IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Enoch Boateng
July 14th, 2022

Outline

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

Executive Summary

- Summary of methodologies
  - ✓ Data Collection using API
  - ✓ Data Collection with Web Scraping
  - ✓ Data Wrangling
  - ✓ Performed Exploratory Analysis with SQL
  - ✓ Performed Exploration Analysis with Visualization
  - ✓ Performed Interactive Visual Analytics folium and Dashboard with Plotly Dash
  - ✓ Performed Machine Learning Prediction Analysis
- Summary of all results
  - ✓ Exploratory Data Analysis Result
  - ✓ Interactive Analytics in Screenshot
  - ✓ Predictive Analytics Result from Machine Learning

Introduction

As SpaceX has advertised their Falcon 9 rocket launch successes on its website, with a cost of 62 million dollars by re-use of it first stage unit; whiles it is cost their competitors in the upwards of 165 million dollars. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. Our goal in this project is to create a machine learning model to predict the successful land outcome of the first stage and a better cost to bid against SpaceX.

These are the following problem to solve:

- ✓ Finding the features that affects the landing outcome.
- ✓ The relationship between each variables and how it is affecting outcome.
- ✓ The best condition needed to increase the probability of successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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 SpaceX launch data was gathered from an API, specifically the SpaceX REST API and also Web Scrapping from Wikipedia.

For REST API, get requests was use to collect data from SpaceX REST API. Then, decoded the content to a JSON and turned it to Pandas dataframe using json normalization method. Data Wrangling was performed to clean the data.

For Web Scrapping , BeautifulSoup was used to collect data from HTML tables that contain valuable records. Then, parse the data from tables and convert them into Pandas data frame for visualization and analysis.

Data Collection – SpaceX API

Request data from SpaceX Rest API

↓

Convert JSON to DataFrame and use json_normalize function to normalize data into flat table

↓

Wrangling data using an API to sample and clean data

[Data Collection and Wrangling Github Link]

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

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

```
[12]: # Use json_normalize meethod to convert the json result into a dataframe
      # decode response content as json
      static_json_df = res.json()

[13]: # apply json_normalize
      data = pd.json_normalize(static_json_df)
```

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

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

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

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

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

- From the `rocket` we would like to learn the booster name

Data Collection - Scraping

Request the Falcon 9 Launch from Wiki page URL

Create a BeautifulSoup Object from the HTML response

Extract all column/variable names from the HTML header

[Data Collection - Scraping Link to Github Notebook]

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

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

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

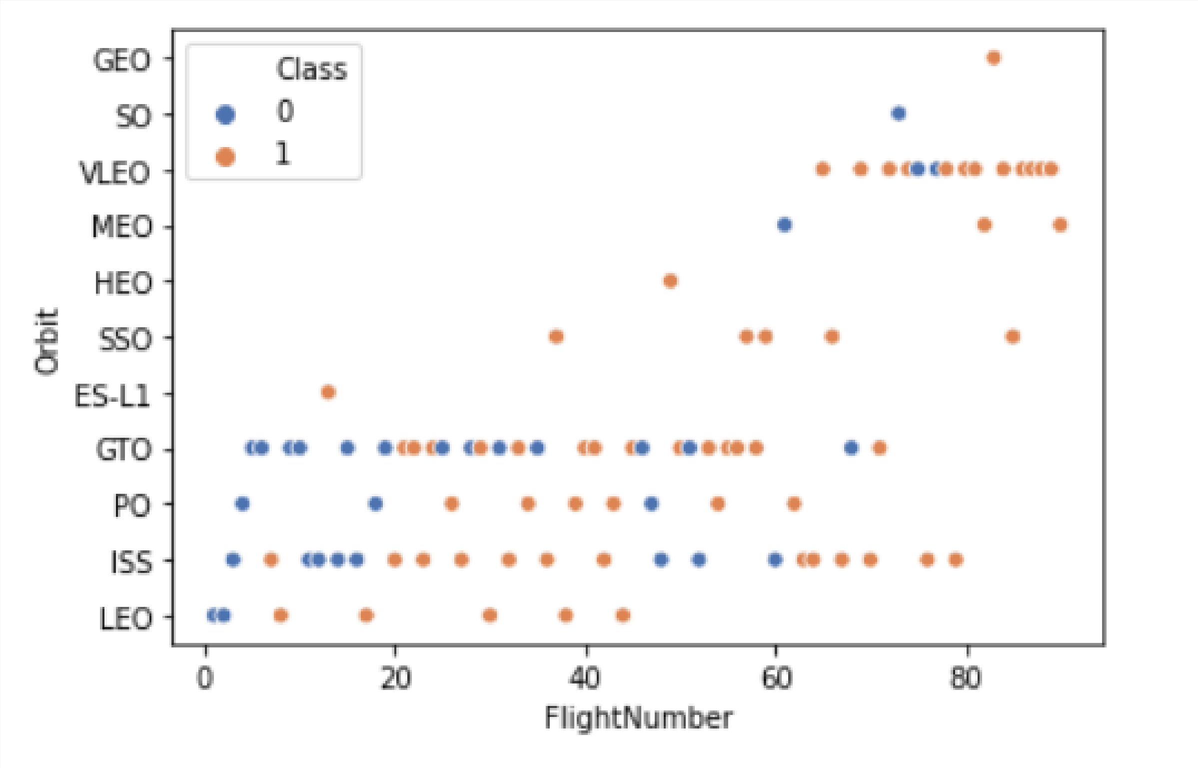
```
[13]: extracted_row = 0
      #Extract each table
      for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
         # get table row
          for rows in table.find_all("tr"):
              #check to see if first table heading is as number corresponding to launch a number
              if rows.th:
                  if rows.th.string:
                      flight_number=rows.th.string.strip()
                      flag=flight_number.isdigit()
              else:
                  flag=False
              #get table element
              row=rows.find_all('td')
              #if it is number save cells in a dictonary
              if flag:
                  extracted_row += 1
                  # Flight Number value
                  # TODO: Append the flight_number into launch_dict with key `Flight No.`
                  #print(flight_number)
                  datatimelist=date_time(row[0])

                  # Date value
                  # TODO: Append the date into launch_dict with key `Date`
                  date = datatimelist[0].strip(',')
                  #print(date)
```

Data Wrangling

- ✓Rows with missing values where identified.
- ✓Dealing with missing values - the mean and the replace functions were used to complete the dataset .
- ✓Create a landing outcome label from the outcome column, for analysis, visualization, and Machine Learning

[Data Wrangling Link Github Notebook]

EDA with Data Visualization



Using scatter graph to find the relationship between the features such as:

✓Payload and Flight Number.

✓Flight Number and launch Site.

✓Flight Number and Orbit Type.

✓Payload and Launch Site.

✓Payload and Orbit Type.

EDA with Data Visualization Link to Github Notebook

EDA with SQL

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

SQL Notebook Link to Github Notebook

Build an Interactive Map with Folium

- ✓For visualization, latitude and longitude coordinates of each launch site from the dataset were located by circle markers with labeled names.
- ✓Map enhancement, by adding launch outcomes(Failure or Success) for each sites. Adding outcome class(0,1) to find high success rate.
- ✓Calculating the distance of the launch sites to various landmark to find answer to the question of:
  - I. How close the launch sites with railways, highways, and coastlines?
  - II. How close the launch sites with nearby cities?

[Interactive Map with Folium Link to Github Notebook]

Build a Dashboard with Plotly Dash

- ✓ Success-pie chart and success-payload-scatter-chart were built to generate interactive dashboard by using Plotly Dash
- ✓ To obtain some insight to answer the following questions:
  - I. Which site has the largest successful launches?
  - II. Which site has the highest launch success rate?
  - III. Which payload range(s) has the highest launch success rate?
  - IV. Which payload range(s) has the lowest launch success rate?
  - V. Which F9 Booster version (`v1.0`, `v1.1`, FT, B4, B5, etc.) has the highest launch success rate?

[Interactive Dashboard with Plotly Dash link to Github Notebook]

Predictive Analysis (Classification)

Building Model

- ✓ Load the dataframe and creating NumPy array.
- ✓ Standardizing and transforming data.
- ✓ Split data into test and training dataset.
- ✓ Models are trained and hyperparameters are selected using GridSearchCV.

Model Evaluation

- ✓ Check the accuracy for each model.
- ✓ Get tuned hyperparameters for each algorithm.
- ✓ Confusion Matrix for visualization of the performance of an algorithm.

Model Tuning

- ✓ Using the best hyperparameter values.

Find the Best Model

- ✓ Determine the model with the best accuracy using the training data.

Predictive Analysis link to Github Notebook

Results

TASK 7

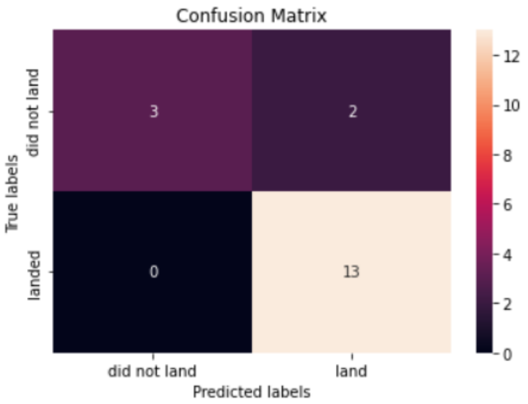
Calculate the accuracy on the test data using the method `score` :

In [69]:
```
svm_cv.score(X_test, y_test)
```

Out[69]:
```
0.8888888888888888
```

We can plot the confusion matrix

In [70]:
```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(y_test,yhat)
```



TASK 5

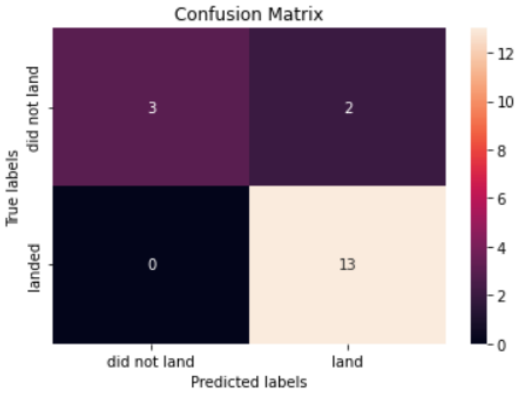
Calculate the accuracy on the test data using the method `score` :

In [63]:
```
logreg_cv.score(X_test, y_test)
```

Out[63]:
```
0.8888888888888888
```

Lets look at the confusion matrix:

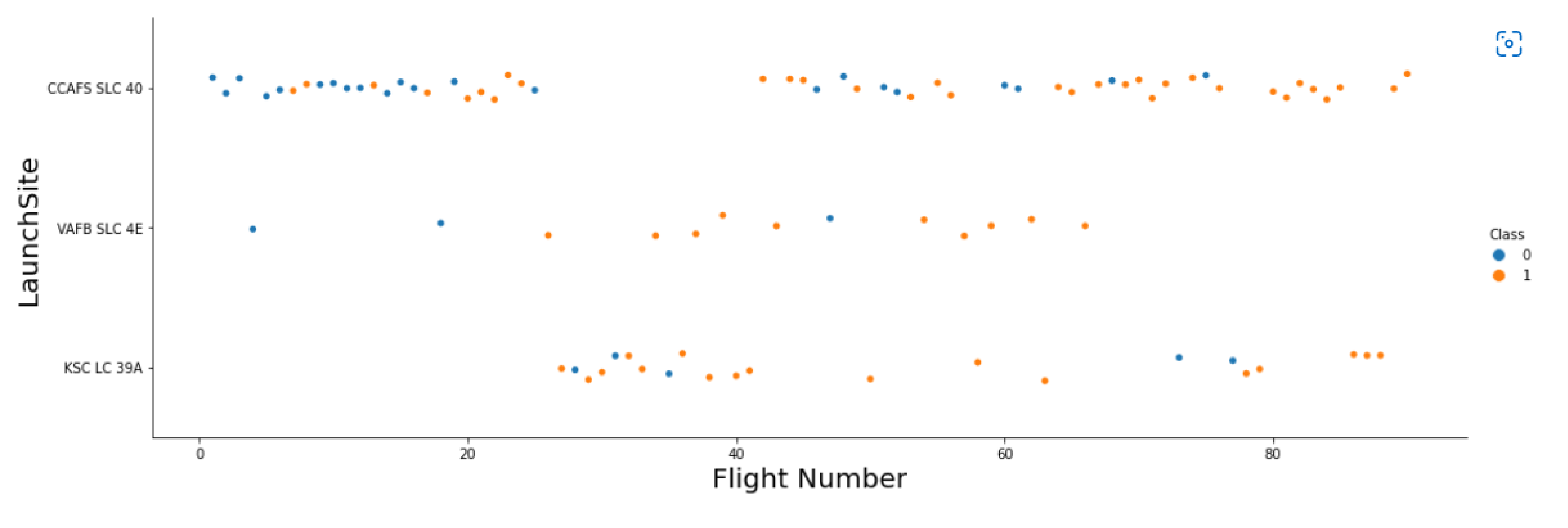
In [64]:
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(y_test,yhat)
```



Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

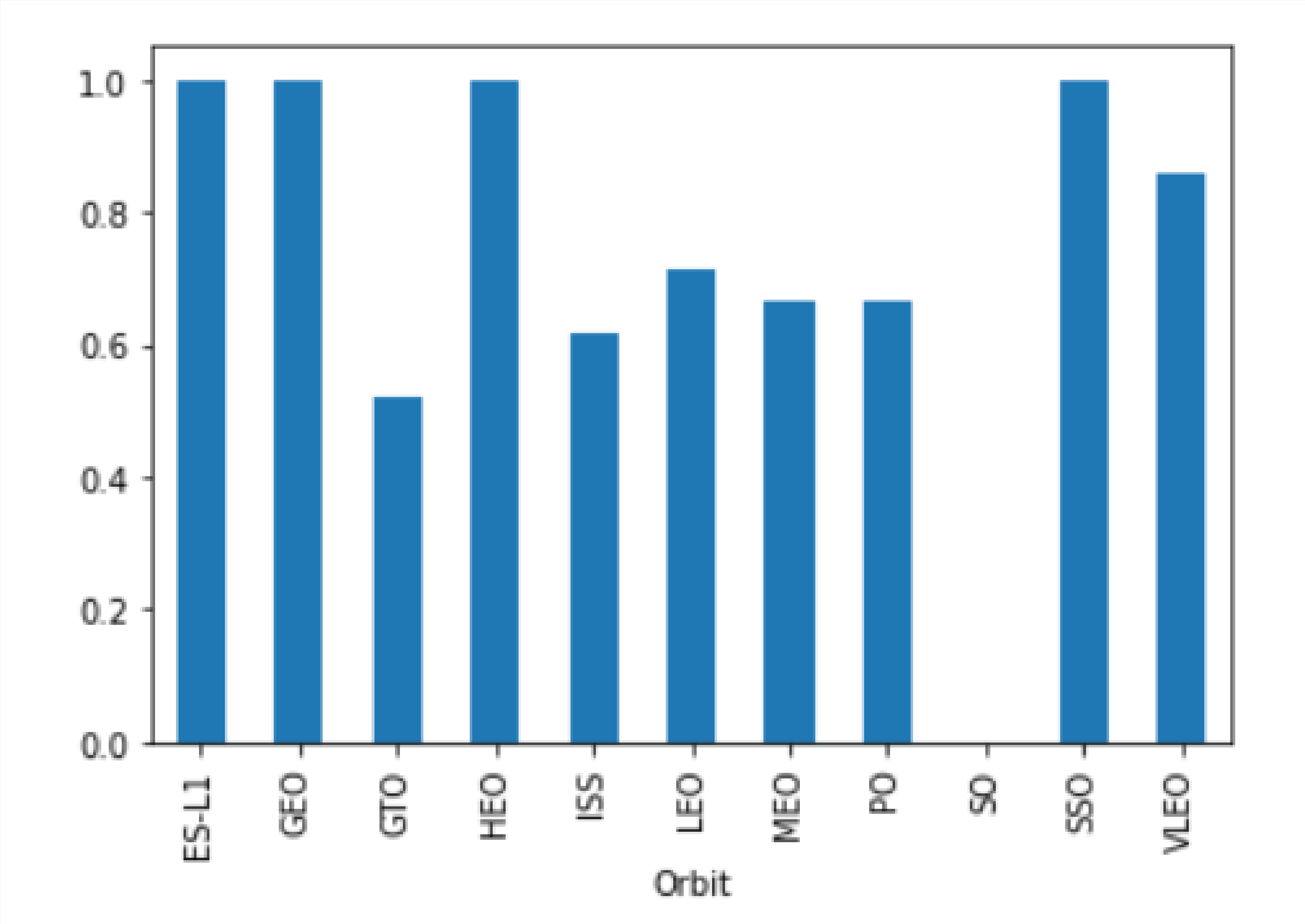


The scatter plot shows success improves as the number increases per launch site.

Payload vs. Launch Site

From the scatter, as the pay load mass increase from 7000kg, the probability of success rate increases highly.

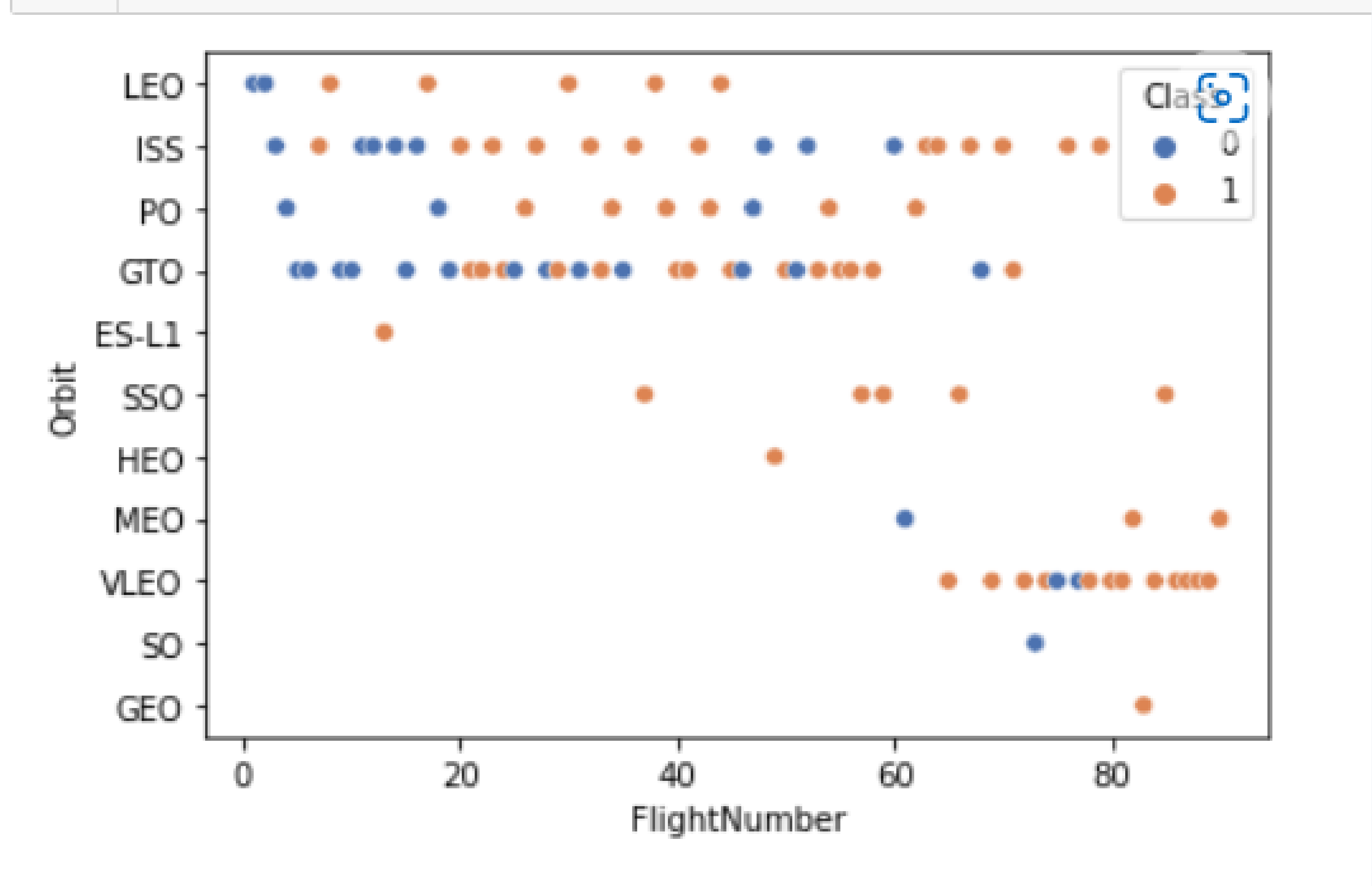
Success Rate vs. Orbit Type



✓The figure shows the outcomes of the orbits to influences the landing outcomes as some orbits has 100% success rate such as SSO, HEO, GEO, and ESL1 while SO orbit produced 0% rate of success.

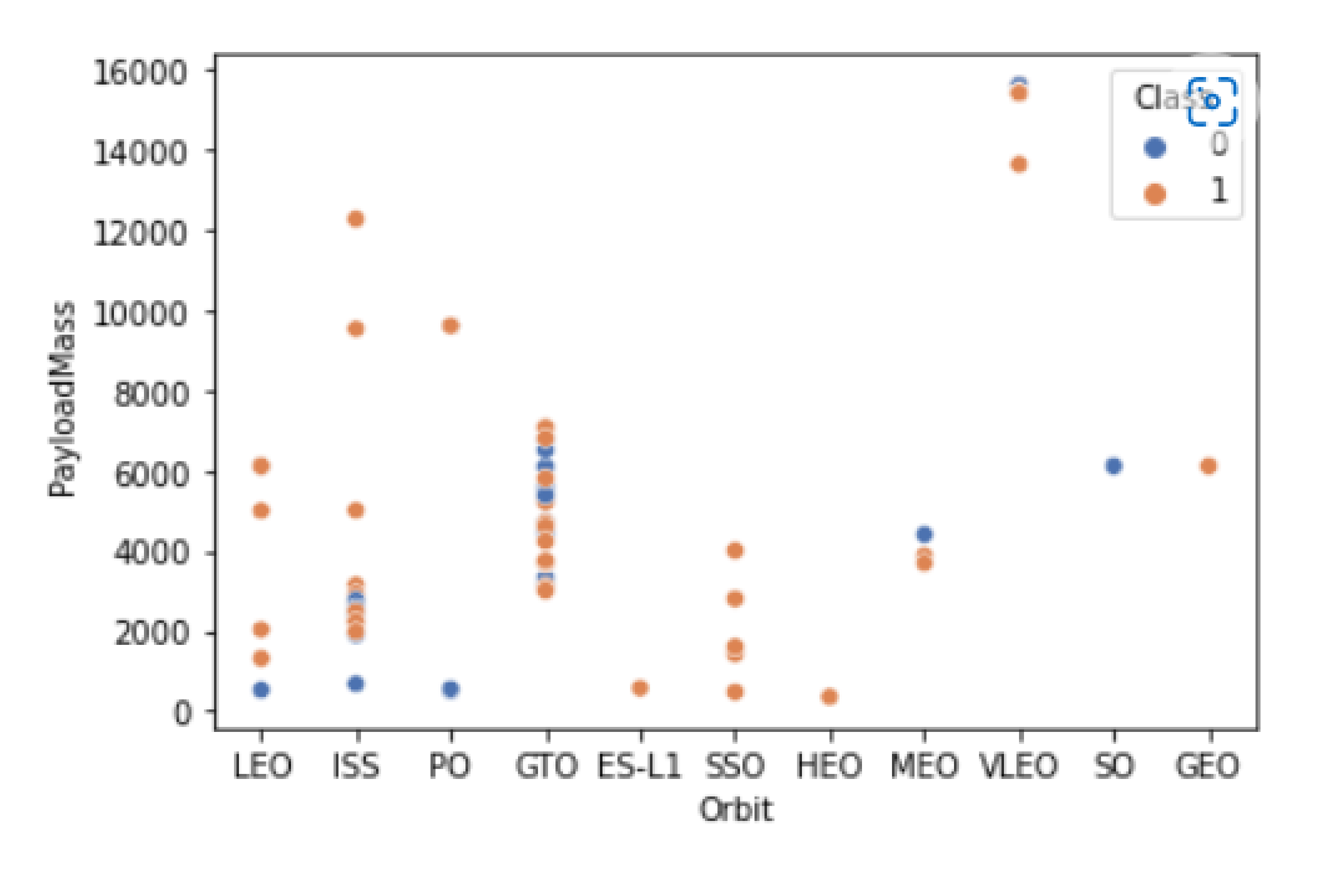
✓Analysis show that some of this orbits has only 1 occurrence such as GEO, SO, HEO, and ES-L1 which mean this data need more dataset to see pattern or trend before we draw any conclusion

Flight Number vs. Orbit Type



The scatter plot , shows a good number of success rate where Flight Number is greater than 60

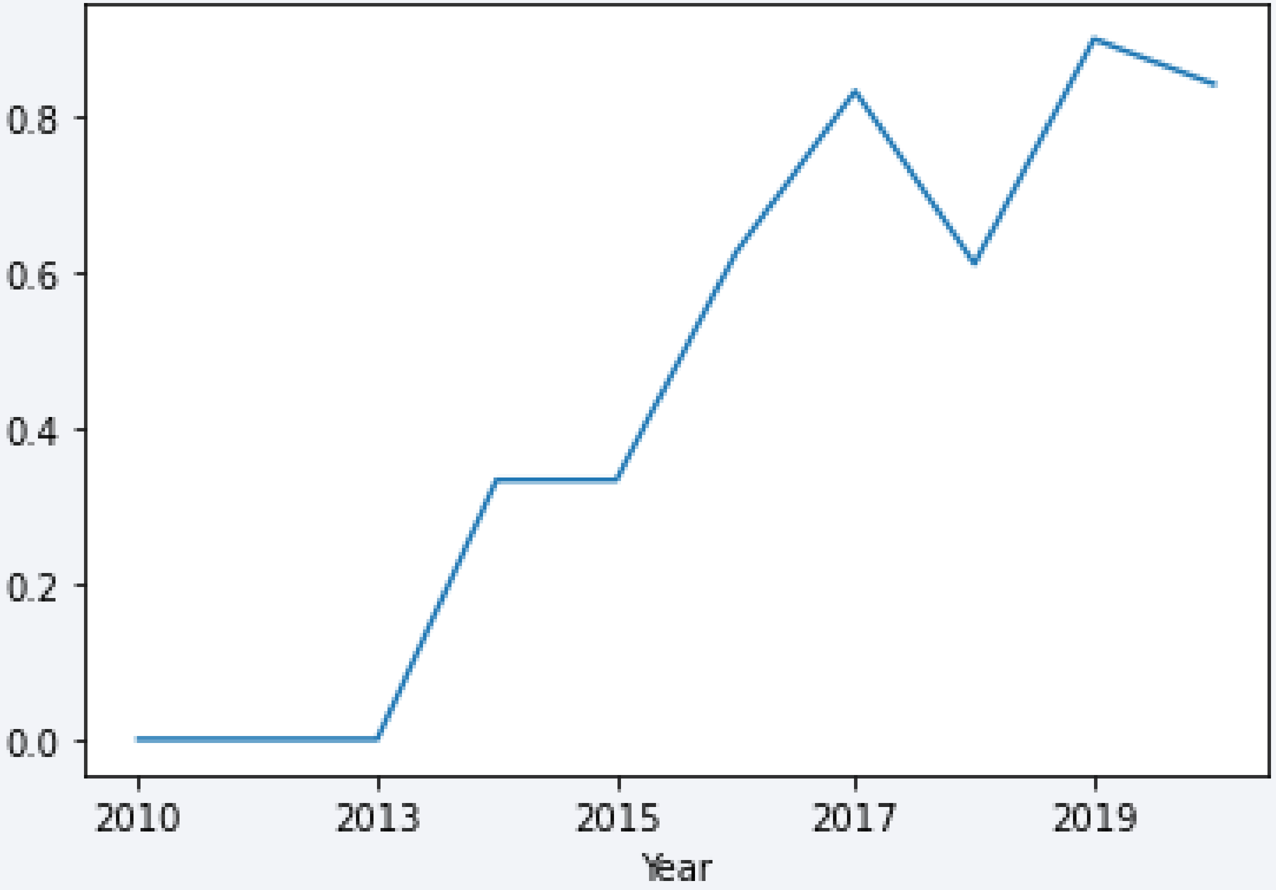
Payload vs. Orbit Type



Heavier payload has positive impact on LEO, ISS, and PO orbit. Negative impact on MEO and VLEO orbit.

GTO orbit seem to depict no relation between the attributes.

Launch Success Yearly Trend



From the graph, there is an increasing trend from 2013 to 2020.

If this continuous increasing in success rate trend, it can be project to reach a success rate of 1.

All Launch Site Names

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

In [9]:

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

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

Out[9]:

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

DISTINCT was used to SELECT launch sites from SPACEXTBL

Launch Site Names Begin with 'CCA'

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

```
0]:   1 %sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

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

ut[10]:

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

The Query outputs launch sites with only 5 records and starting with CCA.

Total Payload Mass

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

```
]:  1 %sql select sum(PAYLOAD_MASS__KG_) as 'total payload mass' from SPACEXTBL where Customer = 'NASA (CRS)';
```

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

t[26]:

total payload mass
45596

The query outputs 619967kg of total payload mass carried by boosters from NASA(CRS)

Average Payload Mass by F9 v1.1

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

```
28]:    1  %sql select Avg(PAYLOAD_MASS__KG_) as 'average payload mass' from SPACEXTBL where Booster_Version = 'F9 v1.1';
```

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

Out[28]:

average payload mass
2928.4

The query outputs average of PAYLOAD_MASS__KG_ of Booster_Version is 2928.4kg

First Successful Ground Landing Date

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

Hint:Use min function

```
6]:  1 %sql select min(DATE) as 'first successful landing outcome'from SPACEXTBL where landing_outcome ='success (ground pad)'

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

The query outputs the first successful ground landing on ground pad date as 2015-12-22.

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
]:    1 3OOSTER_VERSION FROM SPACEXTBL WHERE Landing_Outcome ='Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000

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

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- Present your query result with a short explanation here

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

]:

```
%sql select mission_outcome, count(MISSION_OUTCOME) as missionoutcomes from SPACEXTBL GROUP BY MISSION_OUTCOME;
```

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

t[42]:

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

Query mission_outcome and aggregate mission_outcome group by mission_outcome

Boosters Carried Maximum Payload

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

n [32]:
```
BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

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

Out[32]:

boosterversion
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

The query outputs the booster that have carried the maximum payload using a sub-query in the WHERE clause to the maximum payload mass.

2015 Launch Records

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [44]:

```
%sql SELECT substr(DATE,4,2),BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXTBL
        where substr(Date,7,4)='2015'and landing_outcome ='success (drone ship)' ;
```

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

The query outputs months, booster version, and landing outcome using the WHERE clause.

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

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT DATE, COUNT(LANDING_OUTCOME) as COUNT FROM SPACEXTBL
               WHERE DATE BETWEEN '2010-06-04' and '2017-03-20' AND
                  LANDING_OUTCOME LIKE '%Success%' GROUP BY DATE ORDER BY COUNT(LANDING__OUTCOME) DESC;
```

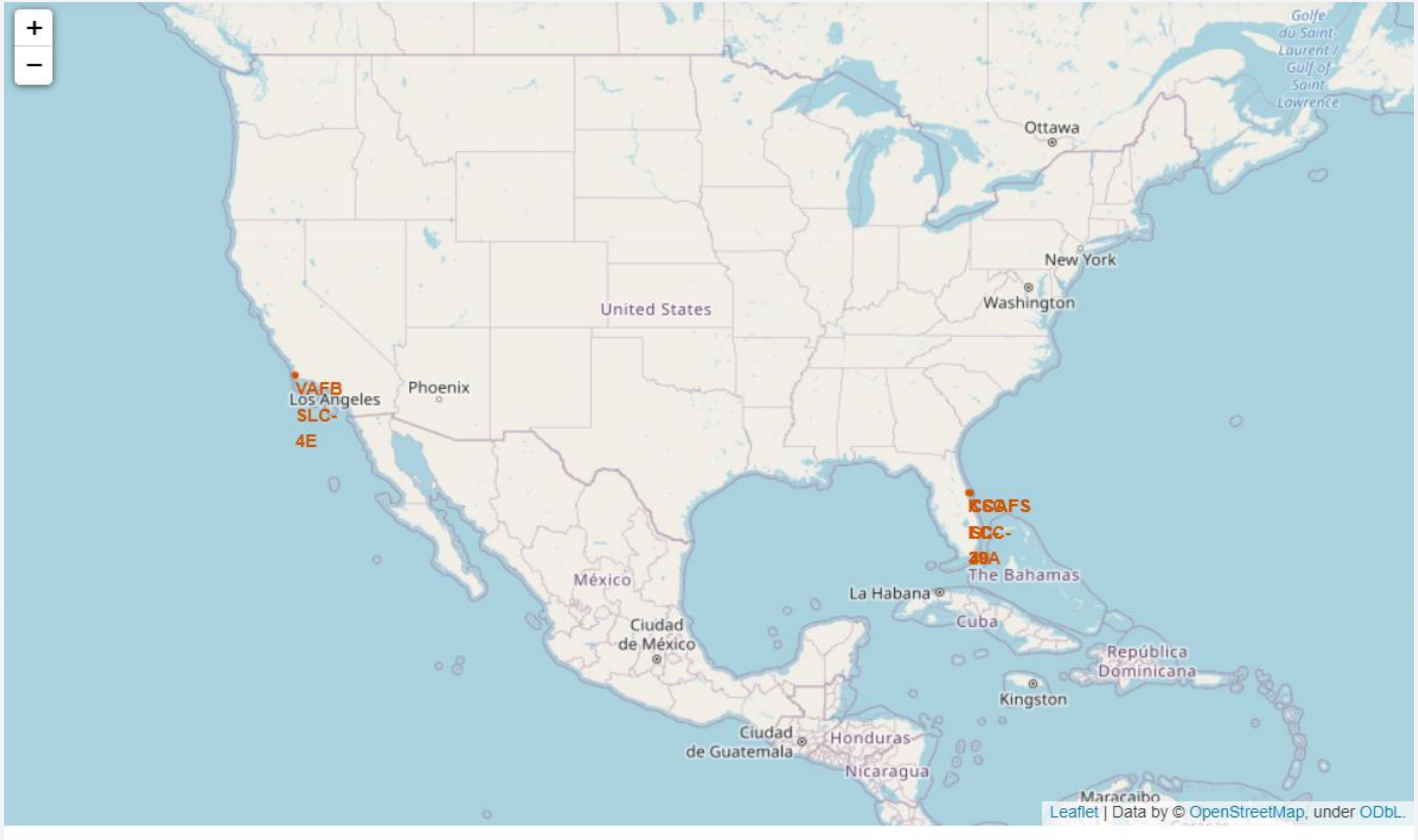
```
 * sqlite:///my_data1.db
```

Selecting landing outcome and the COUNT of landing outcome WHERE landing outcome BETWEEN 04-06-2010 AND 20-03-2017, by GROUP BY AND ORDER BY landing outcome in DESC order.

Section 3

Launch Sites Proximities Analysis

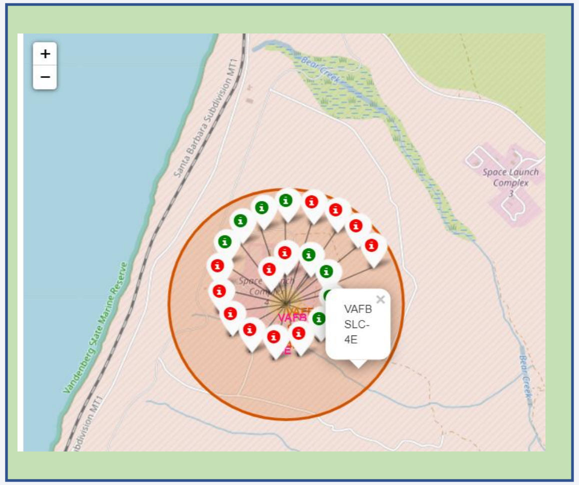
Location of Launch Sites



Display of launch locations in USA

Map Showing Launch Outcomes

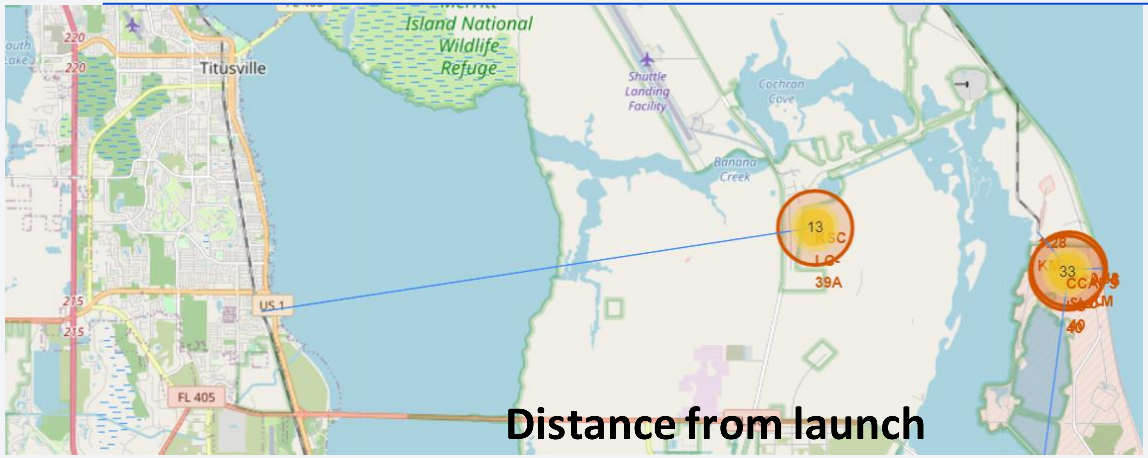
Florida Launch Site



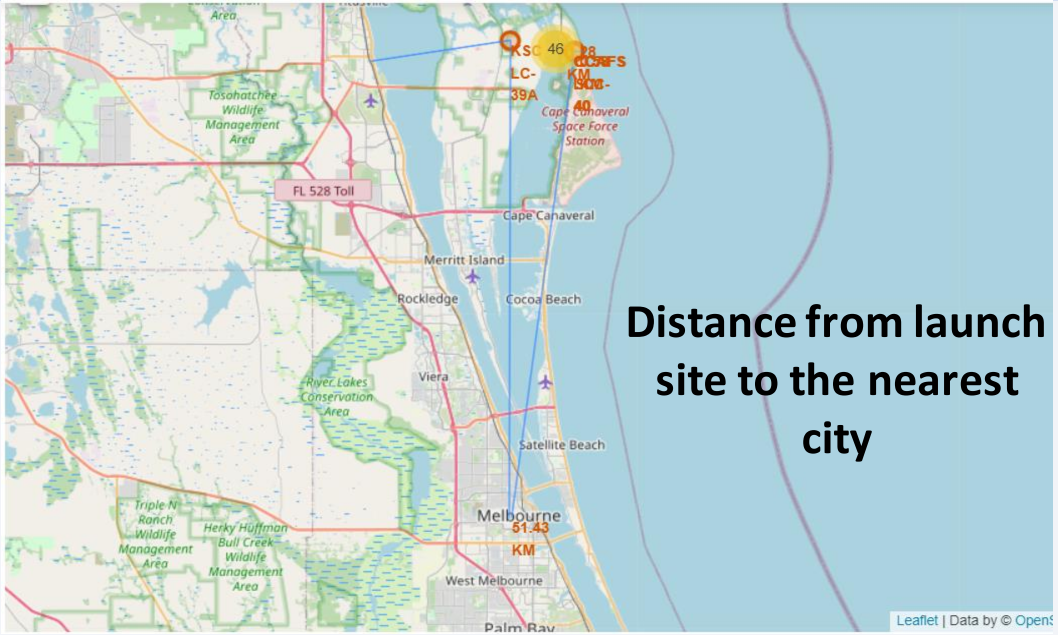
California Launch Site

- ✓ Green Marker — Launch Success
- ✓ Red Marker — Launch Failure

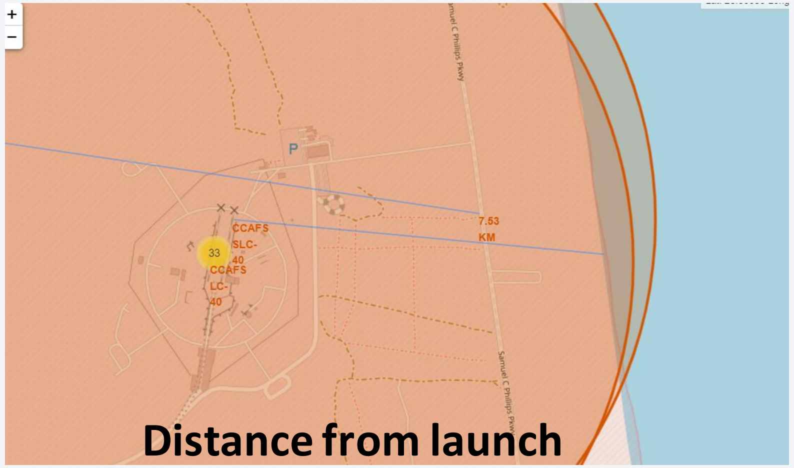
Launch Site Proximity to Landmarks



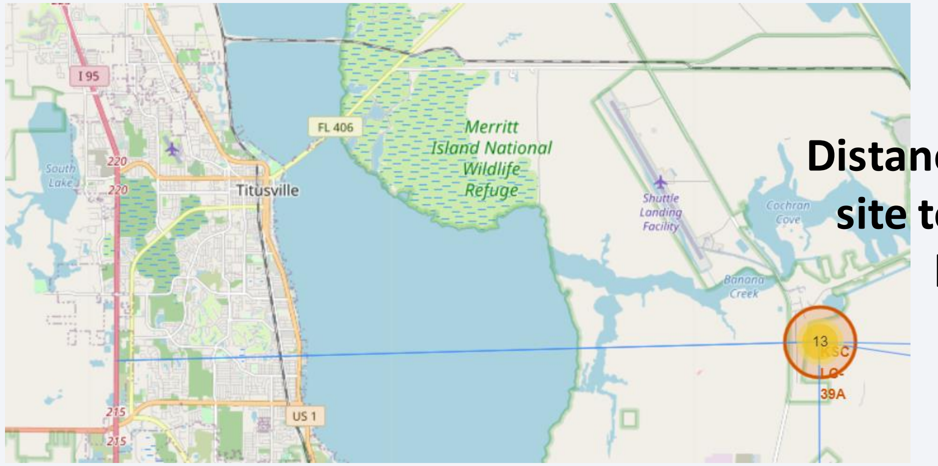
Distance from launch site to the nearest railway



Distance from launch site to the nearest city



Distance from launch site to the nearest coastline



Distance from launch site to the nearest highway

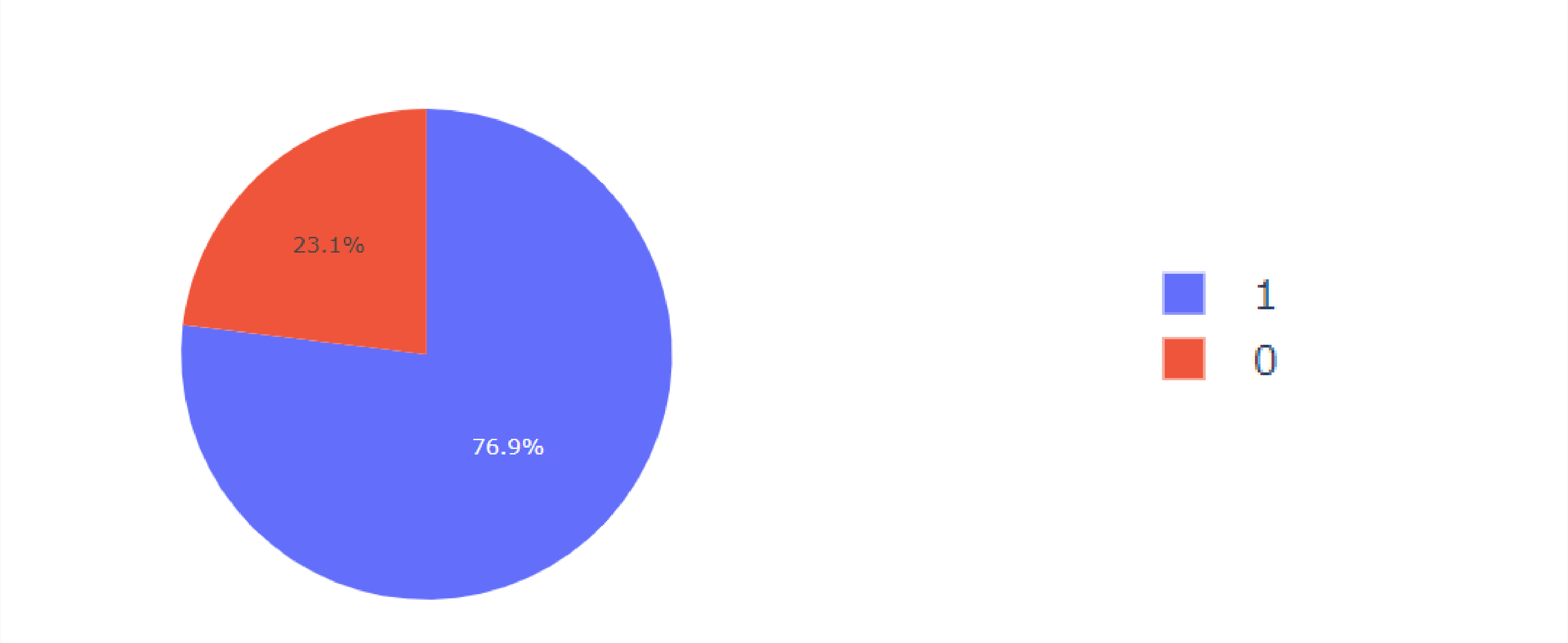
Section 4

Build a Dashboard with Plotly Dash

Success Percentage For All Sites

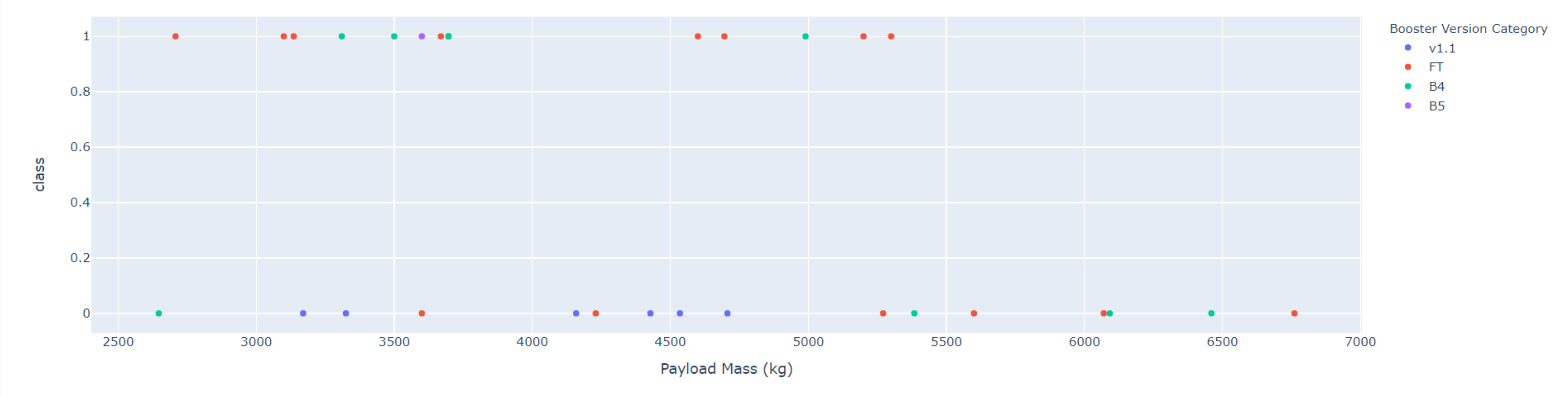
KSC LC-39A had most successful launches and CCAFS SLC-40 had least launches.

Success Ratio for KSC LC-39A

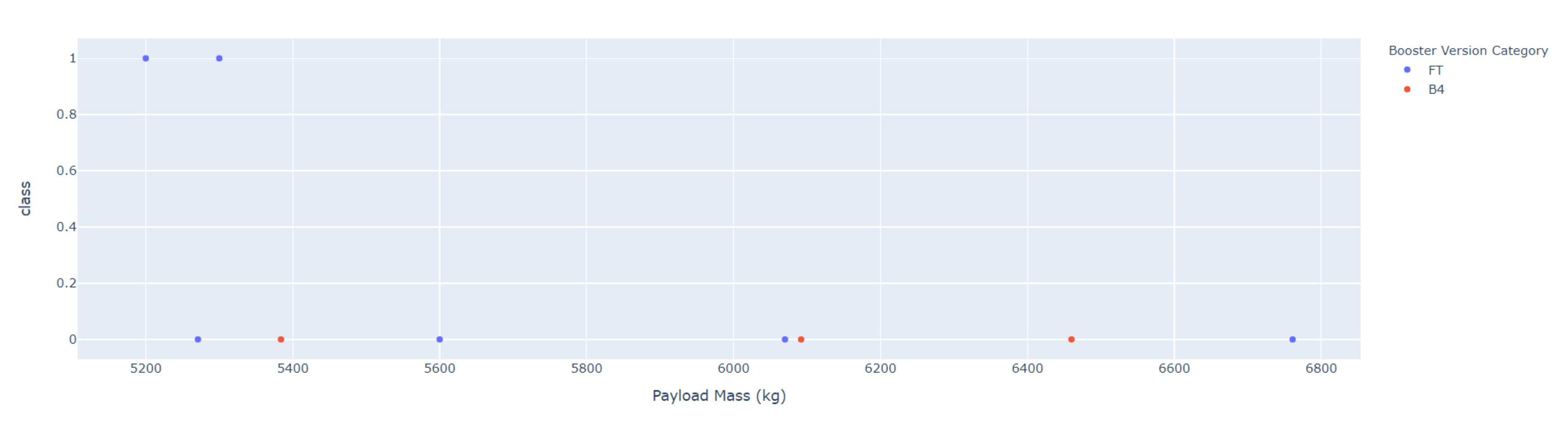


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate.

Payload vs Launch Outcome Scatter Plot



Low Weighted Payload 2500—7500kg



Heavy Weighted Payload 5000—7500kg

Better success rate for low weighted payload is higher than heavy weighted payload.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

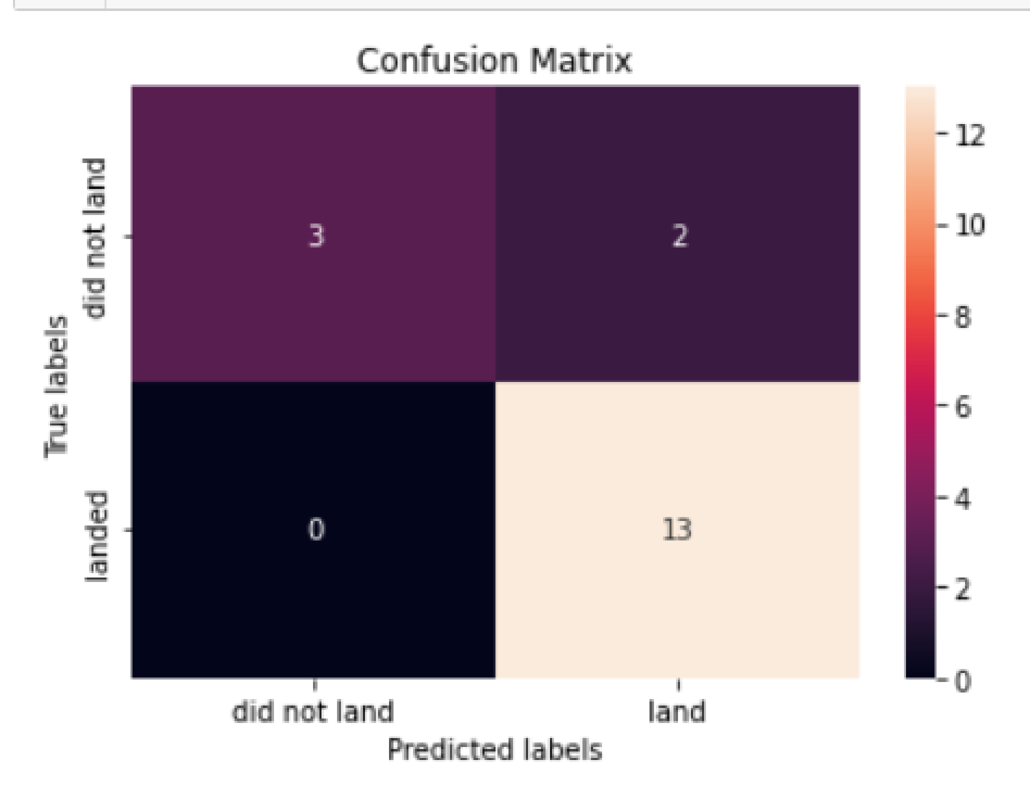
Find the method performs best:

```
algorithms = {'KNN':knn_cv.best_score_, 'Tree':tree_cv.best_score_, 'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm, 'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is : {'criterion': 'gini', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 5, 'splitter': 'best'}
```

From the code, Tree is the best algorithm with the highest classification accuracy.

Confusion Matrix



The false-positive value 2, unsuccessful landing marked as successful landing by the classifier.

Conclusions

- The best algorithm for the Machine Learning is the Tree Classifier.
- There are a lot of success outcomes and better performance when the payloads are below 5000kg.
- SSO orbit had 100% success rate and the payloads were below 5000kg.
- From the study, the result of success rate is positive as the years increases from 2013. There is a highly positive projection in the future.
- The launch site that most success rate is KSC LC-39A.

Thank you!