IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Khairullah Ilyas

Outline

EXECUTIVE SUMMARY

INTRODUCTION

METHODOLOGY

RESULTS

CONCLUSION

APPENDIX

Executive Summary

Objective: Predicting SpaceX Falcon 9 Landing Success/Failure

1. Data Wrangling

- Import and parse SpaceX launch data
- Filter for Falcon 9 launches
- Address missing values

2. EDA and Feature Engineering

- Visualize key relationships:
  - Flight Number vs. Launch Site
  - Payload vs. Launch Site
  - Success rates for each orbit type
  - Flight Number vs. Orbit type
  - Payload vs. Orbit type
  - Yearly trend in launch success
- Feature Engineering:
  - Create dummy variables for categorical columns
  - Cast numeric columns to `float64`

3. Machine Learning Prediction

- Data Split: 80% Training Set, 20% Test Set (Random state: 2)
- Models:
  - Logistic Regression
  - Support Vector Machine
  - Decision Tree
  - k-Nearest Neighbor
- Train models to predict Falcon 9 landing success/failure

Based on the model that has been trained, the Decision Tree model with an F1 score of 0.88 was the fit model.

Introduction

In this project, we will predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Section 1

Methodology

Methodology

Data collection:

- The data was gotten from SpaceX API through get request.

Perform data wrangling

- The Exploratory Data Analysis (EDA) and Feature Engineering stage involve visualizing key relationships within the dataset. This includes examining the correlation between Flight Number and Launch Site, Payload and Launch Site, success rates for each orbit type, relationships between Flight Number and Orbit type, Payload and Orbit type, and exploring the yearly trend in launch success. Additionally, dummy variables are created for categorical columns, and all numeric columns are cast to **float64**.

Methodology

Perform exploratory data analysis (EDA) using visualization and SQL

Perform interactive visual analytics using Folium and Plotly Dash

Perform predictive analysis using classification models

- The dataset is split into train and test sets. Various machine learning models, such as Logistic Regression, Support Vector Machine, Decision Tree, and k-Nearest Neighbor, are employed to train the data and predict the success or failure of Falcon 9 rocket landings. This approach aims to provide insights into the factors influencing landing success and contribute to the enhancement of future launch outcomes.

Data Collection

Set up a client to make HTTP GET requests to the SpaceX API endpoint that provides launch data.

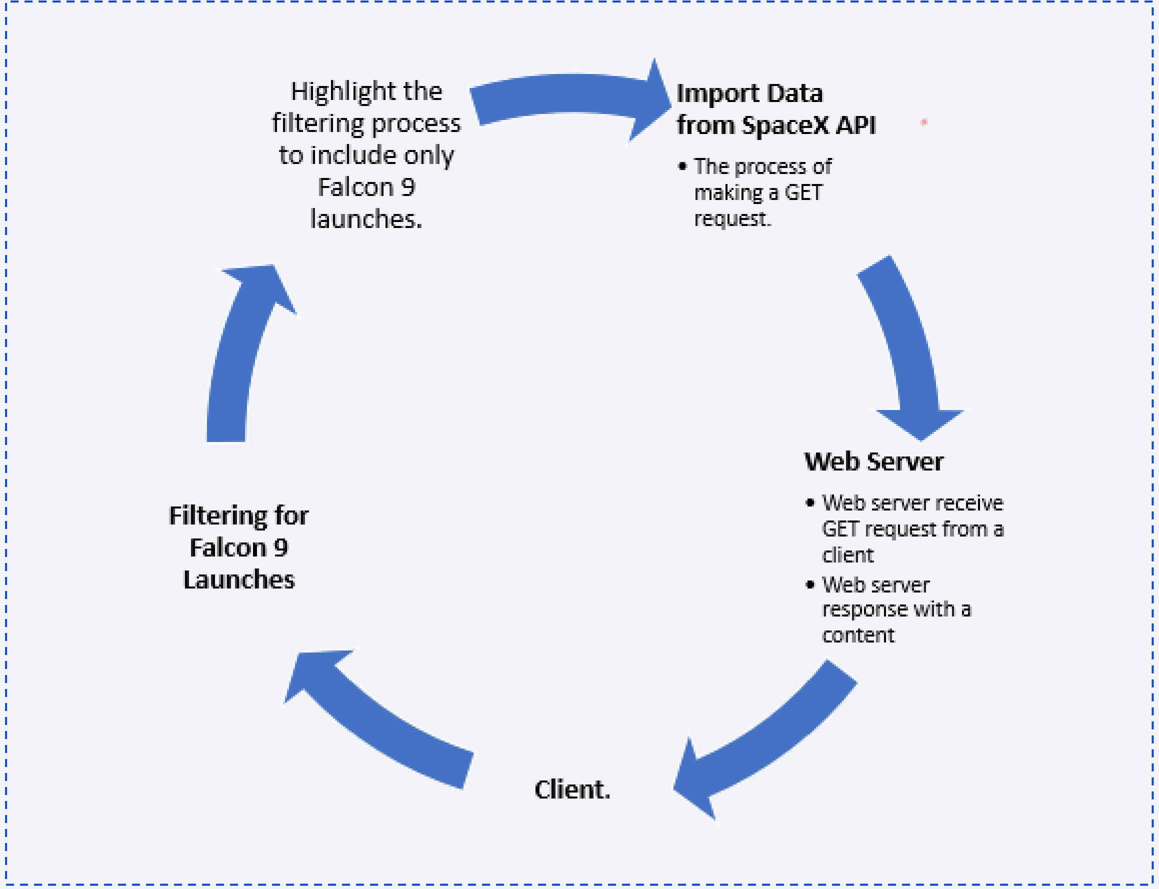
Send GET requests to retrieve the data: The SpaceX API usually returns data in JSON format, which is easy to consume programmatically.

Convert the JSON response into a structured format that is convenient for analysis, such as a pandas DataFrame if you are using Python.

Implement error handling to manage any issues that arise during the API interaction, such as network errors or API limits.

Confirm the success of the requests and capture the response data.

https://github.com/khairul746/Applied-Data-Science-Capstone-IBM/blob/main/1.%20jupyter-labs-spacex-data-collection-api.ipynb



Import Data from SpaceX API

- The process of making a GET request.

Web Server

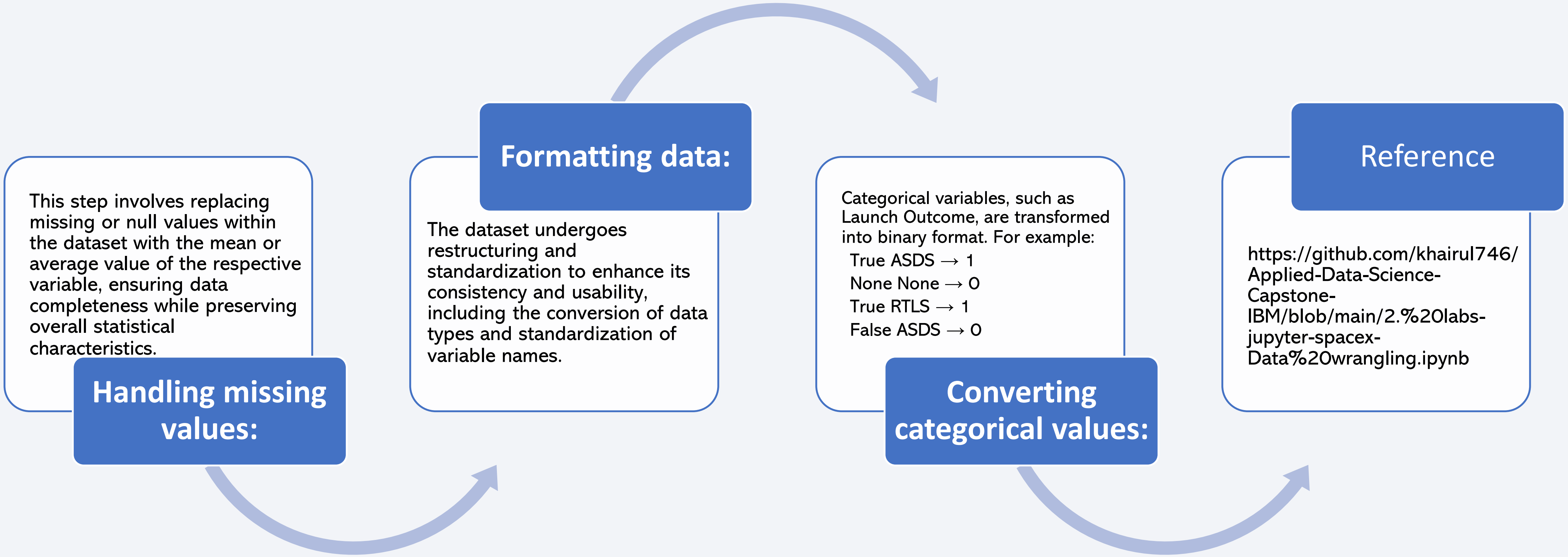
- Web server receive GET request from a client
- Web server response with a content

Client.

Filtering for Falcon 9 Launches

Highlight the filtering process to include only Falcon 9 launches.

Data Wrangling



Handling missing values:

This step involves replacing missing or null values within the dataset with the mean or average value of the respective variable, ensuring data completeness while preserving overall statistical characteristics.

Formatting data:

The dataset undergoes restructuring and standardization to enhance its consistency and usability, including the conversion of data types and standardization of variable names.

Converting categorical values:

Categorical variables, such as Launch Outcome, are transformed into binary format. For example:

- True ASDS → 1
- None None → 0
- True RTLS → 1
- False ASDS → 0

Reference

https://github.com/khairul746/Applied-Data-Science-Capstone-IBM/blob/main/2.%20labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Flight Number vs Launch Site
  - A scatter plot was employed to depict the relationship between **Flight Number** and **Launch Site**, visualizing correlations and patterns between two variables.
  - Based on the plot, there is no particular relationship between the variables
- Payload vs. Launch Site
  - A scatter plot was employed to depict the relationship between **Payload** and **Launch Site**, visualizing correlations and patterns between two variables.
  - Based on the plot, for the **VAFB-SLC** launchsite there are no rockets launched for heavypayload mass(greater than 10000).
- Success rates for each orbit type
  - Bar charts allow for easy comparison on **Success Rates** across multiple categories of **Orbit Type** simultaneously.
  - Based on the plot, there are 4 orbit types with 100% success rate.
- Flight Number vs. Orbit type
  - A scatter plot was employed to depict the relationship between **Flight Number** and **Orbit Type** visualizing correlations and patterns between two variables.
  - Based on the plot, there is no particular relationship between the variables
- Payload vs. Orbit type
  - A scatter plot was employed to depict the relationship between **Payload** and **Orbit Type**, visualizing correlations and patterns between two variables.
  - With heavy payloads the successful landing or positive landing rate is more for **Polar**, **LEO,** and **ISS**.
  - However for **GTO** we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are there here.
- Yearly trend in launch success
  - To visualize data over time the line chart is a good choice.
  - Based on the plot, The **success rate** since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.
- https://github.com/khairul746/Applied-Data-Science-Capstone-IBM/blob/main/3.%20jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- Here are the various SQL queries that have been performed
  - Display the names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first succesful landing outcome in ground pad was acheived.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failure mission outcomes
  - List the names of the booster_versions which have carried the maximum payload mass.
  - List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
  - 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.
- https://github.com/khairul746/Applied-Data-Science-Capstone-IBM/blob/main/4.%20jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- The Folium library provides a wide variety of methods for creating interactive maps in python
- **Markers** are used to mark on the map so we can tell where the launch is in California and Florida.
- **MarkerCluster** are used to group adjacent Markers in the map.
- **PolyLine** is used to create a connecting line between coordinates on the map
- https://github.com/khairul746/Applied-Data-Science-Capstone-IBM/blob/main/5.%20lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- On Plotly Dash, we can create interactive dashboards for data visualization. Dash Plotly supports creating dropdowns, sliders, checkboxes, etc.
- In the **Dropdown** menu that has been created, there are several interactive options, namely "All Sites, CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, and CCAFS SLC-40.".
- When one of the options in the **Dropdown** is clicked, a **pie chart** will appear, displaying the success rate of the launcher.
- The dashboard is also equipped with a **Slider**, whose value contains the mass of the payload.
- When the **slider** is set, a **scatter plot** will be generated displaying the payload mass and the success of the launch.
- https://github.com/khairul746/Applied-Data-Science-Capstone-IBM/blob/main/6.%20spacex_dash_app.py

Predictive Analysis (Classification)

The development of a machine learning model initiates with the division of the dataset into training and testing subsets, employing a test size of 0.2 and a random state of 10.

Divide the dataset into training and testing subsets with a test size of 0.2 and a random state of 10.

Next, several machine learning algorithms such as Decision Tree, SVM, Logistic Regression, and kNN are applied to the training data to train multiple models.

Subsequently, each trained model undergoes evaluation utilizing a confusion matrix to gauge its performance, assessing the accuracy of classifying instances into their respective categories.

https://github.com/khairul746/Applied-Data-Science-Capstone-IBM/blob/main/7.%20SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

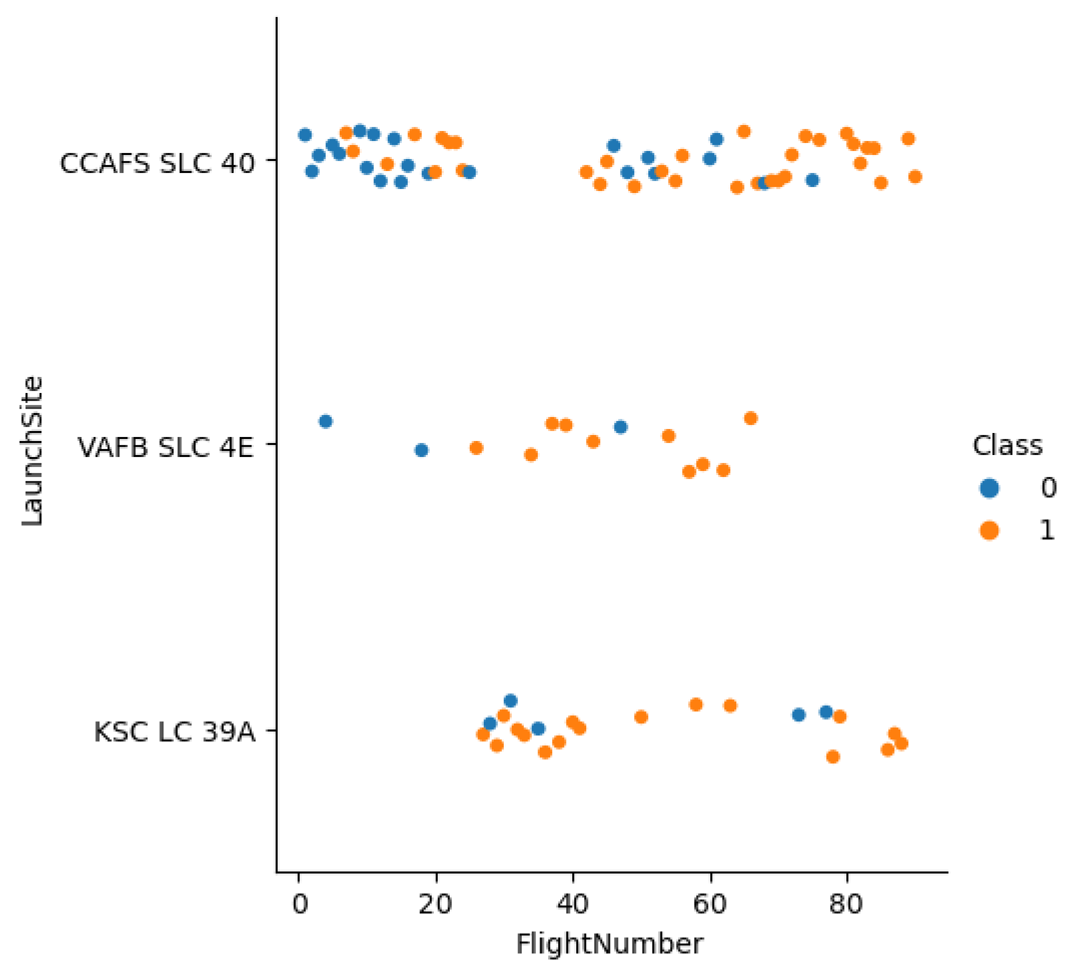
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

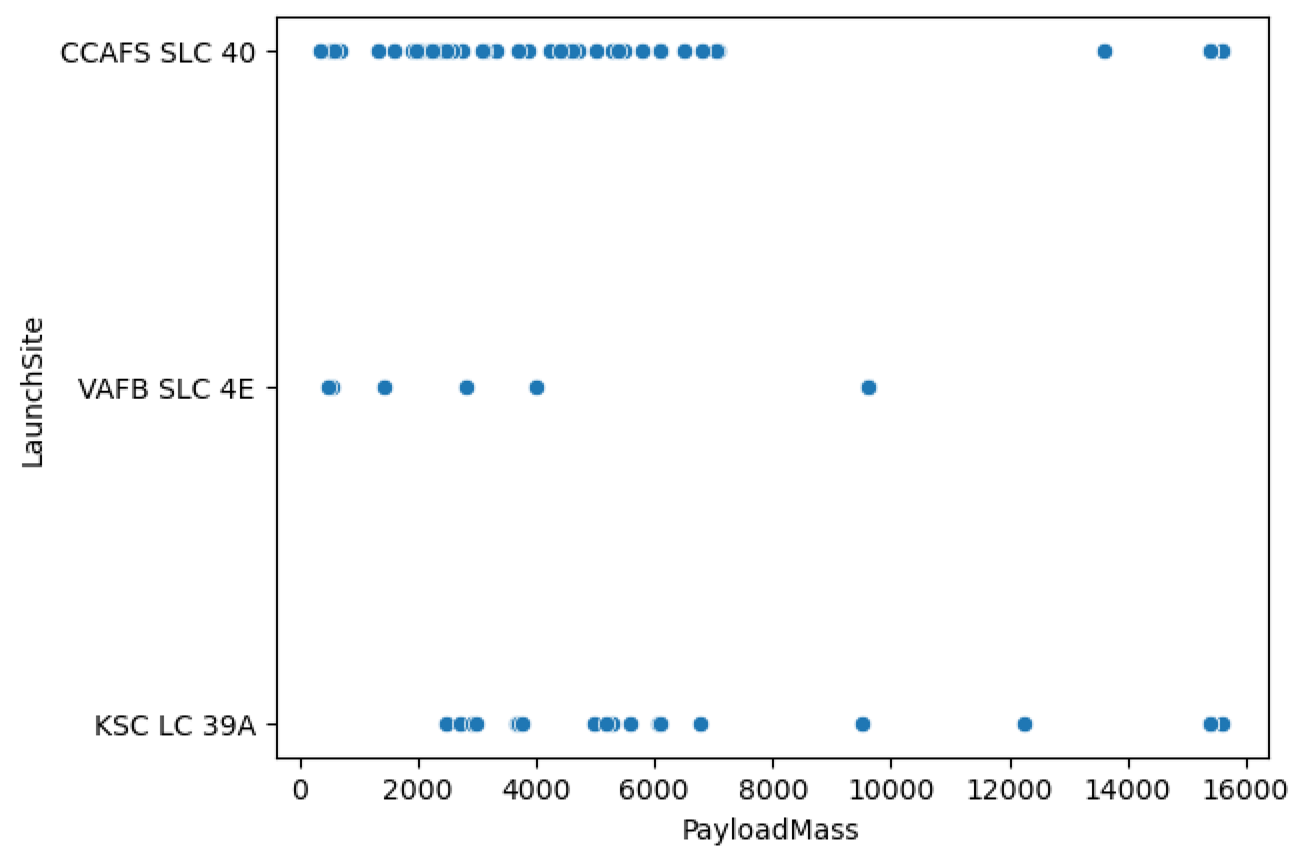
Flight Number vs. Launch Site

- There is no specific relationship between Flight Number, Launch Site and success rate



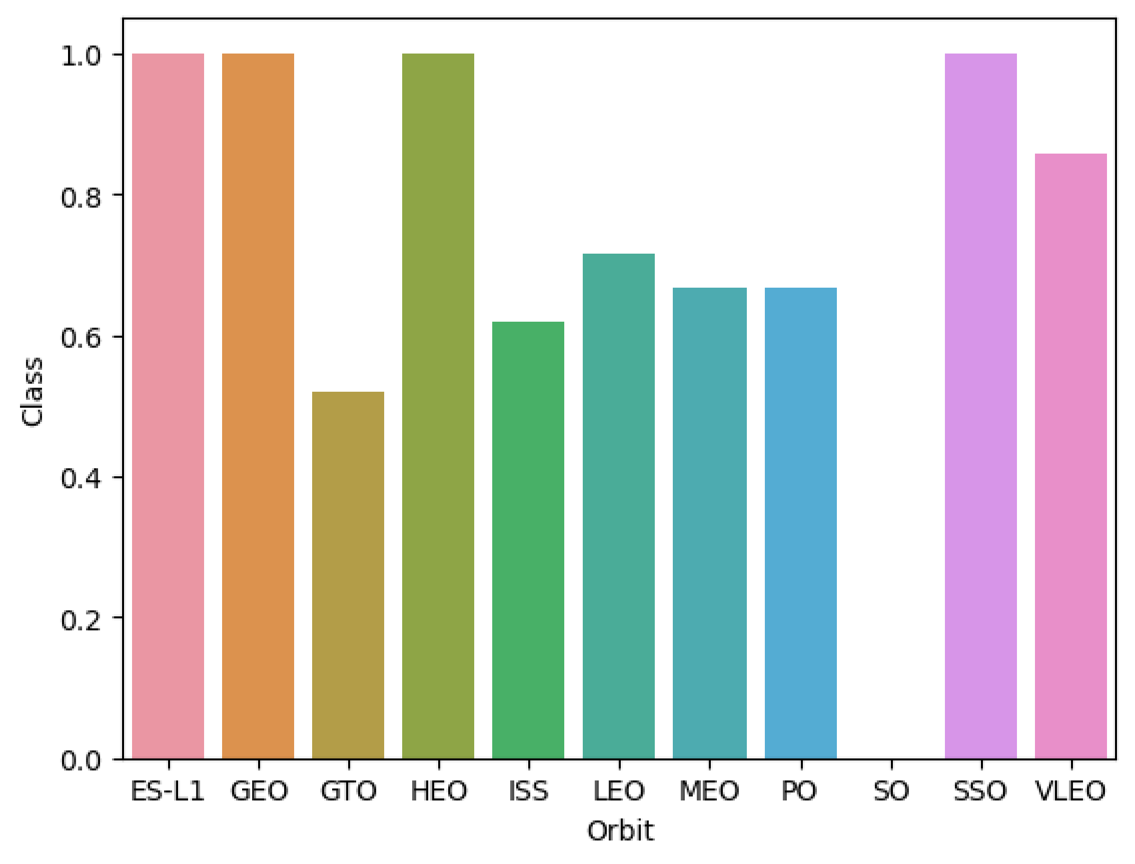
Payload vs. Launch Site

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



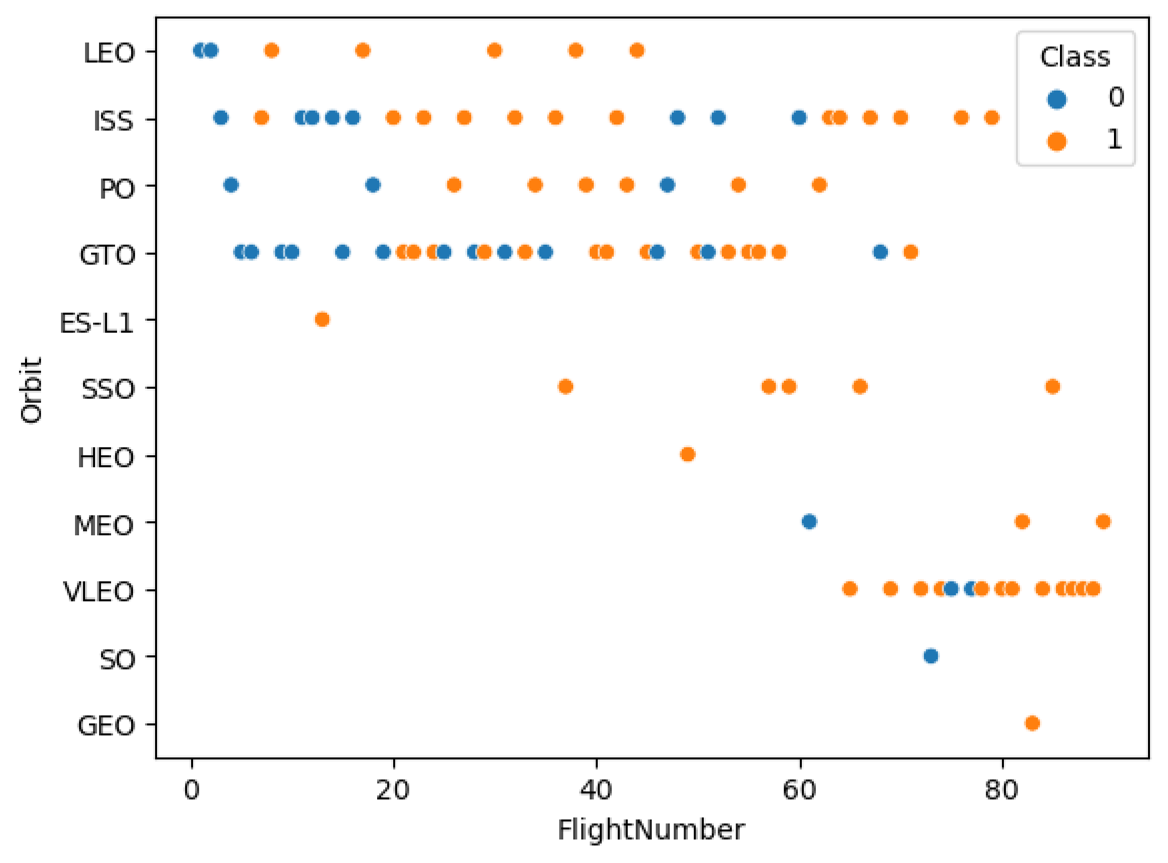
Success Rate vs. Orbit Type

- SSO, ES-L1, GEO, and HEO orbits have the highest landing success rate with 100%.



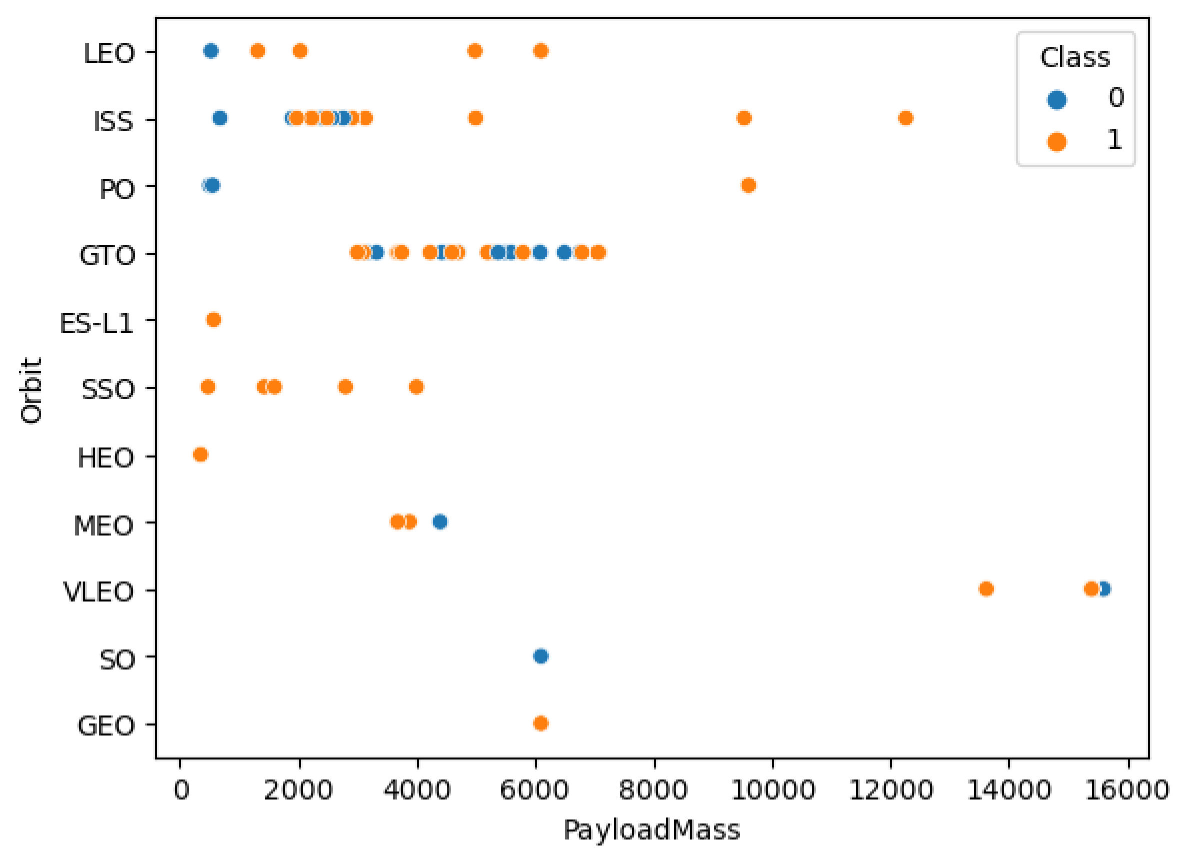
Flight Number vs. Orbit Type

- The LEO orbit success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



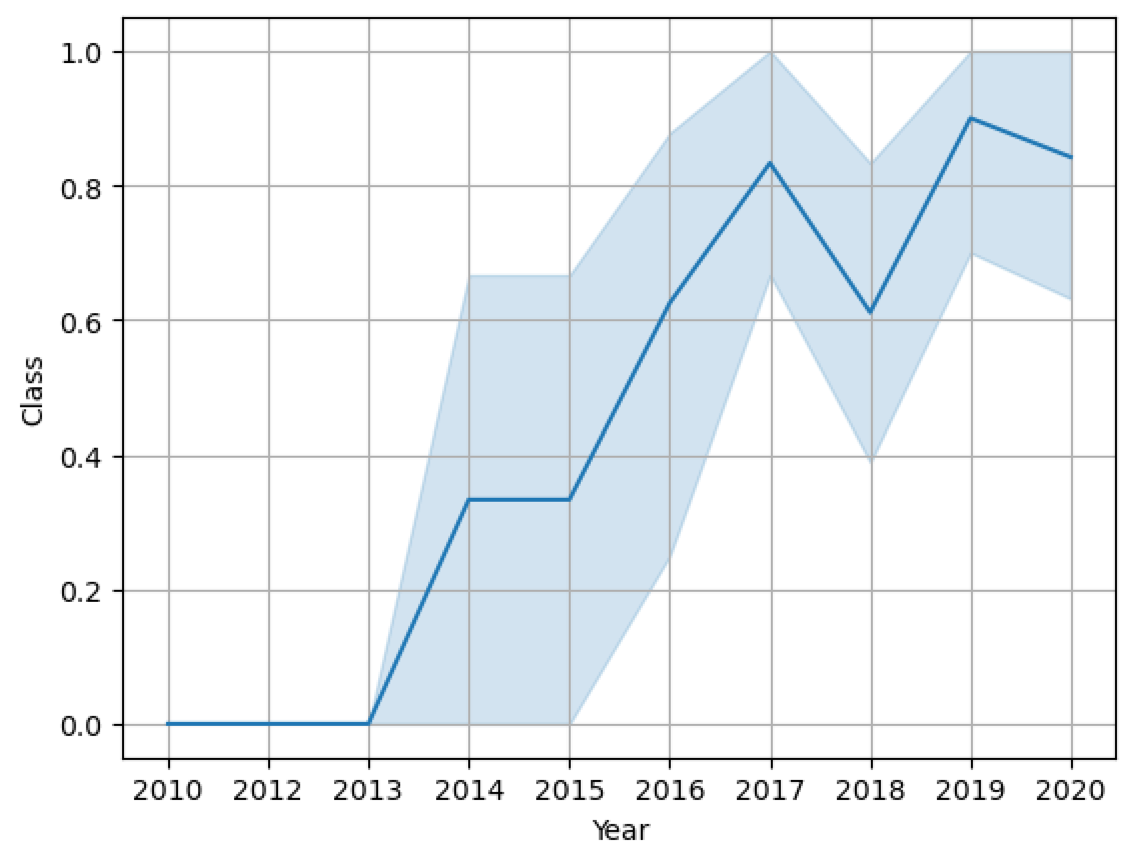
Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here.



Launch Success Yearly Trend

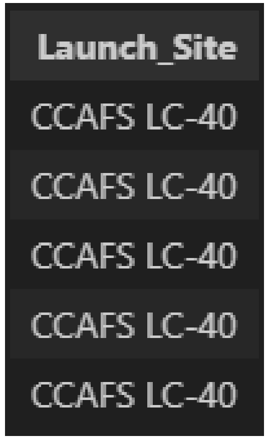
- We can observe that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.



All Launch Site Names

- There is a list of launch site names
  - CCAFS LC-40
  - VAFB SLC-4E
  - KSC LC-39A
  - CCAFS SLC-40
- In Jupyter Notebook, This query %sql select DISTINCT(Launch_Site) from SPACEXTABLE is used to retrieve a list of launch site names.
- SELECT DISTINCT(Launch_Site) : This part of the query is selecting unique/distinct values from the "Launch_Site" column. The DISTINCT keyword is used to eliminate duplicate values and retrieve only unique values.
- FROM SPACEXTABLE : This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".

Launch Site Names Begin with 'CCA'



Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

- In Jupyter Notebook, This query `%sql select Launch_Site from SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' limit 5` is used to retrieve a list of launch site names begin with ‘`CCA`’.
- `SELECT Launch_Site` : This part of the query specifies the column to be selected, which is "`Launch_Site`" in this case.
- `FROM SPACEXTABLE` : This specifies the table from which to retrieve the data, in this case, "`SPACEXTABLE`“
- `WHERE Launch_Site LIKE 'CCA%’` : This is a condition that filters the results. It selects only those records where the "`Launch_Site`" column starts with the string `'CCA'`. The `LIKE` keyword is used for pattern matching, and `'CCA%'` means any value starting with `'CCA'`.
- `LIMIT 5` : This limits the number of rows returned to 5. So, the query will return at most 5 records that satisfy the specified conditions.

Total Payload Mass

- The total payload carried by boosters from NASA is 45,596 kg
- In Jupyter Notebook, This query %%sql select sum(PAYLOAD_MASS__KG_) AS Total_Payload_Mass_kg from SPACEXTABLE where Customer == 'NASA (CRS)’ is used to retrieve total payload mass carried by booster launched by NASA (CRS).
- SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass_kg : This part of the query selects the sum of the "PAYLOAD_MASS__KG_" column and aliases the result as "Total_Payload_Mass_kg". The SUM function is used to calculate the total payload mass.
- FROM SPACEXTABLE : This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".
- WHERE Customer = 'NASA (CRS)’ : This is a condition that filters the results. It selects only those records where the "Customer" column is equal to 'NASA (CRS)'.

Average Payload Mass by F9 v1.1

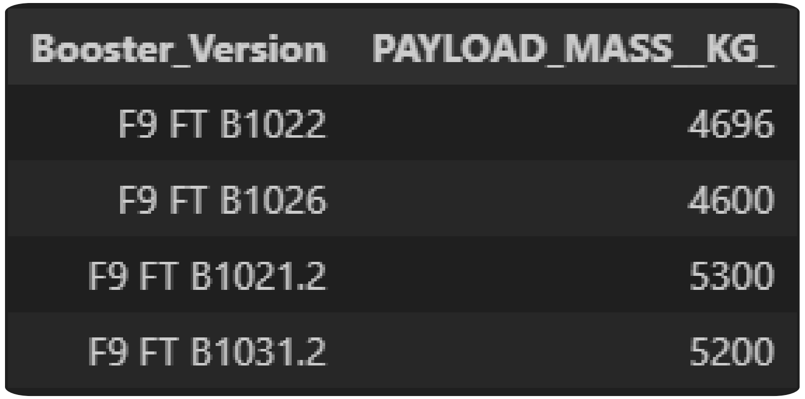
- The average payload mass carried by booster version F9 v1.1is 2928.4 kg
- In Jupyter Notebook, This query %%sql select avg(PAYLOAD_MASS__KG_) AS Average_Payload_Mass from SPACEXTABLE WHERE Booster_Version == 'F9 v1.1’ is used to retrieve average payload mass carried by booster version F9 v1.1.
- SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass_kg : This part of the query selects the sum of the "PAYLOAD_MASS__KG_" column and aliases the result as "Total_Payload_Mass_kg". The SUM function is used to calculate the total payload mass.
- FROM SPACEXTABLE: This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".
- WHERE Customer = 'NASA (CRS)’ : This is a condition that filters the results. It selects only those records where the "Customer" column is equal to 'NASA (CRS)’.

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad is at 2015-12-22
- In Jupyter Notebook, This query %sql select min(Date) as First_Succesful_Landing_Date from SPACEXTABLE where Landing_Outcome == 'Success (ground pad)’ is used to retrieve the first successful landing outcome on ground pad.
- select min(Date) as First_Successful_Landing_Date: This part of the query selects the minimum (earliest) date from the "Date" column and aliases the result as "First_Successful_Landing_Date". The MIN function is used to find the minimum value in a column.
- from SPACEXTABLE: This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".
- where Landing_Outcome = 'Success (ground pad)': This is a condition that filters the results. It selects only those records where the "Landing_Outcome" column is equal to 'Success (ground pad)'.

Successful Drone Ship Landing with Payload between 4000 and 6000

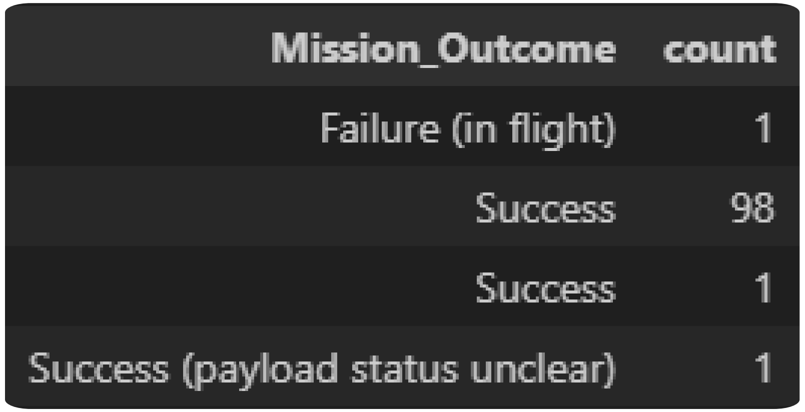
- The table in the left side is a list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000.
- In Jupyter Notebook, This query %%sql select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTABLE where Landing_Outcome == 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000; is used.
- SELECT Booster_Version, PAYLOAD_MASS__KG_: This part of the query specifies the columns to be selected, which are "Booster_Version" and "PAYLOAD_MASS__KG_".
- FROM SPACEXTABLE: This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".
- WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000: This is a condition that filters the results. It selects only those records where the "Landing_Outcome" is 'Success (drone ship)' and the "PAYLOAD_MASS__KG_" is between 4000 and 6000.



Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

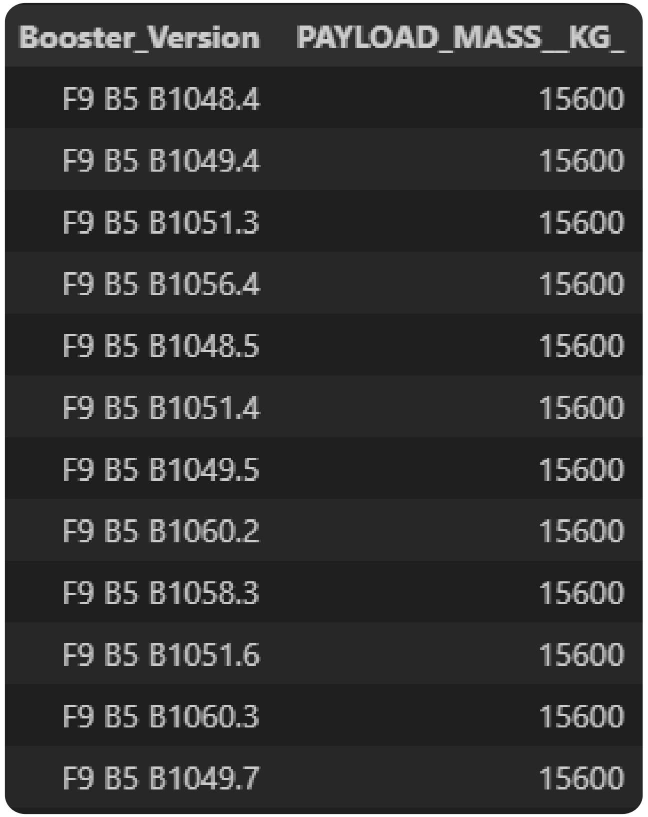
- The table in the left side present the total number of successful and failure mission outcomes
- In Jupyter Notebook, This query %%sql select Mission_Outcome, count(Mission_Outcome) as count from SPACEXTABLE group by Mission_Outcome is used.
- SELECT Mission_Outcome, COUNT(Mission_Outcome) AS count: This part of the query selects the "Mission_Outcome" column and counts the occurrences of each unique value. The result is aliased as "count".
- FROM SPACEXTABLE: This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".
- GROUP BY Mission_Outcome: This groups the results based on the unique values in the "Mission_Outcome" column.



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

Boosters Carried Maximum Payload

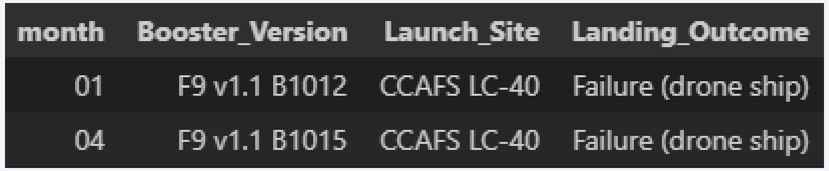
- On the left side there is a list of the booster which have carried the maximum payload mass
- In Jupyter Notebook, This query %%sql select Booster_Version,PAYLOAD_MASS__KG_ from SPACEXTABLE where PAYLOAD_MASS__KG_ == (select max(PAYLOAD_MASS__KG_) from SPACEXTABLE) is used.
- SELECT Booster_Version, PAYLOAD_MASS__KG_: This part of the query specifies the columns to be selected, which are "Booster_Version" and "PAYLOAD_MASS__KG_".
- FROM SPACEXTABLE: This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".
- WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE): This is a condition that filters the results. It selects only those records where the "PAYLOAD_MASS__KG_" is equal to the maximum payload mass obtained from the subquery (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE).



Booster_Version	PAYLOAD_MASS__KG_
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

2015 Launch Records

- The table below list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015.

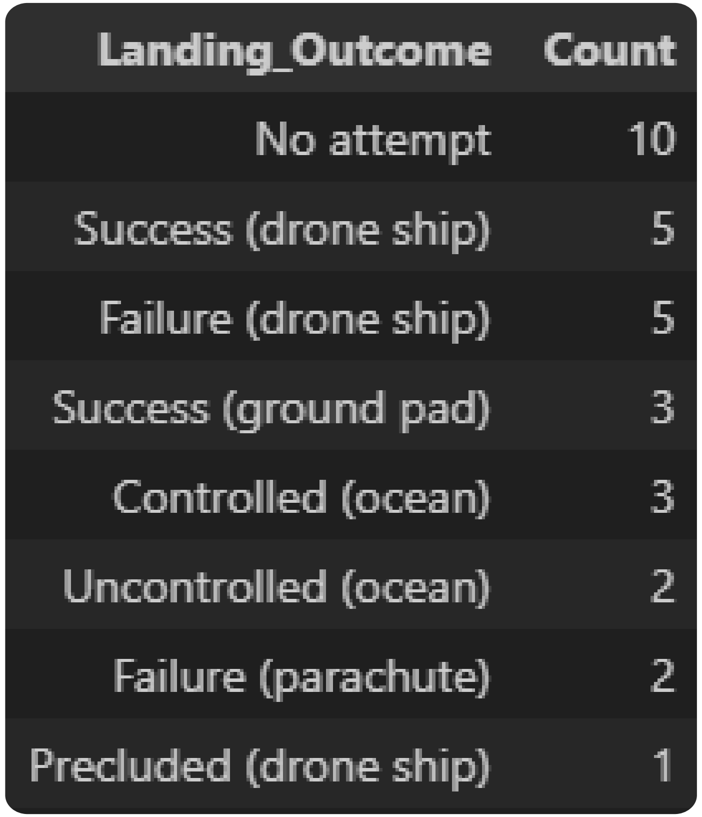


month	Booster_Version	Launch_Site	Landing_Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

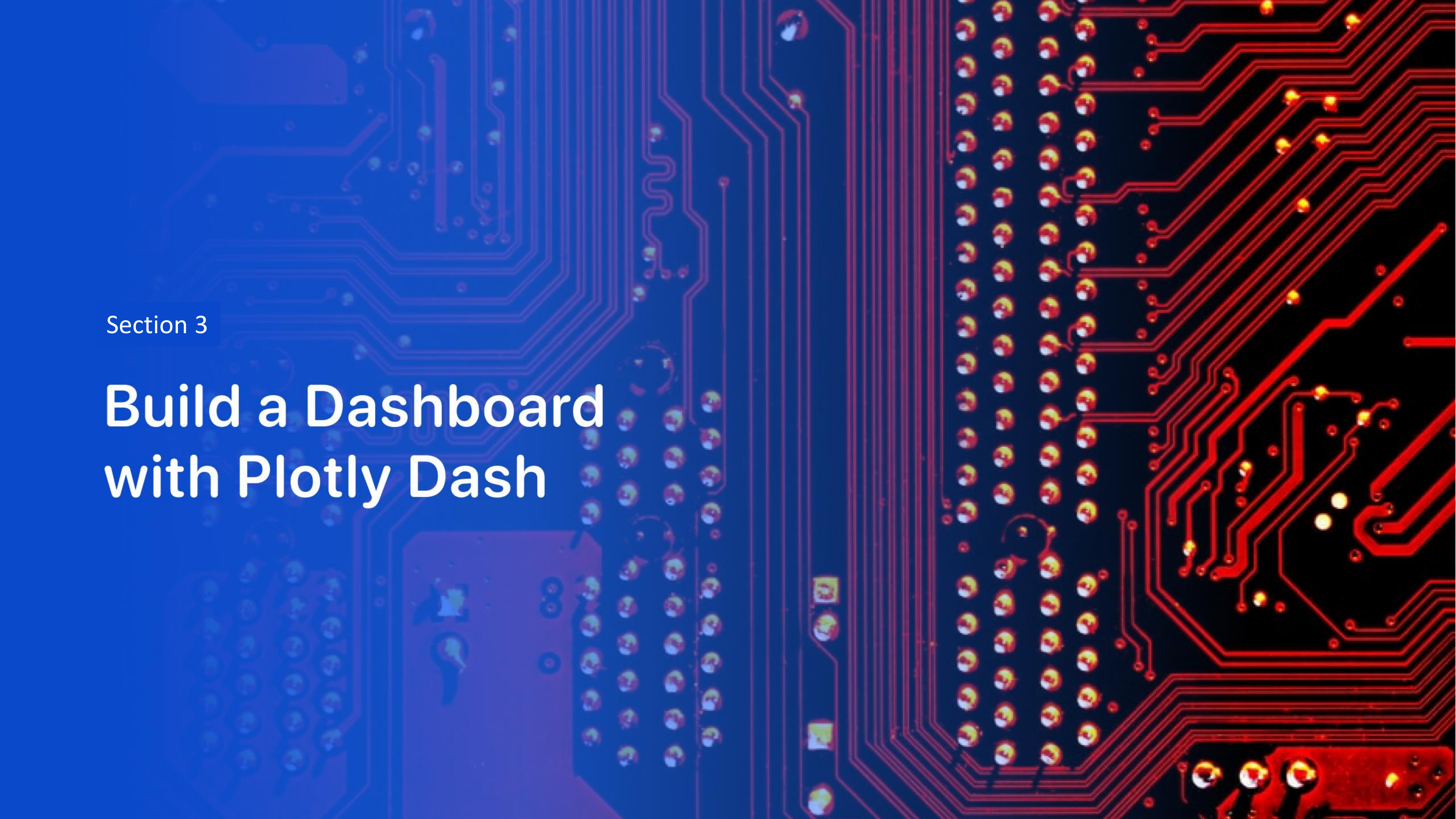
- In Jupyter Notebook, This query `%%sql select substr(Date, 6,2) as month, Booster_Version, Launch_Site, Landing_Outcome from SPACEXTABLE where Landing_Outcome like 'Fail%' and substr(Date,0,5)='2015'`is used.
- `SELECT substr(Date, 6, 2) AS month, Booster_Version, Launch_Site, Landing_Outcome`: This part of the query specifies the columns to be selected. It also uses the `substr` function to extract the month part from the "`Date`" column and aliases it as "`month`".
- `FROM SPACEXTABLE`: This specifies the table from which to retrieve the data, in this case, "`SPACEXTABLE`".
- `WHERE Landing_Outcome LIKE 'Fail%' AND substr(Date, 0, 5) = '2015'`: This is a condition that filters the results. It selects only those records where the "`Landing_Outcome`" starts with '`Fail`' and the year part of the "`Date`" column is '`2015`'.

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

- The table on the left side 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
- In Jupyter Notebook, This query %%sql select Landing_Outcome, count(Landing_Outcome) as Count from SPACEXTABLE where Date between '2010-06-04' and '2017-03-20’ group by Landing_Outcome order by count(Landing_Outcome) desc is used.
- SELECT Landing_Outcome, COUNT(Landing_Outcome) AS Count: This part of the query selects the "Landing_Outcome" column and counts the occurrences of each unique value. The result is aliased as "Count".
- FROM SPACEXTABLE: This specifies the table from which to retrieve the data, in this case, "SPACEXTABLE".
- WHERE Date BETWEEN '2010-06-04' AND '2017-03-20': This is a condition that filters the results to only include records with a "Date" within the specified range.
- GROUP BY Landing_Outcome: This groups the results based on the unique values in the "Landing_Outcome" column.
- ORDER BY COUNT(Landing_Outcome) DESC: This orders the result set based on the count of each "Landing_Outcome" in descending order.



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

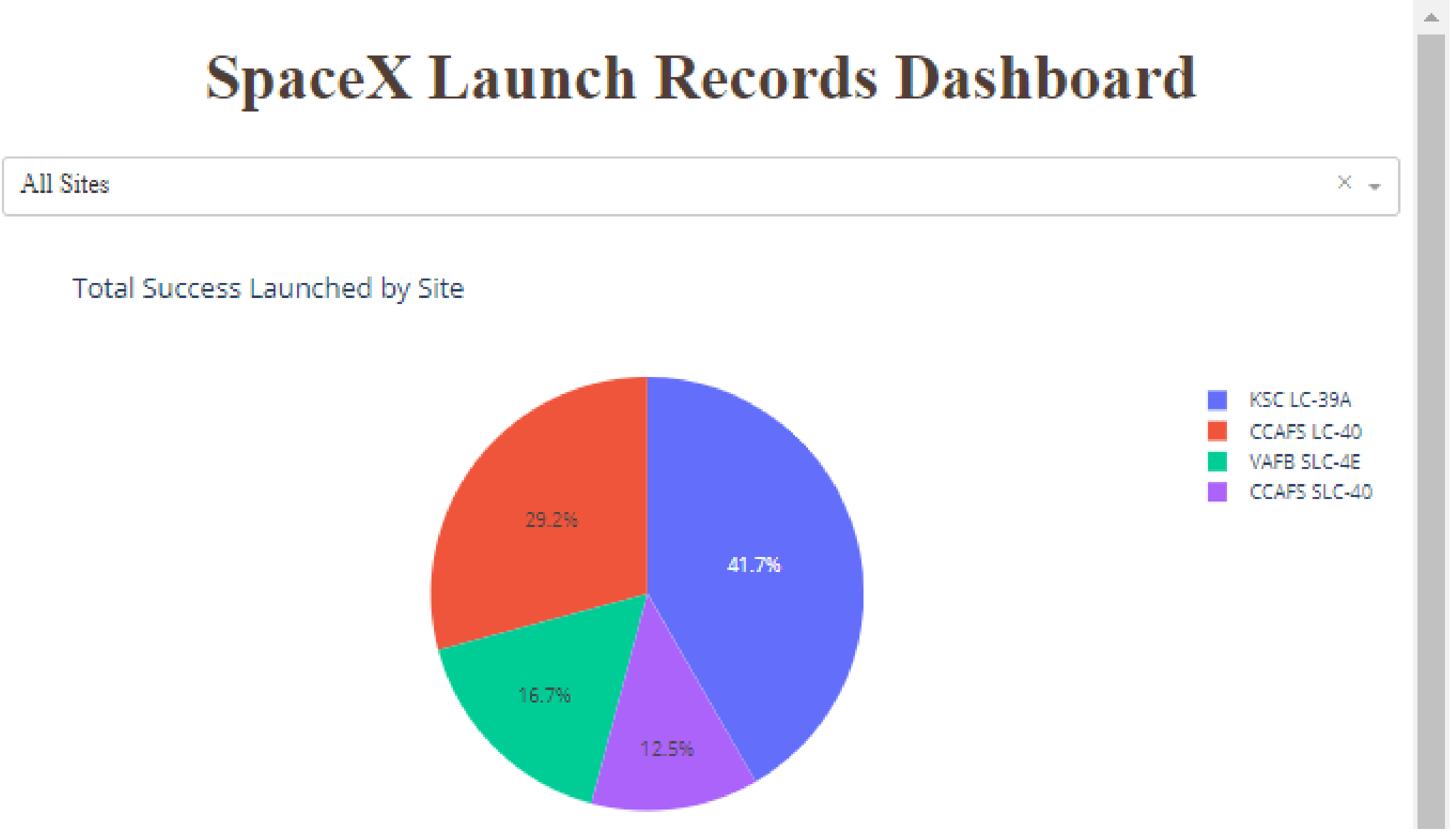


Section 3

Build a Dashboard with Plotly Dash

Total Success Launched by Site

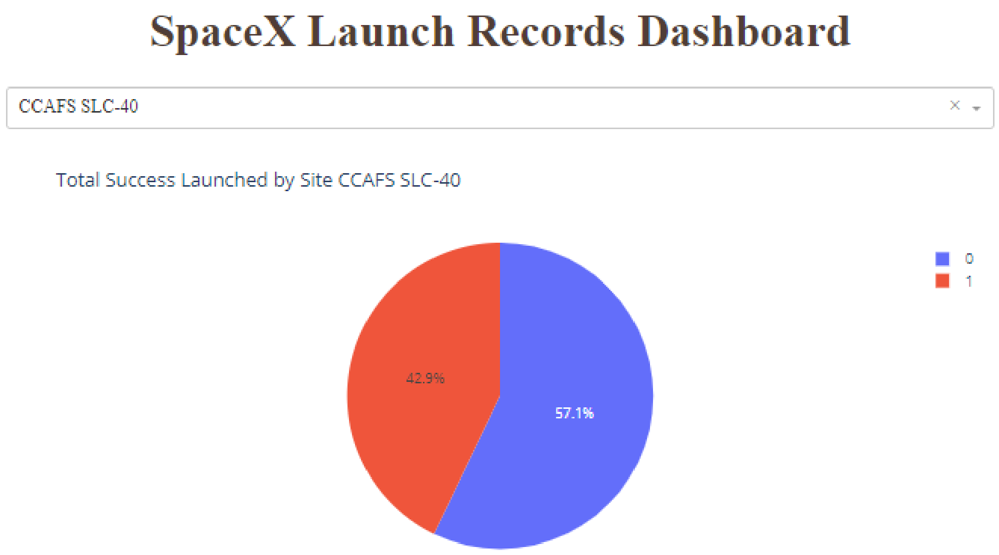
Based on the pie plot, it can be seen that the KSC LC-40 site has the highest number of successes compared to other sites. While the CCAFS SLC-40 site has the smallest number of successes.



SpaceX Launch Records Dashboard

All Sites

Highest Launch Success Ratio



SpaceX Launch Records Dashboard

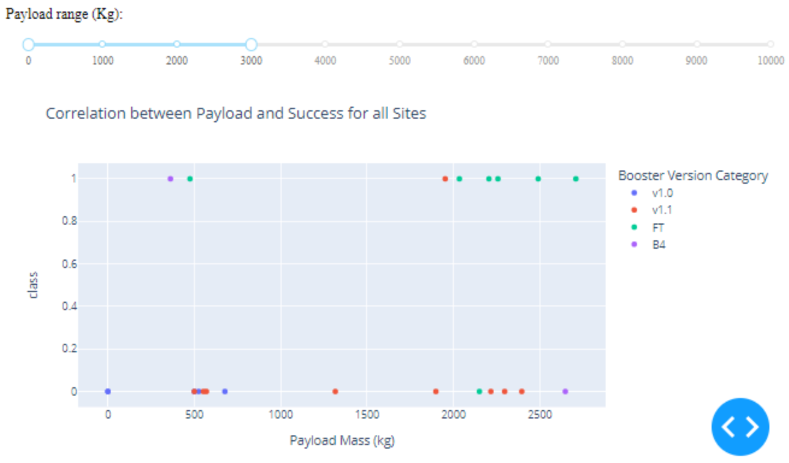
CCAFS SLC-40

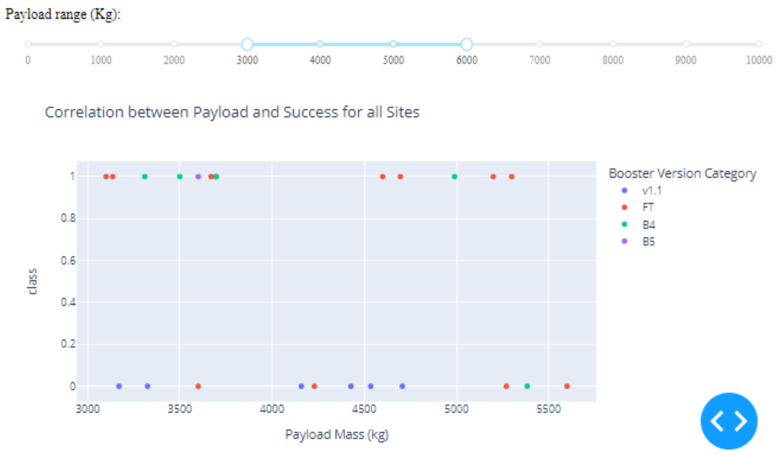
Total Success Launched by Site CCAFS SLC-40

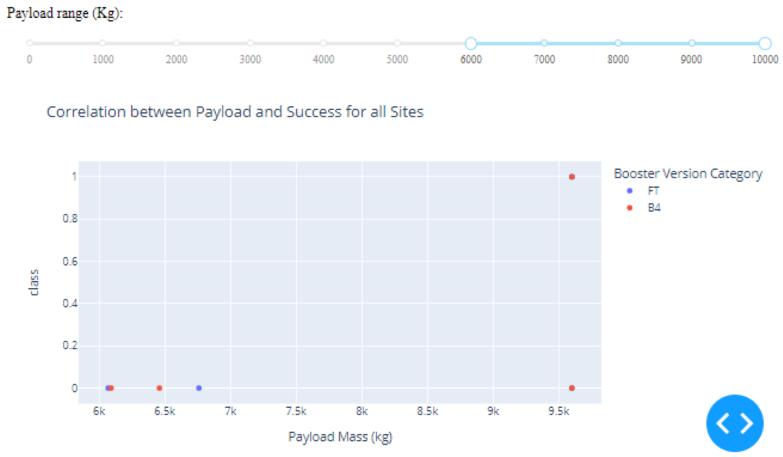
Although the CCAFS SLC-40 site has the smallest number of successful launches, it has the highest successful launch rate at 42.9%.

Payload Mass vs Launch Outcome

- For payload mass range of 0–3000 kg, booster version category “FT” has the highest success launch outcome.
- For the payload mass range of 3000–6000 kg, the booster version category "v1.1" had the most unsuccessful launch outcome.
- For the payload mass range of 6000–10000 kg, almost all booster version categories fail.



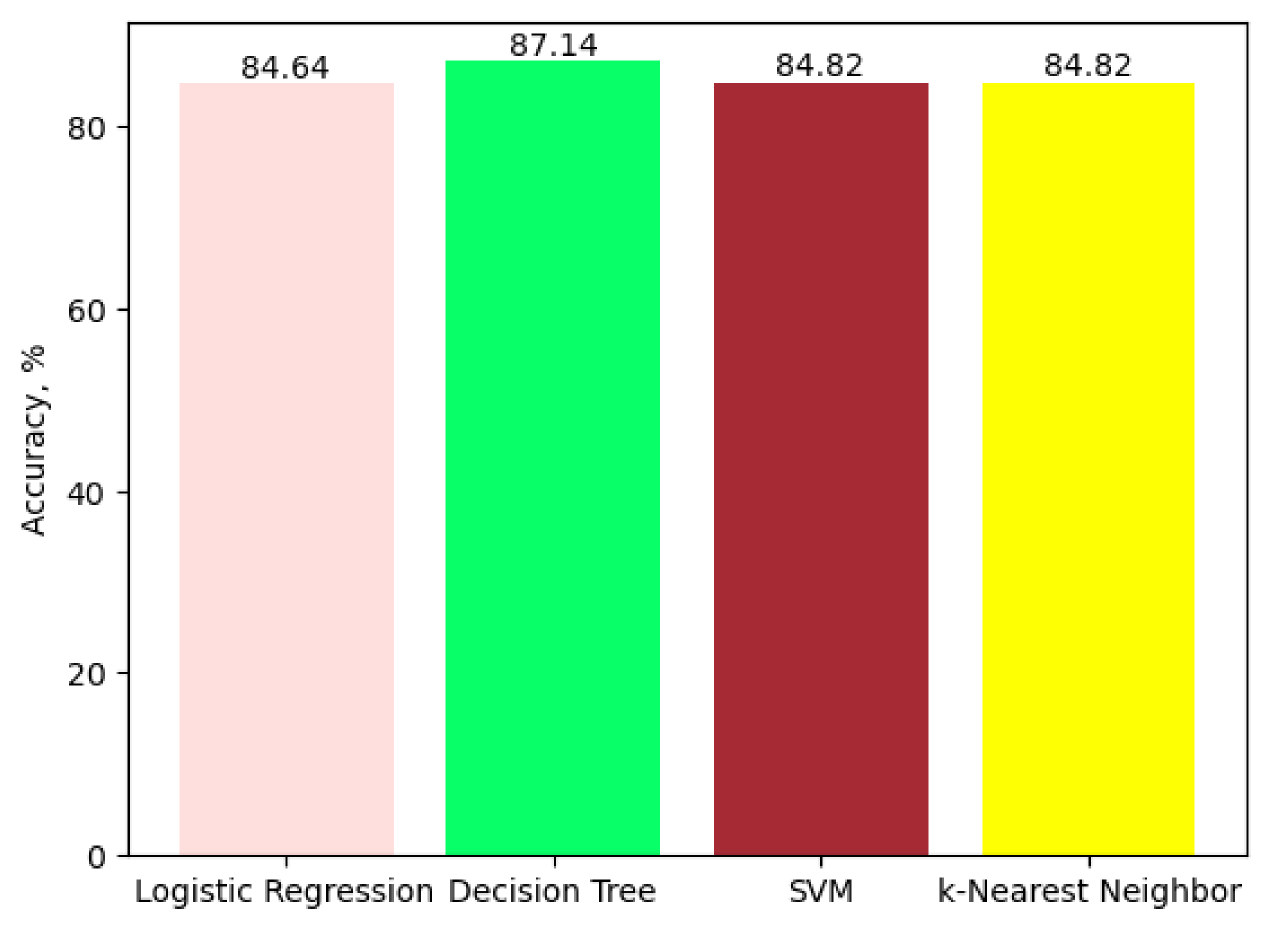




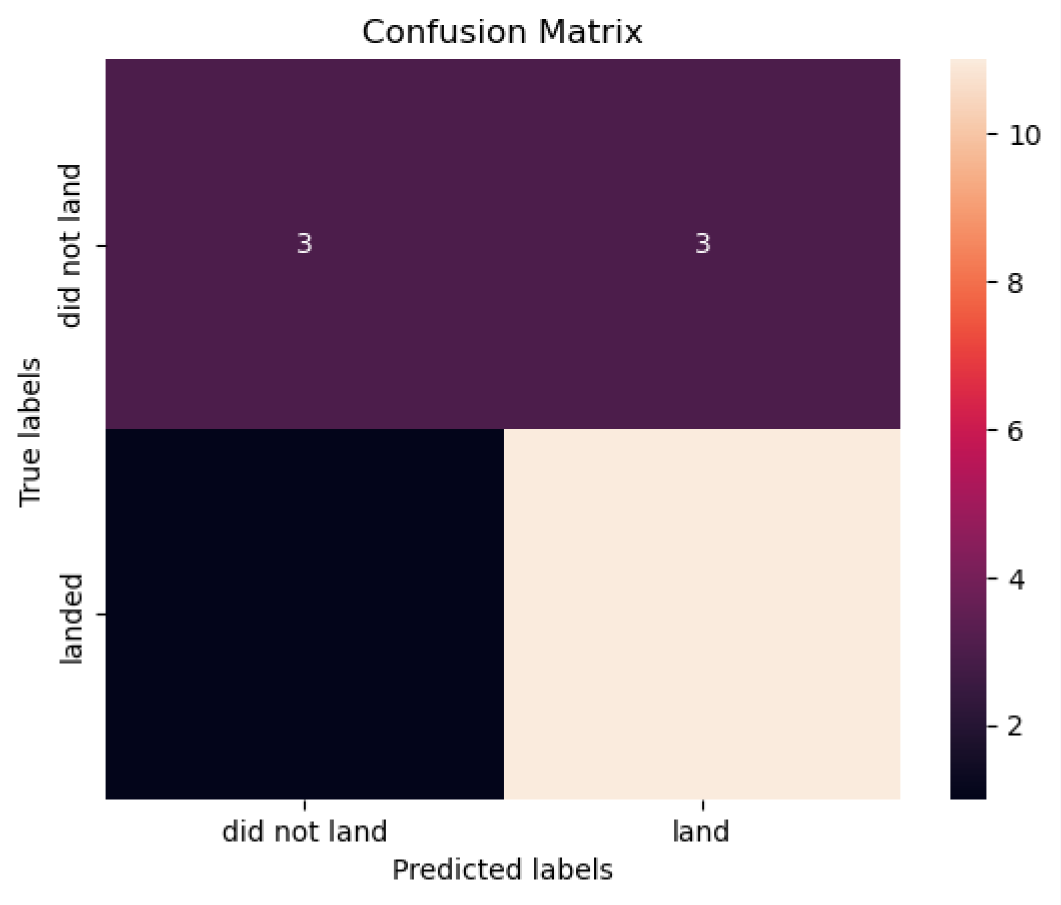
Section 4

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix



- True Positive (TP):
  - Value: 11
  - Description: The number of data that is actually positive and predicted positive by the model. In this case, 11 samples that did land are correctly predicted as landed.
- False Positive (FP):
  - Value: 3
  - Description: The number of data that is actually negative but predicted positive by the model. In this case, 3 samples that did not land are incorrectly predicted as landed.
- False Negative (FN):
  - Value: -
  - Description: The number of data that is actually positive but predicted negative by the model. In this case, no samples that did land are incorrectly predicted as did not land.
- True Negative (TN):
  - Value: 3
  - Description: The number of data that is actually negative and predicted negative by the model. In this case, 3 samples that did not land are correctly predicted as did not land.

Conclusions

- The machine learning decision tree model has the highest accuracy compared to the other 3 models, with 87.14% accuracy.
- Through EDA we can find out the relationship between variables such as flight number, flight site, payload mass, orbit type and success rate. In addition, the annual trend towards success rate can also be seen through the EDA process.
- With the Folium library, we can find out that the launch site is close to the coast of Florida and California.
- With the Plotly Dash library, we can create an interactive dashboard that displays data visualizations with pie charts and scatter charts with interaction buttons like Dropdown and RangeSlider.

Thank you!