IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Charles Batista
December 15, 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection
  - Data Wrangling
  - EDA With Data Visualization
  - EDA With SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive Analysis
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

- Project background and context

The era of commercial space exploration is upon us, with several companies paving the way for affordable space travel. Among these, SpaceX stands out as one of the most successful, thanks in part to the cost-effectiveness of their rocket launches.

SpaceX promotes Falcon 9 rocket launches on its website at a price of 62 million dollars, in stark contrast to other providers whose costs soar to over 165 million dollars per launch. A significant factor contributing to these savings is SpaceX's innovative practice of reusing the first stage of its rockets.

Hence, our objective is to forecast the successful landing of the Falcon 9's first stage. By determining the likelihood of a successful landing, we can ascertain the overall cost of a launch. This predictive information becomes particularly valuable in scenarios where alternative companies may seek to compete with SpaceX in bidding for rocket launches.

- Problems you want to find answers
  - Analyzing the Relationship Between Various Rocket Variables and the Success Rate of Landings
  - Determining Optimal Parameters for Maximizing Successful Rocket Landings

Section 1

Methodology

Methodology

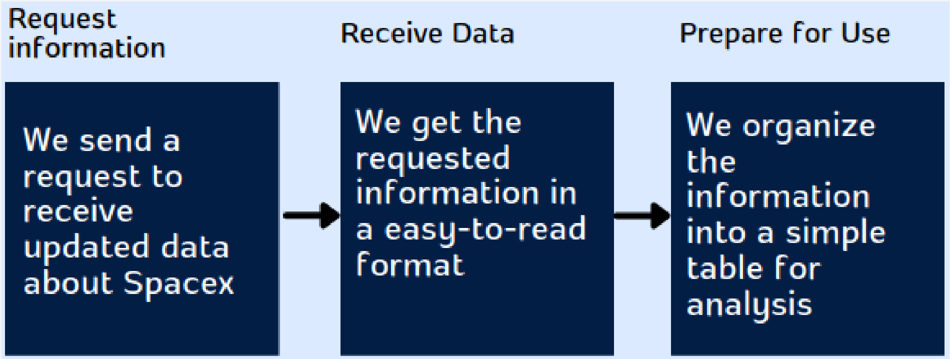
Executive Summary

- Data collection methodology:
    - Data was collected from Wikipedia using Web scraping and the SpaceX API itself
- Perform data wrangling
    - Converting Results to Training Labels for Successful and Unsuccessful Booster Landings
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - Identifying Optimal Hyperparameters for SVM, Classification Trees, and Logistic Regression

Data Collection

- I collected data using two methods: SpaceX's data tool and by extracting information from tables on SpaceX-related Wikipedia pages.
- From SpaceX's tool, I retrieved details like Flight Number, Date, Booster Version, Payload Mass, Orbit, Launch Site, and more. Additionally, I gathered data from Wikipedia tables, which included Flight Number, Launch Site, Payload, Payload Mass, Orbit, Customer, and Launch Outcome.
- This compilation of information provides a comprehensive dataset about SpaceX launches

SpaceX API:



Request information

We send a request to receive updated data about Spacex

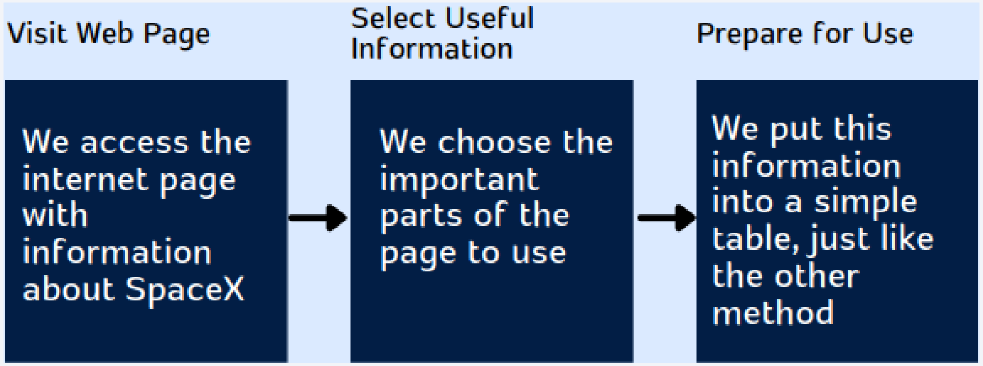
Receive Data

We get the requested information in a easy-to-read format

Prepare for Use

We organize the the information into a simple table for analysis

Web Scraping:



Visit Web Page

We access the internet page with information about SpaceX

Select Useful Information

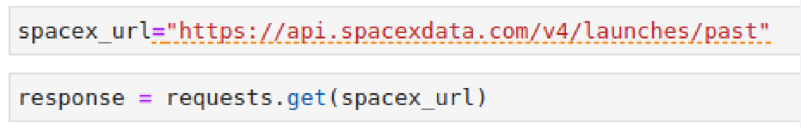
We choose the important parts of the page to use

Prepare for Use

We put this information into a simple table, just like the other method

Data Collection – SpaceX API

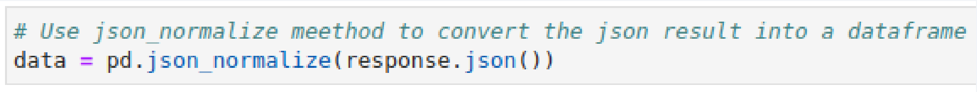
- Requesting rocket launch data from SpaceX API



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

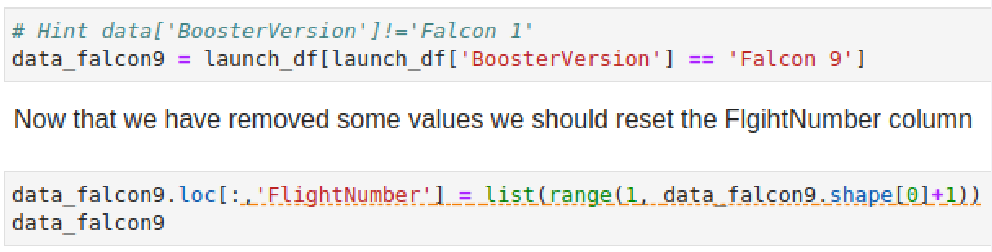
```
response = requests.get(spacex_url)
```

2. Converting Response to a JSON file



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

3. Using custom functions to clean data



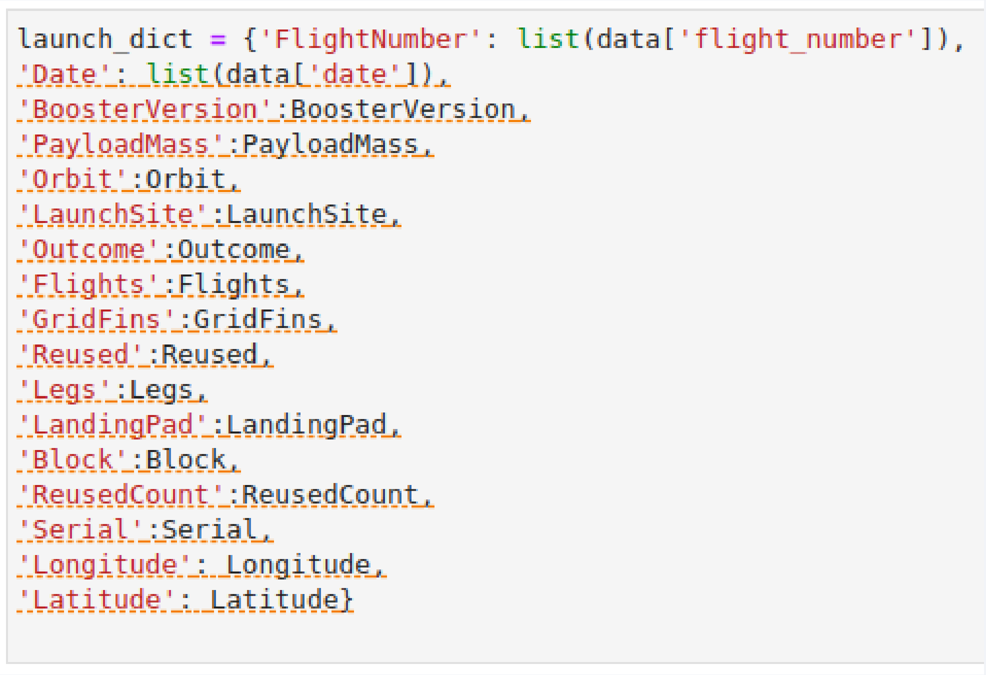
```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = launch_df[launch_df['BoosterVersion'] == 'Falcon 9']
```

Now that we have removed some values we should reset the FlgihtNumber column

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

- [GitHub Data Collection]()

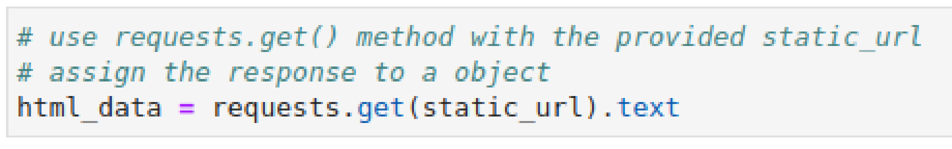
4. Combining the columns into a dictionary to create a data frame



```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

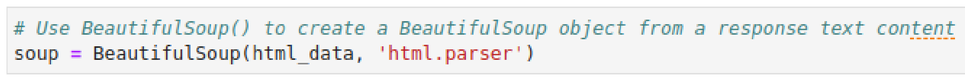
Data Collection - Scraping

1. Getting response from HTML



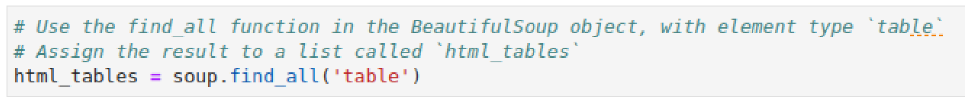
```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url).text
```

2. Creating a BeautifulSoup object



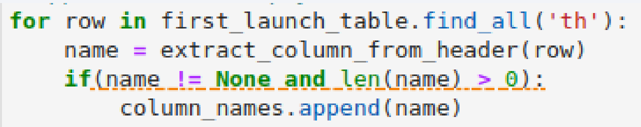
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data, 'html.parser')
```

3. Finding all tables and assigning the result to a list



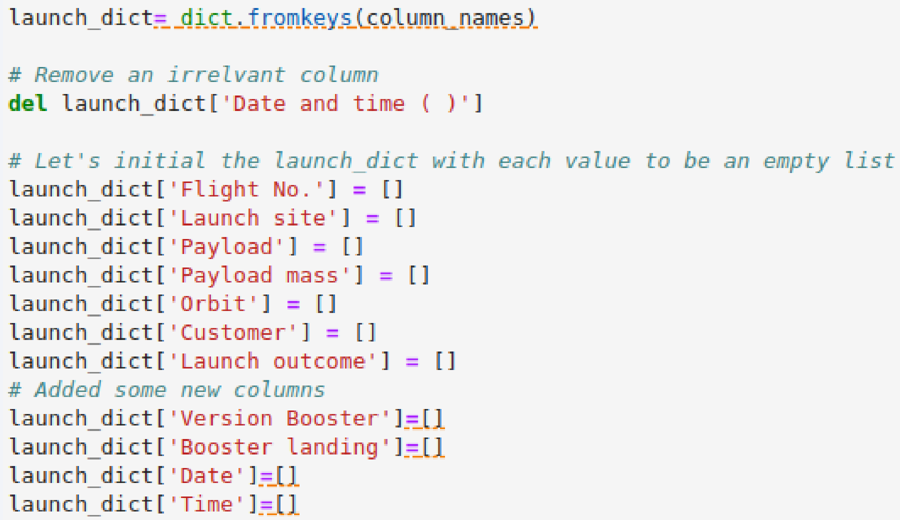
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

4. Extracting column name one by one



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

5. Creating an empty dictionary with keys

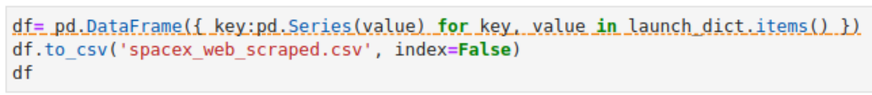


```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Creating a Dataframe and exporting in to a CSV



```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
df.to_csv('spacex_web_scraped.csv', index=False)
df
```

- Github URL - Scraping

Data Wrangling

- In the dataset, some instances show that the booster didn't land successfully, and in some cases, it tried but failed due to accidents
  - True Ocean: The mission successfully landed in a specific part of the ocean.
  - False Ocean: The mission did not successfully land in the designated ocean area.
  - True RTLS: The mission successfully landed on the ground pad.
  - False RTLS: The mission did not successfully land on the ground pad.
  - True ASDS: The mission successfully landed on the drone ship.
  - False ASDS: The mission did not successfully land on the drone ship.

- We can label these outcomes for training purposes.
  - Successful = 1
  - Failed = 0
- GitHub Url - Data Wrangling

EDA with Data Visualization

- Scatter chart:
  - Flight number vs Launch Site
  - Payload vs Launch Site
  - Flight Number vs Orbit Type
  - Payload vs Orbit type
  - A scatter plot illustrates how one thing changes with another. We call this connection a correlation. This type of plot is typically made with a lot of data points.
- Bar chart:
  - Orbit Type vs Success Rate
  - A bar chart makes comparing data between different groups easy. One side represents categories, and the other side shows specific values. The chart helps reveal relationships between the two sides.
- Line chart:
  - Year vs Success rate
  - A line chart clearly displays data variables and trends, making it useful for predicting outcomes of data not yet recorded.

  - GitHub - Data Visualization

EDA with SQL

- Putting the dataset into the right table in a Db2 database and using SQL queries to find answers to the following questions:
    - Displaying the names of the unique launch sites in the space mission
    - Displaying 5 records where launch sites begin with the string 'CCA'
    - Displaying the total payload mass carried by boosters launched by NASA(CRS)
    - Displaying average payload mass carried by booster version F9 v1.1
    - Listing the date when the first successful landing outcome in ground pad was achieved
    - Listing the total number of successful and failure mission outcomes
    - Listing the names of the booster versions which have carried the maximum payload mass
    - Listing the failed landing outcomes in drone ship, their booster version and launch site names for in year 2015
    - Ranking 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.

GitHub URL - EDA with SQL

Build an Interactive Map with Folium

- Objects created and added to a folium map:
  - Markers that show all launch sites on a map
  - Markers that show the success and failed launches for each site on the map
  - Lines that show the distances between a launch site to its proximities

- By adding these objects the following geographical patterns about launch sites are found:
  - Are launch sites in close proximity to railways? Yes
  - Are launch sites in close proximity to highways? Yes
  - Are launch sites in close proximity to coastline? Yes
  - Do launch sites keep certain distance away from cities? Yes

  - GitHub

Build a Dashboard with Plotly Dash

- The dashboard application contains a pie chart and a scatter point chart.
- A pie chart displaying the total number of successful launches at various sites. This chart can be chosen to show the distribution of successful landings across all launch sites or to highlight the success rate of individual sites.
- Scatter chart
- This chart illustrates the connection between outcomes and payload mass (in kg) using various boosters. It has two inputs: all sites and individual sites, with payload mass adjustable on a slider ranging from 0 to 10000 kg. The purpose of this chart is to assess the influence of launch site, payload mass, and booster version categories on mission success.
- [Github URL(Raw code)]

Predictive Analysis (Classification)

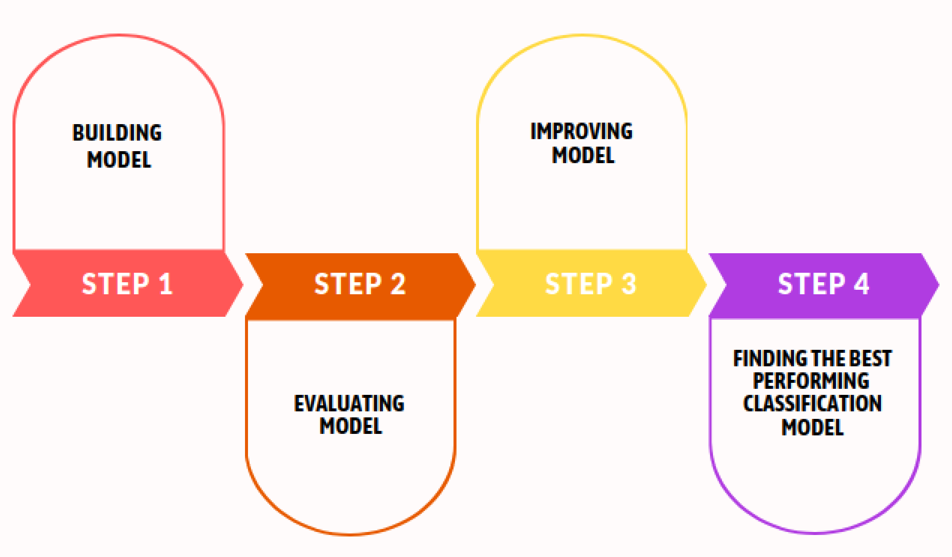
1. **Performing Exploratory Data Analysis (EDA) and Defining Training Labels:**

- Conducting exploratory data analysis to identify patterns and create training labels. This involves adding a class column and standardizing the data. Finally, splitting the dataset into training and test data.

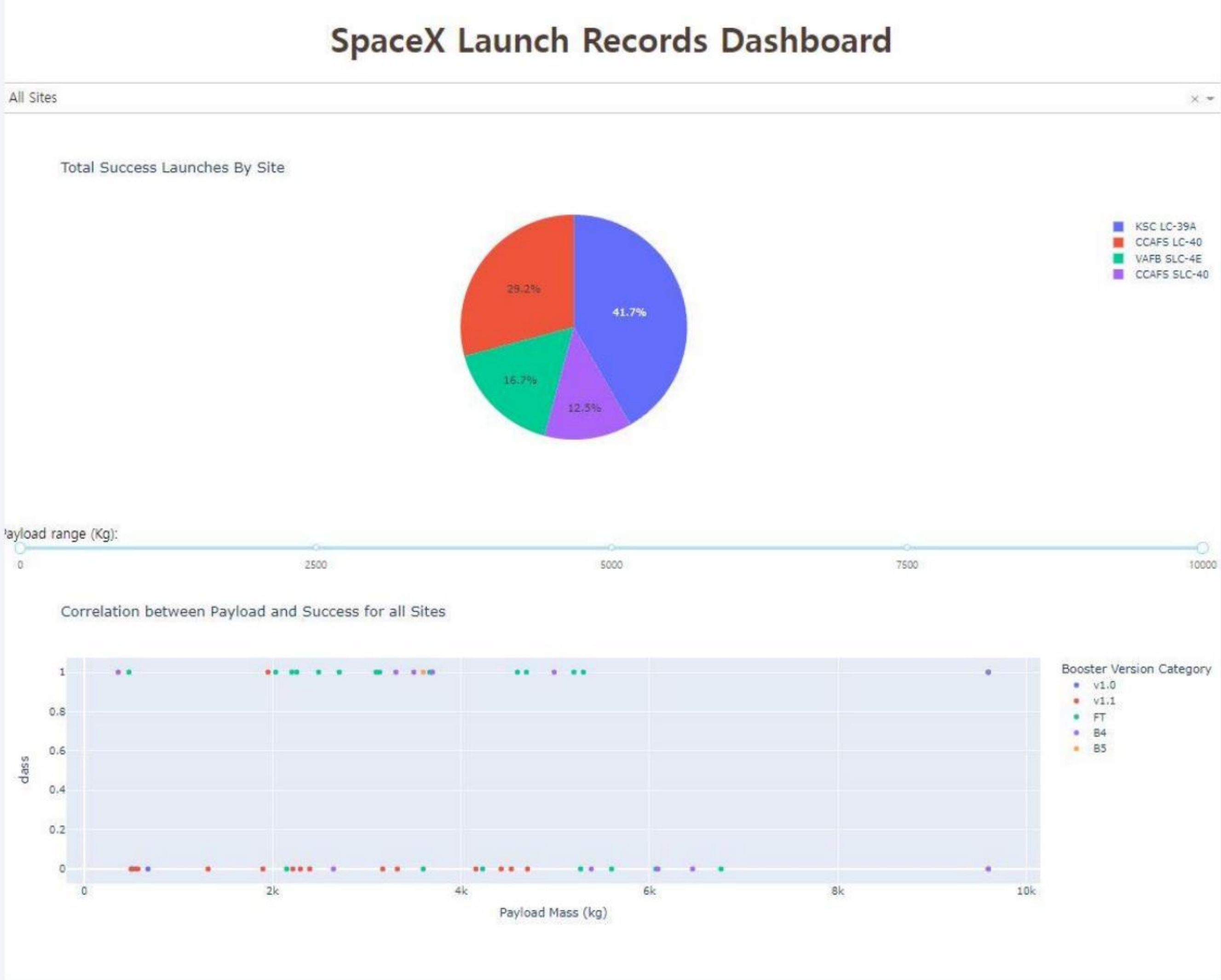
2. Finding Optimal Hyperparameters for SVM, Classification Trees, and Logistic Regression:

- Determining the best hyperparameters for Support Vector Machines (SVM), Classification Trees, and Logistic Regression. Evaluating the performance of each method using test data to identify the most effective approach.

- Github - Predictive Analysis



Results



SpaceX Launch Records Dashboard

All Sites

A sneak peek of the dashboard created using Plotly Dash..

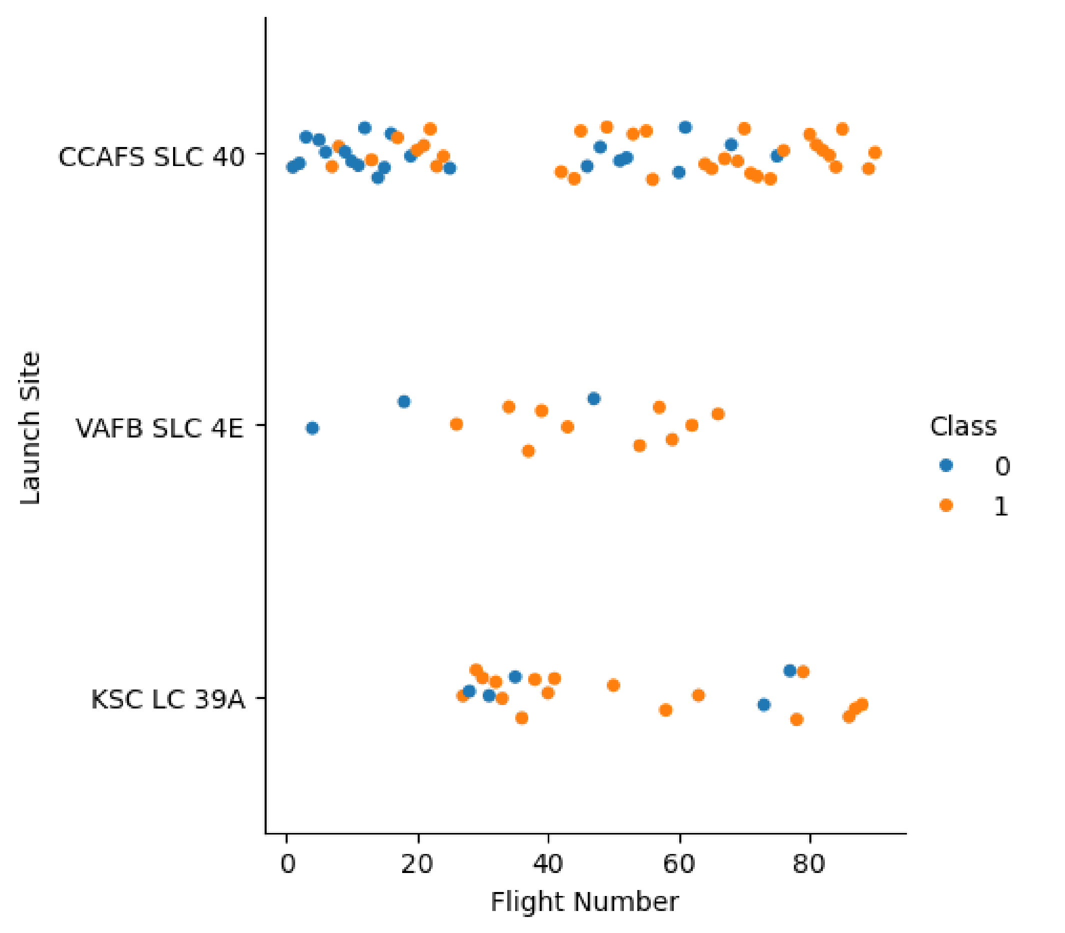
Upcoming slides will showcase outcomes from Exploratory Data Analysis (EDA) using visualization, EDA with SQL, an interactive map created with Folium, and an interactive dashboard.

When comparing the accuracy of the four methods, all demonstrate an accuracy of approximately 83% for the test data.

Section 2

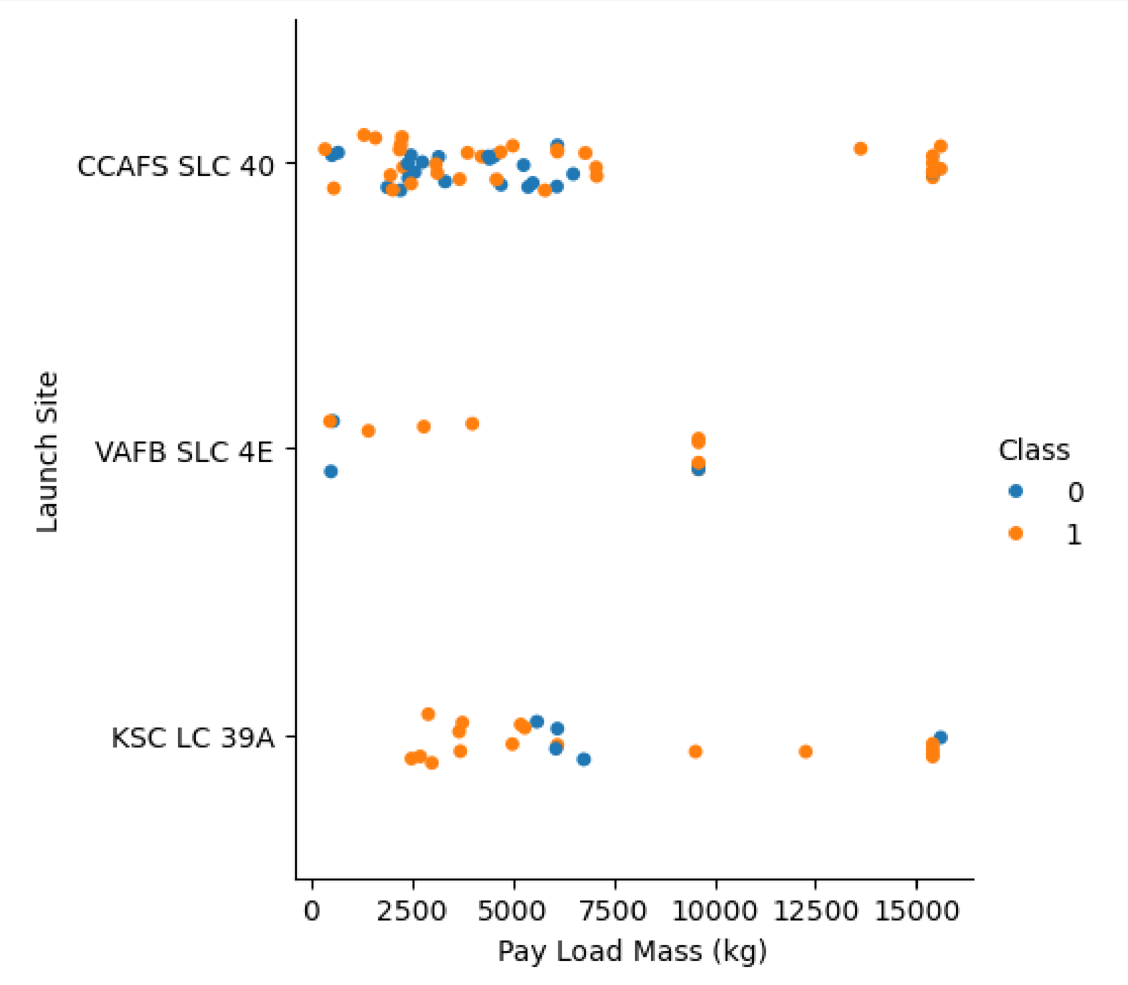
Insights drawn from EDA

Flight Number vs. Launch Site



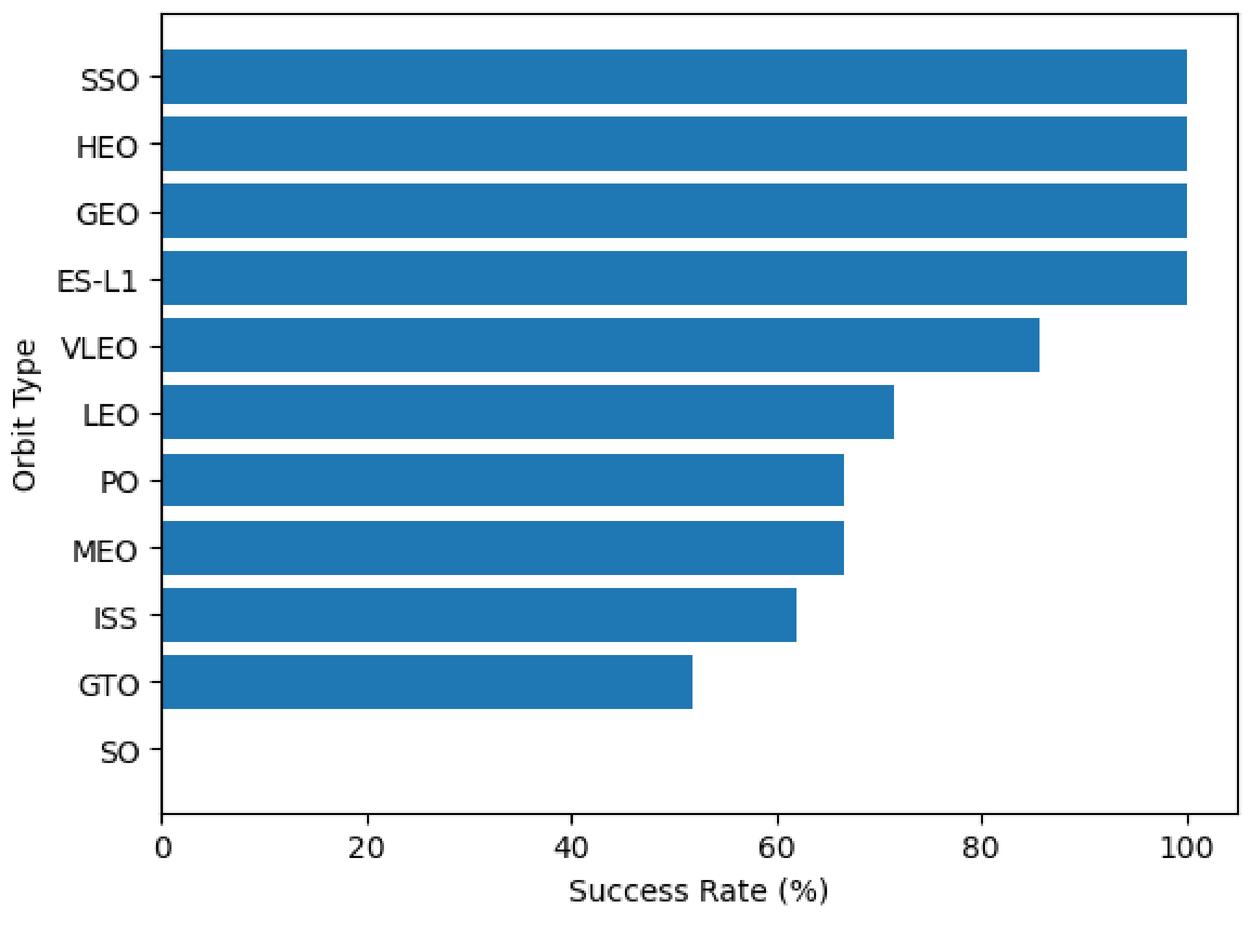
- Class 0 signifies unsuccessful launches, while Class 1 denotes successful launches.
- The figure indicates a rising success rate with an increase in the number of flights.
- A notable breakthrough is observed, particularly after the 20th flight, with a considerable increase in the success rate.

Payload vs. Launch Site



- Class 0 indicates unsuccessful launches, and Class 1 represents successful launches.
- Initially, it appears that a larger payload mass is associated with a higher success rate. However, decision-making based on this figure is challenging as no clear pattern between successful launch and payload mass is evident.

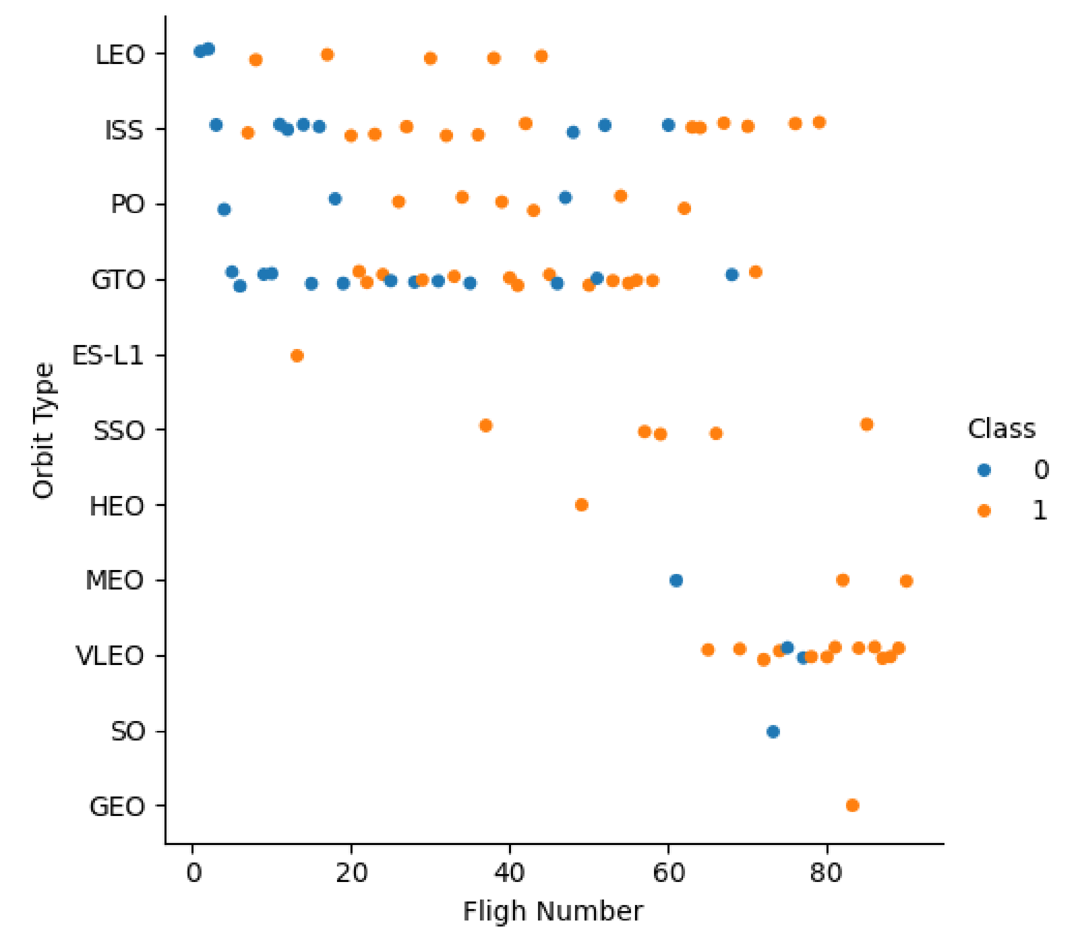
Success Rate vs. Orbit Type



- SSO, HEO, GEO, and ES-L1 orbit types boast the highest success rates, each at 100%.

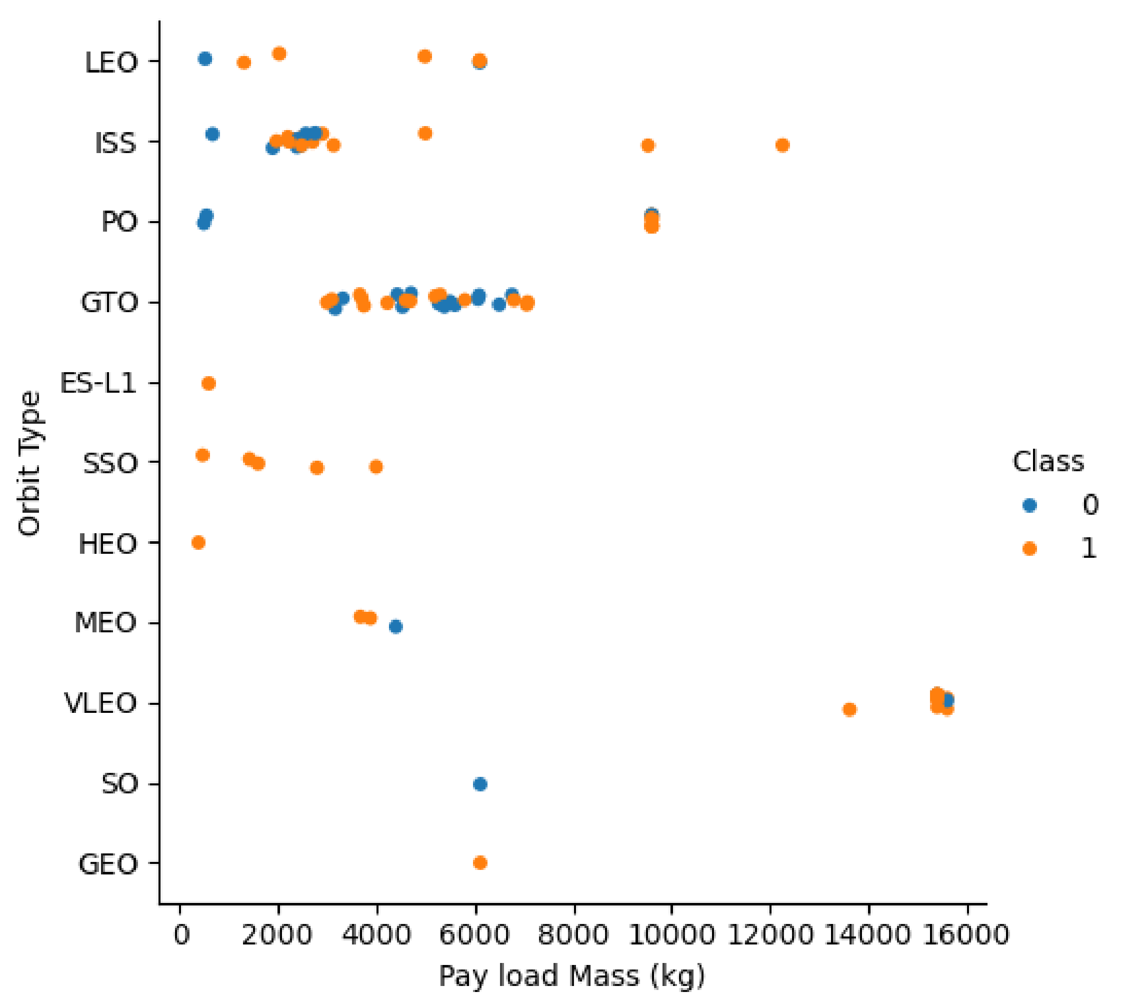
- In contrast, the GTO orbit type exhibits a 50% success rate, the lowest among all. The only exception is the SO type, which faced a single failure.

Flight Number vs. Orbit Type



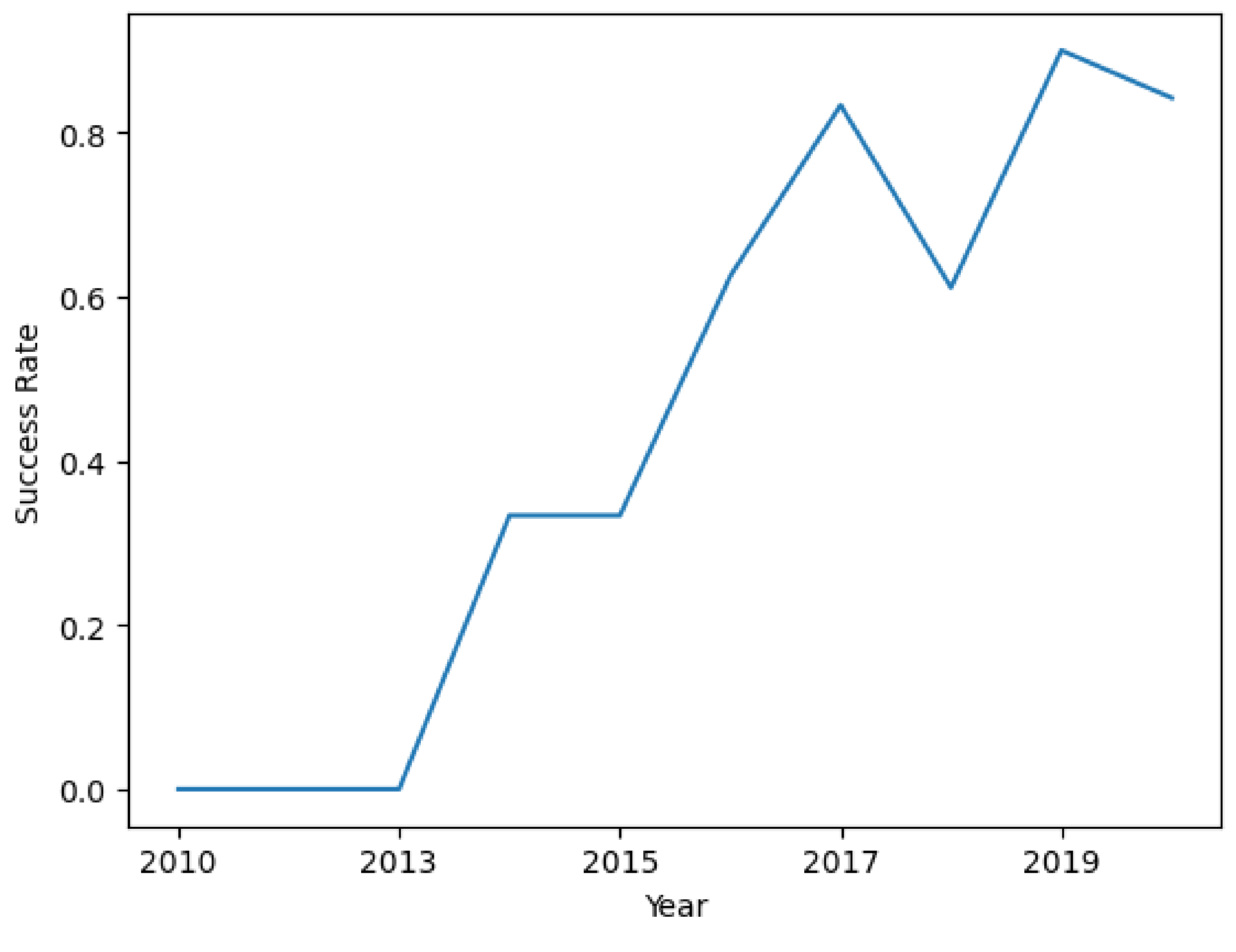
- Class 0 denotes unsuccessful launches, and Class 1 signifies successful launches.
- In most cases, the launch outcome appears to be linked to the flight number.
- Contrastingly, for GTO orbit, there seems to be no clear relationship between flight numbers and success rate.
- SpaceX starts with LEO, showing a moderate success rate. Notably, VLEO, with a high success rate, appears to be the preferred choice in recent launches.

Payload vs. Orbit Type



- Class 0 indicates unsuccessful launches, and Class 1 denotes successful launches.
- For heavy payloads, positive landing rates are more prominent for LEO and ISS. However, distinguishing between positive and negative landings is challenging for GTO due to their close grouping.

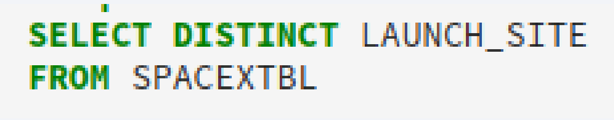
Launch Success Yearly Trend



- From 2013 to 2017, the success rate consistently rose
- In 2018, there was a slight decrease in the success rate
- Currently, there is a success rate of approximately 80%.

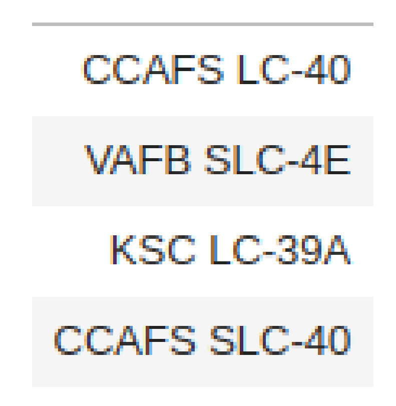
All Launch Site Names

- **Query**



```
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL
```

- **Result**



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

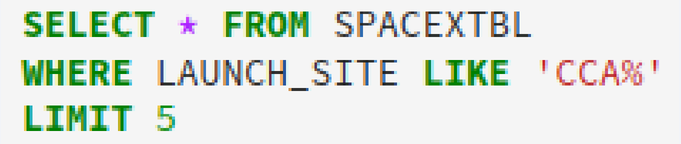
- **SQL DISTINCT Usage:**

When using the SQL DISTINCT clause, only unique values are shown in the Launch_Site column from the SpaceX table.

- **Unique Launch Sites:**

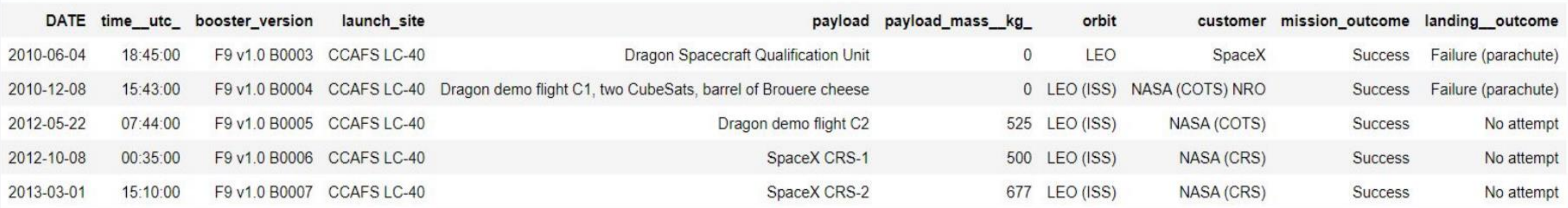
There are four unique launch sites: CCAFS LC-40, CCAFS SLC-40, KSC LC-39A, and VAFB SLC-4E.

Launch Site Names Begin with 'CCA'



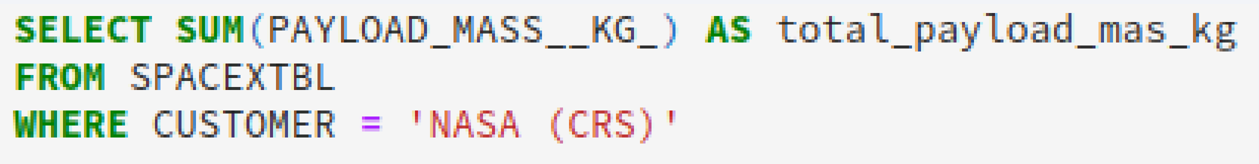
```
SELECT * FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE 'CCA%'
LIMIT 5
```

- **Limited Records Display:**
  - With the use of the LIMIT 5 clause in the query, only five records from the SpaceX table were displayed.
- **Launch Site Name Filter:**
  - By combining the LIKE operator with the percent sign (%), Launch_Site names starting with CAA can be retrieved.



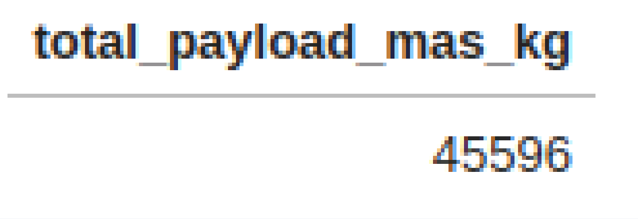
DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass



```
SELECT SUM(PAYLOAD_MASS__KG_) AS total_payload_mas_kg
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)'
```

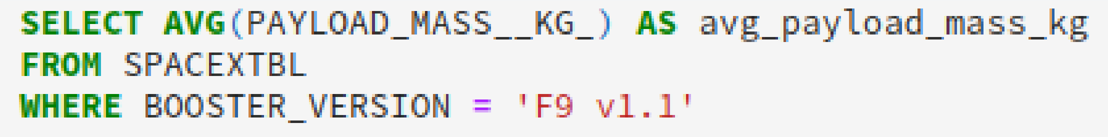
SUM() Function Usage:
The SUM() function is employed to calculate the total of the PAYLOAD_MASS__KG_ column



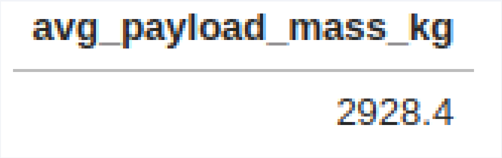
total_payload_mas_kg
45596

Filtering Dataset:
In the WHERE clause, the dataset is filtered to perform calculations exclusively when the Customer is NASA (CRS)

Average Payload Mass by F9 v1.1



```sql
SELECT AVG(PAYLOAD_MASS__KG_) AS avg_payload_mass_kg
FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1'
```

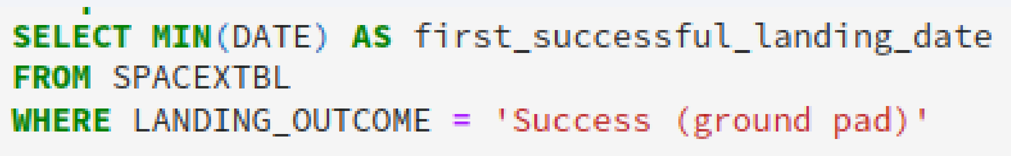


avg_payload_mass_kg
2928.4

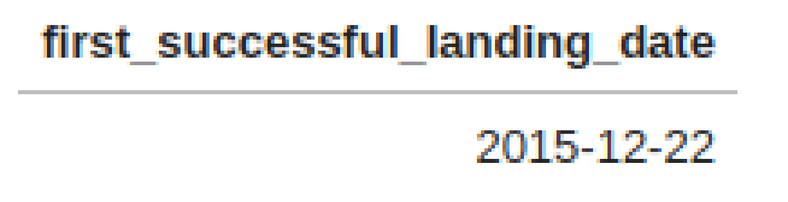
AVG() Function Usage:
The AVG() function is applied to compute the average value of the PAYLOAD_MASS__KG_ column.

Filtering Dataset:
In the WHERE clause, the dataset is filtered to perform calculations exclusively when the Booster_version is F9 v1.1.

First Successful Ground Landing Date



```
SELECT MIN(DATE) AS first_successful_landing_date
FROM SPACEXTBL
WHERE LANDING_OUTCOME = 'Success (ground pad)'
```

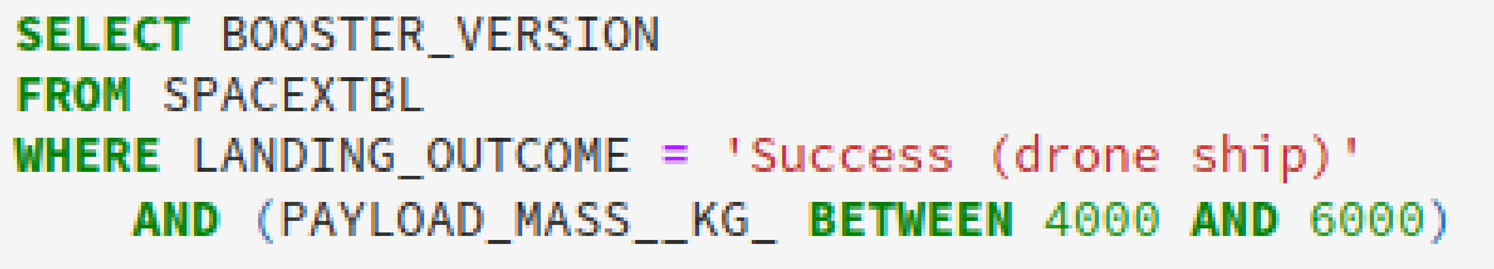


first_successful_landing_date
2015-12-22

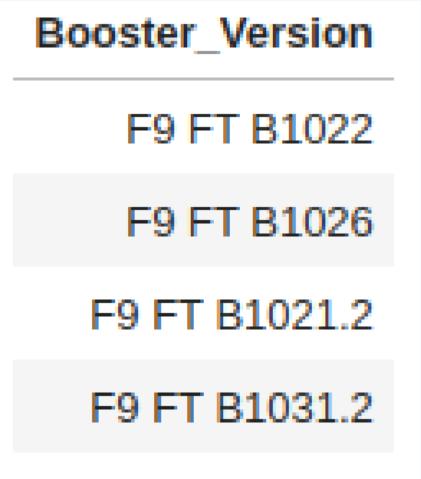
MIN() Function Usage:
The MIN() function is utilized to determine the earliest date in the DATE column.

Filtering Dataset:
In the WHERE clause, the dataset is filtered to conduct the search solely when the Landing__outcome is a Success (ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000



```
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING_OUTCOME = 'Success (drone ship)'
    AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000)
```

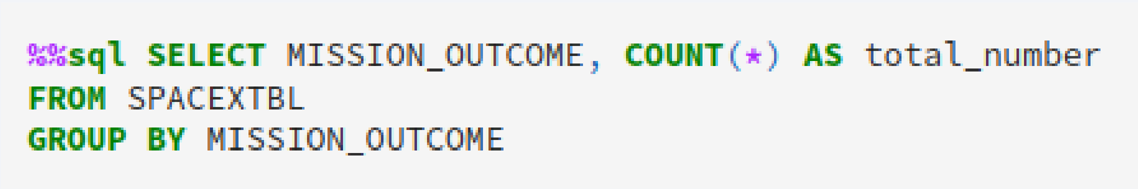


Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

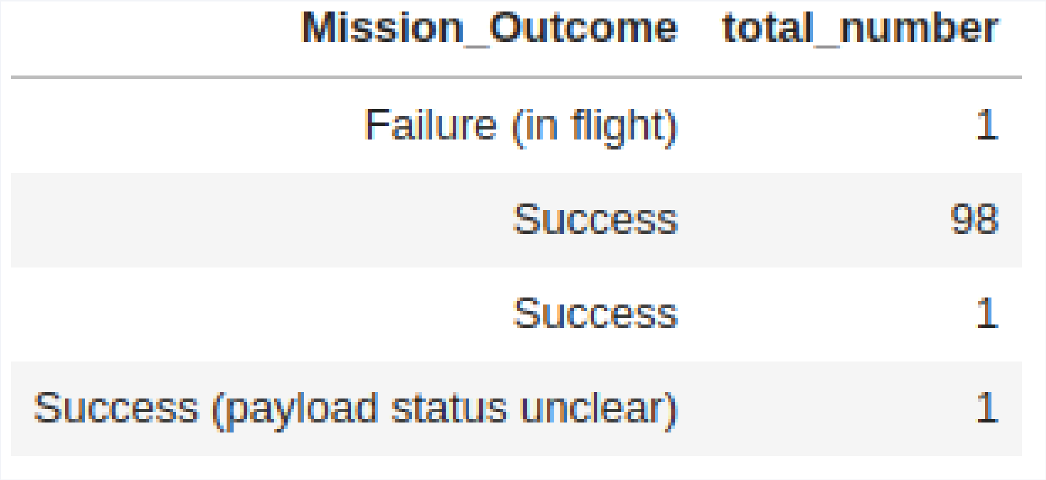
Filtering for Success on Drone Ship:
In the WHERE clause, the dataset is filtered to perform a search when the Landing__outcome is Success (drone ship).

Additional Condition with AND Operator:
Using the AND operator, a record is displayed only if the additional condition PAYLOAD_MASS__KG_ is between 4000 and 6000.

Total Number of Successful and Failure Mission Outcomes



```
%%sql SELECT MISSION_OUTCOME, COUNT(*) AS total_number
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME
```



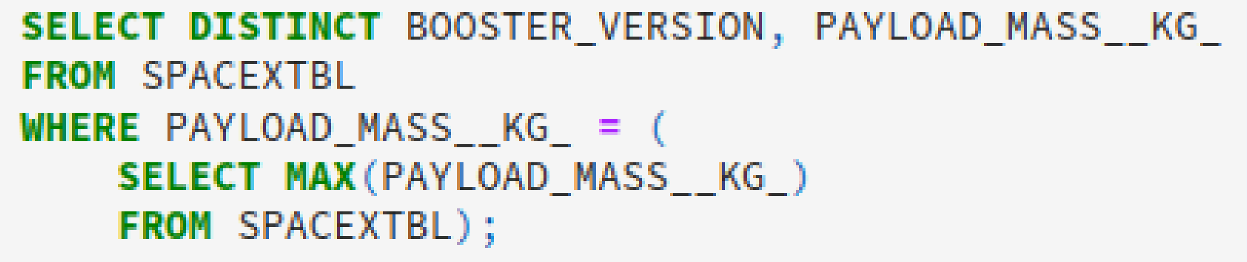
Mission_Outcome	total_number
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

COUNT() Function Usage:
The COUNT() function is applied to calculate the total number of columns.

GROUP BY Statement:
Utilizing the GROUP BY statement, rows with identical values are grouped to find the total number in each Mission_outcome.

Mission Success Observation:
Based on the results, SpaceX has achieved success in nearly 99% of its missions.

Boosters Carried Maximum Payload

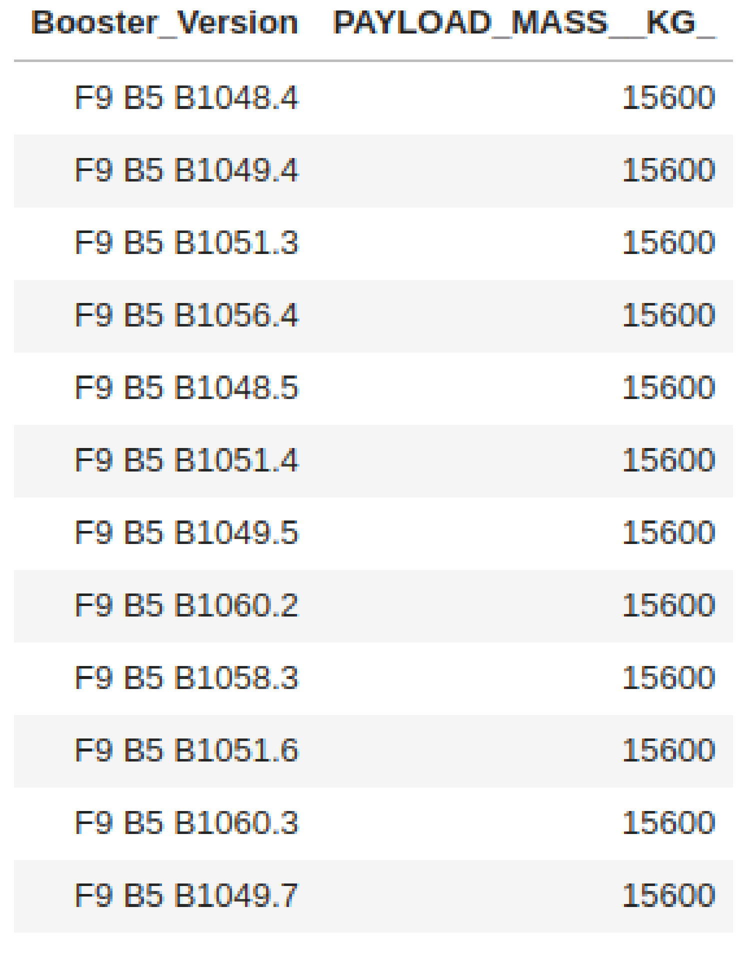


```
SELECT DISTINCT BOOSTER_VERSION, PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL);
```

Subquery with MAX() Function:
Using a subquery, firstly, identify the maximum payload value using the MAX() function.

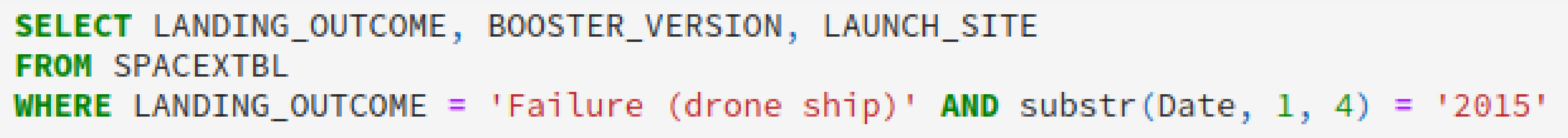
Filtering Dataset for Maximum Payload:
Secondly, filter the dataset to perform a search if PAYLOAD_MASS__KG_ is the maximum payload value.

Observation from Results:
Based on the results, boosters of version F9 B5 B10xx.x could carry the maximum payload.

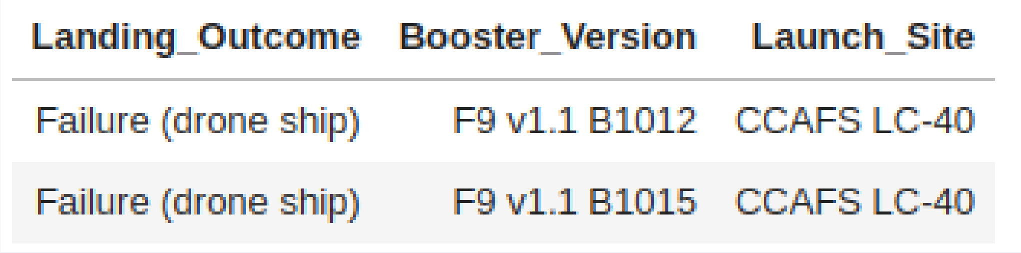


Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records



```
SELECT LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE LANDING_OUTCOME = 'Failure (drone ship)' AND substr(Date, 1, 4) = '2015'
```



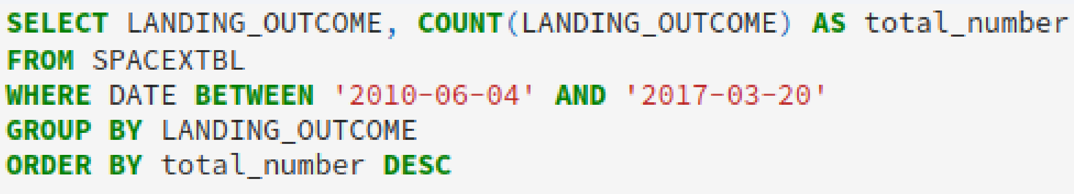
Landing_Outcome	Booster_Version	Launch_Site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Filtering for Drone Ship Landing Failure:
In the WHERE clause, the dataset is filtered to perform a search when the Landing__outcome is Failure (drone ship).

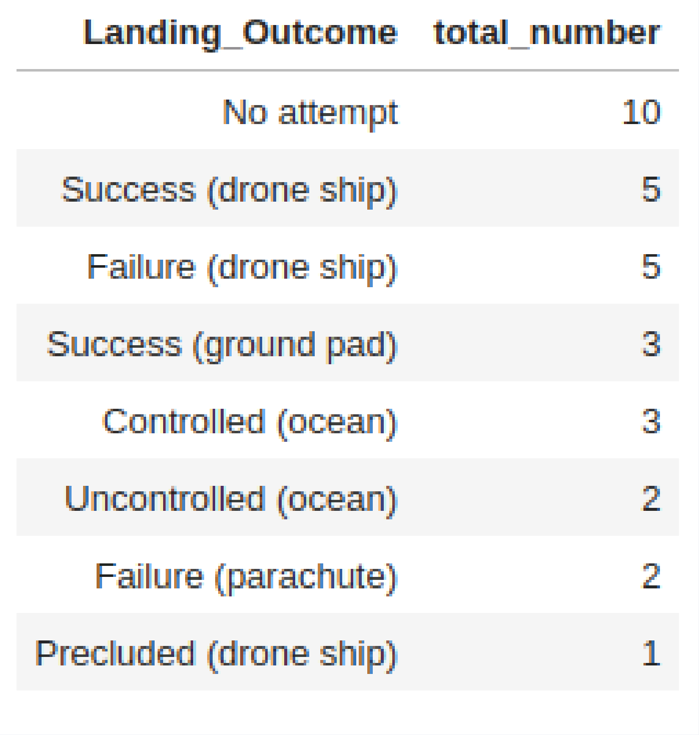
Additional Condition with AND Operator:
Using the AND operator, a record is displayed only if the additional condition YEAR is 2015.

Observation from Results:
In 2015, there were two instances of landing failures on drone ships.

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



```
SELECT LANDING_OUTCOME, COUNT(LANDING_OUTCOME) AS total_number
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING_OUTCOME
ORDER BY total_number DESC
```



Landing_Outcome	total_number
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

Filtering by Date Range:
In the WHERE clause, the dataset is filtered to perform a search if the date is between 2010-06-04 and 2017-03-20.

Sorting Records by Total Landings:
Using the ORDER BY keyword, records are sorted by the total number of landings.

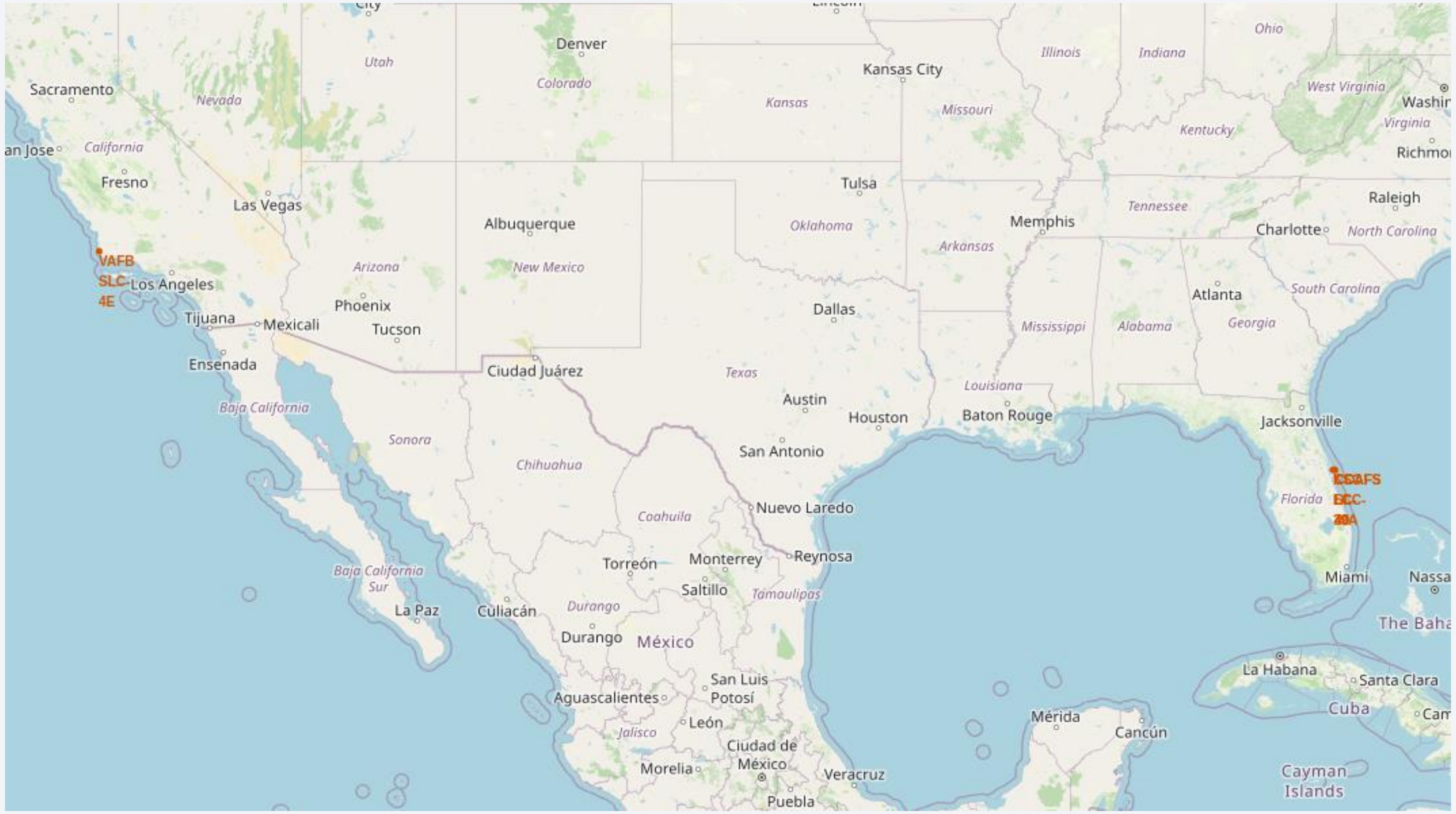
Descending Order:
Using the DESC keyword, the records are sorted in descending order.

Observation from Results:
Based on the results, the number of successes and failures between 2010-06-04 and 2017-03-20 was similar.

Section 3

Launch Sites Proximities Analysis

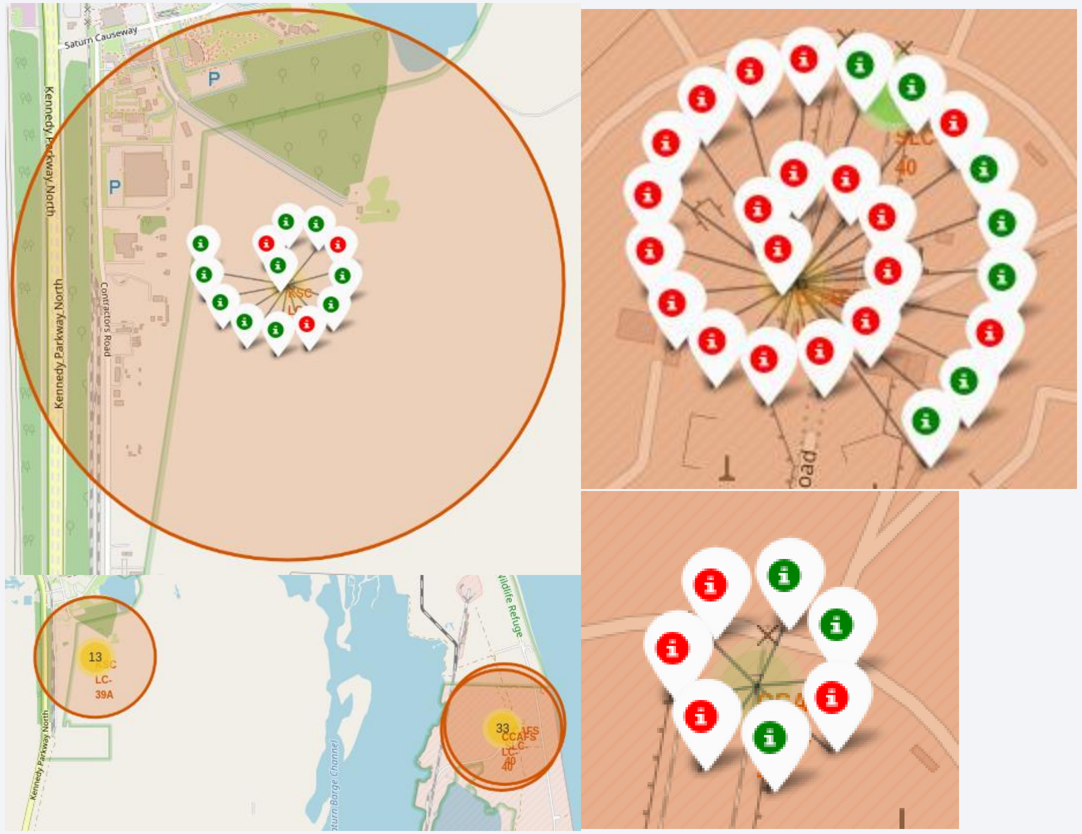
All Locations of Launch Sites



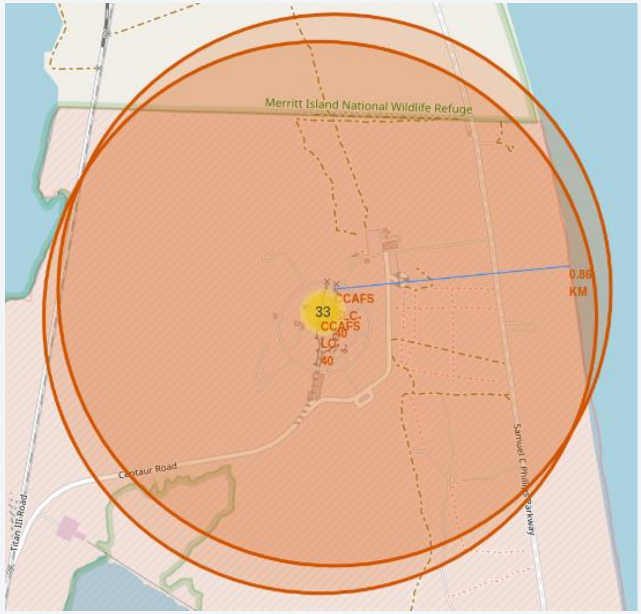
The map on the left displays all SpaceX launch sites, indicating that they are all located in the United States. Additionally, it is evident from the map that all launch sites are situated near the coast.

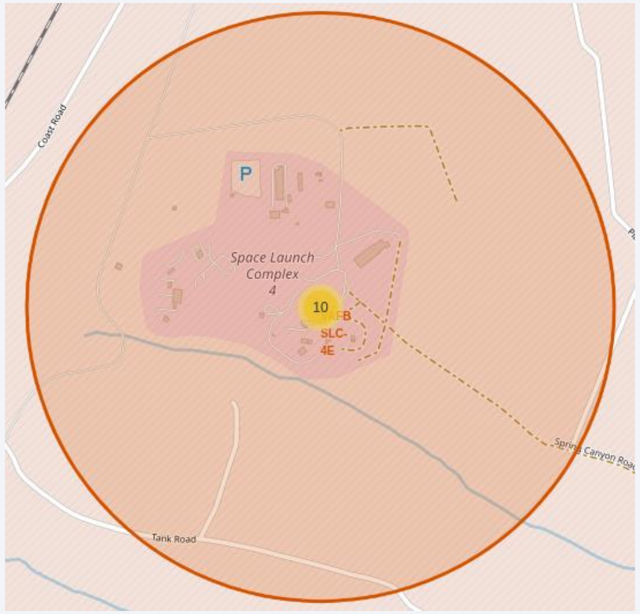
Launch Outcomes Color-Coded

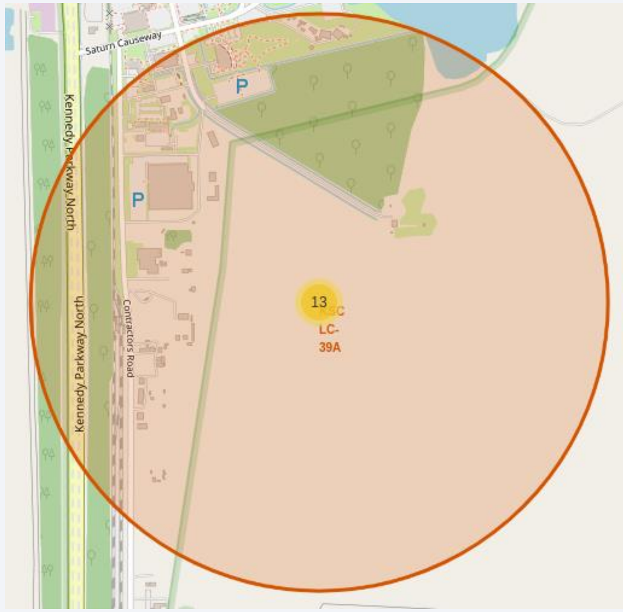
Clicking on marker clusters reveals successful landings (green) or failed landings (red).



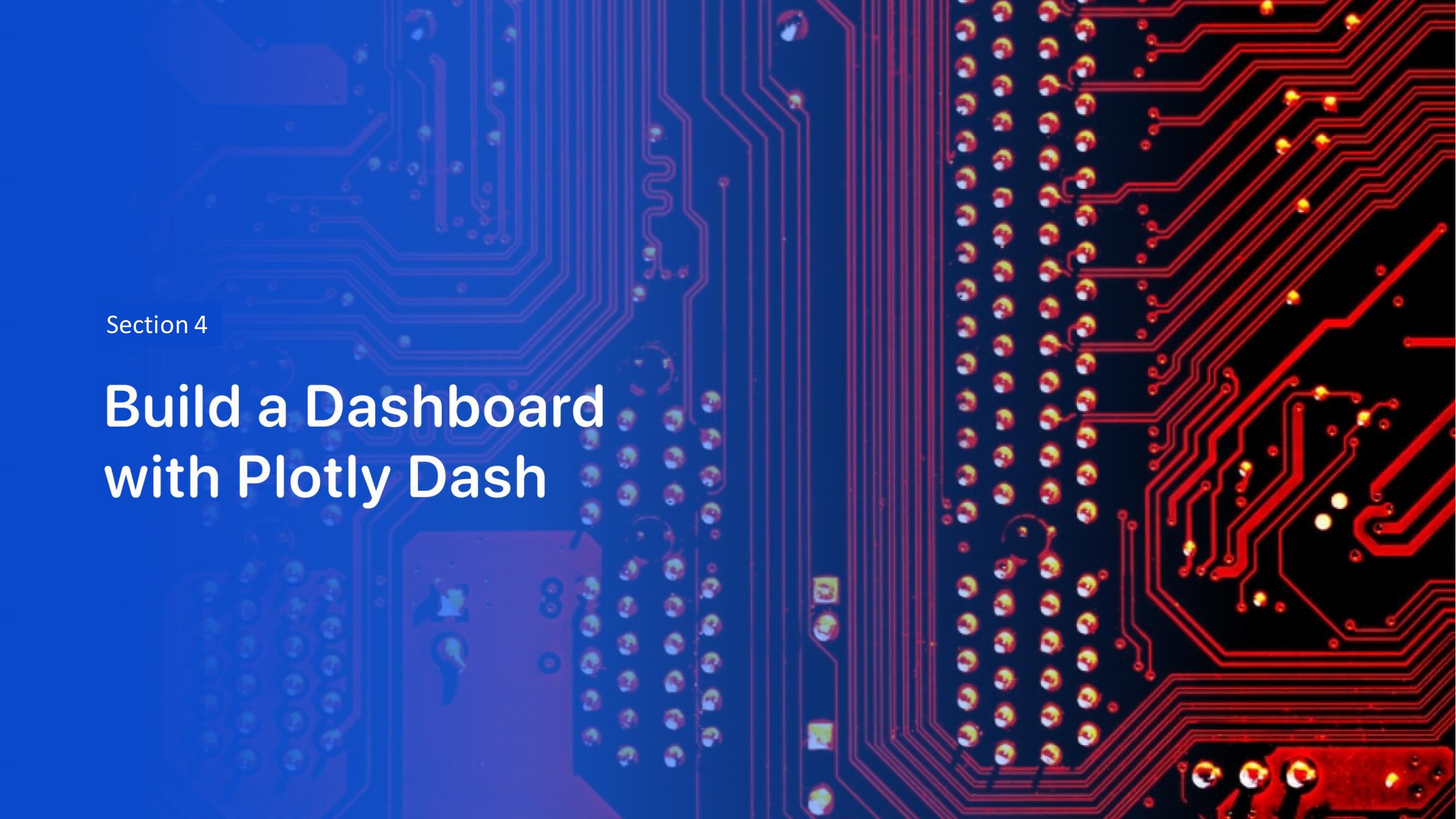
Launch Site Proximity


Merritt Island National Wildlife Refuge
0.86 KM
33
CCAFS LC-40
CCAFS SLC-40
Centaur Road
Titan III Road
Samuel C Phillips Parkway





The launch site is strategically located near railways and highways for easy transportation of equipment and personnel. Additionally, it is positioned close to the coastline and relatively distant from cities to mitigate potential threats in case of a launch failure.



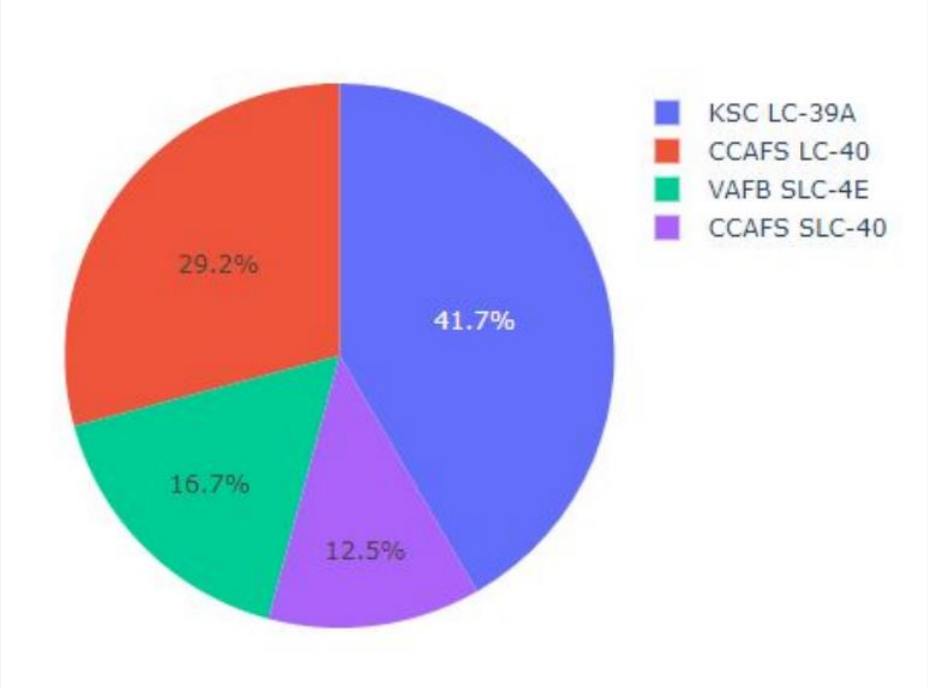
Section 4

Build a Dashboard with Plotly Dash

Total Successful Launches Across All Sites

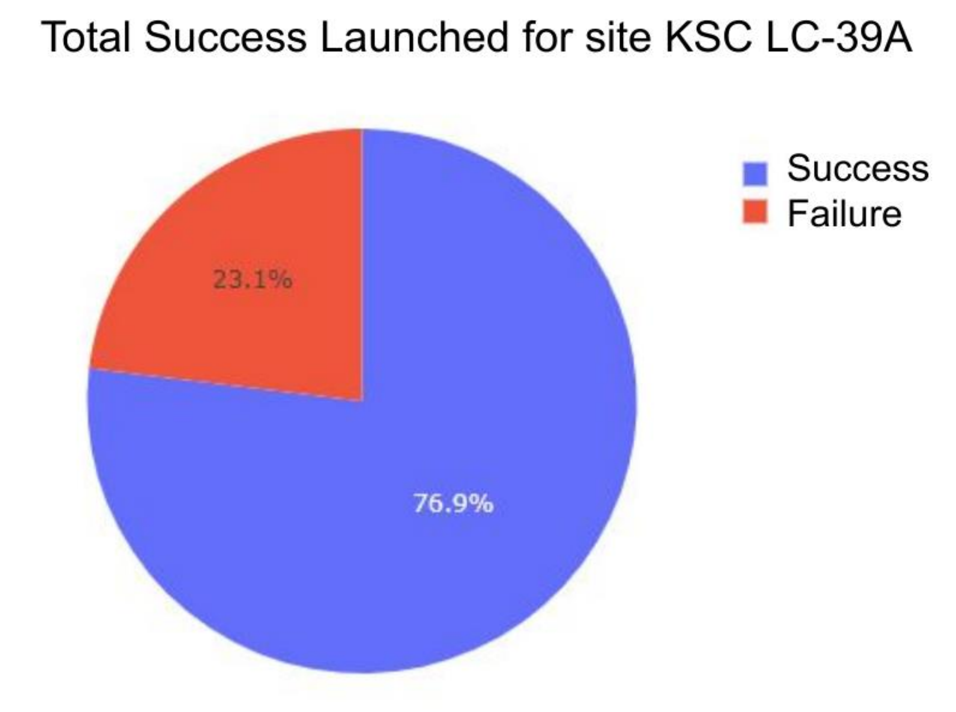
- **KSLC-39A** has the highest number of successful launches among all sites.

- On the other hand, **VAFB SLC-4E** has the fewest successful launches, potentially due to a smaller data sample or the unique challenges of being the only site located in California, making launches on the west coast potentially more challenging than on the east coast.

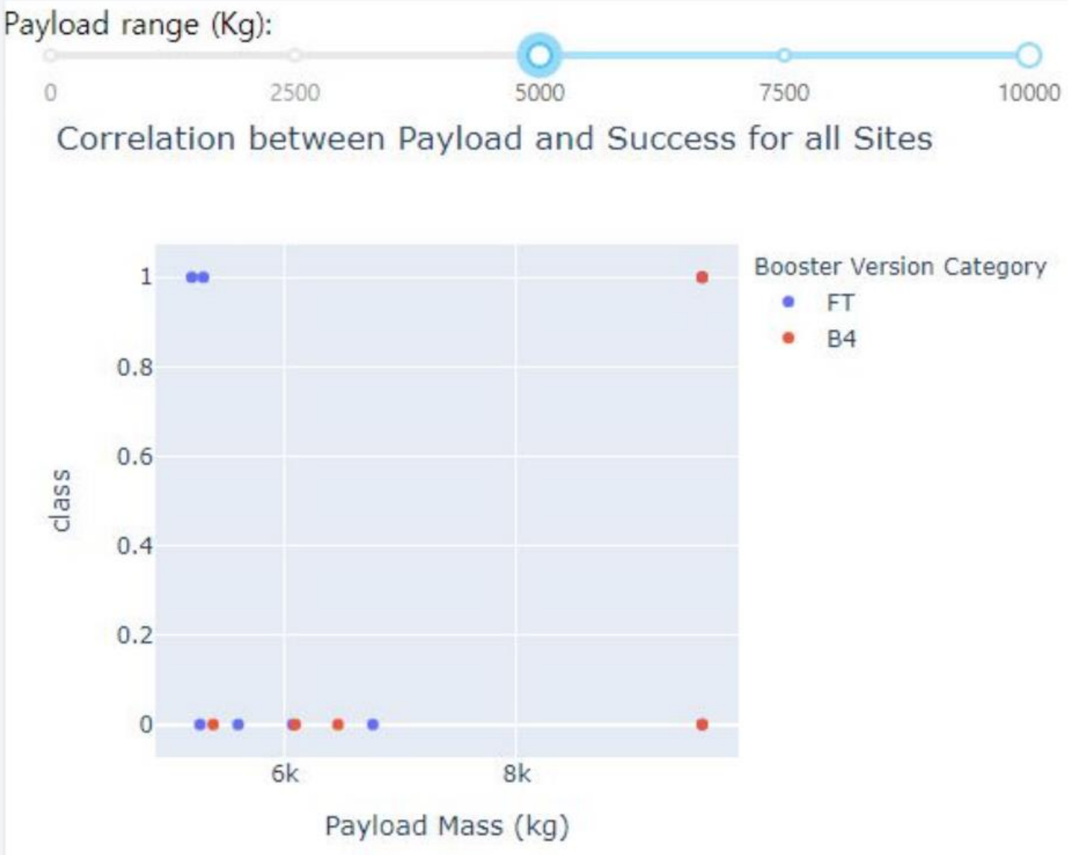


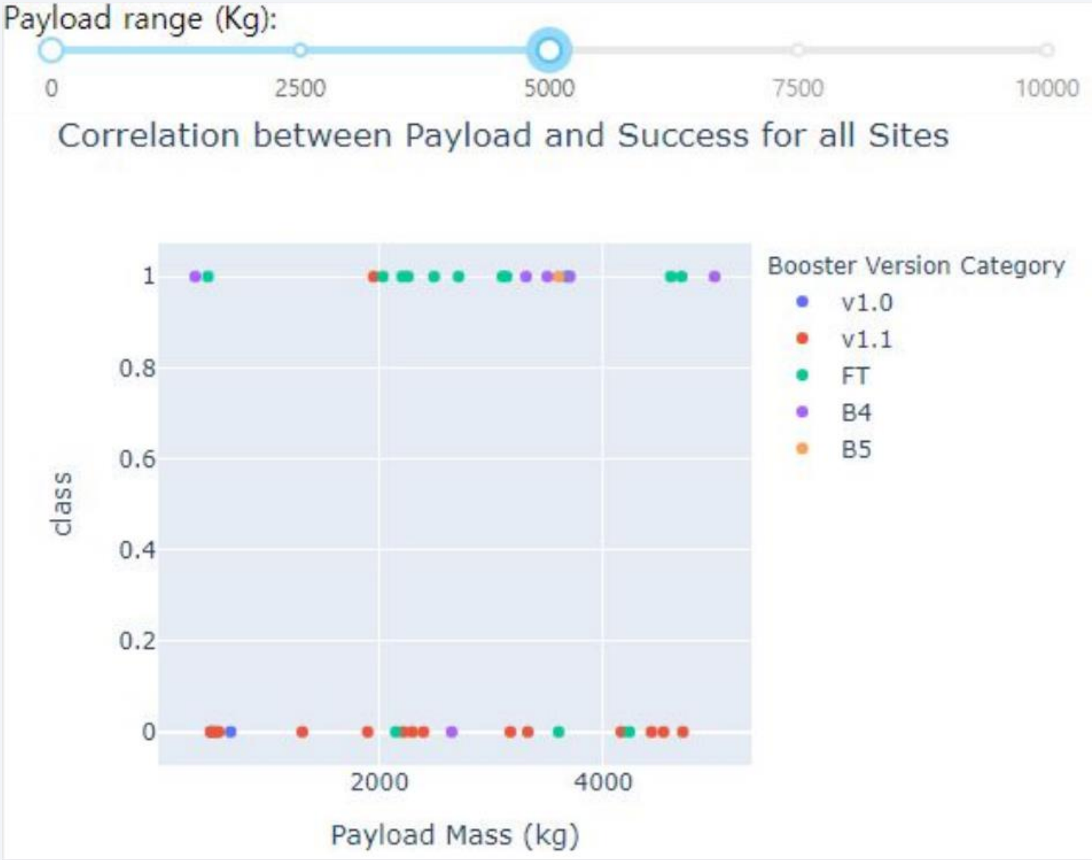
Launch Site with Highest Success Ratio

KSLC-39A boasts the highest success rate, achieving 10 landing successes (76.9%) and 3 landing failures (23.1%).



Payload vs. Launch Outcome Scatter Plot Across All Sites



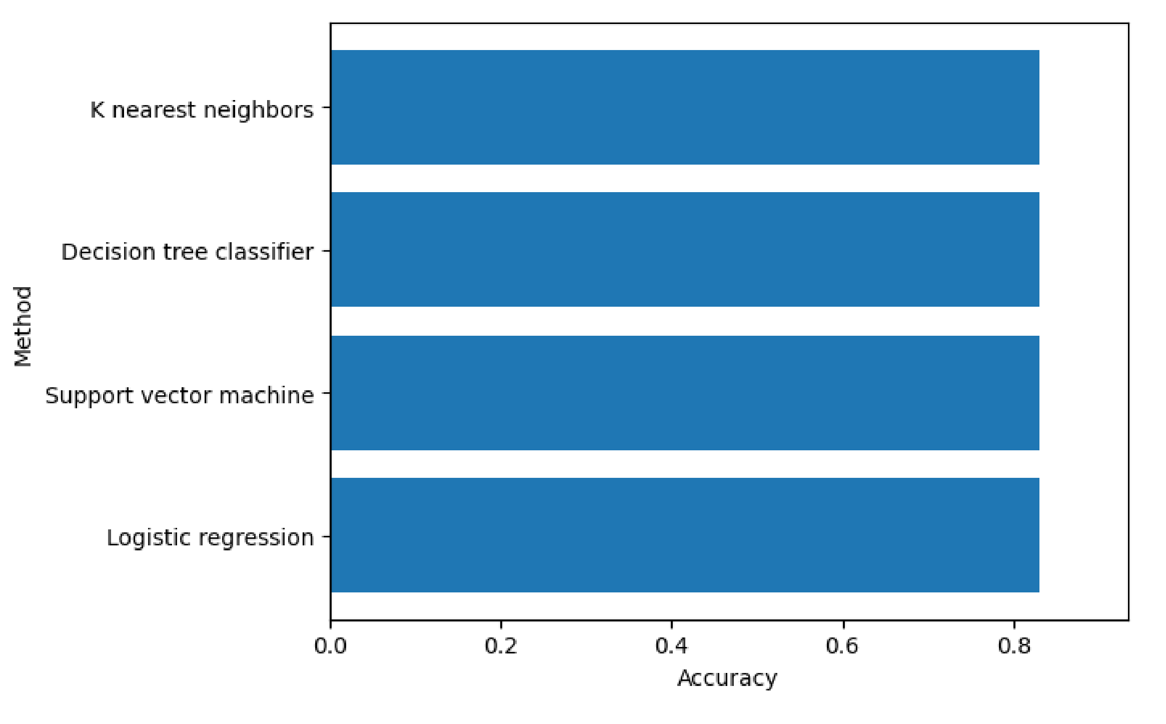


These figures reveal that the launch success rate (class 1) is higher for low-weighted payloads (0-5000 kg) compared to heavy-weighted payloads (5000-10000 kg).

Section 5

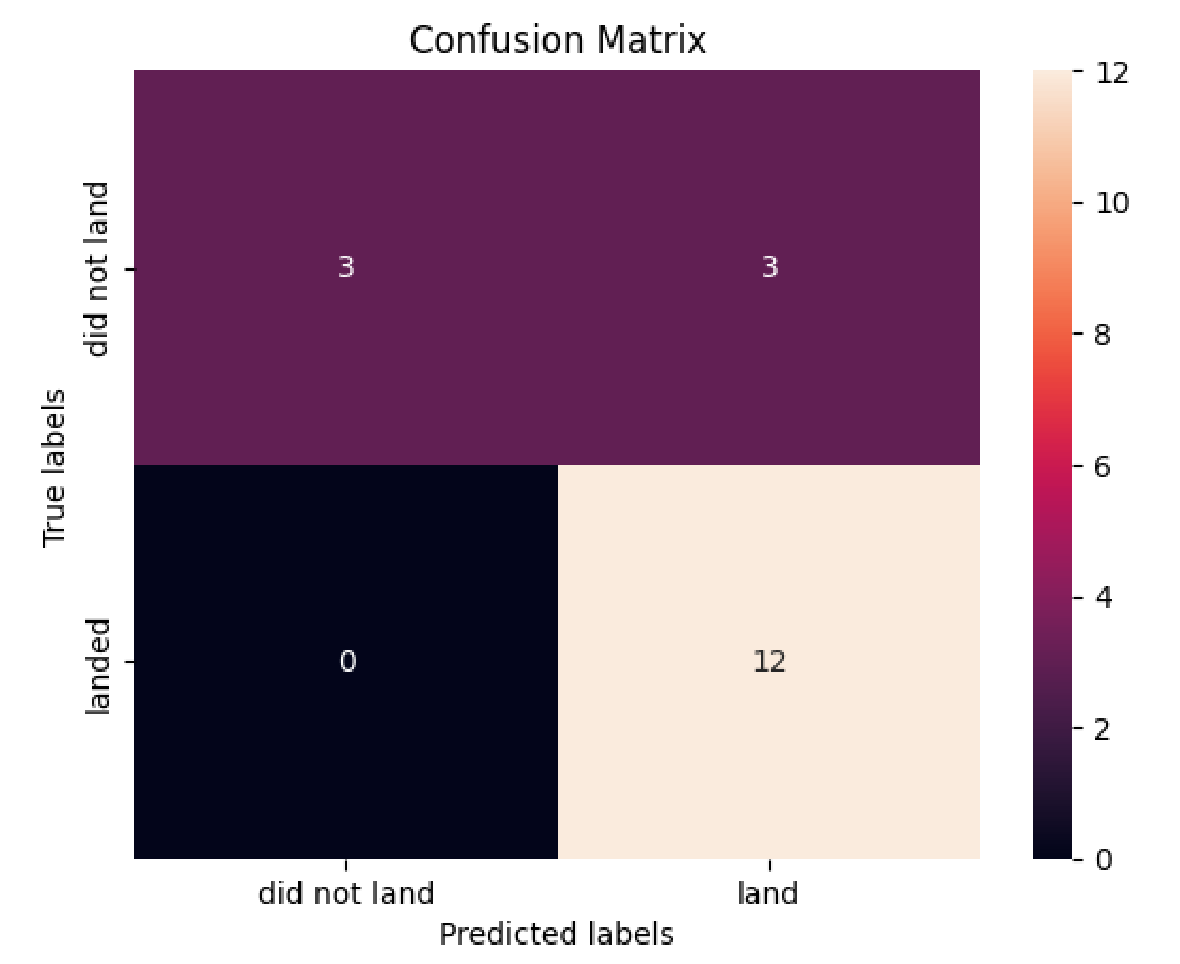
Predictive Analysis (Classification)

Classification Accuracy



- In the test set, all models demonstrated nearly identical **accuracy at 83.33%**.
- It's important to note that the test size was small, with only 18 samples.
- Therefore, gathering more data is necessary to determine the optimal model.

Confusion Matrix



- The confusion matrix is consistent across all models as their performance is the same for the test set.

- The models correctly predicted 12 successful landings and 3 failed landings. However, there were 3 false positive predictions, where the models indicated successful landings when the true label was failure.

- In summary, these models primarily predict **successful landings.**

Conclusions

As the quantity of flights increased, there was a corresponding rise in the success rate, surpassing 80% in recent times.

• Orbital types SSO, HEO, GEO, and ES-L1 exhibit the highest success rates, reaching 100%.

• The launch site is strategically located near railways, highways, and the coastline, while maintaining a considerable distance from urban areas.

• Among all sites, KSLC-39A stands out with the highest number of successful launches and the highest success rate.

• Notably, the success rate for launches with lower payload weights surpasses that of heavier payloads.

• Within this dataset, all models share an identical accuracy of 83.33%. However, the need for more data is evident to ascertain the optimal model, given the dataset's relatively small size.

Appendix

- The whole project on Github

Thank you!