The Race To Space : A Data Science Insight

- OLUJIMI DAVID

- March 19, 2023

OUTLINE

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

Executive Summary

- **Methodology Summary**
  - Data Collection via SpaceX REST API, Web Scrapping from Wikipedia
  - Data Wrangling
  - Exploratory Data Analysis (EDA) with SQL and visualization
  - Interactive Visual Analytics with Folium and Plotly Dash
  - Predictive Analysis
- **Summary of Results**
  - EDA results
  - Interactive Visual Analytics screenshots
  - Predictive Analysis results

Introduction

- **Context & Background Of The Project**
  - This capstone project will predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars each whereas other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- **Problem Statement**
  - Will the first stage of the falcon 9 land successfully?
  - What are the characteristics that will affect the success or failure of a landing?
  - What are the best conditions that result in a successful landing?

METHODOLOGY

- Section 1

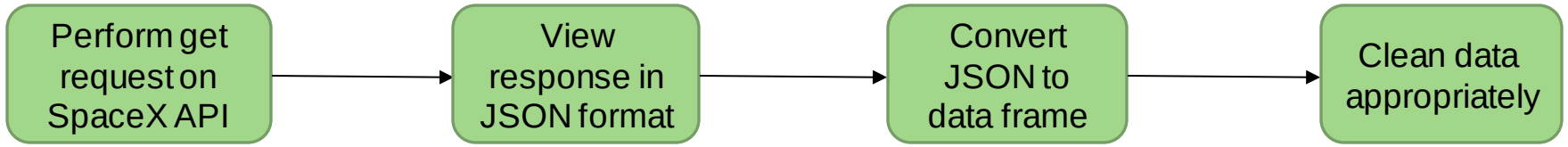
IBM Developer
SKILLS NETWORK

Methodology

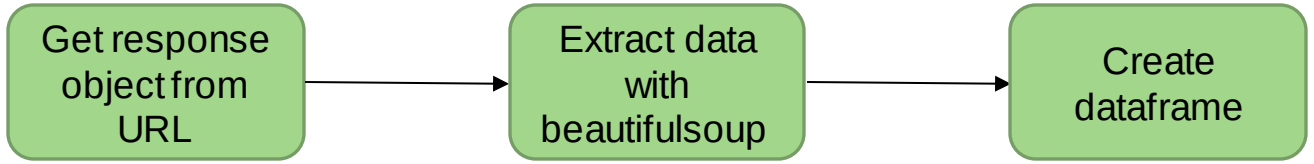
- **Data Collection Methodology:**
  - SpaceX REST API
  - Web Scraping
- **Data Wrangling:**
  - Exploratory Data Analysis
  - Determine Training Labels
- **Exploratory Data Analysis (EDA) using visualization and SQL**
- **Interactive Visual Analytics Using Folium and Plotly Dash**
- **Predictive Analysis (Classification Models):**
  - Logistic Regression
  - Support Vector Machine
  - Tree Classifier
  - K-Nearest Neighbours

Data Collection

- **SpaceX REST API**
  - This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome. Our goal is to use this data to predict whether SpaceX will attempt to land a rocket or not.



- **Web Scraping From Wikipedia**
  - Use the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records from wikipedia.



Data Collection – SpaceX API

SpaceX API Get Request:

Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)
```

Check the content of the response

Decoding the response content as a json using .json() and turn it into a pandas dataframe using .json_normalize():

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

```
In [11]: # Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

Cleaning the data:

```
In [13]: # Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have mu
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection – Web Scraping

Get HTML response and create beatifiulsoup object from response:

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

In [60]:
```
# use requests.get() method with the provided static_url
response = requests.get(static_url).text
# assign the response to a object
```

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

In [61]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, 'html.parser')
```

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

In [62]:
```
# Use soup.title attribute
soup.title
```

Out[62]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Extract data with beatifiulsoup package:

In [71]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
tab = pd.read_html(static_url)
column_names = tab[2].columns
tab[2].head(1)
```

Out[71]:

	Flight No.	Date andtime (UTC)	Version,Booster [b]	Launch site	Payload[c]	Payload mass	Orbit	Customer	Launchoutcome	Boosterlanding
0	1	4 June 2010,18:45	F9 v1.0[7]B0003.1[8]	CCAFS,SLC-40	Dragon Spacecraft Qualification Unit	NaN	LEO	SpaceX	Success	Failure[9][10] (parachute)

Create DataFrame:

After you have fill in the parsed launch record values into `launch_dict`, you can create a dataframe from it.

In [81]:
```
df = pd.DataFrame(launch_dict)
```

In [82]:
```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

- The landing outcomes are either successful or unsuccessful regardless of landing region(Ocean, Ground pad, Drone ship)
  - True Ocean, True RTLS, and True ASDS all mean successful landing
  - False Ocean, False RTLS, False ASDS and None ASDS all mean unsuccessful landing
- Convert the outcomes into training labels, These labels will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully.

No. Of launches on each site

```
In [7]:  # Apply value_counts() on column LaunchSite
         df['LaunchSite'].value_counts()

Out[7]:  CCAFS SLC 40    55
         KSC LC 39A      22
         VAFB SLC 4E     13
         Name: LaunchSite, dtype: int64
```

No. Of occurrence on each orbit

```
In [9]:  # Apply value_counts on Orbit column
         df['Orbit'].value_counts()

Out[9]:  GTO      27
         ISS      21
         VLEO     14
         PO        9
         LEO       7
         SSO       5
         MEO       3
         ES-L1     1
         HEO       1
         SO        1
         GEO       1
         Name: Orbit, dtype: int64
```

No & occurrence of mission outcome

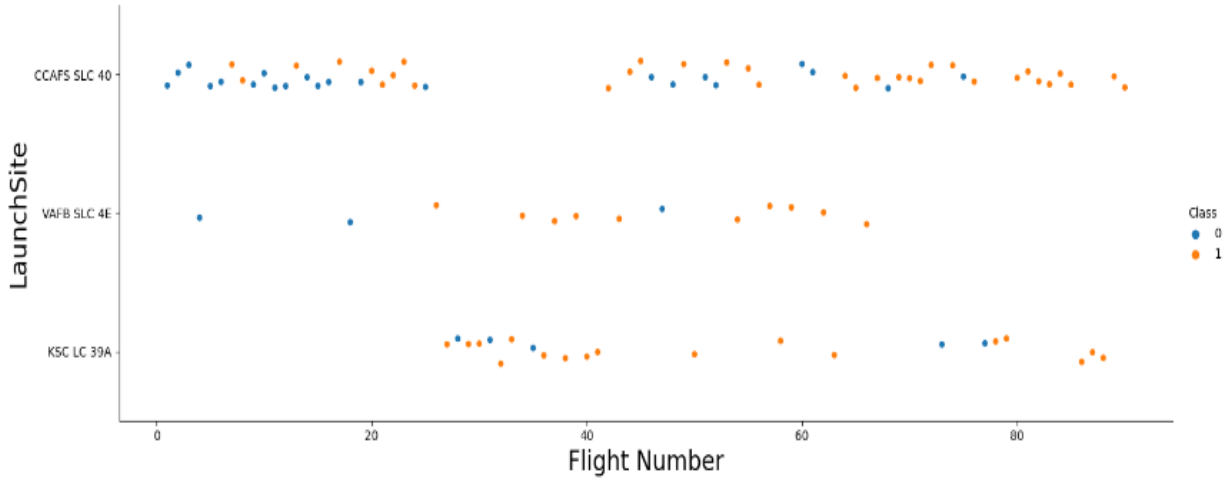
```
In [13]: # landing_outcomes = values on Outcome column
         landing_outcomes = df['Outcome'].value_counts()
         landing_outcomes

Out[13]: True ASDS      41
         None None      19
         True RTLS      14
         False ASDS      6
         True Ocean      5
         False Ocean     2
         None ASDS       2
         False RTLS      1
         Name: Outcome, dtype: int64
```

create landing outcome label from outcome column

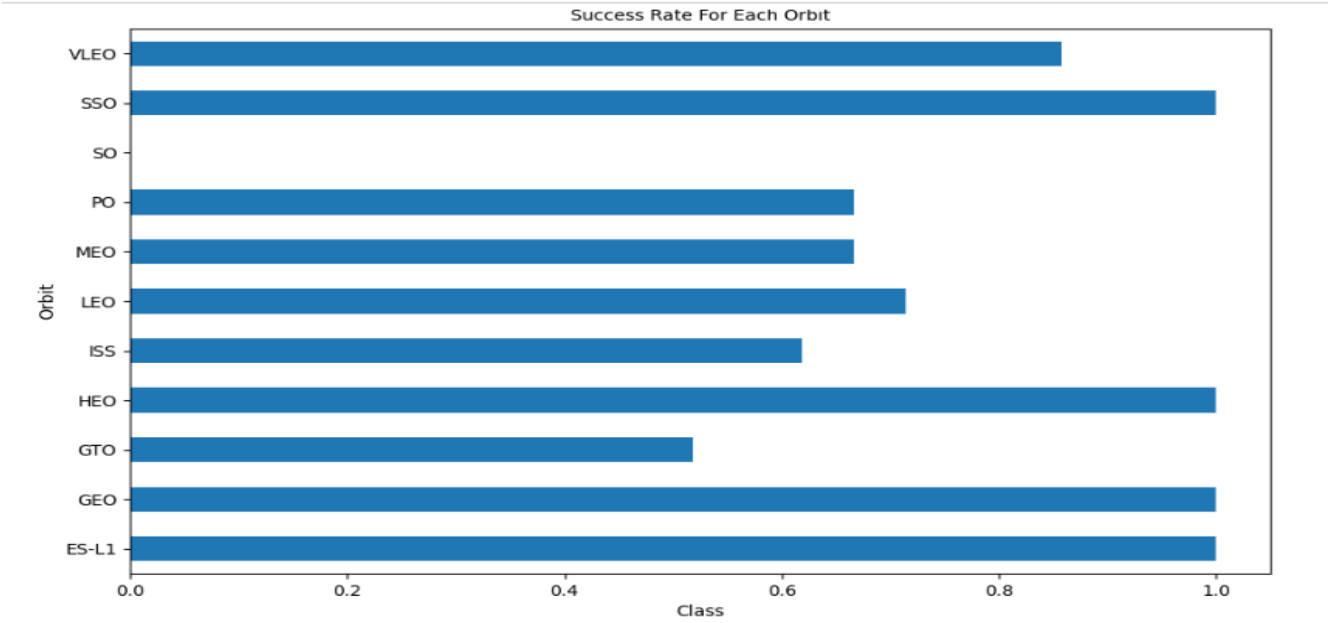
```
In [27]: # landing_class = 0 if bad_outcome
         # landing_class = 1 otherwise

         landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]
```

EDA With Visualization

- The scatter plot visualization revealed the relationships between multiple variables, such as:
  - Flight Number vs Launch Site
  - Payload vs Launch Site
  - Flight Number vs Orbit Type
  - Payload vs Orbit Type
  - Payload vs Flight Number

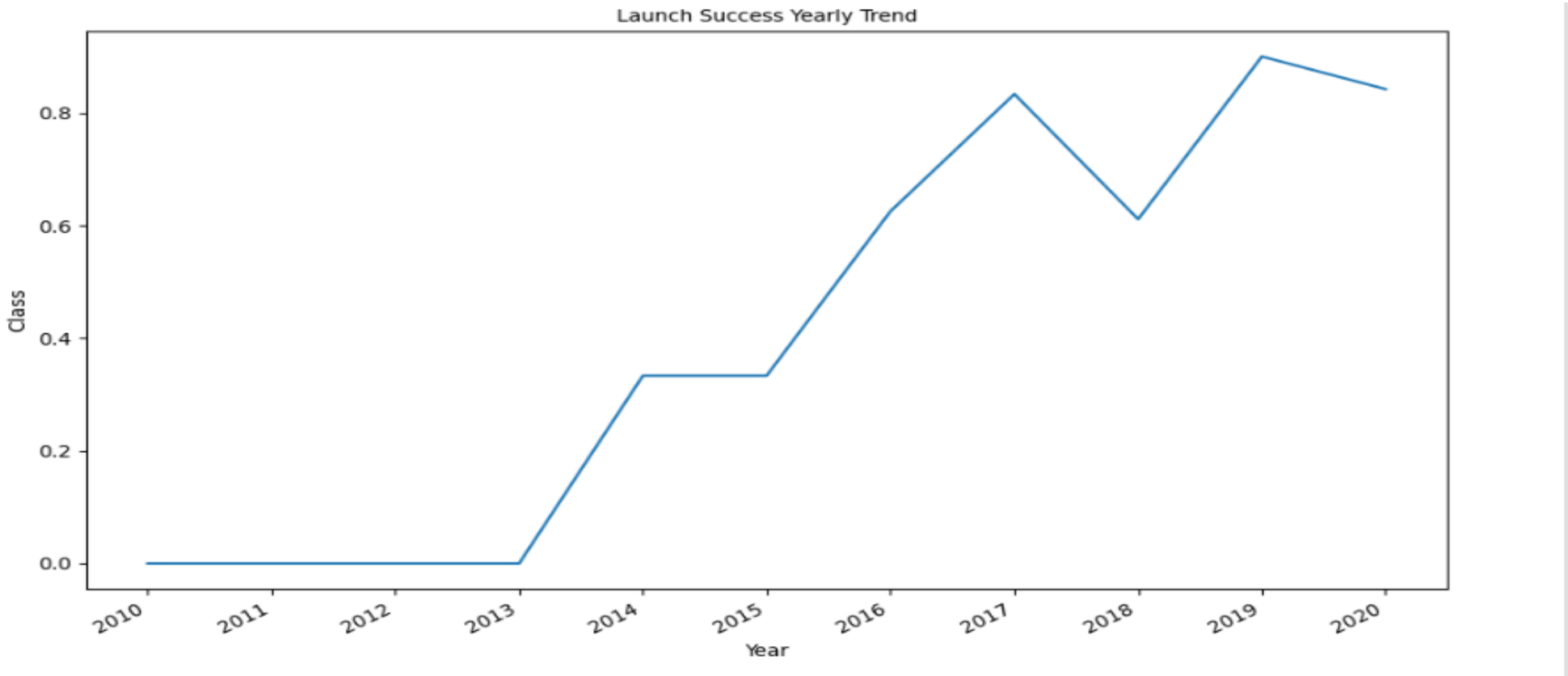


- On the other hand the bar chart revealed the relationship between the success rate of each orbit type.



EDA With Visualization

- To get the average launch success trend on a yearly basis, the year of launch was plotted against the average success rate on a Line Chart.



EDA With SQL

- **To get a better understanding of the data the following SQL queries were used:**
  - *Display the names of the unique launch sites in the space mission*
  - *Display 5 records where launch sites begin with the string 'CCA'*
  - *Display the total payload mass carried by boosters launched by NASA (CRS)*
  - *Display average payload mass carried by booster version F9 v1.1*
  - *List the date when the first successful landing outcome in ground pad was achieved.*
  - *List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000*
  - *List the total number of successful and failure mission outcomes*
  - *List the names of the booster versions which have carried the maximum payload mass. Use a subquery*
  - *List the records which will display the month names, failure landing outcomes in drone ship ,booster versions, launch site for the months in year 2015.*
  - *Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order.*

Building Interactive Map with Folium

- **Markers, Circles, Marker Cluster, Mouse Position and Lines were used to create geospatial visuals**
  - *Markers are used to mark points on the map*
    - *Mark Launch Site Coordinates*
    - *Mark successful and unsuccessful landings (Green(1) for successful, Red(0) for unsuccessful).*
    - *Mark points between launch sites and key points (railway, highway, city, coastline).*
  - *Circles were used to highlight areas*
    - *Red circle at each launch site along with its launch site name.*
    - *Red circle at NASA John Space Centre's coordinates also with its name as label.*
  - *Marker Cluster was used to group launches in a cluster in each coordinate*
  - *Mouse Position was used to get coordinate for a mouse over a point on the map.*
  - *Lines were used to show distance between 2 points*
    - *Plotting lines between launch sites and key points such as coastlines, cities, railways.*

Build a Dashboard with Plotly Dash

- **The dashboard contains the following; dropdown menu, range slider, pie chart and a scatter plot**
  - *Dropdown menu allows the user to select all launch sites or a particular launch site.*
  - *The Range slider allows the user to choose a payload mass in a fixed range.*
  - *The pie chart is used to show successful launches by total or by a specific launch site.*
  - *The scatter plot is to show the correlation between payload and success rate.*

Predictive Analysis - Classification

- **This section is made up of 3 main parts, namely:**
  - *Model Building*
  - *Model Evaluation*
  - *Model Comparison*

Model Building

- Import relevant libraries and load the data set.
- Standardize the data set.
- Split the data set into training set and test set for classification algorithms.
- Set the parameters needed for GridSearchCV and then fit the data.

Model Evaluation

- Calculate the accuracy of each model accordingly.
- Obtain the best and favorable parameters for each model.
- Plot confusion matrix for each model.

Model Comparison

- Find the hyperparameters for the models.
- Find out which model has the highest accuracy.
- The best model has the highest accuracy.

Results

- Exploratory Data Analysis results
- Interactive Visual Analytics screenshots
- Predictive Analysis results

SECTION 2

INSIGHTS FROM EXPLORATORY DATA ANALYSIS

Launch Site Names

```
%sql SELECT DISTINCT (Launch_Site) FROM SPACEXTBL
```

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

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

- Above is the SQL query used to obtain the unique names of each launch site and its result.
- SELECT DISTINCT is used to remove duplicate values of the launch site.

Launch Site Names That Begin With "CCA"

- SQL Query:

```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5
 * sqlite:///my_data1.db
Done.
```

  - WHERE clause specifies the column name.
  - LIKE clause specifies the substring 'CCA'.
  - LIMIT trims the result into 5 rows.

- Query Result:

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

- SQL Query:

```
%sql SELECT Customer, SUM(PAYLOAD_MASS__KG_) AS "TOTAL PAYLOAD LAUNCHED BY NASA (CRS)" FROM SPACEXTBL GROUP BY 1 HAVING Custo
 * sqlite:///my_data1.db
Done.
```

  - SUM calculates the sum of the column selected
  - GROUP BY groups the results by customer
  - HAVING clause filters the row where customer = 'NASA CRS'
- Query Result:

Customer	TOTAL PAYLOAD LAUNCHED BY NASA (CRS)
NASA (CRS)	45596

Average Payload Mass Carried by F9 v1.1

- SQL Query:

```
%sql SELECT Booster_Version, AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL GROUP BY 1 HAVING Booster_Version = 'F9 v1.1'
 * sqlite:///my_data1.db
Done.
```

  - AVG was used to calculate the average payload mass
  - GROUP BY groups the results by booster version
  - HAVING clause filters the row where booster version = 'F9 v1.1'
- Query Result:

Booster_Version	AVG(PAYLOAD_MASS__KG_)
F9 v1.1	2928.4

Date Of First Successful Outcome

- SQL Query:

```
%sql SELECT "Landing _Outcome", MIN(Date) FROM SPACEXTBL GROUP BY 1 HAVING "Landing _Outcome" = "Success (ground pad)"
 * sqlite:///my_data1.db
Done.
```

  - SELECT MIN locates the earliest successful landing date
  - GROUP BY groups the result by landing outcome
  - HAVING filters the rows where outcome is successful
- Query Result:

Landing _Outcome	MIN(Date)
Success (ground pad)	01-05-2017

Success In Drone ship With Payload Mass Between (4000 – 6000 kg)

```
%sql SELECT Booster_Version, "Landing _Outcome", PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone
AND PAYLOAD_MASS__KG_ BETWEEN '4000' AND '6000'
```

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

- SQL Query:
  - The WHERE clause filters the column for specific attributes for the landing outcome and payload mass
  - The BETWEEN clause specifies the range of the payload mass (4000 - 6000 kg)
- Query Result:

Booster_Version	Landing _Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Total Number Of Success And Failure Mission Outcomes

- SQL Query:

```
%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Success%') AS 'SUCCESS', \
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS 'FAILURE'
```

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

  - A subquery was used to select the total number of successful outcomes and total number of failed outcomes

- Query Result:

SUCCESS	FAILURE
100	1

Boosters That Have Carried The Maximum Payload Mass

- SQL Query & Result:
  - To obtain the boosters that have carried the maximum payload mass a subquery was implemented

```
ELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTE
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

Launch Records For 2015

- SQL Query:

```
%sql SELECT substr(Date, 4, 2) AS Month, "Landing _Outcome", Booster_Version, Launch_Site FROM SPACEXTBL WHERE "Landing _Outc
AND substr(Date,7,4)='2015'
```

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

  - This query returns the month, landing outcomes that failed for drone ships, booster version, launch sites for the year 2015.

- Query Result:

Month	Landing _Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- SQL Query:

```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "LANDING _OUTCOME" LIKE '%Success%' AND "DATE" >= '04-06-2010' AND "DATE" <= '20-03-2017'\
GROUP BY 1 \
ORDER BY 2 DESC
```

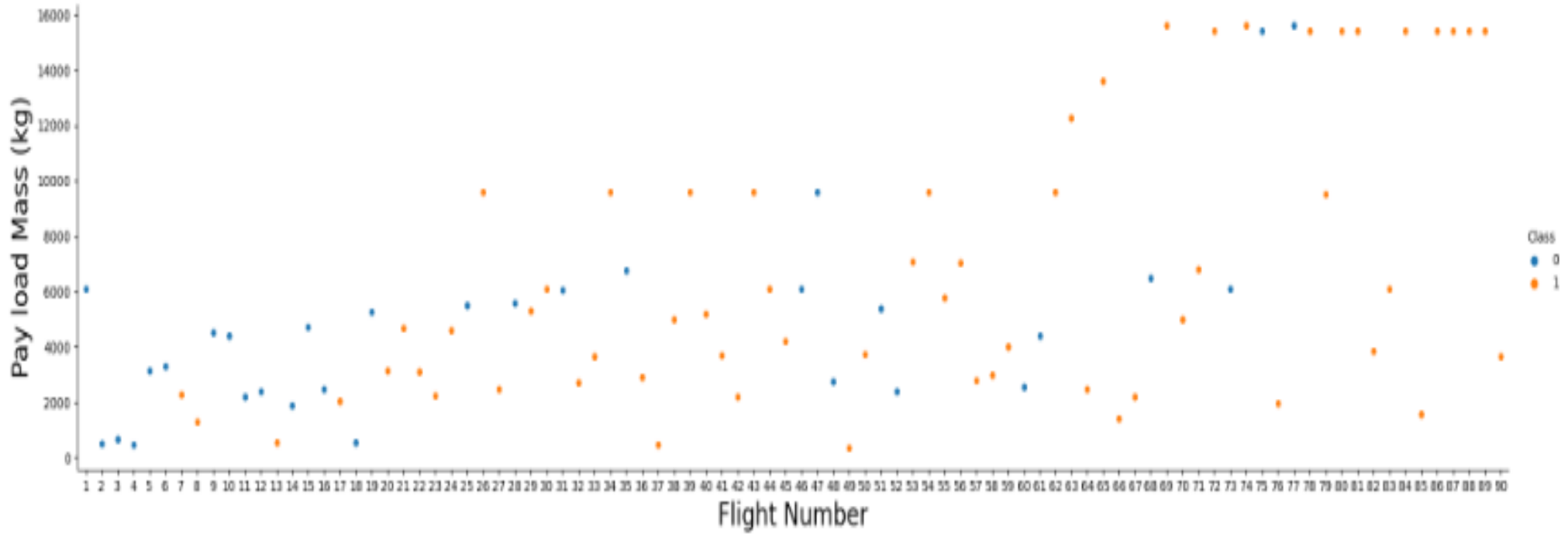
```
 * sqlite:///my_data1.db
Done.
```

  - This query returns the number of successful landing outcomes between 04-06-2010 and 20-03-2017.
  - GROUP BY clause groups the result by the landing outcomes.
  - ORDER BY clause displays the result in descending order with the help of the DESC function.

- Query Result:

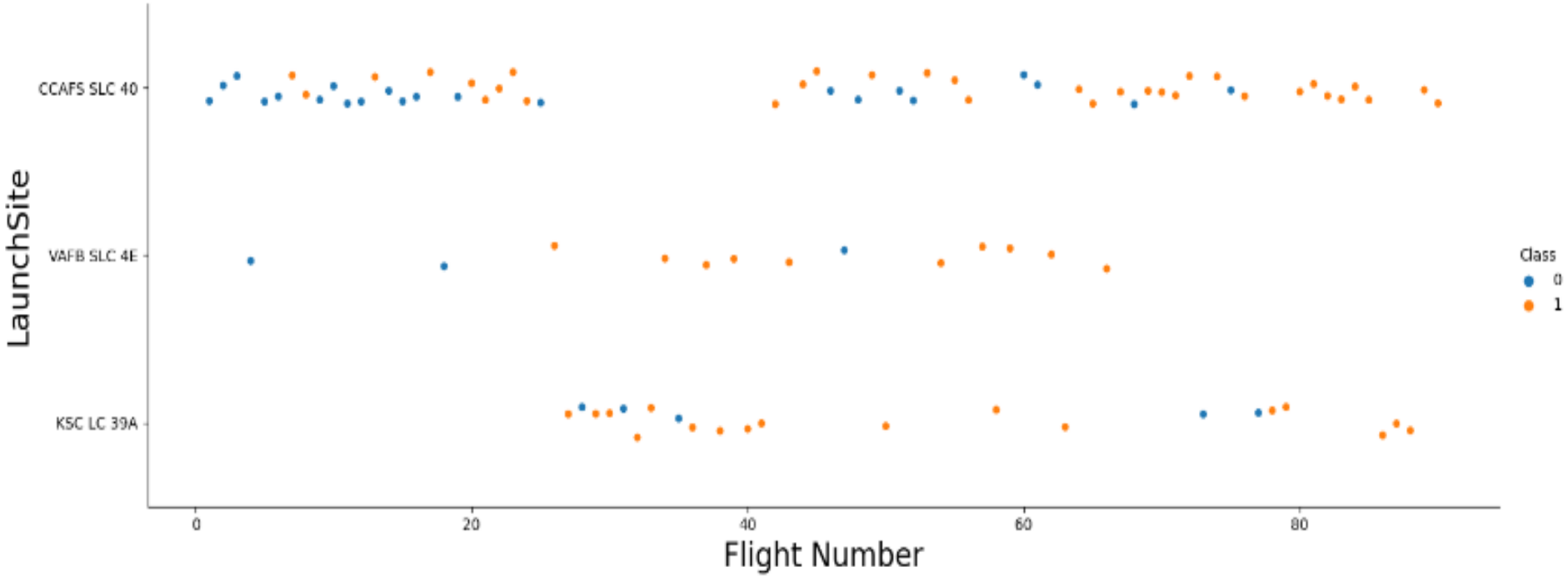
Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

Flight Number vs Pay Load Mass



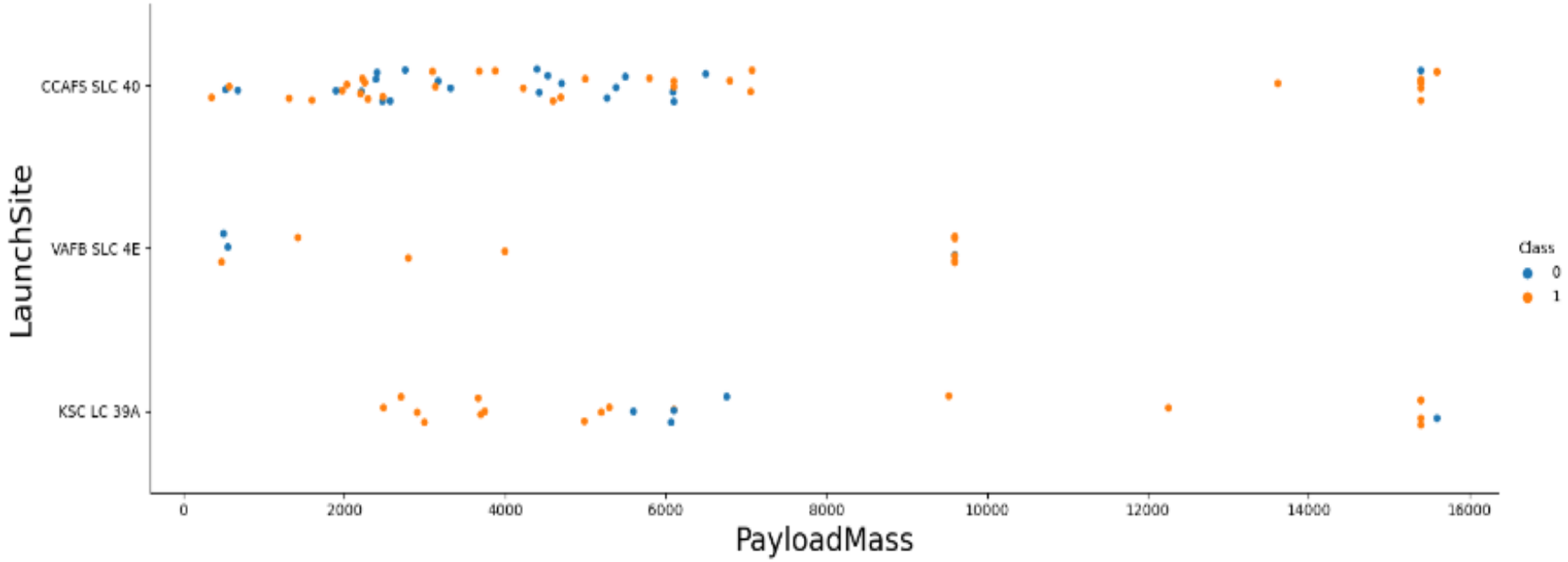
- We see that as the flight number increases, the first stage is more likely to land successfully.
- The payload mass is also important; it seems the more massive the payload, the less likely the first stage will return.

Flight Number vs Launch Site



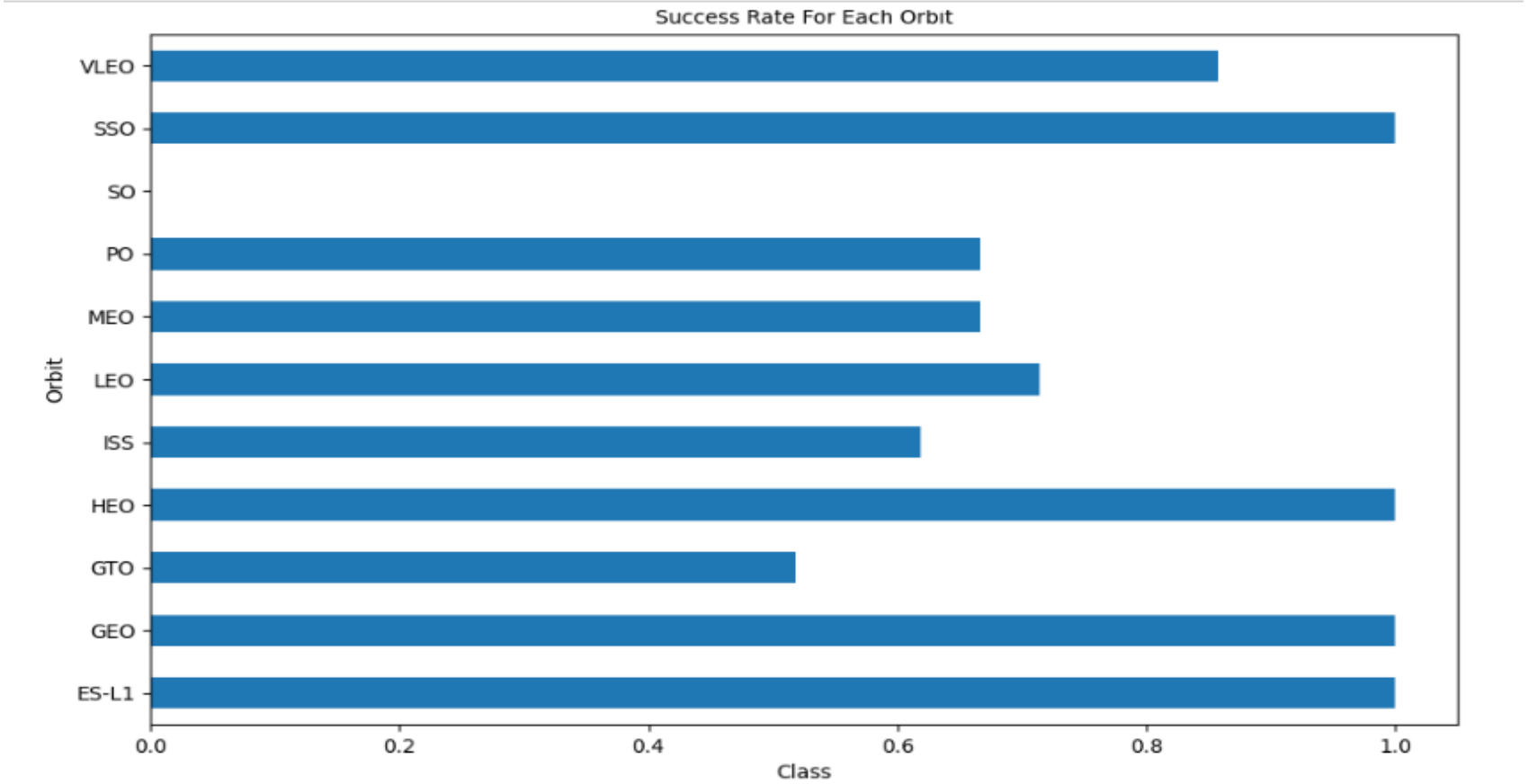
- From the above scatter plot we can observe that CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.
- It is also important to note that launch site CCAFS LC-40 carried out more launches compared to the other launch sites.

Pay Load vs Launch Site



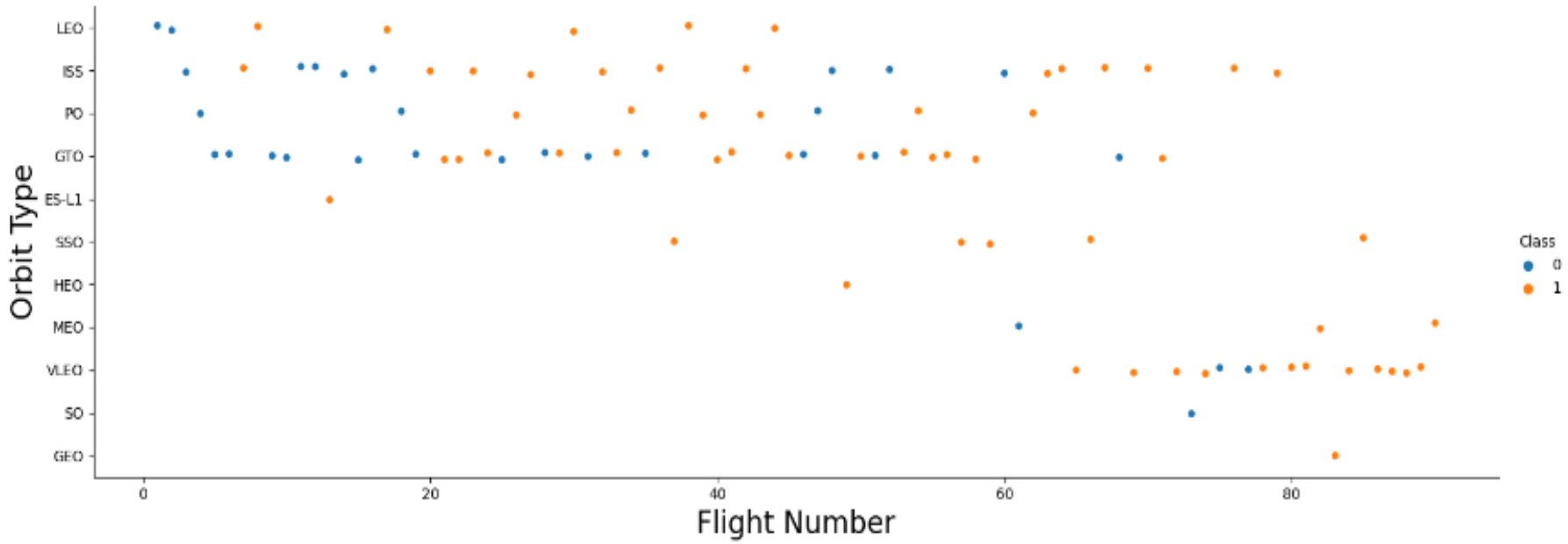
- From the scatter plot above, it is clear that for all the launch sites majority of the launches have pay load mass between (0 - 8000kg) and payloads above 8000kg tend to have a higher success rate.
- You will also find out that for the VAFB-SLC-4E launch site there are no rockets launched for heavy payload mass(greater than 10000).

Success Rate vs Orbit Type



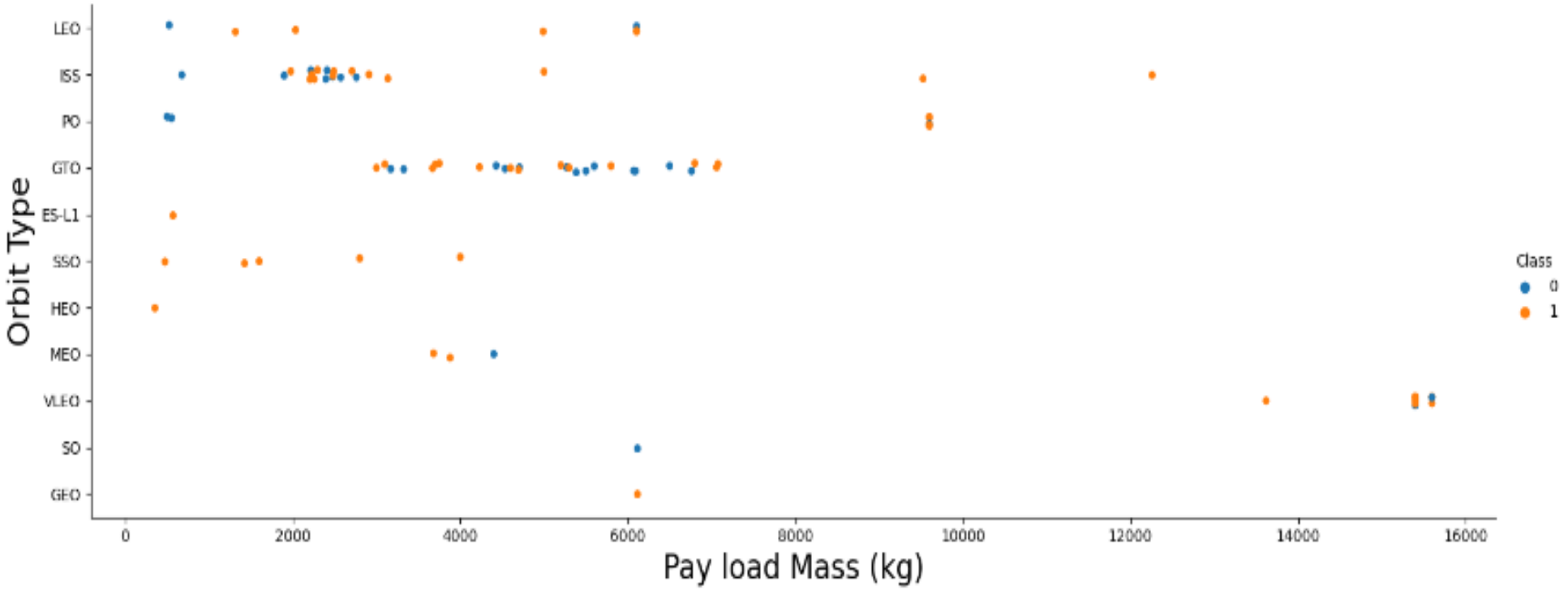
- The orbits with the highest success rate recorded are; SSO, HEO, GEO and ES – L1.
- For the SO orbit there was no success rate recorded.

Flight Number vs Orbit Type



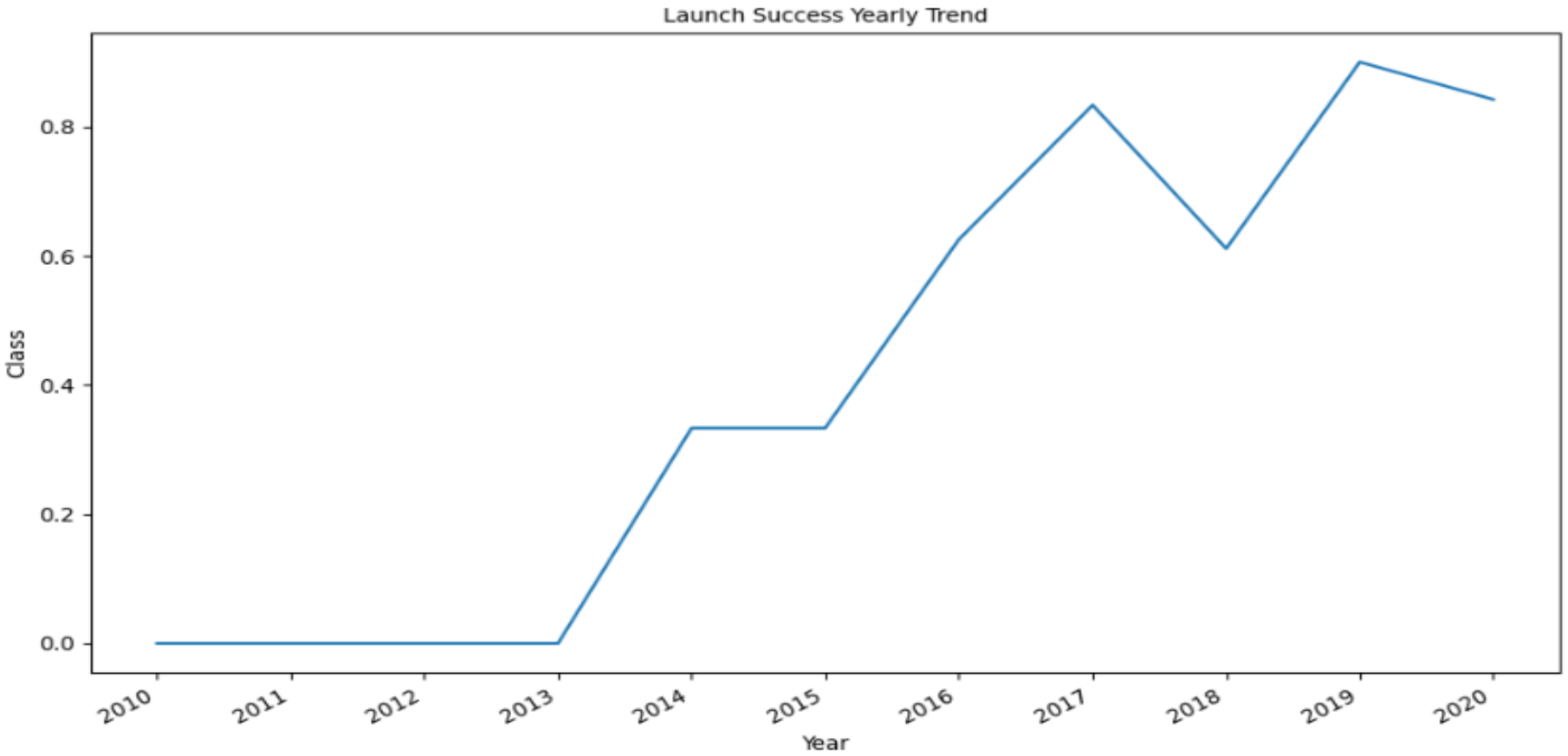
- For the LEO orbit the Success rate appears to be related to the number of flights.
- There seems to be no relationship between flight number when in GTO orbit.

Payload Mass vs Orbit Type



- Polar and ISS orbits have successful outcomes for heavy pay loads (greater than 8000 kg).
- For GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both present.
- For the SSO orbit there were no successful outcomes recorded for payloads above 4000 kg

Yearly Trend For Launch Success



- From the above line plot, it is evident that from year 2013, the success rate has maintained a steady rise.

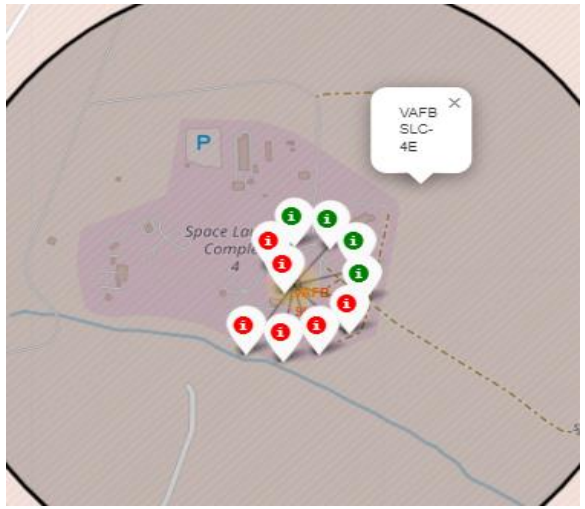
SECTION 3

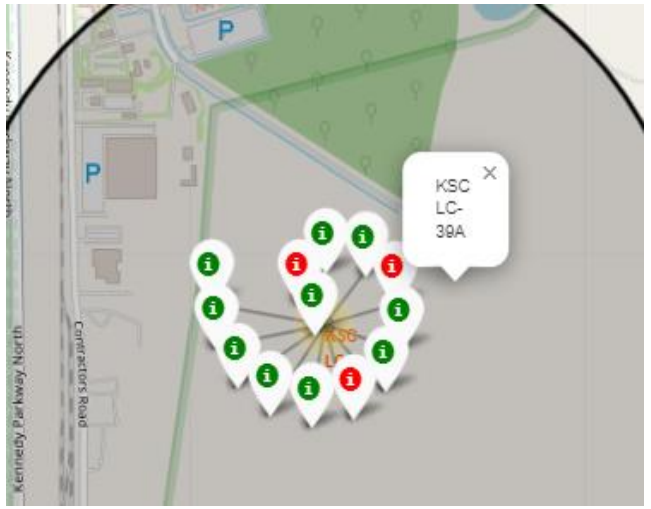
ANALYSIS OF LAUNCH SITE PROXIMITY

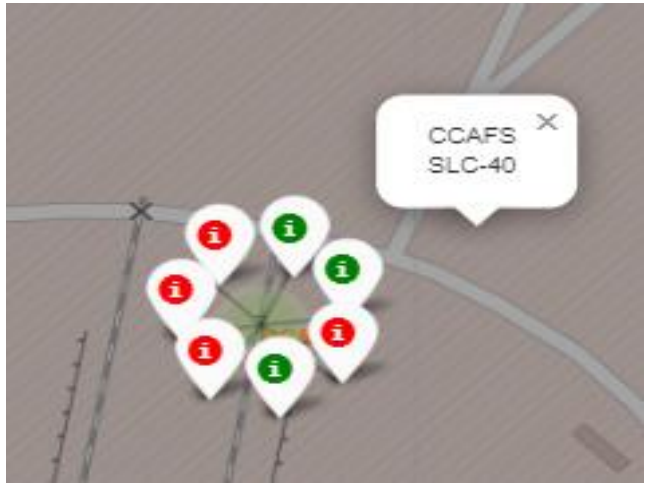
Launch Site Locations

- All launch sites are located within the United States.
- Launch sites are notably located at coast lines most likely due safety reasons and safety measures.

Launch Sites Labelled With Colors

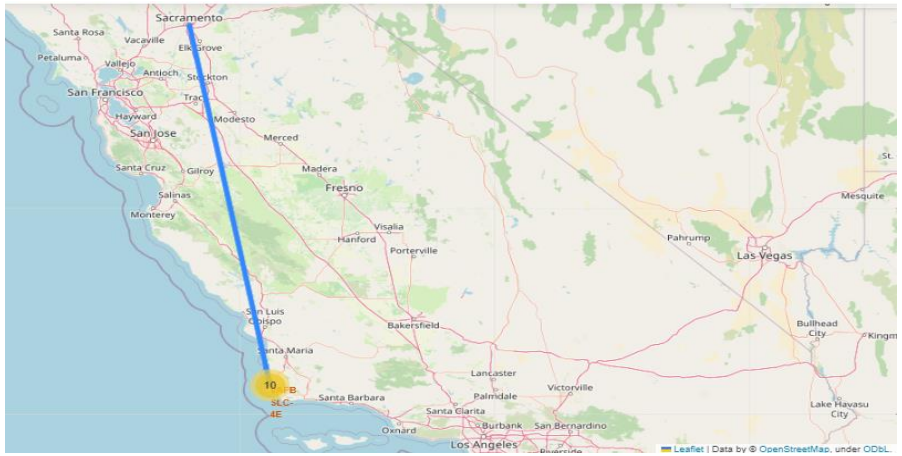






- The Green Markers represent successful launches.
- The Red Markers represent unsuccessful launches.

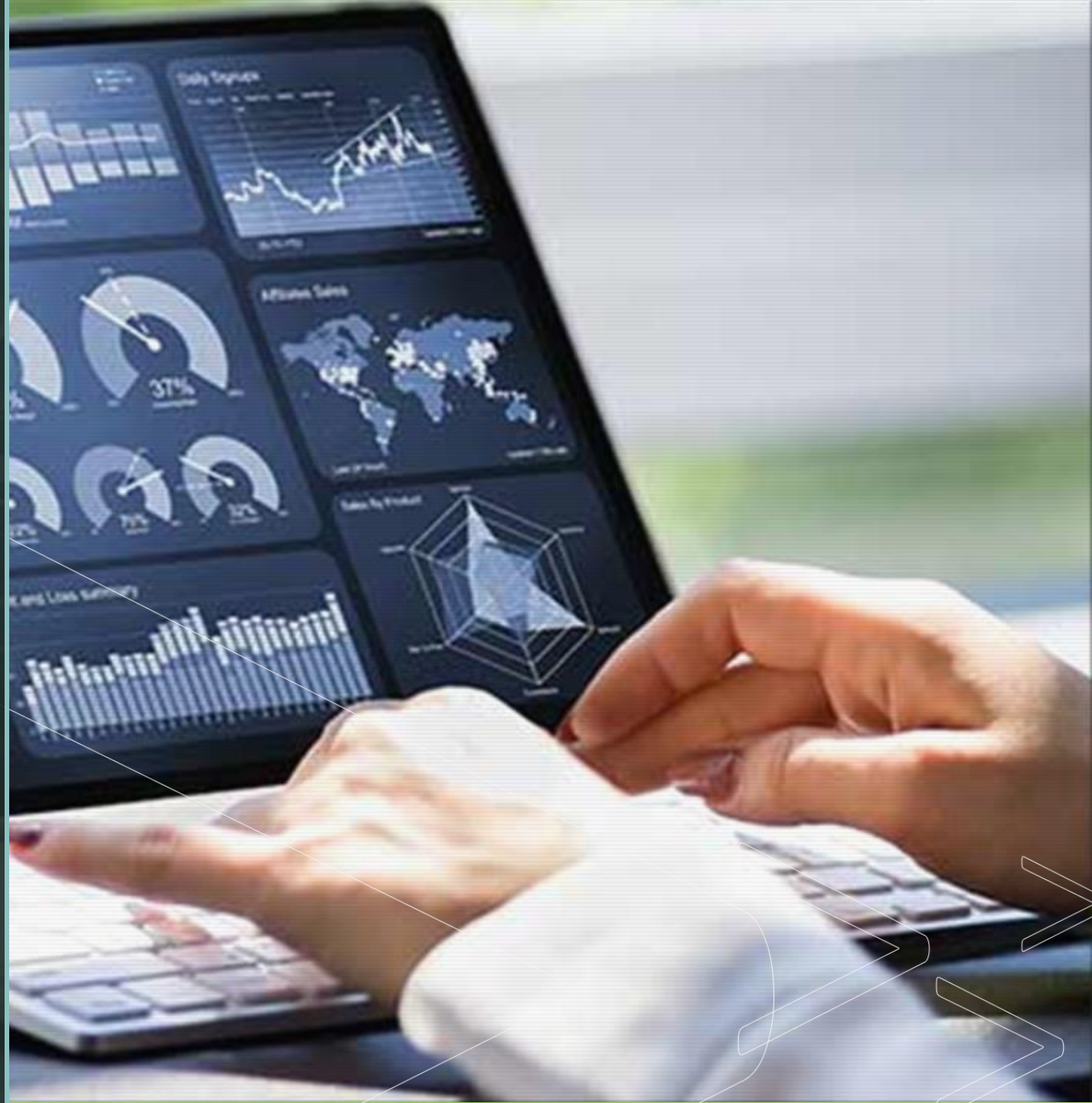
Launch Site Distance And Proximities



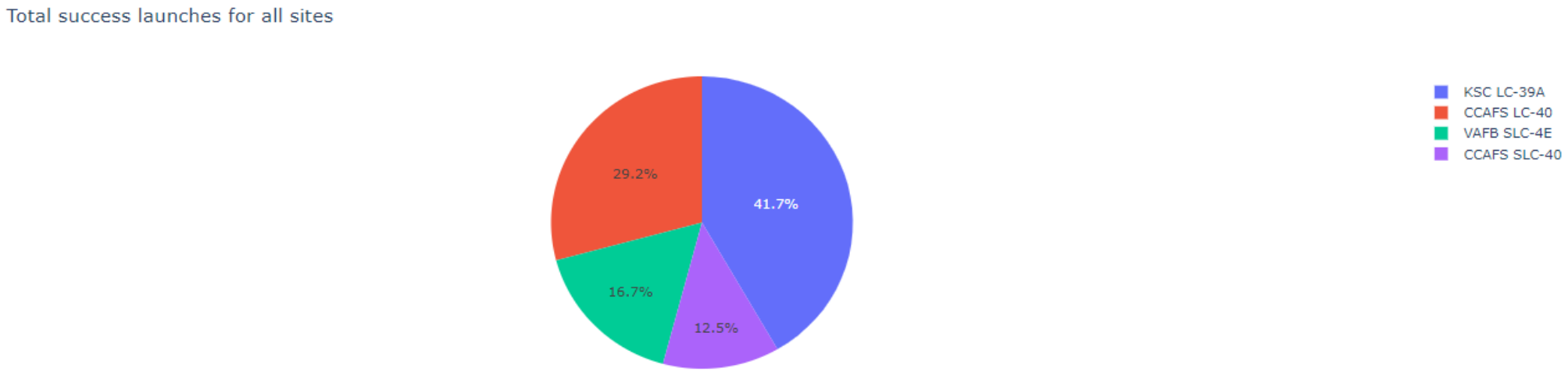
- Launch sites have close proximity to highways.
- Launch sites have close proximity to railways.
- Launch sites have close proximity to coastlines.
- Launch sites keep certain distances from cities.

BUILDING A DASHBOARD WITH PLOTLY DASH

- SECTION 4

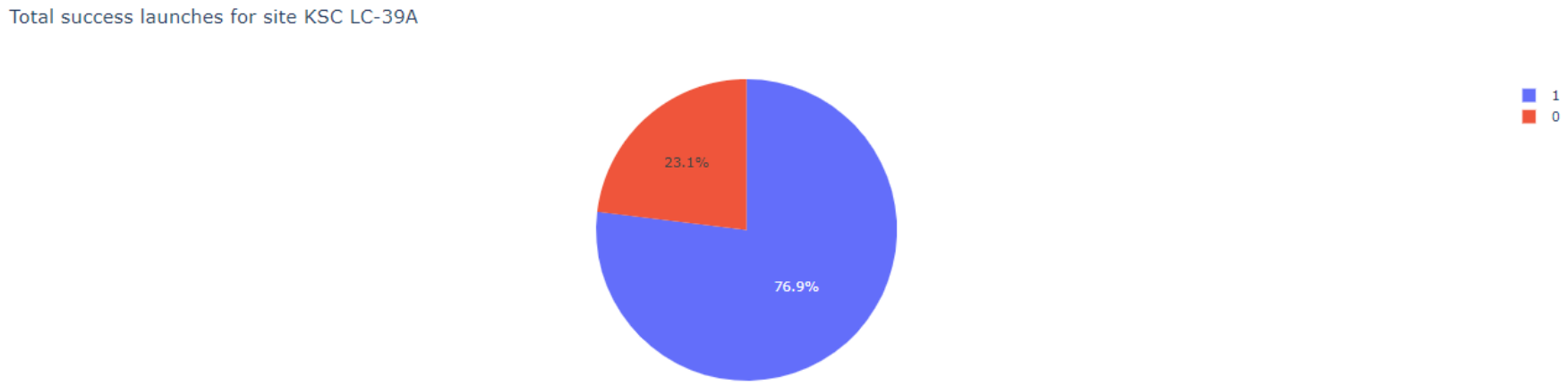


Success Rate For Each Launch Site



- Launch site CCAFS SLC-40 has the lowest success rate
- Launch site KSC LC-39A has the highest success rate

Success Rate For Site KSC LC-39A

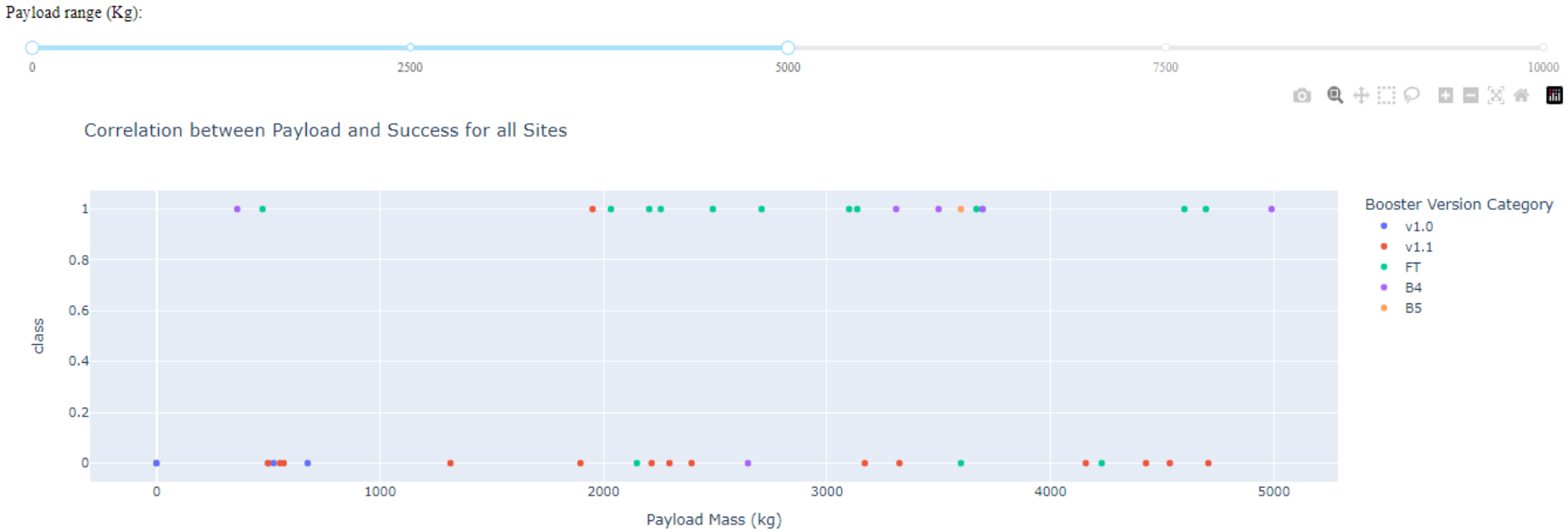


- Launch site KSC LC-39A has a success rate of 76.9% and failure rate of 23.1%

Scatter Plot Of Payload Mass vs Launch Outcome

- Success rate with heavy payload is relatively low.

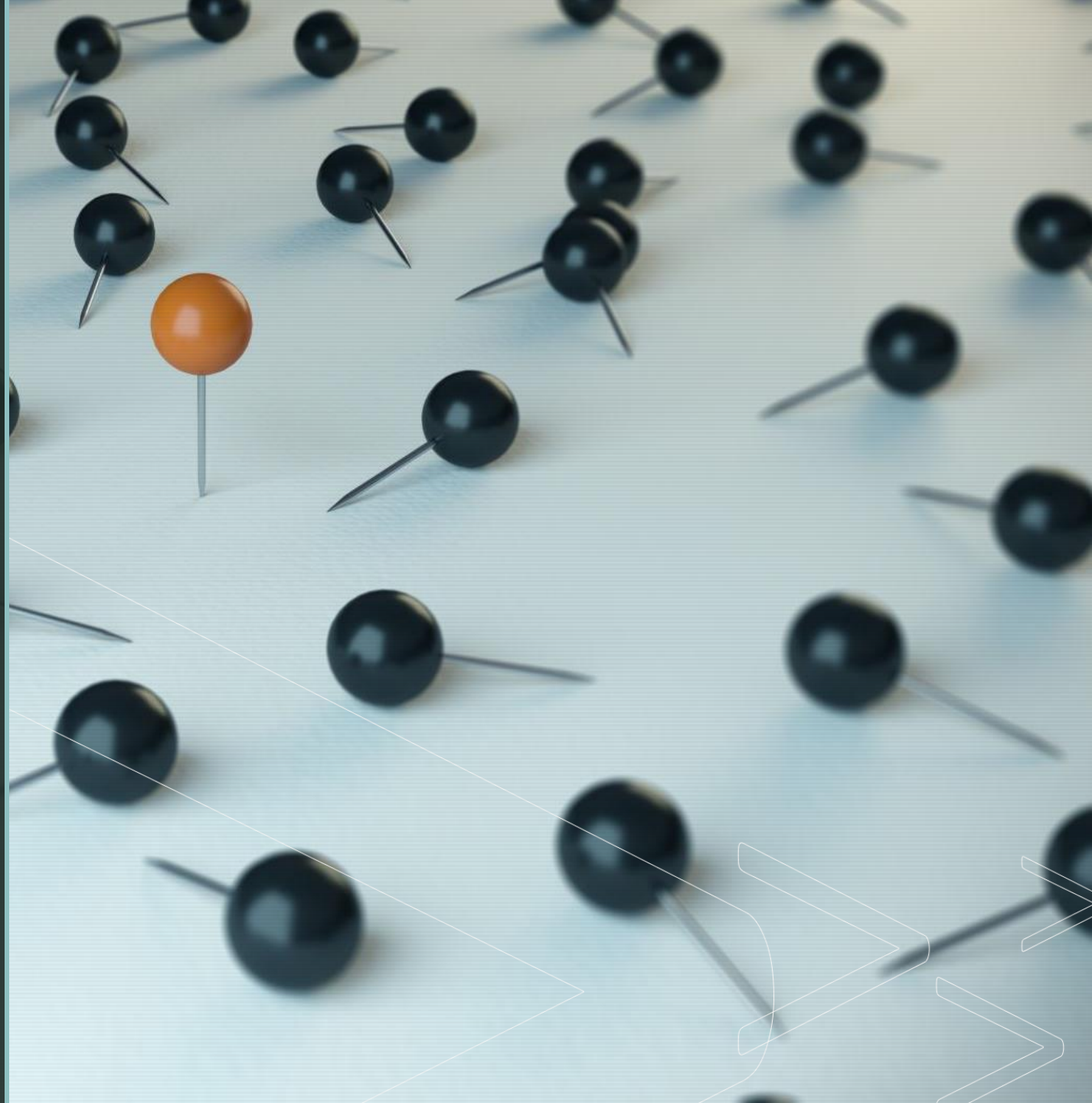
Scatter Plot Of Payload Mass vs Launch Outcome



- Success rate with low weighted payload is high.

SECTION 5

PREDICTIVE ANALYSIS



Classification Accuracy

```
logreg_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

```
svm_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

```
tree_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

```
knn_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

- Logistic Regression Model, SVM Model, Tree Classifier and KNN Model are the models that were built for classification.
- All these models have same accuracy score on the test data.

Classification Accuracy

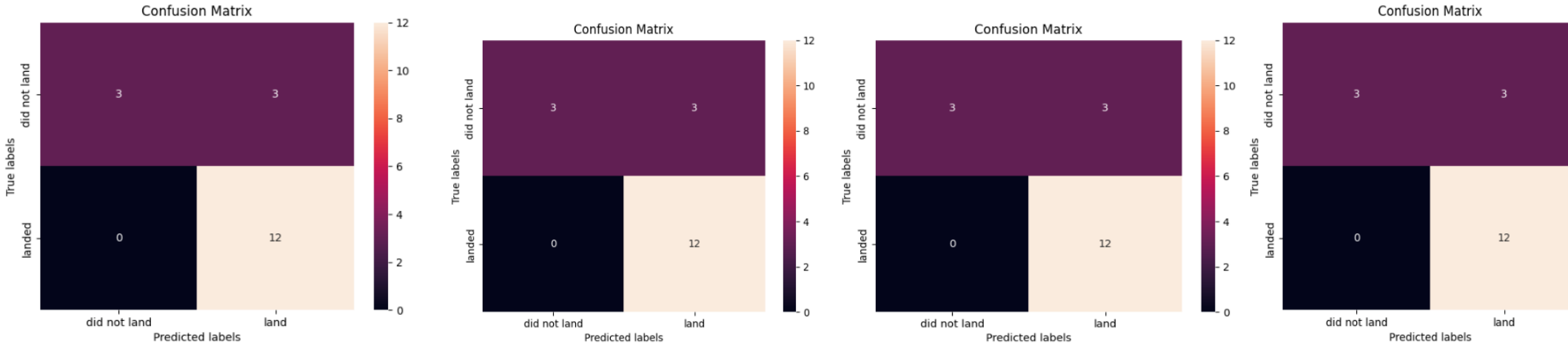
```
methods = {'Logistic Regression': logreg_cv.best_score_, 'SVM': svm_cv.best_score_, 'Tree Classifier':tree_cv.best_score_, 'K
bestmethod= max(methods, key=methods.get)
print('The method that performs best is:', bestmethod, 'with score', methods[bestmethod])
if bestmethod == 'Tree Classifier':
    print('Best Params is :', tree_cv.best_params_)
if bestmethod == 'KNN':
    print('Best Params is :', knn_cv.best_params_)
if bestmethod == 'Logistic Regression':
    print('Best Params is :', logreg_cv.best_params_)
if bestmethod == 'SVM':
    print('Best Params is :', svm_cv.best_score_)
```

```
The method that performs best is: Tree Classifier with score 0.9
Best Params is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 1
0, 'splitter': 'random'}
```

- On the other hand, the tree classifier model has the best accuracy with the training data, hence the tree classifier can be considered as the best model.

Confusion Matrix

Logistic Regression SVM Tree Classifier KNN



- The confusion matrix for all of the classification models are equal as they all have the same accuracy on the test model.

CONCLUSIONS

- The success of a launch can be studied in relation to multiple factors such as the orbit type, flight number, launch site and payload mass.
- ES-L1, GEO, HEO, and SSO orbits recorded the highest success rate.
- Launch site KSC LC-39A has the highest success rate compared to other launch sites.
- Launch site CCAFS SLC-40 has the lowest success rate.
- Low weighted payload performed better than heavy weighted payload.
- Best classification model is Decision Tree Classifier as it has higher accuracy on the training data.

THANK YOU !