IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Alan Chang

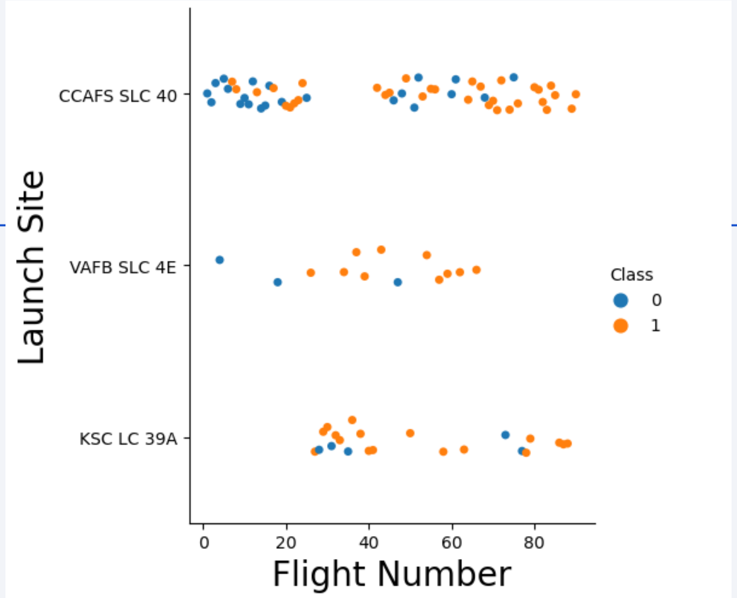
2/16

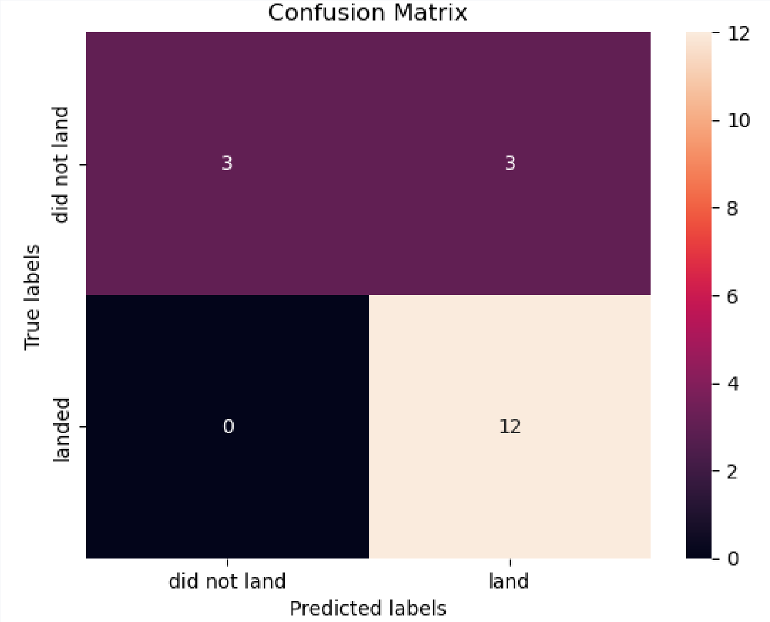
OUTLINE

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

Executive Summary

- Summary of methodologies
  - Collecting the Data
  - Data Wrangling
  - Analysis Using SQL, Pandas, Matplotlib
  - Visual Analytics and Dashboard
  - Predictive Analysis
- Summary of all results
  - Exploratory Data Analysis (EDA)
  - Marking Locations of Sites
  - Confusion Matrix to determine the best results





Introduction

- Background History
  - SpaceX launched a Falcon 9 rocket at a cost of $62million comparing to the usual price of $165million USD
  - SpaceY wants to do a comparison in order to see if the cost is much better than SpaceX
- Issue
  - In order to find out if SpaceY can create a better economical value through different multiple predictions

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - Requesting from SpaceX API
- Perform data wrangling
    - Using methods such as value_counts and .fillna to provide the provide the values and fill in values that did not have any
    - Labeling each launch for successful and failure runs
- Perform exploratory data analysis (EDA) using visualization and SQL
    - Used Pandas and Matlibplot to create visualizations in order to have a better understanding of the data
- Perform interactive visual analytics using Folium and Plotly Dash
    - Create an interactive dashboard using the Plotly Dash
    - Create a geographical image using Folium
- Perform predictive analysis using classification models
    - Processed the data by splitting the data into training and testing data
    - Plot the points using a confusion matrix for the data

Data Collection – Find within SpaceX API

- Data Collection
  - Obtain the SpaceX API

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

  - Define the Variables in order to clean out and call custom functions to get data from the lists

```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

```
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}
```

  - Provide Filtering to only provide all Falcon 9 Launches

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


Github Link

Data Wrangling – Fill in records

- Request from Static URL to respond to object

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

- Create BeautifulSoup Object to find all the tables in the HTML Page

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(http.text, "html.parser")
```

- Use all columns names as keys from the table and create a function to fill "launch_dict"

```
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each value to be an empty list
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'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

Github Link

EDA with SQL

- The SQL Queries were used in order to gather the information necessary for the data
  - Display Launch Site Names
  - Display sites beginning with 'CCA'
  - Display the total payload mass carried by boosters launched by NASA(CRS)
  - Display the average payload mass carried by booster version F9 v1.1
  - List the date when the first successful landing outcome on a ground pad was achieved
  - List the names of the boosters which had success on a drone ship and a payload mass between 4000 and 6000 kg
  - List the total number of successful and failed mission outcomes
  - List the names of the booster versions which have carried the maximum payload mass
  - List the failed landing outcomes on drone ships, their booster versions, and launch site names for 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 Link

EDA with Data Visualization

- Three Charts were used in order to visualize
  - **Scatter Chart**
    - Flight Number and Launch Site
    - Payload and Lanch Site
    - Orbit Type
  - **Bar Chart**
    - Success Rate of Orbit
  - **Line Chart**
    - Success Rate and Year

Build an Interactive Map with Folium

1.Mark all launch Sites

- Start the map using Map Object
- Added folium circles and markers for each launch site on map

2.Mark the Success and Failtures of each launches on site

- Launch all within the same coordinates
- Before clustering them, assign a marker colour of successful (class = 1) as green, and failed (class = 0) as red.
- Put launches into clusters using folium.Marker to the MarkCluster() Object

3.Create a calculation of the distance between its launch site and its proximities

- Use points on the map by determining the Lat and Long Values
- Marking Points of Lat and Long Values with folium.marker
- Displayed the points between the markers using a polyline

Github Link

Build a Dashboard with Plotly Dash

- Pie Chart showing the total success of launches per site
    - Allowing to see which sites were successful
    - Filtered to show success and failure ratio of each site
- Scatter Graph to show the correlation between the success and payload
    - Filtering using RangeSlider() by rnages of payload masses

Github Link

Predictive Analysis (Classification)

- Development Model
  - Preparing the dataset
    - Load Data
    - Create transformations
    - Split Data into training and test sets, train_test_split()
  - Algorithm
    - Used GridsearchCV object and dictionary
    - Fit the object to the parameter

- Evaluation
  - GridSearchCV
    - Finding the best params
    - Finding the best accuracy
  - Plot a Confusion Matrix for better show
  - Used to find the best accuracy score
  - Determining if the performing models are good

Results

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

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

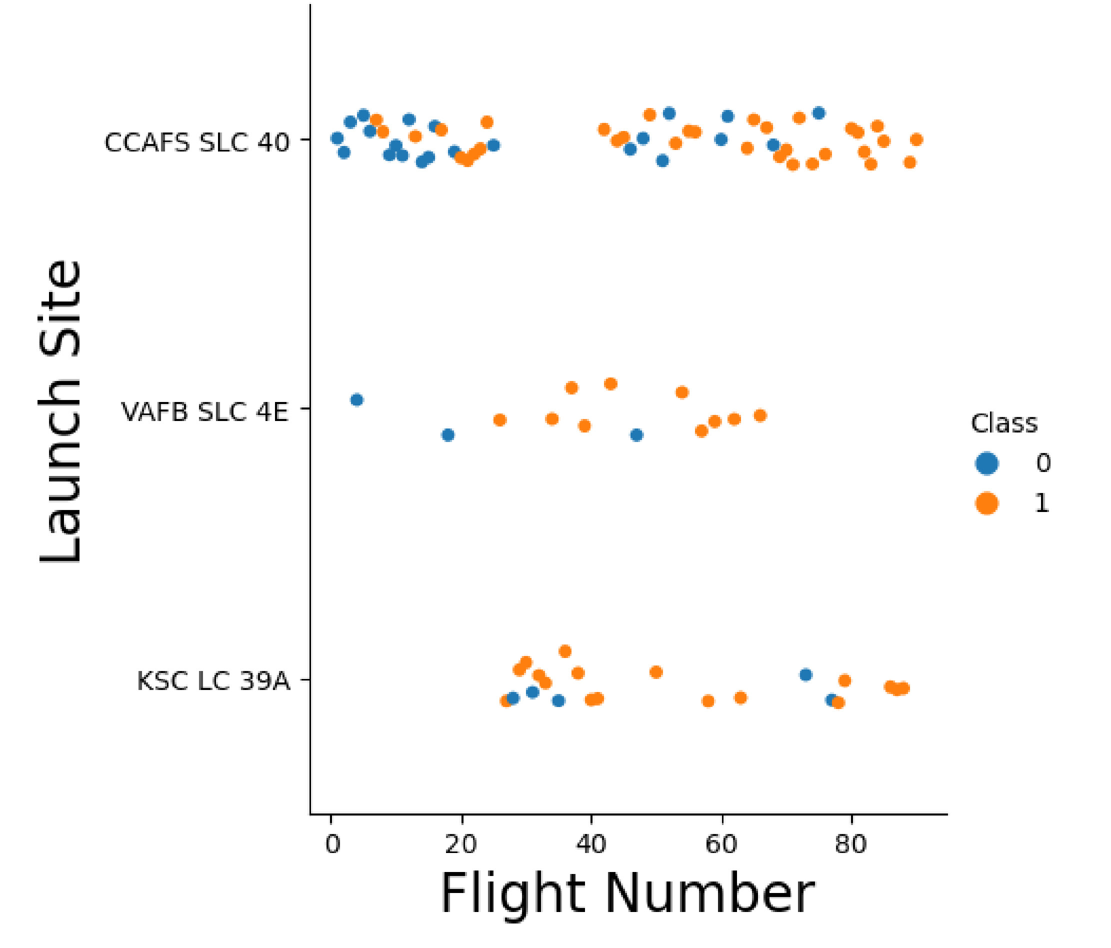
The Scatter Plot of Launch site vs. Flight Number

- Number of flight increases, the rate increases

Flights < 30 from **CCAFS SLC 40** provided a lot of unsuccessful results while >30 had a more better results

Flights for **VAFB SLC 4E** had little of flights but still showed the result of more flight increases, the better success rate

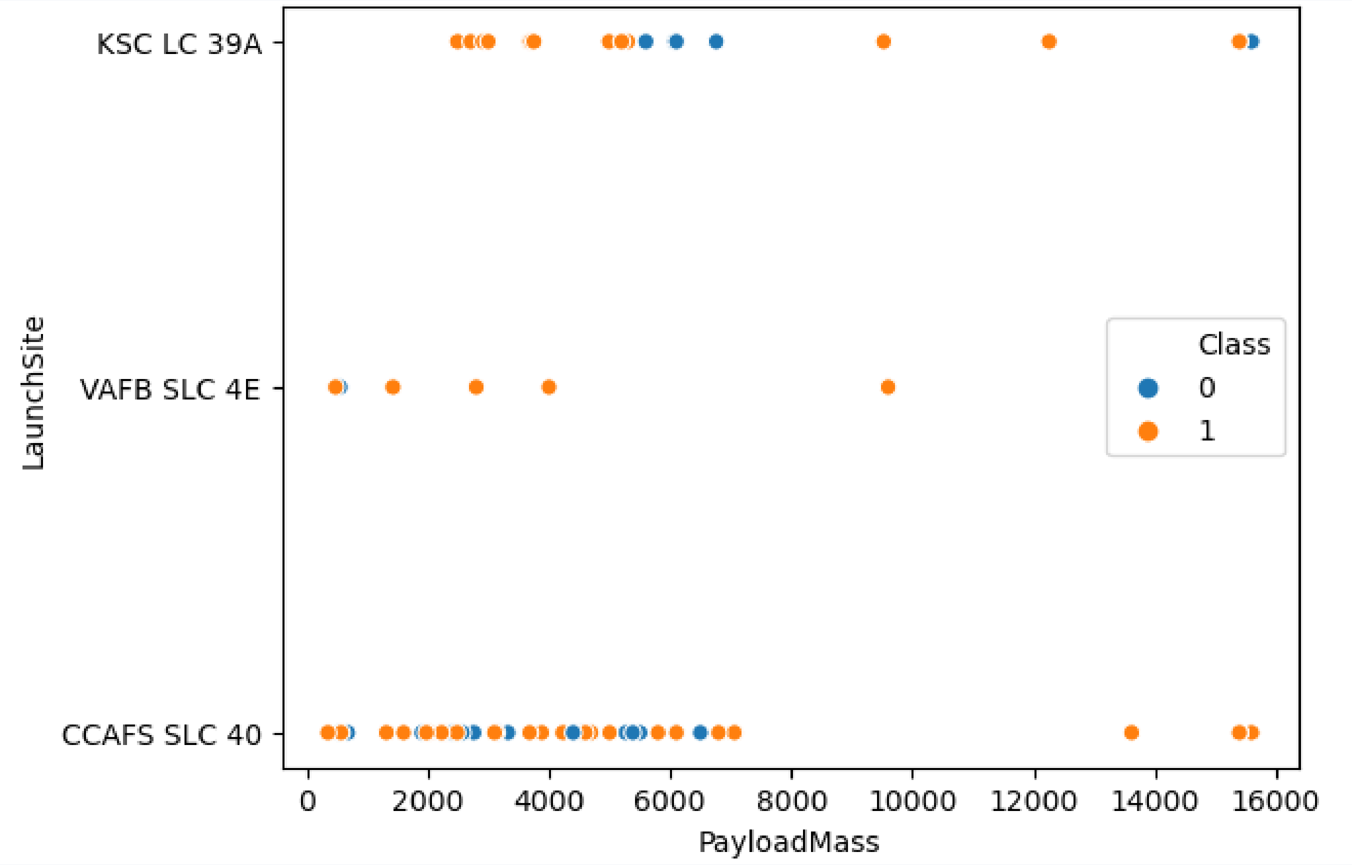
KSC LC 39A follows the same with many successful results in the process



Payload vs. Launch Site

The Payload vs Launch Site scatter shows no real concrete patterns other than having a lot of >6000

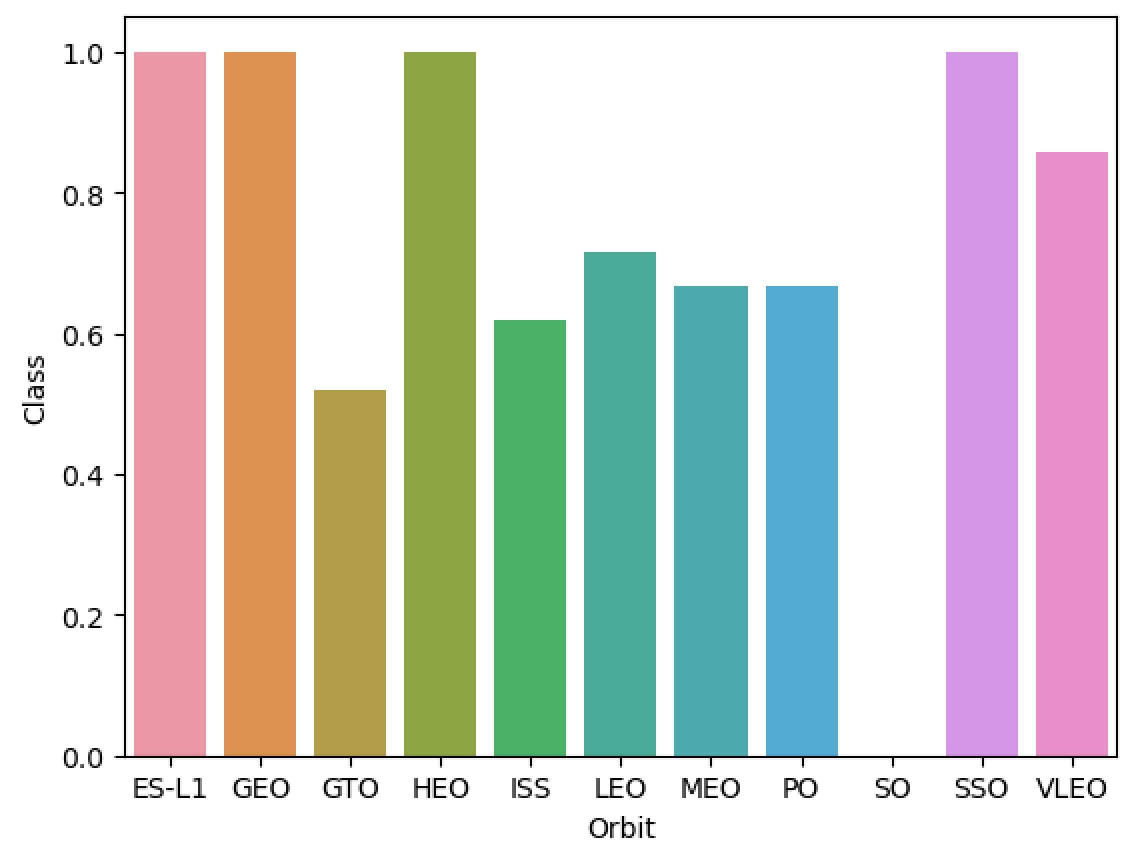
- KSC LC 39A does not have a pattern to show but has a lot of successful results with >6000 with only few outliers
- VAFB SLC 4E had majority of successful results being >6000 as well with 1 outlier
- CCAFS SLC 40 is the only one with a lot of attempts as well as the amount of failures and success >6000



Success Rate vs. Orbit Type

Shows the highest success rates (0% - 100%) from all Orbit types.

- ES-L1, GEO, HEO, SSO have a 100% while SO is the only one with a 0%

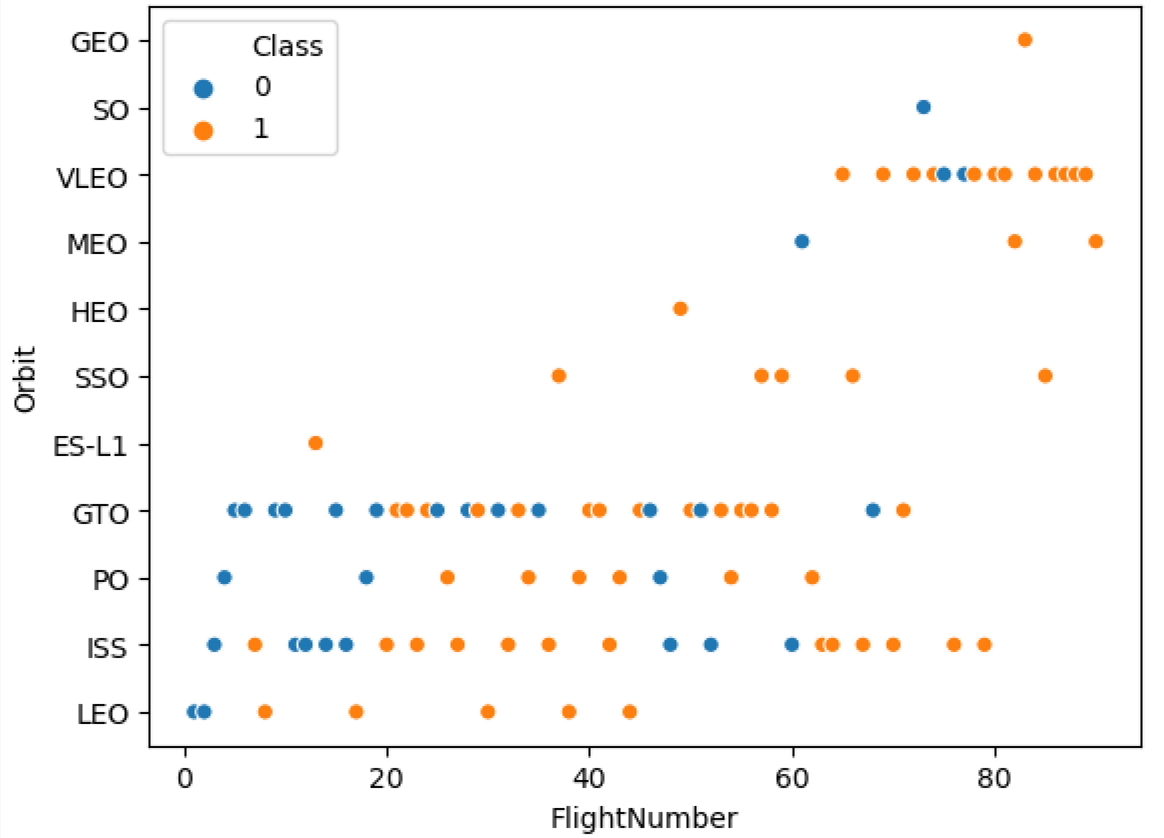


Flight Number vs. Orbit Type

Here shows the number of flights tested from all Orbit Types.

- The graph shows that SO, GEO, ES-L1 and HEO had only one testing which shows the 0 or 100% chance with one attempt
- The other Orbit Types has more testing done but the one that stands out SSO with few attempts but all successful rates.

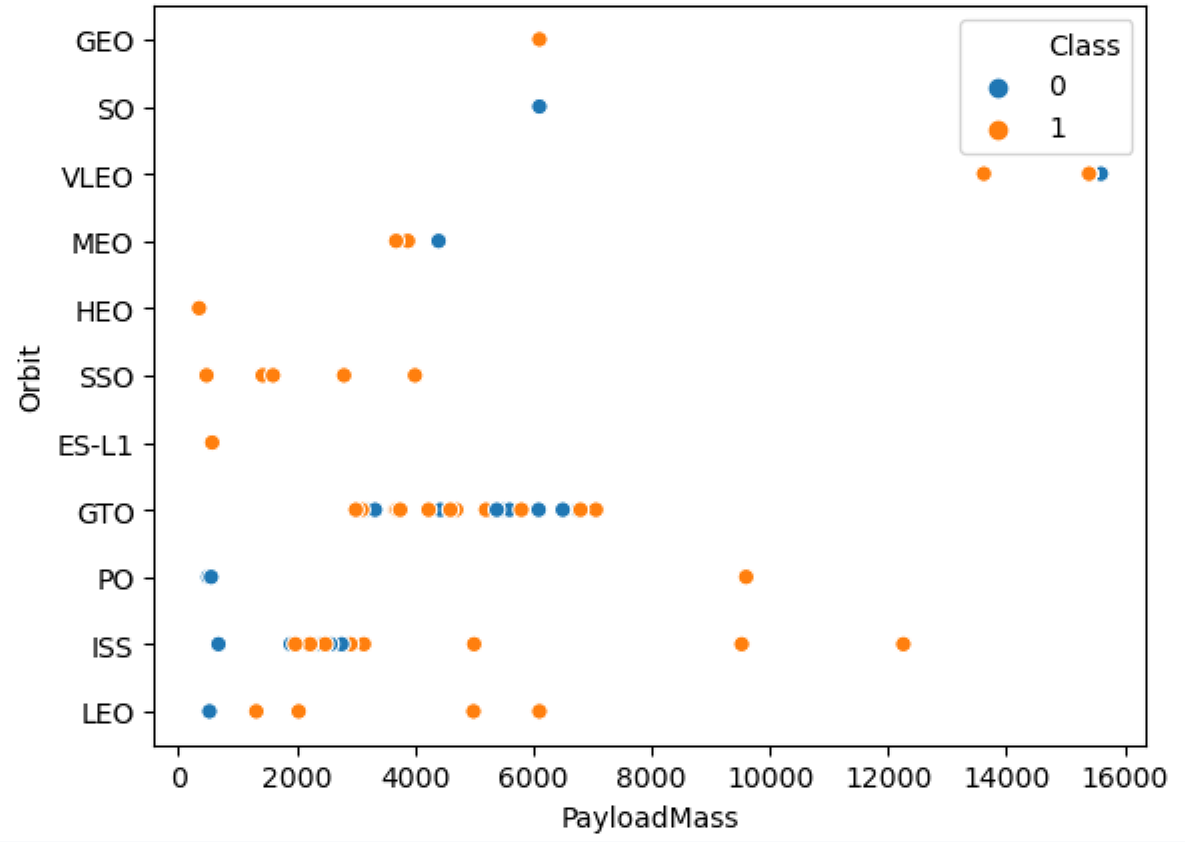
Overall, more failures can occur at a less FlightNumber.



Payload vs. Orbit Type

Here shows the graph of Payload vs Orbit Type.

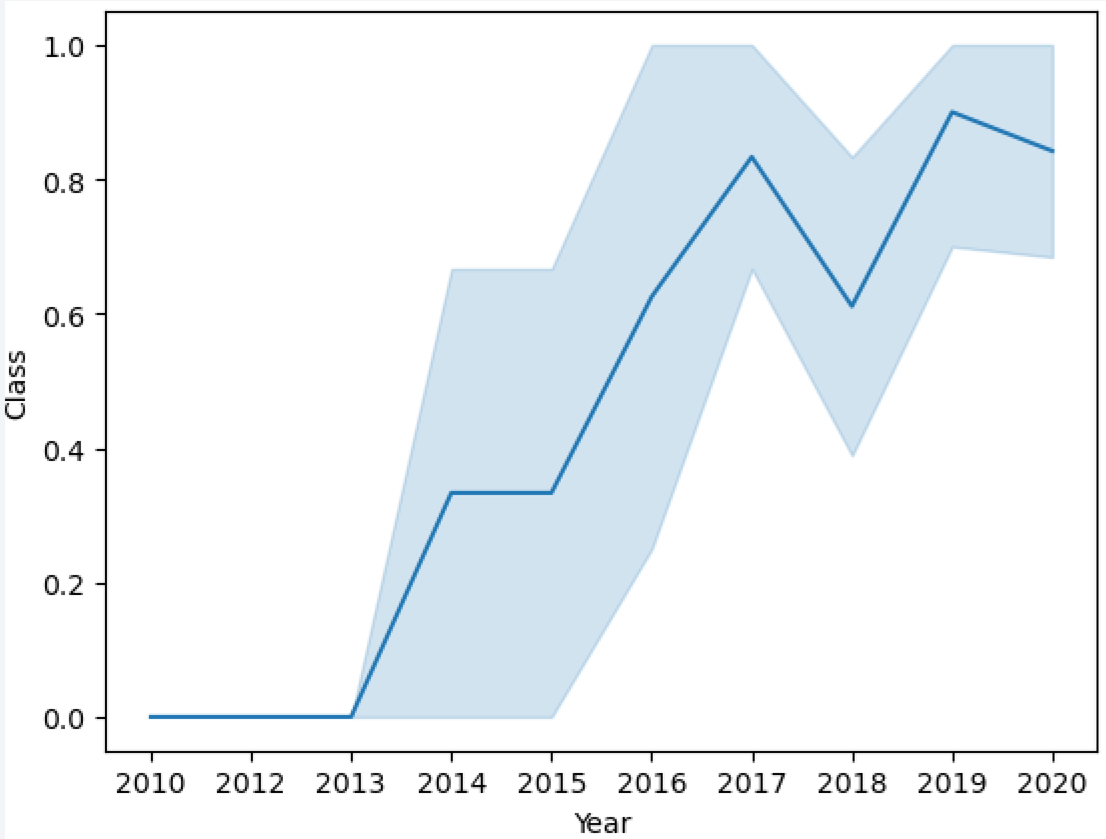
- Something that sticks out is GEO and SO having the same Payload Mass but with opposite results from reach other.
- SSO provided with 100% with >6000 followed by HEO and ES-L1 having the same idea
- VLEO the only Orbit Type with very high Mass but not at the following 100%



Launch Success Yearly Trend

Here shows the trend of yearly success

- Through the year 2013 to 2017 has given the idea of improvement
- 2017 to 2018 is where the decline started
- 2018 – 2020 gradually goes back up but has a very small decline at the end but nothing too drastic.



All Launch Site Names

This finds all the unique Launch sites from their database

```
%sql select DISTINCT LAUNCH_SITE from SPACEXTBL;
```

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

Launch Site Names Begin with 'CCA'

Statement here shows CCAFS LC-40 names to a limit of 5 in the showcase down below

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass from NASA (CRS)

Shows the total mass specifically finding from NASA

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEXTBL \
    WHERE CUSTOMER = 'NASA (CRS)';
```

TOTAL_PAYLOAD_MASS
45596

Average Payload Mass by F9 v1.1

Statement is using Average Function taking where the booster version is F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVERAGE_PAYLOAD_MASS FROM SPACEXTBL \
    WHERE BOOSTER_VERSION = 'F9 v1.1';
```

AVERAGE_PAYLOAD_MASS
2928.4

First Successful Ground Landing Date

The statement uses min to find the first success date

```
%sql select min(date) as Date from SPACEXTBL where mission_outcome like 'Success'
```

Date
01-03-2013

Successful Drone Ship Landing with Payload between 4000 and 6000

The Statement uses a condition to find success of ‘Drone Ships’ and payload between 4000 to 6000

```
%sql select booster_version from SPACEXTBL where ("Landing _Outcome" like 'Success (drone ship)') AND (payload_mass__kg_ BETWEEN 4000 AND 6000)
```

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

Total Number of Successful and Failure Mission Outcomes

Statement uses Count to find the total success and fail mission outcomes

```
%sql SELECT mission_outcome, count(*) as Count FROM SPACEXTBL GROUP by mission_outcome ORDER BY mission_outcome
```

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

Boosters Carried Maximum Payload

Statement uses a condition to locate the Maximum Payload that uniquely separates each Booster Version

```
%%sql SELECT DISTINCT(BOOSTER_VERSION) FROM SPACEXTBL
    WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

- Used the condition Landing outcome to match Failure (Drone Ship)

```
%%sql select substr(DATE,4,2) as Month, "Landing _Outcome", booster_version, launch_site from SPACEXTBL
where substr(Date,7,4) AND "Landing _Outcome" like 'Failure (drone ship)'
```

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
01	Failure (drone ship)	F9 v1.1 B1017	VAFB SLC-4E
03	Failure (drone ship)	F9 FT B1020	CCAFS LC-40
06	Failure (drone ship)	F9 FT B1024	CCAFS LC-40

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

- Conditions of creating the date to match between 4-6-2010 and 20-3-2017) while

```
%%sql Select "Landing _Outcome", count("Landing _Outcome") as Total_Number from SPACEXTBL
where Date between '04-06-2010' and '20-03-2017'
group by "Landing _Outcome"
order by Total_Number desc;
```

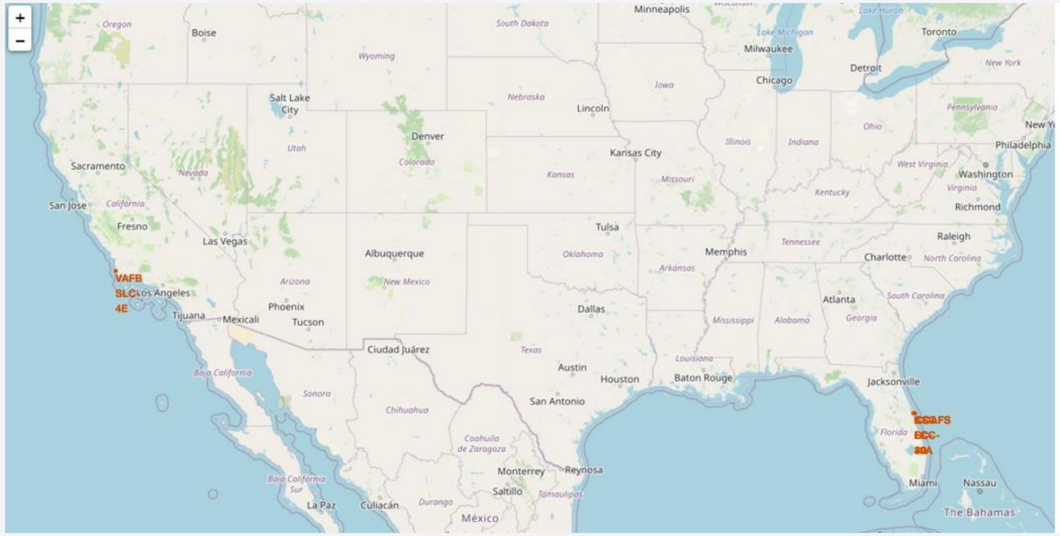
Landing _Outcome	Total_Number
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

%%sql Select "Landing _Outcome", count("Landing _Outcome")
as Total_Number from SPACEXTBL
where Date between '04-06-2010' and '20-03-2017'
group by "Landing _Outcome"
order by Total_Number desc;

Section 3

Launch Sites Proximities Analysis

Marking all Sites on the Map

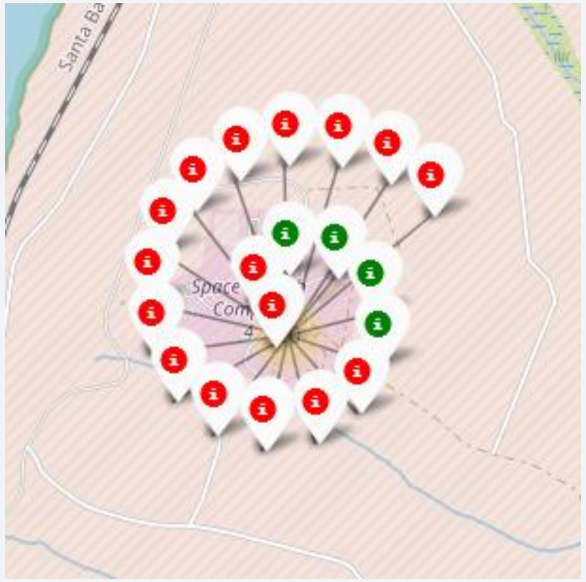


California

Florida

The results of the map locations are on the coast of United States. Majority of the marks are located in California and Florida in this case

Marking the Success and Failures of the Sites

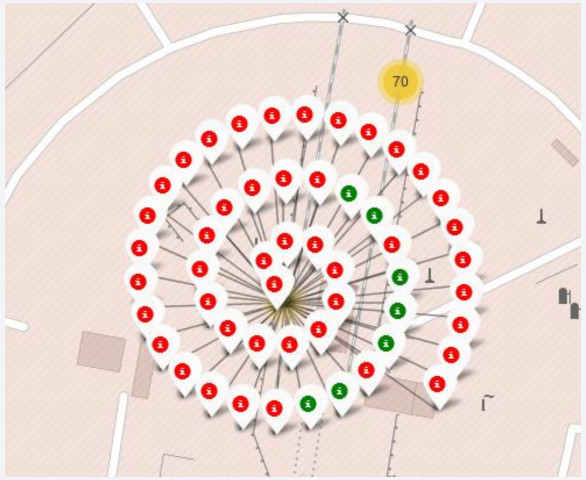


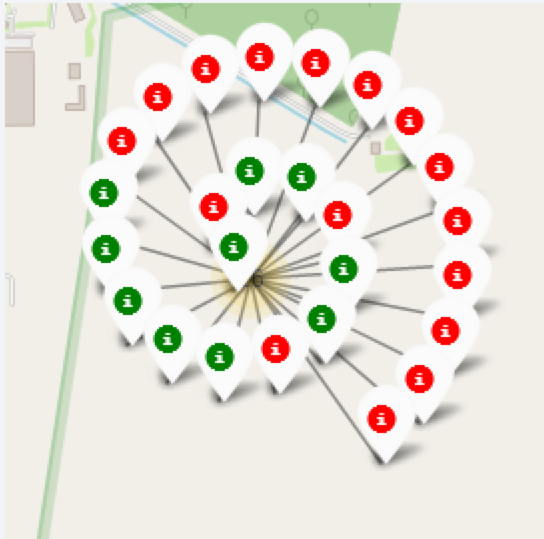
Icons: Success / Failures

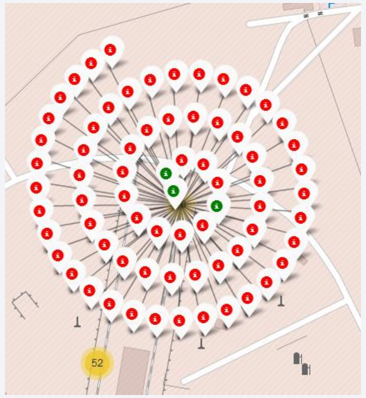
On the left holds the markings for California Testing

On the right holds all the Florida Testing

To conclude, a lot of testing was done in Florida with a lot of filatures surrounding their testing.

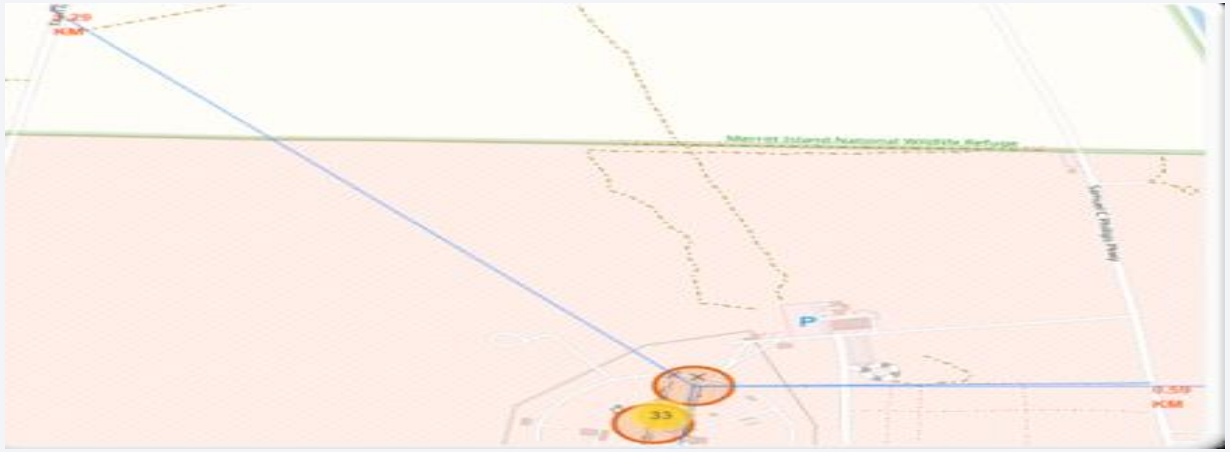


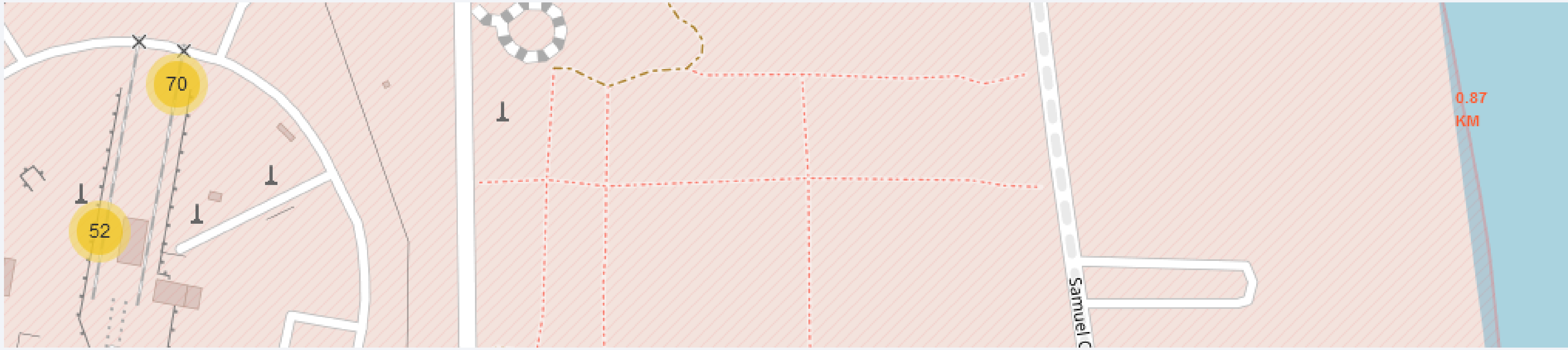




Calculating the distance between launch sites

Following the Launch Sites located in Florida, we can find the distance of the placement for reach launch

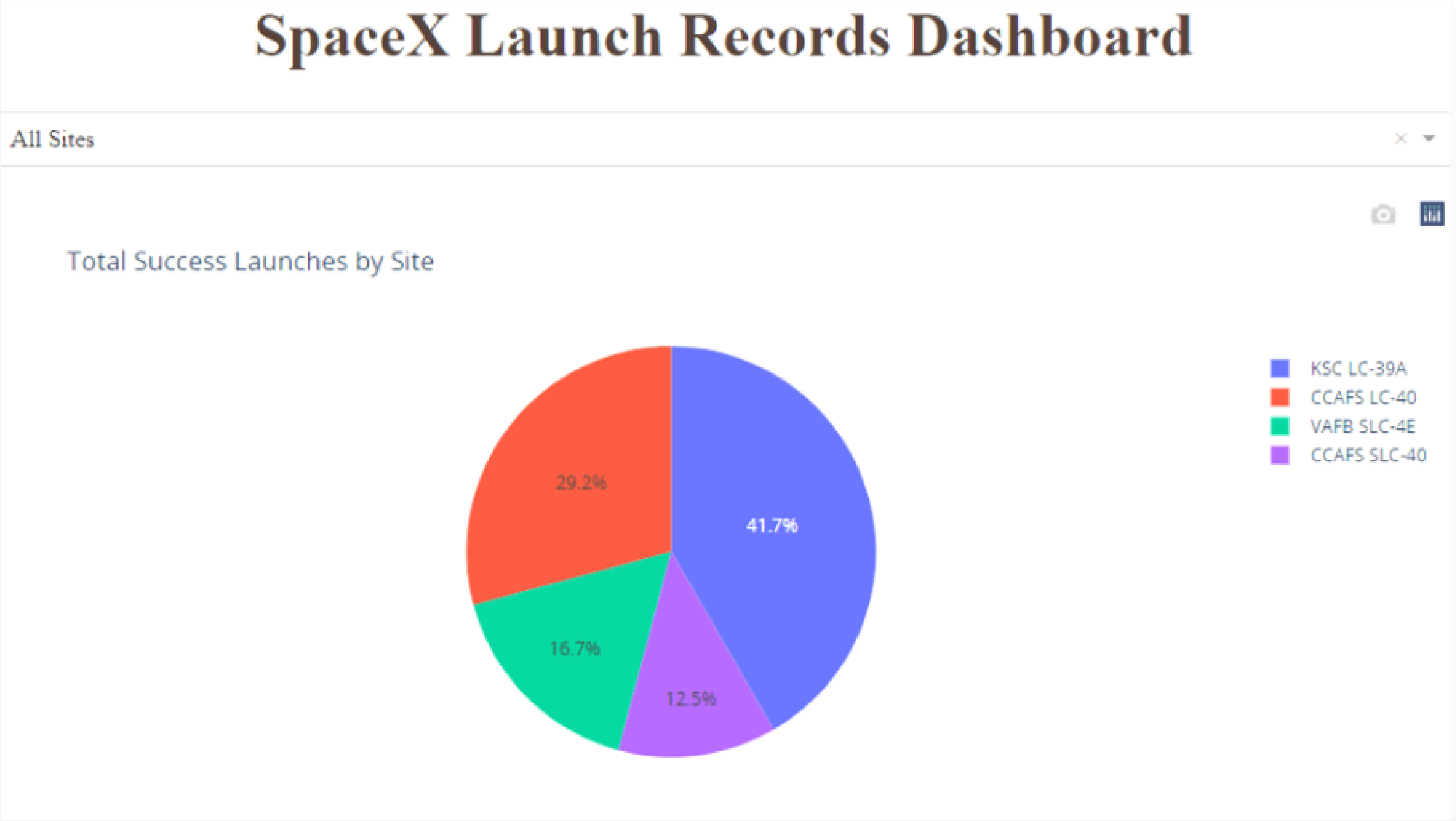




Section 4

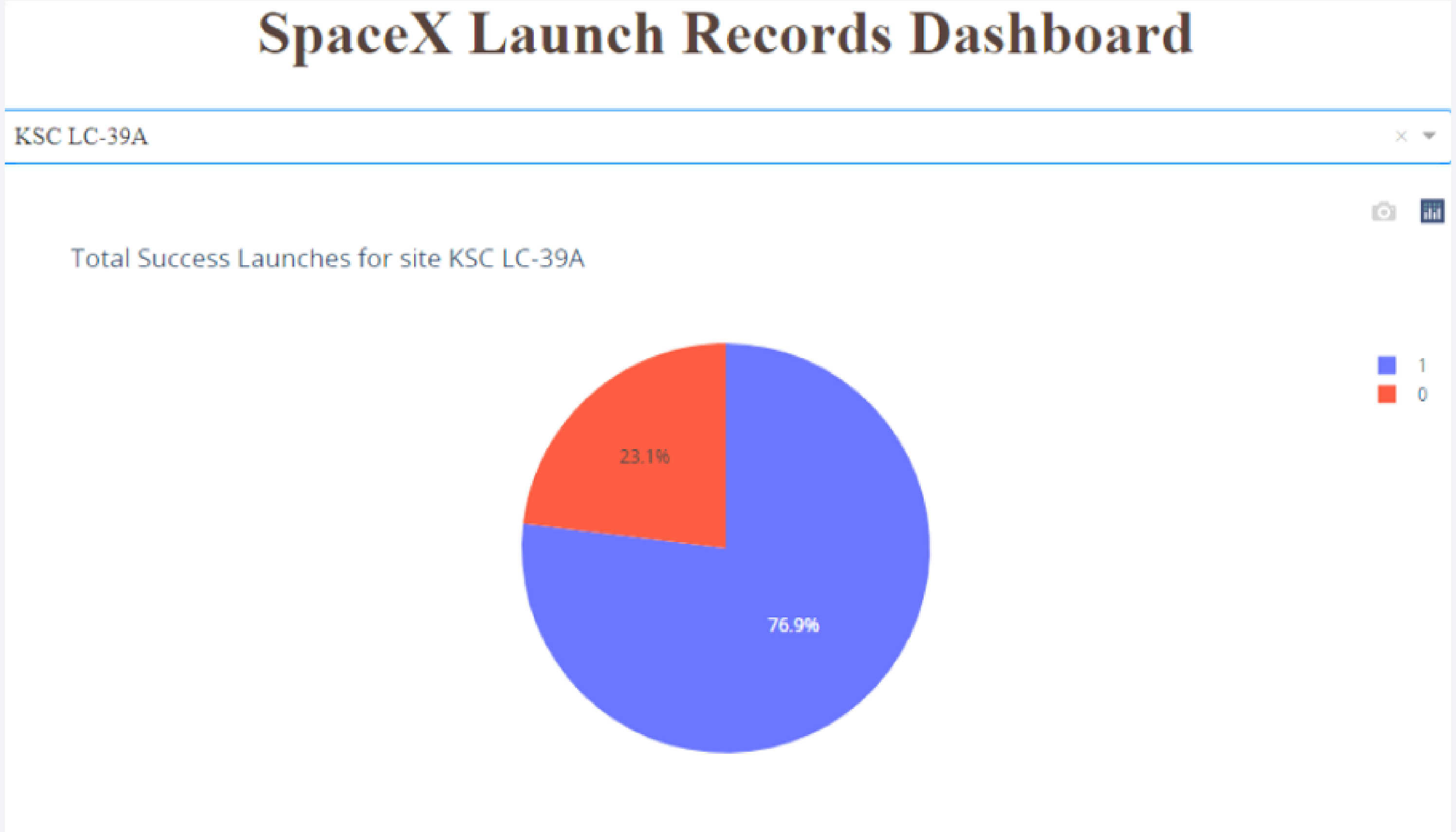
Build a Dashboard with Plotly Dash

Success Launches from Sites



Overall the results show are on the left, it shows that KSC LC-39A (41.7%) happens to have the highest total success out of the 4 sites

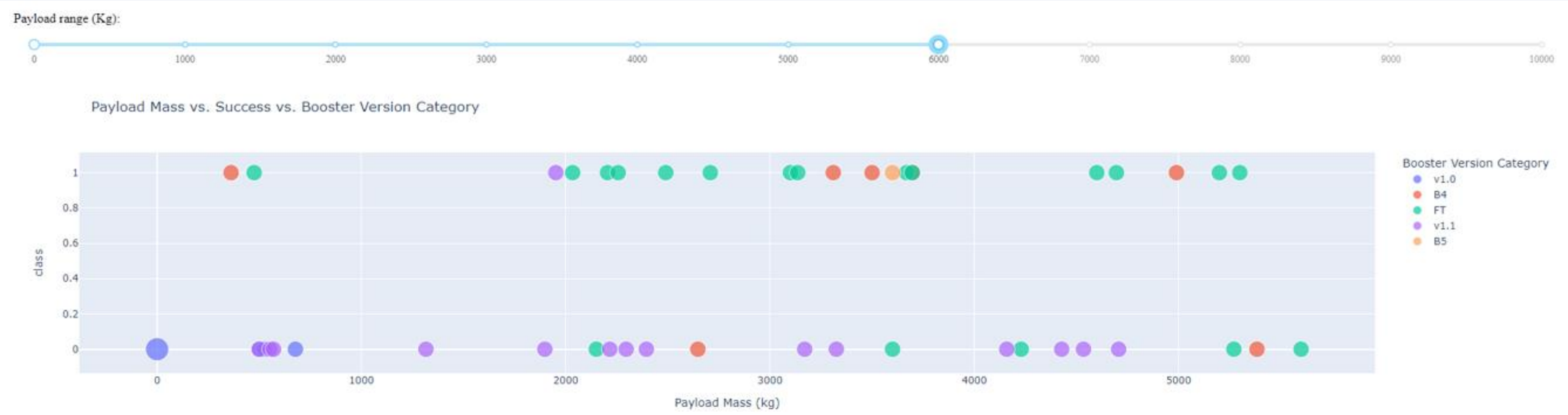
KSC-LC39A Success Ratio



KSC-LC39A success ratio shows where 1 is the Success and 0 is the Fail.

Showing that success out of the total in KSC-LC39A site is 76.9%

Payload Mass Vs Success vs Booster Version Category



The Dashboard shown values the range of 0 to 6000, all put in a Scatter plot
There are only two defining classes, Class 1 (Success) and Class 0 (Failure)

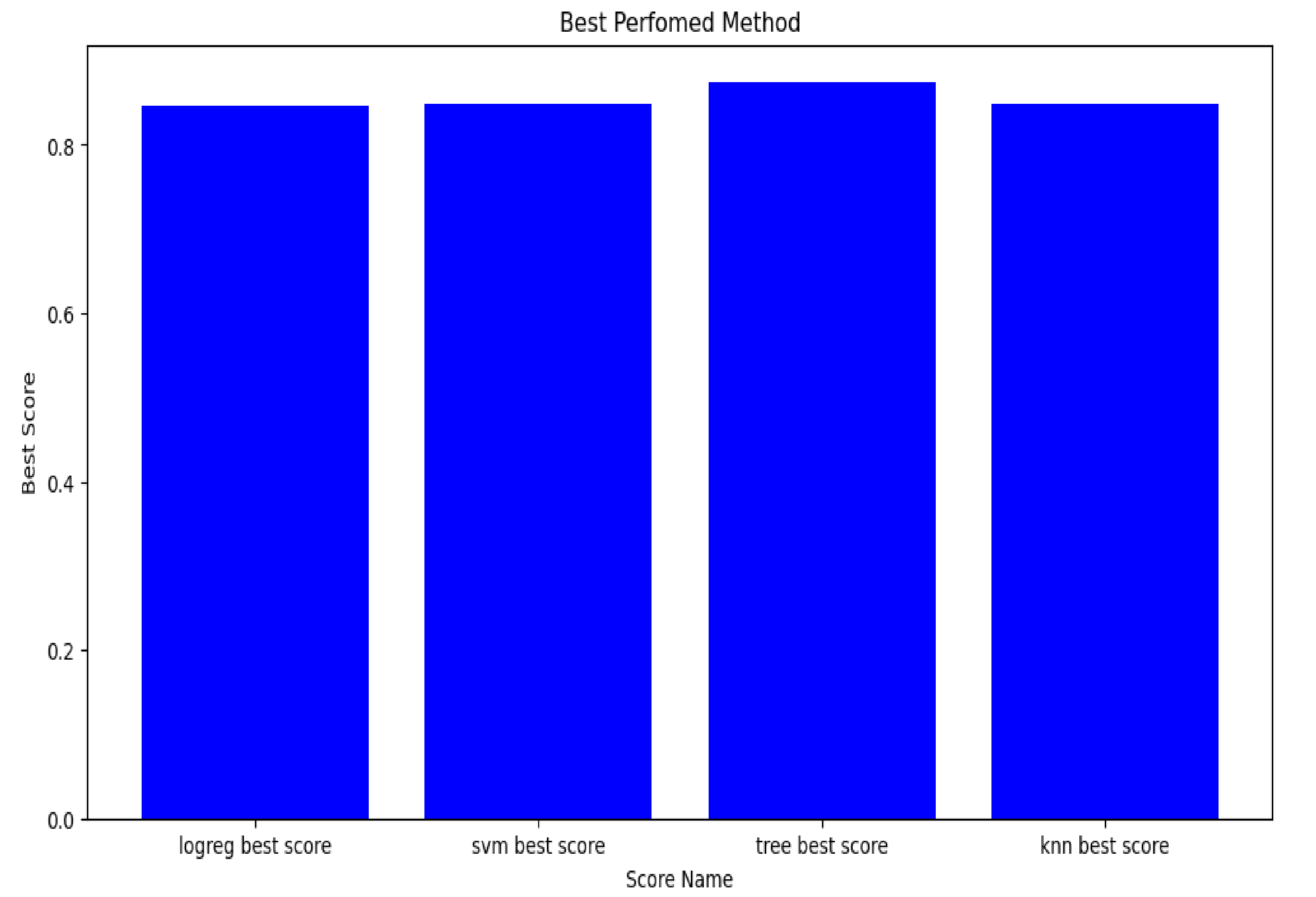
Conclusion, Failures have been all over the place while the success follows a similar grouping with only a few outlier

Section 5

Predictive Analysis (Classification)

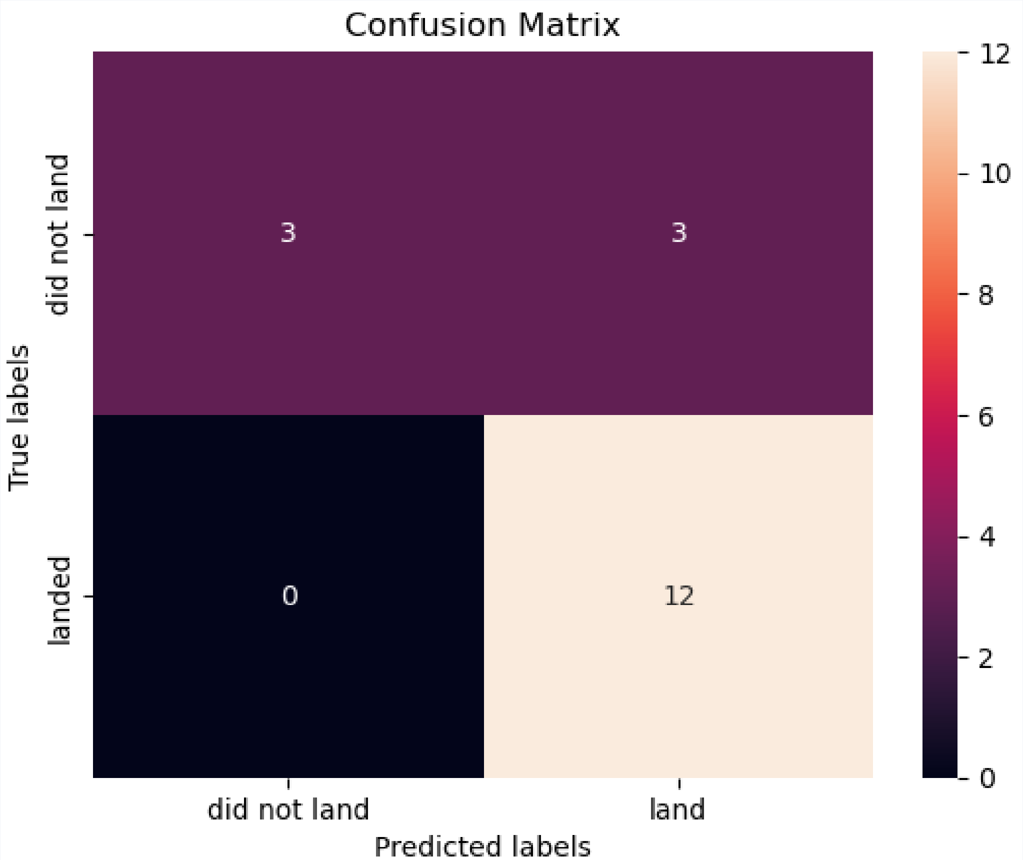
Classification Accuracy

- All of the models provided shows a very similar accuracy. With a being almost 83% with Tree Best Score having a higher accuracy of 87.5%



Confusion Matrix

- Confusion Matrix of the Best Tree Score
- The results has a the same matrix as the other provided scores, where the model predicted 12 success landings but the true labels show that of a split of 3 unsuccessful and 3 successful.



Conclusions

- The task was to provide the results of models for Space Y using Data Science knowledge
- **Methodology -** Using the Data from the public SpaceX API and web scraping off the Wikipedia Page
- **SQL -** Created the Labels and storing in the data through IBM's DB2 Database
- **Plotly Dash/EDA -** Dashboards were created to help provide a better visualization
- **Confusion Matrix -** Creating a predication Analysis with the result of the Best Tree Score at a 87.5%

Appendix

- Github: https://github.com/AlanWChang/IBM-DS-Capstone

Thank you!