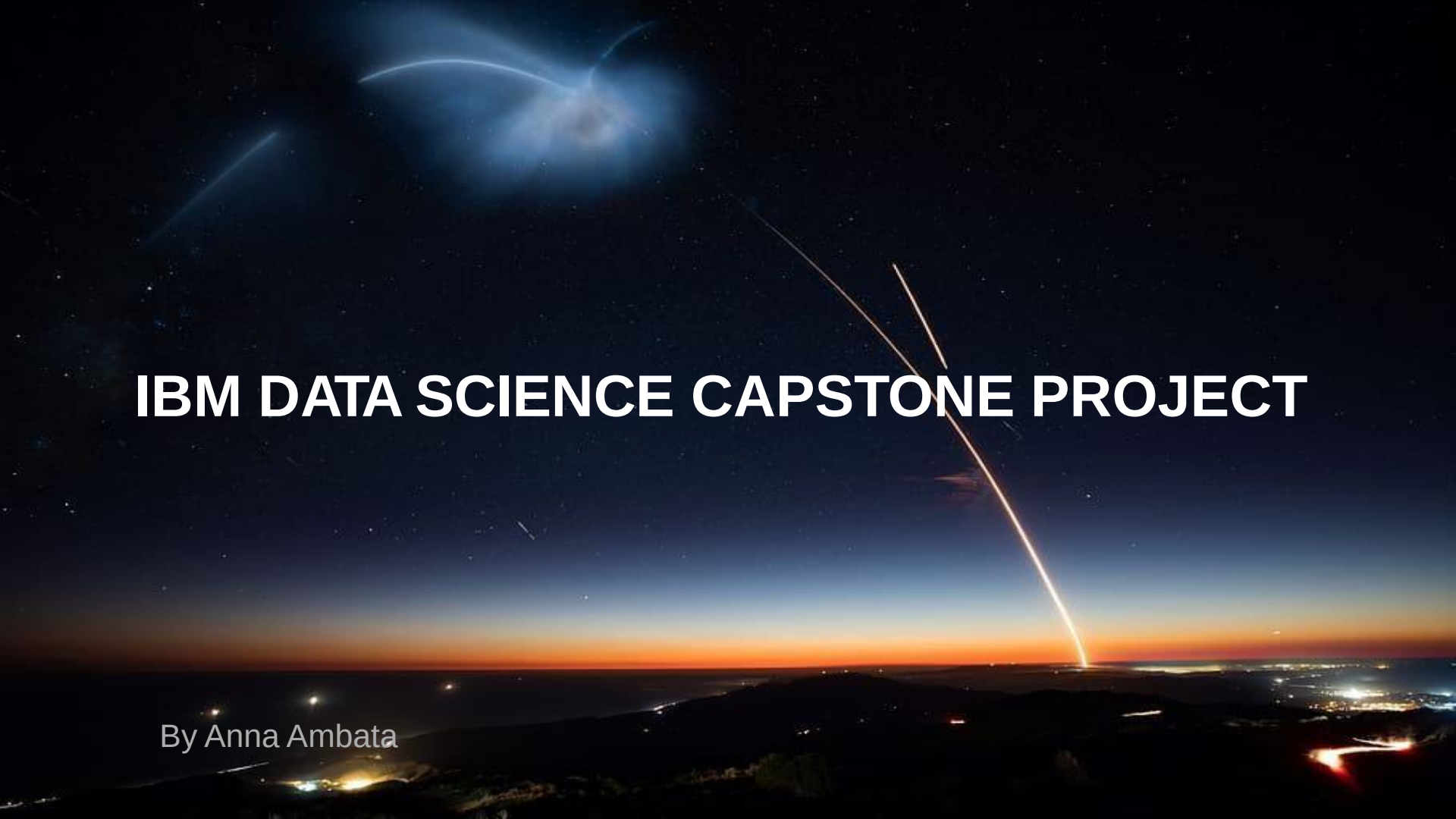


IBM DATA SCIENCE CAPSTONE PROJECT

By Anna Ambata

OUTLINE

1. EXECUTIVE SUMMARY
2. INTRODUCTION
3. METHODOLOGY
4. RESULTS
5. CONCLUSION
6. APPENDIX

1. EXECUTIVE SUMMARY

The In this project, we use the SpaceX REST API and web scraping techniques to collect data on past SpaceX Falcon 9 launches. The dataset contains valuable information about rockets, payloads, landing outcomes, and more. We aim to clean and analyze this data, create visualizations, and build a predictive machine learning model to predict the likelihood of rocket stage reuse and estimate the cost of each launch.

Methodologies Summary

Data collection	Link to Notebook
Data Wrangling	Link to Notebook
EDA Data visualization	Link to Notebook
EDA with SQL	Link to Notebook
Predictive Analysis	Link to Notebook
Interactive Map With Folium	Link to Notebook
Interactive Dashboard Using Plotly	Link to Notebook

2. INTRODUCTION

Space Y is a new aerospace company looking to compete with SpaceX in the commercial rocket launch industry. The project focuses on collecting and analyzing SpaceX's Falcon 9 launch data to predict the likelihood of rocket stage reuse and estimate the cost of each launch. Data is gathered using the SpaceX API and web scraping techniques to create a comprehensive dataset for analysis and machine learning modelling.

The primary objective:

1. **Predicting rocket stage reuse:** Falcon 9 rockets are designed for reusability, a key factor in reducing launch costs. We aim to predict whether SpaceX will reuse the first stage of the rocket based on historical launch data, such as payload weight, launch site, orbit type, and booster status.

2. **Estimating launch costs:** Estimating the price of each SpaceX Falcon 9 launch will help *Space Y* benchmark its own cost structure and remain competitive in the market.

3. METHODOLOGY

To achieve the project goals of predicting rocket stage reuse and estimating launch costs, the following methodology is applied:

1. Data collection
2. Data Wrangling
3. EDA Data visualization
4. EDA with SQL
5. Predictive Analysis
6. Interactive Map With Folium
7. Interactive Dashboard Using Plotly

Data collection

Using Space X API

1.API Endpoint: We gather SpaceX launch data from the endpoint `api.spacexdata.com/v4/launches/past`.

2.Requesting Data: Using the `requests` library, we perform a GET request to retrieve the past launch data.

3.JSON Parsing: The API response is a JSON list of launch objects. We parse the data using `.json()` and then normalize the JSON structure into a flat table using the `json_normalize()` function.

4.Handling Rocket IDs: Some columns, like the rocket column, contain IDs instead of descriptive data. We use additional API endpoints (e.g., `/rockets`, `/launchpads`) to retrieve relevant details.

5.Filtering Data: We filter the dataset to include only Falcon 9 launches and exclude Falcon 1 launches.

6.Handling Null Values: Missing values in the `PayloadMass` column are replaced with the calculated mean, while NULL values in `LandingPad` are left for later handling with one-hot encoding.

7.The cleaned dataset is **exported to a CSV file** for further analysis

REST API

```
json_data=requests.get(static_json_url).json()
```

Make request

```
# Use json_normalize meethod to convert the json result
data=pd.json_normalize(json_data)
```

Normalize to df

Filter Falcon 9 only

```
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}
```

Create a dictionary for creating a dataframe from the dataset collected

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9=df_launch[df_launch['BoosterVersion']!='Falcon 1']
```

Save to CSV

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

Wl

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

Get content of the wiki

Loop Through and add 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
```

Create a Dataframe for the important columns

Loop through the request content and extract data

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

Save to cvs

[Link to Notebook]

4.DATA WRANGLING

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, In 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.

Calculate the number of launches on each site

⇨ Save to CSV

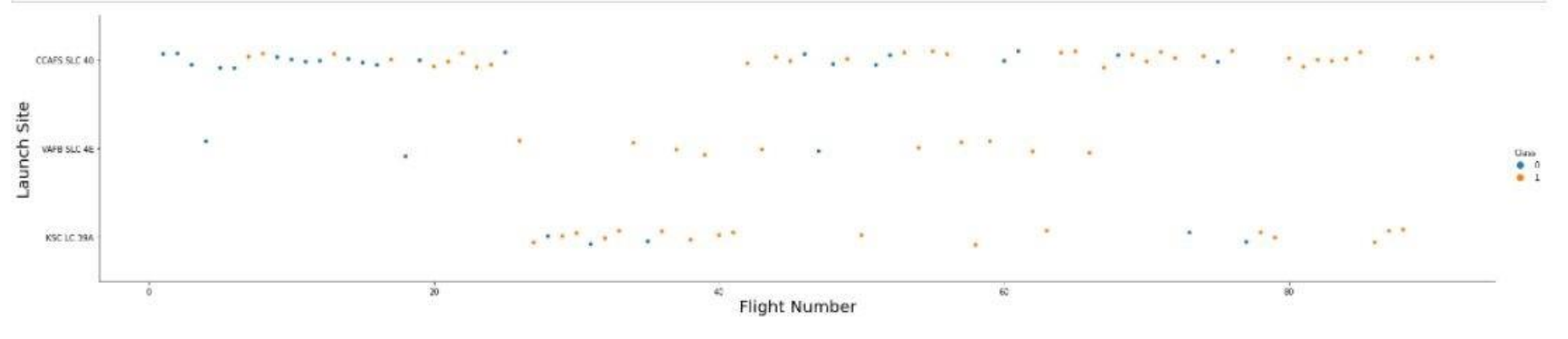
```
df.isnull().sum()/df.count()*100
```

EXPLORATORY DATA ANALYSIS WITH VISUALIZATION

Through EDA on the data from APi and Wiki,we will find some insights on :

- **Flight number & Launch Sites-Visualizing the launch from every site .**
- **Payload & Launch Sites-Payload launch from sites**
- **Success rate & Orbit type-Success rate compared to the orbit type**
- **Flight number & Orbit Type -Type of orbit for each launch**
- **Payload & Orbit type -Payload and the orbit .**
- **Trend of success rate-Trend of the success rate over the years .**

Flight number & Launch Sites



From the Visualization we can concluded that:

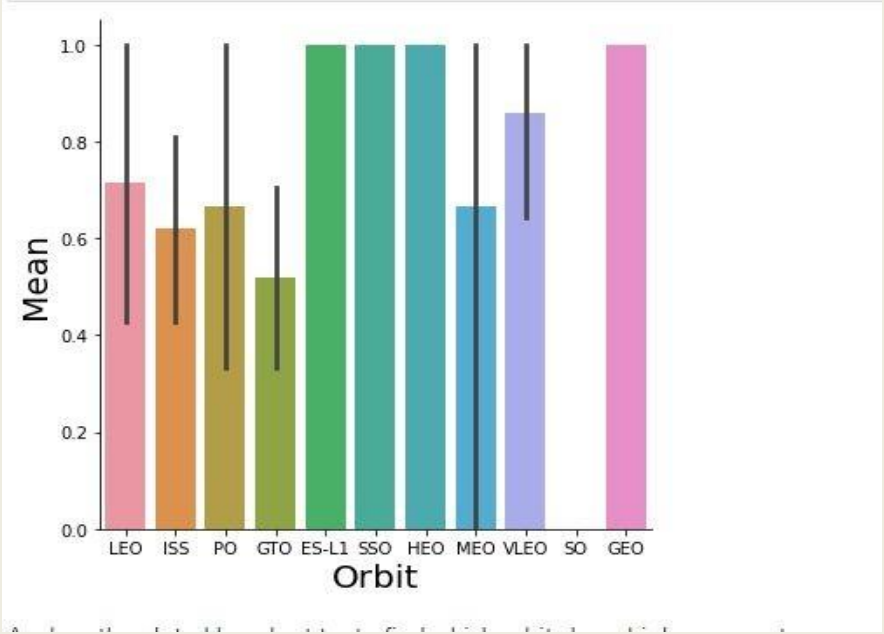
- Earlier flights lauch were from CCAFS-SLC-40 site ,Followed by KSC-LC-39A
- Most Launches are Launched from CCAFS-SLC-40
- Fewer Launches from **VAFB SLC 4E site**

Payload & Launch Sites

From the Visualization we can concluded that:

- VAFB SLC 4E has Low Payload launches
- CCAFS SLC 40 has more Higher Payload Launches and Low Payload Lauches .

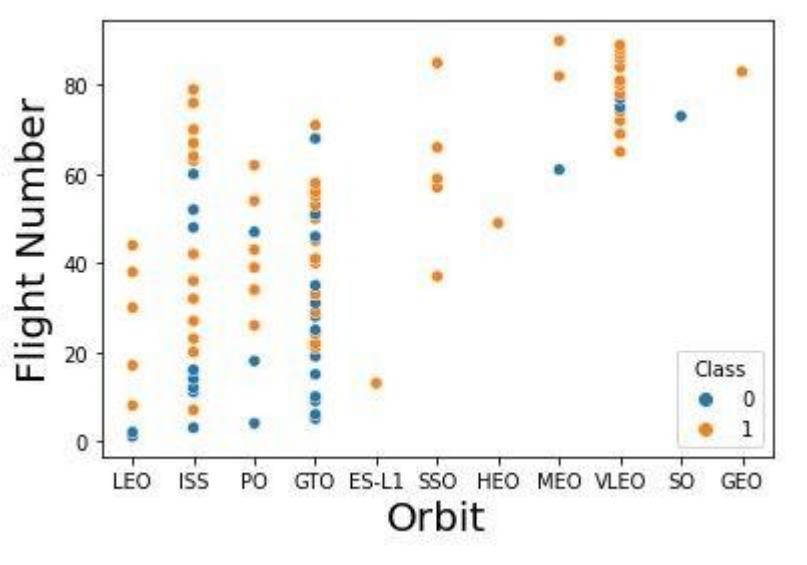
Orbit Success



From the Visualization we can concluded that:

- GEO,HEO & ES-L1,SS) have high success rate .

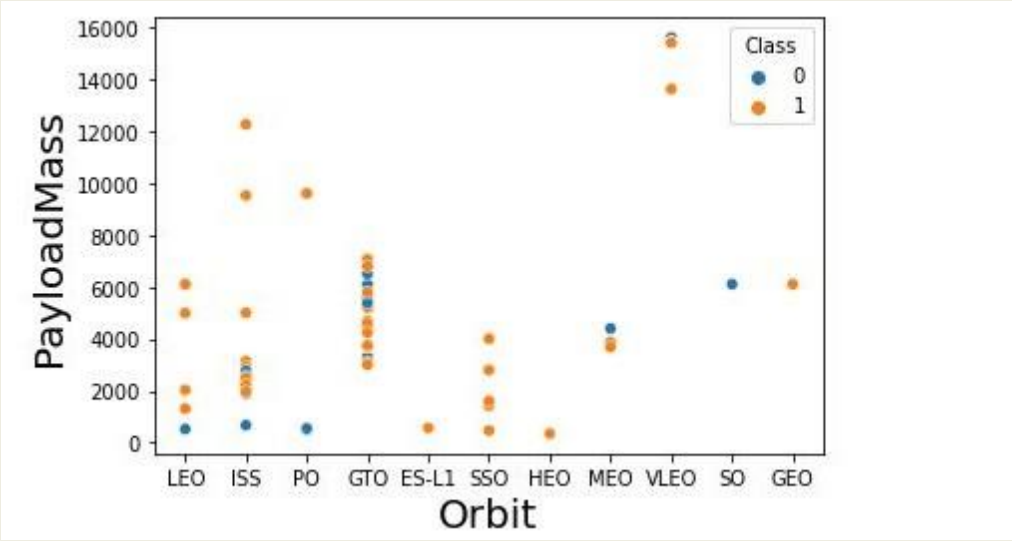
Flight No & Orbit



From the Visualization we can concluded that:

- Most Flight are to ISS,PO,GTO and VLEO
- MOST fails are for ISS,GTO
- SSO & VLEO has high success rate .

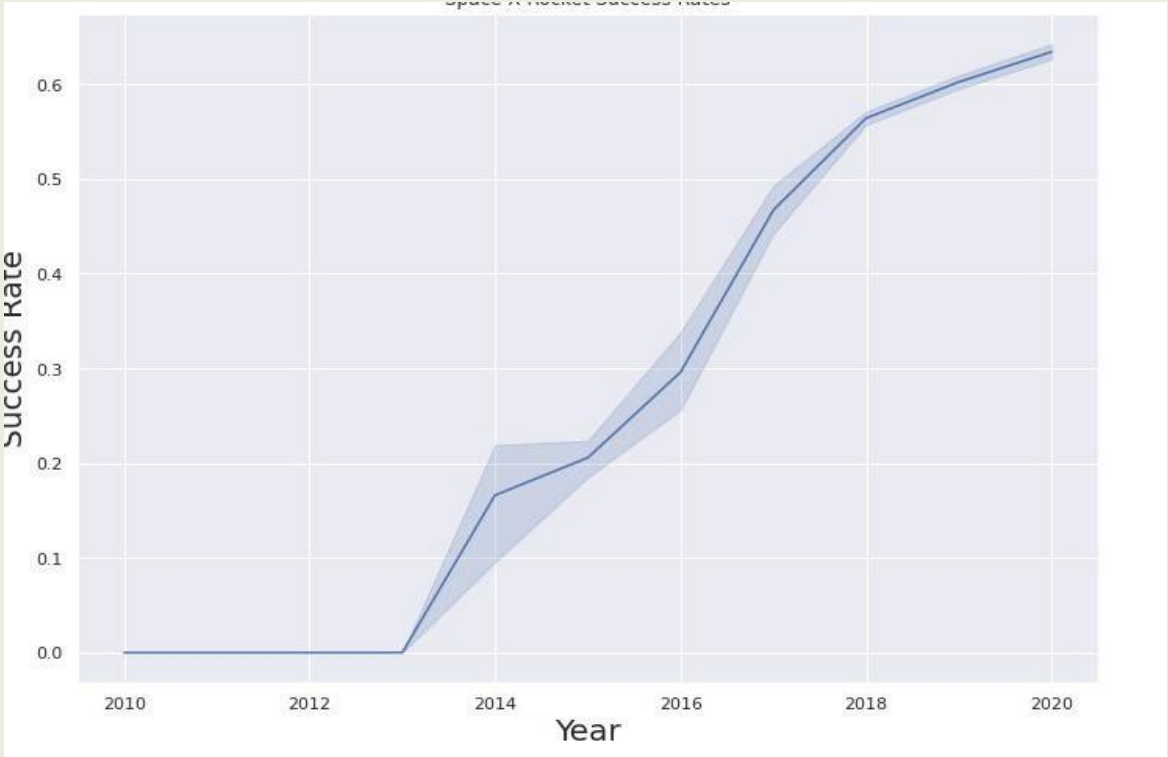
Payload & Orbit



From the Visualization we can concluded that:

- Higher Payload are to the VLEO
- Least Payload are for HEO,ISS,PO,ES-L1
- GTO has average payload size .

Success Rate Trend



The rate of success of the launches increase over time since to the data collected from the previous fails and success launches .

Exploratory Data Analysis With Sql

Exploratory Data Analysis on the follow criteria:

Unique Sites

Max Payload

Average Payload

Day when First Success Landing

Success and Failures count

Boosters With Max Payload

EDA With SQL

For the categories above we find that :

Sites that SpaceX operates in are:

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

Max Payload:48213

Average Payload for all Launches: 2928 Kgs

First Success Landing was Made on:06/05/2016

Booster Version that carry over 4000 kg and 6000 Kg :

F9 FT B1020,F9 FT B1022,F9 FT B1026,F9 FT B1021.2,F9 FT B1031.2

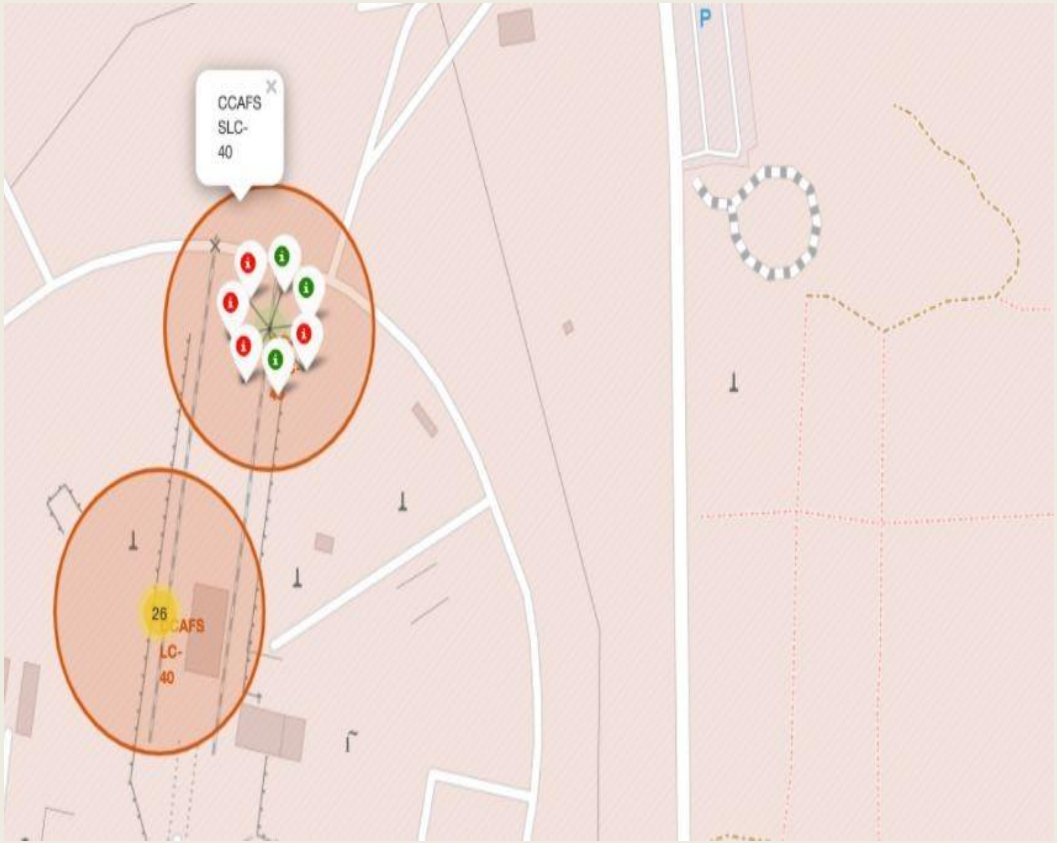
INTERACTIVE MAP WITH FOLIUM

Visualization of the launches for every site and every launch in a Interactive Map

Visualization of:

- Launch Sites
- Visualize the launches on the map base on Fail or Success

Visualize the Launches on Map



Data Set Contained 3 Separate Launch Sites that are displayed on the picture on the left .

This gives us insights to the launches success and failures .

3. PREDICTIVE ANALYSIS (CLASSIFICATION)

Through this model, tuned for best performance we go the insights on the probability if a launch being success or a failures.

GridSearchCV with cross-validation (cv=10) is applied to optimize hyperparameters for several machine learning algorithms:

- Logistic Regression (LogisticRegression())
- Support Vector Machine (SVC())
- Decision Tree (DecisionTreeClassifier())
- K-Nearest Neighbor (KNeighborsClassifier())

INTERACTIVE WITH DASH

Visualization of the Launches from Site in Dashboard

Visualization of:

- Success Launch Launch Sites
- Visualize payload from different sites with rangeSlider

  for interacting with the plot .

After Analyzing all the Models,the KNN was the best Model with accuracy of 77% and best Score of 87%

```
parameters = {'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
              'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
              'p': [1,2]}

KNN = KNeighborsClassifier()
gscv=GridSearchCV(KNN,parameters,scoring="accuracy",cv=10)
KNN_cv=gscv.fit(X_train,y_train)
```

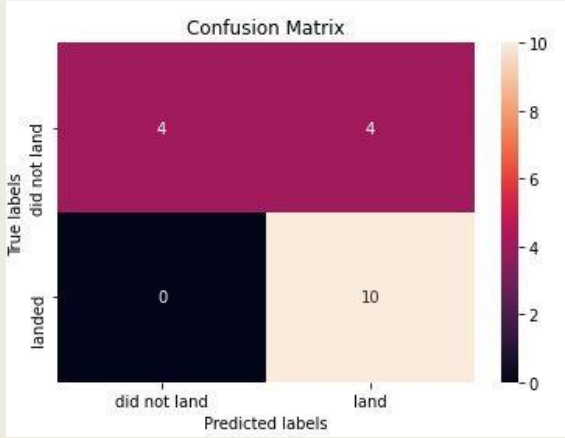
```
print("Accuracy",KNN_cv.score(X_test,y_test))
```

```
Accuracy 0.7777777777777778
```

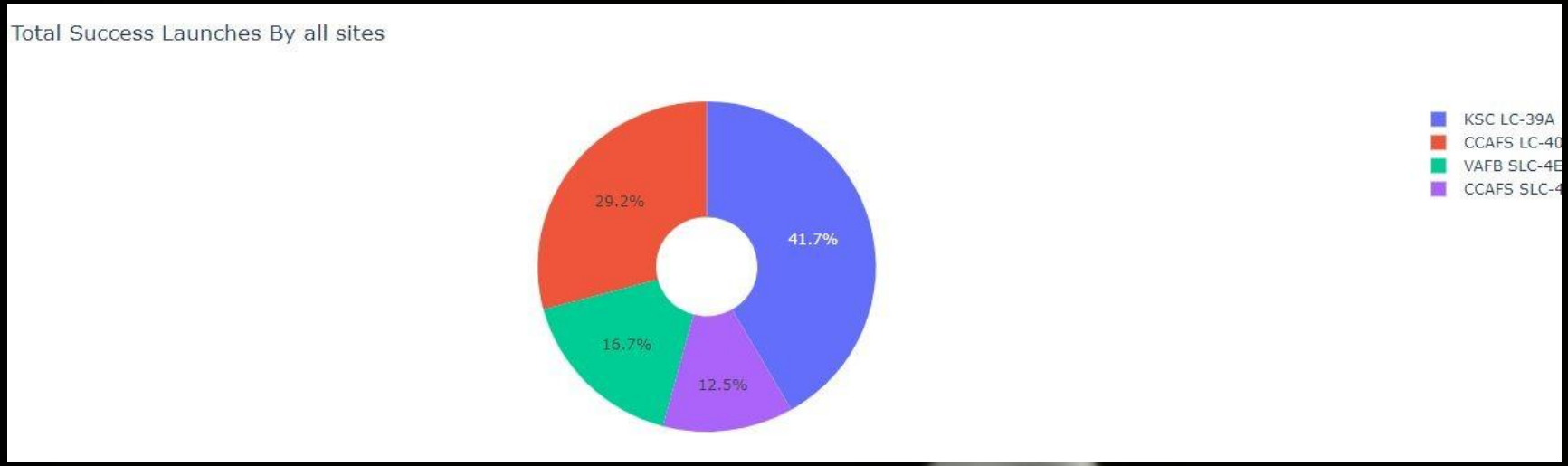
```
print("tuned hpyerparameters :(best parameters) ",KNN_cv.best_params_)
print("accuracy :",KNN_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 4, 'p': 1}
accuracy : 0.8767857142857143
```

```
yhat = KNN_cv.predict(X_test)
plot_confusion_matrix(y_test,yhat)
```



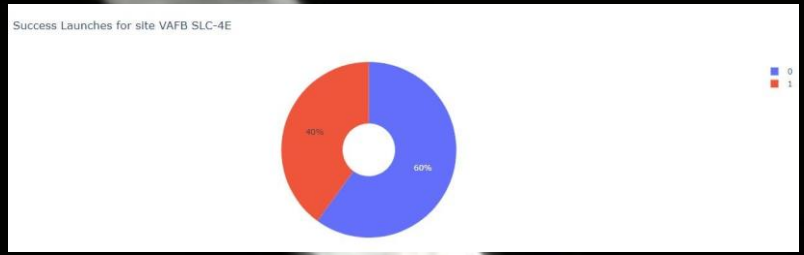
True Positives



Observation is that KSC has more launches compared to other sites

.

Using the drop down on the dashboard it’s possible to view single site launches



Using the range slider we can view the sites that failed and succeed for each booster version and the Payload they were carrying .