Winning Space Race with Data Science

Anurag Srivastava
September 3, 2024

Outline

Executive Summary

The project involved a comprehensive approach to predictive modeling aimed at forecasting the success of SpaceX Falcon 9 first stage landings. Various machine learning techniques were applied, including data collection, data wrangling, and exploratory data analysis. Data was gathered through RESTful APIs and web scraping, followed by in-depth analysis using Python libraries. Interactive maps were developed using Folium to analyze SpaceX launch records and geographical factors influencing launch success. The project also included the creation of interactive dashboards with Plotly Dash, enabling dynamic visualization of the data. The models were rigorously tested, validated, and fine-tuned through hyperparameter optimization to maximize prediction accuracy.

In the results, it is observed that the geographical locations for certain launch sites are favorable due to their proximity to railways, highways, and coastlines. Additionally, launches aimed at specific orbits, such as ES-L1, GEO, HEO, and SSO, have a higher success rate. Moreover, it was noted that the success of launches improves over time. The most accurate model achieved an 83% accuracy rate in predicting the Falcon 9 landing success. These insights provide valuable data for optimizing SpaceX's launch operations, potentially reducing costs and improving efficiency.

Introduction

As a data scientist at Space Y, a new rocket company aiming to compete with SpaceX, our mission is to analyze and predict the outcomes of rocket launches, specifically focusing on the Falcon 9 first stage landing success. SpaceX's ability to reuse the first stage of its rockets is a significant cost-saving advantage, allowing it to offer launch services at a fraction of the cost of its competitors. Understanding and predicting whether the Falcon 9 first stage will successfully land is crucial in estimating launch costs and formulating competitive strategies in the commercial space industry.

Key Questions to Address:

1. Can we accurately predict the likelihood of a Falcon 9 first stage landing success?
   - The goal is to develop a machine learning model that predicts whether the first stage will land successfully, using publicly available data and historical launch records.

2. How can these predictions be used to optimize launch costs and improve competitive positioning?
   - By determining the likelihood of a successful landing, Space Y can better estimate the costs of launches and make informed decisions to compete effectively with SpaceX.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describes how data sets were collected
- Perform data wrangling
  - Describes how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

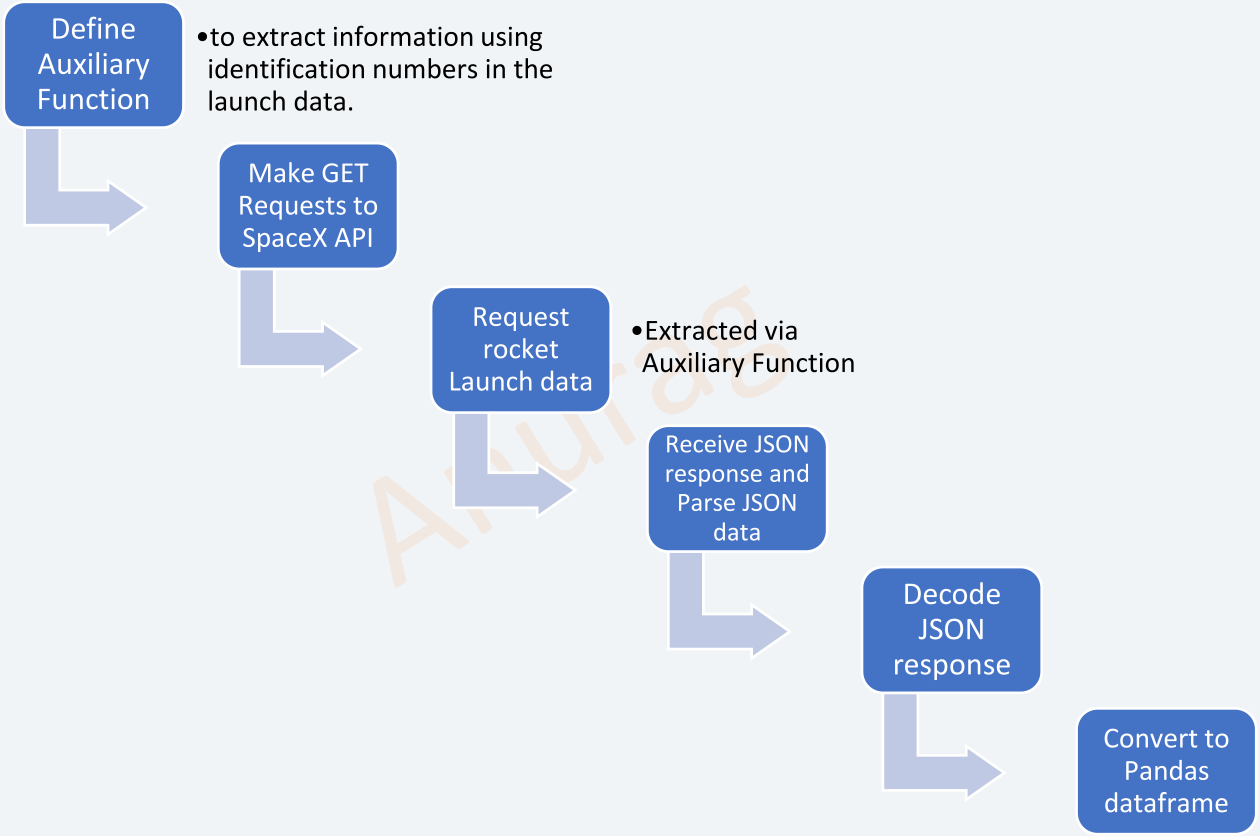
Using SpaceX REST API

- Collected SpaceX launch data using GET requests with helper functions.
- Parsed JSON response and converted it into a pandas dataframe

Using Web Scrapping

- Web scrapped Falcon-9 historical launch records from a Wikipedia page using BeautifulSoup and Request library
- Parsed the HTML table and converted it into a pandas dataframe

Data Collection – SpaceX API



Here is the GitHub URL of the SpaceX API calls notebook- https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(1)%20jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

Performed an HTTP GET method to request the Falcon9 Launch HTML page → Created a BeautifulSoup object from the HTML response → **Extracted all column/variable names from the HTML table header** → **Created a data frame by parsing the launch HTML tables**

Here is the GitHub URL of the web scrapping notebook - https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(2)%20jupyter-labs-webscraping.ipynb

Data Wrangling

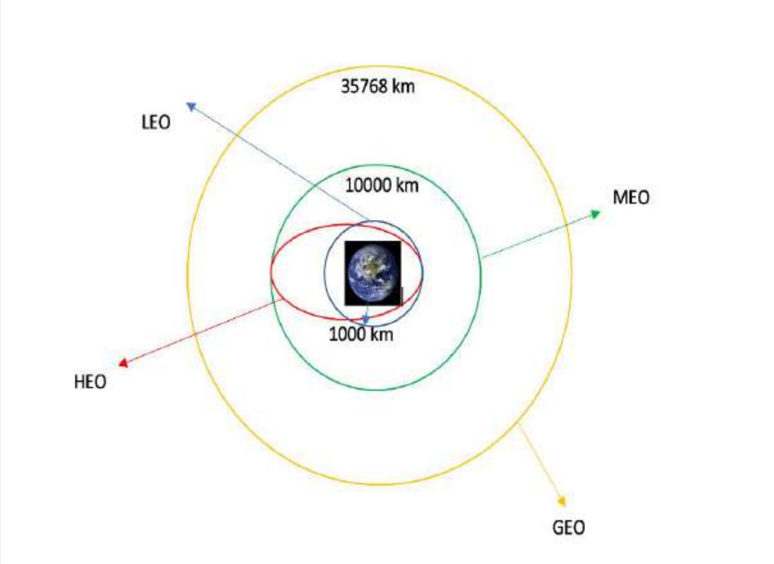
Missing data of Payload Mass column replaced with mean of column

Calculating the number of launches per site

Calculating the number and occurrence of each orbit

Calculating the number and occurrences of mission outcomes per orbit

Creating landing outcome labels from the Outcome column



some common orbit types

Here is the GitHub URL of the Data Wrangling notebook - https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(3)%20labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Used scatter chart to study relationship between various features like Flight Number, Launch Site, Payload Mass (kg) and Orbit Type
- Used bar chart to understand relationship between success rate and orbit type.
- Used line chart to visualize the launch success yearly trend.

Here is the GITHUB URL of the complete EDA with data visualization notebook - https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(4)%20edadataviz.ipynb

EDA with SQL

The following SQL queries were performed:

- Display names of unique launch sites and 5 records where launch site begins with 'CCA'.
- Calculate the total and average payload mass carried by boosters, specifically for NASA (CRS) launches and booster version F9 v1.1.
- List the date of the first successful landing on the ground pad and names of boosters with successful landings on drone ships carrying payloads between 4,000 and 6,000.
- Provide total counts of successful and failed missions, names of booster versions with max payload, failed landing outcomes on drone ships in 2015, and counts of landing outcomes between 2010-06-04 and 2017-03-20 (descending).

Here is the GITHUB URL of the complete EDA with SQL notebook - https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(5)%20jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

The following Map objects were created and added to a folium map:

- Added markers for the closest coastline point and distances to the closest city, railway, highway, etc.
- Used ‘folium.Circle’ to highlight launch sites with text labels.
- Implemented MousePosition to display coordinates when hovering over the map
- Used ‘folium.PolyLine’ to connect launch sites to coastlines, cities, railways, and highways to show distances.

These objects were added to the map to clearly visualize key geographical points and distances. Markers and circles highlight important locations like launch sites, coastlines, cities, railways, and highways. The MousePosition feature helps users easily find coordinates of points on the map, while polylines show distances between launch sites and other significant locations, aiding in spatial analysis.

Here is the GITHUB URL of the complete ‘Interactive map with Folium map’ Notebook - https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(6)%20lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

In the dashboard following elements were added –

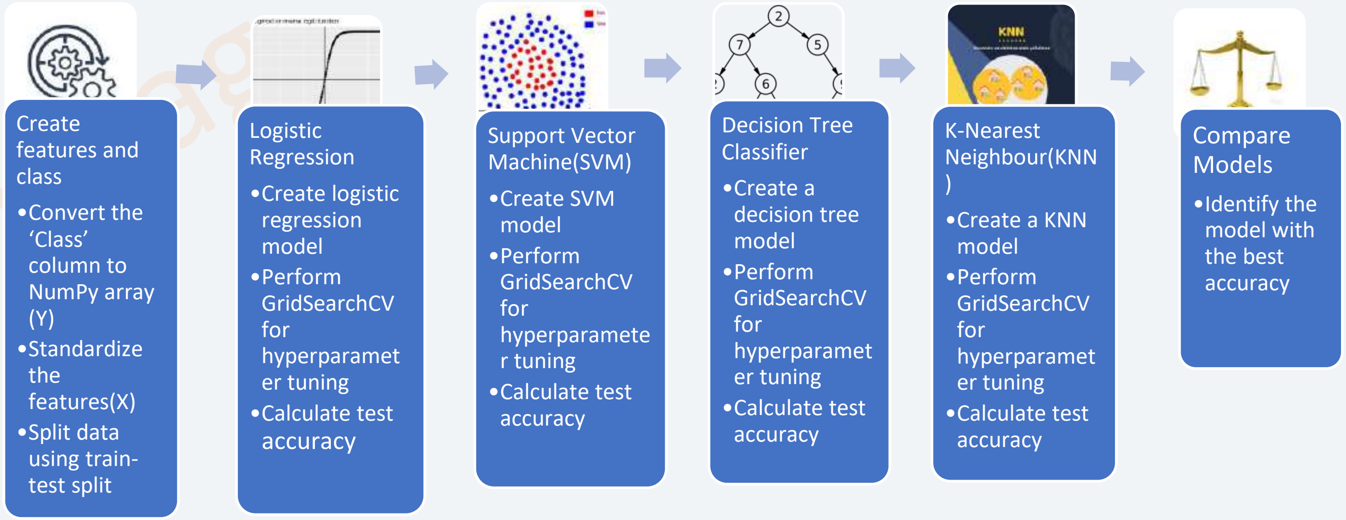
- **Launch Site Drop-down Input**: Added a drop-down component allowing users to select a specific launch site.
- **Success Pie Chart**: Implemented a pie chart that visualizes the success rate of launches based on the selected launch site from the drop-down.
- **Payload Range Slider**: Added a range slider for selecting the payload mass range, allowing dynamic filtering of data.
- **Success-Payload Scatter Plot**: Added a scatter plot that displays the relationship between payload mass and launch success, filtered by the selected launch site and payload range.

The dashboard allows users to filter launch data by site and payload through interactive elements like a drop-down and a range slider. Visuals such as pie charts and scatter plots provide insights into launch success rates and the impact of payload mass on outcomes

Here is the GITHUB URL of the complete 'Dashboard' python file- https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(7)%20Build%20an%20Interactive%20Dashboard%20with%20Ploty%20Dash%20-%20spacex_dash_app.py

Predictive Analysis (Classification)

To build the classification model, we standardized the data through preprocessing and split it into training and testing sets using train-test split method. Several models, including Logistic Regression, Support Vector Machines, Decision Tree Classifier, and K-Nearest Neighbors, were trained and evaluated. Grid Search was used to optimize hyperparameters, improving model performance. The best model was selected based on accuracy using the training data, and performance was further evaluated through a confusion matrix.



Here is the GITHUB URL of the complete predictive analysis lab- https://github.com/Anurag561-tech/Predictive-Modeling-for-Falcon-9-First-Stage-Landing-Success-/blob/main/(8)%20SpaceX_Machine%20Learning%20Prediction.ipynb

Results

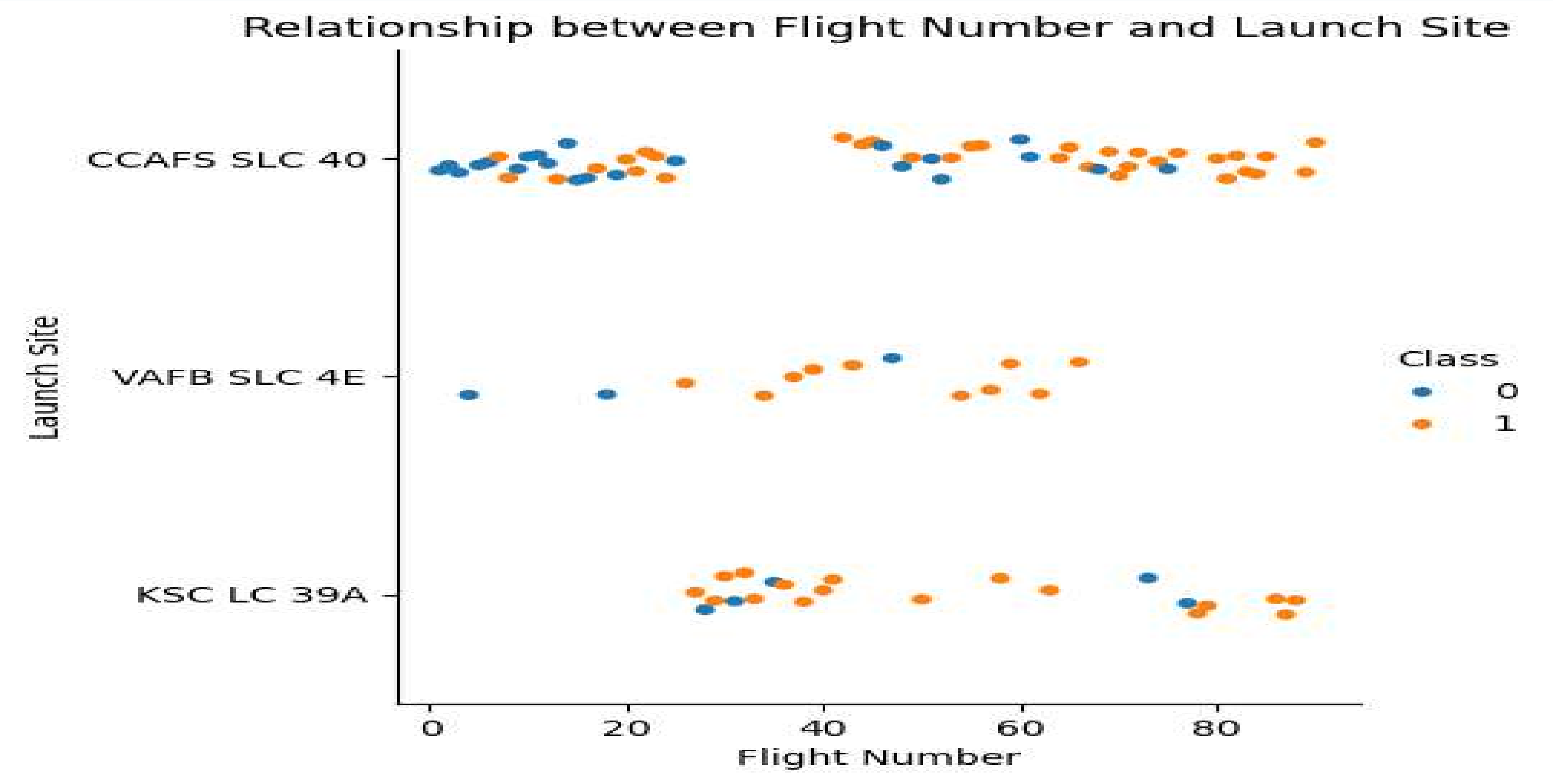
We will review the following results in the report:

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

Section 2

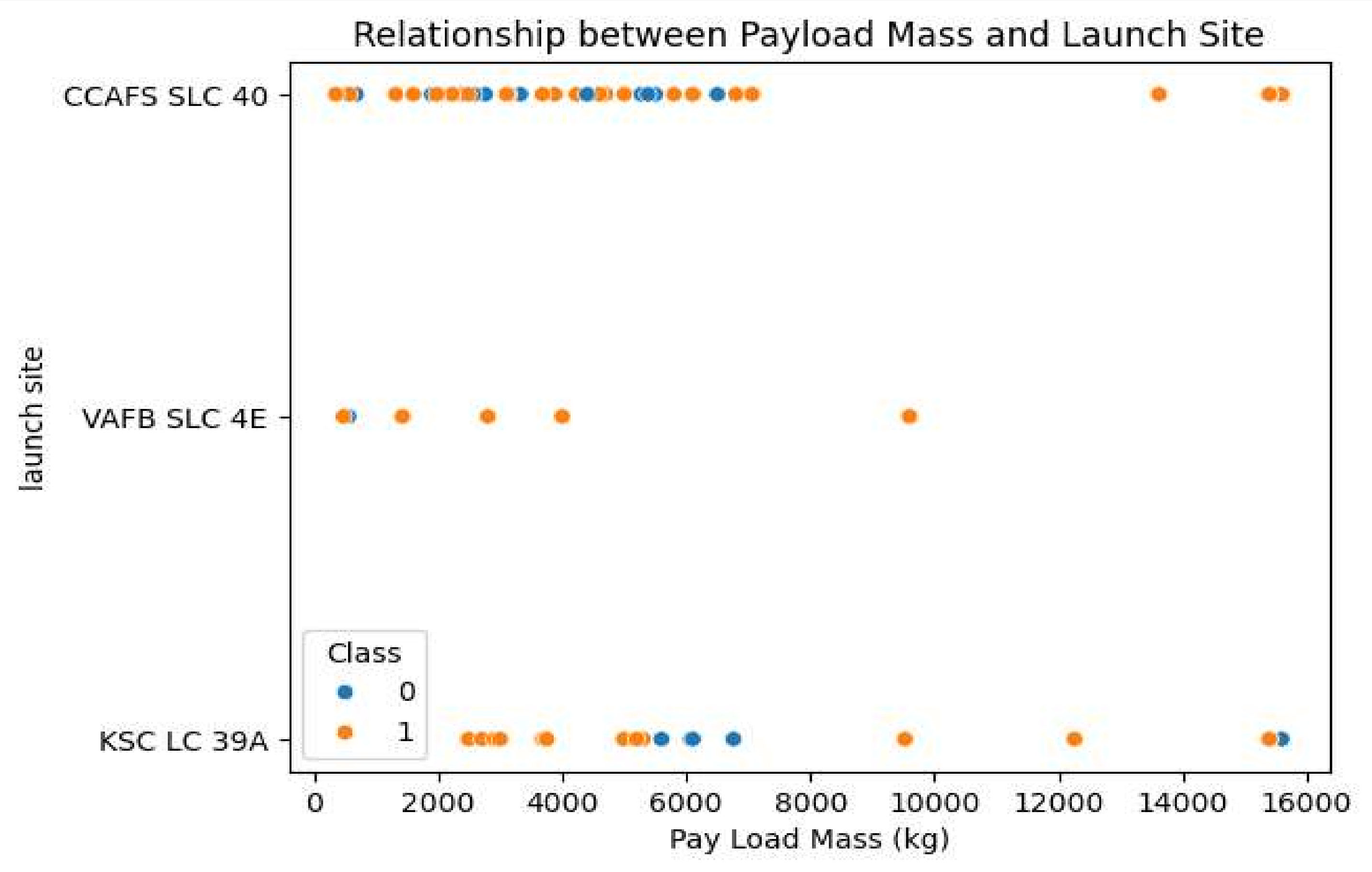
Insights drawn from EDA

Flight Number vs. Launch Site



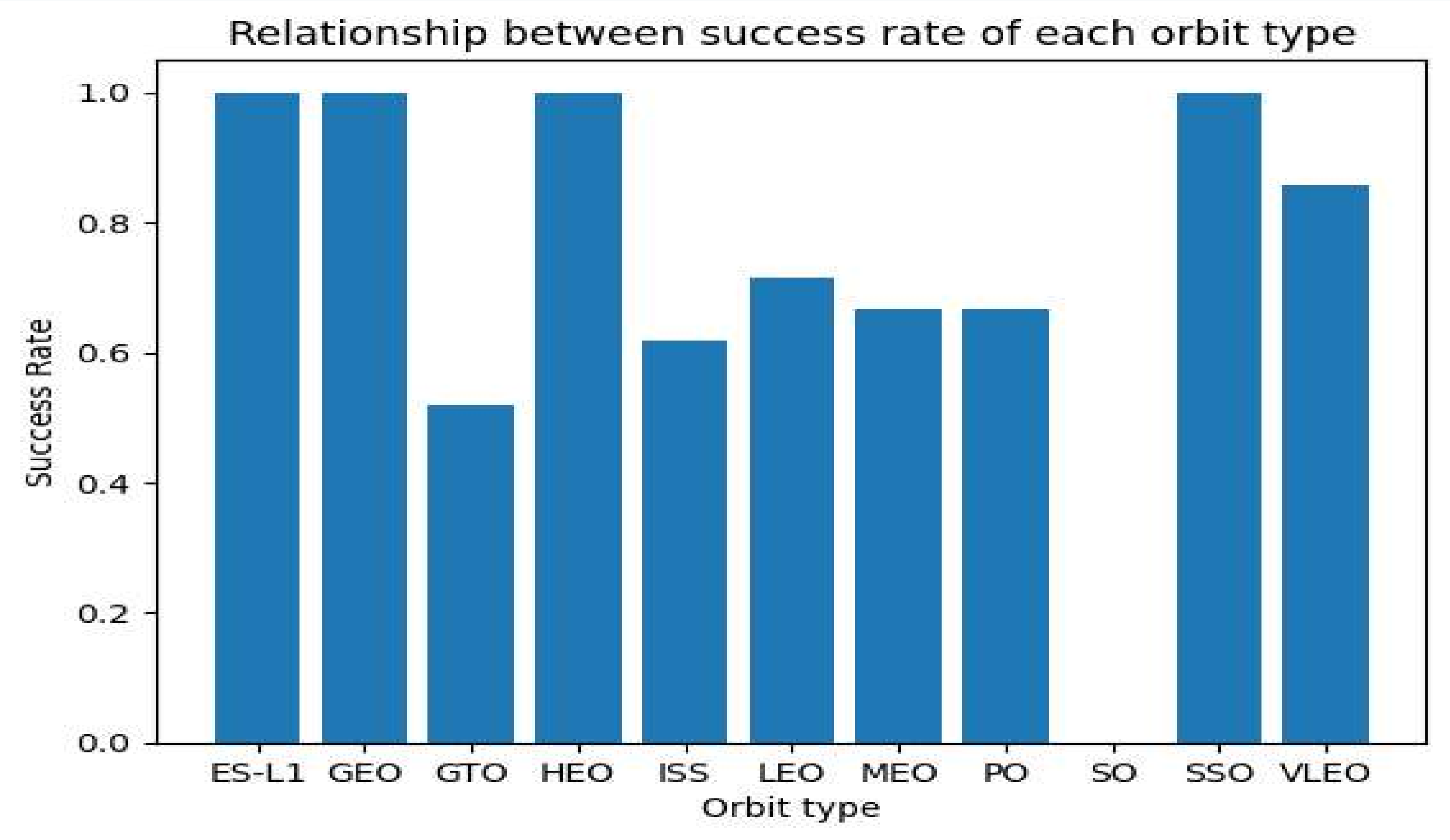
From the scatter plot it is visible that as the flight number for each launch site increases success rate increases

Payload vs. Launch Site



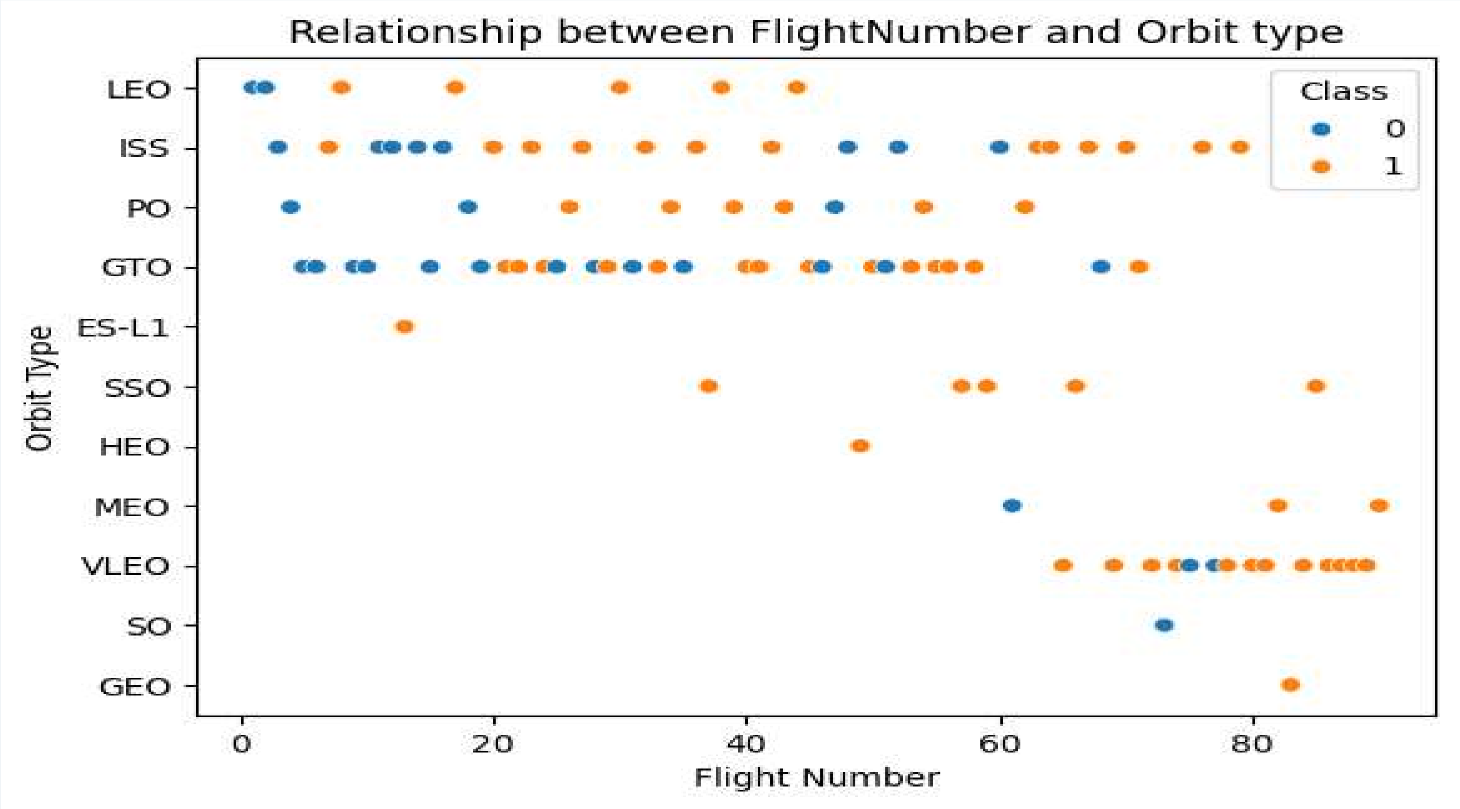
It is observed that in the Payload Mass vs. Launch Site scatter plot, there are no rockets launched with a heavy payload mass (greater than 10,000) from the VAFB-SLC launch site.

Success Rate vs. Orbit Type



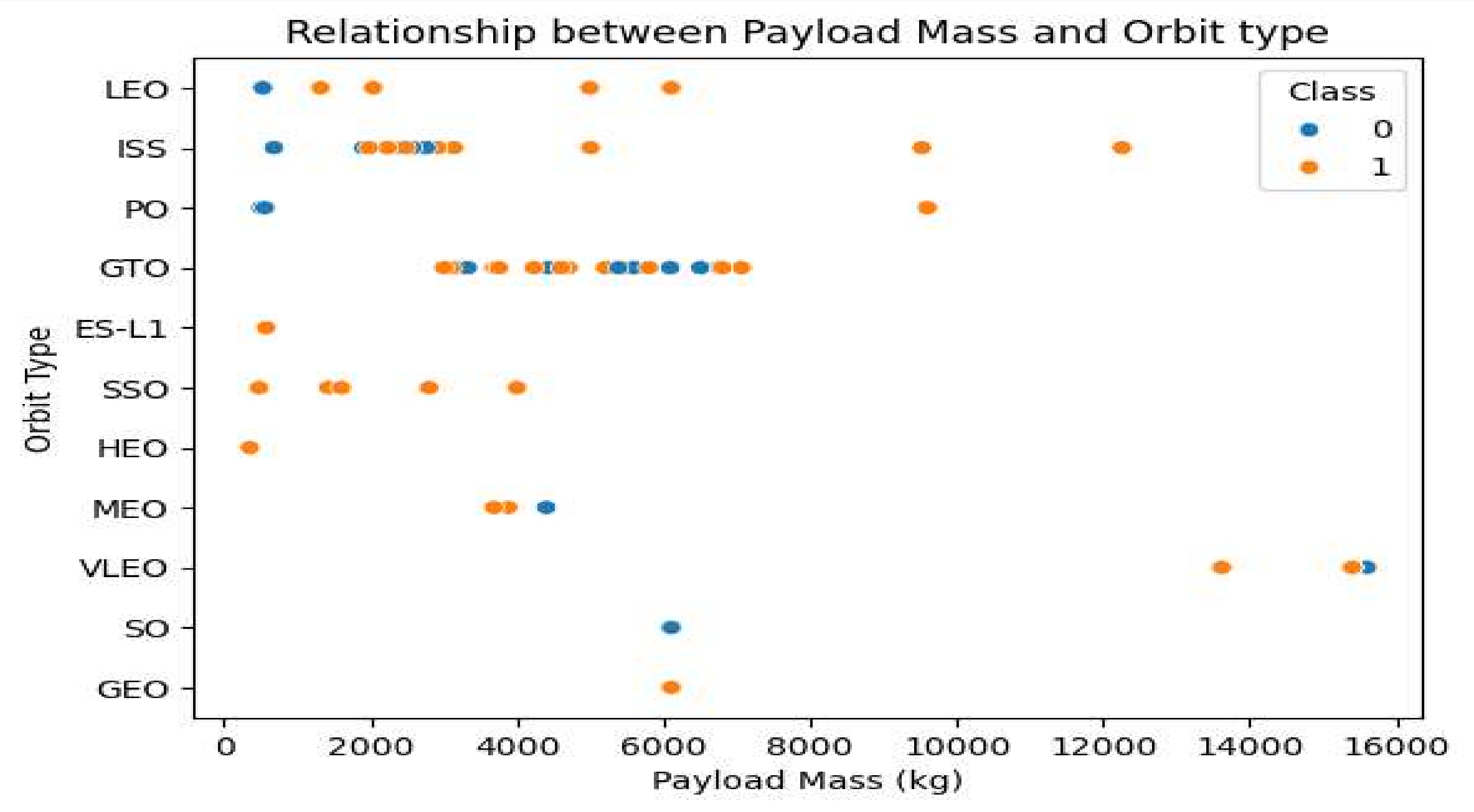
From this bar chart, it is evident that the ES_L1, GEO, HEO, and SSO orbits collectively have the highest success rates.

Flight Number vs. Orbit Type



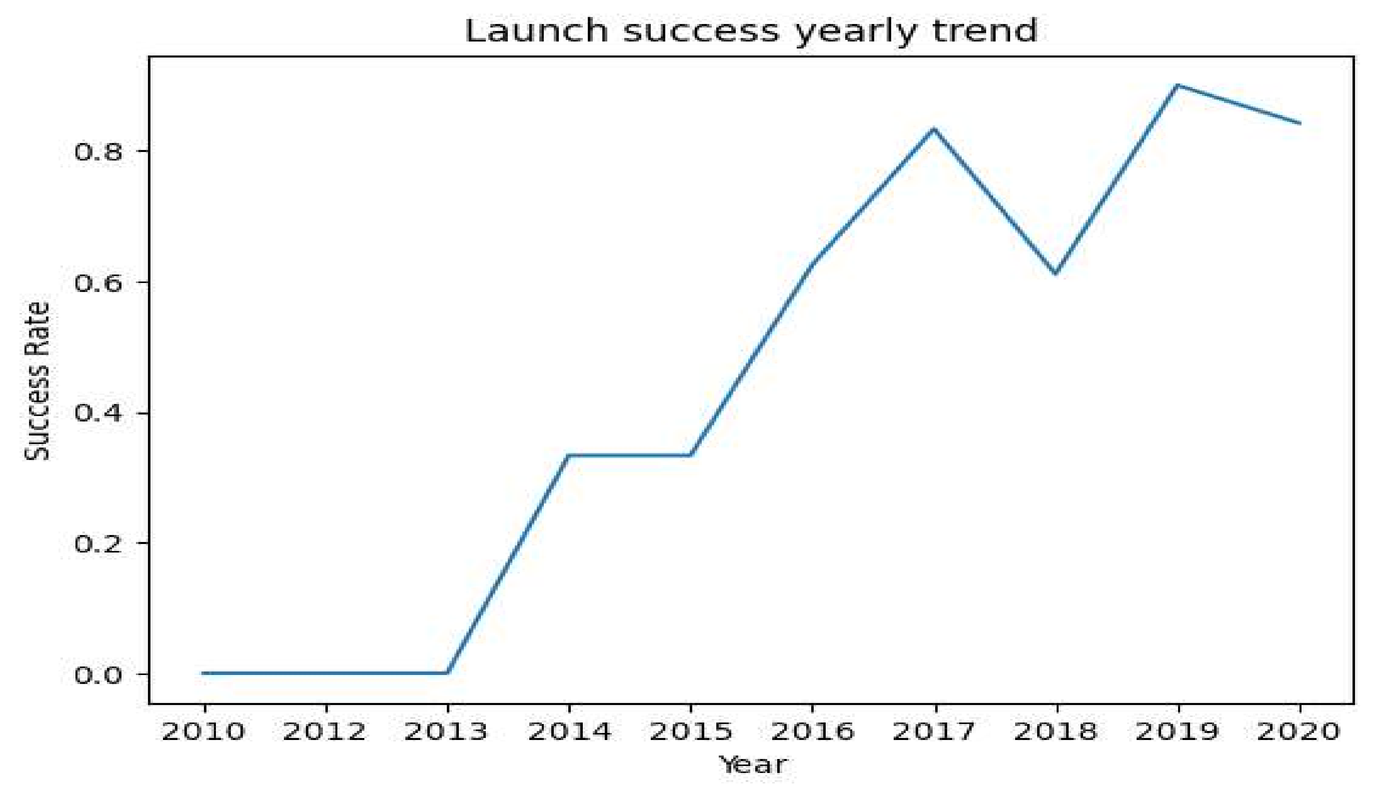
It is observed that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

Payload vs. Orbit Type



With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

Launch Success Yearly Trend



It is observed that the success rate since 2013 kept increasing till 2020.

All Launch Site Names

```
%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

Used ‘SELECT DISTINCT’ statement to return only the unique launch sites from the ‘LAUNCH_SITE’ column of the SPACEXTBL table

Launch Site Names Begin with 'CCA'

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

```
%sql select distinct launch_site from SPACEXTABLE where launch_site like '%CCA%';
```

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

Launch_Site
CCAFS LC-40
CCAFS SLC-40

Used ‘LIKE’ command with ‘%’ wildcard in ‘WHERE’ clause to select and display a table of all records where launch sites have the string ‘CCA’

Total Payload Mass

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

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

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

sum(PAYLOAD_MASS__KG_)
45596

Used the ‘SUM()’ function to return and display the total sum of ‘PAYLOAD_MASS__KG_’ column for Customer ‘NASA (CRS)’

Average Payload Mass by F9 v1.1

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

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

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

avg(PAYLOAD_MASS__KG_)
2928.4

Used the ‘AVG()’ function to return and display the average payload mass carried by booster version F9 v1.1

First Successful Ground Landing Date

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

✓ 0.0s

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

min(Date)
2015-12-22

Used the 'MIN()' function to return and display the first (oldest) date when first successful landing outcome on ground pad 'Success (ground pad)' happened.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

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

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

Used ‘Select’ statement to return and list the names of boosters with payloads between 4000-6000 and landing outcome of ‘Success (drone ship)’.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

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

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

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

Used the ‘COUNT()’ together with the ‘GROUP BY’ statement to return total number of missions outcomes

Boosters Carried Maximum Payload

```
%sql select Booster_Version from SPACEXTABLE where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTABLE);
```

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

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

Using a Subquery to return and pass the Max payload and used it list all the boosters that have carried the Max payload of 15600kgs

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 we will 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, Landing_Outcome, Booster_Version, Launch_Site from SPACEXTABLE \
where [Landing_Outcome] = 'Failure (drone ship)' and substr(Date,0,5)='2015';
```

✓ 0.0s

* sqlite:///my_data1.db
Done.

month	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

Used the ‘substr()’ in the select statement to get the month and year from the date column where substr(Date,0,5)='2015' for year and Landing_outcome was ‘Failure (drone ship)’ and return the records matching the filter.

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

```
%sql SELECT [Landing_Outcome], count(*) as count_outcomes FROM SPACEXTABLE \
WHERE DATE between '2010-06-14' and '2017-03-20' group by [Landing_Outcome] order by count_outcomes DESC;
```

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

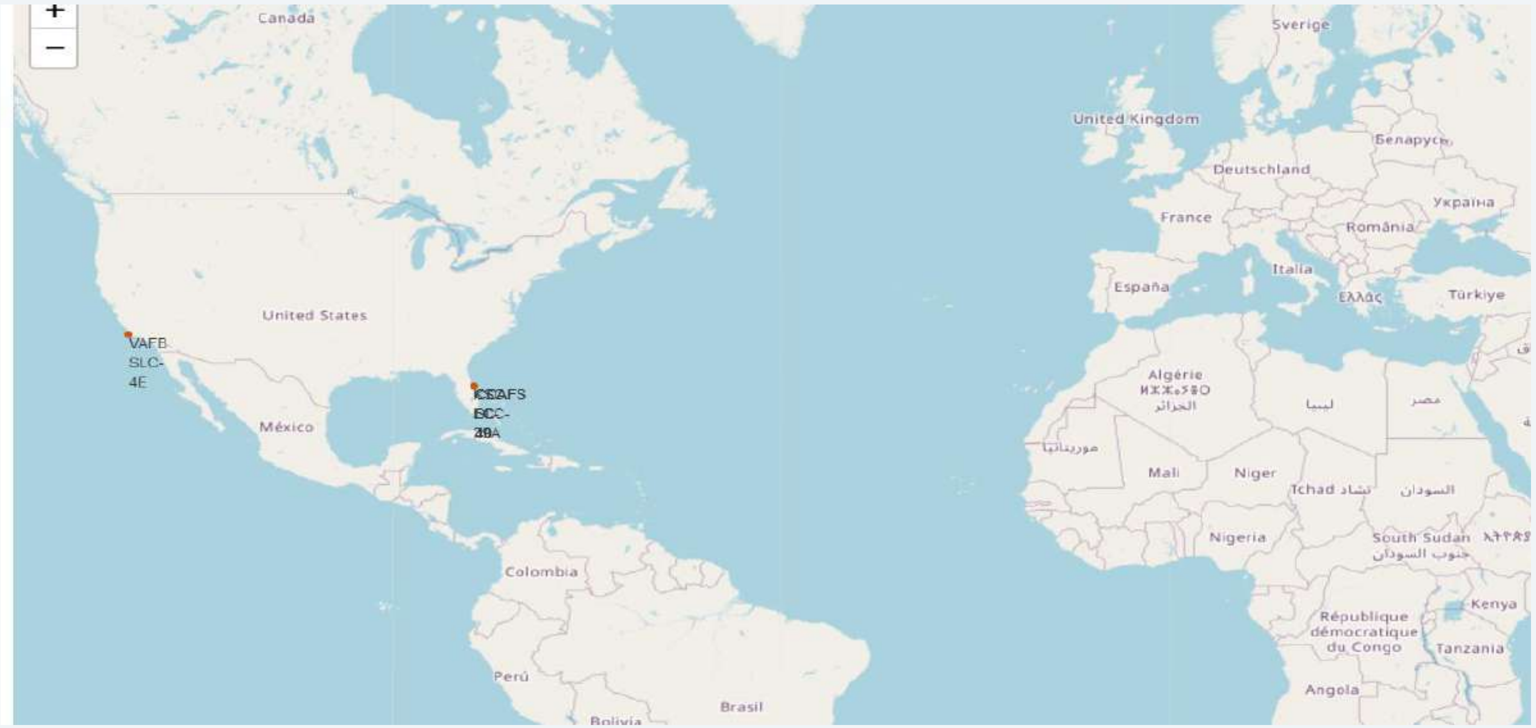
Landing_Outcome	count_outcomes
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Used ‘COUNT’, ‘ORDER’ and ‘WHERE’ and ‘and’ clause to filter out the landing outcomes between ‘2010-06-14’ and ‘2017-03-20’ in descending order and rank them accordingly.

Section 3

Launch Sites Proximities Analysis

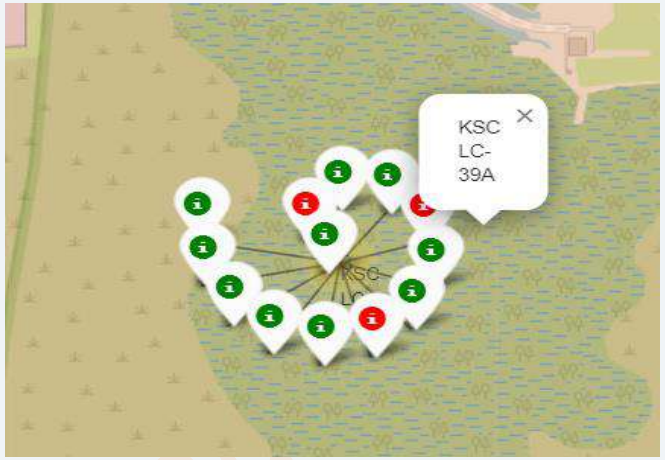
Markers of all launch sites on global map

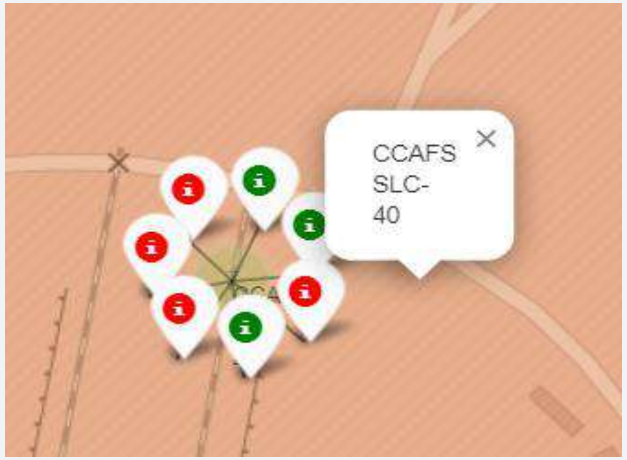


All launch sites are located near the Equator, to the south of the US, and are situated very close to the coast.

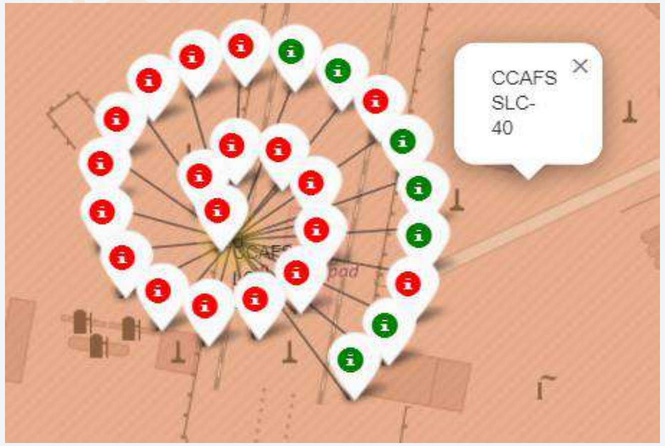
Launch outcomes for each site on the map With Color Markers

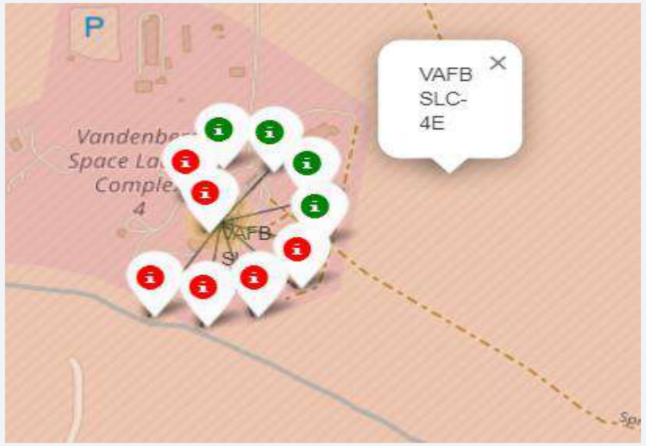
In the Eastern coast (Florida) Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-40



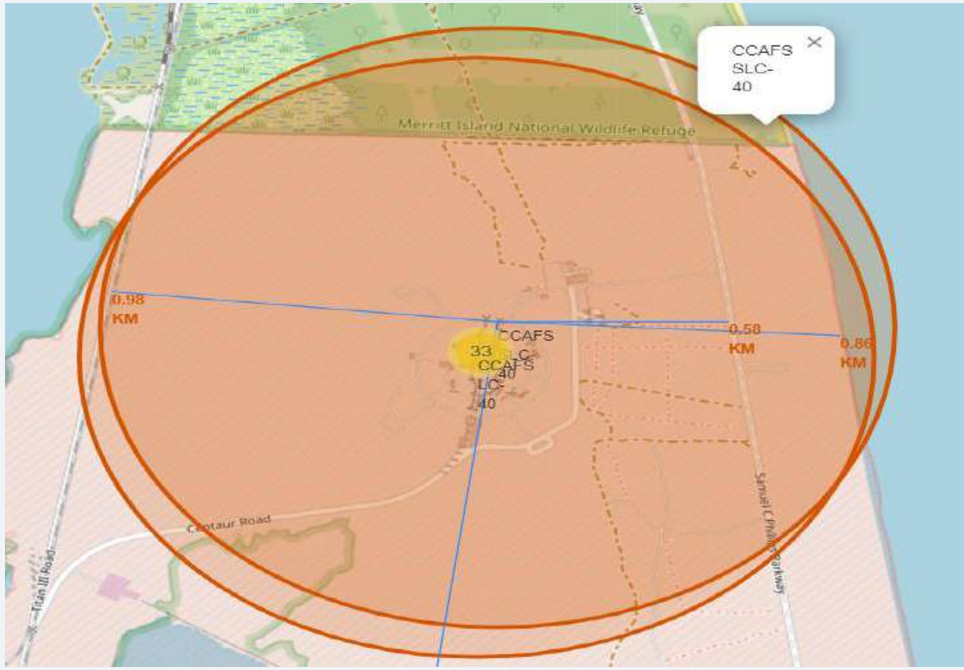


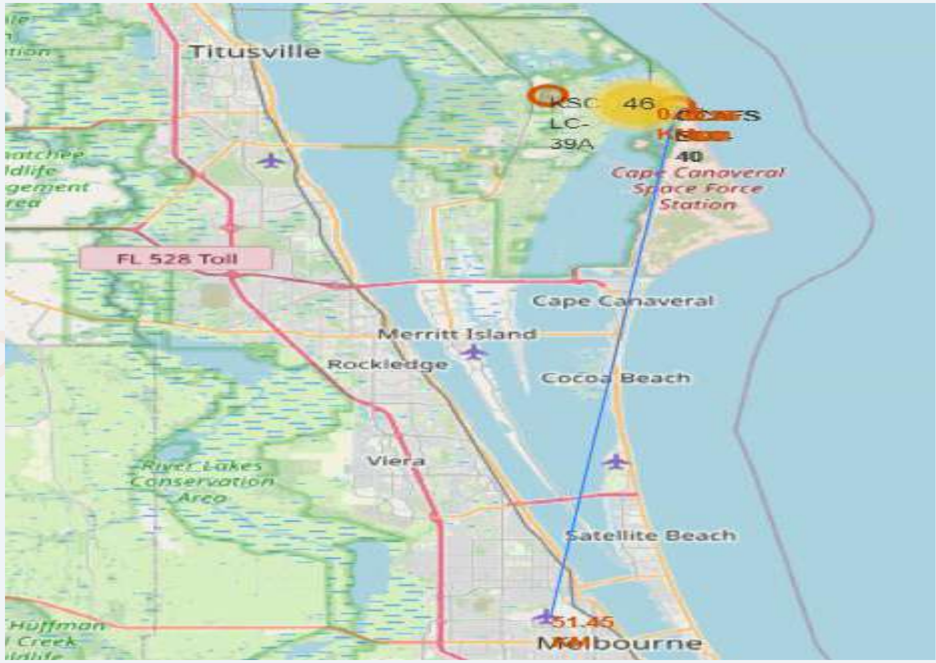
(Green marks for successful launches and red for unsuccessful launches)





Distances between CCAFS SLC-40 launch site to its proximities



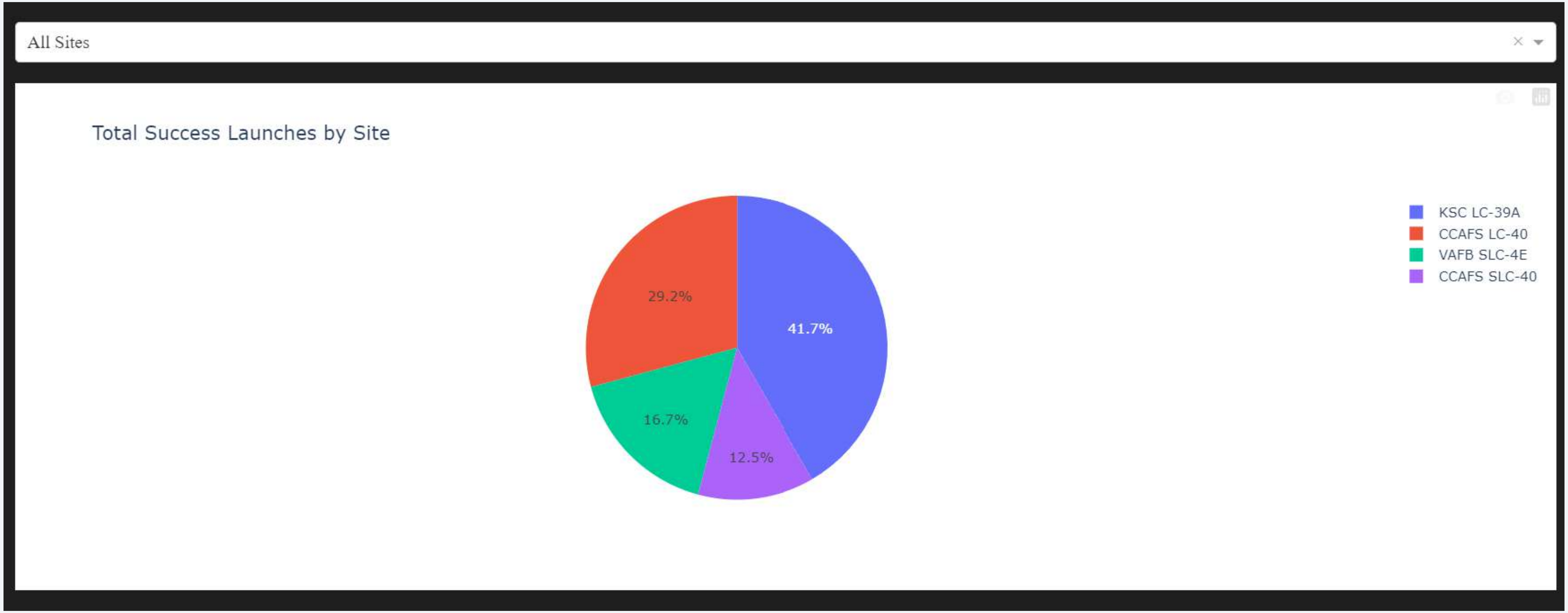


- Railways and Highways are in close to minimize logistic cost.
- Launch Sites are far from cities and close to coastline because crew has option to abort launch and attempt water landing, also to minimize people and property at risk from falling debris

Section 4

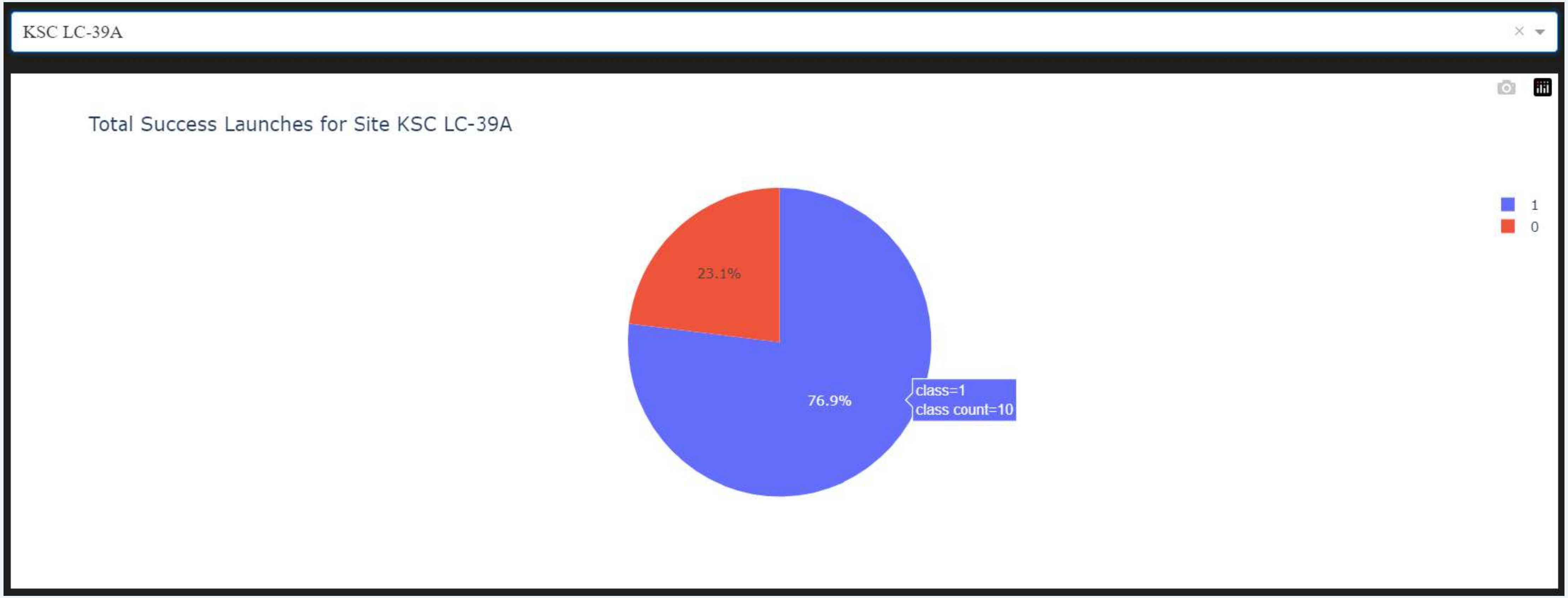
Build a Dashboard with Plotly Dash

Pie-Chart for launch success count for all sites



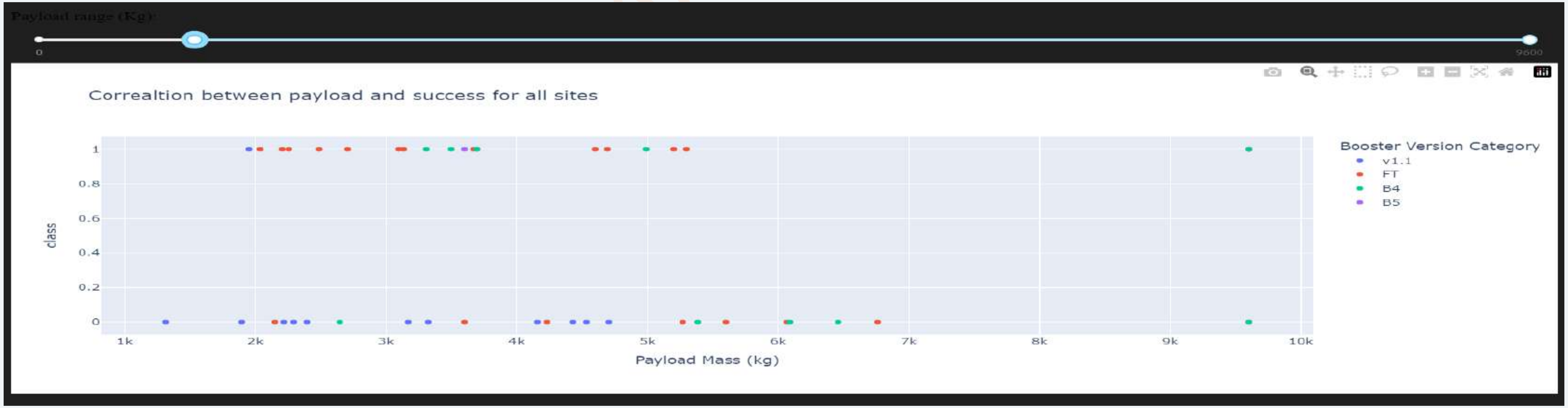
Launch site KSC LC-39A has the highest launch success rate at 41.7% while CCAFS SLC-40 has lowest success rate of 13%

Pie chart for the launch site with highest launch success ratio



It is observed that launch site KSC LC_39A have highest launch success ratio of nearly 3:4

Payload vs. Launch Outcome scatter plot for all sites



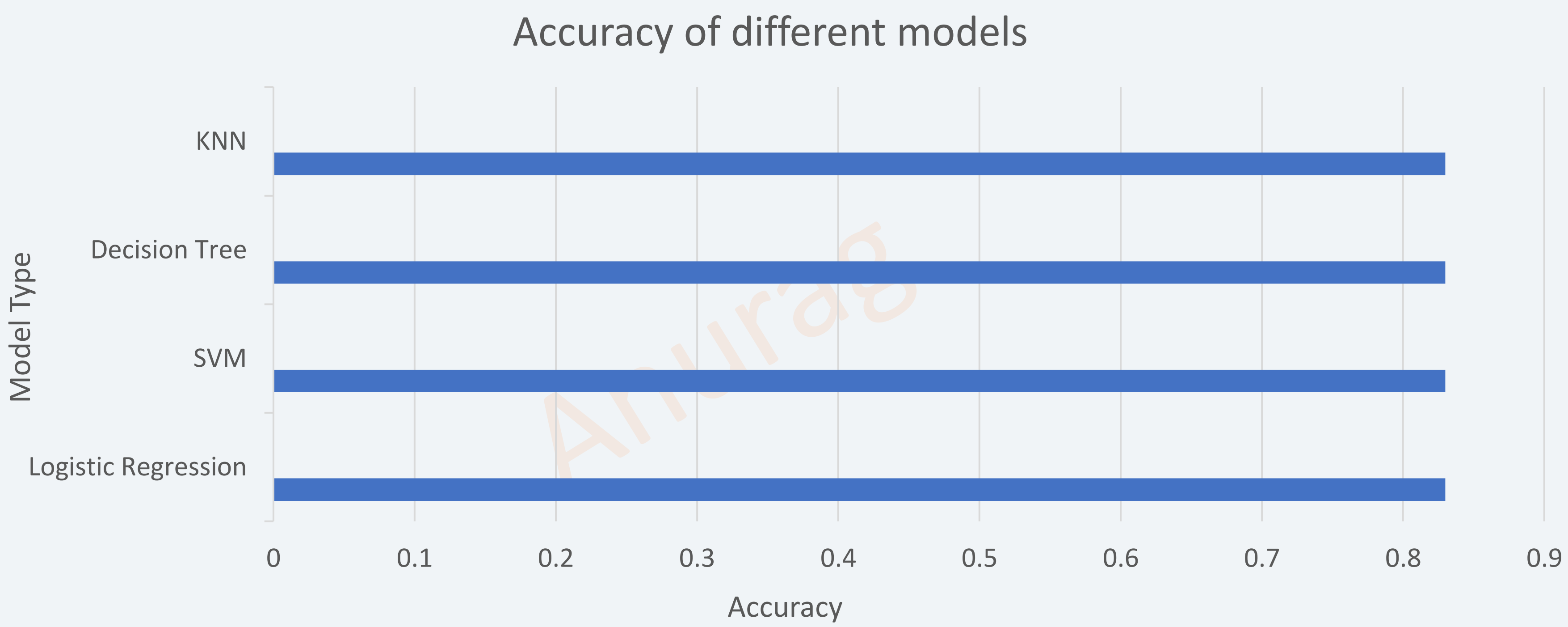
Payload vs. Launch Outcome scatter plot for all sites(Cont…)

- Payload between 1900 and 5300 (kg) and highest success rate while payload between 6000 and 9500(kg) have lowest success rate as evident in the figures.
- FT Booster version has the highest launch success rate.

Section 5

Predictive Analysis (Classification)

Classification Accuracy



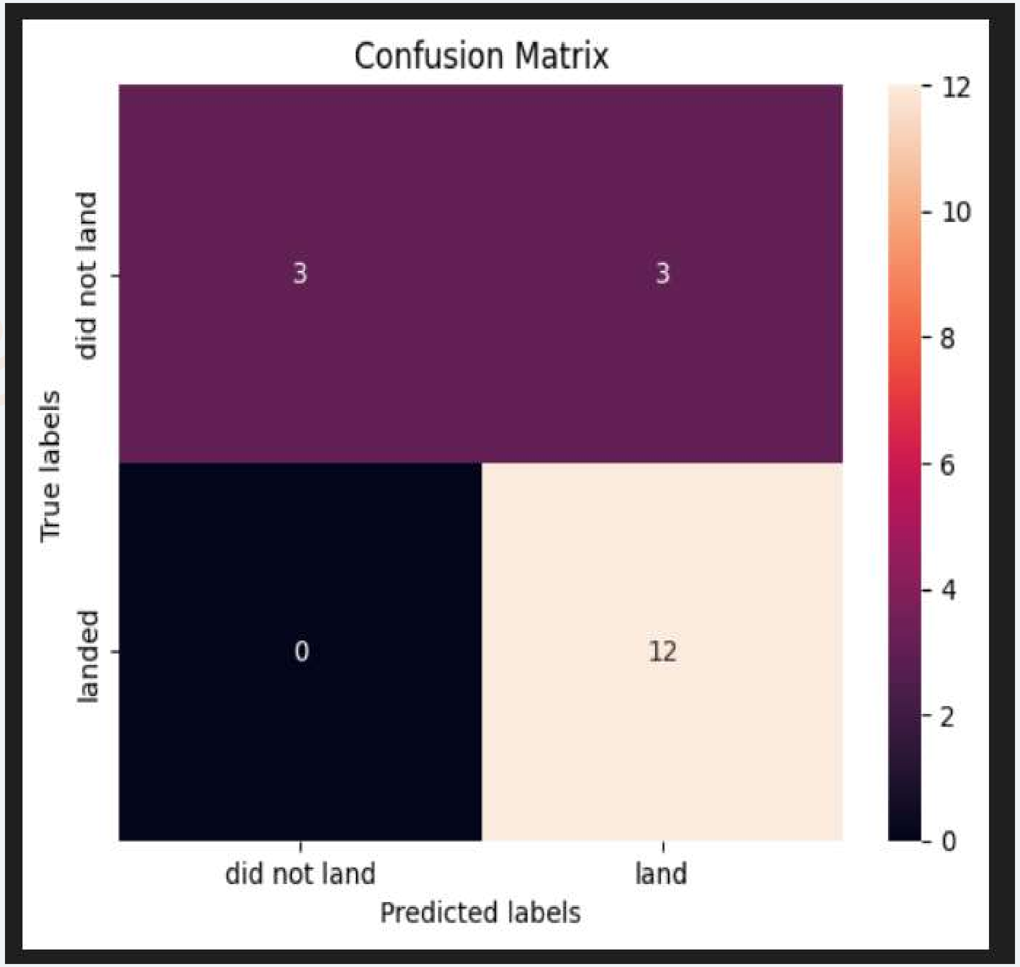
All the methods perform equally on the test data: i.e. They all have the same accuracy of 83.33% on the test Data

Confusion Matrix

As all the models have same accuracy, therefore they have same confusion matrix too.

Examining the confusion matrix, we see that models can distinguish between the different classes. We see that the problem is false positives.
True Positive - 12 (True label is landed, Predicted label is also landed)
False Positive - 3 (True label is not landed, Predicted label is landed)

•**Precision** = TP / (TP + FP) = 12 / 15 = 0.8
•**Recall** = TP / (TP + FN) = 12 / 12 = 1
•**F1 Score** = 2 * (Precision * Recall) / (Precision + Recall) = 2 * (0.8 * 1) / (0.8 + 1) = 0.89
•**Accuracy** = (TP + TN) / (TP + TN + FP + FN) = 0.833



Conclusions

- Overall all model performance similar on the test set.
- Most launch sites are located near the equator to take advantage of the Earth's rotational speed, reducing fuel and booster costs.
- All launch sites are situated close to the coast, railways and highways to reduce logistics cost.
- The success rate of launches has increased over time.
- KSC LC-39A launch site has the highest success rate among launch sites, achieving a 100% success rate for launches with payloads less than 5,500 kg.
- Across all launch sites, a higher payload mass (kg) is associated with a higher success rate.
- Orbits such as ES-L1, GEO, HEO, and SSO have a 100% success rate.

AS future possibility expanding the dataset may enhance the predictive analytics results and determine whether the findings can be generalized to a broader dataset. These results can help Space Y to better estimate the costs of launches and make informed decisions to compete effectively with SpaceX.

Appendix

- To request rocket launch data from SpaceX API following URL was used : https://api.spacexdata.com/v4/launches/past

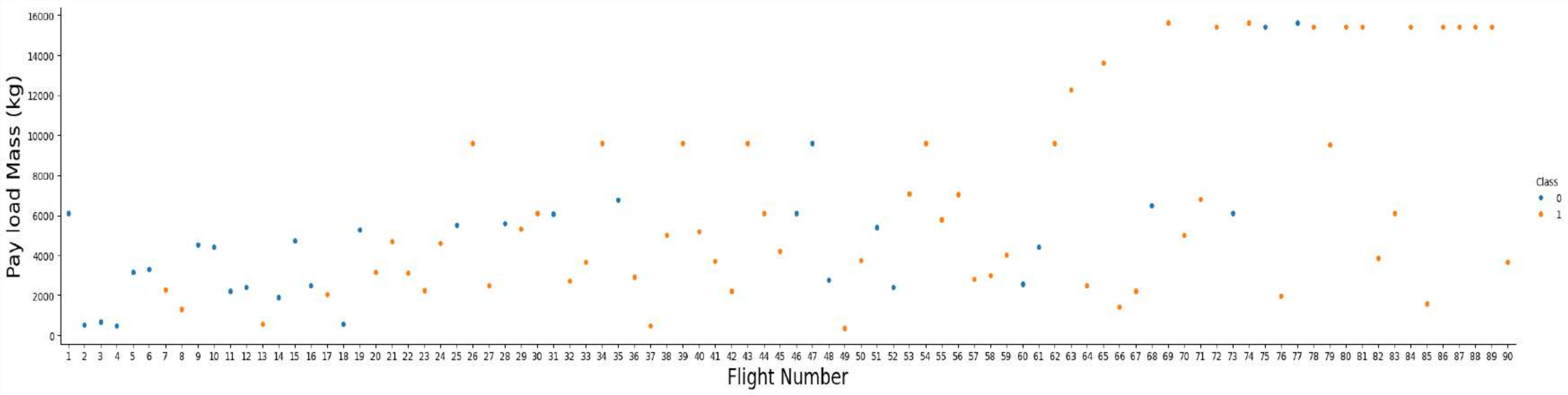
- To make the requested JSON results more consistent, following static response object was used: https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json

- The Wikipedia site used for web scraping SPACEX data is: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

- The dataset used for marking all launch sites on map in folium: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Appendix (Cont…)

- The following scatter plot was created to study the relation between flight number payload mass and launch success –



This plot suggests that as the flight number increases, the likelihood of the first stage landing successfully also rises. Additionally, payload mass seems to play a role; even with heavier payloads, the first stage frequently returns successfully

Appendix (Cont…)

- The python code snippet for creating the app layout-

```python
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color': '#503D36',
                                               'font-size': 40}),

                                dcc.Dropdown(id='site-dropdown',
                                             options=[
                                                 {'label': 'All Sites', 'value': 'ALL'},
                                                 {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},
                                                 {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},
                                                 {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},
                                                 {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}],
                                             value='ALL',
                                             placeholder='place holder here',
                                             searchable=True
                                             ),
                                html.Br(),
                                html.Div(dcc.Graph(id='success-pie-chart')),
                                html.Br(),

                                html.P("Payload range (Kg):"),
                                dcc.RangeSlider(id='payload-slider', min=min_payload, max=max_payload, step=1000,
                                                marks={int(min_payload): str(int(min_payload)), int(max_payload):str(int(max_payload))}, value=[min_payload, max_payload]),


                                html.Div(dcc.Graph(id='success-payload-scatter-chart')),
                                ])

@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))
```

- The python code to add the callback functions-

```python
@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))

def get_pie_chart(entered_site):
    filtered_df = spacex_df
    if entered_site == 'ALL':
        fig=px.pie(filtered_df, values='class',
        names='Launch Site',
        title='Total Success Launches by Site')
        return fig
    else:
        df7 = filtered_df[filtered_df['Launch Site'] == entered_site]
        df7 = df7.groupby(['Launch Site', 'class']).size().reset_index(name='class count')
        fig=px.pie(df7, values='class count', names='class', title='Total Success Launches for Site {}'.format(entered_site))
        return fig

@app.callback(Output(component_id='success-payload-scatter-chart', component_property='figure'),
              [Input(component_id='site-dropdown', component_property='value'), Input(component_id='payload-slider', component_property='value')])

def get_scatter_chart(site_location, payload):
    filtered_df = spacex_df[spacex_df['Payload Mass (kg)'].between(payload[0],payload[1])]
    if site_location == 'ALL':
        fig=px.scatter(filtered_df, x='Payload Mass (kg)', y='class', color="Booster Version Category", title='Correaltion between payload and success for all sites')
        return fig
    else:
        df8 = filtered_df[filtered_df['Launch Site'] == site_location]
        # df8 = df8.groupby(['Launch Site','class'])['Payload Mass (kg)'].mean().reset_index()
        fig=px.scatter(df8, x='Payload Mass (kg)', y='class', color="Booster Version Category", title='Correaltion between payload and success for site {}'.format(site_location))
        return fig
```

Thank you!