Winning Space Race with Data Science

Uppalaiah Erugu
02.20.2024

Outline

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

Executive Summary

The Capstone project follows and presents below methodologies and results

Summary of methodologies

- ❑ Data Collection
- ❑ Data Wrangling
- ❑ Exploratory Data Analysis
- ❑ Interactive Visual Analytics
- ❑ Predictive Analysis

Summary of all results

- ❑ Exploratory Data Analysis (EDA) results
- ❑ Geospatial analytics
- ❑ Interactive dashboard
- ❑ Predictive analysis of classification models

Introduction

- SpaceX launches Falcon 9 rockets at a cost of $62millions while its competitors launches at cost of $165millions. This is much cheaper due to fact that SpaceX can land, and then re-use the first stage of the rocket.

- If we can predict whether the first stage lands successfully or not, the cost of a launch can be determined and this information can used to assess whether or not an alternate company should bid and SpaceX for a rocket launch.

- In this capstone project, the predictions on if the SpaceX Falcon 9 first stage lands successfully are made using classification models.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Making GET requests to the SpaceX REST API
  - Web Scraping
- Perform data wrangling
  - Used the .fillna() method to handle missing values
  - Used the .value_counts() method to determine the following:
    - Number of launches on each site
    - Number and occurrence of each orbit
    - Number and occurrence of mission outcome per orbit type
  - Created a landing outcome labels for successful and failed booster landings
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Used SQL queries to manipulate and analyze the SpaceX dataset
  - Used Pandas and Matplotlib to visualize relationships between variables, and determine patterns
- Perform interactive visual analytics using Folium and Plotly Dash
  - Geospatial analytics using Folium
  - Creating an interactive dashboard using Plotly Dash

Methodology

- Perform predictive analysis using classification models

  Used Scikit-Learn to:

  - Pre-process (standardize) the data
  - Split the data into training and testing data using train_test_split()
  - Train different classification models
  - Find the best hyperparameters using GridSearchCV

  - Plotting confusion matrices for each classification model
  - Assessing the accuracy of each classification model

Data Collection

- Data was collected using following methods
  - Making GET requests to the SpaceX REST API
  - Web Scraping

Data Collection – SpaceX API

- Make a GET response to the SpaceX REST API
- Convert the response to a .json file then to a Pandas DataFrame

- Use custom logic to clean the data
- Define lists for data to be stored in
- Call custom functions to retrieve data and fill the lists
- Use these lists as values in a dictionary and construct the dataset

- Create a Pandas DataFrame from the constructed dictionary dataset

- Filter the DataFrame to only include Falcon 9 launches
- Reset the FlightNumber column
- Replace missing values of PayloadMass with the mean PayloadMass value

SpaceX API is used to retrieve data about rocket launches, payloads, landing specifications, and landing outcomes.

https://github.com/eruguUppi/Capstone-Project/blob/main/1.%20Data%20Collection%20-%20API.ipynb

Data Collection - Scraping

Web scraping was used to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches.

https://github.com/eruguUppi/Capstone-Project/blob/main/1.%20Data%20Collection%20-%20Web%20Scraping.ipynb

- Request the HTML page from the static URL
- Assign the response to an object

↓

- Create a BeautifulSoup object from the HTML response object
- Find all tables within the HTML page

↓

- Collect all column header names from the tables found within the HTML page

↓

- Use the column names as keys in a dictionary
- Use custom functions and logic to parse all launch tables to fill the dictionary values

↓

- Convert the dictionary to a Pandas DataFrame ready for export

Data Wrangling

In order to transform the raw data into usable format

Handle missing values are using ***.fillna()*** method

Use the ***.value_counts()*** method to determine the following:

- Number of launches on each site
- Number and occurrence of each orbit
- Number and occurrence of mission outcome per orbit type

Label landing outcomes as ***1*** for successful ones and ***0*** for failed ones

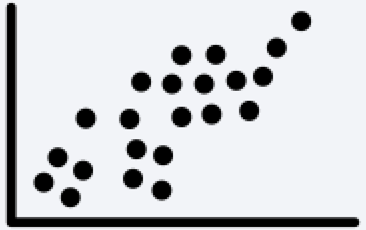
Export the DataFrame as a ***.csv*** file

https://github.com/eruguUppi/Capstone-Project/blob/main/2.%20Data%20Wrangling.ipynb

EDA with Data Visualization

Scatter Plots

Scatter plots are help us to observe relationships or correlations, between two numeric variables.



Scatter charts were created to visualize the relationships between:

- Flight Number Vs Launch Site
- Payload Vs Launch Site
- Orbit Type Vs Flight Number
- Payload Vs Orbit Type

Bar Plots

It presents categorical data with rectangular bars with length proportional to the values they represent.

A bar chart was produced to visualize the relationship between:

- Success Rate and Orbit Type

Line Plots

It connects the data points that display quantitative values over a period of time interval

Line charts were created to visualize the relationships between:

- Success Rate and Year (i.e. the launch success yearly trend)

https://github.com/eruguUppi/Capstone-Project/blob/main/3.%20Exploratory%20Data%20Analysis%20-%20Data%20Visualization.ipynb

EDA with SQL

To understand and analyze the dataset, SQL queries are performed for following purposes

- Display the names of the unique launch sites of SpaceX
- Display 5 records where launch sites begin with the string ‘CCA’
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display the average payload mass carried by booster version F9 v1.1
- List the date when the first successful landing outcome on a ground pad was achieved
- List the names of the boosters which had success on a drone ship and a payload mass between 4000 and 6000 kg
- 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 failed landing outcomes on drone ships, their booster versions, and launch site names for 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

https://github.com/eruguUppi/Capstone-Project/blob/main/3.%20Exploratory%20Data%20Analysis%20-%20SQL.ipynb

Build an Interactive Map with Folium

To visualize the launch data on an interactive map, the following steps or objects are added:

1. Mark all launch sites on a map
   - Initialise the map using a Folium ***Map*** object
   - Add a ***folium.Circle*** and ***folium.Marker*** for each launch site on the launch map

2. Mark the success/failed launches for each site on a map
   - Since most launches have the same coordinates, these locations are clustered together.
   - Before clustering them, assign a marker colour of successful (class = 1) as green, and failed (class = 0) as red.
   - To put the launches into clusters, for each launch, add a ***folium.Marker*** to the ***MarkerCluster()*** object.
   - Create an icon as a text label, assigning the ***icon_color*** as the ***marker_colour*** determined previously.

3. Calculate the distances between a launch site to its proximities
   - To explore the proximities of launch sites, calculations of distances between points can be made using the *Lat* and *Long* values.
   - After marking a point using the *Lat* and *Long* values, create a ***folium.Marker*** object to show the distance.
   - To display the distance line between two points, draw a ***folium.PolyLine*** and add this to the map.

https://github.com/eruguUppi/Capstone-Project/blob/main/4.%20Interactive%20Visual%20Analytics%20-%20Folium.ipynb

Build a Dashboard with Plotly Dash

The following plots and interactions were added to a Plotly Dash dashboard:

1. Pie chart (***px.pie()***) to show the total successful launches per site
   - This makes it clear to see which sites are most successful
   - The chart could also be filtered (using a ***dcc.Dropdown()*** object) to see the success/failure ratio for an individual site

2. Scatter graph (***px.scatter()***) to show the correlation between outcome (success or fail) and payload mass (kg)
   - This could be filtered (using a ***RangeSlider()*** object) by ranges of payload masses
   - It could also be filtered by booster version

https://github.com/eruguUppi/Capstone-Project/blob/main/4.%20Interactive%20Visual%20Analytics%20-%20Plotly%20Dash%20dashboard.py

Predictive Analysis (Classification)

To develop, evaluate, and find the best performing classification model, the below steps were followed:

Model Development

- To prepare the dataset for model development:
  - Load dataset
  - Perform necessary data transformations (standardise and pre-process)
  - Split data into training and test data sets, using ***train_test_split()***
  - Decide which type of machine learning algorithms are most appropriate
- For each chosen algorithm:
  - Create a ***GridSearchCV*** object and a dictionary of parameters
  - Fit the object to the parameters
  - Use the training data set to train the model

Model Evaluation

- For each chosen algorithm:
  - Using the output ***GridSearchCV*** object:
    - Check the tuned hyperparameters (***best_params_***)
    - Check the accuracy (***score*** and ***best_score_***)
  - Plot and examine the Confusion Matrix

Find the Best Classification Model

- Review the accuracy scores for all chosen algorithms
- The model with the highest accuracy score is determined as the best performing model

https://github.com/eruguUppi/Capstone-Project/blob/main/5.%20Predictive%20Analysic%20-%20Classification.ipynb

Results

Exploratory Data Analysis Results

Interactive Analysis Results

Predictive Analysis Results

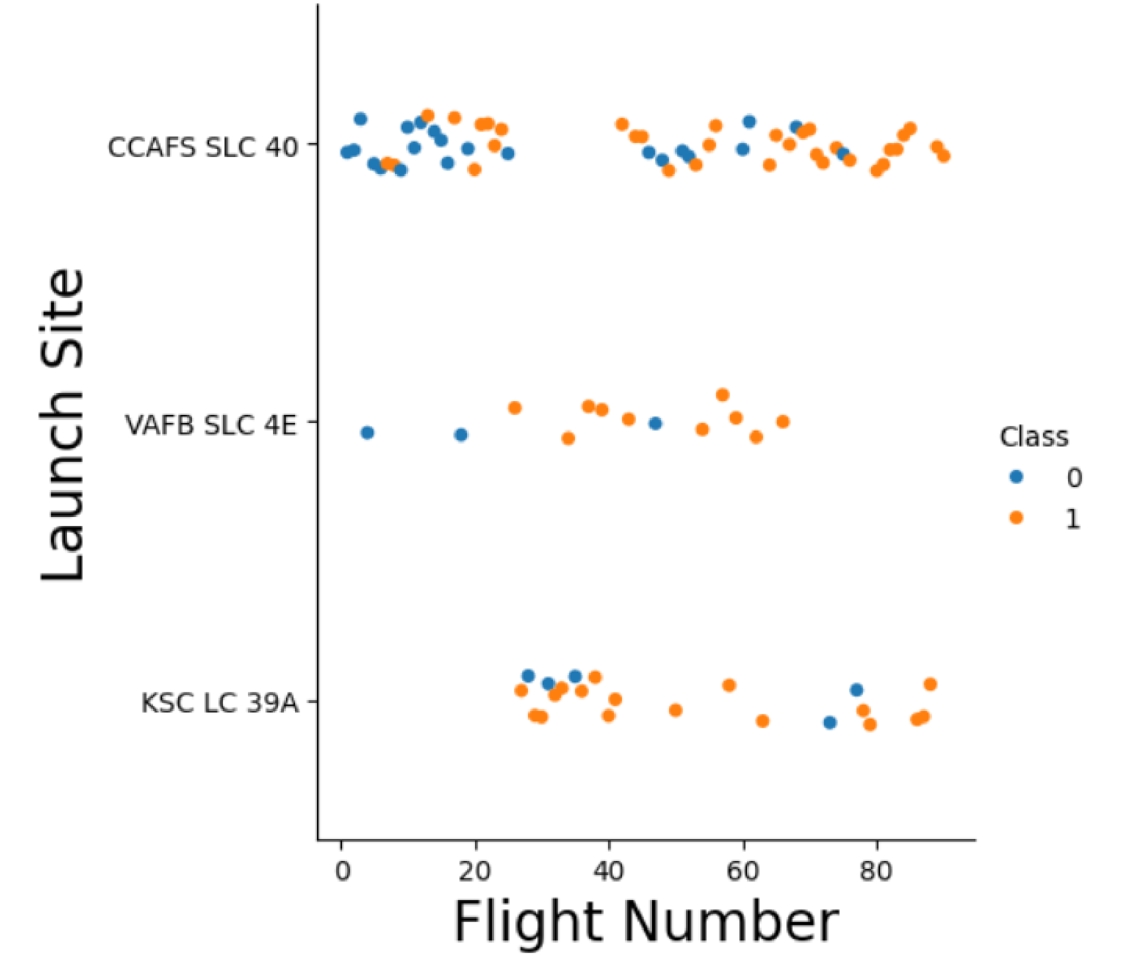
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

The scatter plot of Launch Site vs. Flight Number reveals that:

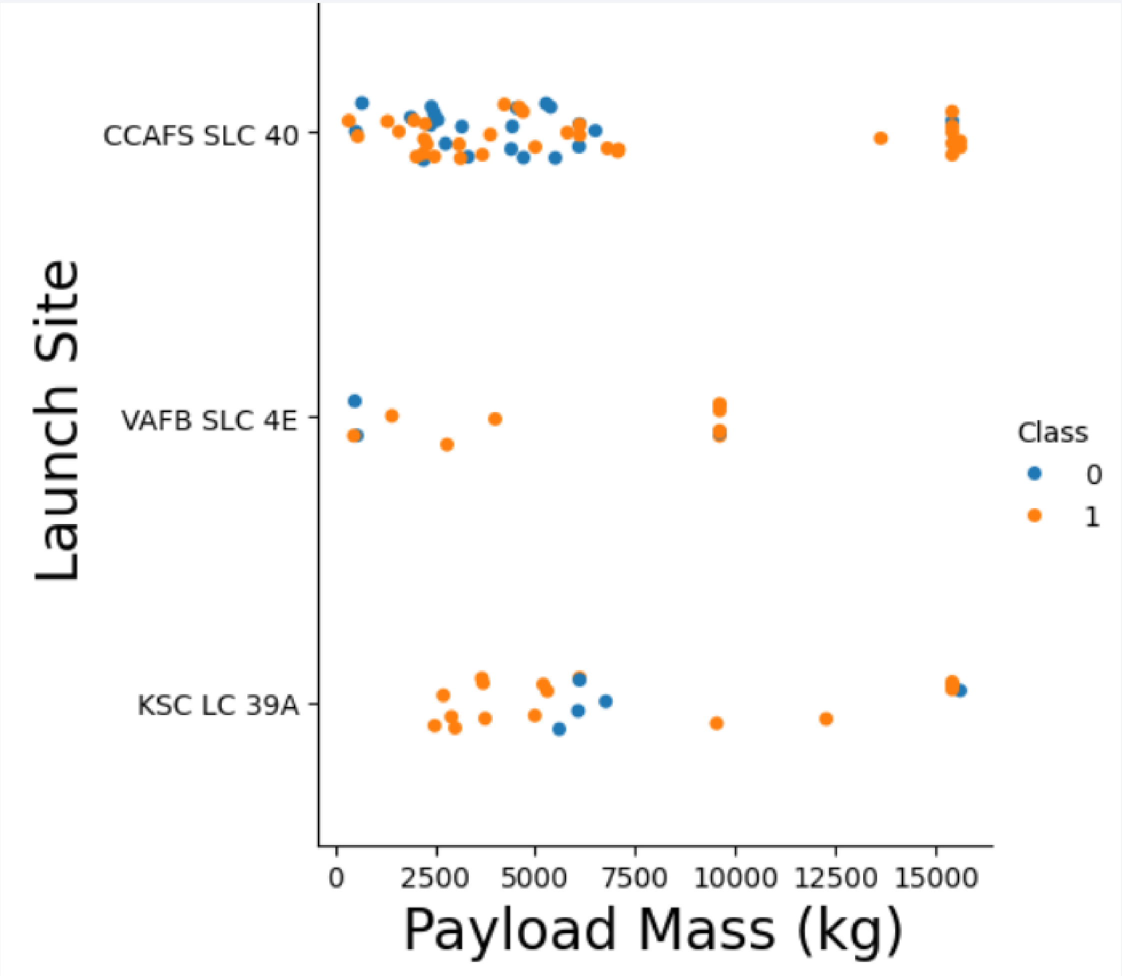
- The rate of success at a launch site increases with the number of flights
- Most of the early are launched from CCAFS SLC 40, and are unsuccessful.
- The flights from VAFB SLC 4E also show this trend, that earlier flights are less successful.
- No early flights are launched from KSC LC 39A, so the launches from this site are more successful.
- Above a flight number of around 30, there are significantly more successful landings (Class = 1).



Payload vs. Launch Site

The scatter plot of Launch Site vs. Payload Mass shows that:

- Most of landings are successful above a payload mass of around 7000 kg
- There are a very few heavier launches and are successful too
- There is no clear correlation between payload mass and success rate for a given launch site but the launches from VAFB SLC 4E are mostly successful for all payloads
- All sites launched a variety of payload masses, with most of the launches from CCAFS SLC 40 being comparatively lighter payloads



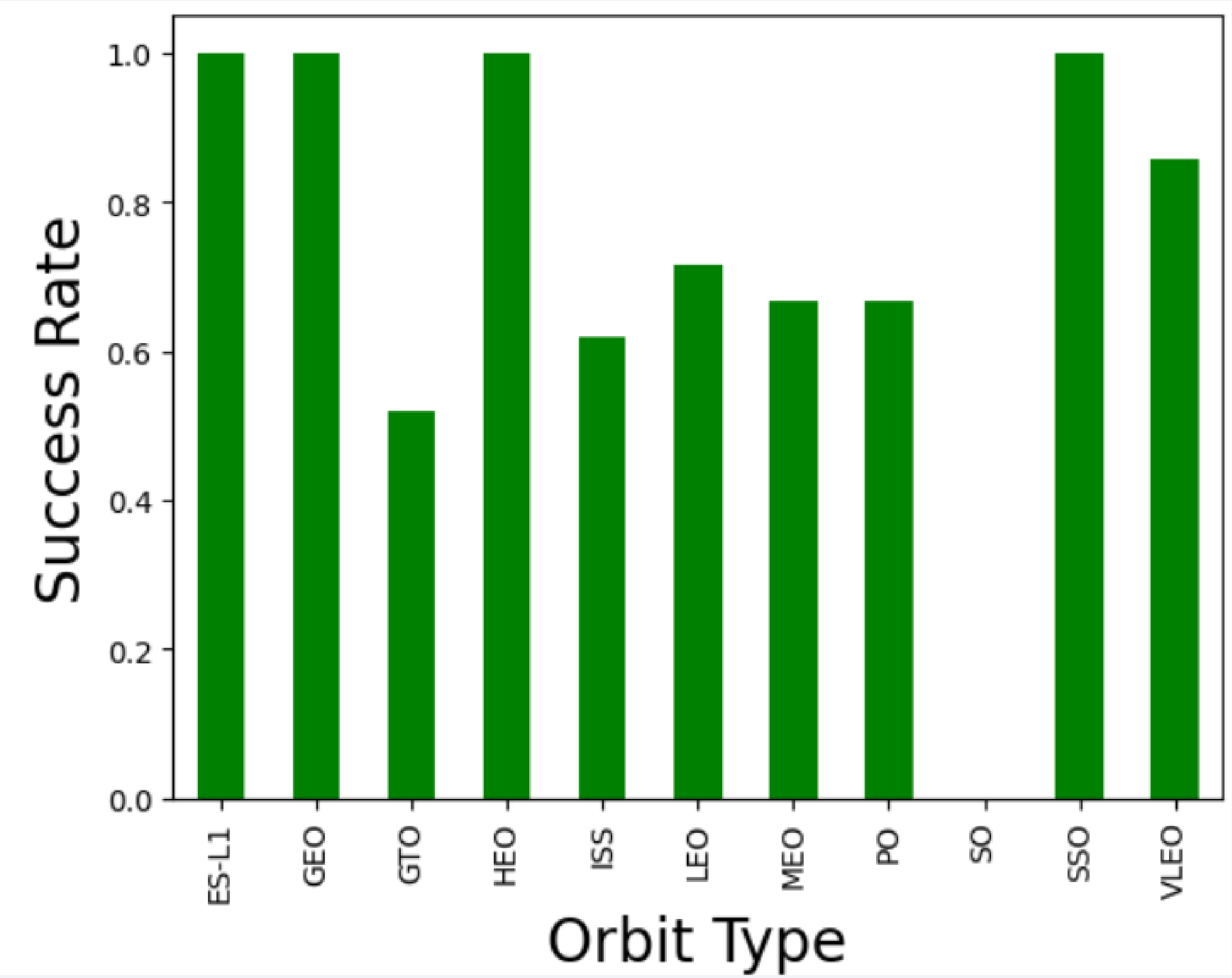
Success Rate vs. Orbit Type

The bar chart of Success Rate vs. Orbit Type shows that the following orbits have the highest (100%) success rate:

- ES-L1 (Earth-Sun First Lagrangian Point)
- GEO (Geostationary Orbit)
- HEO (High Earth Orbit)
- SSO (Sun-synchronous Orbit)

The orbit with the lowest (0%) success rate is:

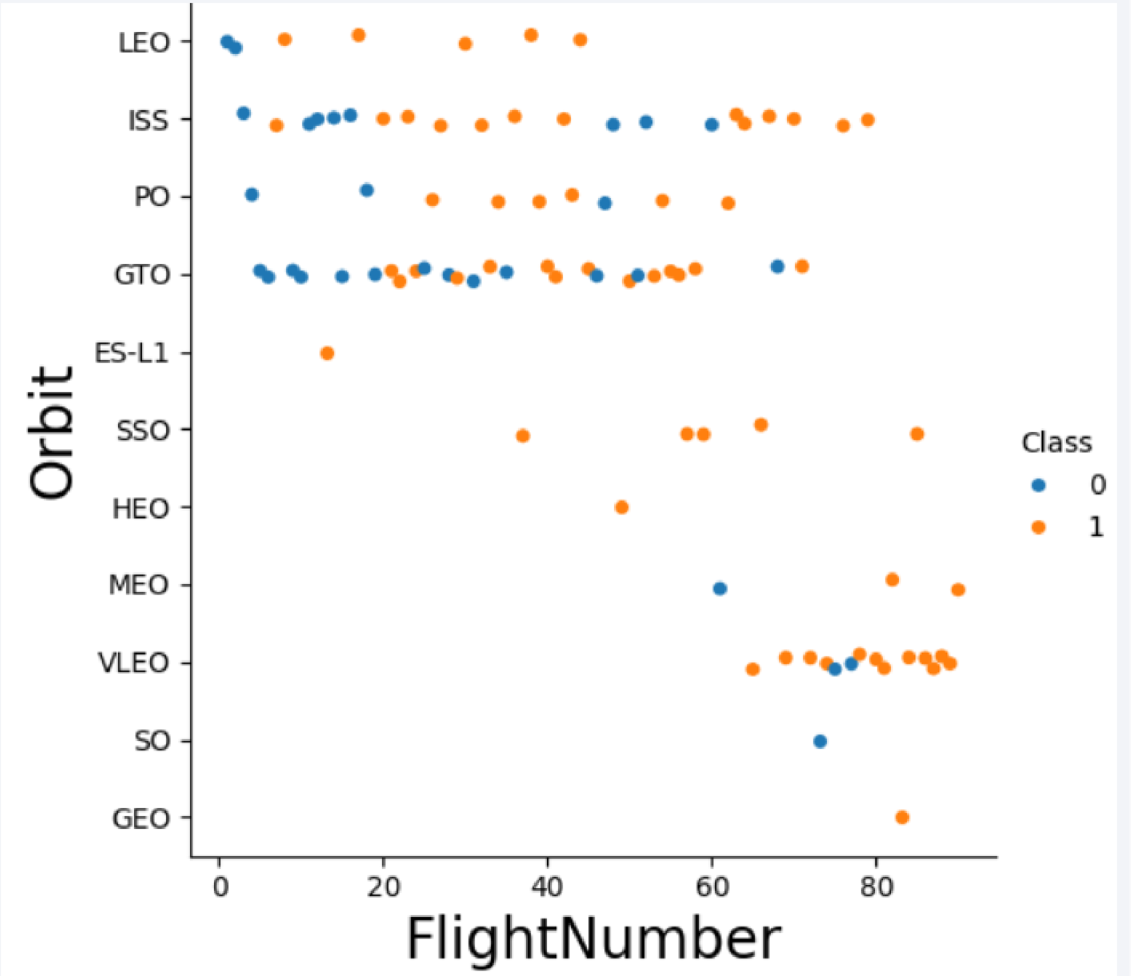
- SO (Heliocentric Orbit)



Flight Number vs. Orbit Type

Scatter plot of Orbit Type vs. Flight number shows:

- GEO, HEO, and ES-L1 orbits have 100% success rate with only one launch.
- The 100% success rate of SSO launch is impressive with all successful flights.
- No relationship between Flight Number and Success Rate for GTO is found.
- The success rate of booster landings seems to increase with Flight Number in all launches except in GTO.



Payload vs. Orbit Type

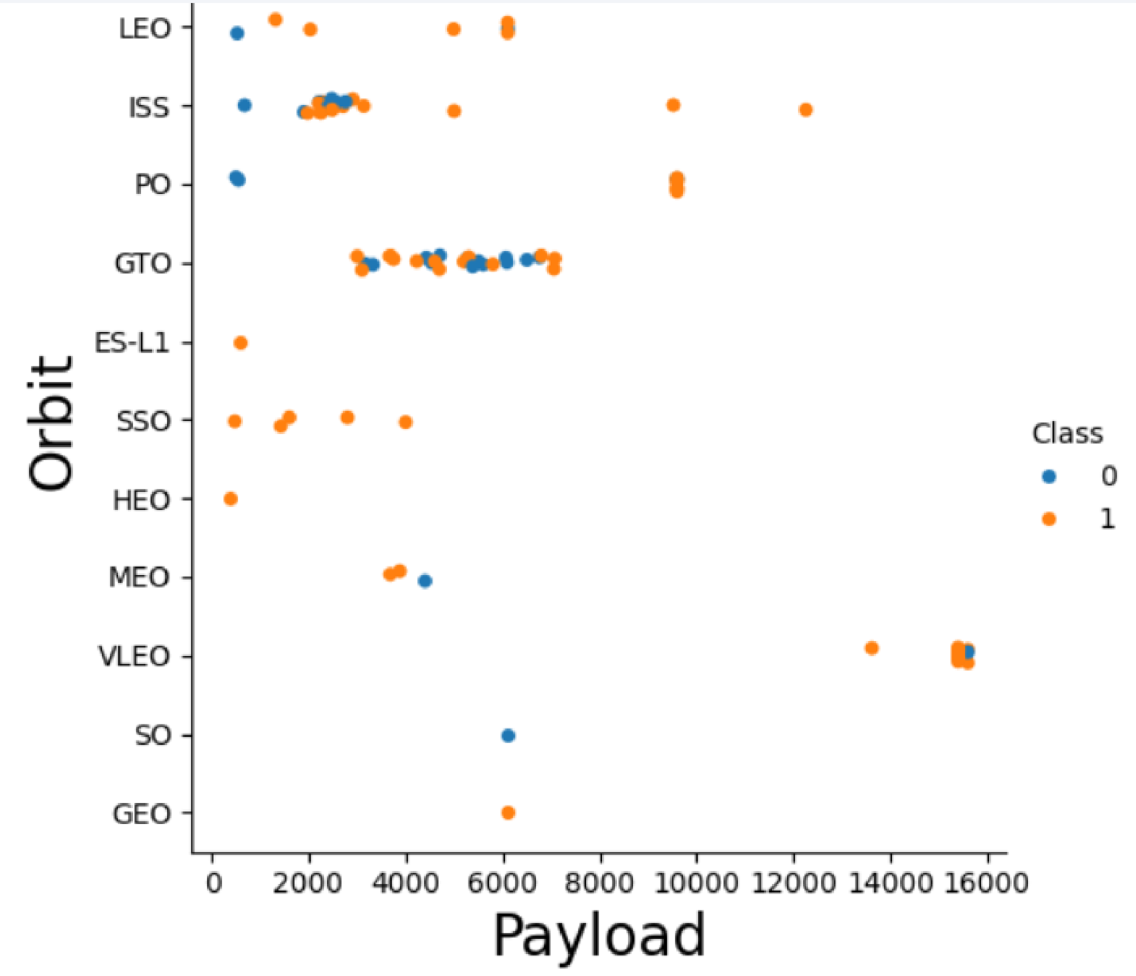
Scatter plot of Orbit Type vs. Payload Mass shows that:

- For heavy payloads below orbit launches' booster landings are successful:
  - PO
  - ISS
  - LEO

- For GTO, no relationship between payload mass and success rate is found.

- All booster landings are successful for SSO launches of all Payloads

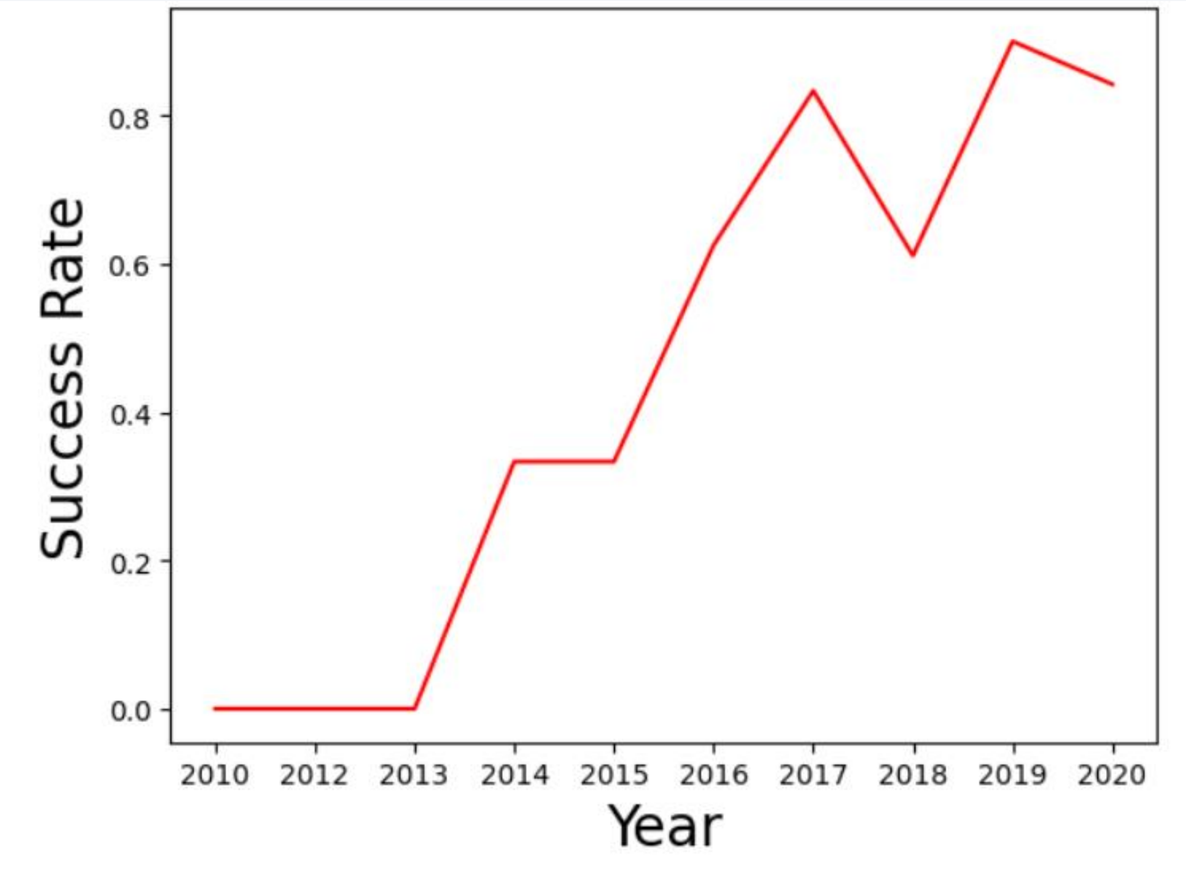
- VLEO (Very Low Earth Orbit) launches are associated with heavier payloads.



Launch Success Yearly Trend

The line chart of yearly average success rate shows that:

- Before 2013, all booster landings are unsuccessful
- After 2013, the success rate increased, despite small dips in 2018 and 2020
- After 2016, the booster landings have more 50% of success rate
- It indicates that SpaceX has mastered the technology of landing first stage boosters for reuse



All Launch Site Names - **EDA with SQL**

SQL query:

```
%sql SELECT UNIQUE(LAUNCH_SITE) FROM SPACEXTBL;
```

Output:

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

The UNIQUE constraint returns only unique values from the LAUNCH_SITE column of the SPACEXTBL table.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

SQL query:

```
%sql SELECT LAUNCH_SITE from SPACEXTBL where LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

Output:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

LIMIT 5 fetches only 5 records, and the LIKE keyword is used with the wild card ‘CCA%’ to retrieve string values beginning with ‘CCA’.

Total Payload Mass

Calculate the total payload carried by boosters from NASA

SQL query:

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

Output:

SUM(PAYLOAD_MASS__KG_)
45596

The SUM keyword is used to calculate the total of the LAUNCH column, and the WHERE clause is used to filter the results to only boosters from NASA (CRS).

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1

SQL query:

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

Output:

AVG(PAYLOAD_MASS__KG_)
2928.4

The AVG() function is used to calculate the average of the PAYLOAD_MASS__KG_ column, and the WHERE clause filters the results to only the F9 v1.1 booster version.

First Successful Ground Landing Date

Find the dates of the first successful landing outcome on ground pad

SQL query:

```
%sql SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)';
```

Output:

MIN(Date)
2015-12-22

The MIN() function is used to calculate the minimum of the DATE column, i.e. the first date, and the WHERE clause filters the results to only the successful ground pad landings.

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

SQL query:

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

Output:

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

The WHERE clause is used to filter the results to include only those that satisfy both conditions. The BETWEEN keyword allows for 4000 < x < 6000 values to be selected.

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

SQL query:

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

Output:

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

The COUNT function is used to calculate the total number of mission outcomes, and the GROUPBY statement is also used to group these results by the type of mission outcome.

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

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

SQL query:

Output:

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

A subquery is used here. The SELECT statement within the brackets finds the maximum payload, and this value is used in the WHERE condition. The DISTINCT keyword is then used to retrieve only distinct /unique booster versions.

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

SQL query:

```
%sql SELECT LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL \
    WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(Date,0,5)='2015';
```

Output:

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

The WHERE clause is used to filter the results for only failed landing outcomes, AND only for the year of 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 query:

```
%%sql
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
```

Output:

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

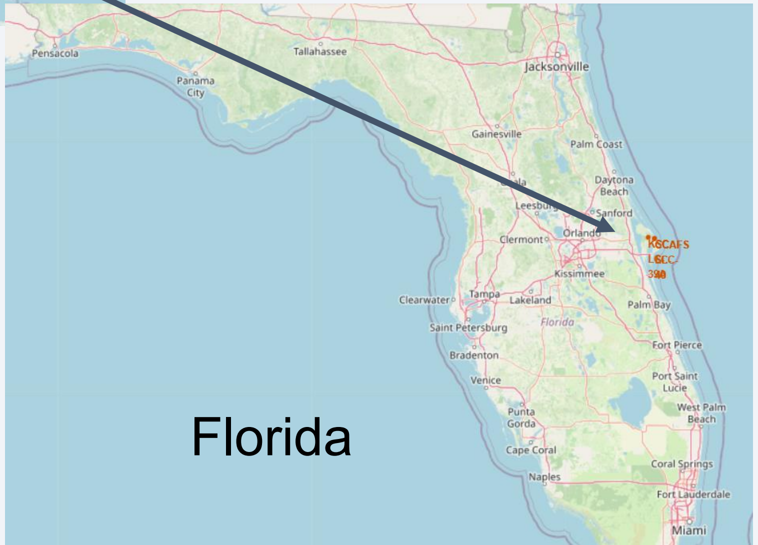
The WHERE clause is used with the BETWEEN keyword to filter the results to dates only within those specified. The results are then grouped and ordered, using the keywords GROUP BY and ORDER BY, respectively, where DESC is used to specify the descending order.

Section 3

Launch Sites Proximities Analysis

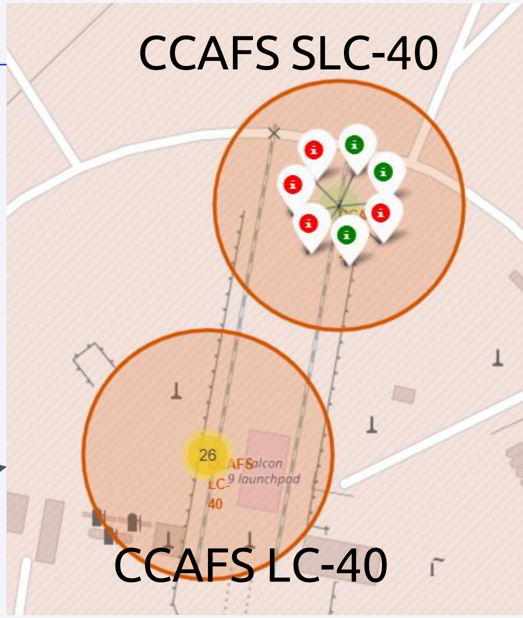
All SpaceX Launch Sites

All SpaceX launch sites are on coasts of Florida and California, USA.

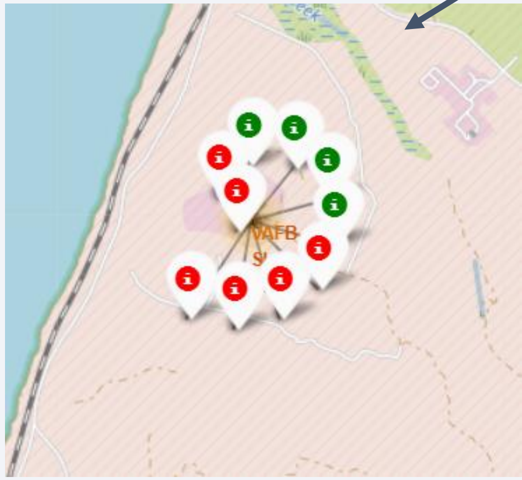


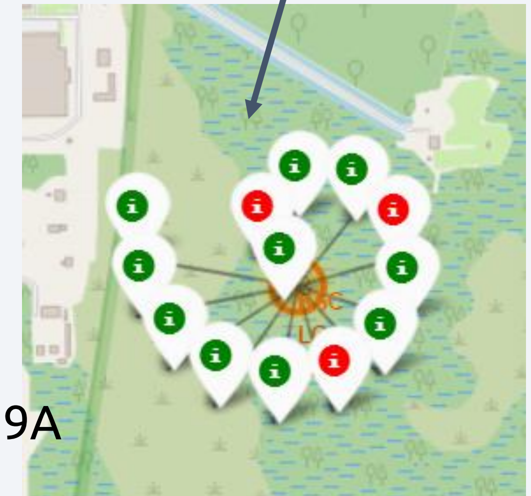
Success/Failed Launches for Each Site

Launches have been grouped into clusters, and annotated with **green** icons for successful landings, and **red** icons for failed ones.

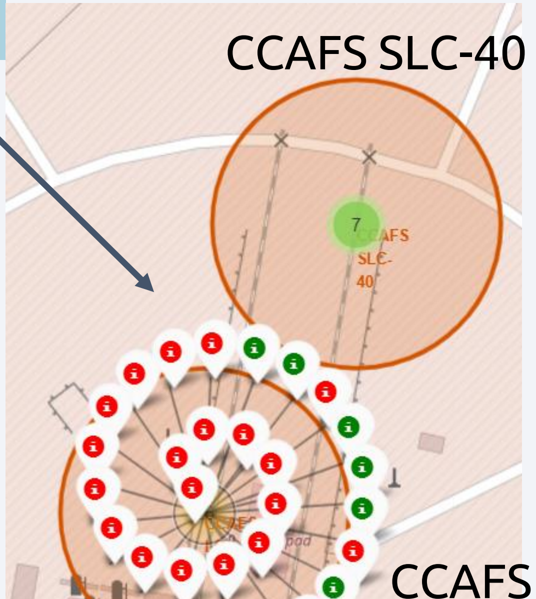


VAFB SLC-4E



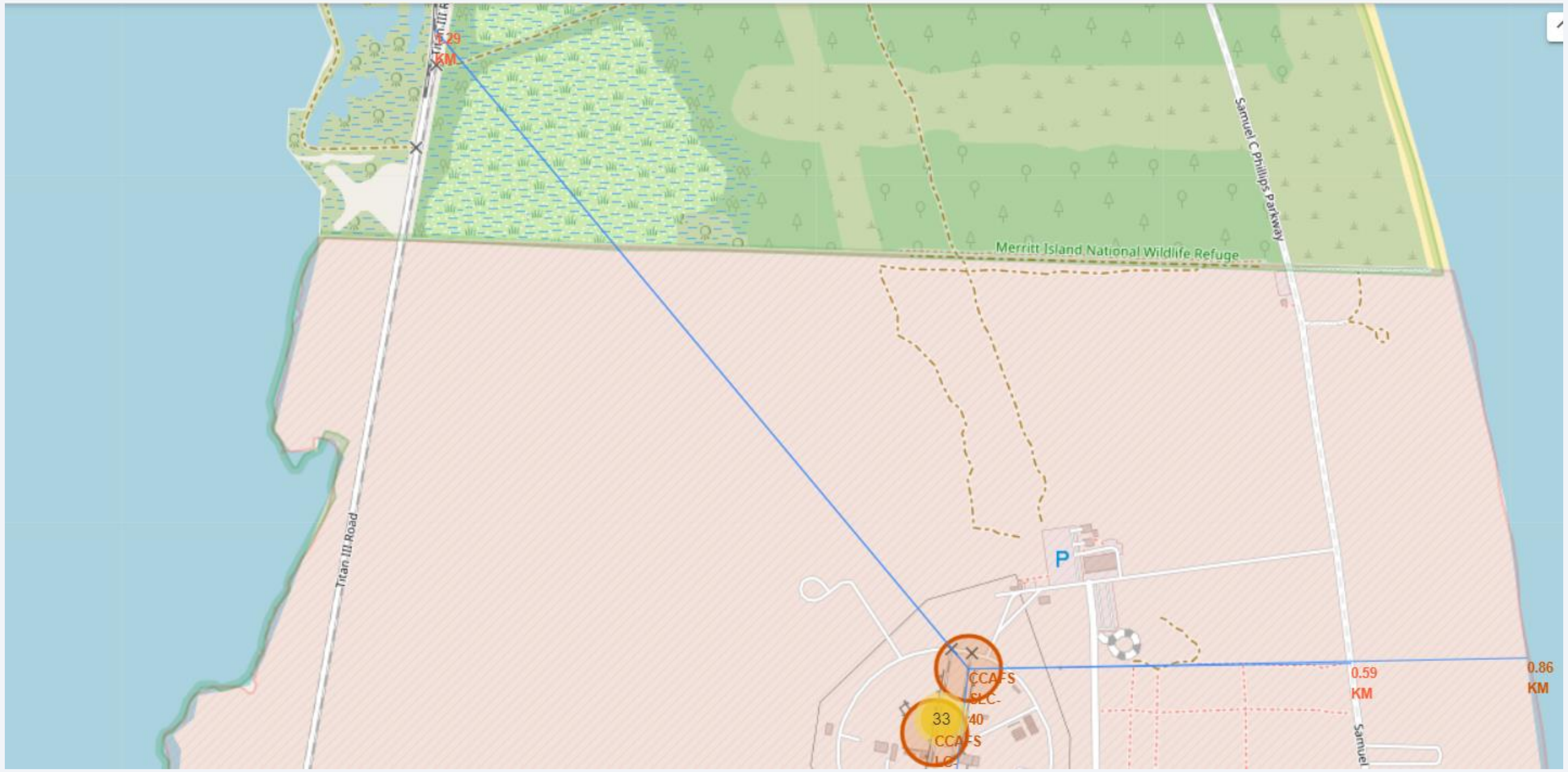


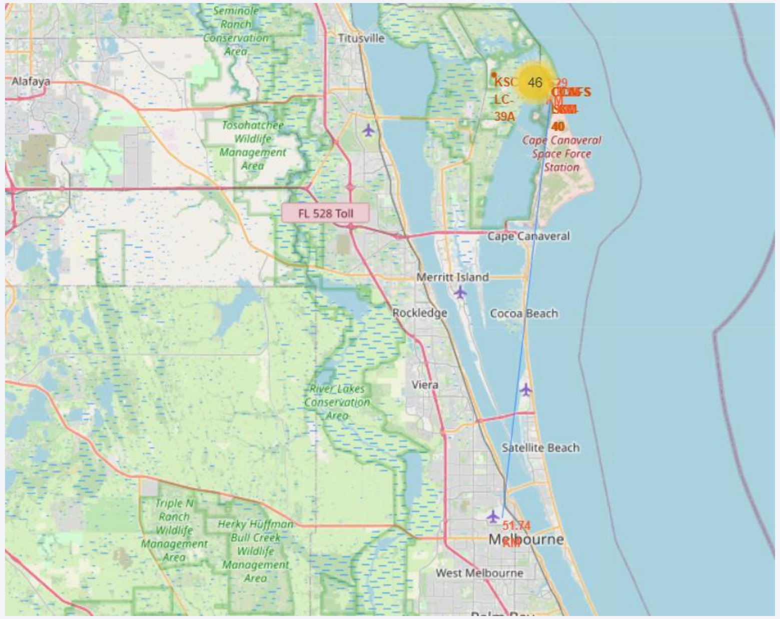
KSC LC-39A



Proximity of Launch Sites to Other Points of Interest

Using the **CCAFS SLC-40** launch site as an example site, the distance between launch site, and nearby highway, coastline, railroad and city are calculated.





Are launch sites in close proximity to railways?

- YES. The coastline is only 0.86 km due East.

Are launch sites in close proximity to highways?

- YES. The nearest highway is only 0.59km away.

Are launch sites in close proximity to railways?

- YES. The nearest railway is only 1.29 km away.

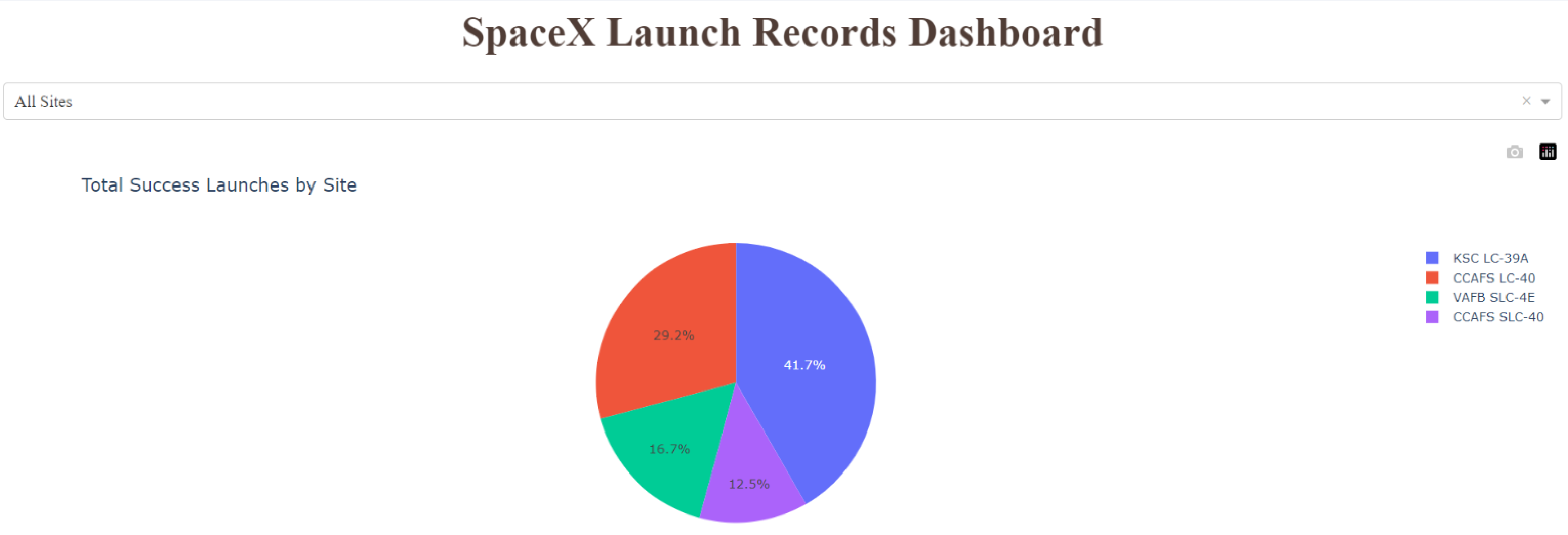
Do launch sites keep certain distance away from cities?

- YES. The nearest city is 51.4 km away.

Section 4

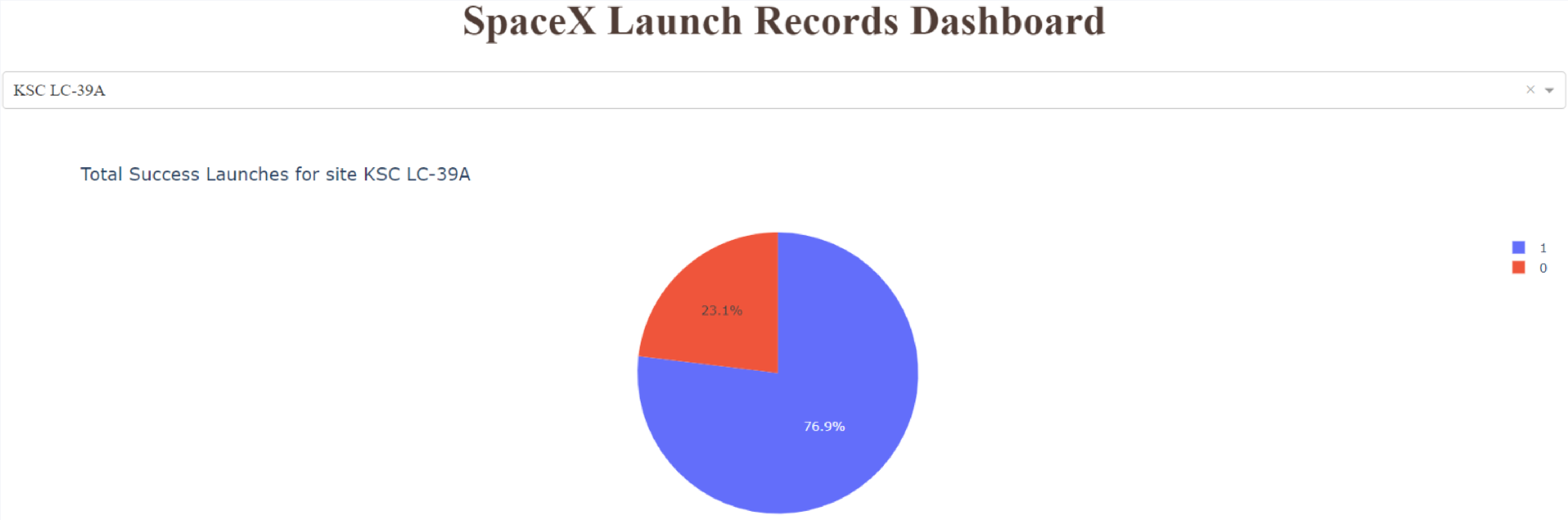
Build a Dashboard with Plotly Dash

Launch Success Count for All Cities



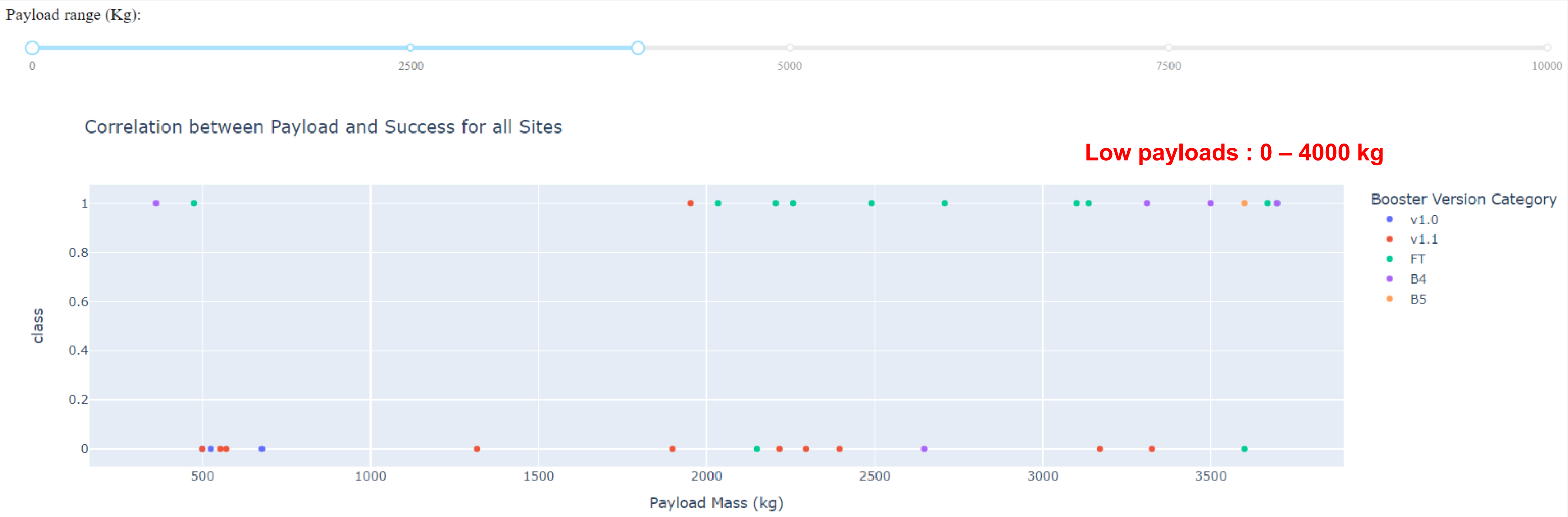
The launch site **KSC LC-39A** had the most successful launches, with 41.7% success rate.

Pie Chart for the Launch Site with the Highest Launch Success Ratio



The launch site **KSC LC-39A** also had the highest rate of successful launches, with a 76.9% success rate for 1st stage booster landings.

Payload vs Launch Outcome Scatter Plot for All Sites



- From the Payload vs Launch outcome plot for all sites, the data is divided into two ranges:
  - **Low payloads : 0 – 4000 kg**
  - **Massive payloads: 4000 – 10000 kg**

For low payloads, the number of successful landings are relatively greater than failed ones

Payload vs Launch Outcome Scatter Plot for All Sites

- For massive payloads, the number of successful landings are less than failed ones
- It is evident from the plots that the success for massive payloads is lower than that for low payloads.
- It is also worth noting that some booster types (v1.0 and B5) have not been launched with massive payloads.

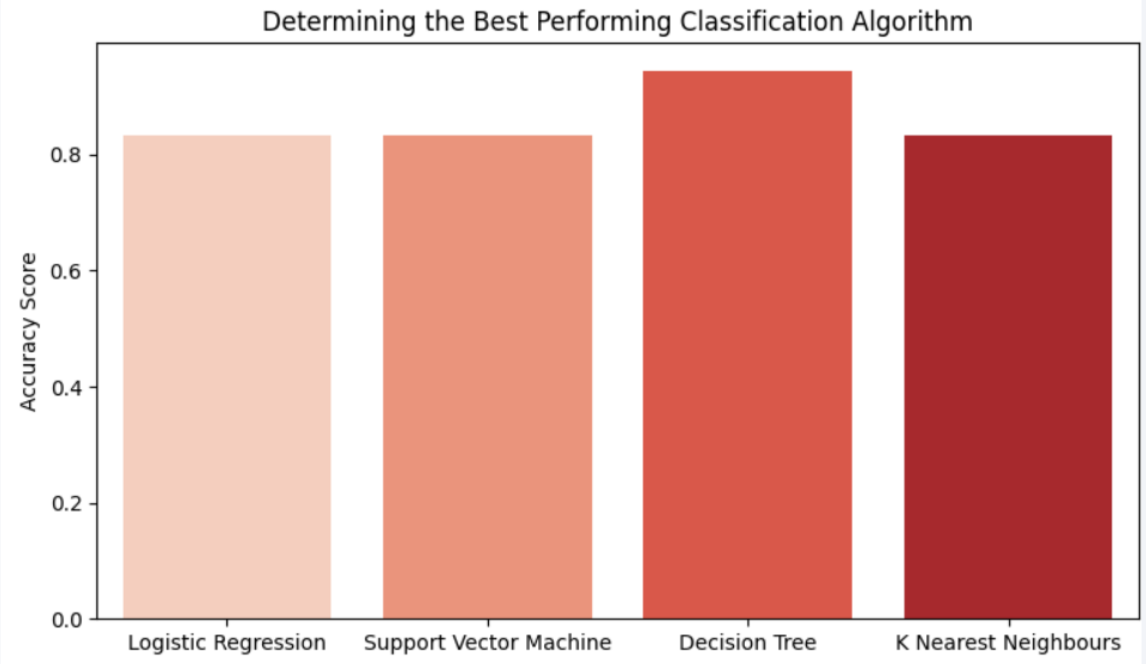
Section 5

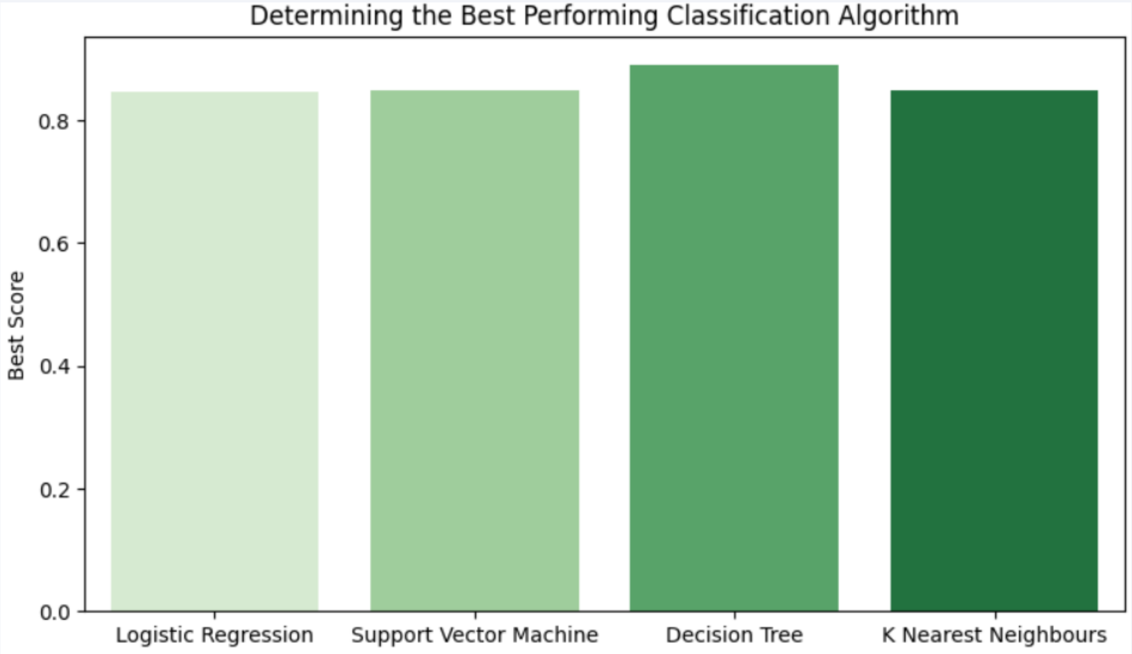
Predictive Analysis (Classification)

Classification Accuracy

From the Accuracy scores and Best scores of all classification models:

- The Decision Tree model has the highest classification accuracy
  - The Accuracy Score is 94.44%
  - The Best Score is 89.1%

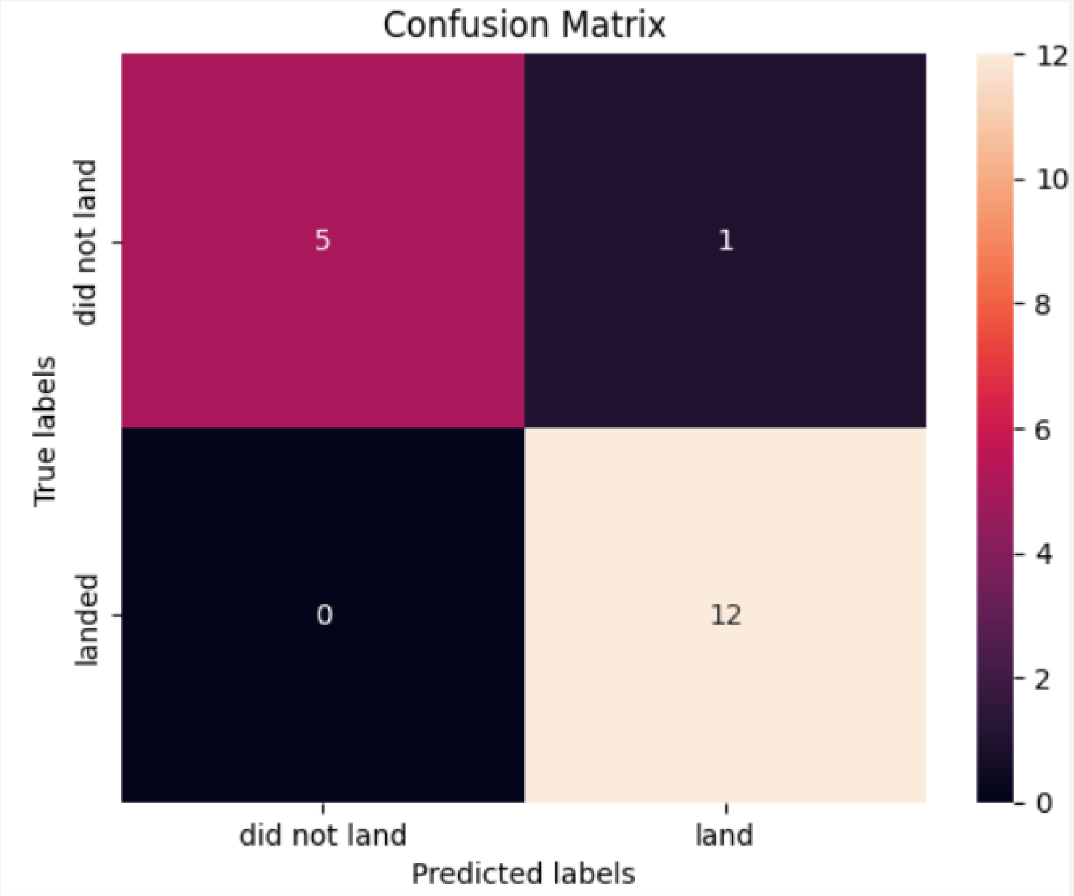




	Model	Accuray score	Best score
0	Logistic Regression	0.833333	0.846429
1	Support Vector Machine	0.833333	0.848214
2	Decision Tree	0.944444	0.891071
3	K Nearest Neighbours	0.833333	0.848214

Confusion Matrix

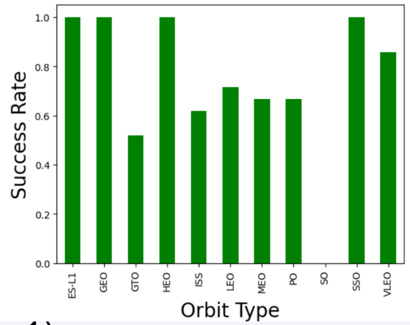
- The best performing classification model is the Decision Tree model, with an accuracy of 94.44%.
- The confusion matrix shows that only 1 out of 18 total results classified incorrectly (a false positive) by Decision Tree model.
- The other 17 results are correctly classified (5 did not land, 12 did land) by Decision Tree model.

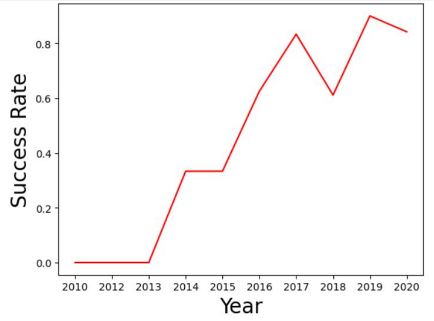


https://github.com/eruguUppi/Capstone-Project/blob/main/5.%20Predictive%20Analysic%20-%20Classification.ipynb

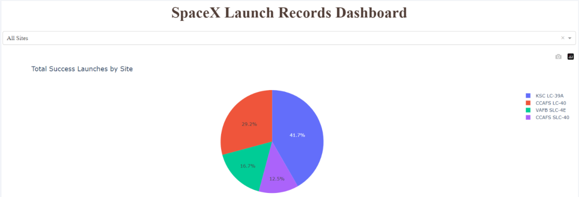
Conclusions

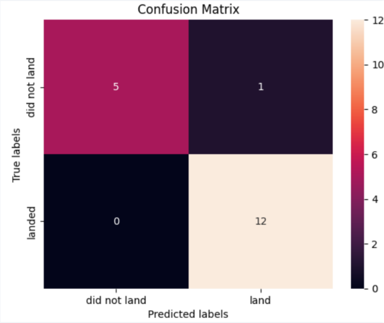
- The rate of success at a launch site increases with number of flights, the early flights being unsuccessful.
  - Before 2013, all booster landings are unsuccessful.
  - After 2013, the success rate increased, despite small dips in 2018 and 2020.
  - After 2016, the booster landings have success rate greater than 50%
- Above a flight number of around 30, there are significantly more successful landings (Class = 1).





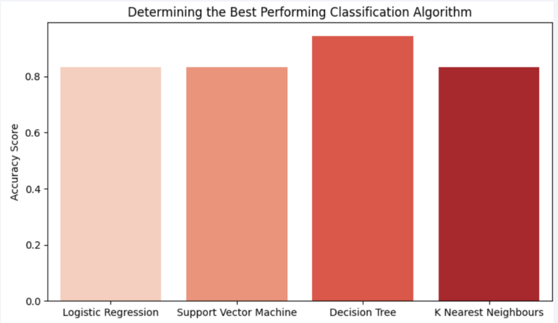
- Orbit types ES-L1, GEO, HEO, and SSO, have the highest (100%) success rate.
  - GEO, HEO, and ES-L1 orbits have 100% success rate with only one launch.
  - The 100% success rate in SSO is more impressive, with 5 successful flights.
  - The orbit types PO, ISS, and LEO, have more success with heavy payloads:
  - VLEO (Very Low Earth Orbit) launches are associated with heavier payloads





- The launch site KSC LC-39 A had the most successful launches, with 41.7% of the total successful launches, and also the highest rate of successful launches, with a 76.9% success rate.

- The success for massive payloads (over 4000kg) is lower than that for low payloads.
- The best performing classification model is the Decision Tree model, with an accuracy of 94.44%.



Appendix

Adding a callback function to render Success Pie Chart based selected launch site dropdown

```
# TASK 2:
# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output
@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[spacex_df['Launch Site'] == entered_site]
    if entered_site == 'All Sites':
        fig = px.pie(spacex_df, values='class', names='Launch Site', title='Total Success Launches by Site')
        return fig
    else:
        # return the outcomes pie chart for a selected site
        site_df = filtered_df.groupby(['Launch Site', 'class']).size().reset_index(name='class count')
        fig = px.pie(site_df,values='class count',names='class',title=f'Total Success Launches for site {entered_site}')
        return fig
```

Appendix

Adding a callback function to render Success rate vs Payload Scatter plot

```python
# TASK 4:
# Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output
@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')]) #no


def get_scatter_chart(entered_site, payload_slider):
    low, high = payload_slider
    slide=(spacex_df['Payload Mass (kg)'] > low) & (spacex_df['Payload Mass (kg)'] < high)
    dropdown_scatter=spacex_df[slide]

    #If All sites are selected, render a scatter plot to display all values for variables Payload Mass (kg) and class.
    #Point colour is set to the booster version category
    if entered_site == 'All Sites':
        fig = px.scatter(
            dropdown_scatter, x='Payload Mass (kg)', y='class',
            hover_data=['Booster Version'],
            color='Booster Version Category',
            title='Correlation between Payload and Success for all Sites')
        return fig
    else:
    #If a specific site is selected, filter the spacex_df dataframe first, then render a scatter plot to display the same as for all sites.
        dropdown_scatter = dropdown_scatter[spacex_df['Launch Site'] == entered_site]
        fig=px.scatter(
            dropdown_scatter,x='Payload Mass (kg)', y='class',
            hover_data=['Booster Version'],
            color='Booster Version Category',
            title = f'Success by Payload Size for site {entered_site}')
        return fig
```

Thank you!