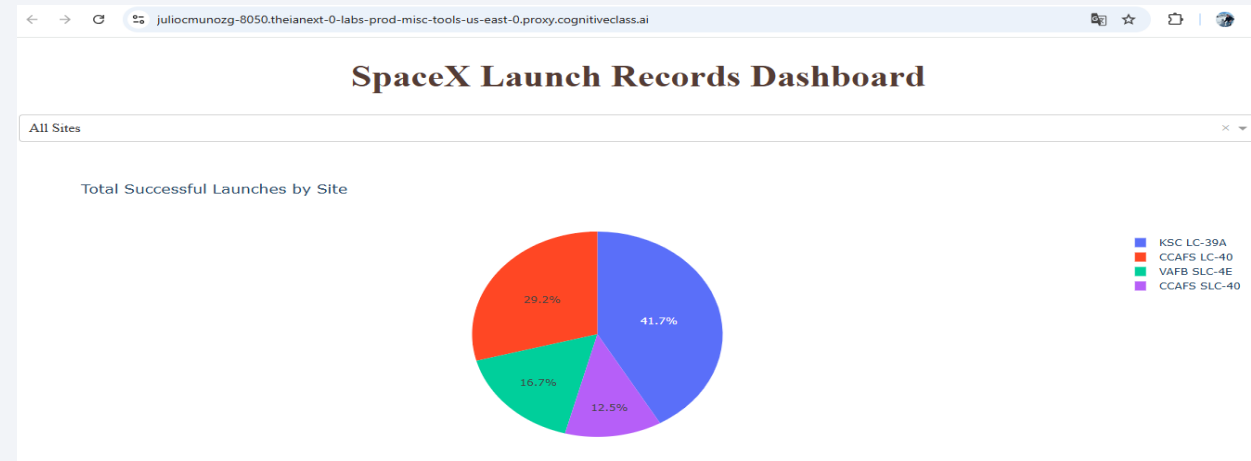
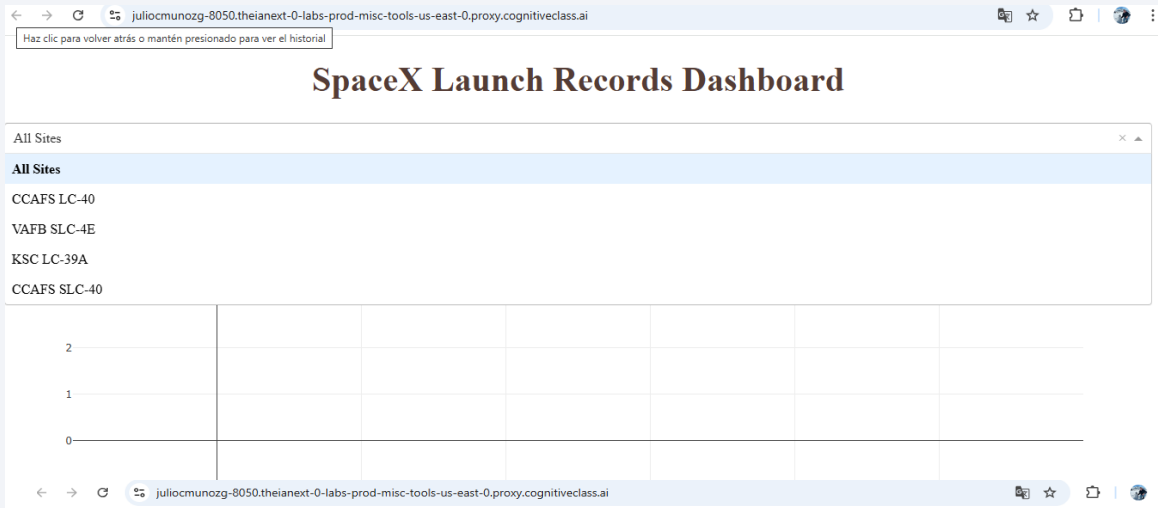
---

- Exploratory data analysis results
  - Launch outcomes vary significantly across different launch sites
  - Most successful launches occurred at KSC LC-39A
  - Payload mass impacts success likelihood: very low or very high payloads had lower success rates
  - Orbit type and booster version also influence mission success
  - Success rates have improved over time, as seen in yearly trends
- Interactive analytics demo in screenshots
- Predictive analysis results



# Results

- Interactive analytics demo in screenshots



# Results

---

- Predictive analysis results

Four models evaluated:

Logistic Regression, SVM, Decision Tree, KNN

Best performance achieved using [Insert Best Model, e.g., SVM with RBF kernel]

Accuracy improved through GridSearchCV and cross-validation

Final model shows reliable prediction of launch success based on payload, site, and booster

✓ Predictive model enables data-driven insights to support mission planning and risk assessment





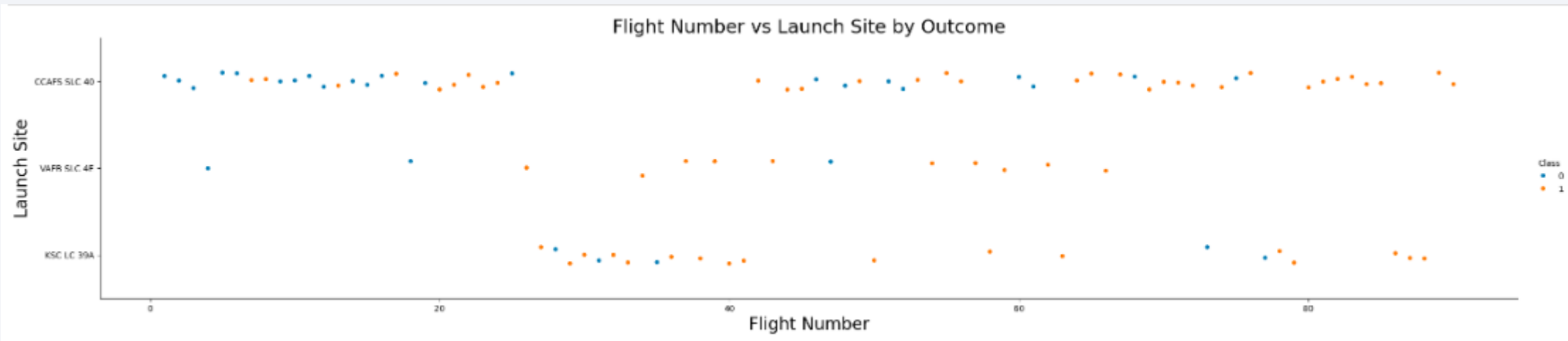
Section 2

# Insights drawn from EDA



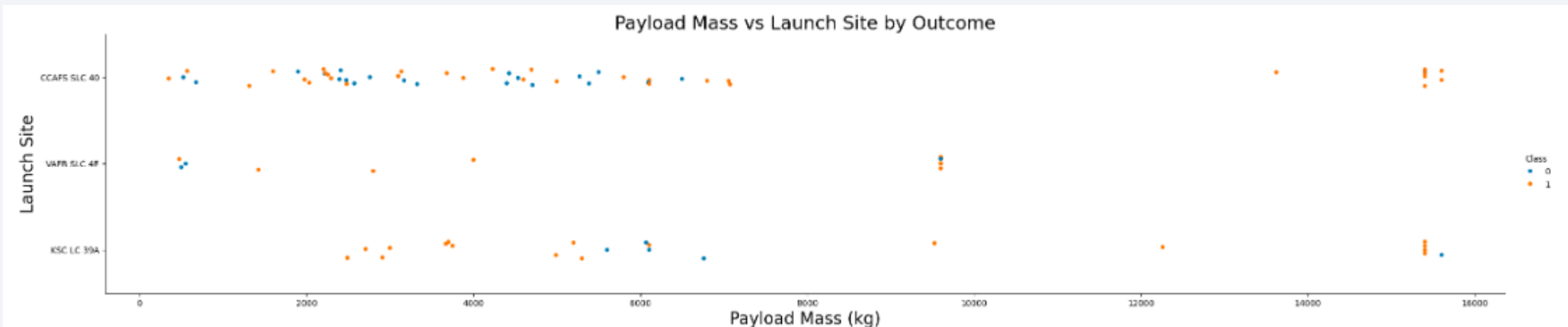
# Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site
- Show the screenshot of the scatter plot with explanation
  - CCAFS SLC 40: Most used site. Early missions show mixed results, but success rates improve over time.
  - VAFB SLC 4E: Fewer launches with a higher proportion of failed landings. No clear improvement trend.
  - KSC LC 39A: Used in later missions with a high success rate, reflecting more mature landing capabilities.
  - Overall Trend: Success rates increase with flight number, indicating learning and technological progress.




# Payload vs. Launch Site

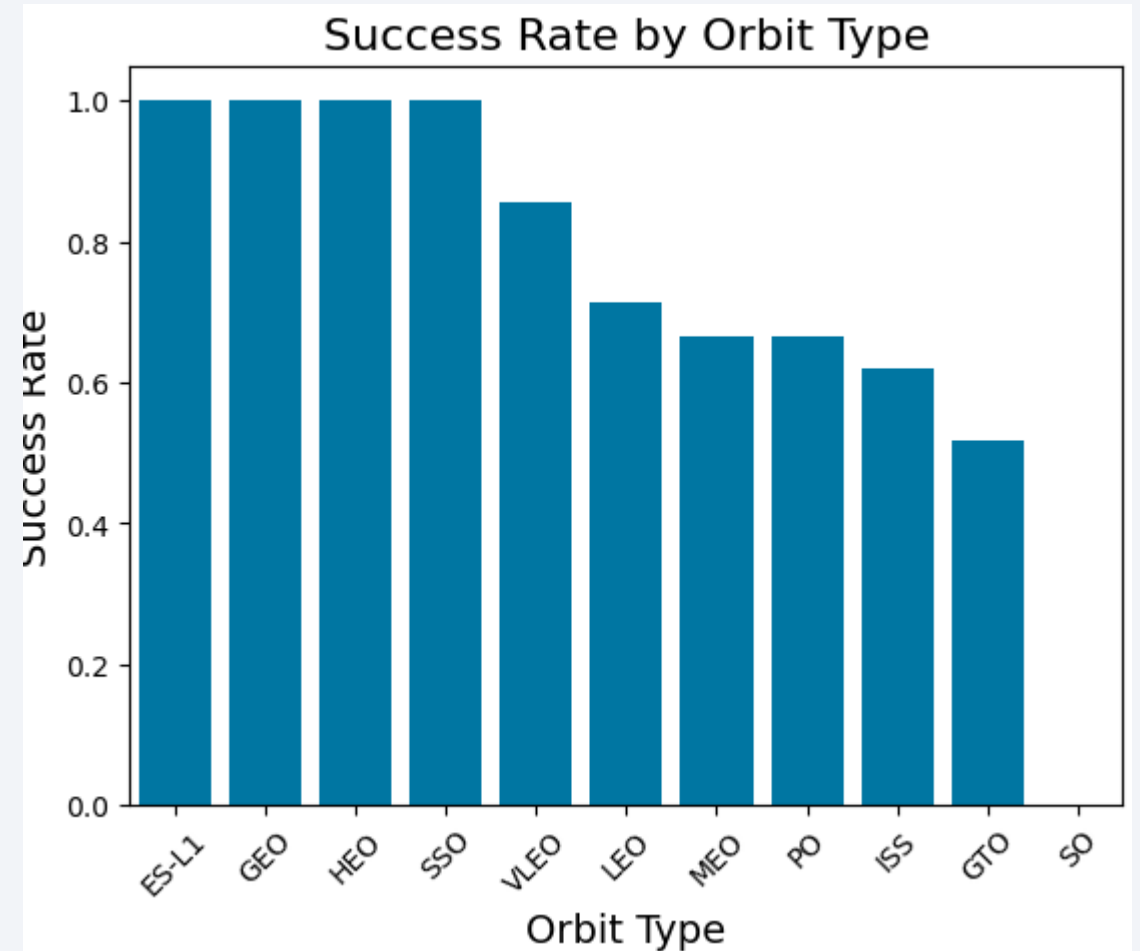
- Show a scatter plot of Payload vs. Launch Site
- Show the screenshot of the scatter plot with explanations
  - VAFB SLC 4E: Only light to medium payloads (<10,000 kg) – no heavy launches.
  - CCAFS SLC 40: Wide range of payloads with mixed outcomes.
  - KSC LC 39A: Handles very heavy payloads (>10,000 kg) with mostly successful landings.
  - Heavier payloads are often launched from KSC, suggesting higher mission confidence and advanced launch infrastructure.








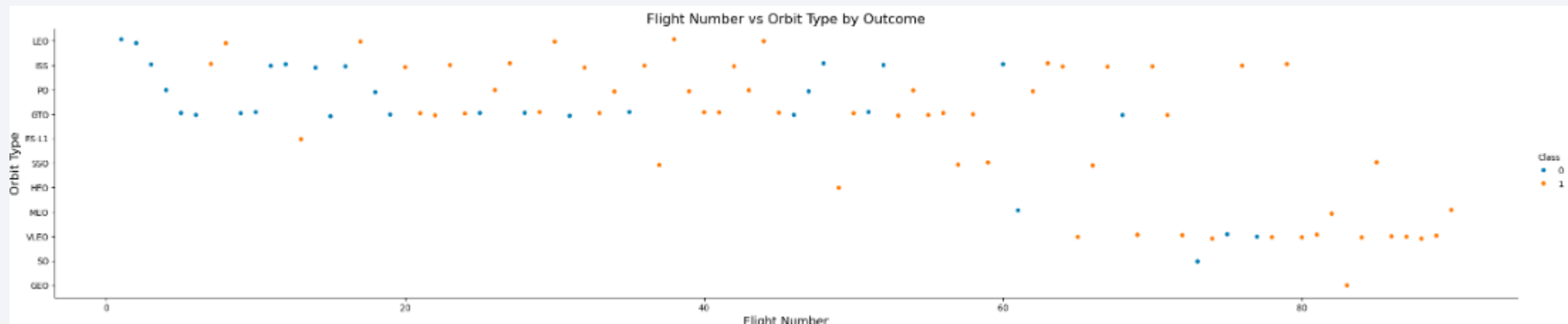
# Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type
- Show the screenshot of the scatter plot with explanations
  -  Highest success: ES-L1, GEO, HEO, SSO – 100% success rate.
  -  Moderate success: VLEO, LEO, MEO, PO, ISS – 60–85% success.
  -  Lowest success: GTO, SO – below 50%, possibly due to mission complexity.






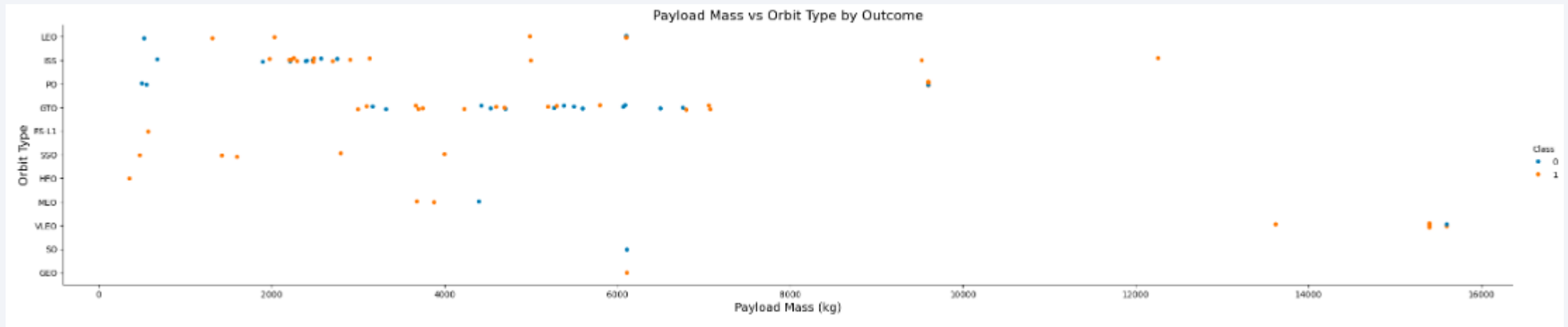
# Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type
- Show the screenshot of the scatter plot with explanations
  -  LEO: Success improves with flight experience, indicating learning and reliability gains.
  -  GTO: No clear trend; outcomes appear independent of flight number, likely due to mission complexity.
  -  Other orbits: Varying success; high-performing orbits like SSO and GEO mostly appear in later flights with strong results.



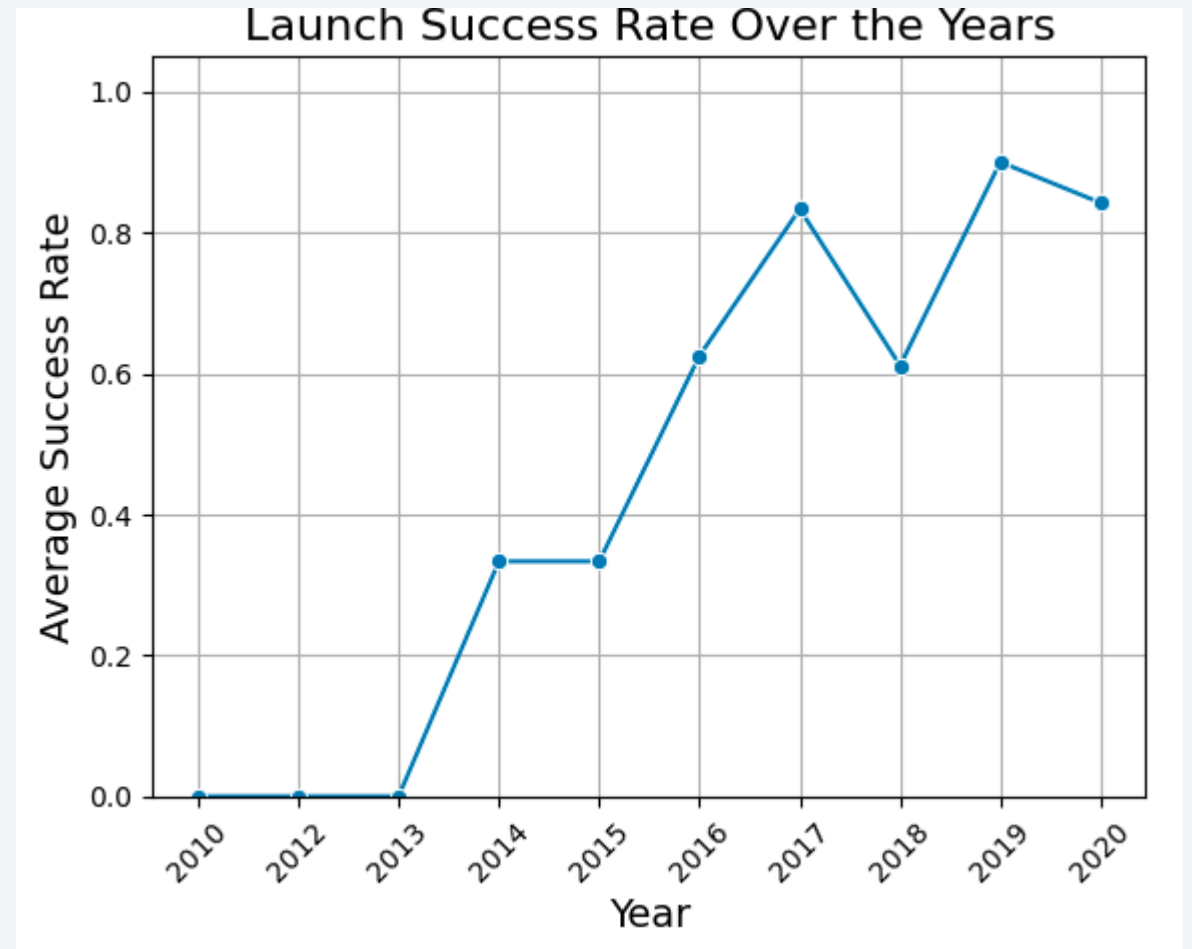
# Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type
- Show the screenshot of the scatter plot with explanations
  -  LEO, ISS, PO: Heavier payloads show strong landing success rates.
-  GTO: No clear success pattern — both successful and failed landings occur across payload sizes.
-  Insight: GTO may involve more complex dynamics that affect landing outcome beyond payload mass alone.



# Launch Success Yearly Trend

- Show a line chart of yearly average success rate
- Show the screenshot of the scatter plot with explanations
  - ✅ Success rate steadily increased from 2013 to 2020.
  - 📈 Major improvements between 2014–2017, reflecting rapid technological advancements.
  - 📉 2018 dip followed by strong recovery in 2019 and 2020, reaching consistent high performance.
  - 💡 Trend suggests continuous learning, system upgrades, and increased mission reliability over time.



# All Launch Site Names

---

- Find the names of the unique launch sites
  - CCAFS SLC 40
  - VAFB SLC 4E
  - KSC LC 39A
- Present your query result with a short explanation here
  - `unique_launch_sites = df['LaunchSite'].unique()`
  - `print(unique_launch_sites)`
  - `df['LaunchSite']` accesses the column containing launch site names.
  - `.unique()` returns an array of unique launch site names (without duplicates).



# Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with 'CCA'

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.5773
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.5773
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.5773
3	4	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005	-80.5773
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.5773
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005	-80.5773

- Present your query result with a short explanation here
  - `df[df['LaunchSite'].str.startswith('CCA')].head(5)`
  - `df['LaunchSite'].str.startswith('CCA')` filters the DataFrame for rows where the LaunchSite column starts with "CCA".
  - `.head(5)` returns the first 5 matching records.

# Total Payload Mass

---

- Calculate the total payload carried by boosters from NASA
  - 0 kg, because launches were made by SpaceX
- Present your query result with a short explanation here
  - `total_nasa_payload = df[df['BoosterVersion'].str.contains('NASA', case=False)]['PayloadMass'].sum()`
  - `print(total_nasa_payload)`
  - `df['BoosterVersion'].str.contains('NASA', case=False):`
  - Filters the DataFrame for rows where the BoosterVersion mentions NASA (case-insensitive).
  - `['PayloadMass']`: Selects the payload mass column.
  - `.sum()`: Calculates the total payload mass for those filtered rows.

# Average Payload Mass by F9 v1.1

---

- Calculate the average payload mass carried by booster version F9 v1.1
  - `f9_v1_1_data = data_falcon9[data_falcon9['BoosterVersion'] == 'F9 v1.1']`
  - `average_payload_f9_v1_1 = f9_v1_1_data['PayloadMass'].mean()`
  - `print("Average payload mass carried by booster version F9 v1.1:", average_payload_f9_v1_1, "kg")`
- Present your query result with a short explanation here
  - Filtering: We select only the rows where the booster version is F9 v1.1.
  - Mean Calculation: We calculate the mean (`.mean()`) of the PayloadMass column for those filtered rows.
  - Result: The result is the average payload mass in kilograms.

# First Successful Ground Landing Date

---

- Find the dates of the first successful landing outcome on ground pad
  - `successful_ground_landings = data_falcon9[data_falcon9['Outcome'] == 'True RTLS']`
  - `first_successful_ground_landing_date = successful_ground_landings['Date'].min()`
  - `print("First successful landing on ground pad (RTLS) occurred on:", first_successful_ground_landing_date.date())`
  - December 22, 2015
- Present your query result with a short explanation here
  - We filtered the launch data for landings with the outcome "True RTLS", indicating a successful landing on a ground pad, and then found the earliest date among those entries.

## Successful Drone Ship Landing with Payload between 4000 and 6000



---

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
  - `filtered_boosters = data_falcon9[`
  - `(data_falcon9['Outcome'] == 'True ASDS') &`
  - `(data_falcon9['PayloadMass'] > 4000) &`
  - `(data_falcon9['PayloadMass'] < 6000)`
  - `]`
  - `booster_names = filtered_boosters['BoosterVersion'].unique().tolist()`
  - `print("Boosters with successful drone ship landing and payload between 4000 and 6000 kg:")`
  - `for name in booster_names:`
  - `print("-", name)`
  - `Falcon 9 Booster`
- Present your query result with a short explanation here
  - We filtered the data for:Landings with outcome "True ASDS" (successful drone ship landing).Payload mass between 4000 kg and 6000 kg.Then we listed the unique BoosterVersion values that met those criteria.



# Total Number of Successful and Failure Mission Outcomes

---

- Calculate the total number of successful and failure mission outcomes
  - `df['MissionStatus'] = df['Outcome'].apply(lambda x: 'Success' if x.startswith('True') else 'Failure')`
  - `mission_outcome_counts = df['MissionStatus'].value_counts()`
  - `mission_outcome_counts.to_dict()`
  -  60 successful mission outcomes
  -  30 failed mission outcomes
- Present your query result with a short explanation here
  - We classified each Outcome as:
  - Success if it starts with "True" (indicating a successful landing or mission),
  - Failure otherwise.
  - Then, we counted how many times each outcome type appeared

# Boosters Carried Maximum Payload

---

- List the names of the booster which have carried the maximum payload mass
  - `booster_payload = df.groupby('Serial')['PayloadMass'].sum().reset_index()`
  - `max_payload = booster_payload['PayloadMass'].max()`
  - `max_payload_boosters = booster_payload[booster_payload['PayloadMass'] == max_payload]`
- Present your query result with a short explanation here
  - The booster with the **maximum total payload mass** is **Serial B1049**, which carried a total of **76,480 kg** across all its missions.
  - To find this, we grouped the dataset by the Serial column (representing each unique booster), summed the PayloadMass for each one, and identified the booster with the highest total. Booster B1049 emerged as the top performer in terms of total mass delivered to orbit.

# 2015 Launch Records

---

- List the failed landing\_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - No failed drone ship landings occurred in the year **2015**
- Present your query result with a short explanation here
  - `df['Date'] = pd.to_datetime(df['Date'])`
  - `filtered_df = df[`
  - `(df['Date'].dt.year == 2015) &`
  - `(df['Outcome'].str.contains('Failure', na=False)) &`
  - `(df['Outcome'].str.contains('drone ship', case=False, na=False))`
  - `]`
  - `result_df = filtered_df[['Outcome', 'BoosterVersion', 'LaunchSite']]`
  - After applying these filters, no matching records were found. This indicates that either no attempts were made to land on a drone ship in 2015, or all such attempts were either successful or not classified as failures in the dataset.

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

---

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
  - None None – 9 times (no landing attempt or data not recorded)
  - True ASDS – 5 successful landings on a drone ship
  - False ASDS – 4 failed drone ship landings
  - True Ocean – 3 successful ocean landings
  - True RTLS – 3 successful ground pad landings
- Present your query result with a short explanation here
  - `df = pd.read_csv("/mnt/data/dataset_part_2.csv")`
  - `df['Date'] = pd.to_datetime(df['Date'])`
  - `filtered_df = df[`
  - `(df['Date'] >= '2010-06-04') &`
  - `(df['Date'] <= '2017-03-20')`
  - `]`
  - `landing_outcome_counts_new =`  
`filtered_df['Outcome'].value_counts().reset_index()`
  - `landing_outcome_counts_new.columns =`  
`['LandingOutcome', 'Count']`

A satellite view of Earth from space, showing the curvature of the planet and city lights at night. The background is a deep blue gradient.

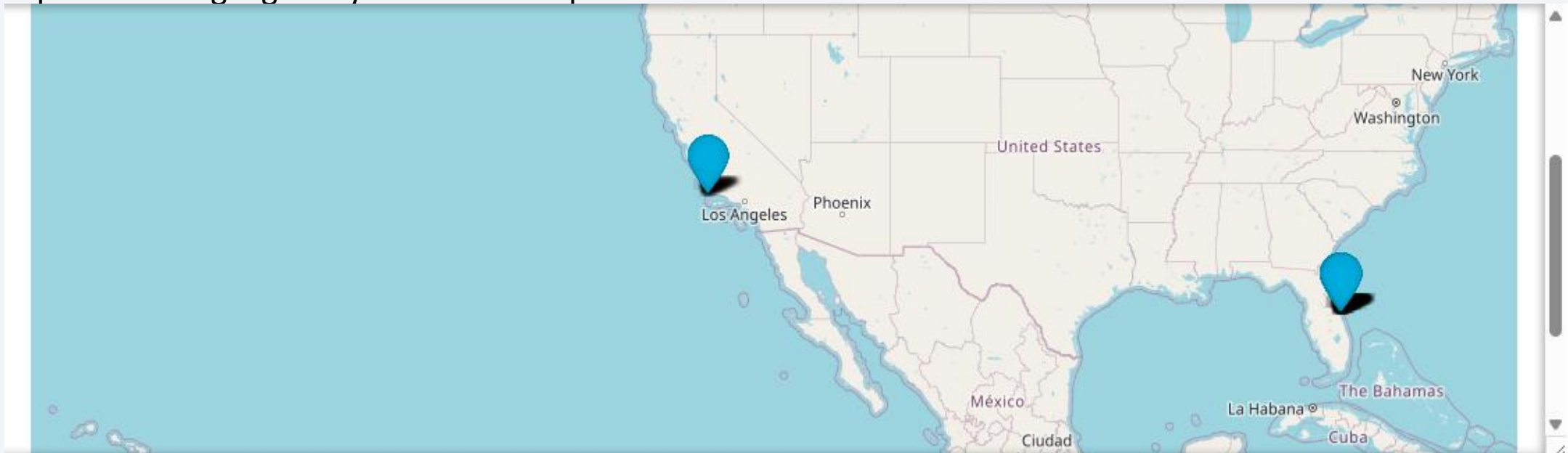
Section 3

# Launch Sites Proximities Analysis



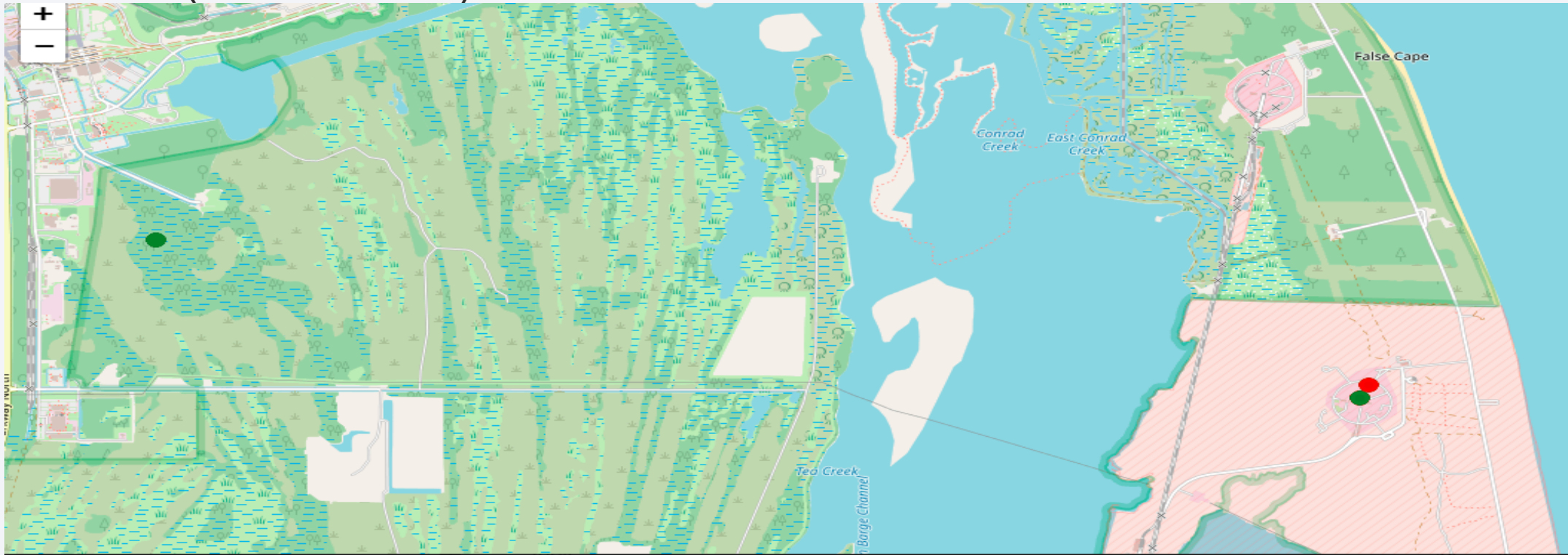
# Launch sites on a map

- Launch Sites on the Map
- Two major U.S. launch sites identified:
  - Los Angeles, CA – near Vandenberg Space Force Base, used for polar orbit launches.
  - Florida – near Cape Canaveral/Kennedy Space Center, used for equatorial and geostationary launches.
- Map markers highlight key locations for space missions.
- Visual emphasis on coastal launch advantages due to orbital mechanics.



# Success/failed launches for each site on the map

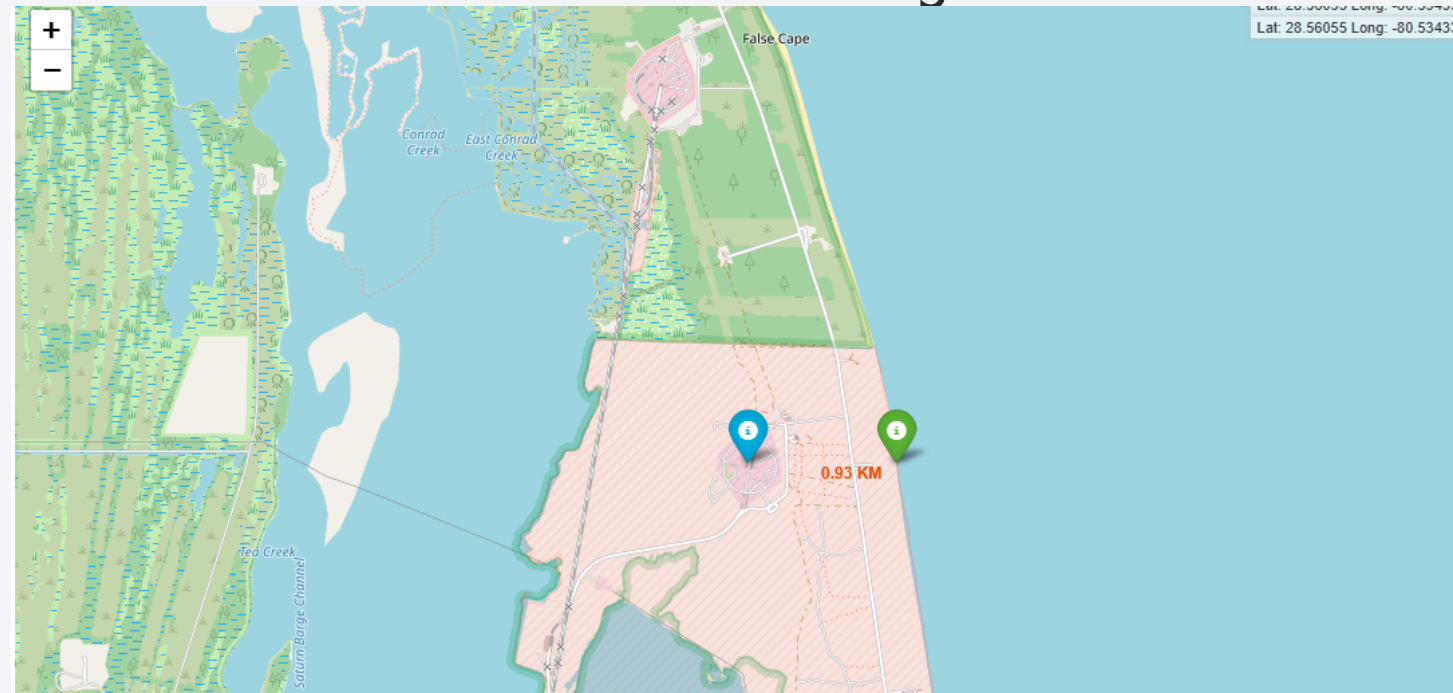
- Launch Outcomes by Site (Map View)
  - Green marker: Successful launch
  - Red marker: Failed launch
  - Multiple launches occurred from the same location (clustered markers)
- Site shown is likely Cape Canaveral, FL — a major U.S. launch center
- Map helps visualize success vs. failure rates by location



# Distances between a launch site to its proximities

---

- Launch site proximity measured using Folium.
- Distance to coastline: 0.93 KM.
- Map markers used to identify and label key infrastructure.
- Highlights strategic location for safe launches and efficient logistics.







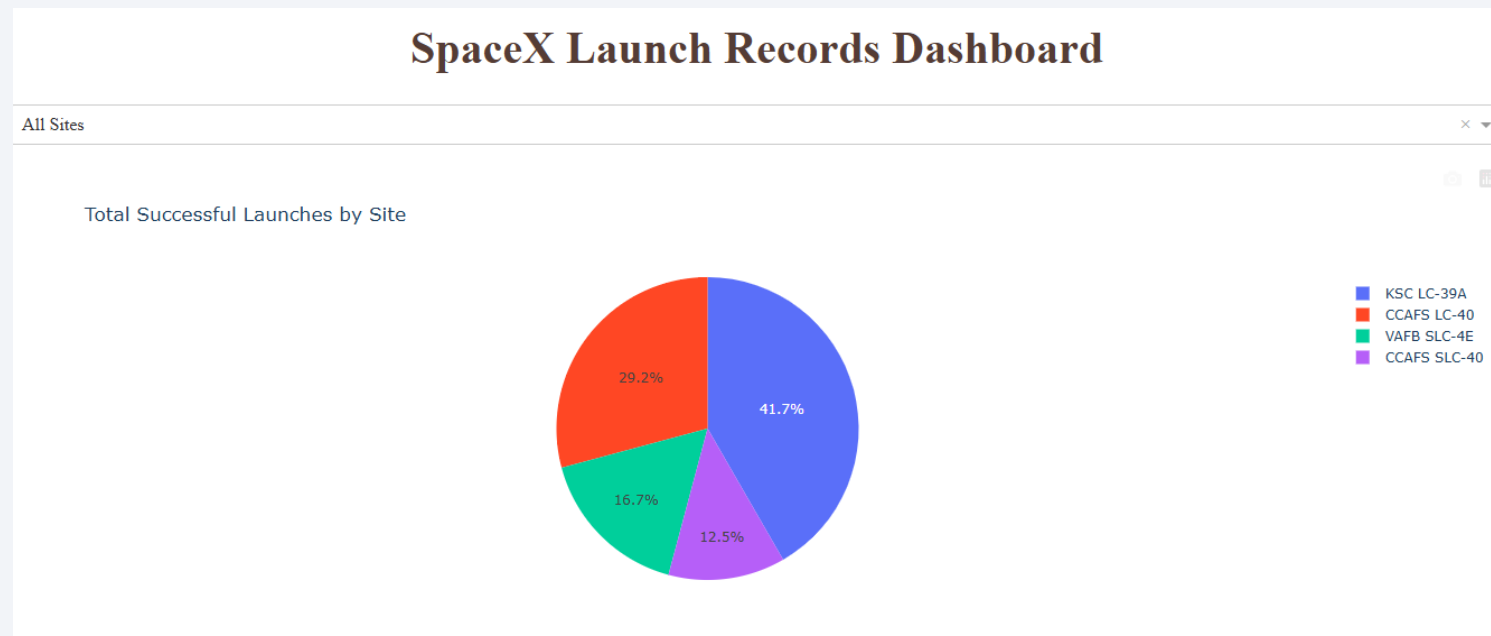
Section 4

# Build a Dashboard with Plotly Dash

# Total Successful Launches By Site

---

- Total Successful Launches by Site
  - KSC LC-39A: 41.7% – highest success rate
  - CCAFS LC-40: 29.2% – second most used
  - VAFB SLC-4E: 16.7%
  - CCAFS SLC-40: 12.5%
- Visual shows launch distribution across SpaceX sites



# Total Successful vs Failures for Site KSC LC-39A

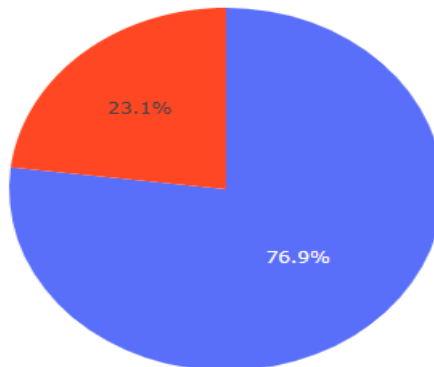
- KSC LC-39A – Success vs. Failure
  - Success rate: 76.9%
  - Failure rate: 23.1%
- KSC LC-39A is the site with the highest total number of successful launches.
- Visual shows strong reliability, making it SpaceX's most successful launch site.

## SpaceX Launch Records Dashboard

KSC LC-39A



Total Success vs. Failure for site KSC LC-39A



■ Success  
■ Failure

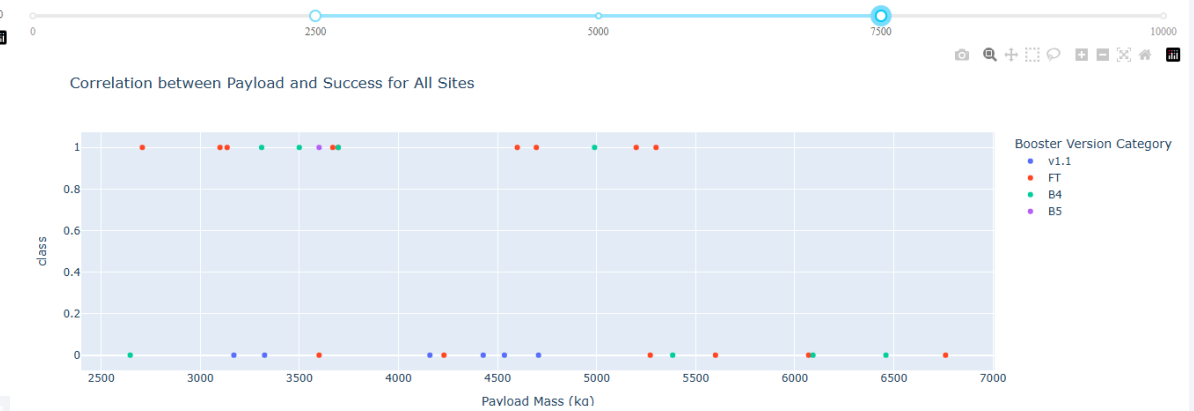


# Correlation between Payload and Success for All Sites

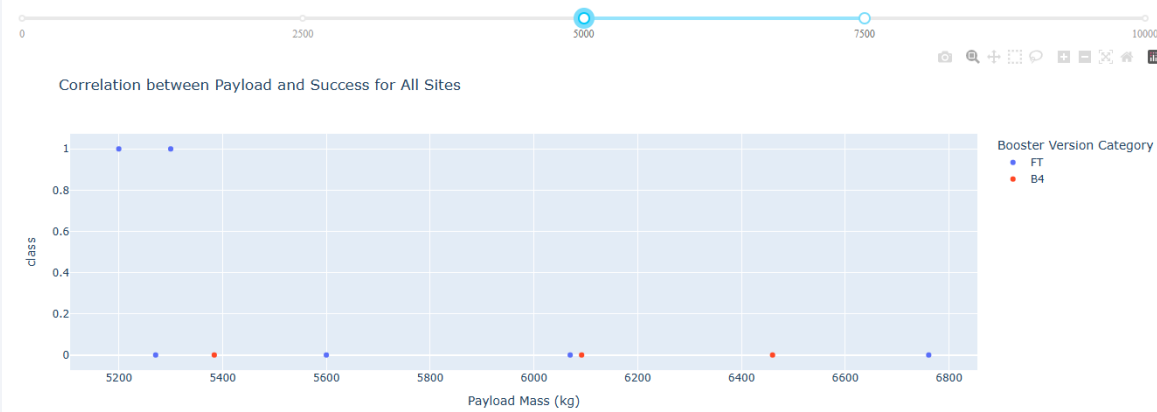
Payload range (Kg):



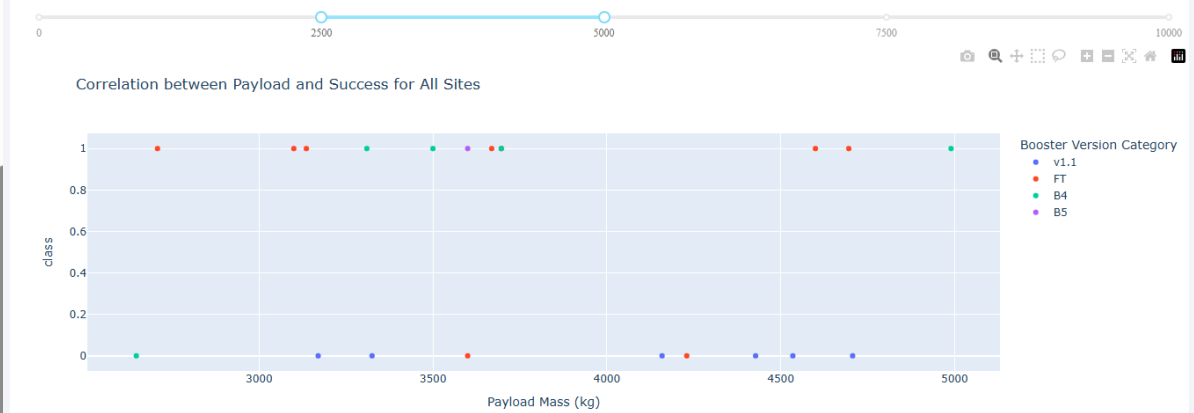
Payload range (Kg):



Payload range (Kg):







Payload range (Kg):



# Correlation between Payload and Success for All Sites

---

- **Key Findings:**

-  **Payloads between 3000–5000 kg show highest concentration of successes, especially for FT and B4 boosters.**
-  **Booster FT** consistently appears in successful launches across all payload ranges.
-  **Larger payloads (>7000 kg) show fewer launches**, with mixed success, and mostly with B5 or FT.
-  **FT and B4** booster versions have the **highest observed success rates**, especially in mid-weight ranges (3000–6000 kg).

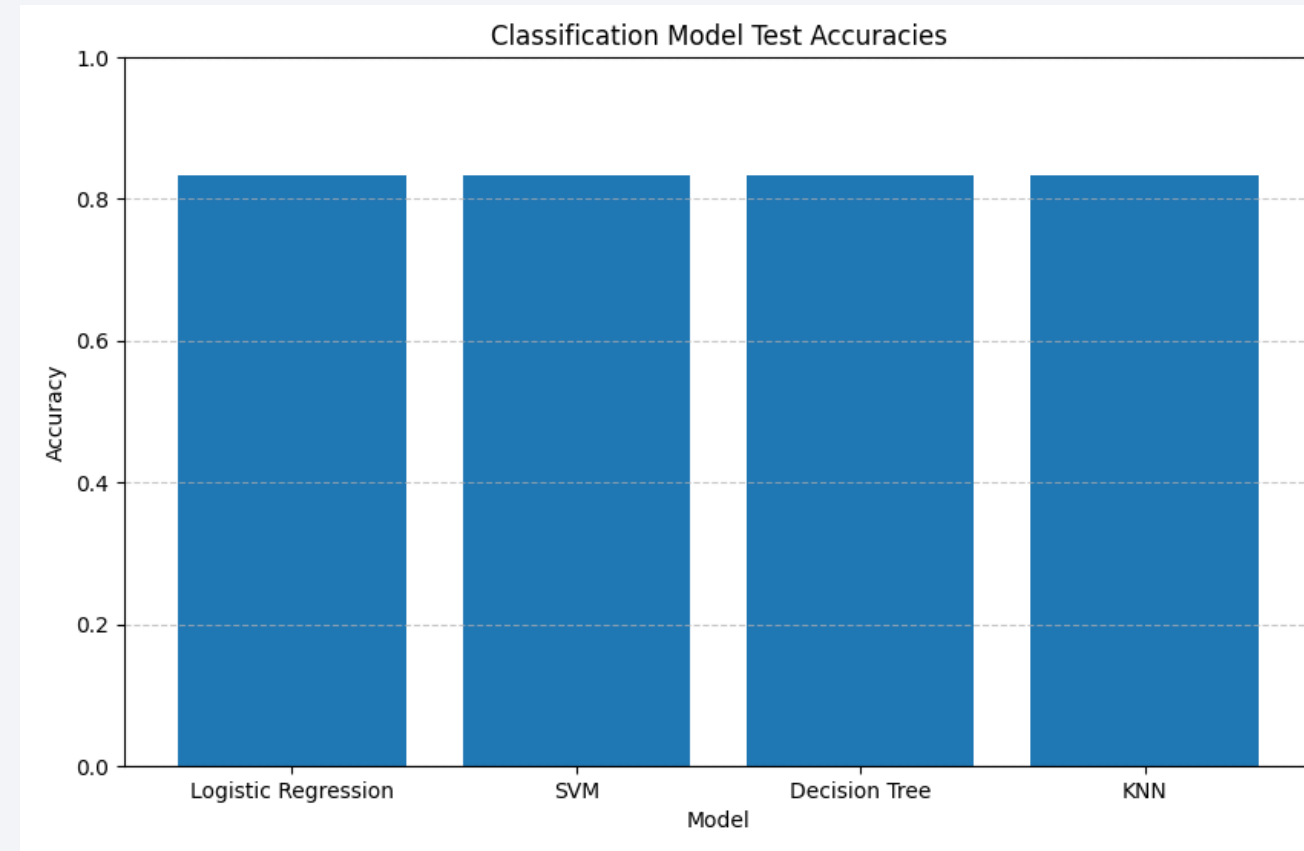
Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

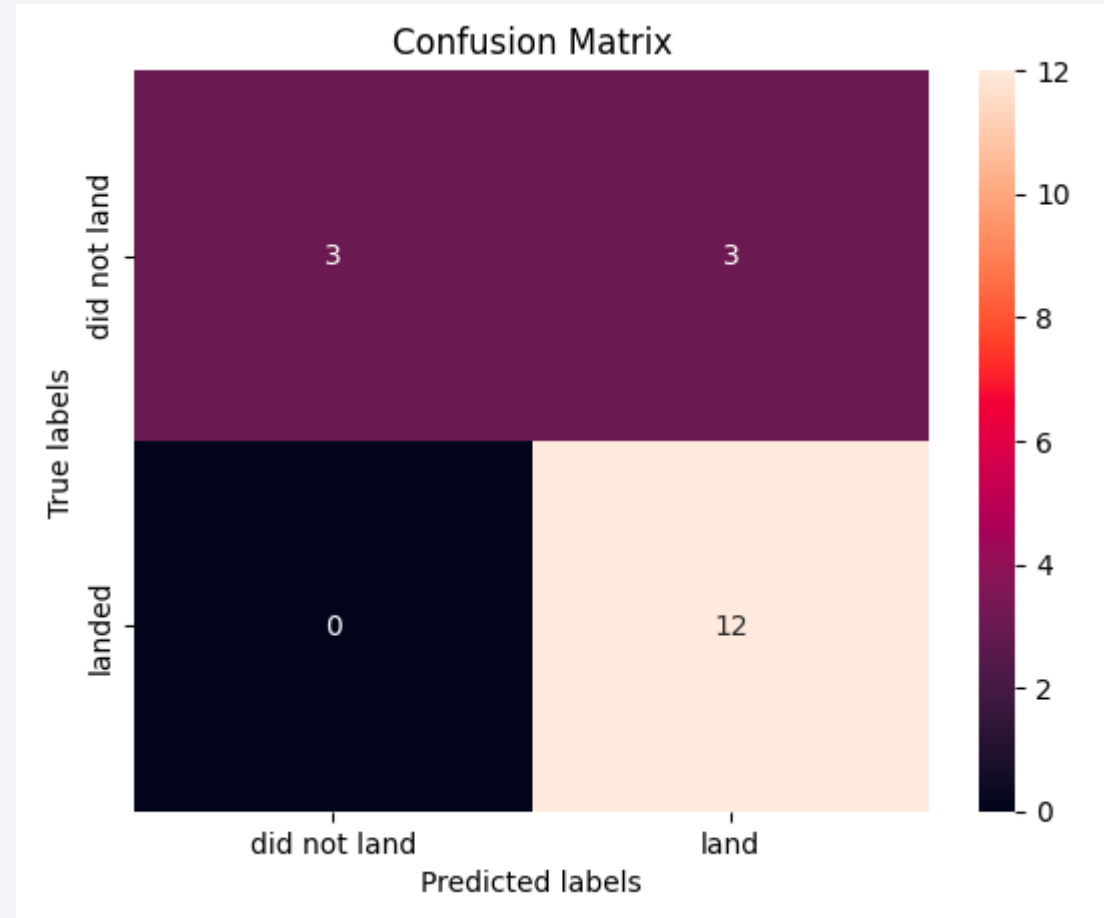
---

- Visualize the built model accuracy for all built classification models, in a bar chart
- Find which model has the highest classification accuracy
  - Logistic Regression, SVM, Decision Tree and KNN tie with the best performances, with 0,8333 each



# Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation
  - True Positives (Landed correctly): 12
  - True Negatives (Did not land correctly): 3
  - False Positives: 3
  - False Negatives: 0
  - ☒ Excellent at predicting successful landings
  - ⚠️ Some false alarms for landings that didn't happen



# Conclusions

---

- 🚀 **Model Performance:** Logistic Regression, SVM, Decision Tree and KNN all achieved high accuracy (83.33%), making them reliable for predicting launch outcomes.
- ✅ **Success Prediction Strength:** These models excel at identifying successful landings (no false negatives), a key metric for SpaceX mission evaluation.
- ⚠️ **Some False Positives:** A few cases predicted landings that didn't happen, indicating overconfidence in borderline cases.
- 📊 **Data-Driven Insights:** Standardized features and machine learning enable effective outcome classification from mission data, supporting operational decisions.
- 🔄 **Improvement Opportunities:** Enhancing feature engineering, adding more mission parameters, or addressing class imbalance could improve accuracy further.
- 🌍 **Business Impact:** Predictive modeling helps optimize costs and reduce risks in future launches, contributing to SpaceX's mission reliability and reusability goals.



# Appendix

---

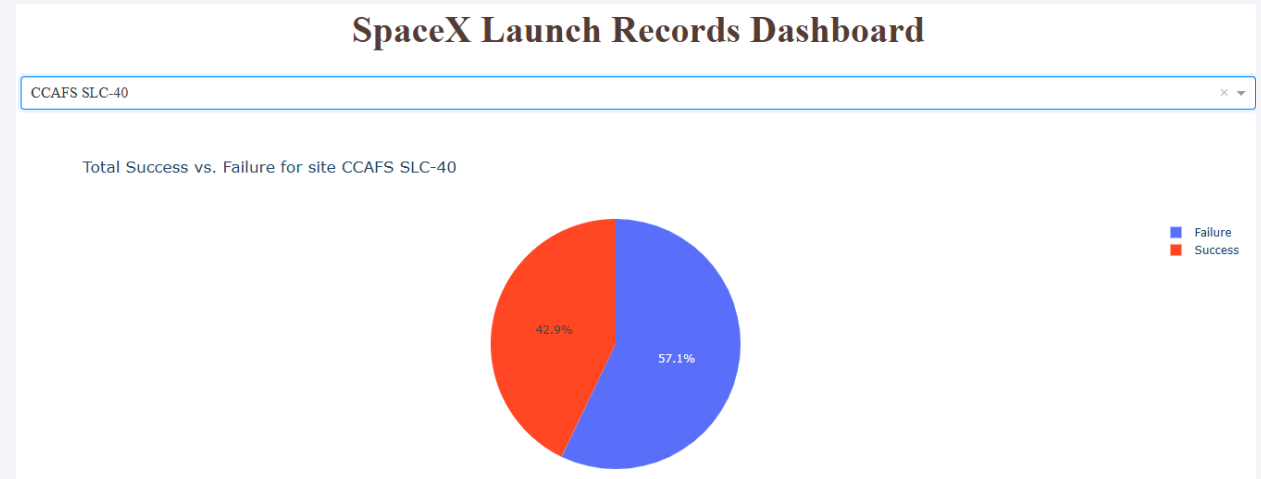
- Python Code for Accuracies Bar Chart

```
# Plotting the bar chart
plt.figure(figsize=(10, 6))
plt.bar(accuracies.keys(), accuracies.values())
plt.title('Classification Model Test Accuracies')
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.ylim(*args: 0, 1)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

- SQL query for Display the names of the unique launch sites in the space mission
  - select distinct Launch\_Site from SPACEXTABLE

# Appendix

- Chart of Success vs Failure for site CCAFS SLC-40



- Data set sample

FlightNumber	Date	BoosterVersion	PayloadMass
1	2010-06-04	Falcon 9	6104.9594117647
2	2012-05-22	Falcon 9	525.0
3	2013-03-01	Falcon 9	677.0
4	2013-09-29	Falcon 9	500.0
5	2013-12-03	Falcon 9	3170.0
6	2014-01-06	Falcon 9	3325.0

Thank you!

