IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Alhassan Osman
14/04/2024

Outline

Executive Summary

In our analysis, we employ various methodologies to extract insights from SpaceX mission data. We are beginning by collecting data via REST API and Web Scraping methods, followed by Data Wrangling to clean and prepare the dataset. Exploratory Data Analysis (EDA) is then conducted to uncover the key insights, while Interactive Analytics provides real-time exploration capabilities. Additionally, Predictive Analysis using classifiers allows us to forecast landing outcomes.

As a result:

- EDA reveals important information regarding launch site performance, payload distributions, and trends.
- Interactive analytics facilitates dynamic exploration, enhancing user engagement.
- Predictive analysis identifies effective classifiers for accurately forecasting landing outcomes.

These methodologies and their corresponding results collectively offer valuable insights for decision-making in SpaceX missions.

Introduction

- **Project background and context**

In modern space exploration, private companies like SpaceX have revolutionized the industry, making space travel more accessible and cost-effective. SpaceX's Falcon 9 rocket, renowned for its reusability, has become a cornerstone of this transformation. However, ensuring the success of Falcon 9 first stage landings remains a critical challenge for optimizing mission efficiency and cost-effectiveness.

- **Problems we want to find answers**

This project seeks to address several key questions and challenges:

1. *Predicting First-Stage Landings*: Can we develop a reliable predictive model to forecast the outcome of Falcon 9 first-stage landings?

2. *Cost Optimization*: How can accurate predictions of first-stage landings contribute to cost optimization for space missions?

Introduction

3. ***Data-driven Insights***: What insights can be gleaned from analyzing past launch data, and how can these insights inform decision-making processes?

4. ***Model Evaluation***: Which machine learning algorithms offer the best performance in predicting first-stage landing outcomes, considering factors such as accuracy, overfitting, and underfitting?

By delving into these questions, this project aims to contribute valuable insights and solutions to the ongoing quest for efficient and cost-effective space exploration.

Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

- We collect data from the SpaceX REST API, focusing on past launches.
- We use web scraping techniques to gather Falcon 9 launch records from relevant Wiki pages

Perform data wrangling

- We perform data wrangling to clean and prepare the collected data, filtering out irrelevant entries “Falcon 1” launches and handling missing values.

Methodology

Executive Summary

Perform exploratory data analysis (EDA) using visualization and SQL

- We analyze attributes like launch site, payload mass, orbit, and outcome.
- We determine correlations between attributes and successful landings.
- We prepare data for predictive modeling by converting categorical variables using one-hot encoding.

Perform interactive visual analytics using Folium and Plotly Dash

- We utilize Folium to visualize launch site locations and proximities.
- We build an interactive dashboard with Plotly Dash to explore insights from the SpaceX dataset.

Methodology

Executive Summary

Perform predictive analysis using classification models

- We construct a machine-learning pipeline to predict the success of Falcon 9 first-stage landings.
- We preprocess data and split it into training and testing sets.
- We implement Grid Search to optimize hyperparameters for different algorithms.
- We evaluate model performance using confusion matrices.

Data Collection and Wrangling

Data collection involves gathering data from REST API and Web Scraping. For REST API, we use a GET request, decode JSON responses, and transform them into Pandas' data frames. Web Scraping involves extracting launch records from Wikipedia tables using BeautifulSoup.

In Data Wrangling, we clean and organize the data, calculate launch site statistics, and create labels for landing outcomes. This prepares the data for Exploratory Data Analysis (EDA) and future analysis.

Data Collection – SpaceX API   NoteBook

Parse the data

- spacex_url = "https://api.spacexdata.com/v4/launches/past"
- response = requests.get(spacex_url)

Convert json into DataFrame

- pd.json_normalize(response.json())

Data Wrangling

- data_falcon9 = dataframe[dataframe.BoosterVersion=="Falcon 9"]
- PayloadMassMean = data_falcon9.PayloadMass.mean()
- data_falcon9.replace(np.nan, PayloadMassMean)

Data Collection – Scraping

NoteBook

HTTP GET Request and Extract the Data

- response = requests.get(static_url)
- soup = BeautifulSoup(response.text, 'html.parser')
- html_tables = soup.find_all('table')

Convert HTML into DataFrame

- launch_dict= dict.fromkeys(column_names)
- df = pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })

Data Wrangling

- Dealing with missing data
- Filtering the related columns

Data Wrangling

[NoteBook]

Figuring out some insight

- df.isnull().sum()/len(df)*100
- df['Launch Site'].value_counts()
- df['Orbit'].value_counts()
- df['Outcome'].value_counts()

Create target column

- bad_outcomes = set(landing_outcomes.keys()[[1,3,5,6,7]])
- landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome’]]
- df['Class’] = landing_class

Export and save the file

- df.to_csv("dataset_part_2.csv", index=False)

EDA with Data Visualization NoteBook

In our exploratory data analysis (EDA), we use a variety of charts to delve into different aspects of our dataset:

Flight Characteristics:

- Explore the relationship between flight number and payload.
- Examine how launch sites were distributed across flight numbers.

Launch Site Analysis:

- Investigate payload distribution across launch sites.
- Explore launch site preferences over time.

EDA with Data Visualization

Orbit Success Rates:

- Compare the success rates of different orbit types using a bar chart.

Orbit Selection Patterns:

- Analyze orbit type distribution across flight numbers.
- Explore payload variations across orbit types.

Trend Analysis:

- Visualize the yearly trend in launch success rates using a line chart.

Each visualization contributes to our understanding of the dataset, revealing patterns, trends, and relationships that informs our analysis and decision-making process.

EDA with SQL

NoteBook

In our exploratory data analysis (EDA) using SQL queries, we investigate different facets of the space mission dataset:

Launch Sites:

- Explore the diversity of launch sites involved in missions.

Payload Analysis:

- Calculate total and average payload masses, providing insights into cargo capacity and distribution.

Landing Outcomes:

- Investigate successful and failure landing events, both on ground pads and drone ships, highlighting operational challenges and achievements.

EDA with SQL

Booster Performance:

- Analyze booster capabilities, considering factors such as payload mass and landing success.

Temporal Analysis:

- Track mission outcomes and landing events over time, revealing historical patterns and milestones.

These analyses provided comprehensive insights into launch site operations, payload characteristics, landing outcomes, and booster performance, offering valuable information for further exploration and decision-making in the space mission domain.

Build an Interactive Map with Folium NoteBook

In our Folium map visualization, we utilize various map objects to enhance the representation of launch sites and their surroundings

Launch Site Highlighting:

- Employ Folium's Circle and Marker objects to highlight launch sites with labeled coordinates, improving their visibility on the map.

Marker Clustering:

- Enhance map readability by implementing the ***MarkerCluster*** object, which consolidates multiple markers at the same coordinates.

Build an Interactive Map with Folium

Mouse Position Tracking:

- Integrate the ***MousePosition*** feature, allowing users to obtain coordinates by hovering over points of interest, enhancing interactivity.

Route Visualization:

- Utilize Folium's ***PolyLine*** object to draw lines connecting launch sites to nearby cities, railways, and highways, providing spatial context and transportation information.

These map objects collectively creates an informative and interactive map visualization, offering insights into launch site locations and their surroundings.

Build a Dashboard with Plotly Dash NoteBook

In our interactive dashboard for SpaceX launch data analysis, we incorporate several key features and interactions to facilitate real-time visual analytics:

Launch Site Dropdown Menu

- Users can select from four different launch sites using a dropdown menu, providing flexibility to focus on specific launch locations of interest.

Success Pie Chart:

- A callback function dynamically renders a pie chart based on the selected launch site from the dropdown. This chart visually represents the distribution of launch success counts, offering insights into the performance of each site.

Build a Dashboard with Plotly Dash

Payload Range Slider:

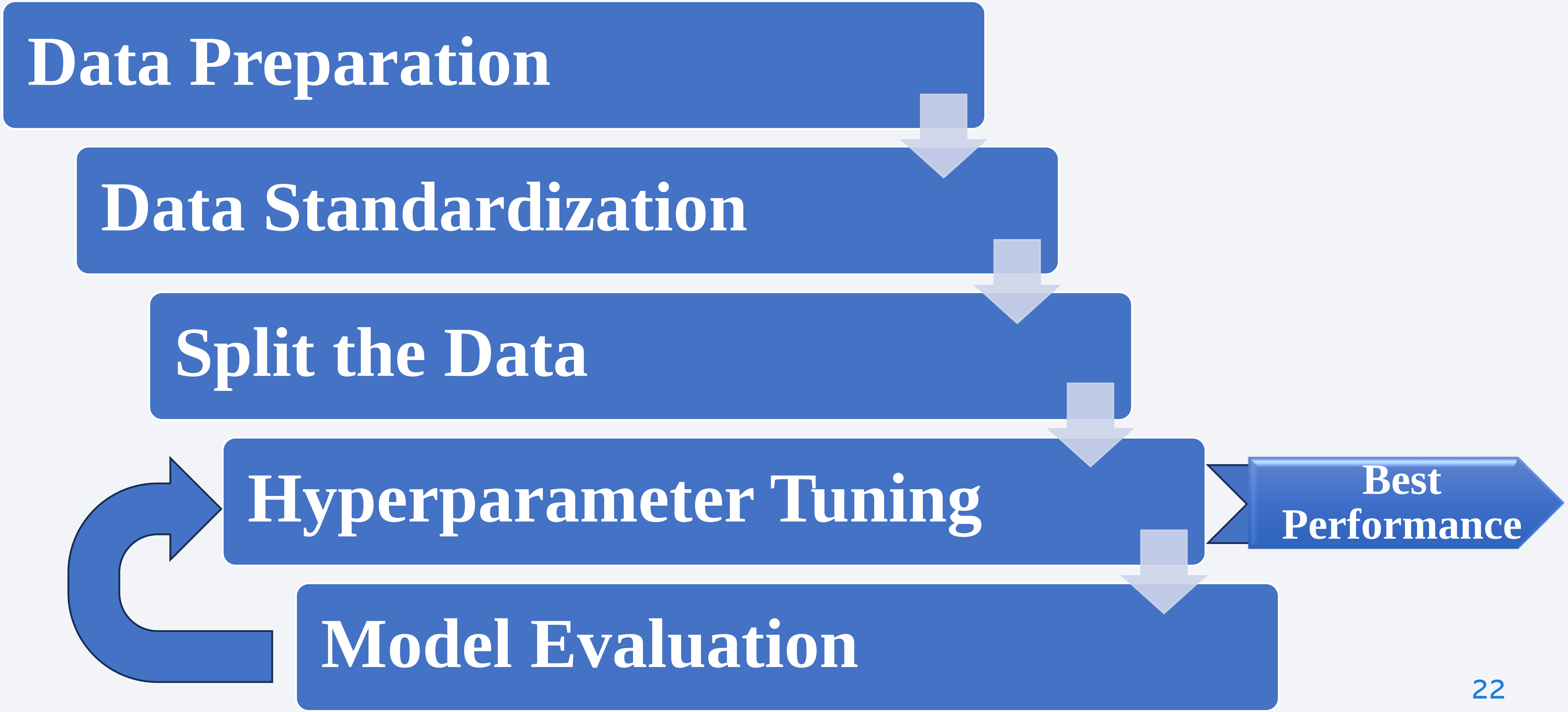
- A range slider allows users to select different payload ranges easily, enabling the exploration of potential correlations between payload mass and mission outcomes.

Success-Payload Scatter Chart:

- Another callback function renders a scatter plot depicting the relationship between payload mass and mission success for the selected launch site(s). This interactive chart provides a visual analysis of how payload influences mission outcomes.

These features enable users to interactively explore SpaceX launch data, uncover patterns, and gain insights into factors influencing mission success.

Predictive Analysis (Classification) NoteBook



Predictive Analysis (Classification)

In our model development process to predict first-stage landing outcomes based on preceding lab data, we follow a systematic approach:

Data Preparation:

- Create a NumPy array from the 'Class' column in the dataset, which represents the first stage landing outcomes.

Data Standardization:

- Standardize the data to ensure uniformity and facilitate model training by scaling the features.

Split the Data:

- Utilize the ***train_test_split*** function to partition the data into training and test sets, separating features (X) and target variables (Y).

Predictive Analysis (Classification)

Hyperparameter Tuning:

- Conduct a search for optimal hyperparameters for four different classifiers: Logistic Regression, SVM, Decision Tree, and KNN. This step involves fine-tuning model parameters to enhance predictive performance.

Model Evaluation:

- Evaluate the performance of each classifier using the test data to determine the method that performs best. This step involves assessing metrics such as accuracy, precision, recall, and F1-score to identify the most effective predictive model.

By systematically following these steps, we aim to develop a robust predictive model capable of accurately forecasting first-stage landing outcomes based on the available data.

Results

Summary of analysis results:

Exploratory Data Analysis (EDA):

- We explore the dataset and identify patterns, distributions, and relationships through visualizations and statistical summaries. This helps us gain a comprehensive understanding of the data's characteristics.

Interactive Analytics Demo:

- Using screenshots or visuals from our interactive dashboard, we demonstrate the dynamic nature of our analysis tool. Users can explore and interact with the data in real time, facilitating deeper insights and informed decision-making.

Results

Predictive Analysis:

- We develop predictive models to forecast first-stage landing outcomes based on the dataset. By evaluating different classifiers and selecting the best-performing model, we gain insights into the factors influencing mission success.

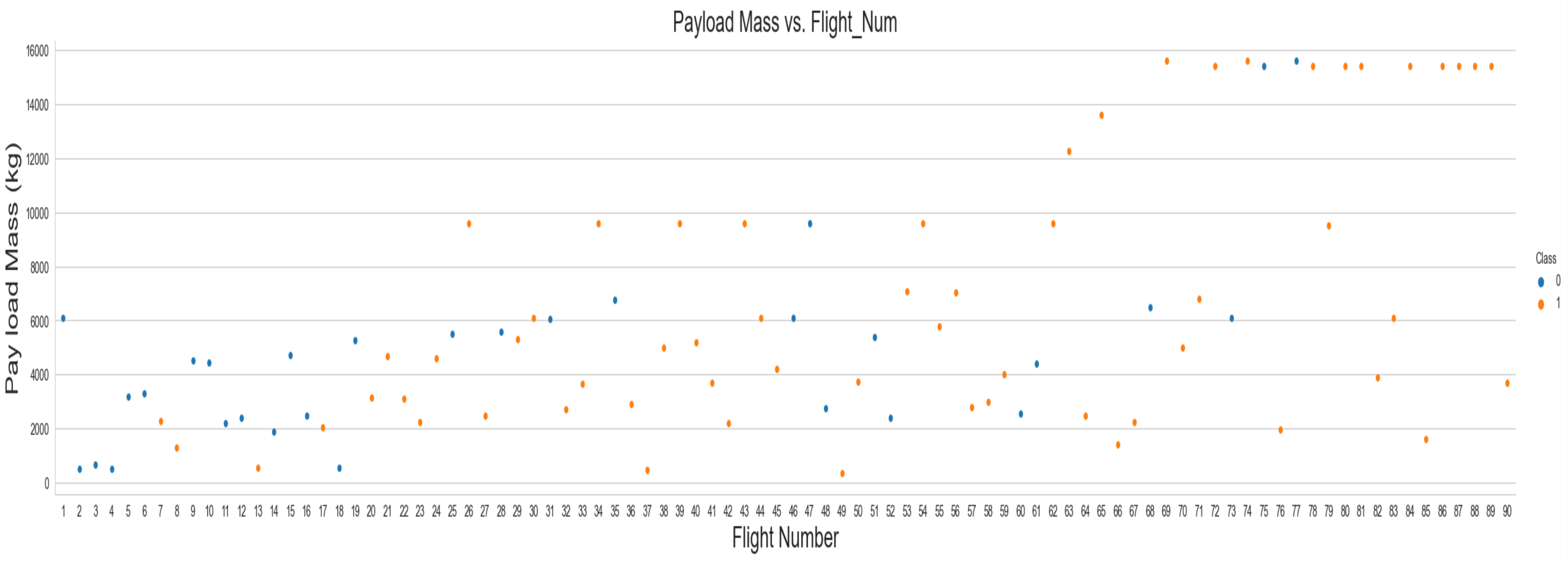
These summaries provide an overview of our analysis process, highlighting key findings and insights obtained from exploring, interacting with, and predicting outcomes using the dataset.

Section 2

Insights drawn from EDA

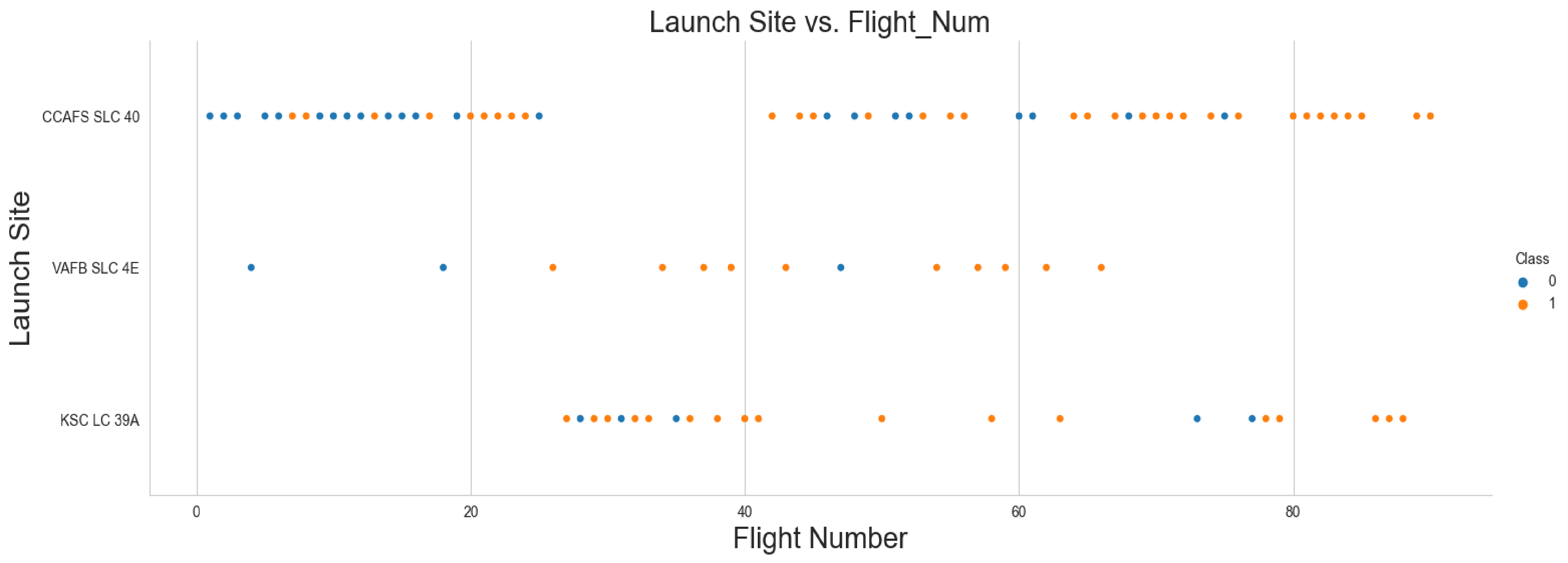
Flight Number vs. Payload

NoteBook



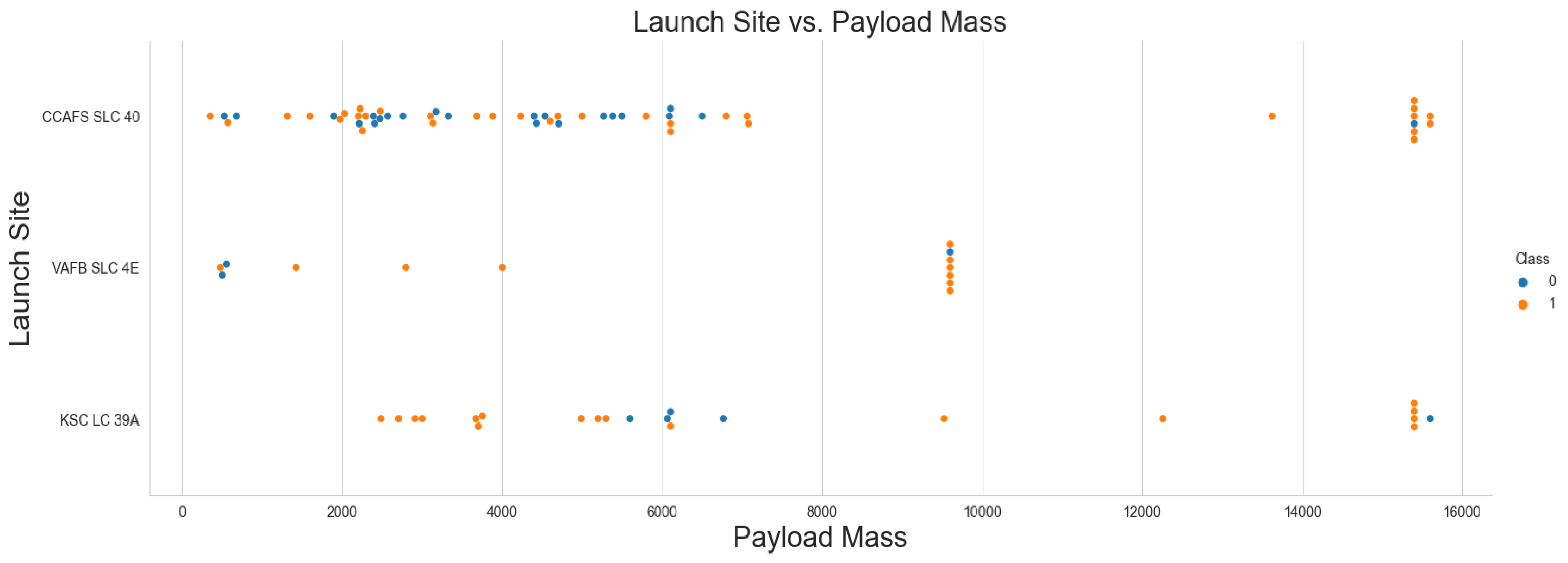
We can see a ***positive moderate correlation*** between flight number and payload, which means that most of the early flights have low to average payloads. In contrast, most newer flights are heavy payloads, which means some more advanced equipment has been added.

Flight Number vs. Launch Site



Most flights launched from ***CCAFS SLC 40.*** For the landing success rate, ***KSC LC 39A*** has the best rate of **77.2%** but its last flight number is **66**, then comes ***VAFB SLC 4E*** with a rate of **76.9%** while it does not have a flight before flight number **27**

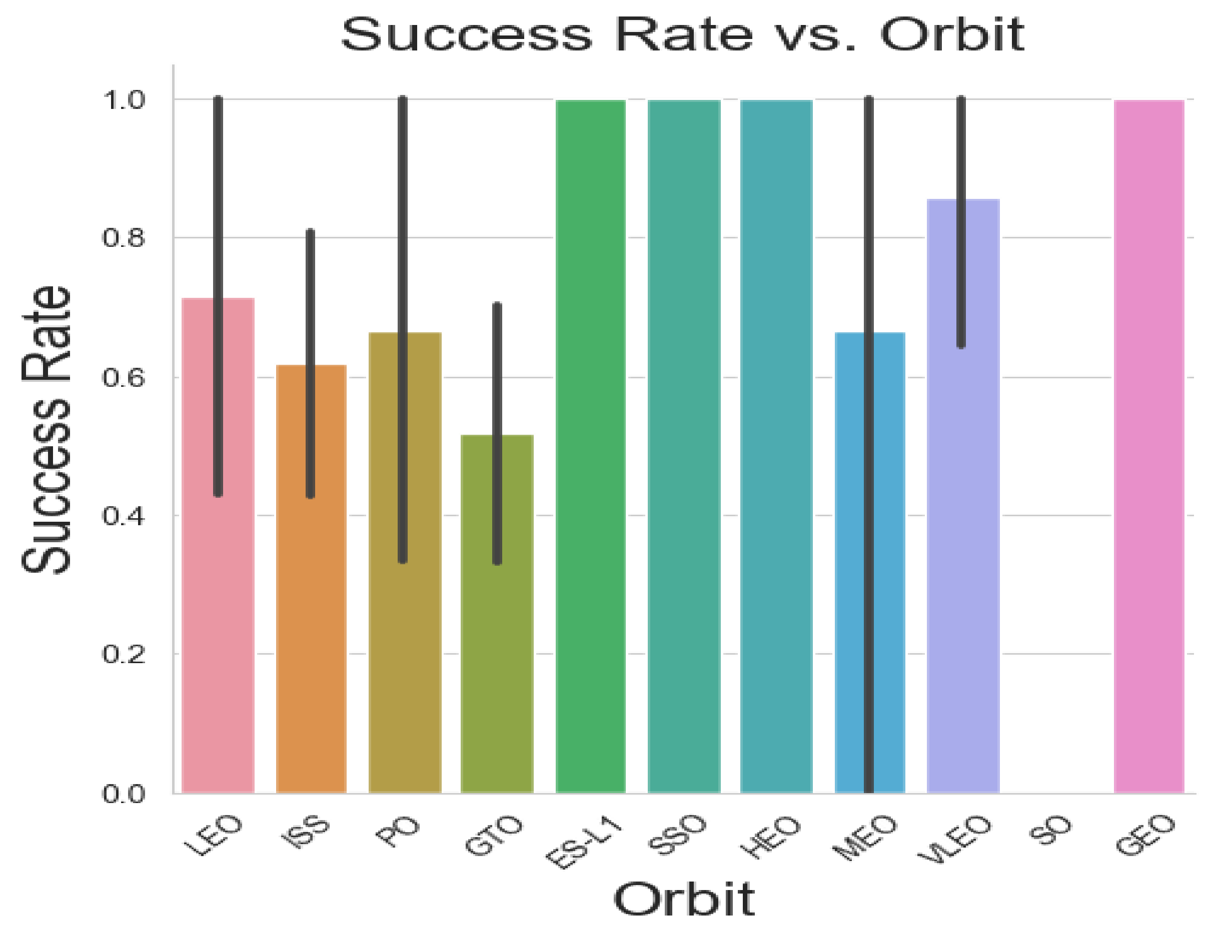
Payload vs. Launch Site



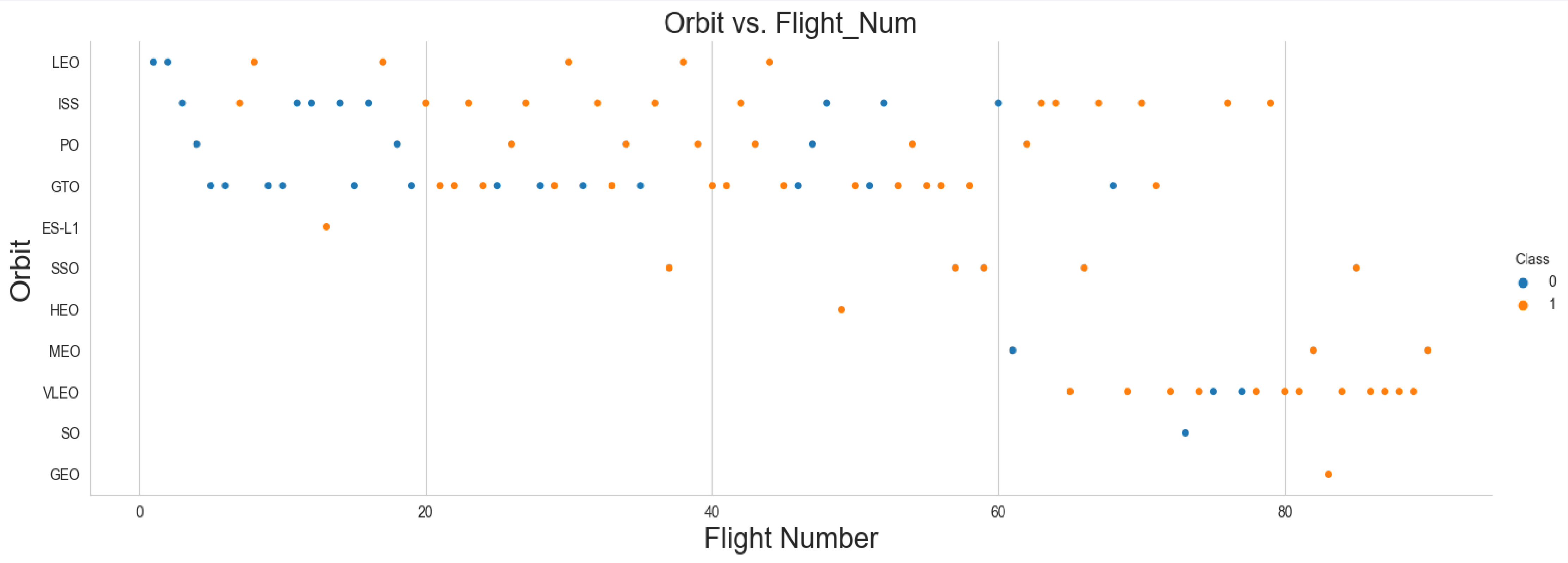
Most of the successful landings have payloads of more than **7000**, and only **3** of them are failures. Also, all the payloads of the flights from the site VAFB SLC 4E are less than **10000**. While all the payloads of the flights from the site KSC LC 36A are more than **20000**.

Success Rate vs. Orbit Type

We can see that: *ES-L1*, *SSO*, *HEO*, and *GEO* orbits have the highest landing success rates, while GTO and ISS have the lowest landing success rates.



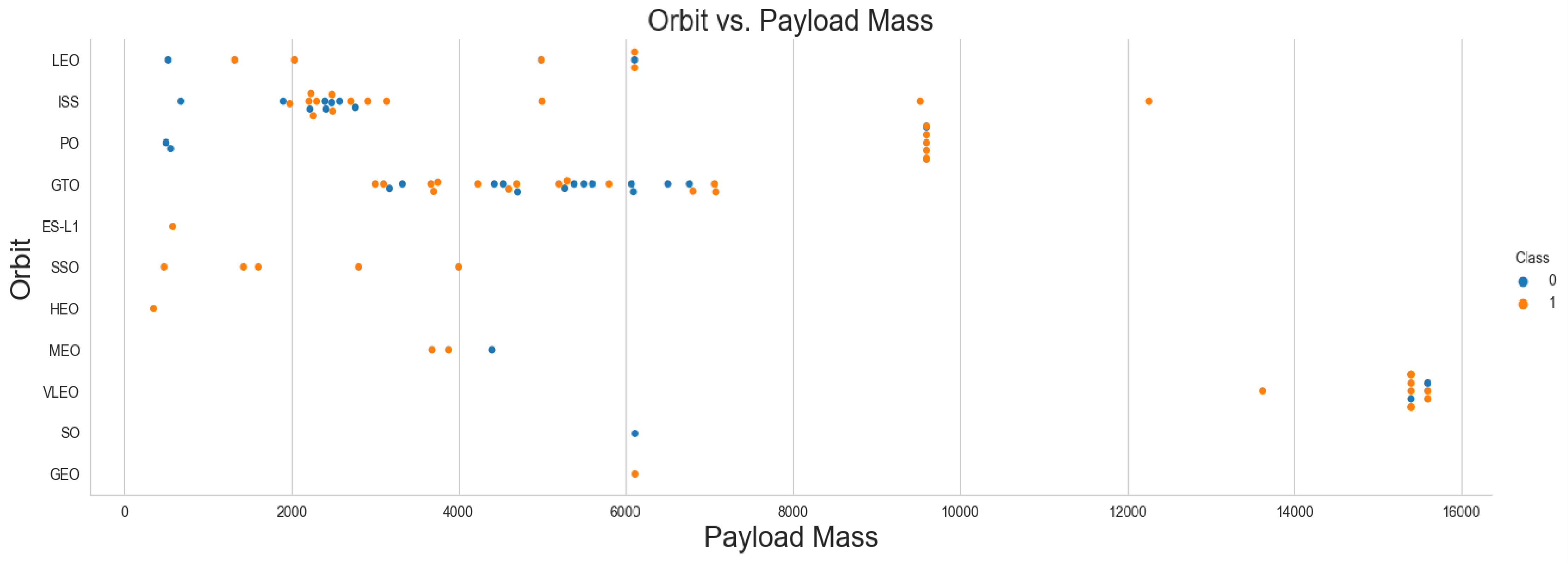
Flight Number vs. Orbit Type



The orbits with the highest landing success rate have one or a few flight numbers.

GTO and ISS have the lowest successful landing rate but have more than 50% of the flight numbers.

Payload vs. Orbit Type

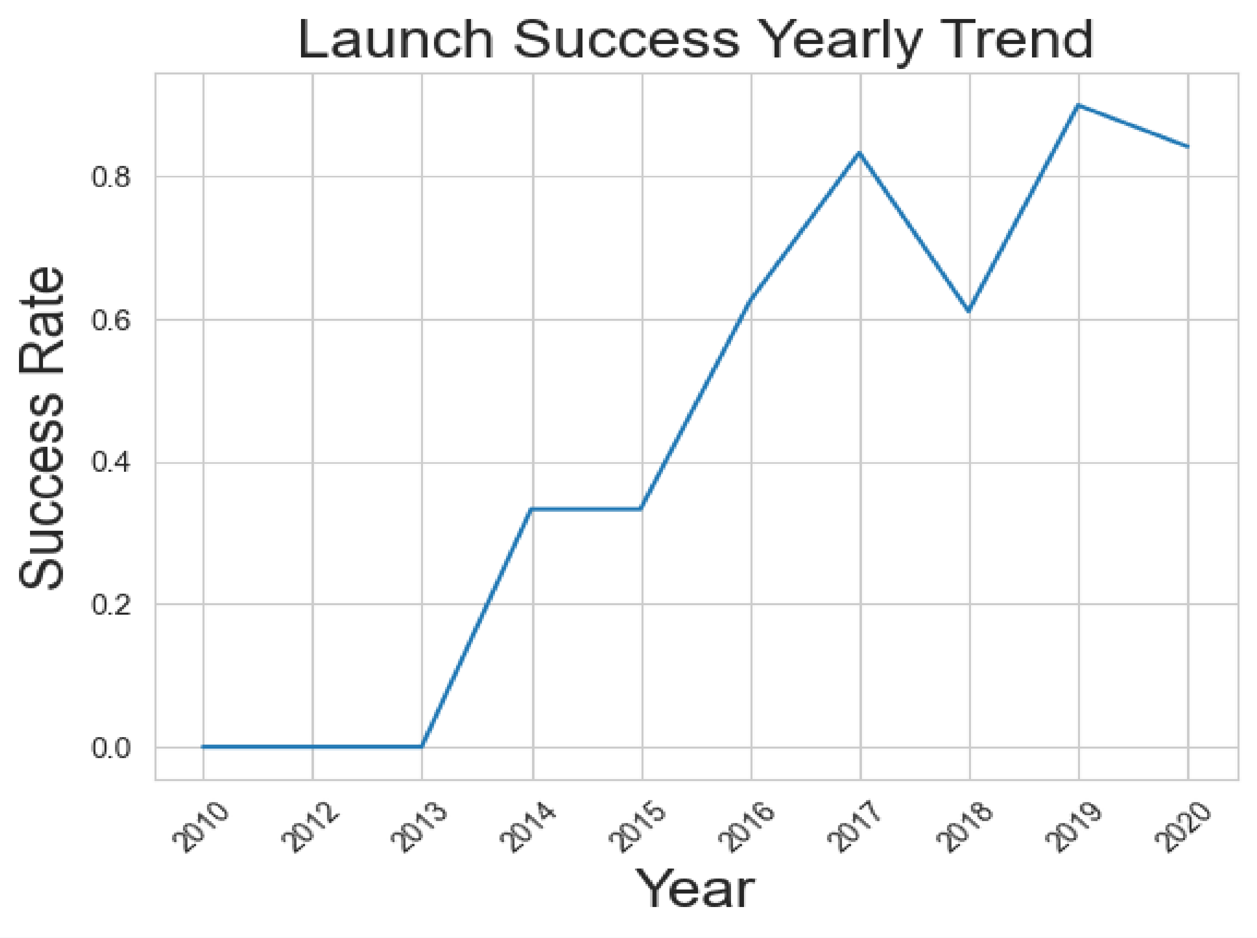


Only *ISS*, *PO*, *GTO*, and *VLEO* orbits have payloads of more than 7000, all of them successful but 3 flights, the remaining orbits its payloads is less than 7000.

Launch Success Yearly Trend

The first successful controlled landing was in **2014**, and since that year the rate of successful landings has increased linearly, except small drop in **2018**.

The best successful landing rate was in 2019 of rate more than **90%**.



All Launch Site Names

[NoteBook]

Using SQL, we can see that we have only three distinct launch sites, at the first glimpse the result shows four launch sites, but CCAFS LC -40 and CCAFS SLC -40 are the same in my opinion.

Click here to ask Blackbox to help you code faster

```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

✓ 0.0s

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

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

Launch Site Names Begin with 'CCA'

💡 Click here to ask Blackbox to help you code faster

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

✓ 0.0s Python

```
 * 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

When browsing the data of the launch site starting with 'CCA' we see that up to **2016** the launch site name was CCAFS LC -40 then as of **2017** the name changed to CCAFS SLC -40. See the first five data for example.

Total Payload Mass

```
Click here to ask Blackbox to help you code faster
%sql SELECT COUNT(PAYLOAD_MASS__KG_) AS 'Count of Flights', SUM("PAYLOAD_MASS__KG_") AS 'Total Payloads of NASA (CRS)' FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

✓ 0.0s

* sqlite:///my_data1.db
Done.

Count of Flights	Total Payloads of NASA (CRS)
20	45596

All flights for customer "**NASA (CRS)**" have small payloads, the total is 45596, only three flights for **SpaceX** can equal the payloads of 20 flights for "**NASA (CRS)**"

Average Payload Mass by F9 v1.1

Click here to ask Blackbox to help you code faster

```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1';
```

✓ 0.0s

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

AVG(PAYLOAD_MASS__KG_)
2928.4

It seems that the early Booster versions have small payloads, while the later ones have large payloads. For example, 'F9 v1.1' is an early booster version that has an average payload of 2928.4 KG for a flight.

First Successful Ground Landing Date

```
💡 Click here to ask Blackbox to help you code faster
%sql SELECT MIN(date) FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)';

✓ 0.0s
```

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

MIN(date)
2015-12-22

The first Success (ground pad) was on **2015-12-22**, but the first successful controlled landing was on **2014-04-18** it landed in the ocean.

Successful Drone Ship Landing with Payload between 4000 and 6000

Click here to ask Blackbox to help you code faster

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

✓ 0.0s

* sqlite:///my_data1.db
Done.

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

All the average payload flights that are weighted between (**4000 KG and 6000 KG**) and have **Successful (drone ship)** landing outcomes are of these booster versions: F9 FT B1022, F9 FT B1026, F9 FT B1021.2, and F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

Click here to ask Blackbox to help you code faster

```
%sql SELECT COUNT(CASE WHEN "Mission_Outcome" LIKE 'Success%' THEN 1 END) AS Successful_Outcomes, COUNT(CASE WHEN "Mission_Outcome" LIKE 'Failure%' THEN 1 END) AS Failure_Outcomes FROM SPACEXTABLE;
```

✓ 0.0s

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

Successful_Outcomes	Failure_Outcomes
100	1

Only one mission outcome failed, all the other **100** mission outcomes have succeeded.

Boosters Carried Maximum Payload

Click here to ask Blackbox to help you code faster

```
#%sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE);
%sql SELECT Booster_Version, (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE) AS 'Maximum Payload Mass' FROM SPACEXTABLE  WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE);
```

✓ 0.0s

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

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

We filtered out the flights that have the maximum payload, they are **12** flights.

2015 Launch Records

Click here to ask Blackbox to help you code faster

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

✓ 0.0s

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

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

Only two flights of drone ships' landing outcomes failed in 2015.

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

Click here to ask Blackbox to help you code faster

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

✓ 0.0s

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

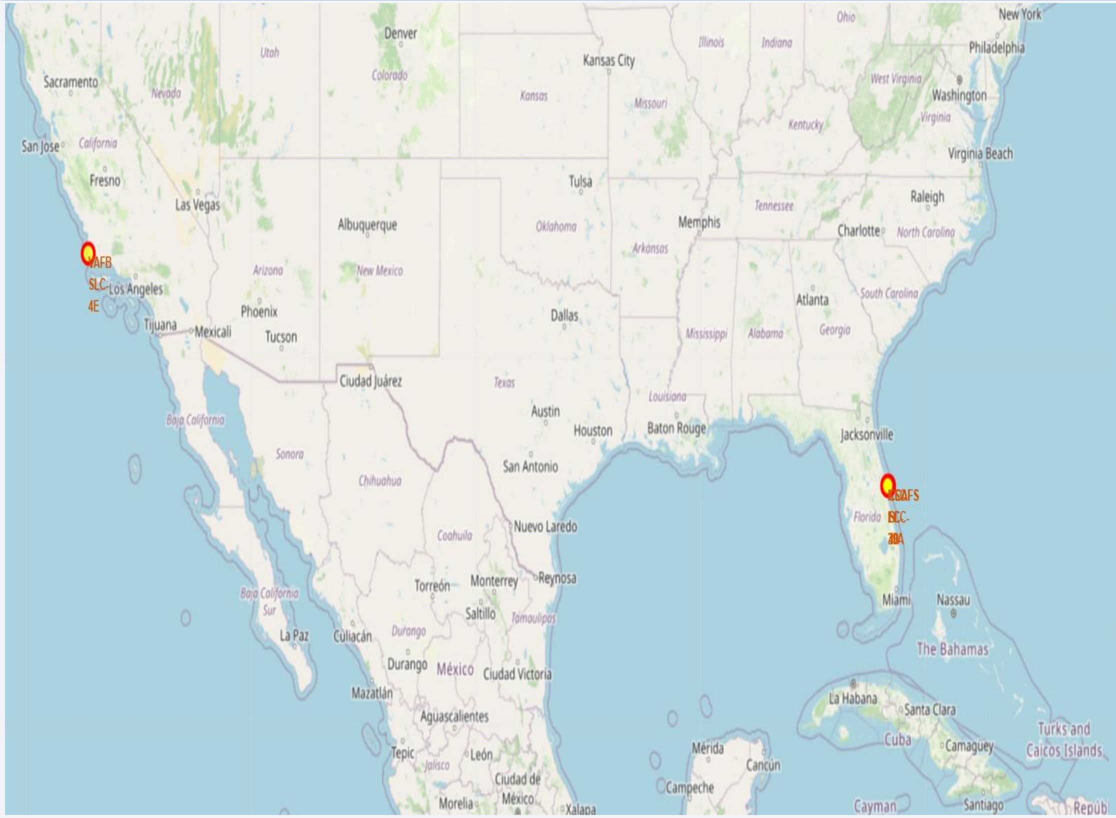
Landing_Outcome	Count_Of_Outcomes
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

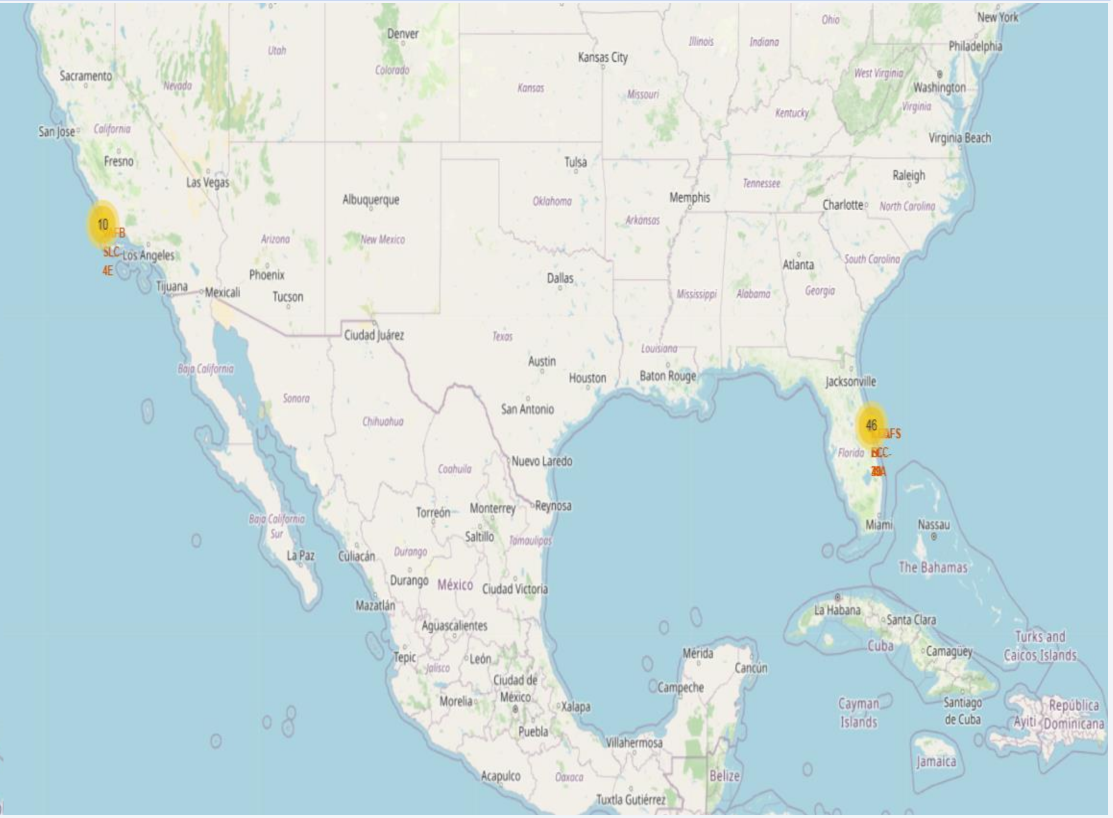
We have *31* flights between **2010-06-04** and **2017-03-20**, and one-third of them have no attempts landing outcomes.

Section 3

Launch Sites Proximities Analysis

Launch Site - Clusters Location [NoteBook]

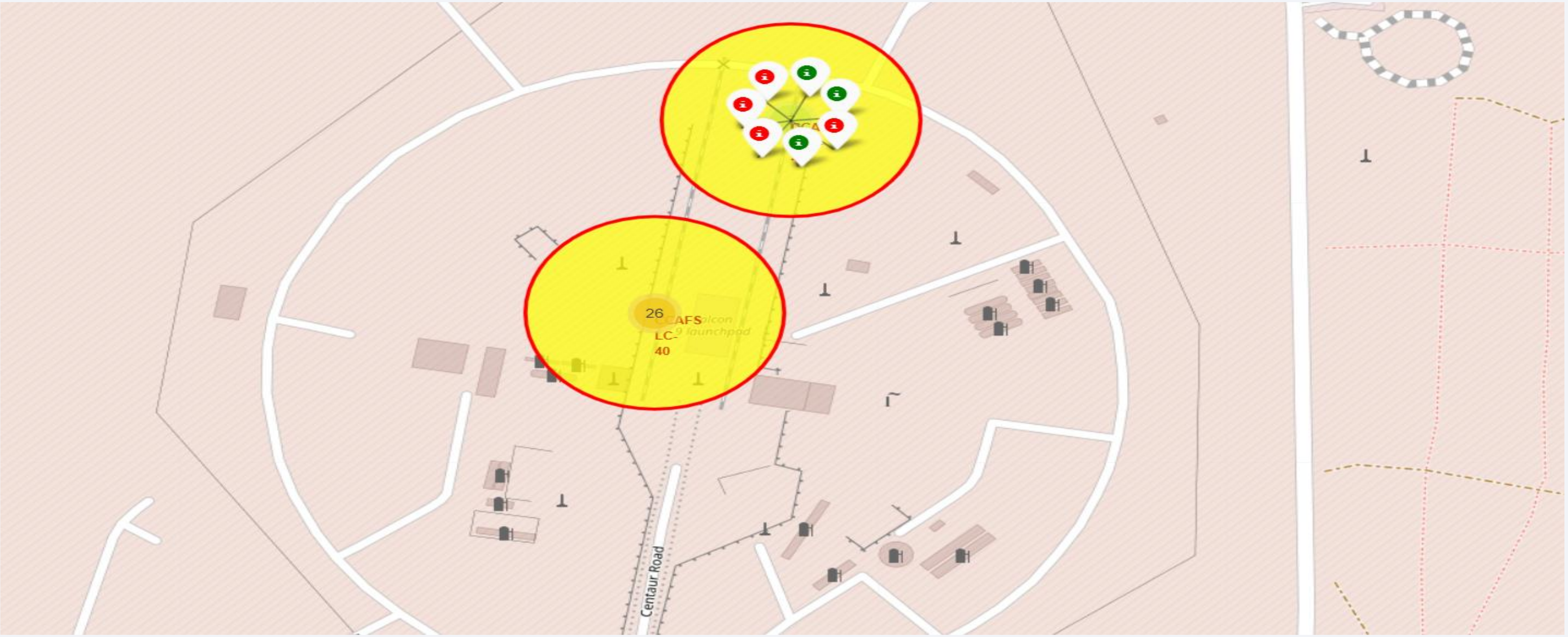




We can see that all launch site locations are near the ocean.

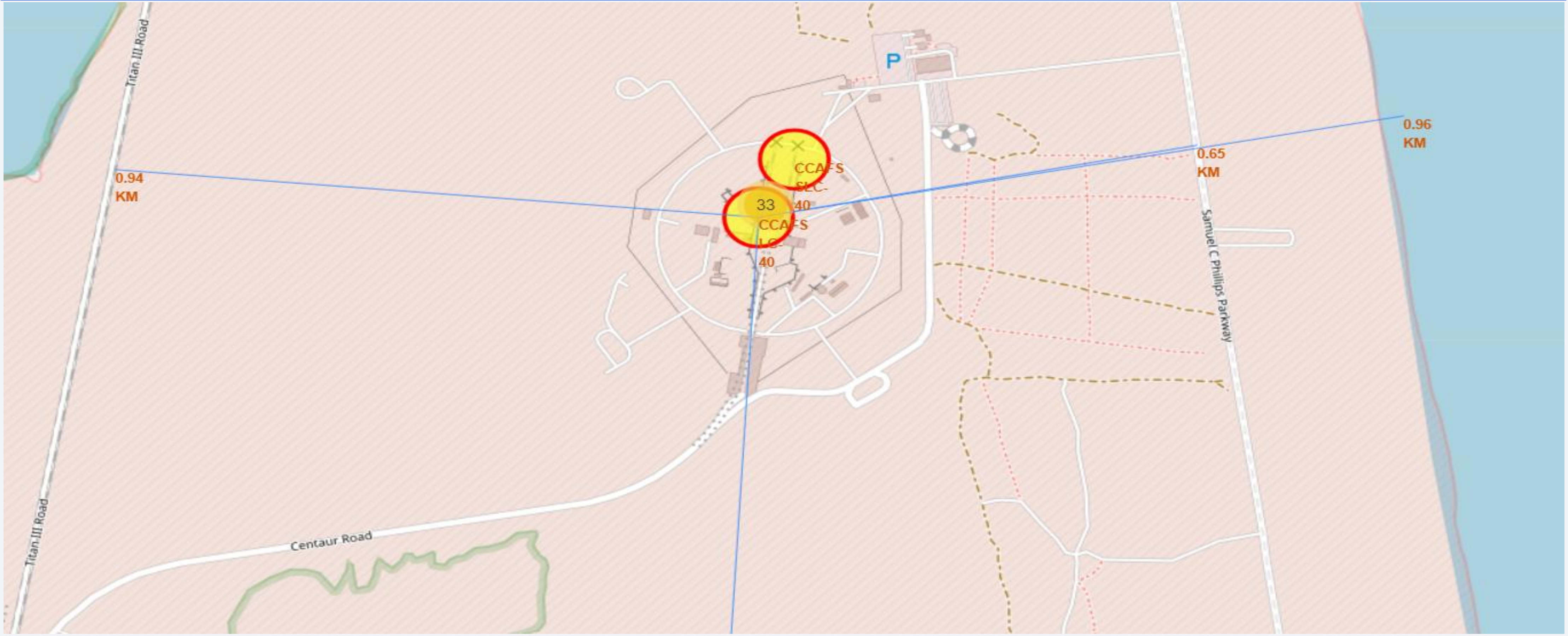
The map on the right is clustered by the number of all flights with the same launch site location.

Success / Fail by Site



Here we have a map that distinguishes successful flights ‘green color’ from failed flights ‘red color’.

Proximities from Town



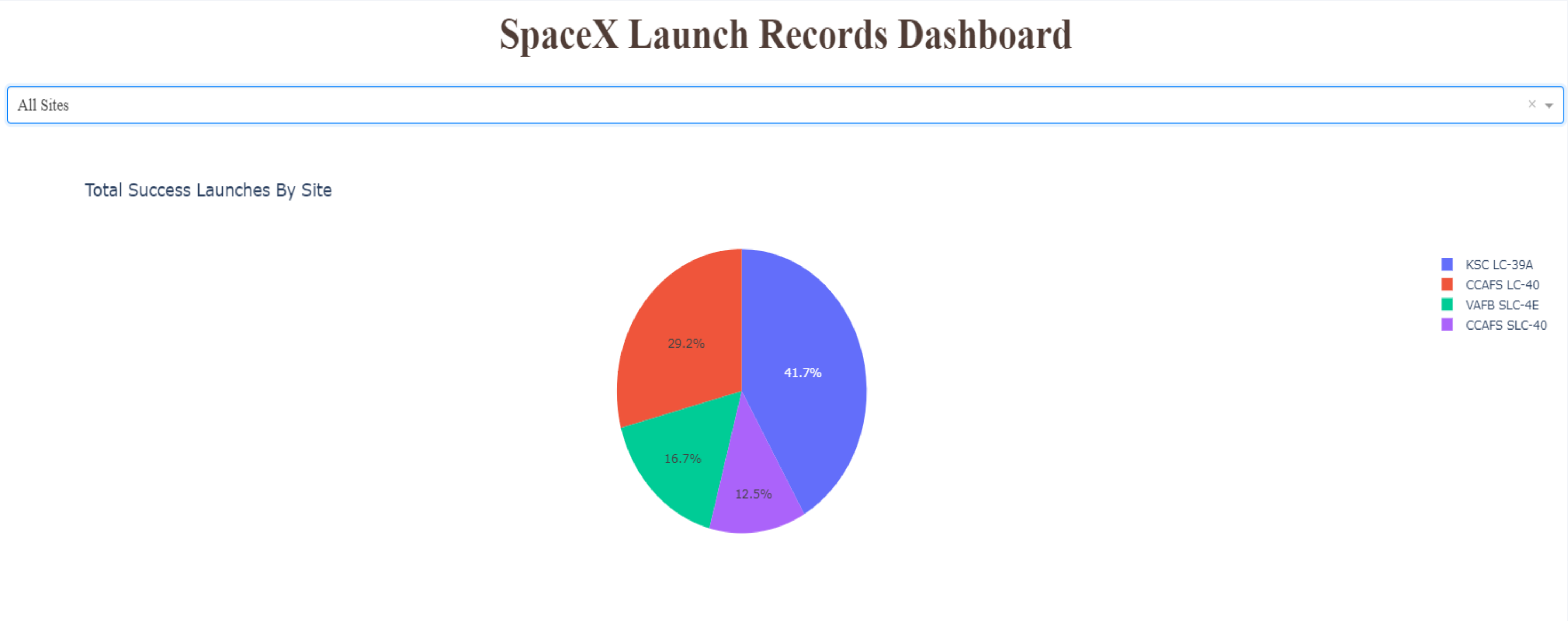
The launch site here is near to **ocean** and **railway**, but so far from **cities** and **main roads**.

Section 4

Build a Dashboard with Plotly Dash

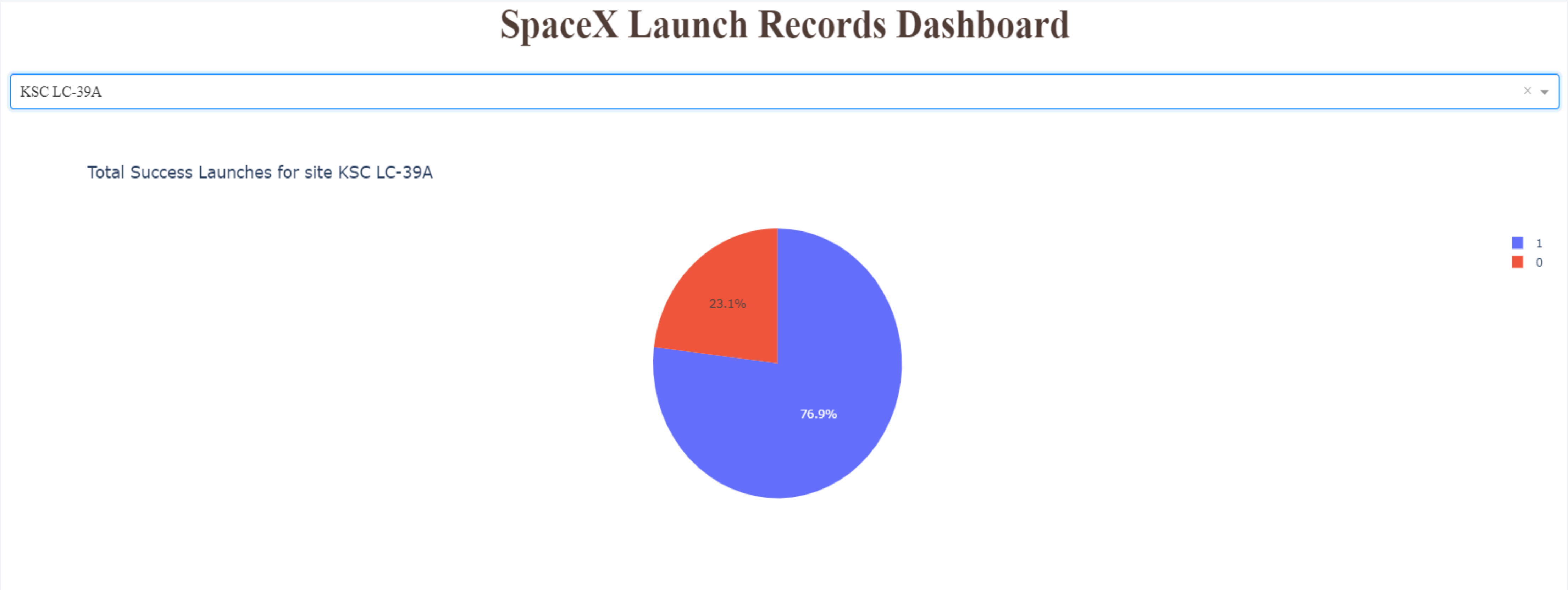
Share of Success for All Launch Site

[NoteBook]



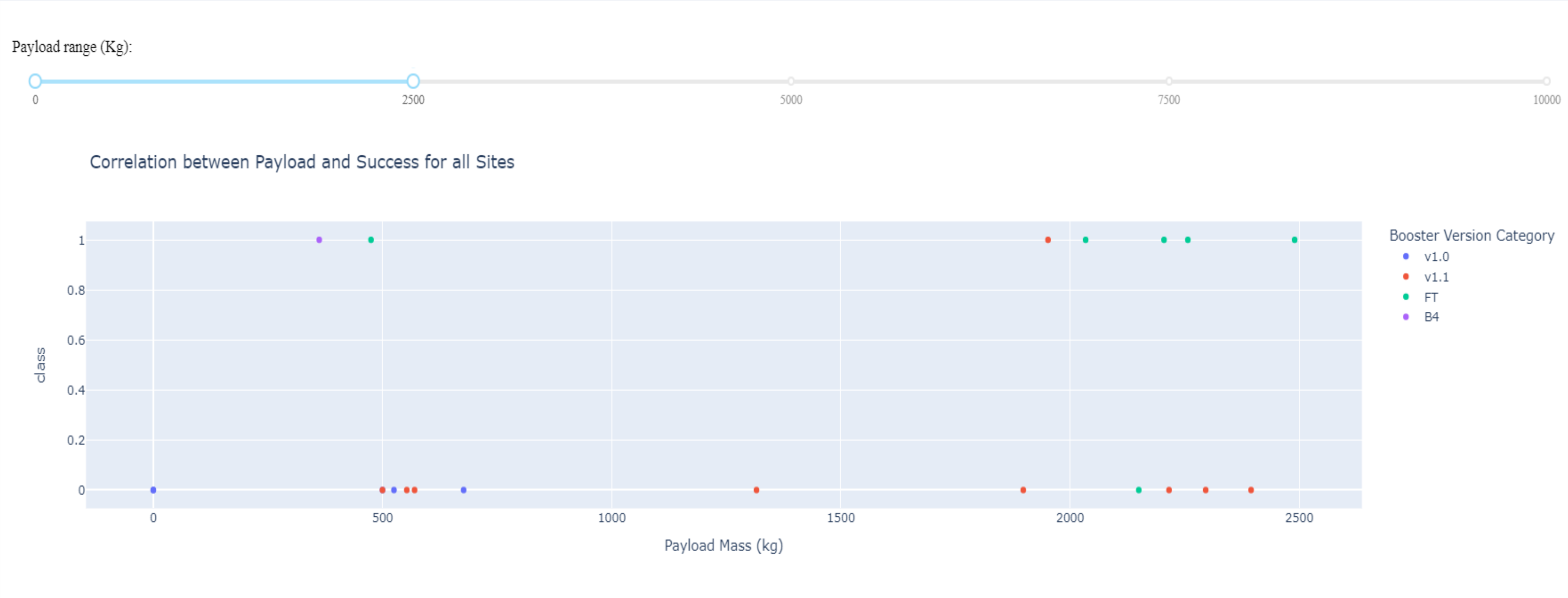
The top launch site with the highest success rate is KSC LC -39A, then comes CCAFS LC -40

Best Launch Site Success Rate



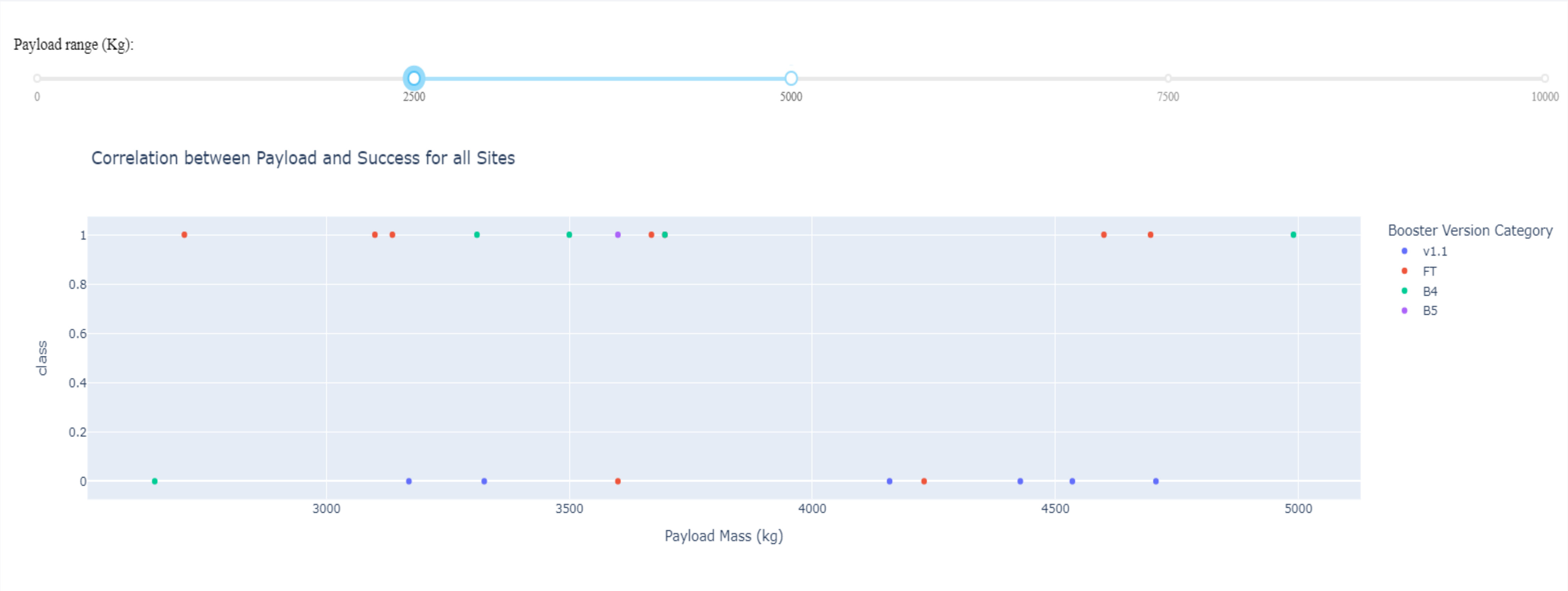
The rate of success of the best launch site is 76.9%

Scatter Plot: Payload (0-2500) vs. Launch Site



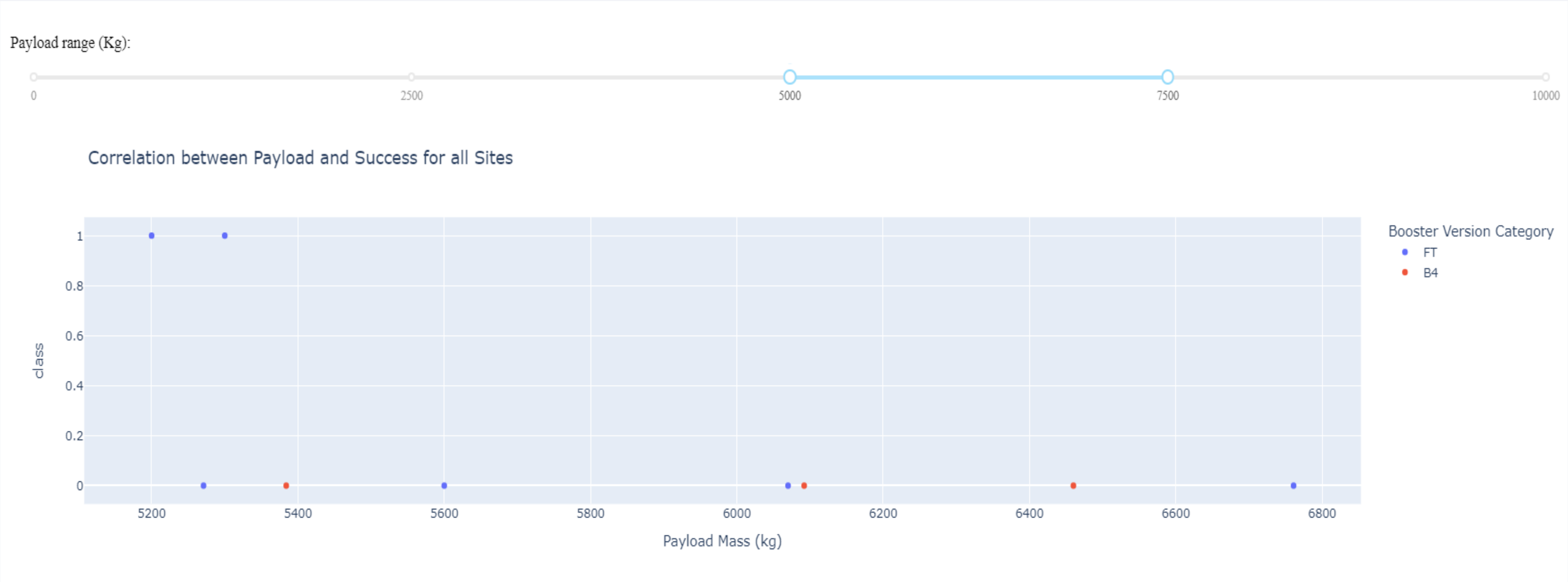
The range of payloads between **0** and **2500** has more failure than success.

Scatter Plot: Payload (2500-5000) vs. Launch Site



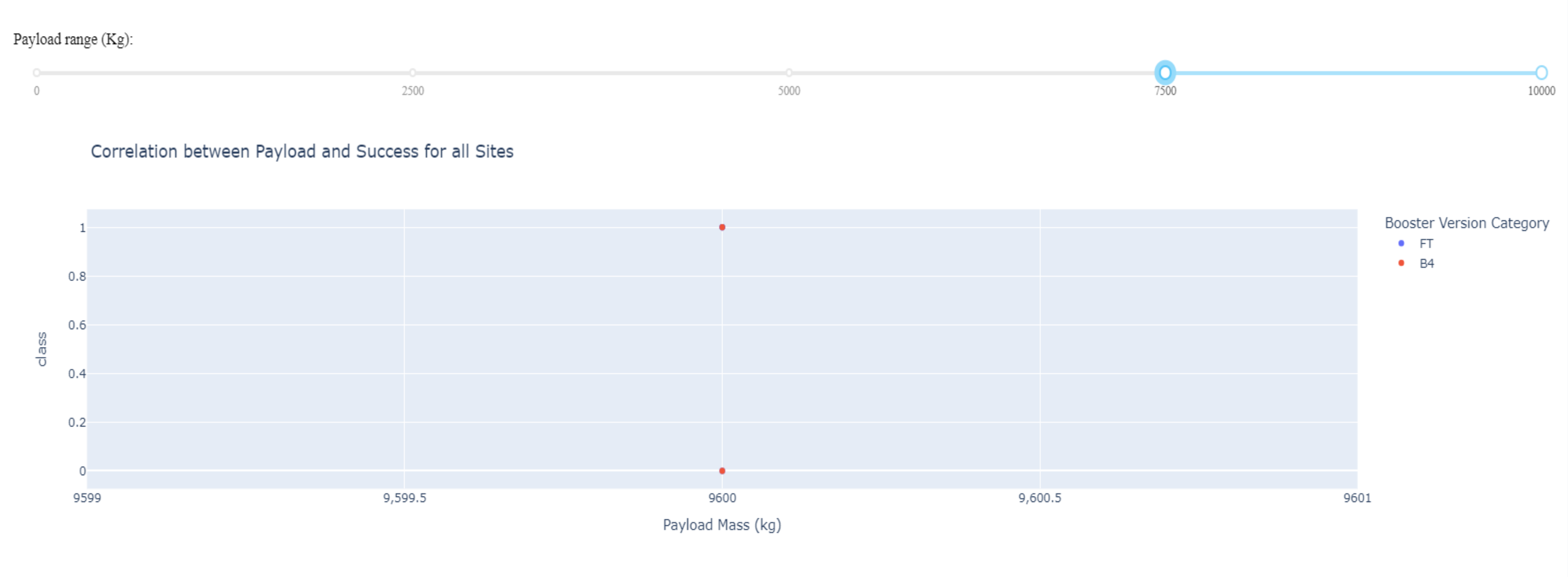
The range of payloads between **2500** and **5000** has the highest success.

Scatter Plot: Payload (5000-7500) vs. Launch Site



The range of payloads between **5000** and **7500** has the highest failure.

Scatter Plot: Payload (7500-1000) vs. Launch Site



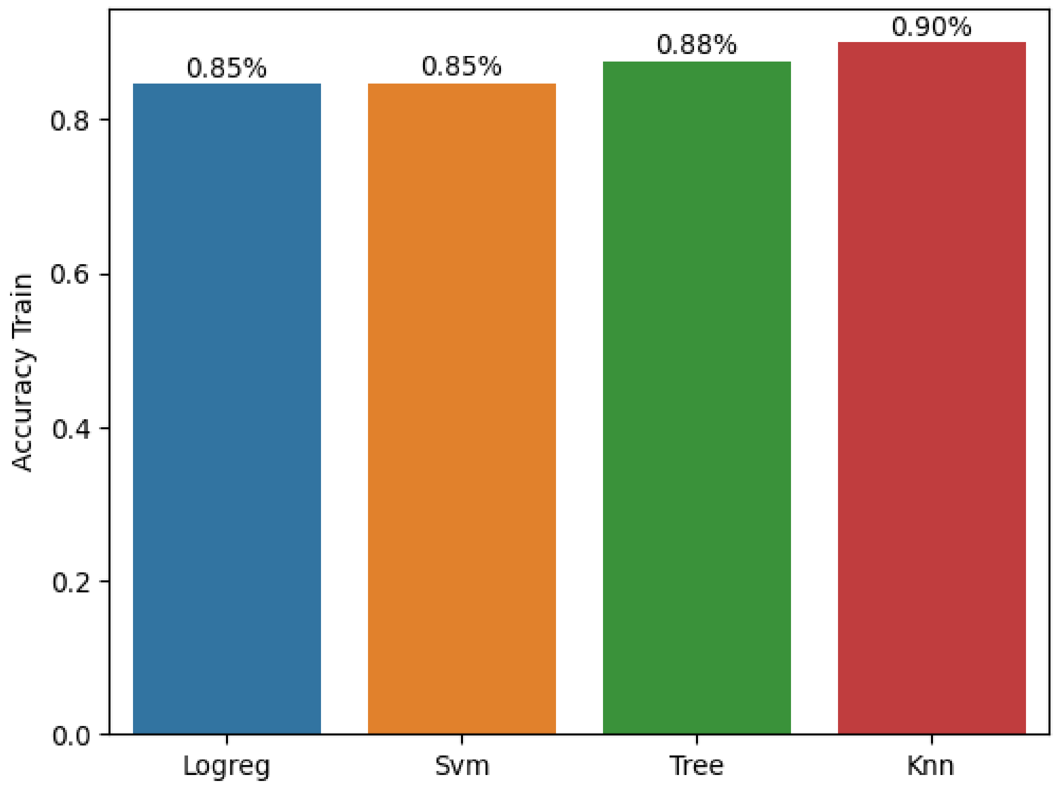
The range of payloads between **7500** and **10000** has a balance of success and failure

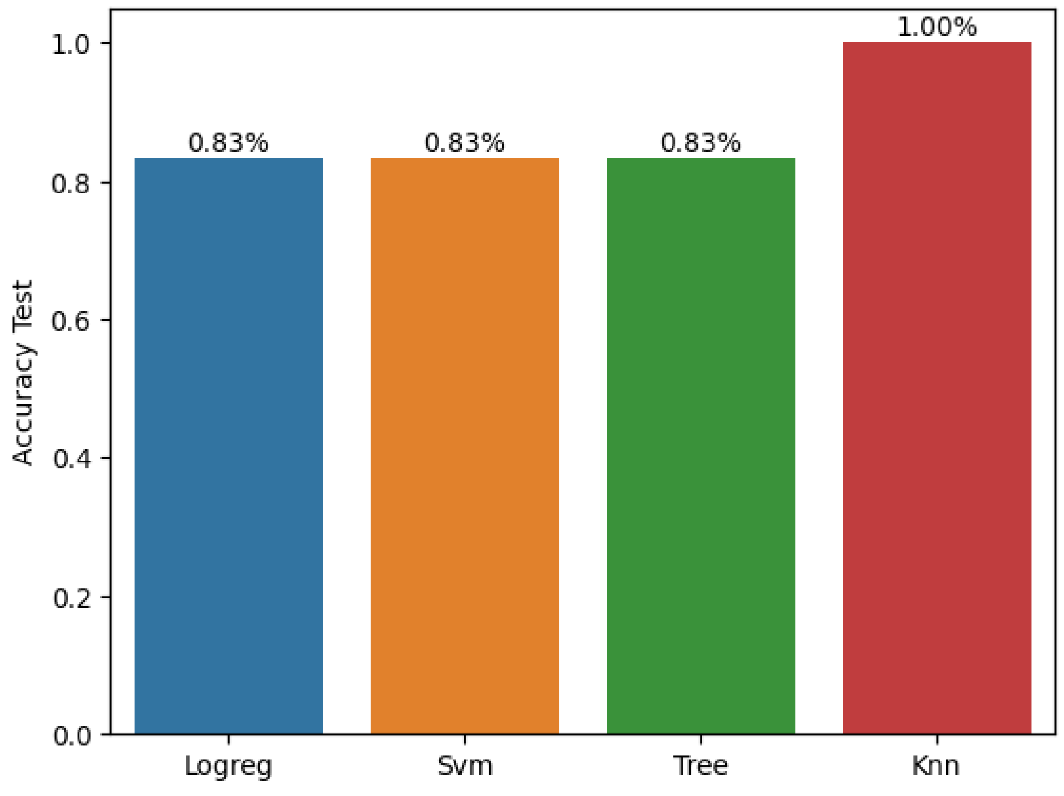
Section 5

Predictive Analysis (Classification)

Classification Accuracy

NoteBook





	Accuracy Train	Accuracy Test	Best Parameters
Logreg	0.846429	0.833333	{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
Svm	0.848214	0.833333	{'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
Tree	0.875000	0.833333	{'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'random'}
Knn	0.900000	1.000000	{'algorithm': 'auto', 'n_neighbors': 1, 'p': 1}

Classification Accuracy

- Three models have similar test accuracies (unseen data), with SVM, Decision Tree, and Logistic Regression achieving a test accuracy of 83%.
- The Logistic Regression and SVM have similar performances in training and test accuracies of 85%.
- The Decision Tree, has a training accuracy of 88%, considering the trade-off between the training and test accuracy, this indicates a potential risk of overfitting, and we can mitigate it through careful validation and testing on additional datasets.

Classification Accuracy

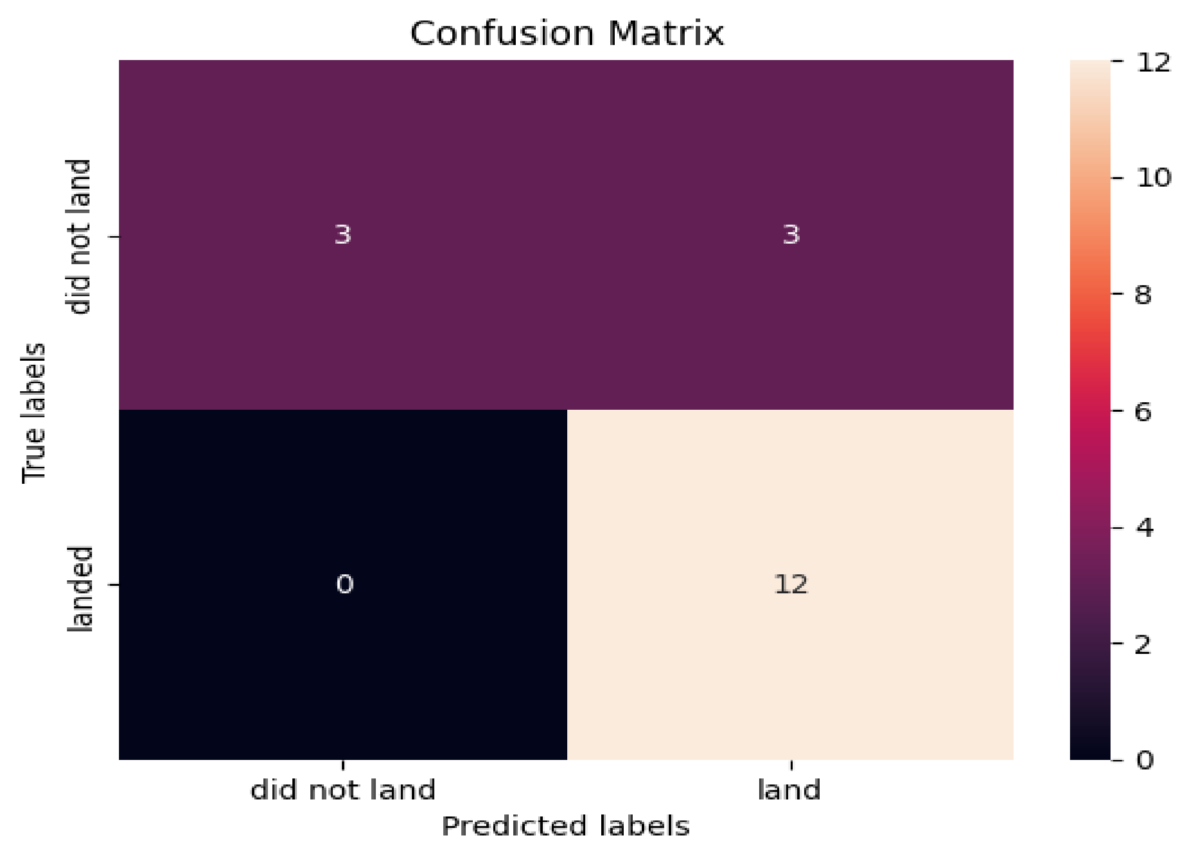
- The KNN model achieves a perfect test accuracy (on unseen data) of 100%.
- The KNN has the highest training accuracy of 90%, suggesting a good fit to the training data, but the trade-off between the test and the training accuracy is significantly large, considering the test accuracy is better than the training accuracy, which is unusual and suggests potential issues with the modeling process or the data, especially considering the small size of the dataset.
- When the model's superior performance on test data relative to training data suggests potential issues such as ***data leakage***, ***imbalance***, ***inappropriate model selection***, or ***random variability***.

Classification Accuracy

Considering the balance between performance, simplicity, and potential issues and overfitting or underfitting, the SVM model with the provided parameters ({'C': 1.0, 'gamma': 0.0316, 'kernel': 'sigmoid'}) could be considered the best choice for this dataset. SVM offers flexibility in handling non-linear relationships and can potentially generalize well to unseen data. However, further evaluation and validation may be necessary to confirm the model's performance. Then comes Logistic Regression, with the parameters {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}

Confusion Matrix

- The SVM classifier has correctly predicted 15 instances (TP + TN).
- There are 3 instances of misclassification, where the classifier predicted the opposite outcome compared to the actual outcome (FP + FN).
- The classifier shows a relatively good performance with a higher number of true positives and true negatives compared to false positives and false negatives. However, ***the number of false positives indicates room for improvement in the classifier's prediction of successful landings***.



Conclusions

- SVM, Decision Tree, and Logistic Regression achieve similar 83% test accuracies, with stable 85% performance in Logistic Regression and SVM.
- Decision Tree's 88% training accuracy raises concerns about overfitting, contrasting with KNN's perfect 100% test accuracy but a significant 10% gap with its 90% training accuracy.
- SVM and Logistic Regression, with specified parameters, is recommended for balanced performance and flexibility, pending further validation.
- Confusion matrix analysis underscores SVM's relatively good predictive performance but indicates room for improvement in predicting successful landings, warranting refinement.

Appendix

All the notebooks and the datasets for this project are available in my GitHub link below:

All notebooks and datasets of SpaceX Project

Thank you!