SpaceX Falcon 9 first stage Landing Prediction

Content

Executive Summary

Summary of methodologies

SpaceX Data Collection using SpaceX API

SpaceX Data Collection with Web Scraping

SpaceX Data Wrangling

SpaceX Exploratory Data Analysis using SQL

Space-X EDA DataViz Using Python Pandas and Matplotlib

Space-X Launch Sites Analysis with Folium-Interactive Visual Analytics and Ploty Dash

SpaceX Machine Learning Landing Prediction

Summary of all results - EDA results

Interactive Visual Analytics and Dashboards Predictive Analysis(Classification)

Project Background and Context:

The Falcon 9 rocket, developed by SpaceX, has gained significant attention in the space industry. SpaceX stands out by offering Falcon 9 rocket launches at a cost of $62 million on its website, while other providers charge upwards of $165 million for similar services. The key factor contributing to this cost difference is SpaceX's ability to reuse the first stage of the Falcon 9 rocket. By reusing the first stage, SpaceX can significantly reduce the overall cost of a launch, making it a more cost-effective option compared to its competitors.

Problem Statement:

In this capstone project, our goal is to predict the successful landing of the Falcon 9 first stage. By analyzing the data available from Falcon 9 rocket launches advertised on SpaceX's website, we aim to determine if the first stage will land successfully. This prediction is crucial as it directly affects the cost of a launch. Knowing whether the first stage will land allows us to estimate the overall expenses involved in a launch. This information becomes particularly valuable if an alternate company wishes to bid against SpaceX for a rocket launch contract, enabling them to assess their competitiveness in terms of cost and reusability.
By addressing this problem, we hope to provide insights into the success rate of Falcon 9 first stage landings, which can inform decision-making processes for potential competitors and stakeholders in the space industry.

Methodology

Data Collection Methodology:

Detail the process of acquiring datasets, including sources, APIs, or any data collection mechanisms employed.

Specify the timeframe and frequency of data collection, ensuring clarity on data acquisition strategies.

Data Wrangling:

Outline the steps taken to clean and preprocess raw data.

Describe how missing values, outliers, and inconsistencies were addressed to ensure data integrity.

Data Processing:

Elaborate on the methods used to process and transform the data into a suitable format for analysis.

Highlight any standardization or normalization procedures applied to enhance data quality.

Exploratory Data Analysis (EDA):

Employ visualization techniques to analyze key patterns and trends within the dataset.

Utilize SQL queries to extract valuable insights, providing a comprehensive understanding of the data.

Methodology

Interactive Visual Analytics using Folium and Plotly Dash:

Illustrate the process of leveraging Folium for interactive maps and Plotly Dash for dynamic visualizations.

Demonstrate how these tools enhance the exploration and communication of complex data structures.

Predictive Analysis using Classification Models:

Detail the steps involved in building classification models.

Discuss the model tuning process, explaining how hyperparameters were adjusted for optimal performance.

Evaluate the models using appropriate metrics, considering factors such as precision, recall, and accuracy.

By providing comprehensive explanations for each step, this methodology ensures transparency and reproducibility in the data analysis process.

Data Collection

Collection of SpaceX Falcon 9 Data Description:

Initial data acquisition involved utilizing the SpaceX API, a RESTful API. A series of helper functions were defined to facilitate the extraction of information by using identification numbers within the launch data. A GET request was then made to the SpaceX API URL to retrieve rocket launch data.

To ensure consistency in the obtained JSON results, the SpaceX launch data was requested through a GET request, and subsequently, the response content was decoded as a JSON result. The decoded JSON was then transformed into a Pandas dataframe for ease of analysis and manipulation.

In addition to API-based data collection, web scraping techniques were employed to gather historical Falcon 9 launch records from a Wikipedia page titled "List of Falcon 9 and Falcon Heavy launches." The launch records were embedded in HTML format. Using the BeautifulSoup and requests libraries, the Falcon 9 launch HTML table records were extracted from the Wikipedia page. The extracted HTML table data was parsed and transformed into a Pandas dataframe for further analysis and integration with other collected data.

Data Collection – SpaceX API

Data from the SpaceX API, a RESTful API, was obtained through a GET request made to the SpaceX API. Subsequently, the SpaceX launch data was requested and parsed using this GET request. The response content was decoded as a JSON result and further converted into a Pandas data frame for comprehensive analysis.

For a detailed overview of the completed SpaceX API calls notebook, please refer to the GitHub URL: SpaceX Data Collection API Notebook.


Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

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

```
response.status_code
```

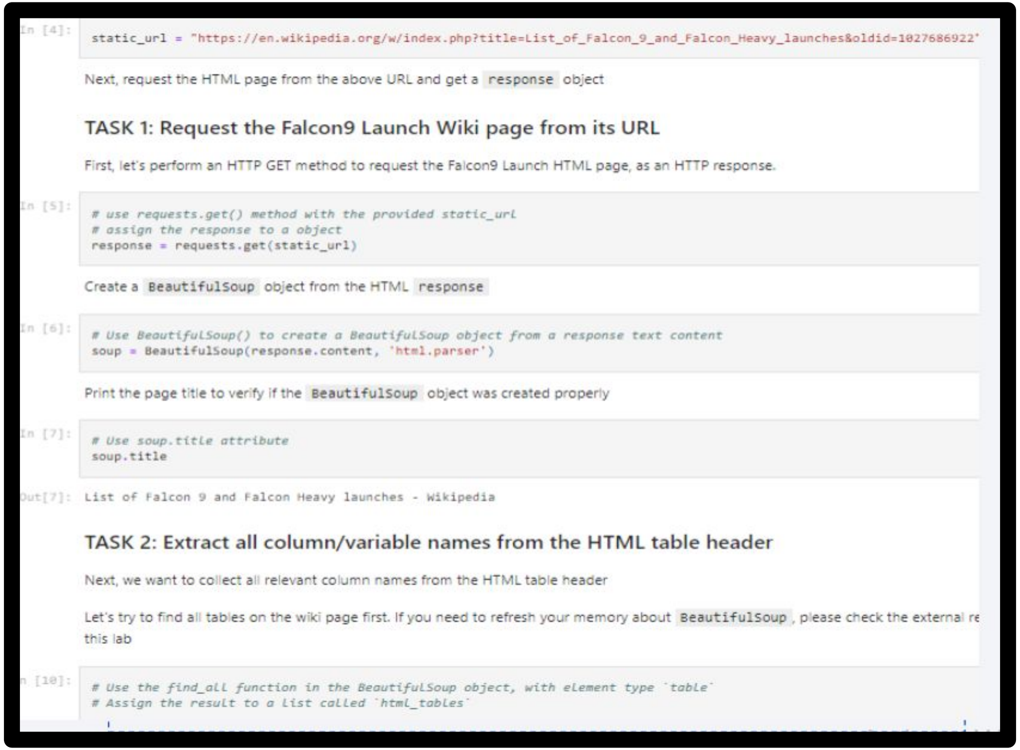
```
200
```

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

Data Collection – SpaceX API

Performed web scraping to collect Falcon 9 historical launch records from a Wikipedia using BeautifulSoup and request, to extract the Falcon 9 launch records from HTML table of the Wikipedia page, then created a data frame by parsing the launch HTML.



In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

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

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

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

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]: List of Falcon 9 and Falcon Heavy launches - Wikipedia

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

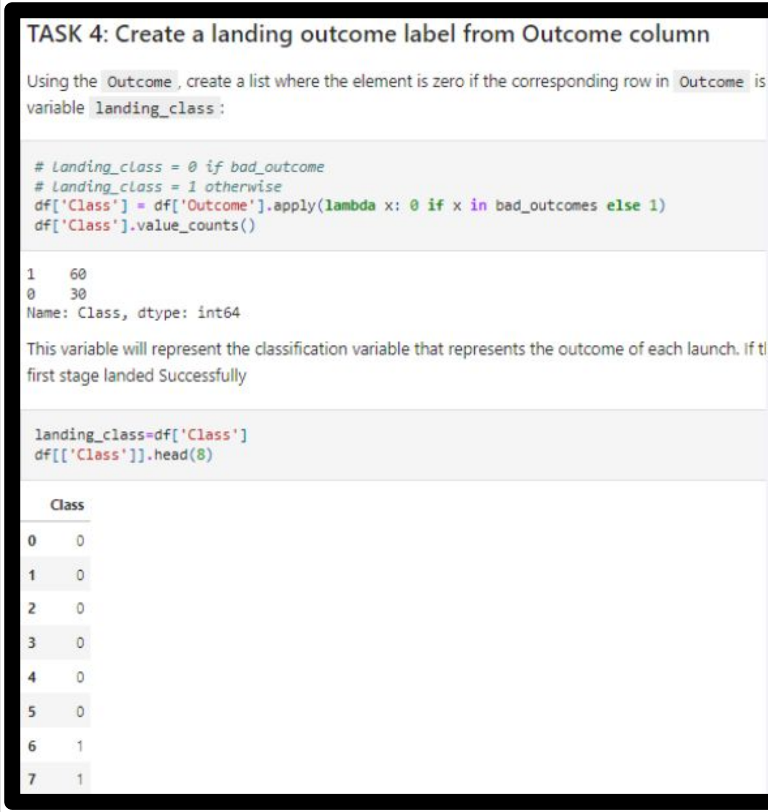
Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external re this lab

n [10]:
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
```

Data Wrangling

After collecting the data and creating a Pandas DataFrame, we filtered the data based on the "Booster Version" column to focus only on Falcon 9 launches. This allowed us to narrow down our analysis to the specific rocket type of interest. To ensure data quality, we addressed missing values in the "Landing Pad" and "Payload Mass" columns. For the "Payload Mass" column, we replaced the missing values with the mean value of the column. This approach helped us maintain the integrity of the dataset and ensure that missing data did not hinder our analysis.

In addition to data preprocessing, we performed Exploratory Data Analysis (EDA) to gain insights and identify patterns within the data. This involved examining various features and their relationships with the target variable. Through EDA, we aimed to determine the appropriate label to be used for training supervised models. By carefully analyzing the data, we identified patterns and correlations that could be used to predict the successful landing of the Falcon 9 first stage.



TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is
variable `landing_class`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

```
1    60
0    30
Name: Class, dtype: int64
```

This variable will represent the classification variable that represents the outcome of each launch. If tl
first stage landed Successfully

```
landing_class=df['Class']
df[['Class']].head(8)
```

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

EDA with Data Visualization

Exploratory Data Analysis (EDA): Explored the dataset to gain insights into its structure and characteristics.

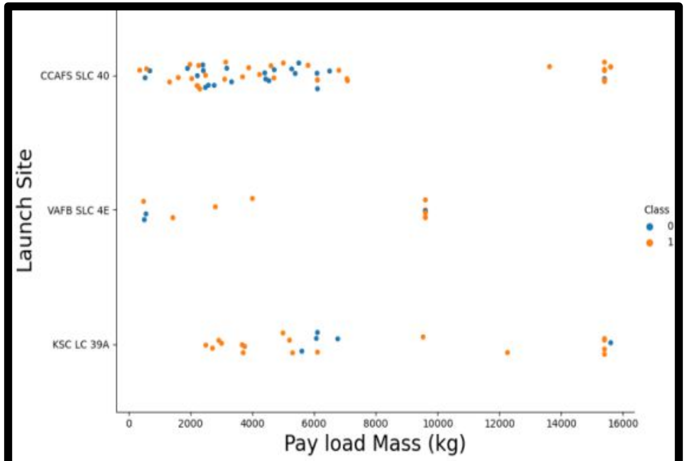
Preparing Data Feature Engineering: Engaged in feature engineering activities to enhance the dataset for analysis.

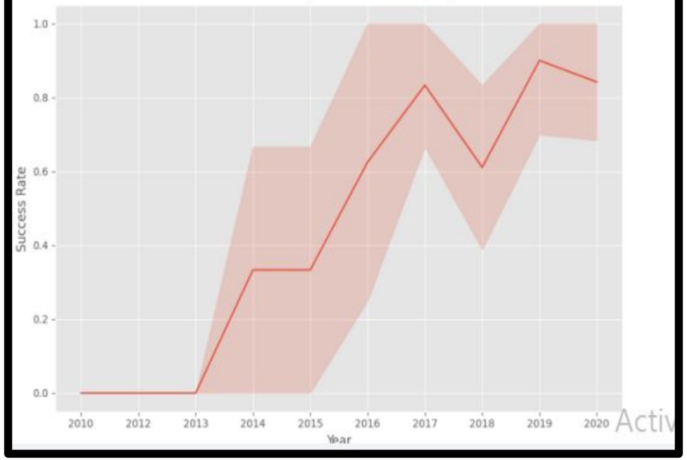
Scatter Plots: Utilized scatter plots to visually represent relationships, including Flight Number and Launch Site, Payload and Launch Site, Flight Number and Orbit Type, and Payload and Orbit Type.

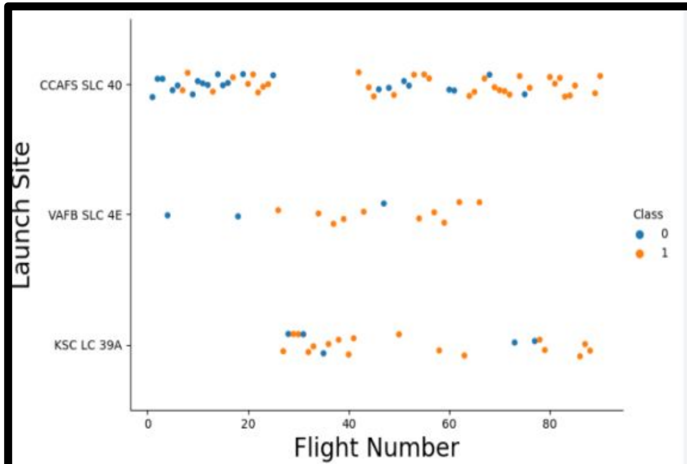
Bar Chart: Employed bar charts to illustrate the success rate of each orbit type, providing a clear visual comparison.

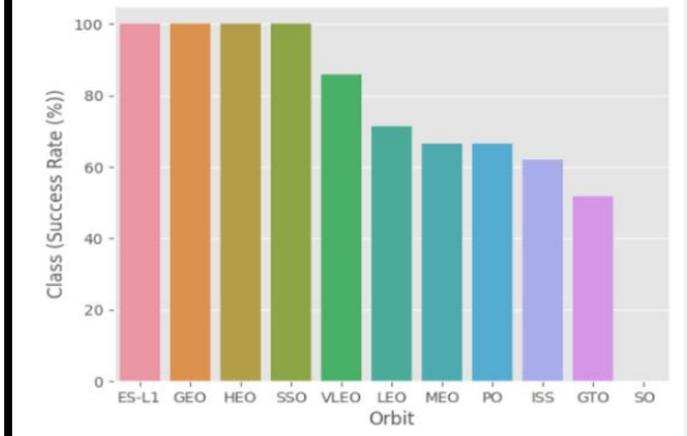
Line Plot: Leveraged line plots to visualize the yearly trend in launch success, offering a comprehensive view of the temporal patterns

EDA with Data Visualization (Plots Cont….)









EDA with SQL

```
# Extract and display the unique launch sites in the space mission

unique_launch_sites = df['LaunchSite'].unique()

print("Unique Launch Sites:")

print(unique_launch_sites)



# Retrieve and display 5 records where launch sites begin with the string 'CCA'


launch_sites_with_CCA =
df[df['LaunchSite'].str.startswith('CCA')].head(5)

print("\nRecords where Launch Sites begin with
'CCA':")

print(launch_sites_with_CCA)
```

```
# Calculate and display the total payload mass carried by boosters launched by NASA (CRS)

total_payload_mass_NASA_CRS =
df[df['Customer'].str.contains('NASA
CRS')]['PayloadMass'].sum()
print("\nTotal Payload Mass carried by boosters
launched by NASA (CRS):")
print(total_payload_mass_NASA_CRS)



# Compute and display the average payload mass carried by booster version F9 v1.1

average_payload_mass_F9_v1_1 =
df[df['BoosterVersion'] == 'F9
v1.1']['PayloadMass'].mean()
print("\nAverage Payload Mass carried by booster
version F9 v1.1:")
print(average_payload_mass_F9_v1_1)
```

EDA with SQL (Cont.…)

```
# List the date when the first successful landing outcome on a ground pad was achieved

first_successful_landing_date = df[df['LandingOutcome'].str.contains('Success') &
df['LandingPad'].notnull()]['Date'].min()

print("Date of the first successful landing on a ground pad:", first_successful_landing_date)



# List the names of boosters with success in drone ship, payload mass > 4000, and < 6000

successful_drone_ship_boosters = df[(df['LandingOutcome'] == 'Success (drone ship)') &

                                    (df['PayloadMass'] > 4000) & (df['PayloadMass'] <
6000)]['BoosterVersion'].unique()

print("\nBoosters with success in drone ship and payload mass between 4000 and 6000:")

print(successful_drone_ship_boosters)
```

```
# List the total number of successful and failure mission outcomes
total_successful_missions = df[df['MissionOutcome'] ==
'Success'].shape[0]
total_failure_missions = df[df['MissionOutcome'] ==
'Failure'].shape[0]
print("\nTotal Number of Successful Missions:",
total_successful_missions)
print("Total Number of Failure Missions:",
total_failure_missions)
```

Build an Interactive Map with Folium

Created folium map to marked all the launch sites, and created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site.

Created a launch set outcomes (failure=0 or success=1)

Build a Dashboard with Plotly Das

Built an interactive dashboard application with Plotly dash by:

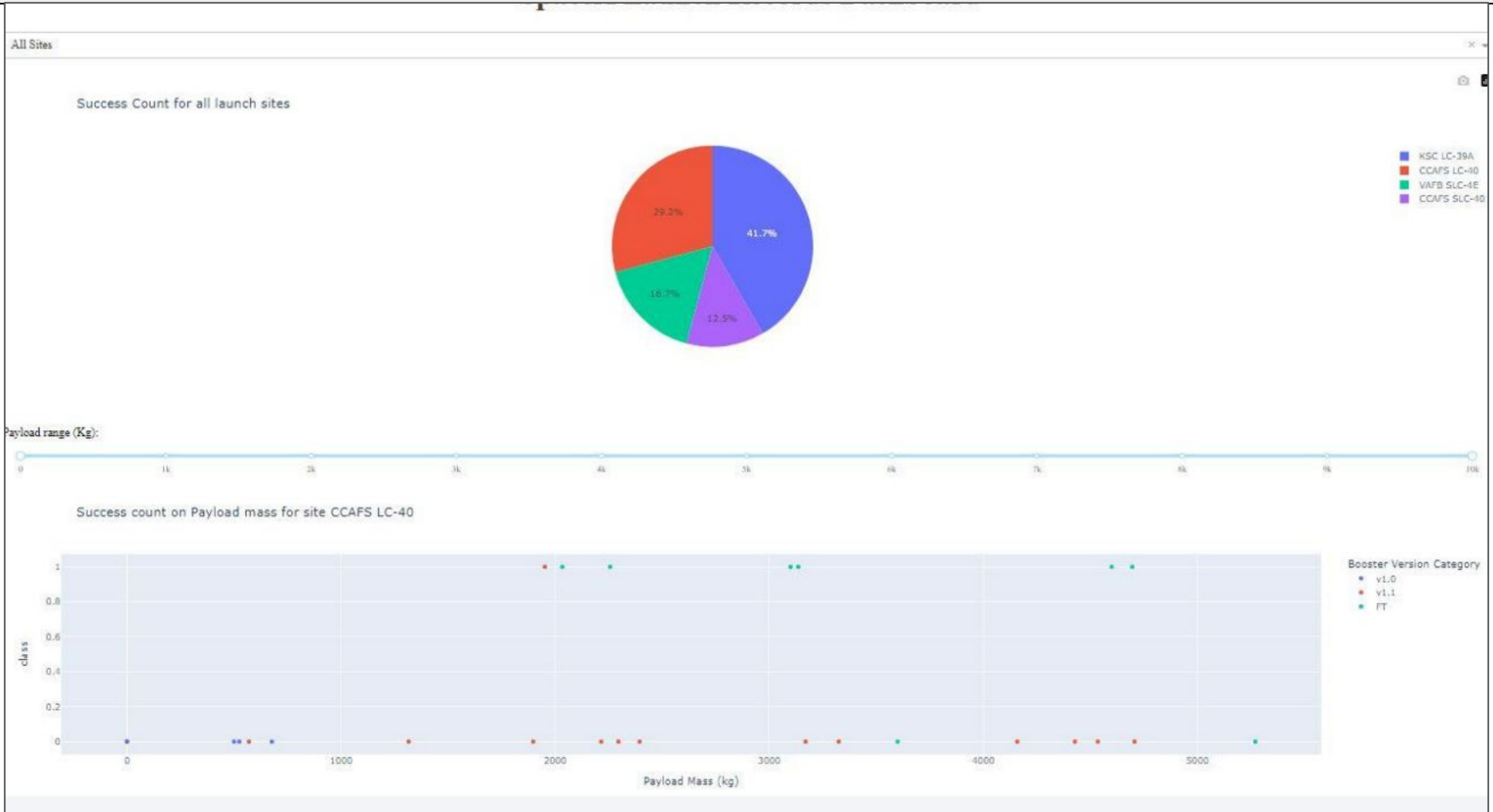
Adding a Launch Site Drop-down Input Component

Adding a callback function to render success-pie-chart based on selected site dropdown

Adding a Range Slider to Select Payload

Adding a callback function to render the success-payload-scatter-chart scatter plot

SpaceX Dash App



Predictive Analysis (Classification)

Data Loading and Exploration:

The data was loaded into a Pandas DataFrame for further analysis.

Exploratory Data Analysis (EDA) was conducted to gain insights into the dataset.

Determination of Training Labels:

The training labels (Y) were derived by creating a NumPy array from the 'Class' column using the to_numpy() method.

The resulting array was assigned to the variable Y as the outcome variable.

Feature Dataset Standardization:

The feature dataset (X) was standardized by applying the preprocessing.StandardScaler() function from the Sklearn library.

Standardization ensures that features are on a similar scale, avoiding dominance by variables with larger magnitudes.

Data Splitting:

The dataset was divided into training and testing sets using the train_test_split function fromsklearn.model_selection.

The test_size parameter was set to 0.2, and random_state was set to 2 for reproducibility.

Model Building and Evaluation:

Various classification models were implemented, and their performance was evaluated using metrics such as accuracy, precision, recall, and F1 score.

The models were trained on the training set and evaluated on the testing set.

Model Improvement:

Model hyperparameters were tuned to enhance performance.

Feature engineering techniques were explored to improve model robustness.

Selection of Best-Performing Model:

The model with the highest performance metrics on the testing set was selected as the best-performing classification model.

Predictive Analysis (Classification)

To identify the most effective machine learning model/method for performance on the test data among SVM, Classification Trees, k-nearest neighbors, and Logistic Regression:

Created an object for each algorithm, followed by the establishment of a GridSearchCV object for each model with predefined sets of parameters.

Conducted a thorough evaluation by creating GridSearchCV objects for each model, setting the cross-validation parameter (cv) to 10, and fitting the training data to identify the best hyperparameters.

After fitting the training set, retrieved the GridSearchCV object for each model, presenting the best parameters using the data attribute best_params_ and the accuracy on the validation data using best_score

The table below illustrates the accuracy scores on the test data for each method, facilitating a comparison to determine the best-performing model among SVM, Classification Trees, k-nearest neighbors, and Logistic Regression

Out[68]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

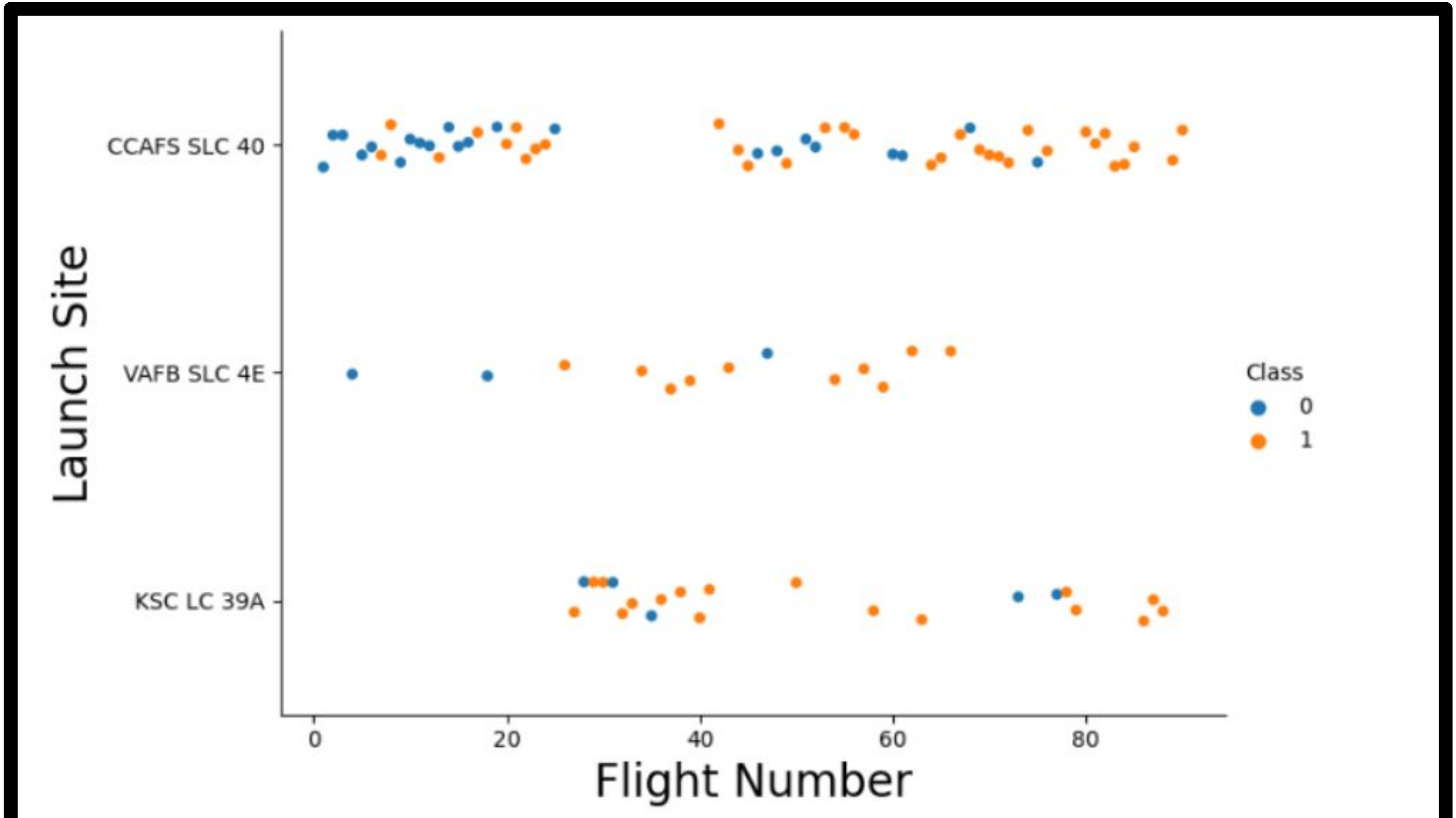
Results

Exploratory data analysis results

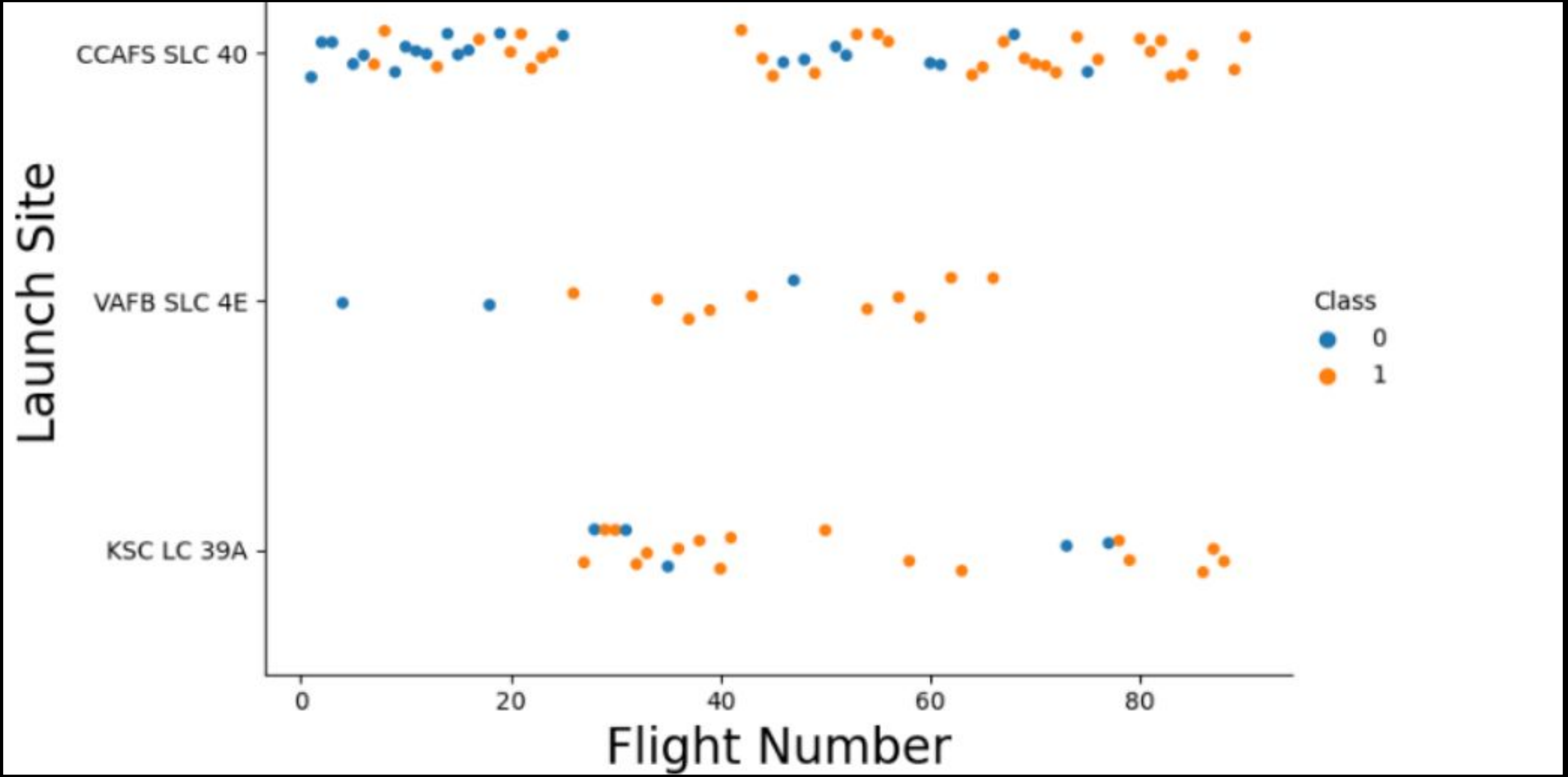
Interactive analytics demo in screenshots

Predictive analysis results 21 R

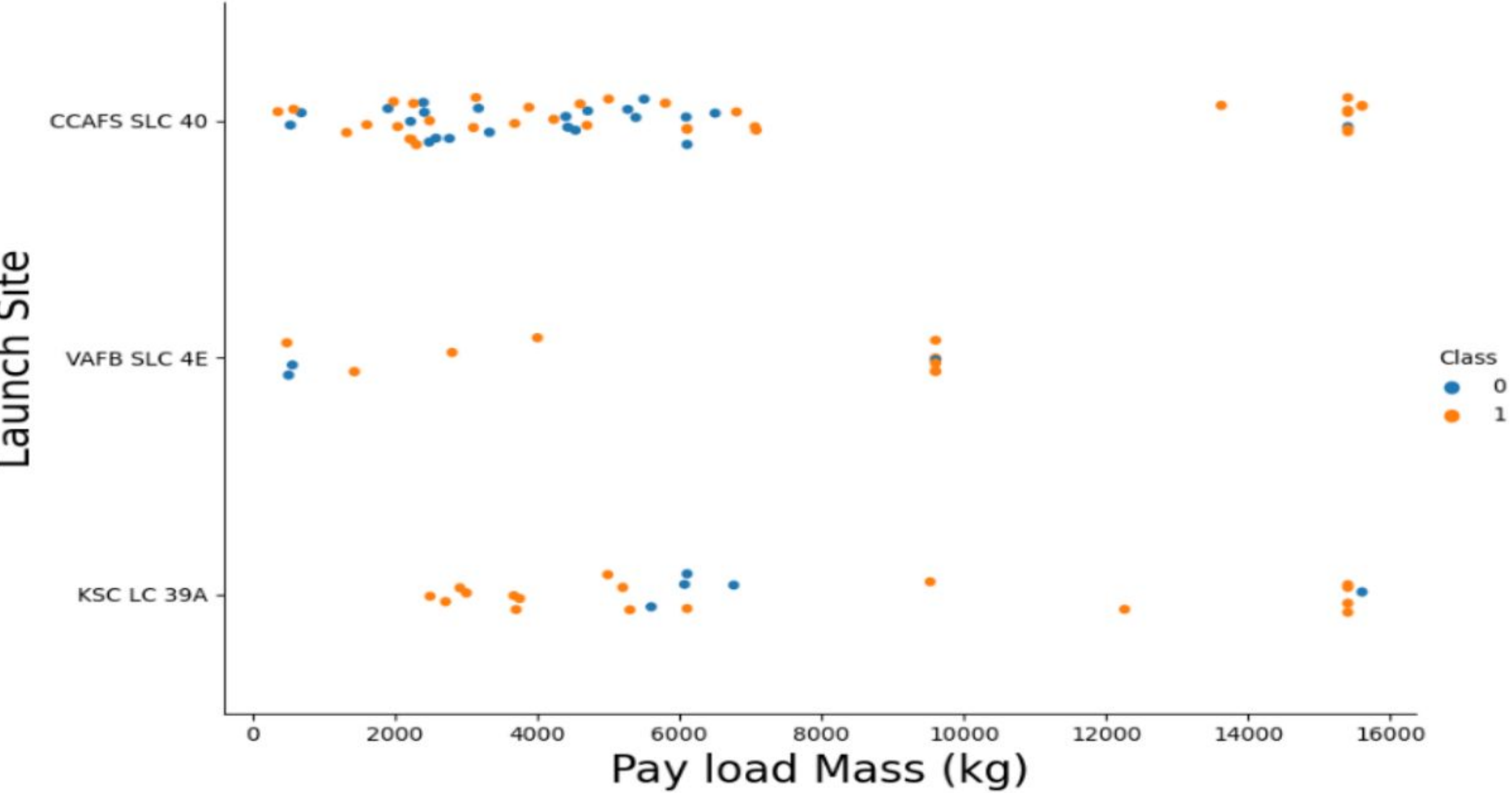
Flight Number vs. Launch Site



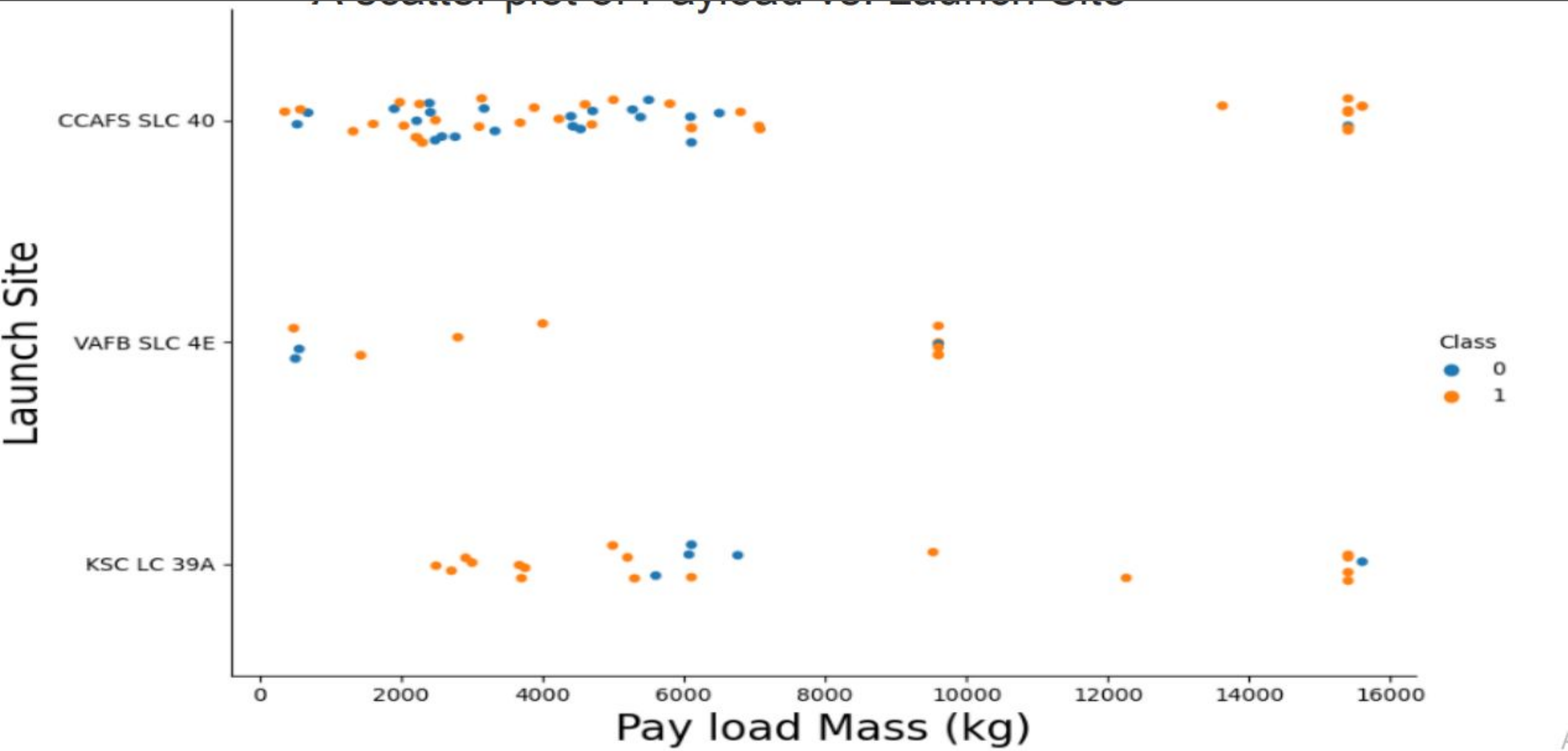
Flight Number vs. Launch Site with explanations



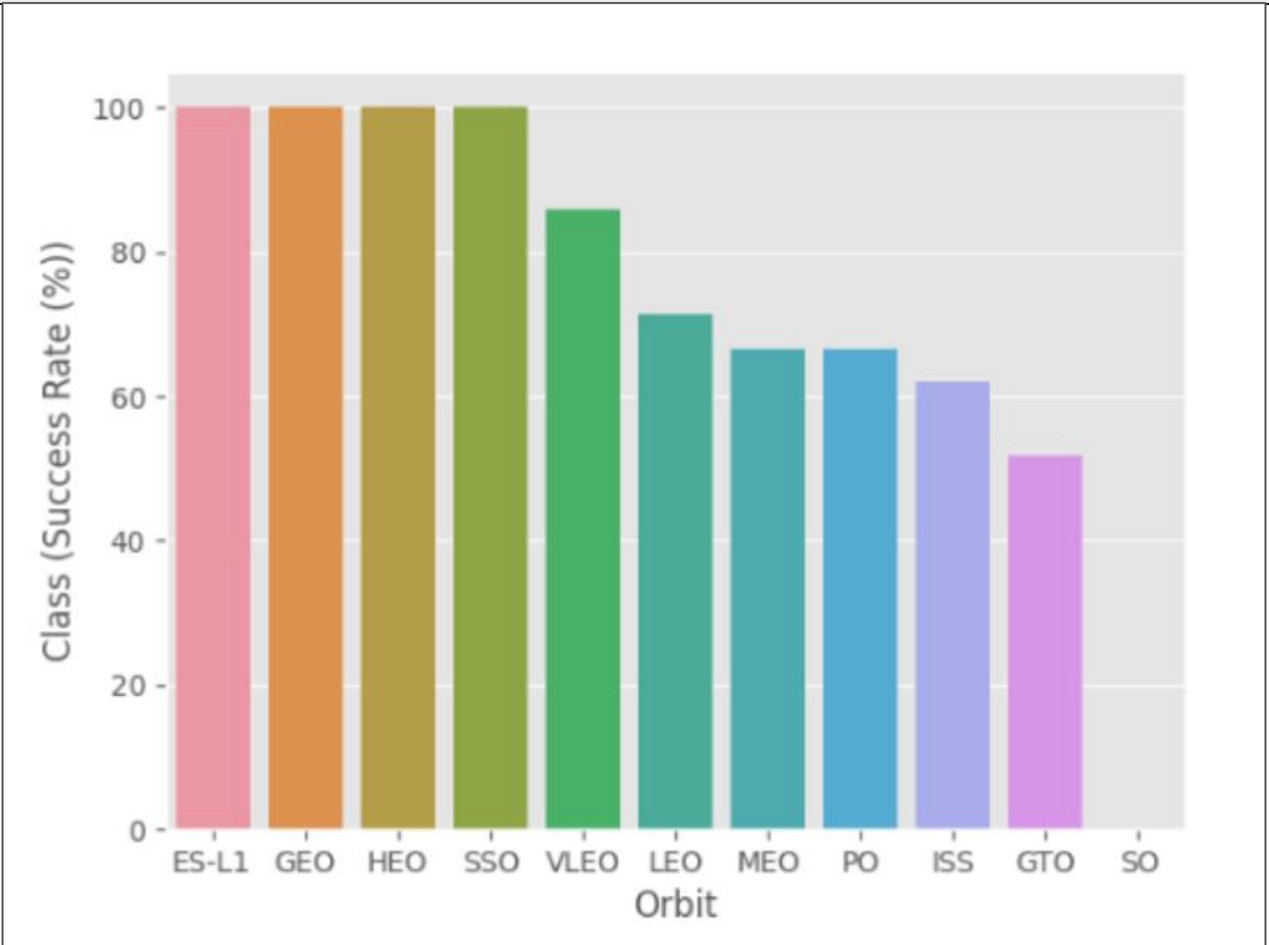
Payload vs. Launch Site



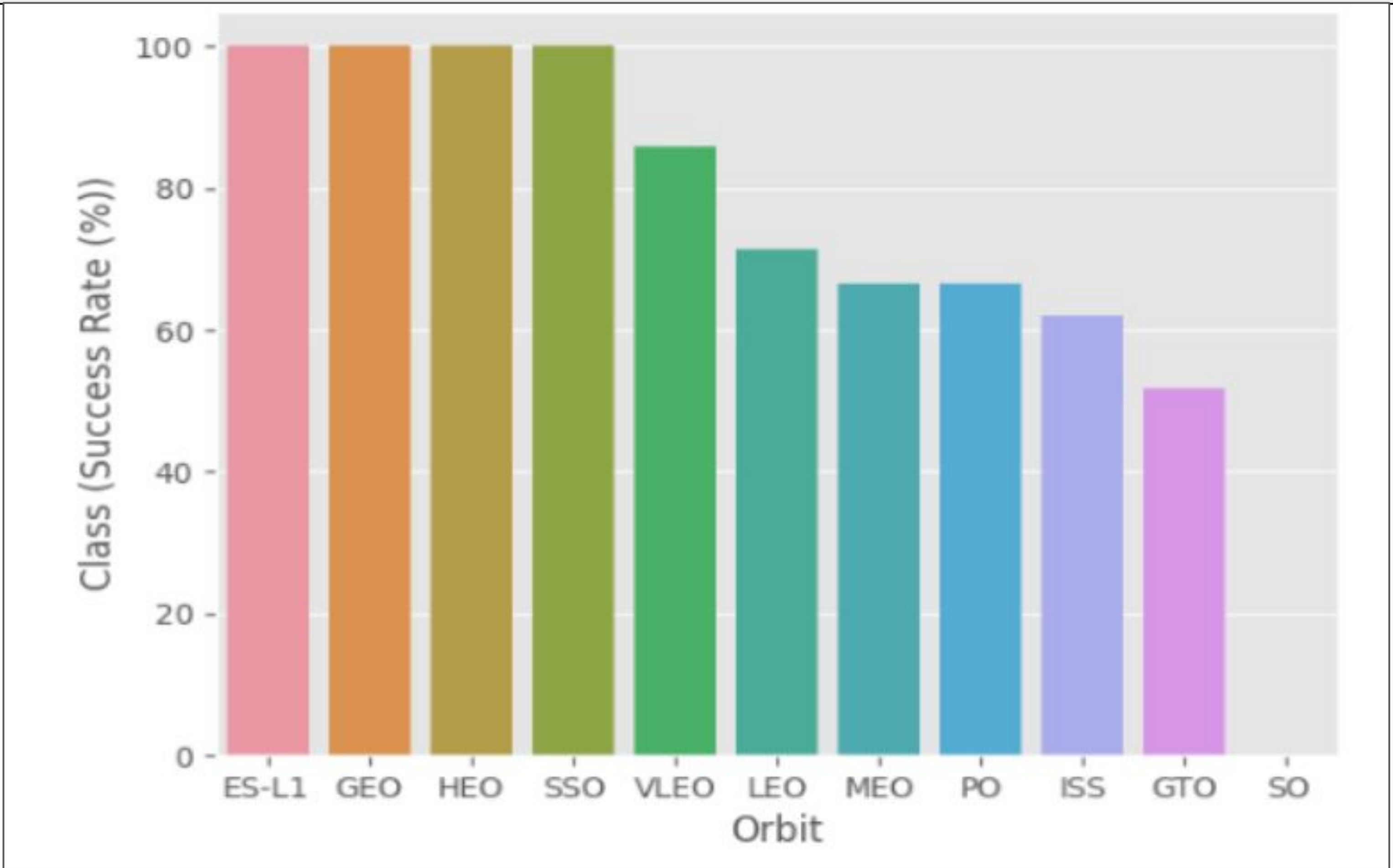
Payload vs. Launch Site with explanations



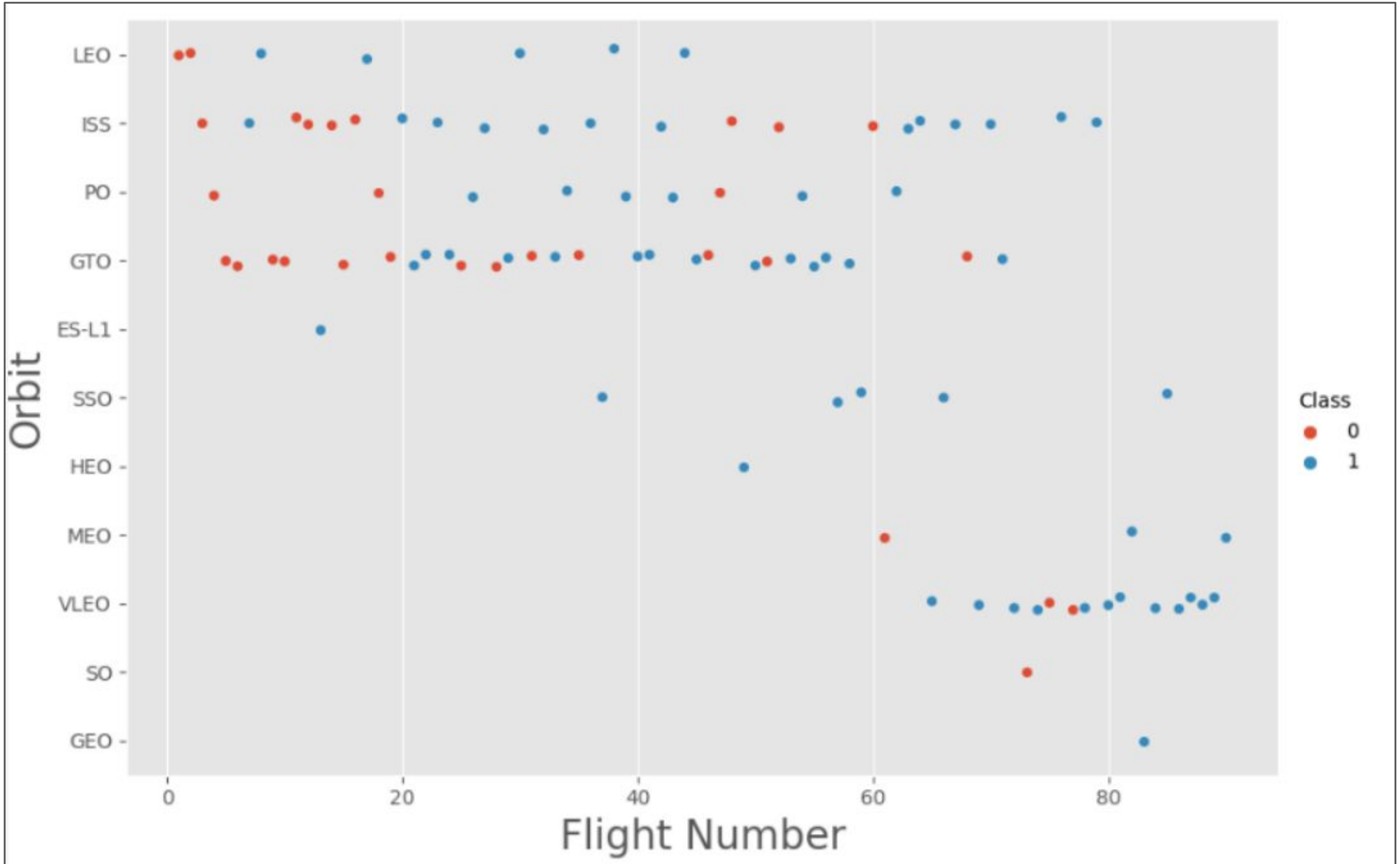
Success Rate vs. Orbit Type



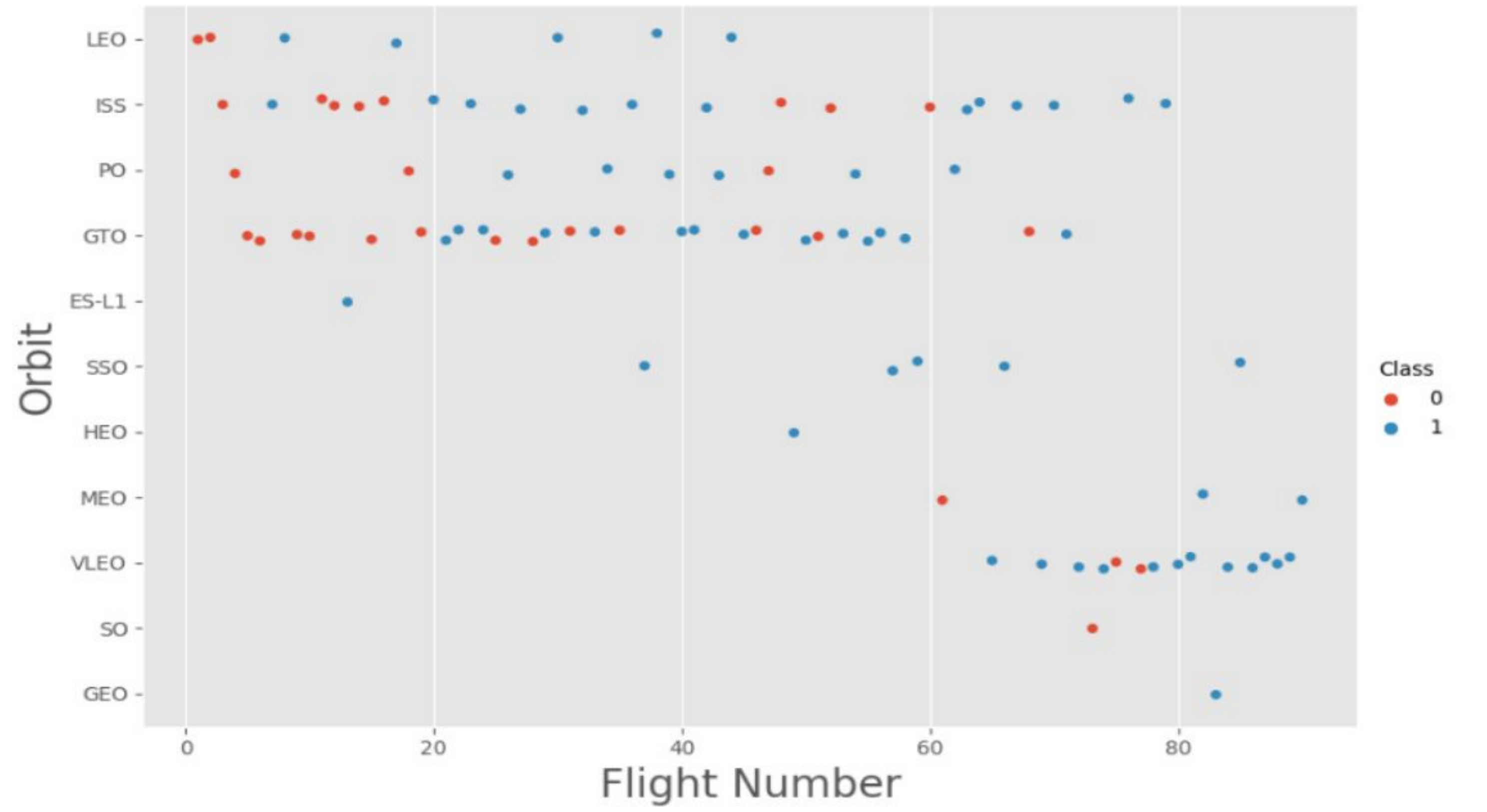
Success Rate vs. Orbit Type with explanations



Flight Number vs. Orbit Type



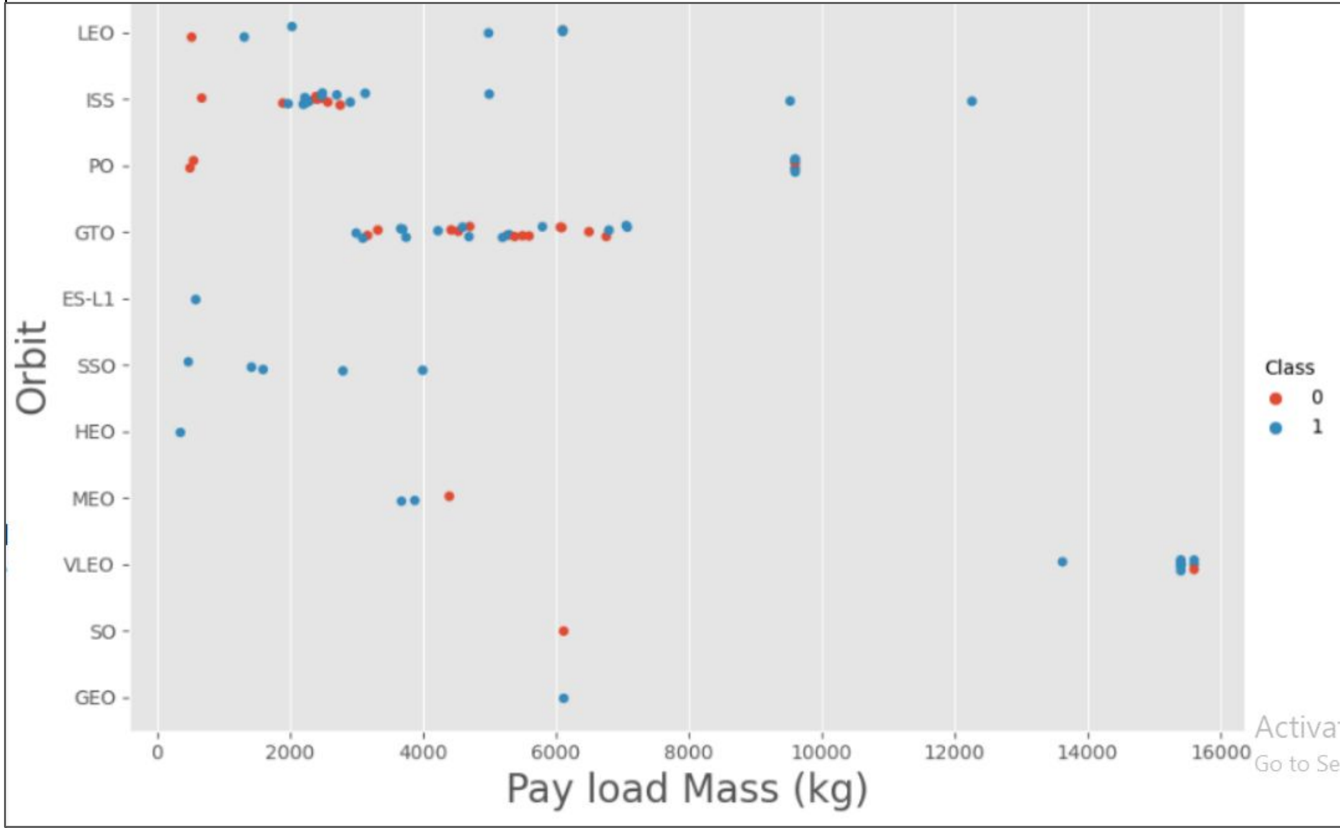
Flight Number vs. Orbit Types With explanations



You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

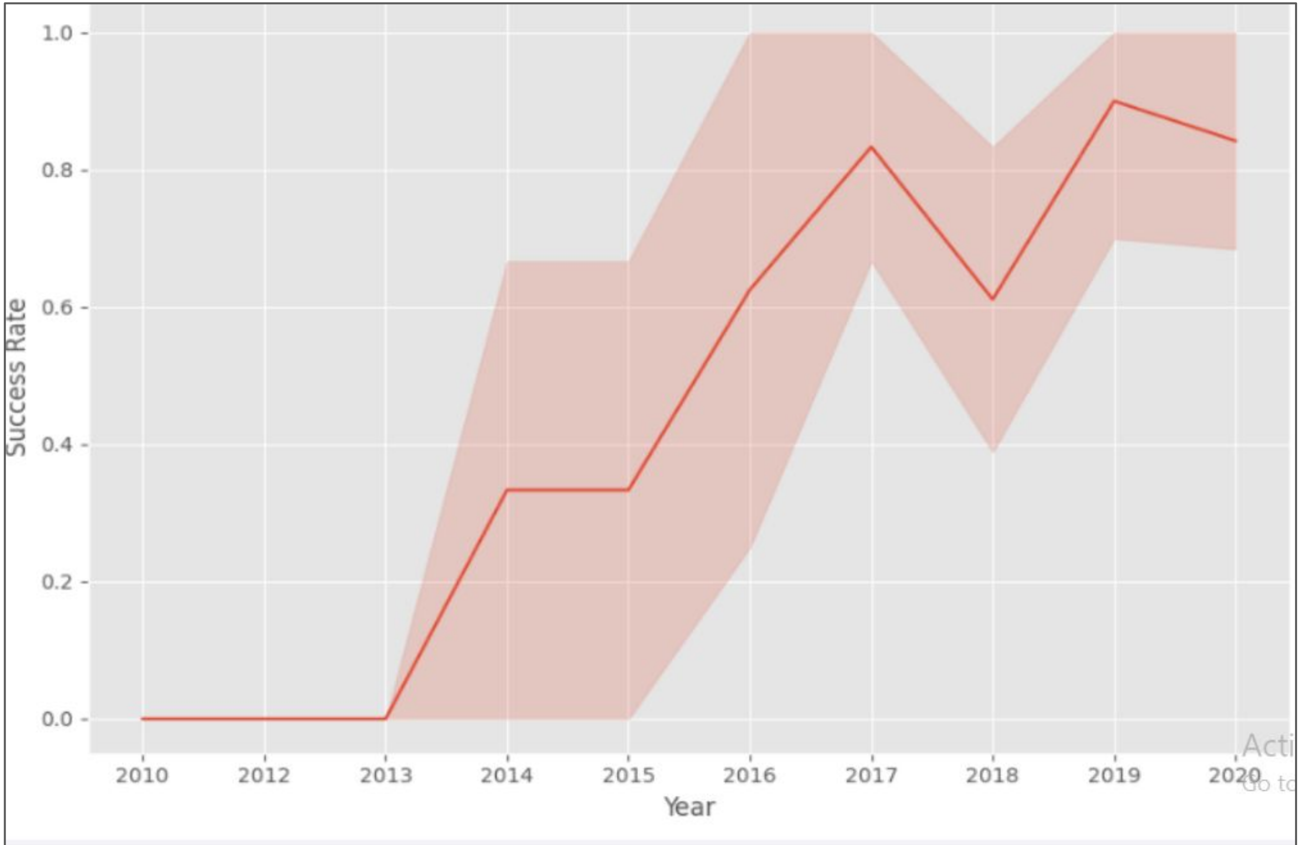
Payload vs. Orbit Type

With substantial payloads, the probability of successful or positive landings is notably higher for Polar, LEO, and ISS trajectories. However, in the case of GTO, distinguishing between positive landing rates and negative landing (unsuccessful missions) is challenging, as both outcomes have nearly equal probabilities.



Launch Success Yearly Trend

From 2013 to 2020, the success rate consistently increased.



All Launch Site Names

Retrieve the names of unique launch sites using the 'SELECT DISTINCT' statement, specifically targeting the 'LAUNCH_SITE' column in the SPACEX TBL table

Task 1

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

In [31]:

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

Out[31]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

n [72]:

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

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

ut[72]:

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

Total Payload Mass

Compute and showcase the total payload carried by boosters from NASA.

Task 3

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

In [17]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

Out[17]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

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

Task 4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

First Successful Ground Landing Date

Determine the dates of the initial successful landing outcomes on a ground pad.

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
```

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

MIN(DATE)
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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 * FROM 'SPACEXTBL'
```

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOA
```

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

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

Compute the total count of successful and unsuccessful mission outcomes.

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

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

Boosters Carried Maximum Payload

```
%sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL)
```

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

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

2015 Launch Records

Provide a list of unsuccessful landing outcomes on drone ships in 2015, including their corresponding booster versions and launch site names.

Task 9

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

```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome"
```

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

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing _Outcome
2015	01	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
2015	04	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	Failure (drone ship)

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

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
19-02-2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-10-2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
18-08-2020	14:31:00	F9 B5 B1049.6	CCAFS SLC-40	Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 1B	15440	LEO	SpaceX, Planet Labs, PlanetIQ	Success	Success
18-07-2016	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-04-2018	22:51:00	F9 B4 B1045.1	CCAFS SLC-40	Transiting Exoplanet Survey Satellite (TESS)	362	HEO	NASA (LSP)	Success	Success (drone ship)

Markers of all launch sites on global map

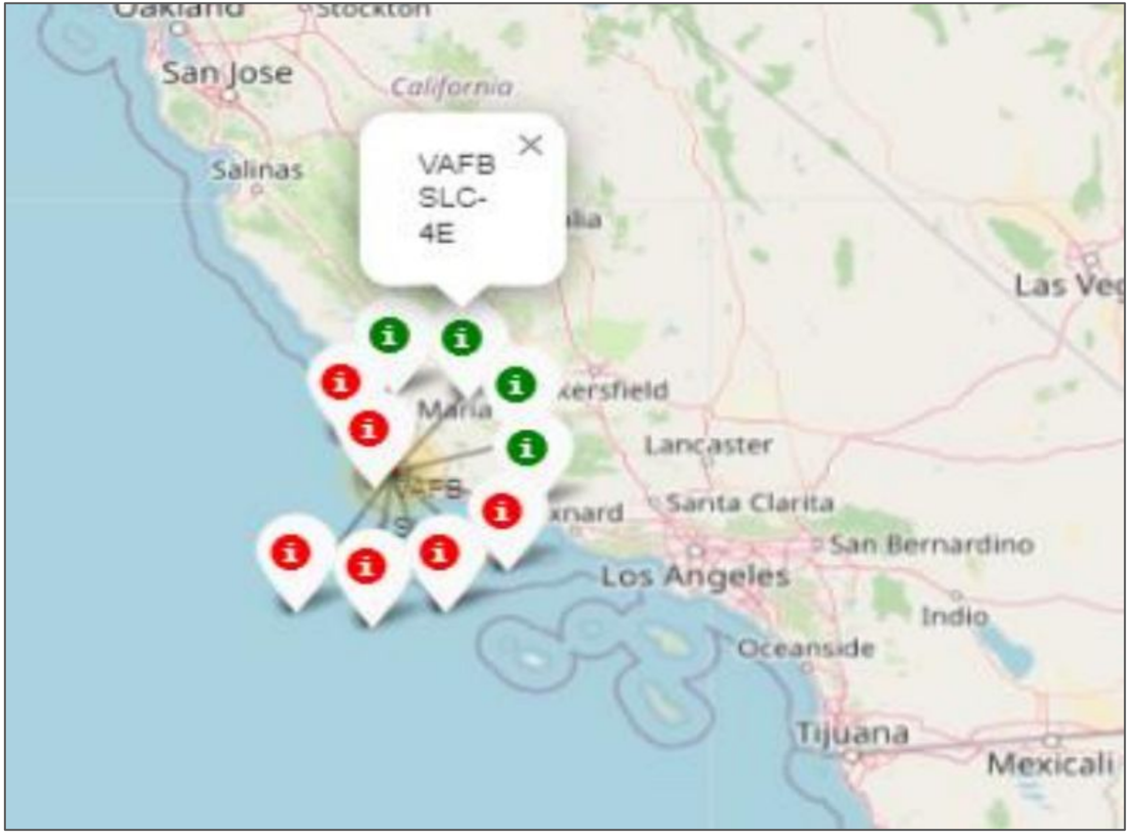
All launch sites are situated near the Equator, positioned southward on the US map. Additionally, each launch site is in close proximity to the coast.

Launch outcomes for each site on the map With Color Markers

On the eastern coast in Florida, Launch site KSC LC-39A exhibits comparatively higher success rates in contrast to CCAFS SLC-40 and CCAFS LC-40.

Launch outcomes for each site on the map With Color Markers

On the West Coast (California), Launch site VAFB SLC-4E demonstrates a relatively lower success rate of 4/10, compared to the KSC LC-39A launch site on the Eastern Coast of Florida.

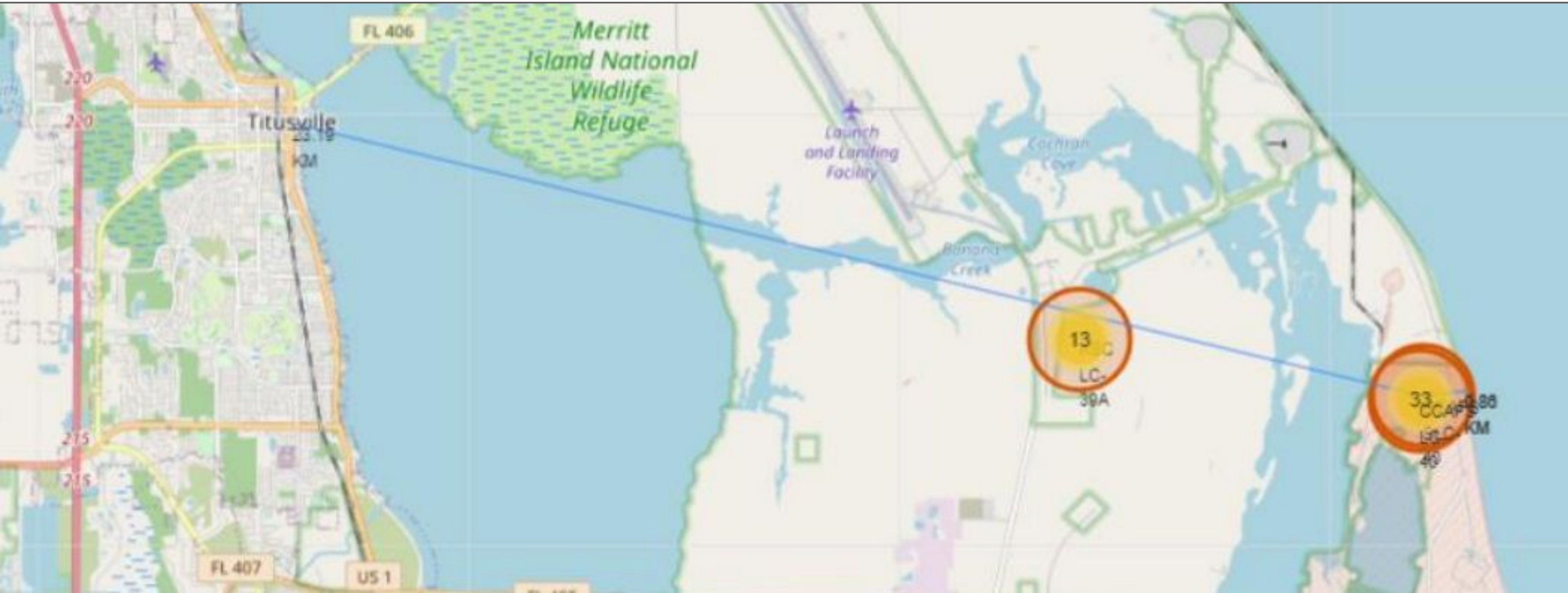


Distances between a launch site to its proximities

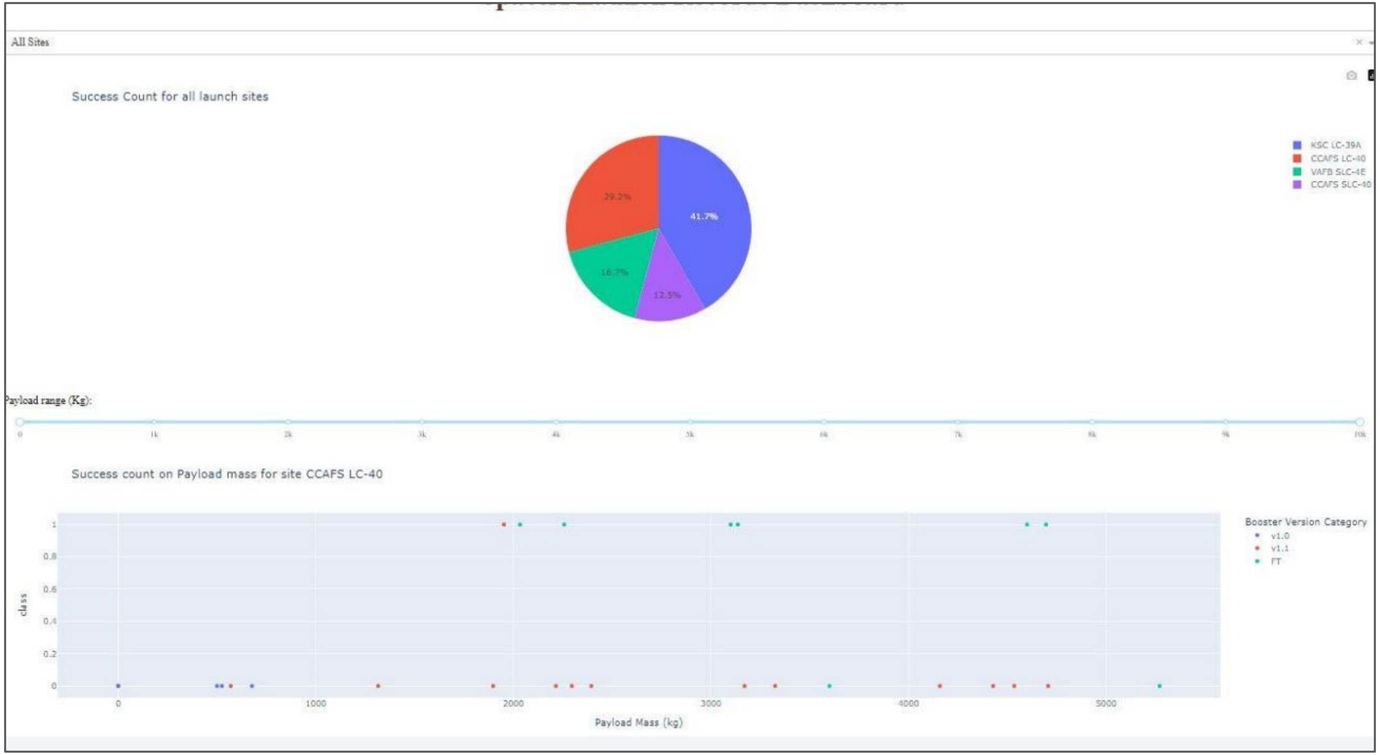
Launch site CCAFS SLC-40 proximity to coastline is 0.86km

Distances between a launch site to its proximities

Launch site CCAFS SLC-40 closest to highway (Washington Avenue) is 23.19km

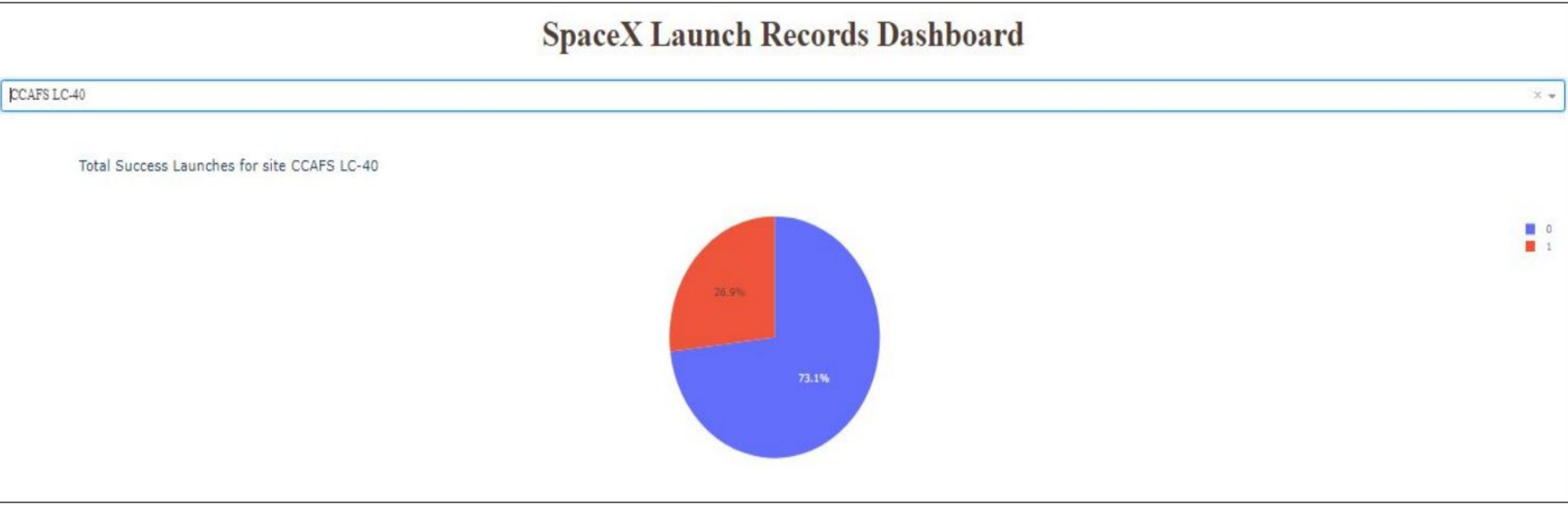


Pie-Chart for launch success count for all sites



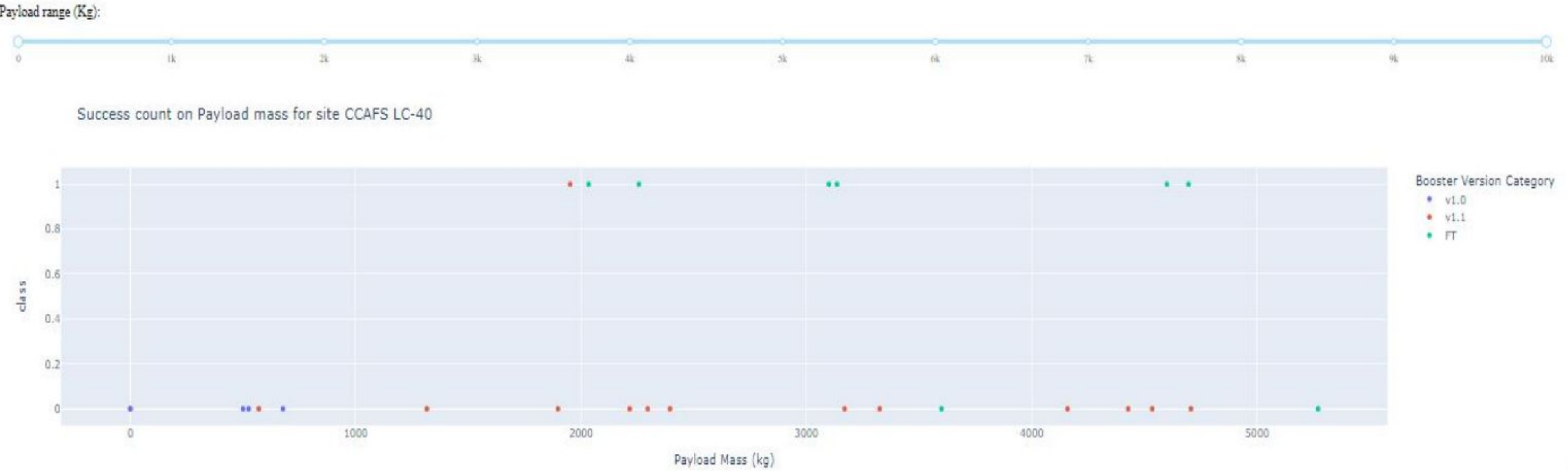
The launch site KSC LC-39A boasts the highest launch success rate at 42%, followed by CCAFS LC-40 at 29%, VAFB SLC-4E at 17%, and lastly, launch site CCAFS SLC-40 with a success rate of 13%.

Pie chart for the launch site with 2 nd highest launch success ratio



Launch site CCAFS LC-40 achieved the second-highest success ratio, with a 73% success rate compared to 27% for failed launches.

Payload vs. Launch Outcome scatter plot for all sites



For Launch site CCAFS LC-40, the booster version FT exhibits the highest success rate when carrying a payload mass greater than 2000kg.

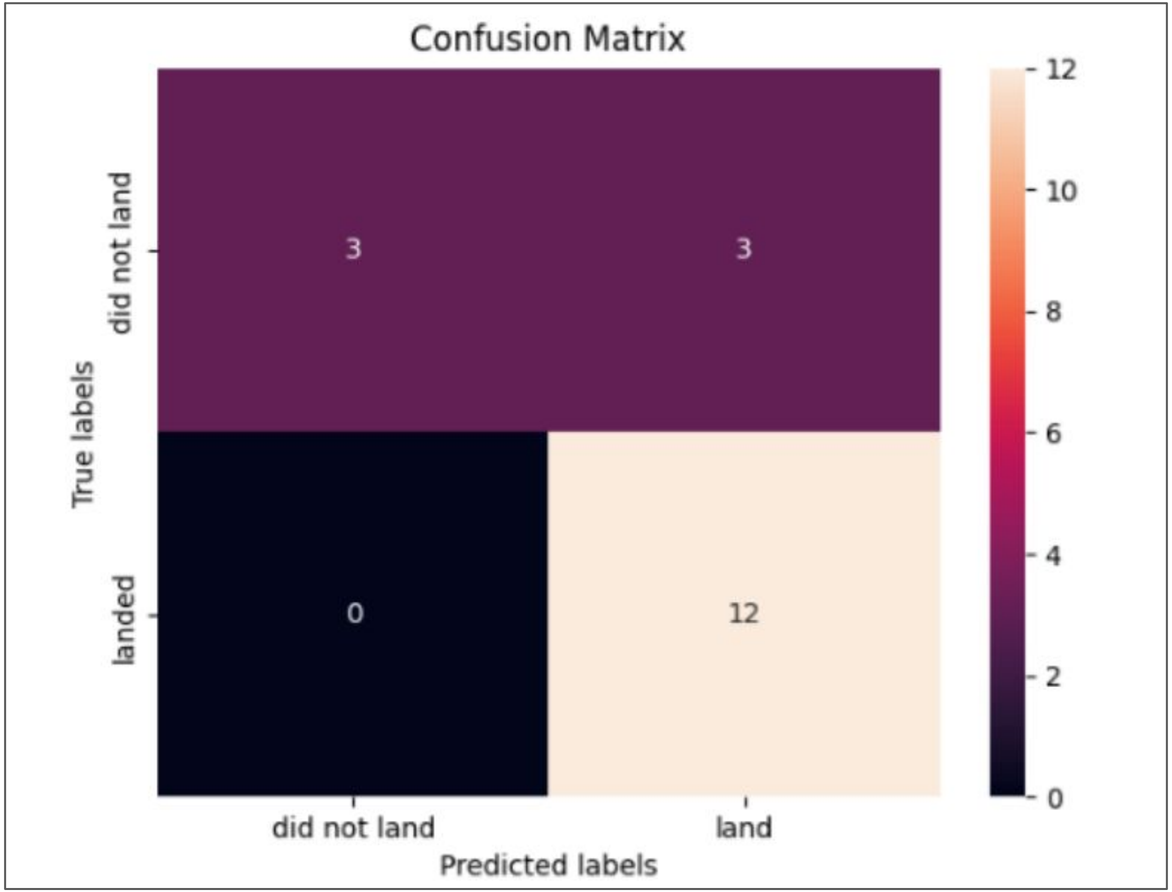
Classification Models Accuracy

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

All the methods perform equally on the test data: i.e. They all have the same accuracy of 0.833333 on the test Data

Confusion Matrix

All four classification models produced identical confusion matrices and demonstrated an equal ability to distinguish between different classes. The predominant issue across all models lies in false positives.



Conclusions

Different launch sites exhibit varying success rates. CCAFS LC-40 has a success rate of 60%, while KSC LC-39A and VAFB SLC 4E boast higher success rates at 77%.

It can be inferred that, with the increase in flight numbers at each of the three launch sites, the success rate also rises. For example, the success rate for the VAFB SLC 4E launch site reaches 100% after Flight number 50, and both KSC LC 39A and CCAFS SLC 40 achieve a 100% success rate after the 80th flight.

Upon observing the scatter point chart for Payload vs. Launch Site, it becomes evident that there are no rockets launched for heavy payload masses (greater than 10000) at the VAFB-SLC launch site.

Orbits ES-L1, GEO, HEO, and SSO demonstrate the highest success rates at 100%, while the SO orbit has the lowest success rate at approximately 50%, with a 0% success rate in the case of Orbit SO.

In the LEO orbit, success appears to be related to the number of flights. However, in the GTO orbit, there seems to be no discernible relationship between flight number and success.

The success rate for heavy payloads is higher for Polar, LEO, and ISS orbits. However, distinguishing success rates for GTO is challenging as both positive and negative landing outcomes are present.

Additionally, the success rate has consistently increased since 2013 and continued to rise until 2020.

Thank You (: