IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ankan Bera
24/03/2020

Outline

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

Executive Summary

Summary of methodologies

- Data Collection
- Data Wrangling
- EDA with Data Visualization
- EDA with SQL
- Building an Interactive map with Folium
- Building a Dashboard with Plotly Dash
- Predictive Analysis

Summary of all results

- Exploratory Data Analysis Results
- Interactive Analytics Demo in Screenshots
- Predictive Analysis Results

Introduction

- **Project background and context**

The commercial space age is here, companies are making space travel affordable for everyone. One such company SpaceX, is doing so in a fabulous rate. One reason SpaceX can do this, is the rocket launches are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards 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.

- **Problems you want to find answers**
  - What factors influences the landing of the rocket.
  - What are the effects each independent factors have on the landing and each other to impact the predictions.
  - What conditions does SpaceX has to fulfill to get the best results from their operation.

Section 1

Methodology

Methodology

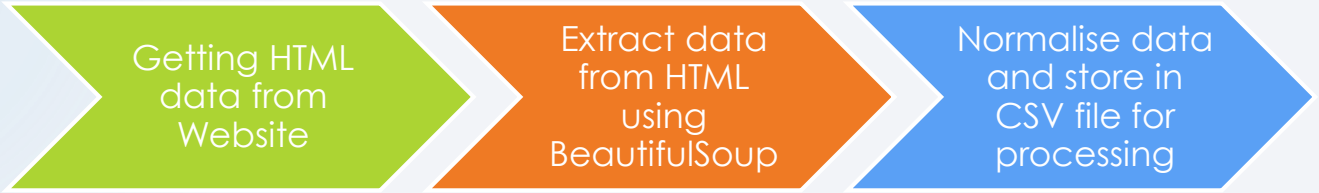
Executive Summary

- **Data collection methodology**:
  - SpaceX REST API
  - Web Scraping
- **Perform data wrangling**
  - Feature encoding for the categorical features to prepare the data for Machine Learning model.
- **Perform exploratory data analysis (EDA) using visualization and SQL**
- **Performed interactive visual analytics using Folium and Plotly Dash**
- **Perform predictive analysis using classification models**
  - How to build, tune, evaluate classification models

Data Collection

- The project used SpaceX launch data which was gathered from the SpaceX REST API.
- From the API, we collected data regarding the launches and the respective technical readings, such as, Type of Rockets used, type of landing pad, location, rocket payload etc.
- The REST API endpoint is https://api.spacexdata.com/v4/launches/latest

- Another way we used to collect Falcon 9 historical launch records from a Wikipedia page using BeautifulSoup Python Library.



Data Collection – SpaceX API

Request to SpaceX REST API → Received data in .json format → Normalise json data and stored in csv file for processing

[GitHub link to Notebook]

1 .Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url).json()
```

2. Converting Response to a .json file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Apply custom functions to clean data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

```
getBoosterVersion(data)
```

4. Assign list to dictionary then dataframe

```
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}
```

```
df = pd.DataFrame.from_dict(launch_dict)
```

5. Filter dataframe and export to flat file (.csv)

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

Getting HTML data from Website → Extract data from HTML using BeautifulSoup → Normalise data and store in CSV file for processing

[Wikipedia page Link]

[GitHub link to Notebook]

1 .Getting Response from HTML

```
page = requests.get(static_url)
```

2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

3. Finding tables

```
html_tables = soup.find_all('table')
```

4. Getting column names

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

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


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'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Appending data to keys (refer) to notebook block 12

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
            # get table row
             for rows in table.find_all("tr")
                 #check to see if first table
```

7. Converting dictionary to dataframe

```
df = pd.DataFrame.from_dict(launch_dict)
```

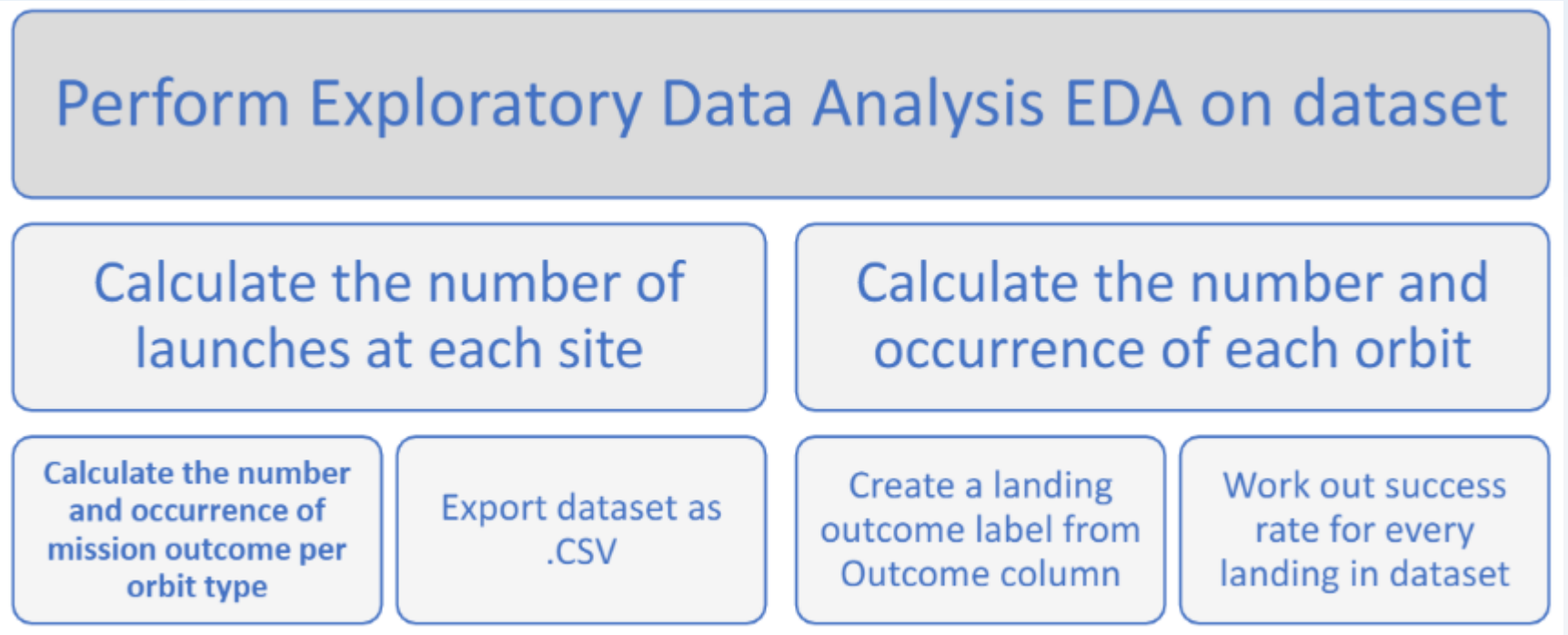
8. Dataframe to .CSV

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad.True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.

Using Data Wrangling, we will mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.



GitHub link to Notebook

EDA with Data Visualization

Performed exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib

Scatter Plots	Bar Plots	Line Plots
Flight Number vs. Payload Mass	Mean vs. Orbit	Success Rate vs Year
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.	A bar diagram makes it easy to compare sets of data between different groups at a glance. The graph represents categories of one axis and a discrete value in the other. The goal is to show the relationship between two axes.	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 to Notebook

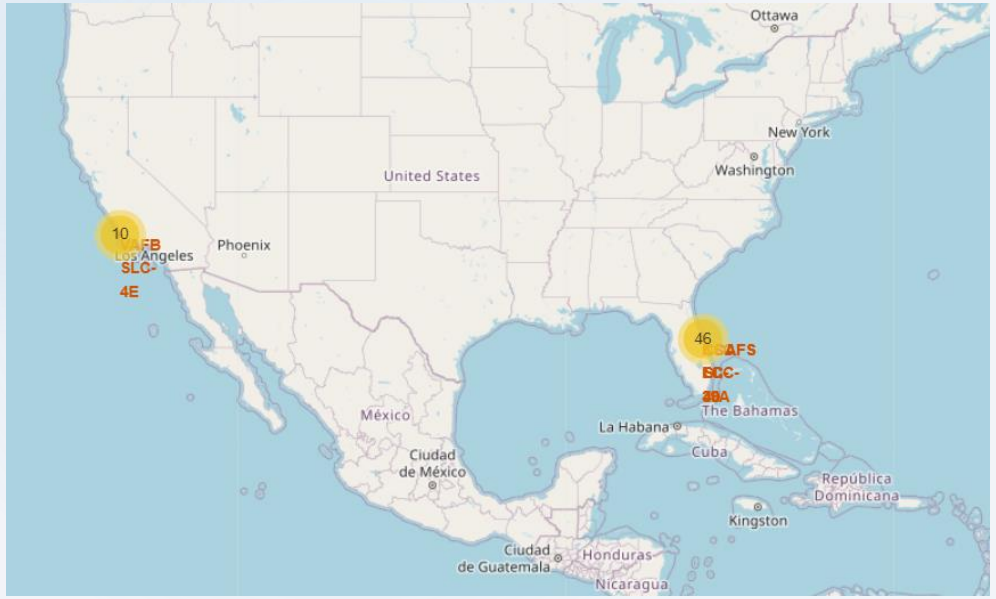
EDA with SQL

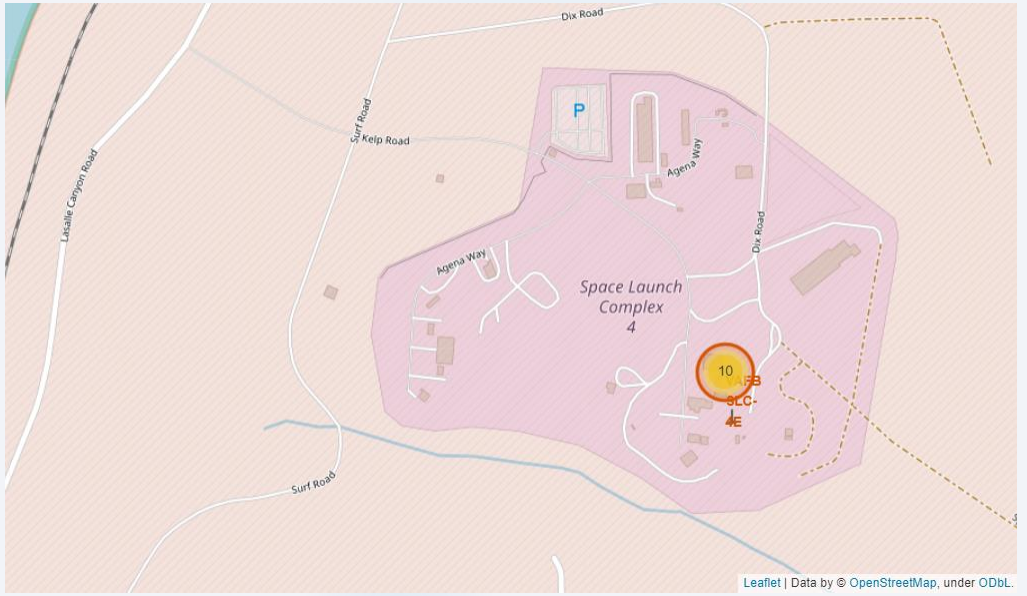
- Displaying the names of the unique launch sites in the space mission
- Displaying 5 records where launch sites begin with the string ‘CCA’
- Displaying the total payload mass carried by booster launched by NASA (CRS)
- Displaying the average payload mass carried by booster version F9 v1.1
- Listing the date where the 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 records which will display the month names, successful landing outcomes in drone ships, booster versions, launch site for the months in the year 2015
- Ranking the count of successful landing outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

GitHub Link to Notebook

Build an Interactive Map with Folium

- To visualize the Launch data into an interactive map, I 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.
- I assigned the dataframe launch_outcomes(failures,successes) to classes 0 and 1. with Green and Red markers in the map using MarkerCluster().
- Using Haversine's Formula I calculated the distance from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. Lines are drawn on the map to measure distance to Landmarks.



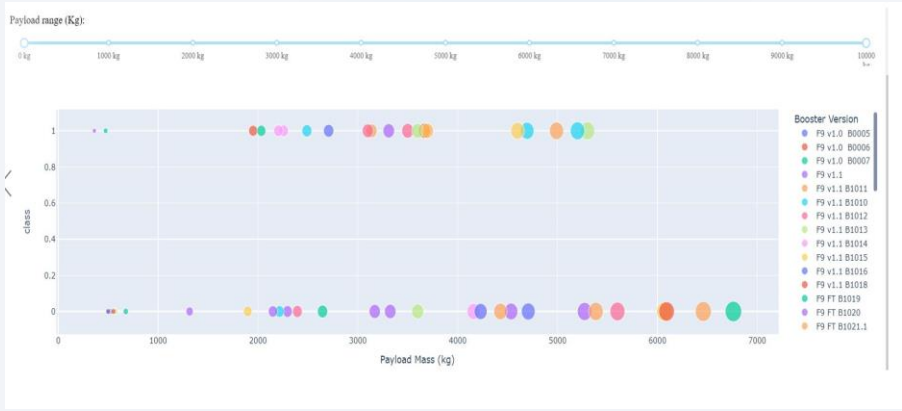


GitHub Link to Notebook

Build a Dashboard with Plotly Dash

- A dashboard is build using Plotly and Dash to Visualize and understand the data in a better manner.
- Pie Chart showing the total launches by a certain site or all sites.
  - Displays relative proportions of multiple classes of data
  - Size of the circle can be made proportional to the total quantity it represents.

- Scatter graph showing the relationship with Outcome and Payload Mass for the different booster versions.
  - Shows the relationship between two variables.
  - It is the best method to show a no-linear pattern.
  - The range of data flow can be determined.
  - Observation and readings are straightforward.



GitHub Link to Source Code

Predictive Analysis (Classification)

Building Model

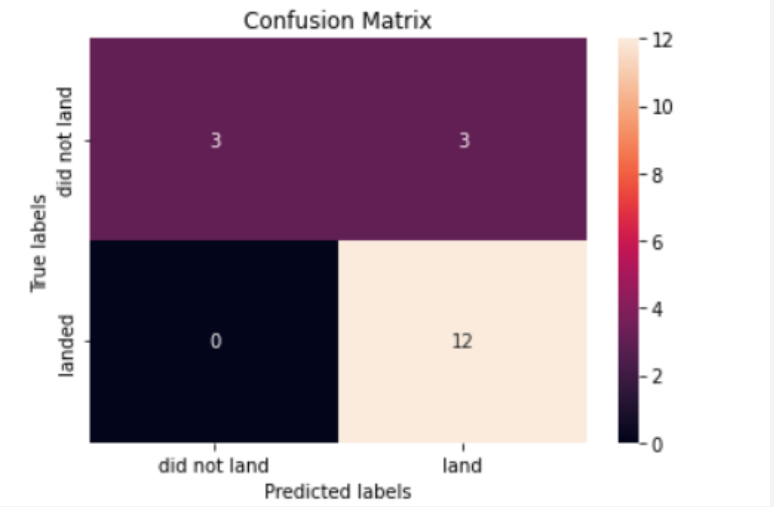
- Load our dataset using NumPy and Pandas
- Transform data as per given logic
- Split our data into Train and Test set.
- Check how many test samples we have.
- Decide which type of machine Learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV and train our dataset.

Evaluating Model

- Check Accuracy for each model
- Hyperparameter tuning for each type of algorithms
- Plot Confusion matrix.

Improving Model

- Feature Engineering
- Algorithm Tuning



[GitHub Link to Notebook]

- 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

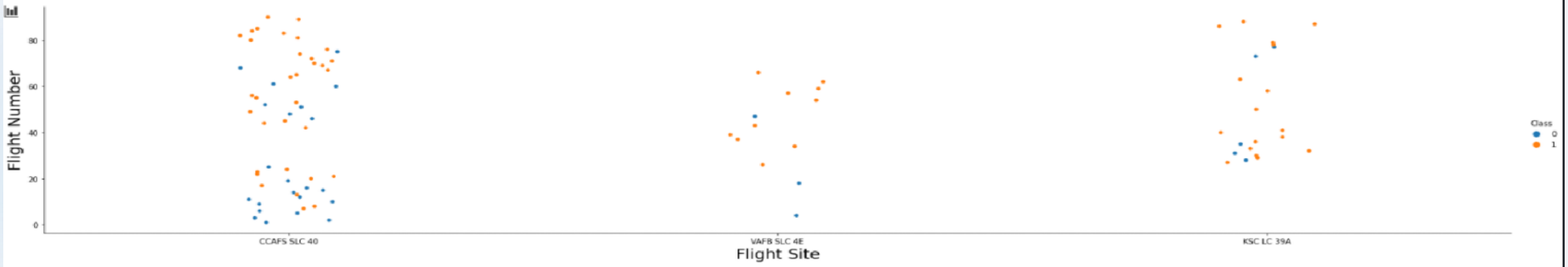
Results

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

Section 2

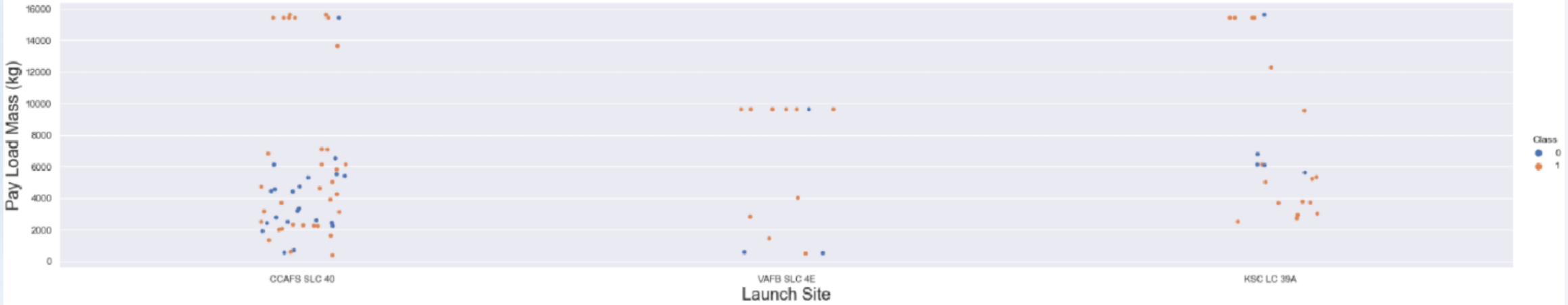
Insights drawn from EDA

Flight Number vs. Flight Site



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

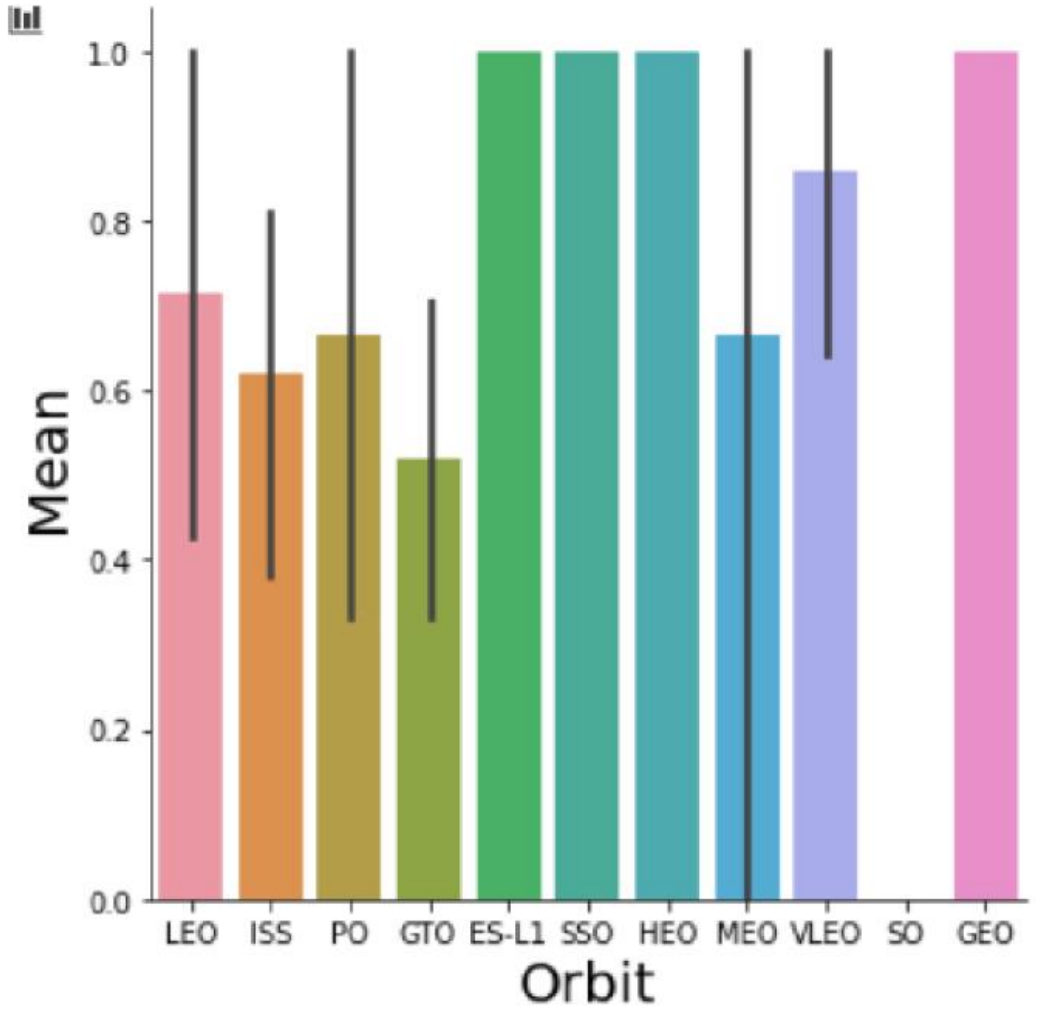
Payload Mass vs. Launch Site



The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket.
There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependant on Pay Load Mass for a success launch.

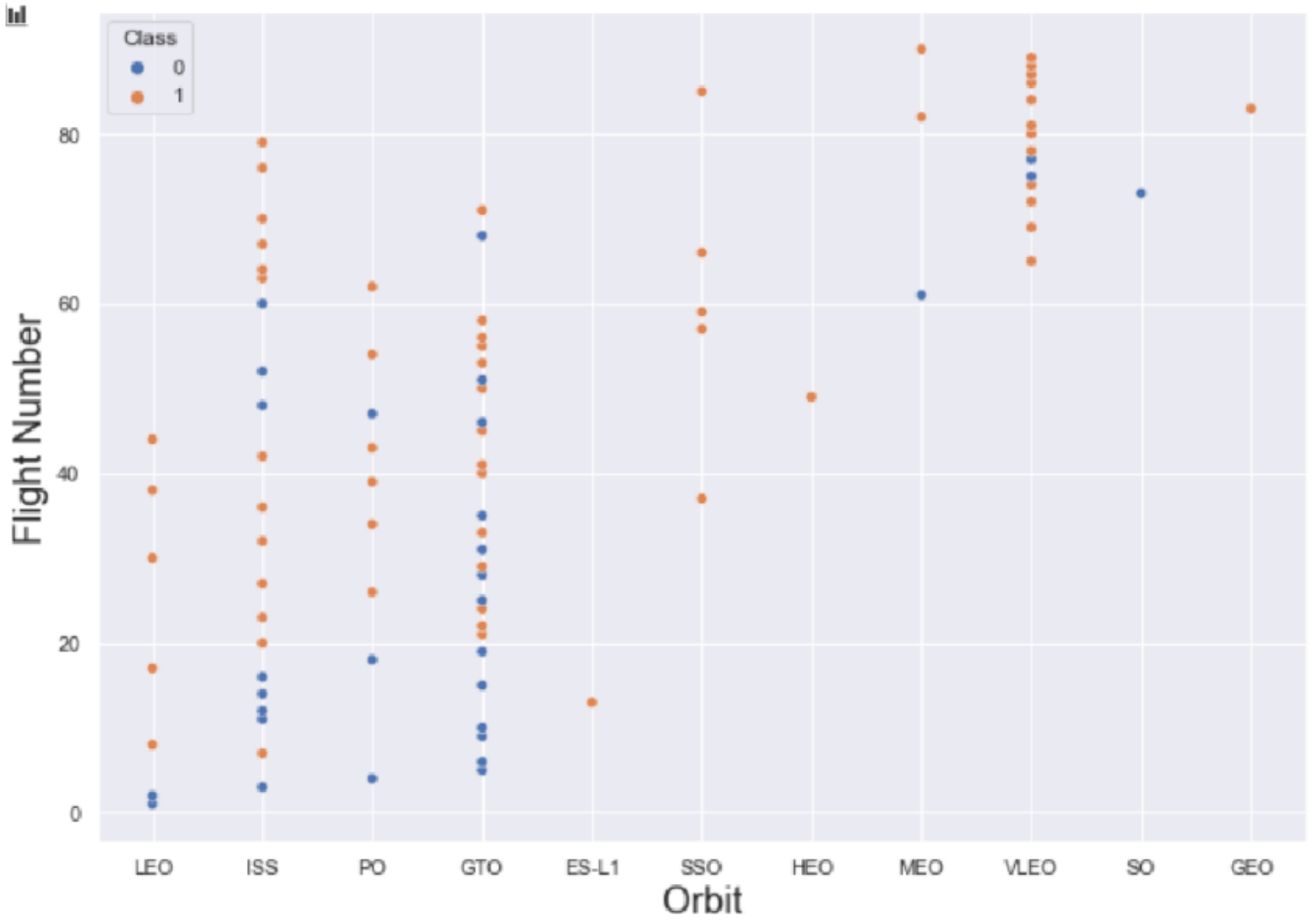
Success rate vs. Orbit type

Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate



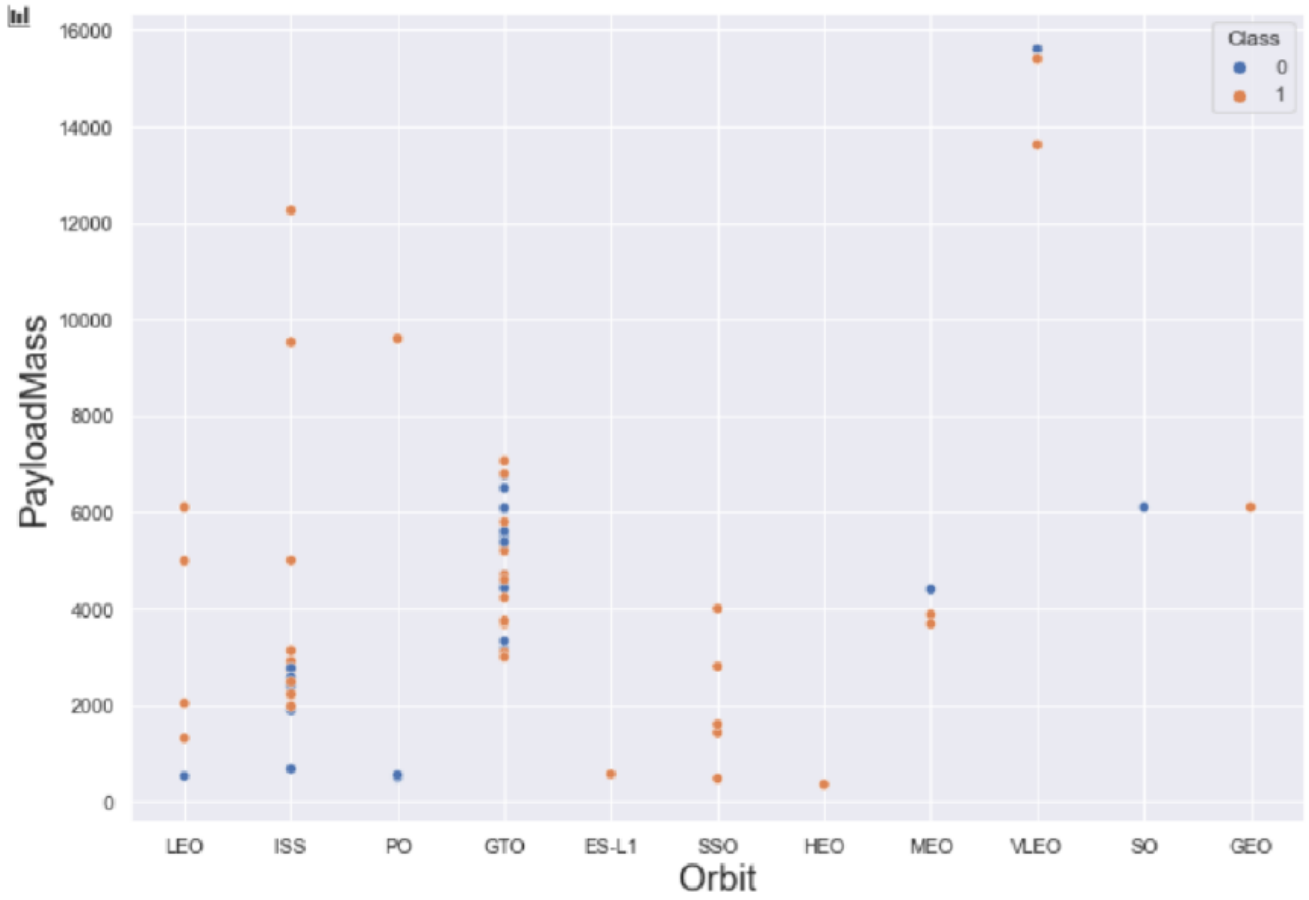
Flight Number vs. Orbit type

You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

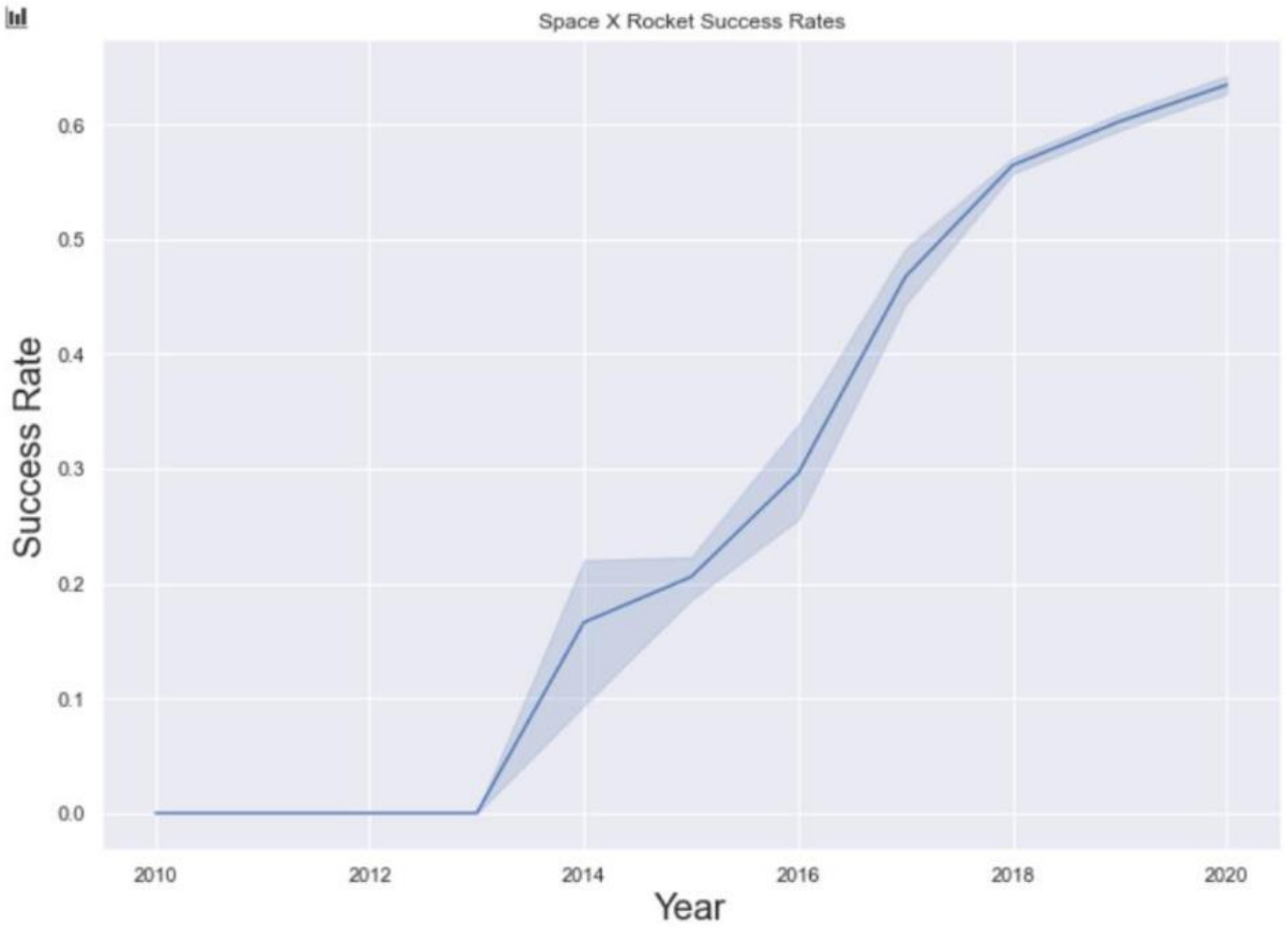


Payload vs. Orbit type

You should observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.



Launch success yearly trend



you can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

SQL QUERY

```
select DISTINCT Launch_Site
from tblSpaceX
```

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

QUERY EXPLAINATION

Using the word ***DISTINCT*** in the query means that it will only show Unique values in the ***Launch_Site*** column from ***tblSpaceX***

Launch Site Names Begin with 'CCA'

SQL QUERY

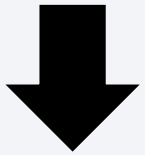
```
select * from spacextbl WHERE Launch_Site LIKE 'CCA%' limit 5
```

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

SQL QUERY

```
select SUM(payload_mass__kg_) as TotalPayloadMass from
spacextbl where Customer = 'NASA (CRS)'
```

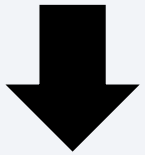


totalpayloadmass
45596

Average Payload Mass by F9 v1.1

SQL QUERY

```
select AVG(payload_mass__kg_) as AveragePayloadMass from spacextbl where
Booster_Version = 'F9 v1.1'
```

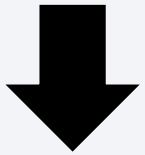


averagepayloadmass
2928

First Successful Ground Landing Date

SQL QUERY

```
select MIN(Date) as first_successful_landing_outcome_in_ground_pad from spacextbl where landing__outcome = 'Success (ground pad)'
```

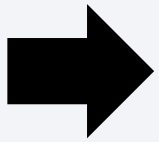


first_successful_landing_outcome_in_ground_pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL QUERY

```
select Booster_Version from spacextbl where landing__outcome
= 'Success (drone ship)' AND payload_mass__kg_ > 4000 AND
payload_mass__kg_ < 6000
```

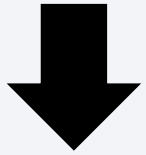


booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

SQL QUERY

```
SELECT (SELECT Count(Mission_Outcome) from spacextbl where Mission_Outcome LIKE '%Success%') as Successful_Mission_Outcomes,(SELECT Count(Mission_Outcome) from spacextbl where Mission_Outcome LIKE '%Failure%') as Failure_Mission_Outcomes from spacextbl
```

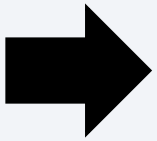


successful_mission_outcomes	failure_mission_outcomes
100	1

Boosters Carried Maximum Payload

SQL QUERY

```
SELECT DISTINCT Booster_Version,
MAX(PAYLOAD_MASS__KG_) AS
Maximum_Payload_Mass FROM SpaceXtbl
GROUP BY Booster_Version ORDER BY
Maximum_Payload_Mass DESC
```

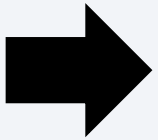


booster_version	maximum_payload_mass
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600
F9 B5 B1049.6	15440

2015 Launch Records

SQL QUERY

```
SELECT Booster_Version,
Launch_Site,
LANDING__OUTCOME,DATE FROM
SpaceXtbl WHERE
(LANDING__OUTCOME LIKE
'%Failure (drone ship)%') and
YEAR(DATE) = 2015
```



booster_version	launch_site	landing__outcome	DATE
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)	2015-01-10
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)	2015-04-14

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

SQL QUERY

```
SELECT COUNT(LANDING__OUTCOME) FROM SpaceXtbl WHERE (LANDING__OUTCOME LIKE '%Success%' or LANDING__OUTCOME LIKE '%Failure%') AND (Date > '2010-06-04' AND Date < '2017-03-20')
```

1
14

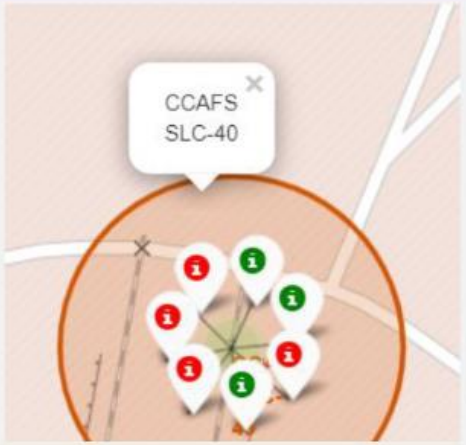
Section 3

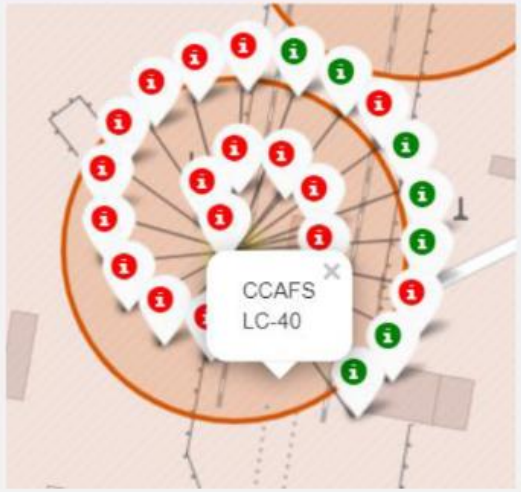
Launch Sites Proximities Analysis

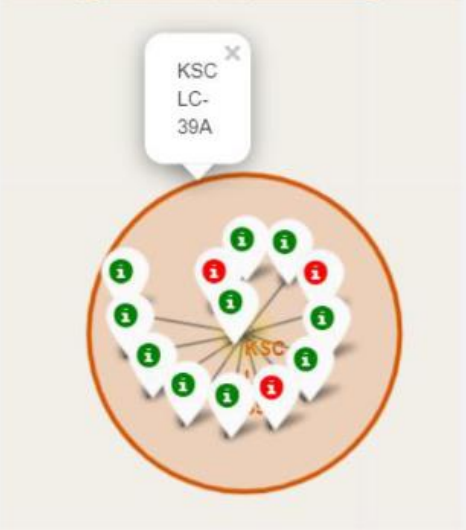
All Launch Sites Global Map Markers

We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

Color Labeled Markers

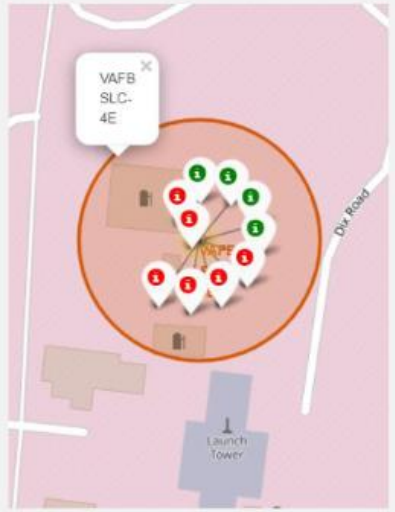






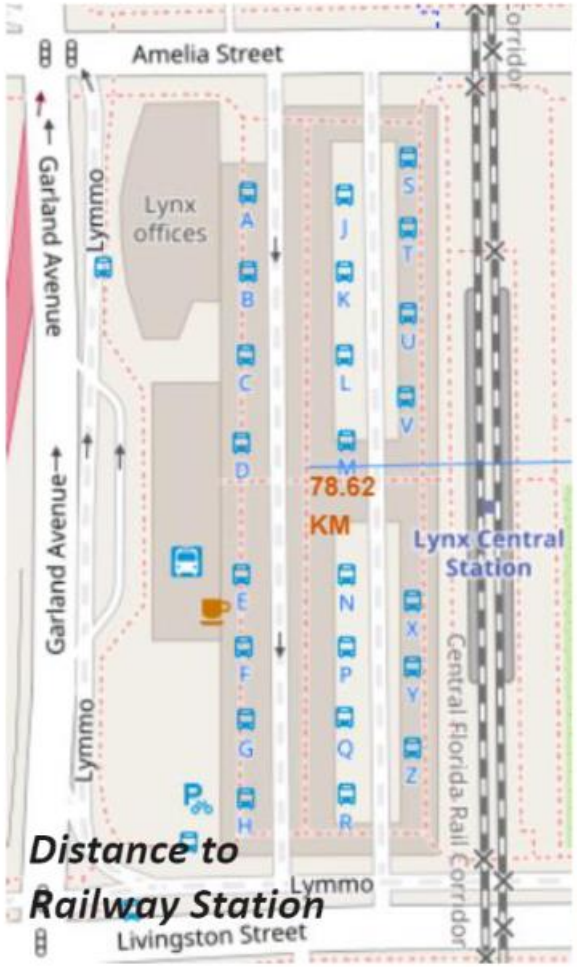
Florida Launch Sites

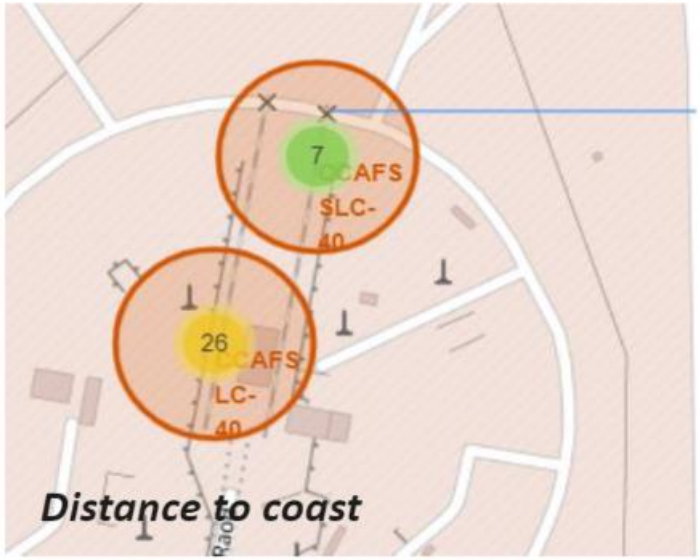
Green Marker shows successful Launches and Red Marker shows Failures

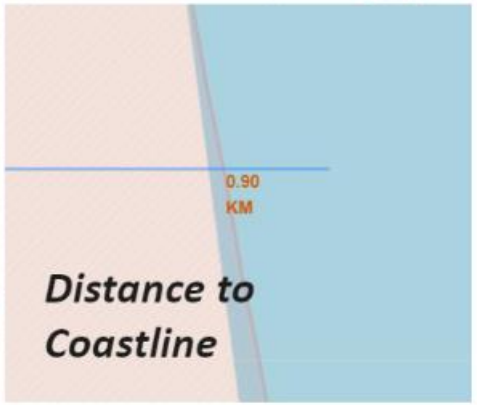


California Launch Site

Launch Site Distance to Landmarks





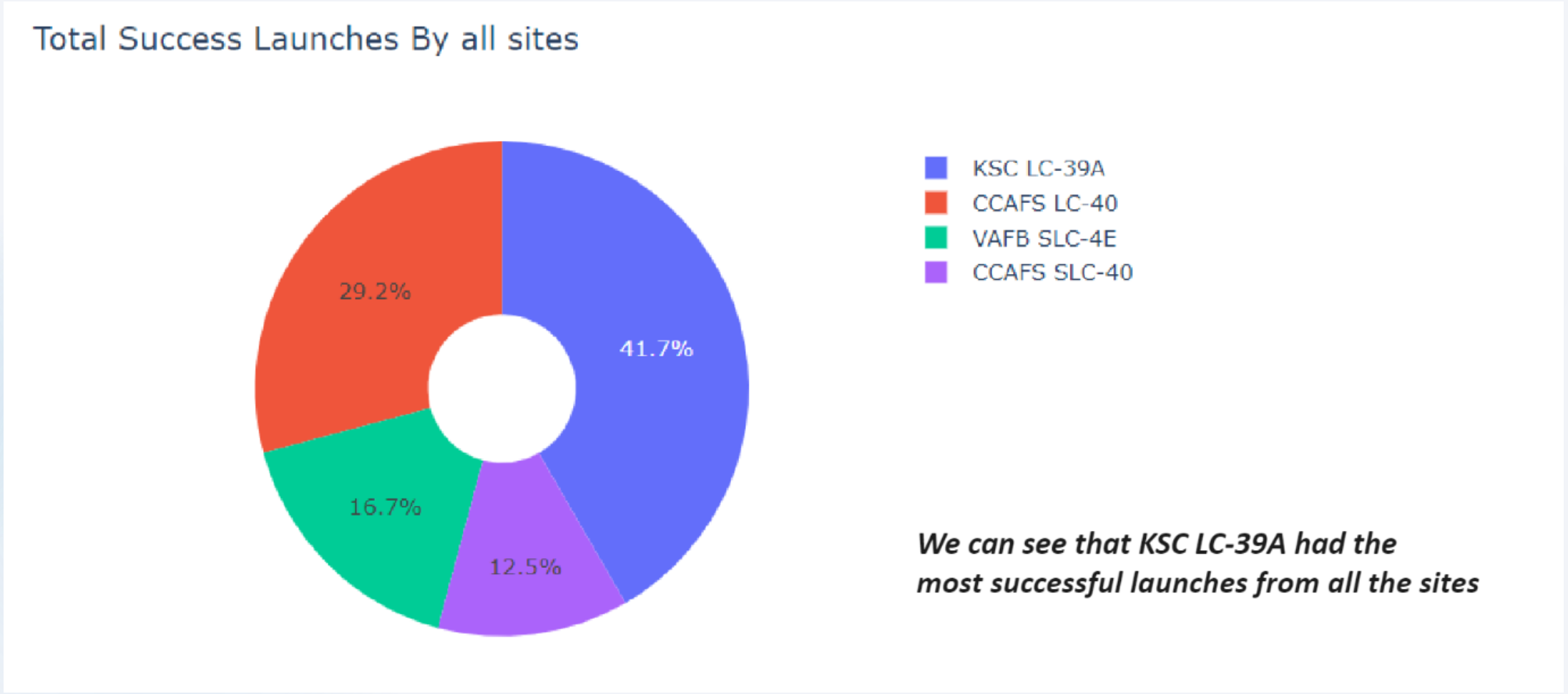


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

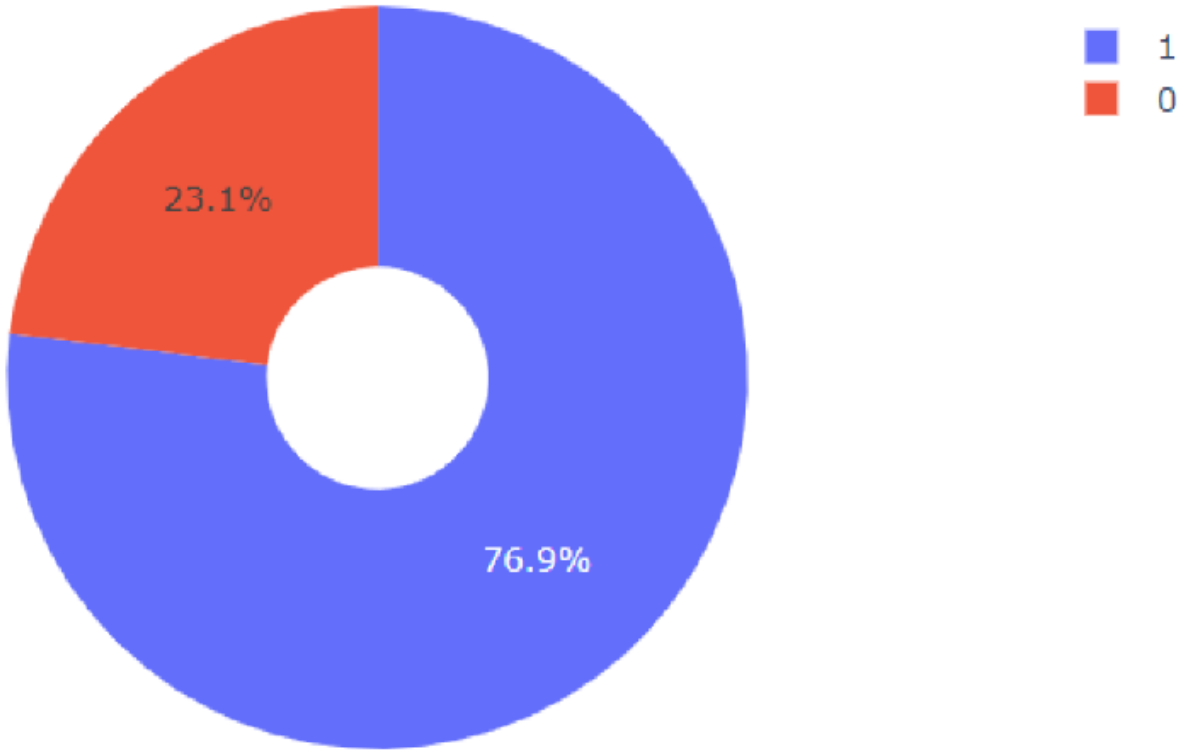
Build a Dashboard with Plotly Dash

Pie Chart Showing the Success Percentage



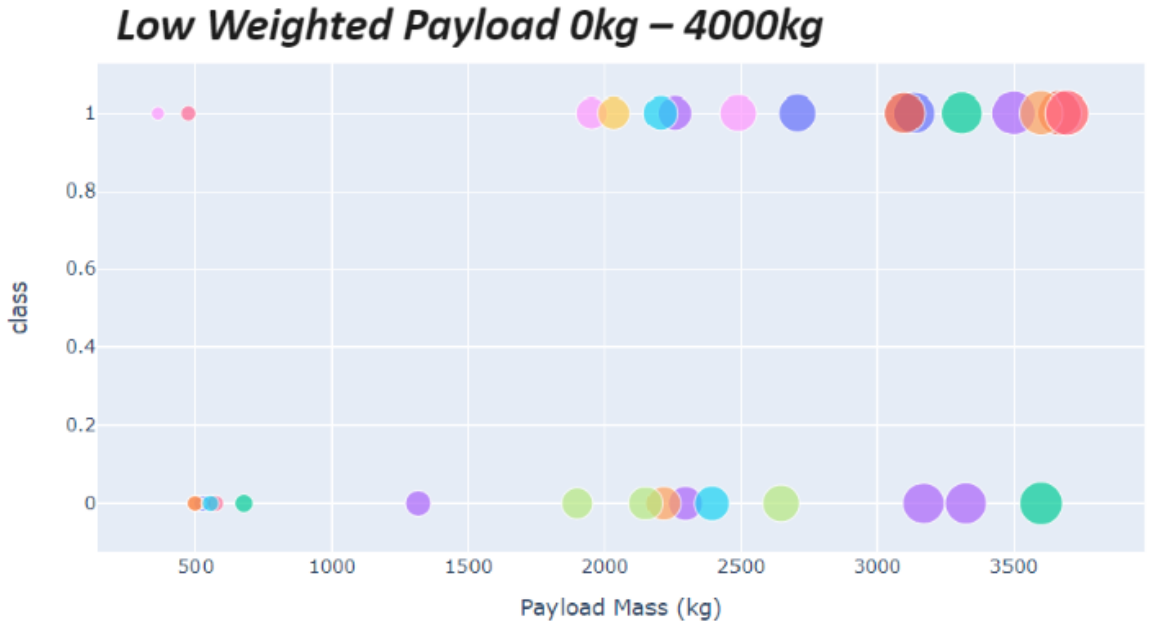
We can see that KSC LC-39A had the most successful launches from all the sites

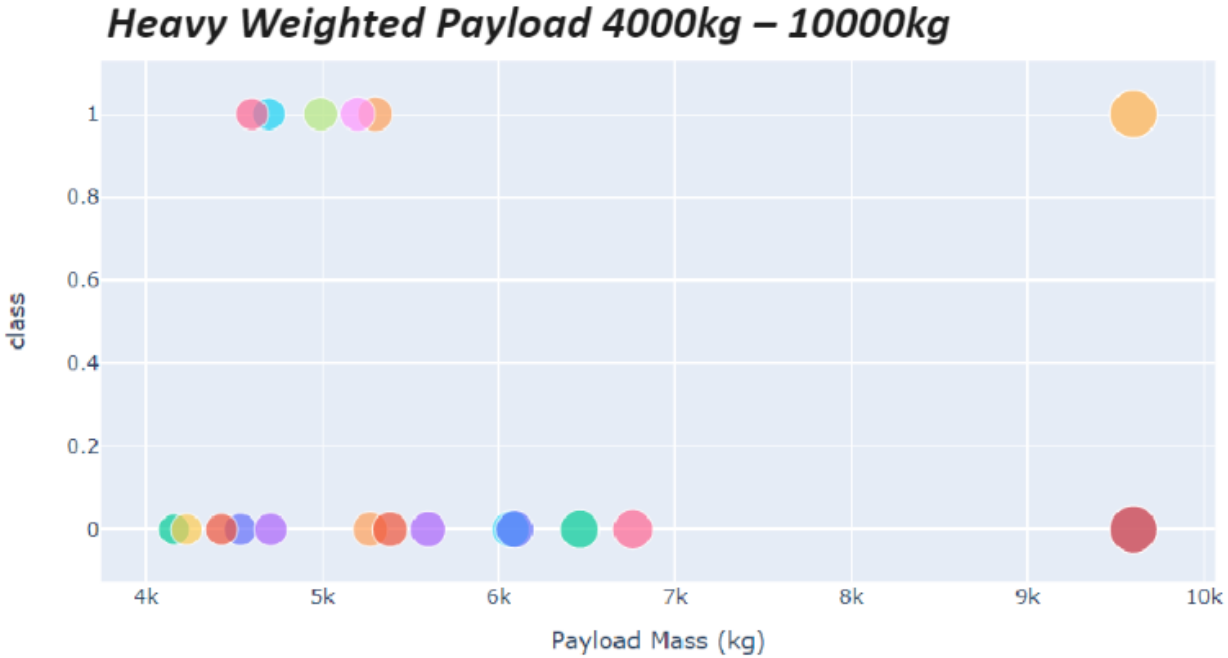
Launch Site with Highest Success Ratio



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 the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

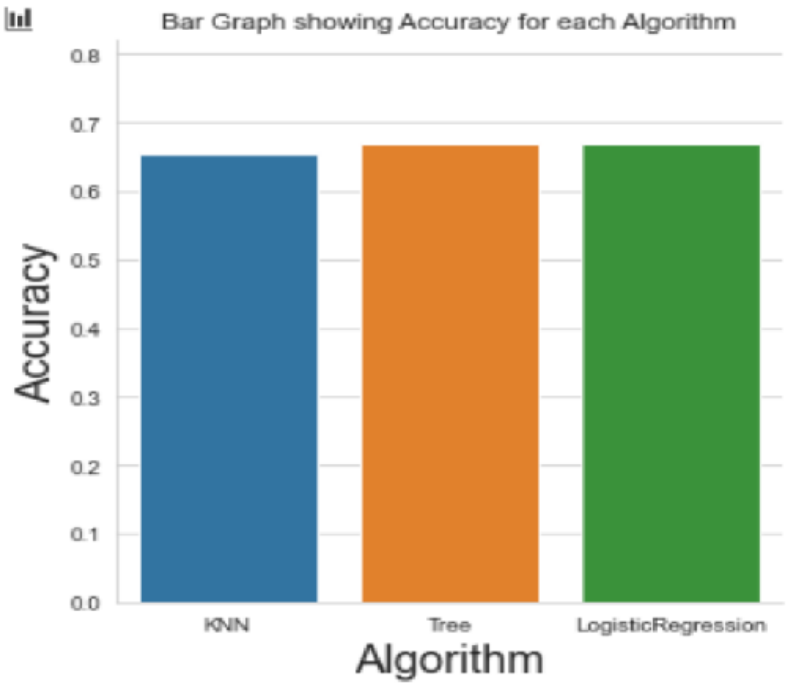
Classification Accuracy

Classification Accuracy using training data

As you can see our accuracy is extremely close but we do have a winner its down to decimal places! using this function

```
bestalgorithm = max(algorithms, key=algorithms.get)
```

	Accuracy	Algorithm
0	0.653571	KNN
1	0.667857	Tree
2	0.667857	LogisticRegression



The tree algorithm wins!!

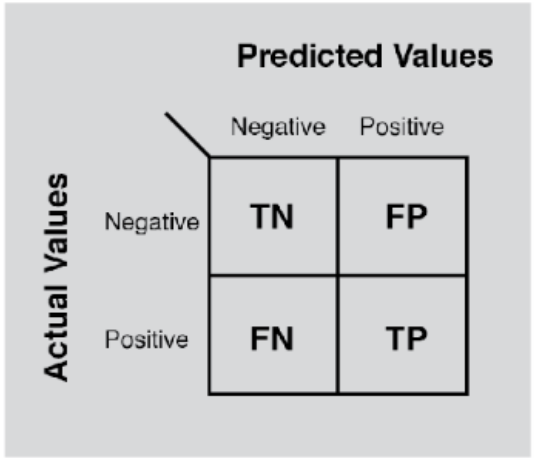
```
Best Algorithm is Tree with a score of 0.6678571428571429
Best Params is : {'criterion': 'gini', 'max_depth': 2, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
```

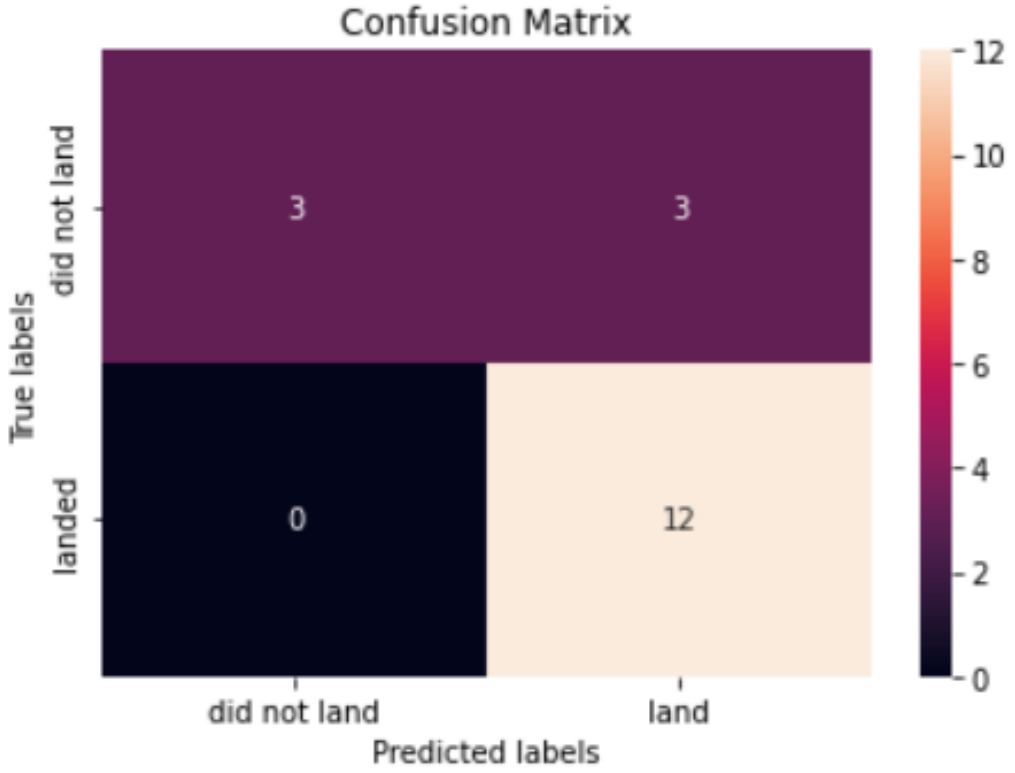
After selecting the best hyperparameters for the decision tree classifier using the validation data, we achieved 83.33% accuracy on the test data.

Confusion Matrix

Confusion Matrix for the Tree

Examining the confusion matrix, we see that Tree can distinguish between the different classes. We see that the major problem is false positives.





Conclusions

- The Decision Tree Classifier algorithm is found out to be the most suitable for this dataset.
- Low weighted payloads perform better than the heavier payloads.
- The success rates for SpaceX launches is directly proportional to the time in years they take to perfect their process.
- WE can see that KSC LC-39A has the most successful launches from all the sites.
- Orbit GEO, HEO, SSO, ES-L1 has the best success rate.

Appendix

- GitHub Repository **Link**
- Haversine Formula. **Wikipedia Link**
- Decision Tree Learning **Wikipedia Link**
- SpaceX API Documentation **Link**

Thank you!