IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Dario Beil
06.11.2025

Outline

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

Executive Summary

This project predicts whether a Falcon 9 first stage will land successfully
an important driver of cost savings through reusability.

We combined official SpaceX API data with a historical Wikipedia snapshot, cleaned and standardized the records, explored key patterns with visualizations and SQL, and delivered interactive analytics with Folium and Plotly Dash.

Finally, i trained and tuned multiple classification models and selected the best-performing approach on held-out data. The pipeline is fully reproducible with versioned datasets and saved model artifacts, enabling transparent peer review and future extension.

Introduction

Project Background and Context:
Under the leadership of Elon Musk, SpaceX has revolutionized spaceflights by making the first stage of its Falcon 9 rockets reusable. This allows SpaceX to reduce the cost per launch to approximately $65 million, which is $100 million less than its competitors. Each successful landing allows the same hardware to be reused, dramatically reducing launch costs and facilitating access to space. Predicting whether a first stage will land successfully is critical to understanding cost savings, planning future missions, and maintaining SpaceX's competitive advantage.

Problems to Answer:

- Which mission factors (e.g., payload mass, launch site, orbit, booster version) most influence first stage landing success?
- Can we build a model that predicts landing success before launch, using available mission data?
- How do landing success rates vary across different launch sites and orbit types?
- What interactive tools best help stakeholders explore and understand these patterns?
- What are the current limitations of the data, and how could future work improve prediction reliability?

Section 1

Methodology

Methodology

Executive Summary

This section provides a concise overview of the analytical pipeline implemented in this project. The workflow began with dual-source data collection (SpaceX API and Wikipedia), followed by rigorous data wrangling to create clean, analysis-ready datasets.

Exploratory data analysis was performed using both visualization and SQL to uncover key patterns and validate findings. Interactive analytics tools (Folium, Plotly Dash) were developed to facilitate stakeholder exploration. Finally, several classification models were trained, tuned, and evaluated, with the best model selected based on test performance. All steps were documented and versioned to ensure full reproducibility and transparency.

Methodology

Data Collection

We used a dual-source approach to maximize data completeness and reliability. Launch records were programmatically retrieved from the SpaceX REST API, focusing on launches, rockets, and launchpads endpoints. The resulting nested JSON data was normalized into flat tables using the pandas library.

To supplement and cross-validate, we scraped a static Wikipedia snapshot of Falcon 9 launches using the requests library for downloading and BeautifulSoup for parsing, extracting tabular data for historical context. All data pulls were timestamped, and snapshot URLs were recorded to ensure reproducibility. Quality checks included HTTP status validation and schema consistency before exporting to CSV.

The SpaceX API provided structured, up-to-date information on each launch, including mission details, payload, and outcome. The Wikipedia snapshot offered a historical reference and additional context, especially for older missions. Both sources were parsed and merged with careful alignment of fields and types using pandas. This dual collection strategy reduced the risk of missing or inconsistent records and enabled robust downstream analysis.

Methodology

Data Wrangling

Data wrangling and preprocessing were carried out using pandas and numpy. Raw data was profiled for missing values and inconsistent types. Outcome strings (e.g., “True ASDS”, “False Ocean”, “None None”) were mapped to a binary landing success label (Class: 1 for success, 0 for failure).

Irrelevant or high-cardinality text fields were removed, categorical variables were encoded, and numerical columns were standardized as needed. The cleaned dataset (dataset_part_2.csv) and a feature matrix for modeling (dataset_part_3.csv) were exported as reproducible checkpoints.

Data processing included quantifying missingness, standardizing data types, and engineering the target variable. Categorical features were encoded, and numerical features were type-cast and scaled where appropriate using pandas and numpy. The resulting datasets were versioned and documented to support reproducibility and peer review.

Methodology

Exploratory Data Analysis (EDA)

EDA was conducted using pandas for data handling, matplotlib and seaborn for visualization, and SQL queries (via pandas and SQLite) for aggregation and validation. Visualizations included bar plots of launch counts by site, charts of orbit distributions, scatter plots of payload mass versus landing success, and correlation matrices to reveal dependencies among features.
These tools highlighted which variables, such as launch site, orbit type, and payload, might influence landing outcomes. SQL queries were used to aggregate statistics like success rates by site and orbit and to calculate average payloads, providing a cross-check for the visual results. This combination of methods ensured that the observed trends were robust and reliable, forming a strong basis for selecting features and building predictive models.

Spatial and interactive analytics were implemented using the Folium library for mapping and Plotly Dash for dashboard development. A Folium map displayed launch site locations and spatial patterns, while a Plotly Dash dashboard enabled dynamic filtering by launch site and payload mass. The dashboard included a pie chart for success distribution and a scatter plot for payload versus landing outcome, colored by booster version. These tools allowed stakeholders to explore scenarios and patterns interactively.

Methodology

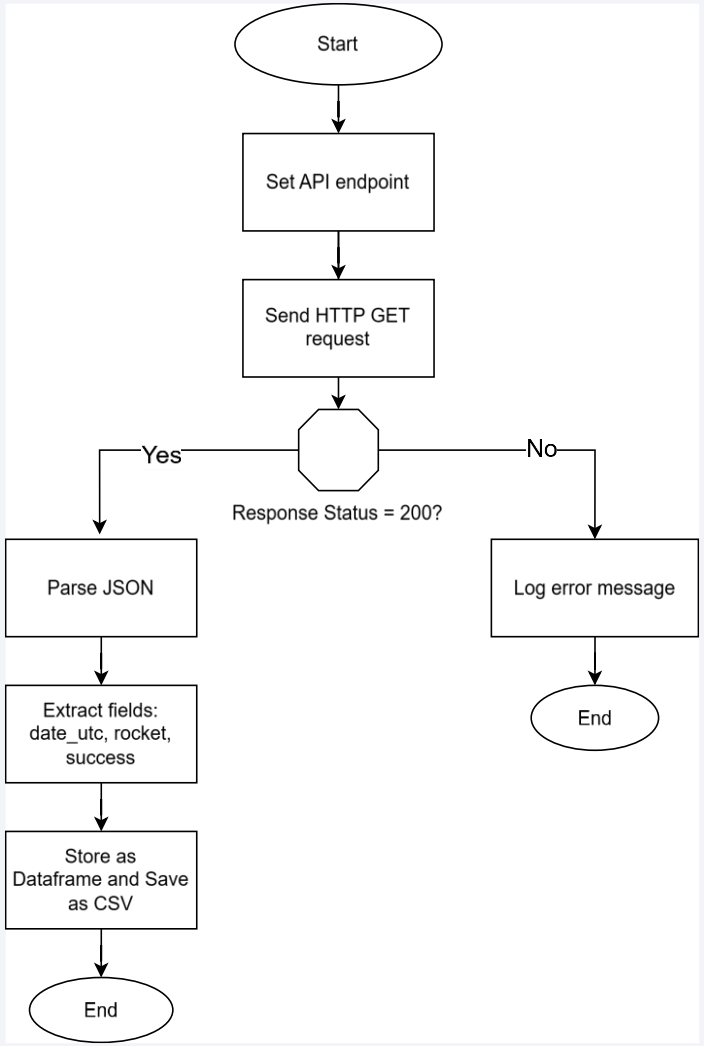
Predictive Analysis using Classification Models

Model development, training, and evaluation were performed using scikit-learn. The cleaned and engineered data was split into training and test sets (80/20). Features were standardized, and four classification models—Logistic Regression, Support Vector Machine, Decision Tree, and k-Nearest Neighbors—were trained. Hyperparameters were tuned using GridSearchCV with 10-fold cross-validation. Model performance was evaluated using accuracy and confusion matrices, with the best-performing model saved for future use.
How to Build, Tune, and Evaluate Classification Models

Model development followed best practices:
- Data was split into train and test sets to prevent information leakage.
- StandardScaler from scikit-learn was fit on the training data and applied to both sets.
- GridSearchCV was used to systematically tune hyperparameters for each model type.
- Evaluation focused on accuracy and confusion matrix interpretation, with special attention to minimizing false positives.
- The best model and its parameters were saved (models.pkl) for reproducibility and deployment.

Data Collection – SpaceX API

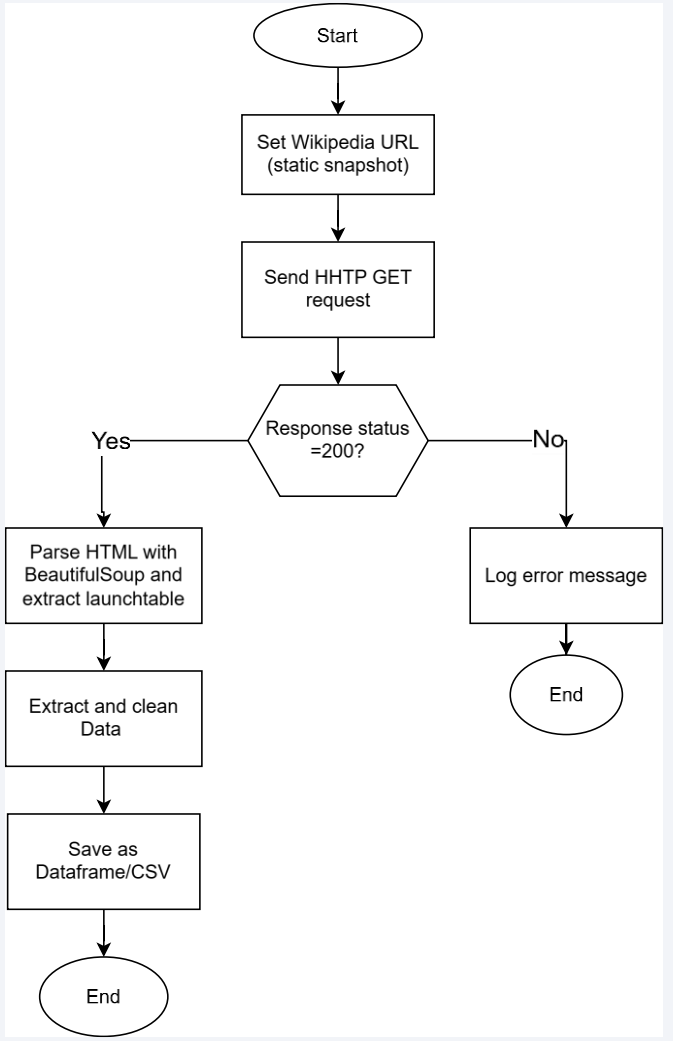
- Retrieve all SpaceX launches using the REST API
- Extract rocket, payload, launchpad, and core information via their IDs
- Combine all data into a structured Data Frame (Pandas)
- Clean and filter the data (e.g., select only Falcon 9 launches, handle missing values)



Github Repository link

Data Collection - Scraping

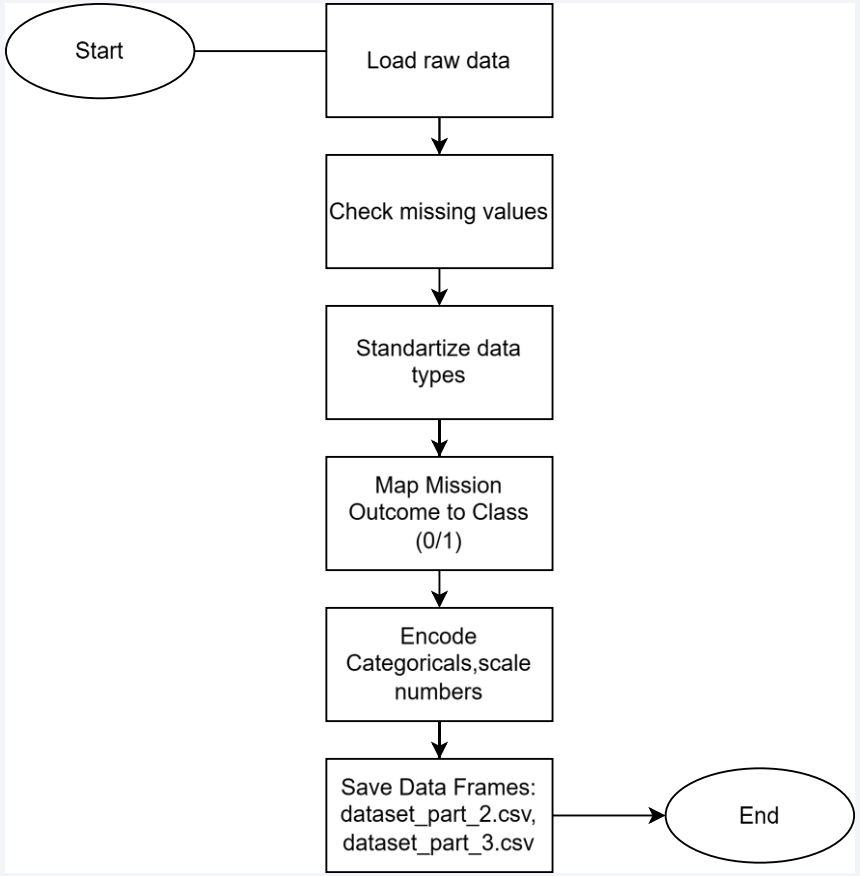
- Request the Falcon 9 and Falcon Heavy launches Wikipedia page (static snapshot)
- Parse the HTML content with BeautifulSoup
- Locate and extract the launch records table
- Extract column names and launch data from the table rows
- Clean and structure the data into a Pandas DataFrame
- Export the DataFrame to CSV for further analysis



Github Repository link

Data Wrangling

- Checked and quantified missing values
- Standardized data types (numerical/categorical)
- Engineered binary landing outcome label (Class: 0/1) from outcome strings
- Encoded categorical variables and scaled numerical features
- Exported cleaned datasets for modeling (dataset_part_2.csv, dataset_part_3.csv)



Github Repository link

EDA with Data Visualization

Charts Plotted and Rationale

- Bar plots of launch counts by site to compare activity and identify the most frequently used launch locations.
- Bar charts of orbit distributions to understand the variety and frequency of mission orbits.
- Scatter plots of payload mass versus landing success to explore the relationship between payload size and landing outcome.
- Correlation matrices to identify dependencies and potential predictive relationships among features.

These visualizations were chosen to highlight key patterns in the data, reveal which variables might influence landing success, and guide feature selection for modeling.

Github Repository link

EDA with SQL

Summary of SQL Queries Performed

- Counted the number of launches per launch site to compare site activity.
- Calculated average payload mass by orbit and by launch site to identify trends and outliers.
- Computed landing success rates (average of Class) by site and by orbit to assess performance differences.
- Selected top missions by payload mass for further inspection.
- Filtered and joined tables to analyze relationships between booster version, site, and outcome.

These queries provided quantitative validation for the visual findings and helped ensure the robustness of the analysis.

Github Repository link

Build an Interactive Map with Folium

Maps Objects created

- **Markers:** Added for each launch site to visualize geographic locations.
  Rationale: Enables users to quickly identify and locate all SpaceX launch sites on the map.
- **Circle Markers:** Used to represent launch sites, with size or color indicating site activity or success rate.
  Rationale: Visually encodes important metrics (e.g., frequency and success) for immediate comparison between sites.
- **Popups:** Included at each marker for interactive exploration of site details.
  Rationale: Provides on-demand access to site-specific information without cluttering the map.
- **Polylines:** Added to show distances or trajectories between launch sites and landing zones.
  Rationale: Illustrates spatial relationships and supports analysis of logistical or safety considerations.

Github Repository link

Build a Dashboard with Plotly Dash

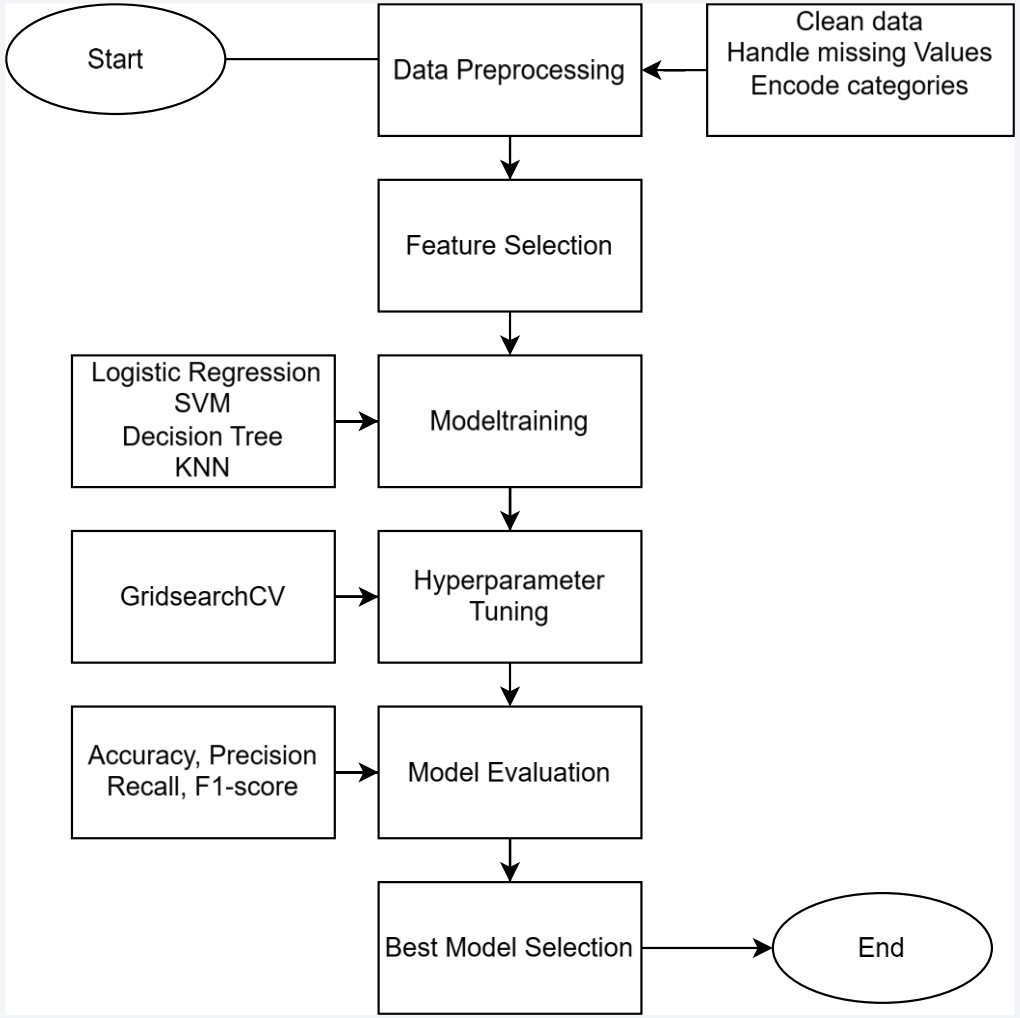
Plots/Graphs and Interactions Added:

- **Pie Chart:**
  Displays the proportion of successful vs. failed launches for each launch site.
  Rationale: Provides a quick overview of the success rates at each site, helping to identify high-performing locations.
- **Scatter Plot:**
  Shows the relationship between payload mass and launch outcome, color-coded by success or failure.
  Rationale: Visualizes potential correlations between payload and outcome, supporting hypothesis testing about influential factors.
- **Dropdown Menus:**
  Allow users to filter data by launch site and payload range.
  Rationale: Enables users to focus on specific sites or payload intervals, facilitating targeted analysis.
- **Interactive Callbacks:**
  Dynamically update the plots based on user selections.
  Rationale: Allow for flexible, exploratory analysis and make the dashboard user-friendly.

Github Repository link

Predictive Analysis (Classification)

- Data preprocessing: Cleaned and encoded features, handled missing values.
- Feature selection: Identified relevant predictors for landing success.
- Model training: Tested multiple classifiers (Logistic Regression, SVM, Decision Tree, KNN).
- Hyperparameter tuning: Used GridSearchCV for optimal parameters.
- Model evaluation: Compared models using accuracy, precision, recall, F1-score.
- Best model selection: Chose the classifier with the highest validation performance.



Github Repository link

Results -Exploratory data Analysis

Flight experience effect:

- Flight Number vs. Payload Mass scatter shows increasing landing success with higher flight numbers.
- Early launches at CCAFS SLC-40 exhibit more failures; later flights improve markedly, indicating process and quality maturation.

Payload and site patterns:

- VAFB-SLC shows no heavy-payload (>10,000 kg) launches in the observed period.
- With heavy payloads, success is more common for Polar, LEO, and ISS orbits; GTO shows mixed Outcomes.

Orbit-type success rates:

- Mean success rate by orbit (bar chart of grouped means) places LEO/ISS near the top; GTO is comparatively less consistent.

Yearly trend:

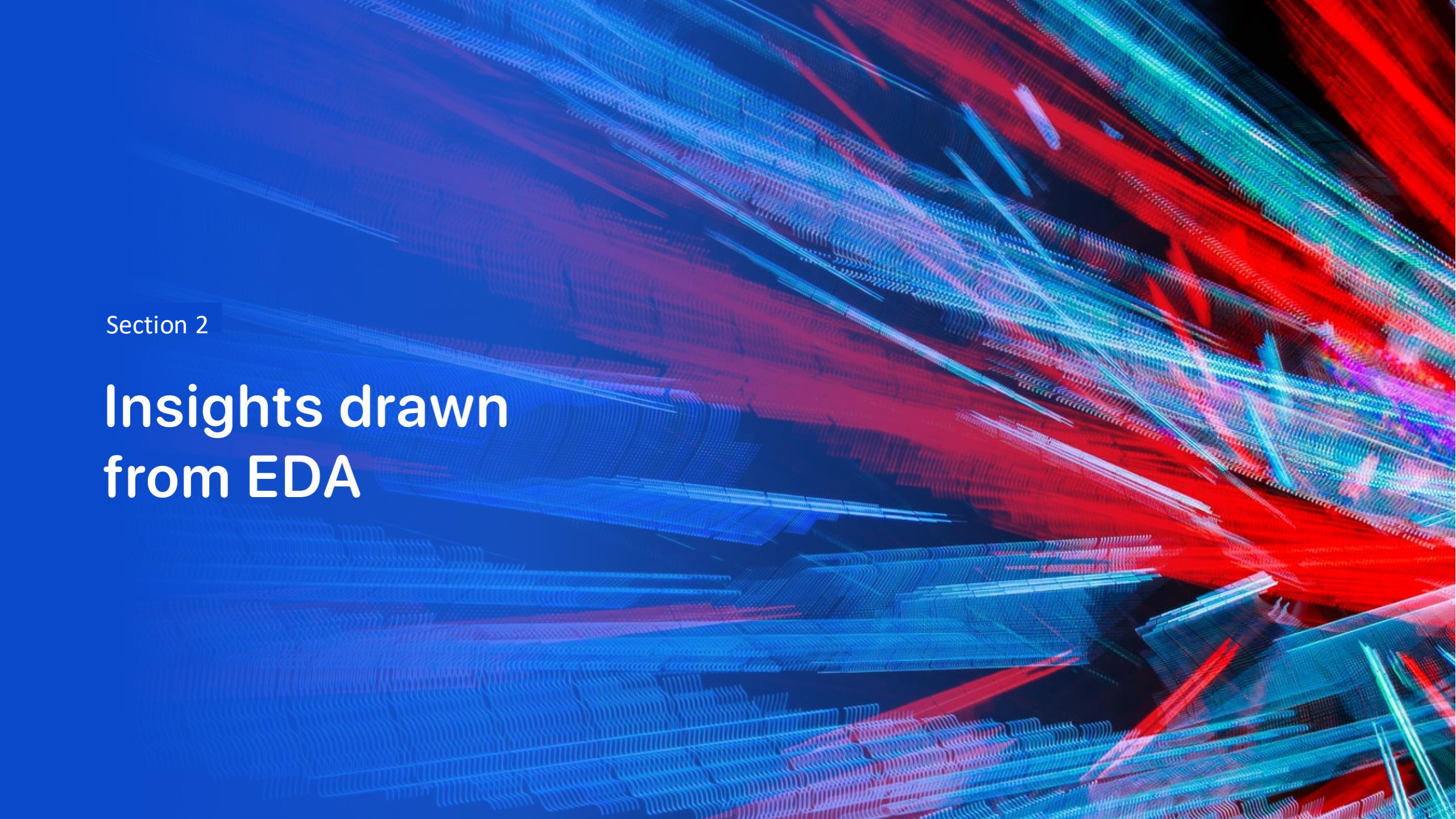
- Average annual success rate increases continuously from 2013 to 2020 (line plot), consistent with operational learning and hardware iteration.

Feature engineering readiness:

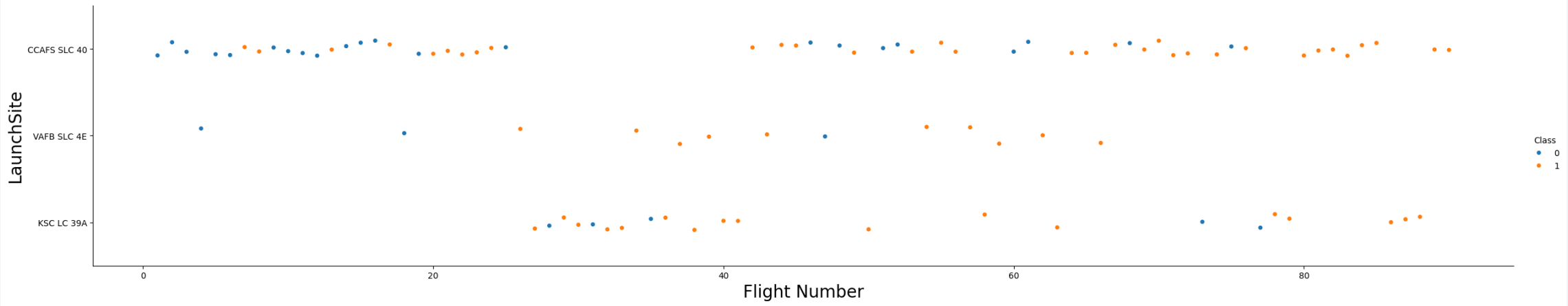
- Dataset cleaned and one-hot encoded; categorical features (Orbit, LaunchSite, LandingPad, Serial) expanded to dummies for modeling.

Results Predictive Analysis Results

- Four classification models were trained and tuned: Logistic Regression, Support vector machine (SVM), Decision Tree, and K-nearest neighbours (KNN).
- All models achieved a test accuracy of **0.8333** on the hold-out set.
- Best Grid search cross-validation (CV) scores:
  - Logistic Regression: 0.8214 (best params: C=1, l2, lbfgs)
  - SVM: 0.8482 (best params: C=1.0, gamma≈0.0316, kernel=sigmoid)
  - Decision Tree: 0.8607 (best params: gini, max_depth=2, splitter=random, etc.)
  - KNN: 0.8339 (best params: n_neighbors=6, p=1)
- The best model by test accuracy was **KNN**.
- Confusion matrix (Logistic Regression example): 12 true positives, 3 false positives, reflecting a small and imbalanced test set (12 successes, 6 failures).
- All tuned models generalize similarly, suggesting the classification task is well captured by the current features and data.
- Recommendation: Increase data size and use stratified cross-validation for more robust model comparison.

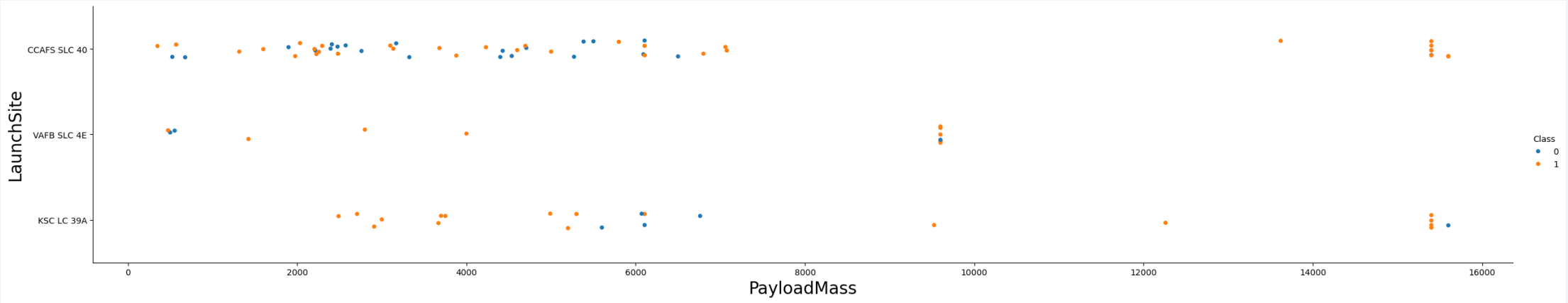

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- This scatter plot visualizes the relationship between the sequential flight number and the launch site. Each point represents a single launch, colored by its outcome (success= yellow, failure= blue).
- The plot reveals that early launches at CCAFS SLC-40 were more prone to failure, while later launches at the same site show improved success rates, indicating operational learning.
- Other sites (e.g., VAFB-SLC, KSC LC-39A) show consistently higher success rates across flight numbers, suggesting site-specific factors or improved processes.

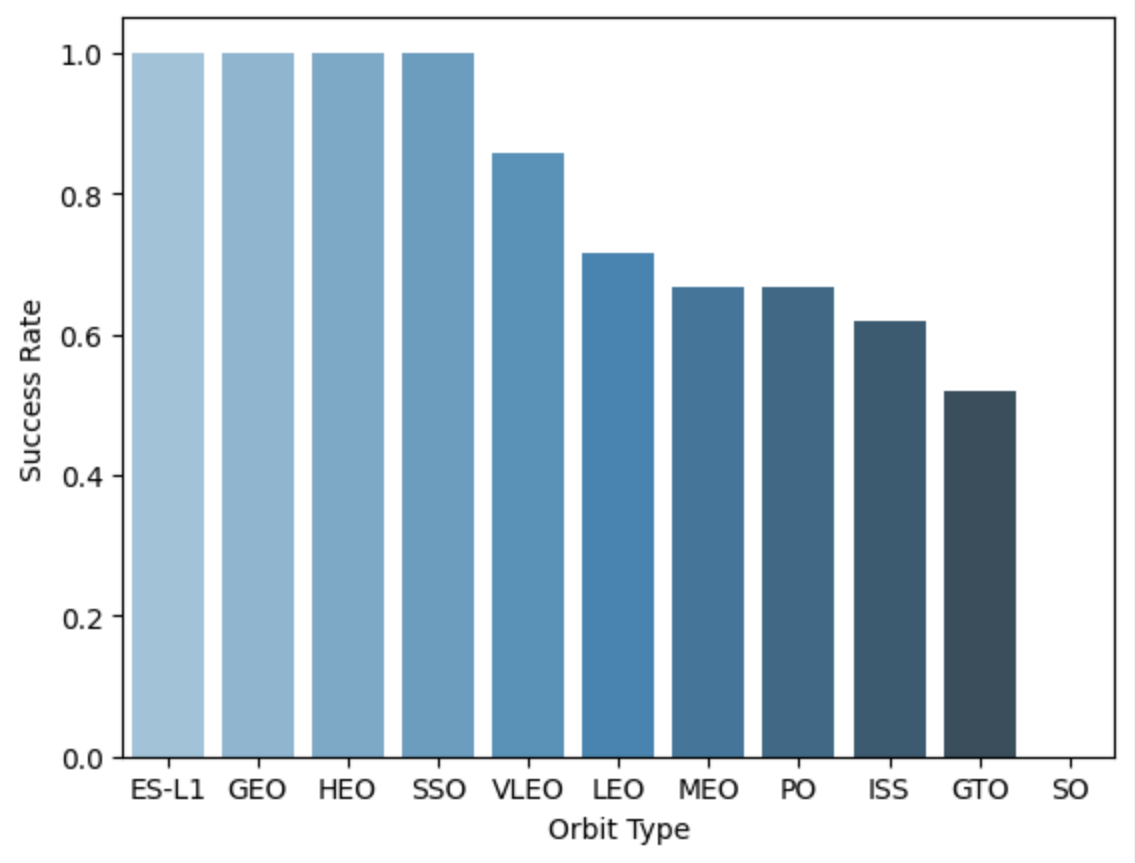
Payload vs. Launch Site



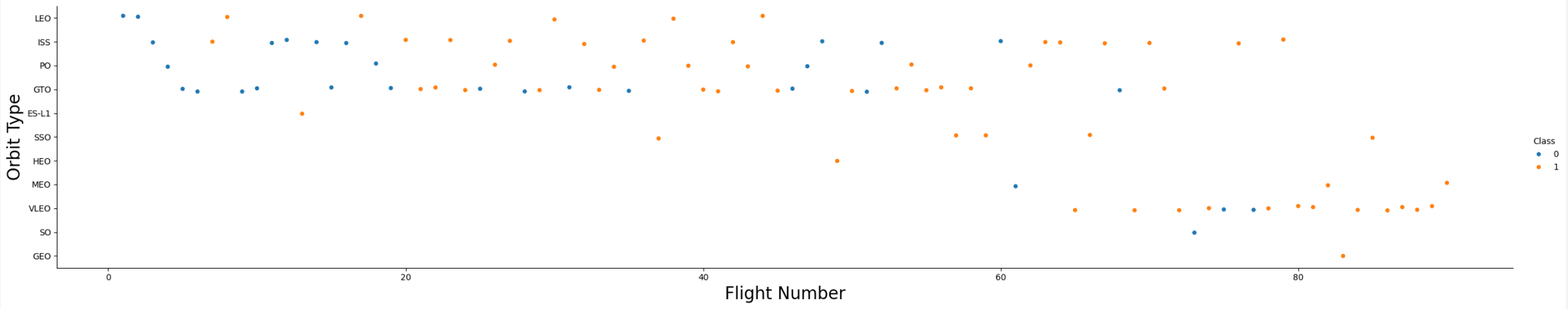
- Visualizes the relationship between payload mass (PayloadMass) and launch site (LaunchSite).
- Colored markers indicate whether the landing was successful (Class=1) or not (Class=0).
- No rockets with heavy payloads (>10,000 kg) were launched from the VAFB-SLC site.
- CCAFS SLC 40 and KSC LC 39A cover a broader range of payloads, including heavy payloads.
- Successful landings occur at all launch sites, but heavy payloads are mainly launched from the larger sites.
- The plot suggests that both launch site and payload mass may influence the probability of a successful landing.

Success Rate vs. Orbit Type

- The bar chart displays the average landing success rate (Class) for each orbit type.
- Success rates are calculated by grouping the data by the "Orbit" column and taking the mean of the "Class" column.
- Orbits with the highest success rates are easily identified at the top of the chart.
- Some orbit types (e.g., LEO, ISS) show higher success rates, indicating more reliable landings for these missions.
- Other orbits (e.g., GTO) have lower success rates, suggesting greater landing challenges.
- The visualization helps to compare how landing success varies depending on the target orbit.

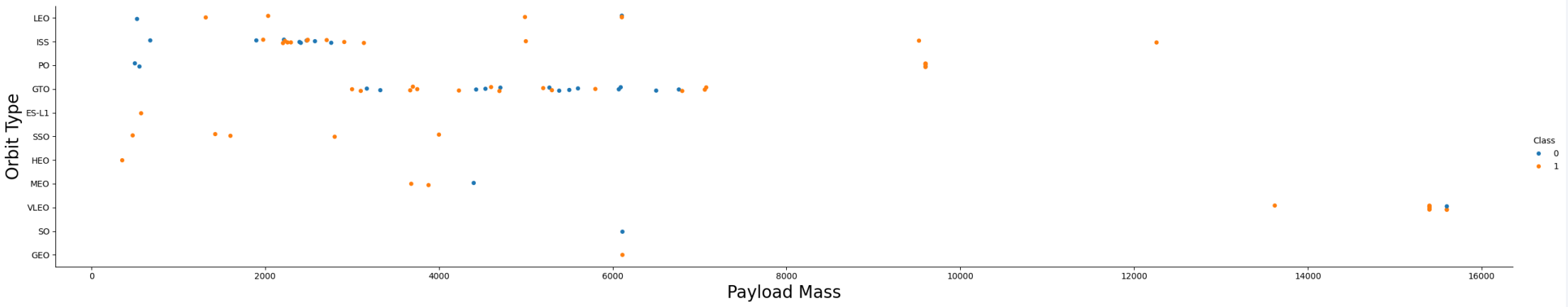


Flight Number vs. Orbit Type



- The scatter plot shows flight number vs. orbit type, colored by landing success (Class).
- For LEO, higher flight numbers correlate with more successful landings - evidence of improvement over time.
- For GTO, no clear trend between flight number and landing success.
- The plot highlights that learning effects and technical advances benefited some orbits (like LEO) more than others.

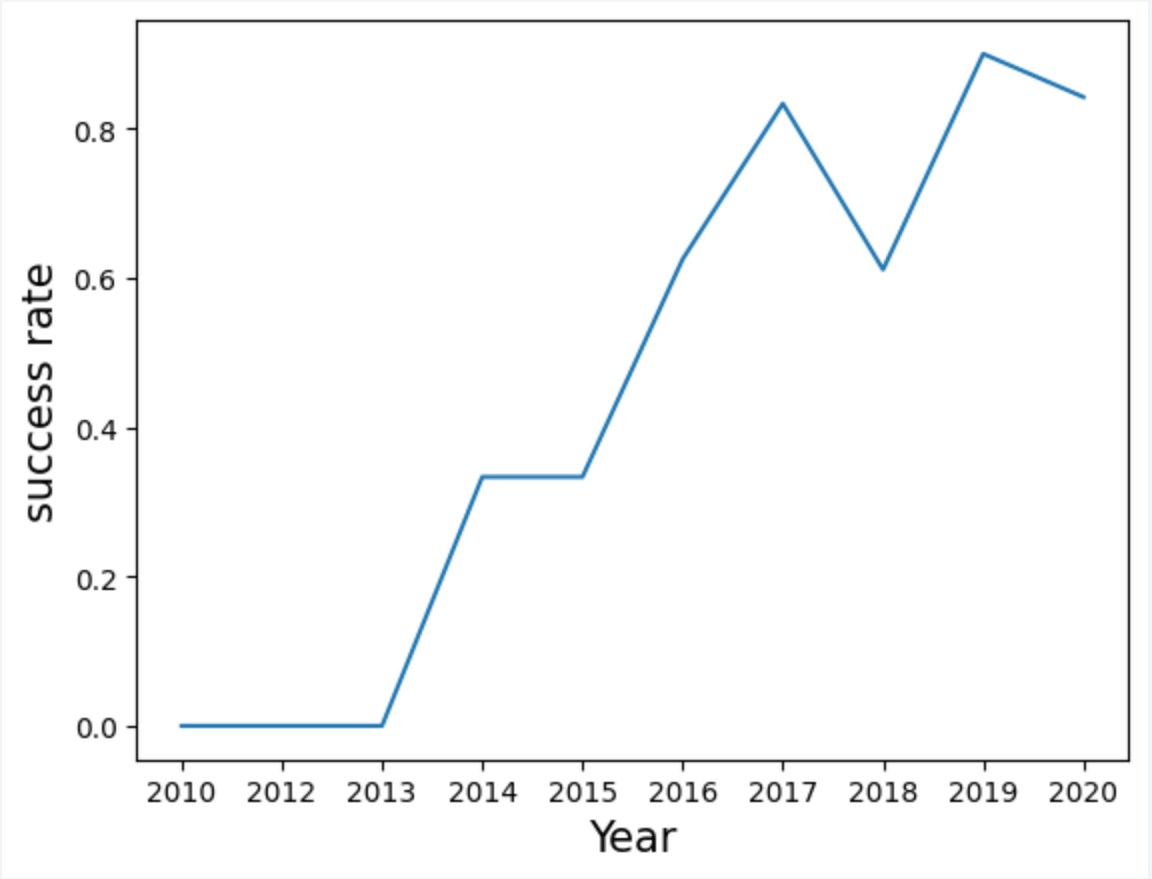
Payload vs. Orbit Type



- The scatter plot shows payload mass (PayloadMass) versus orbit type (Orbit), colored by landing success (Class).
- For Polar, LEO, and ISS orbits, heavy payloads are more often associated with successful landings.
- For GTO, both successful and unsuccessful landings occur across the payload range—no clear pattern.
- The plot highlights that the relationship between payload mass and landing success varies by orbit type.

Launch Success Yearly Trend

- The line chart shows the average landing success rate per year.
- Success rate is calculated as the mean of the "Class" column for each year.
- The plot reveals a clear upward trend: launch success rates have steadily increased from 2013 to 2020.
- This indicates continuous improvement in SpaceX’s landing technology and operational experience over time.



Unique Launch Site Names

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

These are the four distinct launch sites used in the SpaceX missions dataset. Each name represents a different physical location from which launches have occurred.

Launch Site Names Begin with 'CCA'

Date	Booster Version	Launch Site	Payload	Payload Mass (kg)	Orbit	Customer	Mission Outcome	Landing Outcome
2010-06-04	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, …	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

These are the first five launches from sites starting with "CCA" (Cape Canaveral), showing details such as date, booster version, payload, and outcome.

Total Payload Mass

total_payload by NASA

45.596 KG

This query sums the payload mass for all records where the customer is NASA (CRS), giving the total payload carried by NASA missions. The result shows the total payload in kilograms.

Average Payload Mass by F9 v1.1

AVG(PAYLOAD_MASS__KG_)

2928.4

This value represents the average weight of all payloads launched using the F9 v1.1 booster version.

First Successful Ground Landing Date

The date of the first successful landing outcome on a ground pad is the result of the query is 2015-12-21.

This date marks the first time SpaceX successfully landed a booster on a ground pad, representing a major milestone in reusable rocket technology.

Successful Drone Ship Landing with Payload between 4000 and 6000

Booster_Version

F9 FT B1022

F9 FT B1026

F9 FT B1021.2

F9 FT B1031.2

The boosters that successfully landed on a drone ship with a payload mass between 4000 and 6000 kg are the result of the query (e.g., "F9 FT", "F9 B4").

These booster versions achieved a successful drone ship landing while carrying a payload in the specified weight range, demonstrating both heavy lift capability and precise landing technology.

Total Number of Successful and Failure Mission Outcomes

success_cnt
61

failure_cnt
10

SpaceX achieved 61 successful landings and experienced 10 failures, reflecting the overall reliability and progress of their launch and landing technology.

Boosters Carried Maximum Payload

These booster versions achieved the highest recorded payload mass in SpaceX missions, demonstrating their top lifting capability.

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

2015 Launch Records

In 2015, there were two failed drone ship landings:

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

Both failures occurred with F9 v1.1 boosters at the Cape Canaveral LC-40 launch site, showing the challenges of early drone ship landings that year.

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

Most missions in this period had no landing attempt, while successful and failed drone ship landings were equally common. Ground pad successes and ocean landings were less frequent.

This ranking shows the diversity and progress of SpaceX landing outcomes during these years.

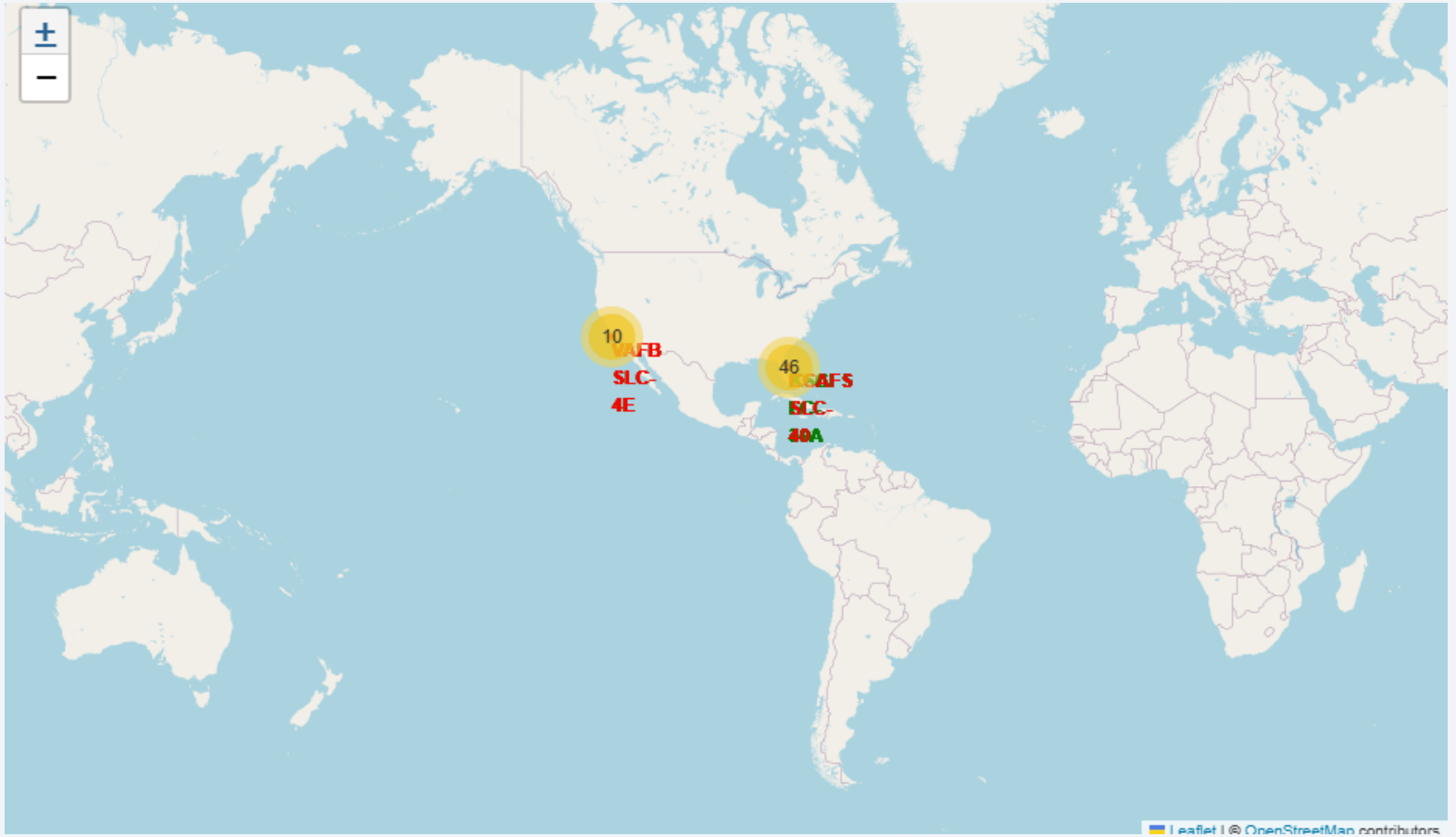
Landing_Outcome	outcome_cnt
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

Results – Folium Map Demo

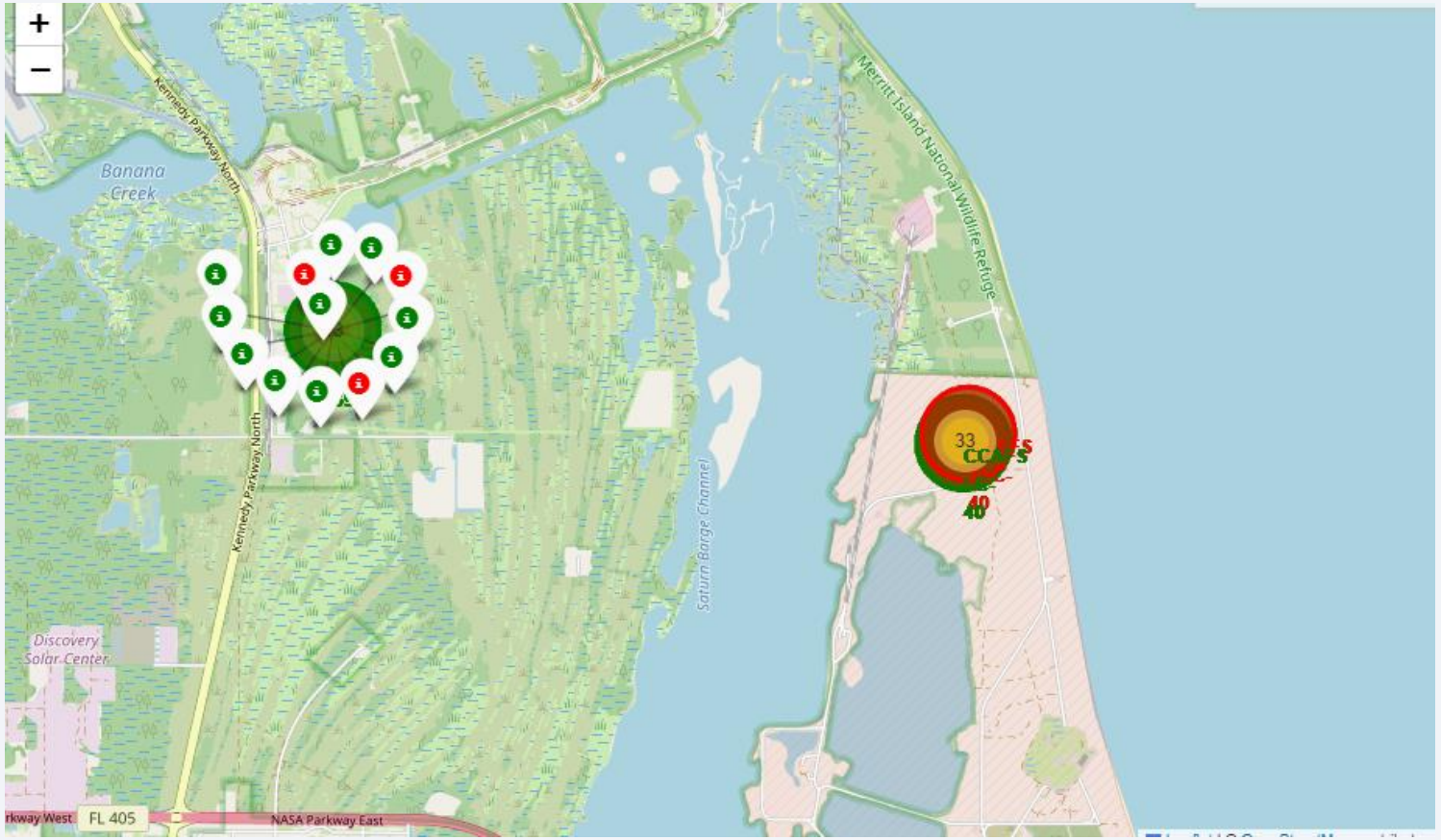
Overview of all SpaceX launch sites, their locations and the total count of launches each site.



Results – Folium Map Demo

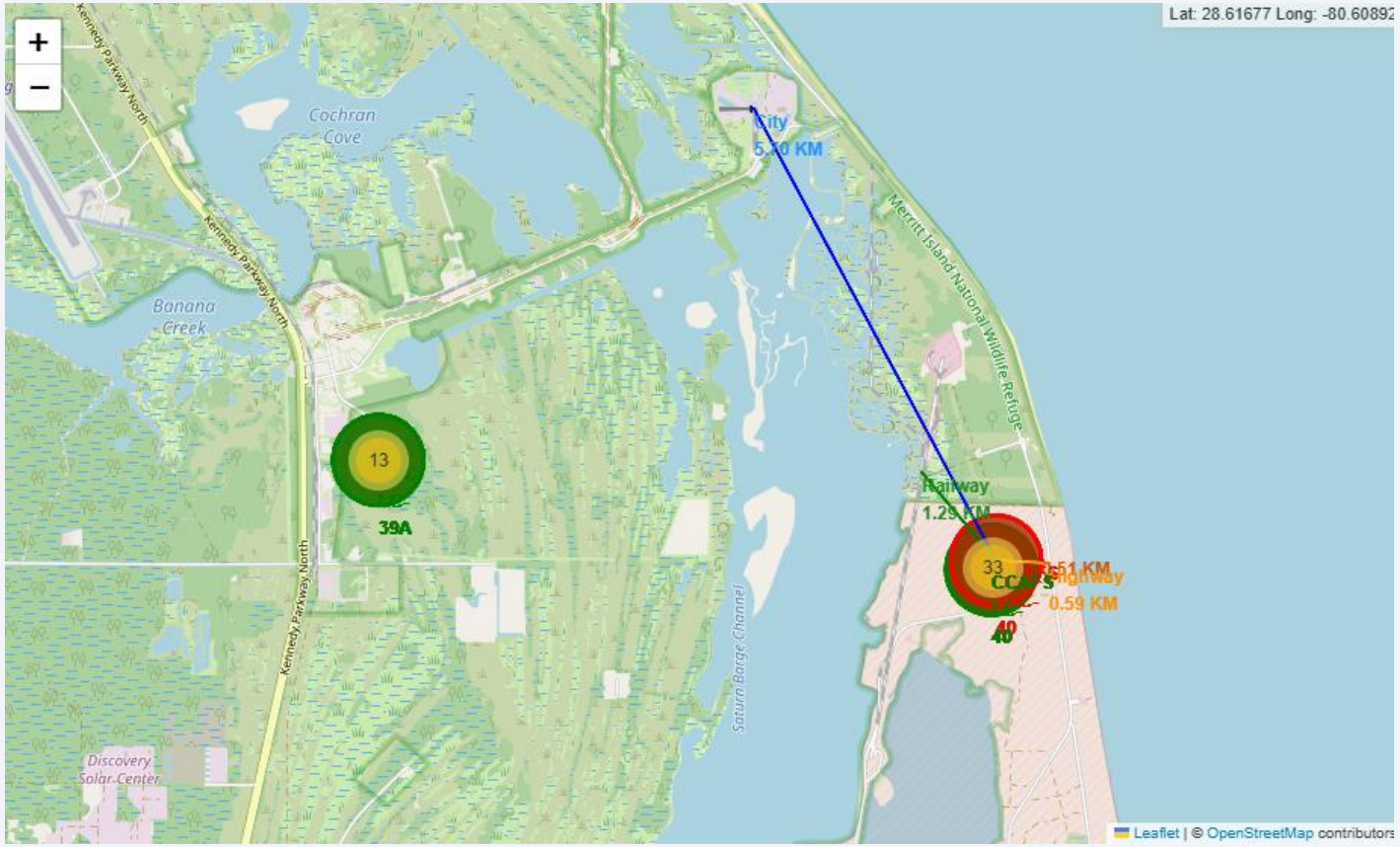
Detailed view of launch outcomes at a specific site.

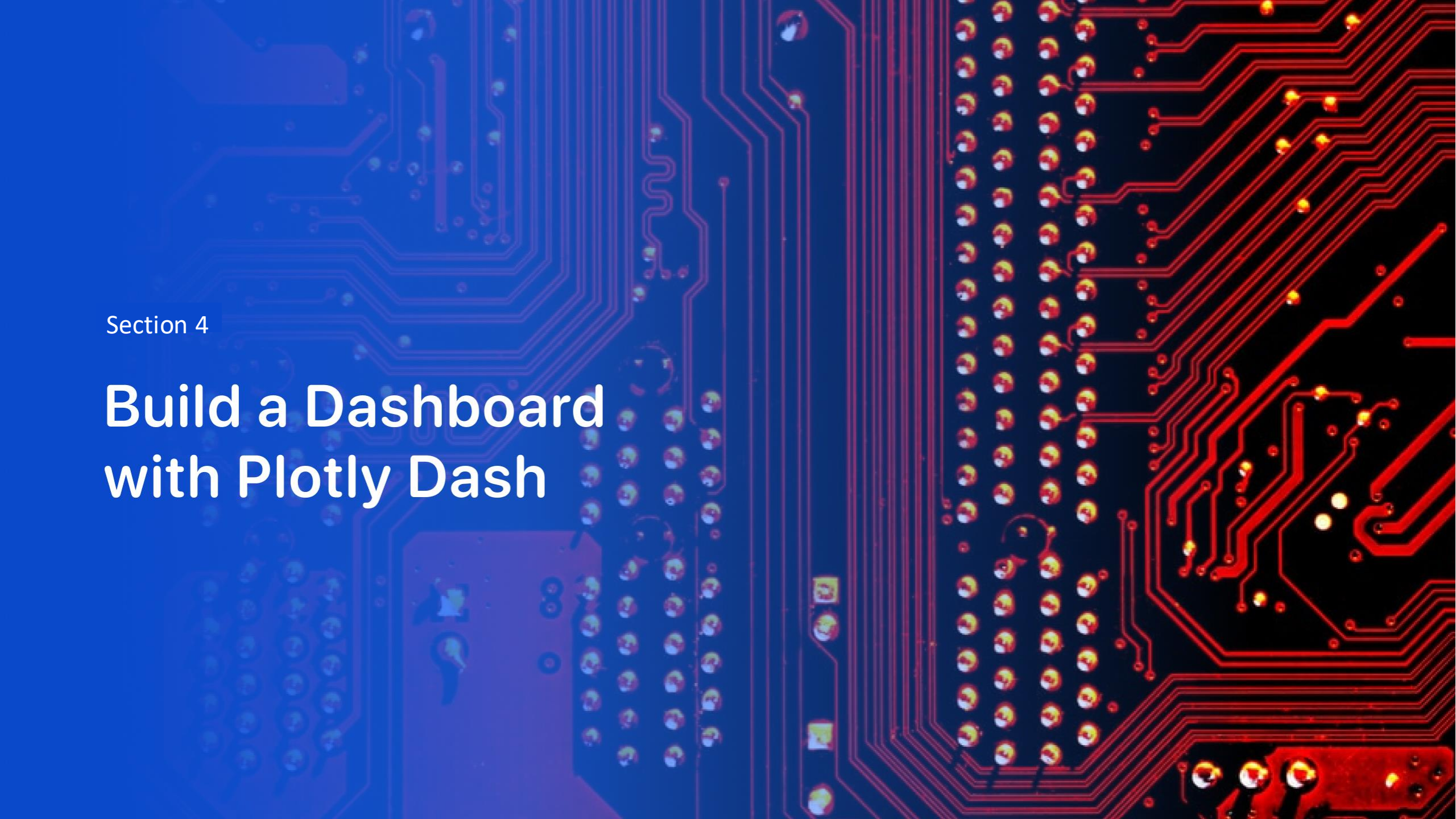
Green Markers = successful launch
Red Markers = failed launch



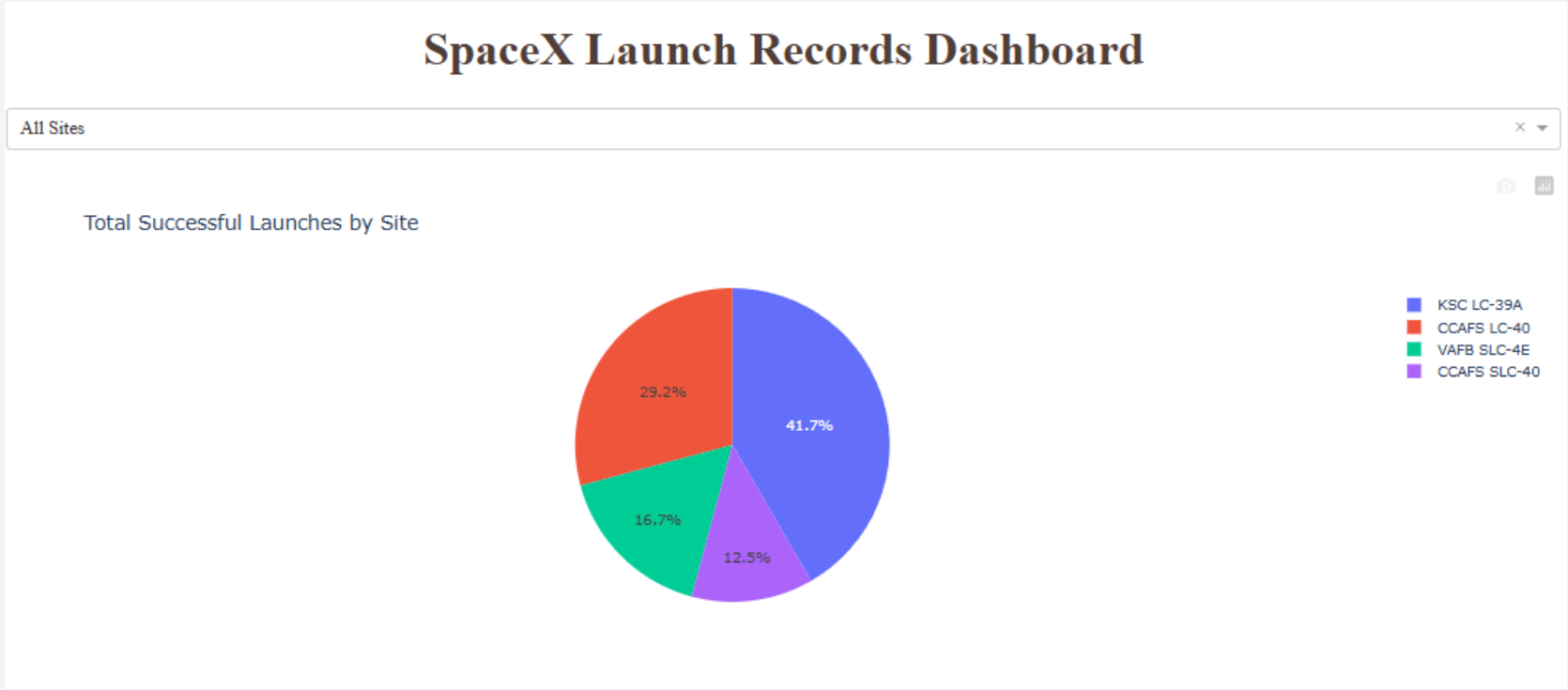
Results – Folium Map Demo

Measured distance from launch site to nearest infrastructure with distance label



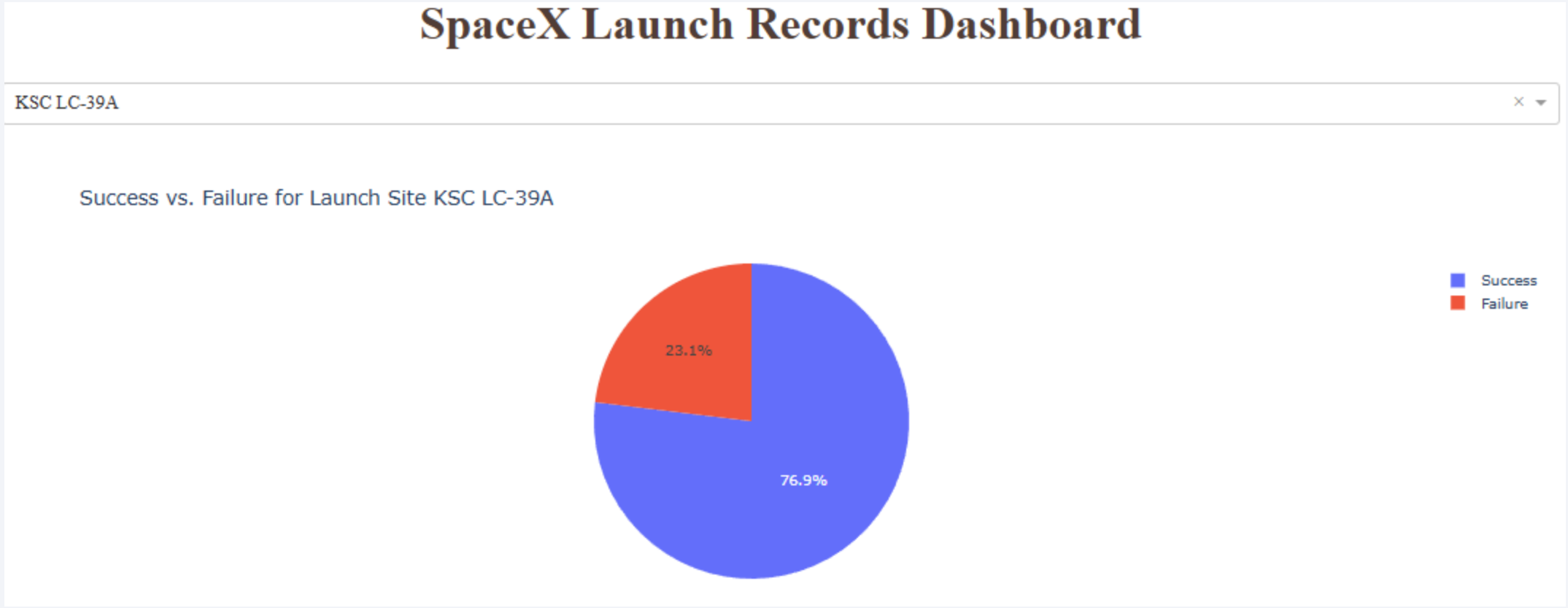

Section 4
Build a Dashboard with Plotly Dash

Results – Dash Dashboard Demo



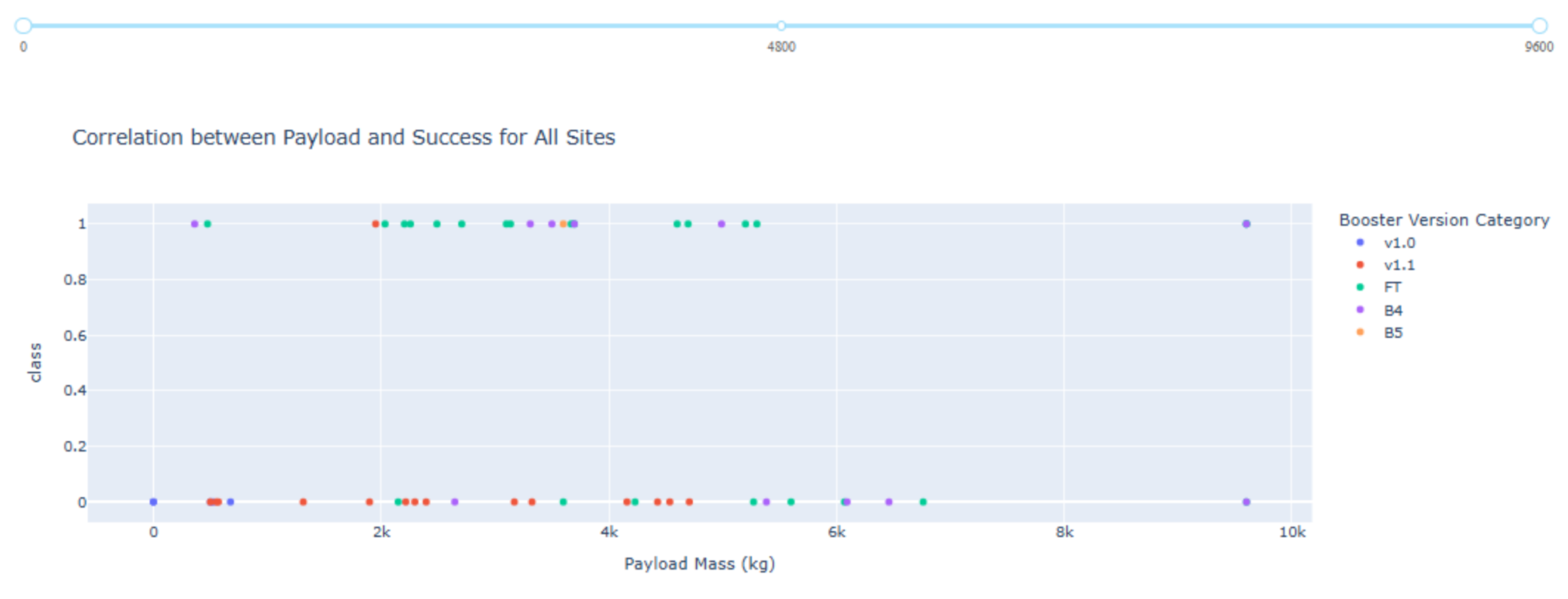
Distribution of successful launches across all sites.

Results – Dash Dashboard Demo



Success vs. failure breakdown for selected site.

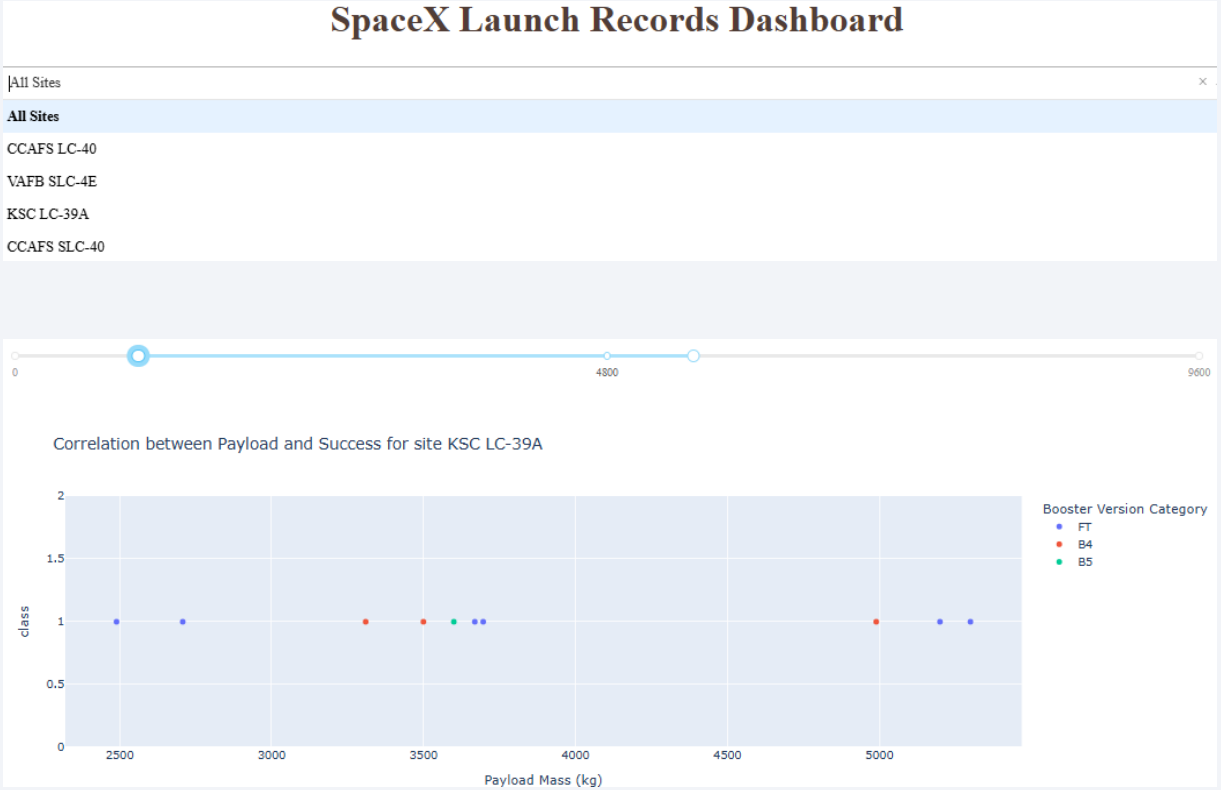
Results – Dash Dashboard Demo



Correlation between payload mass and launch outcome by Booster Version Category.

Results – Dash Dashboard Demo

User selects site and payload range to update all visualizations in real time.



Section 5

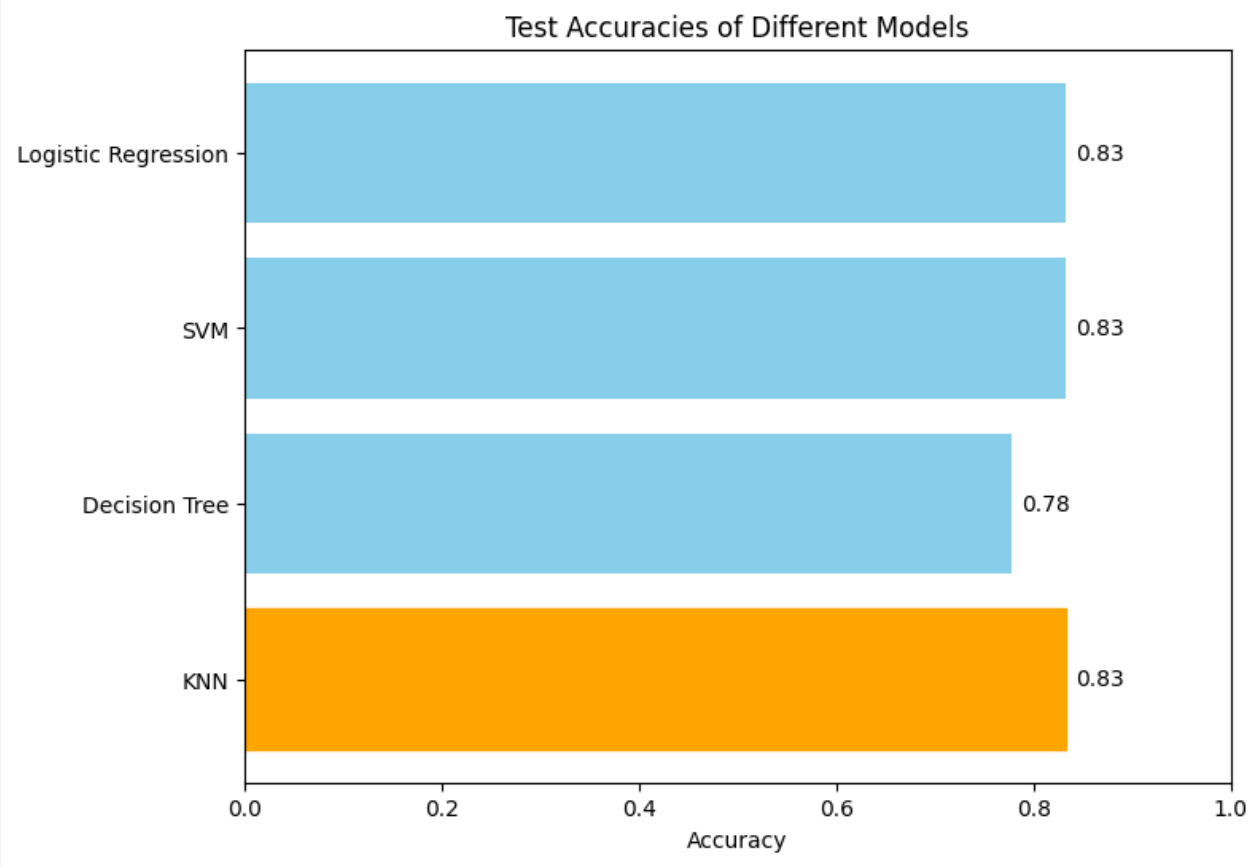
Predictive Analysis (Classification)

Classification Accuracy

KNN achieved the highest test accuracy (83.33%) with almost no difference difference to Logistic Regression and SVM.

Only the Decision Tree shows clear overfitting, performing much better on training than on test data.

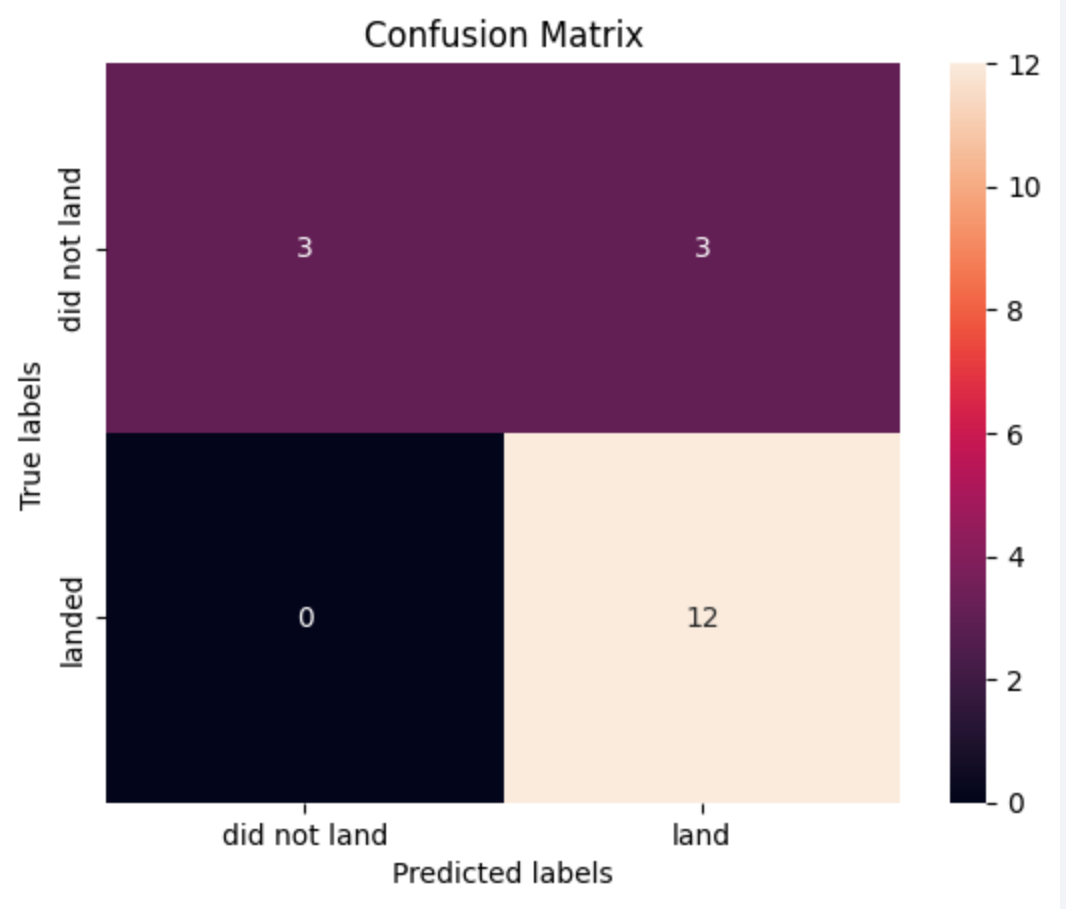
All other models are similarly robust on this task.



Confusion Matrix

- The model correctly predicted 12 successful landings (true positives) and 3 failed landings (true negatives).
- It made 3 false positive errors (predicted “landed” when it was actually “not landed”).
- There were no false negatives (no actual “landed” cases were missed).

This means KNN is very good at identifying successful landings, but sometimes predicts a landing when there was actually a failure. Overall, the model is reliable, with most errors being false alarms rather than missed landings.



Conclusions

- **First Stage Reusability:**
  - SpaceX's ability to land and reuse the Falcon 9 first stage is a major innovation, reducing the need to build new rockets for every launch and making spaceflight more sustainable.
- **Cost Reduction:**
  - Reusing rocket stages leads to significant cost savings. SpaceX can offer launches for around $62 million, compared to traditional costs of $165 million or more.
- **Role of Predictive Models:**
  - The classification models developed in this project (KNN, Logistic Regression, SVM, Decision Tree) aim to predict whether the first stage will land successfully.
  - Accurate predictions support mission planning, risk assessment, and cost estimation.
- **Model Performance:**
  - KNN achieved the highest test accuracy (83.33%), with Logistic Regression and SVM close behind.
  - The Decision Tree model overfit the training data and performed worse on the test set.
  - The small test sample size (18 samples) limits the reliability of these results; more data and further testing are needed for robust conclusions.

Conclusions

- **Business and Industry Impact:**
  - Reliable landing predictions enable better scheduling, resource allocation, and competitive pricing.
  - This strengthens SpaceX's market position and encourages further innovation in the aerospace industry.
- **Data-Driven Decision Making:**
  - Machine learning models empower companies to make informed, data-driven decisions, reducing uncertainty and improving operational efficiency in complex engineering projects like rocket launches.
- **Limitations:**
  - The model accuracy reflects prediction performance, not the actual technical landing success rate.
  - The real Falcon 9 landing success rate is even higher, as seen in operational data.
  - Model improvements and more data could further enhance prediction reliability and cost savings.

Appendix

Data sources and important links:

- SpaceX API: https://github.com/r-spacex/SpaceX-API
- Wikipedia Falcon 9 and Falcon Heavy launches (static snapshot): https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Important Python-snippet summary:

- Notebook for creating a Summary of important snippets : https://github.com/riomio98/falcon9-landing-prediction/blob/main/06_reporting/Appendix_Combined_From_Workspace.ipynb
- Notebook for Summary of important code snippets : https://github.com/riomio98/falcon9-landing-prediction/blob/main/06_reporting/combined_workspace.ipynb

Appendix

Key Python Code Snippets

````
## 1. Data Import and Preprocessing Example
```python
import pandas as pd
from pathlib import Path
# Load cleaned feature matrix and labels
data_path = Path('../05_model_training/dataset_part_2.csv')
X_path = Path('../05_model_training/dataset_part_3.csv')
data = pd.read_csv(data_path)
X = pd.read_csv(X_path)
Y = data['Class'].to_numpy()
```
````

Appendix

Key Python Code Snippets

````
## 2. GridSearchCV Setup for Models
```python
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)

# Logistic Regression
logreg_cv = GridSearchCV(LogisticRegression(max_iter=500),
                         param_grid={"C":[0.01,0.1,1], 'penalty':['l2'], 'solver':['lbfgs']},
                         cv=5)
logreg_cv.fit(X_train, Y_train)

# SVM
svm_cv = GridSearchCV(SVC(),
                      param_grid={'kernel':['linear','rbf'], 'C':[0.1,1]},
                      cv=5)
svm_cv.fit(X_train, Y_train)

# Decision Tree
tree_cv = GridSearchCV(DecisionTreeClassifier(random_state=2),
                      param_grid={'max_depth':[2,4,6], 'criterion':['gini','entropy']},
                      cv=5)
tree_cv.fit(X_train, Y_train)

# KNN
knn_cv = GridSearchCV(KNeighborsClassifier(),
                      param_grid={'n_neighbors':[3,5,7]},
                      cv=5)
knn_cv.fit(X_train, Y_train)
```
````

Appendix

Key Python Code Snippets

````
## 3. Visualization: Model Accuracies (Barplot)
```python
import matplotlib.pyplot as plt
import seaborn as sns

results = {
    'Logistic Regression': logreg_cv.best_score_,
    'SVM': svm_cv.best_score_,
    'Decision Tree': tree_cv.best_score_,
    'KNN': knn_cv.best_score_,
}
plt.figure(figsize=(6,3))
sns.barplot(x=list(results.values()), y=list(results.keys()), orient='h')
plt.xlabel('Cross-validated Accuracy')
plt.title('Model Training Accuracies')
plt.xlim(0,1)
plt.show()
```
````

Appendix

Key Python Code Snippets

````
## 4. Confusion Matrix Output & Visualization
```python
from sklearn.metrics import confusion_matrix

def plot_confusion_matrix(y_true, y_pred):
    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(3,3))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.title('Confusion Matrix')
    plt.show()

# Example for best model (e.g., Logistic Regression)
yhat = logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test, yhat)
print('Confusion matrix values:\n', confusion_matrix(Y_test, yhat))
```
````

Appendix

Key Outputs

```
### Test Accuracies (printed)
```
Logistic Regression: 0.833
SVM: 0.833
Decision Tree: 0.833
KNN: 0.833
```
```

Thank you!