IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Tanmay Chauhan
02-07-2024

Outline

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

Executive Summary

Summary of methodologies

1) We collected the data from SpaceX API
2) Then we performed some Data Wrangling and Data Cleaning.
3) Then we Performed the Exploratory Data Analysis (EDA) using SQL, Pandas and Matplotlib
4) After that we Created Interactive Visual Analysis Using Folium And other tools Like Dash to Build an Interactive Dashboards
5) At last we Performed Predictive Analysis using various Machine Learning Algorithms like
    1) KNN
    2) Linear Regression
    3) Decision Tree Classifier
    4) Grid Search CV
    5) Etc.…

Introduction

- Project background and context
  - The objective of the capstone project was to predict why the next SpaceX falcon 9 was so cheaper than its competitors (cost was 2.5X less then other)
  - The cost can be determined by weather the first stage will land or not.
  - By obtaining this information it can be used for other companies to bit against SpaceX launch.
- Problems you want to find answers to.
  - Problem1 was to predict if the falcon 9 first stage will land successfully of not.
  - If it was successful then the next stage was to determine what were the factors that caused the rocket to land successfully or not.
  - Problem 2 was to determine how can this information be used to bid against SpaceX by other companies to profit from it.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from the official SpaceX API into a Pandas dataset.
- Perform data wrangling
  - Data was preprocessed using various method described in the report below.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash.
- Perform predictive analysis using classification models

Data Collection

Collecting data from the SpaceX API using Python Request.get() Method

Source of the Data > https://api.spacexdata.com/v4/launches/past

Parsing of Dirty data using Pd.json_normalize()

Changing the data format .

Using various self created functions like getBoosterVersion(), getLaunchSite(), getPayloadData() and other methods to preprocess and clean the data according to need.

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- GitHub Link (Data Collection)
- https://github.com/iamtanmayc/New-Repository/blob/089e2bbb2f0b22924d1e67780cdb06be64e29395/jupyter-labs-spacex-data-collection-api.ipynb

Requesting the data

- Request and parse the SpaceX launch data using the GET requests

Data filtering

- Filtered the Data frame to only include Falcon 9 launch

Data Wrangling

- Dealt with missing values.

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- GitHub link of web scrapping notebook >

https://github.com/iamtanmayc/New-Repository/blob/a08238178434efafe54ba808d8f5d3c8a799e761/jupyter-labs-webscraping.ipynb

Requesting the data

- Requested the Falcon 9 Launch Wiki page from its URL

Extracting Columns

- Extracted all column/variable names from the HTML table header

Data Frame Creation

- Created a data frame by parsing the launch HTML

Data Wrangling

GitHub link of data Wrangling https://github.com/iamtanmayc/New-Repository/blob/844c4e64659c25797138baa11f74490dc7fdf11f/labs-jupyter-spacex-Data%20wrangling.ipynb

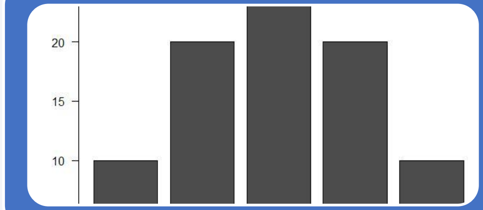
Step 1 Calculating the number of launches on each site

Step 2 Calculating the number and occurrence of each orbit

Step 3 Calculate the number and occurrence of mission outcome of the orbits

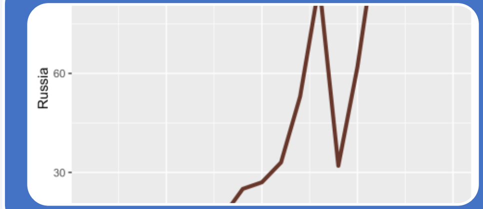
Step 4 Create a landing outcome label from Outcome column

EDA with Data Visualization



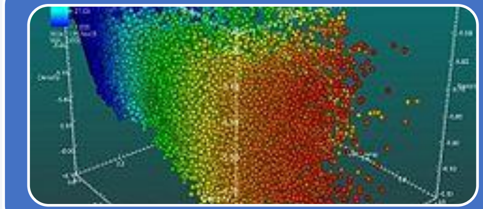
Seaborn Bar Plot

- Helpful for checking the relationship between success rate and orbit type



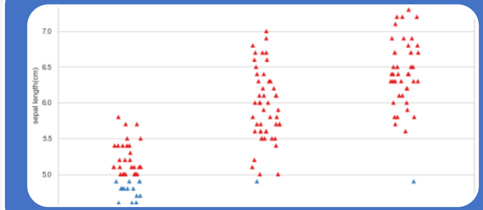
Seaborn Line Plot

- Used to find the trend according to Class



Seaborn Scatter Plot

- Used to observe if there is any relationship between launch site and their payload mass



Seaborn Cat Plot

- Used for finding the relationship between Flight number and Payload
- GitHub link => https://github.com/iamtanmayc/New-Repository/blob/ff484bb01656d8125e833279d9f8a927bfb68f6c/edadataviz.ipynb

EDA with SQL

GitHub Link => https://github.com/iamtanmayc/New-Repository/blob/e9aac028e585f2ab9c14cdf1b2915de62e5569fd/jupyter-labs-eda-sql-coursera_sqllite%20(1).ipynb

Displayed the names of the unique launch sites in space mission.

Displayed top5 records where launch sites begin with the string CCA

The total payload mass carried by boosters launches by NASA

Used MIN function to display landing outcome in ground pad achieved

Used some big Queries like SELECT Mission_Outcome, COUNT(*) AS Total.FROM SPACEXTABLE GROUP BY Mission_Outcome

Build an Interactive Map with Folium

GitHub link => https://github.com/iamtanmayc/New-Repository/blob/c1f7017ba6b53398ae498d7e419381c6057b0ee7/lab_jupyter_launch_site_location.ipynb

Added marker to all the launch of SpaceX sites on the map

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

Colour wise marked the Successful and Unsuccessful launches of all the launch site launches

- RED Marker for Failed/Unsuccessful landing
- GREEN Marker for Successful landing
- Calculated the Distances Between a launch Site to its Proximities

Nearest Railway Track

- Nearest Coastline
- Nearest City
- Nearest Road Track

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The GitHub link is here: https://github.com/iamtanmayc/New-Repository/blob/16542f9da2b2201db7cee7678e16ab2d23769473/5%20spacex_dash_app.py

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- GitHub link :https://github.com/iamtanmayc/Data-science-Capstone/blob/8ba4826c3a1719c6e65b03af370def42bdba1ff3/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

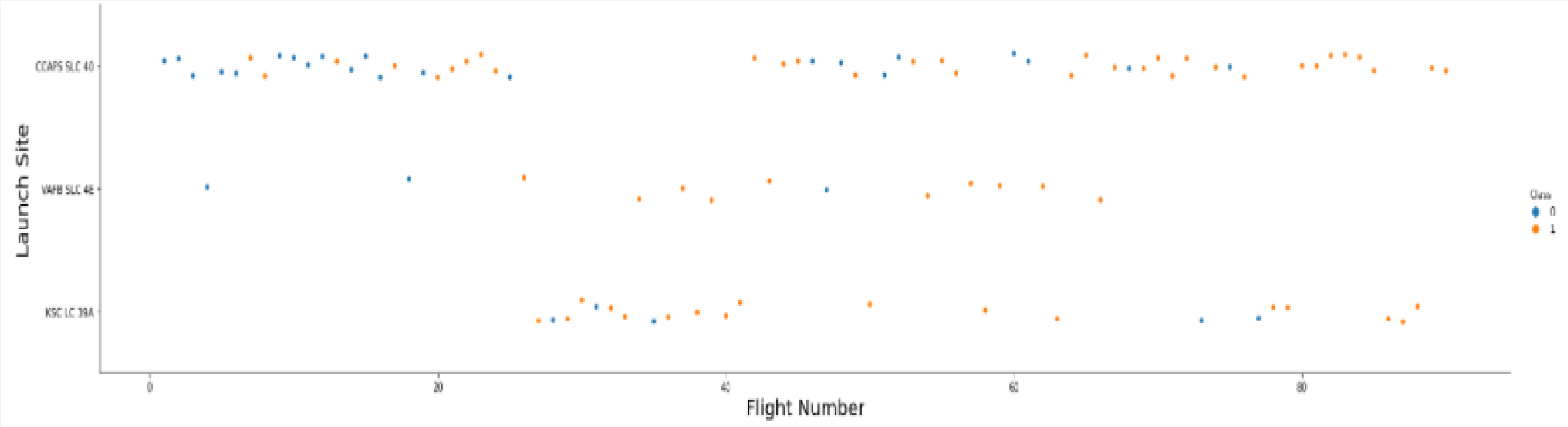
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

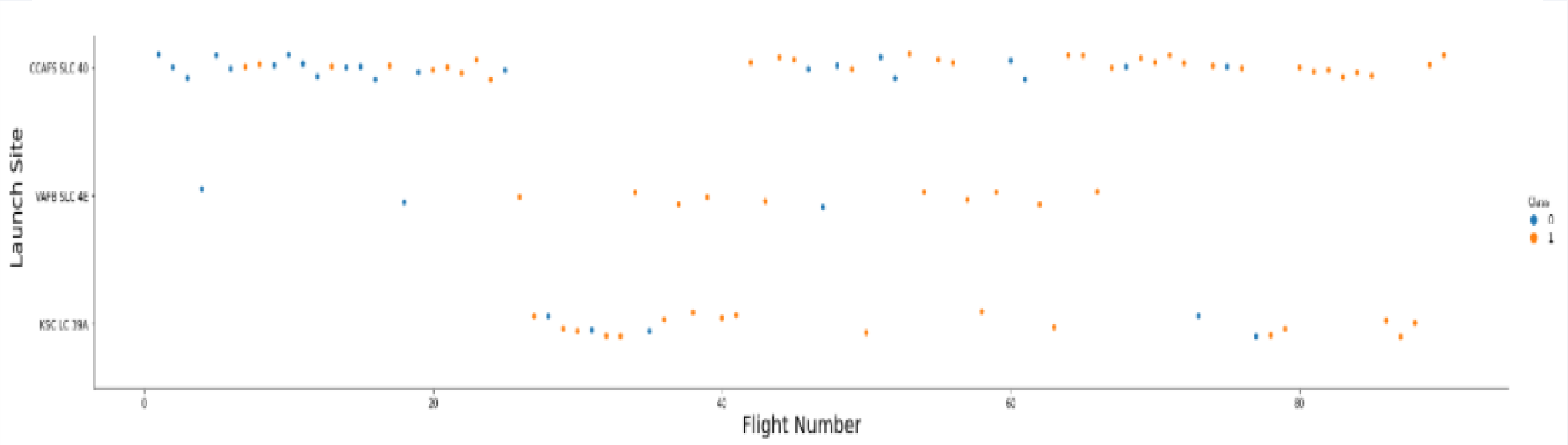
Flight Number vs. Launch Site

- Number vs. Launch Site
  - From the plot we can see that the more rockets a site launches the higher the success rate for that site launches



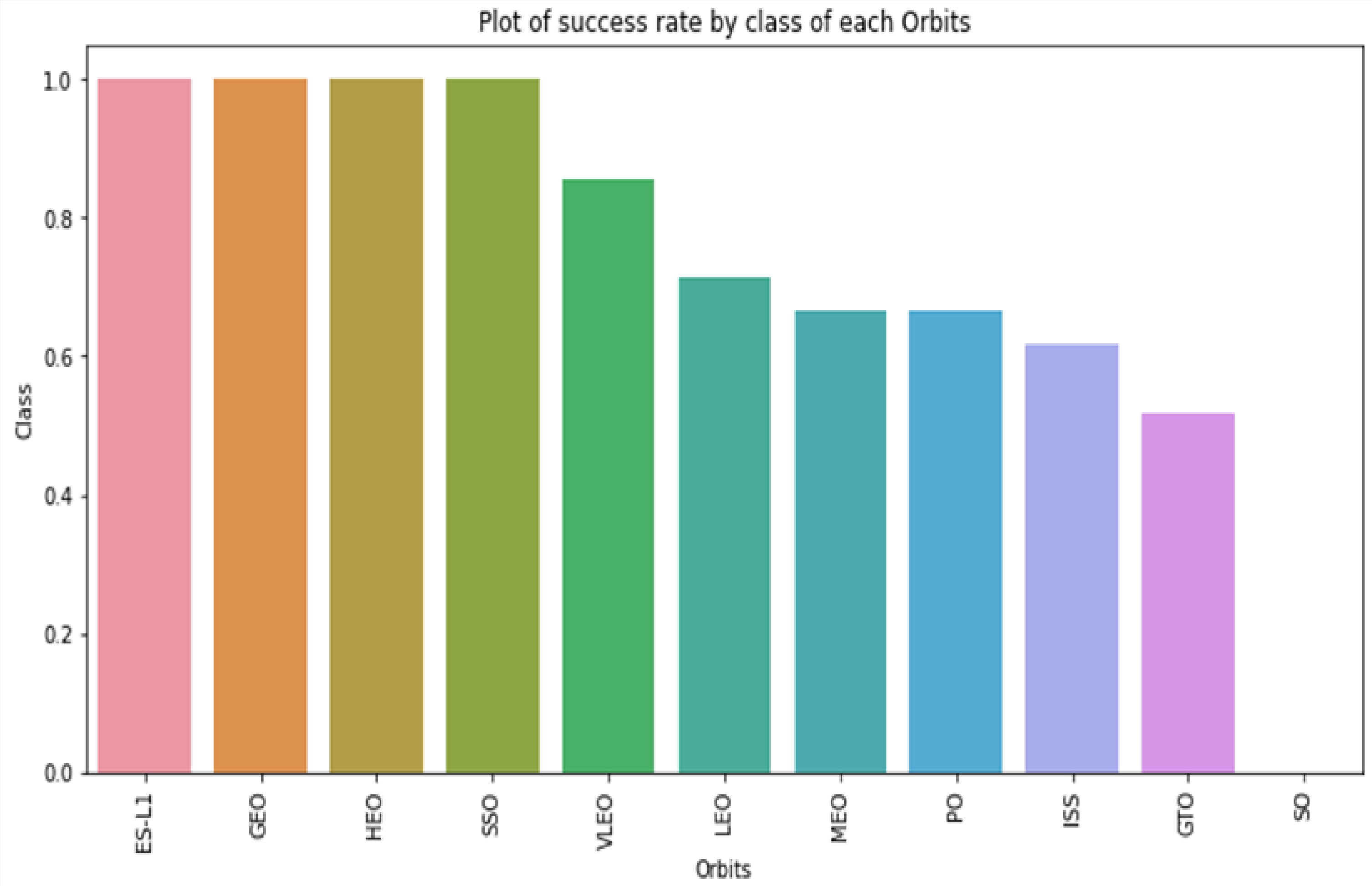
Payload vs. Launch Site

The greater the payload mass for launch site CCAFS SLC 40 the higher the success rate for the rocket.



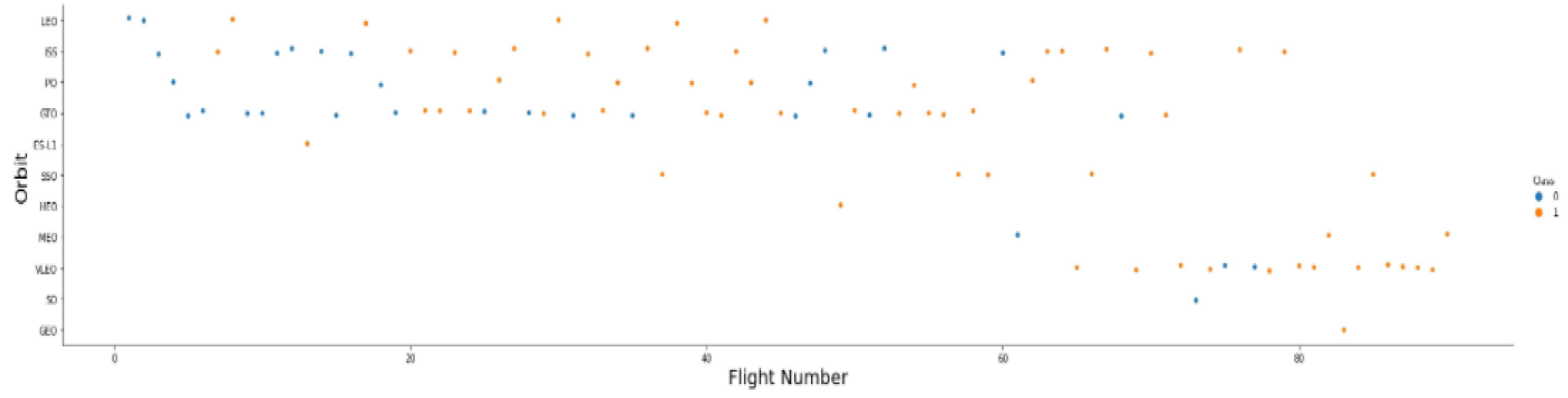
Success Rate vs. Orbit Type

- As show in the bar chart the success rate of ES-L1, GEO, HEO and SSO are relatively higher than the other orbits.



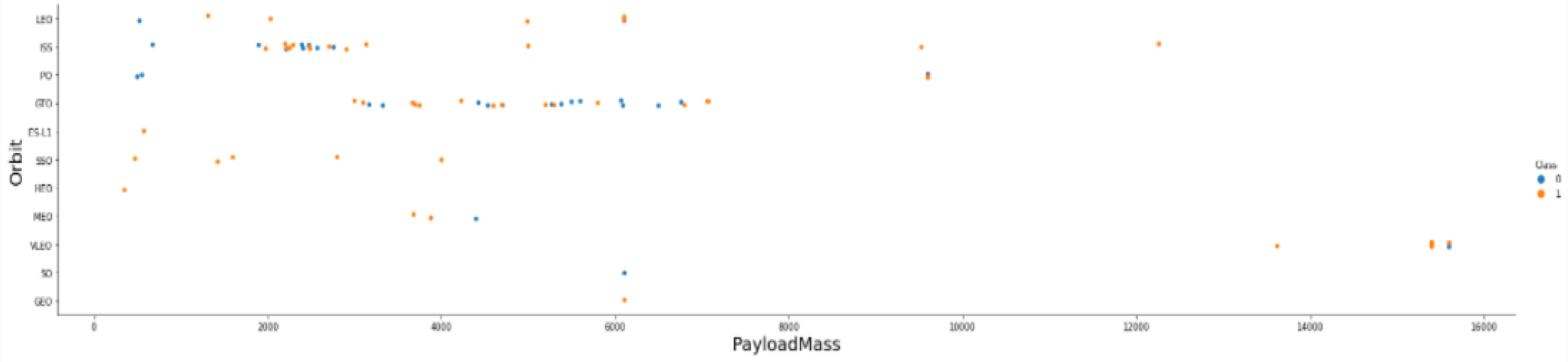
Flight Number vs. Orbit Type

The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



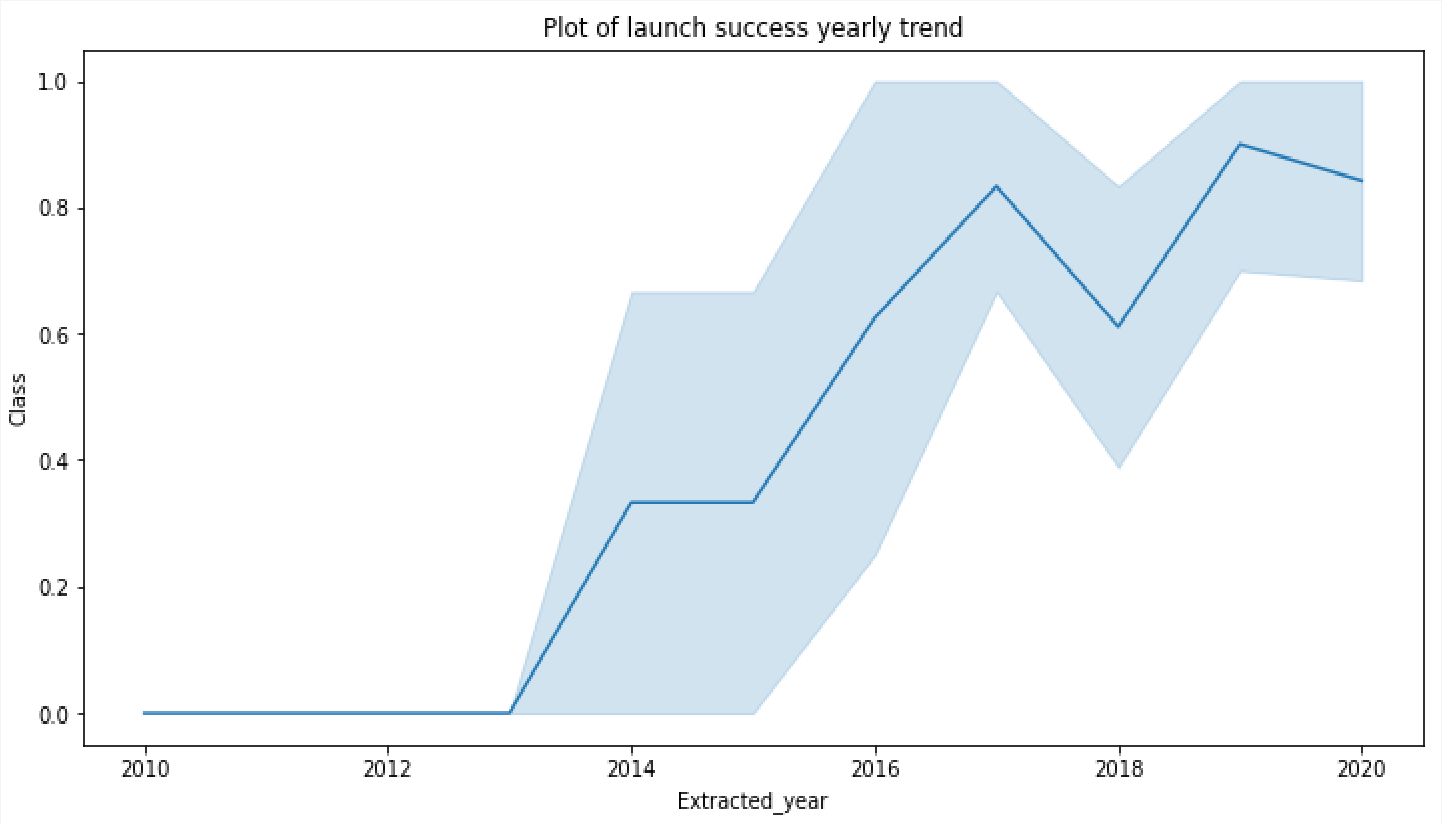
Payload vs. Orbit Type

- The plot below describes that with heavy payload the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- From this plot it can be observed that the success rate since 2013 kept on increasing till 2020



All Launch Site Names

- The SQL magic command Distinct was used to show only the unique launch sites from the SpaceX data.

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

In [10]:

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- We used the query below to display 5 records where launch sites begin with ‘CCA’

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

In [11]:

```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 45596 using the query below

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

In [12]:

```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

Calculating the average payload mass carried by booster version F9 v1.1 as 2928.4

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

In [13]:

```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- From the code below it can be observed that the dates of the first successful landing outcomes on ground pad was 22^{nd} December 2015

In [14]:
```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

In [15]:

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [16]:
```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

```
The total number of successful mission outcome is:
```

	successoutcome
0	100

```
The total number of failed mission outcome is:
```

Out[16]:

	failureoutcome
0	1

- The SQL WHERE query was used to find the count of success outcomes from the SpaceX data

Boosters Carried Maximum Payload

- We determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

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

In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

- We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

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

In [18]:

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

In [19]:

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

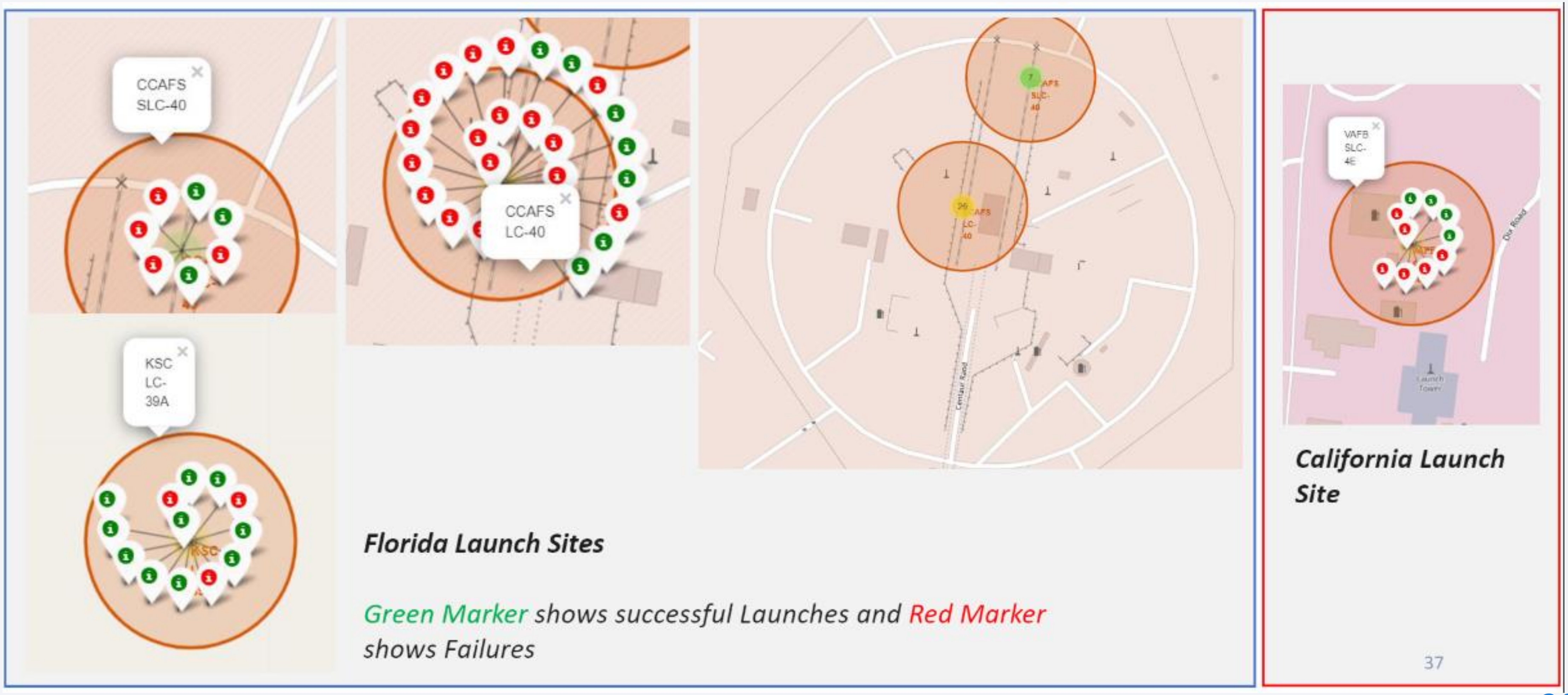
- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

Section 3

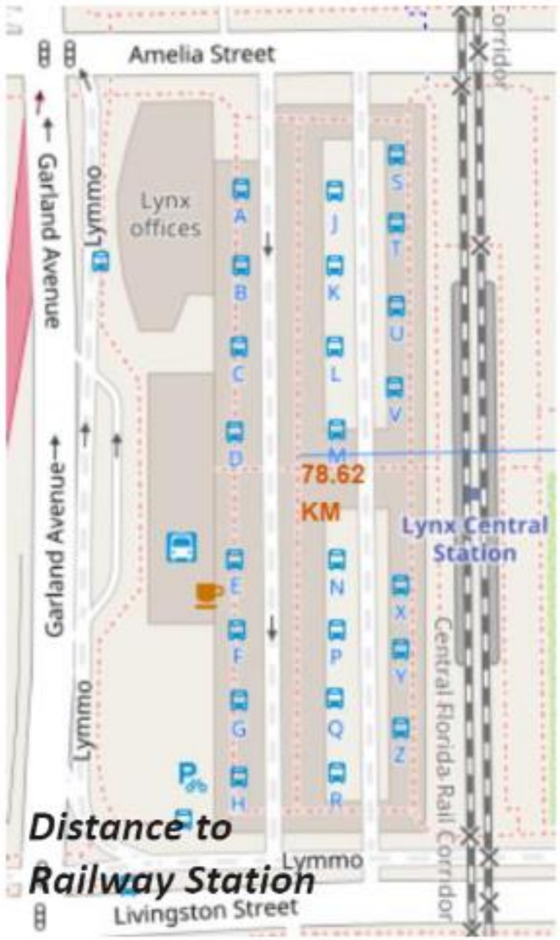
Launch Sites Proximities Analysis

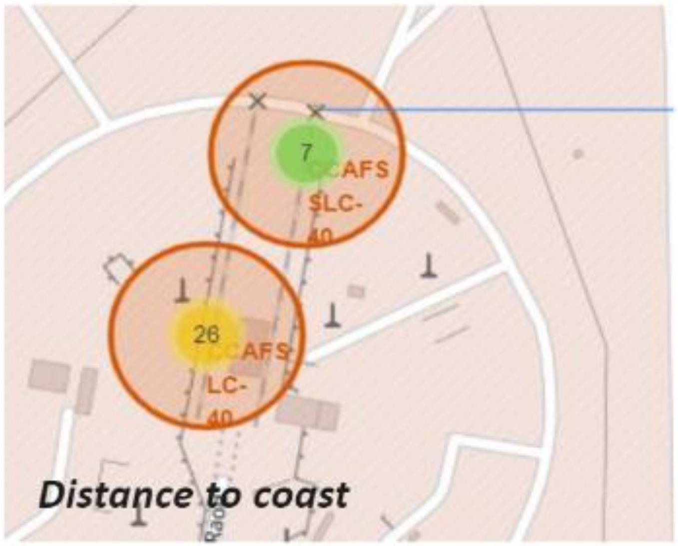
All launch site marked on the global map

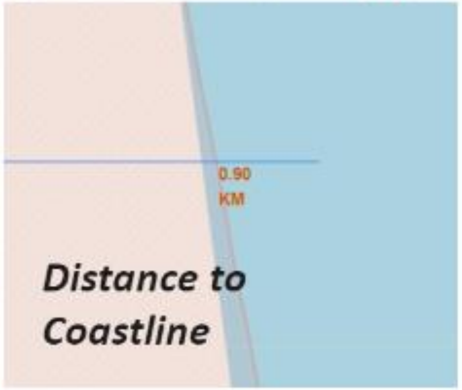
Marking the launch sites with colored labels



Marking the launch site distance of railways, roads etc...





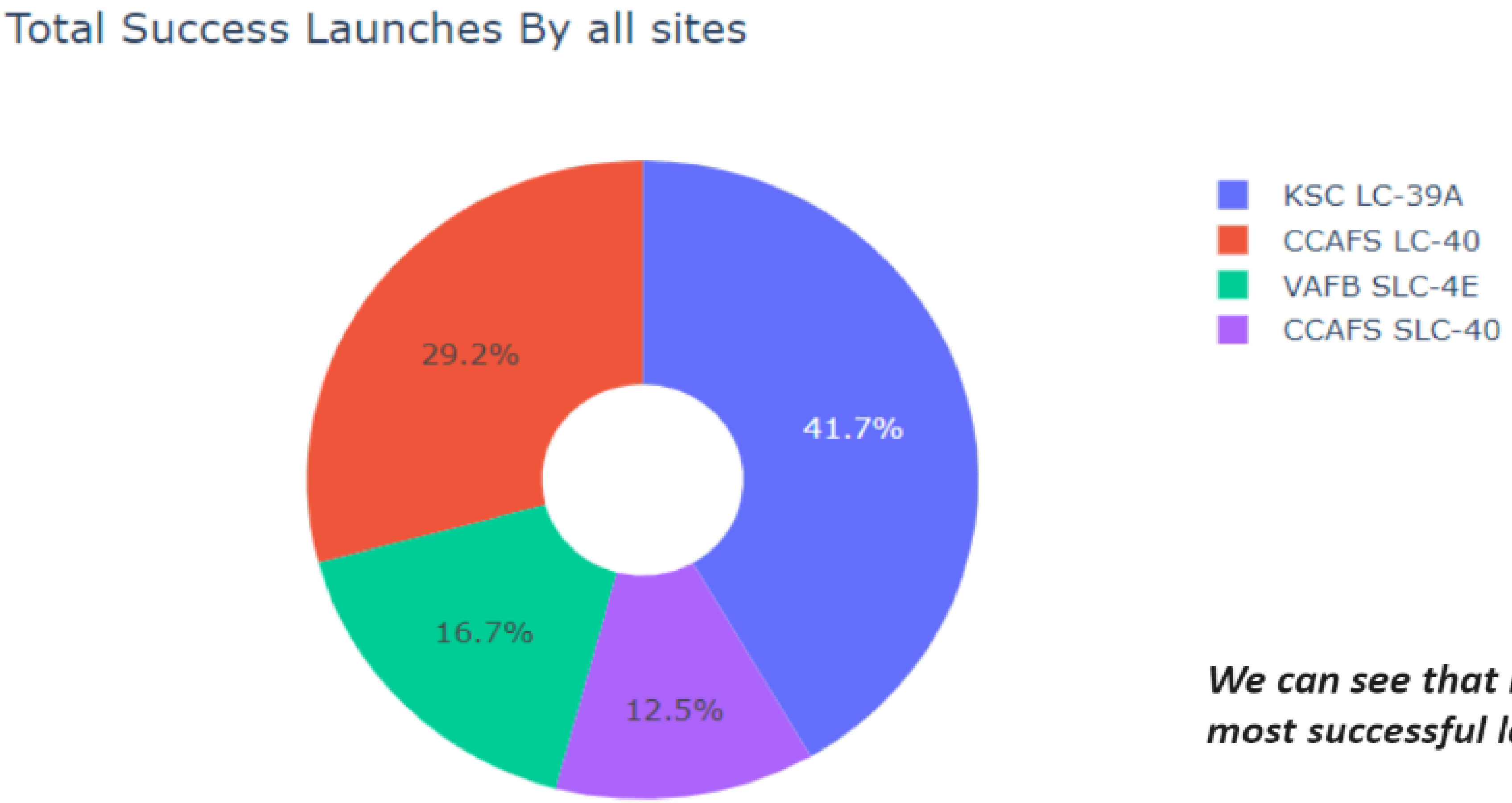


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

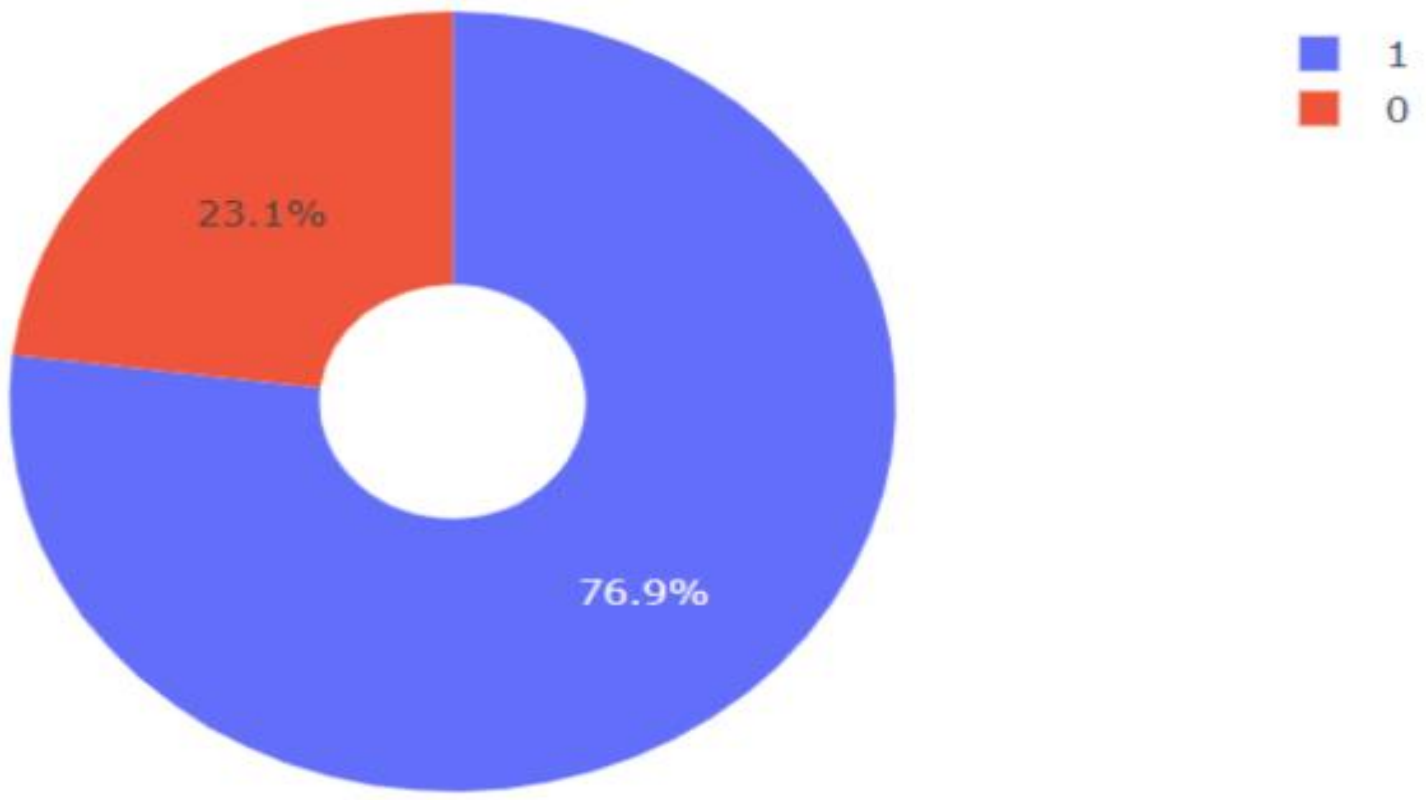
Build a Dashboard with Plotly Dash

Pie chart of success percentage achieved by each launch site.



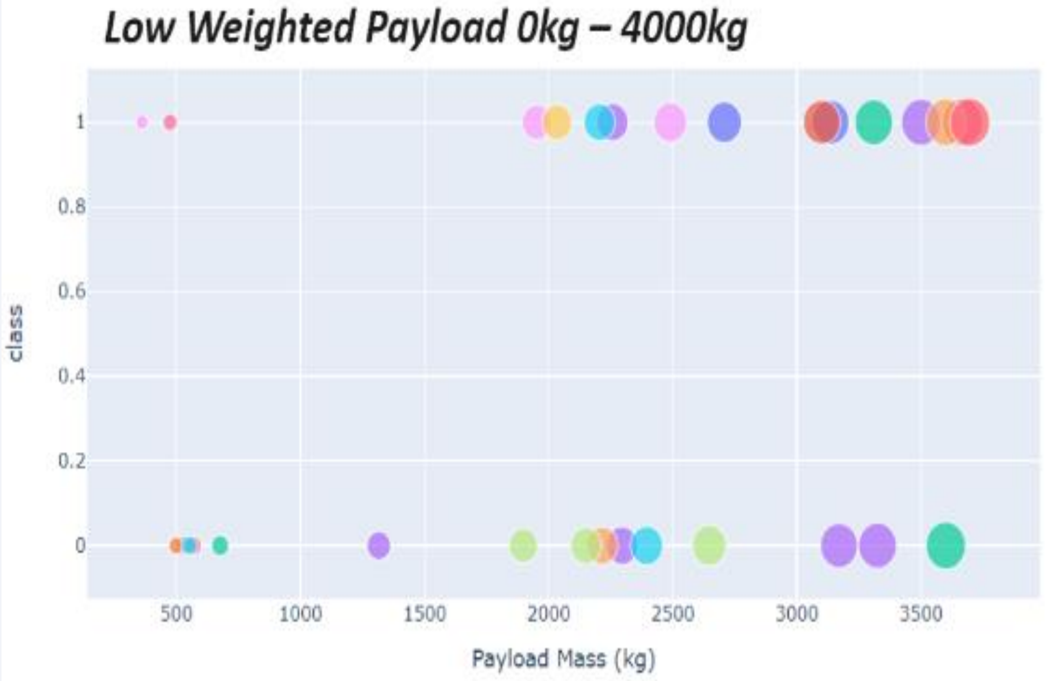
We can see that KSC LC-39A had the most successful launches from all the sites

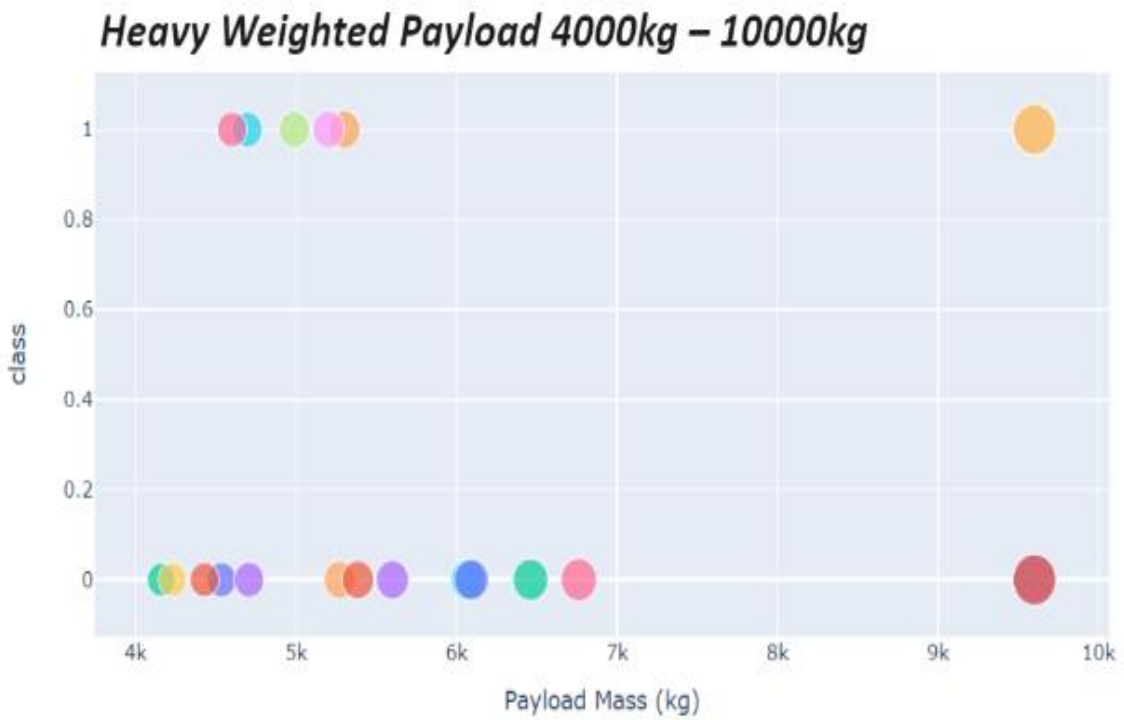
Pie chart showing the Launch site with the highest launch success ratio



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

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider





We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The decision tree classifier is the model with the highest classification accuracy

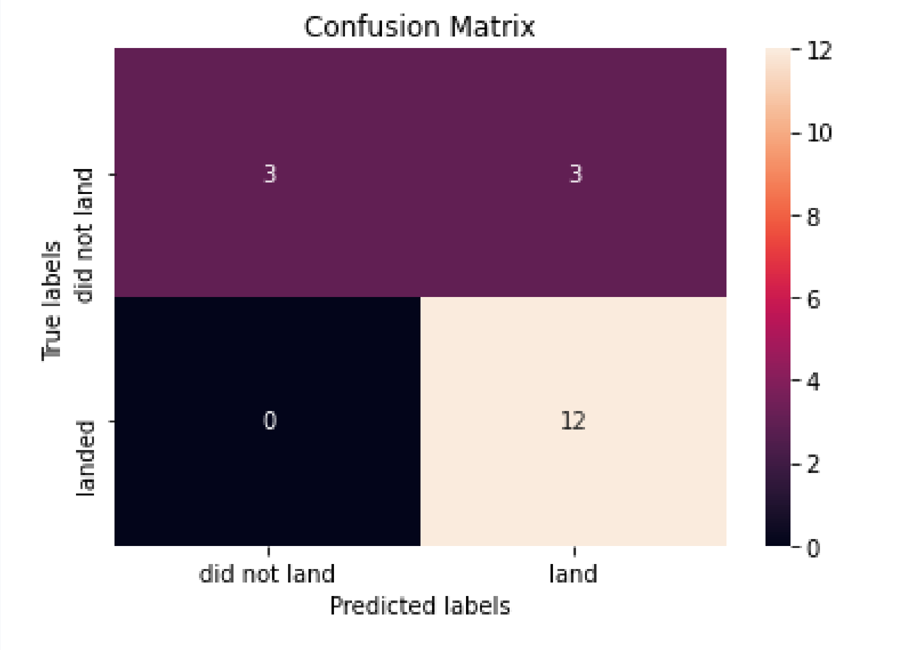
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```


43

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

- We can conclude that:
- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!