IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

DWARIKA PRASAD DASH
25th Jan 2025

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- **Summary of methodologies**
  - ❑ SpaceX Data Collection using SpaceX API
  - ❑ SpaceX Data Collection with Web Scraping
  - ❑ SpaceX Data Wrangling
  - ❑ SpaceX Exploratory Data Analysis using SQL
  - ❑ Space-X EDA DataViz Using Python Pandas and Matplotlib
  - ❑ Space-X Launch Sites Analysis with Folium-Interactive Visual Analytics and Ploty Dash
  - ❑ SpaceX Machine Learning Landing Prediction
- **Summary of all results**
  - ❑ EDA results
  - ❑ Interactive Visual Analytics and Dashboards
  - ❑ Predictive Analysis(Classification)

Introduction

- **Project background and context**
  - 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
- **Problems you want to find answers**
  - In this capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - ❑ Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - ❑ Data was processed using one-hot encoding for categorical features
- 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

- ❑ Data collection is the process of gathering and measuring information on targeted variables in an established system, which then enables one to answer relevant questions and evaluate outcomes. As mentioned, the dataset was collected by REST API and Web Scrapping from Wikipedia.
- ❑ For REST API, its started by using the get request. Then, we decoded the response content as Json and turn it into a pandas dataframe using json.normalize(). We then cleaned the data, checked for missing values and fill with whatever needed.
- ❑ For web scrapping, we will use the BeautifulSoup to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for further analysis

Data Collection – SpaceX API

GET request

Create Data Frame

Data Cleaning

https://github.com/DashDwarika/SpaceX.git

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

```
# 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)]
```

Data Collection - Scraping

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

GET request

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

Create Beautiful Soup object from response

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

https://github.com/DashDwarika/SpaceX.git

Data Wrangling

- Data Wrangling is the process of cleaning and unifying messy and complex data sets for easy access and Exploratory Data Analysis (EDA).
- We will first calculate the number of launches on each site, then calculate the number and occurrence of mission outcome per orbit type.
- We then create a landing outcome label from the outcome column. This will make it easier for further analysis, visualization, and ML. Lastly, we will export the result to a CSV.

https://github.com/DashDwarika/SpaceX.git

EDA with Data Visualization

- We first started by using scatter graph to find the relationship between the attributes such as between:
  - ❑ Payload and Flight Number.
  - ❑ Flight Number and Launch Site.
  - ❑ Payload and Launch Site.
  - ❑ Flight Number and Orbit Type.
  - ❑ Payload and Orbit Type.
- Scatter plots show dependency of attributes on each other. Once a pattern is determined from the graphs. It’s very easy to see which factors affecting the most to the success of the landing outcomes.

https://github.com/DashDwarika/SpaceX.git

EDA with SQL

- Using SQL, we had performed many queries to get better understanding of the dataset, Ex:
  - ❑ Displaying the names of the unique launch sites.
  - ❑ Displaying 5 records where launch sites begin with the string ‘CCA’.
  - ❑ Displaying the total payload mass carried by boosters launched by NASA (CRS).
  - ❑ Displaying the average payload mass carried by booster version F9 v1.1.
  - ❑ Listing the date when the first successful landing outcome in ground pad was achieved.
  - ❑ Listing the names of the boosters which have success in drone ship 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 failed landing outcomes in drone ship, their booster versions, and launch sites names for in year 2015.
  - ❑ Rank the count of landing outcomes failure or success between the date 2010-06-04 and 2017-03-20, in descending order.

https://github.com/DashDwarika/SpaceX.git

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 then assigned the dataframe launch outcomes(failure,success) to classes 0 and 1 with Red and Green markers on the map in MarkerCluster().
- We then used the Haversine's formula to calculated the distance of the launch sites to various landmark to find answer to the questions of:
    - How close the launch sites with railways, highways and coastlines?
    - How close the launch sites with nearby cities?

https://github.com/DashDwarika/SpaceX.git

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash which allowing the user to play around with the data as they need.
- We plotted pie charts showing the total launches by a certain sites.
- We then plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

https://github.com/DashDwarika/SpaceX.git

Predictive Analysis (Classification)

Building the Model

•Load the dataset into NumPy and Pandas
•Transform the data and then split into training and test datasets
•Decide which type of ML to use
•set the parameters and algorithms to GridSearchCV and fit it to dataset.

Evaluating the model

•Check the accuracy for each model
•Get tuned hyperparameters for each type of algorithms.
•plot the confusion matrix.

Improving the Model

•Use Feature Engineering and Algorithm Tuning

Find the best Model

The model with the best accuracy score will be the best performing model.

https://github.com/DashDwarika/SpaceX

Results

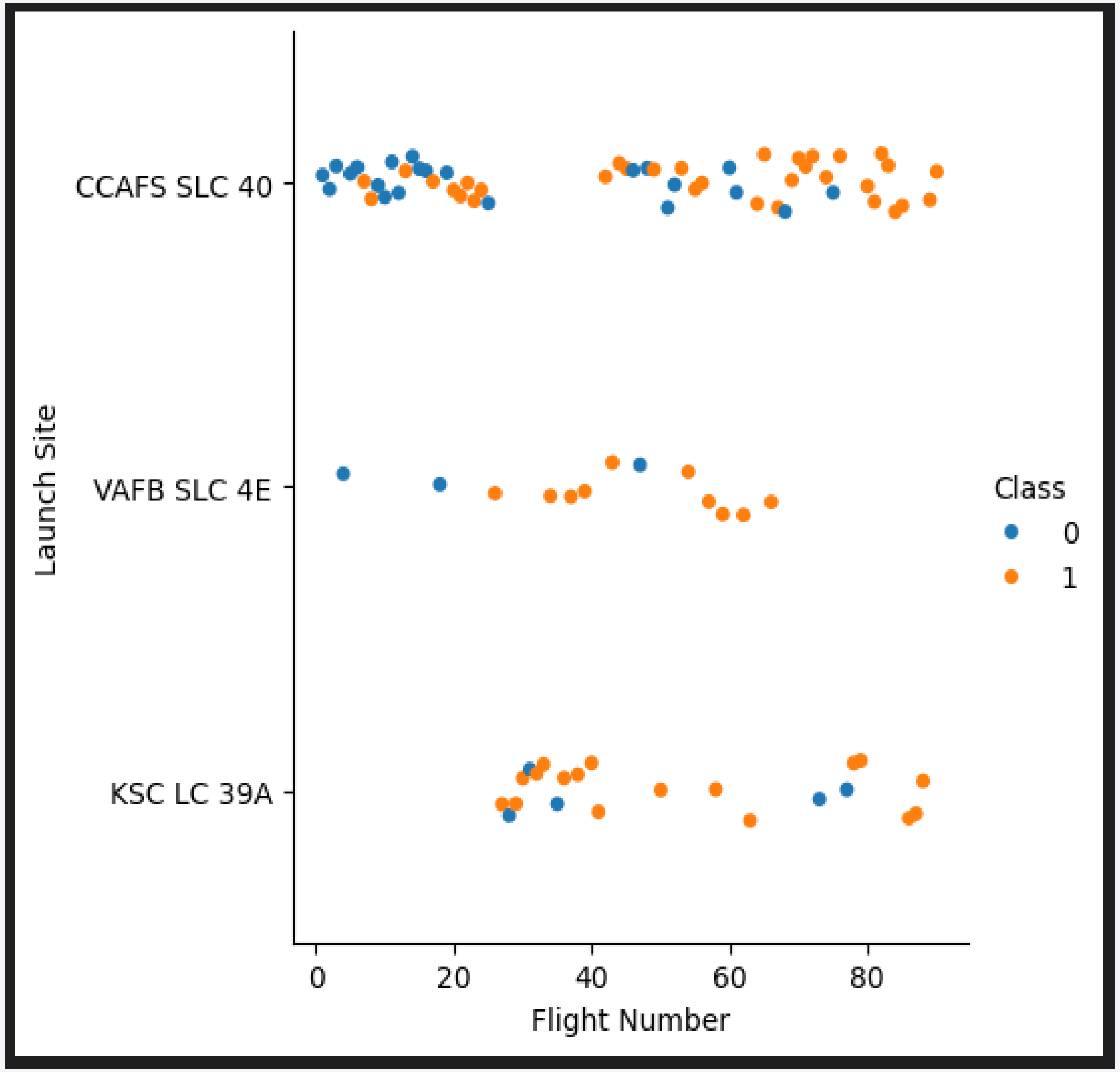
The results will be categorized to 3 main results which is:

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

Section 2

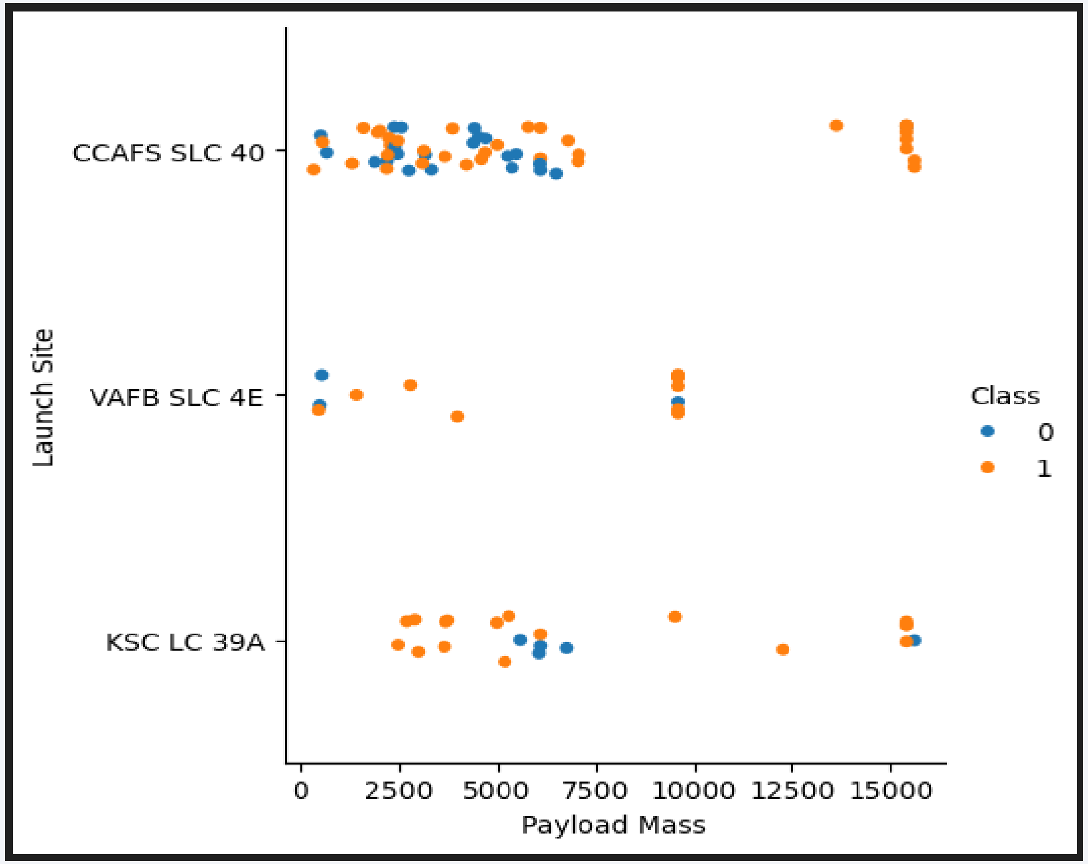
Insights drawn from EDA

Flight Number vs. Launch Site



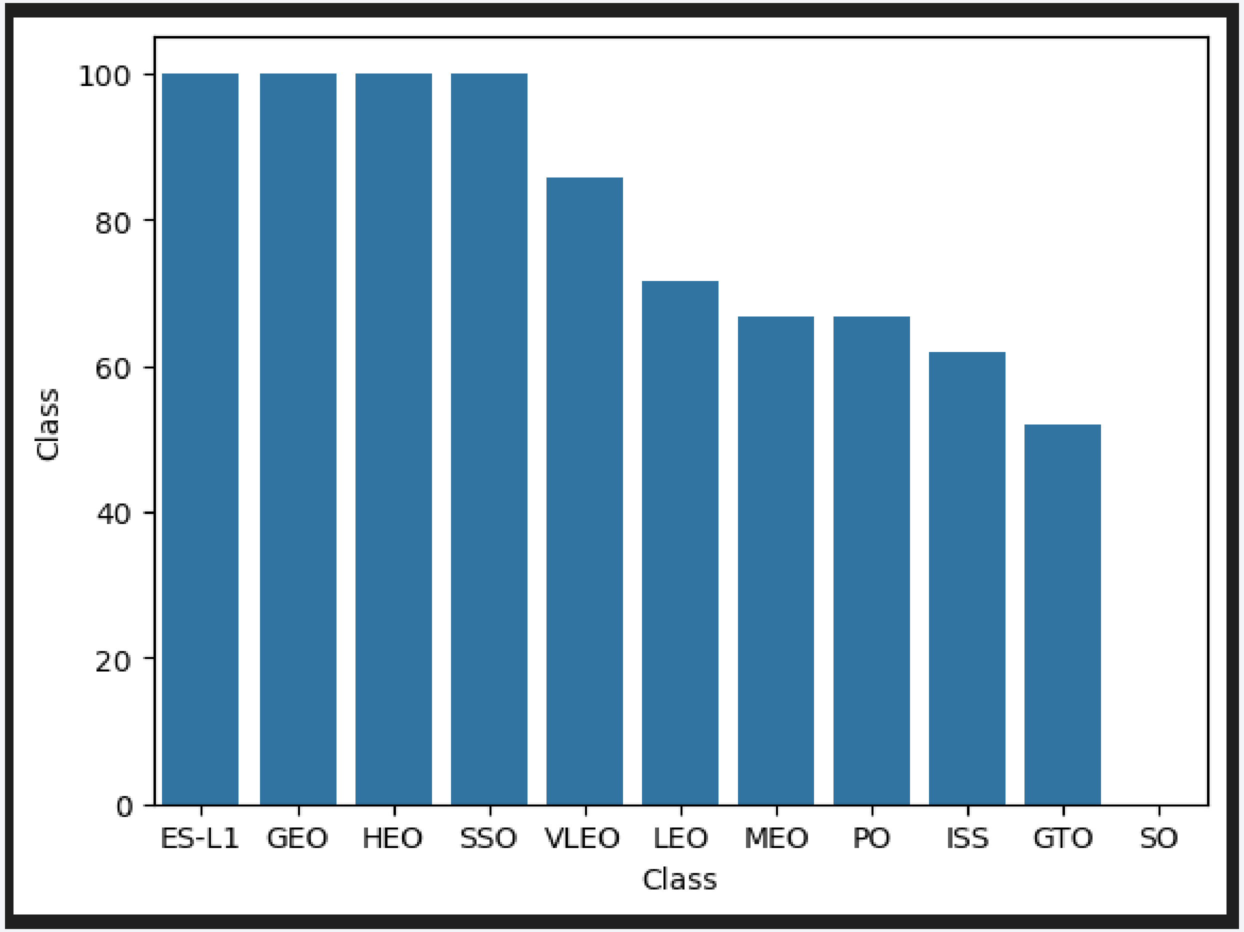
This scatter plot shows that the larger the flights amount of the launch site, the greater the success rate will be. However, site CCAFS SLC40 shows the least pattern of this.

Payload vs. Launch Site



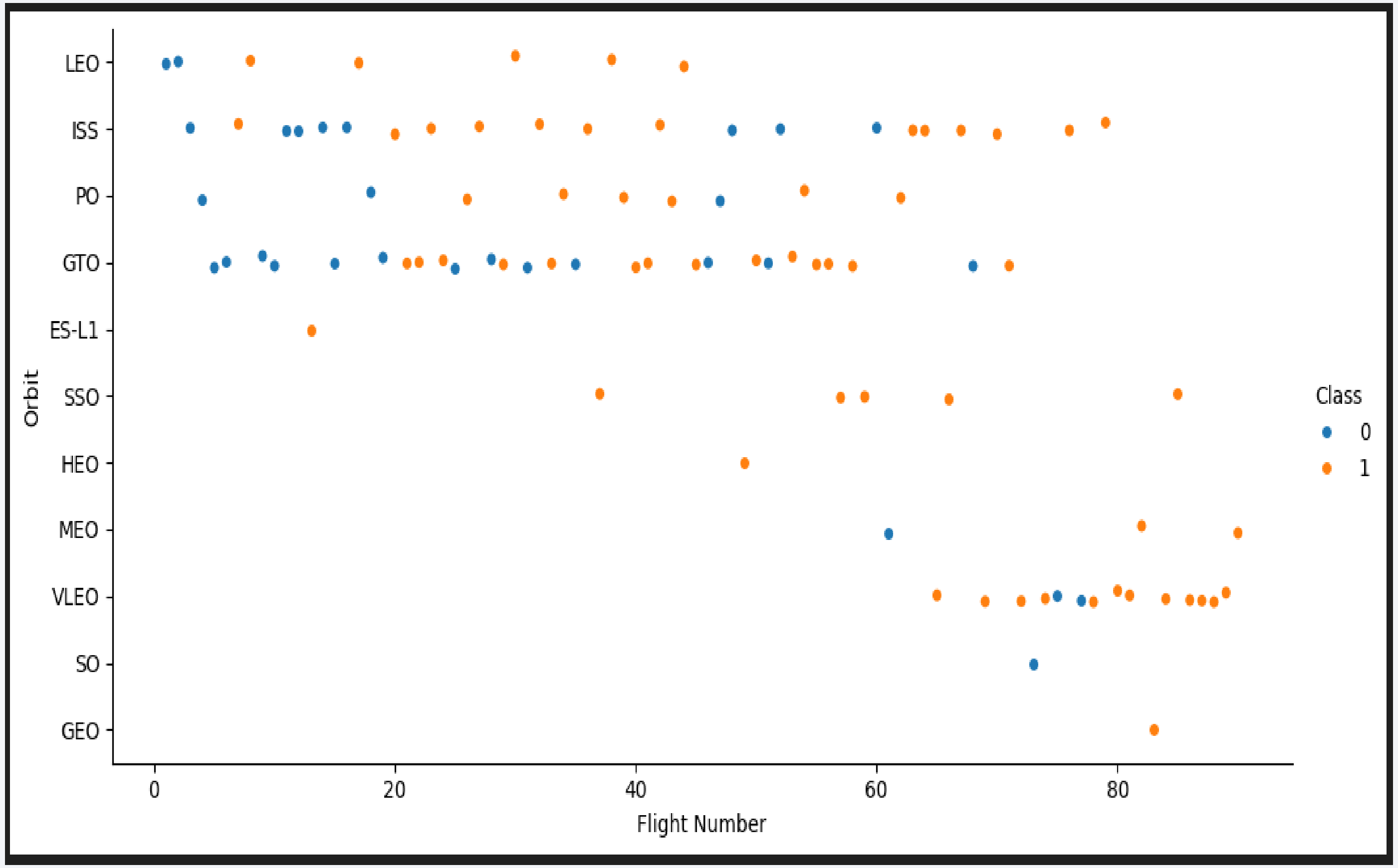
This scatter plot shows once the pay load mass is greater than 7000kg, the probability of the success rate will be highly increased. However, there is no clear pattern to say the launch site is dependent to the pay load mass for the success rate.

Success Rate vs. Orbit Type



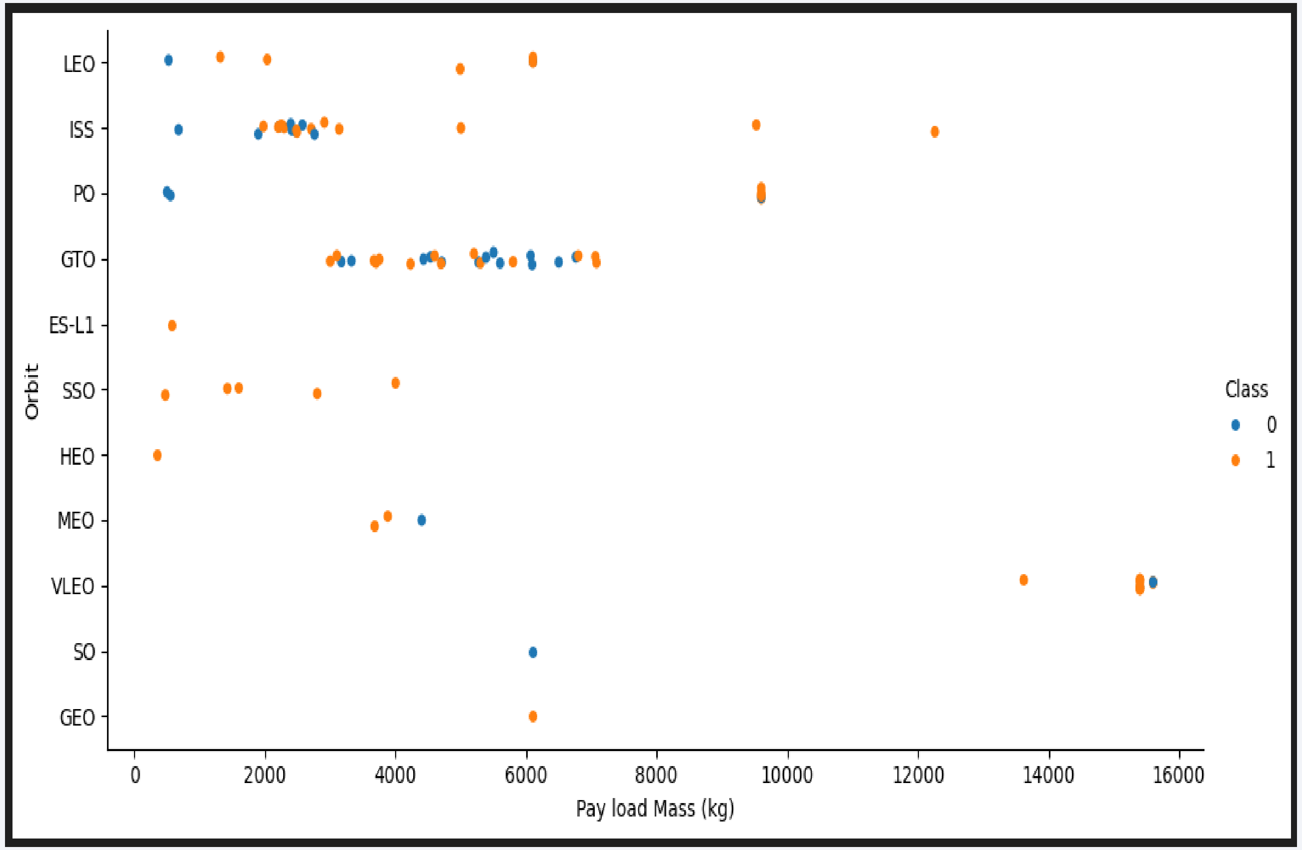
This figure depicted the possibility of the orbits to influences the landing outcomes as some orbits has 100% success rate such as SSO, HEO, GEO AND ES-L1 while SO orbit produced 0% rate of success. However, deeper analysis show that some of this orbits has only 1 occurrence such as GEO, SO, HEO and ES-L1 which mean this data need more dataset to see pattern or trend before we draw any conclusion.

Flight Number vs. Orbit Type



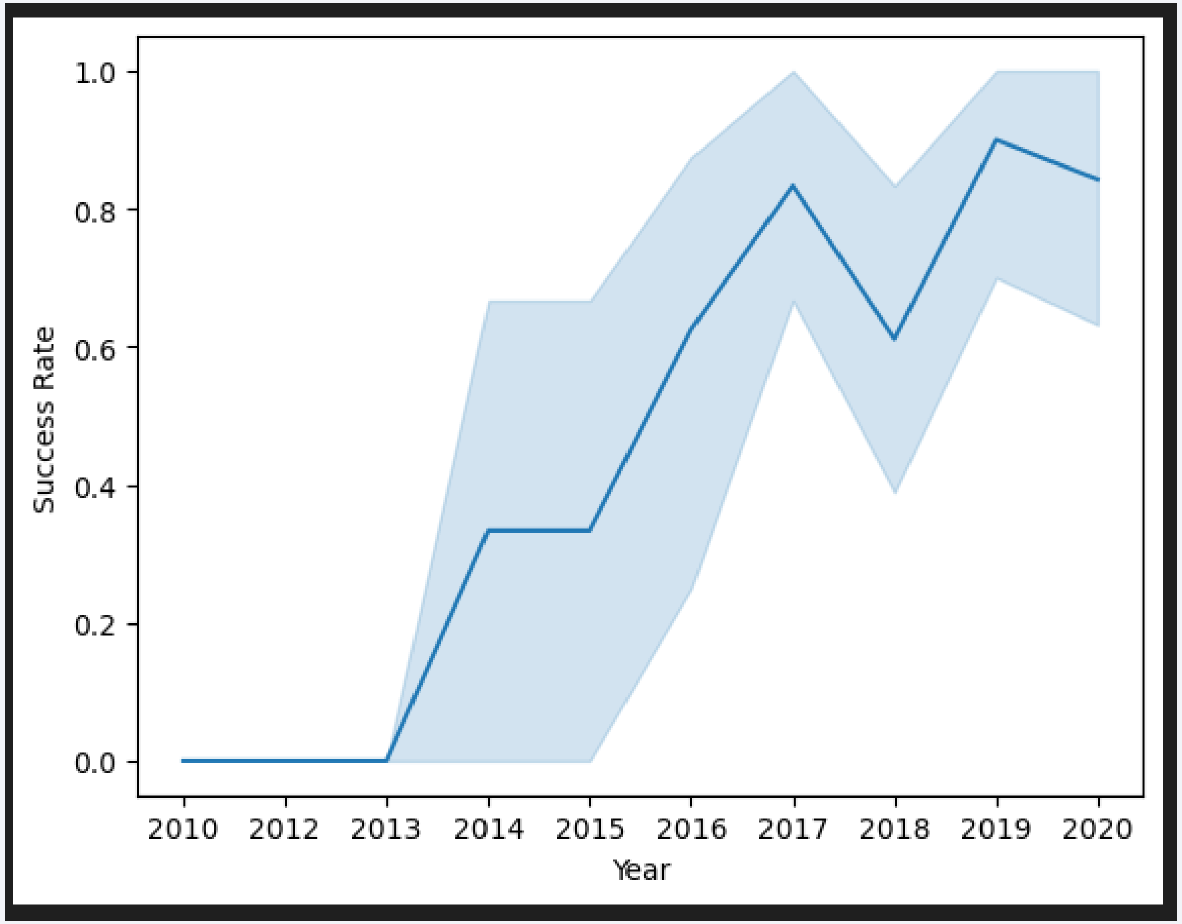
This scatter plot shows that generally, the larger the flight number on each orbits, the greater the success rate (especially LEO orbit) except for GTO orbit which depicts no relationship between both attributes. Orbit that only has 1 occurrence should also be excluded from above statement as it's needed more dataset.

Payload vs. Orbit Type



Heavier payload has positive impact on LEO, ISS and P0 orbit. However, it has negative impact on MEO and VLEO orbit. GTO orbit seem to depict no relation between the attributes. Meanwhile, again, SO, GEO and HEO orbit need more dataset to see any pattern or trend.

Launch Success Yearly Trend



This figures clearly depicted and increasing trend from the year 2013 until 2020. If this trend continue for the next year onward. The success rate will steadily increase until reaching 1/100% success rate.

All Launch Site Names

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

```
%sql select distinct Launch_Site from SPACEXTABLE;
```

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

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

Launch Site Names Begin with 'CCA'

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

```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5;
```

Python

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

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

Total Payload Mass

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

```
%sql select sum(PAYLOAD_MASS__KG_) as TOTAL_PAYLOAD_MASS_KG from spacextable where Customer = 'NASA (CRS)';
```

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

TOTAL_PAYLOAD_MASS_KG
45596

Average Payload Mass by F9 v1.1

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

```
%sql select avg(PAYLOAD_MASS__KG_) as AVERAGE_PAYLOAD_MASS_KG, Customer, Booster_Version from spacextable where Booster_Version like 'F9 v1.1%';
```

Python

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

AVERAGE_PAYLOAD_MASS_KG	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

First Successful Ground Landing Date

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

Hint:Use min function

```
%sql select min(Date) from spacextable where Landing_Outcome = 'Success (ground pad)';
```

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

min(Date)
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

```
%sql select Booster_Version from spacextable where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000;
```

Python

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

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

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 Total FROM SPACEXTABLE GROUP BY "Mission_Outcome";
```

* sqlite:///my_data1.db
Done.

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

Boosters Carried Maximum Payload

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

```
%sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTABLE WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE);
```

Python

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

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

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, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%sql SELECT substr(Date,6,2) as 'Month', substr(Date, 0, 5) as 'Year', Landing_Outcome FROM SPACEXTABLE WHERE substr(Date,0,5)='2015' and
Landing_Outcome = 'Failure (drone ship)';
```

Python

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

Month	Year	Landing_Outcome
01	2015	Failure (drone ship)
04	2015	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)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
%sql SELECT Date, Landing_Outcome, count(Landing_Outcome) as 'Total_Count' FROM SPACEXTABLE WHERE Landing_Outcome in ('Failure (drone ship)',
'Success (ground pad)') AND (Date BETWEEN '2010-06-04' AND '2017-03-20') group by Landing_Outcome;
```

Python

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

Date	Landing_Outcome	Total_Count
2015-01-10	Failure (drone ship)	5
2015-12-22	Success (ground pad)	3

Section 3

Launch Sites Proximities Analysis

Location of all the Launch Sites

We can see that all the SpaceX launch sites are located inside the United States

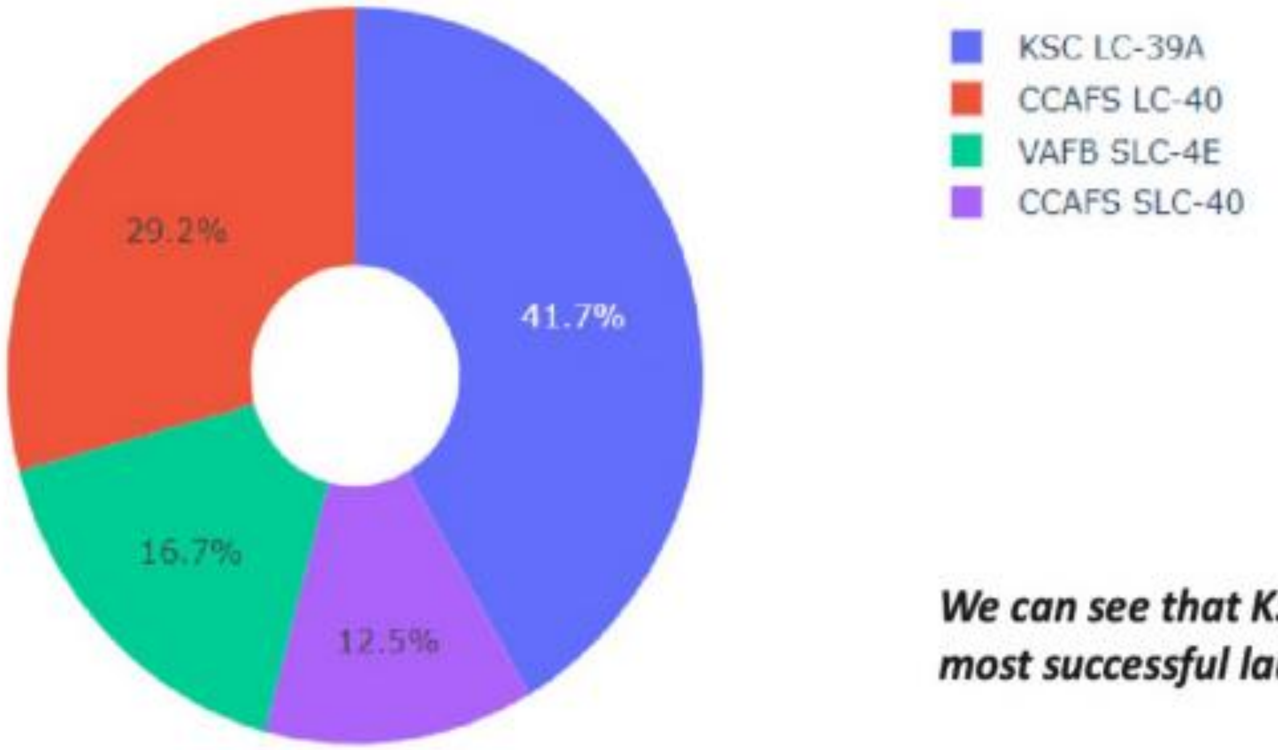
Markers showing launch sites with color labels

Launch Sites Distance to Landmarks

Section 4

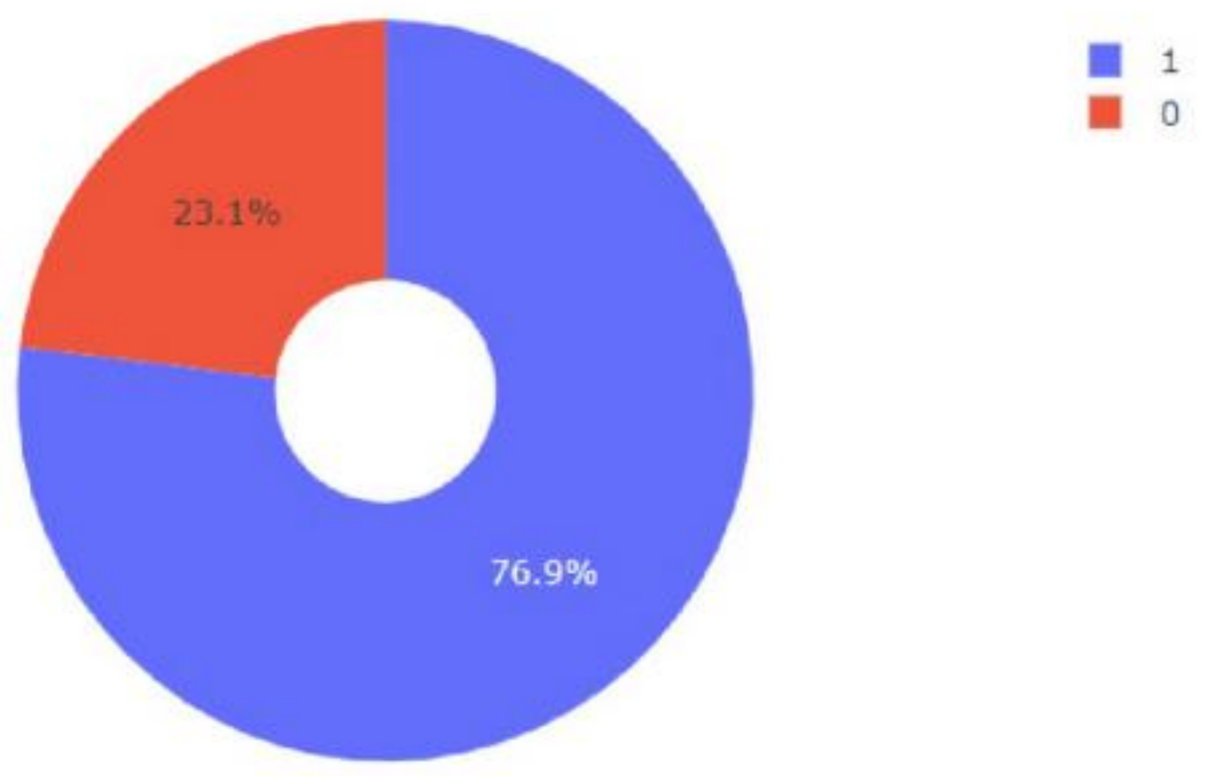
Build a Dashboard with Plotly Dash

The success percentage by each sites.



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

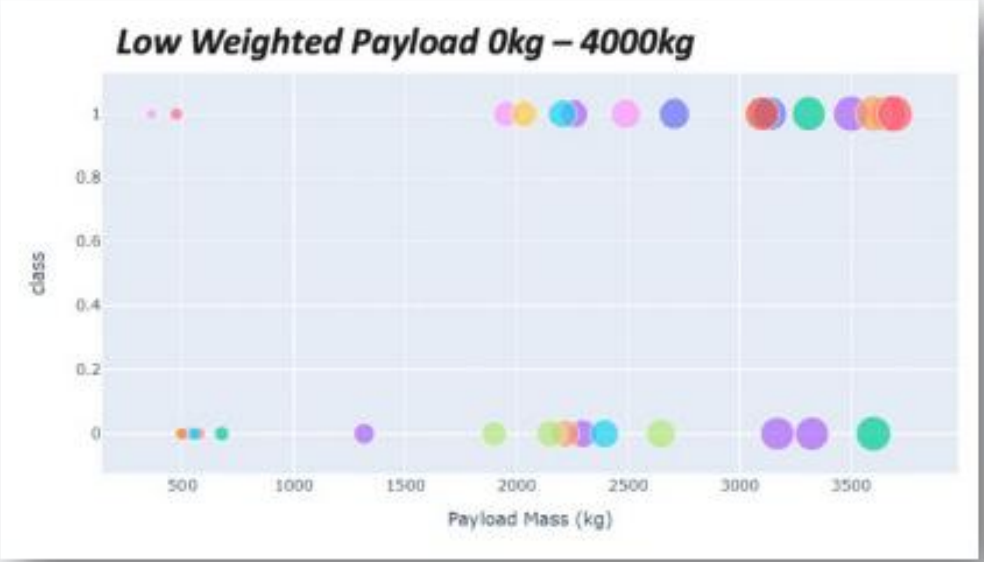
The highest launch-success ratio: KSC LC-39A

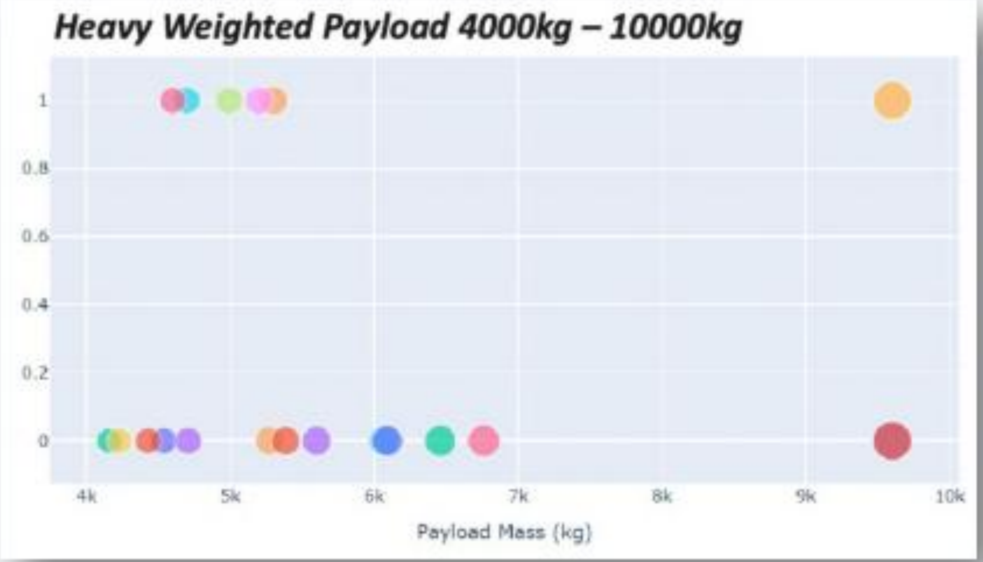


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch Outcome Scatter Plot

We can see that all the success rate for low weighted payload is higher than heavy weighted payload





Section 5

Predictive Analysis (Classification)

Classification Accuracy

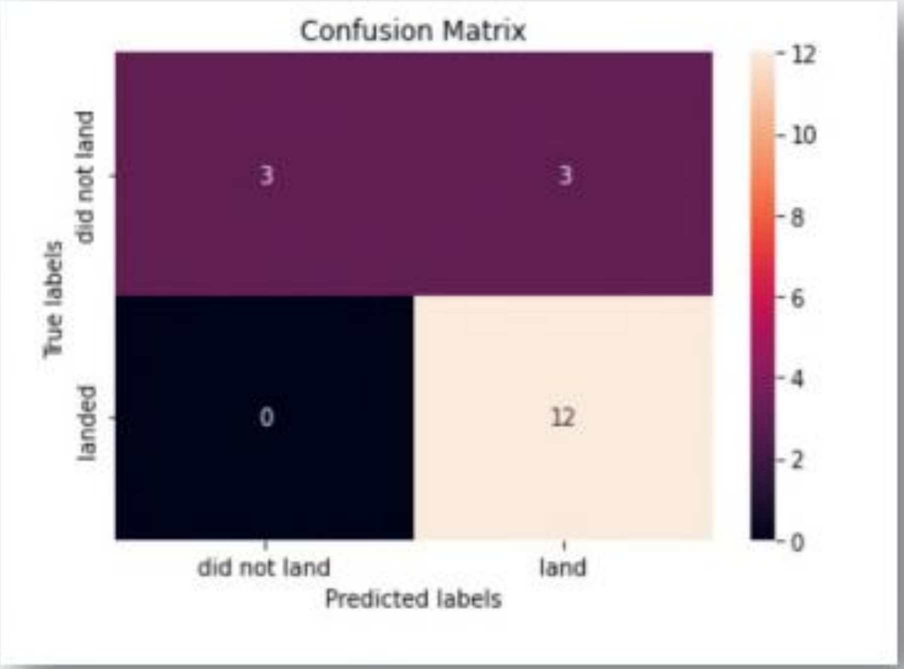
As we can see, by using the code as below: we could identify that the best algorithm to be the Tree Algorithm which have the highest classification accuracy.

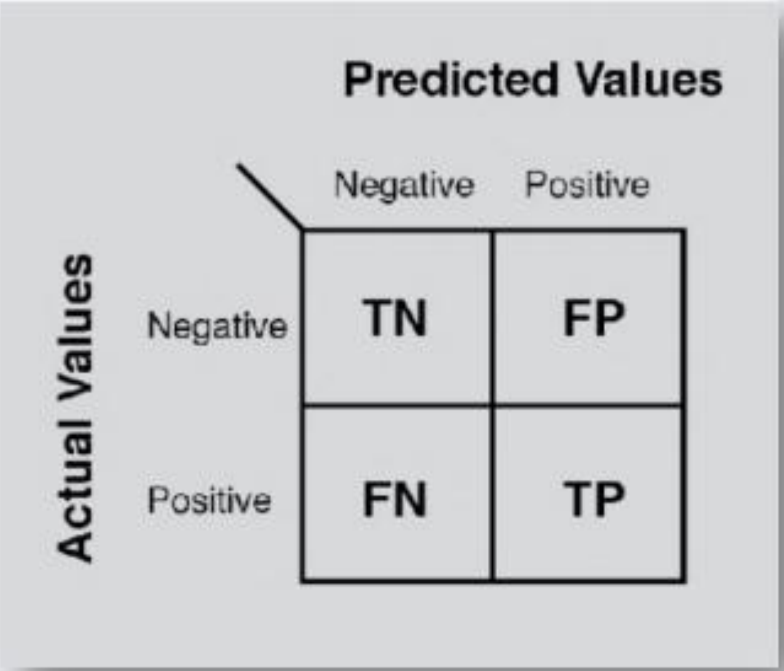
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params 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 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

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 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.
  - ❑ Starting from the year 2013, 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; 76.9%
  - ❑ SSO orbit have the most success rate; 100% and more than 1 occurrence.

Thank you!