IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Ryandra Narlan>
<September 18, 2024>

Outline

Executive Summary

Summary of methodologies

The methodology involves collecting SpaceX launch data via the SpaceX REST API, focusing on rocket specifications, payloads, and landing outcomes. Using Python's `requests` library, the data is extracted in JSON format and transformed into a DataFrame for exploratory data analysis (EDA) with visualizations and SQL. Key attributes like launch sites and success rates are analyzed to identify correlations with successful landings. Interactive visual analytics are conducted using Folium and Plotly Dash, enabling dynamic exploration of data. Data wrangling focuses on attributes such as Flight Number, Date, and Launch Site, categorizing landing outcomes as success (1) or failure (0).

Executive Summary

Summary of all results

Both `FlightNumber` and `Payload` influence launch outcomes, though increased flight numbers have minimal impact across launch sites. No heavy payloads (over 10,000 kg) were launched from the VAFB-SLC site. ES-L1, GEO, HEO, and SSO orbits show the highest success rates, while LEO success correlates with flight numbers; GTO does not. Heavy payloads achieve higher success in Polar, LEO, and ISS orbits. KSC LC-39A boasts the highest overall success rate. The FT Booster Version excels in success based on payload mass, and a Decision Tree classifier achieves an accuracy of 0.86 with a test accuracy of 0.94.

Introduction

- Project background and context : The commercial space era has begun, with companies making space travel more accessible. SpaceX leads by offering cost-effective launches, reusing rockets to reduce expenses.
- Problems you want to find answers : . In this project, as a data scientist for Space Y, you'll analyze SpaceX's data to predict the success of first-stage landings using machine learning.

Section 1

Methodology

Methodology

- Data collection methodology:
  - SpaceX launch data collected via the SpaceX REST API. The data includes details on launches such as rocket specifications, payloads, and landing outcomes. Using Python's `requests` library, this data extracted in JSON format and transform it into a dataframe.
- Perform data wrangling
  - Key attributes to wrangle include Flight Number, Date, Booster version, Payload mass, Orbit, Launch Site, and Outcome. Landing outcomes will be categorized as 0 (failure) or 1 (success), simplifying analysis for launch results.

Methodology

- Perform exploratory data analysis (EDA) using visualization and SQL
    - Exploratory Data Analysis (EDA) performed using visualizations and SQL to explore Falcon 9 landing data. By analyzing features like launch sites, success rates, and payload mass, key attributes correlated with successful landings identified, preparing the data for machine learning predictions.
- Perform interactive visual analytics using Folium and Plotly Dash
    - Interactive visual analytics allows users to explore data dynamically, using tools like zoom, pan, and filter. In this module, interactive maps built with Folium and dashboards built with Plotly Dash to discover visual patterns and analyze launch sites effectively.

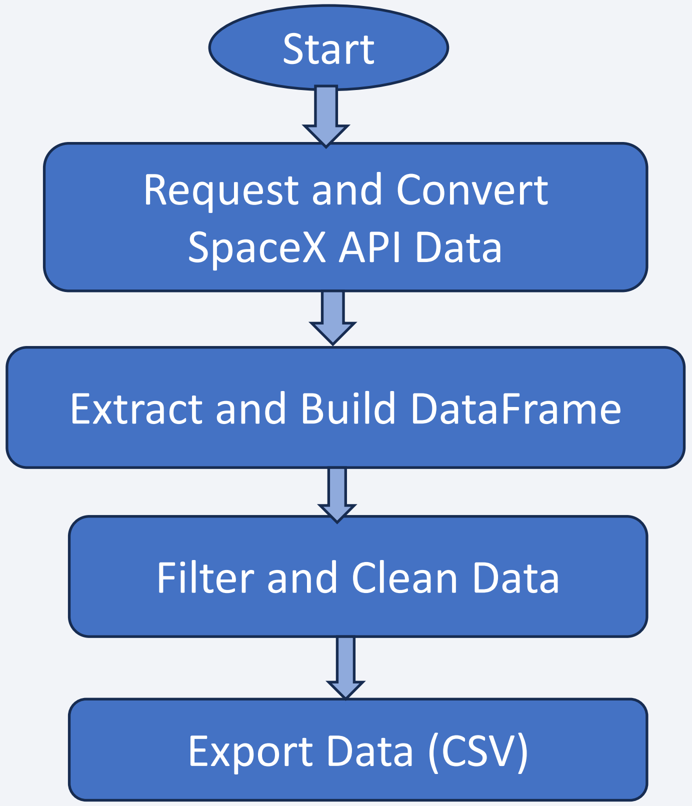
Data Collection

- We began by gathering SpaceX launch data through the API, transforming the extensive information into a Pandas dataframe using `json_normalize()`. With a focus on key columns—rocket, launchpad, payloads, and cores—we extracted valuable details such as booster names, payload mass and orbits, launch site coordinates, and core attributes. This process allowed us to build a comprehensive new dataframe, specifically filtering to include only Falcon 9 launches.

- Next, we tackled data quality issues by identifying and handling missing values. While `LandingPad` column retained `None` to indicate unused pads, we addressed gaps in `PayloadMass` by calculating its mean and filling in `np.nan` values with this mean. The cleaned dataset, now free of missing values except for `LandingPad`, was saved as `data_falcon9`. Finally, we exported this refined data to CSV, readying it for further analysis within a pre-selected date range in the upcoming lab.

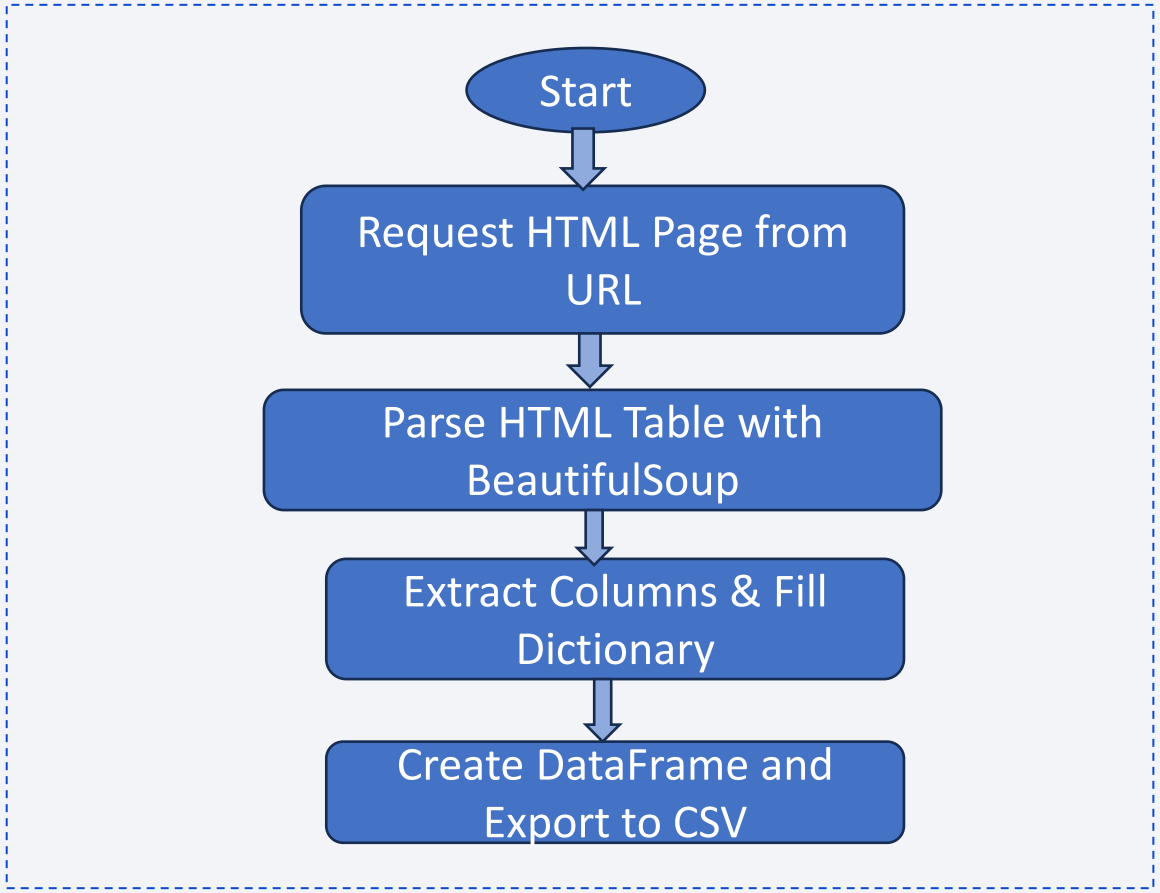
Data Collection – SpaceX API

- Request SpaceX launch data via REST API, Convert API response JSON to Pandas DataFrame using (json_normalize()), Extract specific details from DataFrame: rocket IDs, launchpad IDs, payload IDs, core IDs, Create a new DataFrame with extracted details, Filter DataFrame to include only Falcon 9 launches, Identify and address missing values, Export the cleaned DataFrame to CSV for further analysis.
- https://github.com/Ryandran/space-y/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb


Start
Request and Convert SpaceX API Data
Extract and Build DataFrame
Filter and Clean Data
Export Data (CSV)

Data Collection - Scraping

- To collect Falcon 9 launch records, start by scraping the historical data from a specific Wikipedia page using BeautifulSoup. First, request the HTML page from the URL provided and check the response to ensure successful retrieval. Next, identify and extract relevant column headers from the target table. Create an empty dictionary with these column names as keys. Iterate through the table rows, filling the dictionary with launch records while handling HTML table noise like references and missing values. Convert this dictionary into a Pandas DataFrame and export it to CSV for consistency and further analysis.
- https://github.com/Ryandran/space-y/blob/main/jupyter-labs-webscraping.ipynb

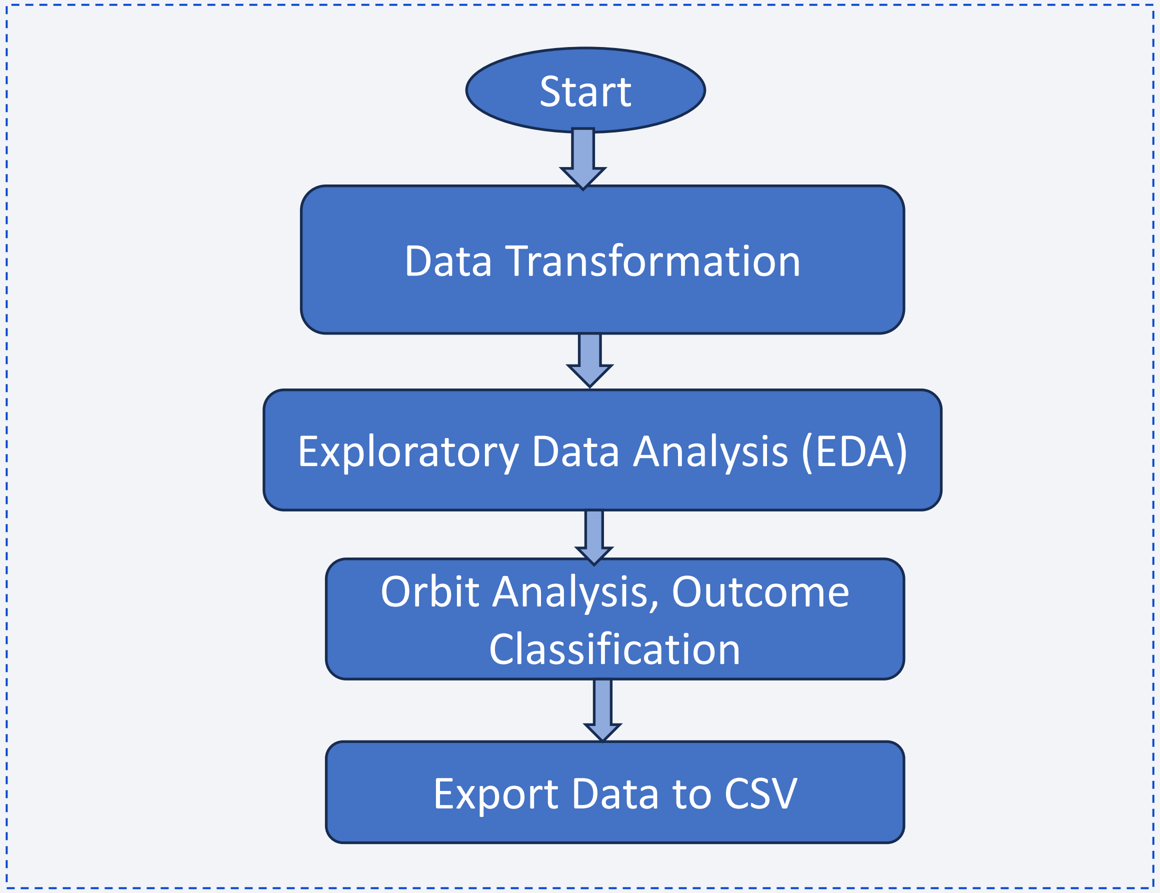

Start
Request HTML Page from URL
Parse HTML Table with BeautifulSoup
Extract Columns & Fill Dictionary
Create DataFrame and Export to CSV

Data Wrangling

- In the journey to refine the Falcon 9 launch data, the process began by transforming landing outcomes into Training Labels, with `1` marking a successful landing and `0` indicating a failure. Exploratory Data Analysis (EDA) then took center stage, identifying missing values and calculating their percentages to ensure data integrity. Analyzing the number of launches at each site using `value_counts()` revealed insights into the operational frequency of different launch sites.

Data Collection - Scraping

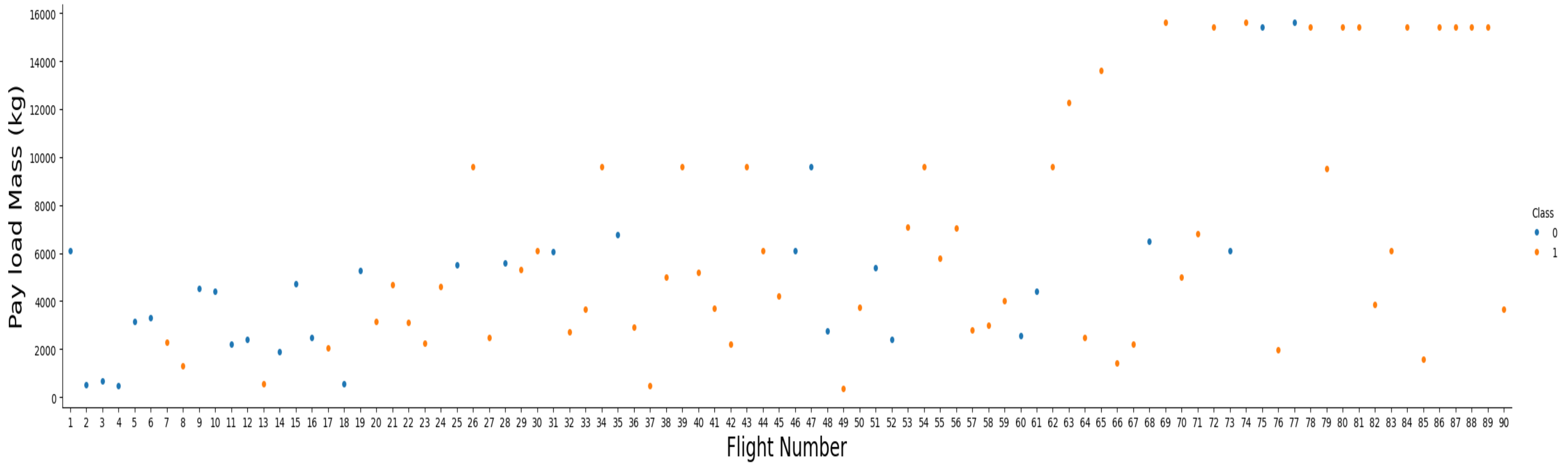
- With a focus on orbits, the types and their occurrences were meticulously examined. Landing outcomes were scrutinized, distinguishing between successful and unsuccessful attempts, and a binary `landing_class` list was created to represent these outcomes. This classification transformed complex data into a format ready for further analysis. The cleaned dataset was then exported to CSV, setting the stage for the next analytical steps while ensuring consistency with a pre-selected date range.
- https://github.com/Ryandran/space-y/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb


Start
Data Transformation
Exploratory Data Analysis (EDA)
Orbit Analysis, Outcome Classification
Export Data to CSV

First, let's try to see how the FlightNumber (indicating the continuous launch attempts.) and Payload variables would affect the launch outcome

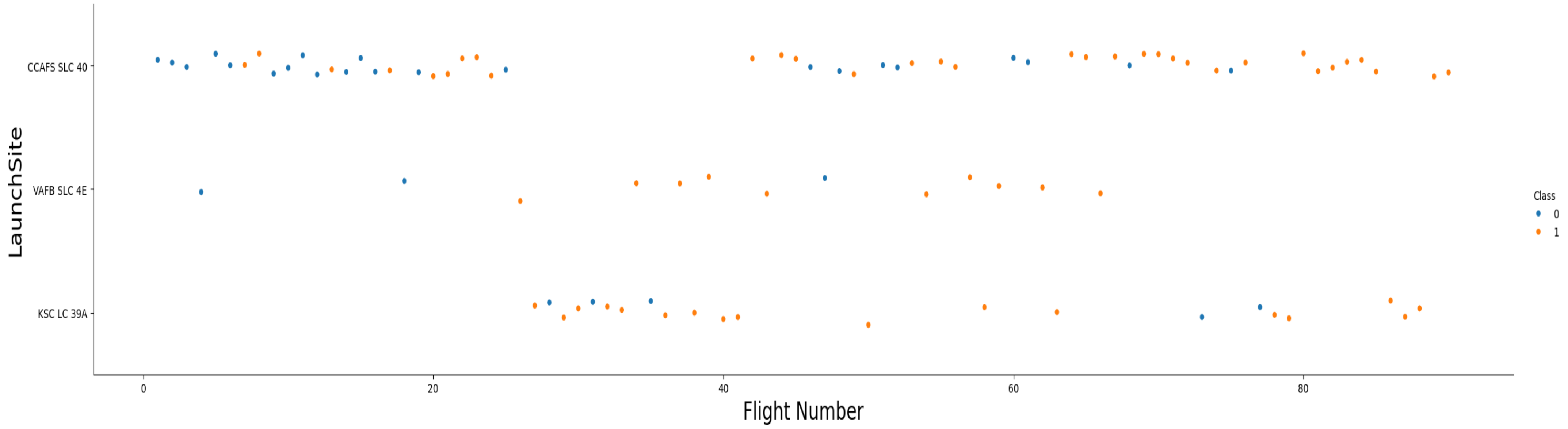
EDA with Data Visualization

- This chart see how the `FlightNumber` and `Payload` variables would affect the launch outcome.
- https://github.com/Ryandran/space-y/blob/main/edadataviz.ipynb



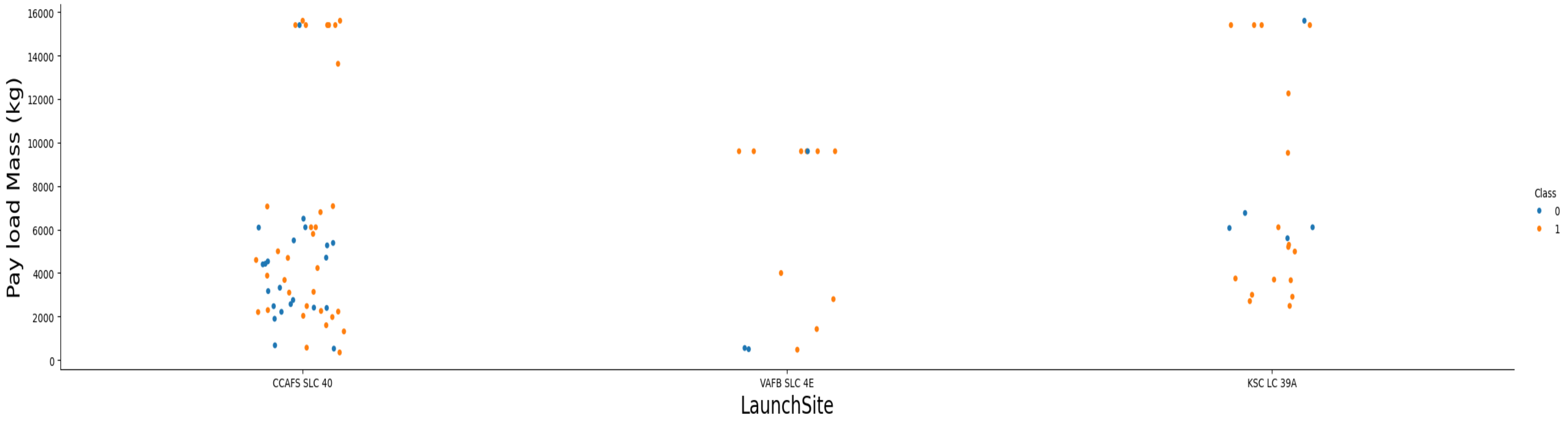
EDA with Data Visualization

- This chart see how the increase in flight numbers does not have a significant impact on the outcome of the launch site.
- https://github.com/Ryandran/space-y/blob/main/edadataviz.ipynb

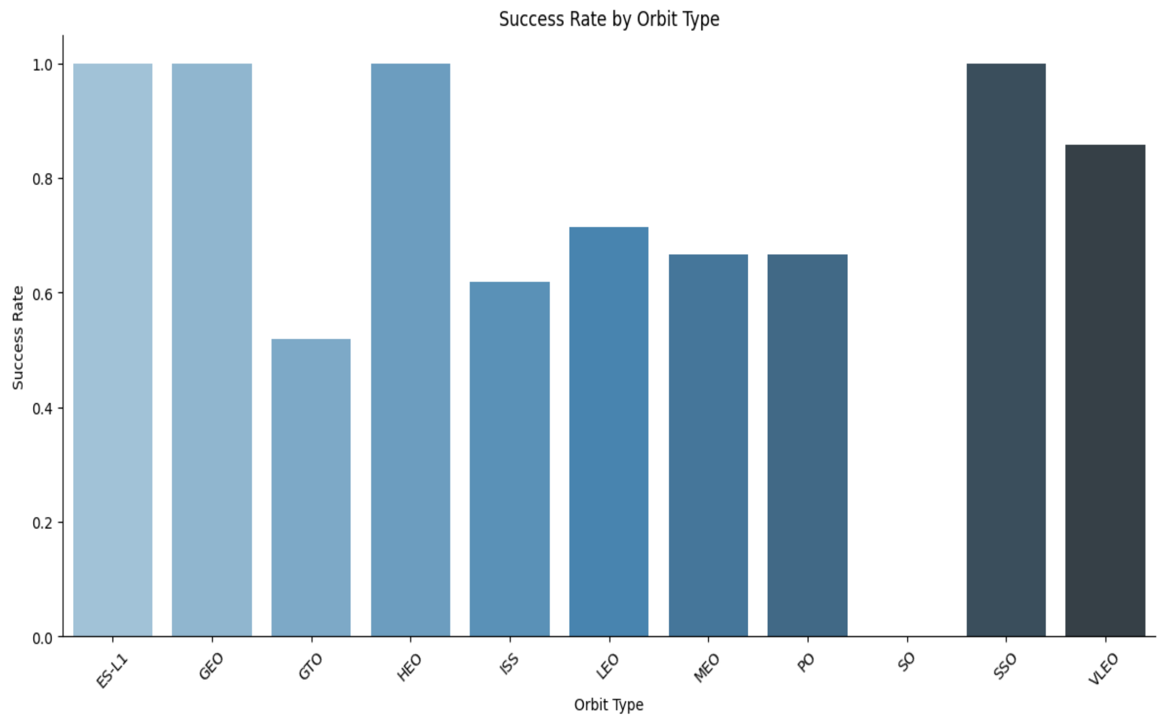


EDA with Data Visualization

- This scatter point chart see how Payload Mass Vs. Launch Site . For VAFB-SLC launchsite, there are no rockets launched for heavypayload mass (greater than 10000)
- https://github.com/Ryandran/space-y/blob/main/edadataviz.ipynb

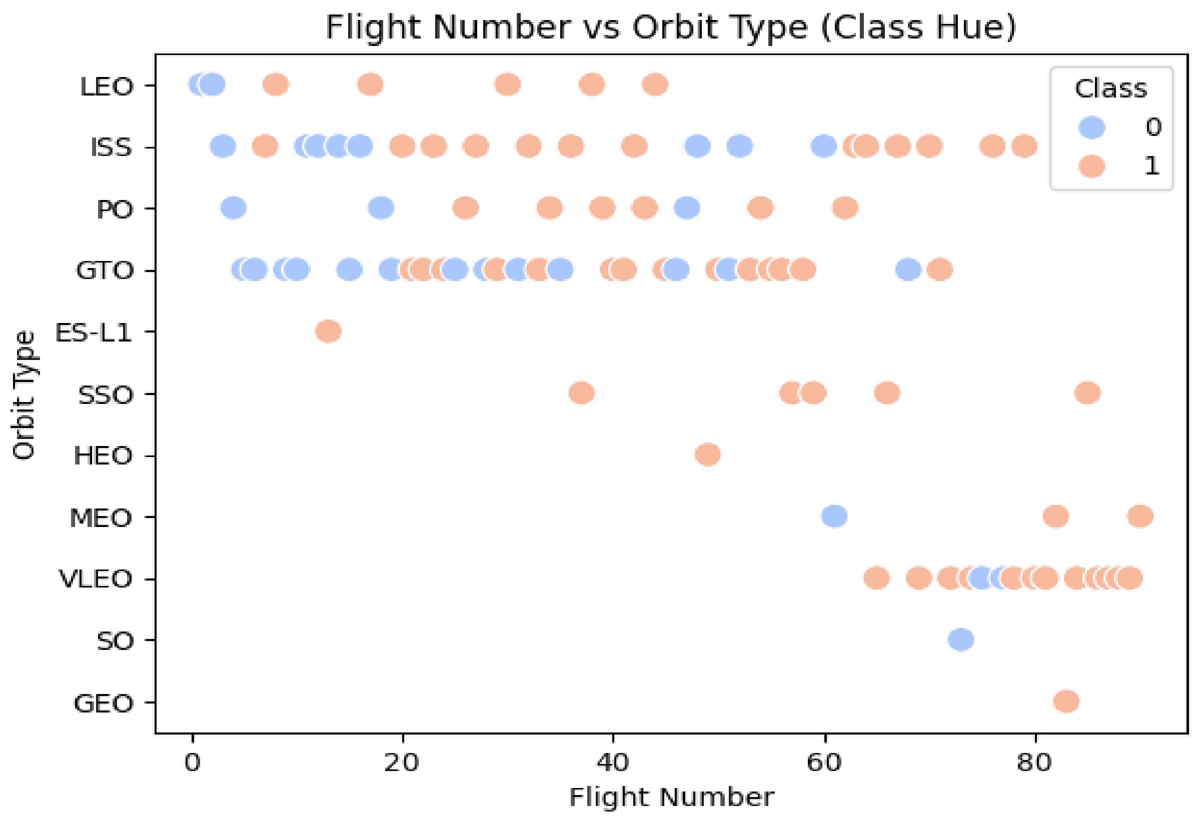


EDA with Data Visualization



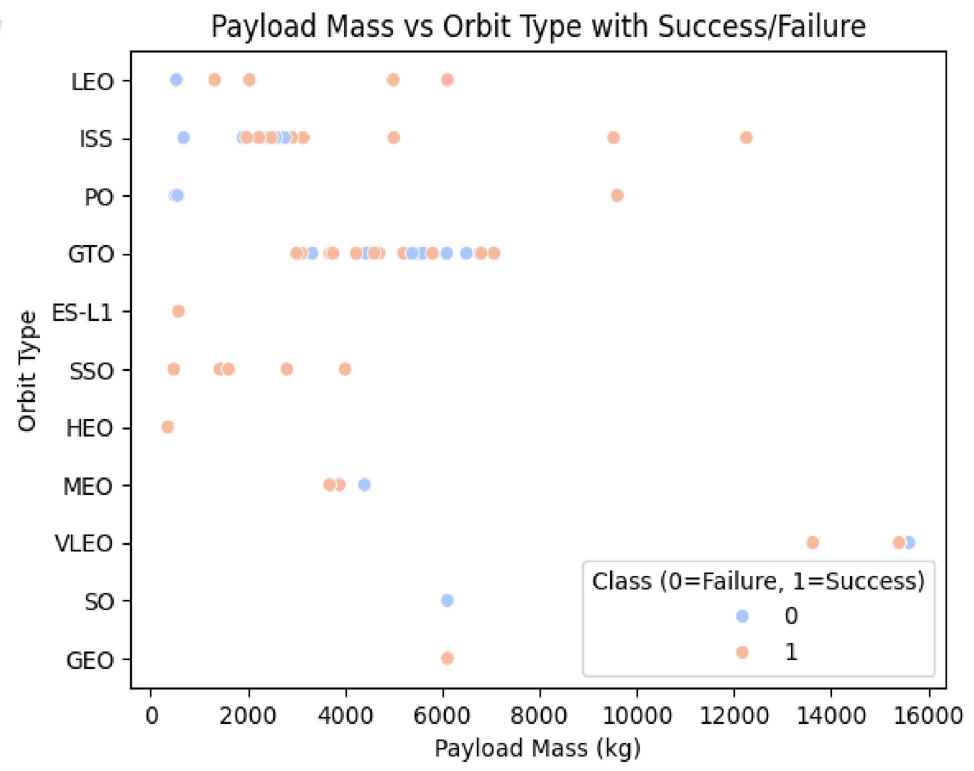
- This plotted bar chart useful to identify which orbits have the highest success rates, these are : ES-L1, GEO, HEO and SSO
- https://github.com/Ryandran/space-y/blob/main/edadataviz.ipynb

EDA with Data Visualization



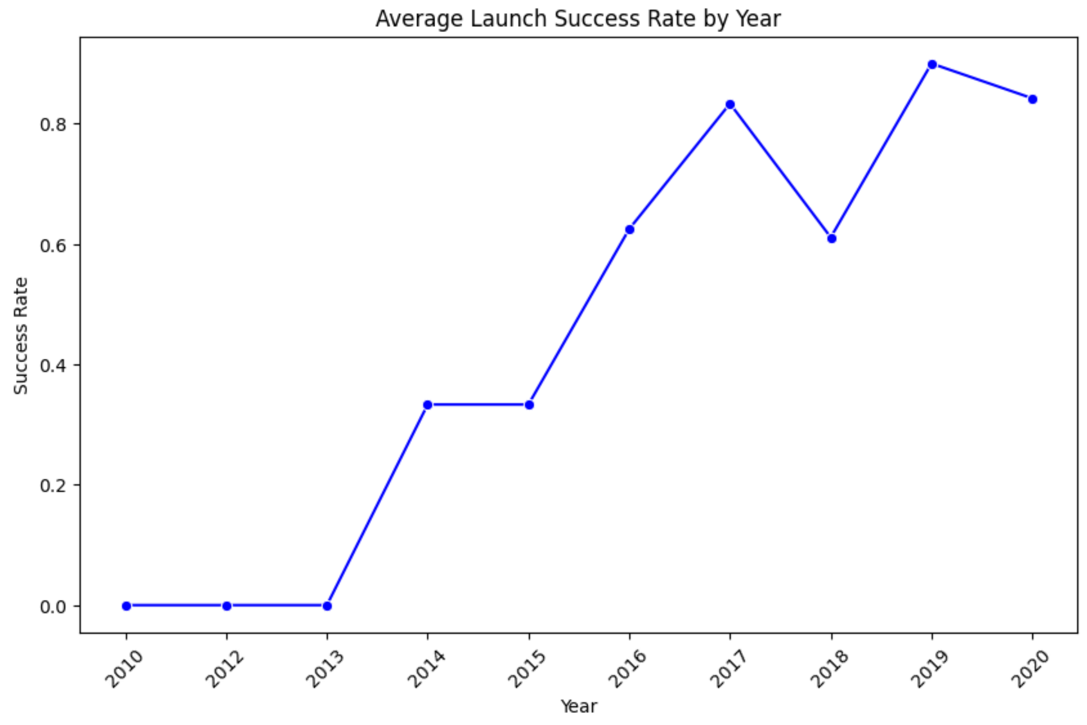
- Using this chart, You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.
- https://github.com/Ryandran/space-y/blob/main/edadataviz.ipynb

EDA with Data Visualization



- 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
- https://github.com/Ryandran/spacey/blob/main/edadataviz.ipynb

EDA with Data Visualization



- Using this chart you can observe that the sucess rate since 2013 kept increasing till 2020
- https://github.com/Ryandran/spacey/blob/main/edadataviz.ipynb

EDA with SQL

- SQL queries performed :
  - Remove blank rows from table : %sql DROP TABLE IF EXISTS SPACEXTABLE;
  - %sql create table SPACEXTABLE as select * from SPACEXTBL where Date is not null
  - Display the names of the unique launch sites in the space mission : query = '''SELECT DISTINCT Launch_SiteFROM SPACEXTBL'‘’
  - Display 5 records where launch sites begin with the string 'CCA’ : query = '''SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5’‘’
  - Display the total payload mass carried by boosters launched by NASA (CRS) : query = '''SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_MassFROM SPACEXTBLWHERE Customer = 'NASA (CRS)‘‘’
  - Display average payload mass carried by booster version F9 v1.1 : query = '''SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_MassFROM SPACEXTBLWHERE Booster_Version = 'F9 v1.1''''

EDA with SQL

- SQL queries performed :
  - List the date when the first succesful landing outcome in ground pad was achieved : query = '''SELECT MIN(Date) AS First_Successful_Landing_DateFROM SPACEXTBLWHERE Landing_Outcome = 'Success (ground pad)'''"
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000 : query = '''SELECT Booster_VersionFROM SPACEXTBLWHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000'"
  - List the total number of successful and failure mission outcomes : query = '''SELECT Mission_Outcome, COUNT(*) AS TotalFROM SPACEXTBLGROUP BY Mission_Outcome'"
  - List the names of the booster_versions which have carried the maximum payload mass. Use a subquery : query = '''SELECT Booster_Version FROM SPACEXTBLWHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)'"

EDA with SQL

- SQL queries performed :
  - 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 : query = '''SELECT substr(Date, 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_SiteFROM SPACEXTBLWHERE substr(Date, 1, 4) = '2015' AND Landing_Outcome LIKE '%Failure (drone ship)%''"
  - 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 : query = '''SELECT Landing_Outcome, COUNT(*) AS Outcome_CountFROM SPACEXTBLWHERE Date BETWEEN '2010-06-04' AND '2017-03-20'GROUP BY Landing_OutcomeORDER BY Outcome_Count DESC'"
- https://github.com/Ryandran/space-y/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- To visualize each Launch Site locations, a highlighted circle area with marker on a specific coordinate added on a map. Marker clusters can be a good way to simplify a map containing many markers having the same coordinate. Also draw a line between a launch site to its closest city, railway, highway, etc. and calculate the distance between them the launch site.

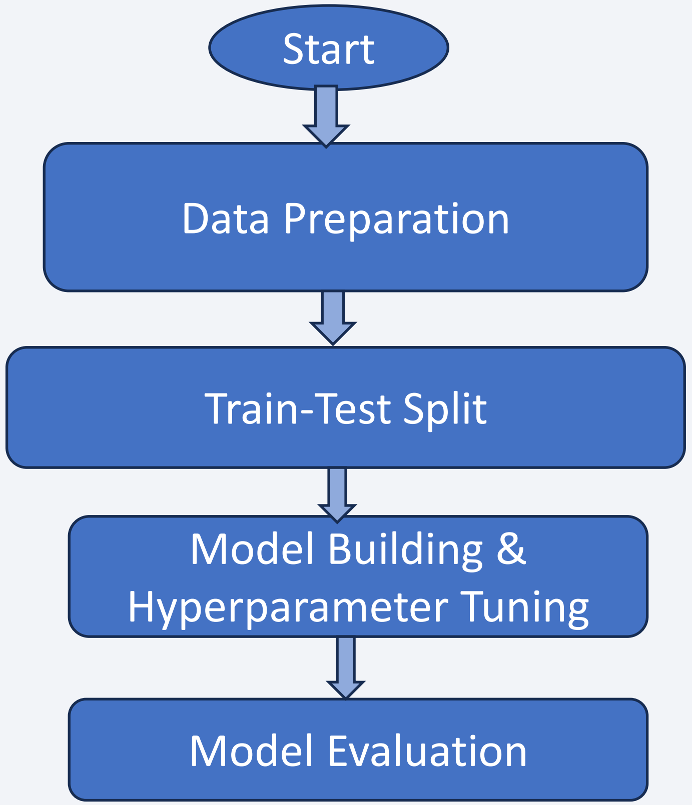
- https://github.com/Ryandran/spacey/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

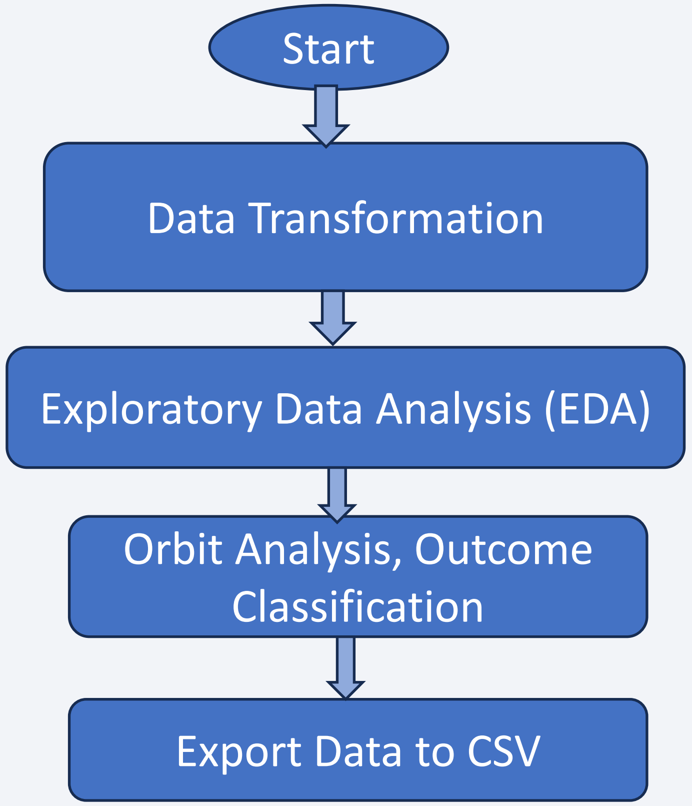
- This dashboard application contains input components such as a dropdown list and a range slider to interact with a pie chart and a scatter point chart .
- A dropdown list useful for select different launch sites and we would like to first see which one has the largest success count. Then, we would like to select one specific site and check its detailed success rate (class=0 vs. class=1).
- Using range slider, we want to find if variable payload is correlated to mission outcome. From a dashboard point of view, we want to be able to easily select different payload range and see if we can identify some visual patterns.
- https://github.com/Ryandran/space-y/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- To build the best classification model, start by preparing the data: clean and standardize features. Split the data into training and testing sets. Next, create models like Logistic Regression, SVM, or Decision Trees and , K Nearest Neighbors. Use GridSearchCV to find the optimal parameters for each model through cross-validation. Evaluate each model's performance on the test data to determine the best one. The model with the highest accuracy on the test set and best cross-validation score is selected as the final, best-performing model. This systematic approach ensures robust and optimized model performance.
- https://github.com/Ryandran/space-y/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb



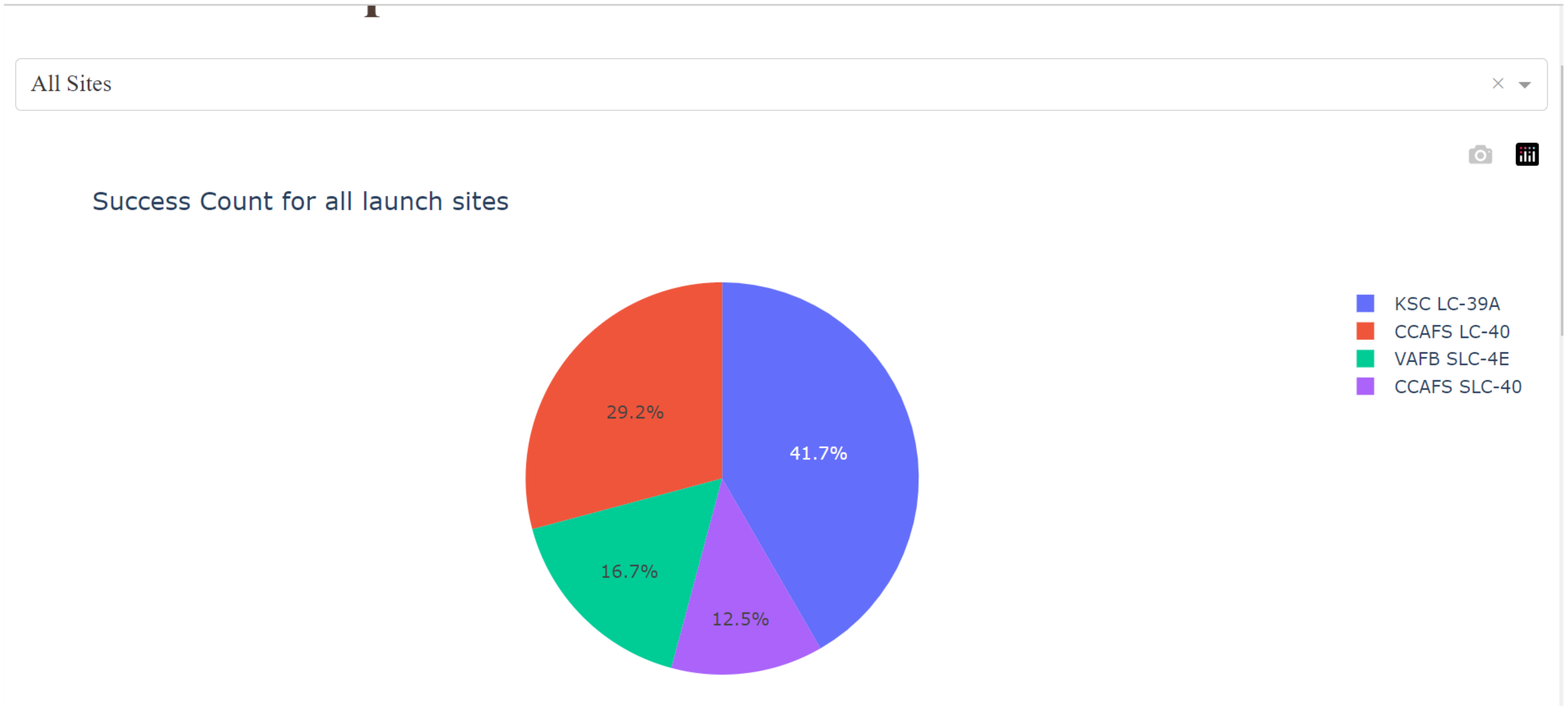
Predictive Analysis (Classification)


Start
Data Transformation
Exploratory Data Analysis (EDA)
Orbit Analysis, Outcome Classification
Export Data to CSV

Results

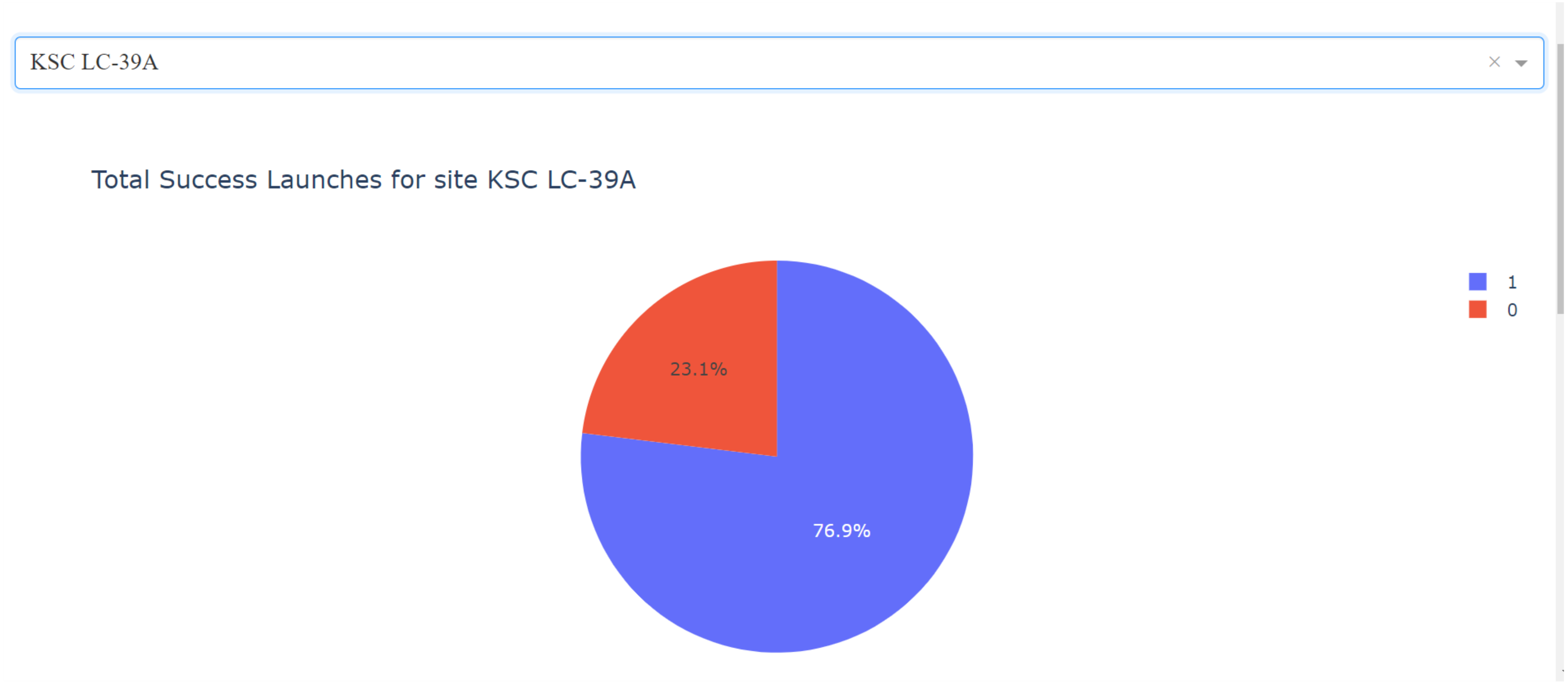
Exploratory data analysis results :

The `FlightNumber` and `Payload` variables influence launch outcomes, but increasing flight numbers don't significantly affect the launch site results. For the VAFB-SLC launch site, no rockets have been launched with heavy payloads (over 10,000). A bar chart shows that the orbits with the highest success rates are ES-L1, GEO, HEO, and SSO. In the LEO orbit, success correlates with the number of flights, while in the GTO orbit, no such relationship is evident. Heavy payloads have higher success rates for Polar, LEO, and ISS orbits. For GTO, distinguishing between successful and unsuccessful landings is challenging, but the success rate has generally increased since 2013.


All Sites

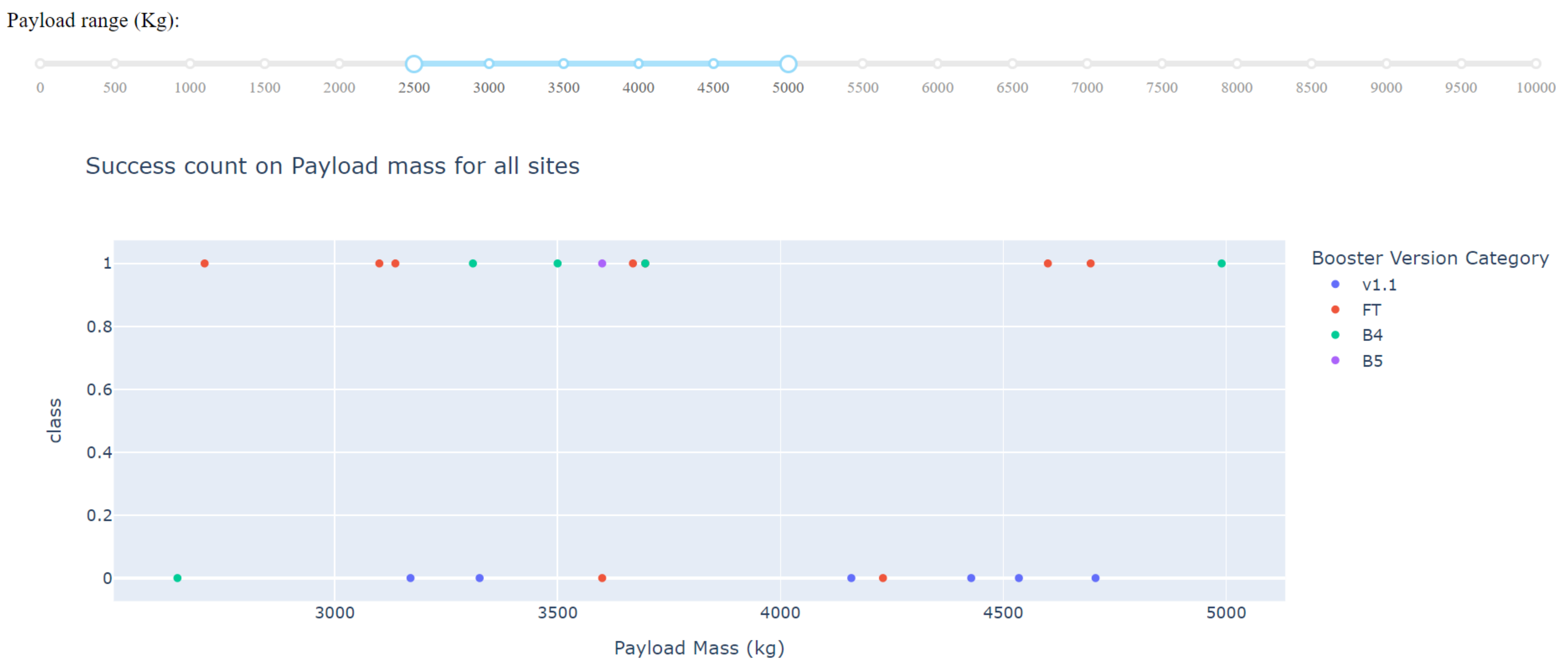
Results

Results

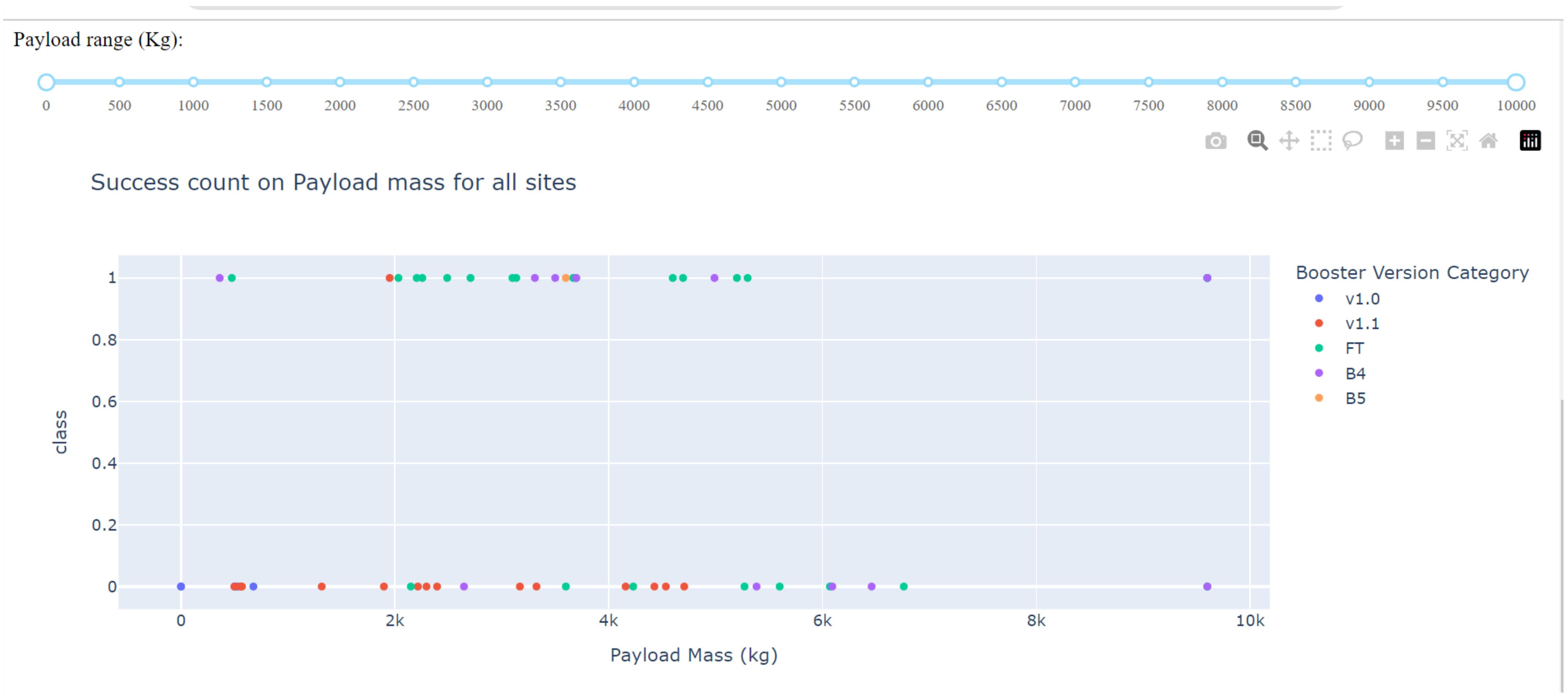

KSC LC-39A

Total Success Launches for site KSC LC-39A

Results


Payload range (Kg):

Results



Payload range (Kg):
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
5500
6000
6500
7000
7500
8000
8500
9000
9500
10000
Success count on Payload mass for all sites
class
1
0.8
0.6
0.4
0.2
0
0
2k
4k
6k
8k
10k
Payload Mass (kg)
Booster Version Category
v1.0
v1.1
FT
B4
B5

Results

Predictive analysis results :

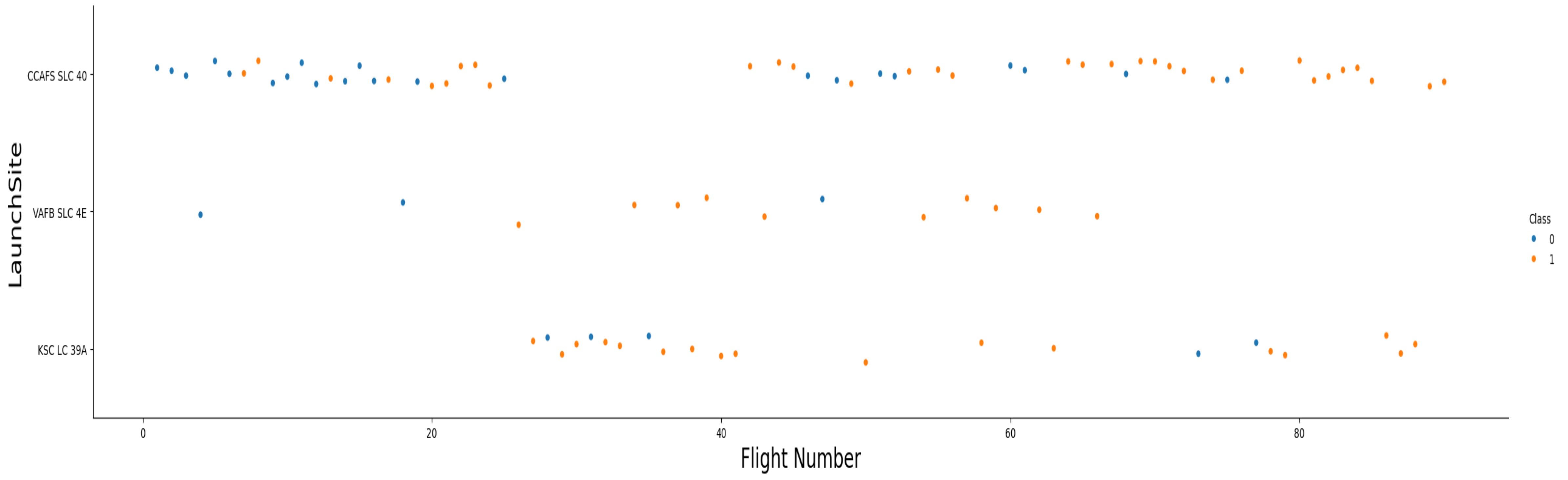
The model with the highest test accuracy and best cross-validation score was selected as the best-performing classification model. Decision tree classifier perform best, with cccuracy 0.86 and accuracy on test data using the method score : 0.94

Section 2

Insights drawn from EDA

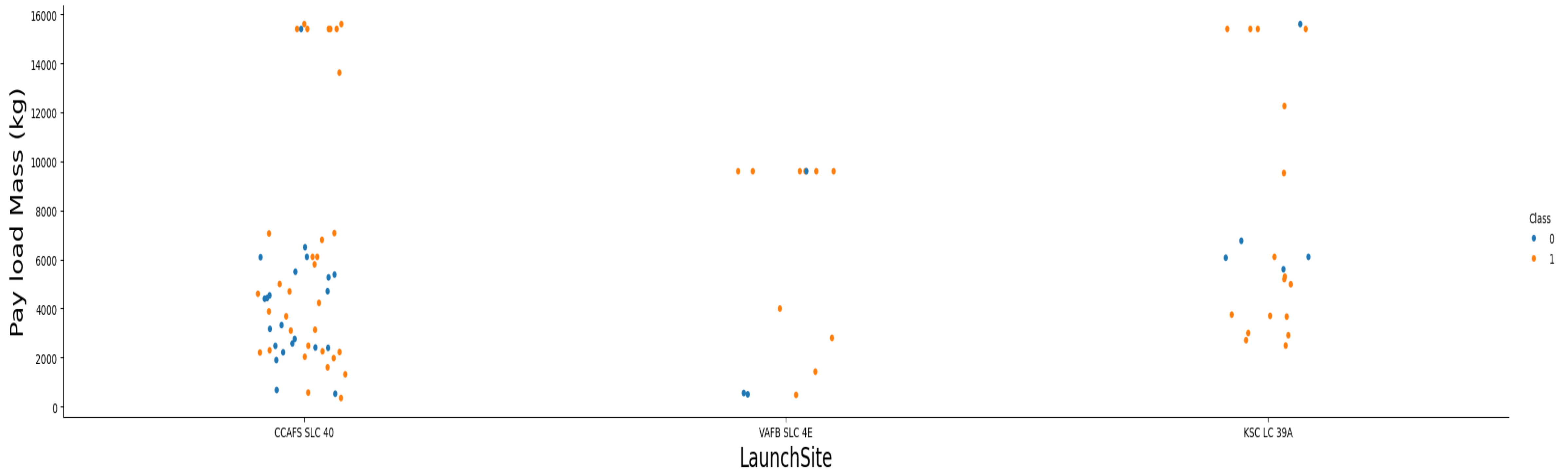
Flight Number vs. Launch Site

This chart see how the increase in flight numbers does not have a significant impact on the outcome of the launch site.

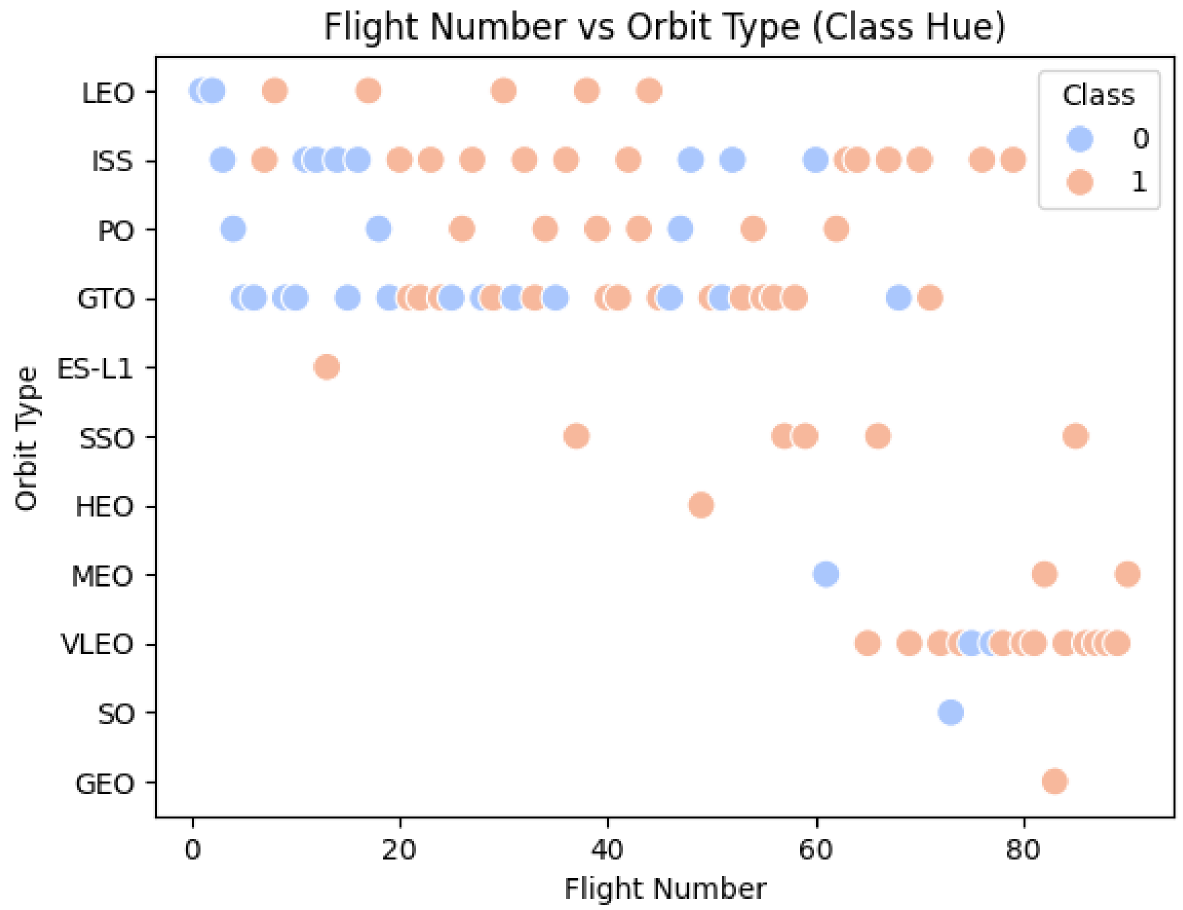


Payload vs. Launch Site

This scatter point chart see how Payload Mass Vs. Launch Site . For VAFB-SLC launchsite, there are no rockets launched for heavypayload mass (greater than 10000)

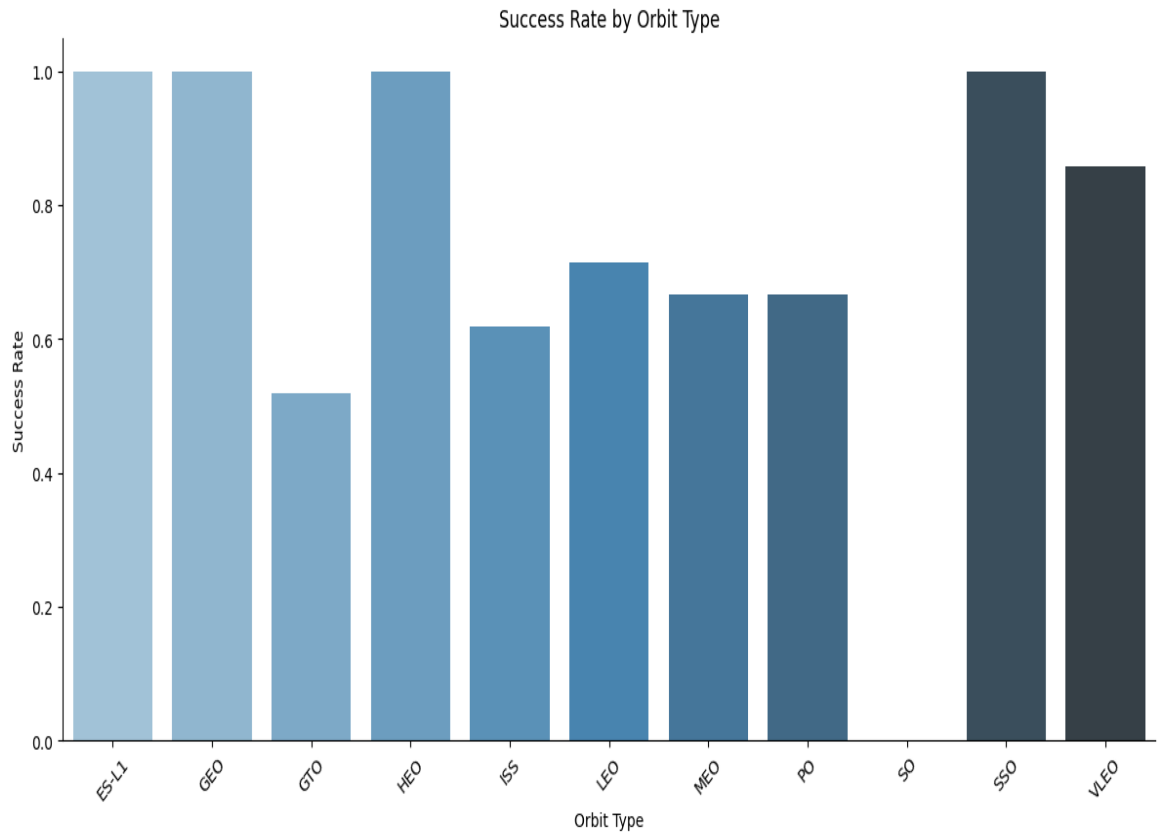


Success Rate vs. Orbit Type



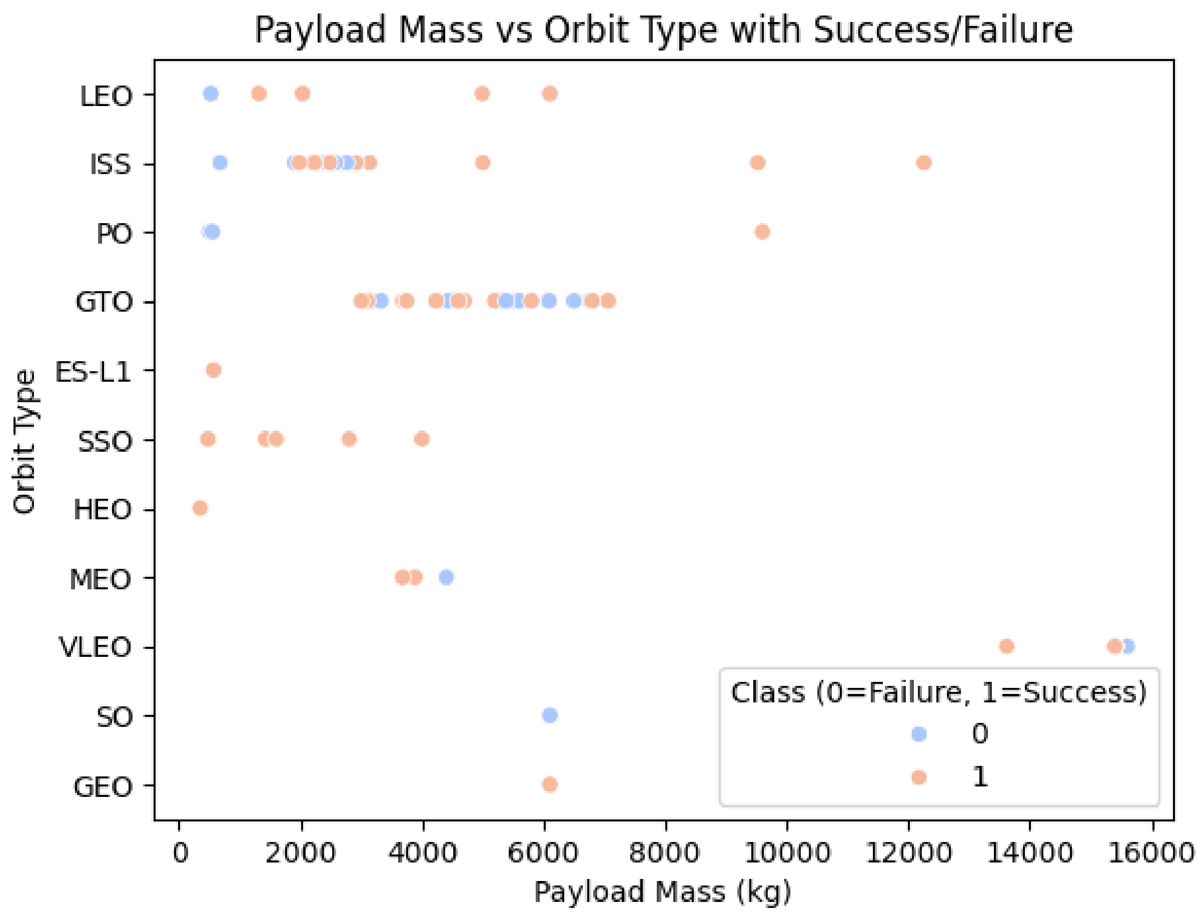
Using this chart, we can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

Flight Number vs. Orbit Type



This plotted bar chart useful to identify which orbits have the highest success rates, these are : ES-L1, GEO, HEO and SSO

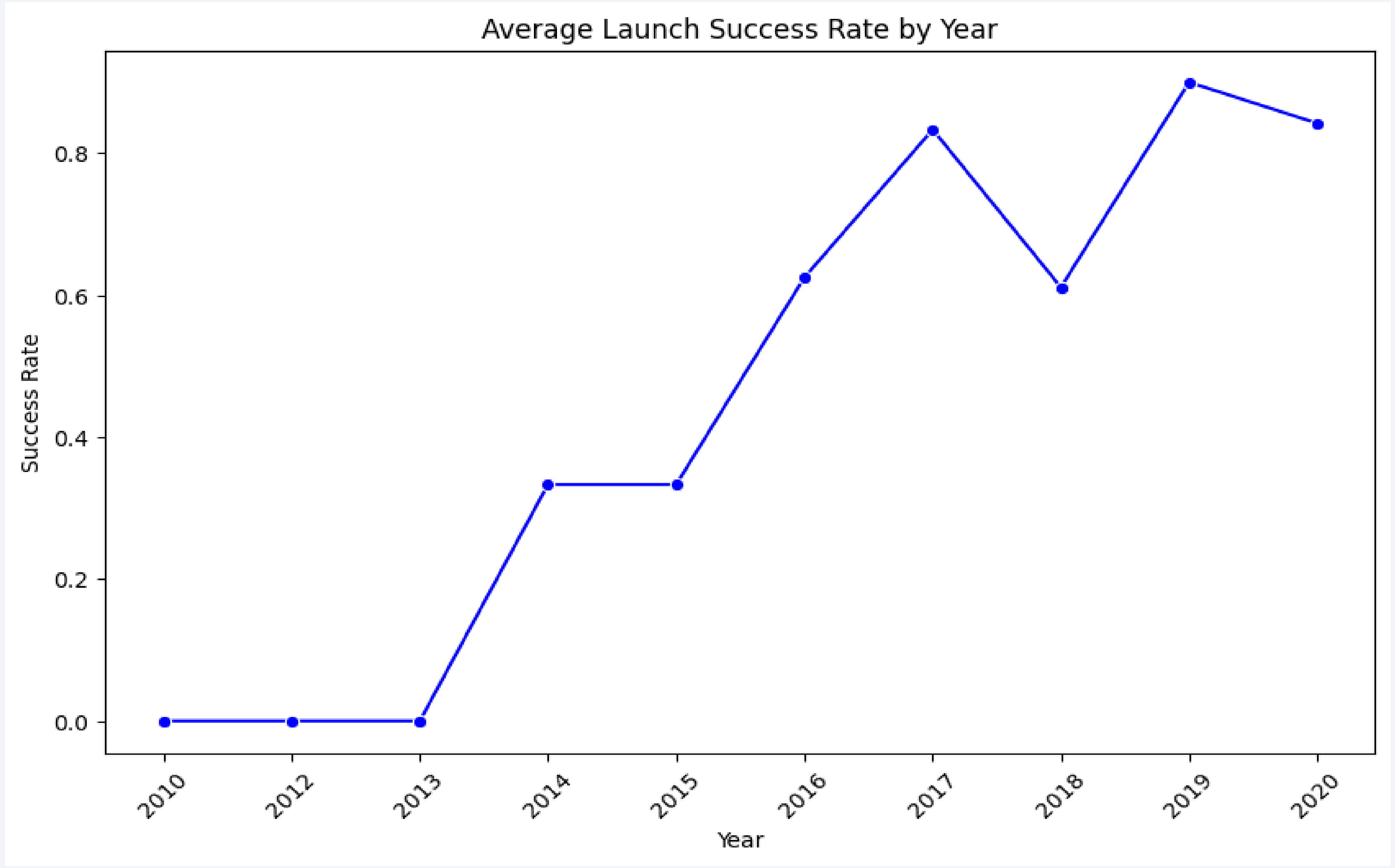
Payload vs. Orbit Type



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

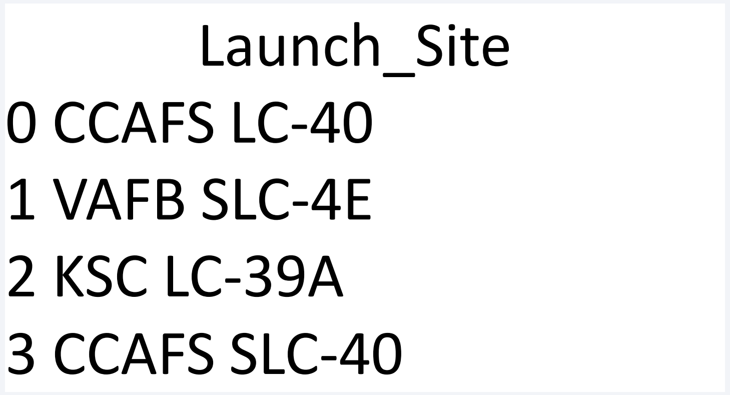
Launch Success Yearly Trend



Using this chart you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

- The names of the unique launch sites



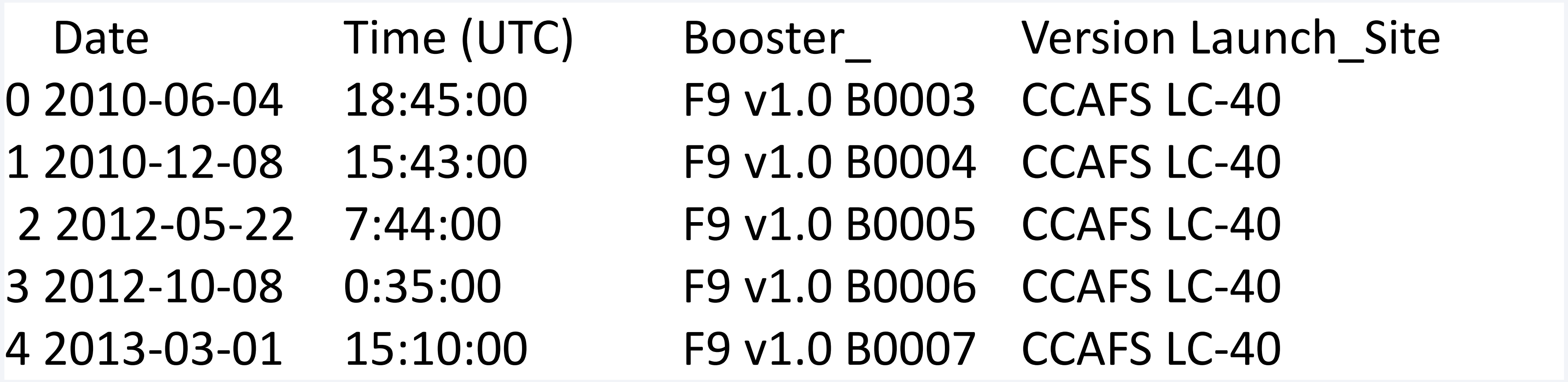
```
          Launch_Site
0 CCAFS LC-40
1 VAFB SLC-4E
2 KSC LC-39A
3 CCAFS SLC-40
```

-
  ```
   query = '''SELECT DISTINCT Launch_Site FROM SPACEXTBL’’’
  unique_launch_sites = pd.read_sql_query(query, con)
  print(unique_launch_sites)
  ```

- This code executes a SQL query to retrieve distinct `Launch_Site` values from the `SPACEXTBL` table, then loads the result into a Pandas DataFrame called `unique_launch_sites`, and finally prints the DataFrame.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- query = '''SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5''’
  records_with_cca_launch_sites = pd.read_sql_query(query, con)
  print(records_with_cca_launch_sites)



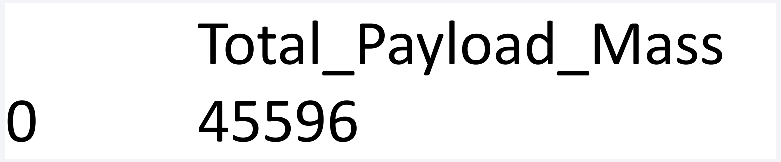
```
    Date          Time (UTC)      Booster_        Version Launch_Site
0 2010-06-04      18:45:00        F9 v1.0 B0003   CCAFS LC-40
1 2010-12-08      15:43:00        F9 v1.0 B0004   CCAFS LC-40
 2 2012-05-22     7:44:00         F9 v1.0 B0005   CCAFS LC-40
3 2012-10-08      0:35:00         F9 v1.0 B0006   CCAFS LC-40
4 2013-03-01      15:10:00        F9 v1.0 B0007   CCAFS LC-40
```

This query selects all columns from the `SPACEXTBL` table, where the `Launch_Site` begins with 'CCA', and limits the result to 5 rows. It then reads the SQL query into a pandas DataFrame (`records_with_cca_launch_sites`) and prints the result.

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- query = '''SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass FROM SPACEXTBLWHERE Customer = 'NASA (CRS)‘“’

```
total_payload_mass = pd.read_sql_query(query, con)
print(total_payload_mass)
```



```
   Total_Payload_Mass
0  45596
```

- This query selects the total payload mass from the "SPACEXTBL" table where the customer is "NASA (CRS)." The result is stored in `total_payload_mass` using pandas' `read_sql_query()` function, which executes the SQL query on the database connection `con`. Finally, the result is printed.

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- query = '''SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass FROM SPACEXTBLWHERE Booster_Version = 'F9 v1.1‘”’

  average_payload_mass = pd.read_sql_query(query, con)

  print(average_payload_mass)

```
        Average_Payload_Mass
0       2928.4
```

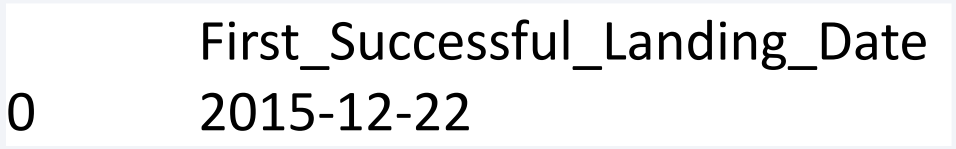
- This SQL query calculates the average payload mass (`PAYLOAD_MASS__KG_`) for SpaceX booster version 'F9 v1.1' from the `SPACEXTBL` table. It runs in Python using `pd.read_sql_query`, which retrieves the result into a pandas DataFrame (`average_payload_mass`).

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- query = '''SELECT MIN(Date) AS First_Successful_Landing_Date FROM SPACEXTBLWHERE Landing_Outcome = 'Success (ground pad)'“‘’

  first_successful_landing_date = pd.read_sql_query(query, con)

  print(first_successful_landing_date)

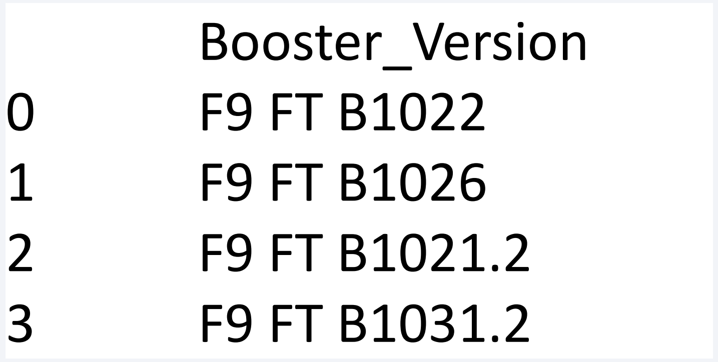


```
        First_Successful_Landing_Date
0       2015-12-22
```

- This query retrieves the earliest date (`First_Successful_Landing_Date`) from the `SPACEXTBL` table where the landing outcome was a success on a ground pad. The result is stored in the `first_successful_landing_date` DataFrame and printed.

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



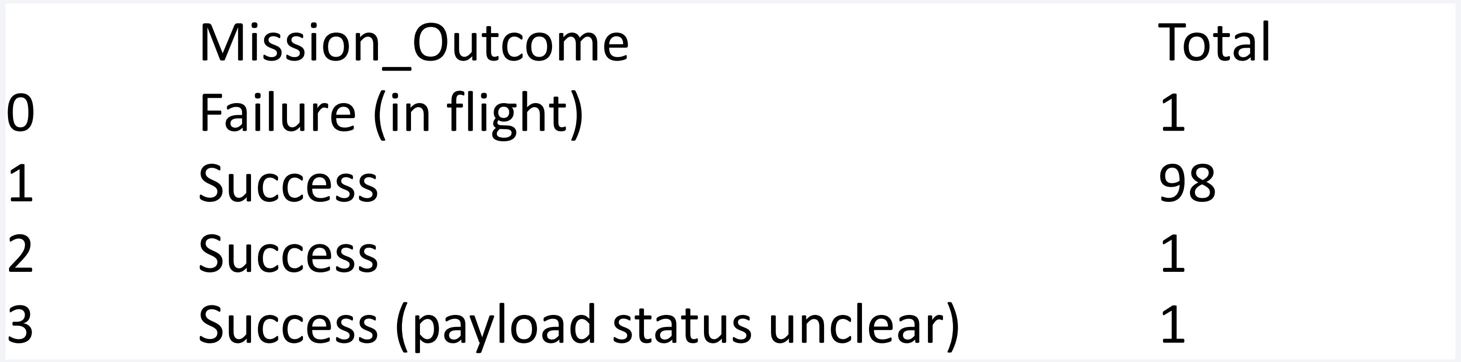
```
          Booster_Version
0         F9 FT B1022
1         F9 FT B1026
2         F9 FT B1021.2
3         F9 FT B1031.2
```

- query = '''SELECT Booster_Version FROM SPACEXTBLWHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000’‘’
  boosters = pd.read_sql_query(query, con)
  print(boosters)

- This SQL query selects the `Booster_Version` from the `SPACEXTBL` table where the `Landing_Outcome` is 'Success (drone ship)' and the `PAYLOAD_MASS__KG_` is between 4000 and 6000. The result is stored in the `boosters` DataFrame and printed using Python's `pandas` library.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes



```
        Mission_Outcome                       Total
0       Failure (in flight)                   1
1       Success                               98
2       Success                               1
3       Success (payload status unclear)      1
```

- query = '''SELECT Mission_Outcome, COUNT(*) AS Total FROM SPACEXTBLGROUP BY Mission_Outcome’‘’
  mission_outcomes = pd.read_sql_query(query, con)
  print(mission_outcomes)

- This SQL query selects the `Mission_Outcome` column from the `SPACEXTBL` table and counts how many times each outcome appears. The results are grouped by `Mission_Outcome`, then retrieved into a pandas DataFrame (`mission_outcomes`) and printed.

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- query = '''SELECT Booster_Version FROM SPACEXTBLWHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_)   FROM SPACEXTBL)'''
  booster_versions_max_payload = pd.read_sql_query(query, con)
  print(booster_versions_max_payload)
- This SQL query selects the `Booster_Version` from the `SPACEXTBL` table where the `PAYLOAD_MASS__KG_` is the maximum. The result is stored in `booster_versions_max_payload` using `pd.read_sql_query`, which is then printed.

```
        Booster_Version
0       F9 B5 B1048.4
1       F9 B5 B1049.4
2       F9 B5 B1051.3
3       F9 B5 B1056.4
4       F9 B5 B1048.5
5       F9 B5 B1051.4
6       F9 B5 B1049.5
7       F9 B5 B1060.2
8       F9 B5 B1058.3
9       F9 B5 B1051.6
10      F9 B5 B1060.3
11      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

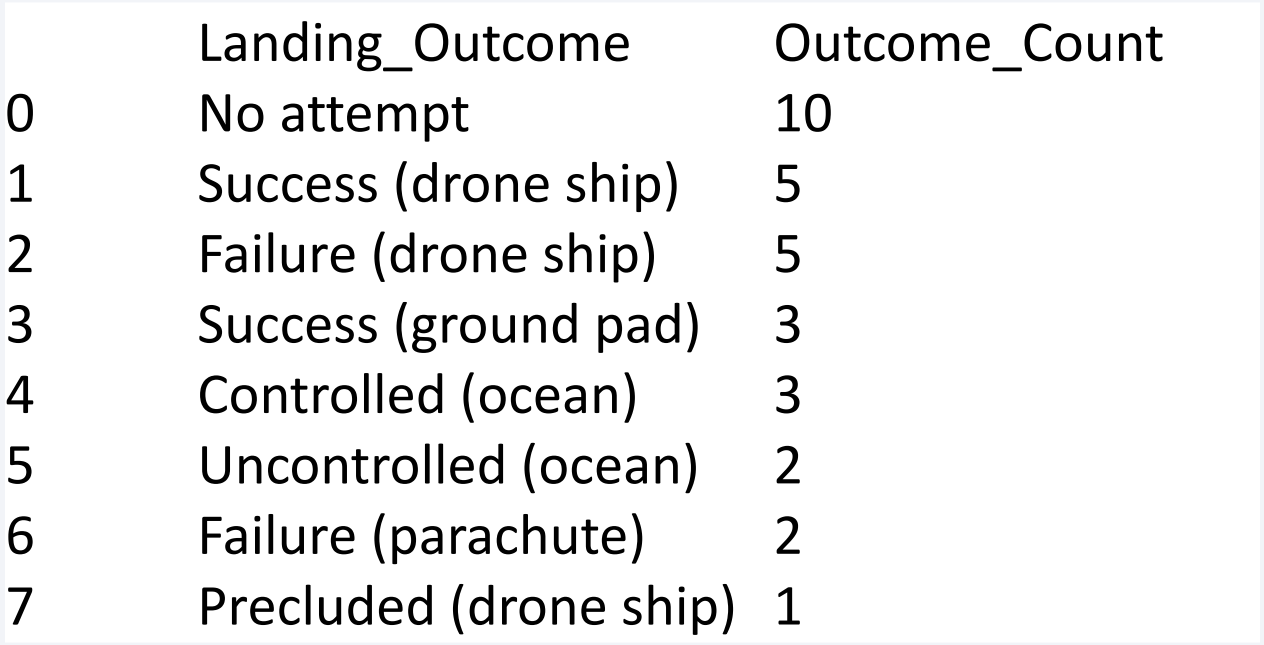
	Month	Landing_Outcome	Booster_Version	Launch_Site
0	01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
1	04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

-
  ```
  query = '''SELECT      substr(Date, 6, 2) AS Month,     Landing_Outcome,
  Booster_Version,     Launch_Site FROM SPACEXTBLWHERE substr(Date, 1, 4) = '2015'
  AND Landing_Outcome LIKE '%Failure (drone ship)%‘’’
   records_2015 = pd.read_sql_query(query, con)
   print(records_2015)
  ```

- This query selects the month, landing outcome, booster version, and launch site from the `SPACEXTBL` table for launches in 2015 that had a landing failure on a drone ship. It retrieves the data using `pd.read_sql_query()` and prints the results.

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



	Landing_Outcome	Outcome_Count
0	No attempt	10
1	Success (drone ship)	5
2	Failure (drone ship)	5
3	Success (ground pad)	3
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Failure (parachute)	2
7	Precluded (drone ship)	1

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

- query = '''SELECT   Landing_Outcome,   COUNT(*) AS Outcome_CountFROM SPACEXTBLWHERE Date BETWEEN '2010-06-04' AND '2017-03-20'GROUP BY Landing_OutcomeORDER BY Outcome_Count DESC''’
  landing_outcome_counts = pd.read_sql_query(query, con)
  print(landing_outcome_counts)

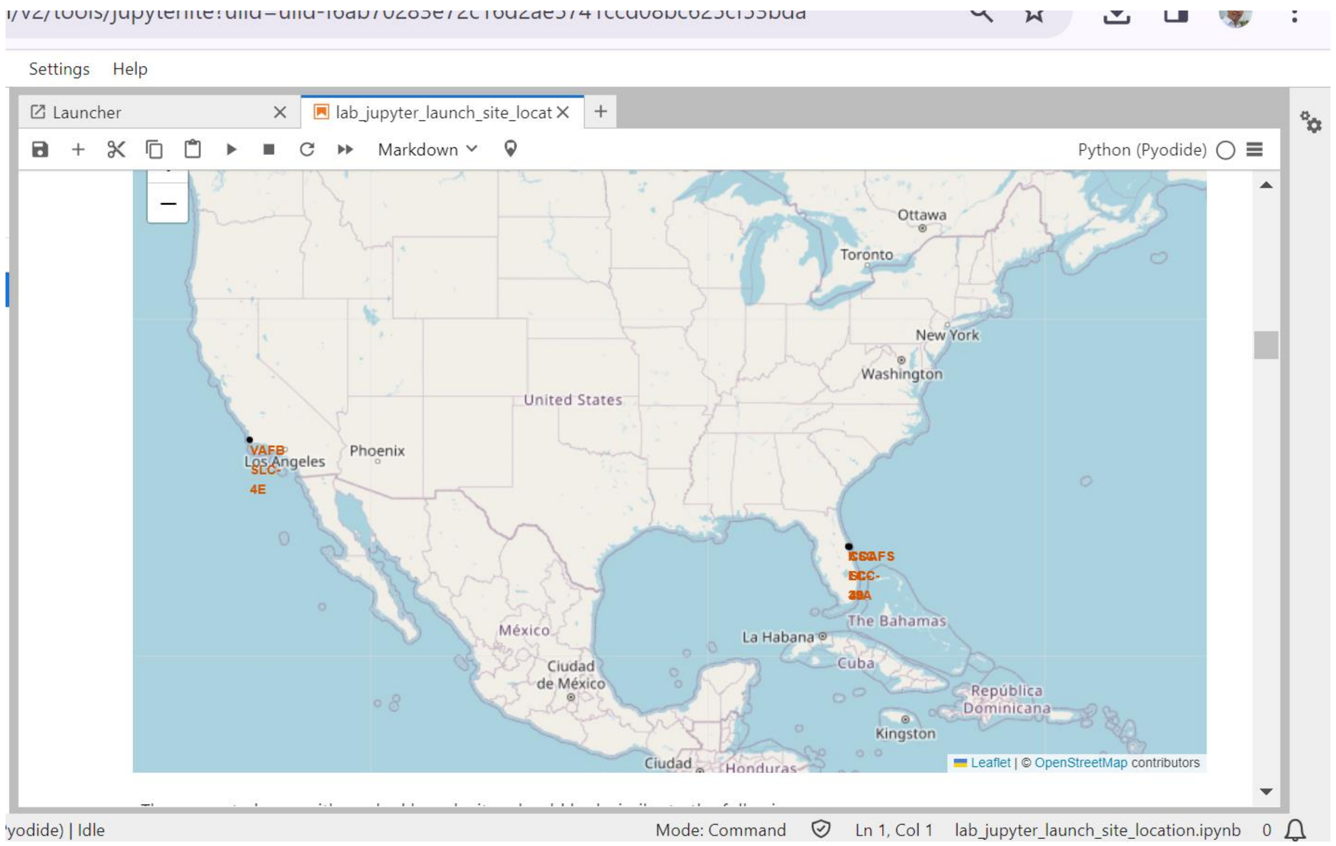
- This SQL query counts the number of landing outcomes from the `SPACEXTBL` table between June 4, 2010, and March 20, 2017. It groups the results by `Landing_Outcome`, orders them by the count in descending order, and stores the result in a DataFrame.

Section 3

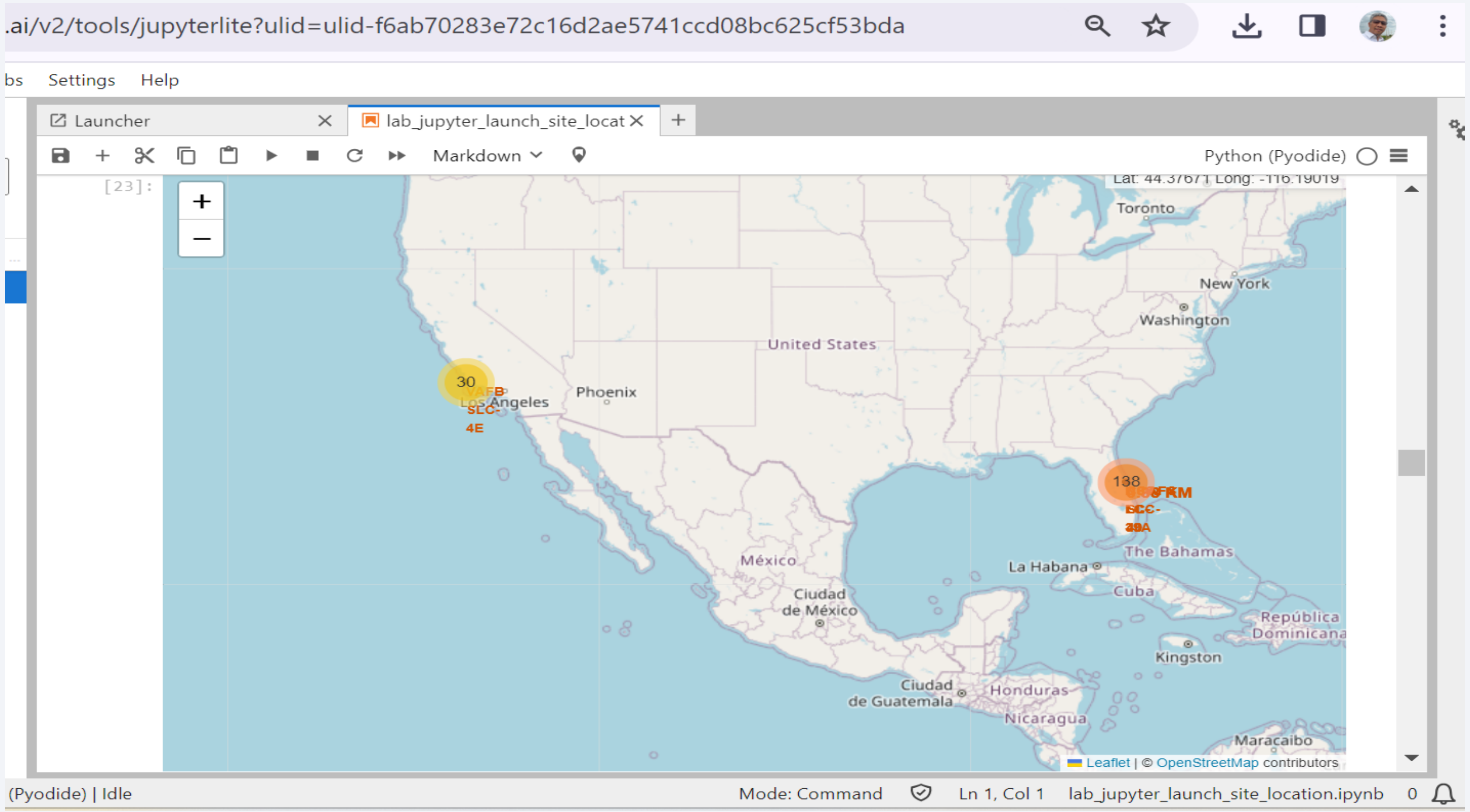
Launch Sites Proximities Analysis

Launch Sites Locations On The Map

- The Folium map centered on average launch site coordinates. It adds circles around each site with a 1,000-meter radius and markers labeled with launch site names, enhancing visualization of the launch sites' locations.

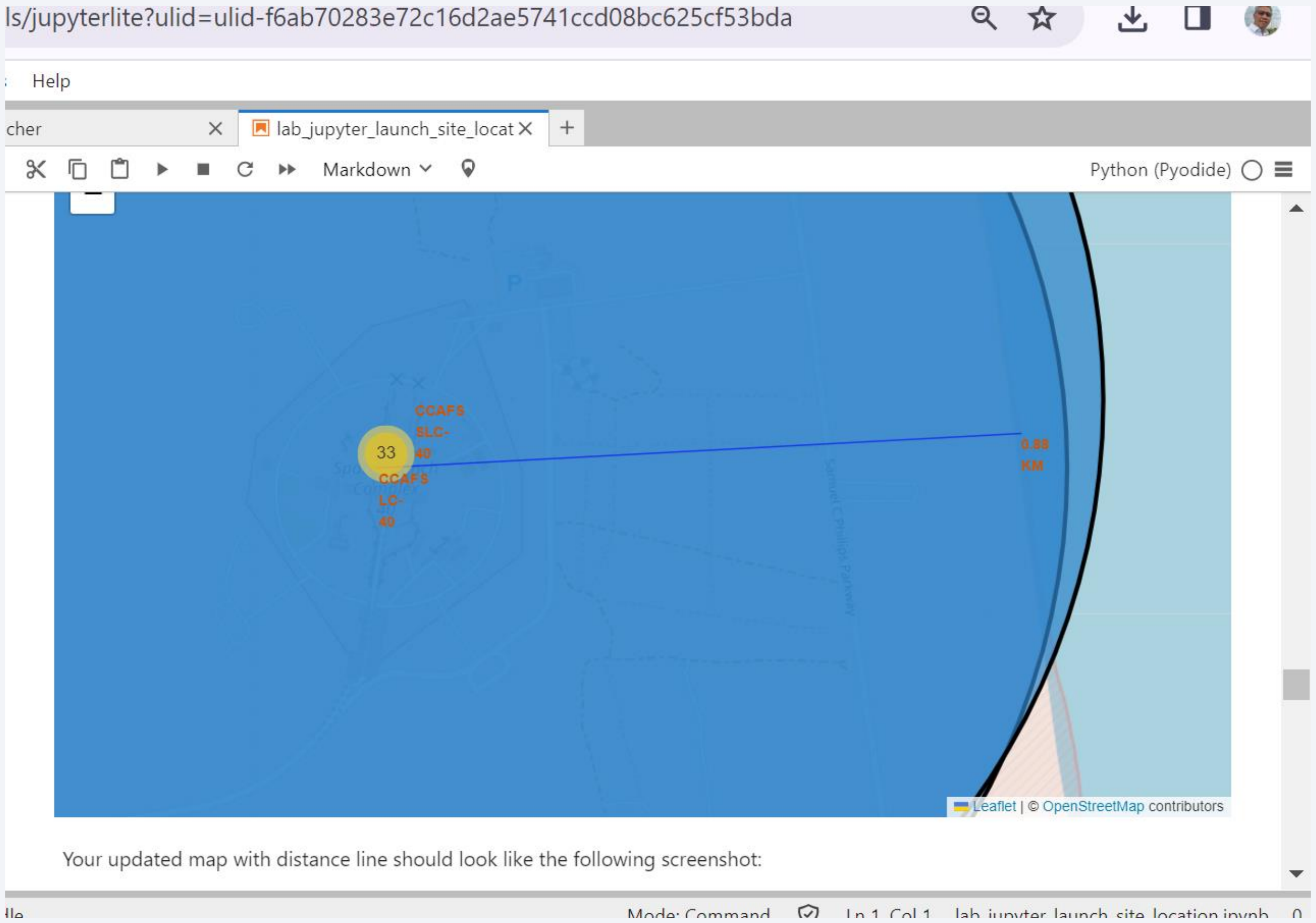


Map markers for all launch outcomes



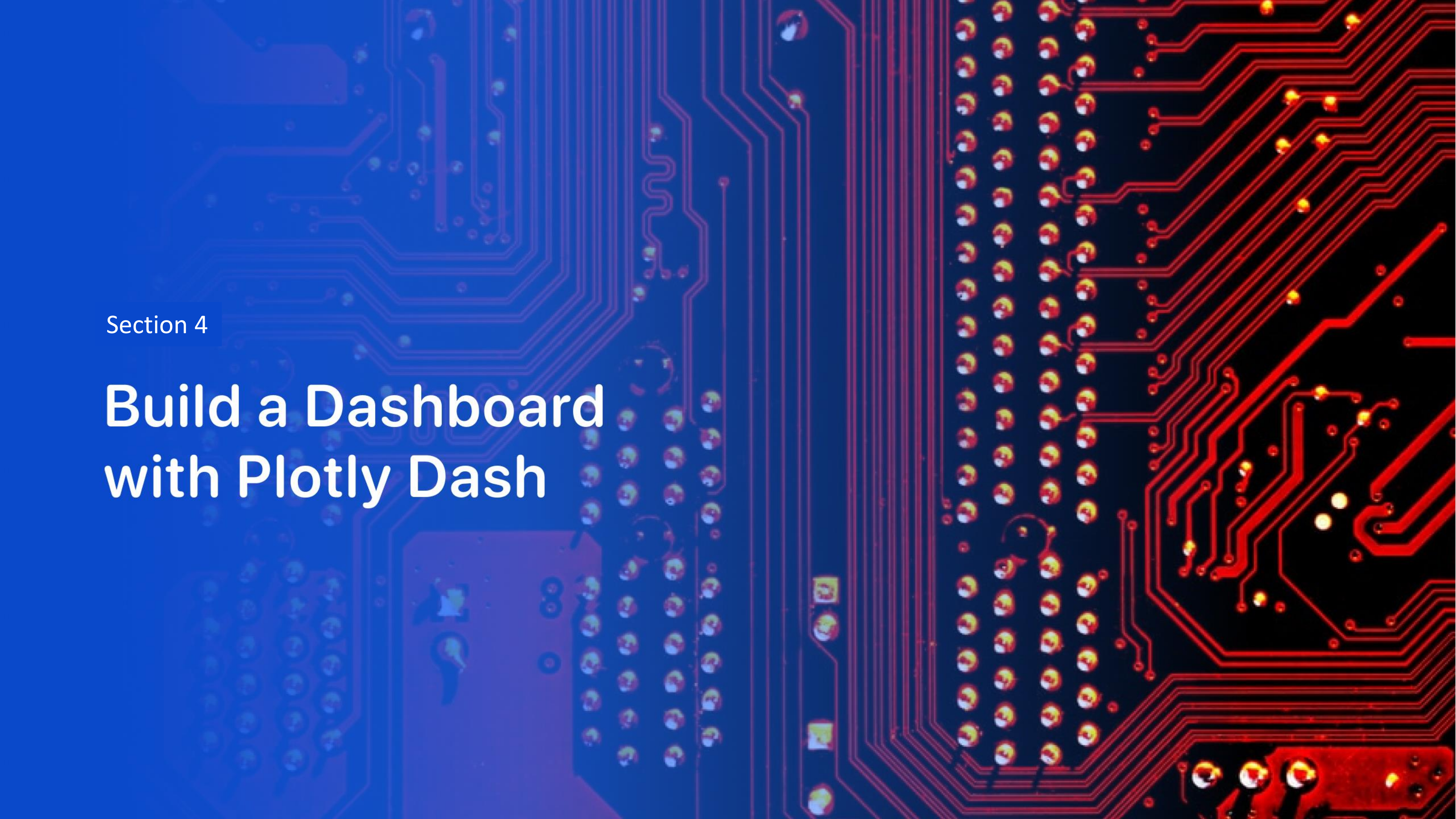
- Each marker represents a launch, with colors indicating success or failure, and is positioned based on latitude and longitude.

Draw a Polyline between a launch site to the coastline point


ls/jupyterlite?ulid=ulid-f6ab70283e72c16d2ae5741ccd08bc625cf53bda
Help
cher
lab_jupyter_launch_site_locat
Markdown
Python (Pyodide)
33
CCAFS SLC-40
CCAFS LC-40
0.88 KM
Leaflet | © OpenStreetMap contributors
Mode: Command
Ln 1, Col 1
lab_jupyter_launch_site_location.ipynb

Your updated map with distance line should look like the following screenshot:

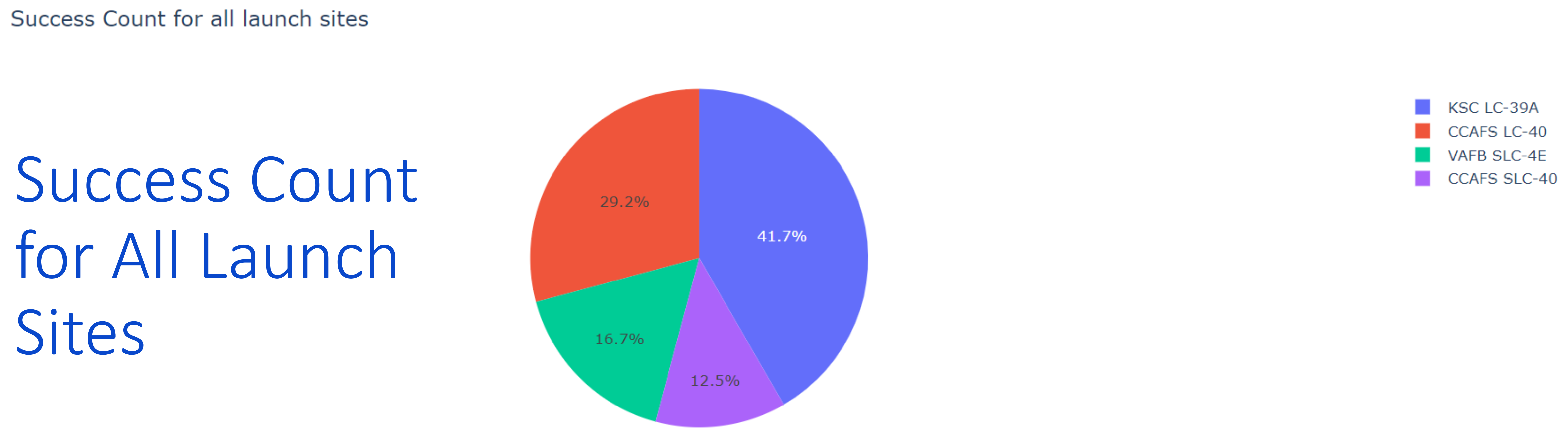
- Creates a blue polyline on a Folium map, connecting a coastline point and a launch site, visually representing the path between the two locations.



Section 4

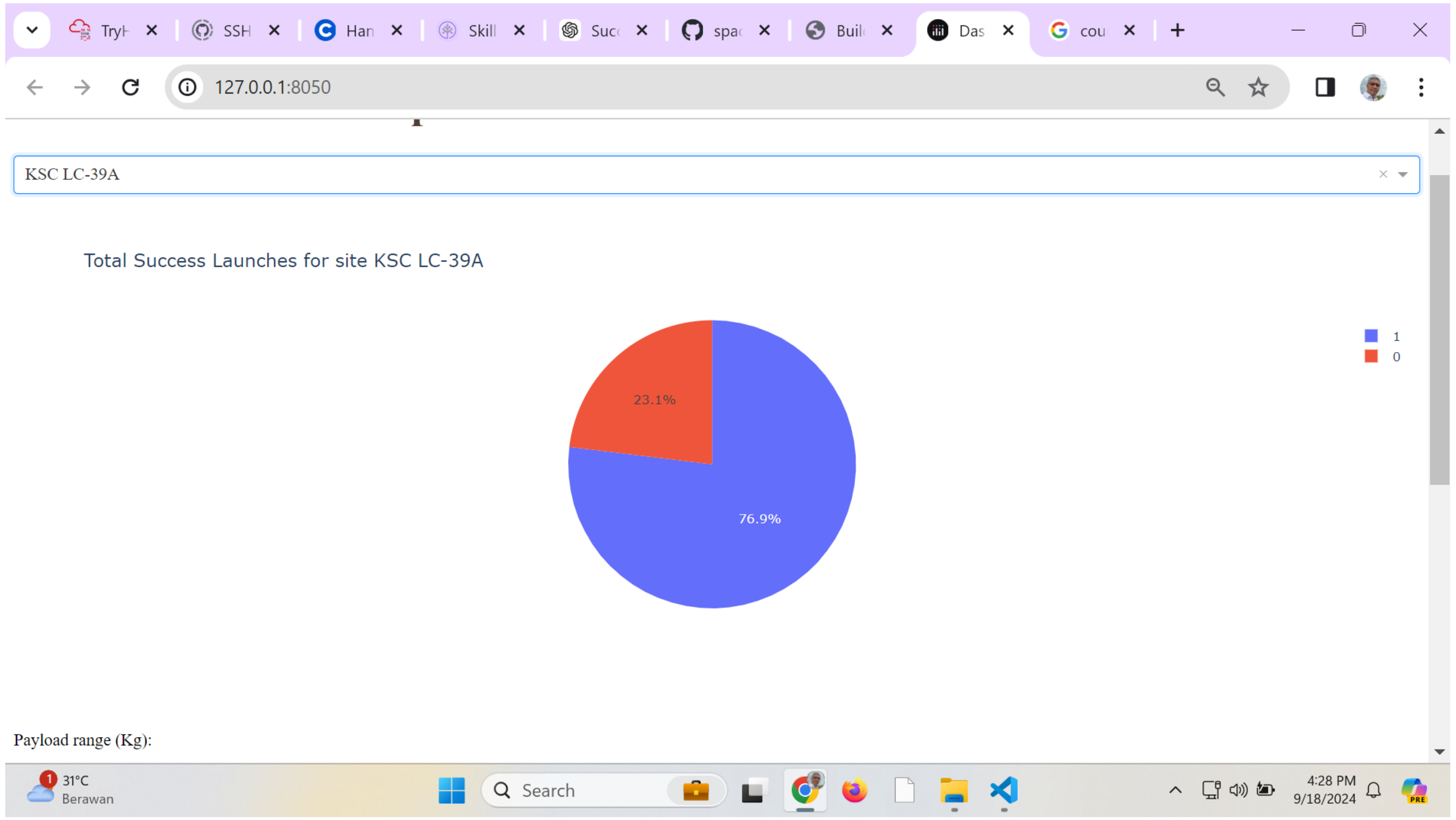
Build a Dashboard with Plotly Dash

Success Count for All Launch Sites



- A callback function that updates a pie chart based on user input from a dropdown. If all sites are selected, it displays total success launches. The pie chart show that KSC LC-39A site has the largest successful launches (41.7 %)

Launch site with highest launch success rate

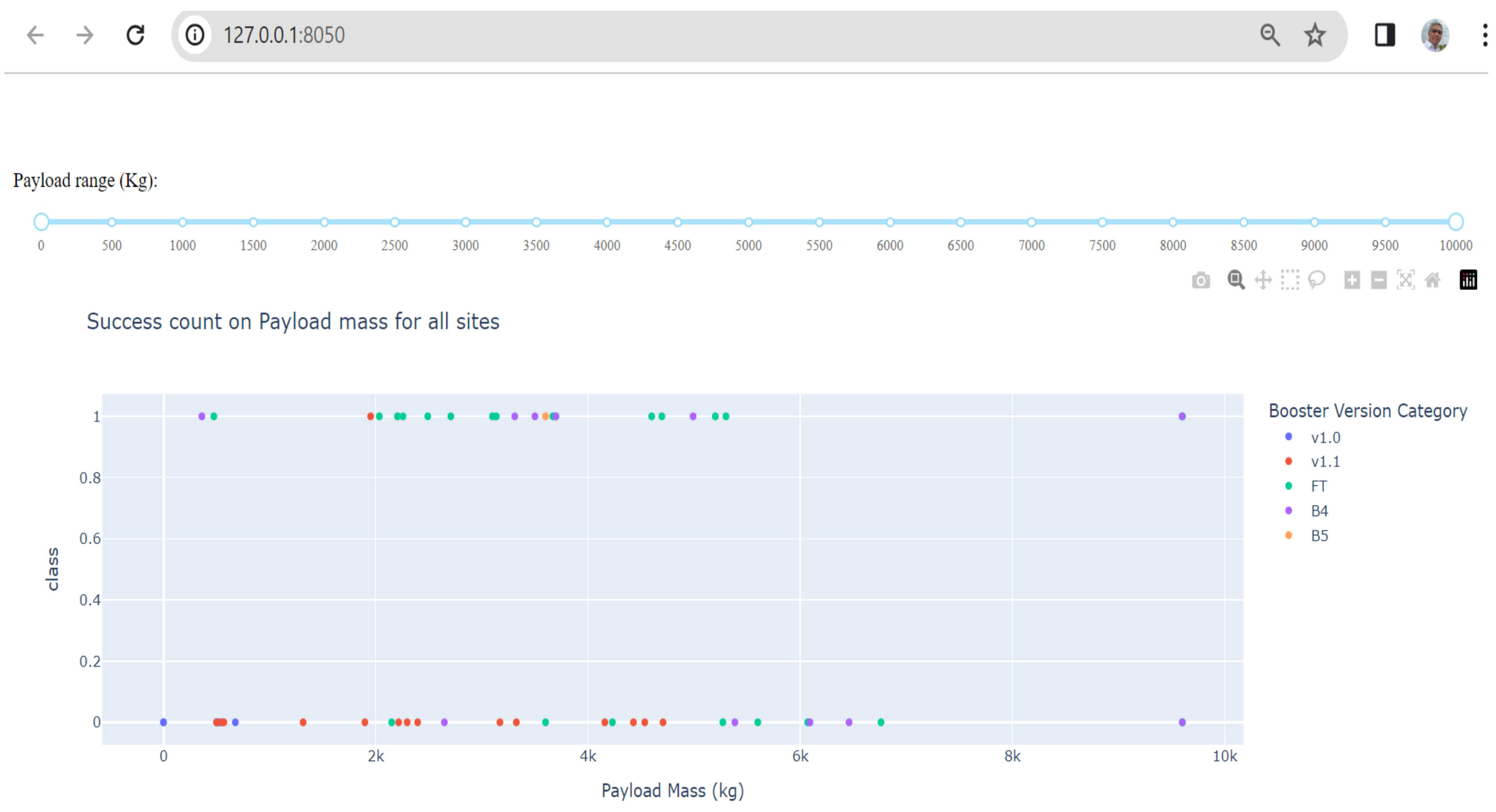

KSC LC-39A

Total Success Launches for site KSC LC-39A

Payload range (Kg):

- On this interactive dashboard if a specific site is chosen, it filters the data to show success and failure counts for that site. The pie chart of KSC LC-39A show the highest launch success rate (76.9 %)

Success count on Payload mass for all sites

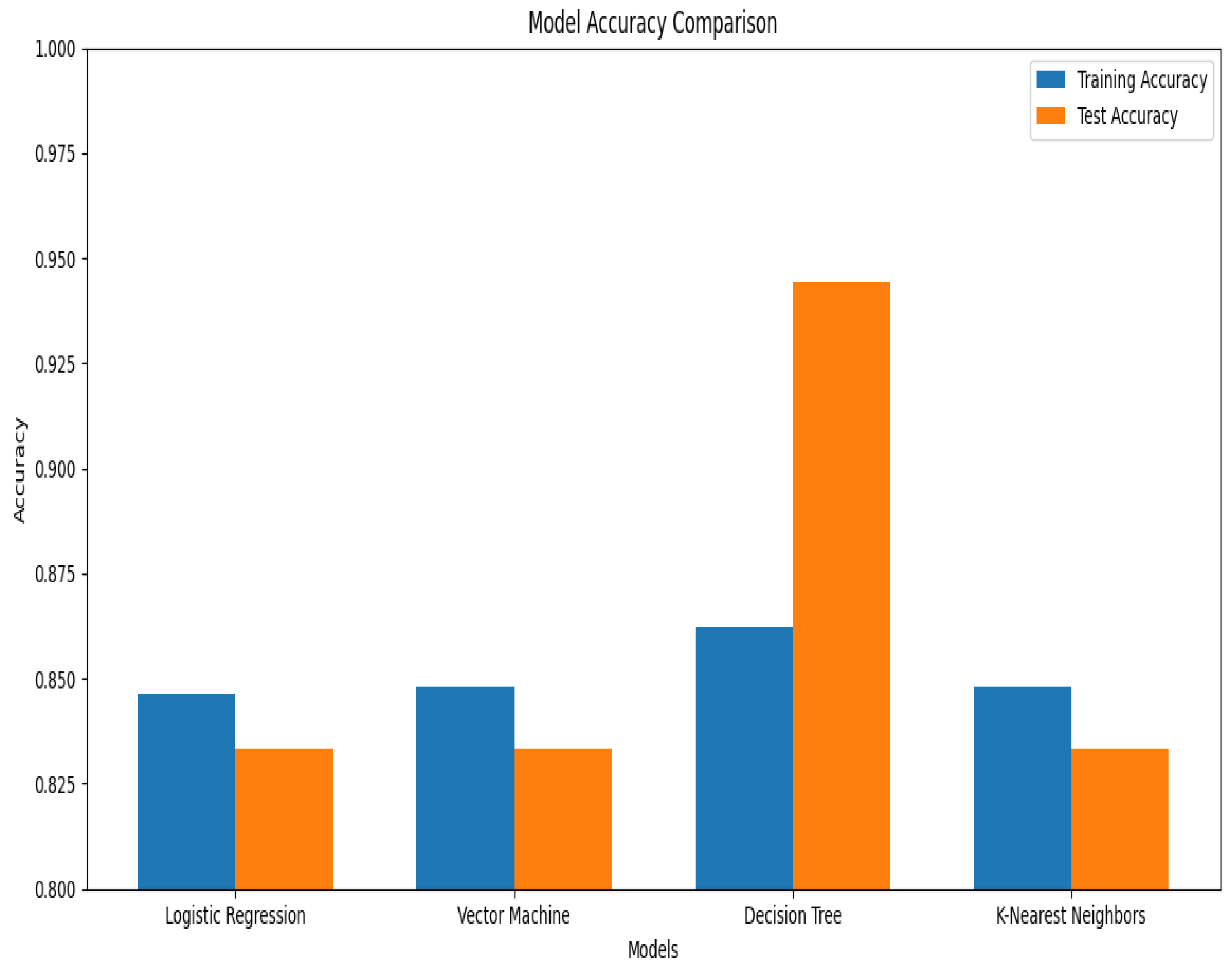


Dashboard Display the success count based on payload mass for all sites, highlighting the FT Booster Version Category as the highest.

Section 5

Predictive Analysis (Classification)

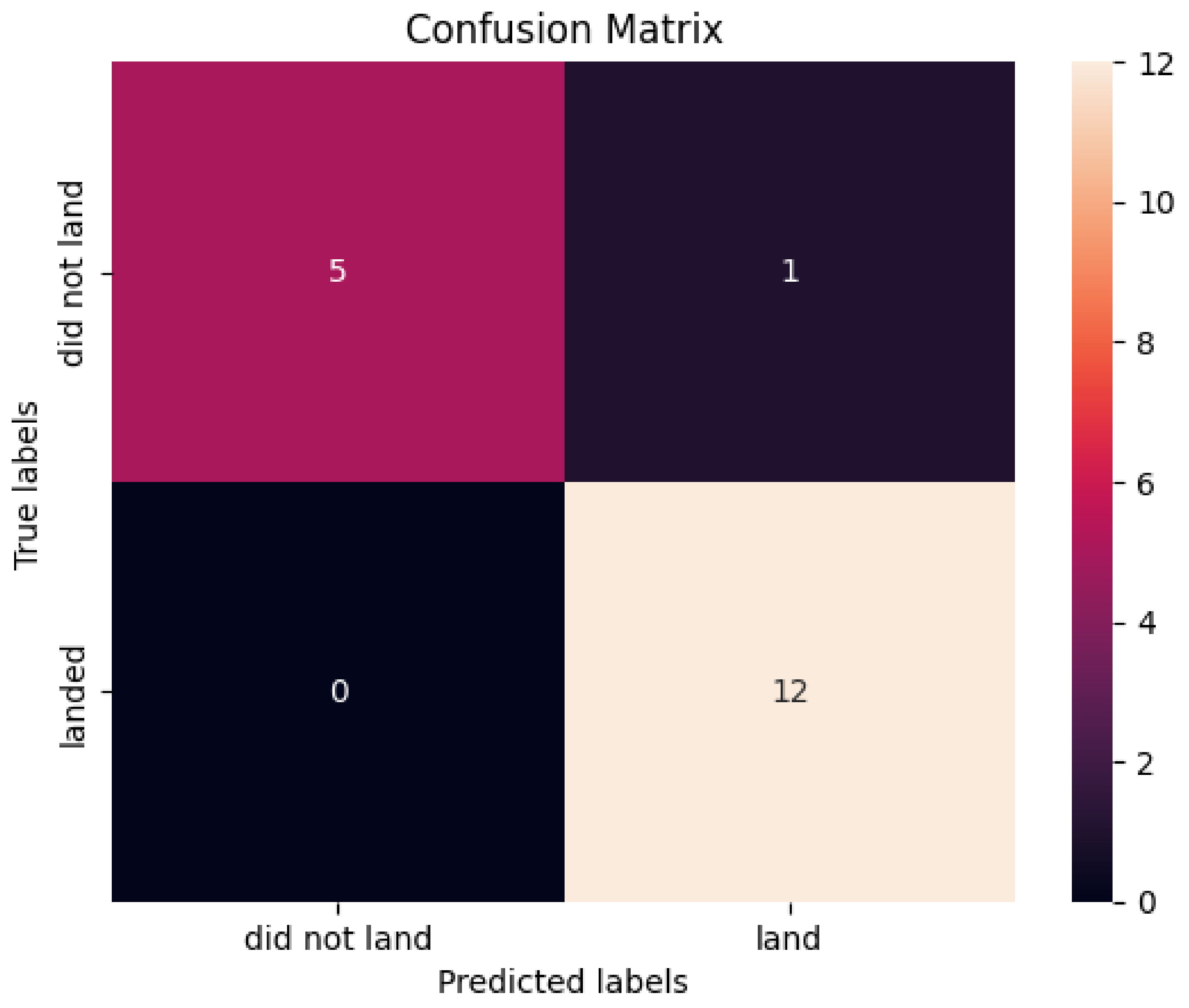
Classification Accuracy



- Decision Tree model has the highest classification accuracy

Confusion Matrix of Decision Tree

The confusion matrix highlight show well the decision tree model performed, with a high number of true positives and minimal false positives or negatives, indicating strong accuracy.



Conclusions

- FlightNumber and Payload Influence**: Both `FlightNumber` and `Payload` impact launch outcomes, but an increase in flight numbers doesn't significantly affect the results at different launch sites.
- VAFB-SLC Launch Site : No rockets with heavy payloads (over 10,000) have been launched from the VAFB-SLC site.
- Orbits with Highest Success Rates : ES-L1, GEO, HEO, and SSO orbits have the highest success rates.
- LEO and GTO Orbit Success : In the LEO orbit, success correlates with the number of flights, while in GTO, no such correlation is evident.
- Heavy Payloads : Heavy payloads have higher success rates in Polar, LEO, and ISS orbits.

Conclusions

- GTO Landings : It's difficult to distinguish between successful and unsuccessful landings for GTO, but overall success rates have increased since 2013.
- KSC LC-39A Success : KSC LC-39A launch site has the highest success rate, with 41.7% of successful launches and a 76.9% success rate.
- FT Booster Version : The FT Booster Version category has the highest success count based on payload mass across all launch sites.
- Decision Tree Model : The Decision Tree classifier performs best, with an accuracy of 0.86 and a test data accuracy score of 0.94.

Appendix

REST API
Definition : A Representational State Transfer (REST) API is a set of web service principles that allows interaction with web resources using a predefined set of stateless operations such as GET, POST, PUT, DELETE.

DataFrame
Definition : A two-dimensional, size-mutable, and potentially heterogeneous tabular data structure with labeled axes (rows and columns) in pandas. It's one of the most widely used structures in data analysis.

CSV (Comma-Separated Values)
Definition : A file format that stores tabular data in plain text, where each line corresponds to a row and each value is separated by a comma. It’s often used for data exchange between different applications.

Appendix

BeautifulSoup
Definition : A Python library used to parse HTML and XML documents. It creates a parse tree from page source code that can be used to extract data or modify the structure of the page.

EDA (Exploratory Data Analysis)
Definition : The process of analyzing datasets to summarize their main characteristics, often using visual methods. It helps to uncover patterns, spot anomalies, and test hypotheses.

GitHub
Definition : A web-based platform for version control using Git. It allows collaboration, sharing code repositories, and managing software projects.

Appendix

SQL (Structured Query Language)
Definition : A domain-specific language used to manage and manipulate relational databases. It allows querying, updating, and managing data within a database.

Folium
Definition : A Python library used for visualizing data on interactive leaflet maps. It enables the creation of interactive maps that can be embedded in Jupyter Notebooks or web apps.

Marker
Definition : A point symbol on a map (in libraries like Folium) that represents a specific location, often used to display important data at that point.

Appendix

Marker Cluster
Definition : A technique used in mapping to cluster nearby markers into a single marker that expands when zoomed in, helping to prevent overlapping and maintain map readability.

Plotly Dash
Definition : An open-source Python framework for building analytical web applications. It allows the creation of interactive dashboards and visualizations using simple Python code.

Dropdown
Definition : A UI component in Dash and other frameworks that allows users to select a single or multiple options from a list.

Appendix

Range Slider
Definition : A UI element in Dash that lets users select a range of values by sliding a handle across a scale, often used to filter data within a specified range.

Callbacks
Definition : In Dash, callbacks are Python functions that are automatically triggered when a component's property changes. They are used to create dynamic interactivity in web apps.

Logistic Regression
Definition : A statistical model used in machine learning for binary classification. It predicts the probability that a given input belongs to a certain class.

SVM (Support Vector Machine)
Definition : A supervised learning model used for classification and regression. It works by finding a hyperplane that best divides the data into classes.

Appendix

Decision Trees
Definition : A supervised learning algorithm used for classification and regression tasks. It splits data into branches based on decision rules derived from input features.

K-Nearest Neighbors (KNN)
Definition : A non-parametric, instance-based learning algorithm used for classification and regression. It assigns labels based on the majority vote from the nearest K neighbors in the feature space.

GridSearchCV
Definition : A technique in machine learning that performs an exhaustive search over a specified parameter grid for an estimator, helping to find the optimal hyperparameters.

Appendix

Cross-Validation Score
Definition : A technique used to assess how the results of a statistical model will generalize to an independent dataset. It involves splitting data into training and validation sets to evaluate model performance.

Confusion Matrix
Definition : A table used to evaluate the performance of a classification algorithm. It contains information about actual and predicted classifications done by a classifier.

True Positive (TP)
Definition : In a confusion matrix, these are cases where the model correctly predicted the positive class.

Appendix

True Negative (TN)
Definition : In a confusion matrix, these are cases where the model correctly predicted the negative class.

False Positive (FP)
Definition : In a confusion matrix, these are cases where the model incorrectly predicted the positive class (type I error).

False Negative (FN)
Definition : In a confusion matrix, these are cases where the model incorrectly predicted the negative class (type II error).

Thank you!