IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Yasin Azarhazin
March 1, 2024

Outline

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

Executive Summary

1. Summary of methodologies

- Data Collection with API
- Data Collection with Web Scraping
- Data Wrangling
- EDA with SQL
- EDA with Data Visualization
- Interactive Visual Analytics with Folium
- Interactive Dashboard with Ploty Dash
- Machine Learning Prediction

2. Summary of all results

- EDA results
- Interactive Analytics results
- Predictive Analytics results

Introduction

- **Project background and context**

  In this capstone, 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.

- **Problems you want to find answers**
  - What key elements contribute to the successful landing of a rocket?
  - How do different features interact to influence the success rate of a rocket's landing?
  - What specific operating conditions are essential to guarantee the success of a landing program?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - We gathered data by using the SpaceX API and scraping information from Wikipedia.
- Perform data wrangling
  - We converted categorical features into one-hot encoding.
- 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

Two data collection methods were employed:

1. **Using API** : Data collection involved utilizing a GET request to access the SpaceX API. Subsequently, we decoded the response content by employing the .json() function and transformed it into a Pandas dataframe using .json_normalize().

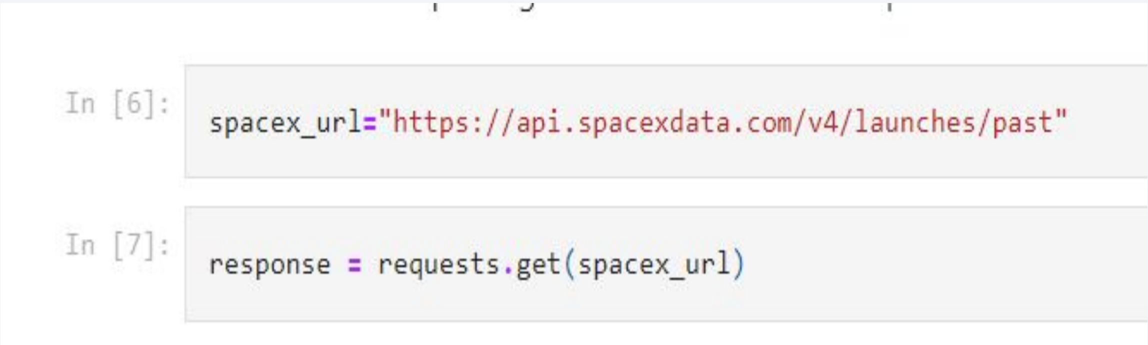
2. **Web Scraping** : We conducted web scraping on Wikipedia to retrieve Falcon 9 launch records using BeautifulSoup. Our goal was to extract the launch records presented in the form of an HTML table, parse the table content, and then convert it into a Pandas dataframe for subsequent analysis.

Data Collection – SpaceX API

- We used get request to get a response from API then we converted it to a json file

- https://github.com/Yasin-Az/IBM-DataScience/blob/main/SpaceX%20Capstone%20Project/SpaceX_Data_Collection_API.ipynb

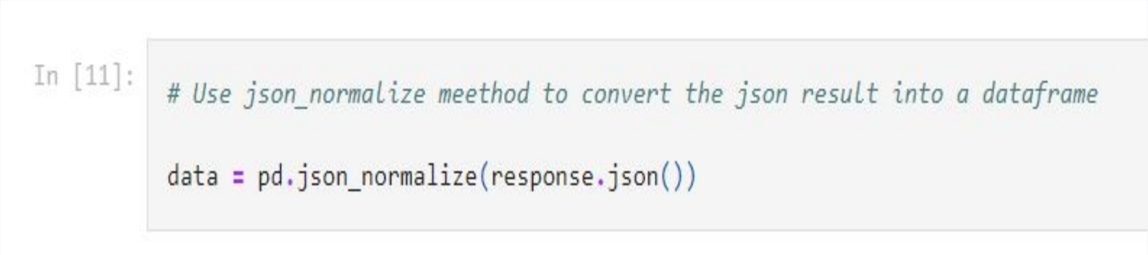
Getting a response from API



```
In [6]:  spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]:  response = requests.get(spacex_url)
```

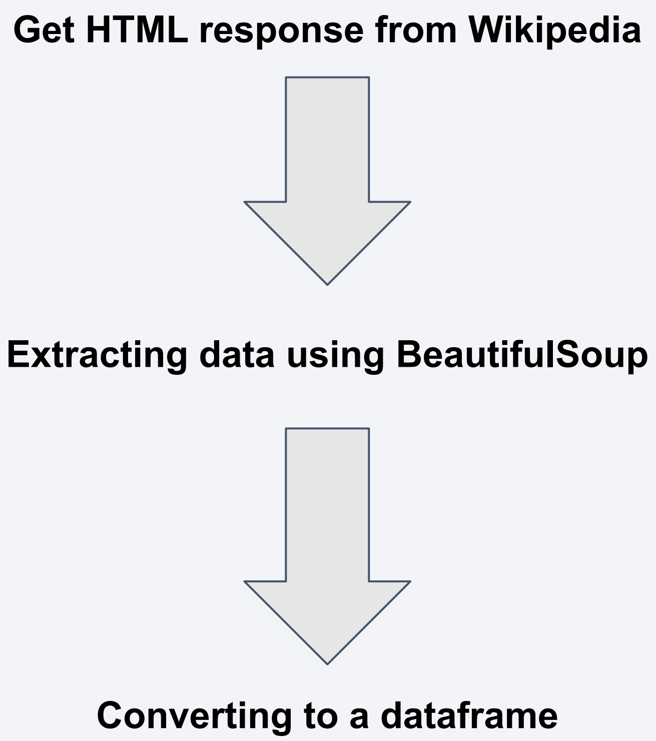
Converting the response to json file



```
In [11]:  # Use json_normalize meethod to convert the json result into a dataframe

          data = pd.json_normalize(response.json())
```

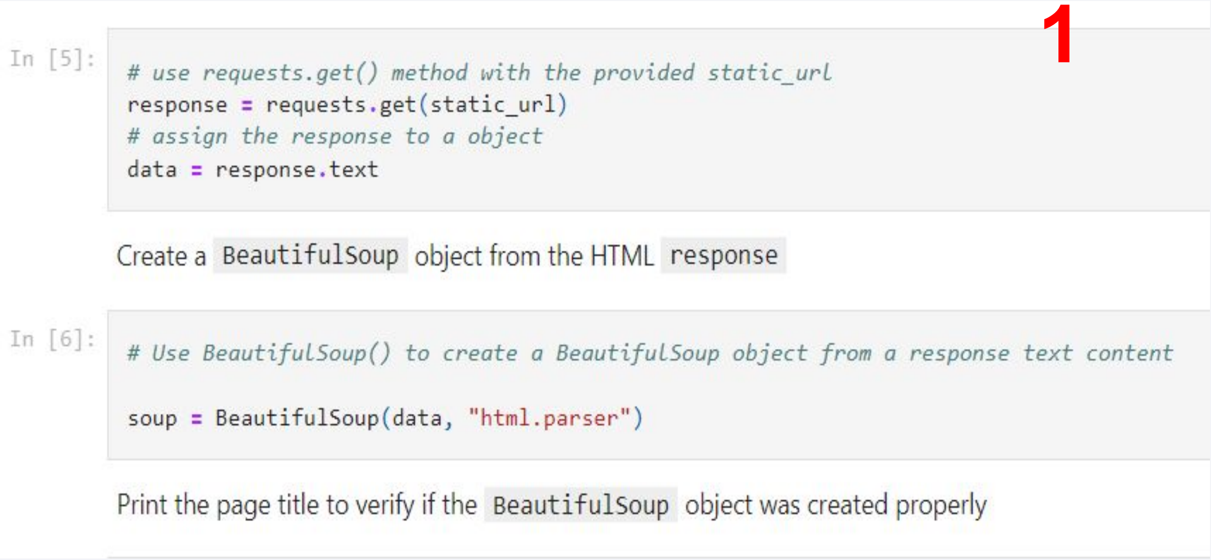
Data Collection - Scraping



Get HTML response from Wikipedia

Extracting data using BeautifulSoup

Converting to a dataframe



1

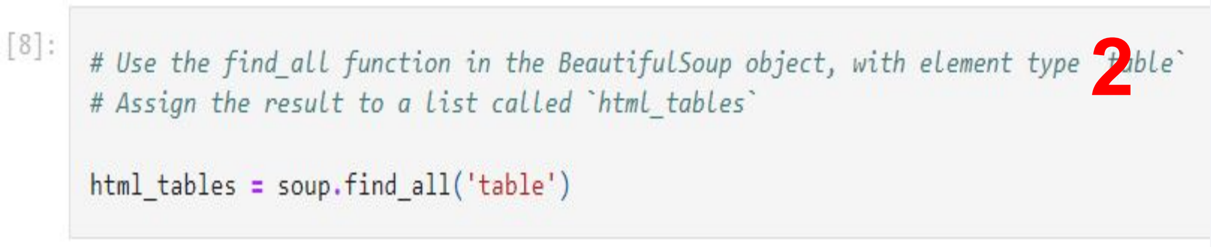
```
In [5]:  # use requests.get() method with the provided static_url
         response = requests.get(static_url)
         # assign the response to a object
         data = response.text
```

Create a `BeautifulSoup` object from the HTML `response`

```
In [6]:  # Use BeautifulSoup() to create a BeautifulSoup object from a response text content

         soup = BeautifulSoup(data, "html.parser")
```

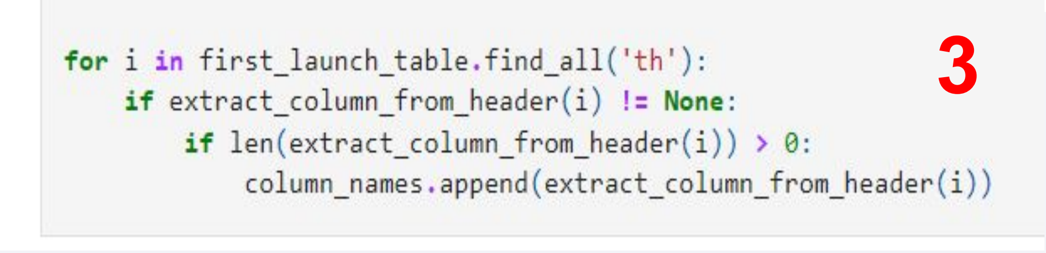
Print the page title to verify if the `BeautifulSoup` object was created properly



2

```
[8]:  # Use the find_all function in the BeautifulSoup object, with element type `table`
      # Assign the result to a list called `html_tables`

      html_tables = soup.find_all('table')
```

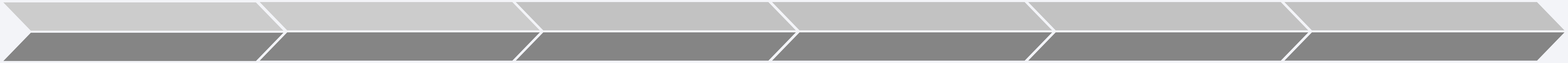


3

```
for i in first_launch_table.find_all('th'):
    if extract_column_from_header(i) != None:
        if len(extract_column_from_header(i)) > 0:
            column_names.append(extract_column_from_header(i))
```

Data Wrangling

The dataset includes cases of unsuccessful booster landings due to accidents. Examples include successful ocean landings ("True Ocean") and unsuccessful ones ("False Ocean"). Similarly, "True RTLS" indicates a successful ground pad landing, while "False RTLS" denotes an unsuccessful attempt. "True ASDS" signifies a successful drone ship landing, and "False ASDS" indicates an unsuccessful one. These outcomes are converted into training labels, where '1' means successful landing, and '0' means unsuccessful.



Calculate the number of launches at each site

Calculate the number and occurrence of each orbit

Calculate the number and occurrence of mission outcome per orbit type

Create a landing outcome label from Outcome column

Work out success rate for every landing in dataset

Export dataset as .CSV

https://github.com/Yasin-Az/IBM-DataScience/blob/main/SpaceX%20Capstone%20Project/SpaceX_Data_Wrangling.ipynb

EDA with Data Visualization

Type of Charts were used :

1. **Scatter plots** : Scatter plots display the correlation between two variables, showcasing their relationship with a visual representation of data.

   1.1. Flight Number vs Payload Mass

   1.2. Flight Number vs Launch Site

   1.3. Payload vs Launch Site

   1.4. Orbit vs Flight Number

   1.5. Payload vs Orbit Type

   1.6. Orbit vs. Payload Mass

2. **Barplots** : Bar diagrams facilitate quick comparisons of data across different groups. With categories on one axis and discrete values on the other, they visually illustrate the relationship between the two axes, capturing significant data changes over time.

   2.1. Mean vs Orbit

3. **Line plots** : Line graphs provide clear visibility into data variables and trends, aiding in predictions for unrecorded results.

   3.1. Success Rate vs Year

https://github.com/Yasin-Az/IBM-DataScience/blob/main/SpaceX%20Capstone%20Project/EDA_DataViz.ipynb

EDA with SQL

We smoothly brought the SpaceX dataset into a PostgreSQL database right from our Jupyter Notebook. Using SQL for some data digging (EDA), we fished out some interesting nuggets, like:

- The names of unique launch sites in the space mission.
- The total payload mass carried by boosters launched by NASA.
- The average payload mass carried by booster version F9 v1.1
- The total number of successful and failure mission outcomes.
- The failed landing outcomes in drone ship, their booster version and launch site names.

https://github.com/Yasin-Az/IBM-DataScience/blob/main/SpaceX%20Capstone%20Project/EDA-SQL-Coursera_Sqllite.ipynb

Build an Interactive Map with Folium

We labeled all launch sites and adorned the Folium map with markers, circles, and lines to visually depict the success or failure of launches at each spot.

Assigning launch outcomes as class 0 for failure and 1 for success, we color-coded marker clusters to easily pinpoint launch sites with a notably high success rate.

Exploring the distances between launch sites and their surroundings, we delved into answering questions such as:

I. Are launch sites situated in close proximity to railways, highways, and coastlines?
II. Do launch sites maintain a specific distance from urban areas?

Build a Dashboard with Plotly Dash

We opted for pie charts and scatter graphs to effectively convey different aspects of the data:

Pie Charts for Total Launches by Sites:

- Why: Pie charts are excellent for displaying parts of a whole. In this context, they provide a clear visual representation of the distribution of total launches across different sites. The audience can quickly grasp the proportional significance of each launch site in the overall dataset.

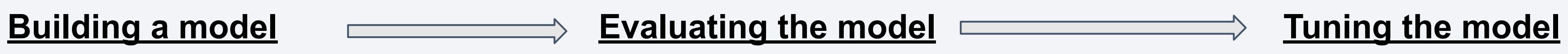
Scatter Graphs for Relationship between Outcome and Payload Mass:

- Why: Scatter graphs are ideal for visualizing relationships between two numerical variables. By plotting launch outcomes against payload mass for different booster versions, we can discern patterns, trends, or correlations. This type of graph helps in understanding if there is any discernible connection between the payload mass and the success or failure of launches for various booster versions.

https://github.com/Yasin-Az/IBM-DataScience/blob/main/SpaceX%20Capstone%20Project/Ploty_Dash.py

Predictive Analysis (Classification)

- We used NumPy and Pandas to handle and tweak our data. Then, we split it into parts for training and testing.
- The dataset was intelligently partitioned into training and testing sets to facilitate robust model evaluation.
- Employing diverse machine learning models, we meticulously fine-tuned hyperparameters through GridSearchCV.
- Our model's performance was gauged using accuracy as the primary metric, guiding us in refining the model through feature engineering and algorithm tuning.
- Ultimately, we identified the most effective classification model through thorough evaluation.



Building a model → **Evaluating the model** → **Tuning the model**

https://github.com/Yasin-Az/IBM-DataScience/blob/main/SpaceX%20Capstone%20Project/SpaceX_Machine_Learning_Prediction.ipynb

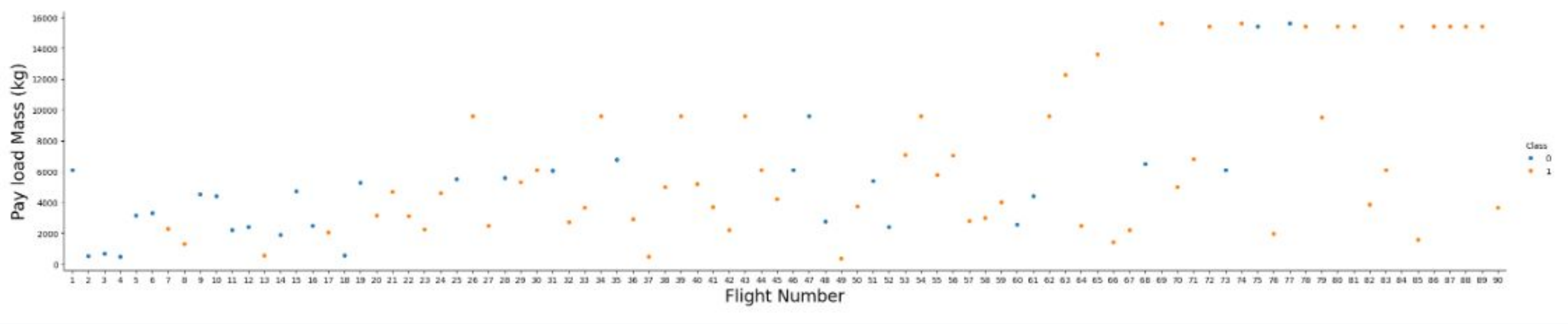
Results

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

Section 2

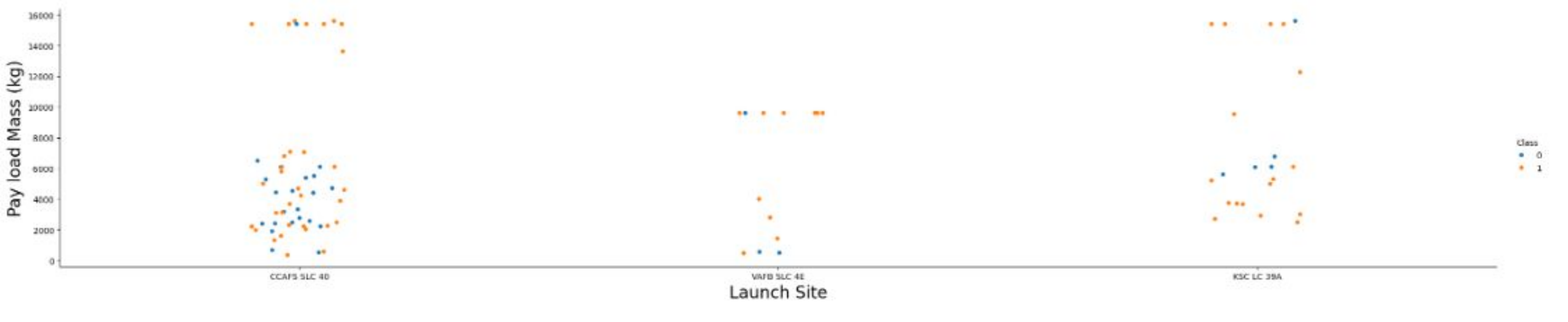
Insights drawn from EDA

Flight Number vs. Launch Site



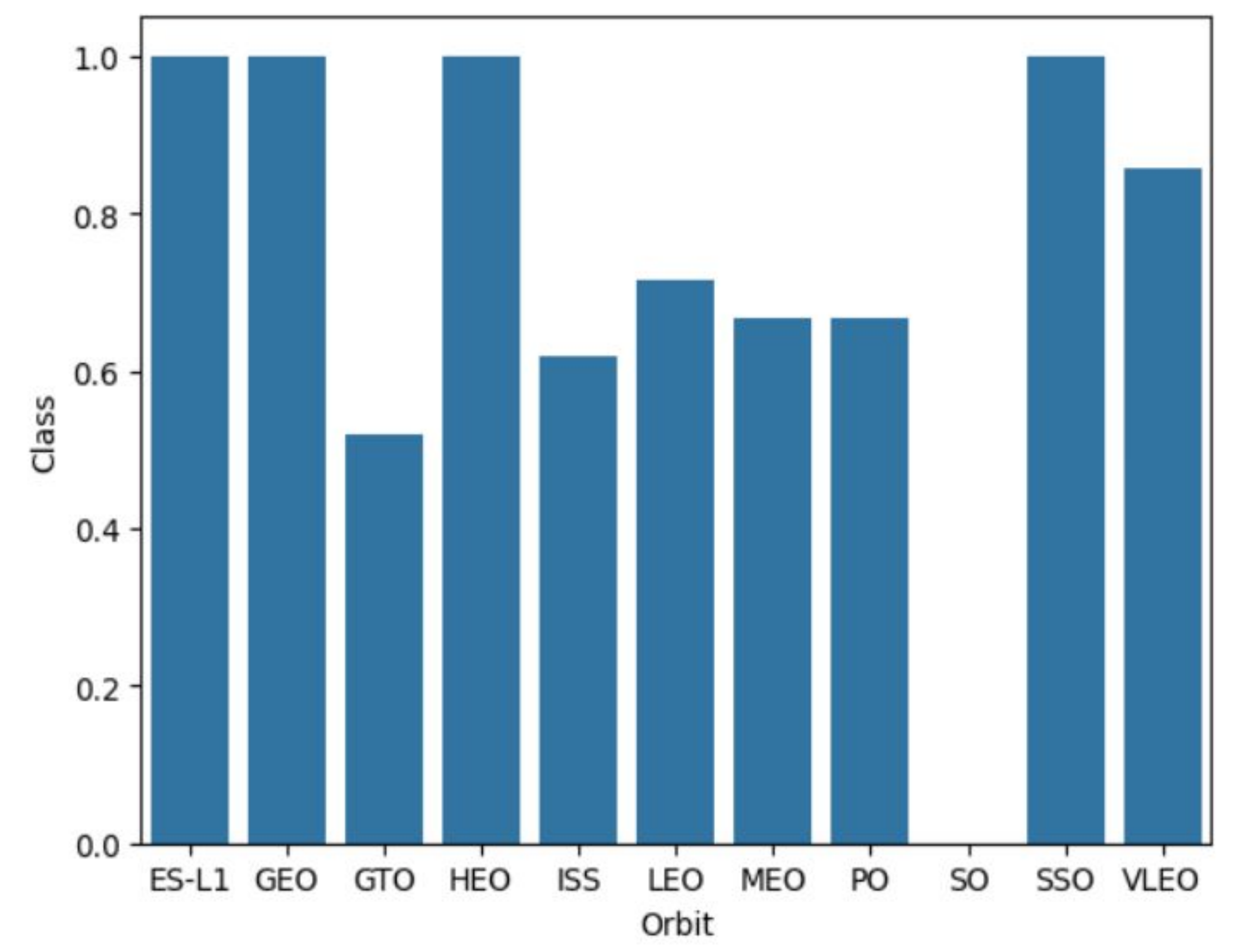
The higher the number of flights at a launch site, the better the chances of success at that site.

Payload vs. Launch Site



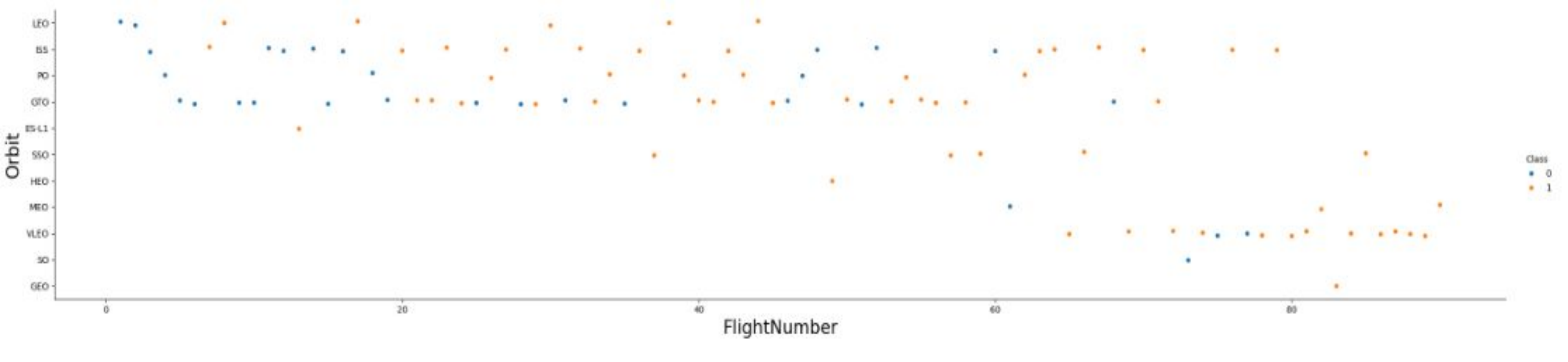
The correlation suggests that, for Launch Site CCAFS SLC 40, a greater payload mass tends to result in a higher success rate for the rocket. However, based on the visualization, it's not entirely clear whether the launch site's success is significantly dependent on the payload mass. Further analysis or additional data might be needed to make a conclusive decision.

Success Rate vs. Orbit Type



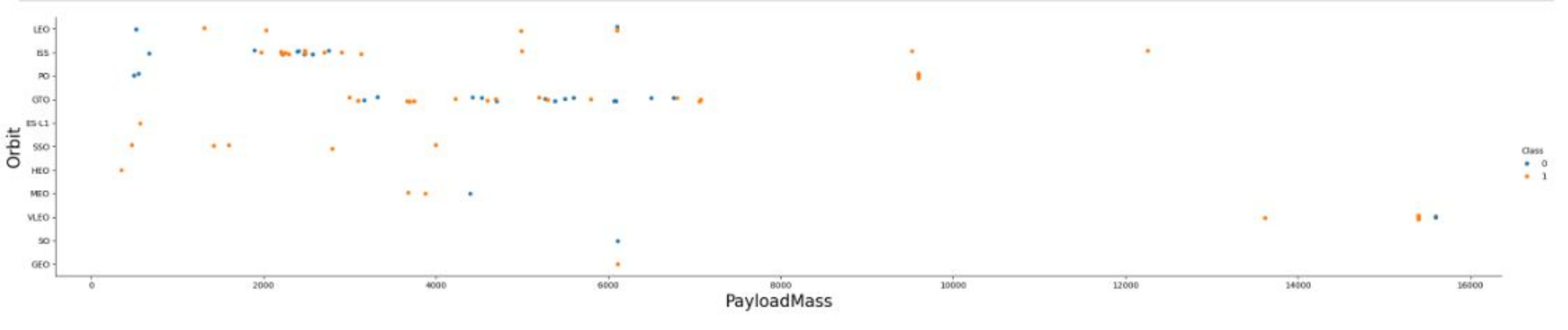
Orbit **GEO,HEO,SSO,ES-L1** has the best success rate

Flight Number vs. Orbit Type



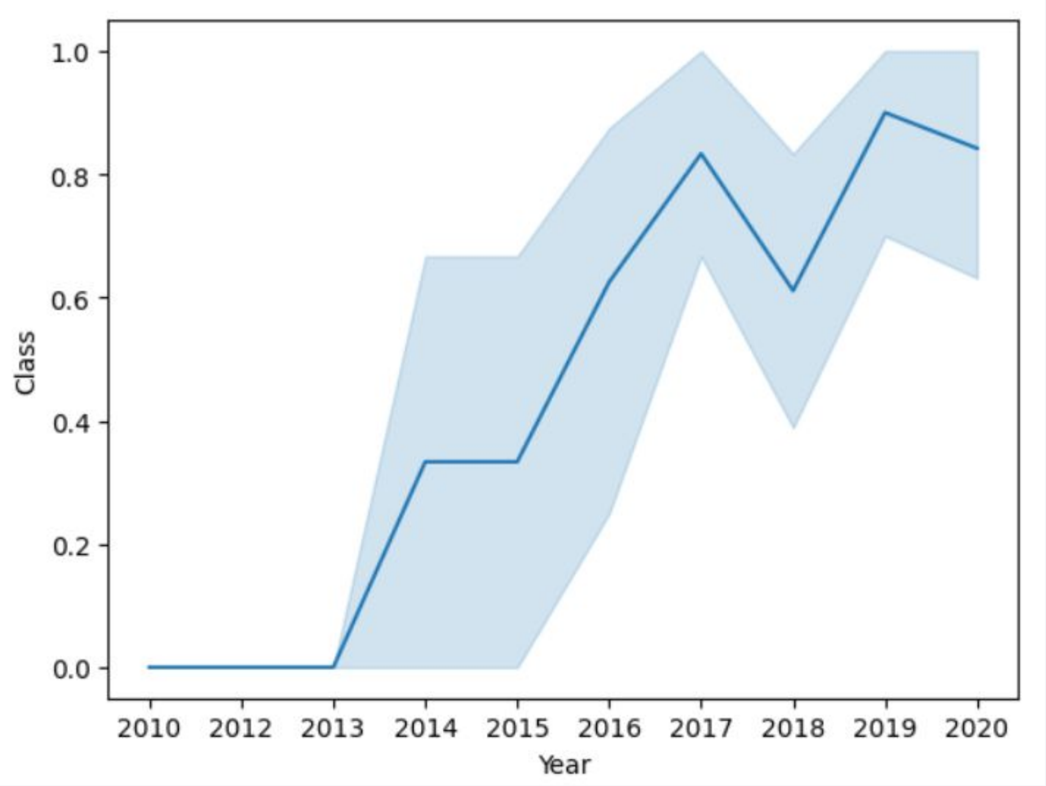
For rockets going to Low Earth Orbit (LEO), it looks like the more flights they have, the more successful they are. But when they're headed to Geostationary Transfer Orbit (GTO), it seems like the number of flights doesn't really affect how successful they are.

Payload vs. Orbit Type



You might notice that heavy payloads tend to have a bad effect on rockets going to (GTO), but they're usually a good thing for rockets heading to (GEO) and Polar (LEO), like (ISS).

Launch Success Yearly Trend



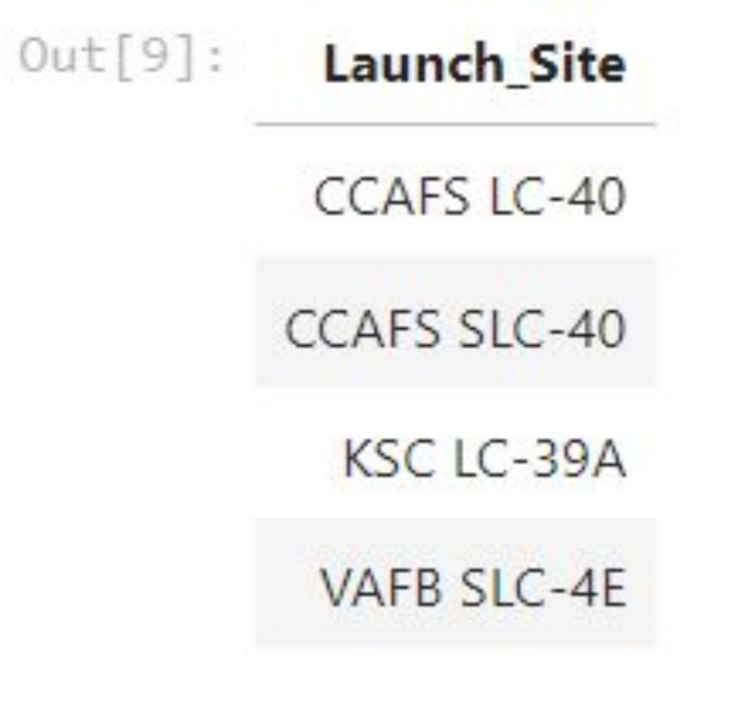
success rate has kept increasing since 2013 till 2020

All Launch Site Names

Code we used :

```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL ORDER BY 1;
```

Launch Sites Names :

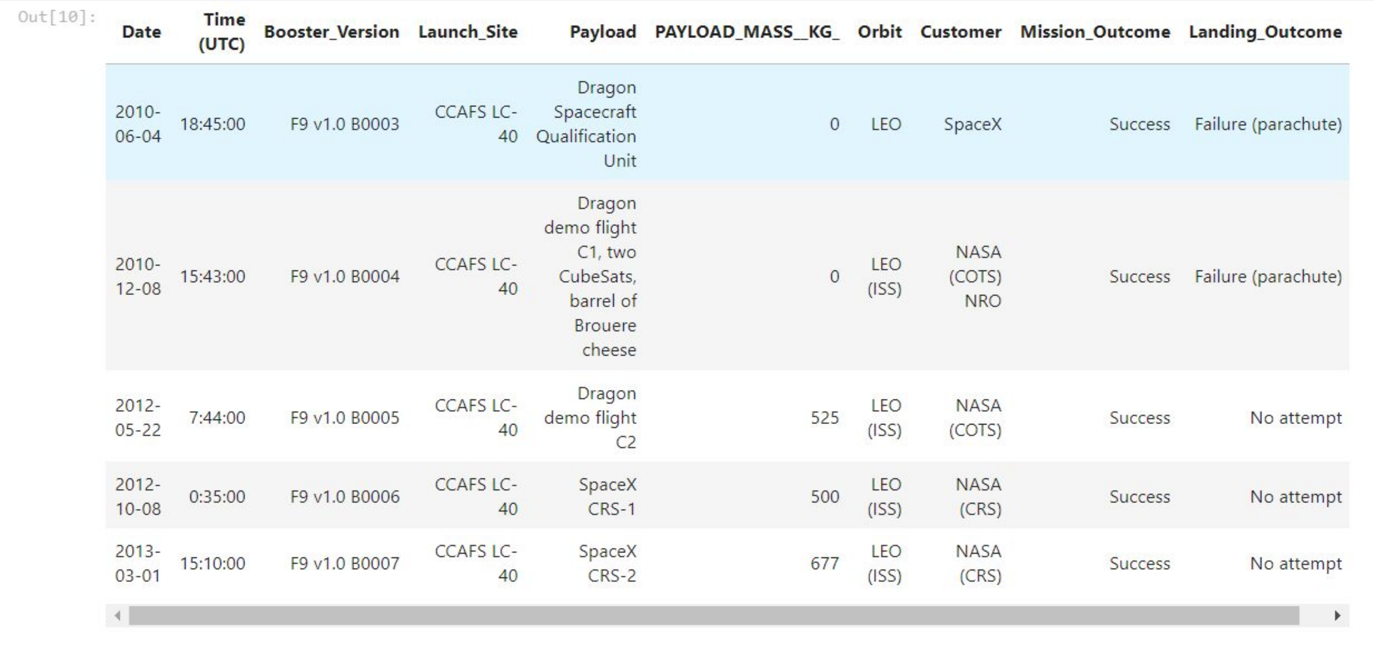


Out[9]:

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

We used DISTINCT to display only unique launch sites

Launch Site Names Begin with 'CCA'



Out[10]:

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

Including the phrase "LIMIT 5" in the query ensures that only 5 records will be displayed. The LIKE keyword with the wildcard 'CCA%' indicates that the Launch_Site name must start with CCA.

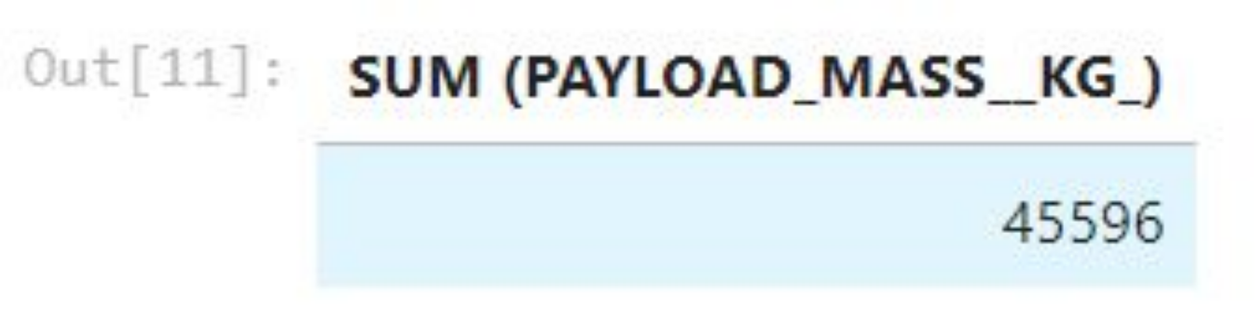
Code :

```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

Total Payload Mass

We determined that the total payload carried by NASA boosters is 45596 using the following query :

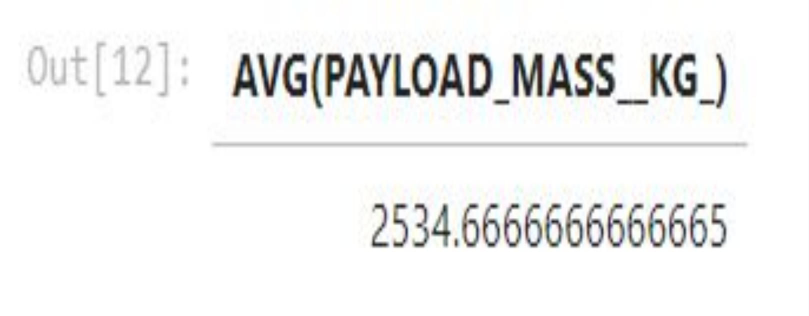
%sql SELECT SUM (PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER='NASA (CRS)'



Out[11]:

SUM (PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1



```
Out[12]:   AVG(PAYLOAD_MASS_KG_)
              2534.6666666666665
```

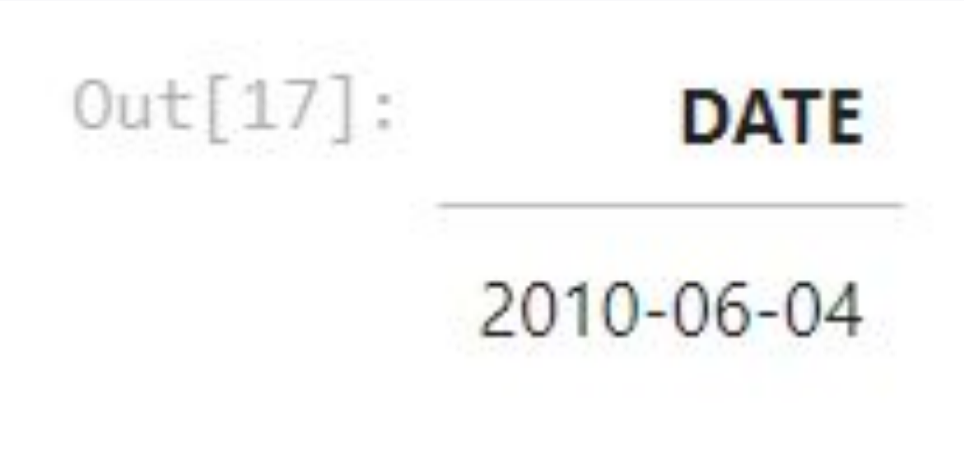
By employing the AVG function, we computed the average payload mass in the column PAYLOAD_MASS_KG_. The WHERE clause was used to narrow down the dataset, ensuring that the calculations specifically pertained to the Booster_version F9 v1.1.

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE booster_version LIKE 'F9 v1.1%'
```

First Successful Ground Landing Date

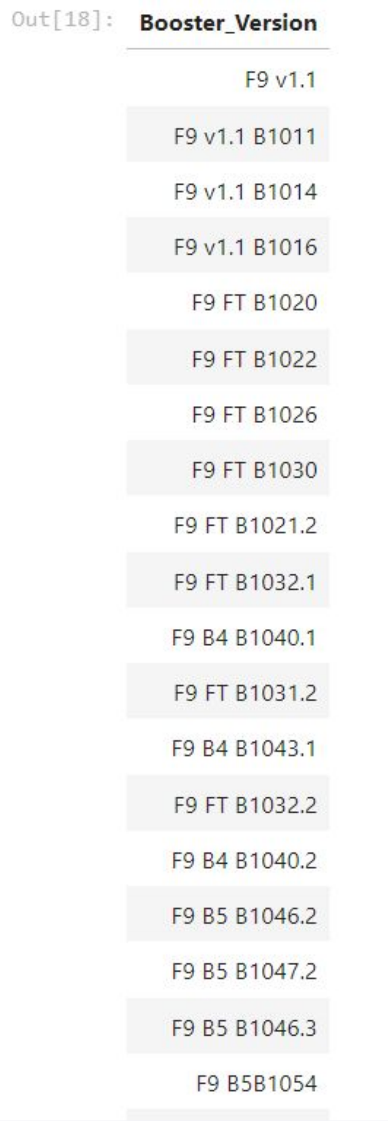
This SQL code retrieves the earliest date (MIN(DATE)) from the SPACEXTBL table WHERE the mission_outcome is marked as 'Success'. It essentially identifies the earliest date associated with successful missions in the dataset.

```
%sql SELECT MIN(DATE) AS DATE FROM SPACEXTBL WHERE mission_outcome LIKE 'Success'
```



```
Out[17]:        DATE
          ----------
          2010-06-04
```

Successful Drone Ship Landing with Payload between 4000 and 6000



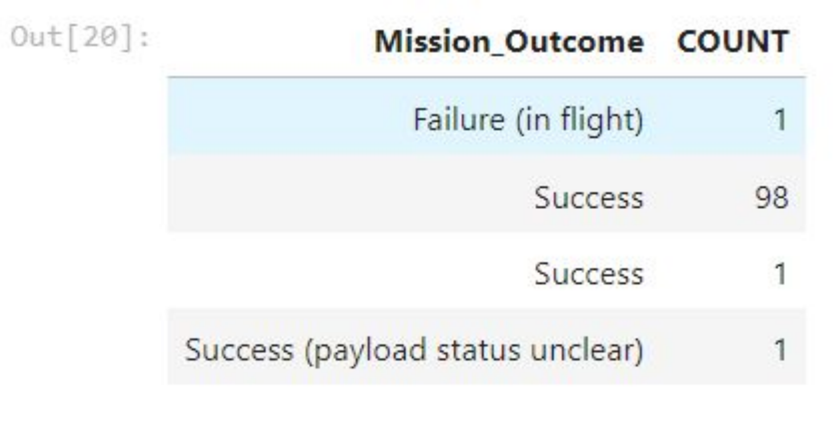
Out[18]:

Booster_Version
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1021.2
F9 FT B1032.1
F9 B4 B1040.1
F9 FT B1031.2
F9 B4 B1043.1
F9 FT B1032.2
F9 B4 B1040.2
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5 B1046.3
F9 B5B1054

This SQL code extracts the BOOSTER_VERSION from the SPACEXTBL table where the PAYLOAD_MASS__KG_ falls within the range of 4000 and 6000. It retrieves information about booster versions associated with payloads within that specific weight range.

```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

Total Number of Successful and Failure Mission Outcomes



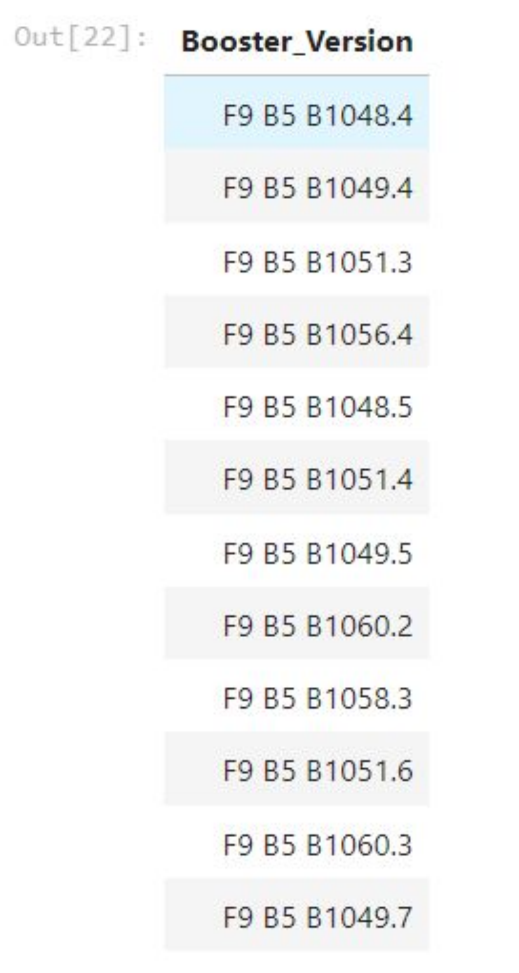
Out[20]:

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

This SQL code counts and displays the occurrences of mission outcomes (success or failure) from the SPACEXTBL table, presenting a summary ordered alphabetically by outcome.

%sql SELECT mission_outcome, COUNT(*) AS COUNT FROM SPACEXTBL GROUP BY mission_outcome ORDER BY mission_outcome

Boosters Carried Maximum Payload



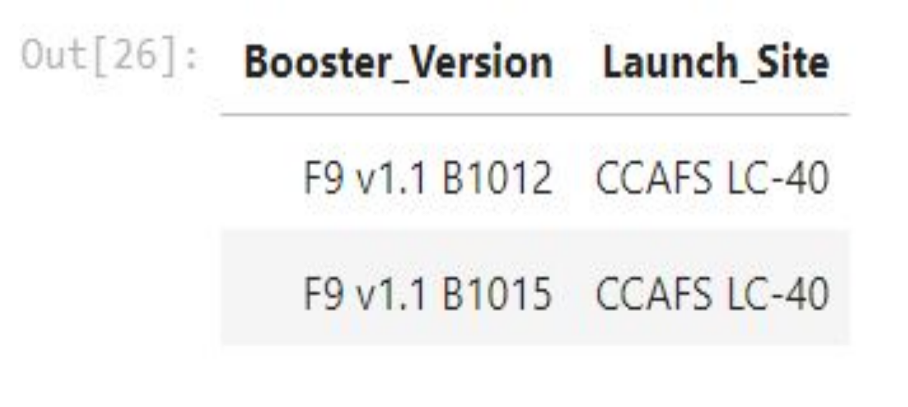
Out[22]:

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

This SQL code retrieves the booster_version from the SPACEXTBL table where the payload_mass__kg_ is equal to the maximum payload mass found in the entire dataset. In simpler terms, it identifies the booster version associated with the heaviest payload in the given dataset.

```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_=(SELECT
MAX(payload_mass__kg_) FROM SPACEXTBL)
```

2015 Launch Records



Out[26]:

Booster_Version	Launch_Site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

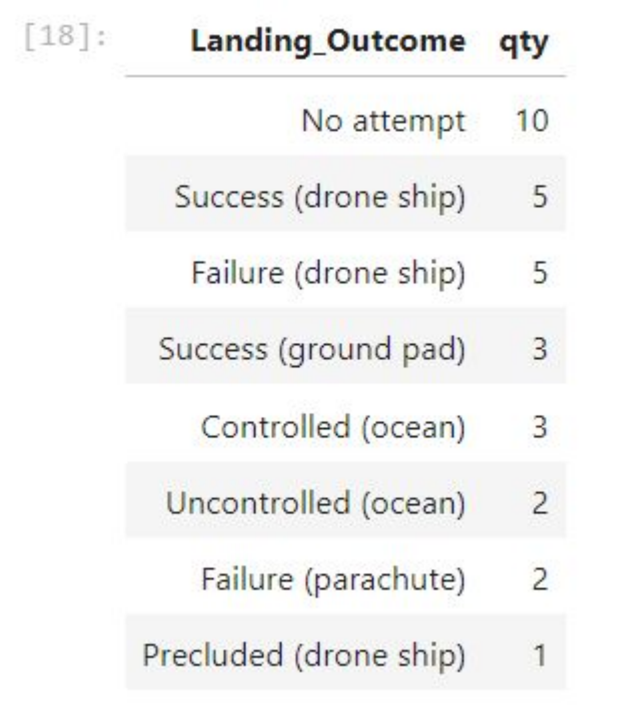
This SQL code selects the BOOSTER_VERSION and LAUNCH_SITE columns from the SPACEXTBL table where the "Landing_Outcome" is marked as 'Failure (drone ship)' and the DATE column starts with '2015%'. In other words, it retrieves information about booster versions and launch sites specifically associated with failed landings on drone ships in the year 2015.

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE  "Landing_Outcome"='Failure (drone ship)'
AND DATE LIKE '2015%'
```

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

This SQL code counts and displays the quantity (qty) of each unique "Landing_Outcome" from the SPACEXTBL table, focusing on the time period between '2010-06-04' and '2017-03-20'. The results are grouped by "Landing_Outcome" and arranged in descending order based on the quantity.

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



[18]:

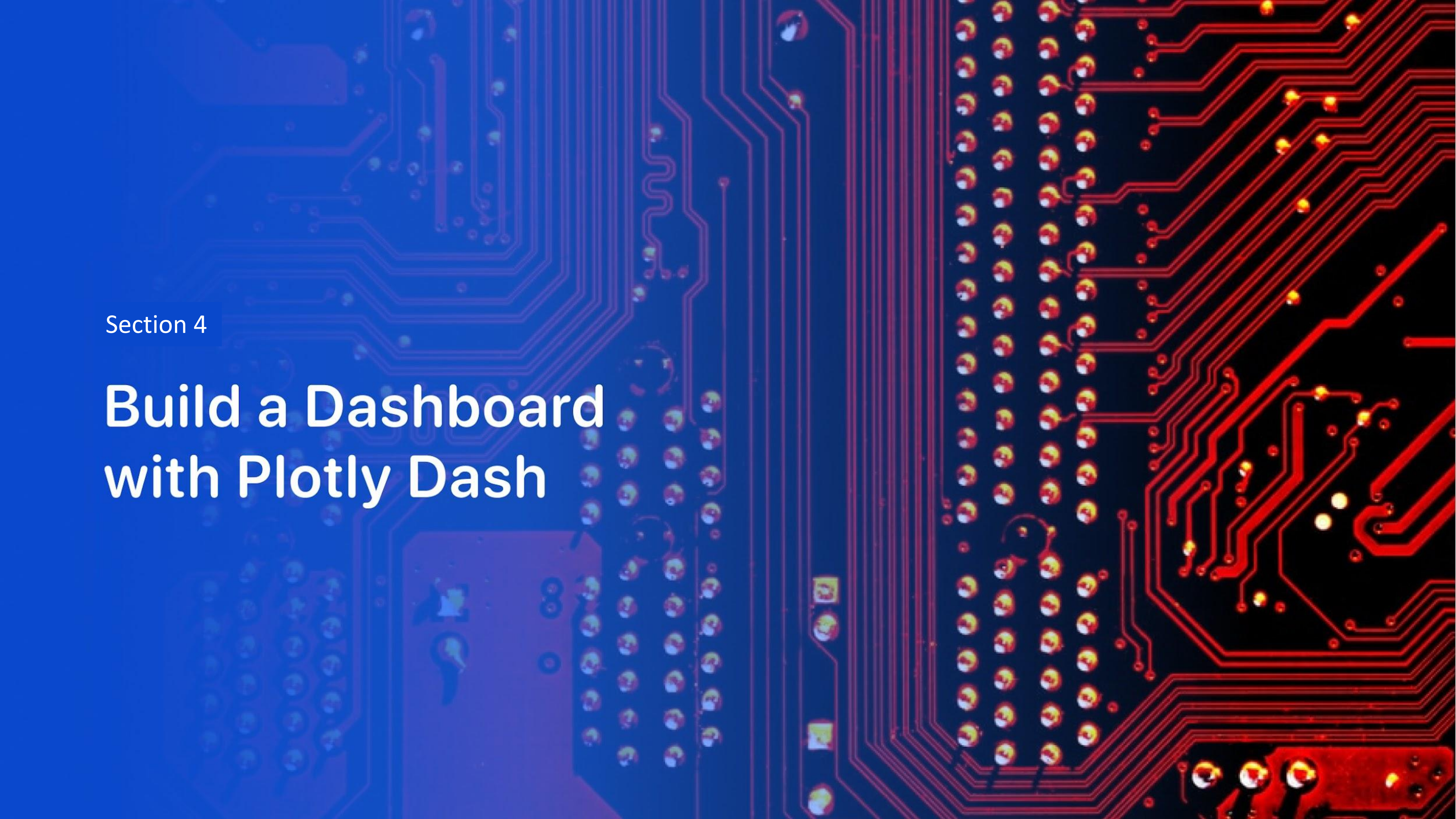
Landing_Outcome	qty
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

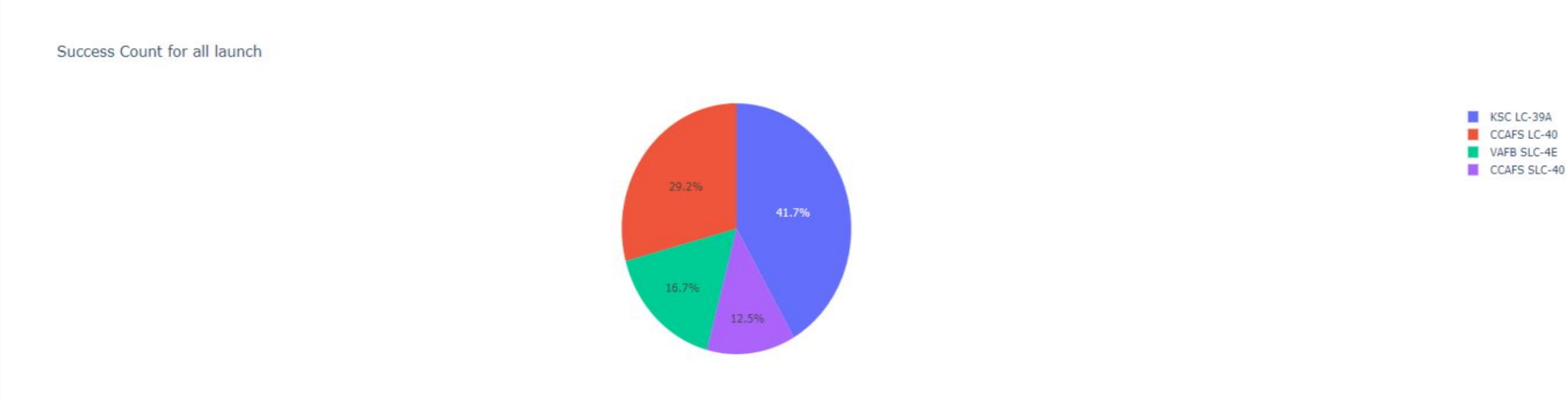
Launch Sites Proximities Analysis

Section 4

Build a Dashboard with Plotly Dash

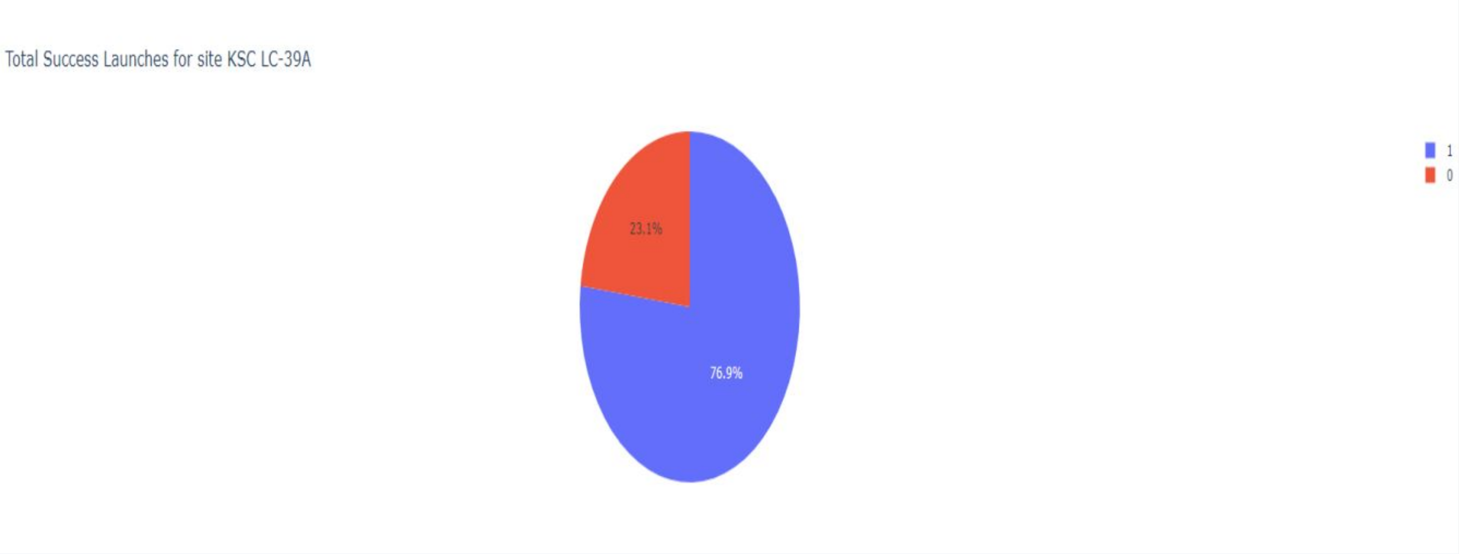


Success Rates Across Launch Sites



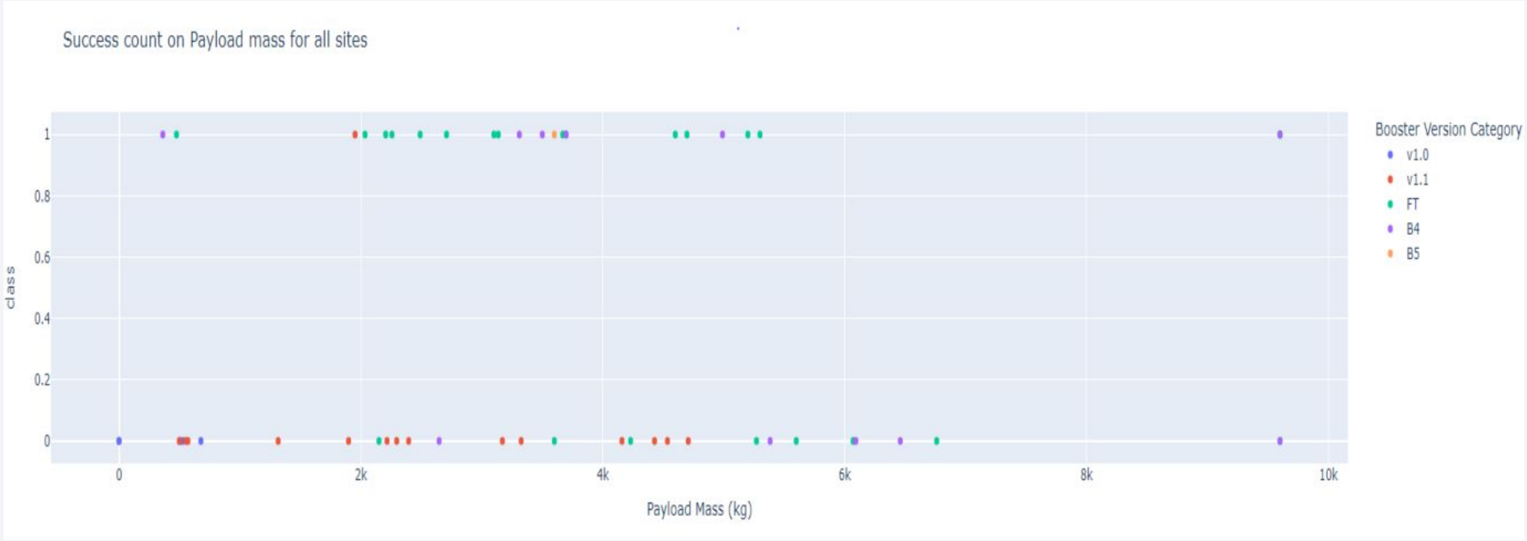
KSC LC-39A boasts the highest success rate among launch sites, standing at an impressive 41.7%.

The success ratio for the launch site with the highest success rate.



KSC LC-39A holds the record for the highest success rate, achieving successful launches at 76.9% and experiencing failures at 23.1%.

<Dashboard Screenshot 3>

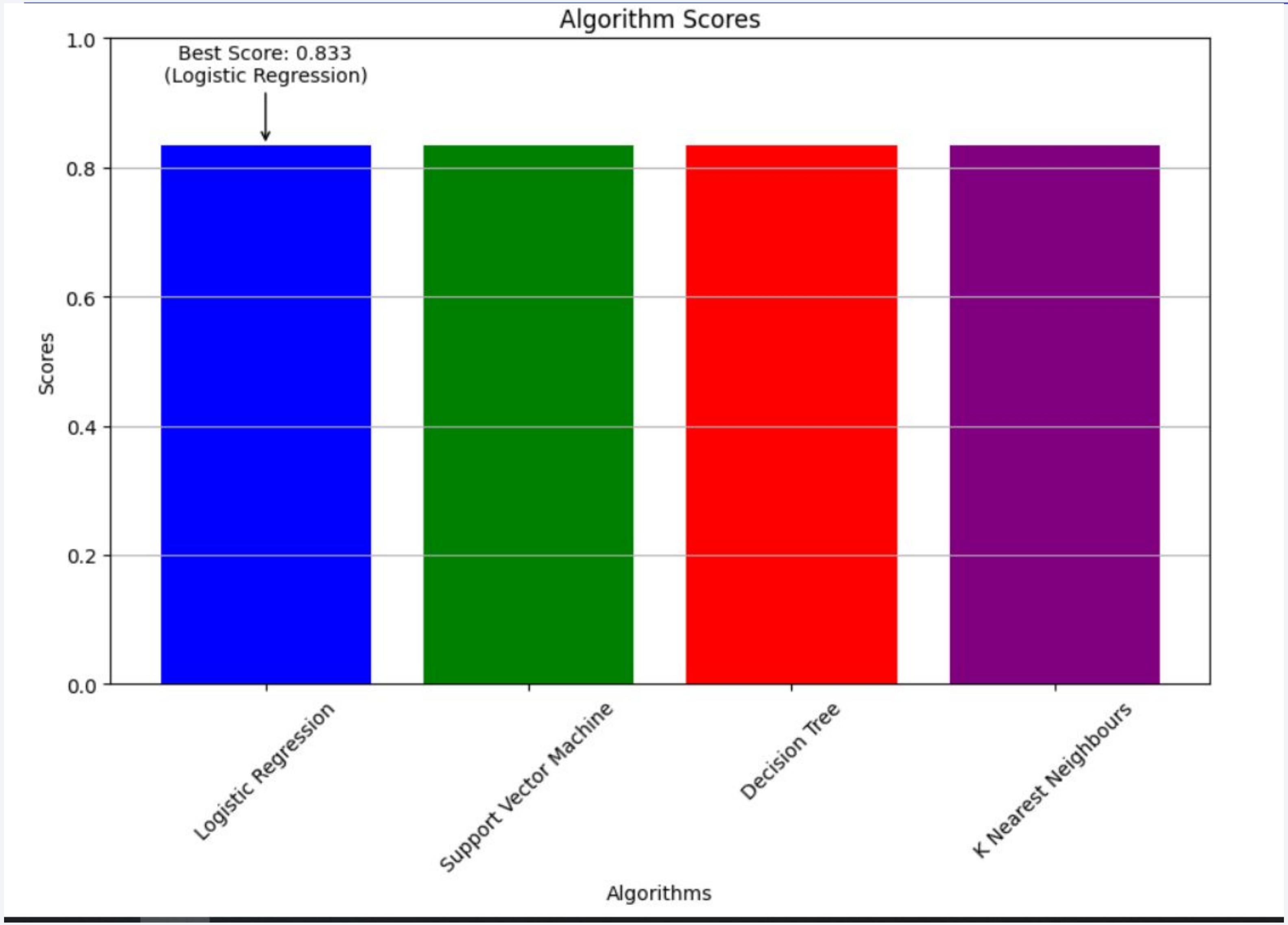


The plot indicates a lower success rate for payloads exceeding 6000kg.

Section 5

Predictive Analysis (Classification)

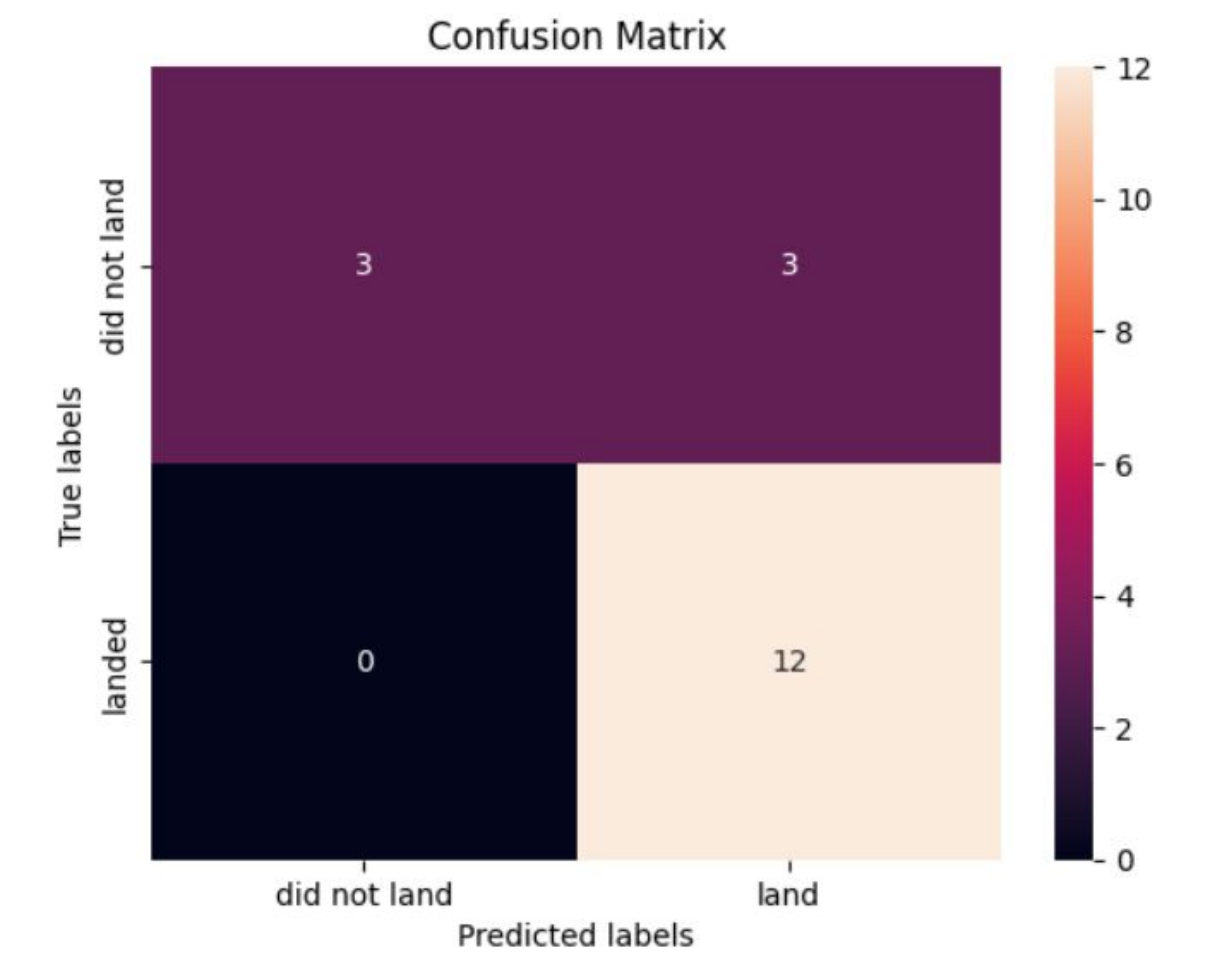
Classification Accuracy



Logistic Regression achieved the highest score among all the models.

Confusion Matrix

In this analysis, the confusion matrix for logistic regression reveals that out of 18 missions, 15 were labeled as landed. Among these, 12 were accurately predicted, while 3 were predicted incorrectly.



Conclusions

In our analysis, several significant patterns and insights emerged:

- The success rate at launch sites demonstrates a positive correlation with the volume of flights, indicating that higher activity fosters greater success.
- Over the years, from 2013 to 2020, there's a discernible upward trajectory in launch success rates.
- Specific orbits, including ES-L1, GEO, HEO, SSO, and VLEO, consistently exhibit higher success rates.
- KSC LC-39A stands out as the launch site with the most successful launches among all.
- Leveraging machine learning algorithms, Logistic Regression emerged as the most effective model for predicting mission outcomes, showcasing its suitability for this task.

This comprehensive summary encapsulates the key findings, providing a clear and succinct overview of the insights obtained from the data analysis.

Thank you!