IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Mukaila Akinbola
04 July 2024

Outline

Executive Summary

- Summary of methodologies
  - Data Collection
  - Data Wrangling
  - EDA with Data Visualization
  - EDA with SQL
  - Building Interactive Maps with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive Analysis (Classification)
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

- Project background and context
  - SpaceX is the most successful company among commercial rocket providers
  - The competitive price offered by SpaceX relies on the ability to reuse the first stage of the rocket
  - While SpaceX launches its Falcon 9 rocket at a cost of $62 Million, other providers cost upward of $165 Million.
  - The competitiveness of SpaceX hinges on the probability of successful landing of the first stage
  - If the probability of successful landing of the first stage can be determined, then the cost of the launch can be estimated.
  - Information on predicting the first stage landing, as well as launch cost estimation, can be used by other competitors of SpaceX in the commercial space business
- Problems to be addressed
  - How do we determine if SpaceX will be able to reuse the first stage?
  - How do we determine the price of each launch?
  - Do we have the necessary information to predict if SpaceX will reuse the first stage?

Section 1

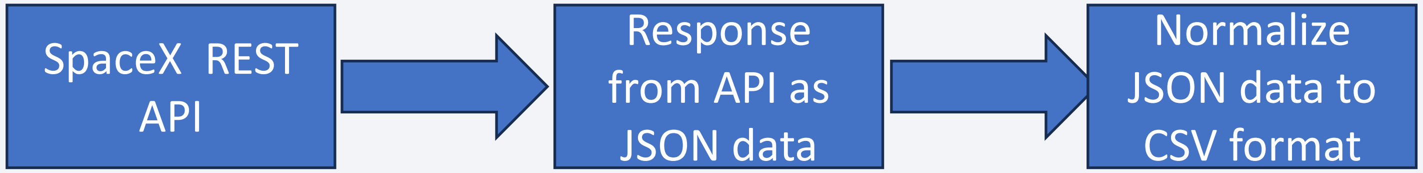
Methodology

Methodology

- Data collection methodology:
  - Gathering of launch data from SpaceX REST API
  - Web scraping of related Wiki pages containing valuable Falcon 9 launch records
- Perform data wrangling
  - Convert landing outcomes (categorical variables) into classes (0 or 1)
  - Drop irrelevant columns and replace missing values
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Building a machine learning pipeline, including preprocessing, training, testing, and evaluating the performance of the classification model

Data Collection

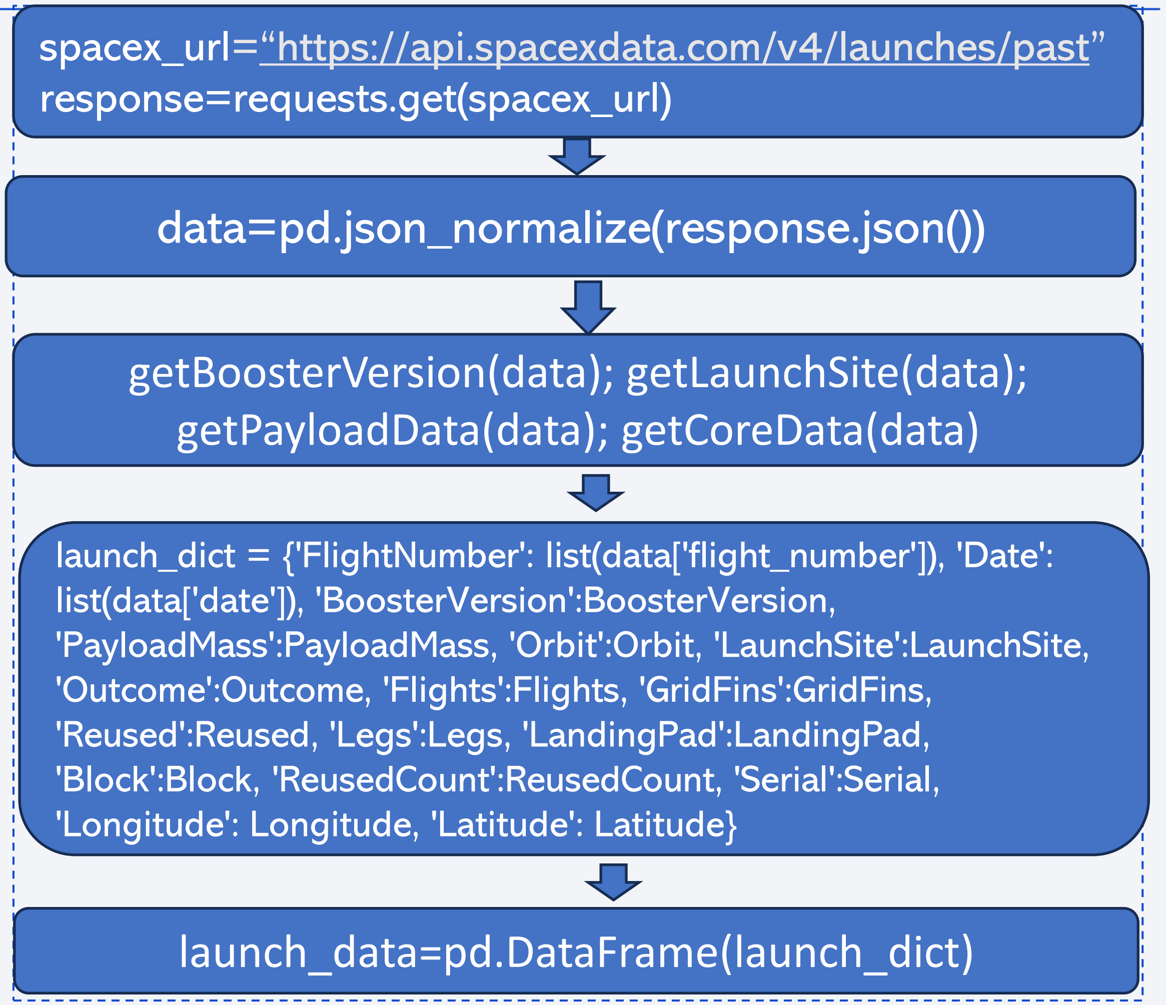
- The data collection process could be summarized as follows:
  - Launch data was gathered from SpaceX REST API.
  - The API provides information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
  - The API's URL was used to perform a get request to obtain launch data from the API.
  - The response comes in the form of JSON objects; we used the json_normalize function to convert this JSON to a dataframe.
  - In addition to the REST API, the BeautifulSoup package was used to web-scrape relevant wiki pages containing Falcon 9 launch data.
- A typical flowchart representing the data collection process is shown below:


SpaceX REST API
Response from API as JSON data
Normalize JSON data to CSV format

Data Collection – SpaceX API

- Request launch data from SpaceX API ===➔
- Convert the JSON response to dataframe =➔
- Apply function method to launch data===➔
- Combine the columns into a dictionary ➔
- Create pd dataframe from the dictionary➔

GitHub URL



```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response=requests.get(spacex_url)
```

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

```
getBoosterVersion(data); getLaunchSite(data);
getPayloadData(data); getCoreData(data)
```

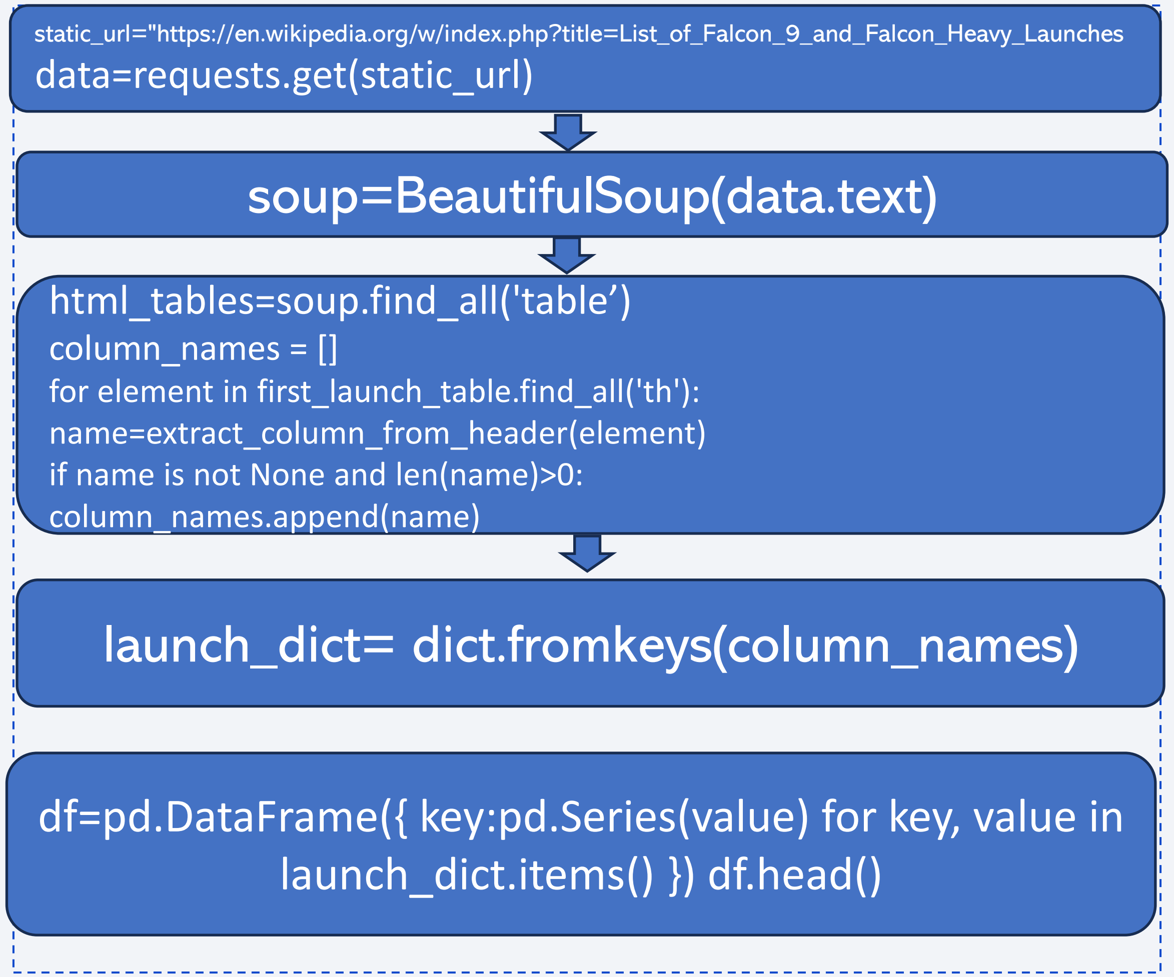
```
launch_dict = {'FlightNumber': list(data['flight_number']), 'Date':
list(data['date']), 'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass, 'Orbit':Orbit, 'LaunchSite':LaunchSite,
'Outcome':Outcome, 'Flights':Flights, 'GridFins':GridFins,
'Reused':Reused, 'Legs':Legs, 'LandingPad':LandingPad,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

```
launch_data=pd.DataFrame(launch_dict)
```

Data Collection - Scraping

- Request Falcon 9 HTML Wiki page ===➔
- Create a BeautifulSoup object from the HTML response text content============➔
- Find all tables on the Wiki page and extract all column names from the table header==➔
- Create an empty dictionary from the extracted column names================ ➔
- Create pd dataframe from the dictionary➔

[GitHub URL]



```
static_url="https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_Launches
data=requests.get(static_url)
```

```
soup=BeautifulSoup(data.text)
```

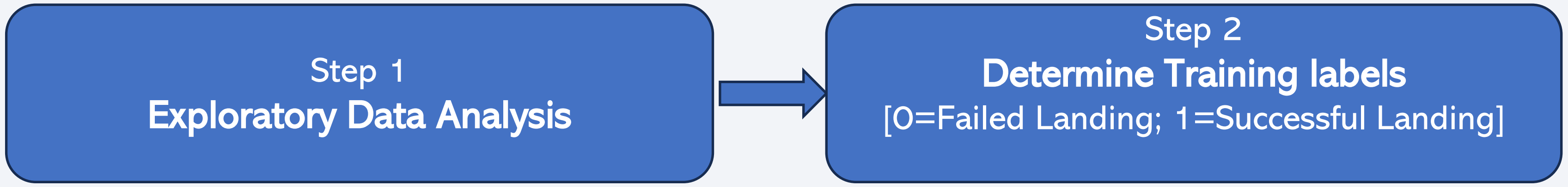
```
html_tables=soup.find_all('table')
column_names = []
for element in first_launch_table.find_all('th'):
name=extract_column_from_header(element)
if name is not None and len(name)>0:
column_names.append(name)
```

```
launch_dict= dict.fromkeys(column_names)
```

```
df=pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() }) df.head()
```

Data Wrangling

- Perform Exploratory Data Analysis on SpaceX dataset
  - Load SpaceX dataset → **df=pd.read_csv("url")**
  - Identify and calculate the percentage of missing values → **df.isnull().sum()/len(df)*100**
  - Identify which columns are numerical or categorical → **df.dtypes**
- Use the method value_counts() to determine:
  - The number of launches from each launch site → **df['LaunchSite'].value_counts()**
  - The number and occurrence of each orbit → **df['Orbit'].value_counts()**
  - The mission outcome of the orbits → **landing_outcomes=df['Outcome'].value_counts()**
- Create a landing outcome label from Outcome column, with two classes ('0' or '1'):
  - **df['Class']=landing_class**


Step 1
Exploratory Data Analysis
Step 2
Determine Training labels
[0=Failed Landing; 1=Successful Landing]

GitHub URL

EDA with Data Visualization

- Seaborn's catplot, scatterplot , barplot, and lineplot were used to explore the relationship between principal variables in the launch data.
- The catplot (with an overlay of launch outcome) explores the relationship between:
  - **FlightNumber vs. PayloadMass ------** to view how the two variables affect launch outcome.
- The scatterplot (with an overlay of launch outcome) explores the relationships:
  - **FlightNumber vs. LaunchSite** ----- to view the concentration of launches at each site.
  - **PayloadMass vs. LaunchSite** ----- to view the size of rockets launched from each site.
  - **PayloadMass vs. Orbit** ------ to see which orbit has more success with heavy payloads
  - **FlightNumber vs. Orbit** ------ to see which orbit's success is related to the number of flights.
- The barplot explores the relationship between:
  - **Orbit vs. Success Rate** ------- to find which orbit has the highest and lowest success rate.
- The lineplot explores the relationship between:
  - **Success Rate vs. Year** ------- to view the trend of average success from year to year.

GitHub URL

EDA with SQL

- SQL queries were performed to:
  - Display the names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA’
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first successful landing outcome in the ground pad was achieved
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failed mission outcomes
  - List the names of the booster versions which have carried the maximum payload mass
  - List the records which will display the month names, failure landing outcomes in drone ship, booster versions, and launch site for the months in year 2015.
  - Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

GitHub URL

Build an Interactive Map with Folium

- To find some geographical pattern to the location of each launch site:
  - Each site location was added on a map using the site's latitude and longitude coordinates
  - For better visualization of the sites, a folium.Map object was created with an initial center located at NASA Johnson Space Center in Houston, Texas.
  - Folium.Marker was added to the map to mark each launch site.
  - Folium.Circle was added to the map to explore the map by zooming in and out of the marked areas.
- Enhance the map by adding launch outcomes for each site:
  - For successful launch (class=1), we used a green marker
  - For failed launch (class=0) we used a red marker
  - A MarkerCluster object was created to simplify multiple markers with the same coordinate.
- Explore and analyze the proximities of each launch site:
  - A MousePosition was added to get the coordinate of the mouse over a point on the map
  - A marker was created with the distance to the closest city, highway, railway, and coastline
  - A PolyLine was drawn between the marker and the launch sites to calculate the distances
  - Some noteworthy observations from the Interactive Map:
    - Launch sites are located in close proximity to highways, railways, and coastlines
    - Launch sites keep a certain distance away from the cities

GitHub URL

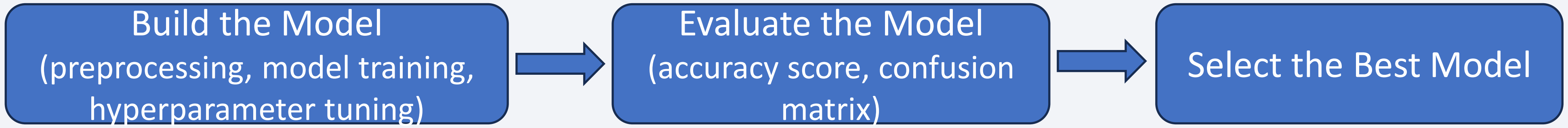
Build a Dashboard with Plotly Dash

- The dashboard application provides a real-time capability for interactive visual analytics of SpaceX data
- Interaction with the pie chart and scatter plot was enabled by adding:
    - A launch site drop-down input component
    - A callback function to render success-pie-chart based on selected site drop-down
    - A range slider to select payload
    - A callback function to render the success-payload-scatter-plot
- Adding a drop-down allows us to see the launch site with the largest successful launches
- Adding a callback function allows us to get the selected launch site from the drop-down and render a pie chart visualizing the launch success count
- Adding a range slider allows us to select different payload ranges and observe if variable payload is correlated to the mission outcome
- When we color-label the Booster versions on the scatter plot, adding a callback function allows us to observe mission outcomes with different Boosters.

GitHub URL

Predictive Analysis (Classification)

- Built the classification model through the following steps:
  - Load the data
  - Create a Numpy array from the column ‘Class’; output to a Pandas Series
  - Standardize the data
  - Perform train_test_split to split the data into training and testing data
  - Train the model and perform a GridSearchCV using four different methods: logistic regression, support vehicle machine, decision tree, and k-nearest neighbors
  - Fit the GridSearchCV object to find the best parameters and best score
- Evaluated the classification model through the following steps:
  - Calculate the accuracy of the test data using the method score()
  - Plot the confusion matrix
  - Examine the confusion matrix to identify major problems with the model
- Selected the best method:
  - We found that the method which performed best is ‘Decision Tree’, with a score of 0,8625


Build the Model (preprocessing, model training, hyperparameter tuning)
Evaluate the Model (accuracy score, confusion matrix)
Select the Best Model

GitHub URL

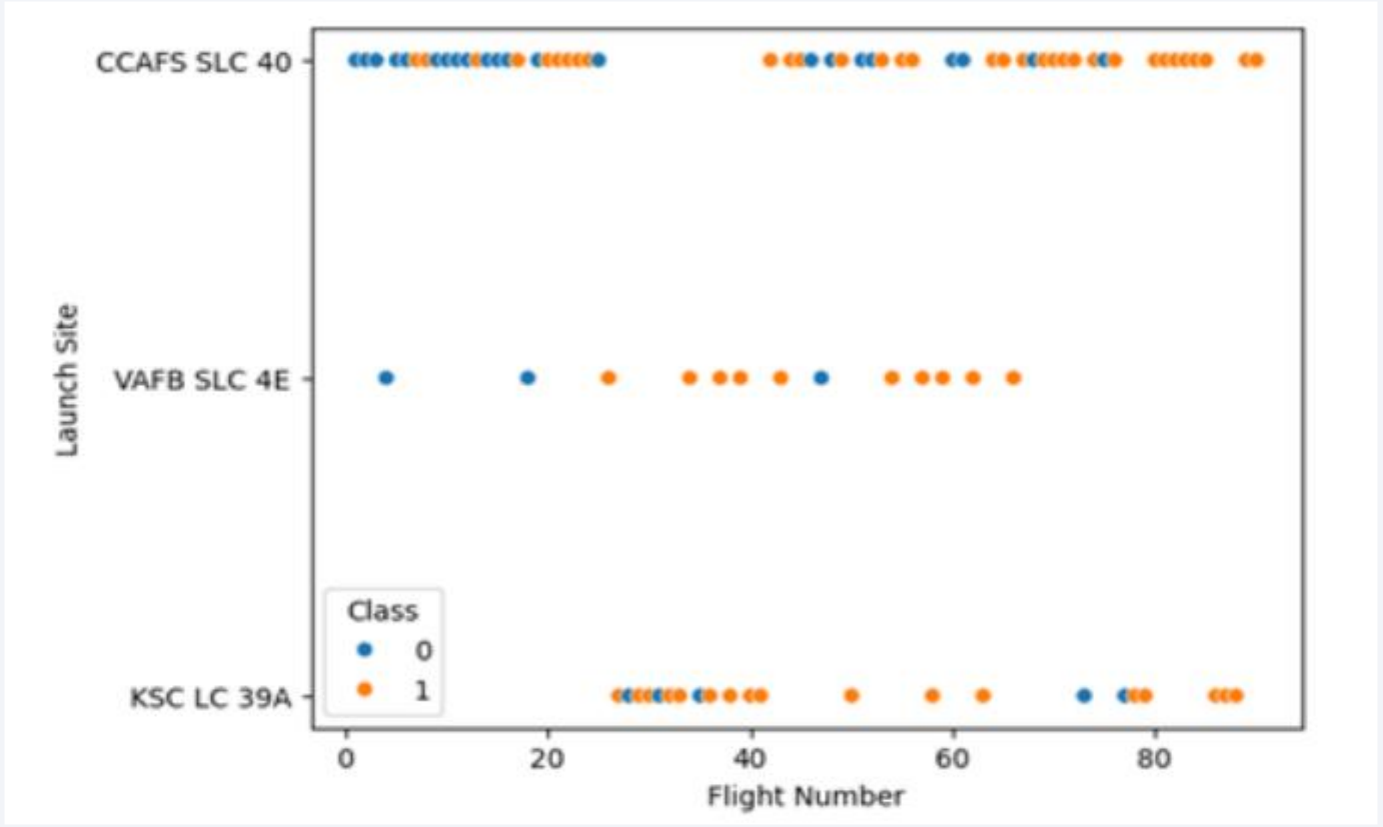
Results

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

Section 2

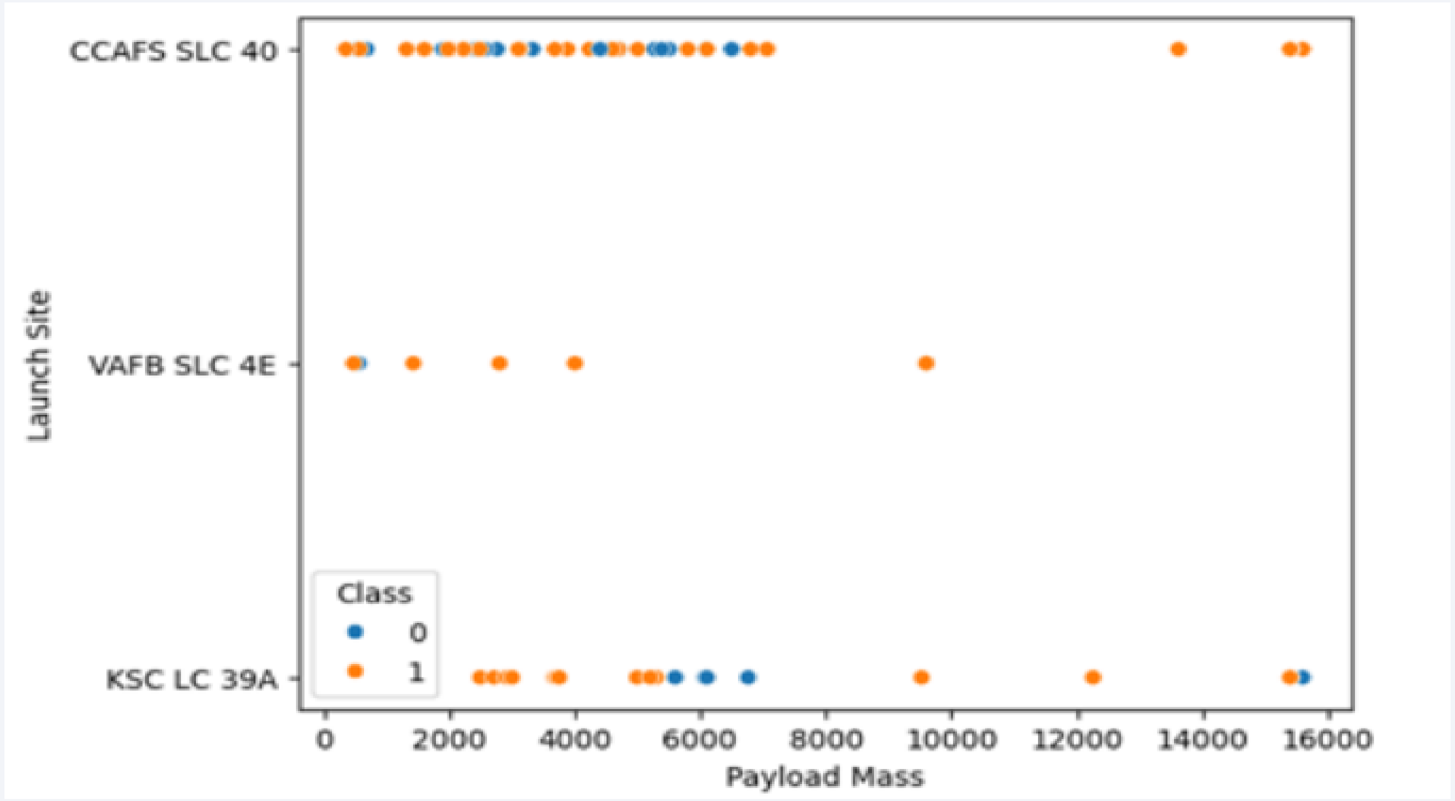
Insights drawn from EDA

Flight Number vs. Launch Site



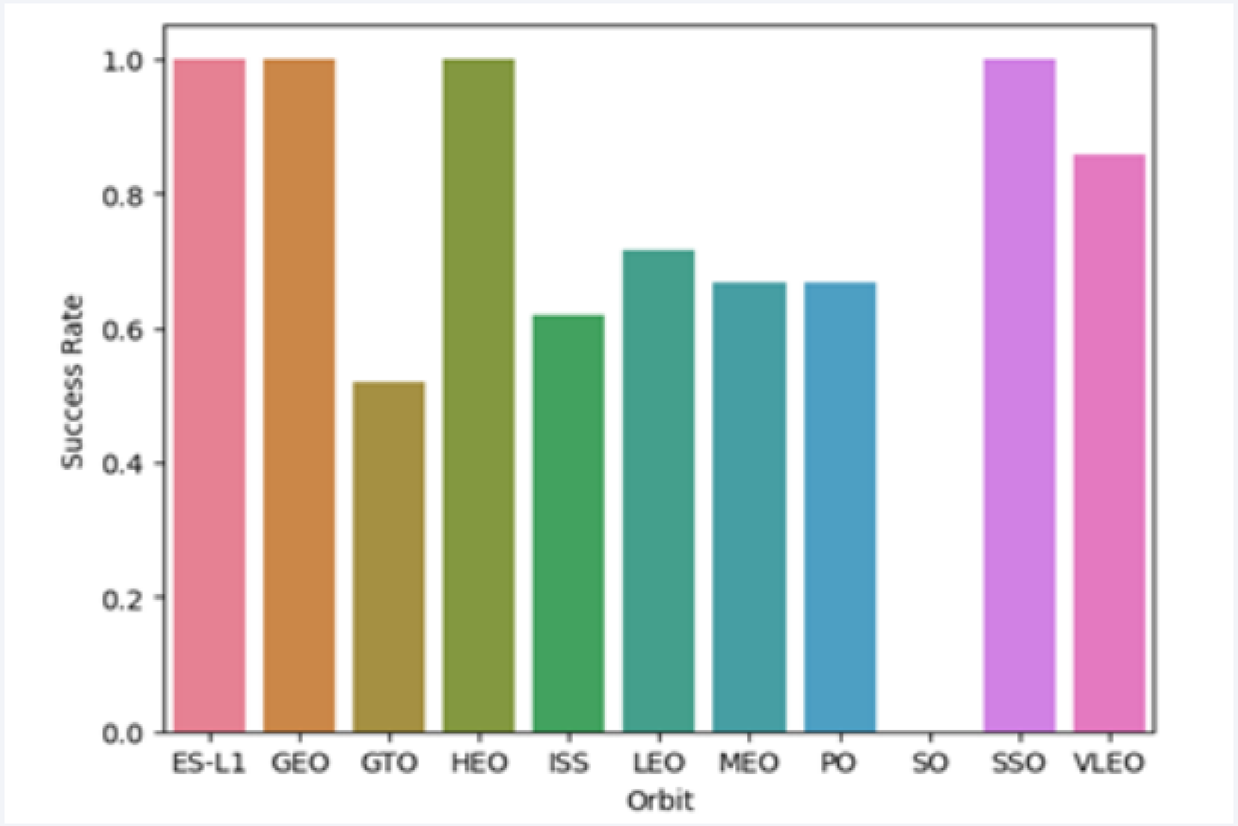
For each launch site, the number of flights appears to be related to success. As the number of flights increases, more successful landings are observed.

Payload vs. Launch Site



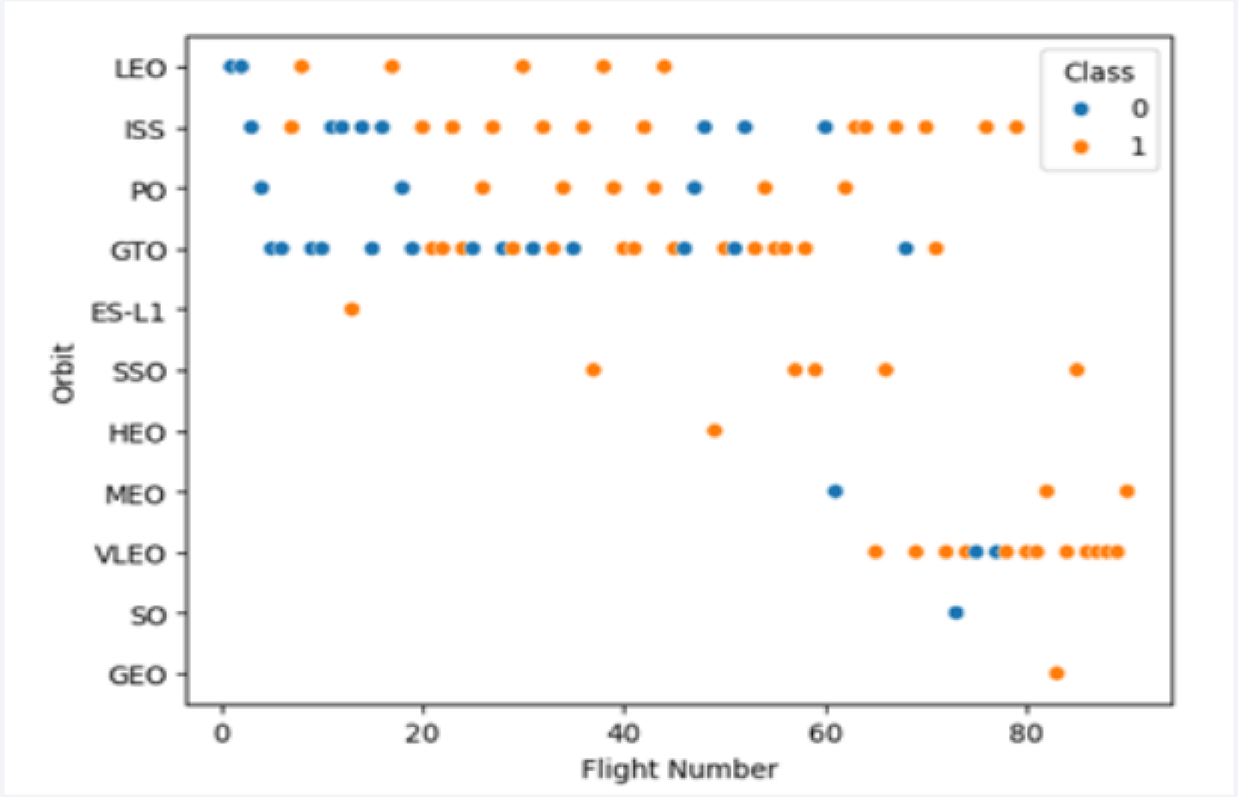
A significant number of rockets launched from the three sites have a payload mass below 8000 kg. For VAFB SLC 4E, there are few rockets launched with a payload mass greater than 10,000 kg. Success is correlated to low payload mass.

Success Rate vs. Orbit Type



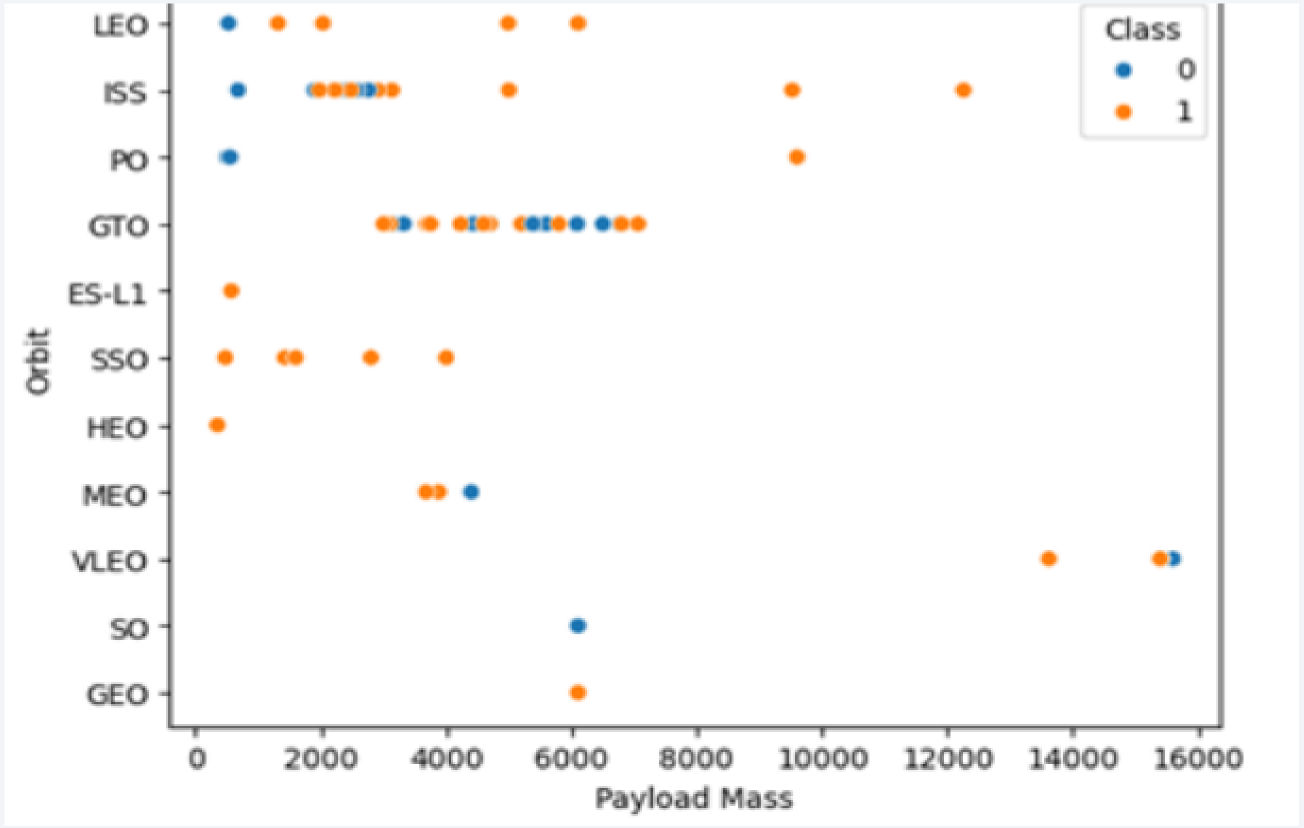
From the bar plot, it is evident that ES-L1, GEO, HEO, and SSO have the highest success rate, while GTO has the lowest success rate.

Flight Number vs. Orbit Type



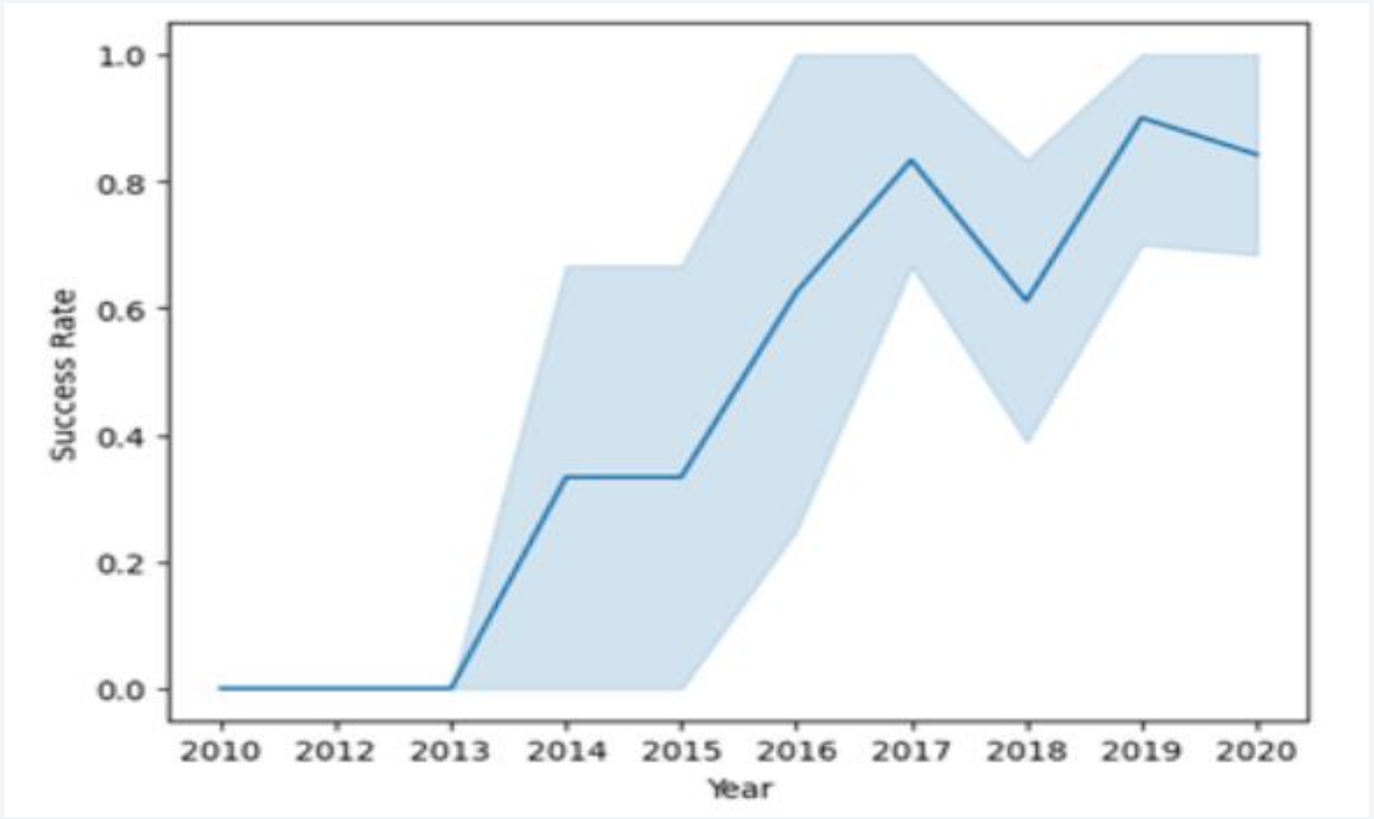
In the LEO orbit, the success appears to be related to flight number; whereas, in the GTO orbit, the success is unrelated to the number of flights.

Payload vs. Orbit Type



For heavy payload, the rate of successful landing is higher in LEO, ISS, and PO. This is not quite distinguishable in GTO, as we have both successful and failed landings here and there.

Launch Success Yearly Trend



As observed from the trend plot, the success rate since 2013 has been increasing till 2020

All Launch Site Names

```
SELECT DISTINCT Launch_Site FROM SPACEXTBL
```

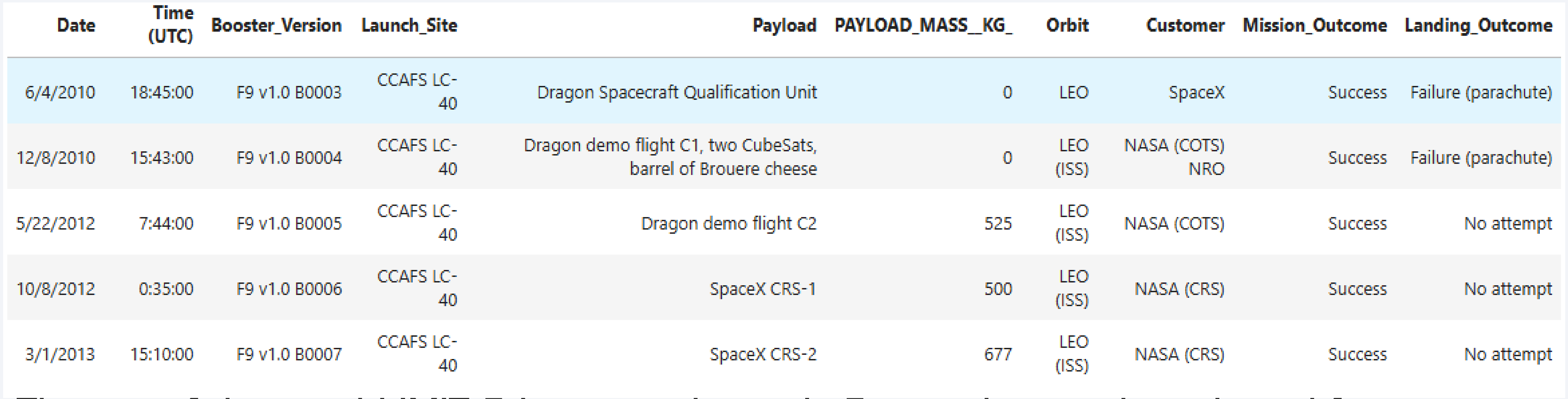
```
Launch_Site

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

- Using the DISTINCT statement in the SQL query ensured that only the four unique launch sites were returned from SPACEXTBL

Launch Site Names Begin with 'CCA'

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



Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
6/4/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
12/8/2010	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)
5/22/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
10/8/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
3/1/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

The use of the word LIMIT 5 instructs that only 5 records must be selected from SPACEXTBL. The where clause and the wild card suggest that the selected 5 records must have a launch site with the first three letters of the name being CCA.

Total Payload Mass

```
SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer='NASA (CRS)'
```

```
SUM(PAYLOAD_MASS__KG_)

45596
```

The SUM function totaled the amounts in the column PAYLOAD_MASS_KG_ of the SPACEXTBL table. The WHERE clause ensures that only the rows in which NASA (CRS) appears in the Customer column are affected.

Average Payload Mass by F9 v1.1

```
SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version='F9 v1.1’
```

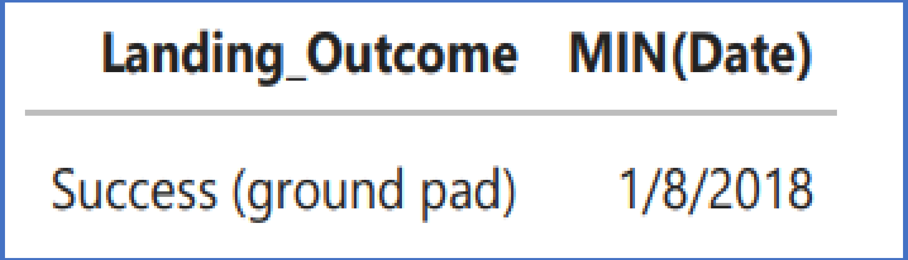
```
AVG(PAYLOAD_MASS__KG_)

2928.4
```

- The AVG function calculates the mean of the amounts in column PAYLOAD_MASS_KG_ of the SPACEXTBL table. The WHERE clause ensures that only the rows in which F9v1.1 appears in the Booster_Version column are affected.

First Successful Ground Landing Date

```
SELECT Landing_Outcome, MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome='Success(ground pad)’
```

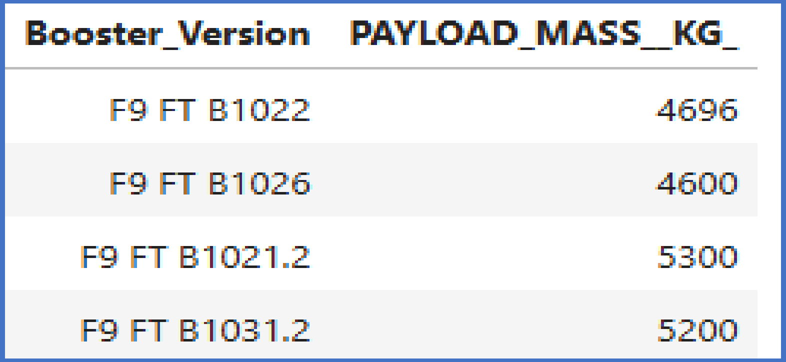


Landing_Outcome	MIN(Date)
Success (ground pad)	1/8/2018

The WHERE clause in the query ensures that only rows in which Success (ground pad) appears in the Landing_Outcome column are affected. The MIN function ensures that the record with the earliest date in the Date column is selected.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
SELECT Booster_Version, PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE Landing_Outcome='Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

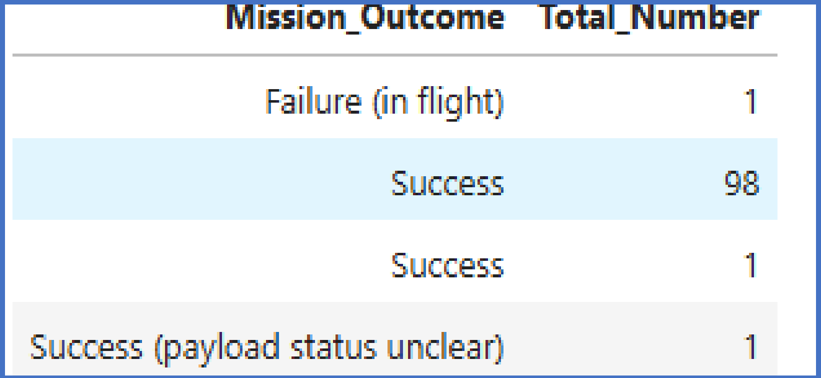


Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

The WHERE clause in the query ensures that only the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 are selected from SPACEXTBL table.

Total Number of Successful and Failure Mission Outcomes

```
SELECT Mission_Outcome, COUNT(*) Total_Number
FROM SPACEXTBL
GROUP BY Mission_Outcome
```

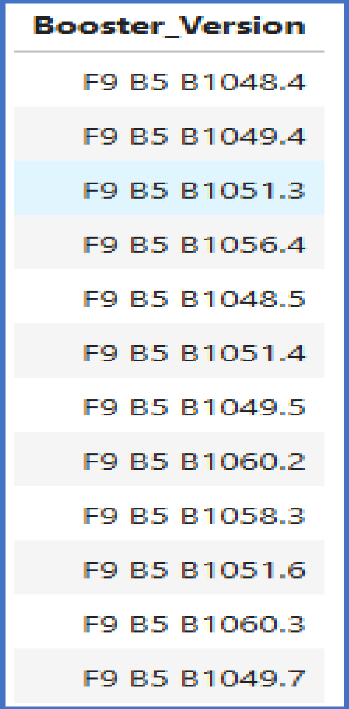


Mission_Outcome	Total_Number
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

The GROUP BY clause returns unique items in the Mission_Outcome column from the SPACEXTBL table. The function COUNT(*) calculates the total number of records in each of these groups (Success or Failure).

Boosters Carried Maximum Payload

```
SELECT Booster_Version
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT
MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

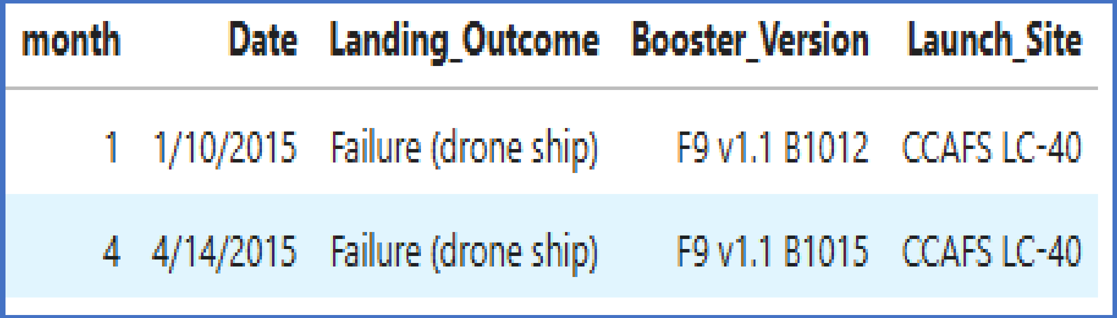


Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

The query returned 12 records. The use of a subquery ensured that only the instances of Booster_Version in which the highest amount appears in the column PAYLOAD_MASS_KG_ were selected from SPACEXTBL

2015 Launch Records

```
SELECT substr(Date, 0, 2) as month, Date, Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTBL
WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(Date, 6, 4)='2015'
```

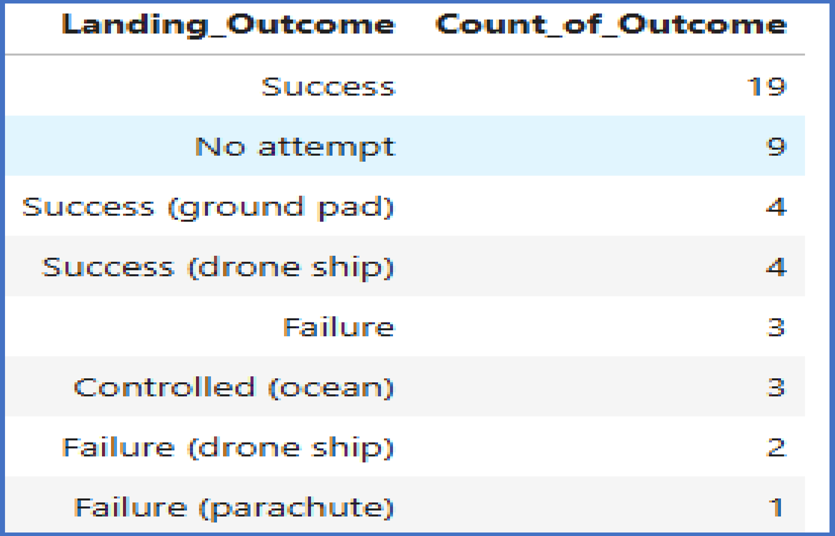


month	Date	Landing_Outcome	Booster_Version	Launch_Site
1	1/10/2015	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
4	4/14/2015	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- The WHERE clause ensures that launch records in which ‘Failure(drone ship)’ appears in column Landing_Outcome and 2015 appears in the column Date are selected from the SPACEXTBL table. The function substr() allows us to return the date in the desired format.

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

```
SELECT Landing_Outcome, COUNT(*) AS Count_of_Outcome
FROM SPACEXTBL
WHERE Date BETWEEN '06/04/2010' AND '3/20/2017'
GROUP BY Landing_Outcome
ORDER BY COUNT(*) DESC
```



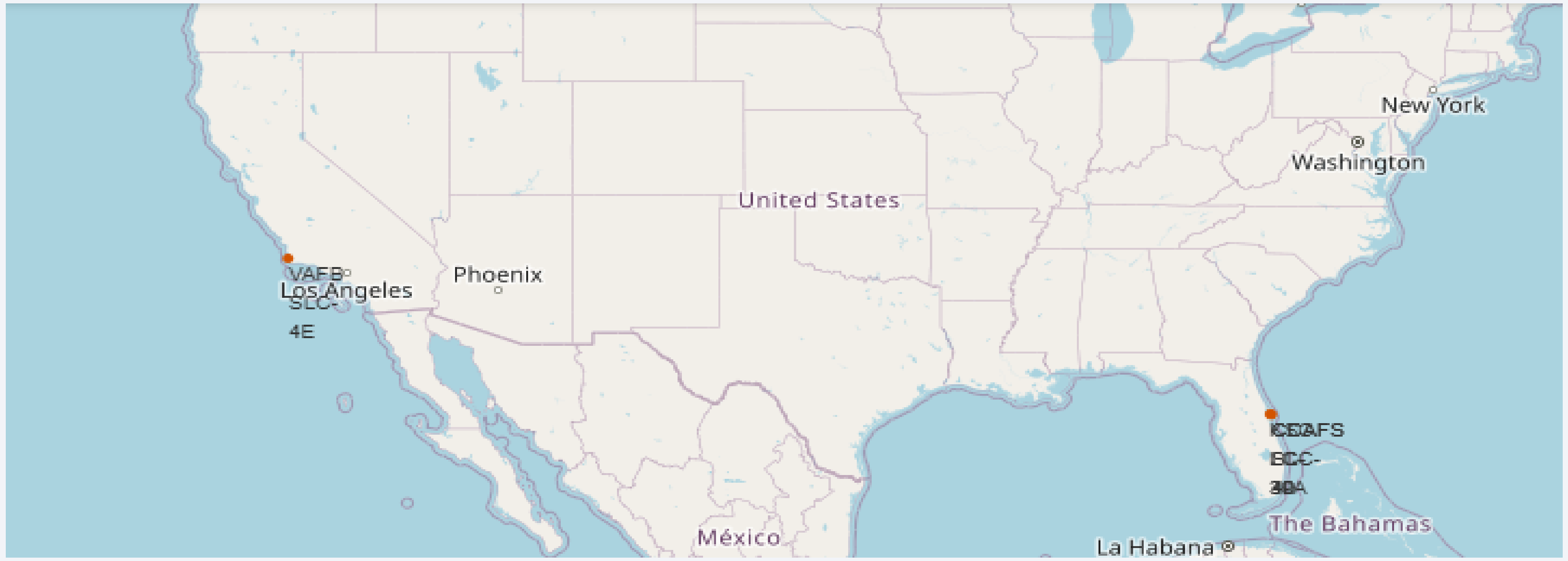
Landing_Outcome	Count_of_Outcome
Success	19
No attempt	9
Success (ground pad)	4
Success (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (drone ship)	2
Failure (parachute)	1

Using the BETWEEN clause with the WHERE clause ensures that the query selects launch records within the specified date interval. The GROUP BY clause returns the total number for each unique Landing_Outcome. The ORDER BY clause allows us to rank the landing outcomes.

Section 3

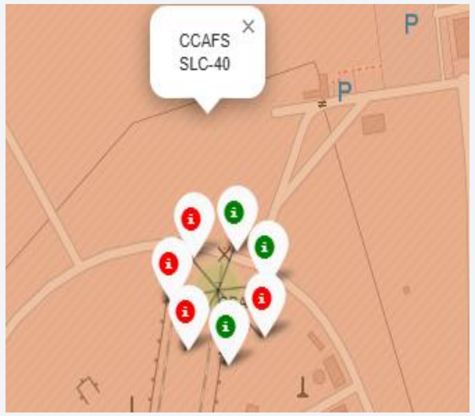
Launch Sites Proximities Analysis

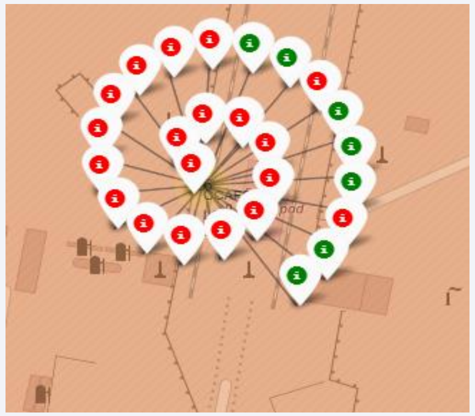
Folium Map of Marked Launch Sites

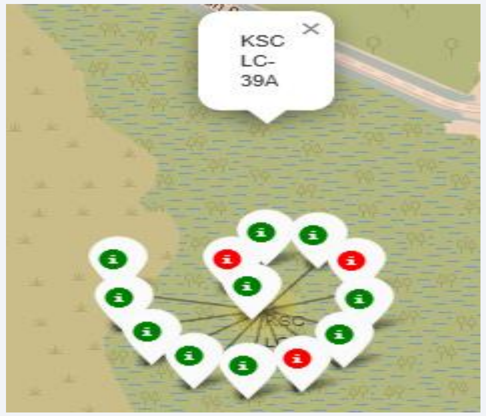


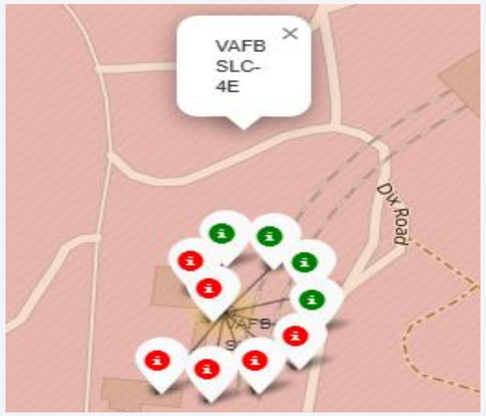
- The folium map shown above includes all launch sites as well as their location markers on a global map.
- One important observation from the map is that all four launch sites are located close to the coastline; the West Coast and East Coast of the United States.

Color-labeled Launch Outcomes with Folium





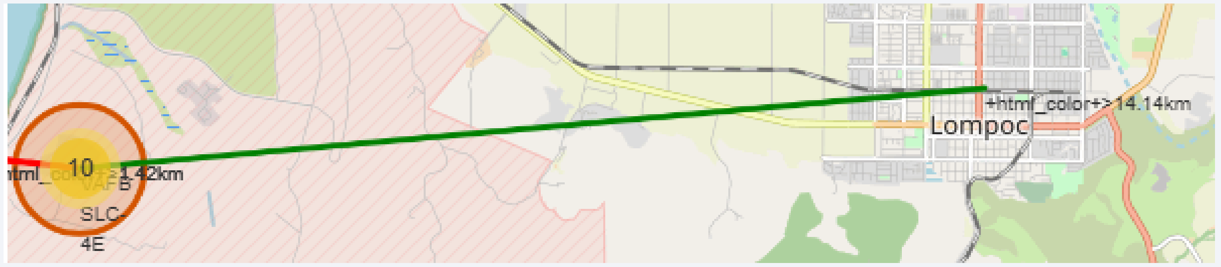




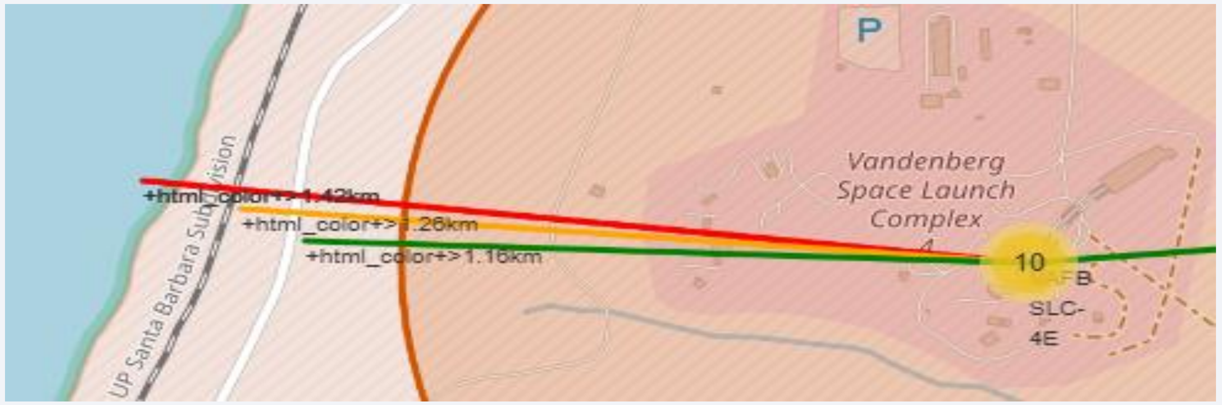
- The above screenshots are the color-labeled launch outcomes with Folium Map for the launch sites CCAFS SLC-40, CCAFS LC-40, KSC LC-39A, and VAFB SLC-4E, respectively.
- The GREEN color represents a successful launch while the RED represents a failed launch.
- From the color-labeled markers in the marker cluster, we observed that KSC LC-39A has the highest success rate of all the four launch sites because it contains the highest percentage of green markers.

Launch Site Proximities with Folium Map

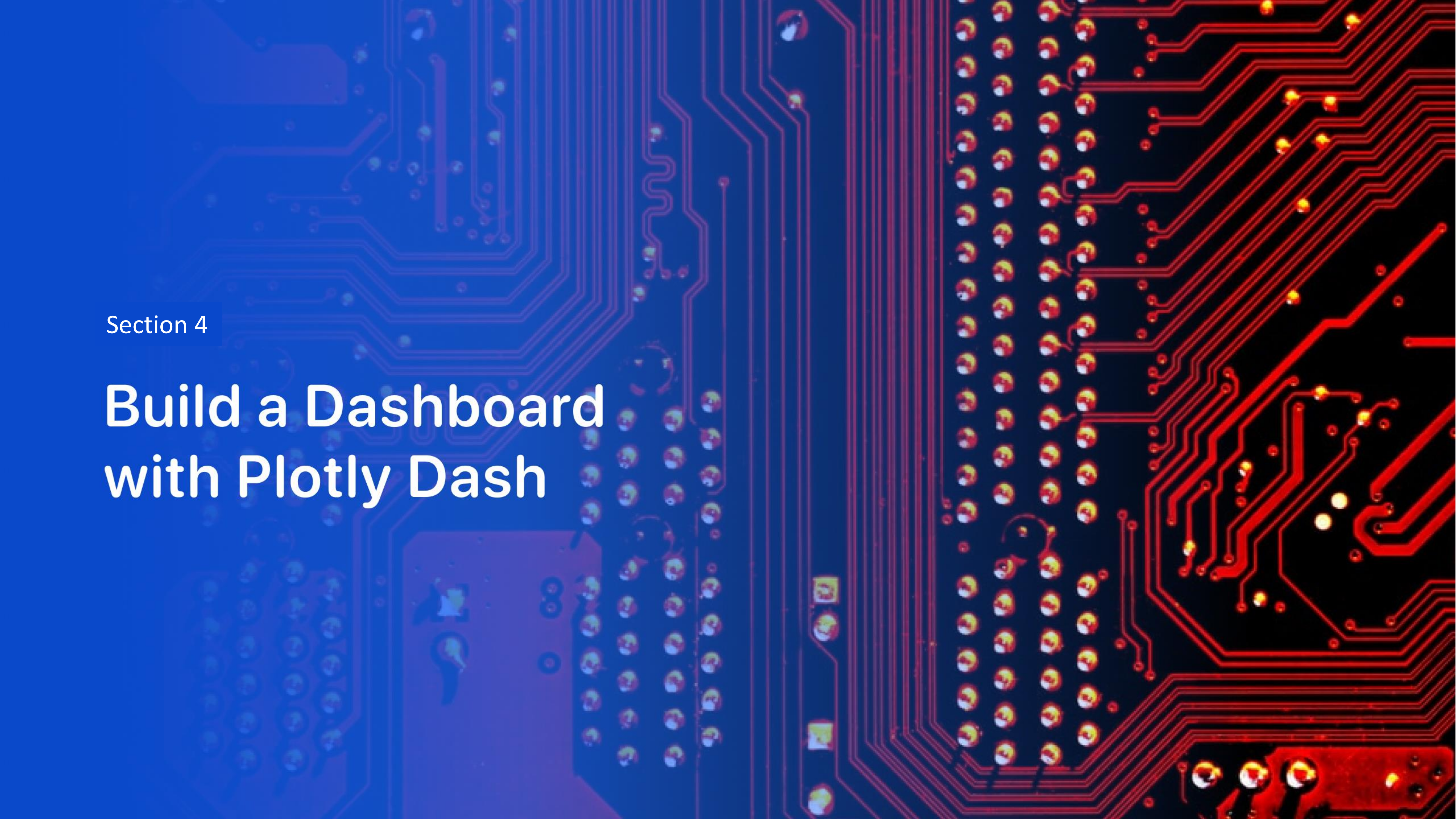
Map of VAFB SLC-4E showing the site's proximity to the closest city, Lompoc:
City Distance = 14.14 km



Map of VAFB SLC-4E showing the site's proximity to the highway, railway, and the coastline:
Highway Distance = 1.15 km
Railway Distance = 1.26 km
Coastline Distance = 1.42 km



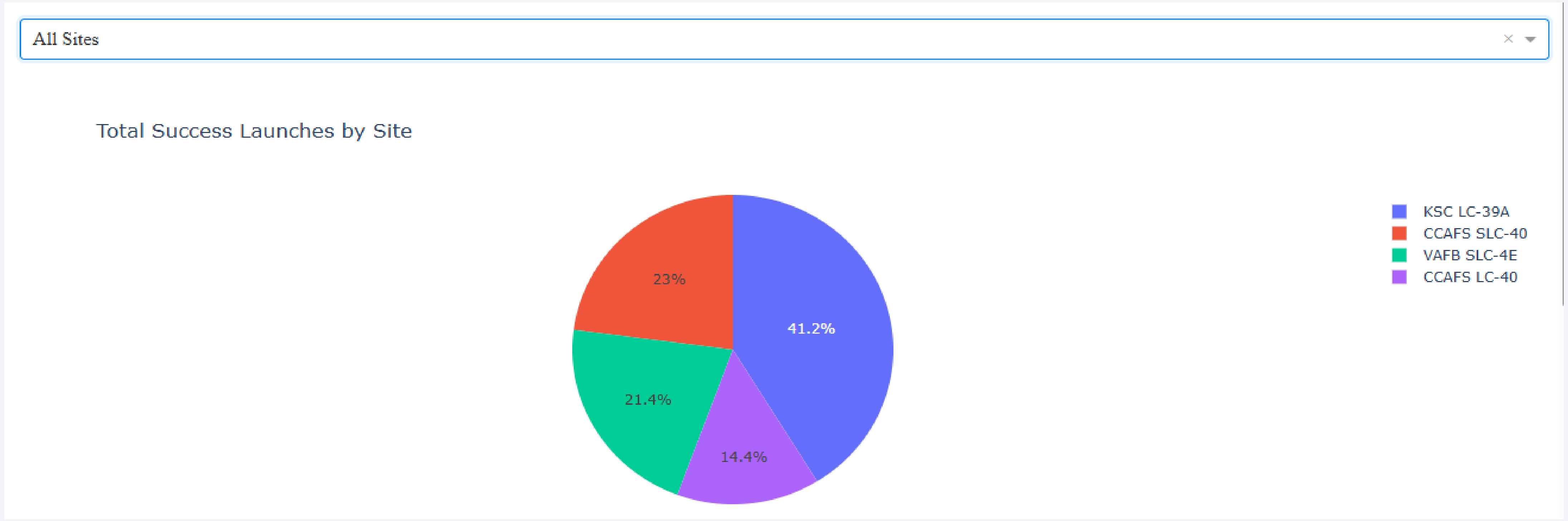
- The important finding on the screenshot is that the launch sites are located relatively far away from cities. On the other hand, launch sites are located in close proximity to highways, railways, and coastlines, understandably for ease of access and logistic support.



Section 4

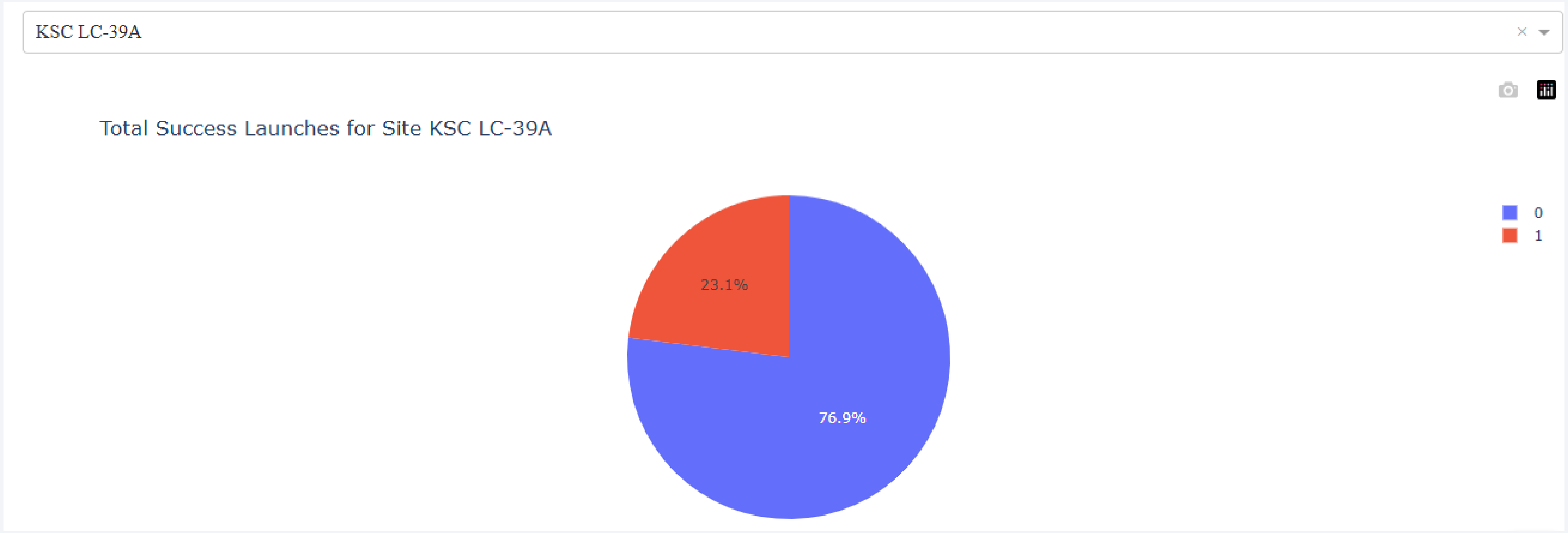
Build a Dashboard with Plotly Dash

Launch Success Count with Plotly Dash


All Sites

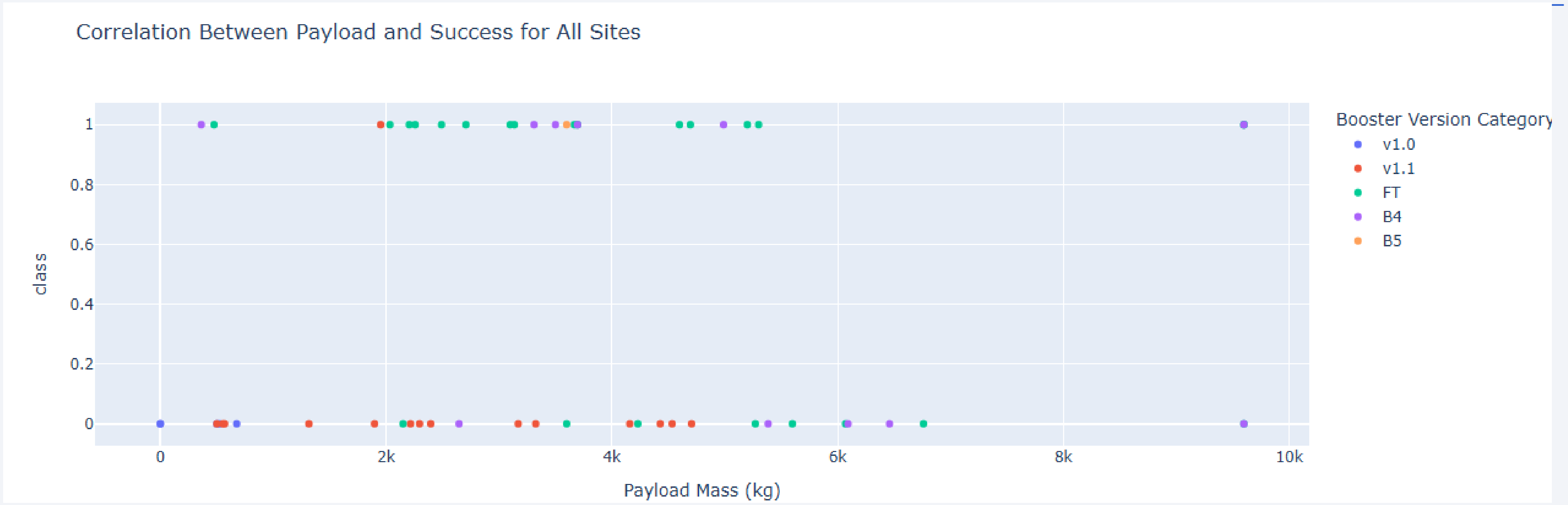
- The pie chart screenshot shows the launch success count for all sites
- Observed that KSC LC-39A has the highest number of successful launches

Launch Success Ratio with Plotly Dash



- The pie chart screenshot shows KSC LC-39A being the launch site with the highest launch success ratio
- Observed that this site has a 76.9% success rate.

Payload vs. Launch Outcome with Plotly Dash

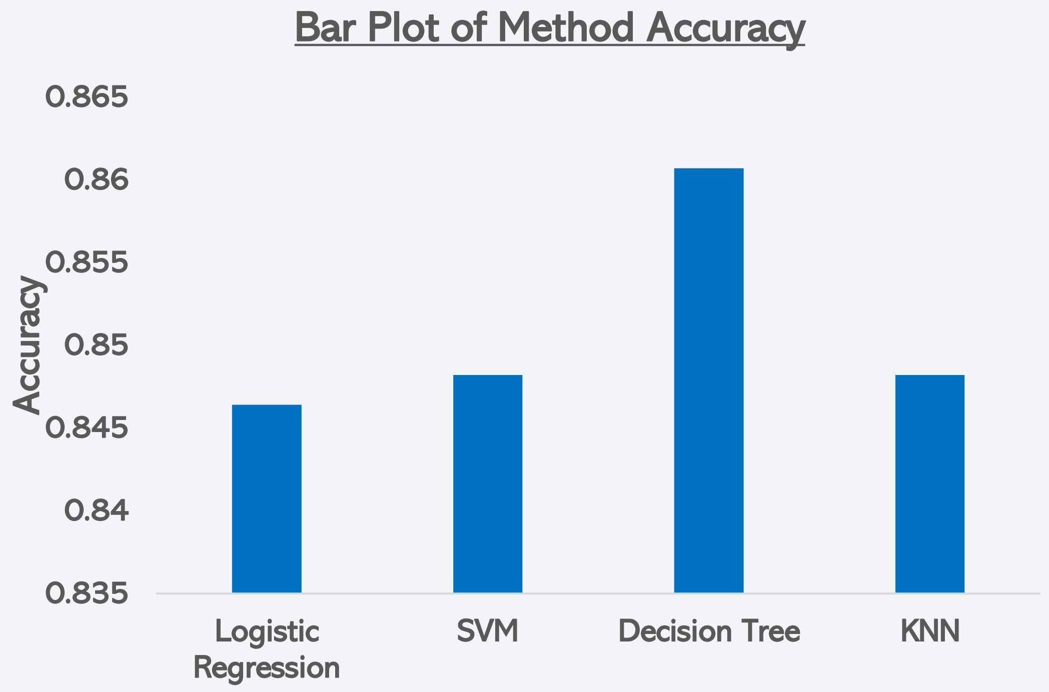


The scatter plot's screenshot shows the correlation of the Payload vs. Launch Outcome for all sites. We observe that Booster Version FT has the largest success rate at a payload range of up to 6000 kg. For a payload range above 6000 kg, it becomes almost impossible to achieve a successful landing with any of the booster versions.

Section 5

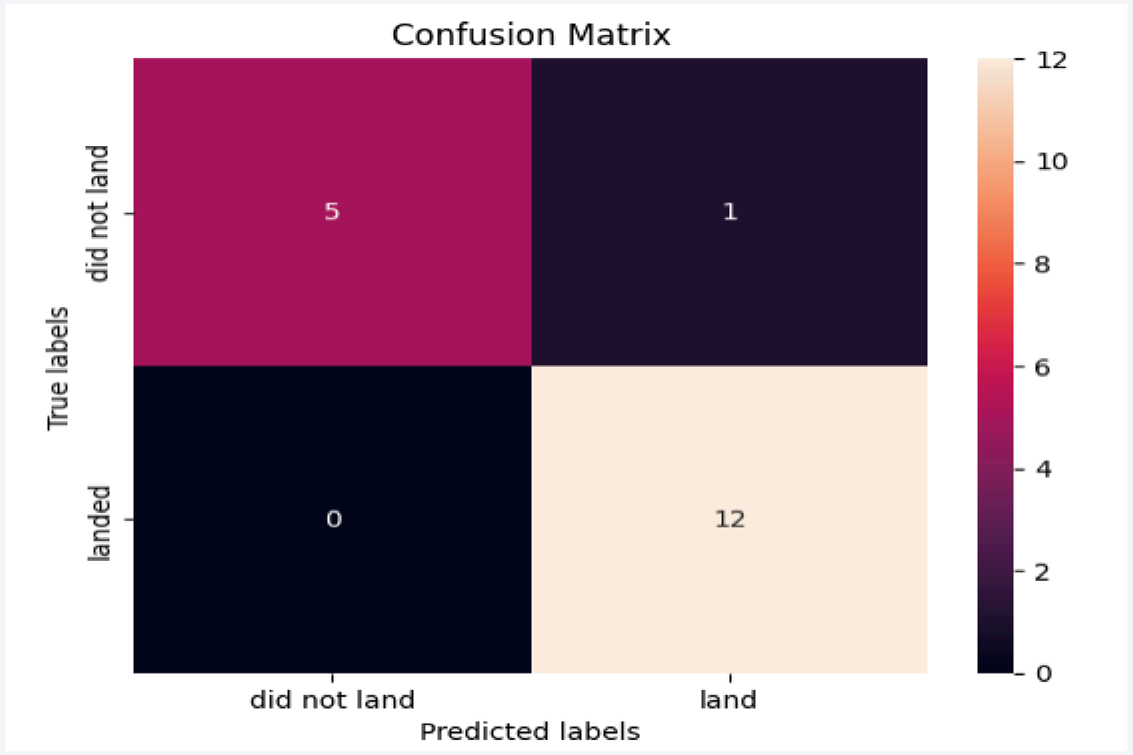
Predictive Analysis (Classification)

Classification Accuracy



From the above bar plot, it is clear that 'Decision Tree' gave us the highest accuracy of 0.86

Confusion Matrix for Decision Tree Model



Examining the confusion matrix, we see that the Decision Tree model can distinguish between the different classes, except for a minor problem with just one false positive (FP). However, the true positives (TP), the false negatives (FN), and the true negatives (TN) are all accurate.

Conclusions

- Launch sites are located relatively far from cities, but close to railways, highways, and coastlines.
- Successful landing is hard to achieve at a payload range above 6000kg.
- KSC LC-39A is the launch site with the highest success ratio.
- GEO, HEO, ES-L1, and SSO orbits have the highest success rate. GTO orbit has the lowest success rate.
- The Booster Version FT has the largest success rate.
- The success rate of launches has been increasing steadily from 2013 up till 2020.
- Successful landing is correlated with the number of flights recorded at the launch sites.
- The best classification algorithm for predicting launch outcomes is the Decision Tree method.

Appendix

- Plotly Express in Python
  - https://plotly.com/python/plotly-express/
- Dash HTML Components
  - https://dash.plotly.com/dash-html-components
- Dash Core Components
  - https://dash.plotly.com/dash-core-components
- Dash Documentation & User Guide
  - https://dash.plotly.com/

Thank you!