IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Majed Ghazi
2024/10/02

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context

The commercial space age has made significant advancements, enabling companies to make space travel more accessible. Companies such as Virgin Galactic, Rocket Lab, Blue Origin, and SpaceX are at the forefront of this development. SpaceX, in particular, has made remarkable strides with its affordable space missions. A key factor contributing to SpaceX's success is the ability to reuse the first stage of their Falcon 9 rockets, significantly lowering launch costs compared to competitors. This has led to a substantial reduction in costs per launch, as SpaceX's Falcon 9 costs about $62 million, compared to upwards of $165 million for other providers. Therefor we will be building a model that will predict if the Falcon 9 first stage will land successfully to determine the cost of a launch.

- Problems we want to answer

- What factors determine whether the first stage of SpaceX's Falcon 9 rockets will successfully land and be reused?
- How can machine learning models be used to predict the likelihood of successful first-stage recovery for future launches?
- How do the mission parameters, such as payload size, orbit destination, and customer requirements, affect the recovery of the first stage?
- What impact does successful or failed first-stage recovery have on the overall cost of a launch?
- How can Space Y use this information to optimize launch costs and compete effectively with SpaceX?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

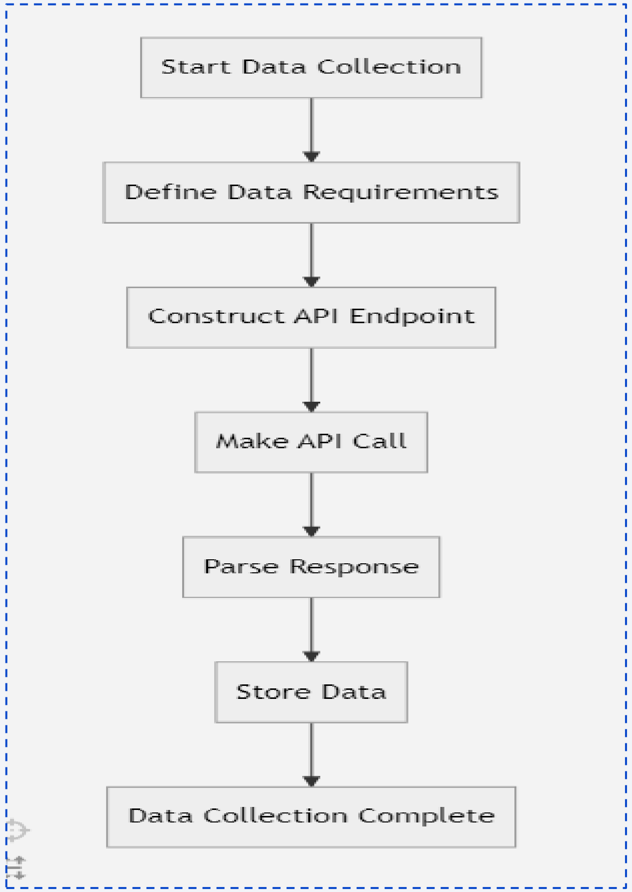
- Describe how data sets were collected.
- You need to present your data collection process use key phrases and flowcharts

Data Collection

- Data was collected from the SpaceX API and conducted web scraping to gather information regarding Falcon 9 launches. This dual approach allowed us to obtain the same dataset through different methods, enhancing the robustness of our data collection process.

Data Collection – SpaceX API

- Define the Data Requirements:
  - Identify the specific data needed (e.g., launch dates, payload masses, landing outcomes).
- Construct the API Endpoint:
  - Use the appropriate endpoint to access the desired data (e.g., /launches).
- Make the API Call:
  - Use the requests library to send an HTTP GET request to the SpaceX API.
  - Retrieve data in JSON format.
- Parse the Response:
  - Check for a successful response (status code 200).
  - Extract relevant information from the JSON response.
- Store the Data:
  - Convert the JSON data into a structured format (e.g., pandas DataFrame).
  - Save the data for further analysis.



[Click To Show Project]

Data Collection - Scraping

Identify

Identify Target Website:

- Determine the specific webpage(s) to scrape for the required data (e.g., Falcon 9 launch details).

Inspect

Inspect the Web Page:

- Use browser developer tools to examine the HTML structure.
- Identify the specific elements (tags, classes, IDs) that contain the data of interest.

Set Up

Set Up Environment:

- Install necessary libraries (e.g., requests, BeautifulSoup, pandas).

Send

Send HTTP Request:

- Use the requests library to send a GET request to the target URL.
- Retrieve the HTML content of the webpage.

Parse

Parse HTML Content:

- Use BeautifulSoup to parse the retrieved HTML.
- Locate and extract the relevant data elements.

Clean and Structure

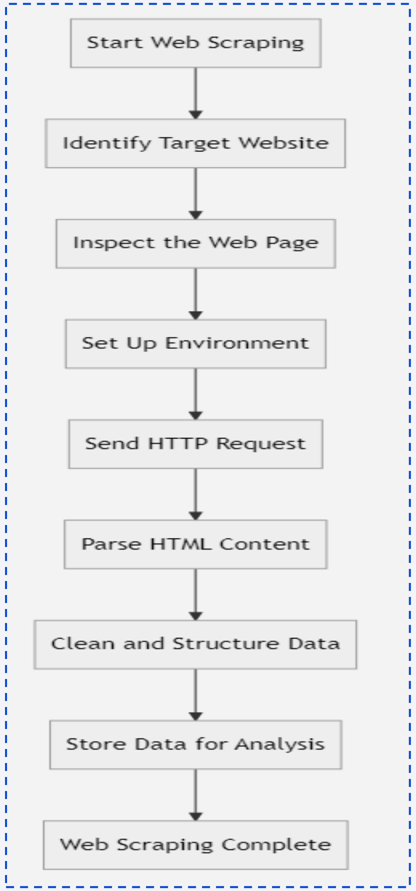
Clean and Structure Data:

- Process the extracted data to remove any unnecessary information.
- Store the cleaned data in a structured format (e.g., DataFrame).

Store

Store Data for Analysis:

- Save the structured data to a file (e.g., CSV) or database for further analysis.

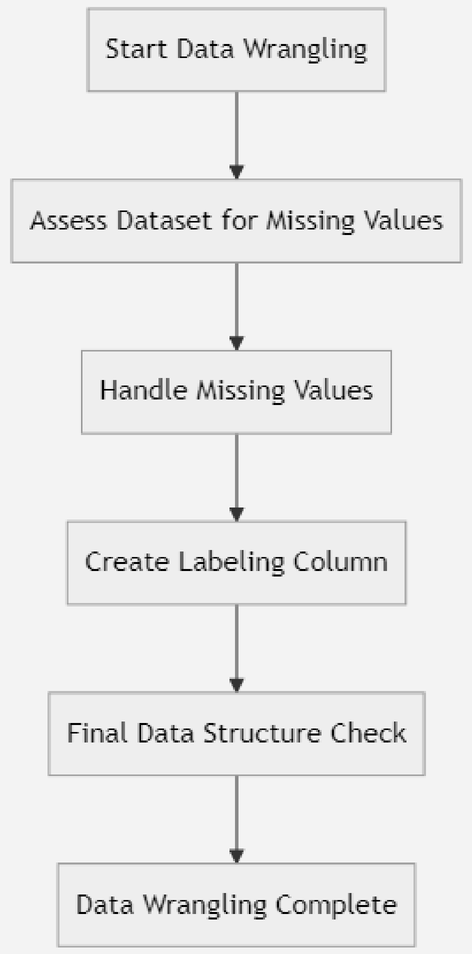


Click To Show Project

Data Wrangling

- Our primary objective was to identify patterns within the data and define labels for training supervised models. This process involves cleaning and transforming the dataset to make it suitable for analysis.

- **Assess Dataset for Missing Values**:
- Analyze the dataset to identify any missing data.
- Use functions to check for null values in the columns.
- **Handle Missing Values**:
- Discover that only one column contained missing values.
- Replace missing values with the mean of the respective column to maintain data integrity.
- **Create Labeling Column**:
- Generate a new column to facilitate analysis.
- Assign values for landing outcomes:
    - Unsuccessful landings = 0
    - Successful landings = 1
- **Final Data Structure Check**:
- Review the modified dataset to ensure accuracy and completeness.
- Verify that the new labeling column has been added correctly.



Click To Show Project

EDA with Data Visualization

- **Visualization Summaries** We visualized our findings through several types of charts:
- **Catplot:** This was used to illustrate the relationship between flight numbers and launch sites.
- **Scatterplot:** This helped us visualize relationships between various columns in our dataset.
- **Bar Chart:** This was employed to depict relationships among categorical variables.
- **Line Chart:** We utilized this to visualize the trend of launch success rates over the years.

Click To Show Project

EDA with SQL

•**Query 1:** Displayed the names of the unique launch sites in the space mission.
•**Query 2:** Retrieved 5 records where launch sites begin with the string 'CCA'.
•**Query 3:** Calculated the total payload mass carried by boosters launched by NASA (CRS).
•**Query 4:** Determined the average payload mass carried by booster version F9 v1.1.
•**Query 5:** Identified the date when the first successful landing outcome on the ground pad was achieved.
•**Query 6:** Listed the names of boosters that successfully landed on a drone ship with a payload mass between 4000 kg and 6000 kg.
•**Query 7:** Counted the total number of successful and failed mission outcomes.
•**Query 8:** Retrieved the names of booster versions that carried the maximum payload mass using a subquery.
•**Query 9:** Listed records displaying the month names, failed landing outcomes in drone ships, booster versions, and launch sites for the year 2015.
•**Query 10:** Ranked the count of landing outcomes (e.g., Failure (drone ship), Success (ground pad)) between the dates 2010-06-04 and 2017-03-20, in descending order.

Click To Show Project

Build an Interactive Map with Folium

- **Markers**:
  - **Purpose**: Indicated the exact location of each launch site on the map.
  - **Explanation**: Markers provide a visual representation of launch site locations, making it easy for users to identify and distinguish between different sites.
- **Circles**:
  - **Purpose**: Defined the boundaries of each launch site.
  - **Explanation**: Circles help visualize the operational area of each site, offering insights into the spatial extent and operational limits of each launch facility.
- **Launch Outcomes (Color-Coded Markers)**:
  - **Purpose**: Differentiated between successful and failed launches for each site.
  - **Explanation**: Using different colors for markers based on launch outcomes allows users to quickly assess the performance of each site at a glance, facilitating performance comparisons.
- **Proximity Analysis (Lines)**:
  - **Purpose**: Analyzed the proximity of each launch site to transportation infrastructure (railways, highways, cities).
  - **Explanation**: By incorporating lines to indicate distances to nearby infrastructure, users can evaluate how well-connected each launch site is, aiding in understanding potential logistical advantages or challenges.

Click To Show Project

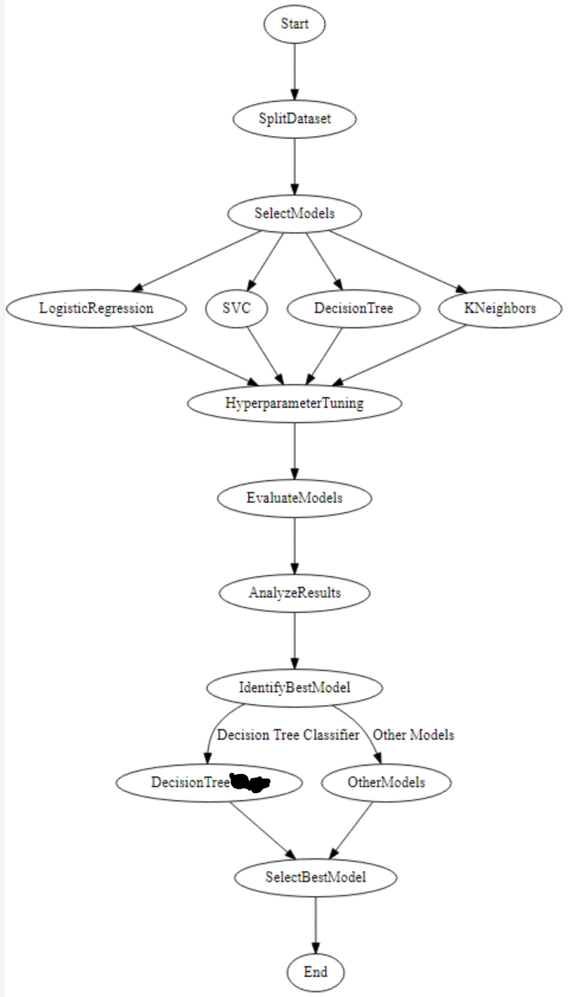
Build a Dashboard with Plotly Dash

- **Plots/Graphs Added:**
- **Pie Chart**:
  - **Purpose**: Visualize the success rates of launch sites.
  - **Explanation**: The pie chart provides a clear and intuitive way for users to see the proportion of successful vs. failed launches across different sites. This visual representation allows users to quickly grasp which sites have higher success rates, enabling comparisons at a glance.
- **Scatter Plot**:
  - **Purpose**: Examine the correlation between payload mass and launch success.
  - **Explanation**: The scatter plot displays payload mass on the x-axis and launch outcomes (i.e., the class column) on the y-axis. This allows users to visually observe how payload may correlate with mission outcomes for selected site(s). Additionally, each scatter point is color-labeled according to the booster version used for the launch, enabling users to differentiate between mission outcomes across different boosters. This visualization facilitates a deeper understanding of how varying payloads and booster types impact the success of launches, helping to identify trends and patterns within the data.
- **Interactions Added:**
- **Dropdown Component**:
  - **Functionality**: Allows users to select either all launch sites or a specific site for analysis.
  - **Explanation**: This interaction enables users to customize their view based on their interests. By selecting a specific site, users can focus on detailed metrics and success rates relevant to that location, enhancing the user experience and making the dashboard more versatile.
- **Payload Range Selector**:
  - **Functionality**: Enables users to filter results based on specific payload mass ranges.
  - **Explanation**: This filter allows users to analyze launches within a specified payload range, which can be crucial for understanding trends or anomalies in success rates for specific types of missions. It enhances the interactivity of the dashboard and allows for more tailored insights based on user-defined criteria.

Click To Show Project

Predictive Analysis (Classification)

- **Model Development Process**
- **Data Splitting**:
  - Split the dataset into training and testing sets to prepare for model training and evaluation.
- **Model Selection**:
  - Employed the following classification models:
    - **Logistic Regression**
    - **Support Vector Classifier (SVC)**
    - **Decision Tree Classifier**
    - **K-Neighbors Classifier**
- **Hyperparameter Tuning**:
  - Utilized **GridSearchCV** to optimize model performance by tuning hyperparameters for each model.
- **Model Evaluation**:
  - Evaluated model predictions using a **confusion matrix** to assess the accuracy and effectiveness of each model.
- **Performance Analysis**:
  - Analyzed the findings:
    - **Decision Tree Classifier**: Identified as the best-performing model.
    - Other models (Logistic Regression, SVC, K-Neighbors) exhibited comparable accuracy levels.
- **Best Performing Model Selection**:
  - Selected the best-performing model based on accuracy metrics and evaluation results.



[Click To Show Project]

Results

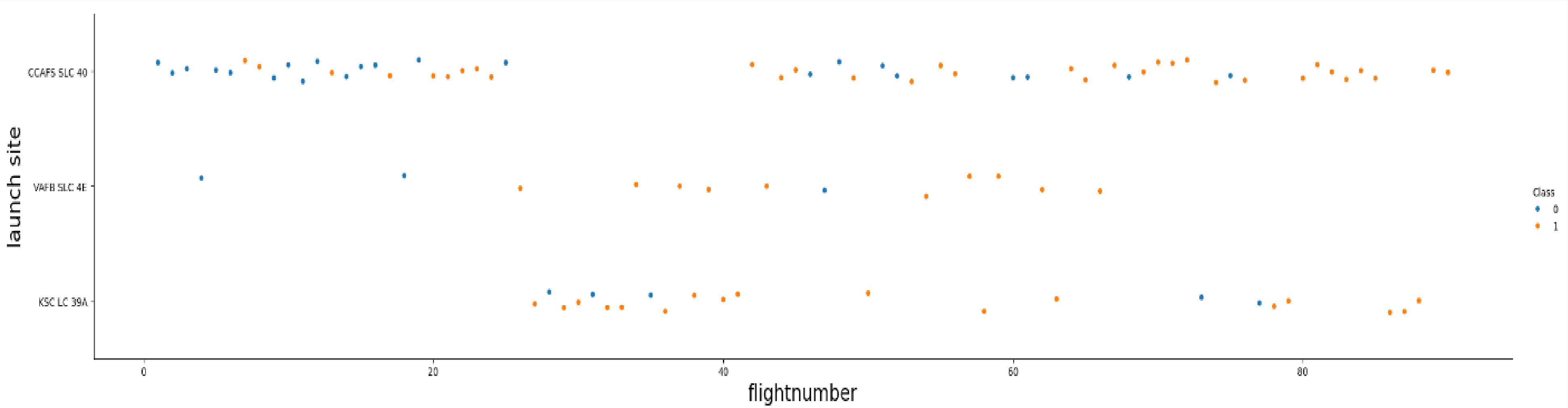
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- CCAFS SLC 40 has the most flights that is successful

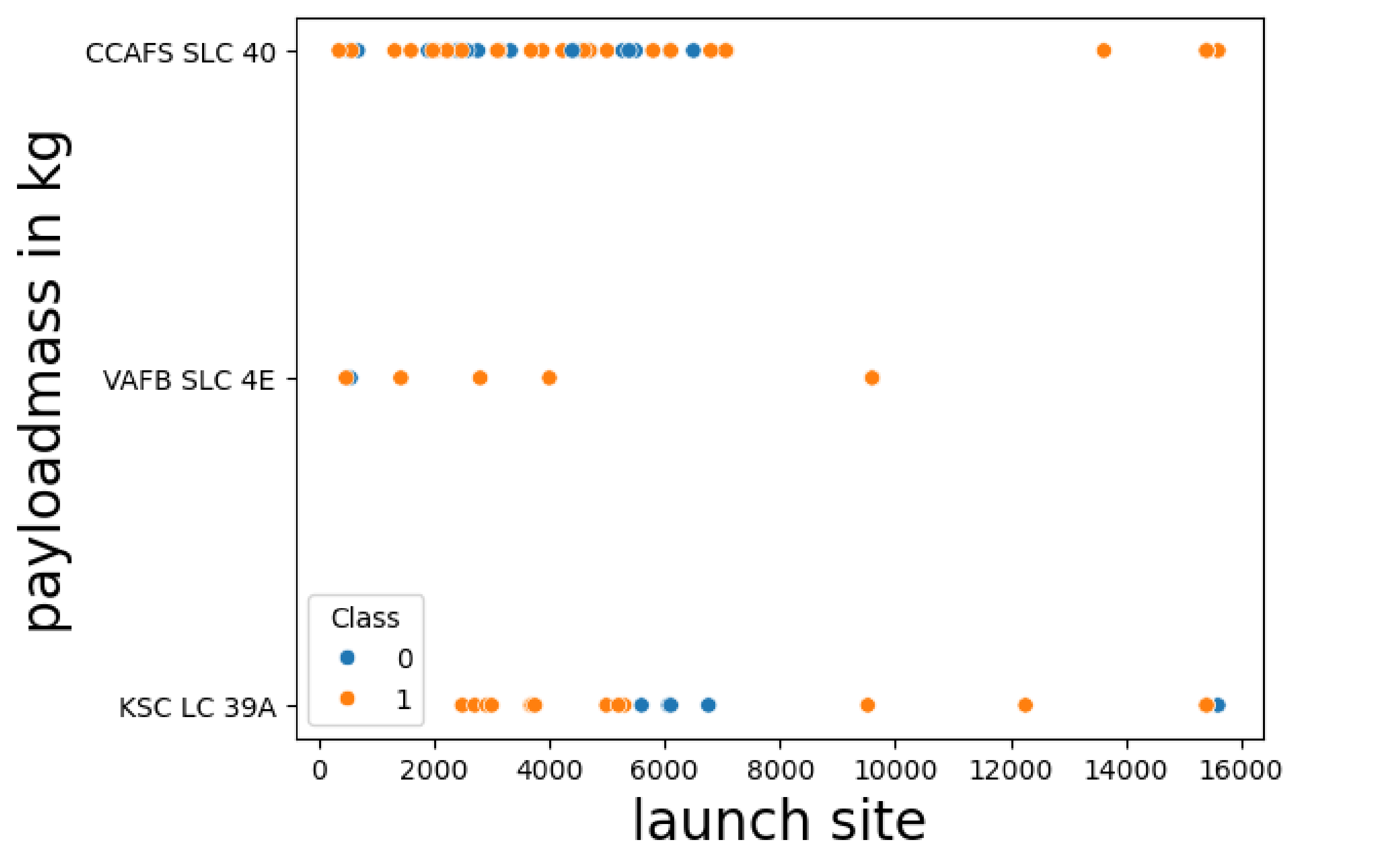


[Click To Show Project]

Payload vs. Launch Site

- VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

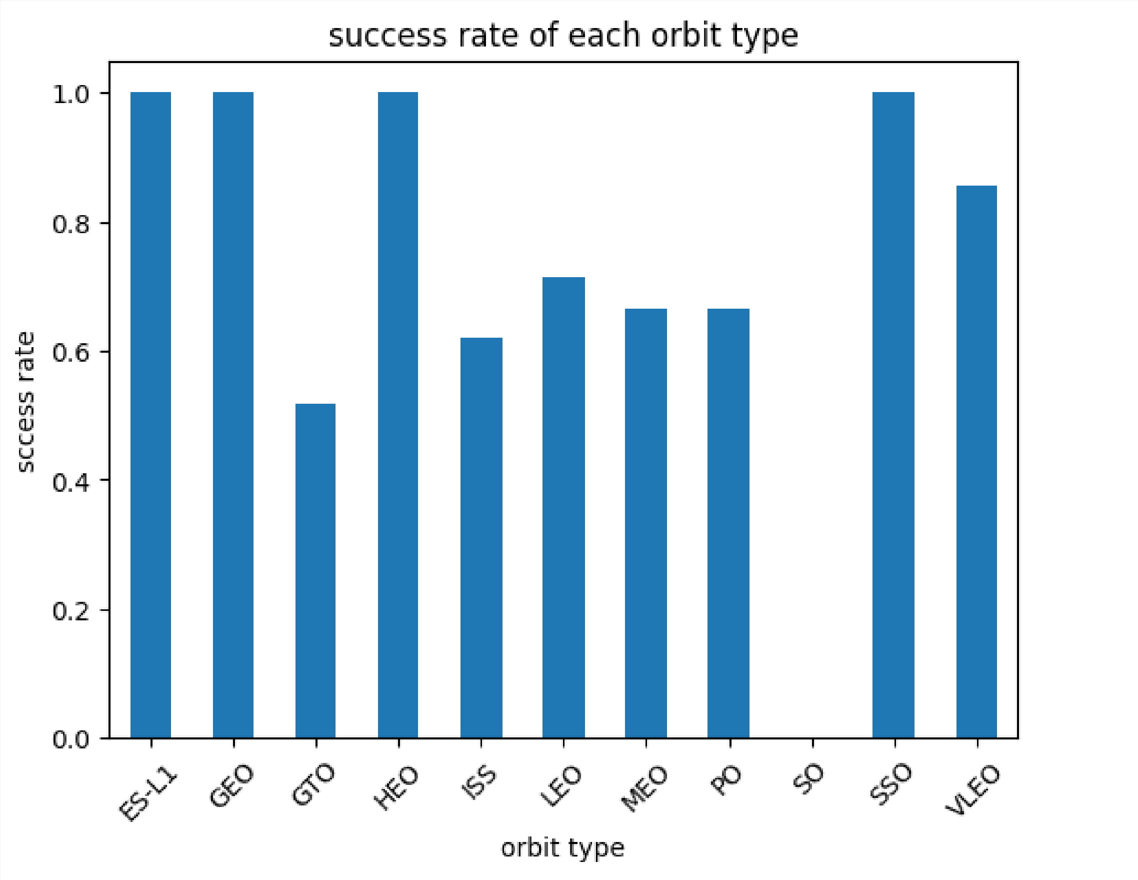
[Click To Show Project]



Success Rate vs. Orbit Type

- ES-L1,GEO,HEO,SSO Has the highest success rate.

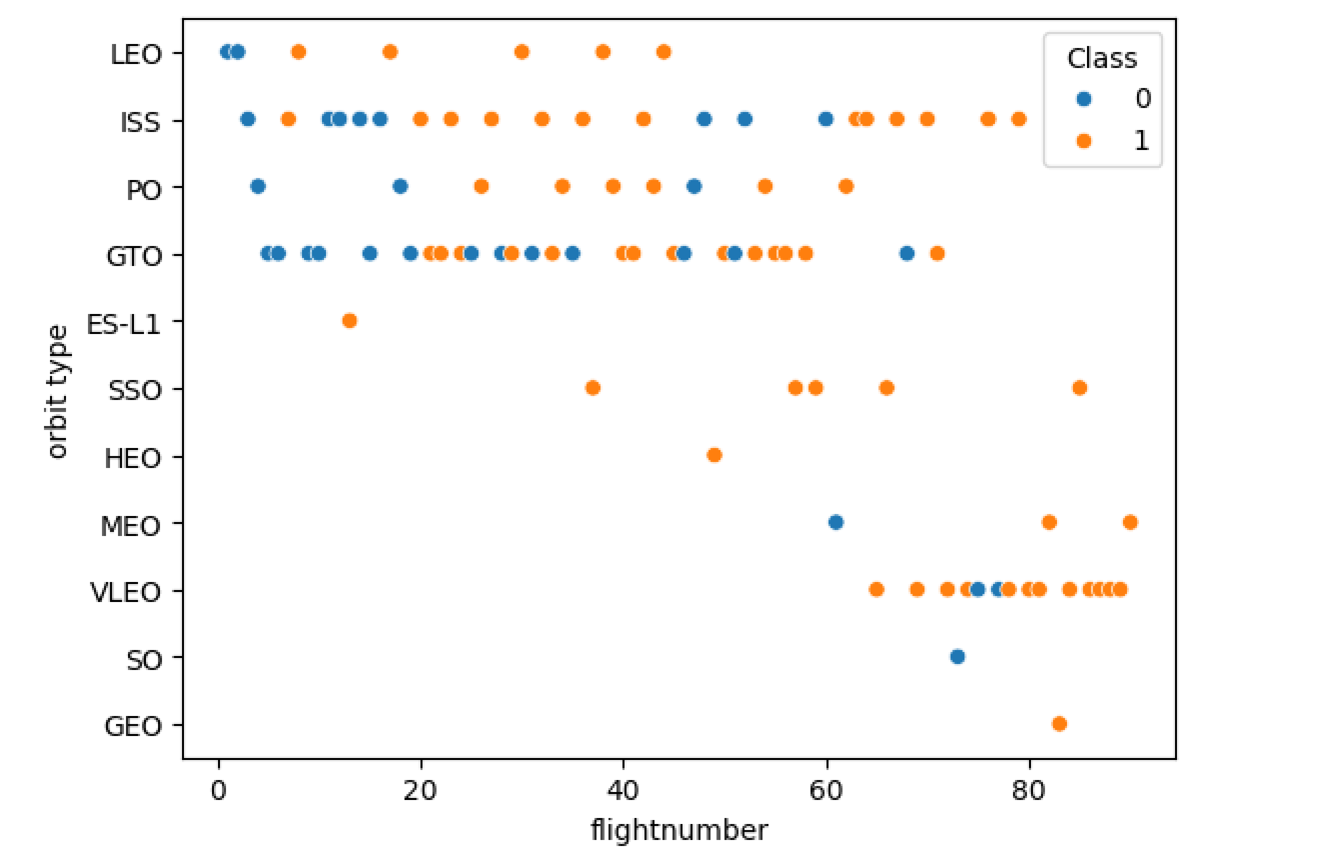
Click To Show Project



Flight Number vs. Orbit Type

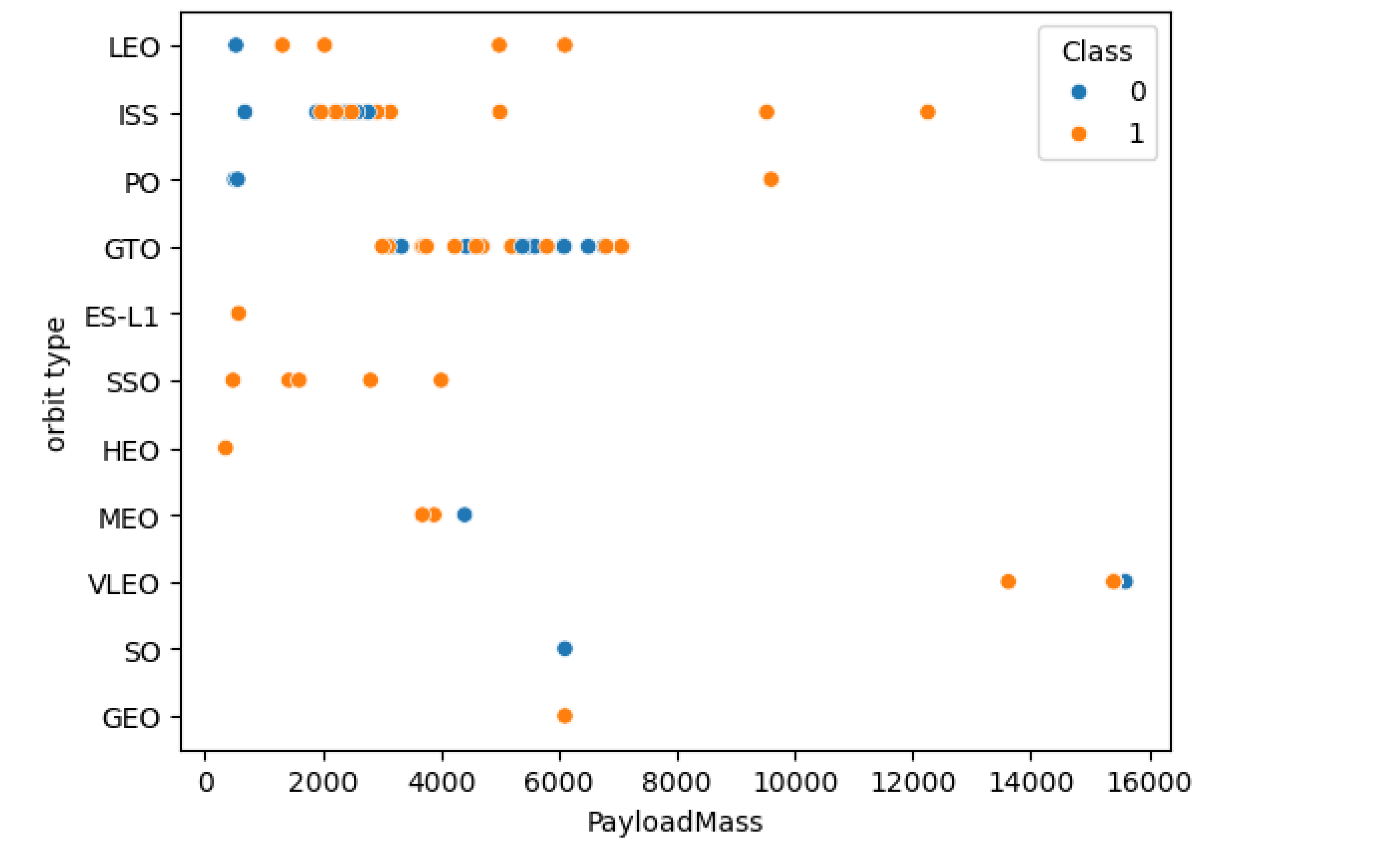
- LEO orbit success is related to the number of flights GTO orbit has no relationship between flight number and success

Click To Show Project



Payload vs. Orbit Type

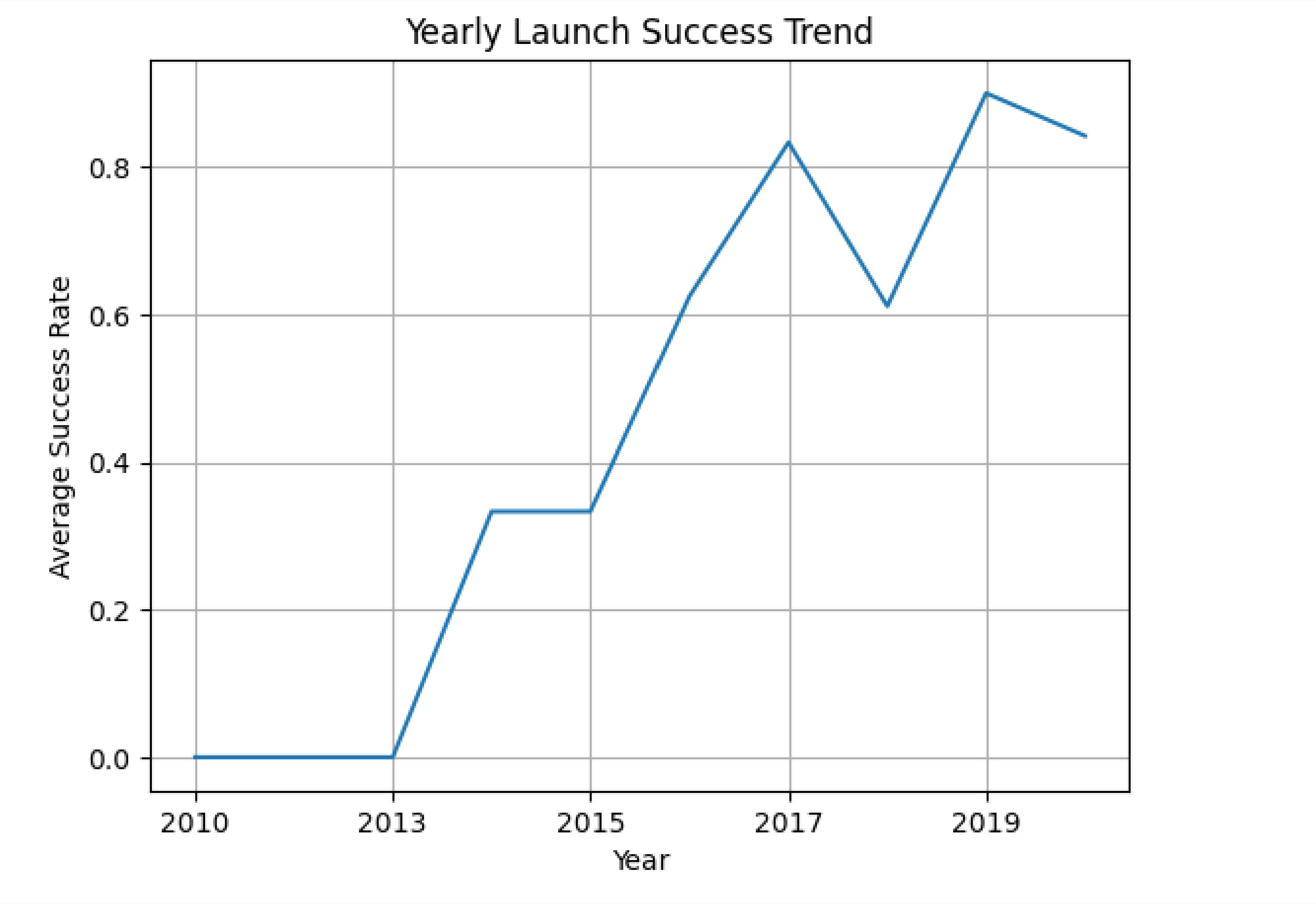
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.



[Click To Show Project]

Launch Success Yearly Trend

- sucess rate since 2013 kept increasing till 2020



Click To Show Project

All Launch Site Names

- Find the names of the unique launch sites

```
[11]: %sql select DISTINCT("Launch_Site") from SPACEXTABLE

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

[11]:

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

Launch Site Names Begin with 'CCA'

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

```
[12]: %sql select * from SPACEXTABLE WHERE "Launch_Site" Like "CCA%" limit 5
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
[13]: %sql SELECT SUM(PAYLOAD_MASS__KG_)FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';

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

[13]:

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

[14]:
```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where Booster_Version = 'F9 v1.1'
```

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

[14]:

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
[15]: %sql select min(date) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'

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

[15]:

min(date)
2015-12-22

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

```
[16]: #Failure (drone ship)
      %sql select Booster_Version from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000

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

[16]:

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
[17]: %sql select MISSION_OUTCOME,COUNT(*) from SPACEXTBL GROUP BY MISSION_OUTCOME

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

[17]:

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

Boosters Carried Maximum Payload

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

```
[18]: %sql select BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)

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

[18]:

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
[20]: %sql select substr(Date, 6,2) as Month, LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Failure (drone ship)' and substr(Date,0,5) = '2015';

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

[20]:

Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

```
[21]: %sql select LANDING_OUTCOME,count(*) as count from SPACEXTBL where Date between '2010-06-04' and '2017-03-20' group by LANDING_OUTCOME order by count DESC

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

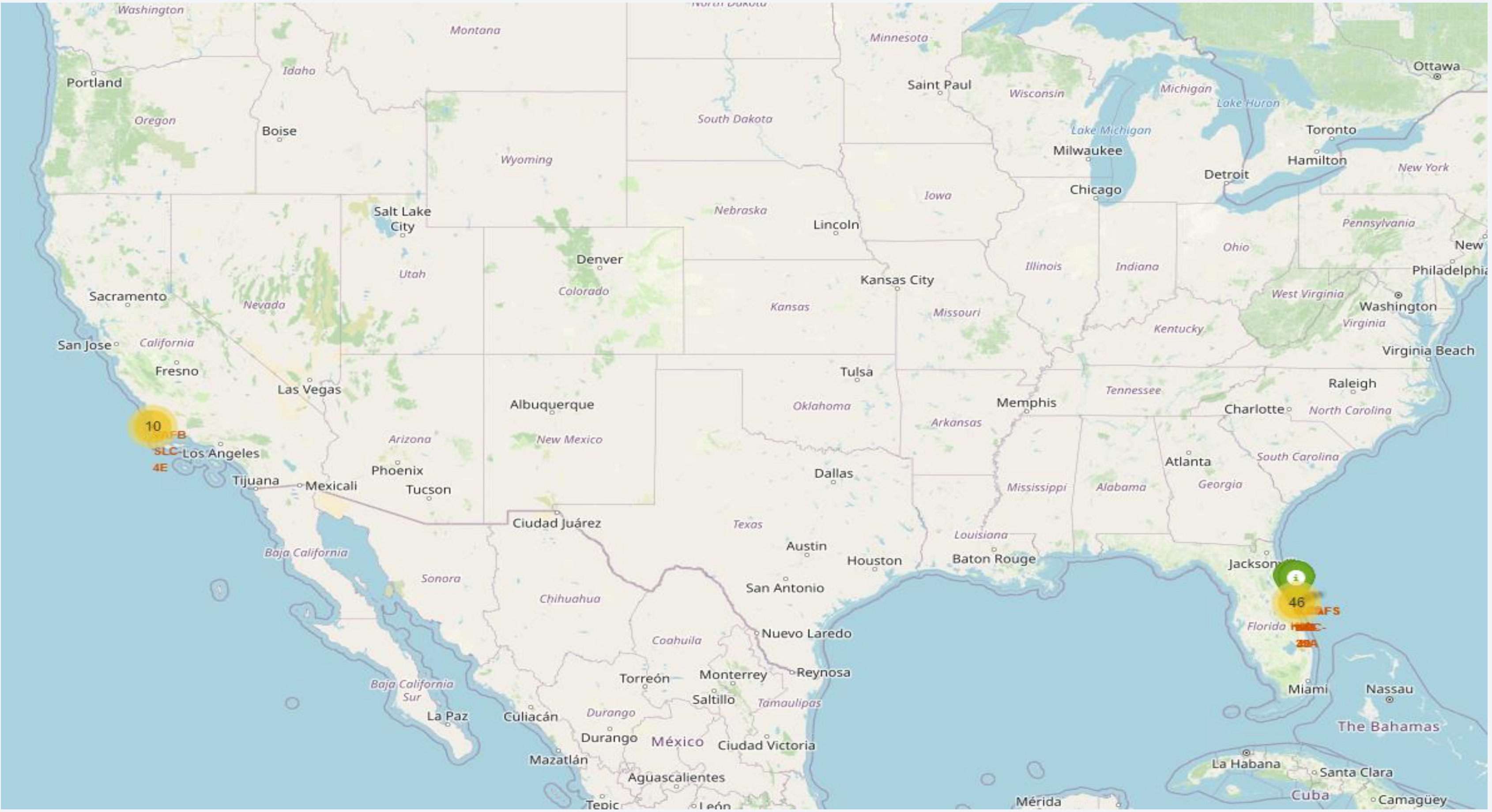
[21]:

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

Section 3

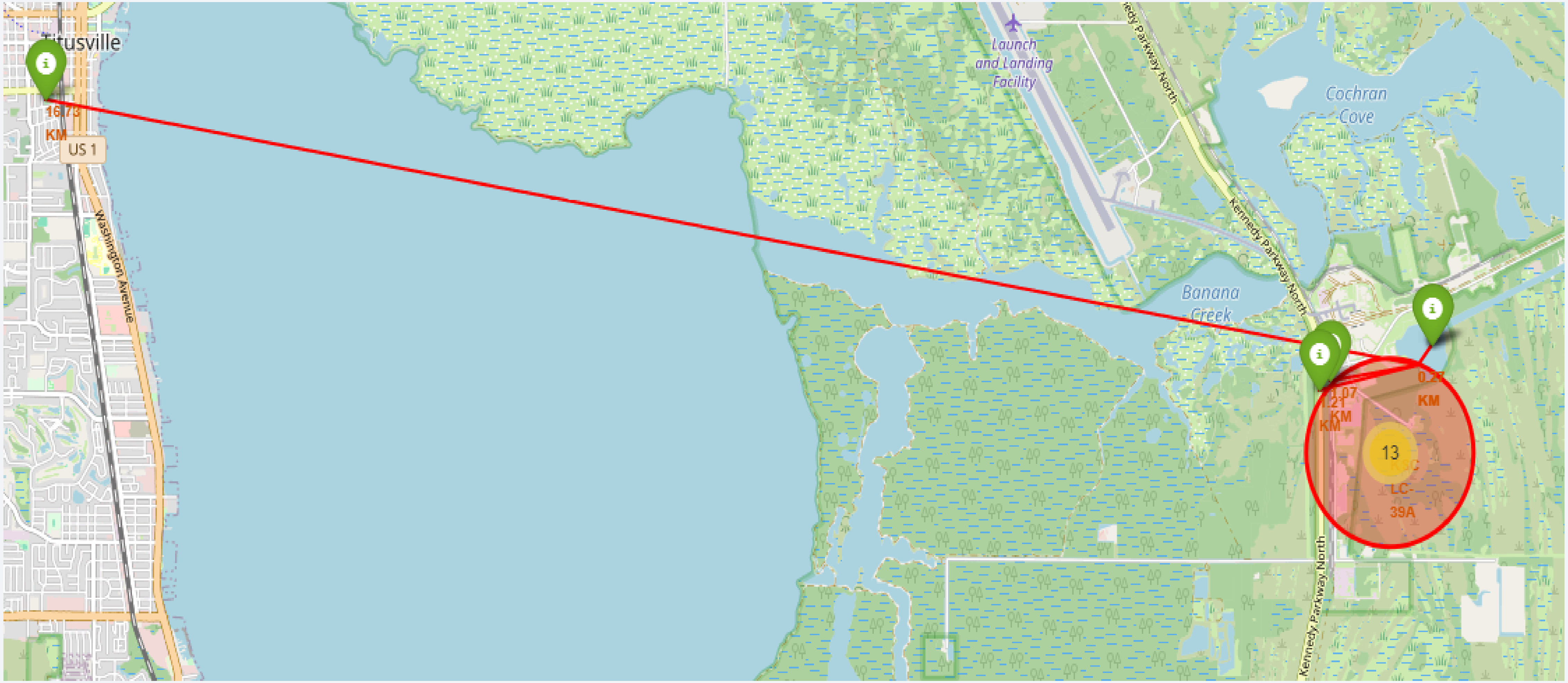
Launch Sites Proximities Analysis

All launch sites



Successful `launches`

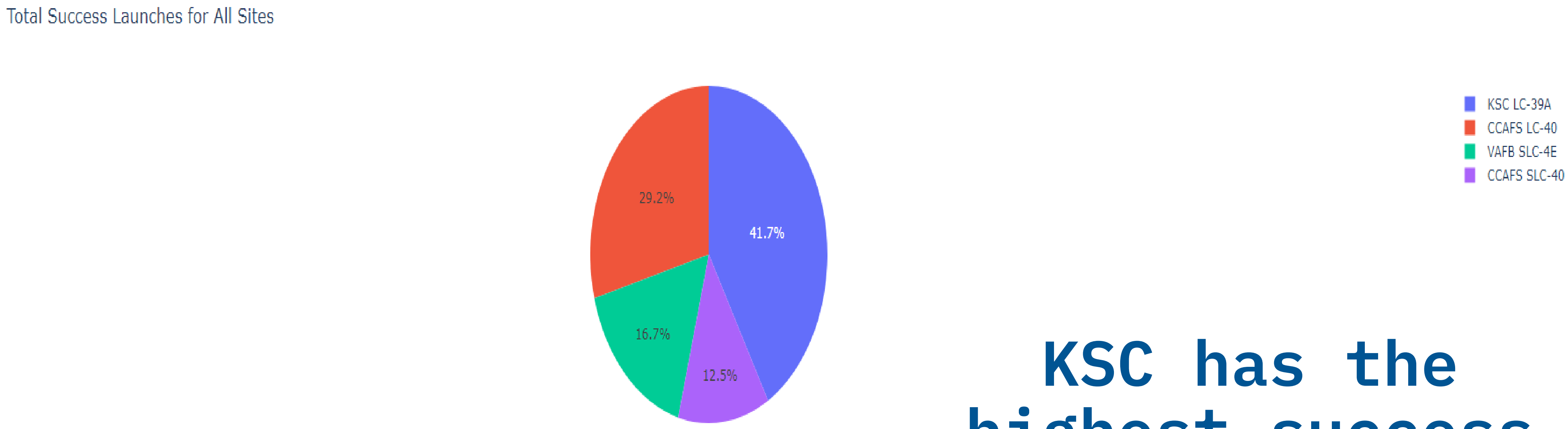
Proximities to site



Section 4

Build a Dashboard with Plotly Dash

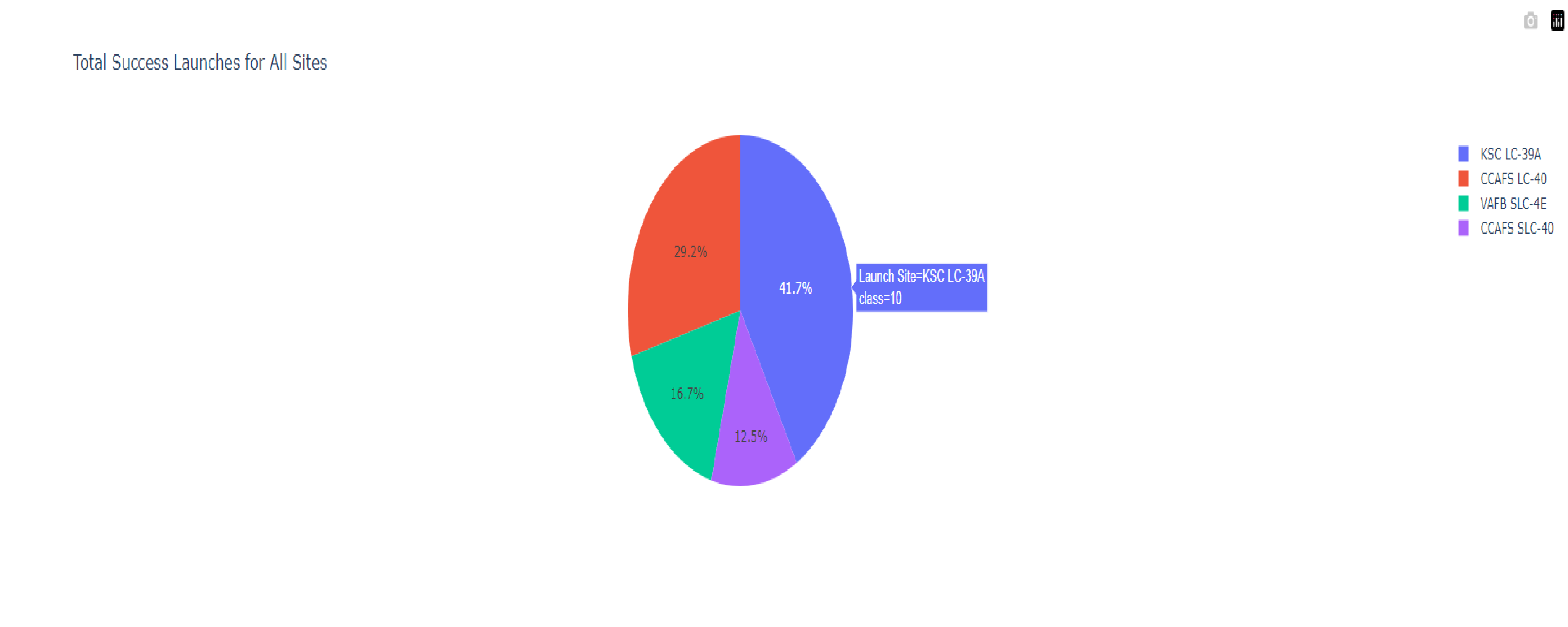
Launch success count for all sites



KSC has the highest success rate

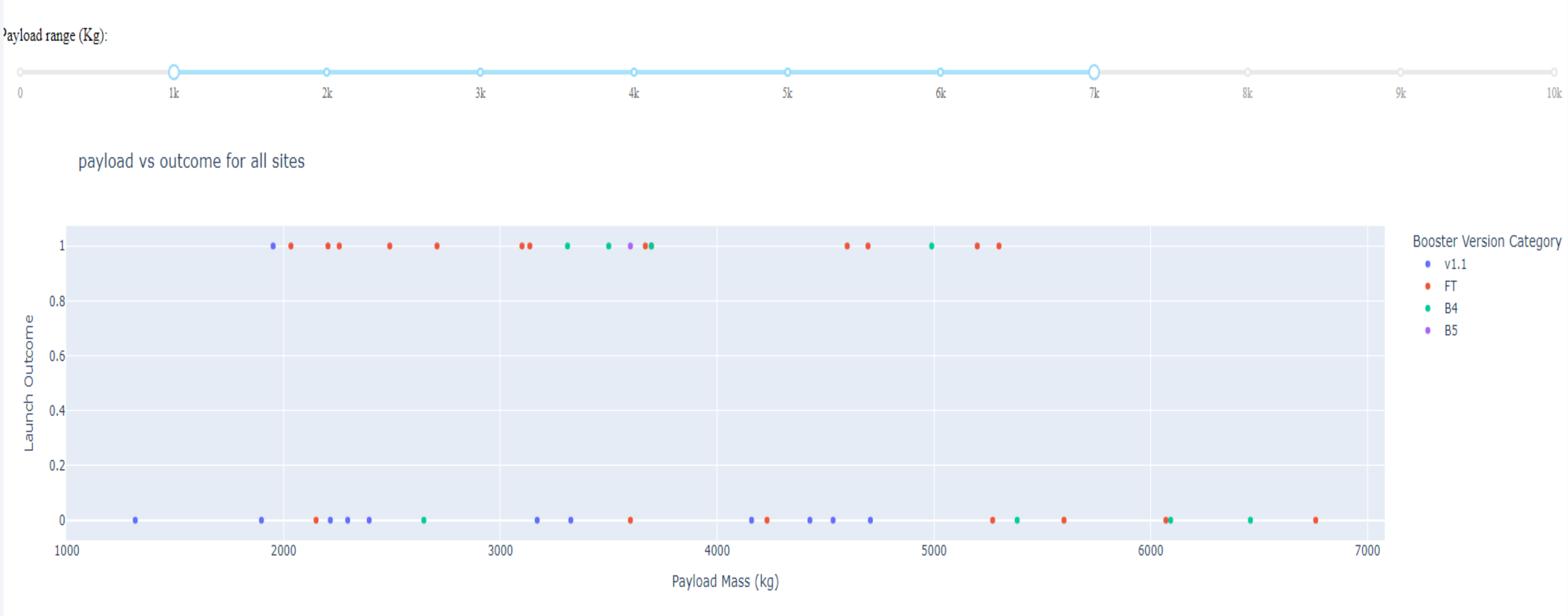
yload range (Kg):

Launch site with highest launch success



'ayload range (Kg):

Payload vs. Launch Outcome scatter plot

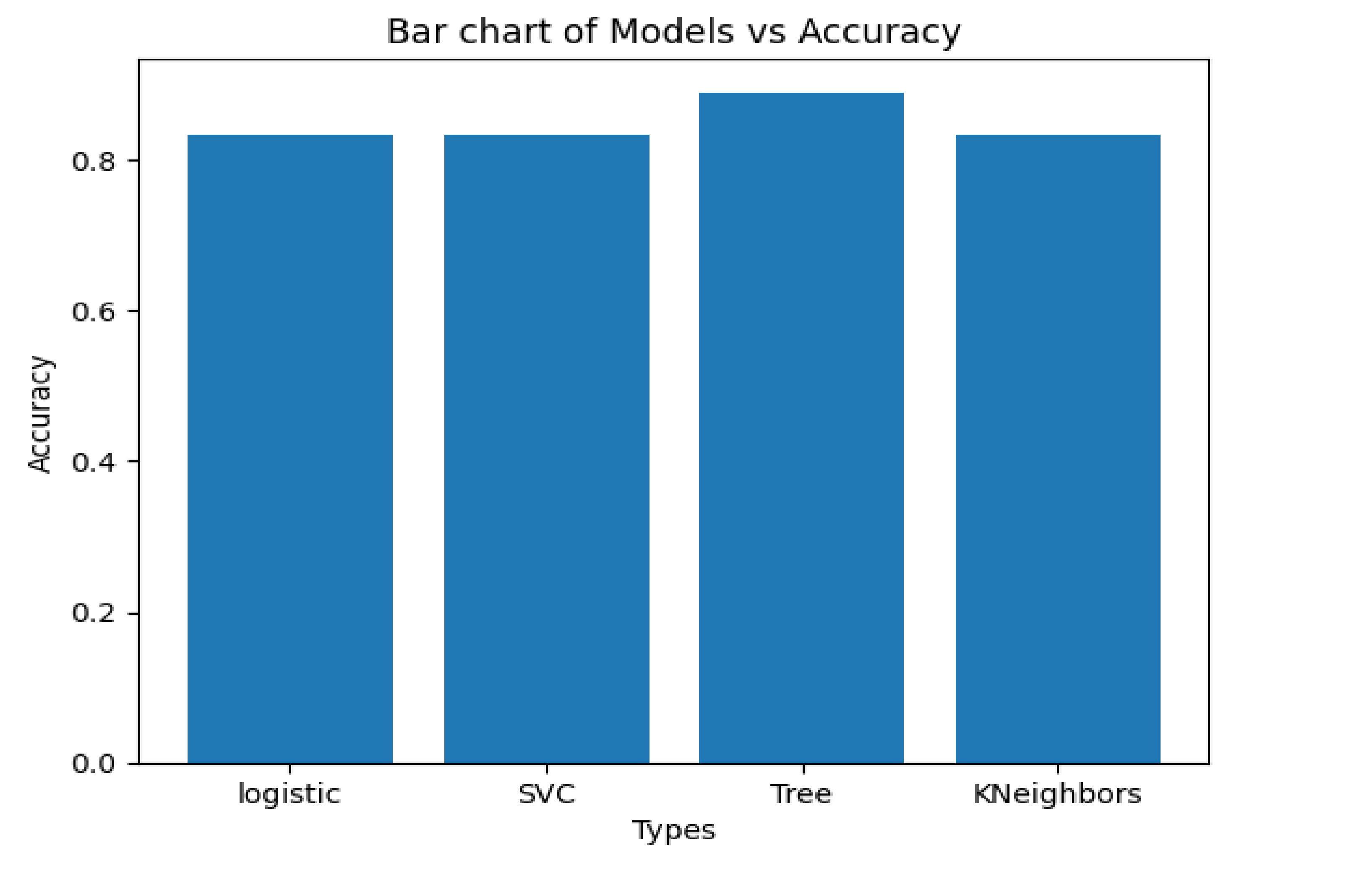


FT booster has the most successful launches

Section 5

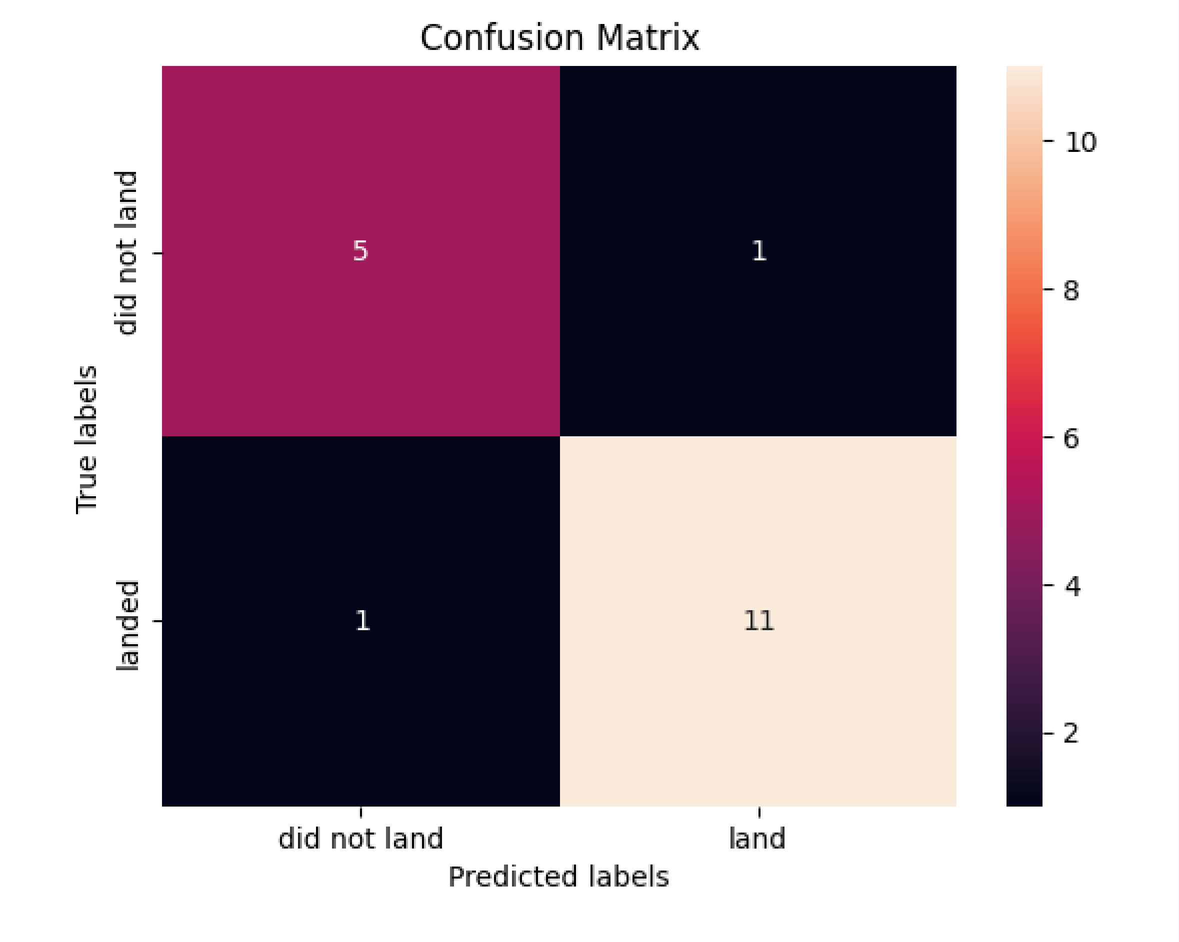
Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix

- The model shows 5 true predicted and 1 false for “did not land”
- The model shows 11 true predicted and 1 false for “landed”



Conclusions

- **Best Model:**
  - The **DecisionTreeClassifier** emerged as the best model for predicting whether Falcon 9 will land successfully or not.

- **Accuracy:**
  - This model demonstrated **high accuracy**, outperforming other models.

- **Confusion Matrix:**
  - The **confusion matrix** for DecisionTreeClassifier supported its strong performance, showcasing better classification results.

- **Other Models:**
  - Other models exhibited **similar accuracy** and had nearly identical confusion matrix results, but none outperformed the DecisionTreeClassifier.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!