IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

GBADEBO-OGUNMEFUN SODIQ

3/01/2022

# Outline

# Executive Summary

Summary of methodologies

Data collection, Data wrangling, EDA with data visualization, EDA with SQL, Building an interactive map with Folium, Building a Dashboard with Plotly Dash, Predictive analysis (Classification)

Summary of all results

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

# Introduction

## Project background and context

- We predicted 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; 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.

## Problems to be solved

- The effect each relationship with certain rocket variables will impact in determining the success rate of a successful landing.
- What influences if the rocket will land successfully?
- What conditions does SpaceX have to achieve to get the best results and ensure the best rocket success landing rate.

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - SpaceX Rest API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

# Data Collection

- The data collection process
  - The SpacesX data was collected from the SPACEX REST API.
  - Our goal is to use this data to predict whether SpaceX will attempt to land a rocket or not.
  - The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/
  - Web scrapping from Wikipedia is another way the data can be collected using Beautiful Soup.

**SpaceX API**

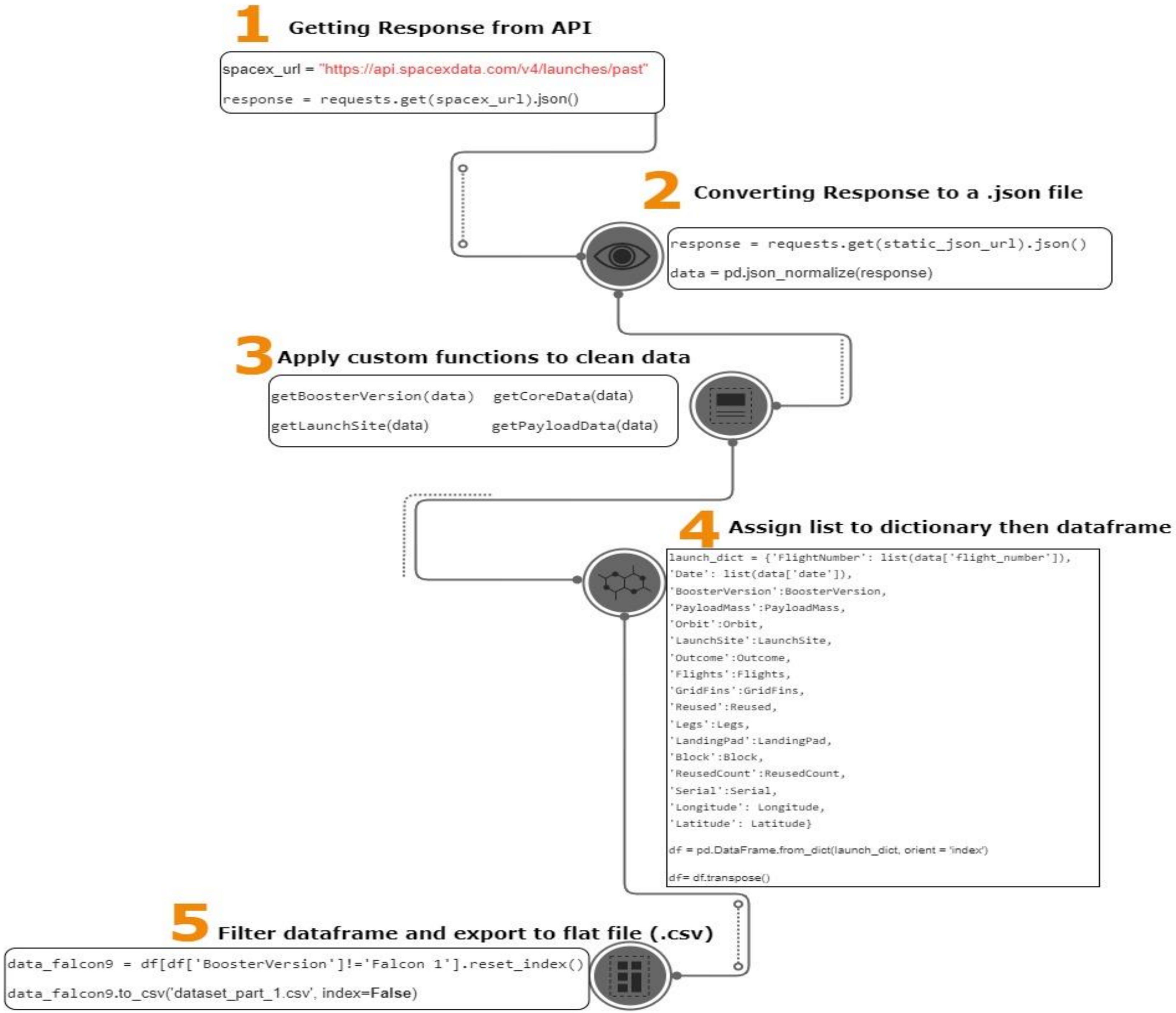
Request rocket launch data from SpaceX API → API returns SpaceX data in .JSON → Normalize data into Data Frame file such as .csv

**Web Scrapping**

Request the Falcon9 Launch Wiki page from its URL → Extract data using Beautiful soup → Create a data frame by parsing the launch HTML tables

# Data Collection – SpaceX API

[Github Url to Notebook]



# Data Collection - Scraping

[Github Url to Notebook]

## 1 .Getting Response from HTML

```
page = requests.get(static_url)
```

## 2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

## 3. Finding tables

```
html_tables = soup.find_all('table')
```

## 4. Getting column names

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

## 5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']


launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

## 6. Appending data to keys (refer) to notebook block 12

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
           # get table row
            for rows in table.find_all("tr")
              #check to see if first table
```

## 7. Converting dictionary to dataframe

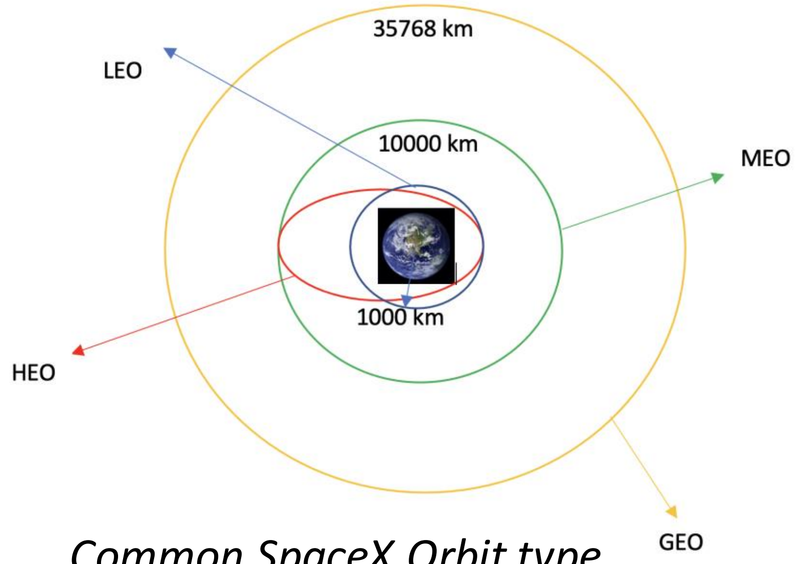
```
df = pd.DataFrame.from_dict(launch_dict)
```

## 8. Dataframe to .CSV

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

# Data Wrangling

In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad. True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.




*Common SpaceX Orbit type*

**Exploratory Data Analysis**

Calculate the number of launches on each site

Calculate the number and occurrence of each orbit

Calculate the number and occurrence of mission outcome per orbit type

Create a landing outcome label from Outcome column

Export dataset as .CSV

[Github Url to Notebook]()

# EDA with Data Visualization

## Scatter plots:

- Flight Number VS. Payload Mass
- Flight Number VS. Launch Site
- Payload VS. Launch Site
- Orbit VS. Flight Number
- Payload VS. Orbit Type
- Orbit VS. Payload Mass

Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation . Scatter plots usually consist of a large body of data

## Bar plots:

- Success rate vs orbit type

A bar diagram makes it easy to compare sets of data between different groups at a glance. The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time.

## Line plots:

- Success Rate VS. Year

Line graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded



Github Url to Notebook

# EDA with SQL

**Performed SQL queries to gather information about the dataset**

- Displayed the names of the unique launch sites in the space mission.
- Displayed 5 records where launch sites begin with the string 'CCA'.
- Displayed the total payload mass carried by boosters launched by NASA (CRS).
- Display average payload mass carried by booster version F9 v1.1.
- Listed 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.
- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015.
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

Github Url

# Build an Interactive Map with Folium

To visualize the Launch Data into an interactive map. We took the Latitude and Longitude Coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.

We assigned the dataframe launch_outcomes(failures, successes) to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()

Using Haversine's formula we calculated the distance from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. Lines are drawn on the map to measure distance to landmarks



Github Url

# Build a Dashboard with Plotly Dash

**The dashboard is built with Dash web framework.**

Pie Chart shows the total launches by a certain site/all sites

- Display relative proportions of multiple classes of data.
- Size of the circle can be made proportional to the total quantity it represents.

Scatter plot shows the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions

- It shows the relationship between two variables.
- It is the best method to show you a non-linear pattern.
- The range of data flow, i.e. maximum and minimum value, can be determined.
- Observation and reading are straightforward.

[Github Url to notebook]

# Predictive Analysis (Classification)

**BUILDING MODEL**

- Load the dataset and Create a NumPy array from the column Class
- Standardized the Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms to be used
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

**EVALUATING MODEL**

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

**IMPROVING MODEL**

- Feature Engineering
- Algorithm Tuning

**FINDING THE BEST PERFORMING CLASSIFICATION MODEL**

- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

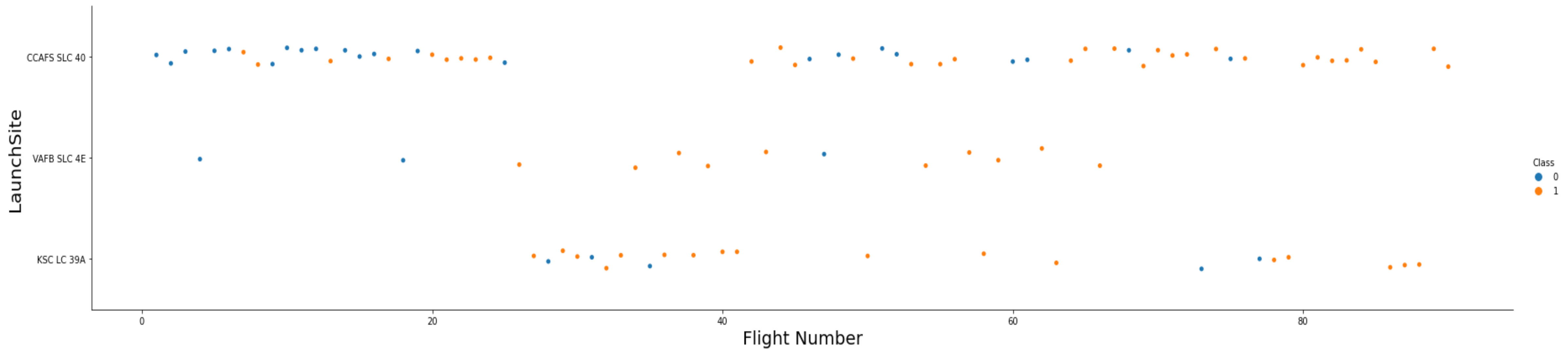
Github Url

# Results

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

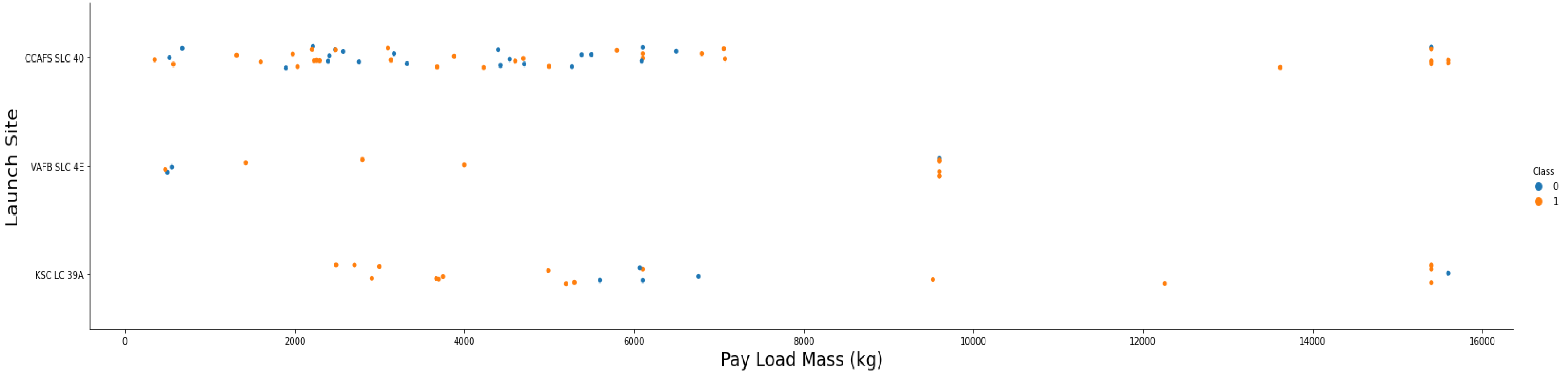
Section 2

# Insights drawn from EDA



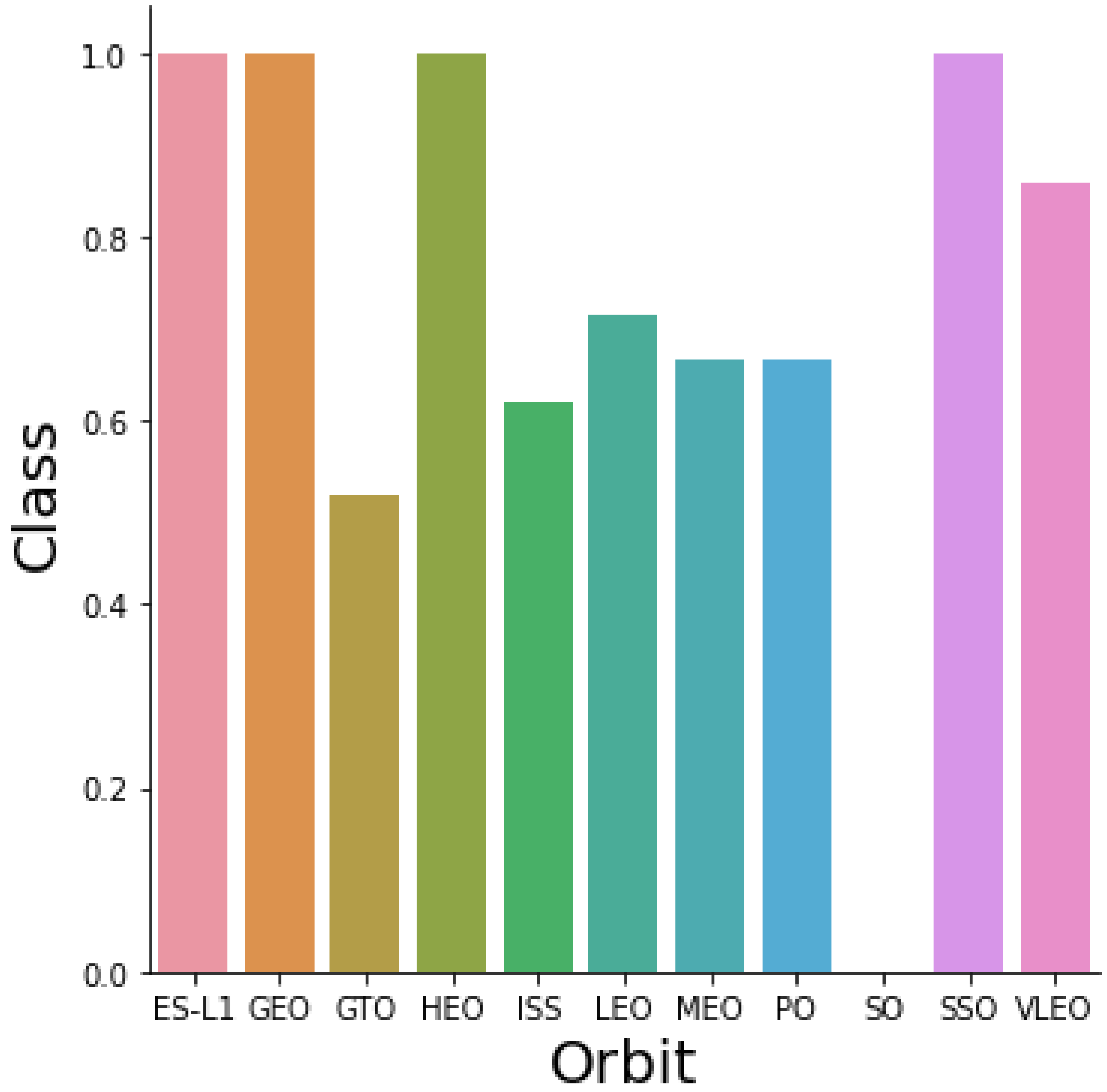

# Flight Number vs. Launch Site

- The more amount of flights at a launch site the greater the success rate at a launch site.
- From the scatter it can be seen that there are more success with increase in flight number.



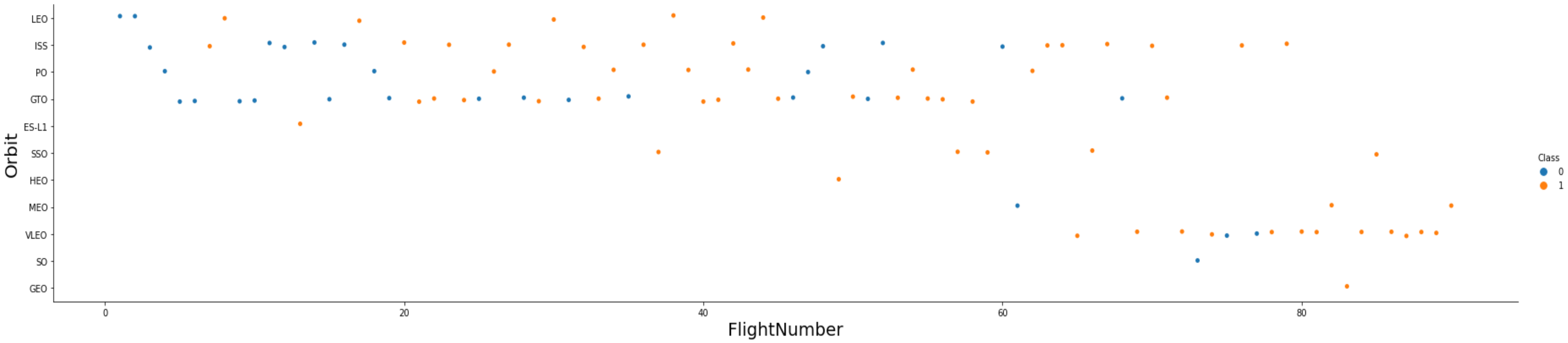

# Payload vs. Launch Site

- The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket.
- There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependent on Pay Load Mass for a success launch.



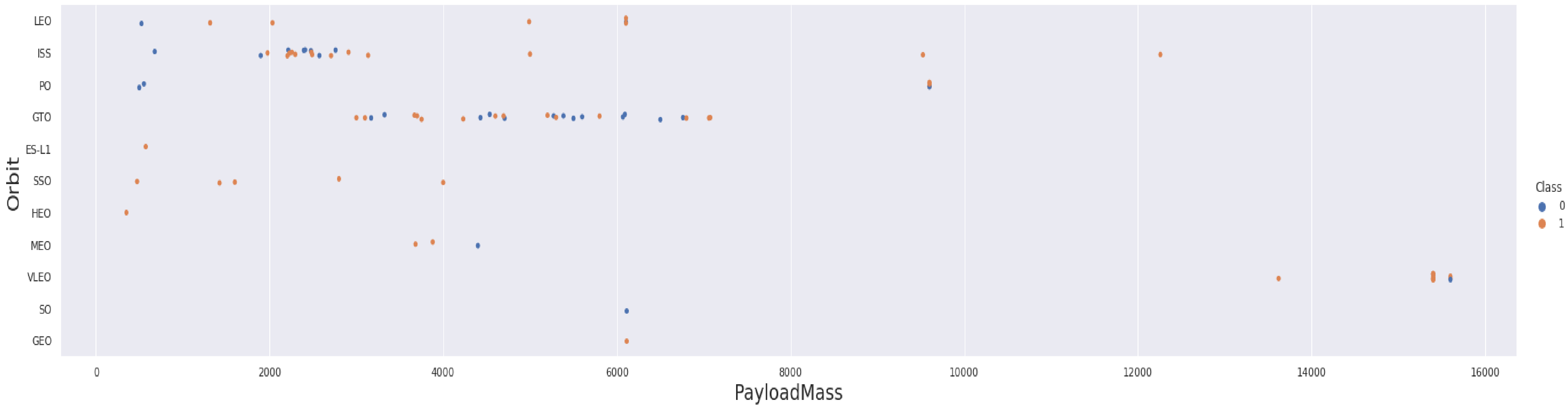

# Success Rate vs. Orbit Type

- From the bar chart, it can be seen that Orbit GEO, HEO, SSO, ES-L1 has the best Success Rate.



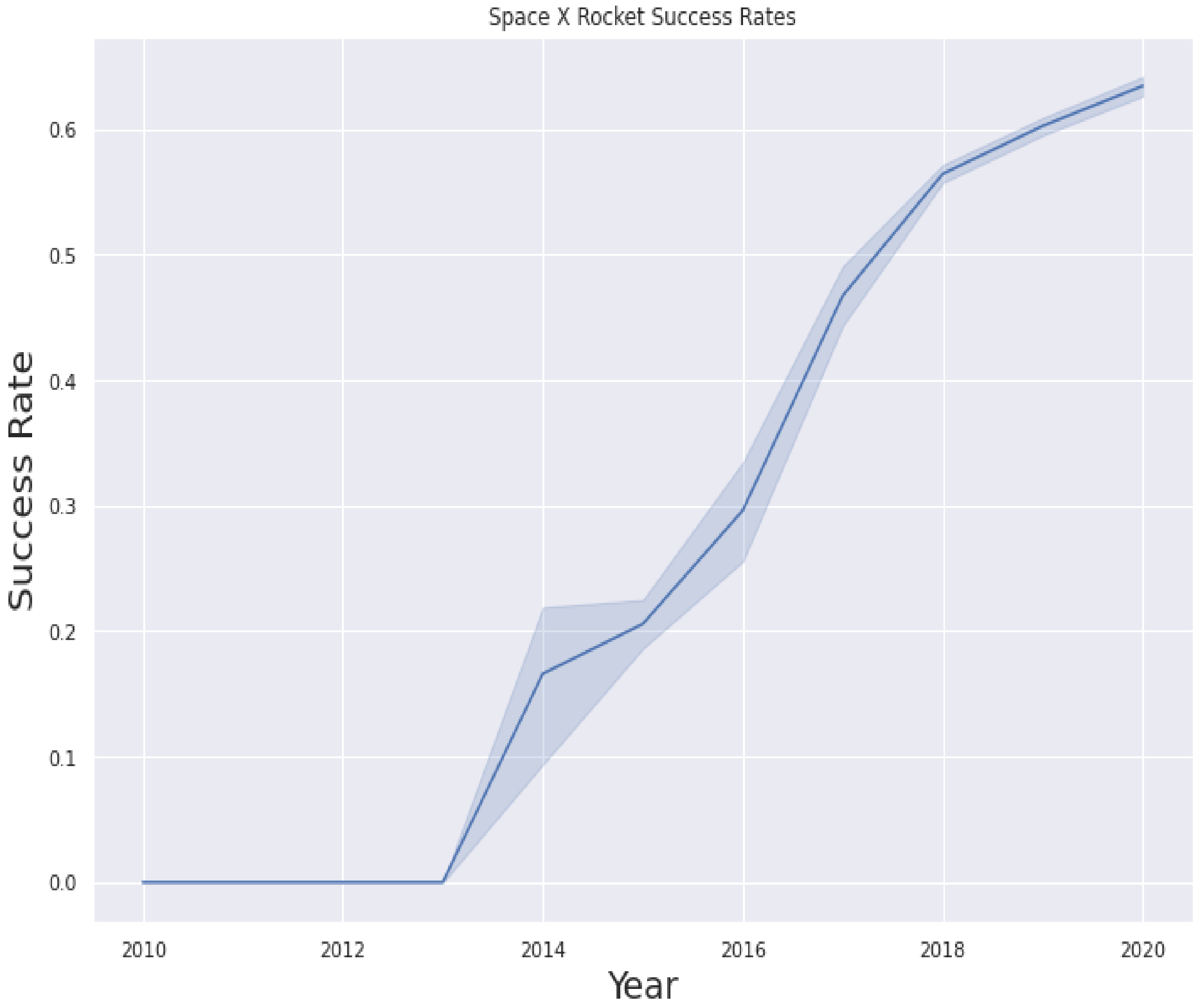

# Flight Number vs. Orbit Type

- It can be seen 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

- You should observe that Heavy payloads have a negative influence on GTO orbits and positive on SSO, LEO, ISS orbits.




# Launch Success Yearly Trend

- It can be observed that the success rate since 2013 kept increasing all the way to 2020.

### Task 1

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

In [8]:
```
%%sql
SELECT UNIQUE(LAUNCH_SITE) FROM SPACEXDATASET;
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[8]:

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

# All Launch Site Names

QUERY EXPLAINATION

- Using the word DISTINCT OR UNIQUE in the query means that it will only show Unique values in the Launch_Site column from SPACEXDATASET.

# Launch Site Names Begin with 'CCA'

QUERY EXPLAINATION

- Using the word LIMIT 5 in the query means that it will only show 5 records from SPACEXDATASET and LIKE keyword has a wild card with the words '%CCA%' the percentage in the end suggests that the Launch_Site name must start with CCA.

### Task 2

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

In [9]:
```
%%sql
SELECT * FROM SPACEXDATASET
WHERE LAUNCH_SITE LIKE '%CCA%'
LIMIT 5;
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[9]:

| DATE | Time (UTC) | booster_version | launch_site | payload | payload_mass__kg_ | orbit | customer | mission_outcome | Landing _Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 2010-06-04 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success | Failure (parachute) |
| 2010-12-08 | 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 |
| 2012-10-08 | 00:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success | No attempt |
| 2013-03-01 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677 | LEO (ISS) | NASA (CRS) | Success | No attempt |

### Task 3

# Total Payload Mass

QUERY EXPLAINATION

- Using the function SUM summates the total in the column PAYLOAD_MASS_KG_ The WHERE clause filters the dataset to only perform calculations on Customer = NASA (CRS)

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

In [10]:

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXDATASET
WHERE CUSTOMER = 'NASA (CRS)';
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[10]:

| 1 |
| --- |
| 45596 |

# Average Payload Mass by F9 v1.1

QUERY EXPLAINATION

- Using the function AVG works out the average in the column PAYLOAD_MASS_KG_ The WHERE clause filters the dataset to only perform calculations on Booster_version LIKE %F9 v1.1%

**Task 4**

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

In [17]:
```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXDATASET
WHERE BOOSTER_VERSION LIKE '%F9 v1.1%';
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[17]:

| 1 |
|---|
| 2534 |

# First Successful Ground Landing Date

QUERY EXPLAINATION

- Using the function MIN works out the minimum date in the column Date The GROUP BY clause filters the dataset to only perform calculations on Landing_Outcome
- The first successful Ground Landing Date is 2015-12-22

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

*Hint:Use min function*

In [41]:
```
%%sql
SELECT MIN(DATE), "Landing _Outcome" FROM SPACEXDATASET
GROUP BY "Landing _Outcome"  ;
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[41]:

| 1 | Landing _Outcome |
|---|---|
| 2014-04-18 | Controlled (ocean) |
| 2018-12-05 | Failure |
| 2015-01-10 | Failure (drone ship) |
| 2010-06-04 | Failure (parachute) |
| 2012-05-22 | No attempt |
| 2015-06-28 | Precluded (drone ship) |
| 2018-07-22 | Success |
| 2016-04-08 | Success (drone ship) |
| 2015-12-22 | Success (ground pad) |
| 2013-09-29 | Uncontrolled (ocean) |

# Successful Drone Ship Landing with Payload between 4000 and 6000

QUERY EXPLAINATION

- Selecting only Booster_Version The WHERE clause filters the dataset to Landing_Outcome = Success (drone ship) AND clause specifies additional filter conditions Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 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***

In [44]:

```
%%sql
SELECT BOOSTER_VERSION FROM SPACEXDATASET
WHERE "Landing _Outcome" = 'Success (drone ship)' AND
PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[44]:

| booster_version |
|---|
| F9 FT B1022 |
| F9 FT B1026 |
| F9 FT B1021.2 |
| F9 FT B1031.2 |

# Total Number of Successful and Failure Mission Outcomes

## QUERY EXPLAINATION

- Selected the count of the Mission Outcome grouping them by Successful and Failure Mission Outcomes.

### Task 7

***List the total number of successful and failure mission outcomes***

In [47]:
```
%%sql
SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS NUMBER_OF_OUTCOMES FROM SPACEXDATASET
GROUP BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[47]:

| mission_outcome | number_of_outcomes |
|---|---|
| Failure (in flight) | 1 |
| Success | 99 |
| Success (payload status unclear) | 1 |

# Boosters Carried Maximum Payload

QUERY EXPLAINATION

- Using Subquery as a clause where Payload = selecting maximum payload from SPACEXDATASET. Then Booster_version are now selected based on the maximum payload.

**Task 8**

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

```
In [54]: %%sql
SELECT BOOSTER_VERSION FROM SPACEXDATASET
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATASET) ;

 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[54]:

| booster_version |
|---|
| F9 B5 B1048.4 |
| F9 B5 B1049.4 |
| F9 B5 B1051.3 |
| F9 B5 B1056.4 |
| F9 B5 B1048.5 |
| F9 B5 B1051.4 |
| F9 B5 B1049.5 |
| F9 B5 B1060.2 |
| F9 B5 B1058.3 |
| F9 B5 B1051.6 |
| F9 B5 B1060.3 |
| F9 B5 B1049.7 |

# 2015 Launch Records

QUERY EXPLAINATION

- Selecting Landing_Outcome in drone_ship,their booster version and launch site names WHERE clause filters Year to be 2015

### Task 9

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

In [55]:
```
%%sql
SELECT "Landing _Outcome", BOOSTER_VERSION, LAUNCH_SITE  FROM SPACEXDATASET
WHERE "Landing _Outcome" = 'Failure (drone ship)' AND YEAR(DATE) = 2015;
```

```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[55]:

| Landing _Outcome | booster_version | launch_site |
|---|---|---|
| Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40 |
| Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40 |

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

- QUERY EXPLAINATION
- Function COUNT counts landing outcome column WHERE Date is filtered Between 2010-06-04 and 2017-03-20. Then grouped by the landing outcome and then arranged in Descending Order.

### Task 10

***Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order***

In [65]:

```
%%sql
SELECT "Landing _Outcome", COUNT("Landing _Outcome") AS NUMBER_OF_LANDING FROM SPACEXDATASET
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing _Outcome"
ORDER BY NUMBER_OF_LANDING DESC;
```

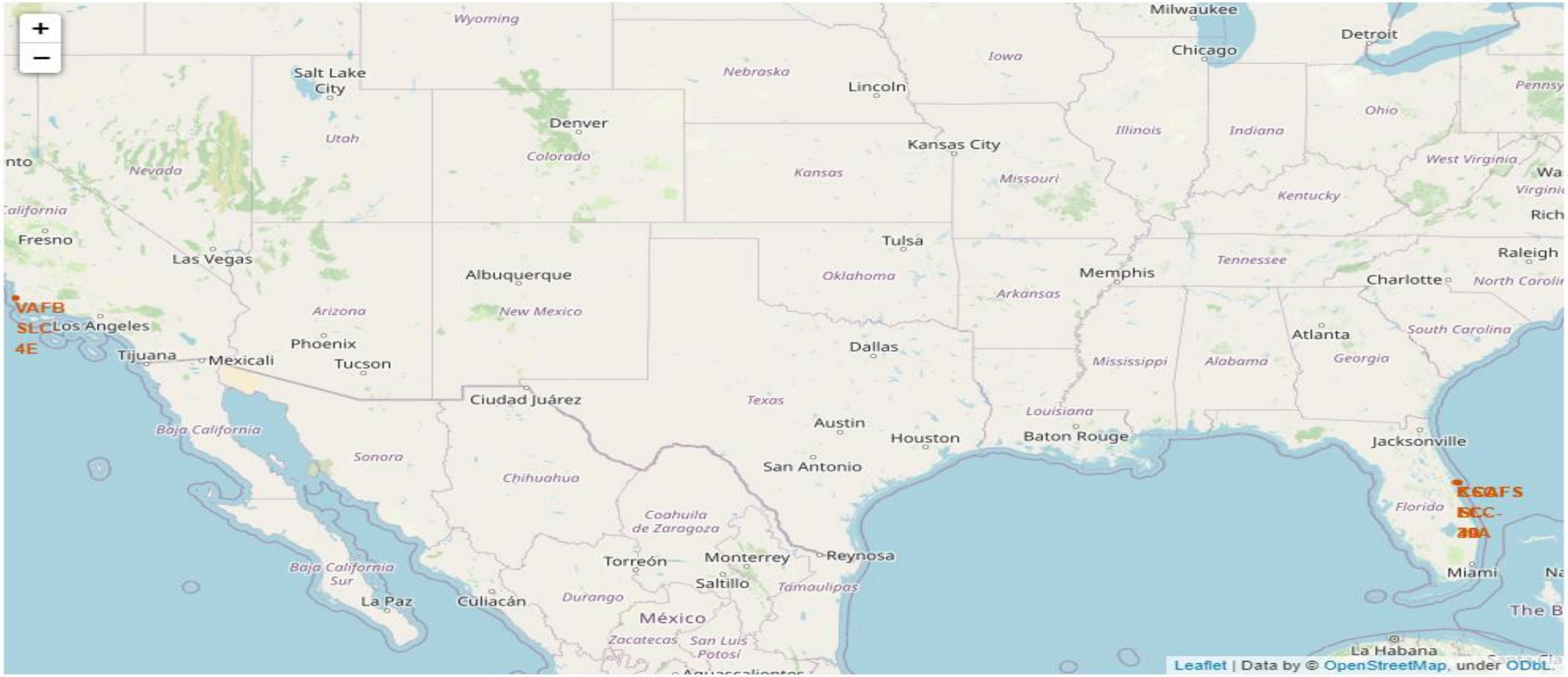
```
 * ibm_db_sa://wff24947:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[65]:

| Landing _Outcome | number_of_landing |
|---|---|
| No attempt | 10 |
| Failure (drone ship) | 5 |
| Success (drone ship) | 5 |
| Controlled (ocean) | 3 |
| Success (ground pad) | 3 |
| Failure (parachute) | 2 |
| Uncontrolled (ocean) | 2 |
| Precluded (drone ship) | 1 |

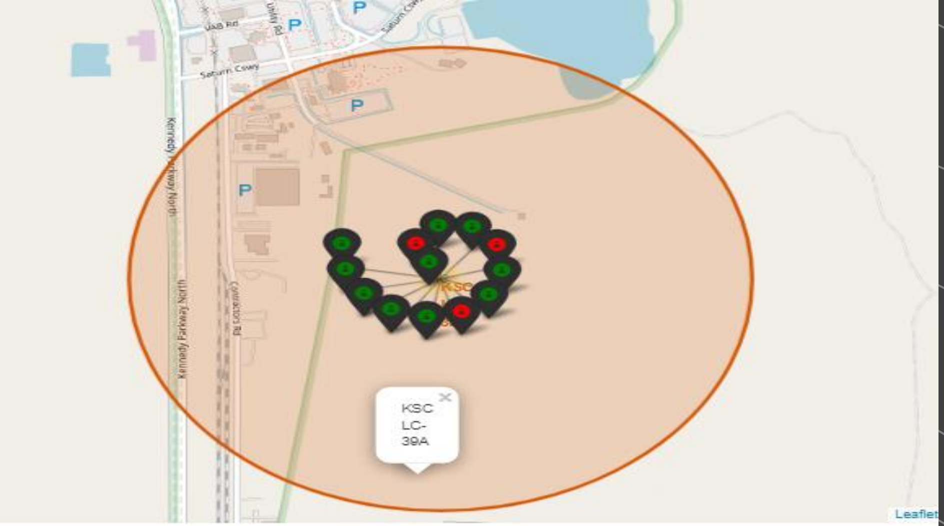
Section 4

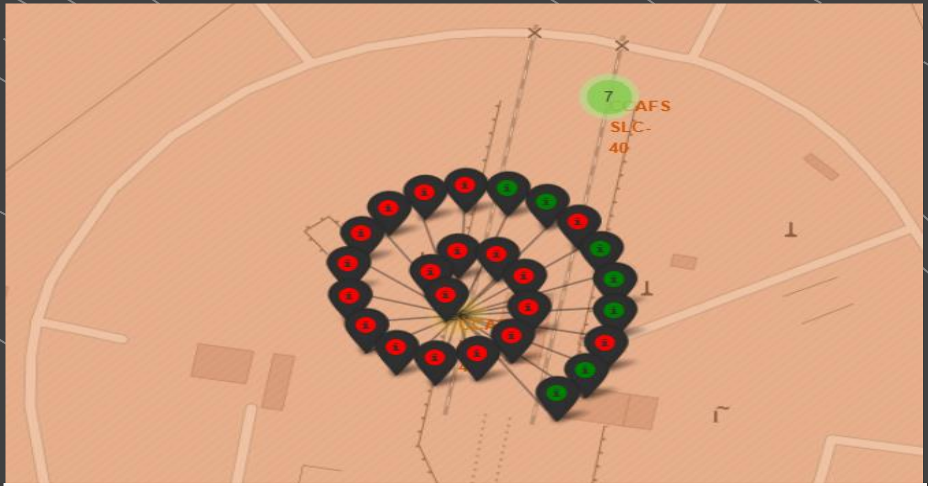
# Launch Sites Proximities Analysis




# All Launch Site on the Global Map

- We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

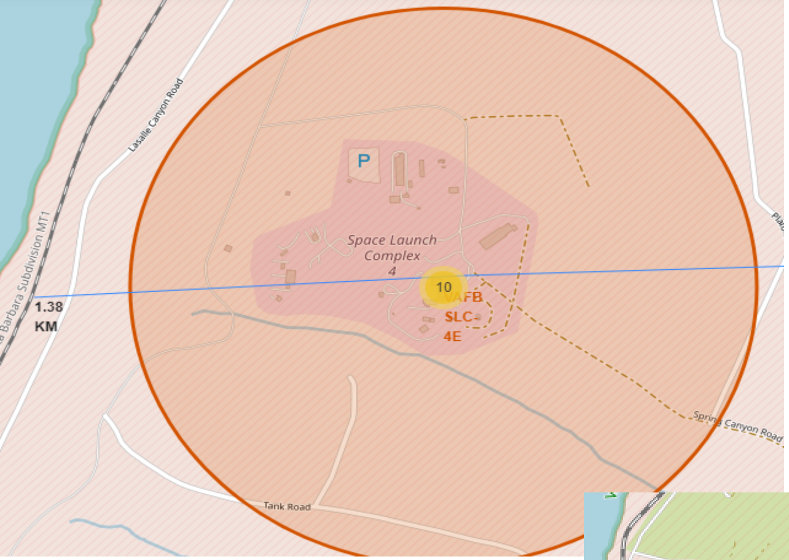


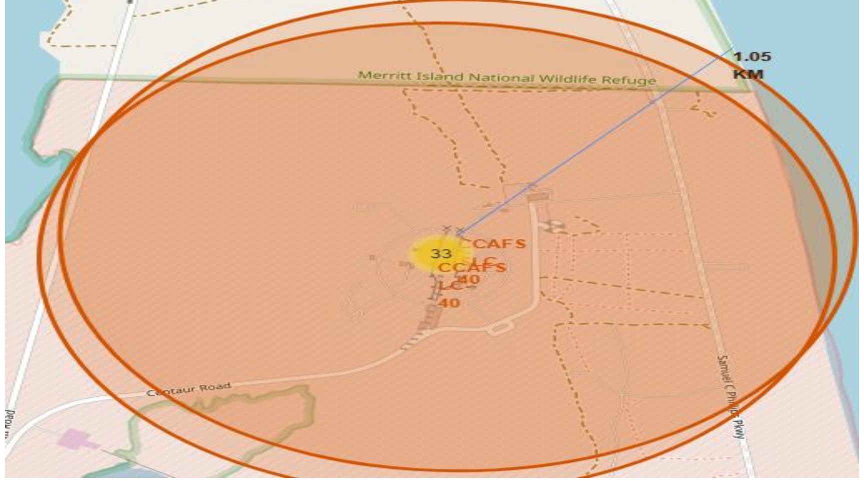



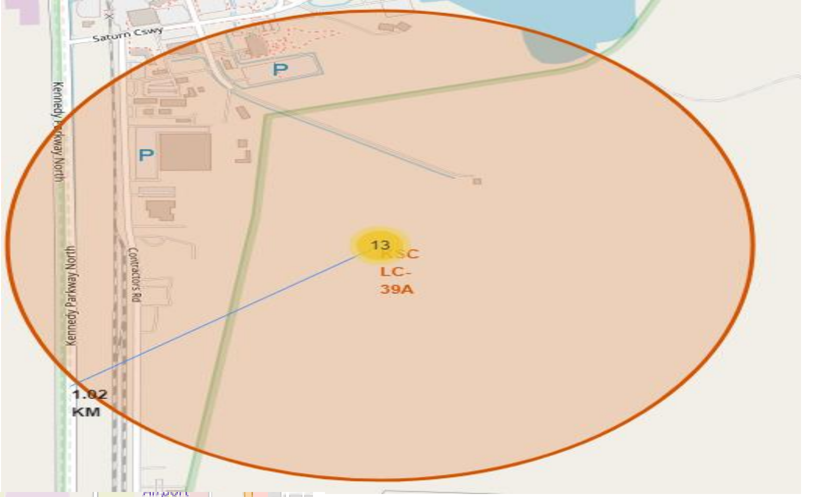


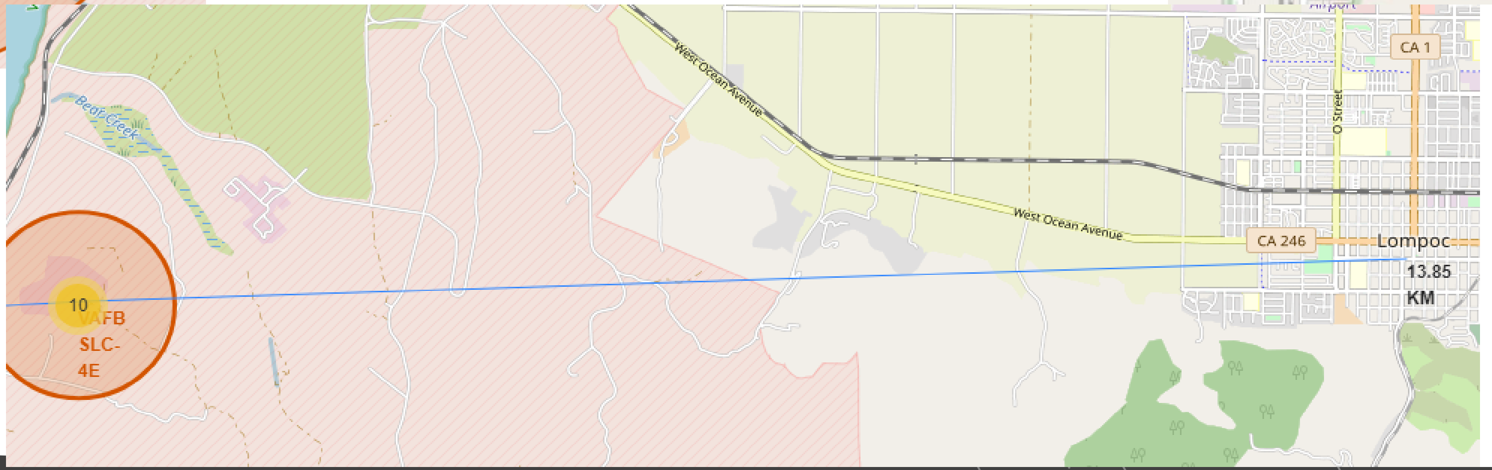

# Colour Labelled Markers

- Green Marker shows successful Launches and Red Marker shows Failures



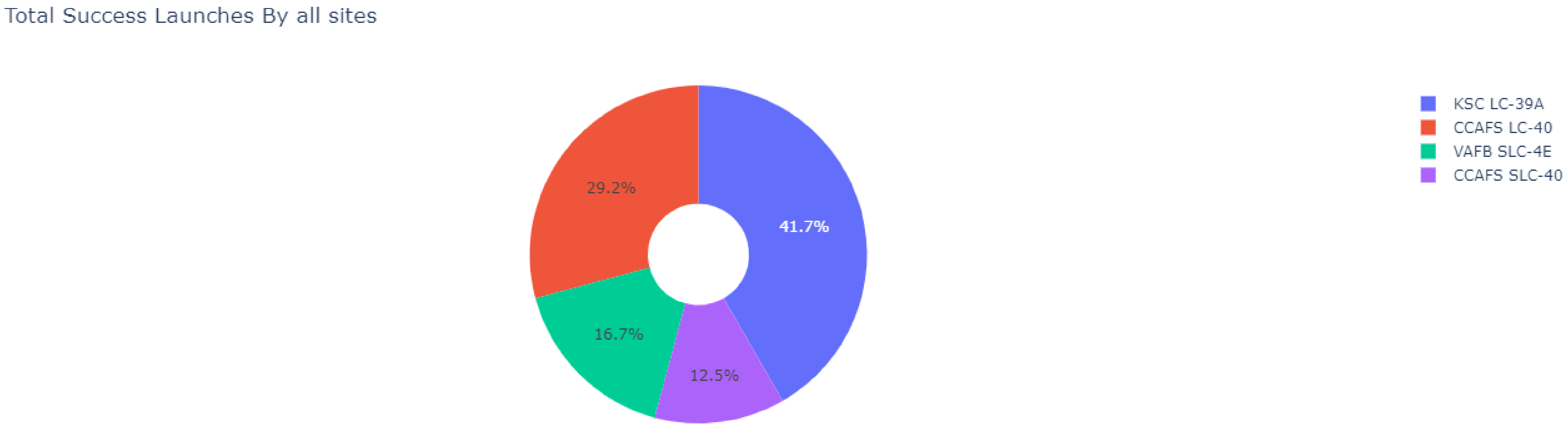




# Launch Sites distance to landmarks to find trends with Haversine formula using CCAFS-SLC-40 as a reference

- •Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

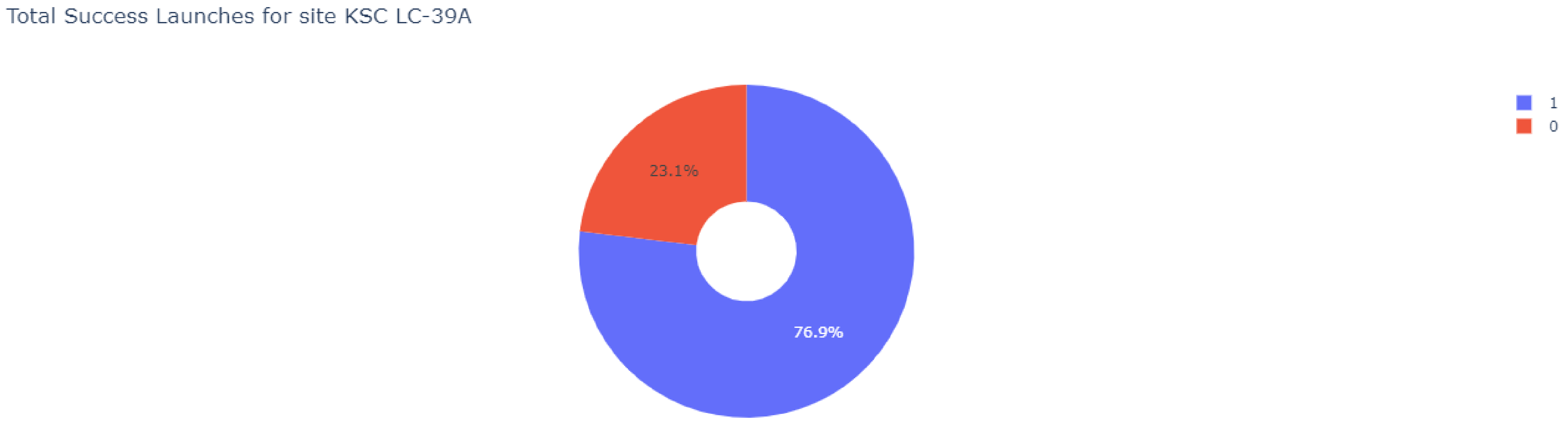
Section 5

# Build a Dashboard with Plotly Dash




# Total Launch Success for All Sites

It can be seen that KSC LC-39A had the most successful launches from all the sites.

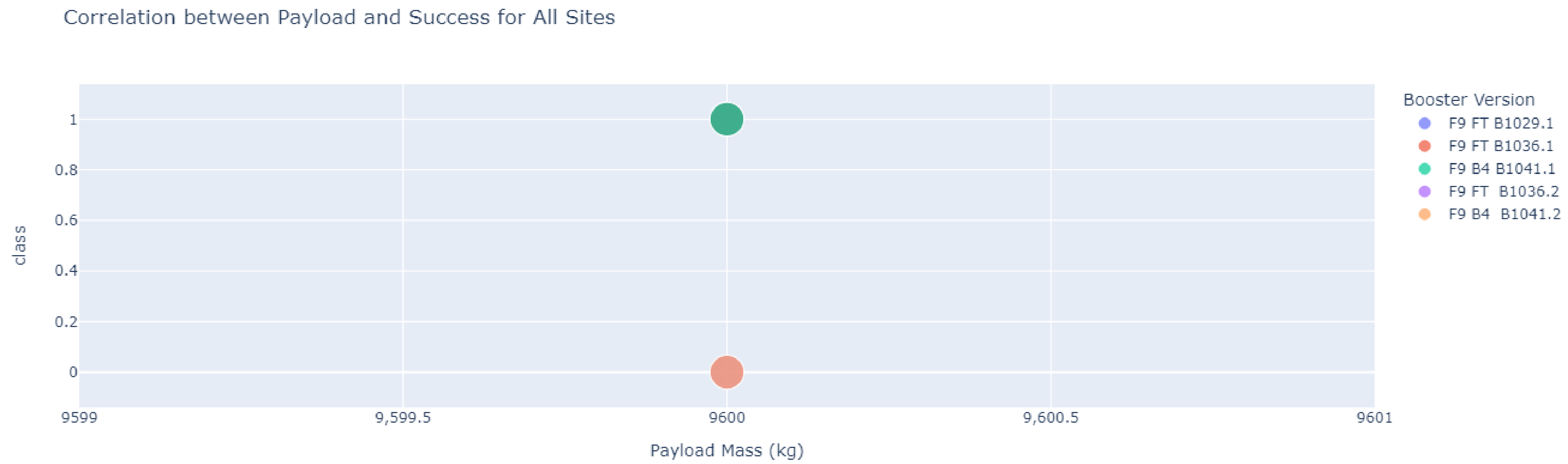



# Total Success for Site KSC LC 39A

KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate.

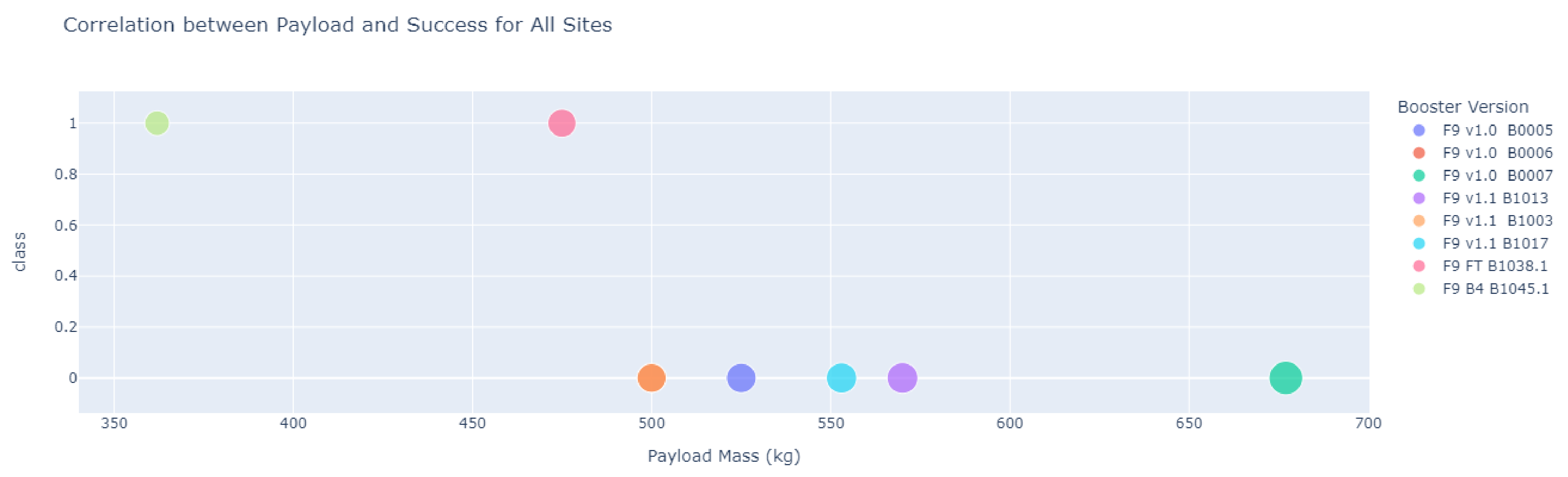

# Payload vs Launch Outcome for All Sites

- From the scatter plot it can be seen that there are more success rate when the Payload is low(< 5k). But as the Payload increases the success rate decreases.




# Payload vs Launch Outcome for All Sites

- Booster version F9 FT B1041.1 is the success rate with the highest Payload.




# Payload vs Launch Outcome for All Sites

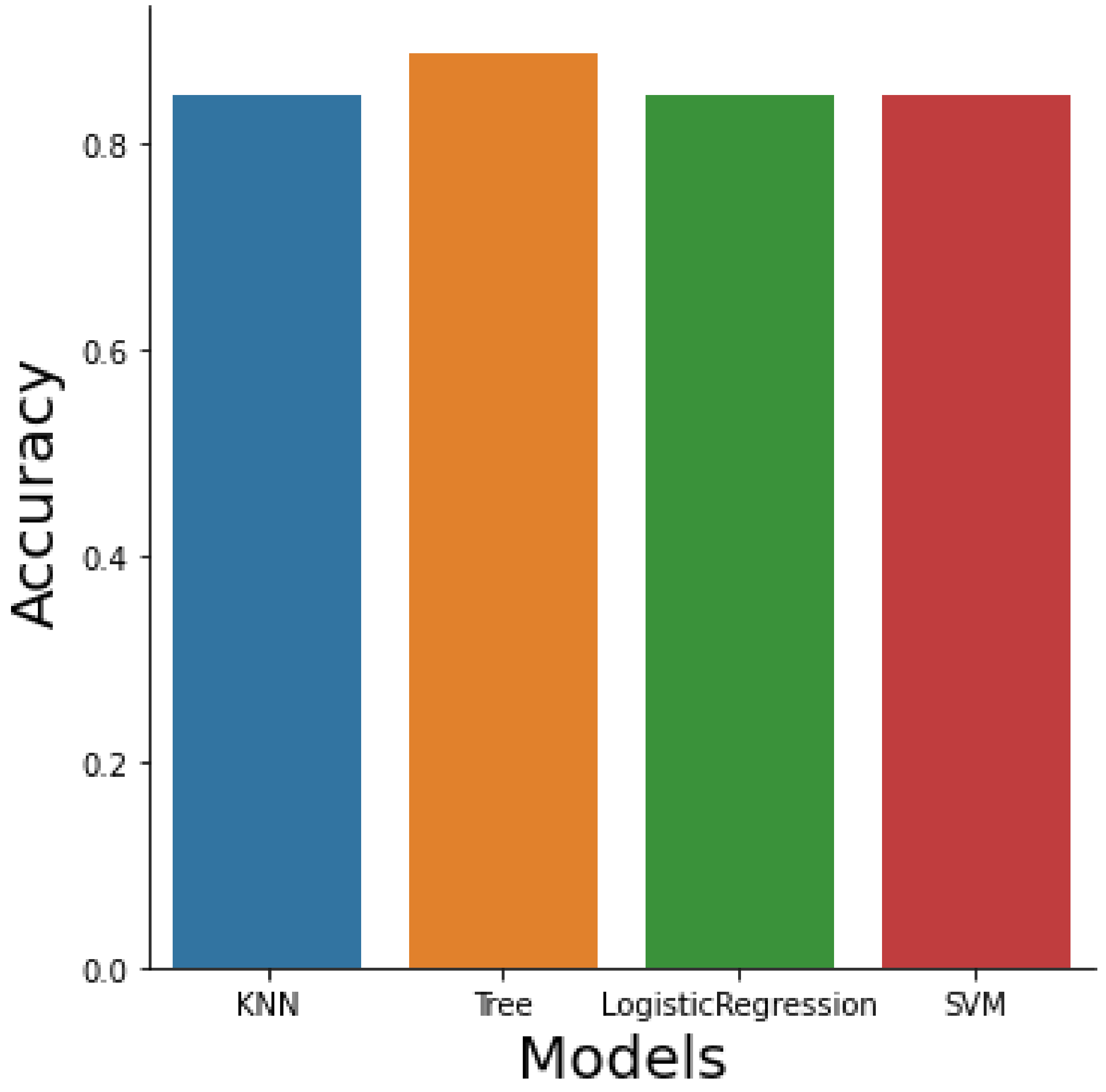
- Booster version F9 B4 B1045.1 is the success rate with the lowest Payload.

Section 6

# Predictive Analysis (Classification)

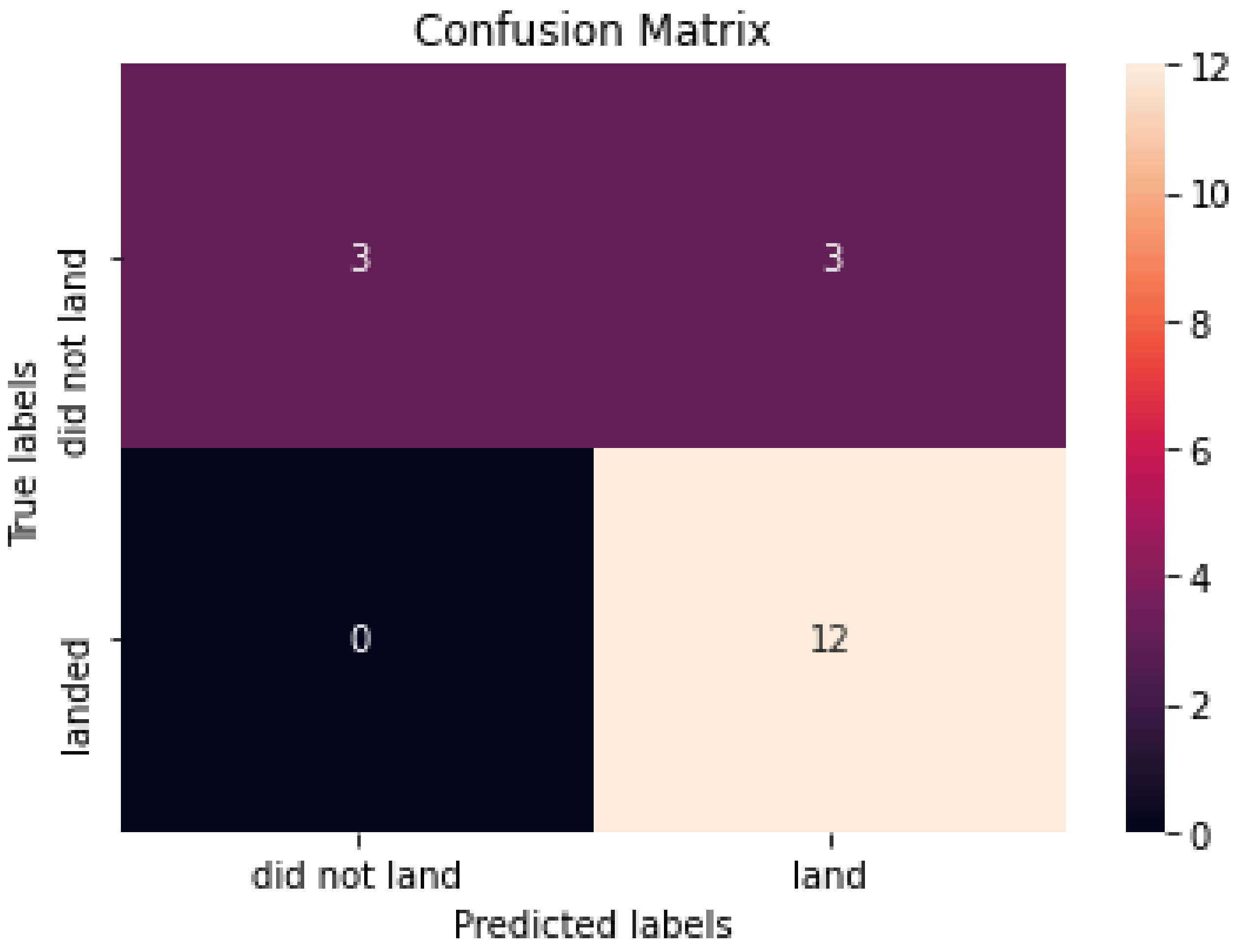
# Classification Accuracy

- From the bar chart we can see that Tree Classification model has the highest accuracy with an accuracy of over 0.8.



# Confusion Matrix

- The best performing model is the Tree classification model.
- From the confusion matrix out of the 18 records were used for the model prediction
- 12 to be True positive (land).
- 3 to be True Negative ( did not land).
- 3 to be False positive (land which is wrong).



# Conclusions

- The Tree Classifier Algorithm is the best for Machine Learning for the SpaceX Falcon9 dataset.
- Low weighted payloads have more success rate than the heavier payloads.
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches.
- We can see that KSC LC-39A had the most successful launches from all the sites.
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate.

# Appendix

- Haversine formula
- Module sqlserver
- Dashbaord with Dash

# Thank you!