IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Name>
<Date>

Outline

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

Executive Summary

SUMMARY OF METHODOLOGIES:

- Data collection
- Data wrangling
- EDA with data visualization and SQL
- Interactive map with folium
- Dashboard with plotly
- Predictive analysis

SUMMARY OF ALL RESULTS:

- EDA results
- Interactive analysis
- Predictive analysis

Introduction

PROJECT BACKGROUND AND CONTEXT:

- ➢ SpaceX advertises that Falcon 9 rocket launches with the cost of 62 million dollars where others cost up to 165 million dollars. That is because spaceX reuses the first stage. If we can predict if the first stage can land successfully or not, then we can determine the cost of launch.

PROBLEMS YOU WANT TO FIND ANSWERS:

- What does SpaceX need to do to ensure that the first stage of landing is successful
- What variables will impact the landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Web scrapping from Wikipedia pages and Rest APIs
- Perform data wrangling:
  - Dropping unwanted columns and transforming the data using one-hot encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Scatter plots and bar graphs
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

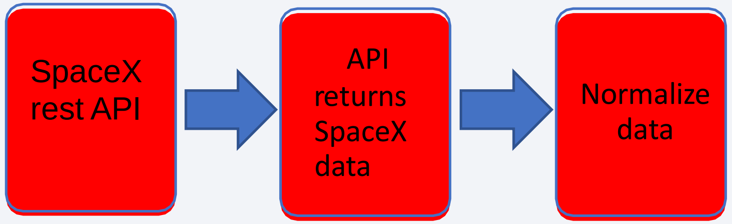
Data Collection

Data was collected from:

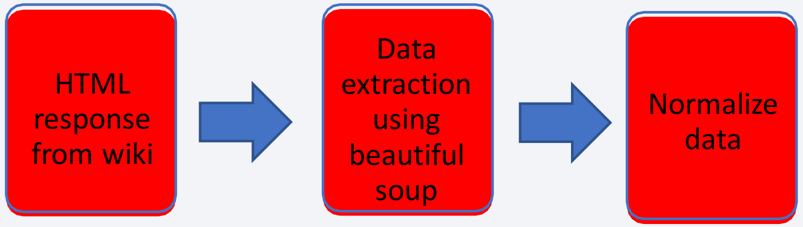
- SpaceX Rest API data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- Webscraping from Wikipedia

Rest API URL: api.spacexdata.com/v4/.

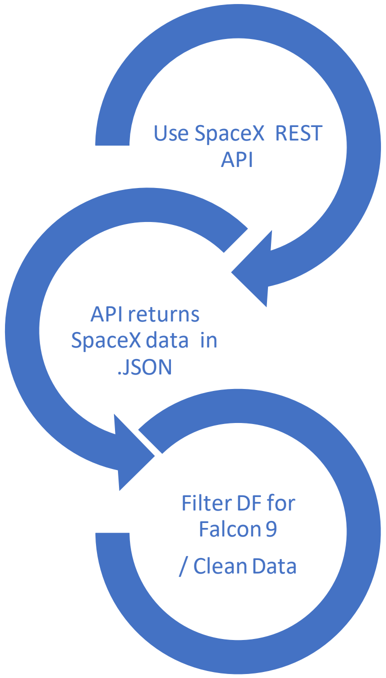
SPACEX REST API



WEB SCRAPING



Data Collection SpaceX API



1. Response from API:

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

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

2. Response into JSON file:

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

3. Functions to clean Data:

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

4. List -> Dictionary -> Dataframe:

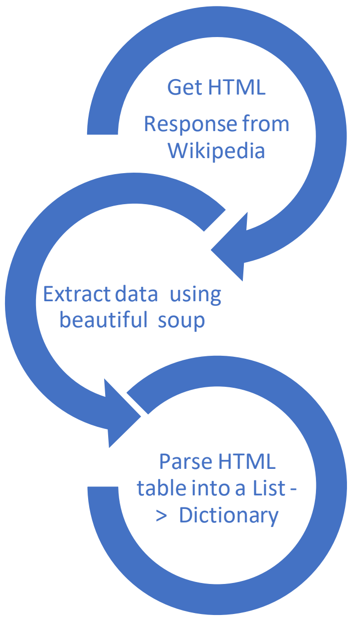
```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5. Filter data and convert to .csv file

```
# Hint data['BoosterVersion']!='Falcon 1'
data_fal = df1['BoosterVersion'] =='Falcon 9'
data_falcon9 = df1[data_fal]
data_falcon9.head()
```

```
data_falcon9.to_csv('dataset_part1.csv', index=False)
```

DATA WRANGLING



1.Getting response from wiki

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

2. Creating beautiful soup object

```
soup = BeautifulSoup(response.text, "html.parser")

print(soup.prettify())
```

3. Find tables and get column names

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

4. Create a 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']=[]
```

5. Converting dictionary into Dataframe

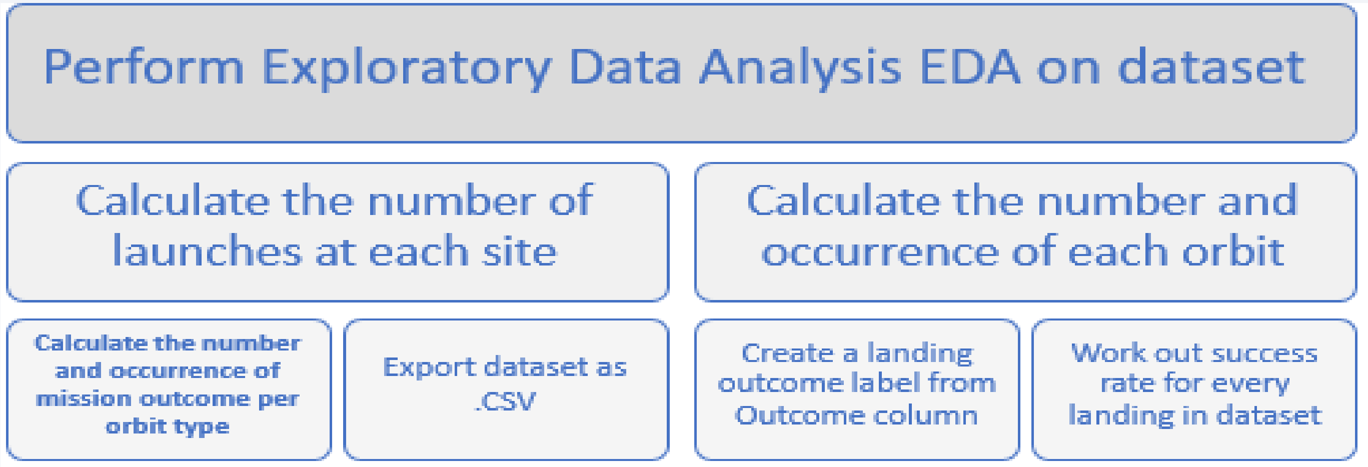
```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',
ders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
```

6. Dataframe to .csv

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

Data Collection - Scraping

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. We mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.



GitHub URL: https://github.com/swathireddy14/IBM-DS-capstone-project

EDA WITH DATA VISUALIZATION

SCATTER PLOTS:

- Flight number vs Launch Site
- Flight number vs Payload Mass
- Payload vs launch Site
- Payload vs Orbit Type
- Orbit vs Flight number
- Orbit vs Payload Mass

LINE GRAPH:

- Success Rate vs Year

BAR GRAPH:

- Mean vs Orbit

EDA WITH SQL

QUESTIONS RELATED TO SQL:

1. Display the names of the unique launch sites in the space mission
2. Display 5 records where launch sites begin with the string 'CCA‘
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display average payload mass carried by booster version F9 v1.1
5. List the date where the succesfulL landing outcome in drone ship was achieved
6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. List the total number of successful and failure mission outcomes
8. List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
9. List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
10. 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

GITHUB URL: https://github.com/swathireddy14/IBM-DS-capstone-project

Build an Interactive Map with Folium

To visualize the Launch Data into an interactive map, Latitude and Longitude are taken, Coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.

Assigned the dataframe launch_outcomes(failures, successes) to *classes 0 and 1* with Green and Red markers on the map in a MarkerCluster()

- 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

PREDICTIVE ANALYSIS

BUILDING MODEL

- Load the dataset
- Transform data
- Splitting into training and testing sets
- Selecting machine learning alogorithms
- Parameter tuning and selection
- Fit train and test test into GridSearchCV and train the dataset

MODEL EVALUATION:

- Check accuracy for each model
- Plot confusion matrix

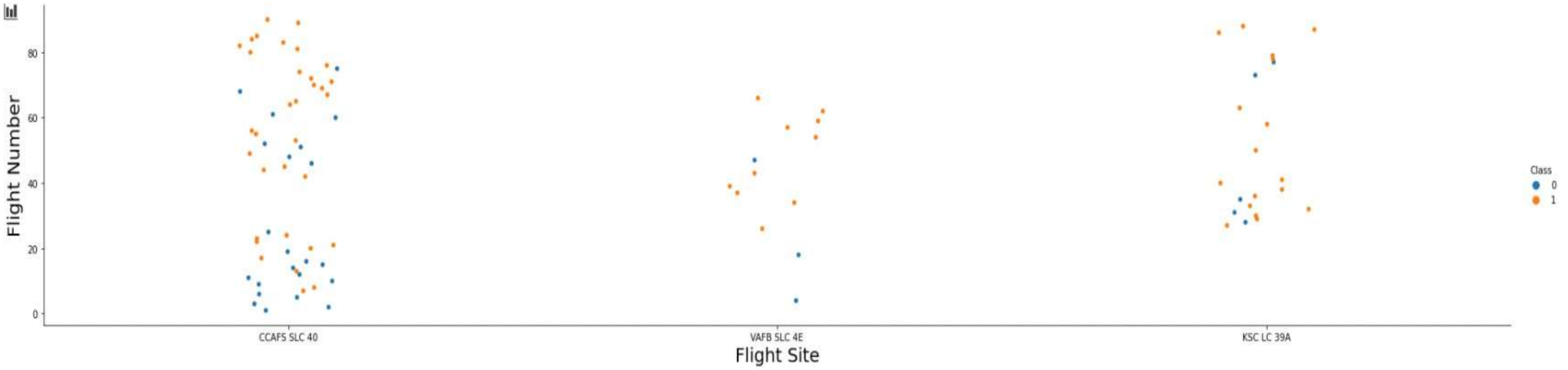
BEST CLASSIFICATION MODEL:

- Model with best accuracy is considered to be the best classification model for this problem.

EDA RESULTS

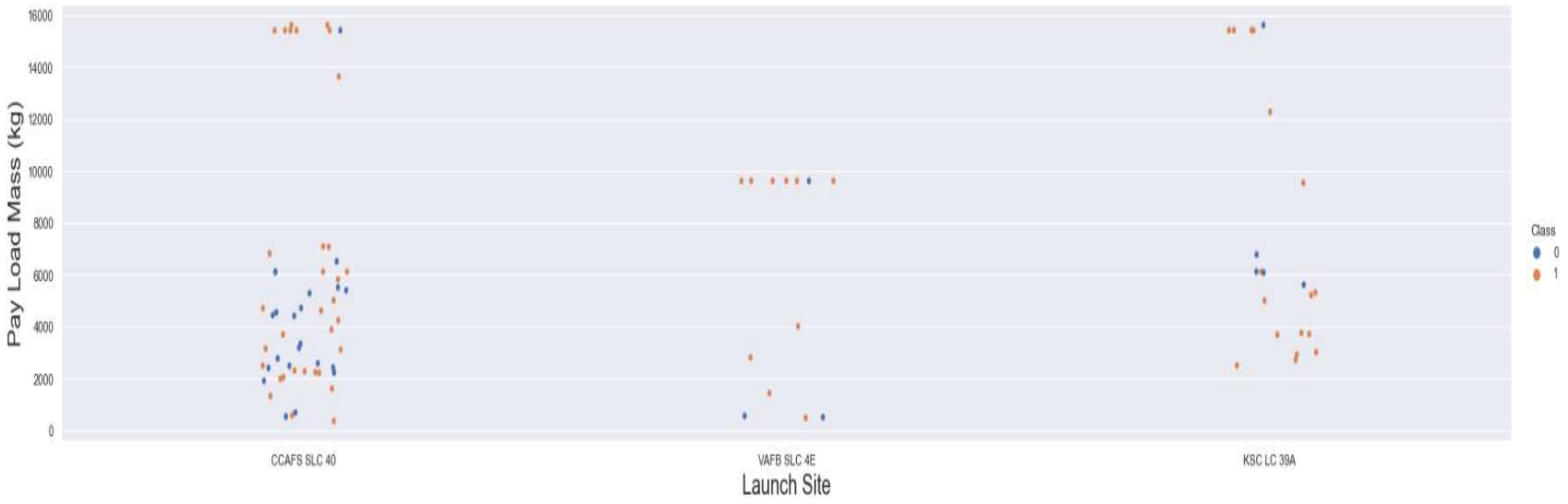
FLIGHT NUMBER VS LAUNCH SITE:

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



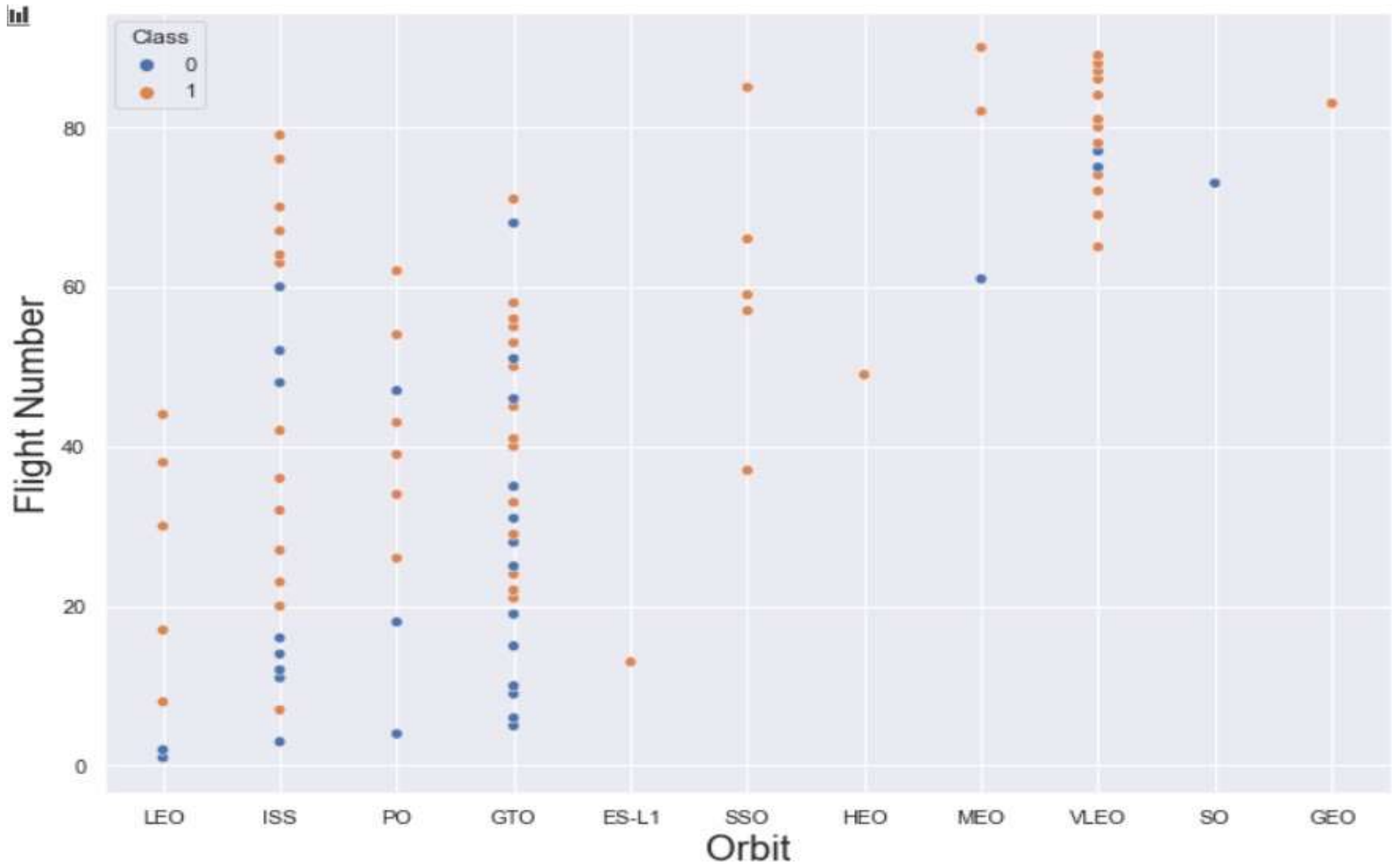
PAYLOAD MASS VS LAUNCH SITE

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



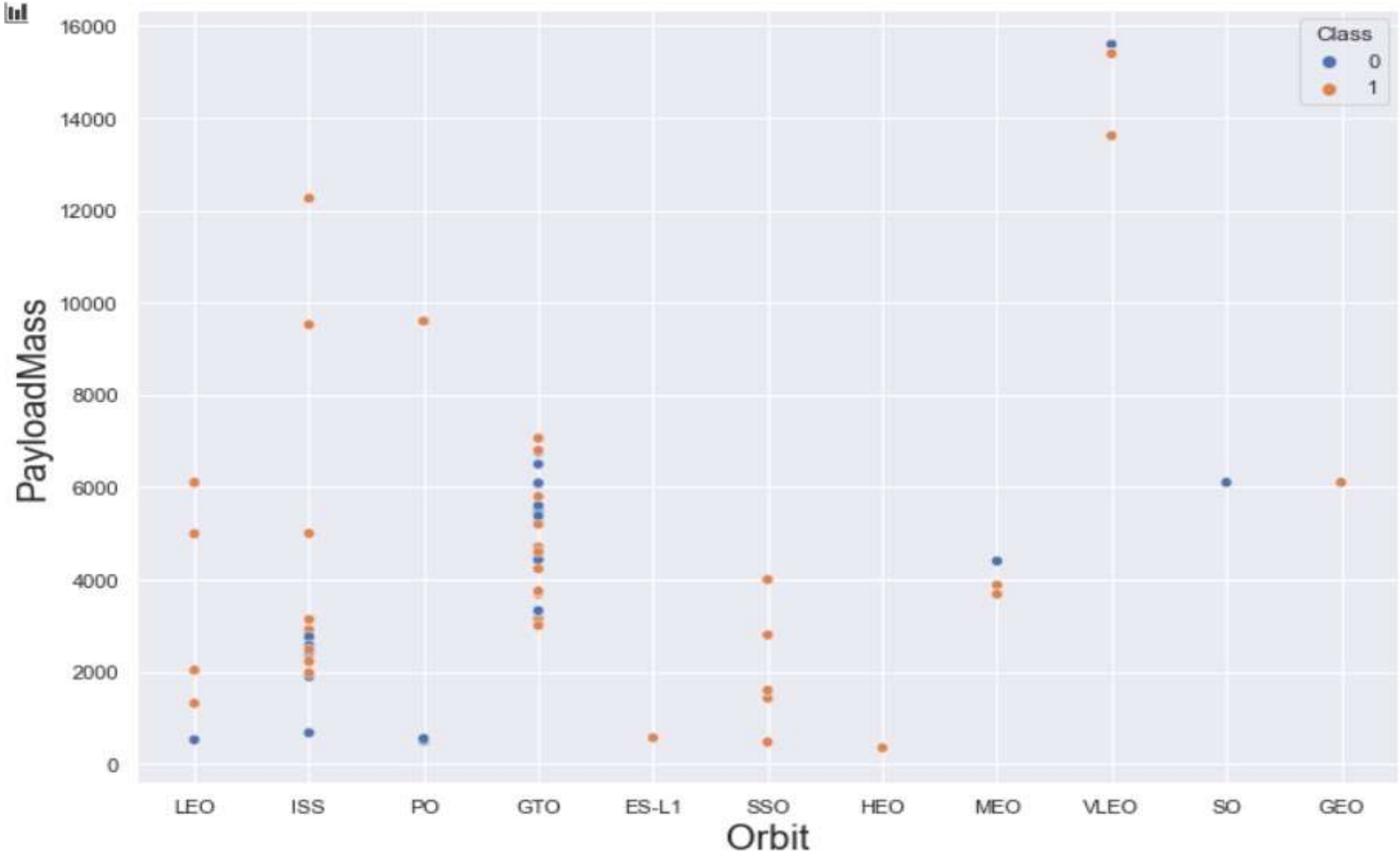
FLIGHT NUMBER VS ORBIT TYPE

in LEO orbit the Success is related to the number of flights



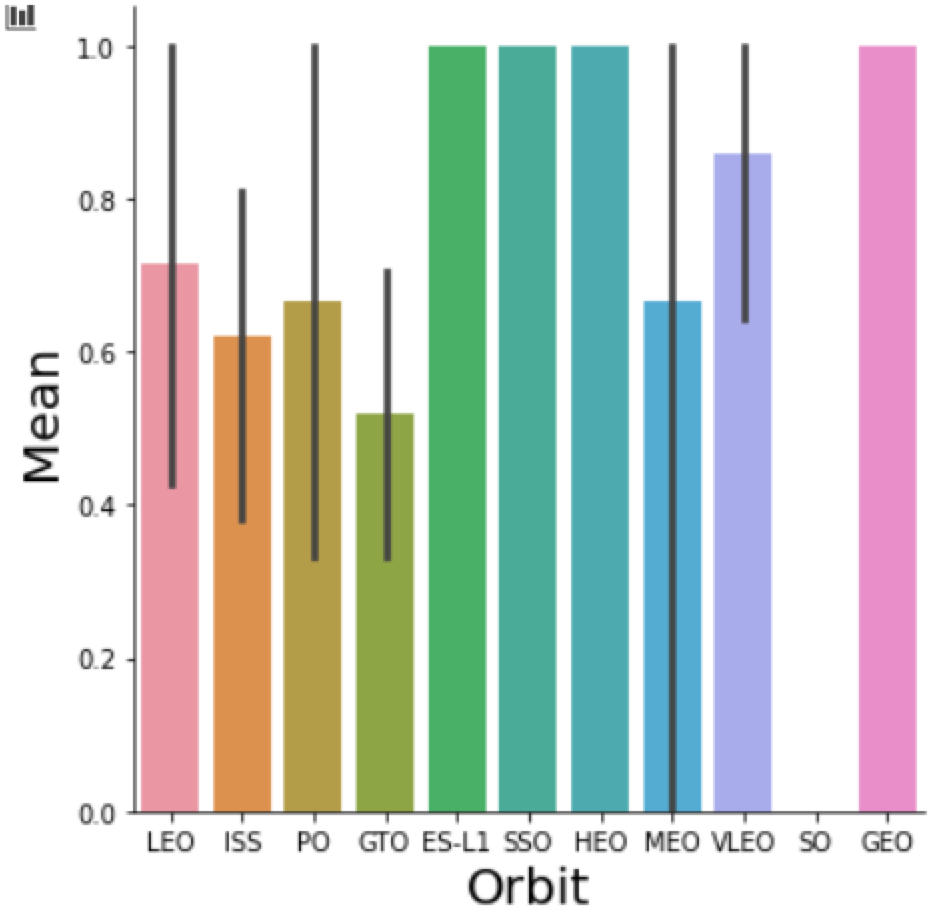
PAYLOAD VS ORBIT TYPE

Payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits



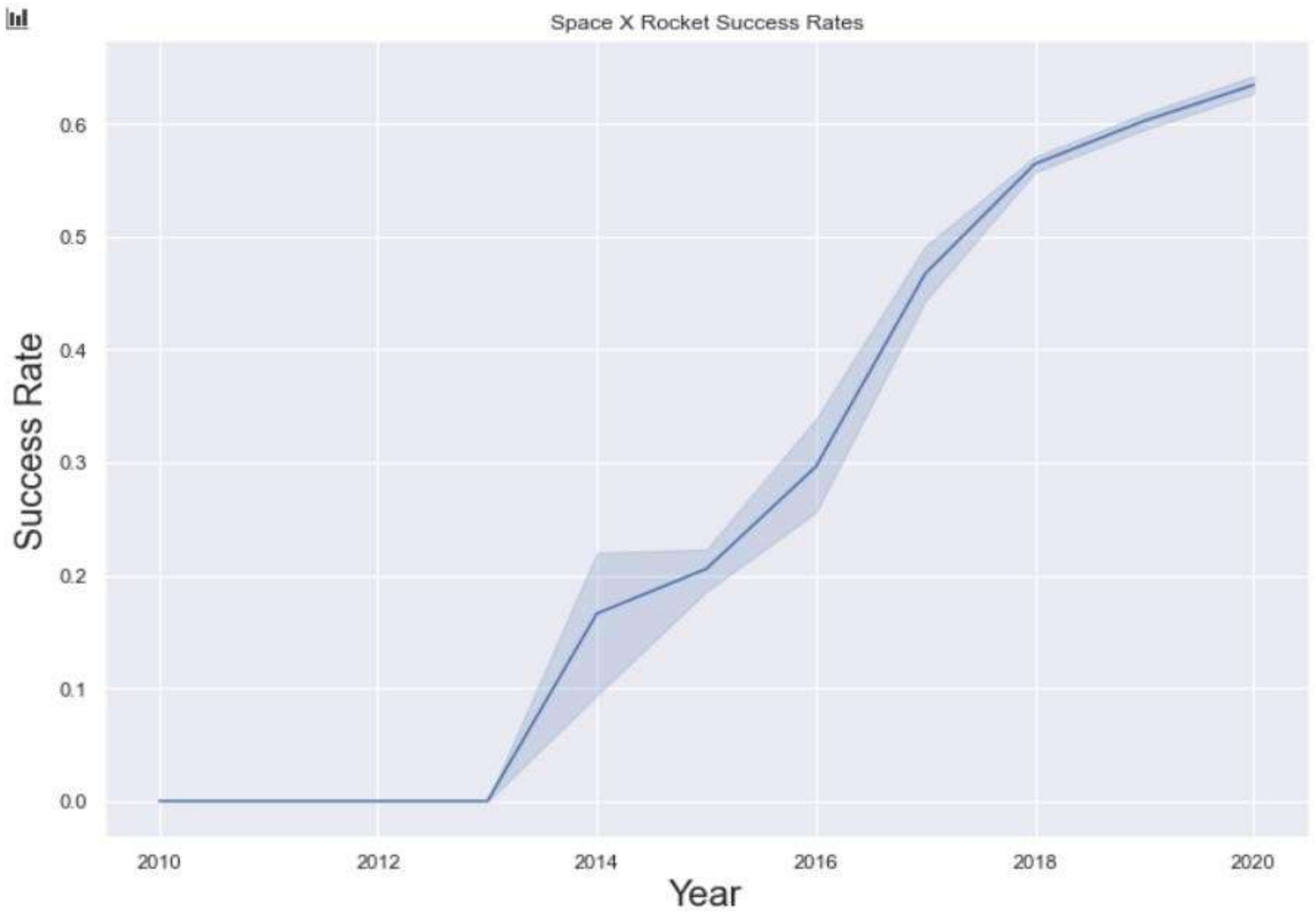
SUCCESS RATE VS ORBIT TYPE

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



LAUNCH SUCCESS YEARLY TREND

Success rate since 2013 kept increasing till 2020



EDA SQL RESULTS

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

```
output1 = sqldf("select DISTINCT Launch_Site from df")
output1
```

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

Launch Site Names Begin with 'CCA'

```
output2 = sqldf("select * from df WHERE Launch_Site LIKE 'CCA%'")
output2.head(5)
```

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-06-05	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	2010-06-06	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2010-06-07	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2010-06-08	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
output3= sqldf("select SUM(PAYLOAD_MASS__KG_) TotalPayloadMass from df where Customer = 'NASA (CRS)'")
output3
```

	TotalPayloadMass
0	45596

Average Payload Mass by F9 v1.1

```
[15] output4=sqldf("select AVG(PAYLOAD_MASS__KG_) from df where Booster_Version = 'F9 v1.1'")
     output4
```

	AVG(PAYLOAD_MASS__KG_)
0	2928.4

First Successful Ground Landing Date

```
[17] output5 = sqldf("select MIN(Date) from df where Landing_Outcome  = 'Success (ground pad)'")
     output5
```

	MIN(Date)
0	2010-06-23

Successful Drone Ship Landing with Payload between 4000 and 6000

```
[18] output6 = sqldf("select Booster_Version from df where Landing_Outcome = 'Success (drone ship)' AND Payload_MASS__KG_ > 4000 AND Payload_MASS__KG_ < 6000")
     output6
```

	Booster_Version
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

```
output7 = sqldf("SELECT(SELECT Count(Mission_Outcome) from df where Mission_Outcome LIKE '%Success%'),(SELECT Count(Mission_Outcome)
from df where Mission_Outcome LIKE '%Failure%')")
output7
```

	Successful_Mission_Outcomes	Failure_Mission_Outcomes
0	100	1

Boosters Carried Maximum Payload

```
sqldf("SELECT DISTINCT Booster_Version, MAX(PAYLOAD_MASS__KG_) AS [Maximum Payload Mass] FROM df GROUP BY Booster_Version ORDER BY [Maximum Payload Mass] DESC")
```

	Booster_Version	Maximum Payload Mass
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
...	...	...
92	F9 v1.1 B1003	500
93	F9 FT B1038.1	475
94	F9 B4 B1045.1	362
95	F9 v1.0 B0003	0
96	F9 v1.0 B0004	0

97 rows × 2 columns

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

```
sqldf("SELECT COUNT(Landing_Outcome) FROM df WHERE (Landing_Outcome LIKE '%Success%') OR (Landing_Outcome LIKE '%Failure%') AND (Date >'04-06-2010') AND (Date <
```

	COUNT(Landing_Outcome)
0	61

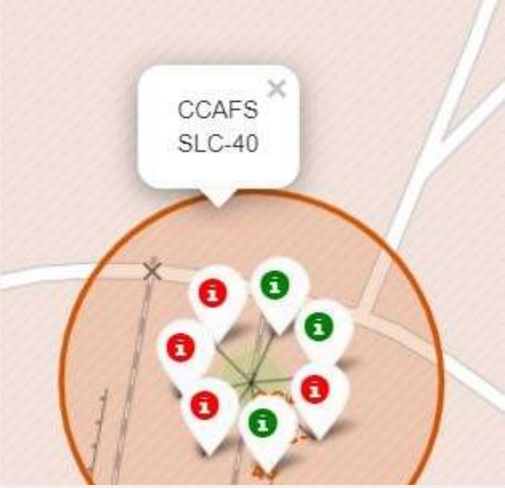
Section 4

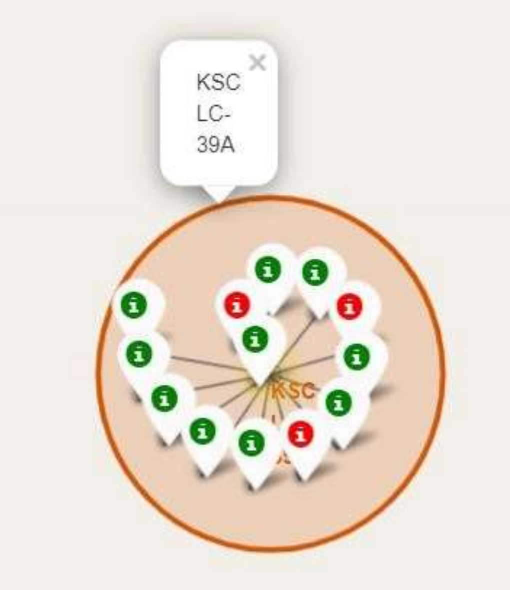
Launch Sites Proximities Analysis

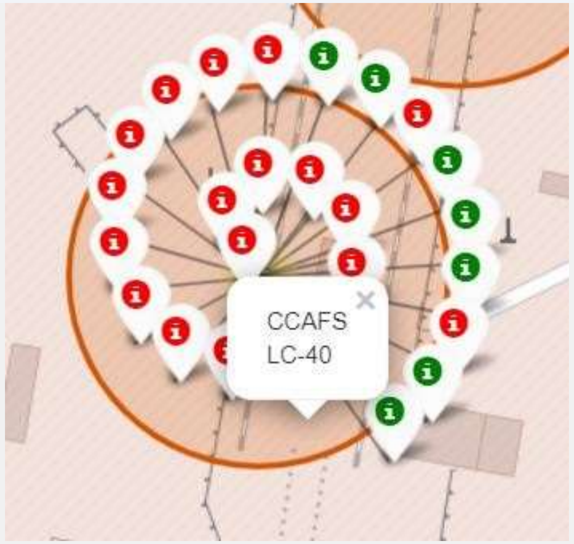
<Folium Map Screenshot 1>

SpaceX launch sites are in the United States of America coasts that is, Florida and California

<Folium Map Screenshot 2>





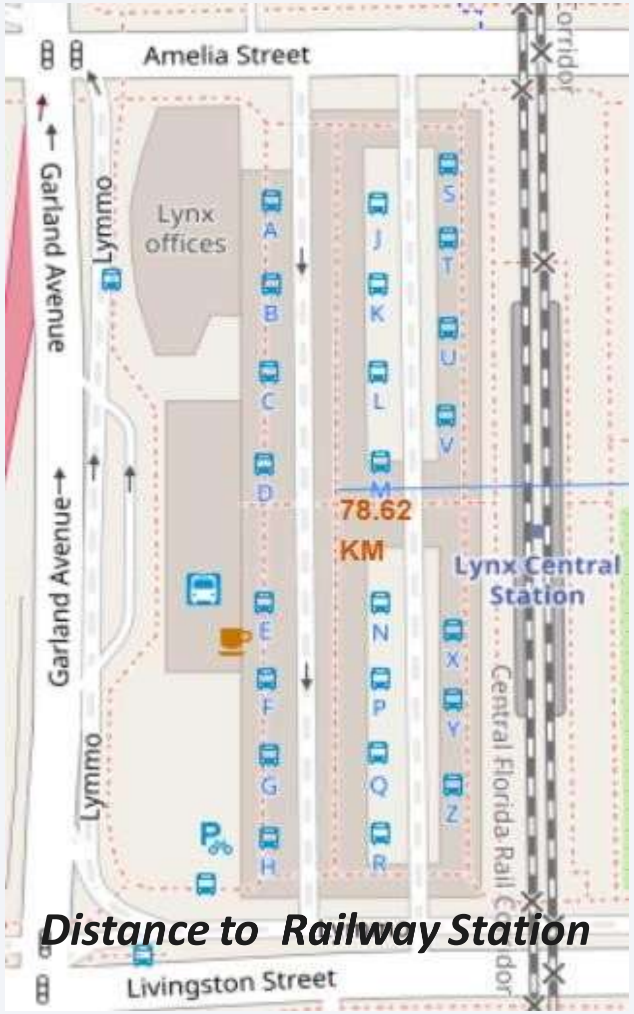


Florida Launch Sites

Green Marker: success
Red Marker : Failures

California Launch Site

<Folium Map Screenshot 3>



Distance to Railway Station

Distance to Coast

Distance to closest highway

Distance to Coastline

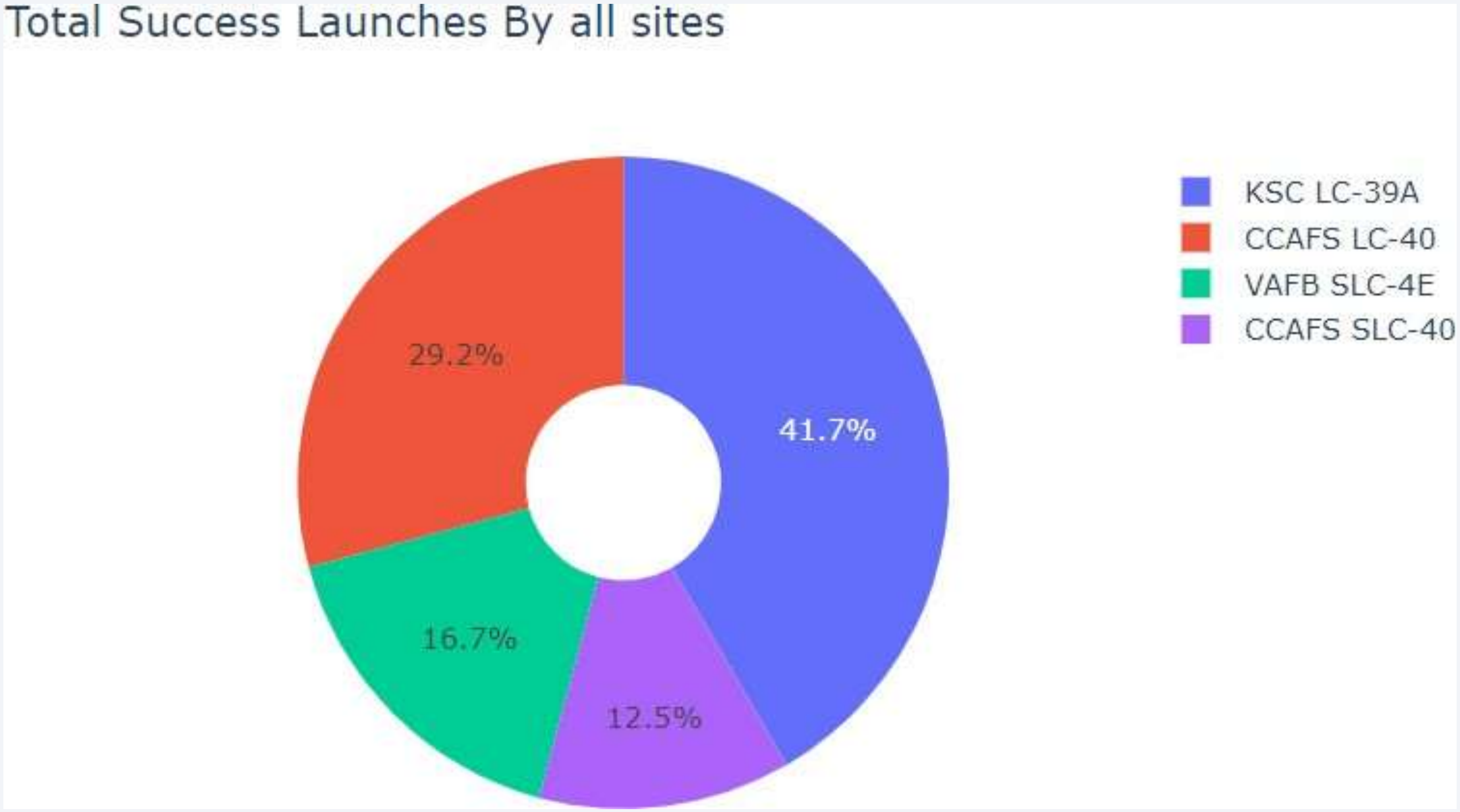
Distance to city

- 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 5

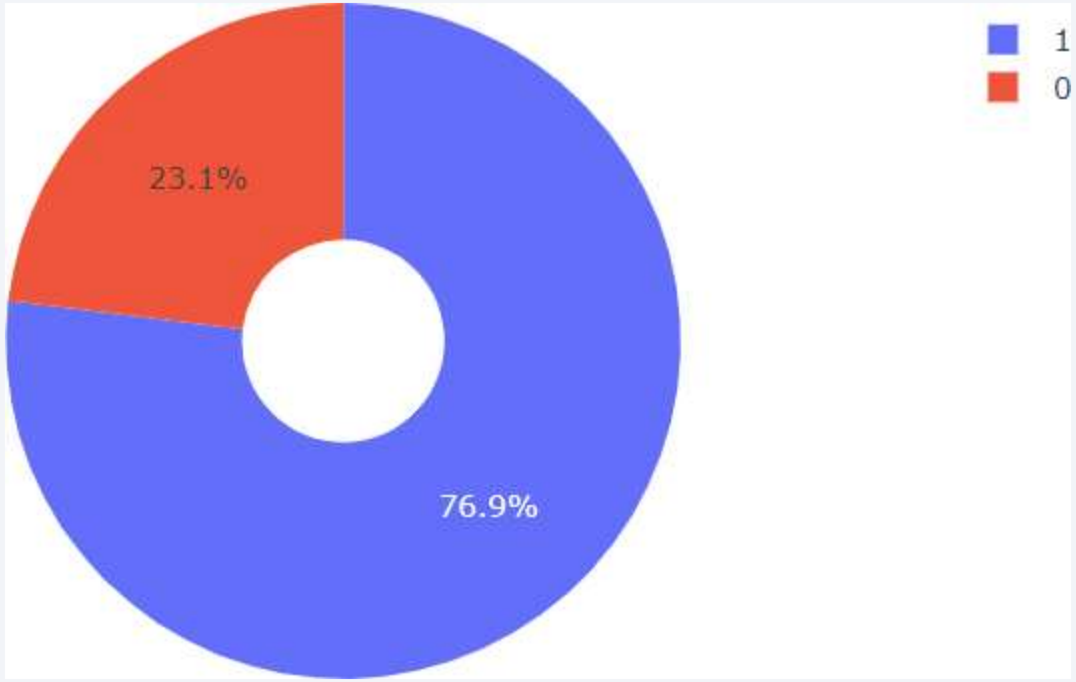
Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>



KSC LC-39A had the most successful launches from all the sites

<Dashboard Screenshot 2>



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

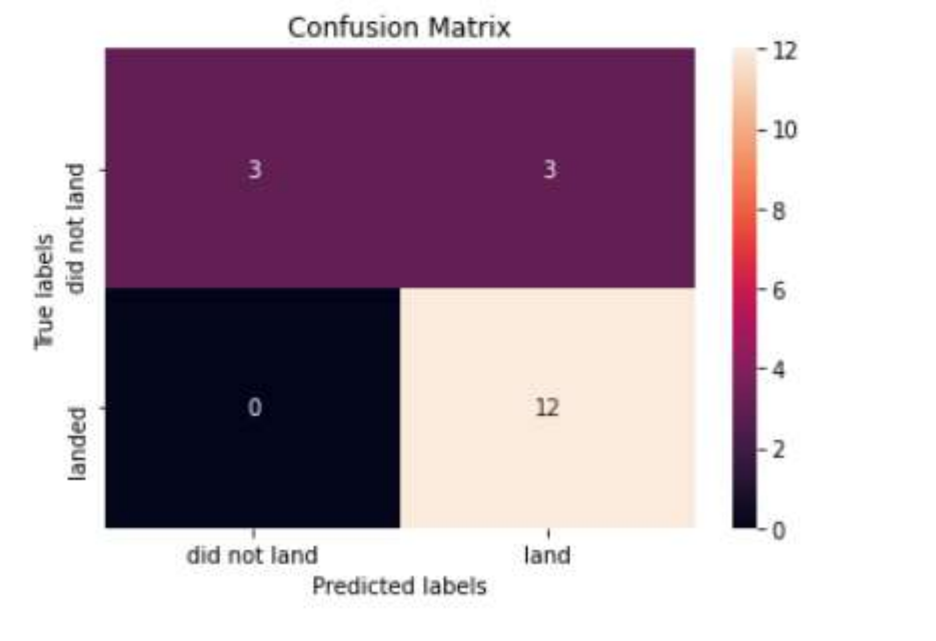
Section 6

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 0.8767857142857143
Best Params is : {'criterion': 'entropy', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 2,
'min_samples_split': 10, 'splitter': 'random'}
```



After selecting the best hyperparameters for the decision tree classifier using the validation data, we achieved 88.92% accuracy on the test data.

Tree can distinguish between the different classes.

Conclusions

- The Tree Classifier is the best for Machine Learning algorithm for this dataset
- Low weighted payloads perform better than the heavier payloads
- The success rates for SpaceX launches is directly proportional time
- KSC LC-39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Thank you!