Winning Space Race with Data Science

Lulwah AlSuwaidan
Jan 14, 2022

Outline

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

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

Section 1

Methodology

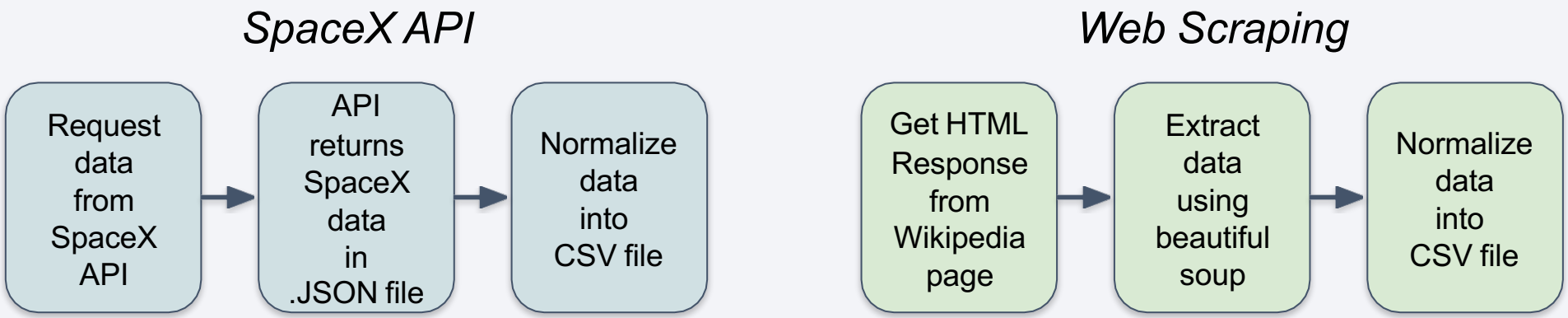
Methodology

Executive Summary

- **Data collection methodology:**
  - SpaceX Rest API
  - (Web Scrapping) from Wikipedia
- **Performed data wrangling (Transforming data for Machine Learning)**
  - One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- **Performed exploratory data analysis (EDA) using visualization and SQL**
  - Plotting : Scatter Graphs, Bar Graphs to show relationships between variables to show patterns of data.
- **Performed interactive visual analytics**
  - using Folium and Plotly Dash
- **Performed predictive analysis using classification models**
  - How to build, tune, evaluate classification models

Data Collection

- The data collection process includes a combination of API requests from the SpaceX API and web scraping data from a table in the Wikipedia page of SpaceX, *Falcon 9 and Falcon Heavy Launches Records*.
  - SpaceX API Data Columns: `FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude`
  - Wikipedia Web Scrape Data Columns: `Flight No., Launch site, Payload, PayloadMass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time`



Data Collection – SpaceX API

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

```
In [7]: spacex_url="https://api.spacexdata.com/v4/launches/past"
In [8]: response = requests.get(spacex_url)
```

Now, let's apply `getBoosterVersion` function method to get the booster version

```
In [23]: # Call getBoosterVersion
         getBoosterVersion(data)
```

the list has now been update

```
In [24]: BoosterVersion[0:5]
Out[24]: ['Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 9']
```

we can apply the rest of the functions here:

```
In [25]: # Call getLaunchSite
         getLaunchSite(data)
In [26]: # Call getPayloadData
         getPayloadData(data)
In [27]: # Call getCoreData
         getCoreData(data)
```

```
In [28]: launch_dict = {'FlightNumber': list(data['flight_number']),
         'Date': list(data['date']),
         'BoosterVersion':BoosterVersion,
         'PayloadMass':PayloadMass,
         'Orbit':Orbit,
         'LaunchSite':LaunchSite,
         'Outcome':Outcome,
         'Flights':Flights,
         'GridFins':GridFins,
         'Reused':Reused,
         'Legs':Legs,
         'LandingPad':LandingPad,
         'Block':Block,
         'ReusedCount':ReusedCount,
         'Serial':Serial,
         'Longitude': Longitude,
         'Latitude': Latitude}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
In [29]: # Create a data from launch_dict
         launch_dict_df = pd.DataFrame.from_dict(launch_dict)
```

```
data_falcon9 = launch_dict_df[launch_dict_df['BoosterVersion']!='Falcon 1']
data_falcon9
```

Getting Responses from API → Covert Response to json file → Apply Custom Function to clean data → Assign list to dictionary then create dataframe → Filter dataframe and export file

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

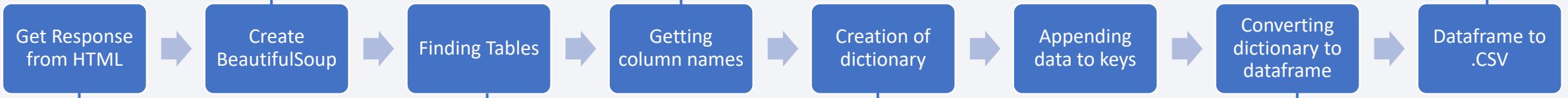
```
# Use json_normalize meethod to convert the json result into a dataframe
# Flatten data
r = requests.get(static_json_url)
r.json()

data = pd.json_normalize(r.json())
data
```

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```

Data Collection - Scraping




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

```
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
```

```
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
```

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

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

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

```
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']=[]
```

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

Data Wrangling

Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose [Data Wrangling GitHub]

- **Description:**

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

Process

Perform Exploratory Data Analysis EDA on dataset

Calculate the number of launches at each site

Calculate the number and occurrence of each orbit

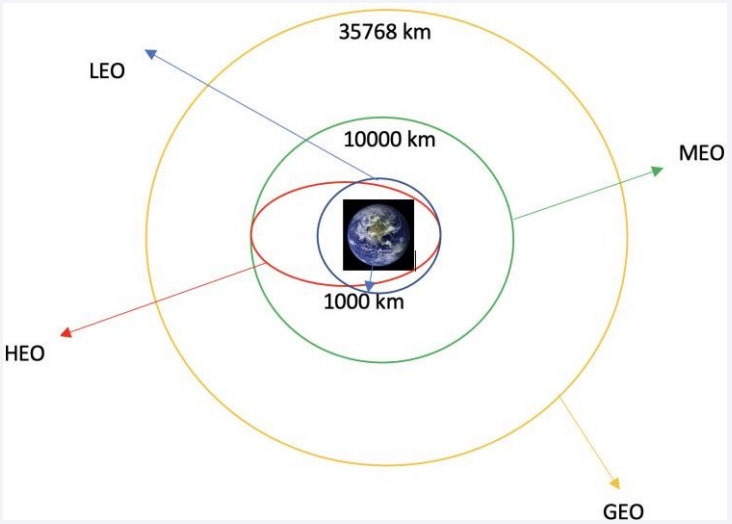
Calculate the number and occurrence of mission outcome per orbit type

Export dataset as .CSV

Create a landing outcome label from Outcome column

Work out success rate for every landing in dataset

Each launch aims to an dedicated orbit, and here are some common orbit types:



EDA with Data Visualization

Data Viz GitHub

Scatter Graphs being drawn:

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 Graph being drawn:

Mean VS. Orbit

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 Graph being drawn:

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

EDA with SQL

EDA with SQL

- Performed SQL queries to gather information about the dataset.
- For example of some questions we were asked about the data we needed information about. Which we are using SQL queries to get the answers in the dataset :
- Displaying the names of the unique launch sites in the space mission
- Displaying 5 records where launch sites begin with the string 'KSC'
- Displaying the total payload mass carried by boosters launched by NASA (CRS)
- Displaying average payload mass carried by booster version F9 v1.1
- Listing the date where the successful landing outcome in drone ship was achieved.
- Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
- Listing the total number of successful and failure mission outcomes
- Listing the names of the booster_versions which have carried the maximum payload mass.
- Listing the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
- Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

Build an Interactive Map with Folium

Visual Analytics 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

- **Example of some trends in which the Launch Site is situated in.**

•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

Build a Dashboard with Plotly Dash

Dashboard

- The dashboard is built with Flask and Dash web framework.
- Graphs
  - - Pie Chart showing 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 Graph showing 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.

Predictive Analysis (Classification) ML Prediction

- **BUILDING MODEL**
  - Load our dataset into NumPy and Pandas
  - Transform Data
  - Split our data into training and test data sets
  - Check how many test samples we have
  - Decide which type of machine learning algorithms we want to use
  - Set our parameters and algorithms to GridSearchCV
  - Fit our datasets into the GridSearchCV objects and train our dataset.

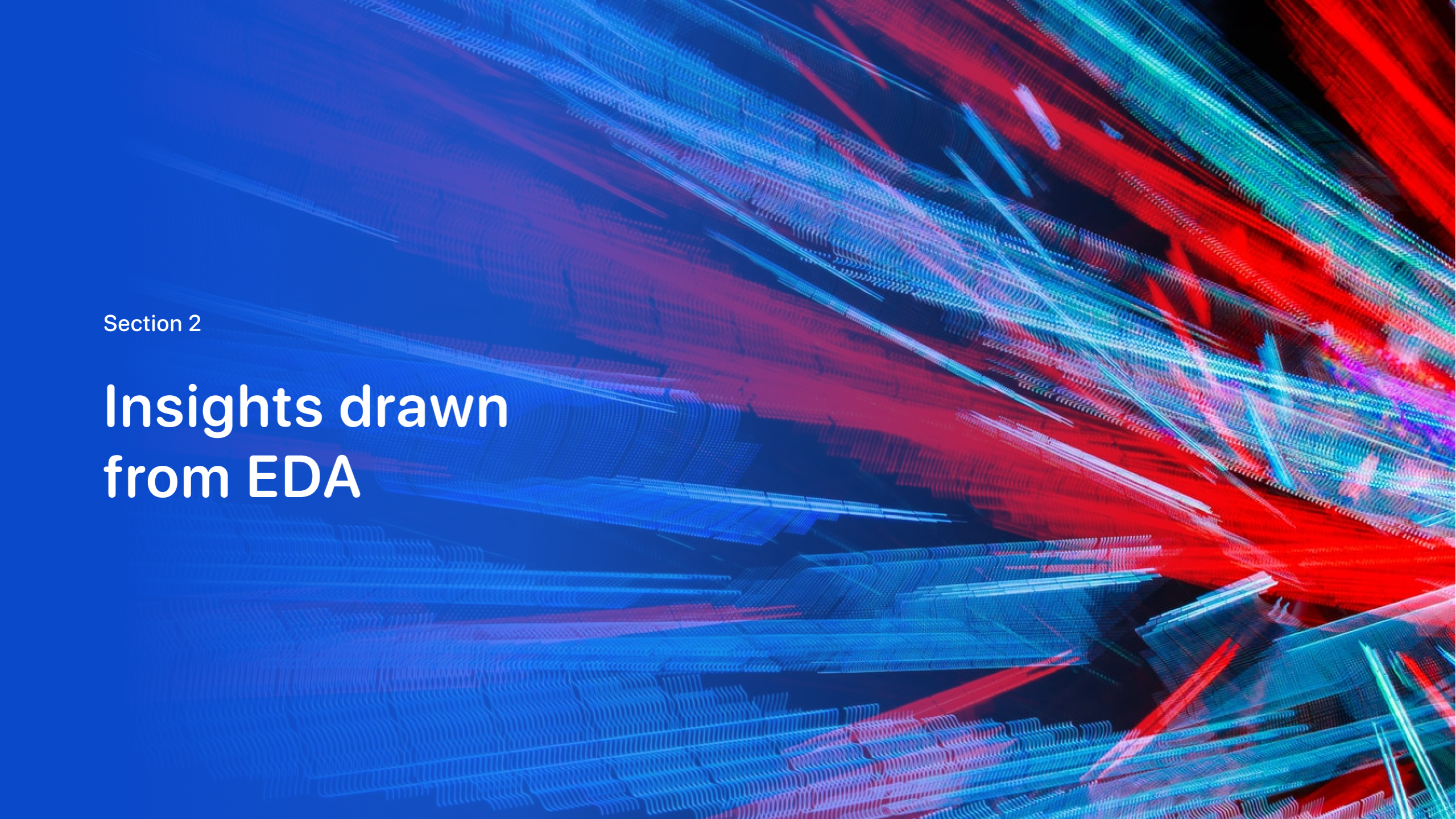
- **EVALUATING MODEL**
  - Check accuracy for each model
  - Get tuned hyperparameters for each type of algorithms
  - Plot Confusion Matrix

- **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.

Results

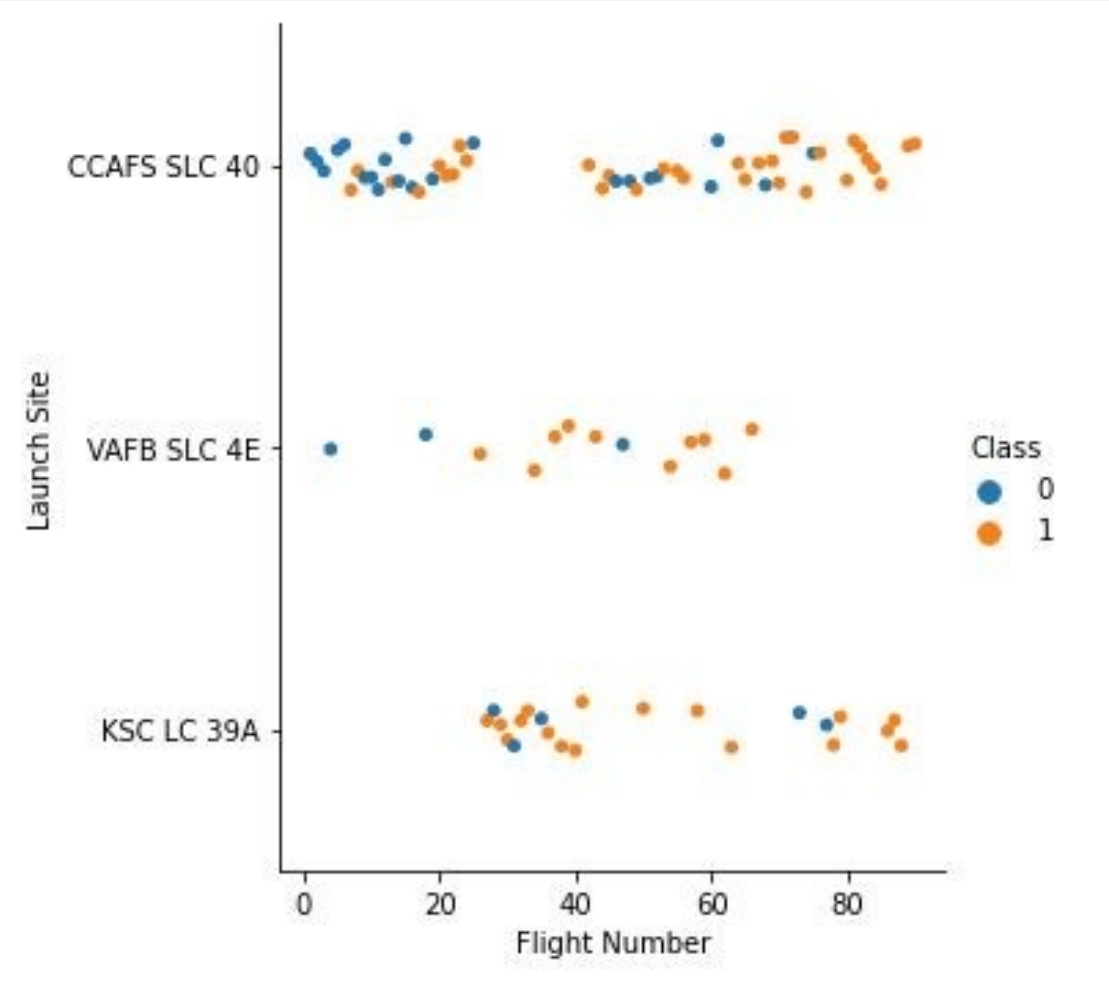
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

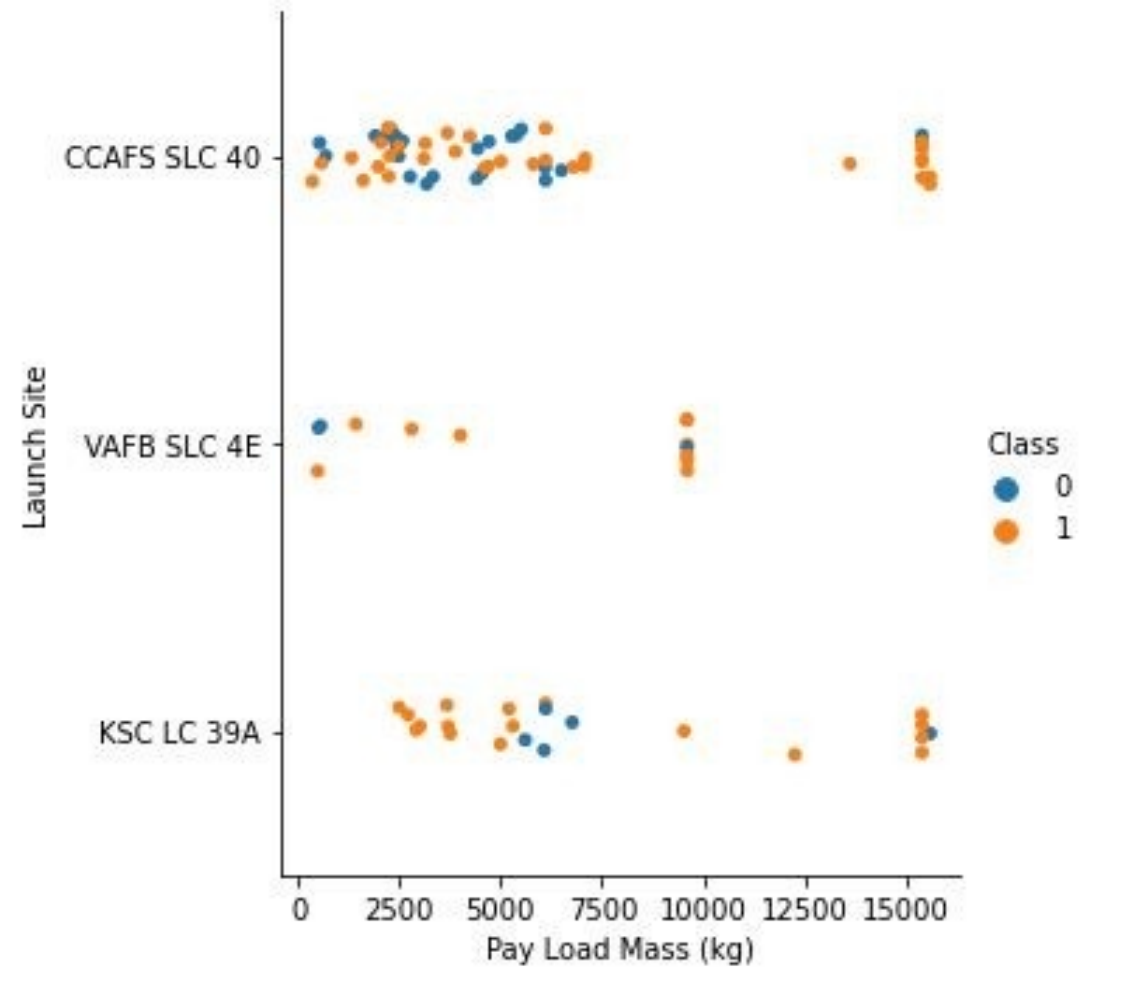
Insights drawn from EDA

Flight Number vs. Launch Site



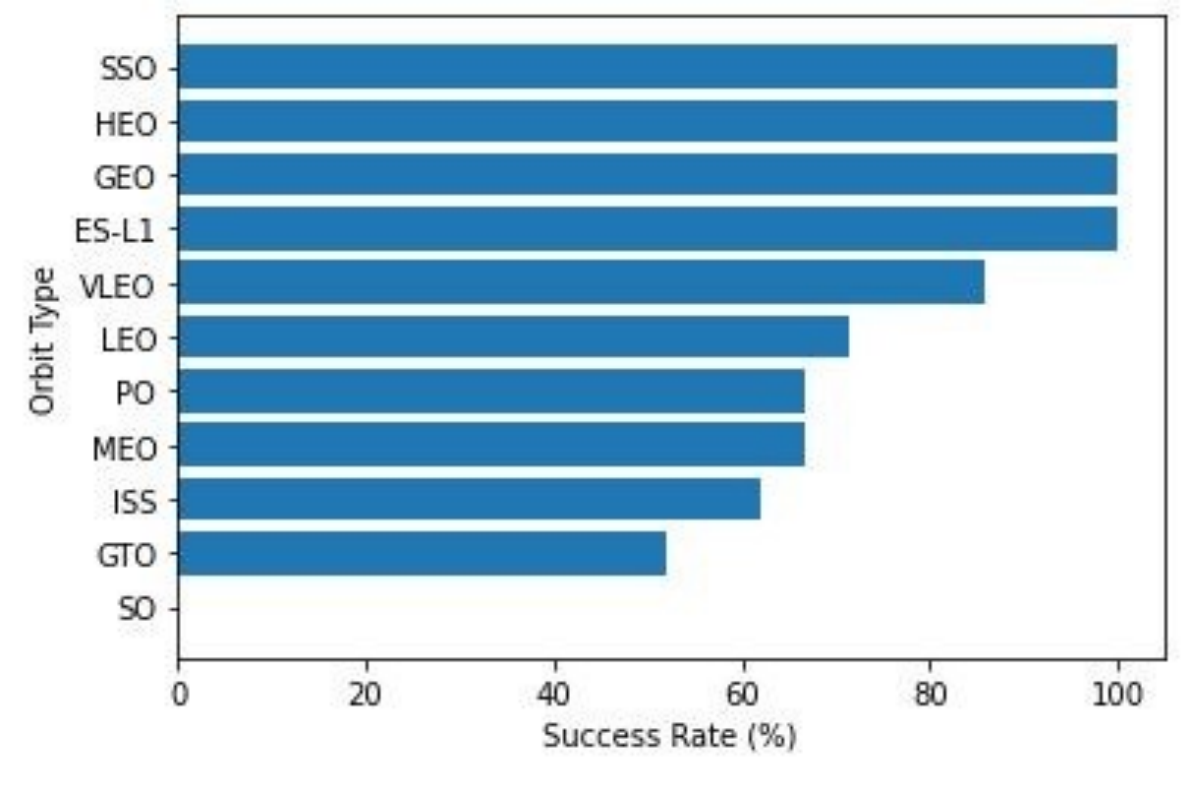
- `Class 0` (blue) represents unsuccessful launch, and `Class 1` (orange) represents successful launch.
- This figure shows that **the success rate increased as the number of flights increased**.
- As the success rate has increased considerably since the *20th* flight, this point seems to be a big breakthrough.

Payload vs. Launch Site



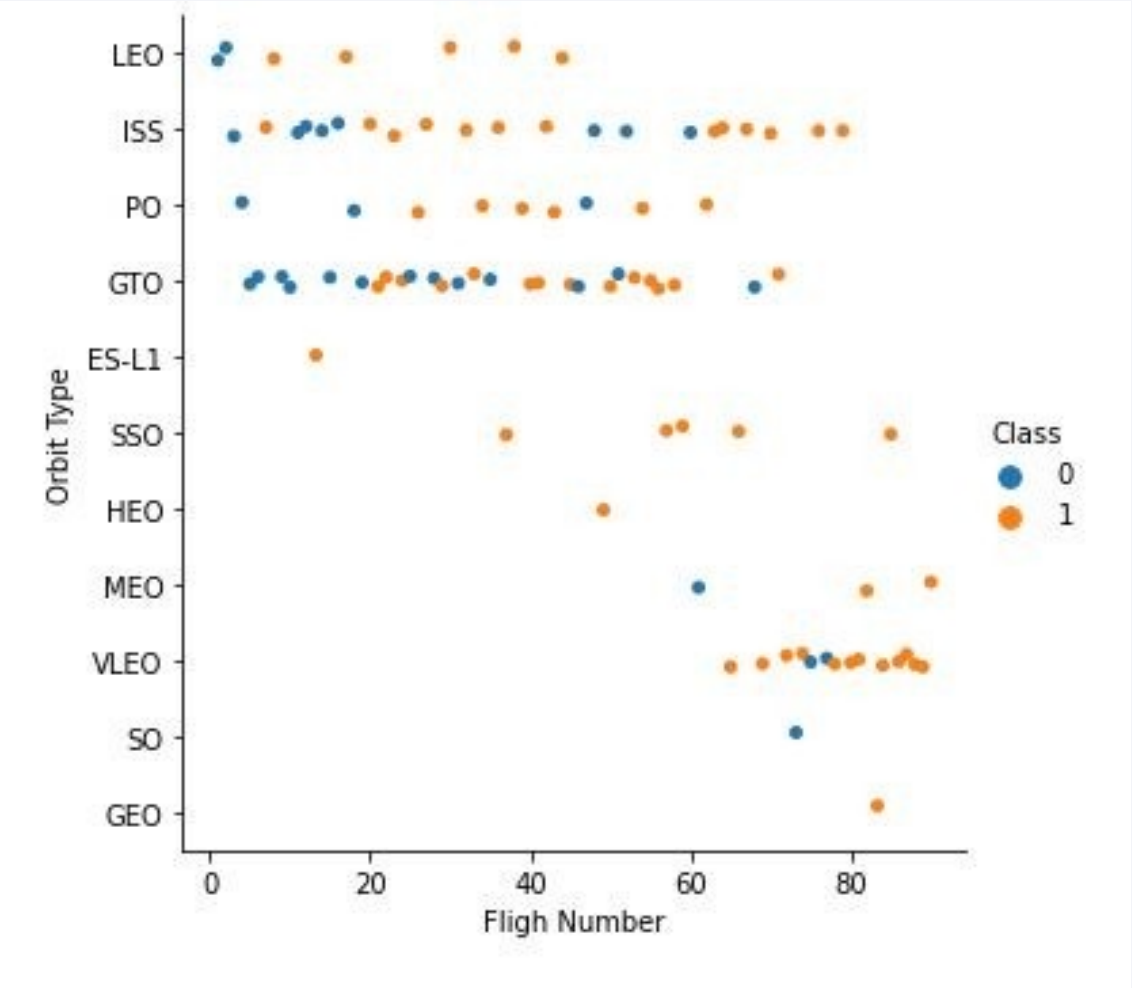
- `Class 0` (blue) represents unsuccessful launch, and `Class 1` (orange) represents successful launch.
- At first glance, the larger pay load mass, the higher the rocket's success rate, but it seems difficult to make decisions based on this figure because **no clear pattern can be found between successful launch and Pay Load Mass**.

Success Rate vs. Orbit Type



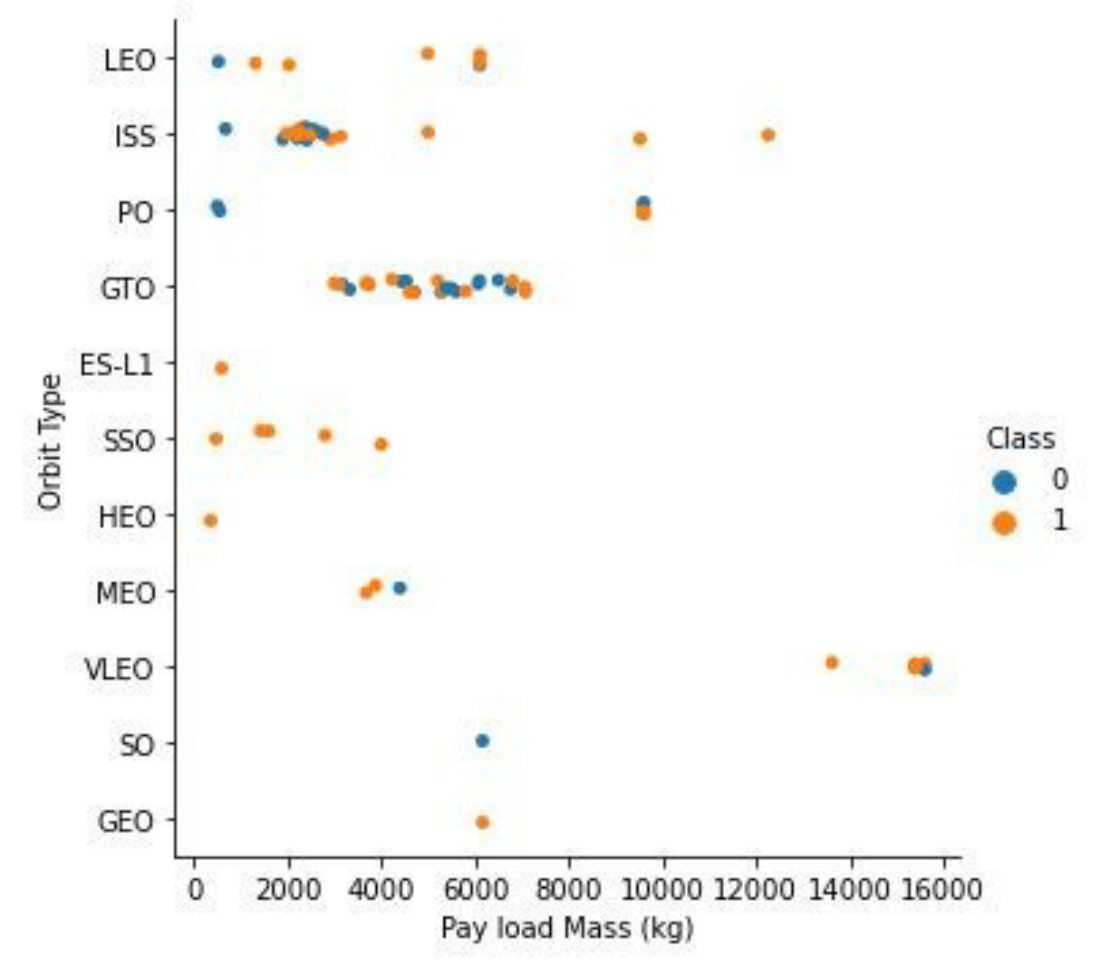
- Orbit types **SSO**, **HEO**, **GEO**, and **ES-L1** have **the highest success rates** (*100%*).
- On the other hand, the success rate of orbit type **GTO** is only *50%*, and it is the **lowest** except for type SO, which recorded failure in a single attempt.

Flight Number vs. Orbit Type



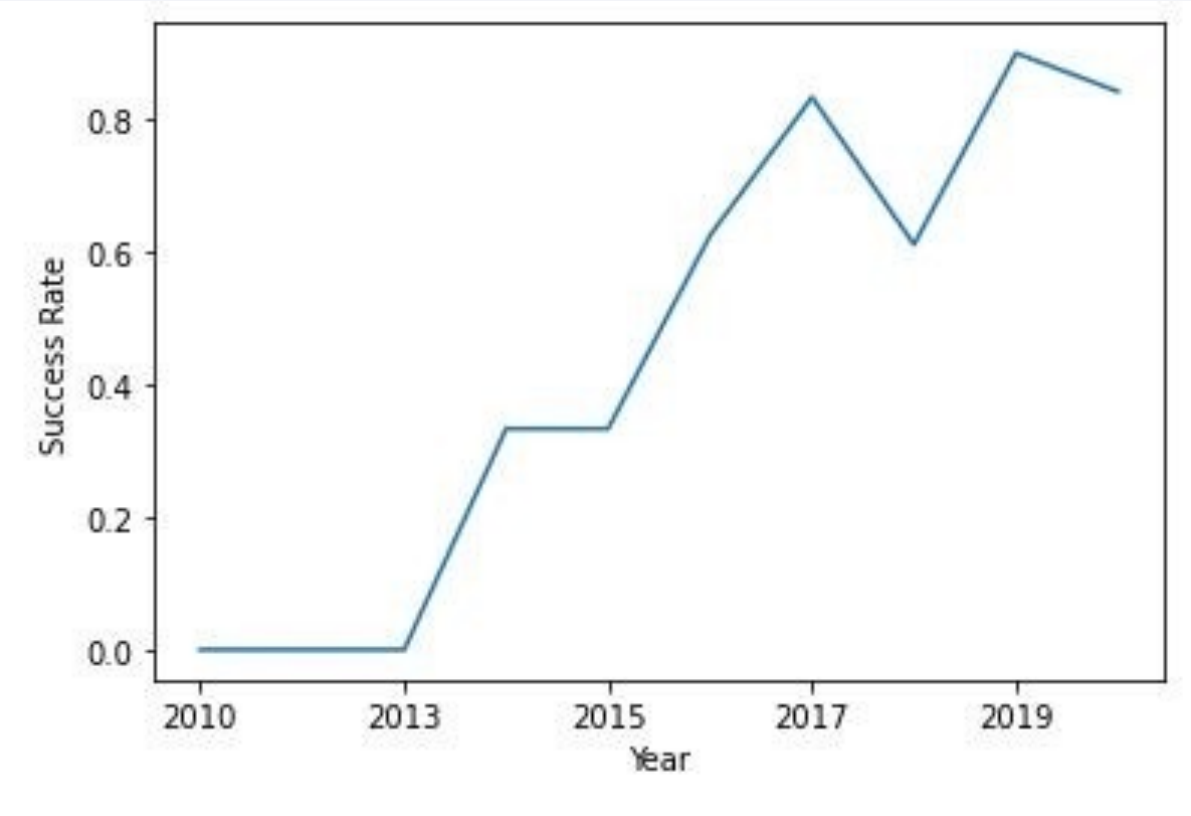
- `Class 0` (blue) represents unsuccessful launch, and `Class 1` (orange) represents successful launch.
- In most cases, the launch outcome seems to be correlated with the flight number.
- On the other hand, in **GTO** orbit, there seems to be **no** relationship between flight numbers and success rate.
- SpaceX starts with LEO with a moderate success rate, and it seems that VLEO, which has a high success rate, is used the most in recent launches.

Payload vs. Orbit Type



- `Class 0` (blue) represents unsuccessful launch, and `Class 1` (orange) represents successful launch.
- With heavy payloads the successful landing or positive landing rate are more for LEO and ISS.
- However, in the case of GTO, it is hard to distinguish between the positive landing rate and the negative landing because they are all gathered together.

Launch Success Yearly Trend



- Since 2013, the success rate has continued to **increase** until 2017.
- The rate decreased slightly in 2018.
- Recently, it has shown a success rate of about *80%*.

All Launch Site Names

- Query

```
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL
```

- Result

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

- When the SQL `DISTINCT` clause is used in the query, only unique values are displayed in the `Launch_Site` column from the `SpaceX` table.
- There are four unique launch sites: CCAFS LC-40, CCAFS SLC-40, KSC LC-39A, VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- Query

```
SELECT * FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE 'CCA%'
LIMIT 5
```

- Result

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

- Only five records of the SpaceX table were displayed using `LIMIT 5` clause in the query.
- Using the `LIKE` operator and the percent sign (%) together, the `Launch_Site` name starting with `CAA` could be called.

Total Payload Mass

- Query

```
SELECT SUM(PAYLOAD_MASS__KG_)
       AS total_payload_mass_kg
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)'
```

- Result

total_payload_mass_kg
45596

- Using the `SUM()` function to calculate the sum of column `PAYLOAD_MASS__KG_`.
- In the `WHERE` clause, filter the dataset to perform calculations only if `Customer` is `NASA (CRS)`.

Average Payload Mass by F9 v1.1

- Query

```
SELECT AVG(PAYLOAD_MASS__KG_)
       AS avg_payload_mass_kg
FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1'
```

- Result

avg_payload_mass_kg
2928

- Using the `AVG()` function to calculate the average value of column `PAYLOAD_MASS__KG_`.
- In the `WHERE` clause, filter the dataset to perform calculations only if `Booster_version` is `F9 v1.1`.

First Successful Ground Landing Date

- Query

```
SELECT MIN(DATE)
  AS first_successful_landing_date
FROM SPACEXTBL
WHERE LANDING__OUTCOME
      = 'Success (ground pad)'
```

- Result

first_successful_landing_date
2015-12-22

- Using the `MIN()` function to find out the earliest date in the column `DATE`.
- In the `WHERE` clause, filter the dataset to perform a search only if `Landing__outcome` is `Success (ground pad)`.

Successful Drone Ship Landing with Payload between 4000 and 6000

- Query

```
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (drone ship)'
    AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000)
```

- Result

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

- In the `WHERE` clause, filter the dataset to perform a search if `Landing__outcome` is `Success (drone ship)`.
    - Using the `AND` operator to display a record if additional condition `PAYLOAD_MASS_KG_` is between `4000` and `6000`.

Total Number of Successful and Failure Mission Outcomes

- Query

```
SELECT MISSION_OUTCOME,
       COUNT(*) AS total_number
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME
```

- Result

mission_outcome	total_number
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- Using the `COUNT()` function to calculate the total number of columns.
- Using the `GROUP BY` statement, groups rows that have the same values into summary rows to find the total number in each `Mission_outcome`.
- According to the result, *SpaceX* seems to have **successfully completed nearly 99% of its missions**.

Boosters Carried Maximum Payload

- Query

```
SELECT DISTINCT BOOSTER_VERSION,
       PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
       SELECT MAX(PAYLOAD_MASS__KG_)
       FROM SPACEXTBL)
```

- Using a subquery, first, find the maximum value of the payload by using `MAX()` function, and second, filter the dataset to perform a search if `PAYLOAD_MASS__KG_` is the maximum value of the payload.
- According to the result, version `F9 B5 B10xx.x` boosters could carried the maximum payload.

- Result

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

2015 Launch Records

- Query

```
SELECT LANDING__OUTCOME,
       BOOSTER_VERSION,
       LAUNCH_SITE
FROM SPACEXTBL
WHERE LANDING__OUTCOME
          = 'Failure (drone ship)'
       AND YEAR(DATE) = '2015'
```

- Result

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

- In the `WHERE` clause, filter the dataset to perform a search if `Landing__outcome` is `Failure (drone ship)`.
  - Using the `AND` operator to display a record if additional condition `YEAR` is `2015`.
- In 2015, there were two landing failures on drone ships.

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

- Query

```
SELECT LANDING__OUTCOME,
       COUNT(LANDING__OUTCOME) AS total_number
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING__OUTCOME
ORDER BY total_number DESC
```

- In the `WHERE` clause, filter the dataset to perform a search if the date is between `2010-06-04` and `2017-03-20`.
- Using the `ORDER BY` keyword to sort the records by total number of landing, and using `DESC` keyword to sort the records in descending order.
- According to the results, the number of successes and failures between 2010-06-04 and 2017-03-20 was similar.

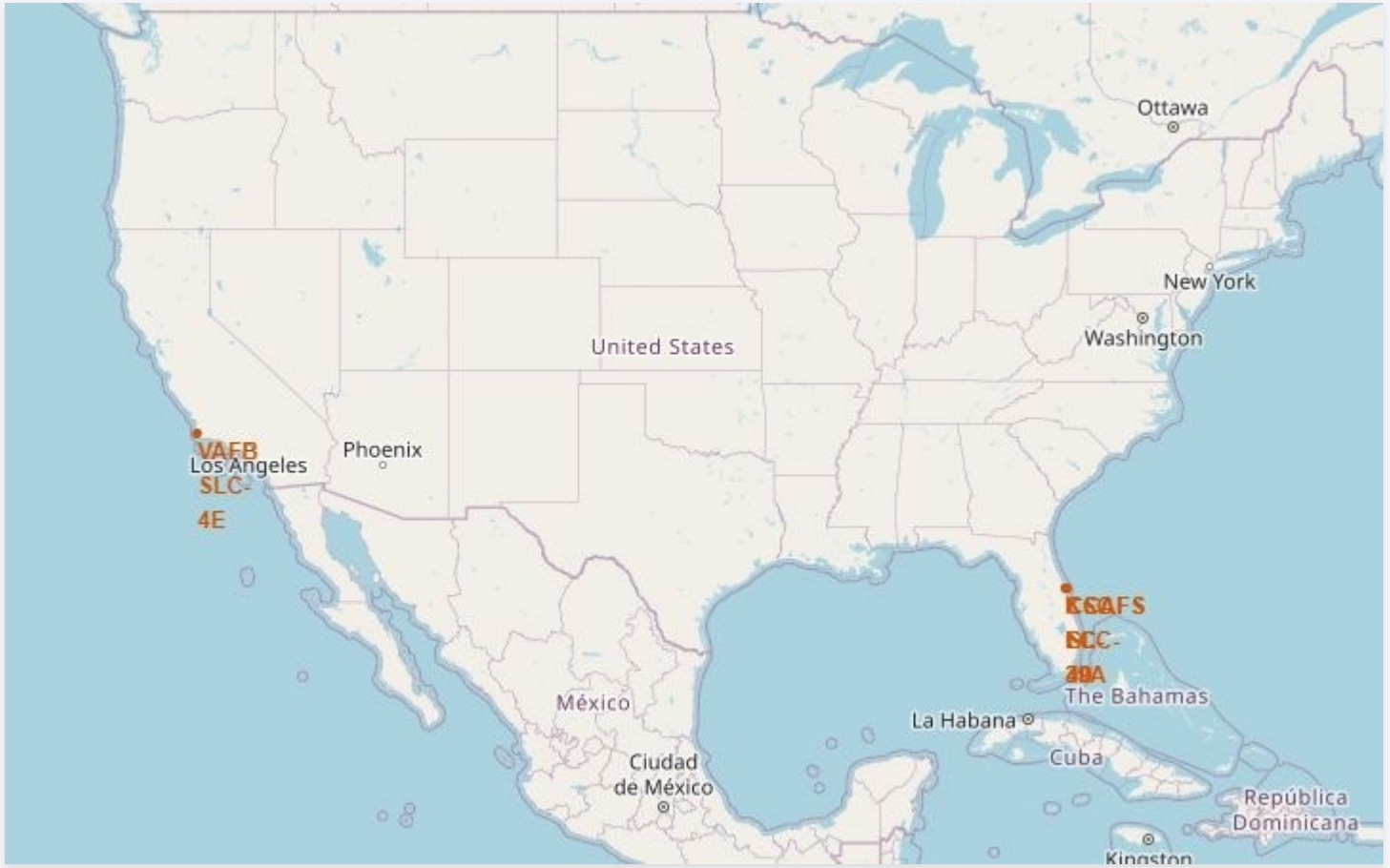
- Result

landing__outcome	total_number
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 Sites' Locations

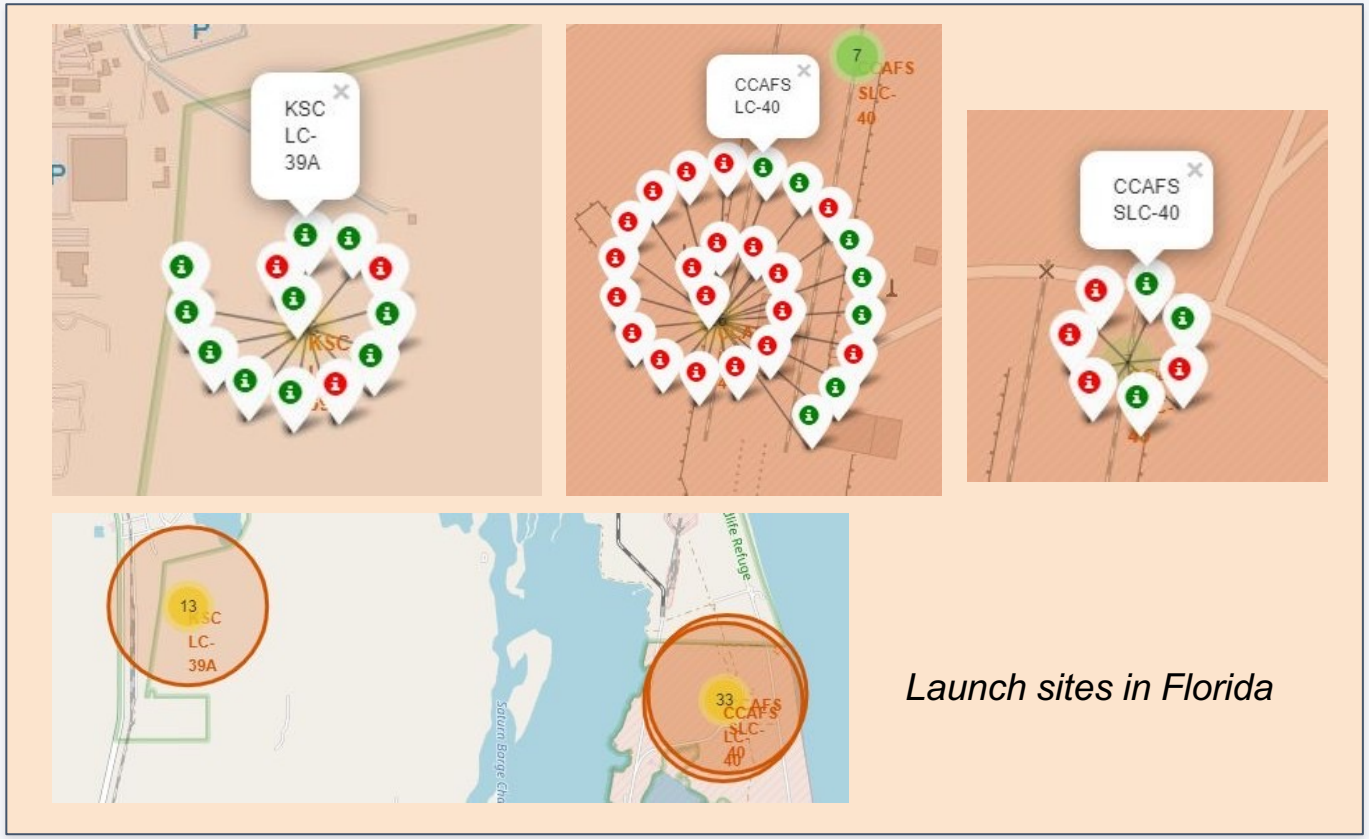


- The left map shows all *SpaceX* launch sites, and the right map also shows that all launch sites are in the United States.
- As can be seen on the map, all launch sites are near the coast.

Color-labeled Launch Outcomes

Launch site in California

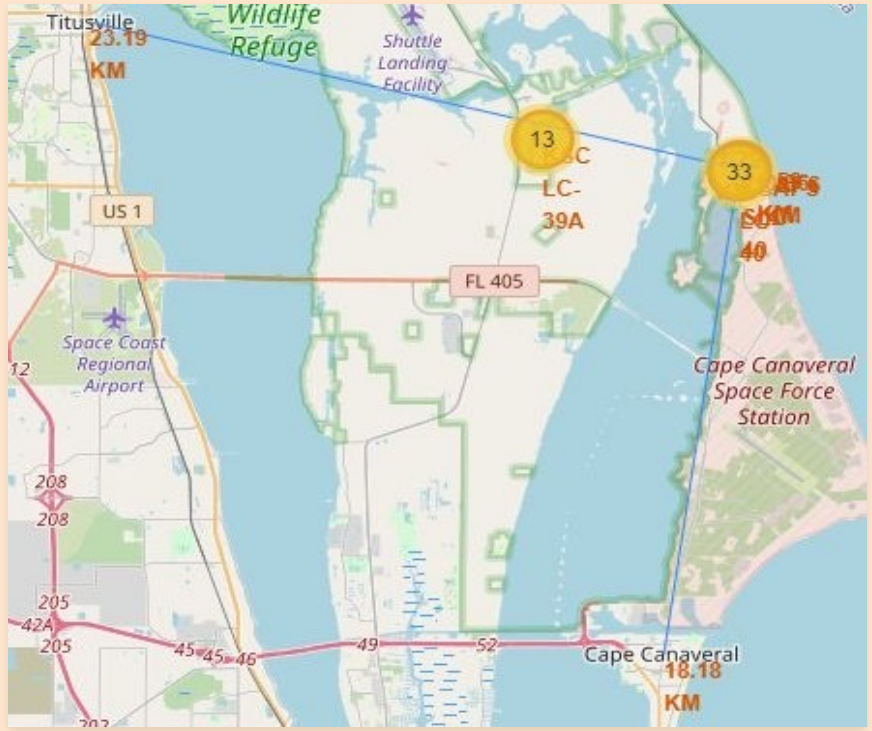
- By clicking on the marker clusters,
  successful landing (green)
  or failed landing (red) are
  displayed.



Launch sites in Florida

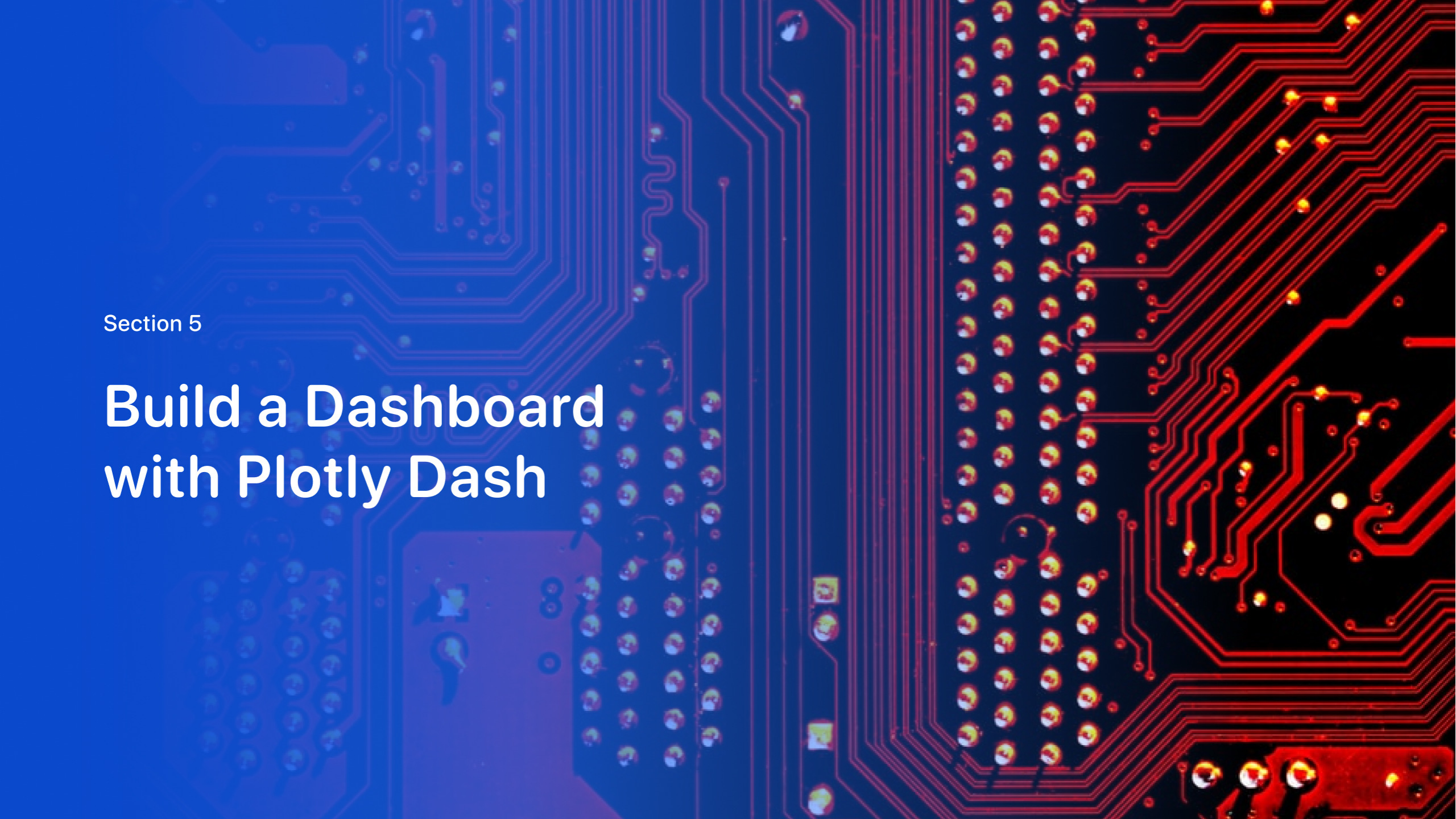
Proximities of Launch Sites

Are launch sites in close proximity to railways? ***Yes***
Are launch sites in close proximity to highways? ***Yes***
Are launch sites in close proximity to coastline? ***Yes***



Do launch sites keep certain distance away from cities? ***Yes***

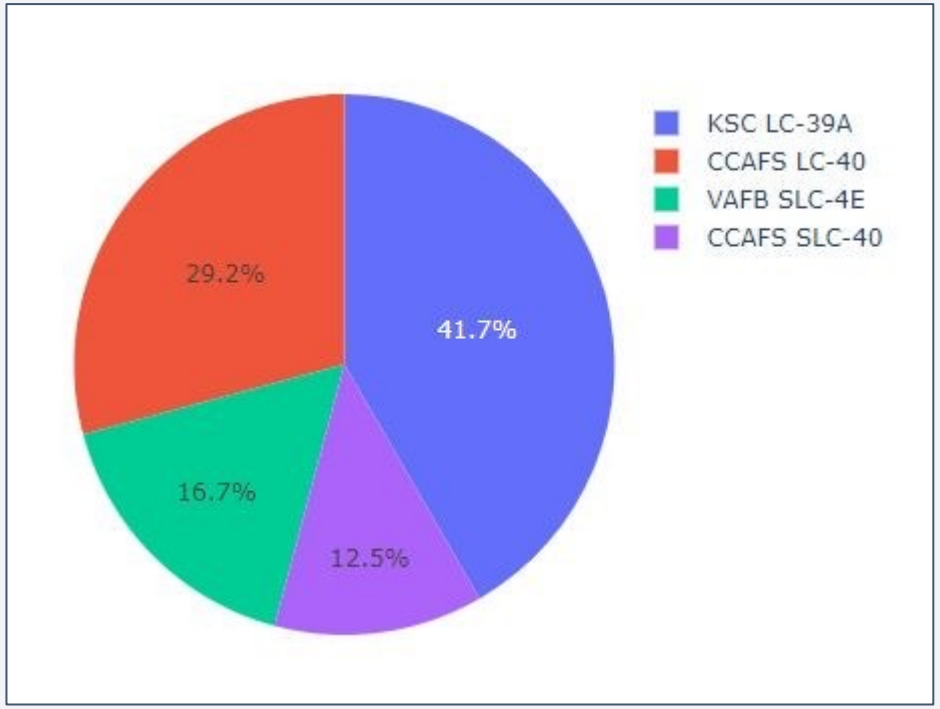
- It can be found that the launch site is **close to railways and highways** for transportation of equipment or personnel, and is also **close to coastline** and relatively **far from the cities** so that launch failure does not pose a threat.



Section 5

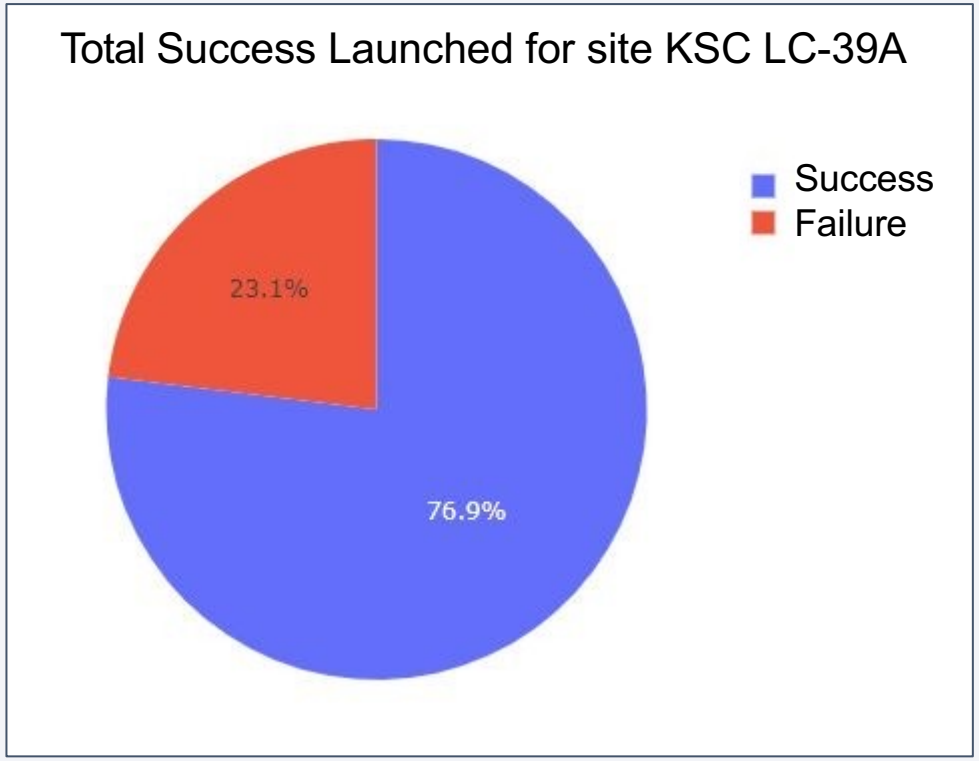
Build a Dashboard with Plotly Dash

Total Success Launches By all sites



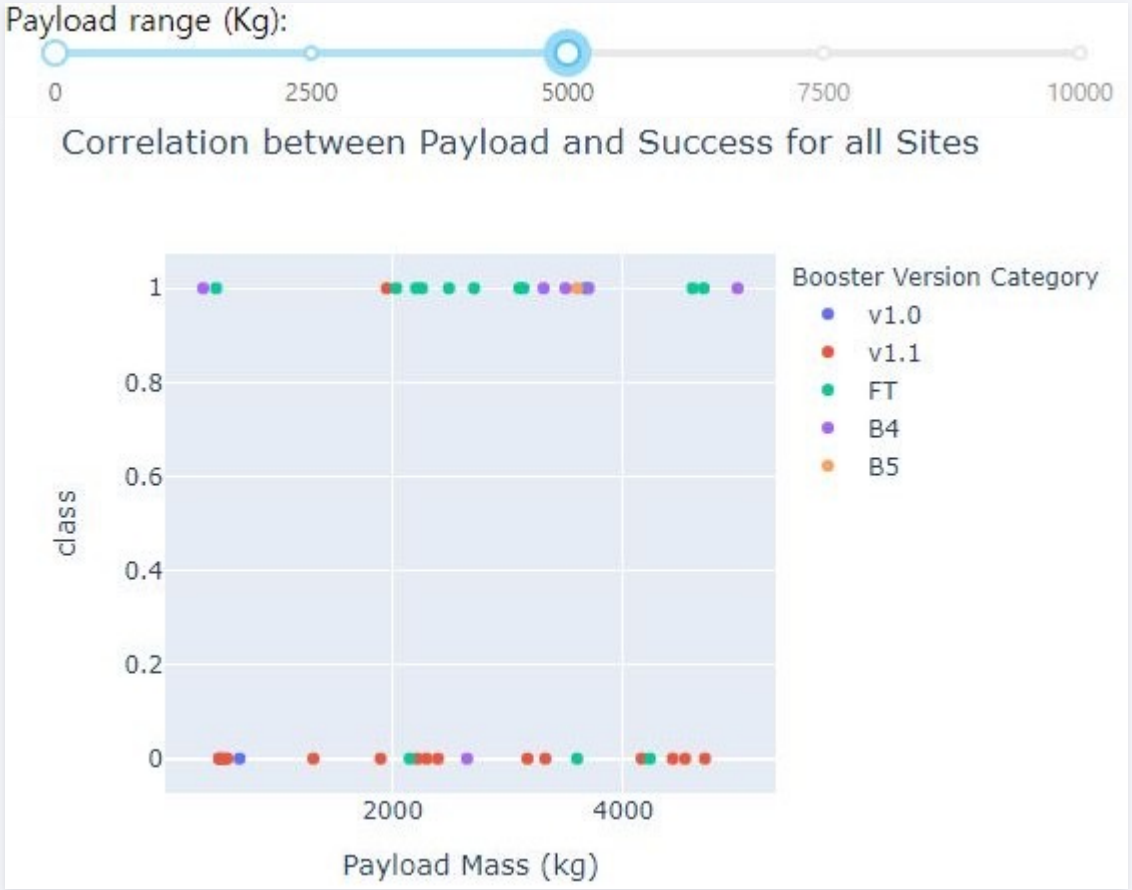
- KSLC-39A records the most launch success among all sites.
- The VAFB SLC-4E has the fewest launch success, possibly because
  - the data sample is small, or
  - because it is the only site located in California, so the launch difficulty on the west coast may be higher than on the east coast.

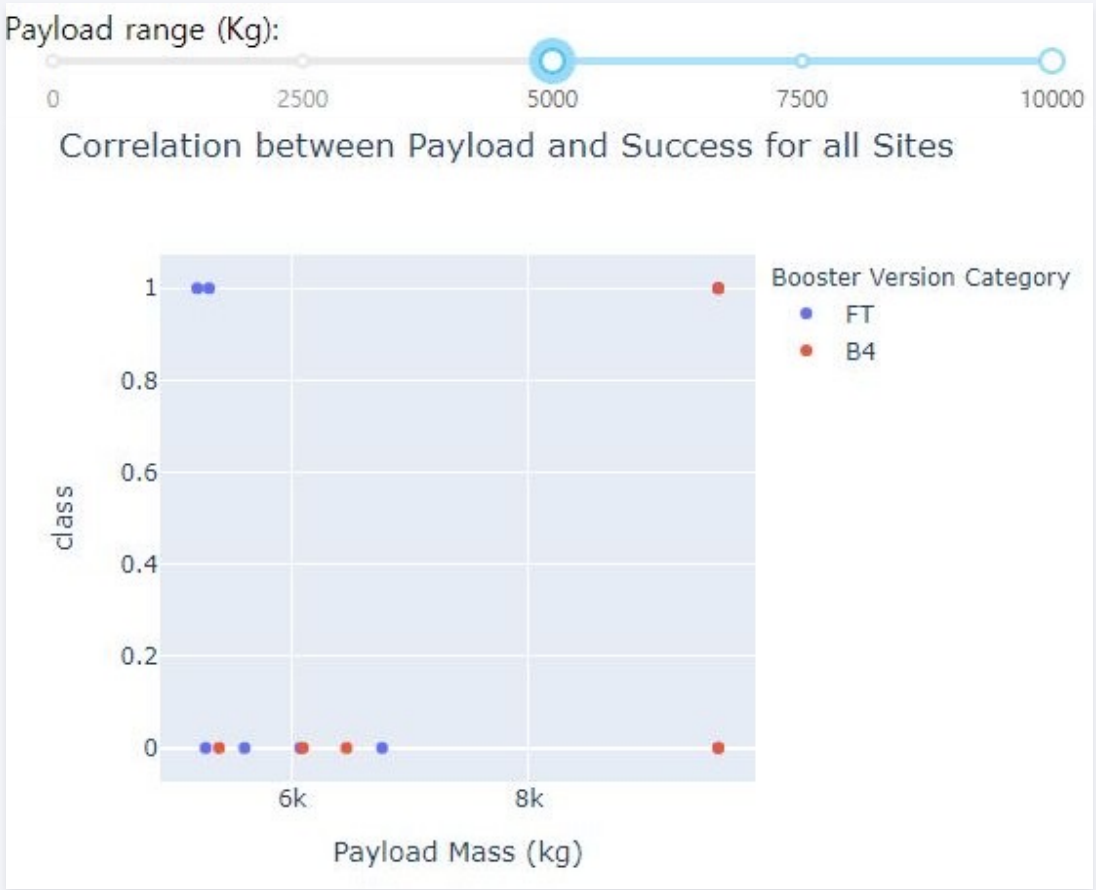
Launch Site with Highest Launch Success Ratio



- KSLC-39A has the highest success rate with 10 landing successes *(76.9%)* and 3 landing failures *(23.1%)*.

Payload vs. Launch Outcome Scatter Plot for all sites



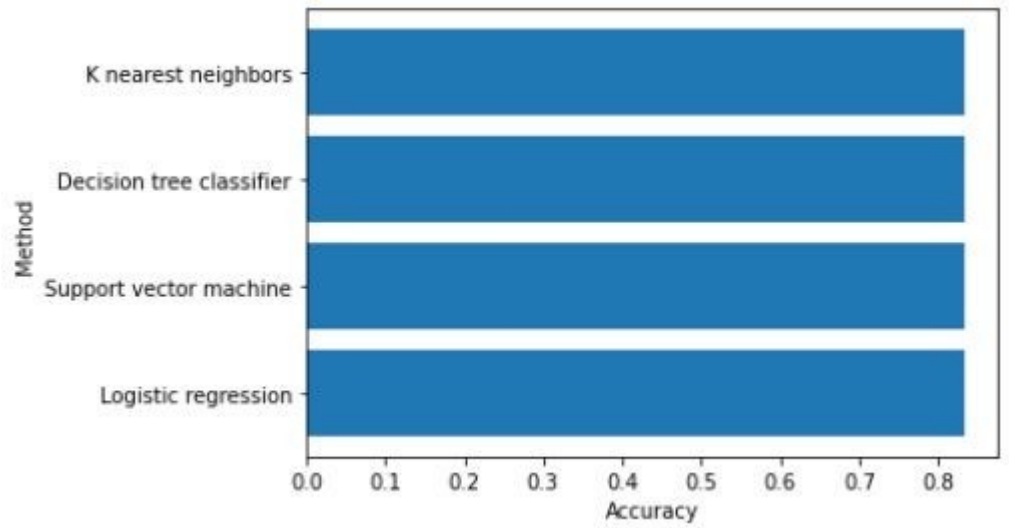


- These figures show that **the launch success rate** *(class 1)* **for low weighted payloads*(0-5000 kg)* is higher** than that of heavy weighted payloads*(5000-10000 kg)*.

Section 6

Predictive Analysis (Classification)

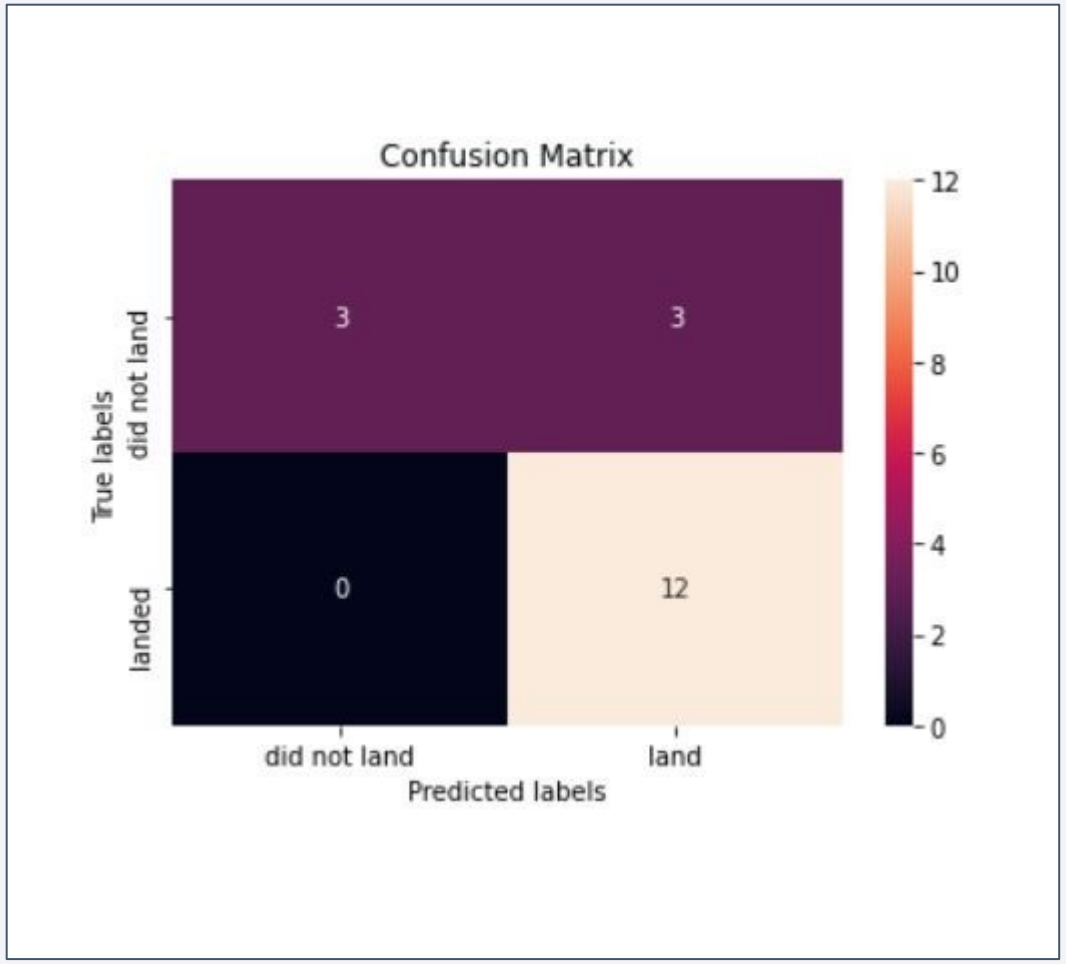
Classification Accuracy



	Method	Accuracy
0	Logistic regression	0.833333
1	Support vector machine	0.833333
2	Decision tree classifier	0.833333
3	K nearest neighbors	0.833333

- In the test set, **the accuracy of all models** was virtually the **same** at **83.33%**.
- It should be noted that the test size was small at 18.
- Therefore, more data is needed to determine the optimal model.

Confusion Matrix



- The confusion matrix is the same for all models because all models performed the same for the test set.
- The models predicted 12 successful landings when the true label was successful and 3 failed landings when the true label was failure. But there were also 3 predictions that said successful landings when the true label was failure *(false positive)*.
- Overall, **these models predict successful landings**.

Conclusions

- As the number of flights increased, the success rate increased, and recently it has exceeded 80%.
- Orbital types SSO, HEO, GEO, and ES-L1 have the highest success rate (100%).
- The launch site is close to railways, highways, and coastline, but far from cities.
- KSLC-39A has the highest number of launch successes and the highest success rate among all sites.
- The launch success rate of low weighted payloads is higher than that of heavy weighted payloads.
- In this dataset, all models have the same accuracy (83.33%), but it seems that more data is needed to determine the optimal model due to the small data size.

Thank you!