BY MEHUL SUJI

IBM Data Science Capstone

Outline

Executive Summary

- Summary of Methodology

  - Data Collection: AP, Web scrapping, SQL
  - Data Wrangling and Analysis
  - Interactive Maps
  - Predictive Analytics for each classification model

- Summary of all Results:
  - *Data Analysis with Interactive Visuals*
  - *Best Model for Predictive Analysis*

Introduction

- SpaceX stands as a trailblazing enterprise that has significantly disrupted the space industry by introducing rocket launches, notably the Falcon 9, at remarkably low costs—starting as low as 62 million dollars. This is in stark contrast to other providers whose charges soar to a staggering 165 million dollars per launch. The lion's share of these cost savings can be attributed to SpaceX's ingenious concept of reusing the initial stage of the rocket. This involves skillfully re-landing the rocket for subsequent missions, thereby driving costs down even further with each successive reuse.

As a data scientist employed by a startup that is in direct competition with SpaceX, the primary objective of this endeavor revolves around establishing a robust machine learning pipeline. The aim is to forecast the outcome of the first stage's landing in future missions. This undertaking carries immense significance as it aids in determining the optimal bidding price against SpaceX for securing a rocket launch contract.

The problems included:

• Identifying all factors that influence the landing outcome

• The relationship between each variables and how it is affecting the outcome.

• The best condition needed to increase the probability of successful landing.

Methodology

- Data collection methodology:
    - Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
    - Data was processed using one-hot encoding for categorical features
- 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 mode

Data Collection

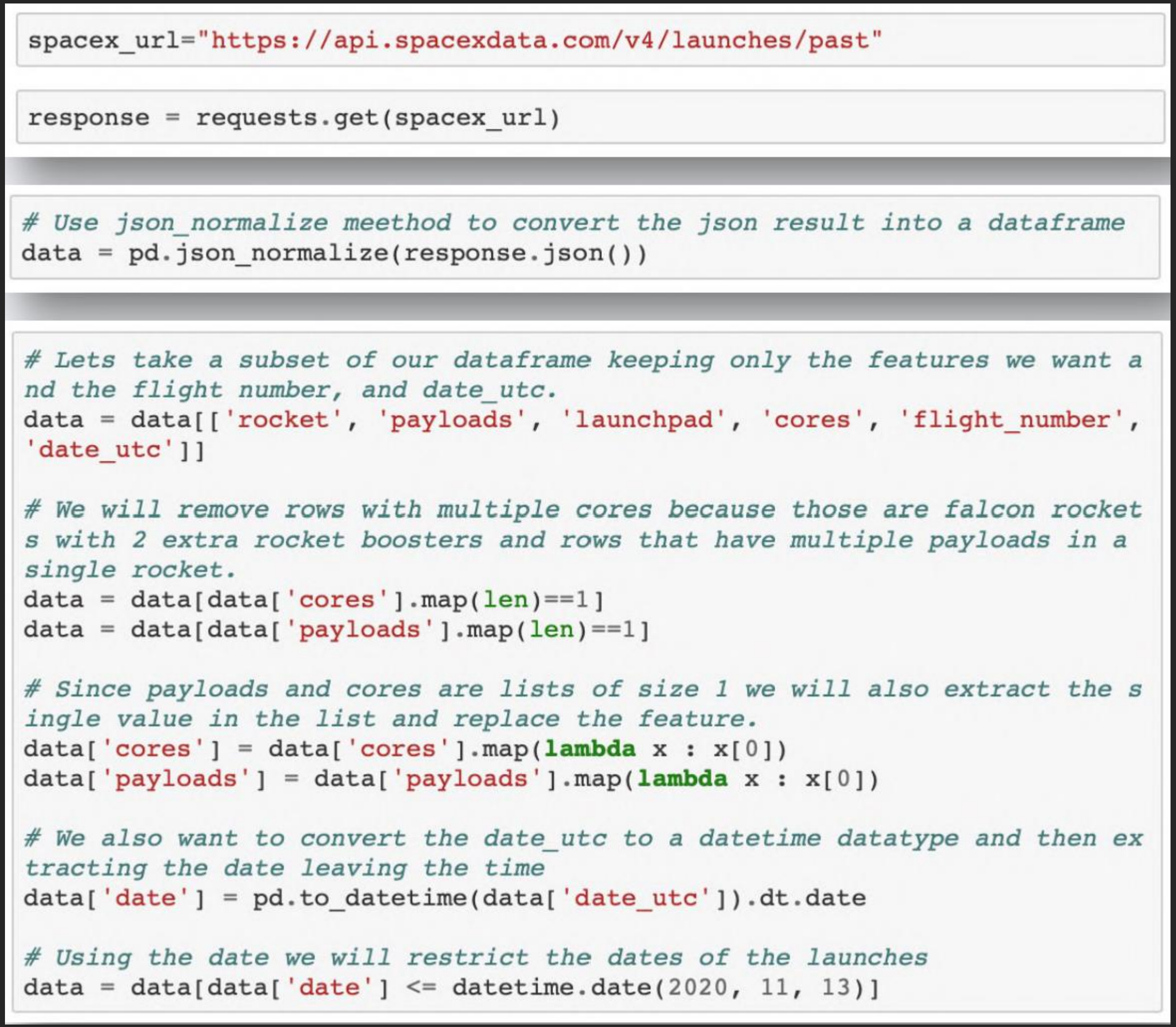
Data collection involves the systematic procedure of amassing and quantifying information related to specific variables within a pre-established framework. This accumulation of data subsequently empowers individuals to address pertinent inquiries and assess eventual outcomes. In this context, the dataset has been assembled through a combination of REST API utilization and web scraping techniques, with the primary data source being Wikipedia.

The process of gathering data through a REST API commences with initiating a GET request. Subsequently, the received response content is decoded as JSON and transformed into a structured pandas dataframe using the `json_normalize()` function. Following this, a data cleansing phase is undertaken, involving the identification and handling of missing values through appropriate filling methods.

On the other hand, web scraping entails the application of BeautifulSoup for the extraction of launch records from the HTML table format. This scraped data is then parsed and translated into a pandas dataframe, facilitating its integration into further analytical processes.

DATA COLLECTION – SPACEX API

- Get request for rocket launch using API
- Use json_normalize method to convert json result to data frame
- Performed data cleaning and filling the missing value



```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

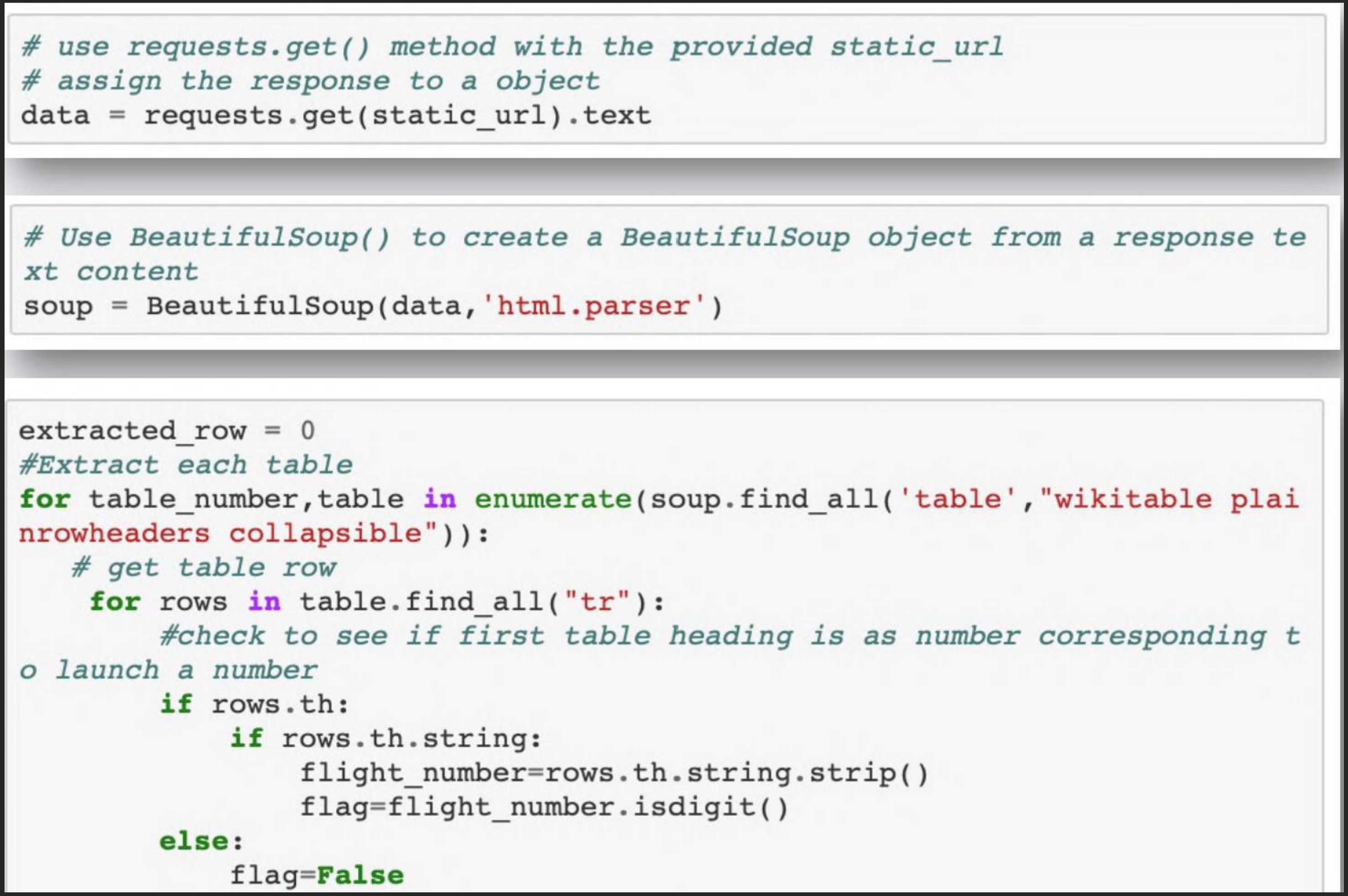
```
response = requests.get(spacex_url)
```

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

```
# Lets take a subset of our dataframe keeping only the features we want a
nd the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number',
'date_utc']]

# We will remove rows with multiple cores because those are falcon rocket
s with 2 extra rocket boosters and rows that have multiple payloads in a
single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the s
ingle value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then ex
tracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

DATA COLLECTION - SCRAPING

- Request the Falcon9 Launch Wiki page from url
- Create a BeautifulSoup from the HTML response
- Extract all column/variable names from the HTML header



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

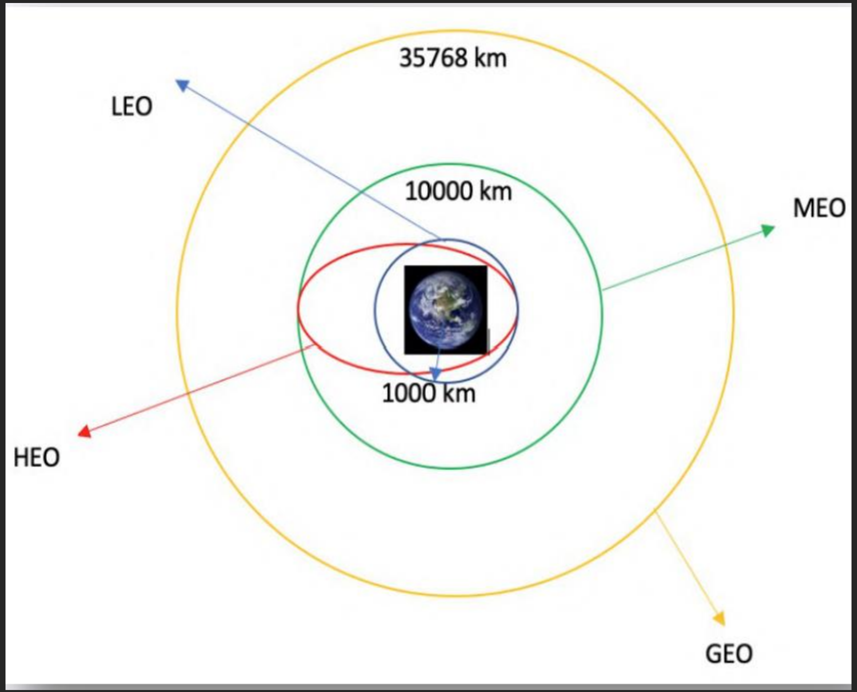
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response te
xt content
soup = BeautifulSoup(data,'html.parser')
```

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plai
nrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding t
o launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

DATA WRANGLING

Data Wrangling entails the systematic procedure of refining and harmonizing intricate and disorderly datasets, rendering them more accessible and suitable for both exploratory data analysis and streamlined usage. Our initial step involves the computation of launch quantities across distinct launch sites, followed by the derivation of counts and frequencies associated with mission outcomes according to orbit types.

Subsequently, we generate a designated label for landing outcomes by leveraging information from the existing outcome column. This categorical labeling significantly simplifies subsequent tasks, including analysis, visualization, and the implementation of machine learning techniques. Concluding the process, we conclude by exporting the refined dataset, complete with these enhancements, into a CSV format.



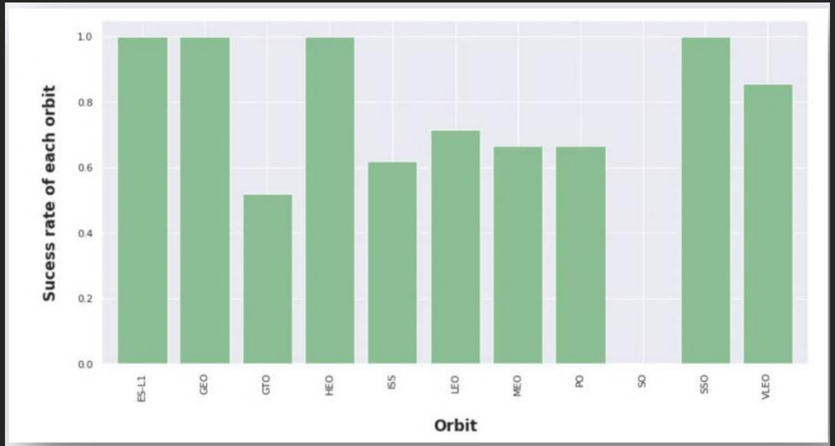
EDA WITH DATA VISUALIZATION

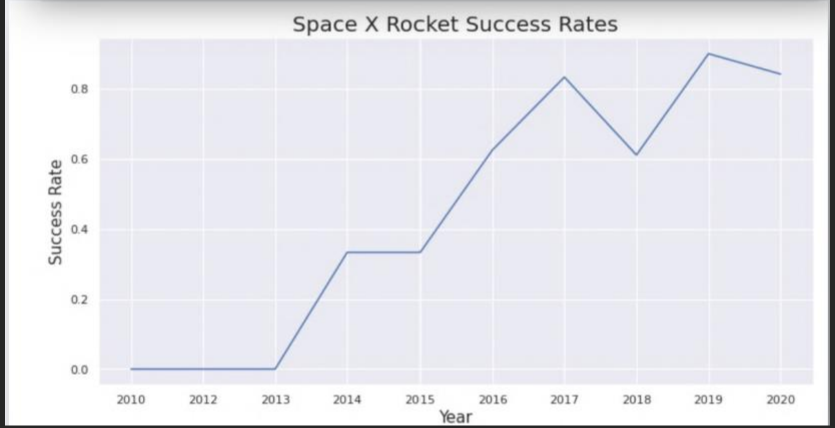
After obtaining initial insights from scatter plots that help reveal underlying relationships, we will expand our exploration by employing additional visualization techniques such as bar graphs and line plots.

Bar graphs emerge as a straightforward means of comprehending relationships between attributes. In this context, our intention is to employ bar graphs to discern the orbits that exhibit the highest likelihood of success. This visual representation will provide a clear understanding of these relationships.

Furthermore, we will leverage line graphs to unveil trends and patterns associated with specific attributes over time. In the context of this analysis, the line graph serves to showcase the annual trajectory of launch success.

Subsequently, we will engage in Feature Engineering, a pivotal step to support future success prediction modules. This involves the creation of dummy variables from categorical columns. These engineered features will contribute to enhancing the accuracy of predictive models by encapsulating important categorical information in a numerical format.





EDA with SQL

Utilizing SQL, we have executed a series of queries to gain a more comprehensive understanding of the dataset. Examples of these queries include:

- Retrieving the names of the launch sites.
- Displaying five records where launch sites commence with the string 'CCA'.
- Calculating the total payload mass transported by boosters launched by NASA (CRS).
- Determining the average payload mass carried by booster version F9 v1.1.
- Enumerating the date of the inaugural successful landing outcome on a ground pad.
- Listing the names of boosters that achieved success on a drone ship, with payload mass ranging between 4000 and 6000.
- Enumerating the total count of successful and failed mission outcomes.
- Listing the names of booster versions that carried the maximum payload mass.
- Displaying the landing outcomes that failed on a drone ship, along with their corresponding booster versions and launch site names, specifically for the year 2015.
- Ranking the count of landing outcomes, distinguishing between success and failure, within the timeframe spanning from June 4, 2010, to March 20, 2017, in descending order.

Build an Interactive Map with Folium

For the purpose of creating an interactive map visualizing launch data, we integrated latitude and longitude coordinates from each launch site. Subsequently, we incorporated circle markers around these sites, each marked with the corresponding launch site's name label.

The dataframe containing launch outcomes, categorized as either failure or success, was then allocated to classes 0 and 1. These classes were depicted on the map using Red and Green markers, respectively, employing the MarkerCluster() technique.

To further investigate proximity-related inquiries, we leveraged Haversine's formula to compute distances between launch sites and various landmarks. This analysis aimed to address queries such as:

- What is the proximity of launch sites to railways, highways, and coastlines?

- How close are launch sites to neighboring cities?

Build a Dashboard with Plotly Dash

We developed an interactive dashboard using Plotly Dash, offering users the flexibility to manipulate the data according to their requirements.

Key features of the dashboard include:

- Pie charts illustrating the cumulative launches from specific launch sites.

- Scatter plots portraying the connection between Outcome and Payload Mass (measured in kilograms) across various booster versions.

Predictive Analysis - Classification

1. Building the Model: Load the dataset into NumPy and Pandas •Transform the data and then split into training and test datasets •Decide which type of ML to use •set the parameters and algorithms to GridSearchCV and fit it to dataset.
2. Evaluating the Model: Check the accuracy for each model •Get tuned hyperparameters for each type of algorithms. •plot the confusion matrix.
3. Improving the Model: Use Feature Engineering and Algorithm Tuning
4. Find the Best Model: The model with the best accuracy score will be the best performing model

Results:

• Exploratory data analysis results

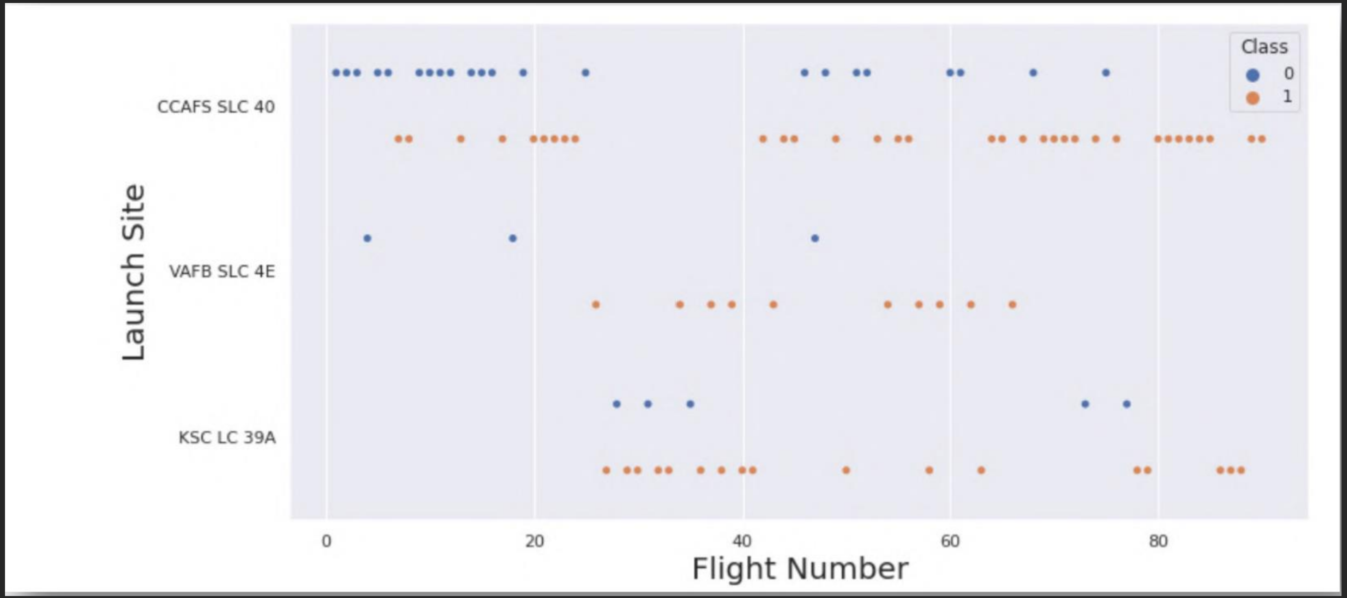
• Interactive analytics demo in screenshots

• Predictive analysis results

EDA Results

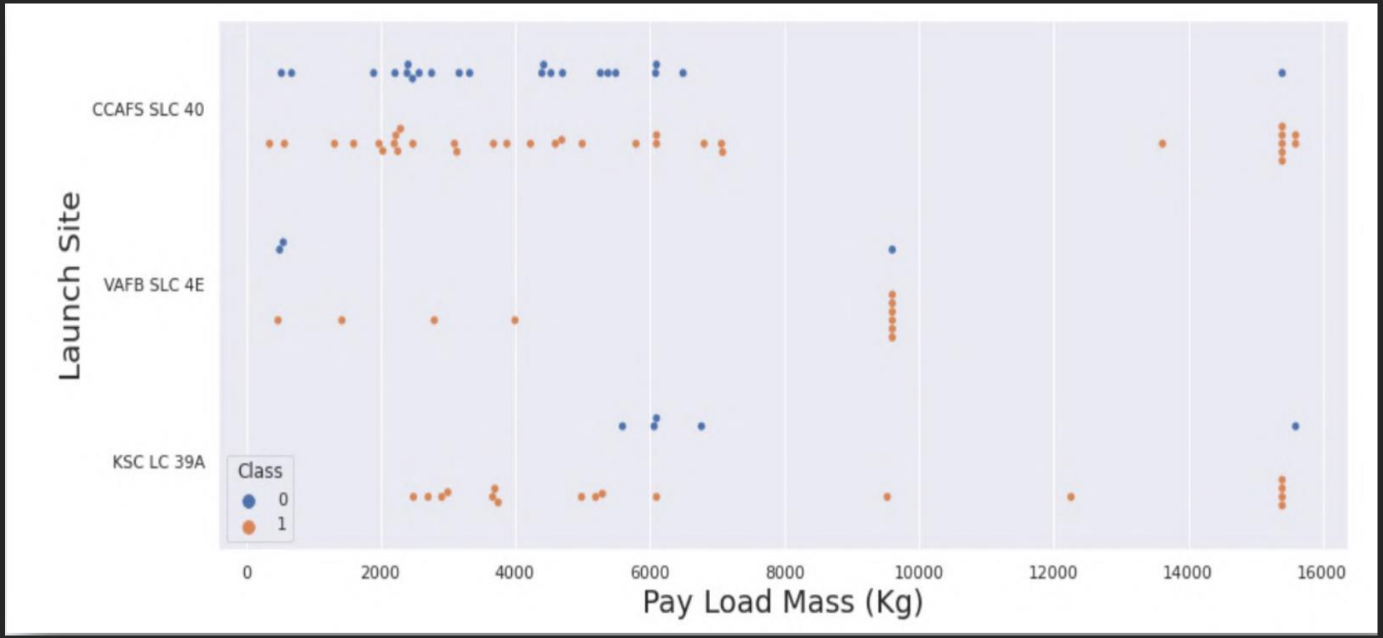
FLIGHT NUMBER VS. LAUNCH SITE

This scatter plot illustrates a relationship where an increase in the number of flights from a launch site is associated with a higher success rate. However, the launch site CCAFS SLC40 contrasts this trend by exhibiting the least consistent pattern in this regard.



PAYLOAD VS. LAUNCH SITE

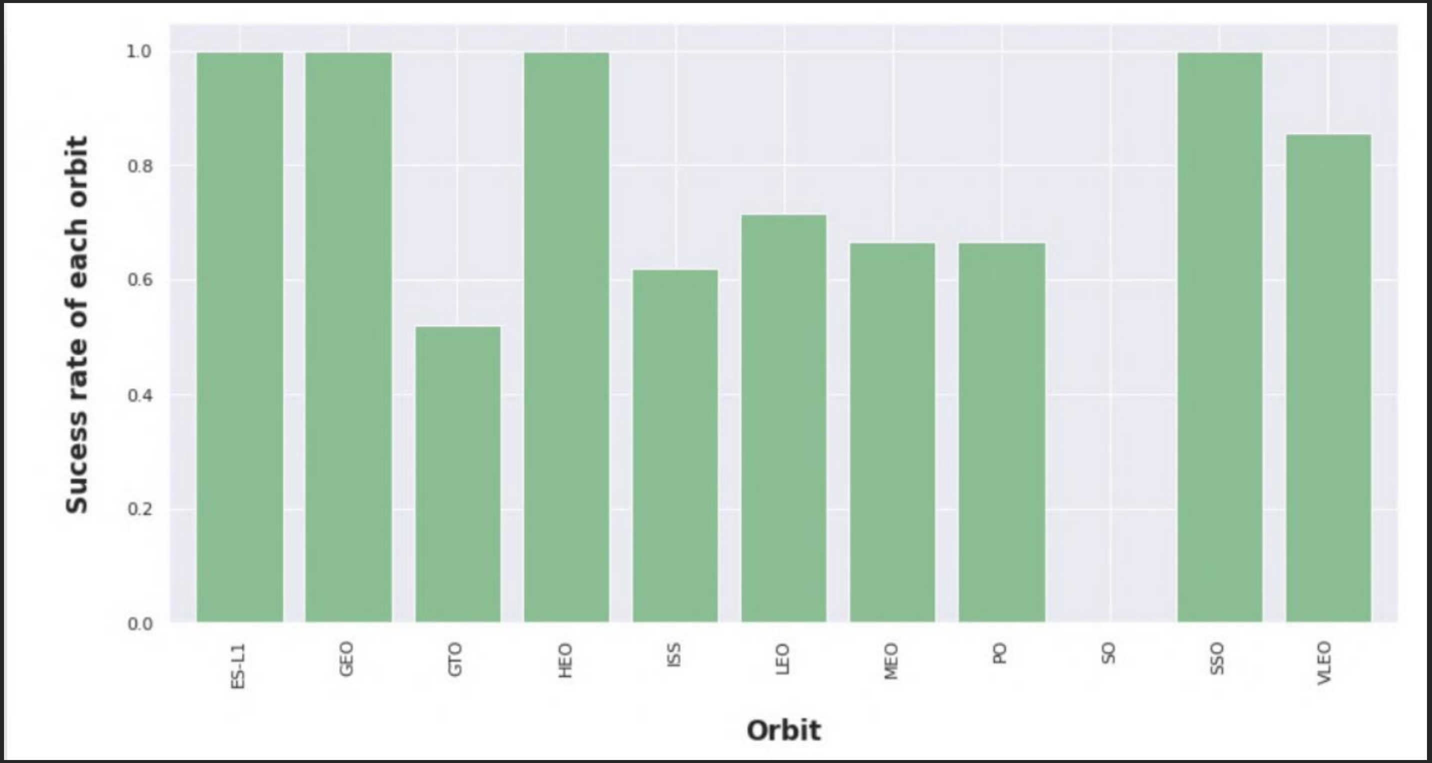
This scatter plot indicates that once the payload mass exceeds 7000kg, there is a significant increase in the probability of achieving a successful outcome. However, there isn't a distinct pattern suggesting that the launch site's success rate is dependent on the payload mass.



SUCCESS RATE VS. ORBIT TYPE

This illustration portrays how different orbits can impact landing outcomes. Certain orbits, like Sun-Synchronous (SSO), High Earth Orbit (HEO), Geostationary Orbit (GEO), and Earth-Sun L1 (ES-L1), exhibit a 100% success rate. Conversely, the Sun Orbit (SO) shows a 0% success rate.

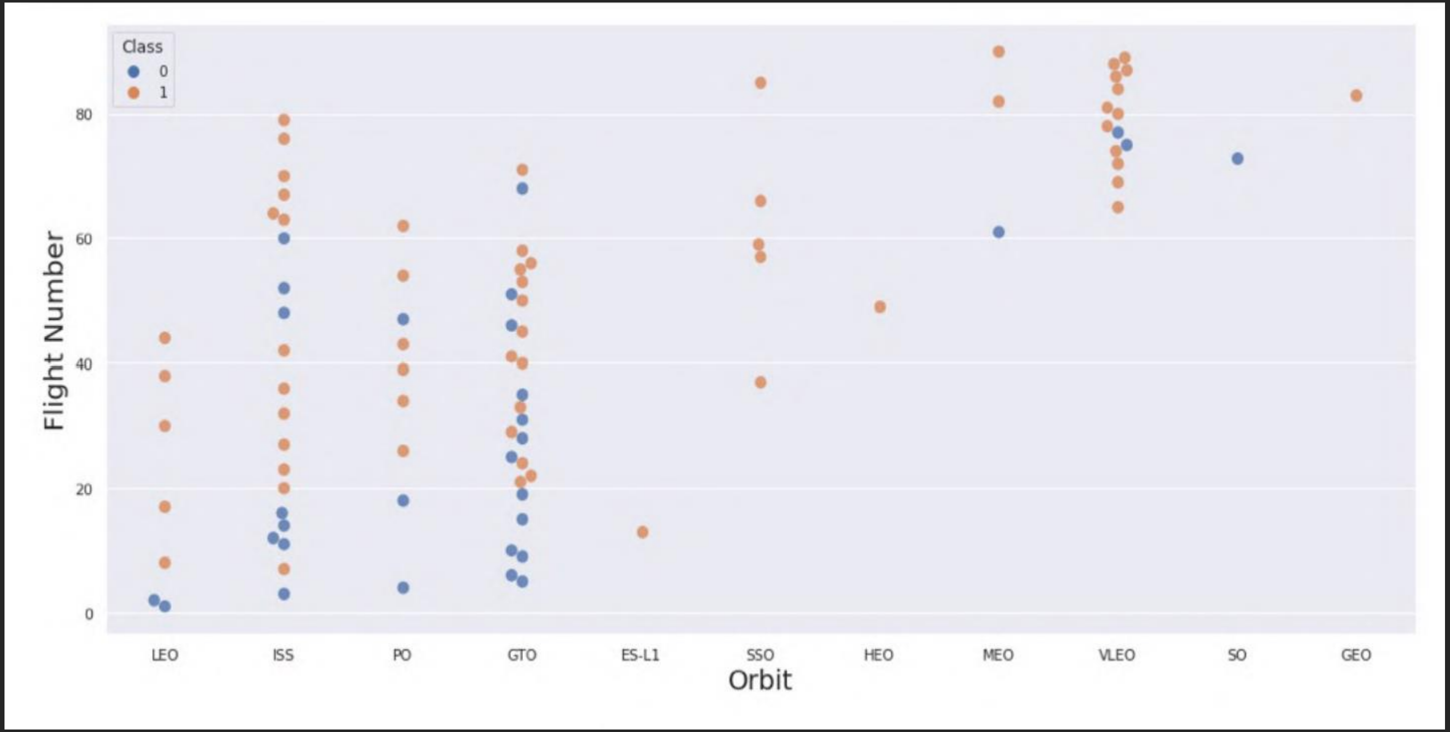
However, upon closer examination, it becomes evident that some of these orbits, such as GEO, SO, HEO, and ES-L1, have only occurred once. This limited data suggests that more datasets are necessary to identify any meaningful patterns or trends before drawing any conclusive observations.



FLIGHT NUMBER VS. ORBIT TYPE

This scatter plot demonstrates a general trend where a higher flight count within each orbit is associated with an increased success rate, particularly in Low Earth Orbit (LEO). However, this relationship doesn't hold true for Geostationary Transfer Orbit (GTO), as there seems to be no correlation between the two variables.

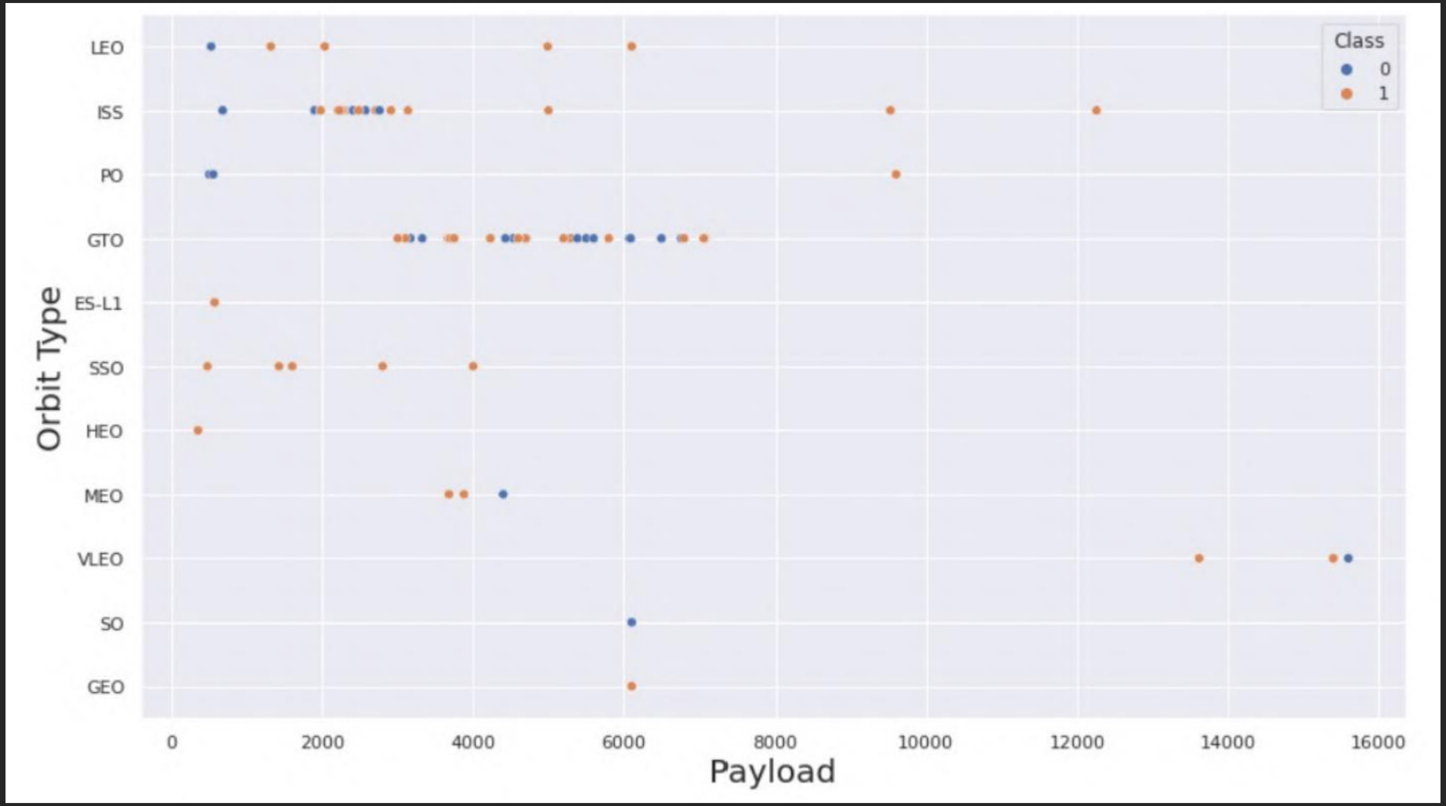
It's important to note that orbits with only a single occurrence should be excluded from the observation, as more data is required to make any conclusive statements about their patterns.



PAYLOAD VS. ORBIT TYPE

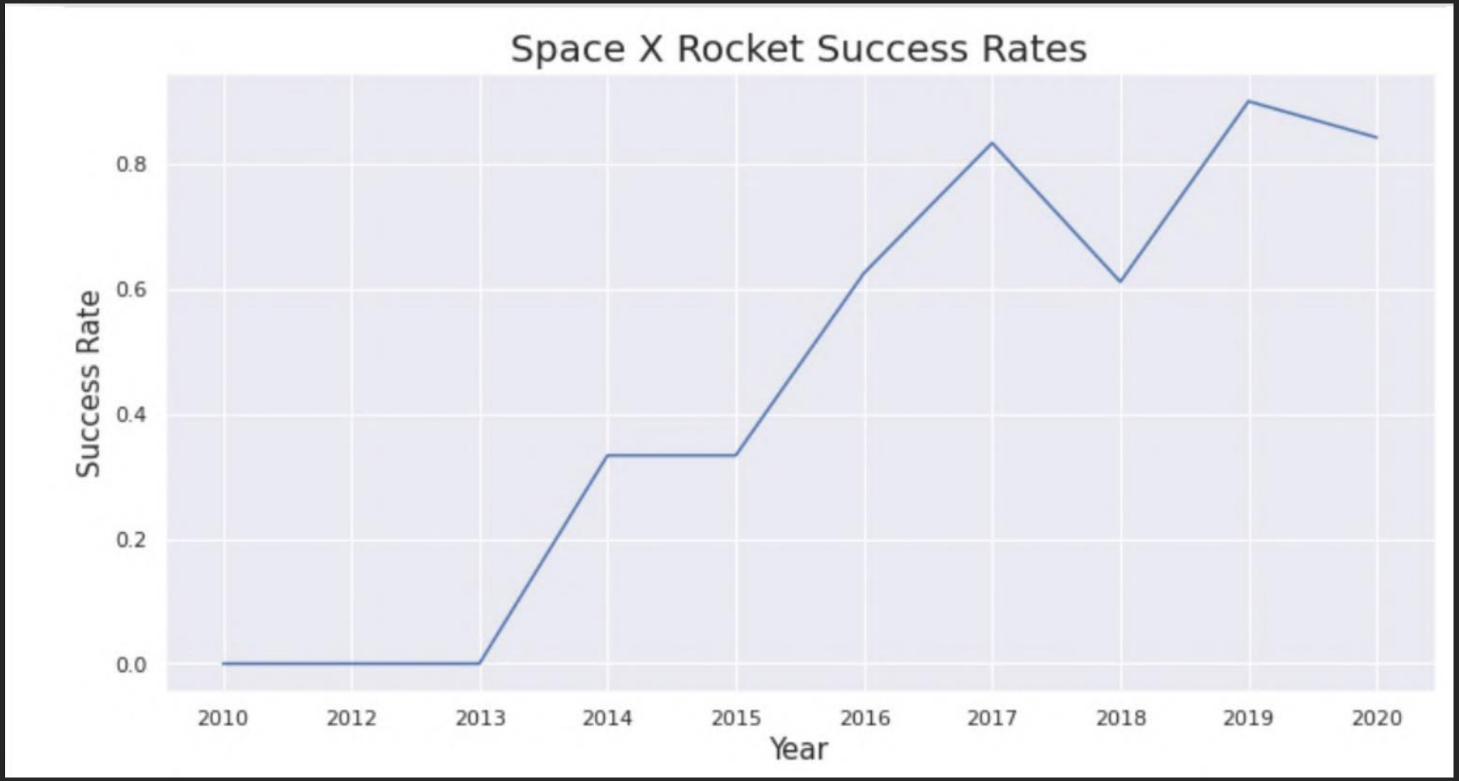
Increased payload weight has a beneficial effect on Low Earth Orbit (LEO), International Space Station (ISS), and Polar Orbit (P0). However, it adversely affects Medium Earth Orbit (MEO) and Very Low Earth Orbit (VLEO). The Geostationary Transfer Orbit (GTO) appears to lack a discernible correlation between these attributes.

Meanwhile, Sun Orbit (SO), Geostationary Orbit (GEO), and High Earth Orbit (HEO) orbits require more data to identify any discernible patterns or trends.



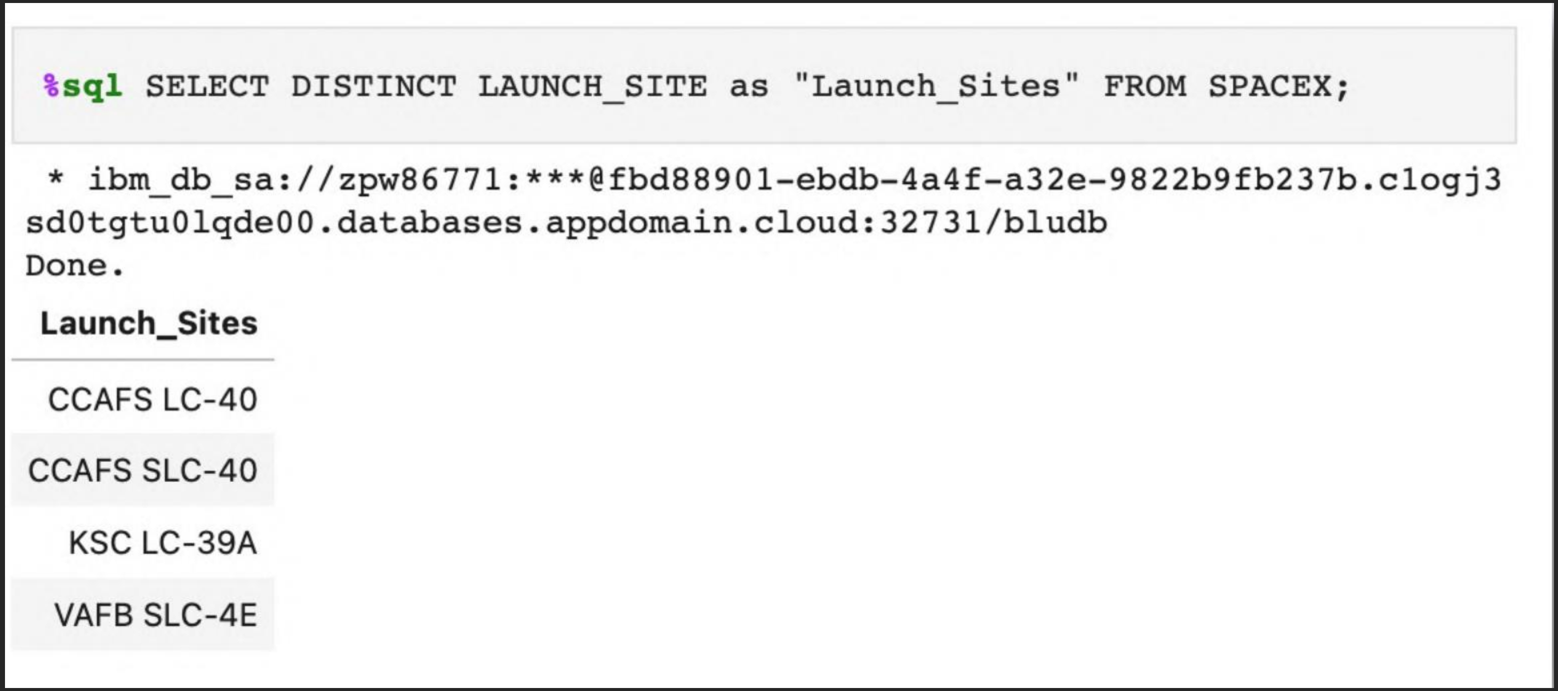
LAUNCH SUCCESS YEARLY TREND

These figures clearly illustrate a consistent upward trend from the year 2013 to 2020. If this trend persists in the coming years, the success rate is projected to steadily rise until it reaches a perfect 100% success rate.



LAUNCH SITE NAMES

We utilized the keyword "DISTINCT" to display exclusively unique launch sites within the SpaceX dataset.



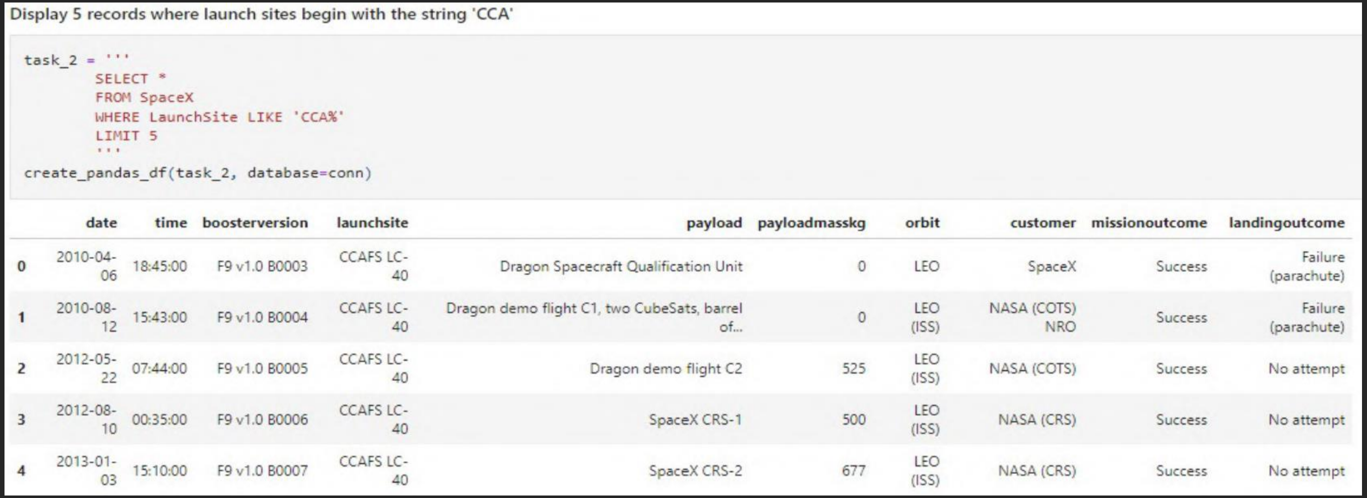
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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

'CCA' LAUNCH SITE NAMES

We employed the query to exhibit five records where the launch sites commence with the prefix "CCA."



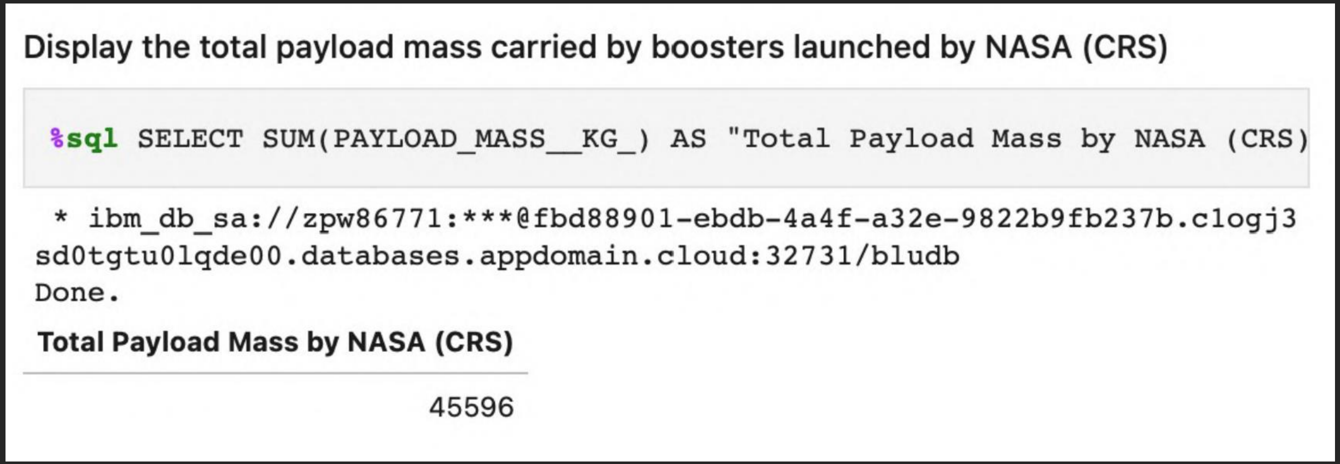
Display 5 records where launch sites begin with the string 'CCA'

```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

TOTAL PAYLOAD MASS

By executing the following query, we computed the cumulative payload transported by NASA's boosters to be 45,596.



Display the total payload mass carried by boosters launched by NASA (CRS)

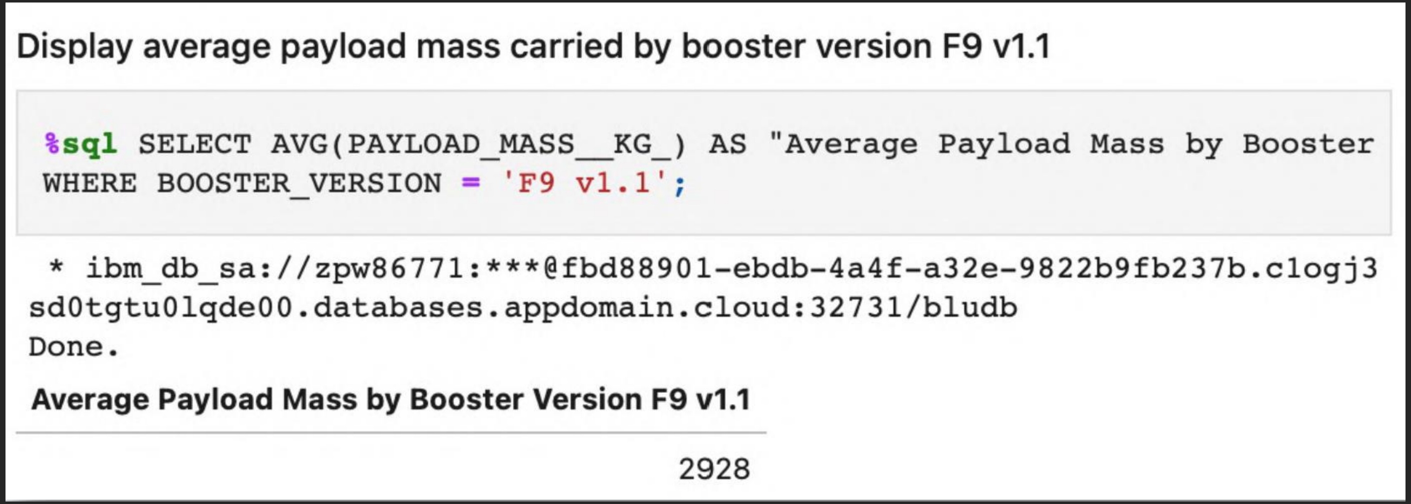
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Payload Mass by NASA (CRS)
45596

AVERAGE PAYLOAD MASS BY F9 V1.1

We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4



Display average payload mass carried by booster version F9 v1.1

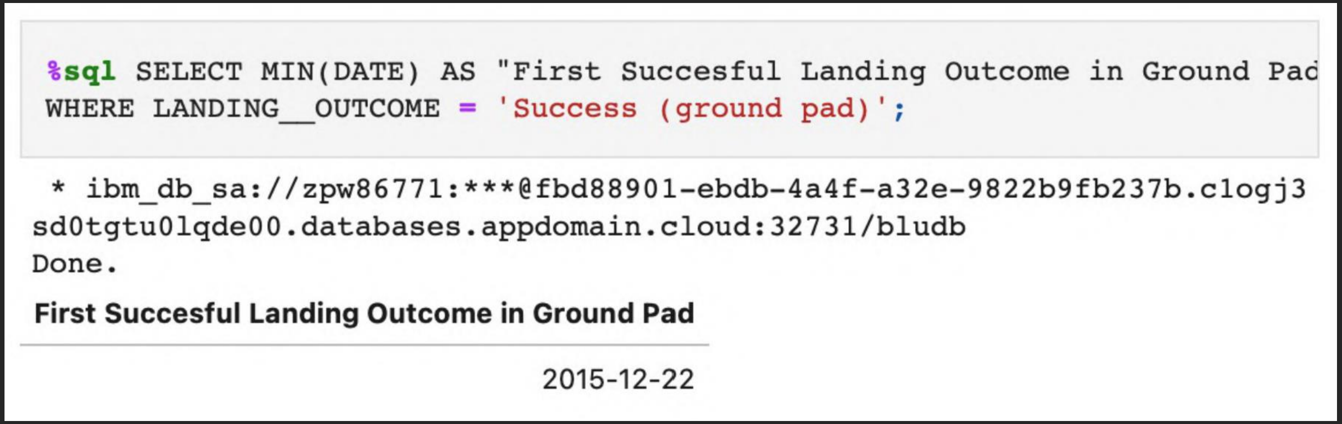
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

THE DATE OF THE FIRST SUCCESSFUL GROUND LANDING.

We utilized the min() function to derive the result. Upon observation, we noted that the date of the earliest successful ground landing was December 22nd, 2015.



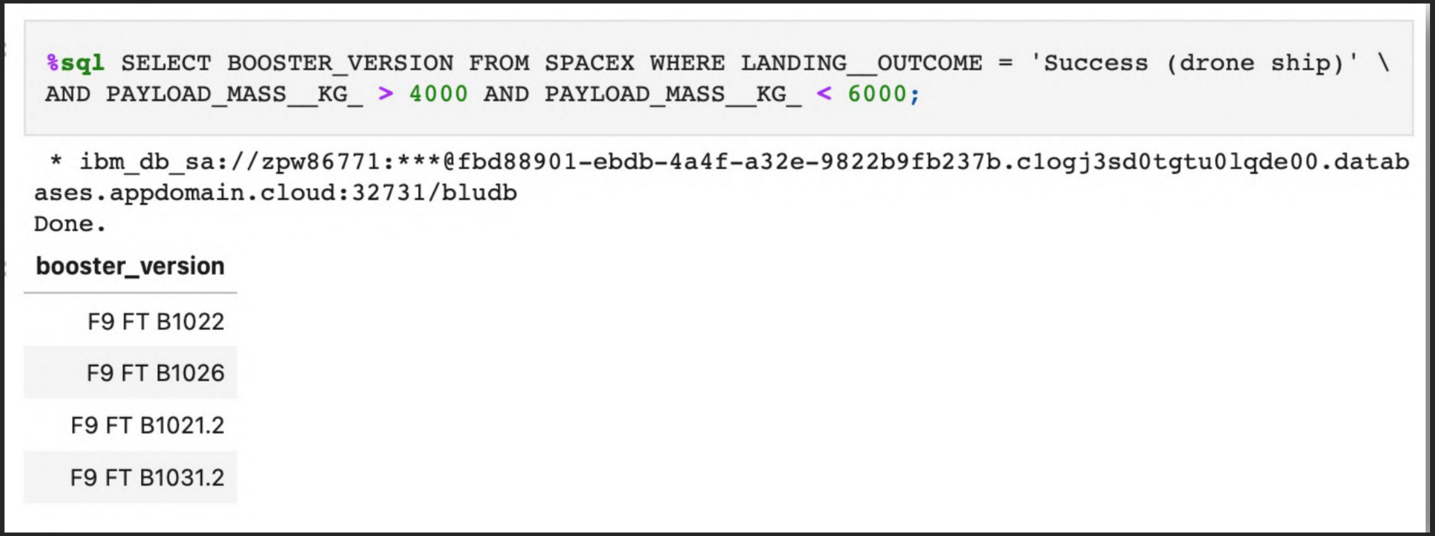
```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

ACHIEVED SUCCESSFUL LANDINGS ON DRONE SHIPS WITH PAYLOADS RANGING BETWEEN 4000 AND 6000.

We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000



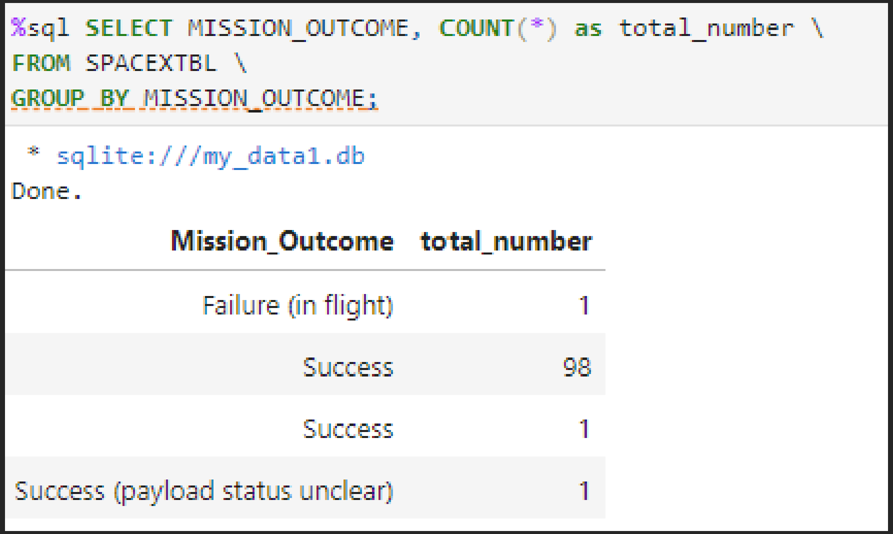
```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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

TOTAL NUMBER OF SUCCESSFUL AND FAILURE MISSION OUTCOMES

We used wildcard like '%' to filter for WHERE Mission_Outcome was a success or a failure.



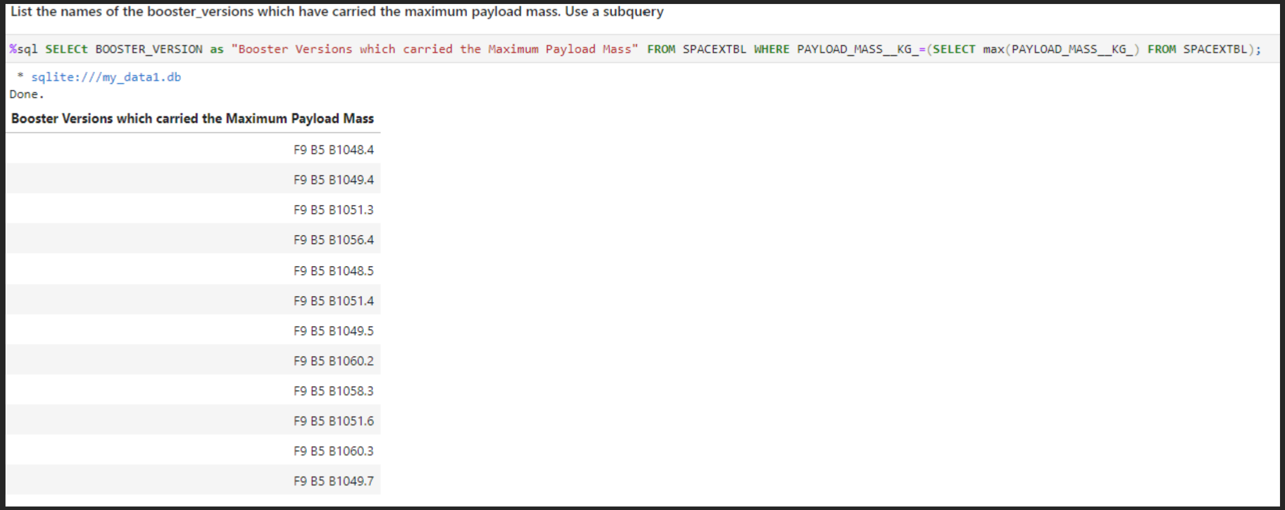
```
%sql SELECT MISSION_OUTCOME, COUNT(*) as total_number \
FROM SPACEXTBL \
GROUP BY MISSION_OUTCOME;
```

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

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

BOOSTERS CARRIED MAXIMUM PAYLOAD

We identified the booster that transported the highest payload by utilizing a subquery within the WHERE clause in conjunction with the MAX() function.



List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

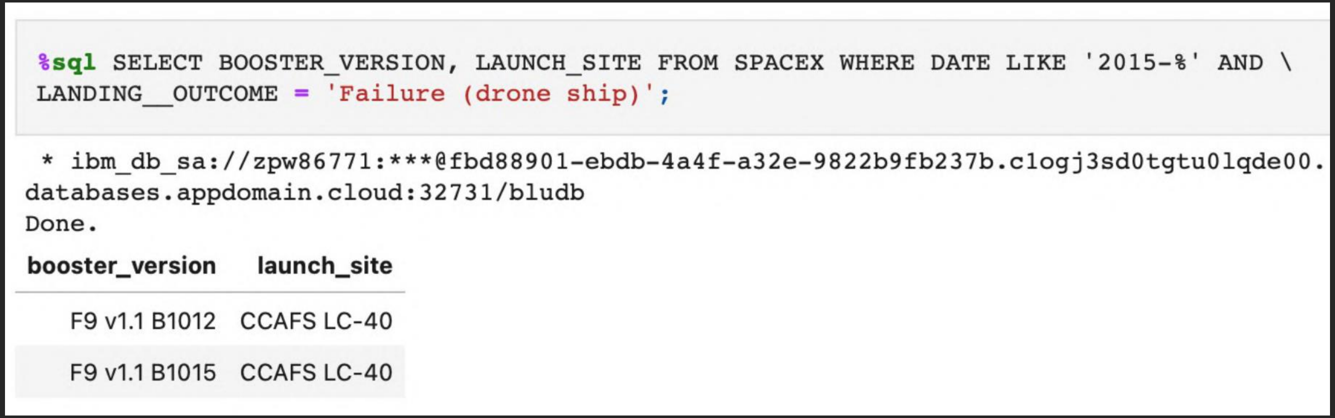
```
%sql SELECt BOOSTER_VERSION as "Booster Versions which carried the Maximum Payload Mass" FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_=(SELECT max(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Booster Versions which carried the Maximum Payload Mass
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

We employed a combination of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter and extract information about unsuccessful landing outcomes on drone ships, including their booster versions and launch site names, specifically for the year 2015.



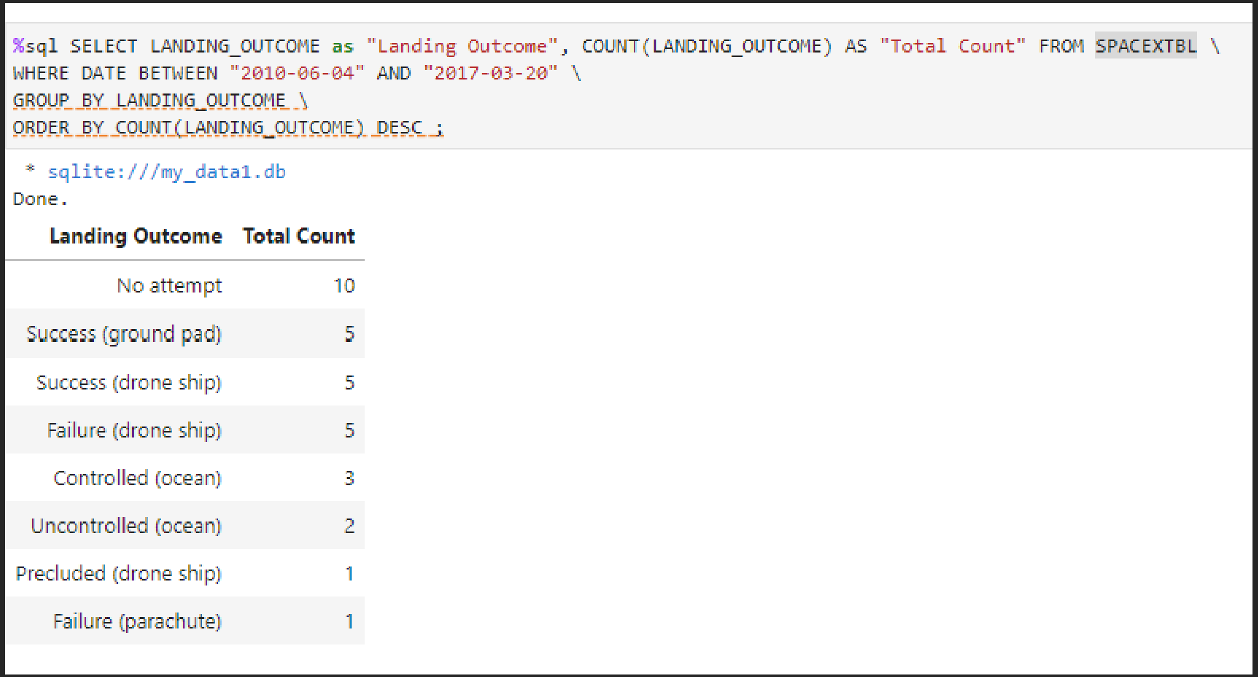
```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.
databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

RANK LANDING OUTCOMES BETWEEN 2010-06-04 AND 2017-03-20

We chose the Landing outcomes and the COUNT of landing outcomes from the dataset. Then, we utilized the WHERE clause to filter for landing outcomes that fall between the dates June 4, 2010, and March 20, 2010. Subsequently, we applied the GROUP BY clause to group the landing outcomes and employed the ORDER BY clause to arrange the grouped landing outcomes in descending order.

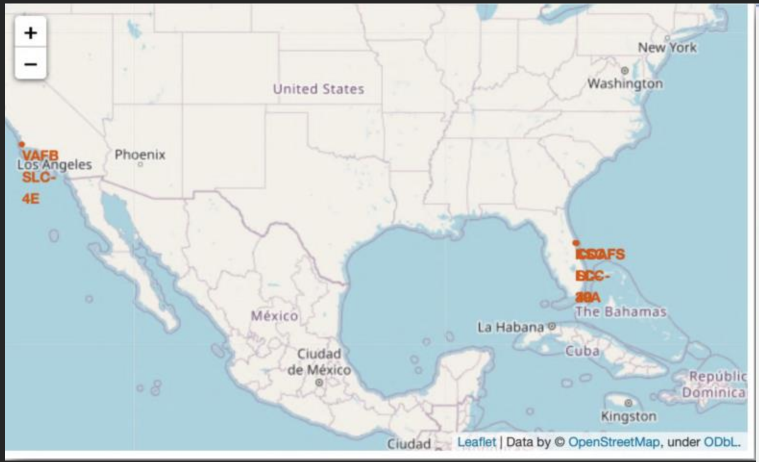


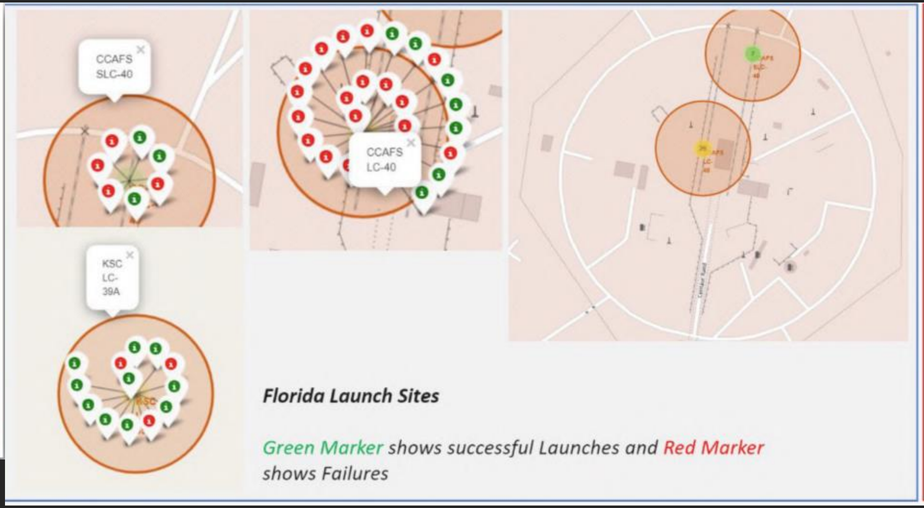
```
%sql SELECT LANDING_OUTCOME as "Landing Outcome", COUNT(LANDING_OUTCOME) AS "Total Count" FROM SPACEXTBL \
WHERE DATE BETWEEN "2010-06-04" AND "2017-03-20" \
GROUP BY LANDING_OUTCOME \
ORDER BY COUNT(LANDING_OUTCOME) DESC ;
```

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

Landing Outcome	Total Count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

LAUNCH SITES ANALYSIS



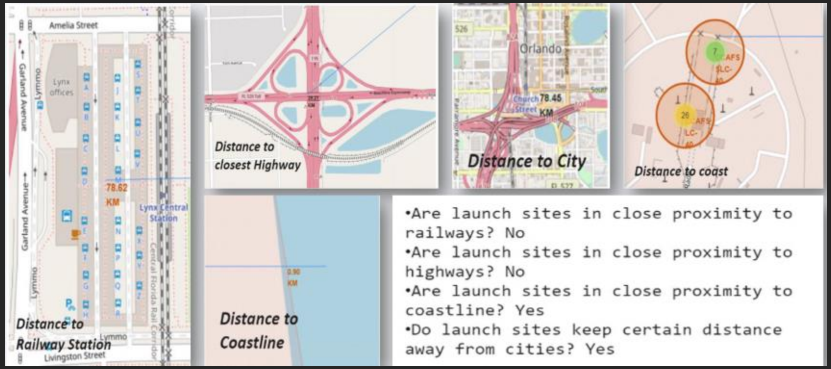


Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

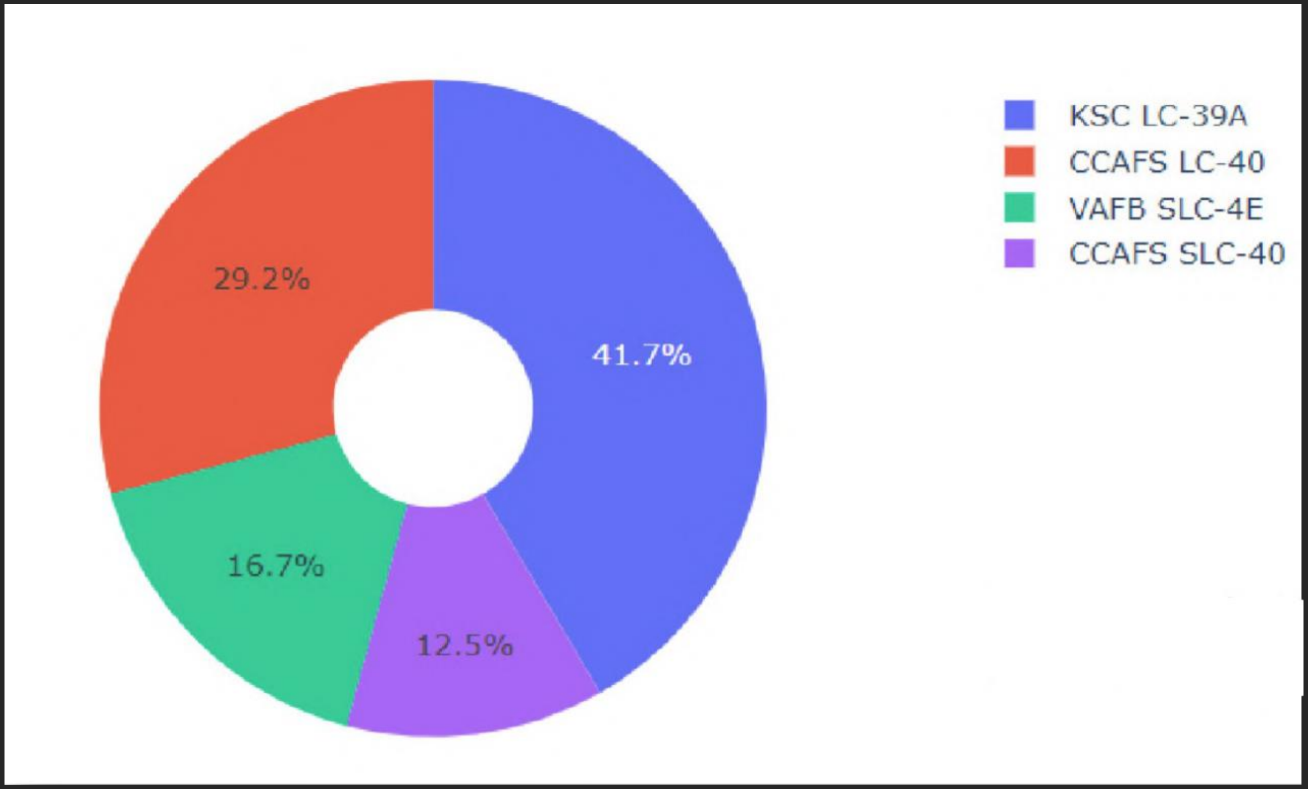
37



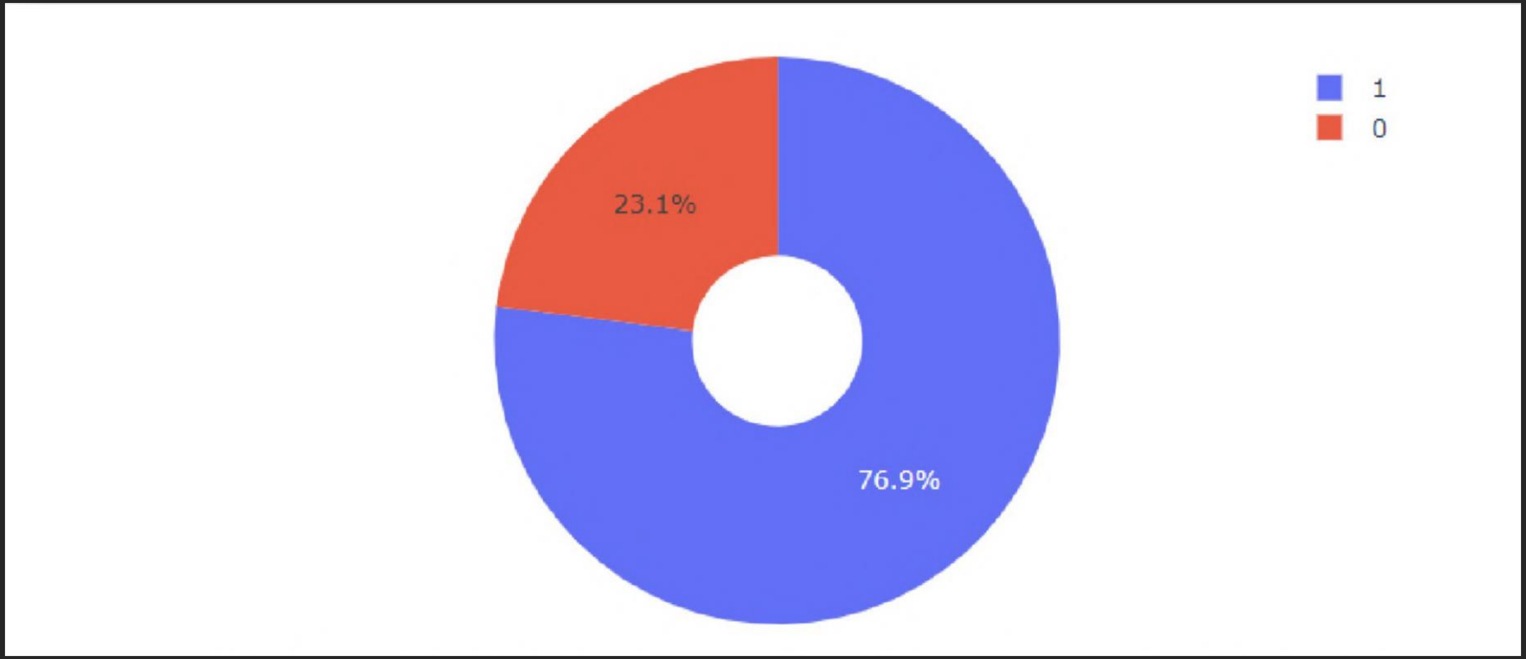
•Are launch sites in close proximity to railways? No
•Are launch sites in close proximity to highways? No
•Are launch sites in close proximity to coastline? Yes
•Do launch sites keep certain distance away from cities? Yes

Dashboard with Plotly Dash

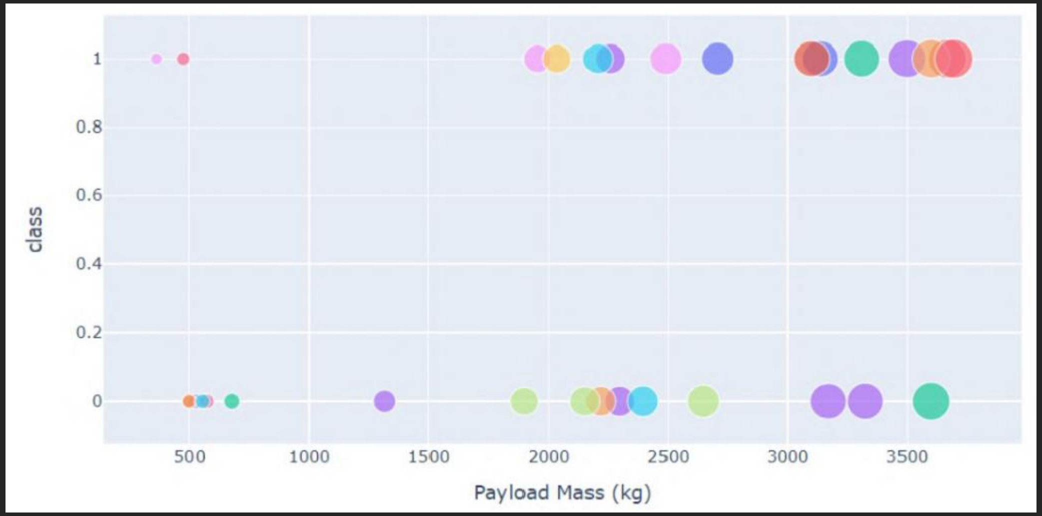
Success Percentage by each Sites

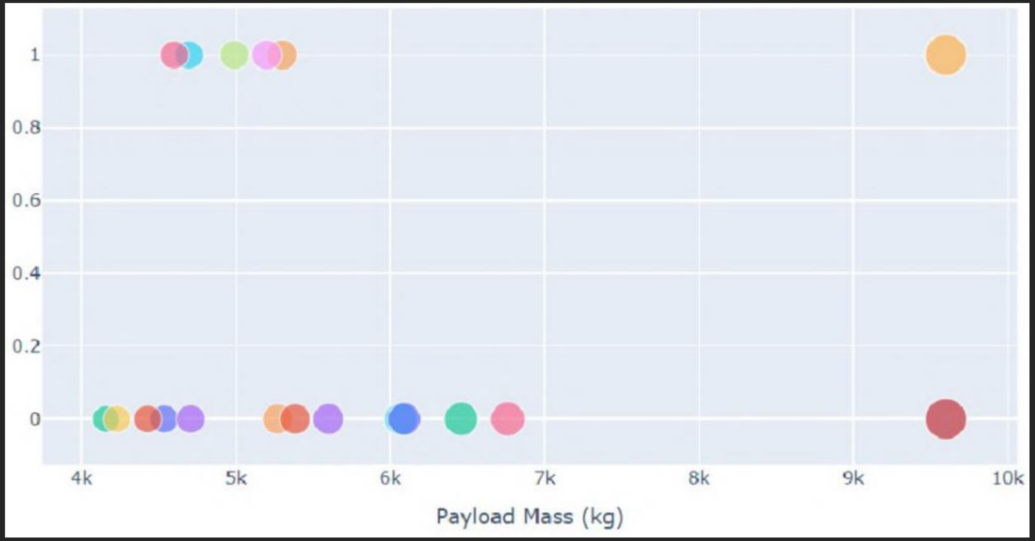


Highest Launch-Success Ratio for KSC LC-39A



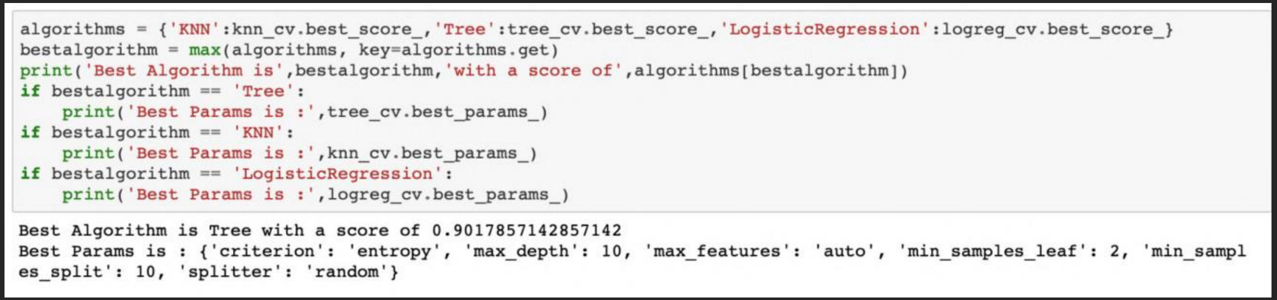
Payload vs Launch Outcome Scatter Plot





Predictive Analysis

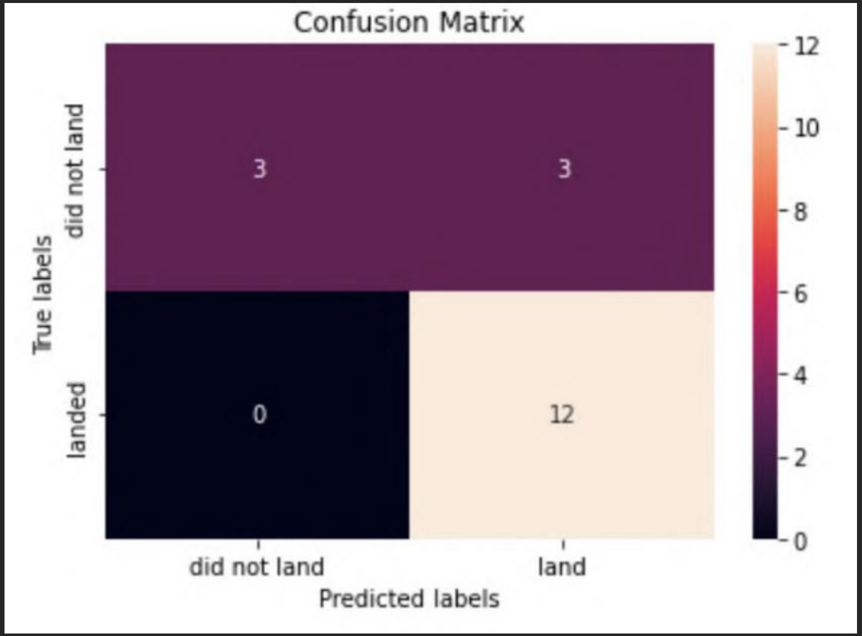
CLASSIFICATION ACCURACY(TREE ALGORITHM)



```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

CONFUSION MATRIX



Conclusion

- We can draw the following conclusions based on the analysis:
- Lighter payloads (defined as 4000kg and below) exhibited superior performance compared to heavier payloads.
- Commencing from 2013, SpaceX's launch success rate has been consistently increasing, proportionate to the passing years up to 2020, indicating a promising trajectory towards perfect launches in the future.
- Among all launch sites, KSC LC-39A demonstrated the highest success rate with 76.9% of launches being successful.
- The Sun-Synchronous Orbit (SSO) showcased the highest success rate of 100%, and this success was observed across multiple occurrences.
- The Tree Classifier Algorithm stands out as the most effective Machine Learning approach for this dataset.

These conclusions provide valuable insights into the dataset and can guide decision-making and further analysis.