IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sam Frost
June 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection
  - Data Wrangling
  - Exploratory Data Analysis with Data Visualization
  - Exploratory Data Analysis 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 in screenshots
  - Predictive analysis results

Introduction

Project background and context

This project is to predict if the Falcon 9 first stage will land successfully. SpaceX offers a Falcon 9 rocket launch as low as 62 million dollars; other providers cost upward of 165 million dollars each which is a huge cost saving. Much of of the cost savings is because SpaceX can reuse the first stage. The goal of this project is to use machine learning to predict the landing outcome of the first stage. In turn, this will enable us to determine the cost of a launch along with a right price an alternate company should seek to bid against SpaceX for a rocket launch

Problems you want to find answers

- What are the factors that influence the landing outcome?
- What are the relationship of certain rockets variables and how they will impact the landing outcome?
- What are the best conditions needed to increase the probability of a successful landing?

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
- Perform exploratory data analysis (EDA) using visualization and SQ
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- One of the first steps in the Machine Learning Lifecycle is to identify what data is needed. Then evaluate the various means available for collecting this data.

  REST API - We use the get request and decode the response content as JSON format and use json_normalize() which turns the data into a pandas dataframe. Once this is done we clean the data, checking for missing values and fill where required.

  Web scrapping – We use BeautifulSoup and extract the records as HTML table and convert it to Pandas dataframe for further analysis.

- You need to present your data collection process use key phrases and flowcharts

Data Collection – SpaceX API

1. Get Response from API
2. Converting Response using json_normalize method to convert into a pandas dataframe
3. Perform data cleaning and fill missing value

[Github Link]

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

```
response = requests.get(spacex_url)
```

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

```
# 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 sing
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)]
```

Data Collection - Scraping

1. Get Response from HTML
2. Use BeautifulSoup to create a BeautifulSoup object from response
3. Extract data from the table

[Github Link]

```
# use requests.get() method with the provided static_url
# assign the response to a object

data = requests.get(static_url).text
```

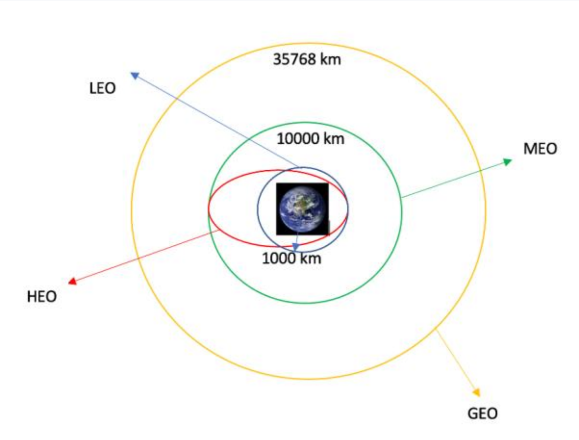
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data,'html.parser')
```

```
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
```

Data Wrangling

Data Wrangling is a process of cleaning messy and complex data sets ready for Exploratory Data Analysis (EDA).

First step is to calculate the number of launches per site, then calculate the amount of mission outcome per orbit type, labeling a landing outcome from the outcome column and export the result to a CSV



Github Link

EDA with Data Visualization

- Scatter Graphs being drawn:

  Flight Number VS. Payload Mass

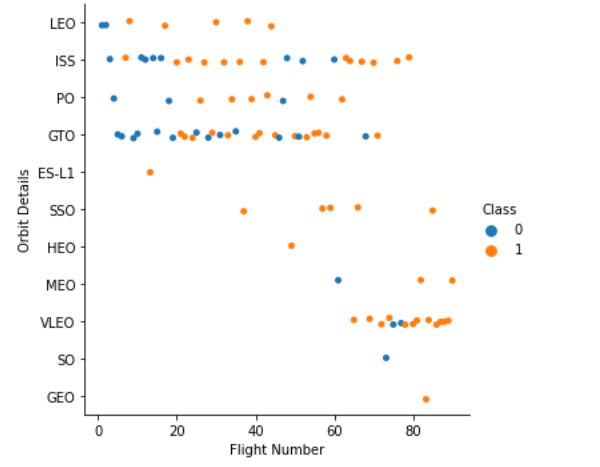
  Flight Number VS. Launch Site

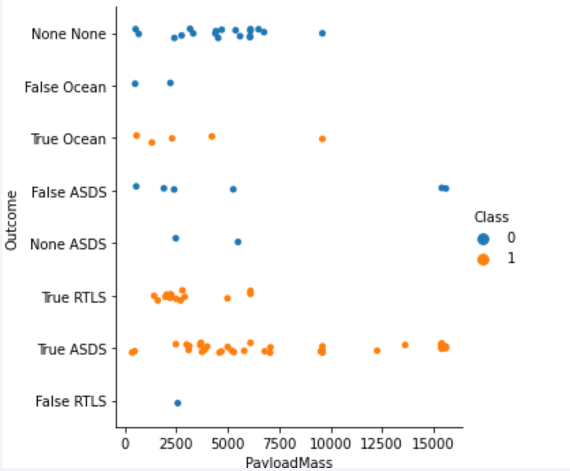
  Payload VS. Launch Site

  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.

EDA with Data Visualization

Bar Graph being drawn:

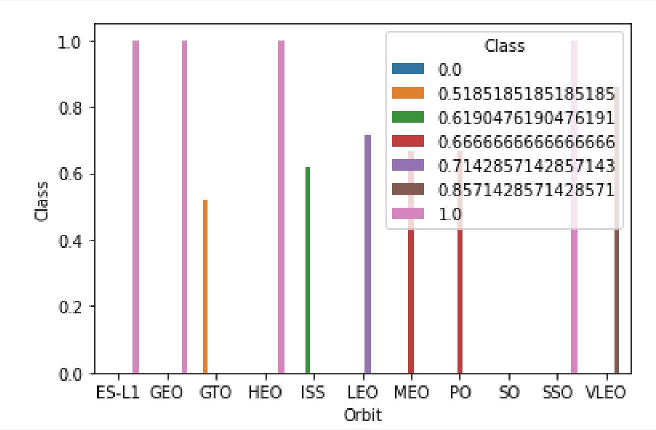
Mean VS. Orbit

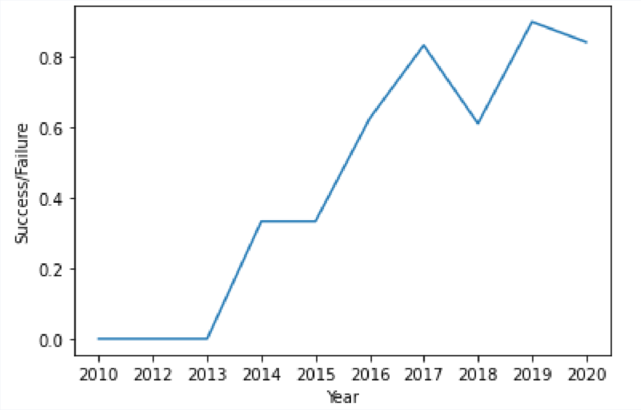
Line Graph being drawn:

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

[Github Link]





EDA with SQL

- 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.

Github Link

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().

We then used the Haversine's formula to calculated the distance of the launch sites to landmarks

[Github Link]

Build a Dashboard with Plotly Dash

Using Plotly Dash we created an interactive dashboard app. It allows users to play around with the date.

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

Created a plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

Github Link

Predictive Analysis (Classification)

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.

Github Link

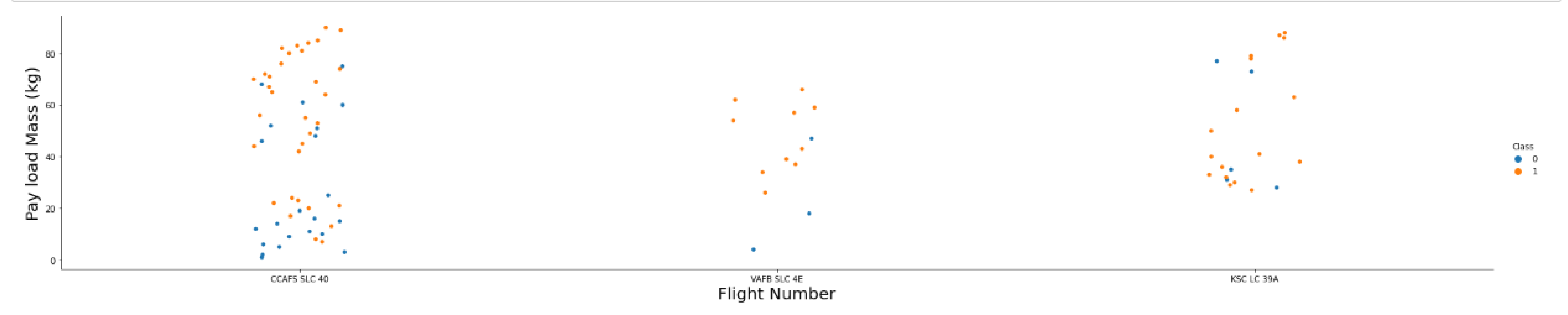
Results

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

Section 2

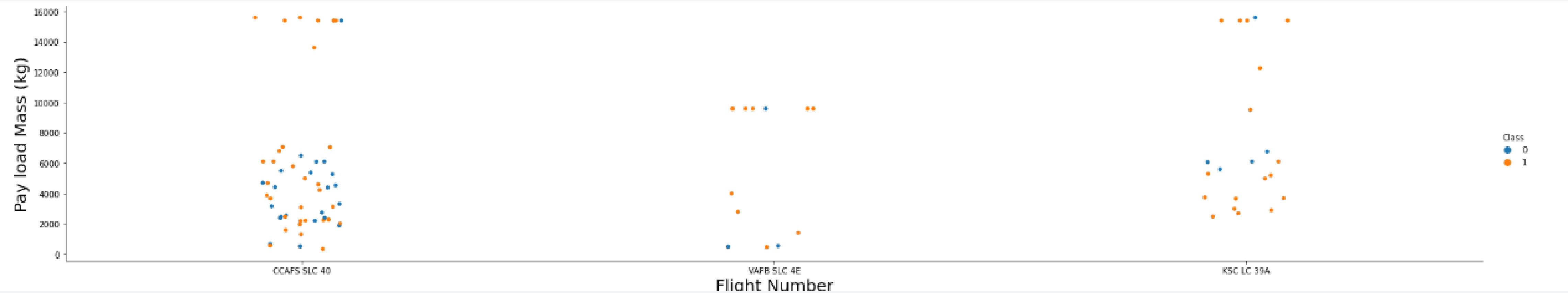
Insights drawn from EDA

Flight Number vs. Launch Site



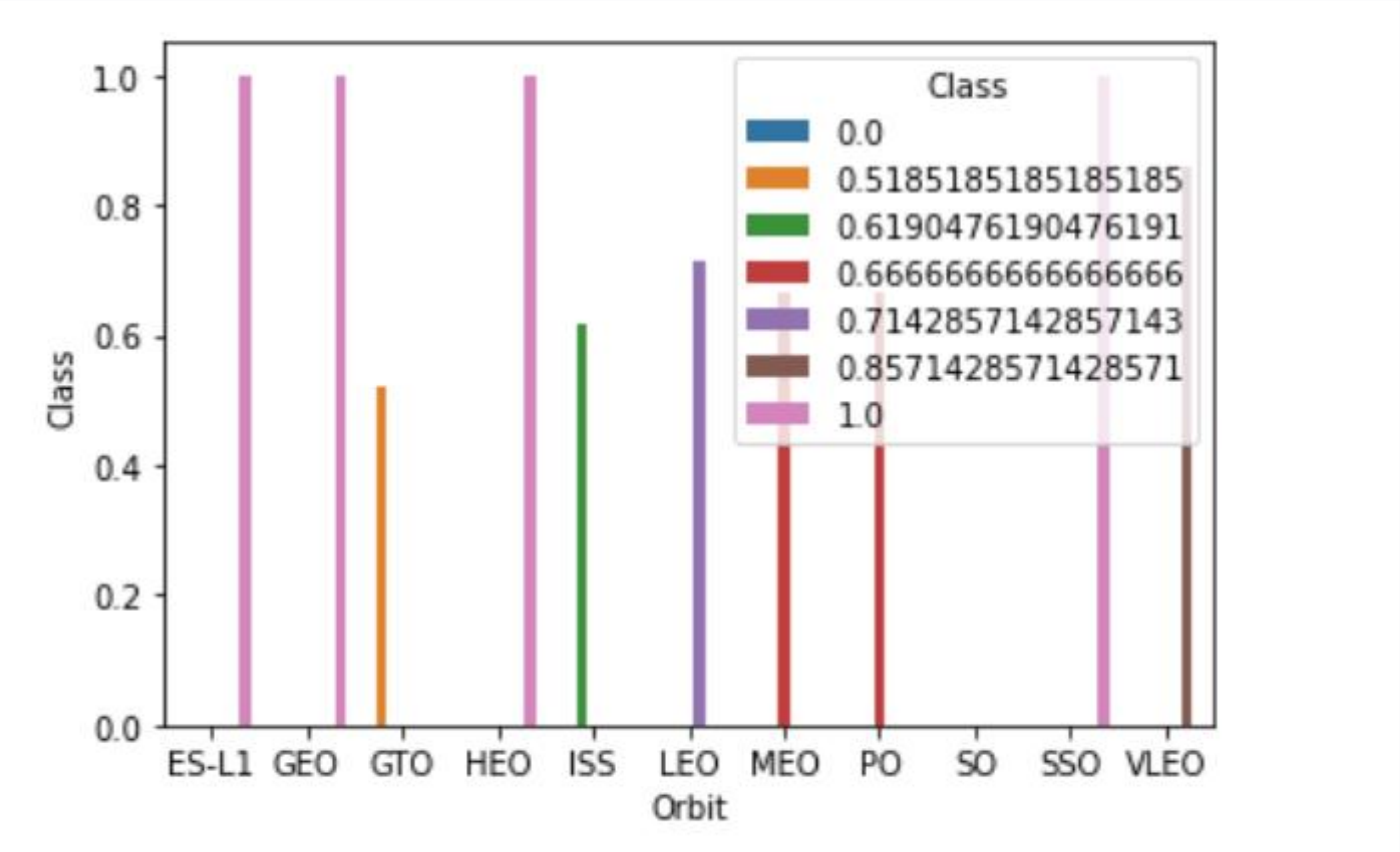
- The graph shows the more amount of flights at a launch site the greater the success rate at a launch site.

Payload vs. Launch Site

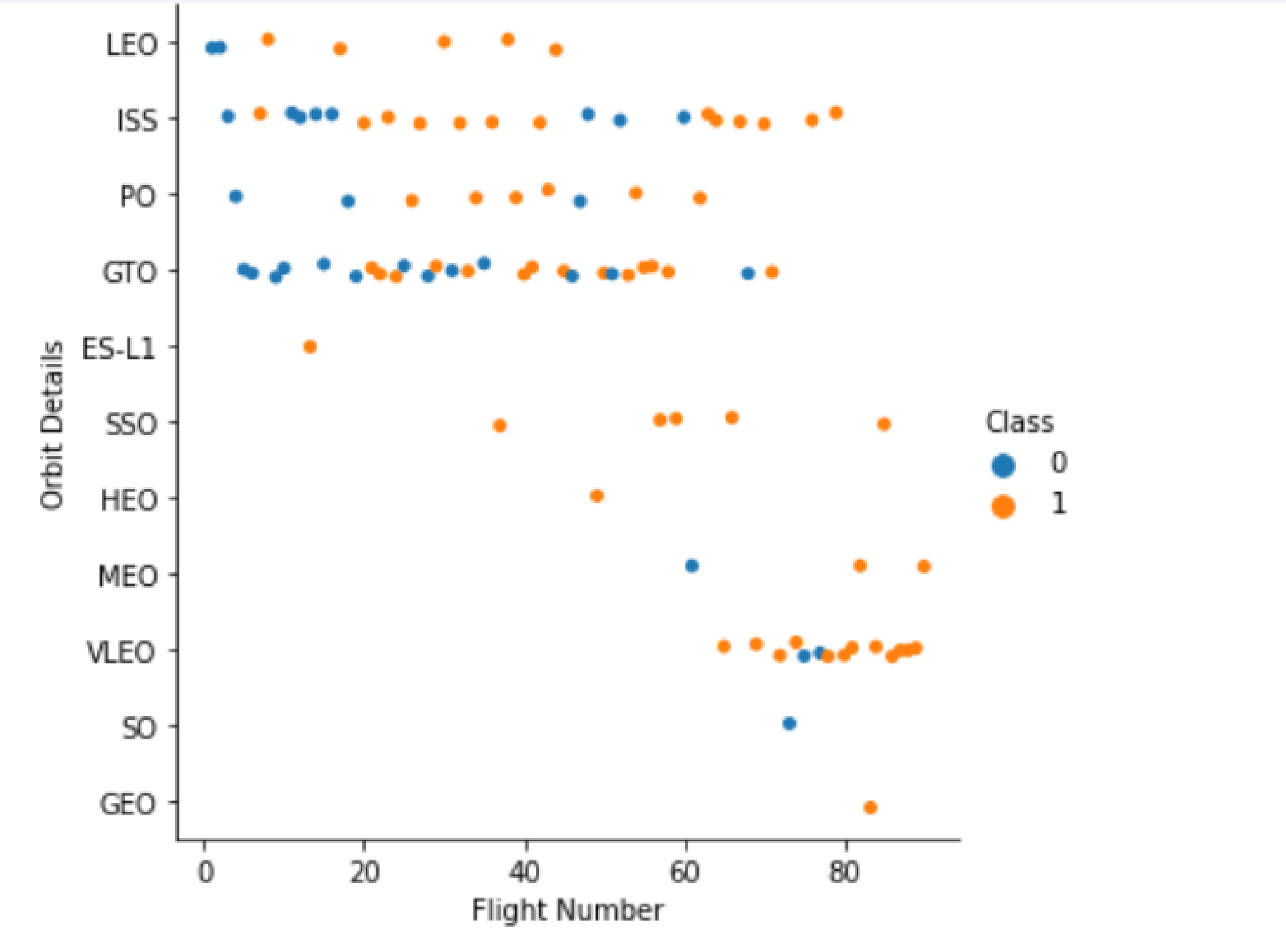


The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket.

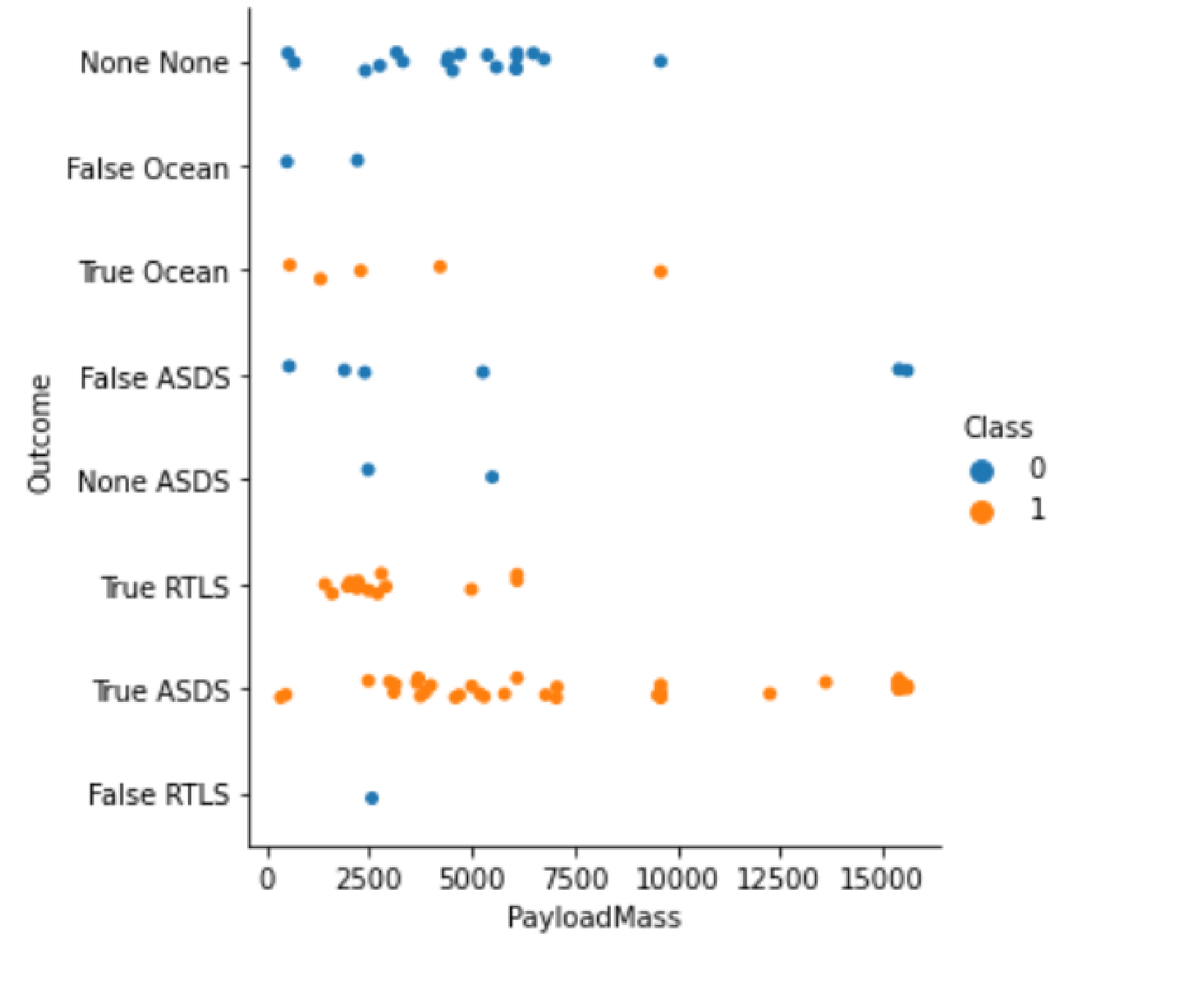
Success Rate vs. Orbit Type



Flight Number vs. Orbit Type

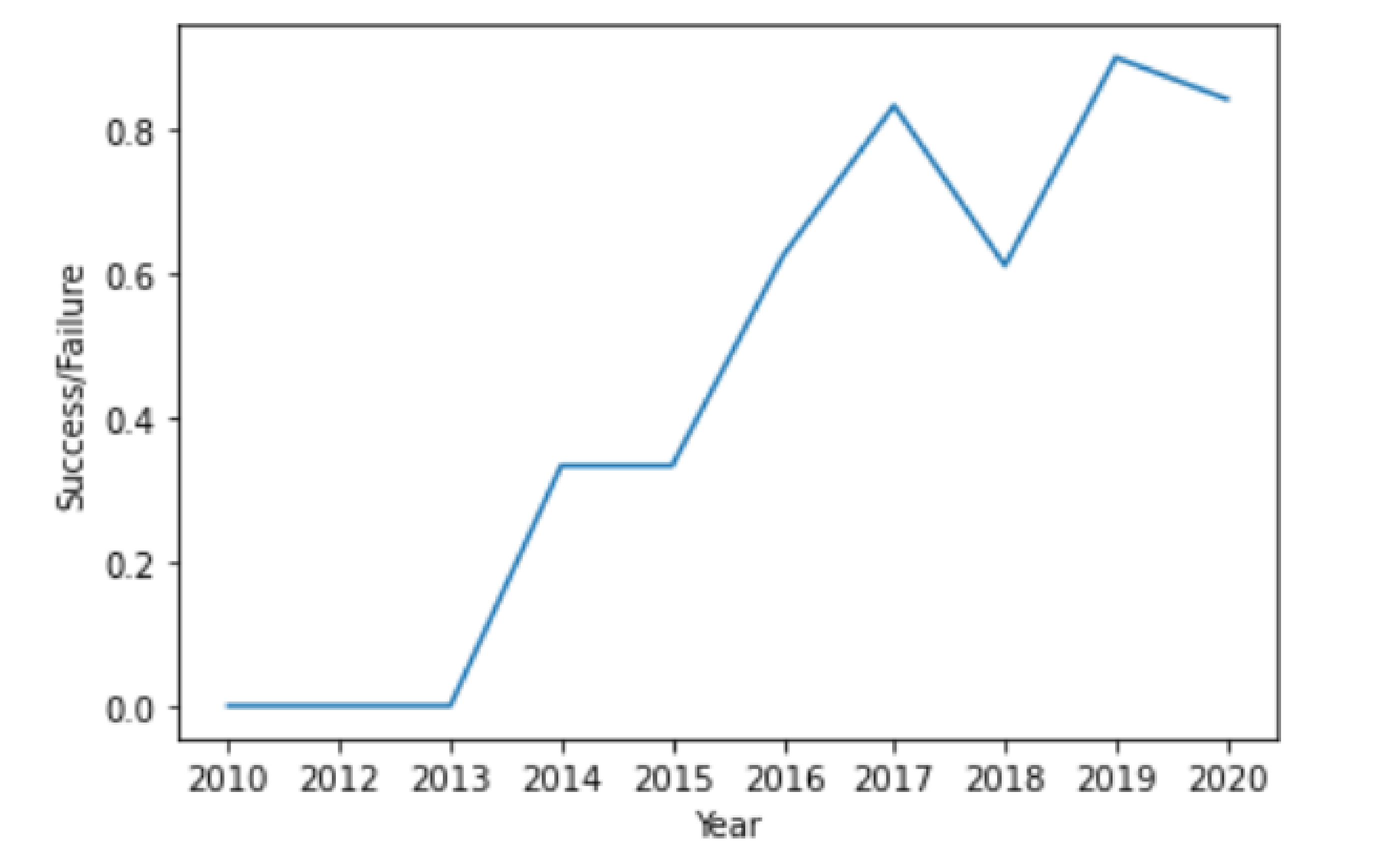


Payload vs. Orbit Type



Launch Success Yearly Trend

As we can see success rate since 2013 kept increasing till 2020



All Launch Site Names

We used the following SQL query to get all launch site names:

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXDATASET;
```

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

Launch Site Names Begin with 'KSC'

We use the following query to gather 5 sites beginning with KSC

```
%sql SELECT LAUNCH_SITE FROM SPACEXDATASET WHERE LAUNCH_SITE LIKE 'KSC%' LIMIT 5;
```

launch_site
KSC LC-39A
KSC LC-39A
KSC LC-39A
KSC LC-39A
KSC LC-39A

Total Payload Mass

We Calculate the total payload carried by boosters from NASA with the following Query:

```
%sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXDATASET;
```

payloadmass
619967

Average Payload Mass by F9 v1.1

Average payload mass carried by booster version F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXDATASET;
```

payloadmass
6138

First Successful Ground Landing Date

We get the first successful date by using the min date

```
%sql select min(DATE) from SPACEXDATASET;
```

1
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select BOOSTER_VERSION from SPACEXDATASET where LANDING__OUTCOME='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

```
%sql select count(MISSION_OUTCOME) as missionoutcomes from SPACEXDATASET GROUP BY MISSION_OUTCOME;
```

missionoutcomes
1
99
1

Boosters Carried Maximum Payload

```
%sql select BOOSTER_VERSION as boosterversion from SPACEXDATASET where PAYLOAD_MASS__KG_=(select max(P
AYLOAD_MASS__KG_) from SPACEXDATASET);
```

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

2015 Launch Records

```
%sql SELECT MONTH(DATE),MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXDATASET where EXTRACT(YEAR FROM DATE)='2017' AND \
MISSION_OUTCOME = 'Success';
```

Here the SQL query will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017

1	mission_outcome	booster_version	launch_site
1	Success	F9 FT B1029.1	VAFB SLC-4E
2	Success	F9 FT B1031.1	KSC LC-39A
3	Success	F9 FT B1030	KSC LC-39A
3	Success	F9 FT B1021.2	KSC LC-39A
5	Success	F9 FT B1032.1	KSC LC-39A
5	Success	F9 FT B1034	KSC LC-39A
6	Success	F9 FT B1035.1	KSC LC-39A
6	Success	F9 FT B1029.2	KSC LC-39A
6	Success	F9 FT B1036.1	VAFB SLC-4E
7	Success	F9 FT B1037	KSC LC-39A
8	Success	F9 B4 B1039.1	KSC LC-39A
8	Success	F9 FT B1038.1	VAFB SLC-4E
9	Success	F9 B4 B1040.1	KSC LC-39A
10	Success	F9 B4 B1041.1	VAFB SLC-4E
10	Success	F9 FT B1031.2	KSC LC-39A
10	Success	F9 B4 B1042.1	KSC LC-39A
12	Success	F9 FT B1035.2	CCAFS SLC-40
12	Success	F9 FT B1036.2	VAFB SLC-4E

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

```
%sql SELECT LANDING__OUTCOME FROM SPACEXDATASET WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC;
```

This query will rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order

landing__outcome
No attempt
Success (ground pad)
Success (drone ship)
Success (drone ship)
Success (ground pad)
Failure (drone ship)
Success (drone ship)
Success (drone ship)
Success (drone ship)
Failure (drone ship)
Failure (drone ship)
Success (ground pad)
Precluded (drone ship)
No attempt
Failure (drone ship)
No attempt
Controlled (ocean)
Failure (drone ship)
Uncontrolled (ocean)
No attempt
No attempt
Controlled (ocean)
Controlled (ocean)
No attempt
No attempt
Uncontrolled (ocean)
No attempt
No attempt
No attempt
Failure (parachute)
Failure (parachute)

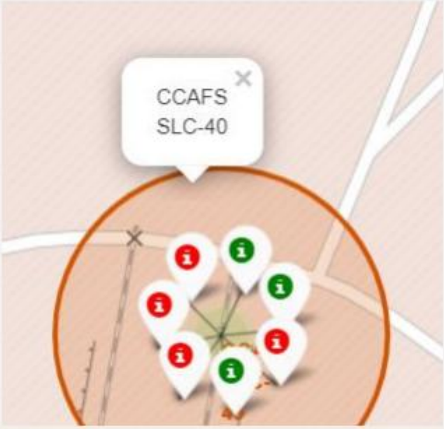
Section 3

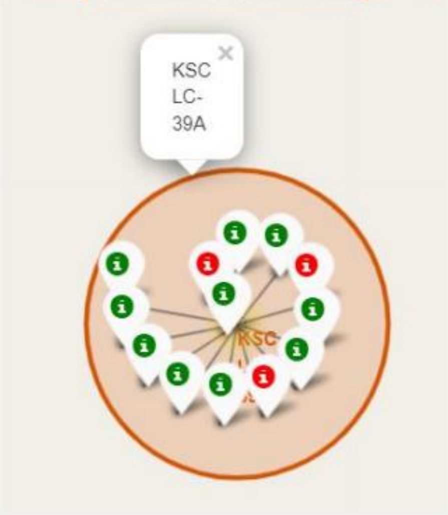
Launch Sites Proximities Analysis

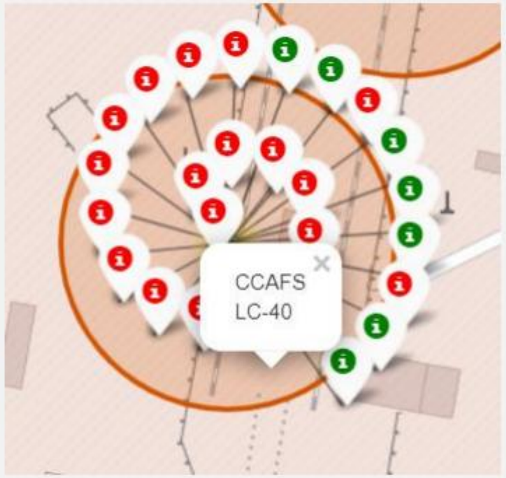
Location of Launch Sites

Our Map shows the locations of the launch sites in USA

Colour Labelled Markers





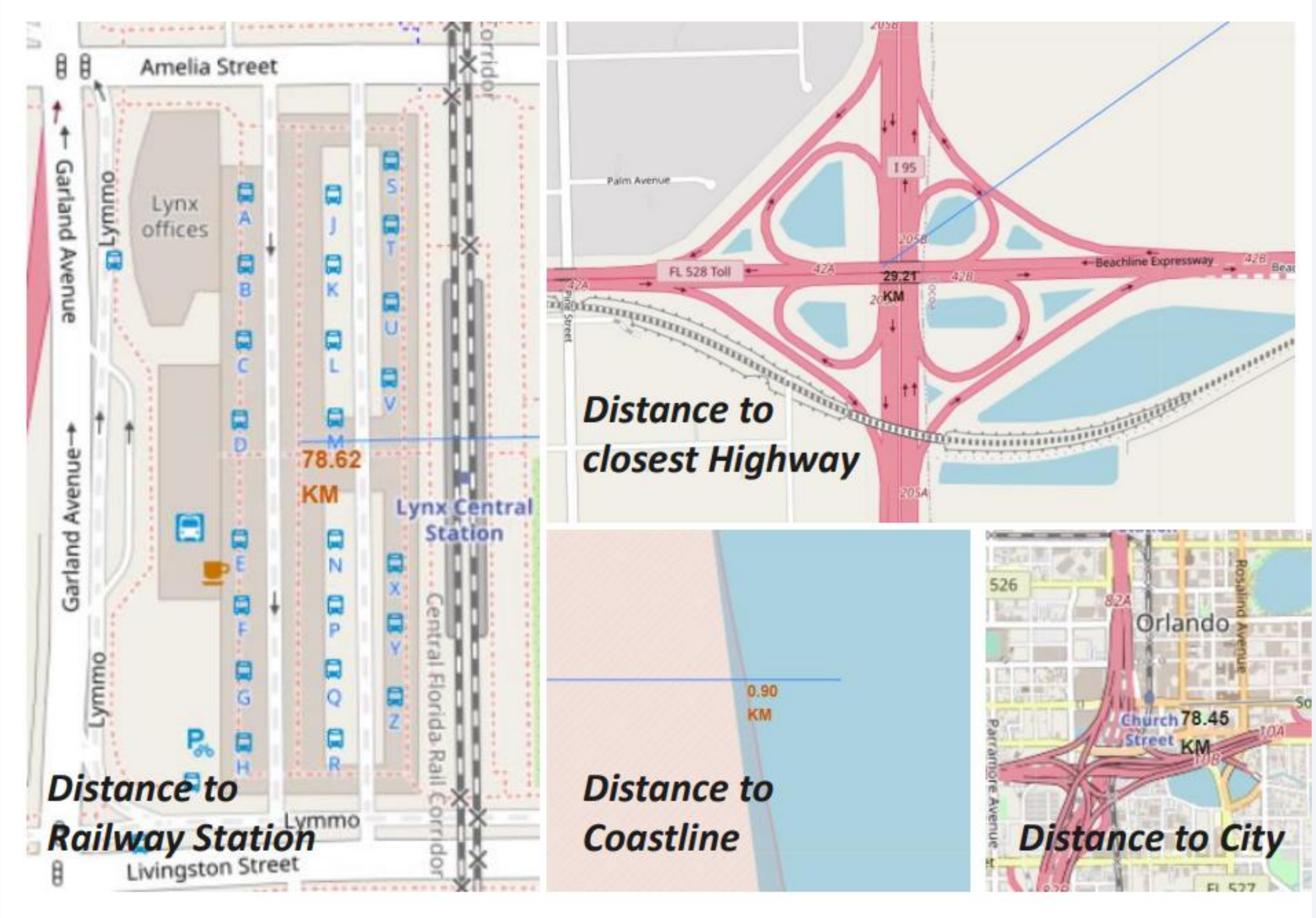


California Launch Site

Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

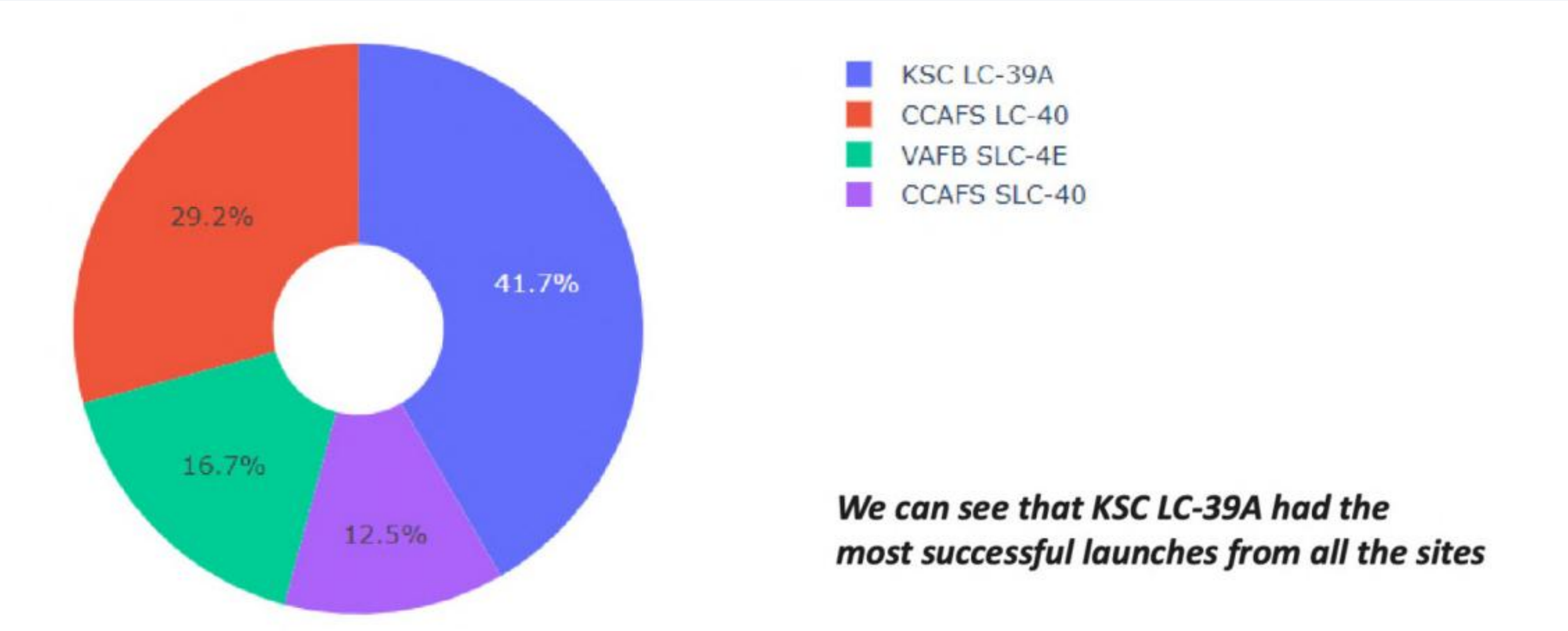
Launch Sites Distance from Landmarks



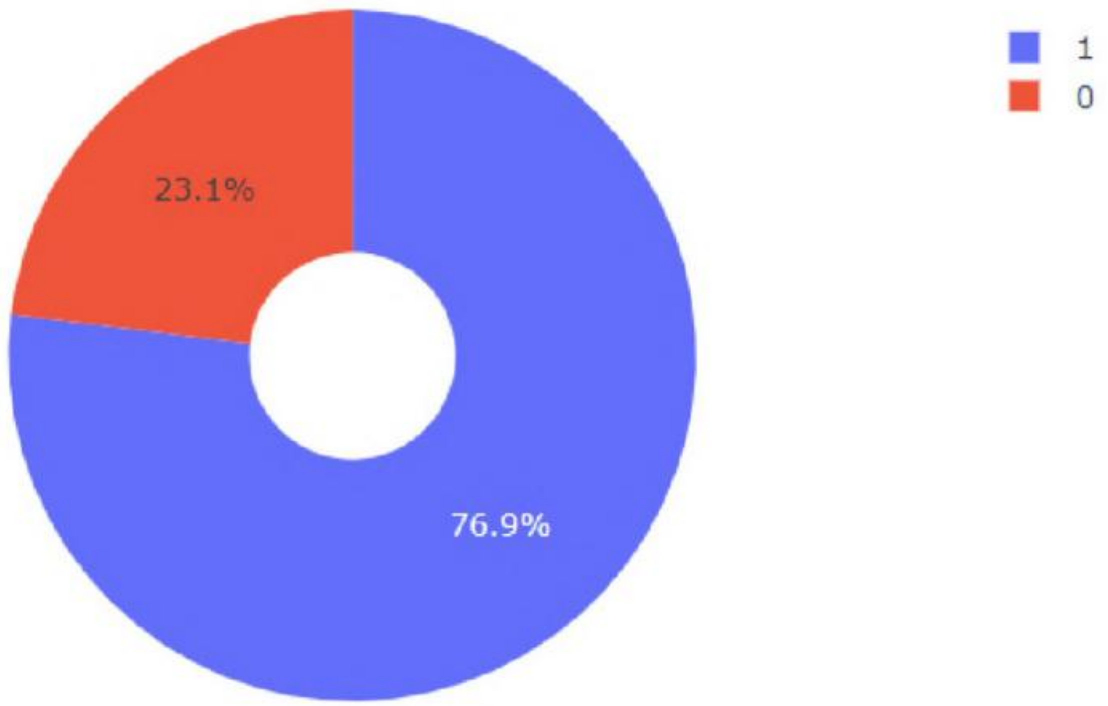
Section 4

Build a Dashboard with Plotly Dash

Total Success Launches % per site

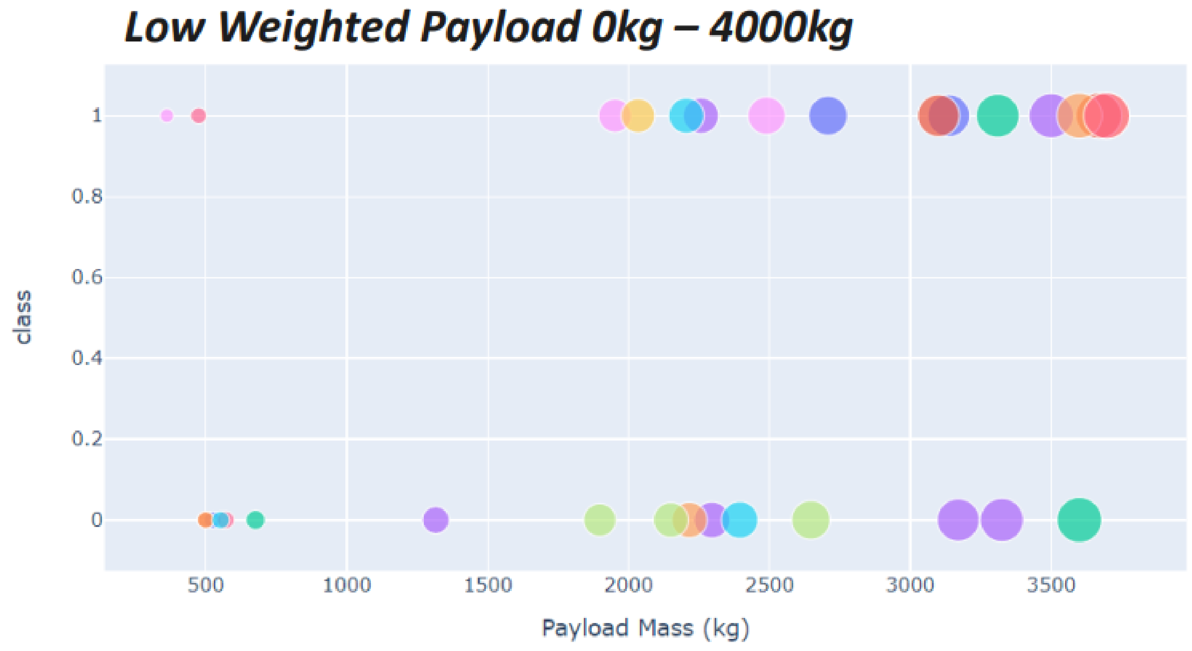


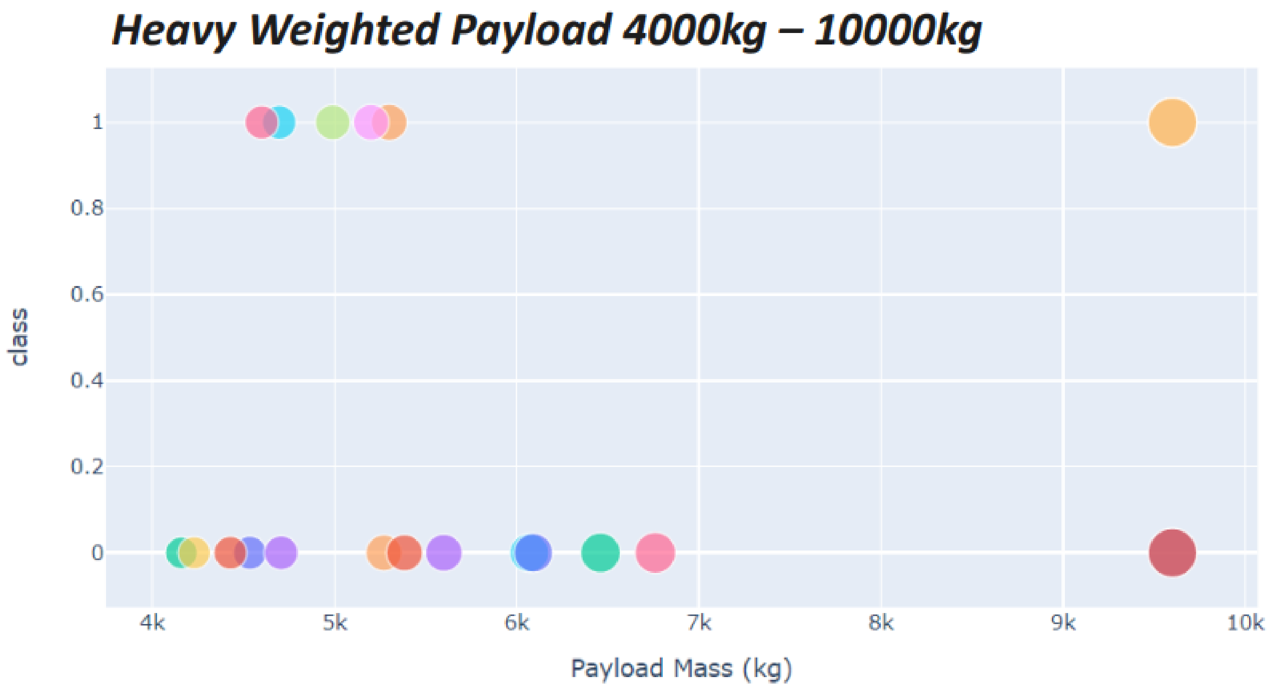
Site with highest launch success



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

Payload vs. Launch Outcome Scatter Plott for all sites





Section 5

Predictive Analysis (Classification)

Classification Accuracy

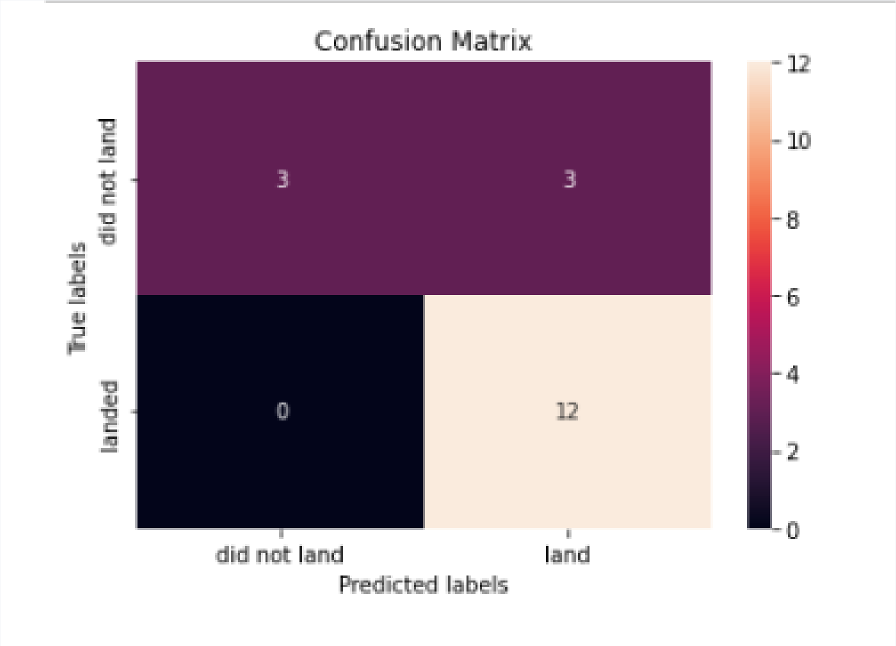
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params 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 0.875
Best Params is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 2, 's
plitter': 'random'}
```

We can see from the above code that we were able to identify that the best algorithm to be the Tree Algorithm which have the highest classification accuracy.

Confusion Matrix

Examining the confusion matrix, we see that Tree can distinguish between the different classes.



Conclusions

So to conclude:

- The Tree Classifier Algorithm is the best Machine Learning approach for this dataset.
- The low weighted payloads (which define as 4000kg and below) performed better than the heavy weighted payloads.
- The success rate for SpaceX launches is increased, directly proportional time in years to 2020, which it will eventually perfect the launches in the future.
- KSC LC-39A have the most successful launches of any sites;

Thank you!