IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

CY
December 26, 2023

Outline

Executive Summary

Introduction

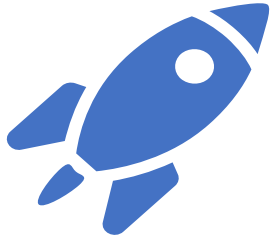
Methodology

Results

Conclusion

Appendix

Introduction



Project background and context

The goal of this project is to create a machine learning pipeline to predict if the SpaceX Falcon 9 rocket first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost an upward of 165 million dollars each. Much of the savings is because SpaceX can reuse the first stage. Thus, by determining if the first stage will land, we can determine the launch cost. 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 factors affect the first-stage landing success?

How does the relationship among different features affect the success rate of the launch outcome?

What is an optimal location for building a launch site?

What is the best predictive model for a successful first-stage landing?

Executive Summary

Summary of methodologies

- Data Collection through API
- Data Collection with Web scraping
- Data Wrangling
- Exploratory Data Analysis using SQL
- Exploratory Data Analysis with Data Visualization
- Interactive Visual Analytics with Folium
- Interactive Dashboard with Plotly Dash
- Machine Learning Prediction

Summary of all results

- Exploratory Data Analysis
- Interactive Visual Analytics and Dashboard
- Predictive Analysis

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Collect rocket launch data from the SpaceX REST API using the GET request and web scraping from the Falcon9 Launch Wiki page
- Perform data wrangling
  - Process the data by filtering, calculating missing values, and applying one-hot encoding to 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
  - Tune and evaluate the classification models to find the best model and hyperparameters

Data Collection

- Collect rocket launch data from the SpaceX REST API using the GET request
- Convert the response content to a dataframe using .json_normalize()
- Clean the data by checking for missing values and replacing them with the mean
- Scrape the Falcon 9 launch records HTML table from Wikipedia using BeautifulSoup and the HTTP GET method
- Parse the HTML table into a dictionary to create a dataframe
- Export data to a CSV file for analysis

Data Collection – SpaceX API

- Step 1: Collect SpaceX launch data using the GET request
- Step 2: Filter the dataframe to only include Falcon 9 launches
- Step 3: Replace missing values for PayloadMass by calculating the mean value
- GitHub: Link

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-S
```

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

```
data_falcon9=data[data['BoosterVersion']!='Falcon 1']
```

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

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

```
# Calculate the mean value of PayloadMass column
PayloadMassMean=data_falcon9['PayloadMass'].mean()
print('The mean value of PayloadMass is:', PayloadMassMean)

# Replace the np.nan values with its mean value
data_falcon9.replace(np.nan, PayloadMassMean, inplace=True)
```

```
The mean value of PayloadMass is: 6123.547647058824
```

Data Collection - Scraping

- Step 1: Collect Falcon9 launch data using BeautifulSoup and the HTTP GET method
- Step 2: Extract column names from the HTML table header
- Step 3: Parse the launch HTML tables into a Pandas dataframe using a dictionary with keys from the extracted column names
- GitHub: Link

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launche
```

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

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

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

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

```
# Use soup.title attribute
print(soup.title)
```

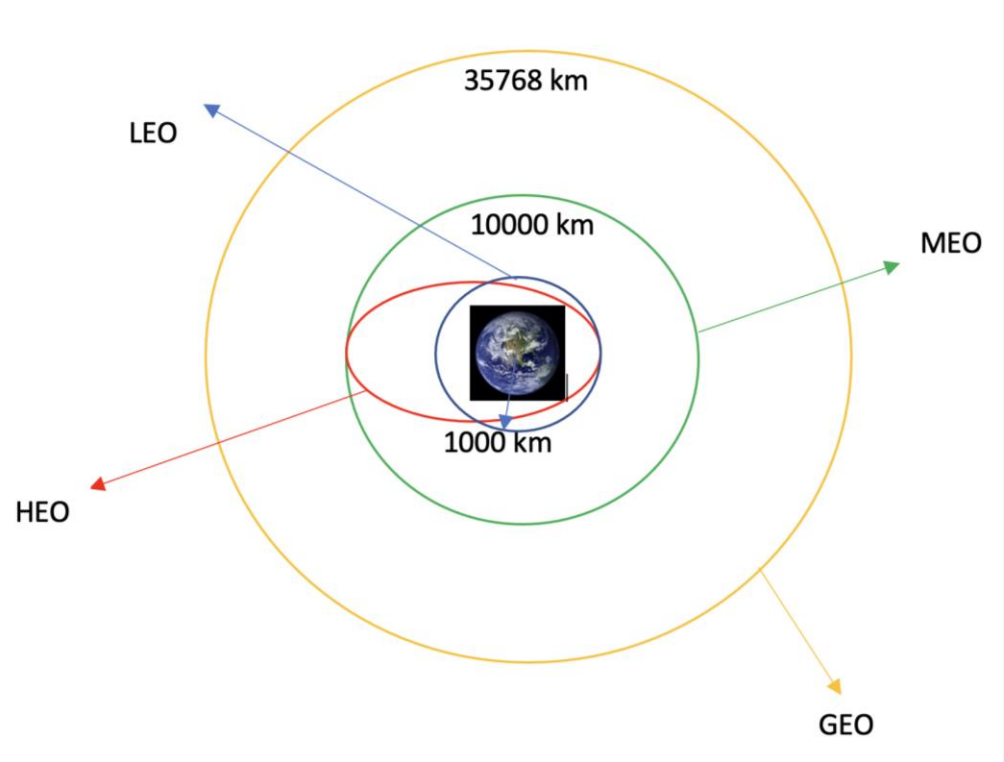
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

```
column_names = []
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)
```

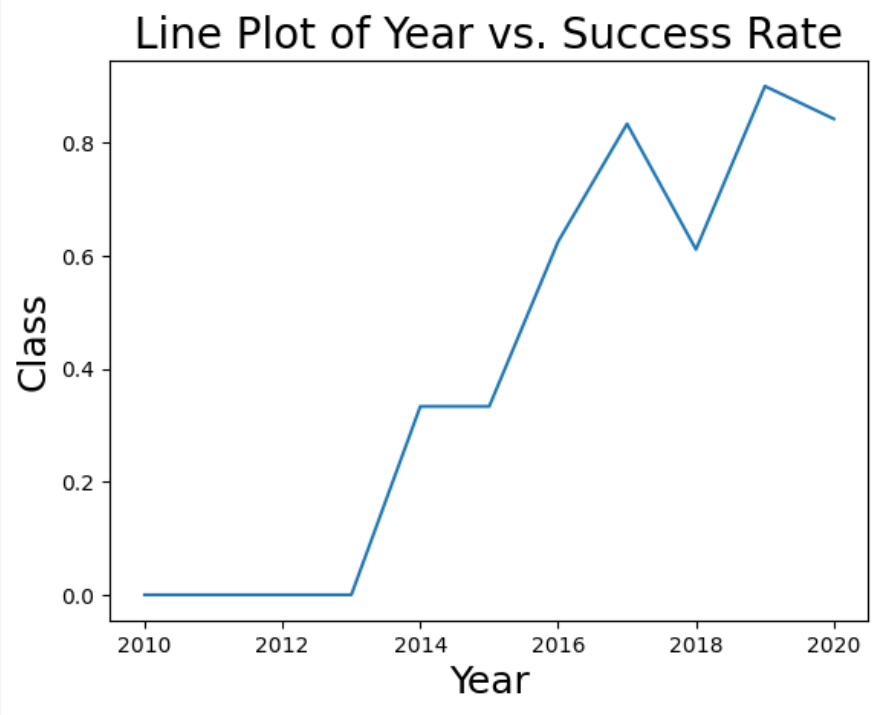
```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

Data Wrangling

- Perform Exploratory Data Analysis and determine training labels
- Calculate the number of launches per site, and the number and occurrence of each orbit and mission outcome per orbit type
- Create a landing outcome label from the outcome column and export to a CSV
- GitHub: Link

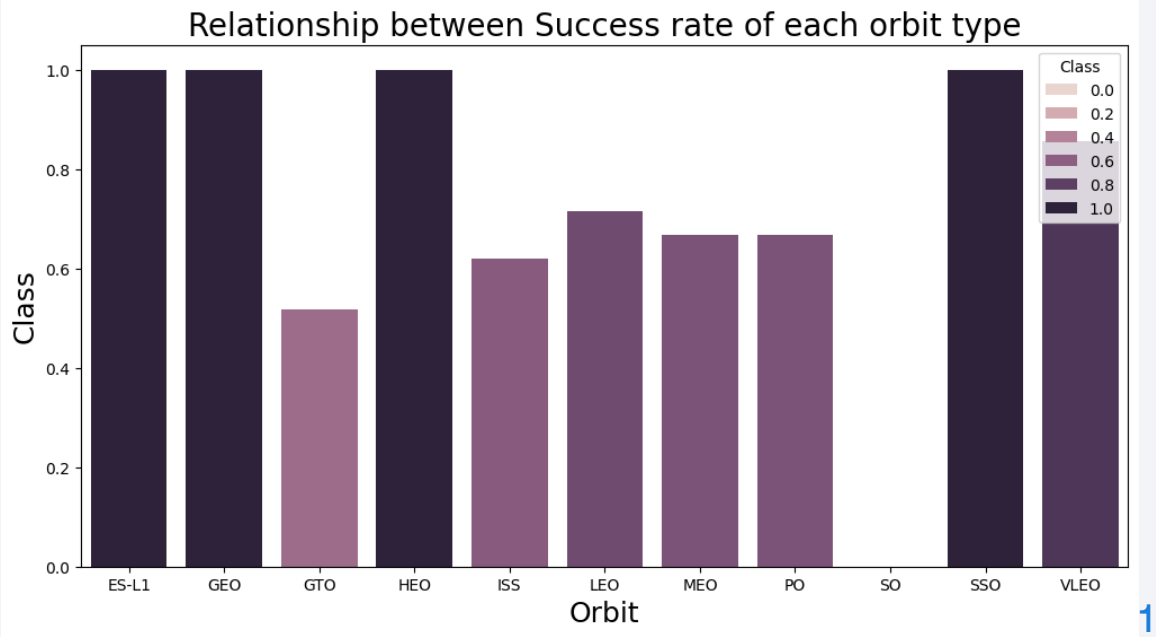


EDA with Data Visualization



- Visualize the launch success yearly trend using a line plot to observe the success rate trend over time
- GitHub: Link

- Visualize the relationship between payload mass and flight number, flight number and launch site, payload mass and launch site, and success rate of each orbit type using categorical plots and a bar chart



EDA with SQL

- Load SQL extension and establish a connection with the SQLite database.
- Write and execute queries to find out:
  - Names of the unique launch sites in the space mission
  - Total payload mass carried by boosters launched by NASA (CRS)
  - Average payload mass carried by booster version F9 v1.1
  - Date when the first successful landing outcome in ground pad was achieved
  - Names of boosters which have success in drone ship and payload mass between 4000 and 6000 kg
  - Total number of successful and failure mission outcomes
  - Names of booster versions which carried the maximum payload mass
  - Failure landing outcomes in drone ship for months in the year 2015
- GitHub: Link

```
%load_ext sql
```

```
import csv, sqlite3

con = sqlite3.connect("my_data1.db")
cur = con.cursor()
```

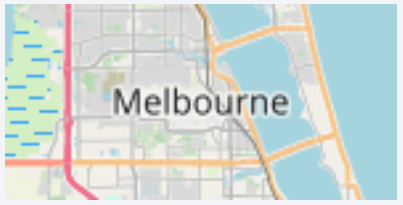
```
!pip install -q pandas==1.1.5
```

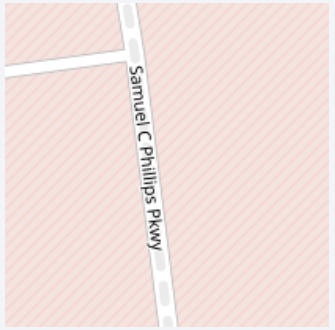
```
%sql sqlite:///my_data1.db
```

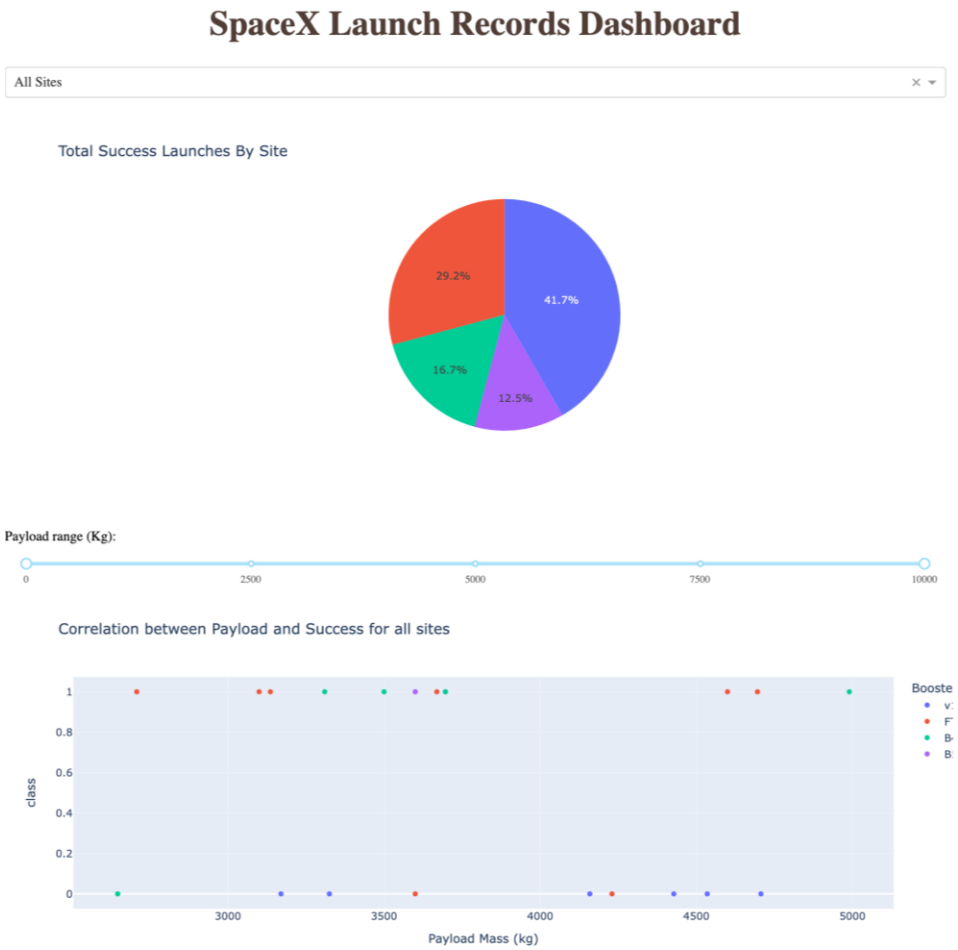
```
'Connected: @my_data1.db'
```

Build an Interactive Map with Folium

- Mark all launch sites on a map and add map objects such as markers, circles, and lines to indicate successful and failed launches for each site on the map.
- Assign class values to markers for all launch records indicating if a launch was successful with class=1 and if a launch was failed with class=0.
- Use color-labeled markers to easily identify which launch sites have relatively high success rates.
- Calculate the distances between a launch site to its proximities such as railways, highways, coastlines, and cities using MousePosition and PolyLine to determine if launch sites keep a certain distance away from cities and if launch sites are in close proximity to highways, coastlines, and railways.
- GitHub: Link







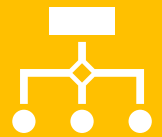
Build a Dashboard with Plotly Dash

- Use Plotly Dash to build an interactive dashboard.
- Plot pie charts to show the total successful launches by launch site and scatter chart to show the correlation between Payload Mass (kg) and Success Outcome for different booster versions.
- GitHub:
  - Code link
  - Dashboard link

Predictive Analysis (Classification)

Load data using pandas and numpy, standardize and transform the data, and split the data X and Y into training and test data.

Build and fit KNN, Decision Tree, Logistic Regression, and SVM machine learning models and tune different hyperparameters using GridSearchCV.



Calculate the accuracy scores, plot confusion matrices, and improve the model using feature engineering and algorithm tuning to find the method that performs the best.

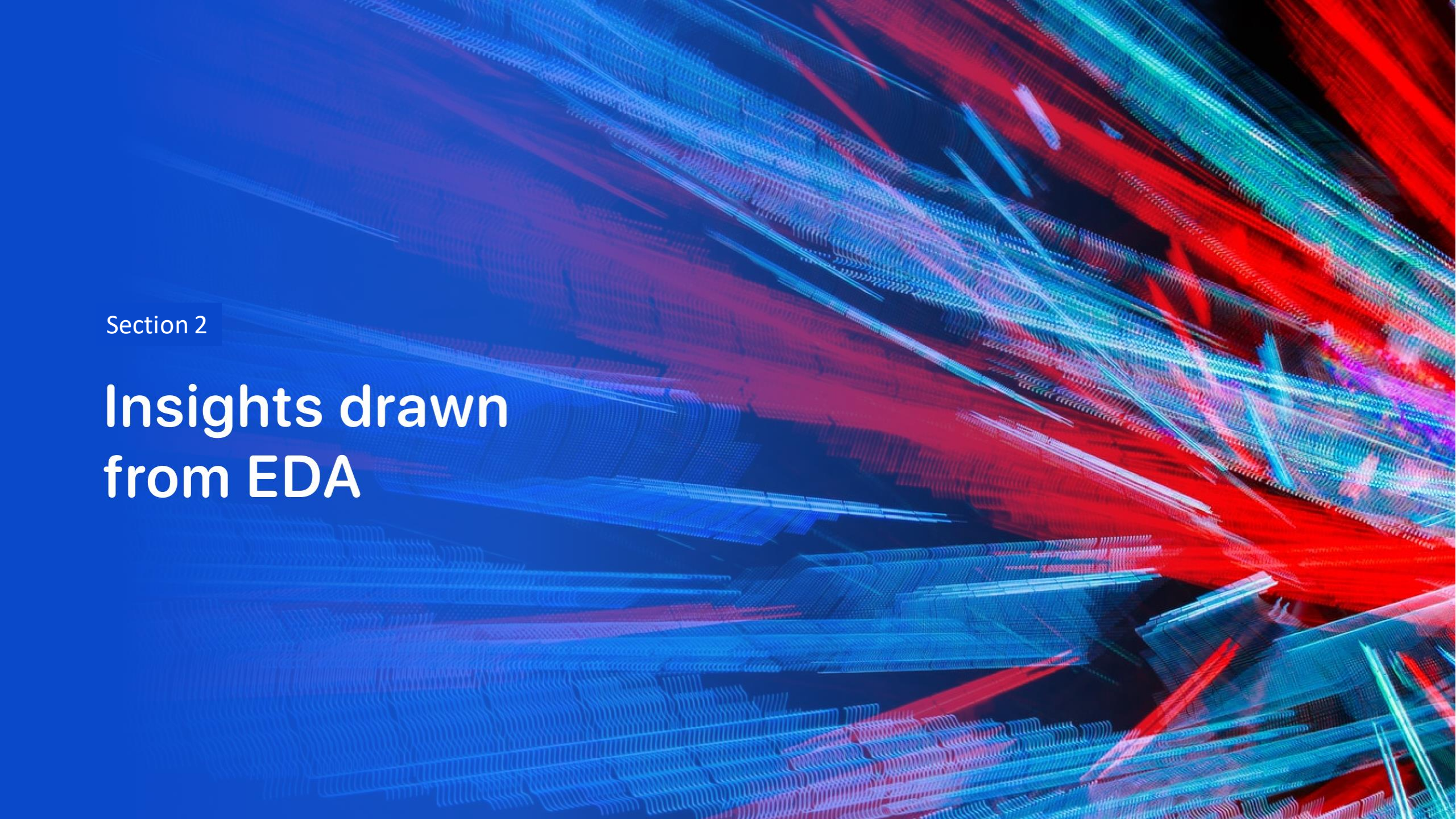
GitHub: Link

Results

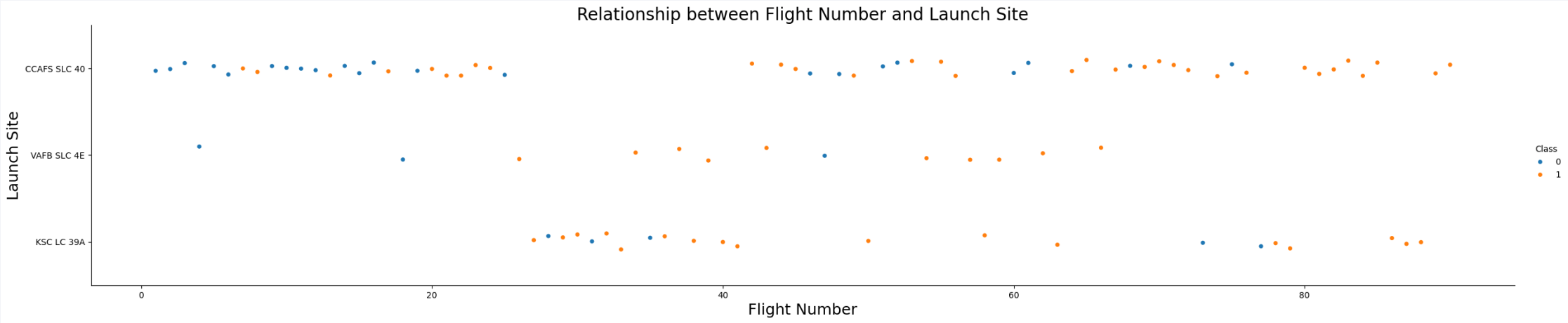
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

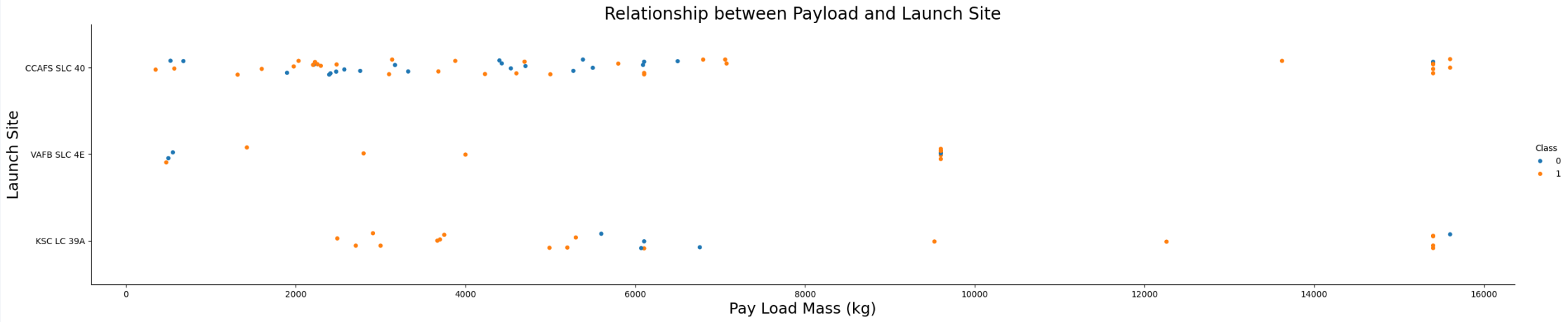


Flight Number vs. Launch Site

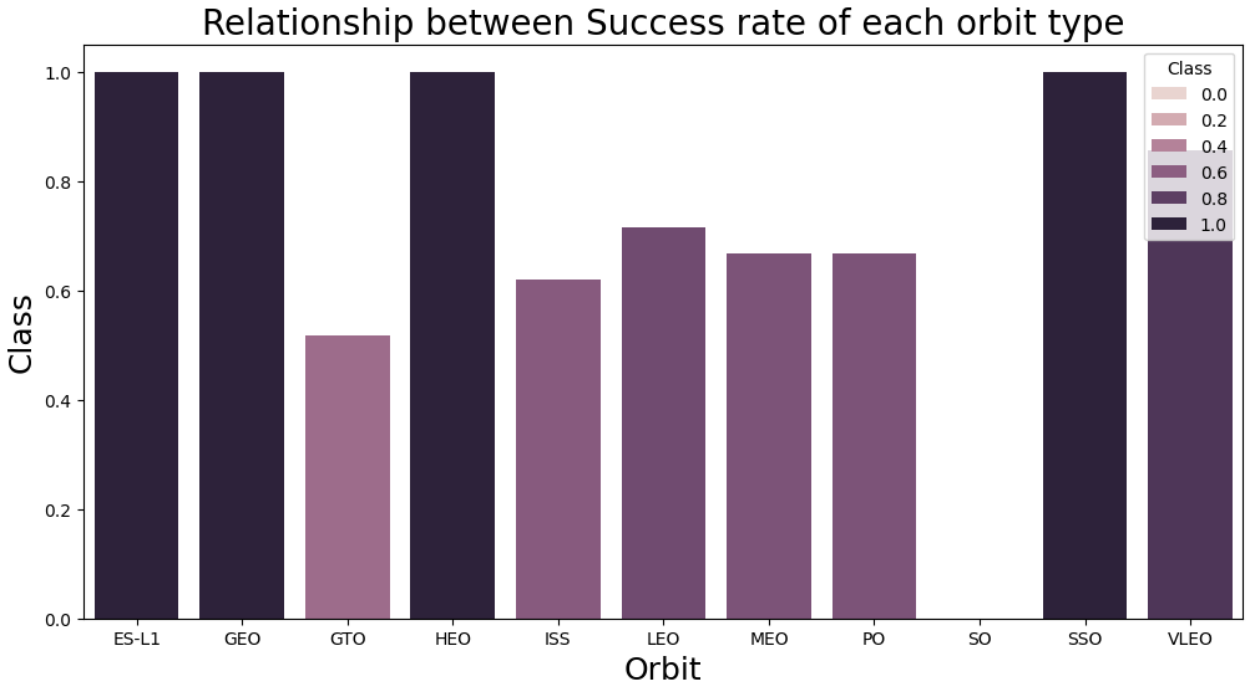


- The scatter plot of Flight Number vs. Launch Site shows that as the number of flights increases, the success rate at a launch site also increases.

Payload vs. Launch Site



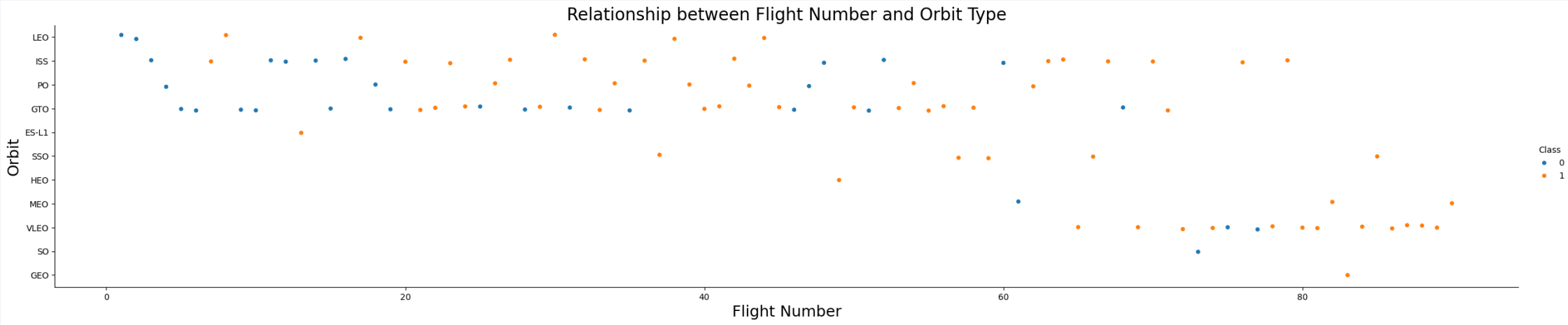
- The scatter plot of Payload vs. Launch Site shows that the launch success rate increases for a heavy payload mass (kg) greater than 12,000 for CCAFS SLC-40.
- For VAFB SLC-4E launch site, there are no rockets launched for heavy payload mass (kg) greater than 10,000.



Success Rate vs. Orbit Type

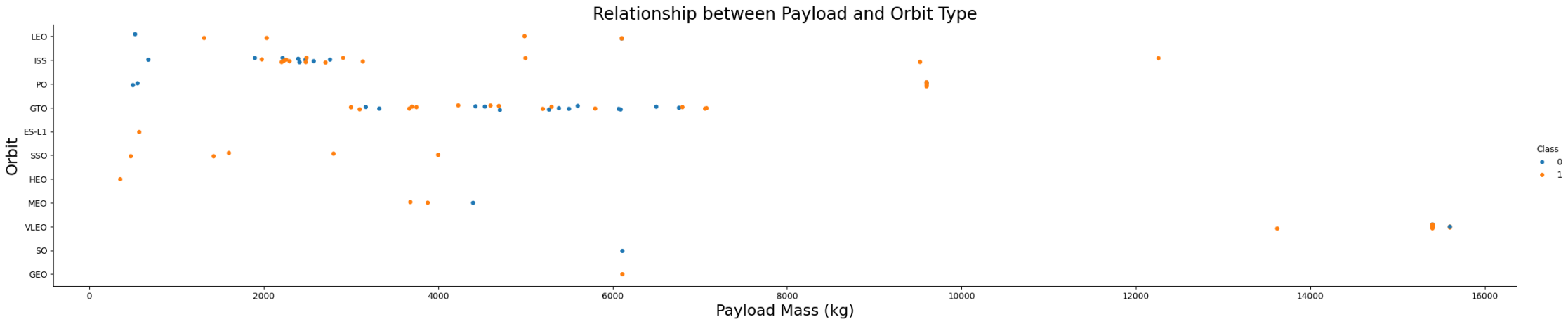
- The bar chart for the success rate of each orbit type shows that orbits ES-L1, GEO, HEO, and SSO have the highest success rates.
- Orbit SO has the lowest success rate.

Flight Number vs. Orbit Type



- The scatter plot of Flight number vs. Orbit type shows that in the LEO orbit, the success appears to be related to the number of flights.
- On the other hand, there seems to be no relationship between flight number for the GTO orbit.

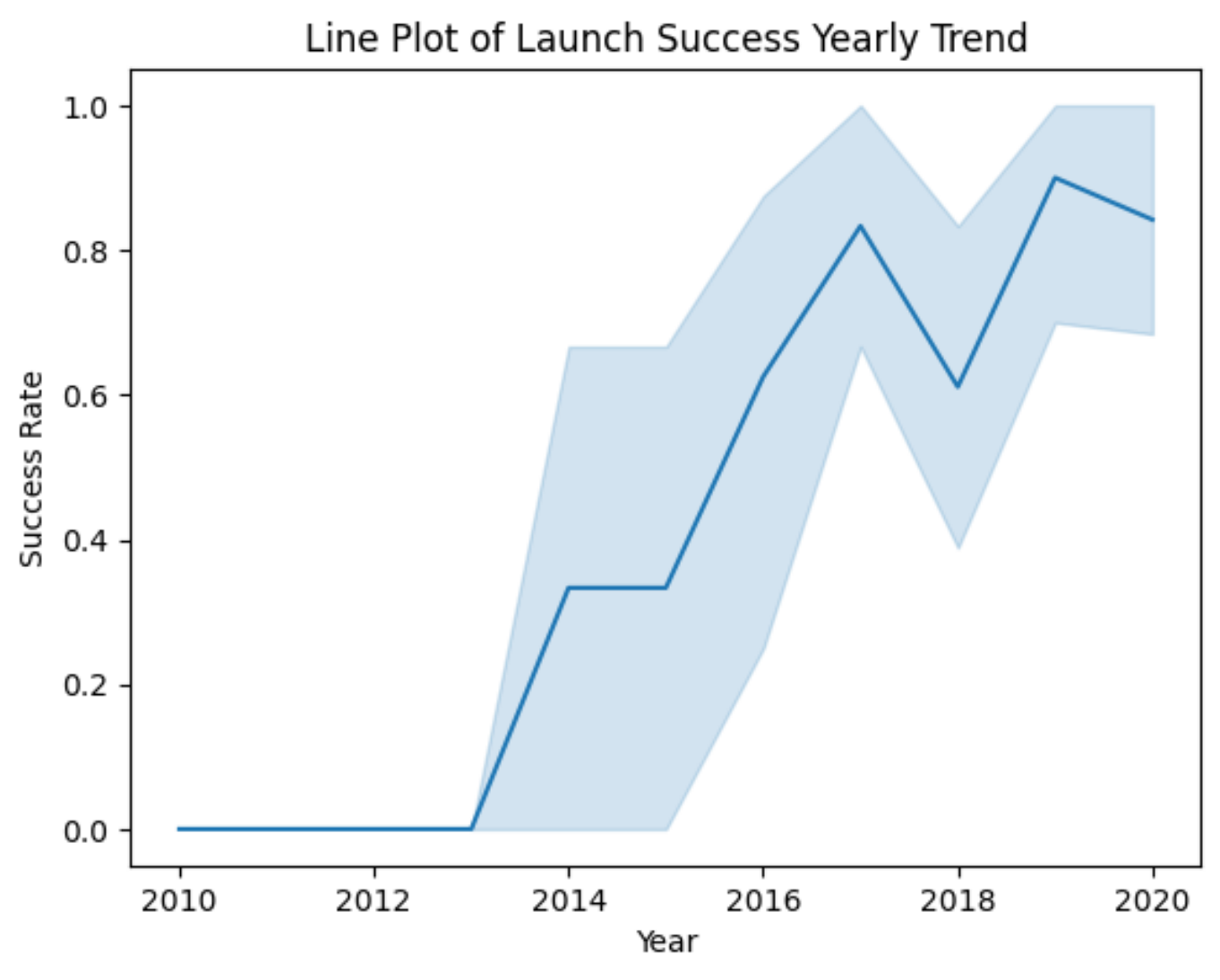
Payload vs. Orbit Type



- The scatter plot of payload vs. orbit type shows that with heavy payloads over 5000 kg, the success rates are higher for orbits PO, LEO, and ISS.

Launch Success Yearly Trend

- The line chart of yearly average success rate shows that success rate has been increasing since 2013 until 2020, with a slight dip in 2018.



All Launch Site Names

Display the names of the unique launch sites in the space mission

```
%sql select distinct Launch_Site from SPACEXTABLE;
```

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

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

- Find the names of the unique launch sites using SELECT DISTINCT.
- The SELECT DISTINCT statement is used to return only unique rows/values from a table.
- 4 unique launch sites are displayed: CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, and CCAFS SLC-40.

Launch Site Names Begin with 'CCA'

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

```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5;
```

Python

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

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

- Use LIMIT 5 to display 5 records where launch site names begin with 'CCA' and the LIKE keyword with the wild card 'CCA%' to find string values beginning with 'CCA'.

Total Payload Mass

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

```
%sql select SUM(PAYLOAD_MASS__KG_) as total_payload_mass from SPACEXTABLE where Customer = 'NASA (CRS)';
```

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

total_payload_mass
45596

- Calculate the total payload mass carried by boosters from NASA (CRS) using the SUM() function as 45596 kg.

Average Payload Mass by F9 v1.1

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

```
%sql select AVG(PAYLOAD_MASS__KG_) as avg_payload_mass from SPACEXTABLE where Booster_Version = 'F9 v1.1';
```

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

avg_payload_mass
2928.4

- Calculate the average payload mass carried by booster version F9 v1.1 using the AVG() function as 2928.4 kg.

First Successful Ground Landing Date

List the date when the first successful landing outcome in ground pad was achieved.

Hint:Use min function

```
%sql select MIN(Date) as first_successful_landing_outcome from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)';
```

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

first_successful_landing_outcome
2015-12-22

- Using the MIN() function, determine that the date of the first successful landing outcome on ground pad is December 22, 2015.

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql select Booster_Version from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and  (PAYLOAD_MASS__KG_ < 6000) AND (PAYLOAD_MASS__KG_ > 4000);
```

Python

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

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

- Use the WHERE clause to filter for the names of boosters which have successfully landed on drone ship and the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000.
- The booster names are F9 FT B1022, F9 FT B1026, F9 FT B1021.2, and F9 FT B1031.2.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql select Mission_Outcome, count(*) from SPACEXTABLE group by Mission_Outcome;
```

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

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

- Using the COUNT() function, determine that the total number of successful and failure mission outcomes to be 100 successful and 1 failure.

Boosters Carried Maximum Payload

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

```
%sql select Booster_Version from SPACEXTABLE where PAYLOAD_MASS__KG_ = (select MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE) ORDER BY Booster_Version;
```

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

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

- Use a subquery in the WHERE clause and the MAX() function to determine the names of the booster which have carried the maximum payload mass.

2015 Launch Records

List the records which will display the month names, failure landing_outcomes in drone ship, booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%%sql
select substr(Date, 6, 2) as month_name, Landing_Outcome, Booster_Version, Launch_Site
from SPACEXTABLE
where substr(Date,0,5)='2015' and Landing_Outcome = 'Failure (drone ship)';
```

Python

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

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

- Use a combination of the SUBSTR() function and WHERE clause to determine the months January and April of 2015, failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015.

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

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
%sql select Landing_Outcome, count(*) from SPACEXTABLE where Date between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by count(*) desc;
```

Python

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

Landing_Outcome	count(*)
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

- Select landing outcomes and the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the dates 2010-06-04 and 2017-03-20, in descending order, using COUNT().
- Apply the GROUP BY clause to group by the landing outcomes and the ORDER BY clause to order by the landing outcomes in descending order.

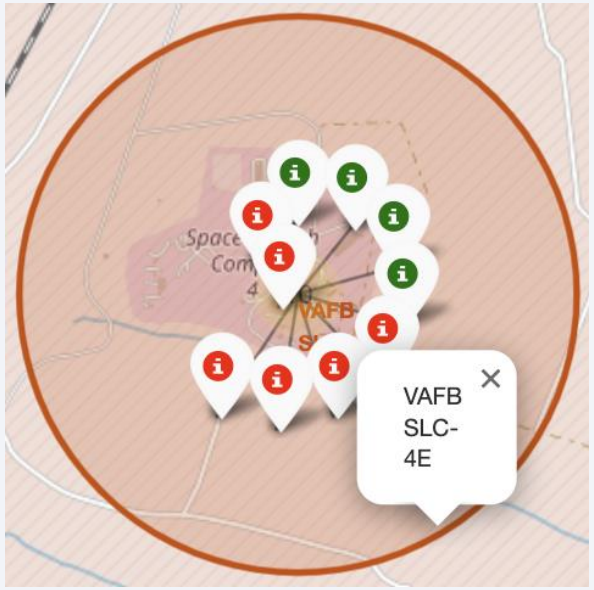

Section 3
Launch Sites
Proximities Analysis

All launch sites global map markers

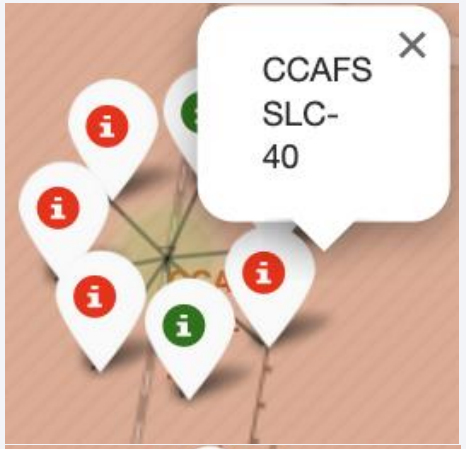
- The launch sites are in very close proximity to the United States coasts. They are indicated by the red markers on the map in California and Florida.

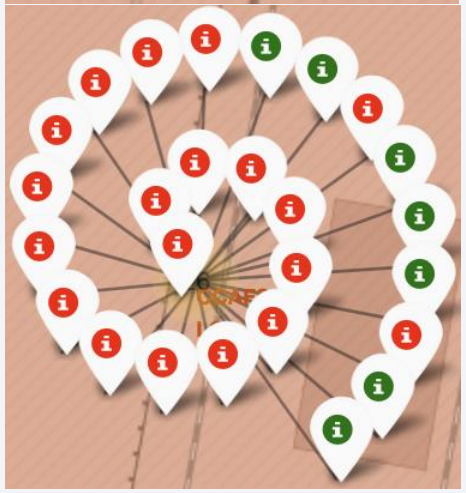
Colored Markers showing Launch Outcomes for each site

- Green markers indicate successful launches whereas red markers indicate failed launches.



California Launch Site: VAFB SLC-4E

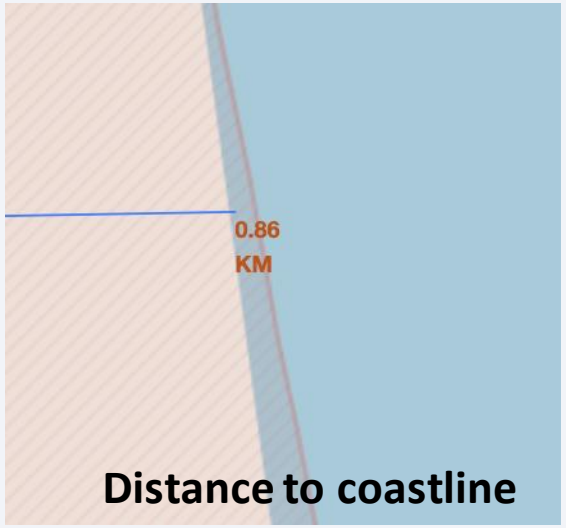


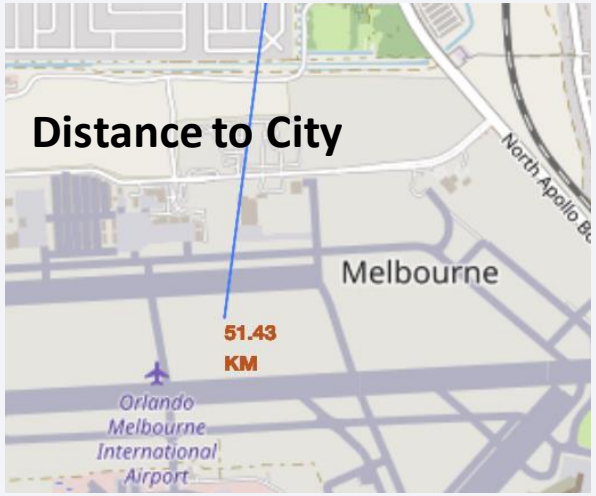


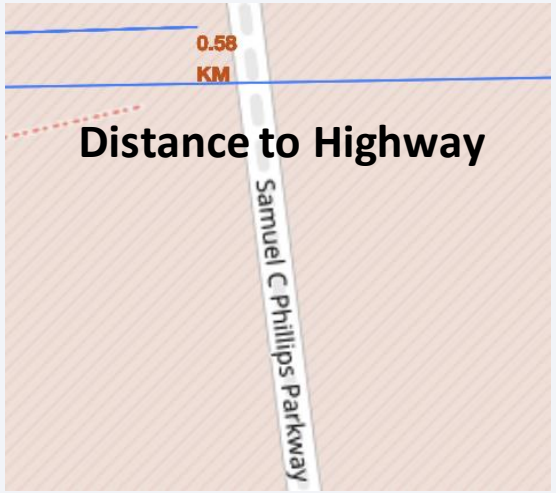
Florida Launch Sites: CCAFS SLC-40, KSC LC-39A, CCAFS LC-40

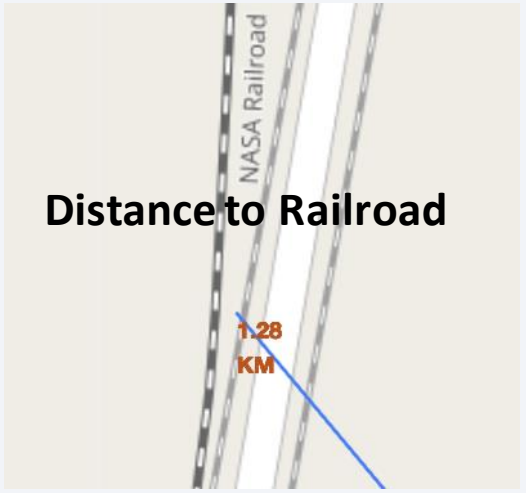
Launch Sites with distance lines to proximities

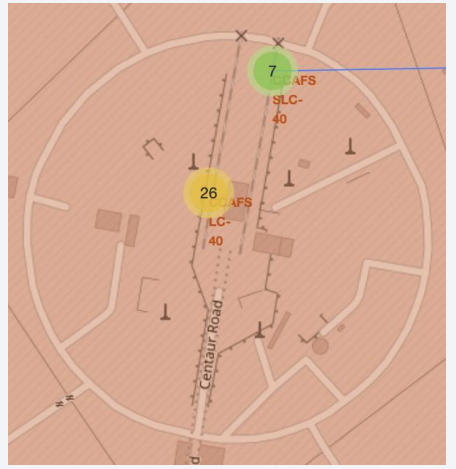
- Launch sites are in close proximity to the coastline, railways, and highways, but they keep certain distance away from cities.
- Close proximity to railways and highways makes it easier to transport heavy cargo.

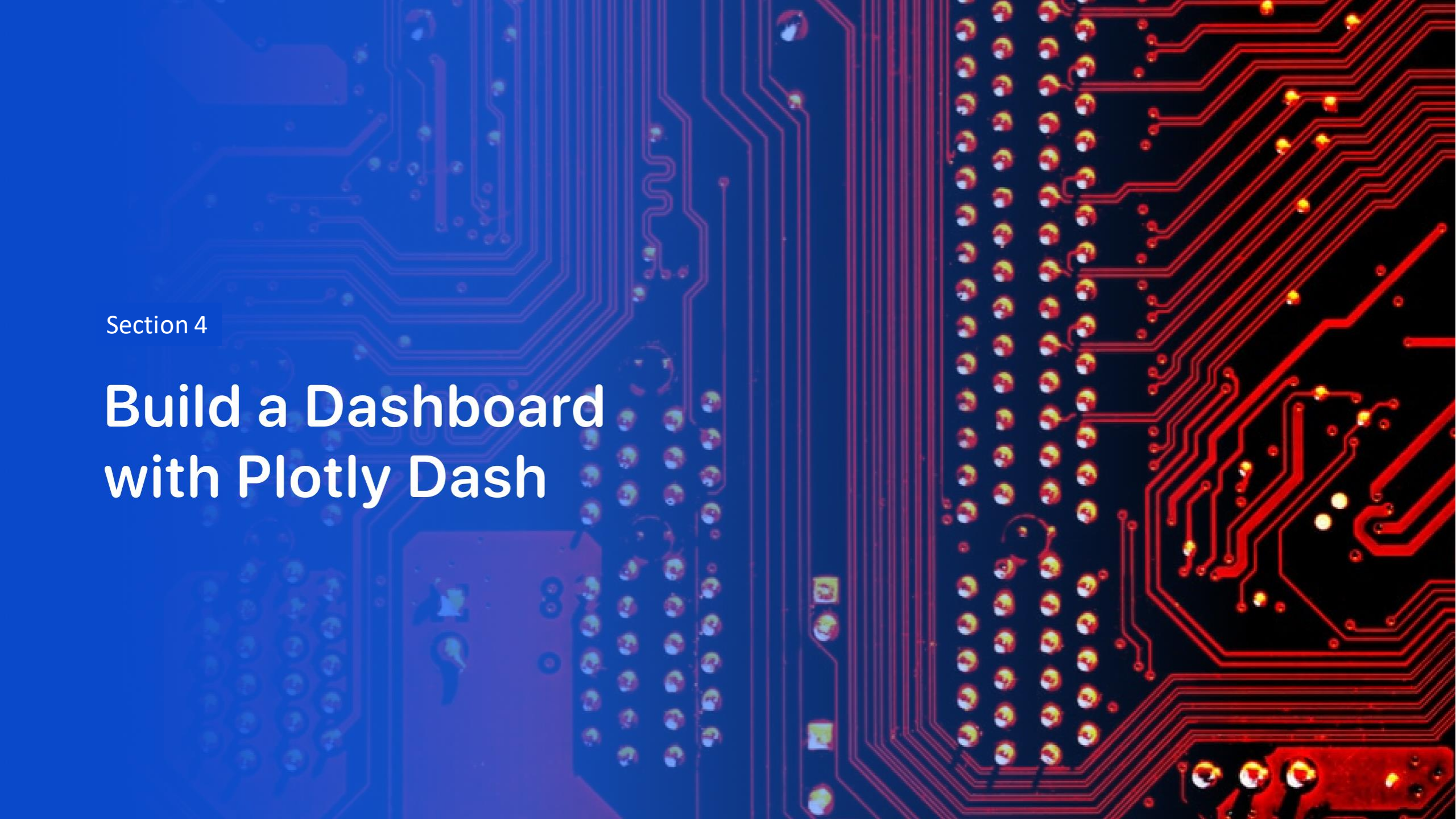




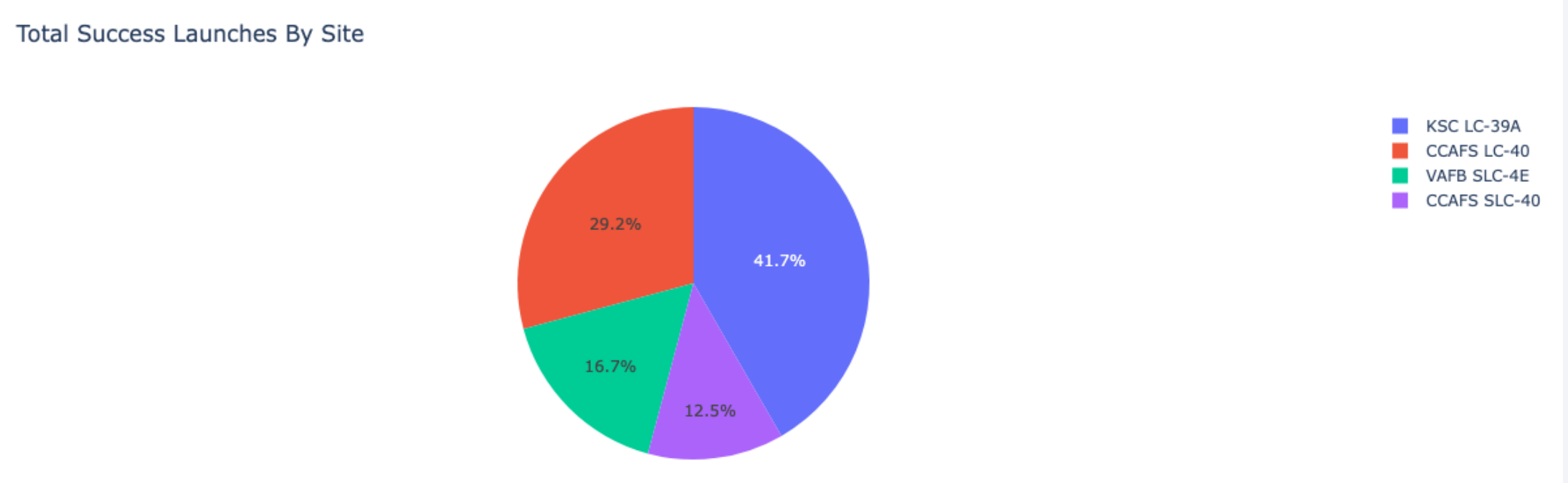






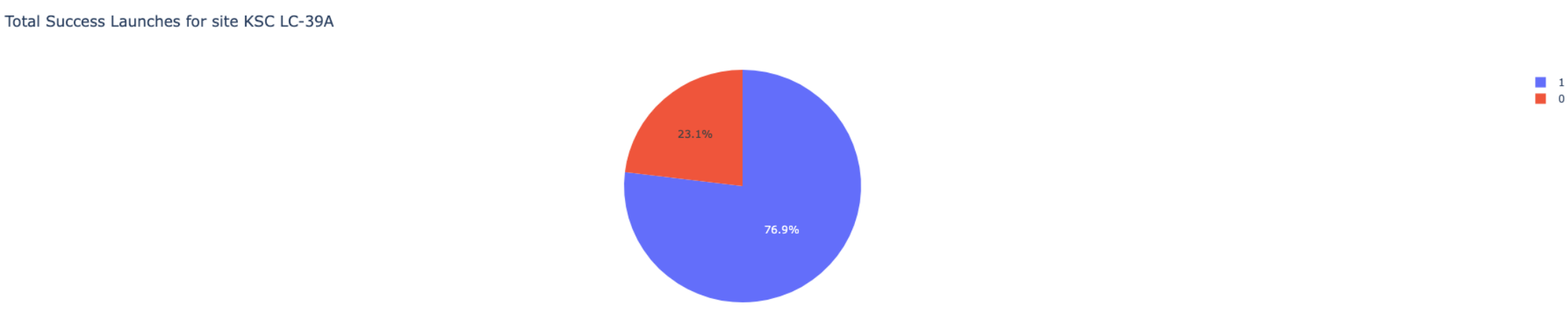

Section 4
Build a Dashboard with Plotly Dash

Launch Success Count for all sites



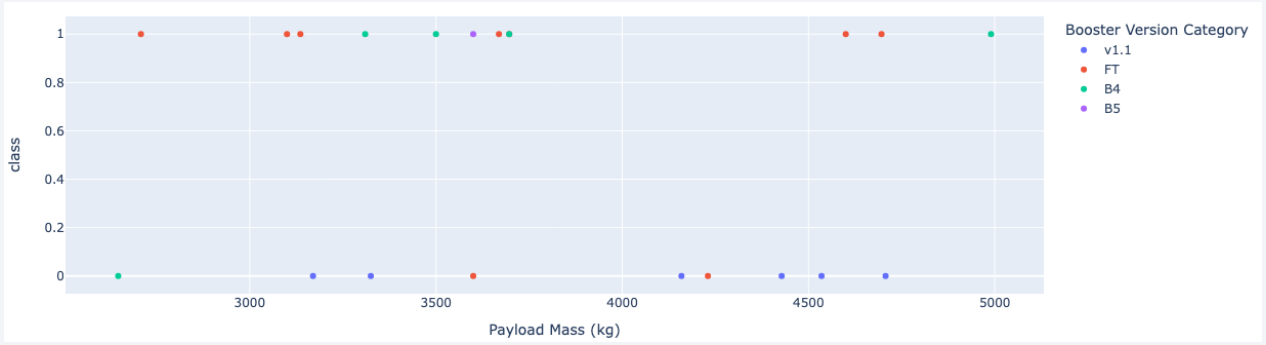
- The pie chart shows that KSC LC-39A had the highest launch success percentage out of all sites with a count of 10 successes.

Launch Site with Highest Launch Success Ratio



- The pie chart shows that KSC LC-39A, the launch site with the highest success rate, achieved a 76.9% success rate.

Payload vs. Launch Outcome Scatter Plot for all sites



Lower Weighted Payload 2500-5000 kg

Booster Versions: v1.1, FT, B4, B5

Higher Weighted Payload 7500-10000 kg

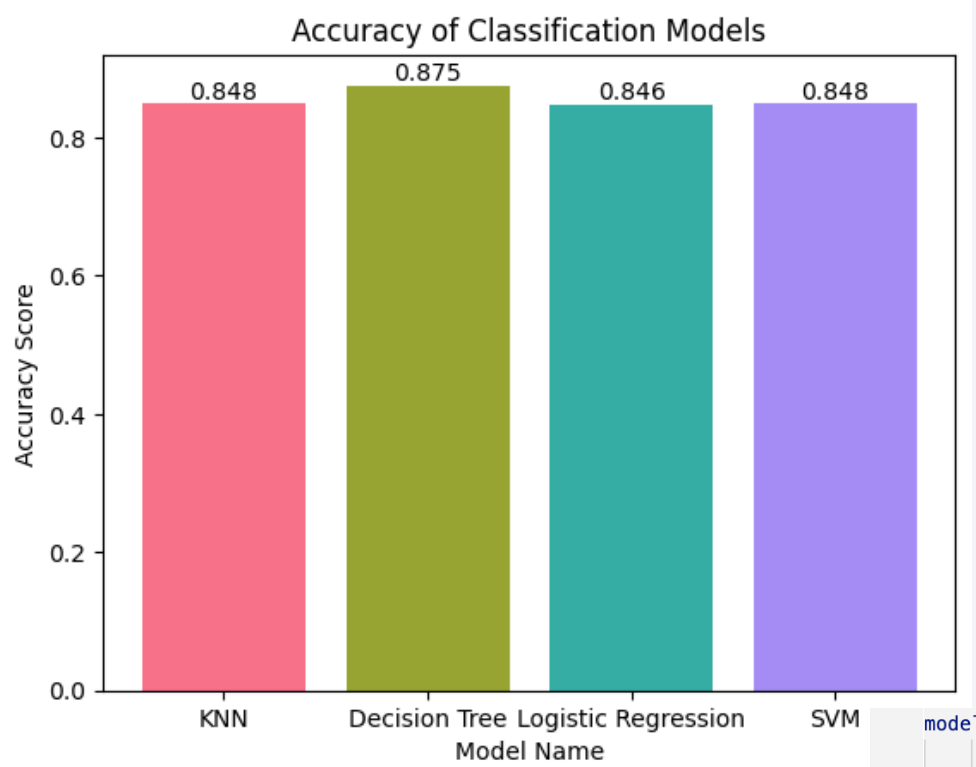
Booster Versions: FT, B4

- From the scatter plots, the payload mass range of 2500-5000 kg has the highest launch success rate whereas the payload range of 7500-10000 kg has the lowest launch success rate. The success rates for lower weighted payloads is higher than the success rates for heavier weighted payloads.

Section 5

Predictive Analysis (Classification)

Classification Accuracy



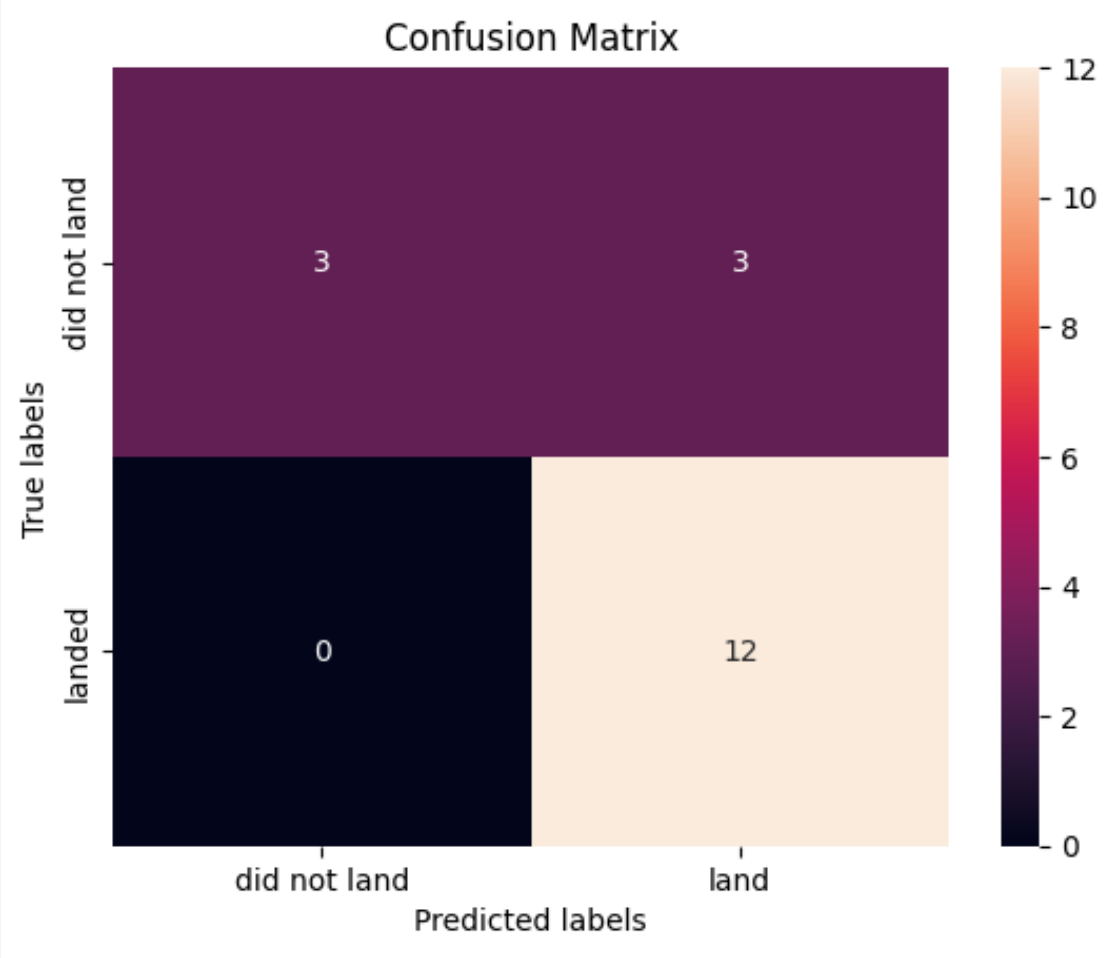
- The Decision Tree Classifier model performs the best with the highest accuracy score of 0.875.

```
model_accuracy_scores = {'KNN': knn_cv.best_score_,
                         'Decision Tree': tree_cv.best_score_,
                         'Logistic Regression': logreg_cv.best_score_,
                         'SVM': svm_cv.best_score_}
print(model_accuracy_scores)
```

```
{'KNN': 0.8482142857142858, 'Decision Tree': 0.875, 'Logistic Regression': 0.8464285714285713, 'SVM': 0.8482142857142856}
```

Confusion Matrix

- The Decision Tree Classifier confusion matrix indicates that it can distinguish between the different classes. The major problem is the false positives- e.g. when a rocket launch failed to land but was predicted to land successfully by the classifier.



Conclusions and Innovative Insights

- The larger the number of flights at a launch site, the higher the success rate at the launch site.
- Launch sites are in close proximity to the coastline, railways, and highways, but they keep certain distance away from cities.
- The launch success rate began to increase in 2013 until 2020, with a slight dip in 2018.
- For heavy payloads over 5000 kg, the success rates are higher for orbits PO, LEO, and ISS.
- KSC LC-39A had the highest launch success rate out of all sites with a total of 10 successful launches.
- Orbits ES-L1, GEO, HEO, and SSO have the highest launch success rates, followed by orbit VLEO.
- The Decision Tree Classifier model performs the best for this project.

Appendix

- GitHub Repository: Link
- Data Collection through API: Link
- Data Collection with Web scraping: Link
- Data Wrangling: Link
- Exploratory Data Analysis using SQL: Link
- Exploratory Data Analysis with Data Visualization: Link
- Interactive Visual Analytics with Folium: Link
- Interactive Dashboard with Plotly Dash: Code Link; Dashboard Link
- Machine Learning Prediction: Link

Thank you!