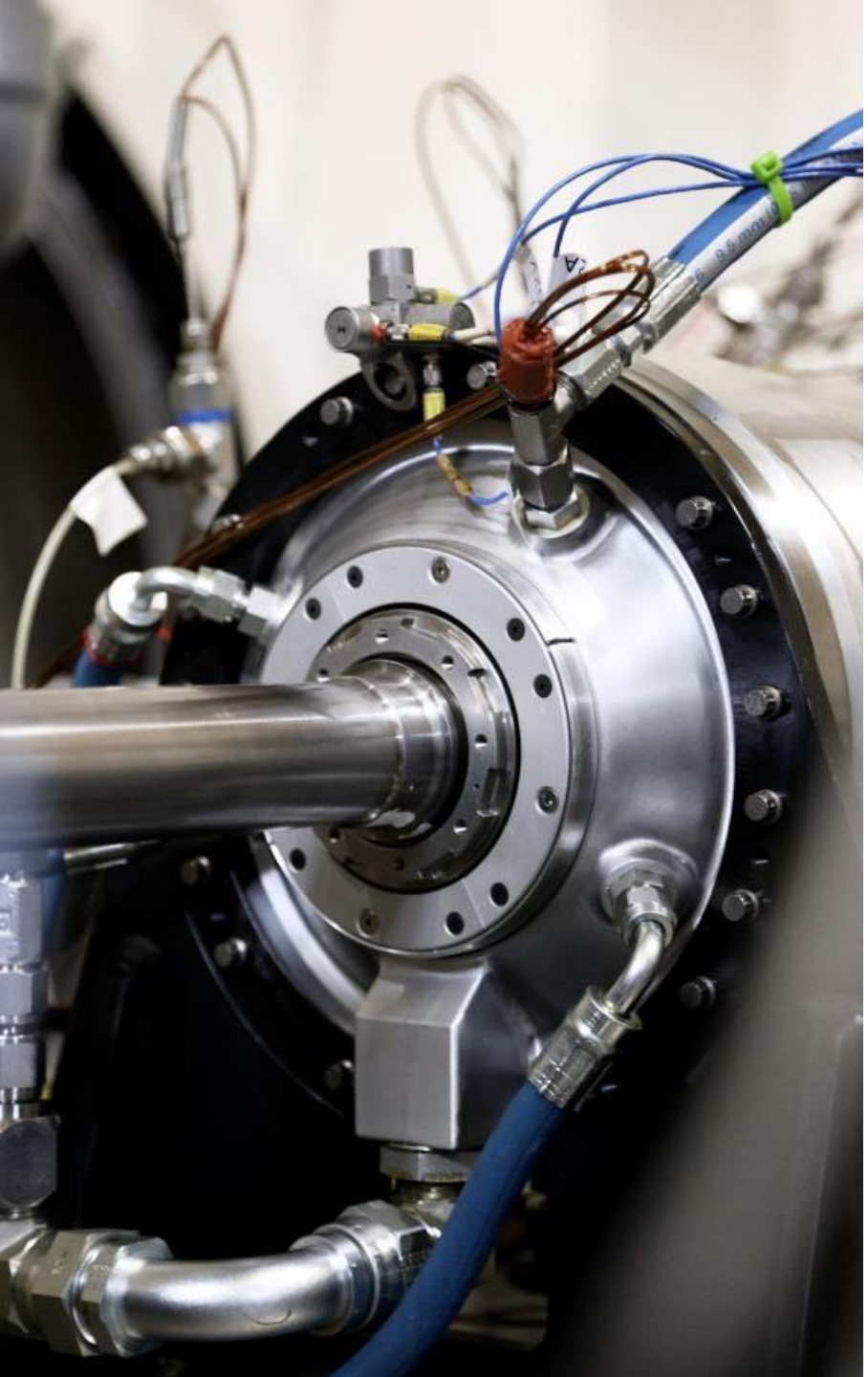
IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Uyen Tran
2024/09/19

Content outline

1. Executive Summary
2. Introduction
3. Methodology
4. Results
5. Conclusion
6. Appendix



Executive summary

Summary of methodologies

This project follows below steps to identify the factors for a successful rocket landing:

1. Data Collection
2. Data Wrangling
3. Exploratory Data Analysis
4. Interactive Visual Analytics
5. Predictive Analysis (Classification)

Summary of all results

Highlight result including:

1. Exploratory Data Analysis
2. Screenshots of Interactive Analysis
3. Result of Predictive Analysis

Introduction

Background

Leading space company SpaceX works to lower the cost of space travel for all people. Its achievements include sending manned space flights, satellite constellation launches that enable internet access, and spacecraft to the International Space Station.

SpaceX can accomplish this because of their creative reuse of the Falcon 9 rocket's first stage, which makes rocket launches relatively cheap ($62 million per launch). Some suppliers cost more than $165 million apiece and are unable to repeat the first stage.

We can calculate the launch cost by ascertaining if the first phase will land. To accomplish this, we forecast whether SpaceX or a rival business will be able to reuse the first phase using open data and machine learning models.

Problems to explore

The number of payload mass, launch site, number of flights, and orbits affect first-stage to land success

The rate of successful landings over time

Select predictive model for successful landing

Section 1
Methodology

Methodology

Flow chart of data collection:

1. Collect data	2. Wrangle data	3. Collect data
Using REST API and web scrapping from SpaceX	Ensuring data quality: filtering, handling missing value, feature engineering	Appling SQL and visualization techniques

4. Visualize data	5. Build models
Using Folium and Plotly Dash	Using classification models to predict. Tune and evaluate models to find the best model and parameter

Data Collection – API

GitHub URL of the completed SpaceX API calls notebook

The SpaceX API is a RESTful API that provides various endpoints for retrieving information about launches, rockets, missions, and more.

Step 1: Request Data from the SpaceX API	• **Set up environment and Import libraries:** Start Python script by importing the necessary libraries. • **Define the API endpoint:** The endpoint for launch data is https://api.spacexdata.com/v4/launches. • **Make the Request:** Use the requests library to fetch the data.
Step 2: Convert Response to Data Frame and Create Custom Function to Filter	• **Load data into DataFrame:** Use pd.json_normalize() to convert the JSON response into a DataFrame.python • **Define a Function**: Create a function that filters for Falcon 9 launches.
Step 3: Create Data Frame from the Dictionary	• **Create dictionary:** from the Data • **Create DataFrame:** from Dictionary
Step 4: Filter Data Frame and Replace Missing Values	• **Filter data**: Using the Custom Function • **Calculate the Mean Payload Mass:** Find the mean of the payload_mass_kg column and fill missing values.
Step 5: Export Data	• **Export Data:** to CSV File

Data Collection – Web Scraping

GitHub URL of the completed SpaceX web scrapping

Web scraping is a technique used to extract data from websites. It involves making HTTP requests to a webpage, parsing the HTML content, and extracting the desired information

Step 1: Request the Falcon9 Launch Wiki page from its URL	• **Make an HTTP:** request to the Wikipedia page to fetch the HTML content containing the Falcon 9 launch data.
Step 2: Create BeautifulSoup Object	• **Initialize a BeautifulSoup object:** using the HTML response, allowing for easy navigation and parsing of the HTML structure.
Step 3: Extract all column/ variable names from the HTML table header	• **Identify** the relevant HTML table and extract column names from the header (<th> elements**).** Iterate through the table rows (<tr> elements) to collect data from the cells (<td> elements).
Step 4: Create a data frame by parsing the launch HTML tables	• **Create** an empty dictionary with keys from the extracted column names in the previous task. Later, this dictionary will be converted into a Pandas data frame.

Data Wrangling

GitHub URL of the completed data wrangling

Data wrangling is the process of converting raw data into a usable form. It may also be called data munging or data remediation

Step 1: Load Data	• **Import data:** from API and web scrapping, import libraries and load the dataset into a DataFrame.

Step 2: Explore Data	• **Examine:** the dataset's structure and identify issues such as:
	  • Summary Statistics
	  • Data Types Overview
	  • Missing Values Analysis

Step 3: Clean Data	• **Qualify data**
	  • Handle Missing Values: Imputation or Removal
	  • Remove Duplicates
	  • Correct Data Types (e.g., converting strings to dates)
	  • Outlier Detection and Treatment

Step 4: Transform Data	• **Create new features and normalize or scale data as needed for analysis**
	  • Normalization/Standardization
	  • Encoding Categorical Variables (One-hot, Label Encoding)
	  • Feature Engineering (Creating new variables)
	  • Aggregation or Restructuring Data (Pivoting, Melting)

EDA with Data Visualization

GitHub URL of the completed data visualization

Perform exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib

- *Exploratory Data Analysis*
- *Preparing Data Feature Engineering*

Step 1: Initial Overview and Data Distribution	• **Import data:** read the SpaceX dataset into a Pandas DataFrame and print its summary • **Visualize data distribution** by plot Histogram, Box Plot, etc.

Step 2: Visualize Relationships Between Variables	• **Understand data structure and categorical variable analysis:**
	  • Scatter Plots: To assess relationships between two continuous variables.
	  • Pair Plots: To visualize relationships in a matrix format for multiple variables.
	  • Heatmaps: To visualize correlation matrices

Step 3: Featuring Engineering	• **Select, manipulate and transform raw data into features**
	  • Create dummy variables to categorical columns
	  • Feature selection
	  • Cast all numeric columns to float64

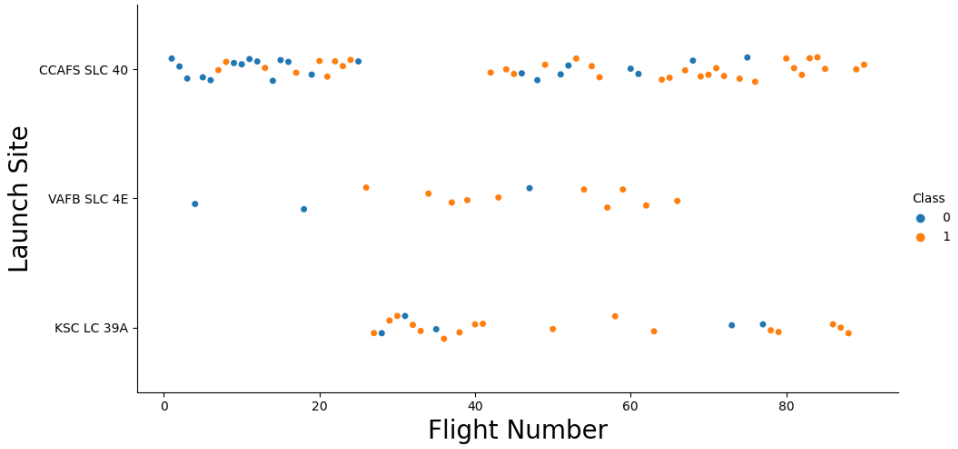
Step 4: Final Insights and Reporting	• **Dashboards:** Use tools like Tableau or Plotly Dash for interactive visualizations • **Key findings:** Summarize insights from the visualizations

EDA with Data Visualization - Scatter Plot

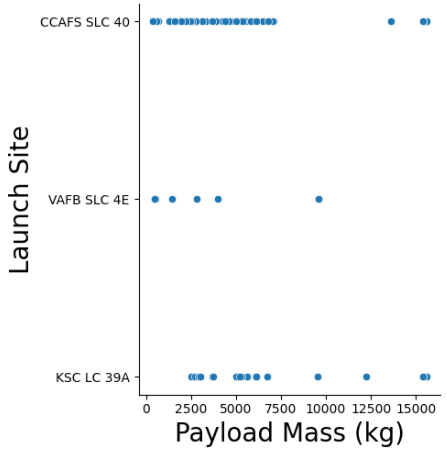
Scatter Plot

- Scatter plot uses dots to represent values for two different numeric variables.
- The position of each dot on the horizontal and vertical axis indicates values for an individual data point.

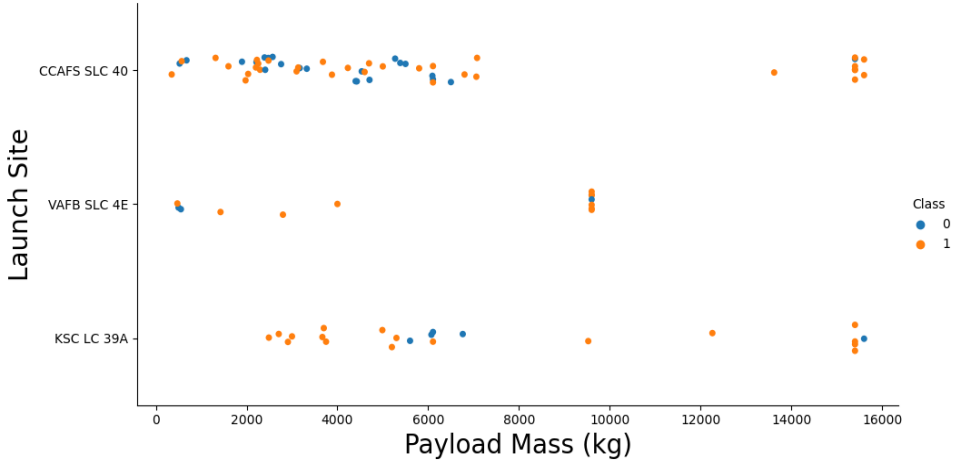
Flight Number and Launch Site



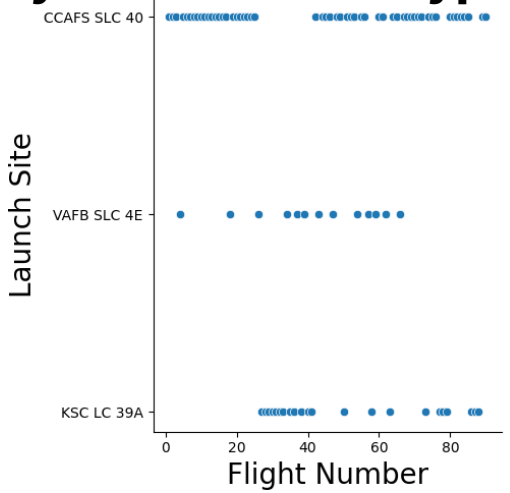
Flight Number and Orbit type



Payload and Launch Site



Payload and Orbit type



EDA with Data Visualization – Line Chart, Bar Chart

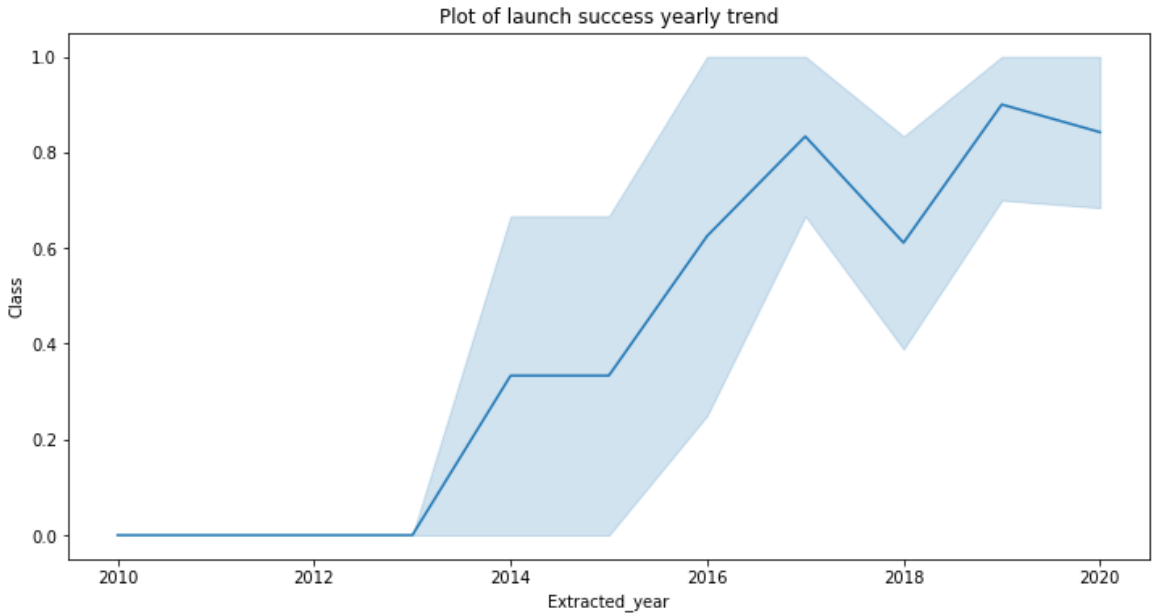
Line Chart

- See the data patterns over time and show how variables change over time

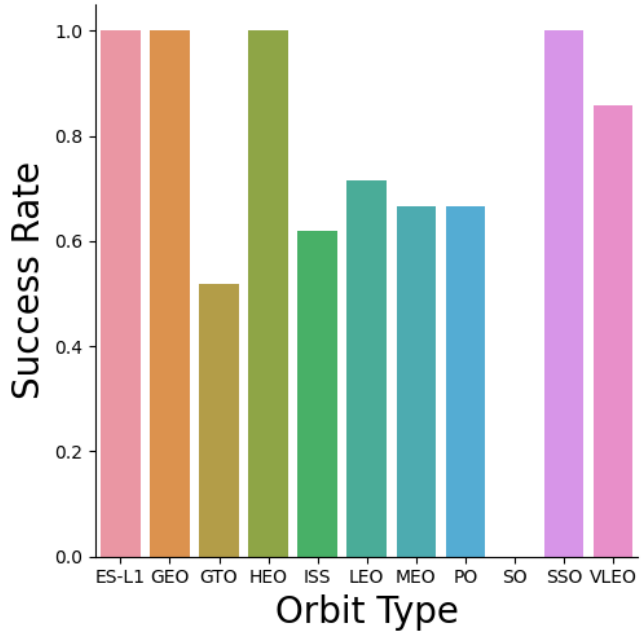
Bar Chart

- Show a distribution of data points or perform a comparison of metric values across different subgroups of your data
- Compare which groups are highest or most common, and how other groups compare against the others.

The launch success yearly trend



Success rate of each orbit type



EDA with SQL

GitHub URL

Display:

- Names of unique launch sites
- Five records where launch site begins with 'CCA'
- Total payload mass carried by boosters launched by NASA (CRS)
- Average payload mass carried by booster version F9 v1.1.

Image: Rank the count of landing outcomes

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

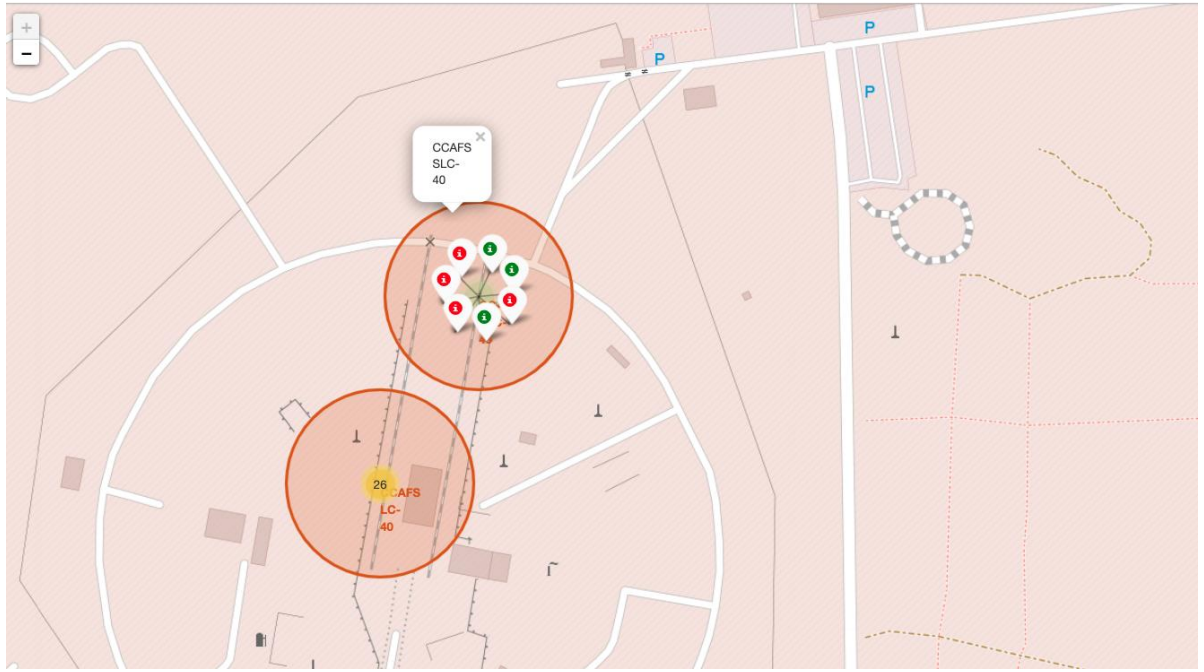
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

List:

- Date of first successful landing on ground pad
- Names of boosters which had success landing on drone ship and have payload mass greater than 4,000 but less than 6,000
- Total number of successful and failed missions
- Names of booster versions which have carried the max payload
- Failed landing outcomes on drone ship, their booster version and launch site for the months in the year 2015
- Count of landing outcomes between 2010-06-04 and 2017-03-20 (desc)

Build an Interactive Map with Folium

GitHub URL



From the color-labeled markers in marker cluster, that be able to easily identify which launch sites have relatively high success rates.

Markers Indicating Launch Sites

- Added **blue** circle at NASA Johnson Space Center's coordinate with a popup label showing its name using its latitude and longitude coordinates
- Added **red** circles at all launch sites coordinates with a popup label showing its name using its name using its latitude and longitude coordinates

Colored Markers of Launch Outcomes

- Added colored markers of successful (**green**) and unsuccessful (**red**) launches at each launch site to show which launch sites have high success rates

Distances Between a Launch Site to Proximities

- Added colored lines to show distance between launch site CCAFS SLC- 40 and its proximity to the nearest coastline, railway, highway, and city

Build a Dashboard with Plotly Dash

GitHub URL

Dropdown List with Launch Sites

- Allow user to select all launch sites or a certain launch site

Slider of Payload Mass Range

- Allow user to select payload mass range

Pie Chart

- Allow user to see successful and unsuccessful launches as a percent of the total

Scatter Chart

- Allow user to see the correlation between Payload and Launch Success

Scatter Chart Showing Payload Mass vs. Success Rate by Booster Version

Predictive Analysis (Classification)

1. Import Data and Assign Feature

- Standardize the data with Standard Scaler.
- Fit and transform the data.

2. Feature Engineering

- Create NumPy array from the Class column
- Standardize the data with Standard Scaler. Fit and transform the data.

3. Split Data and Build Models

- Split the data using train_test_split
- Create a GridSearchCV object with cv=10 for parameter optimization
- Apply GridSearchCV on different algorithms:
  - logistic regression
  - support vector machine
  - decision tree
  - K-Nearest Neighbor

4. Evaluate

- Calculate accuracy on the test data using .score() for all models
- Assess the confusion matrix for all models
- Identify the best model using Jaccard_Score, F1_Score and Accuracy

Section 2
Insights drawn from EDA

Results

Exploratory Data Analysis

- Launch success has improved over time
- KSC LC-39A has the highest success rate among landing sites
- Orbits ES-L1, GEO, HEO and SSO have a 100% success rate

Visual Analytics

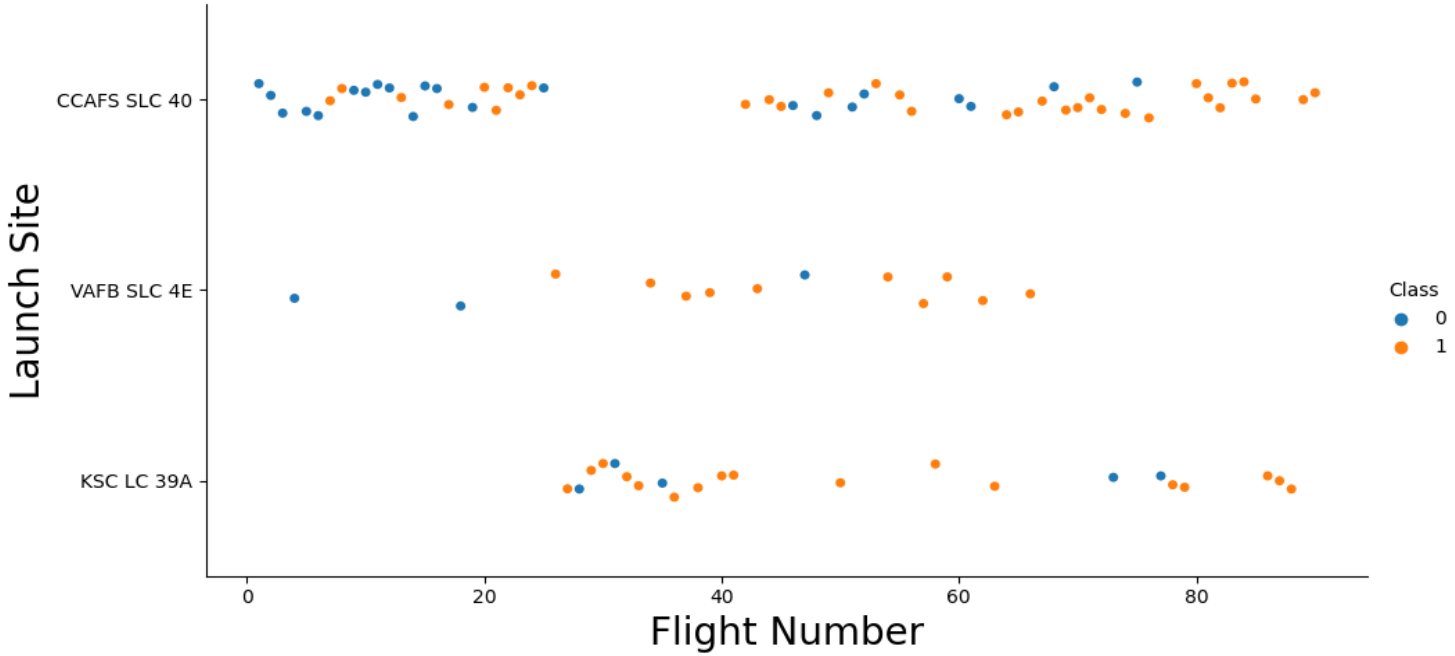
- Most launch sites are near the equator, and all are close to the coast
- Launch sites are far enough away from anything a failed launch can damage (city, highway, railway), while still close enough to bring people and material to support launch activities

Predictive Analytics

- Decision Tree model is the best predictive model for the dataset

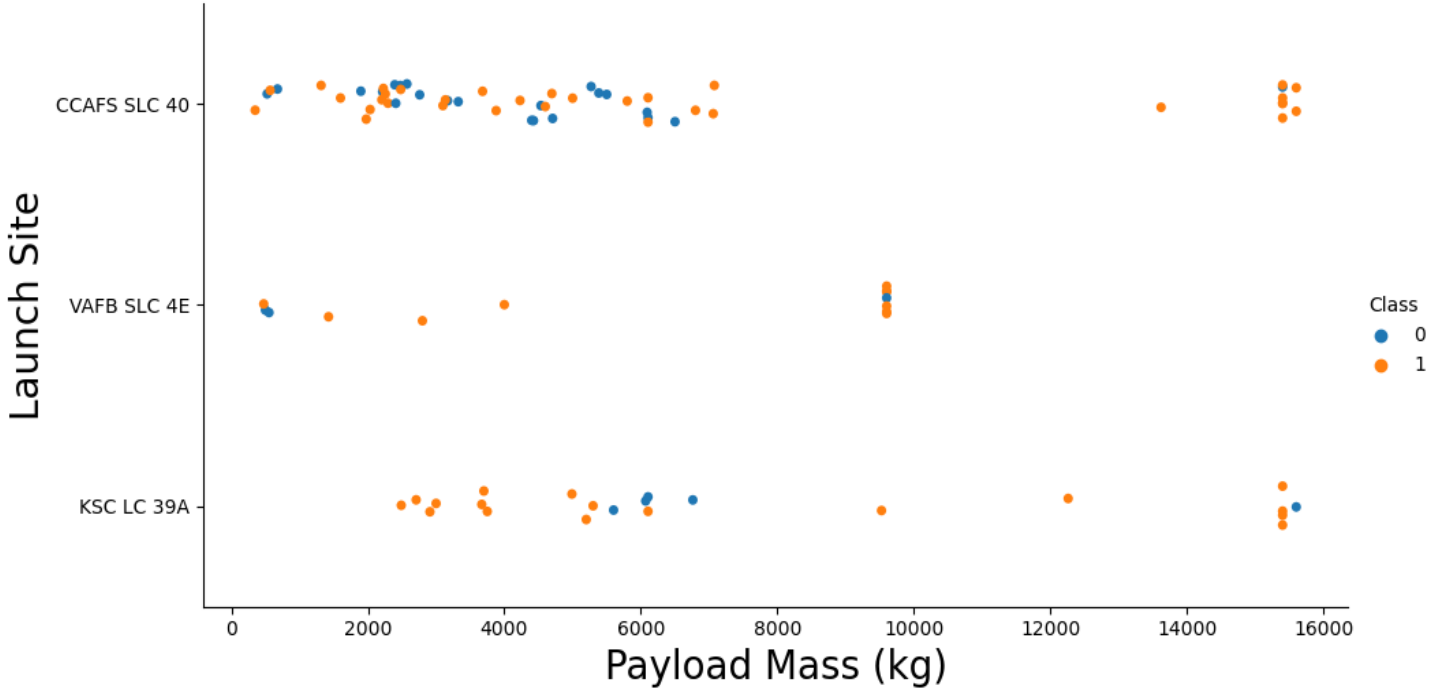
EDA - Flight Number vs. Launch Site

- **Earlier flights** had a **lower success rate** (**blue = fail**)
- **Later flights** had a **higher success rate** (**orange = success**)
- We can infer that new launches have a higher success rate
- Lower flight number (20) no failure or success metric for “KSC LC 39A”,
- But two failures for the “VAFB SLC 4E”. the flight number increases past 80, more success “CCAFS SLC 40” and none for “VAFB SLC 4E”.



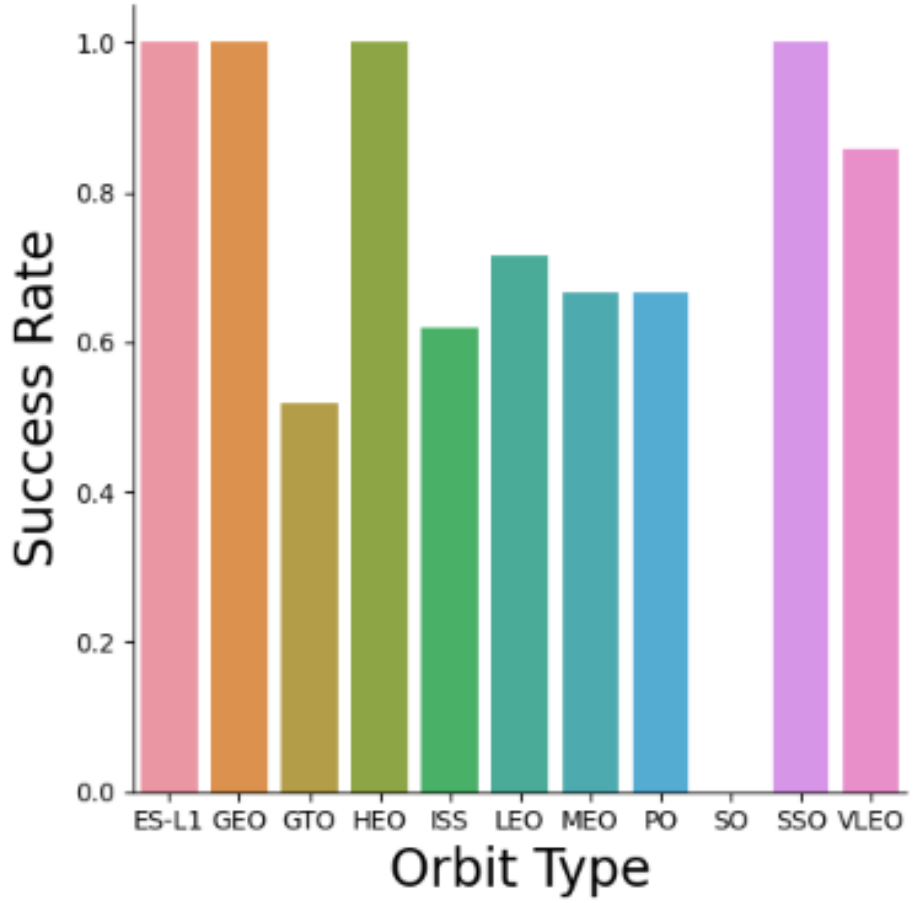
EDA - Payload vs. Launch Site

- The higher the payload mass (kg), the higher the success rate
- Most launces with a payload greater than 7,000 kg were successful
- KSC LC 39A has a 100% success rate for launches less than 5,500 kg
- VAFB SKC 4E has not launched anything greater than ~10,000 kg



EDA - Success Rate vs. Orbit Type

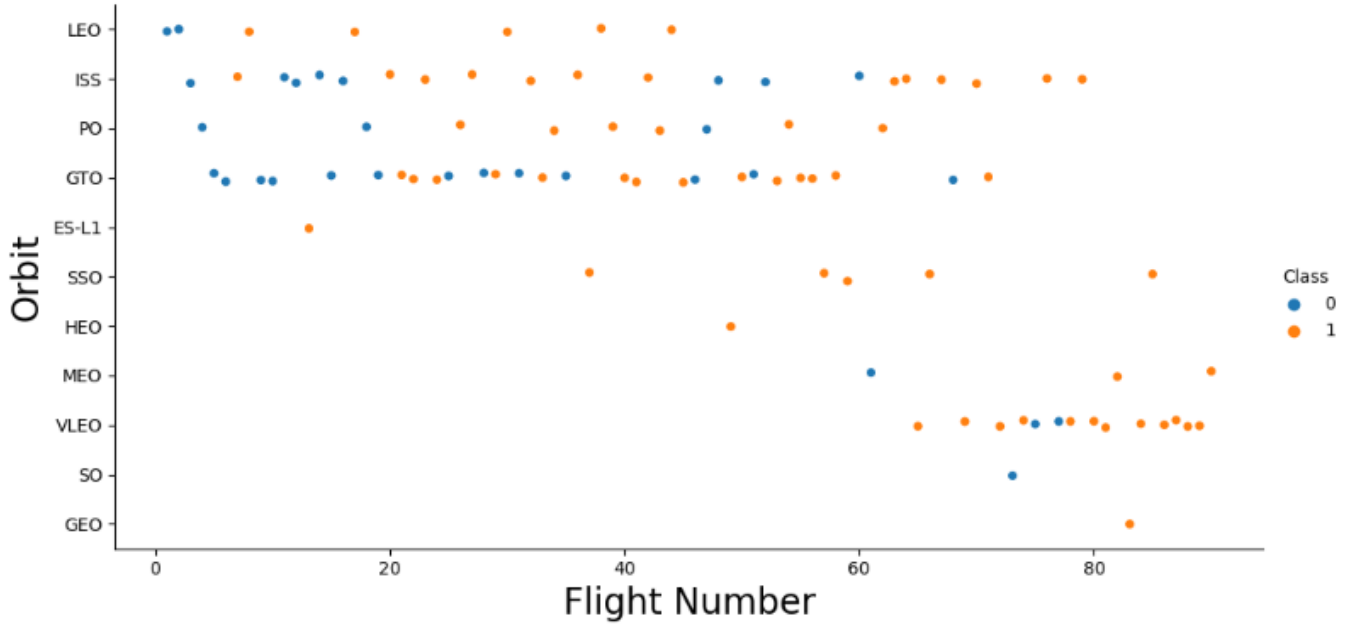
- 100% Success Rate: ES-L1, GEO, HEO and SSO
- 50%-80% Success Rate: GTO, ISS, LEO, MEO, PO
- 0% Success Rate: SO



EDA - Flight Number vs. Orbit Type

- The success rate typically increases with the number of flights for each orbit
- This relationship is highly apparent for the LEO orbit
- The GTO orbit, however, does not follow this trend

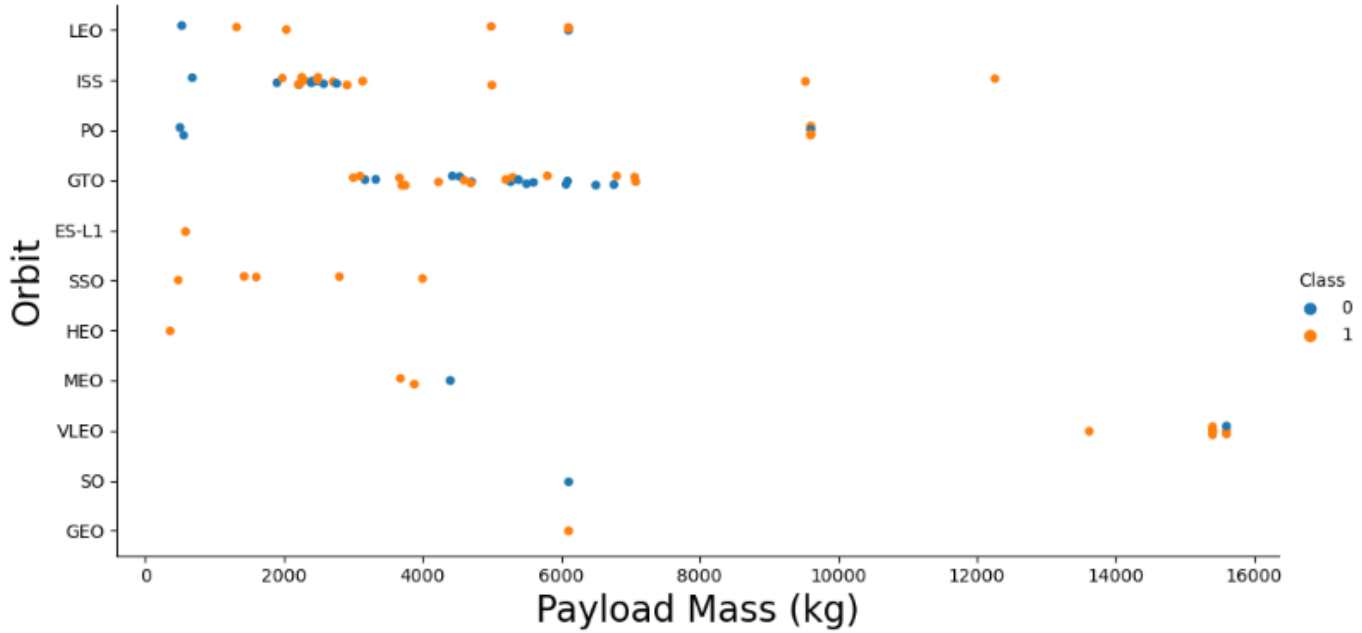
⇒ **success** is **related** to the number of **flights whereas in the GTO orbit**, there is no relationship between flight number and the orbit.



EDA - Payload vs. Orbit Type

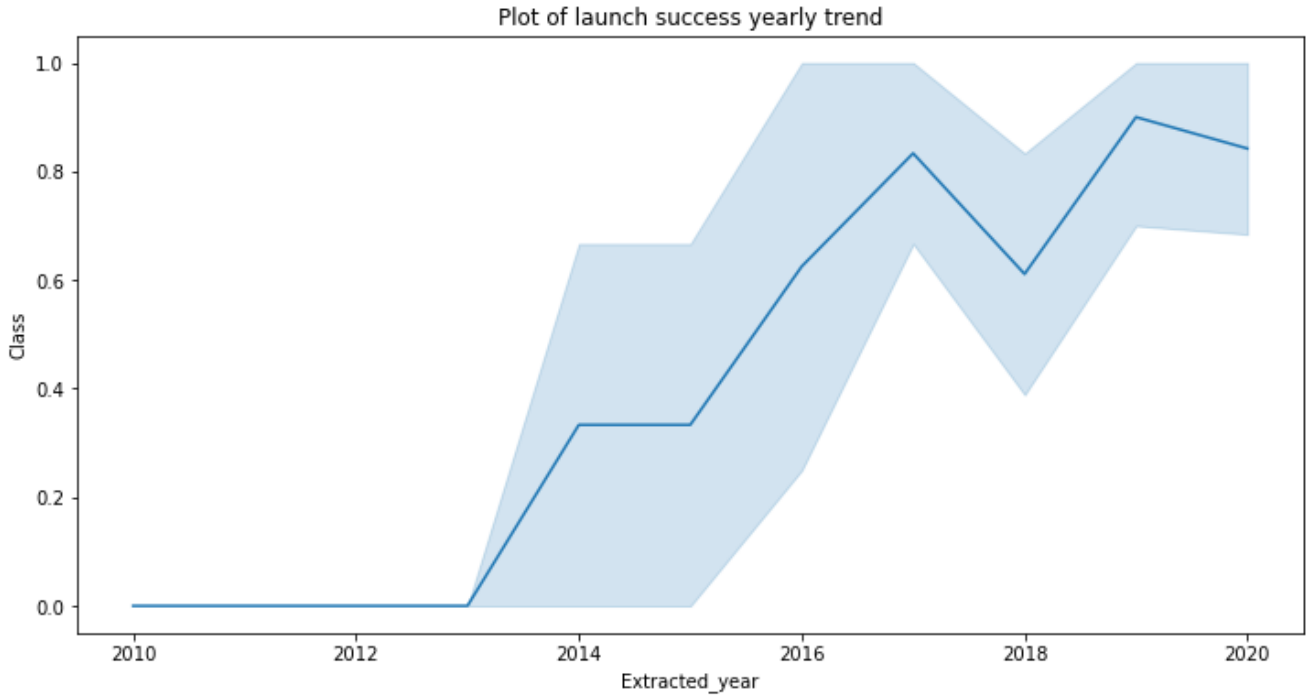
- The GTO orbit has mixed success with heavier payloads

⇒ with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



EDA - Launch Success Yearly Trend

- The success rate improved from 2013-2017 and 2018-2019
- The success rate decreased from 2017-2018 and from 2019-2020
- Overall, the success rate has improved since 2013 and kept on increasing till 2020



All Launch Site Names

Launch Site Names

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

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

In [10]:
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

CCAFS LC-40: Space Launch Complex 40 formerly Launch Complex 40 (LC-40) is an orbital launch pad located in northern Cape Canaveral , Florida (Wikipedia).

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

The total payload mass of boosters from NASA is 45596.

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

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

total_payloadmass
45596

Average Payload Mass by F9 v1.1

Using SELECT function to calculate the average payload mass carried by booster version F9 v1.1 as 2928.4

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

avg_payloadmass
2928.4

First Successful Ground Landing Date

We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

In [14]:
```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000

In [15]:

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

We used wildcard like '%' to filter for WHERE MissionOutcome was a success or a failure.

- 1 Failure in Flight
- 99 Success
- 1 Success (payload status unclear)

```
%sql SELECT MISSION_OUTCOME, COUNT(*) as total_number \
FROM SPACEXTBL \
GROUP BY MISSION_OUTCOME;
```

Mission_Outcome	Total (Success or failure)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

The booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.

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

In [17]:
```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

Using a combinations of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for **failed** landing outcomes in drone ship, their booster versions, and launch site names for year 2015

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

In [18]:
```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20.
- We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

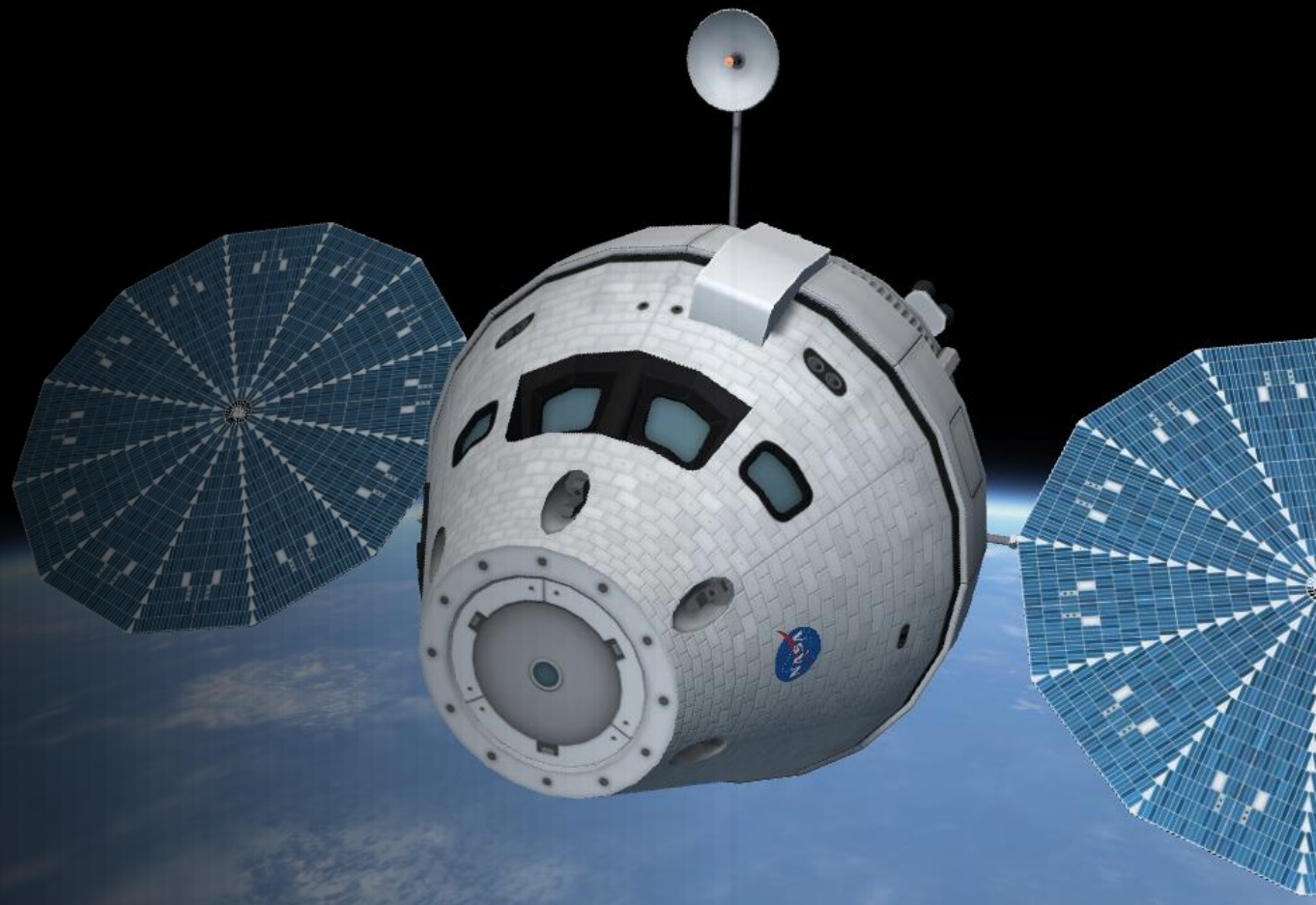
In [19]:

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

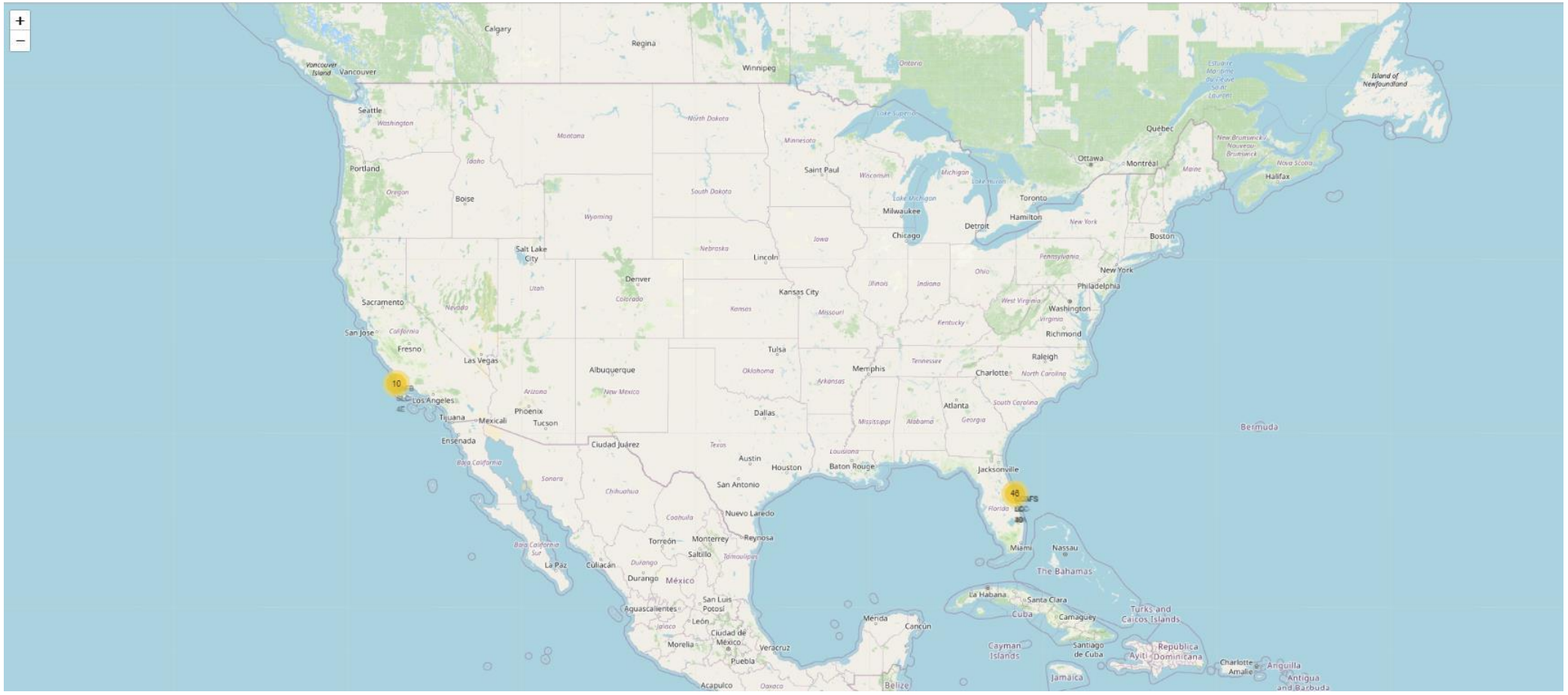
Section 3 Launch Sites Proximities Analysis



All Launch Sites Global Map Markers

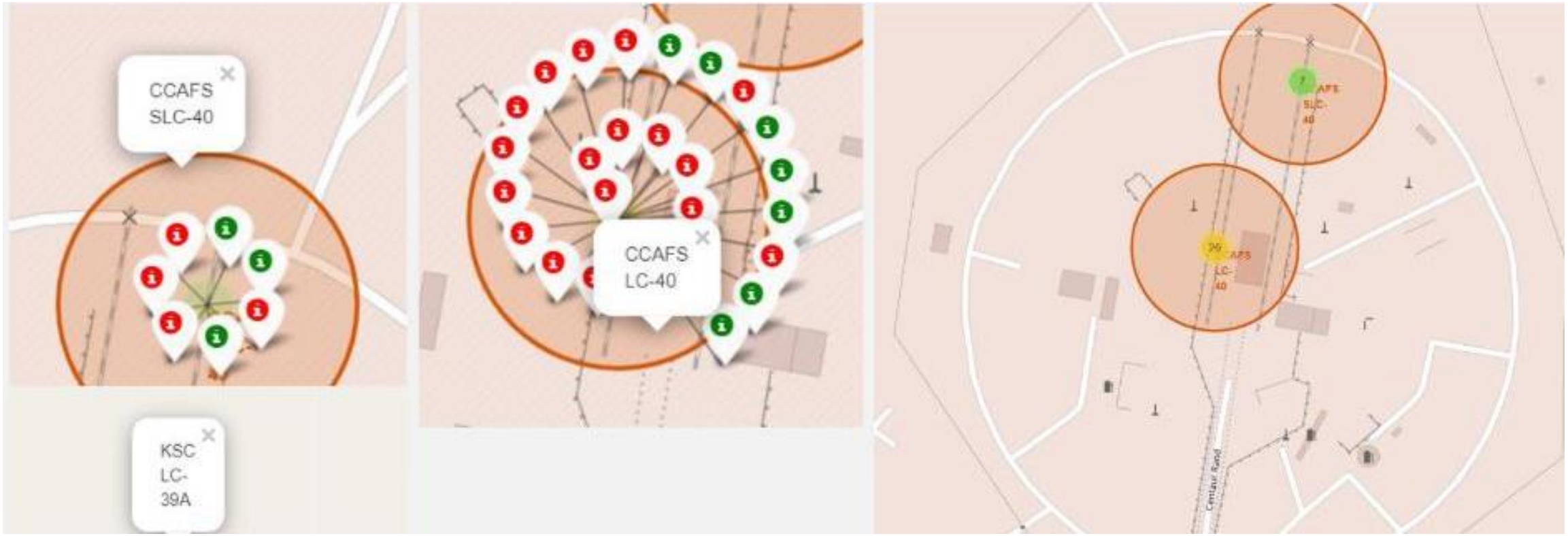
Near Equator: the closer the launch site to the equator, the easier it is to launch to equatorial orbit, and the more help you get from Earth's rotation for a prograde orbit.

Rockets launched from sites near the equator get an additional natural boost - due to the rotational speed of earth - that helps save the cost of putting in extra fuel and boosters.



Launch Outcomes

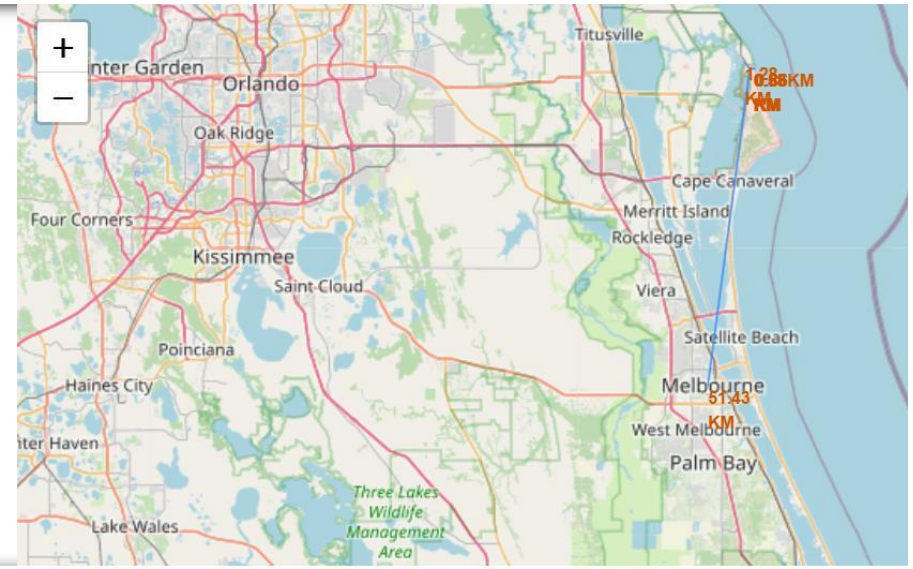
- **Green** markers for successful launches
- **Red** markers for unsuccessful launches
- Launch site **CCAFS SLC-40** has a **3/7 success rate** (**42.9%)**

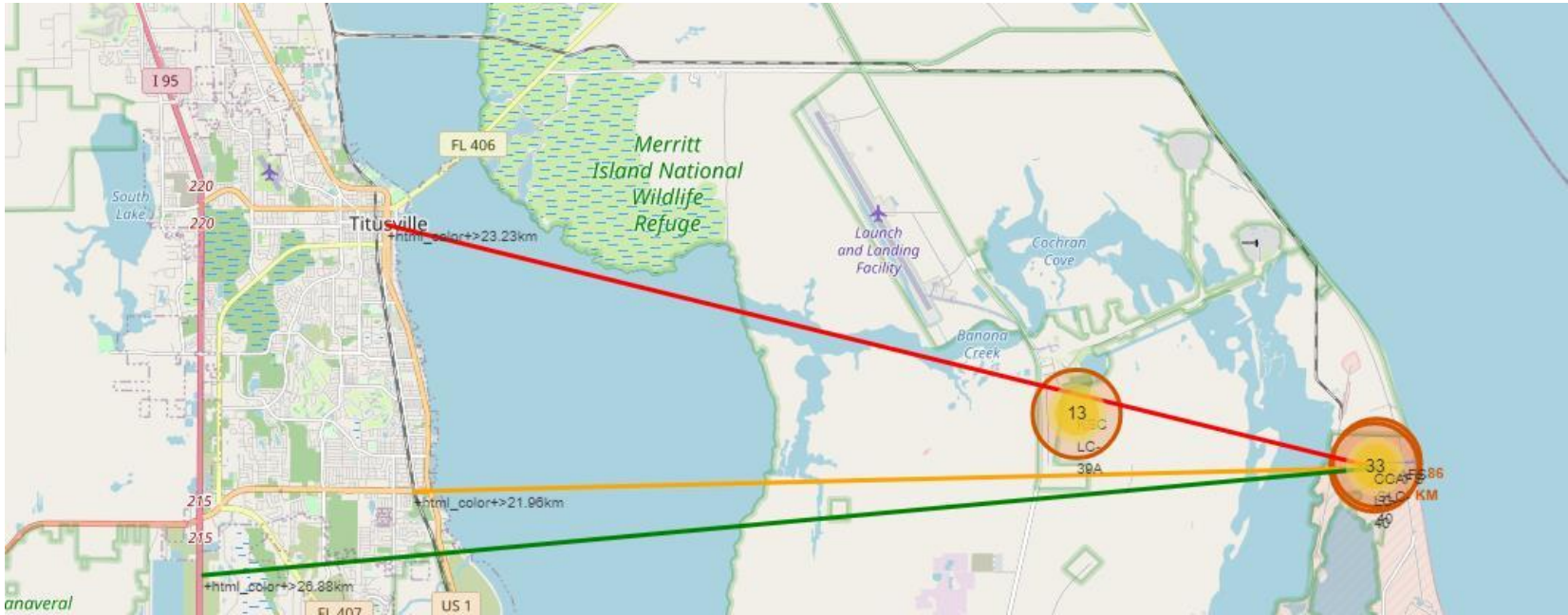


Launch Site Distance

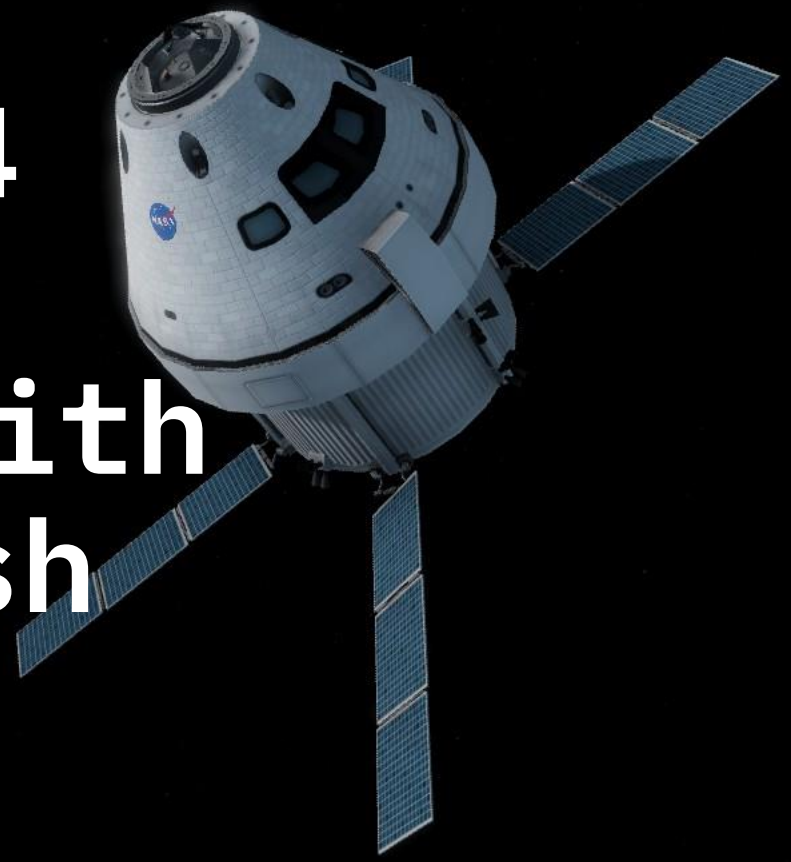
CCAFS SLC-40

- **Coasts**: help ensure that spent stages dropped along the launch path or failed launches don't fall on people or property.
- **Safety/ Security:** needs to be an exclusion zone around the launch site to keep unauthorized people away and keep people safe.
- **Transportation/Infrastructure and Cities**: need to be away from anything a failed launch can damage, but still close enough to roads/rails/docks to be able to bring people and material to or from it in support of launch activities.





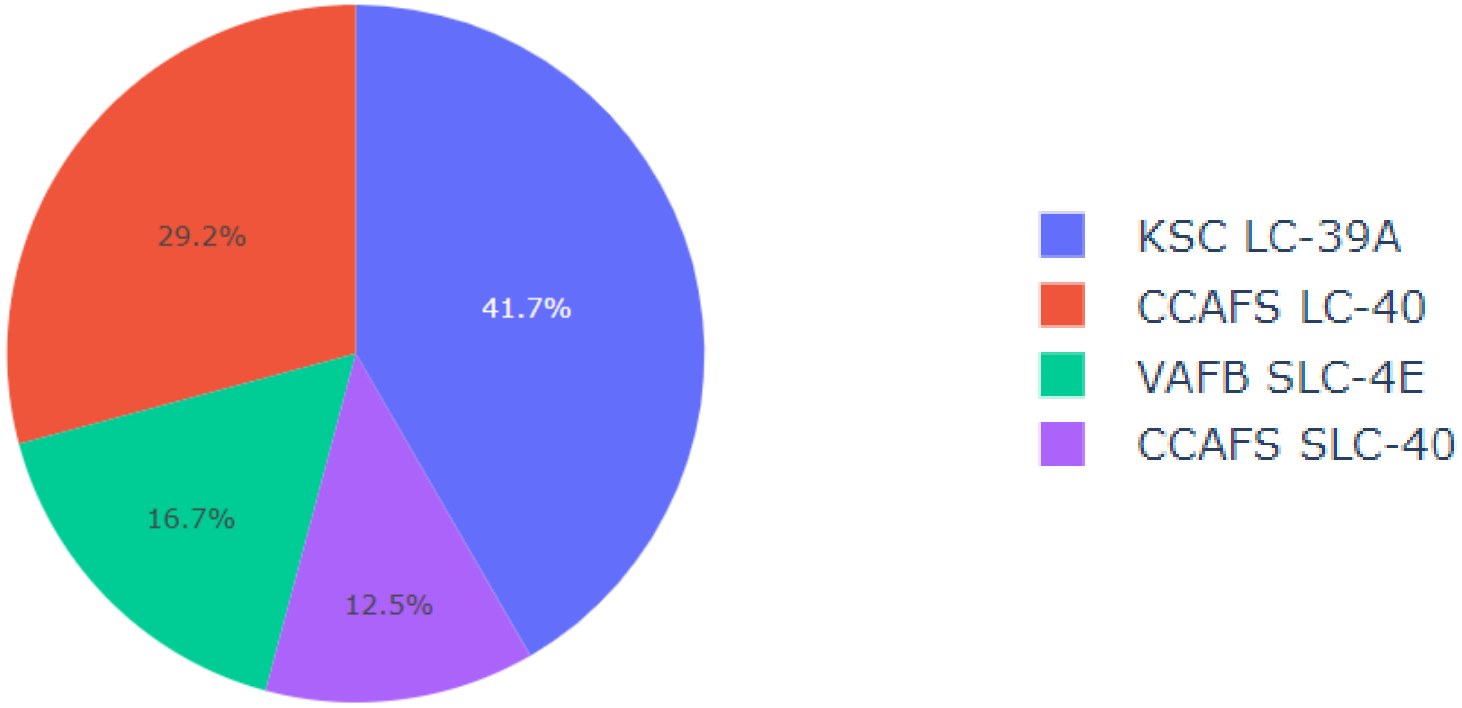
Section 4 Build a Dashboard with Plotly Dash



Launch Success by Site

Pie chart of all the launch sites

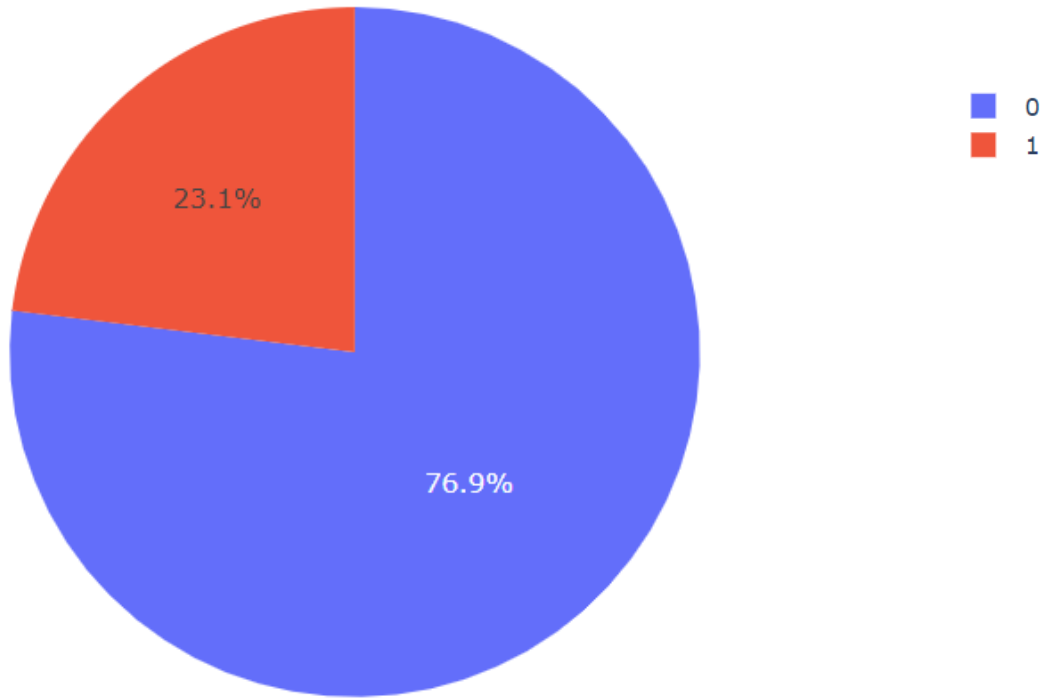
KSC LC-39A, KSC LC-39A has the most successful launches amongst launch sites, makes up 41.7% of the pie chart showing it has more launches than the other sites, followed by CCAFS LC-40 with 29.2%.



Launch Success – KSC LC – 29A

Pie chart of KSC LC 39A

- KSC LC-39A has the highest success rate amongst launch sites (76.9%)
- 10 successful launches and 3 failed launches



Payload vs. Launch Outcome scatter

By Booster Version

- Payloads between 2,000 kg and 5,000 kg have the highest success rate
- 1 indicating successful outcome and 0 indicating an unsuccessful outcome

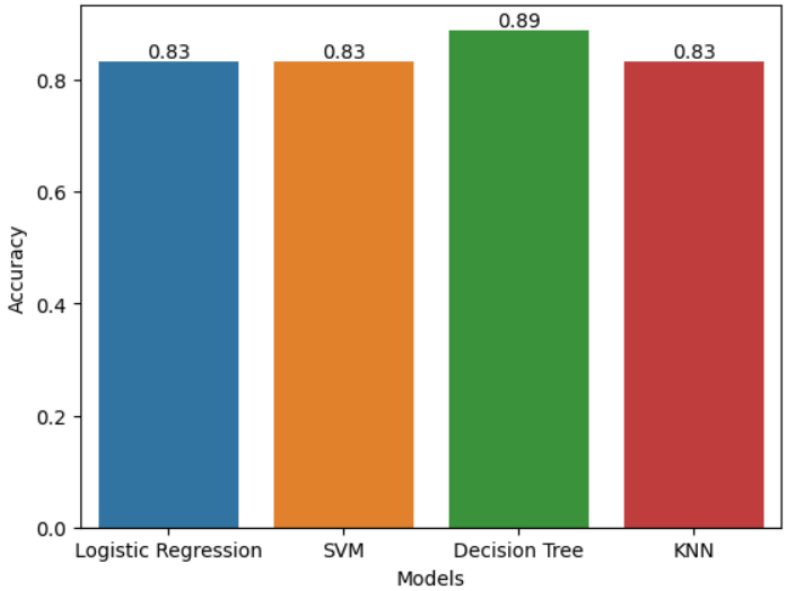
=> as payload mass grow so does the success rate for the booster version FT, while the opposite for booster version v1.1 which has more failures.

Section 5 Predictive Analysis (Classification)

Classification Accuracy

The decision tree classifier is the model with the highest classification accuracy with 88.88%

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.800000	0.800000	0.800000	0.800000
F1_Score	0.888889	0.888889	0.888889	0.888889
Accuracy	0.833333	0.833333	0.833333	0.833333



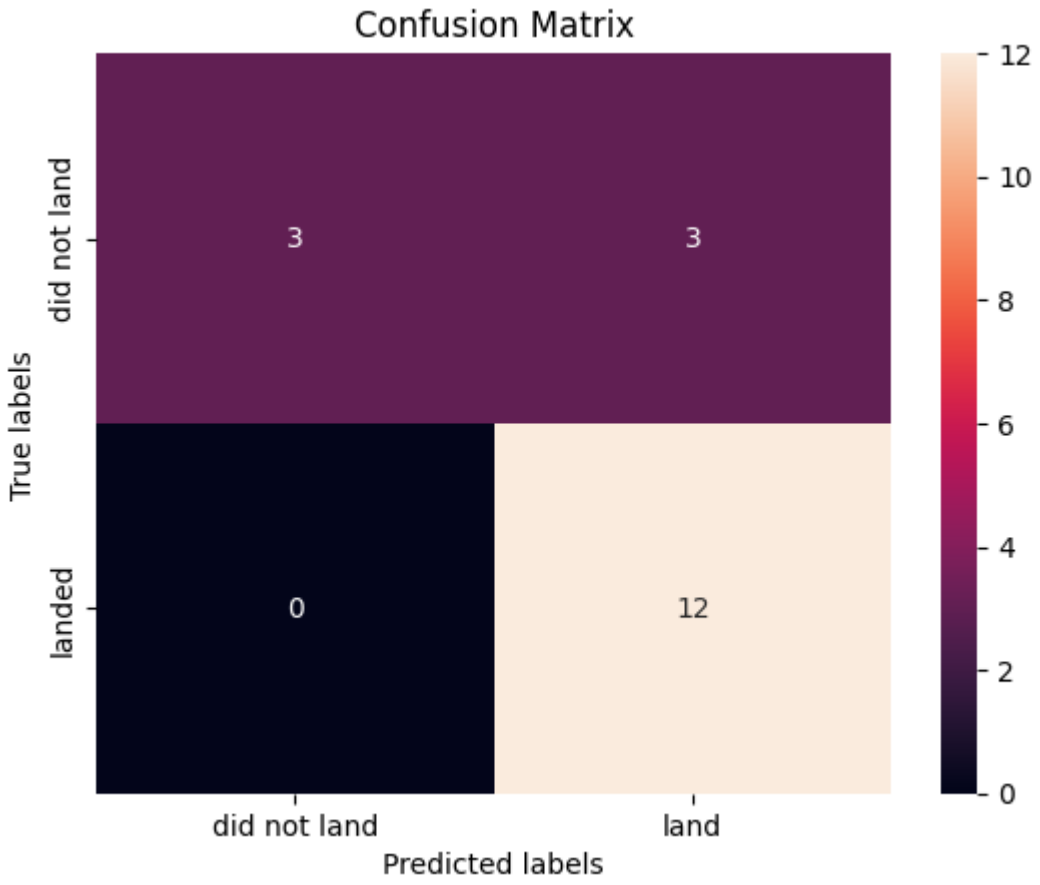

```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.9017857142857142
Best params is : {'criterion': 'gini', 'max_depth': 16, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
```

Confusion Matrix

The Decision tree model has 2 false-positive and no false negative and achieved a precision accuracy of 85.71% while recall is 100%



Performance Summary

- A confusion matrix summarizes the performance of a classification algorithm
- All the confusion matrices were identical
- The fact that there are false positives (Type 1 error) is not good

Confusion Matrix Outputs:

- **Precision** = TP / (TP + FP) = .80
- **Recall** = TP / (TP + FN) = 1
- **F1 Score** = 2 * (Precision * Recall) / (Precision + Recall) = .89
- **Accuracy** = (TP + TN) / (TP + TN + FP + FN) = .833

Conclusions

Model Performance: The models performed similarly on the test set with the decision tree model slightly outperforming

Equator: Most of the launch sites are near the equator for an additional natural boost - due to the rotational speed of earth - which helps save the cost of putting in extra fuel and boosters

Coast: All the launch sites are close to the coast

Successful rate:

- KSC LC-39A: Has the highest success rate among launch sites. Has a 100% success rate for launches less than 5,500 kg
- Orbits: ES-L1, GEO, HEO, and SSO have a 100% success rate
- Payload Mass: Across all launch sites, the higher the payload mass (kg), the higher the success rate

Appendix

All relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, and data sets included in this presentation can be found on my GitHub

Thank you!