IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

RAHUL ABHISHEK
18/07/2023

OUTLINE

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

EXECUTIVE SUMMARY

Summary of methodology

- Data collection through API
- Data collection through web scrapping
- Data wrangling
- Exploratory data analysis with SQL
- Exploratory data analysis with data visualization
- Interactive Visual Analytics (map) with Folium
- Model building (classification algo)

INTRODUCTION

SpaceX is the only company which has brough down the price of rocket launching by nearly 62 % when compared with other companies. SpaceX offers the launching price of the rocket for about 62 million dollar where other giant companies offer 165 million dollar for the same and SpaceX manage to reduce the launching price of the rocket by continuously reusing the first stage ,

Now being a data scientist on the rival company of SpaceX my job is to build the machine learning model which can predict the optimum launching price of the Falcon 9 rocket using which we can bid with our competitor company SpaceX

Problem involves :

1. Finding the useful insights from the data obtain like what are the factor responsible for the success and failure launch of the rocket
2. What is the relationship among various feature with respect to the outcome feature or target feature
3. What are the parameter that has to be consider which best define the effective cost of the rocket launch

Section 1

Methodology

METHODOLOGY

EXECUTIVE SUMMARY :

1. DATA COLLECTION (using SpaceX Rest API and Web Scraping using BeautifulSoup)
2. Data Wrangling
3. Exploratory Data Analysis using Visualization and SQL
4. Creating interactive dashboard using python Plotly and interactive visual of map using Folium
5. Model building for final prediction using classification algorithm namely (KNN, SVM. Logreg, Decission Tree)
6. Finding best model among these

DATA COLLECTION

Data collection : it is the process of gathering, recording data form various sources as per the requirement of the project. Since data are coming from various sources so we have to be careful while collecting data that it is relevant or not or it can be able to answer our questions or not because entire data science project depends on the source ie data so it play vital role in the data science project

Since in our project we are collecting data of spacex launch using Rest Api.

1. First we get the url of the source
2. Then we used request method followed by get () in order to fetch the data.
3. Then we used json() method as the response we get we have to convert in some suitable readable format followed by this we have used normalize_json() method to convert the content we get into pandas data frame
4. Since the data we obtained was containing other rocket as well but we are interested in falcon 9 and in some columns not entire in order to proceed further so we filter our data to get only valuable data from the big content
5. Now once this process is done we moved on to the process called data wrangling where we removed missing values from the data set

In second part we used Web Scraping technique to fetch the data from wiki

1. url of the relevant source
2. Used request.get() method to web scrape the data which we need
3. Then we created beautiful soup object by name soup and using that we convert the html file to content along with this we also used html parser method to get the desire result in content form so we can extract it useful data from it
4. Then we converted it to a data frame

DATA COLLECTION USING REST API

Get Request for Rocket Launch Data Using SpaceX Rest Api

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
print(response.content)
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/
response.status_code
```

Json normalize to convert the content into pandas data frame

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

Data cleaning removing missing values and filtering data to get falcon 9 result only

```
# 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 ha
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_falcon9=df[df['BoosterVersion']=='Falcon 9']

data_falcon9.isnull().sum()

# Calculate the mean value of PayloadMass column
mean_payload=data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass']=data_falcon9['PayloadMass'].replace(np.nan,mean_payload)
```

[link of the code]

DATA COLLECTION USING WEB SCRAPING

Request falcon9 launch data from wiki via url

Created BeautifulSoup object soup using that we scrap wiki data and use html parse to convert it into readable text document

further we extracted data using loop

link of code

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

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

```
# Use soup.title attribute
print(soup.title)
```

```
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
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            launch_dict['Date'].append(date)
            #print(date)

            # Time value
            # TODO: Append the time into launch_dict with key `Time`
```

DATA WRANGLING

Data Wrangling: Data wrangling is the process of cleaning, organizing and transforming raw data into desired format . Data wrangling enables businesses to tackle more complex data in less time and produce more accurate results and make better decision.

In the falcon 9 data set we performed following steps :

1. Tackle missing value
2. Then we calculate number of launches on each site
3. Then we calculate numbers and occurrence of each orbit
4. Then we calculate the number and occurrence of mission outcome per orbit type
5. Then we created landing outcome label from outcome column
6. Then we export csv file

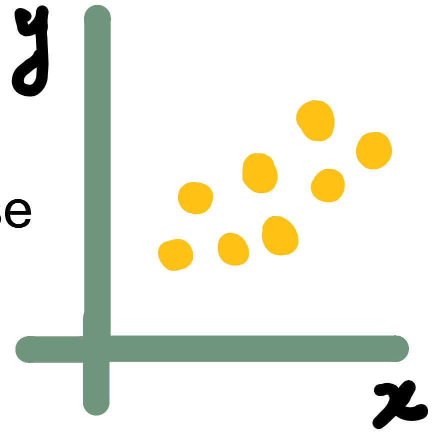
link of code :

EDA and VISUALIZATION

Scatter plot :

- Relation between flight number vs payload
- Relation between flight no vs launch site
- Relation between payload vs launch site
- Relation between success rate vs orbit
- Relation between flight no vs orbit
- Relation between payload vs orbit

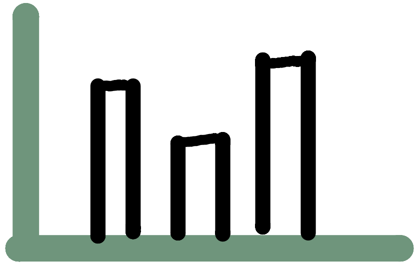
With the help of cat and scatter plot we understand the relationship between these variable basically there correlation



link of code :

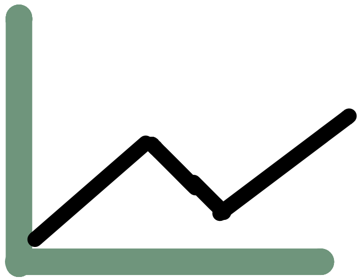
Bar graph :

Using bar graph understood success rate of each orbit



Line Chart : help us to understand the trend between the selected variables

Success rate vs yearly trend



EDA with SQL

Performed following SQL queries :

- Name of unique launch site
- Launch site begin with cca
- Total payload mass carried by booster launched by NASA
- Average payload mass carried by booster version v1.1
- First successful landing outcome in ground pad
- Name of the booster which have success in drone ship and have payload mass between 4000 and 6000
- Total number of successful and failure mission outcome
- names of the booster versions which have carried the maximum payload mass.
- records which will display the month names, failure landing outcomes in drone ship ,booster versions, launch site for the months in year 2015.
- 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.

link of code

INTERACTIVE MAP WITH FOLIUM

- Folium map object is a map centered on NASA Johnson Space Center at Houson, Texas
- Red circle at NASA Johnson Space Center's coordinate with label showing its name (folium.Circle, folium.map.Marker).
- Red circles at each launch site coordinates with label showing launch site name (folium.Circle, folium.map.Marker, folium.features.DivIcon).
- Points are grouped in a cluster to display multiple and different information for the same coordinates (folium.plugins.MarkerCluster).
- Markers show successful and unsuccessful landings. Green color marker for successful landing and Red color marker for unsuccessful landing. (folium.map.Marker, folium.Icon).
- Markers show distance between launch site to key locations (railway, highway, coastway, city) and plot a line between them. (folium.map.Marker, folium.PolyLine, folium.features.DivIcon)
- These objects are created in order to understand the problem and the data .using these map and markers now we can easily see all launch sites, their surroundings and the number of successful and unsuccessful landings.

link of code

DASH BOARD WITH PLOTLY

Launch site dropdown which includes all 4 launch site

Added callback function to render success pie chart based on selected site from dropdown

Added range slider to select payload

Added callback function to render the success payload scatter chart

At the end we found some valuable insight from the dash board such as

1. Which site has largest successful launch
2. Which site has the highest launch success rate
3. Which payload range has the highest launch success rate
4. Which payload range has the lowest success rate
5. Which F9 booster version such as v1.0,v1.1,FT,B4,B5 has the highest launch success rate

link of code

MODEL BUILDING (CLASSIFICATION ALGO)

Data preparation :

- Data is loaded to the system using pandas lib (csv) file
- Dependent variable [‘class’] is being converted to NumPy array
- Independent variable or features x is being standardize in order to bring all the feature to similar scale
- Splitting the data using train test split method

Model preparation:

- Various classification algorithm were selected such as SVM, KNN, Logreg, Decision Tree
- Using gridsearchcv hyper parameter were tune for each model and best parameter were selected
- All model were trained on the training data set using hyper parameter

Model Evaluation:

- For each model we have used GridsearchCv to obtain best parameter
- Accuracy of each model has been tested on test data set
- Confusion matrix is plotted for each model

Lastly comparison is done between the models based on best parameter and accuracy

link of code

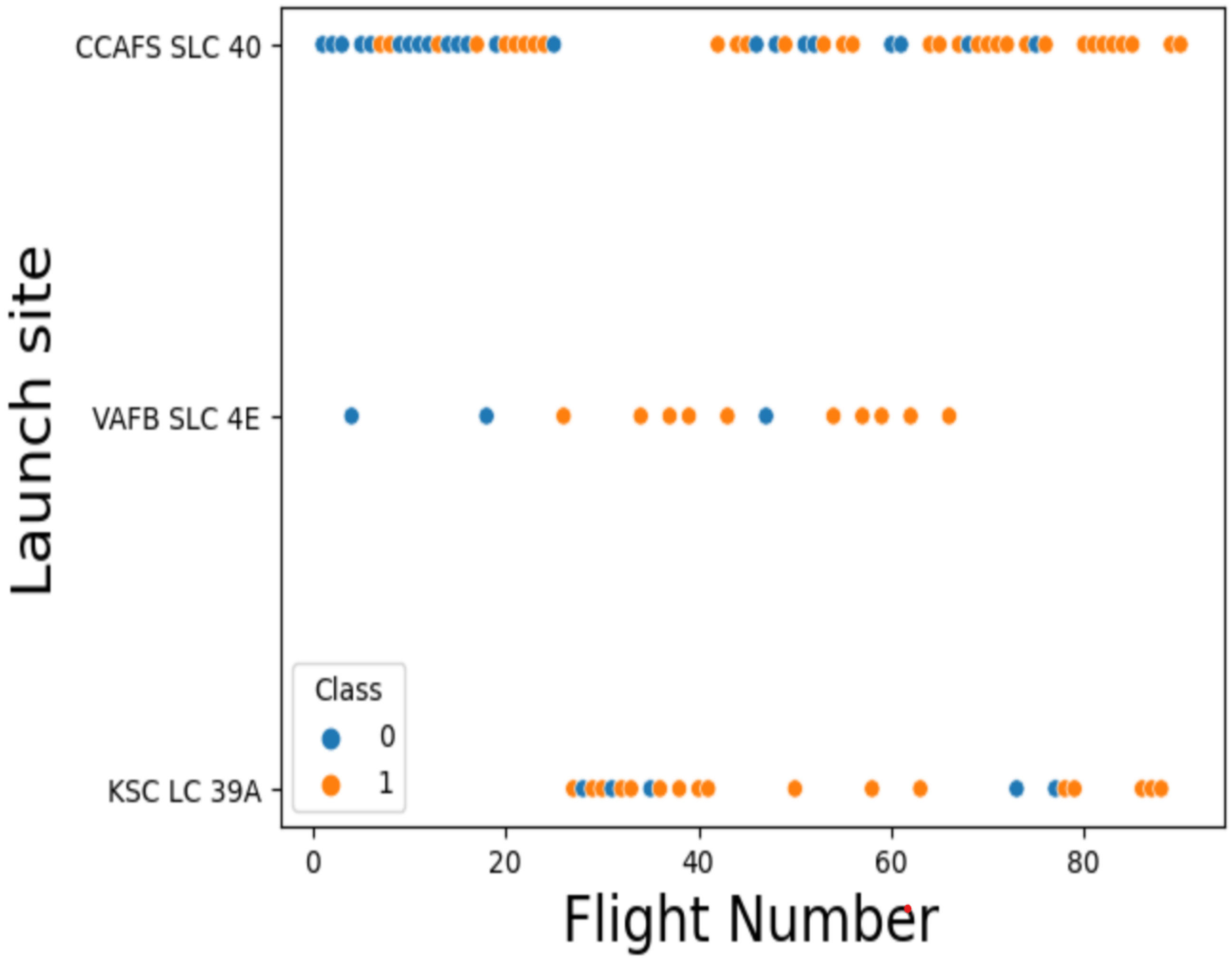
RESULT

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

Section 2

Insights drawn from EDA

PLOT OF FLIGHT NUMBER VS LAUNCH SITE



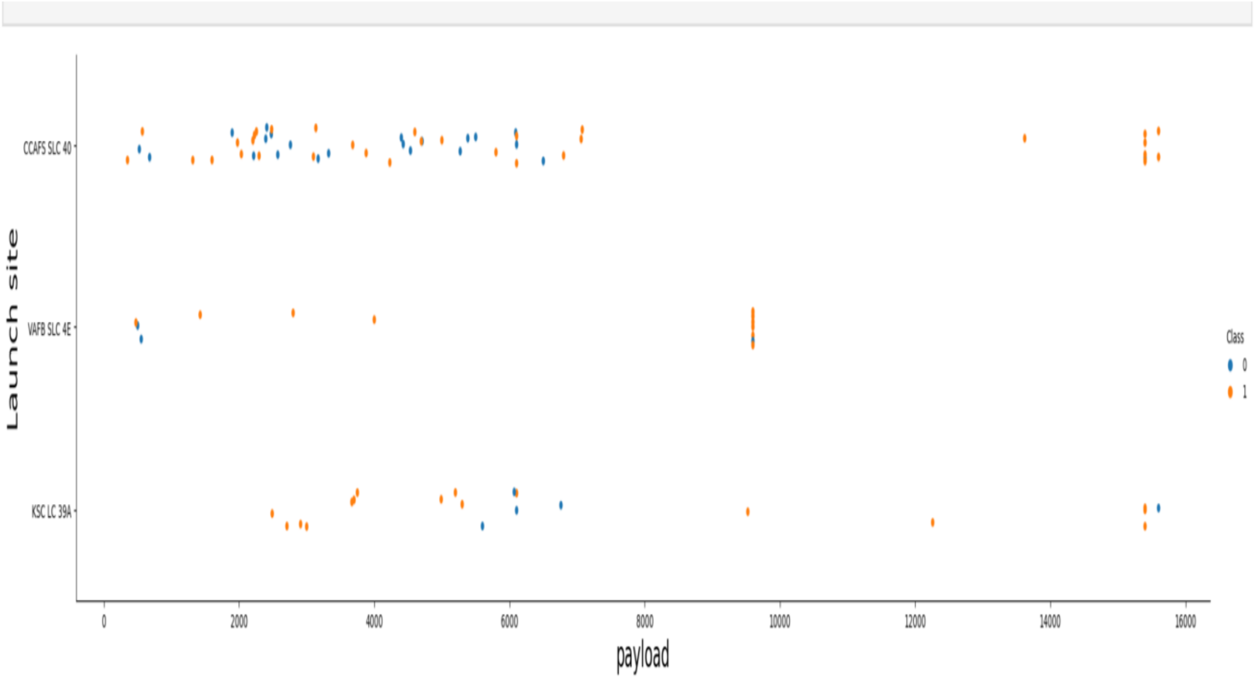
As we can see as we are increasing flight number the success rate for launch site namely CCAFS SLC 40 seems to increase when we compare the same with less flight number above 80 flight number its success rate is at top notch

Whereas for launch site VAFB SLC 4E we can observe from the scatterplot when flight number crosses 60 its success rate vanish completely

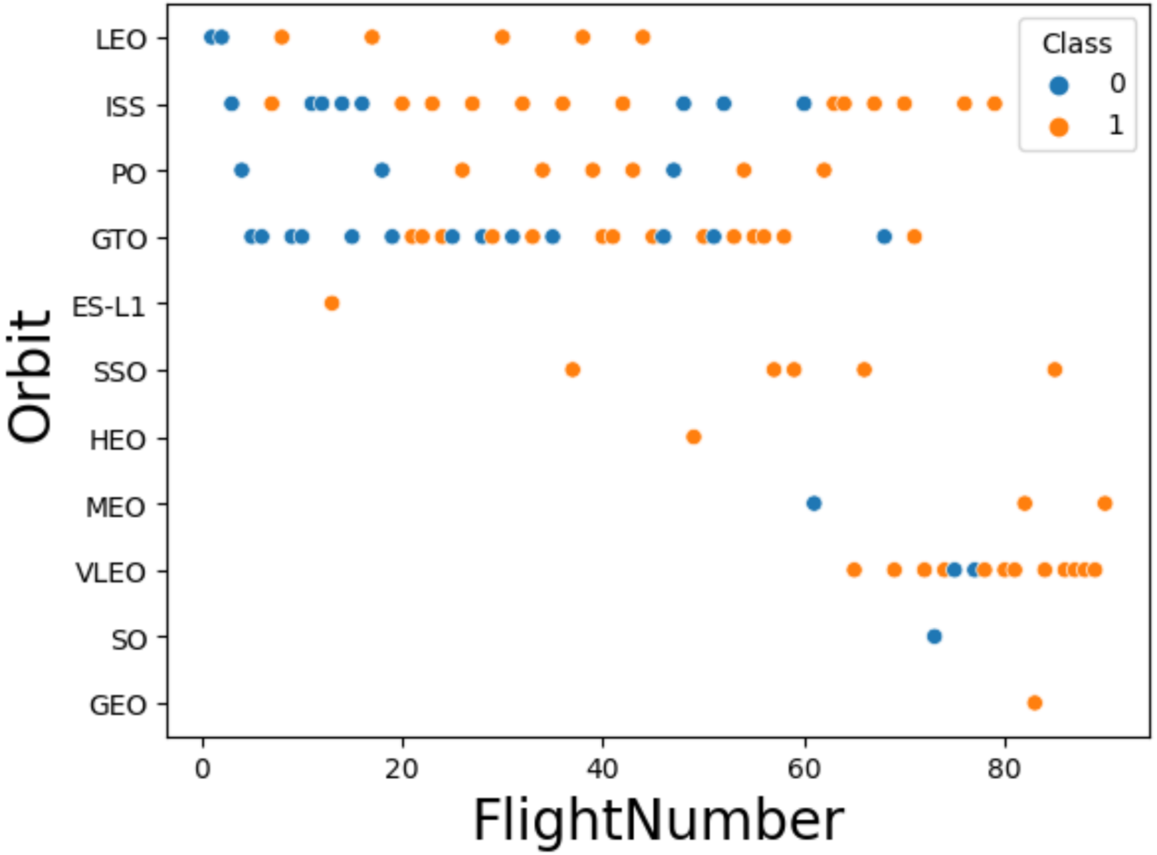
PLOT OF PAYLOAD VS LAUNCH SITE

As we can see from the scatter plot that when the payload is > 7000 success rate increases drastically for all the launch site

However, there is no clear linear relation encounter between payload and launch site



PLOT OF FLIGHT NUMBER VS ORBIT

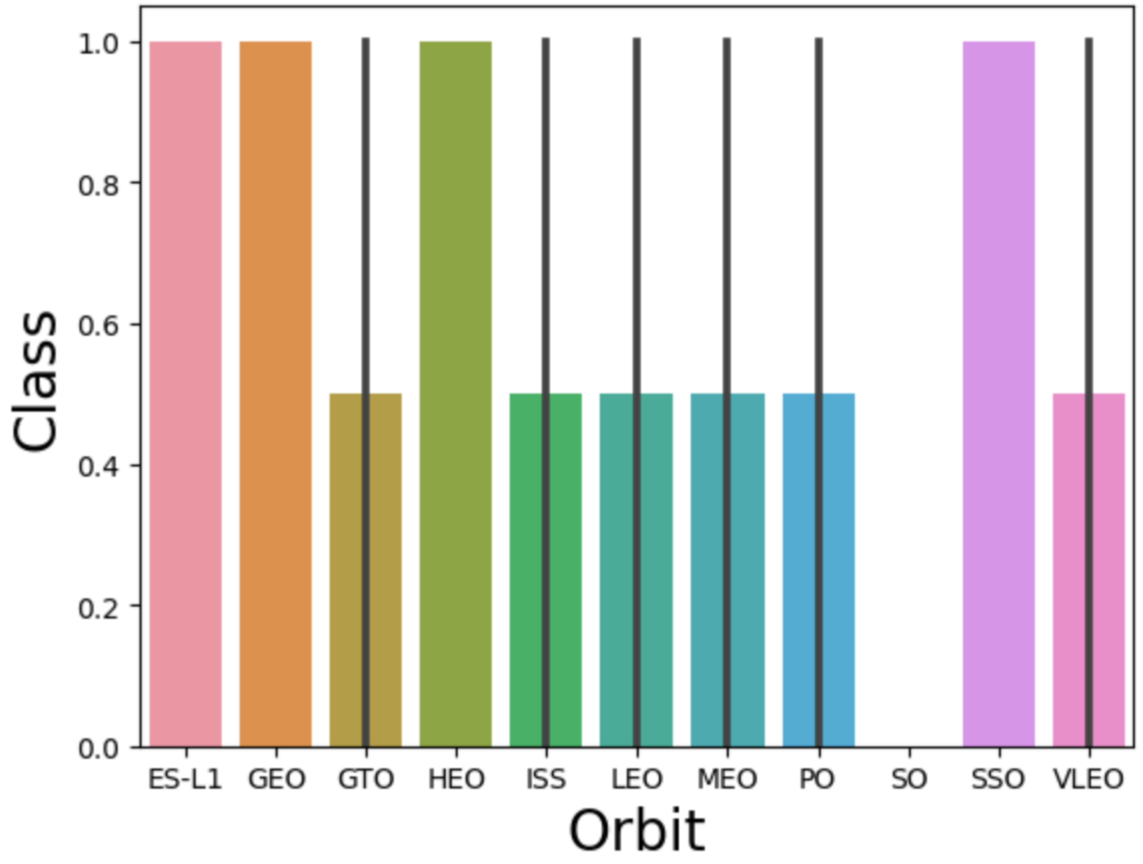


As we can see for the GTO orbit till flight number 50 more failure rate is observed in comparison to success rate,
However orbit like ES-L1, HEO, GEO, SO have only one observation so, it hard to say anything in favour or in against as we need more data to conclude.
Additionally, orbit LEO,ISS,PO,SSO,VLEO have more success rate in comparison to failure rate

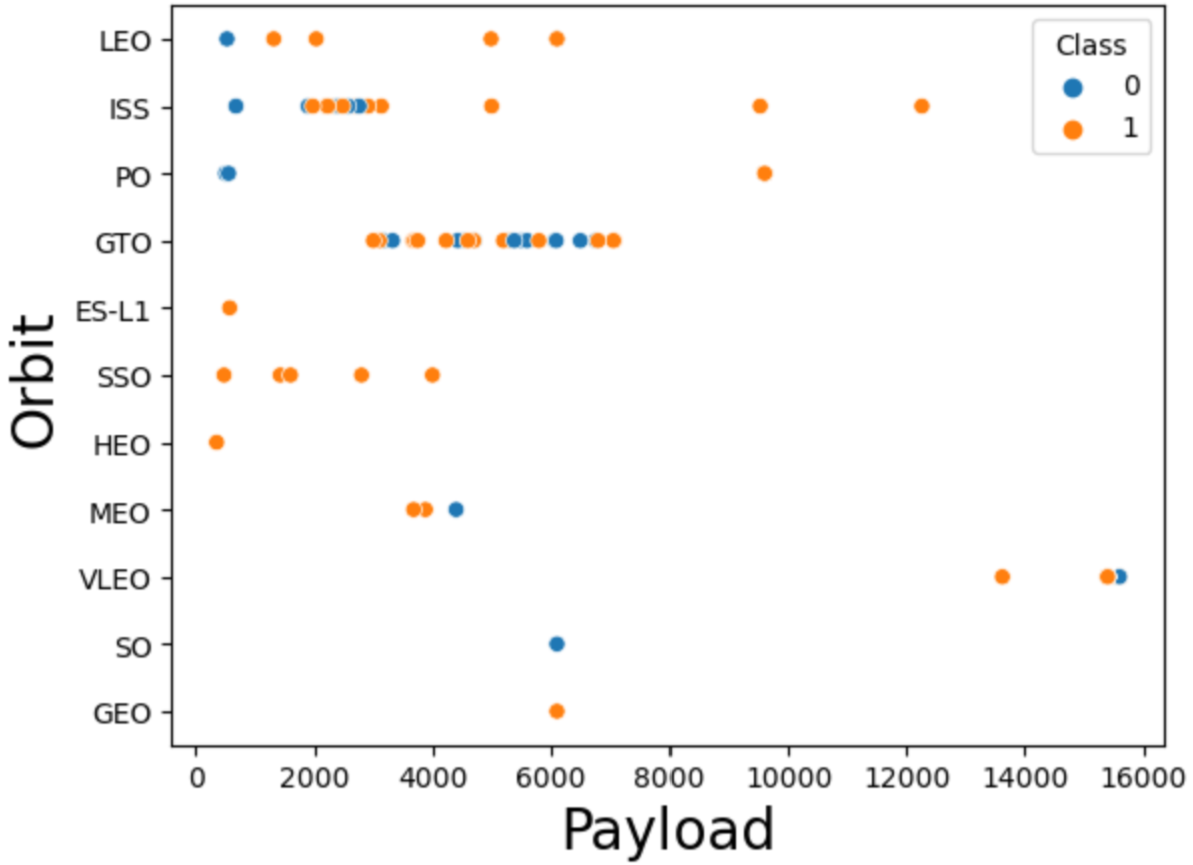
PLOT OF ORBIT VS SUCCESS RATE

SO orbit has 0% success rate at the same time orbit like ES-L1, GEO, HEO, SSO found to have success rate of 100% and orbit like GTO,ISS,LEO,MEO,PO,VLEO all are at the same level of success rate that is 50%

Additionally as discussed In pervious slide in orbit like ES-L1, HEO, GEO, SO has only one observation so it hard to conclude further.



PLOT OF PAYLOAD VS ORBIT

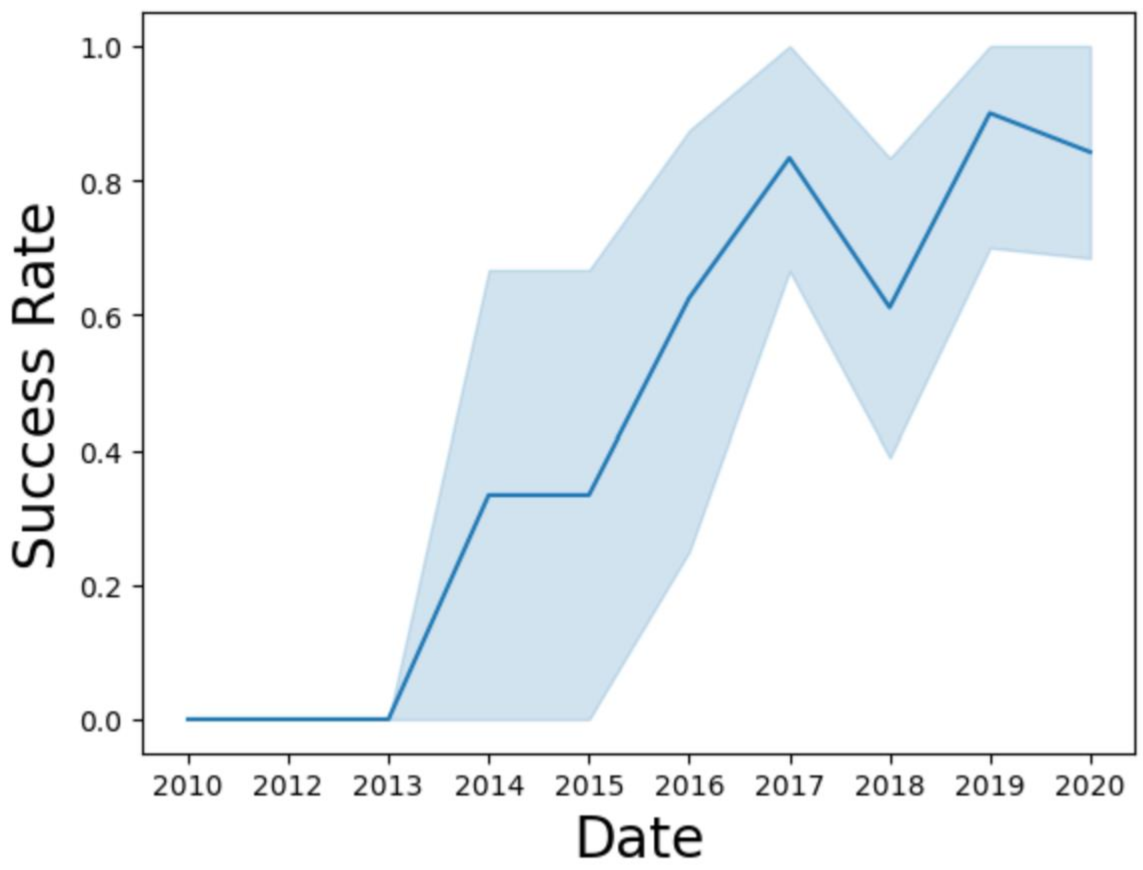


Orbit SSO reported 100% success rate when payload is up to 4000 kg , however for orbit GTO is 50%-50% success and failure till 8000kg of payload, in addition to these orbit ISS has reported more success rate when payload was below 6000kg and when payload is above 6000 kg.

For orbit like HEO, ES-L1 and GEO(100% success rate) while SO(which reported 100% failure) ,) these orbit has only one observation again it is quite difficult to predict or put any insight for these orbits

SUCCESS RATE TREND

From the plot we can see that till 2013 there was no success rate reported but after 2013 till 2020 the trend line continuously increased that means it has continuous success rate



Launch site name

```
%sql select  distinct(Launch_Site) from SPACEXTBL

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

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

Using distinct clause we have fetched unique launch site name as show in the above figure

Launch site which contain CCA in there name

To find launch site name which contain cca in there name we have used like operator to find pattern in string

```
%sql select Launch_Site from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total payload where customer is NASA

Using sum() aggregation function to find sum of payload mass and used filtering clause where to filter for only customer whose name is NASA(CRS)

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596.0

Avg payload for booster version F9 v1.1

Using avg() aggregation function we have found average payload mass and filter it with where clause for booster version = F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

First successful landing outcome

First successful landing outcome we wrote the query by using min() function for date and filter string using like operator to find similar pattern like success (ground pad)

```
%sql select min(Date) from SPACEXTBL where Landing_Outcome like '%Success (ground pad)%'
```

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

min(Date)
01/08/2018

Listing booster version

Listing all booster version based on landing outcome if it matches success(drone ship) and payload mass is between 4000 and 6000 kg

```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and
```

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

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

Total success and failure rate

Counting total success and failure rate based on mission outcome when it matches to pattern success or failure in the string and those rows which matches the required condition has to be counted

```
%%sql SELECT COUNT(*) as totalsuccessandfailure FROM SPACEXTBL
WHERE MISSION_OUTCOME LIKE '%Success%' OR MISSION_OUTCOME LIKE '%Failure(in flight)%'
```

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

totalsuccessandfailure
100

Booster version

Listing Booster version based on maximum payload criteria here to select maximum payload criteria I have used subquery concept of sql

```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ in (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

Booster_Version
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

Listing month, booster version, launch site

Listing month, booster version, launch site based on criteria where landing outcome matches Failure drone ship

```
%%sql SELECT substr(Date, 4, 2) AS MONTH, Booster_Version,Launch_Site
FROM SPACEXTBL
WHERE Landing_Outcome like '%Failure (drone ship)%' and substr("DATE", 7,4) = '2015'
```

* sqlite:///my_data1.db

Done.

MONTH	Booster_Version	Launch_Site
10	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

Listing Landing outcome and counting the same

Listing landing outcome which is based on the criteria that it should match pattern success and for that we have used like operator and date should be between 4/6/2010 to 20/3/2017 that should be grouped by landing outcome and order by landing outcome in descending order so that we can have top counts in the beginning itself

```
%%sql
select Landing_Outcome, COUNT (Landing_Outcome)
FROM SPACEXTBL
WHERE Landing_Outcome like '%Success%' and Date between '04/06/2010' and '20/03/2017'
GROUP BY Landing_Outcome
ORDER BY COUNT (Landing_Outcome) DESC
```

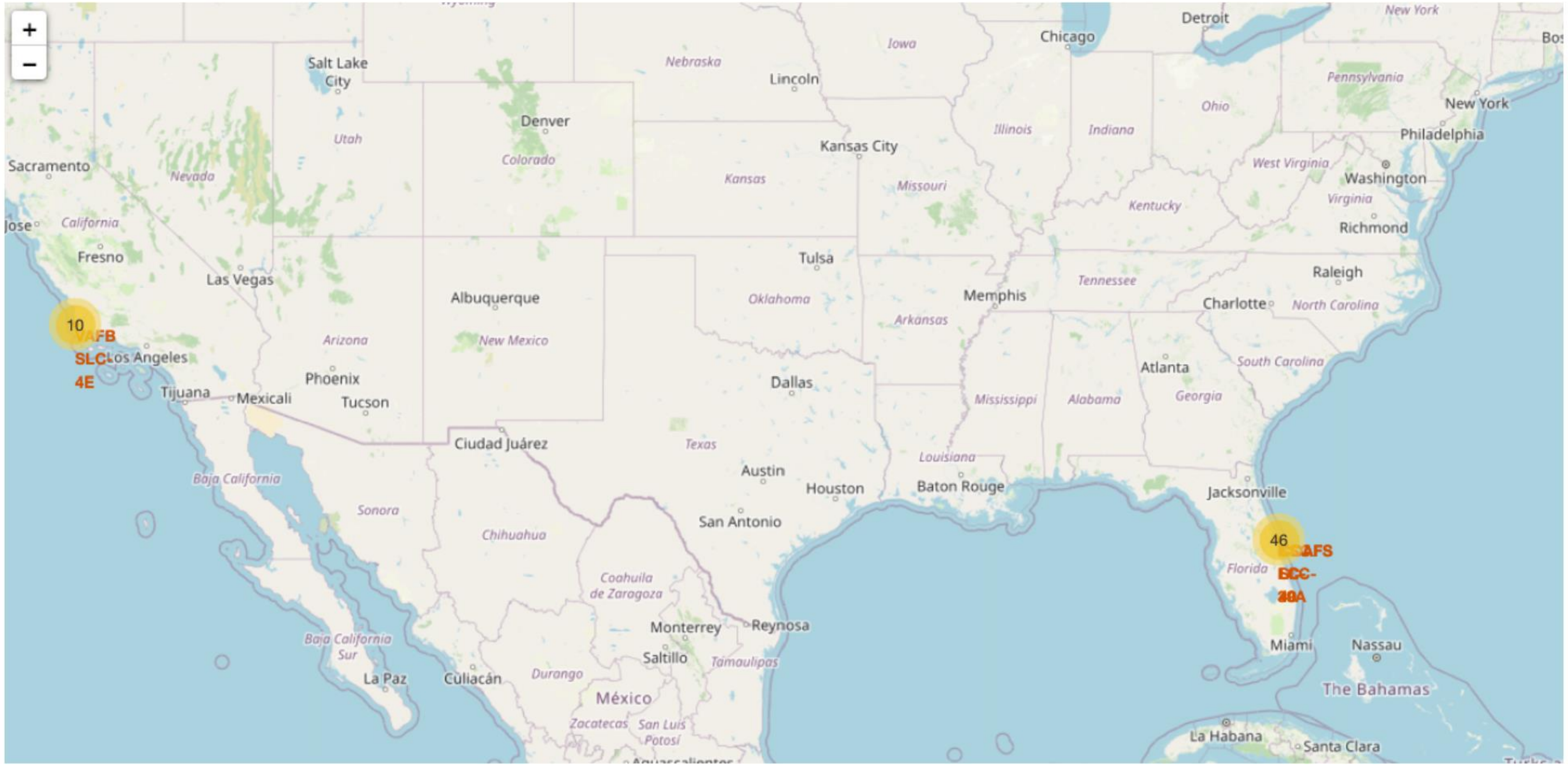
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	COUNT (Landing_Outcome)
Success	20
Success (drone ship)	8
Success (ground pad)	7

Section 4

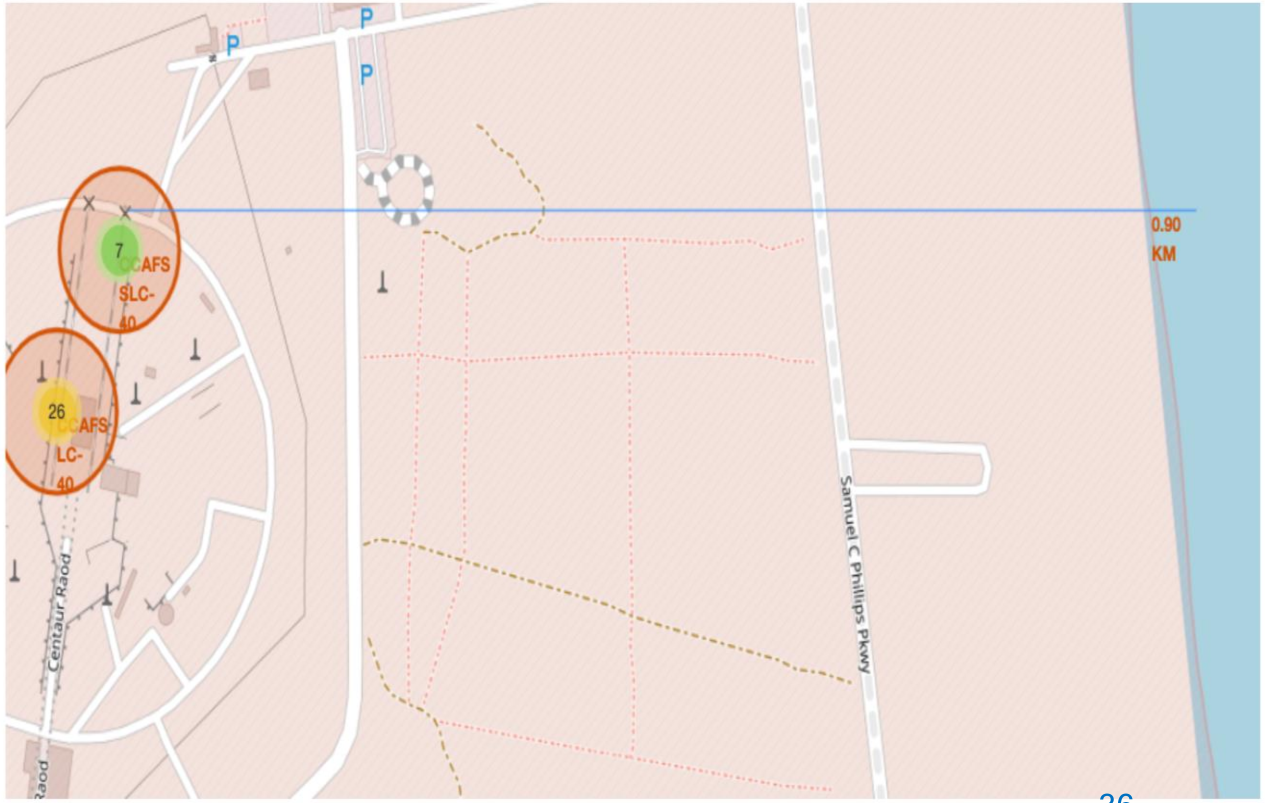
Launch Sites Proximities Analysis

LAUNCH SITES DISTANCE TO LANDMARKS



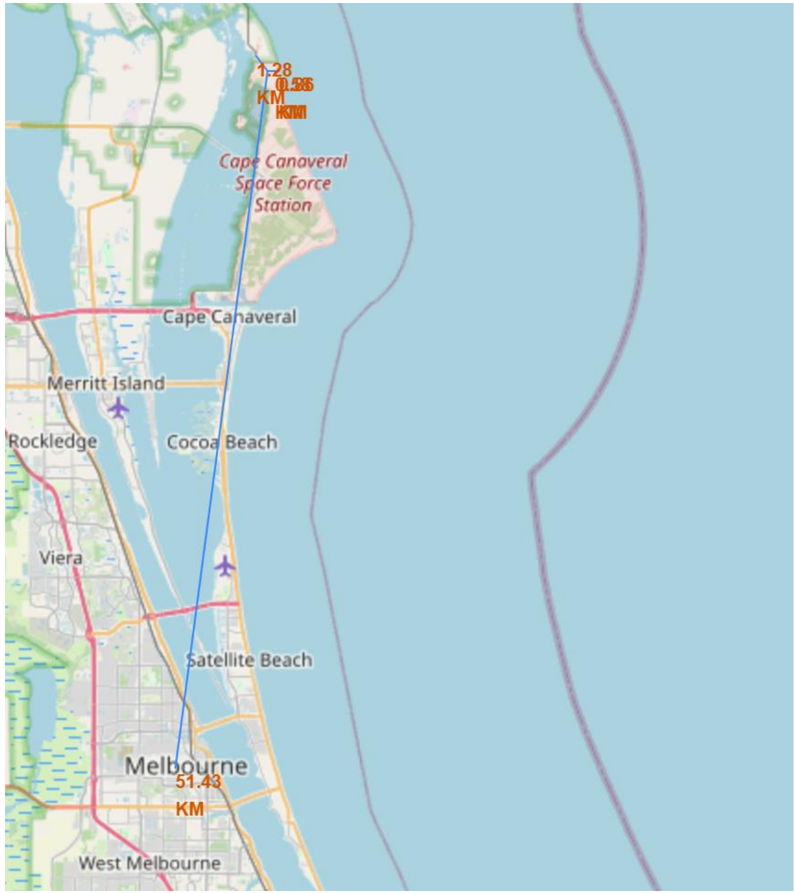
All launch site of SpaceX are near to the coastal region

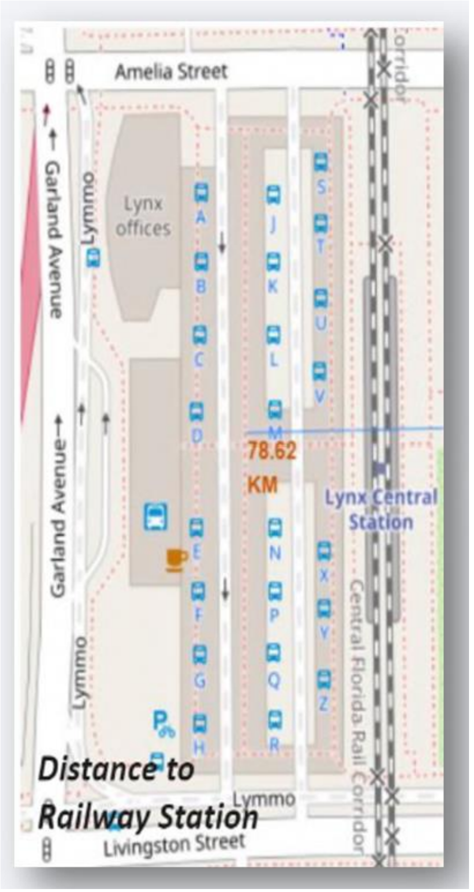
LAUNCH SITES DISTANCE TO LANDMARKS



All Launch site of SpaceX are at approximate .9 km away from the coastal region of USA

LAUNCH SITES DISTANCE TO LANDMARKS



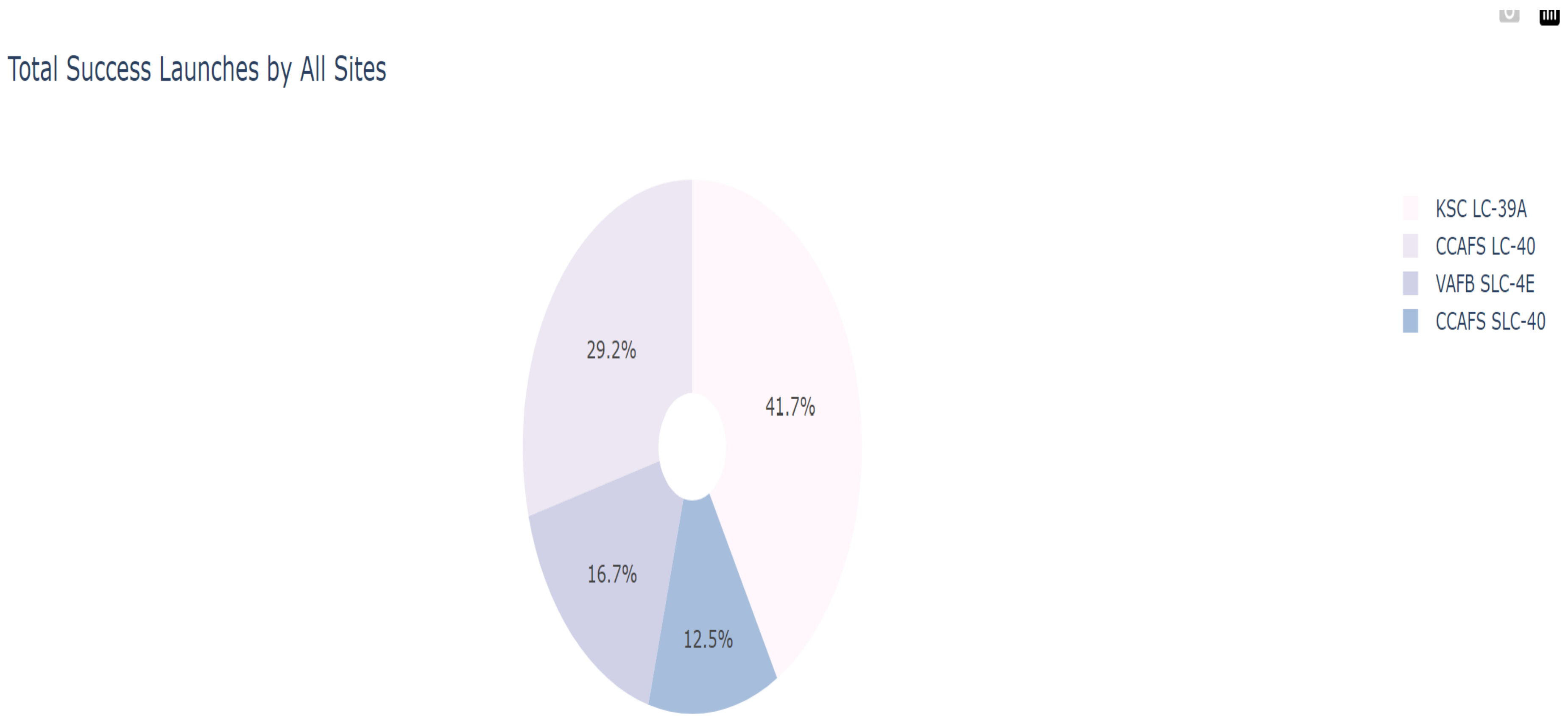


Distance of launch site for airport is around 51.43km and from railway it is around 78.62 km

Section 5

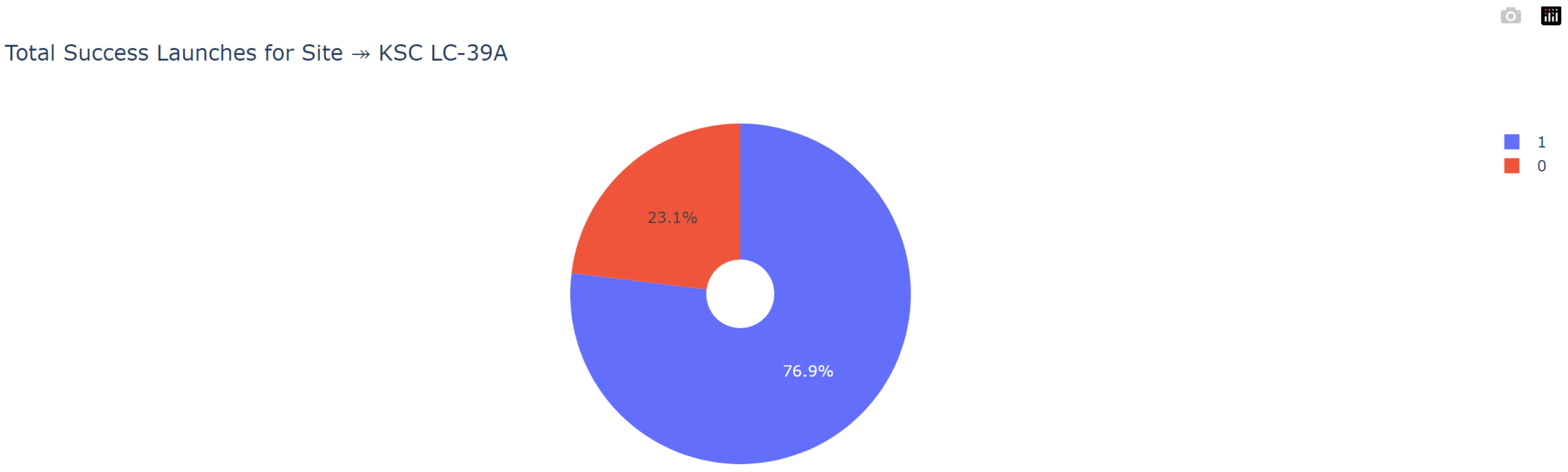
Build a Dashboard with Plotly Dash

DASH BOARD FOR SUCCESS LAUNCH



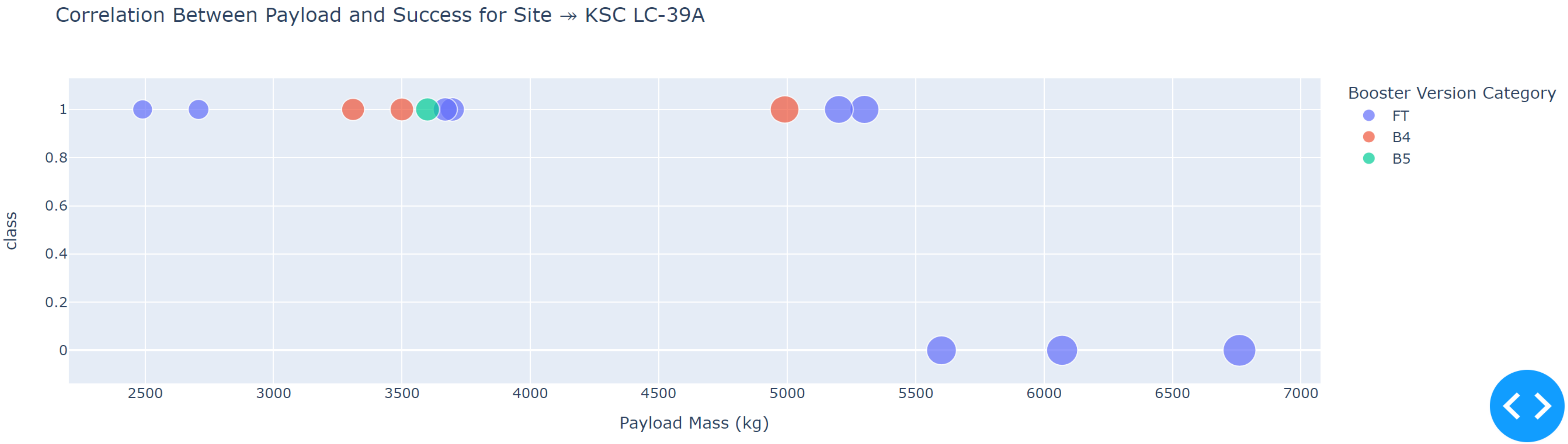
As we can see we have success rate for all the site in the form of pie chart success rate of KSC LC-39A is around 41.7% which is highest among all other site and least is for site CCAFS SLC-40 ie 12.5%

SUCCESS RATE FOR KSC LC-39A



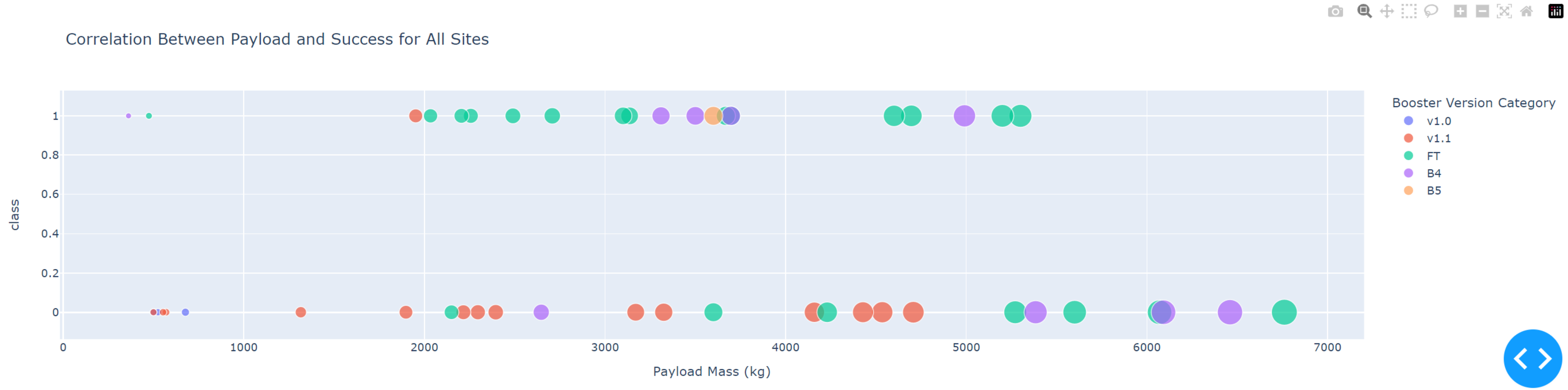
From above pie chart it is clear that success rate for site KSC LC -39A is around 76.9% which is highest success rate

PAYLOAD AND LAUNCH SITE



For KSC LC -39A when we have payload below 5500 kg its success rate is quite good almost near to 100% for all the booster version but when we increase the payload beyond 5500 kg its unable to perform well and its success rate drop to 0%

Success rate vs payload



It is observed that when the payload is beyond 5000kg the success rate is quite less as compared to failure rate

SUCCESS RATE VS PAYLOAD

It is observed that success rate is quite good when we have less payload (less than 4000kg)

Section 6

Predictive Analysis (Classification)

LOGISTIC REGRESSION ALGORITHM

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

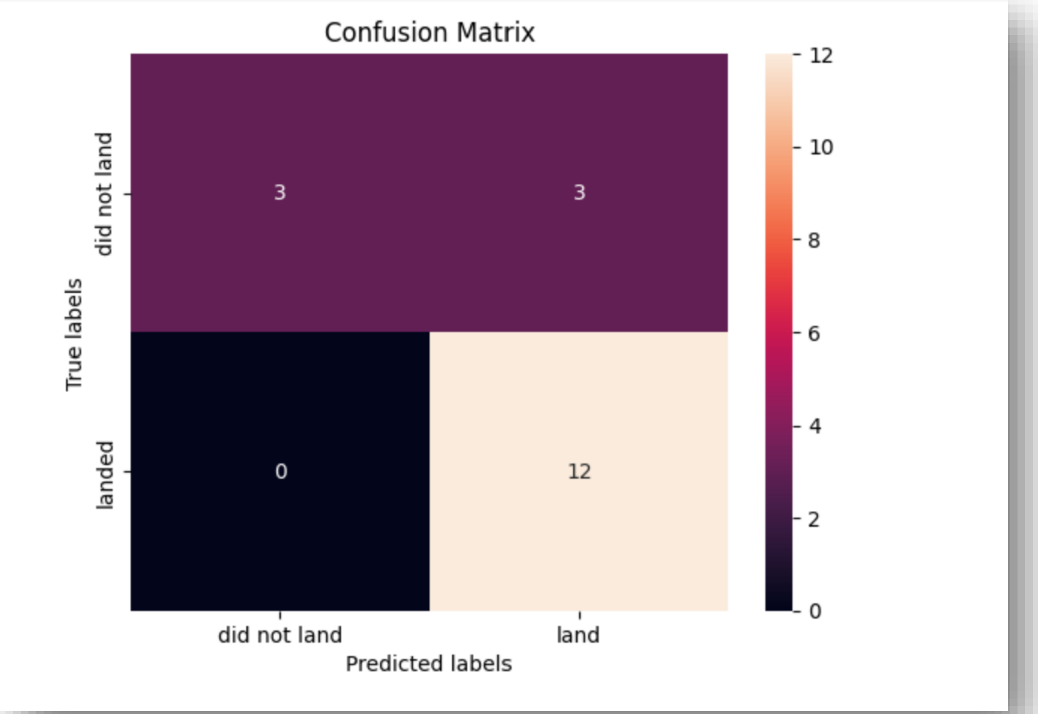
```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

TASK 5

Calculate the accuracy on the test data using the method `score` :

```
logreg_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```



Test score obtained for log reg is 83% that means it is 83% accurate

Explanation about confusion matrix

Actual value : rocket has landed and predicted value: rocket landed = 12 times and that is true positive(TP)

AV: land , predicted : didn’t land = 0 time ie (no False negative)

AV: didn’t land, PV: didn’t land =3 times (True negative)

AV: didn’t land, PV: landed = 3 times (false positive)

SVM ALGORITHM

```
svm_cv = GridSearchCV(svm, parameters, cv=10)

svm_cv.fit(X_train, Y_train)
```

▸ GridSearchCV
▸ estimator: SVC
▸ SVC

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

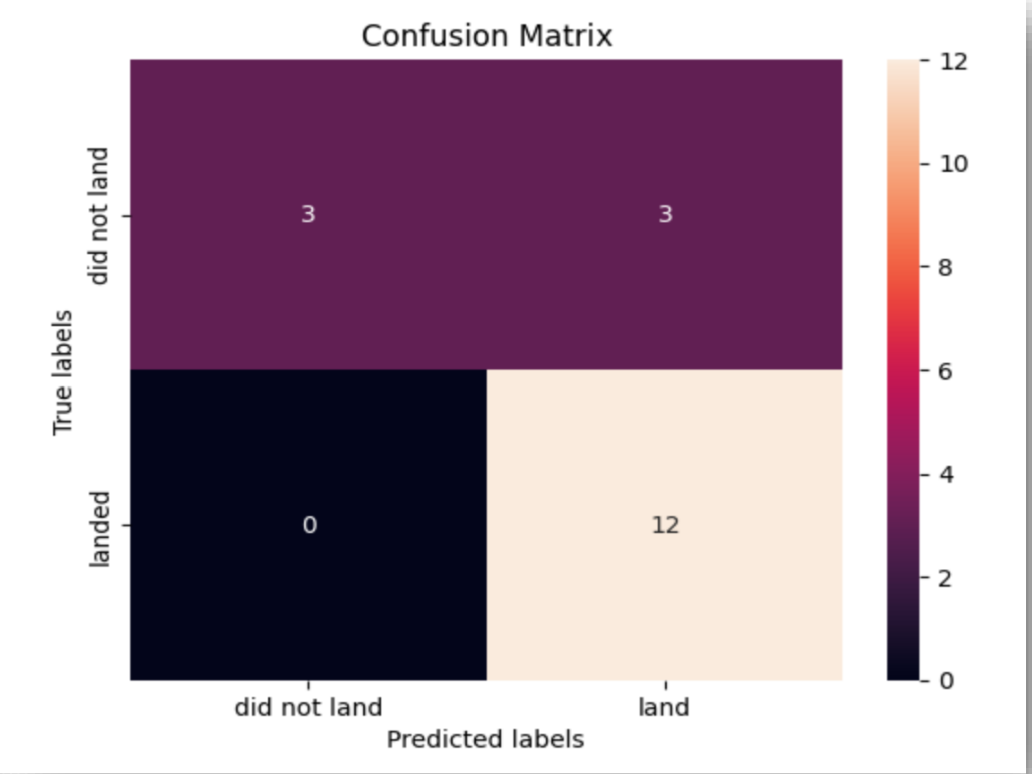
```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

TASK 7

Calculate the accuracy on the test data using the method `score` :

```
svm_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```



Test score obtained for SVM is 83% that means it is 83% accurate

Explanation about confusion matrix

Actual value : rocket has landed and predicted value: rocket landed = 12 times and that is true positive(TP)

AV: land , predicted : didn’t land = 0 time ie (no False negative)

AV: didn’t land, PV: didn’t land =3 times (True negative)

AV: didn’t land, PV: landed = 3 times (false positive)

KNN ALGORITHM

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

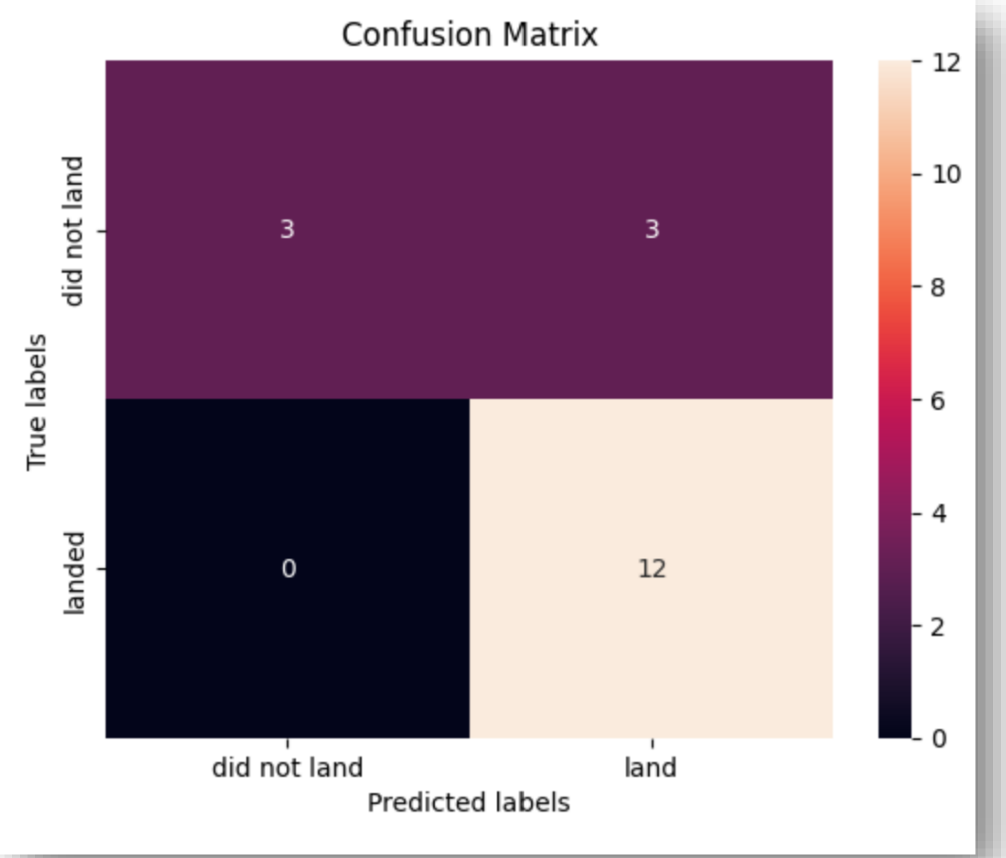
```
tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

TASK 11

Calculate the accuracy of knn_cv on the test data using the method `score` :

```
KNN_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```



Test score obtained for KNN is 83% that means it is 83% accurate

Explanation about confusion matrix

Actual value : rocket has landed and predicted value: rocket landed = 12 times and that is true positive(TP)

AV: land , predicted : didn’t land = 0 time ie (no False negative)

AV: didn’t land, PV: didn’t land =3 times (True negative)

AV: didn’t land, PV: landed = 3 times (false positive)

DECISION TREE ALGORITHM

```
tree_cv = GridSearchCV(tree, parameters, cv=10)

tree_cv.fit(X_train, Y_train)
```

•••

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

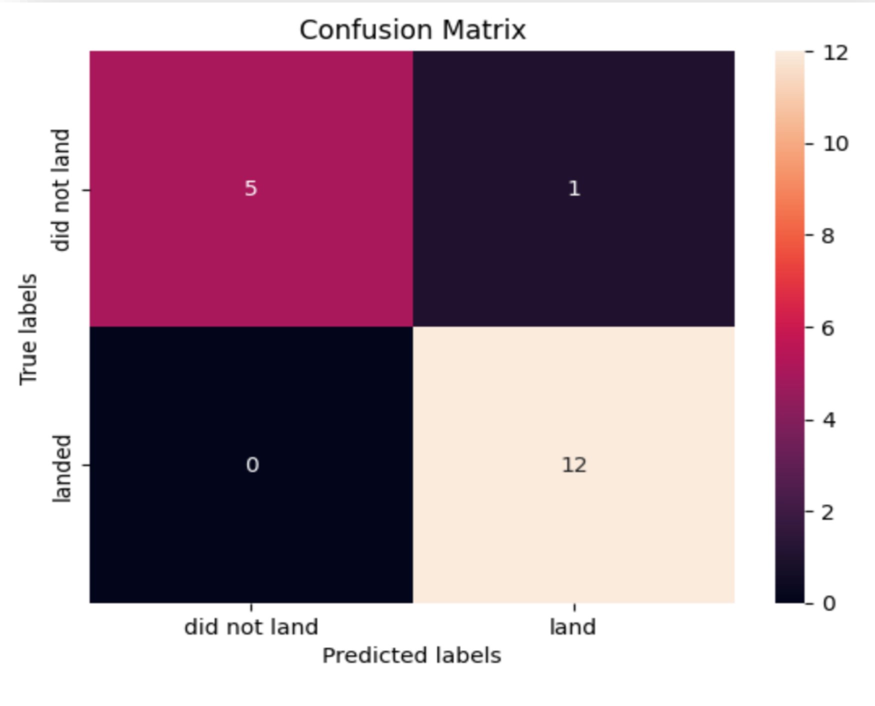
```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'best'}
accuracy : 0.9017857142857142
```

TASK 9

Calculate the accuracy of tree_cv on the test data using the method `score` :

```
tree_cv.score(X_test, Y_test)
```

```
0.9444444444444444
```



Test score obtained for Decision Tree is 94% that means it is 94% accurate

Explanation about confusion matrix

Actual value : rocket has landed and predicted value: rocket landed = 12 times and that is true positive(TP)

AV: land , predicted : didn't land = 0 time ie (no False negative)

AV: didn't land, PV: didn't land =5 times (True negative)

AV: didn't land, PV: landed = 1 times (false positive)

COMPARISON BASED ON BEST PARAMETER AND 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_)
elif bestalgorithm == 'KNN':
    print('Best Params is:', KNN_cv.best_params)
elif bestalgorithm == 'LogisticRegression':
    print('Best Params is:', logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter':
'best'}
```

As we have gone through various algorithm but the best algorithm we got is decision tree with test score of 90%

Conclusion

we can conclude following observations :

- Best classification algorithm we found was decision tree with highest accuracy and best test score of about 94% and 90% respectively
- Success rate for site KSC LC -39A is around 76.9% which is highest success rate among other launch sites
- Low payload has higher success rate as compared to high payload
- Orbit like LEO,ISS,PO,SSO,VLEO have more success rate in comparison to failure rate
- After 2013 till 2020 the trend line continuously increased that means it has continuous success rate

Thank you!