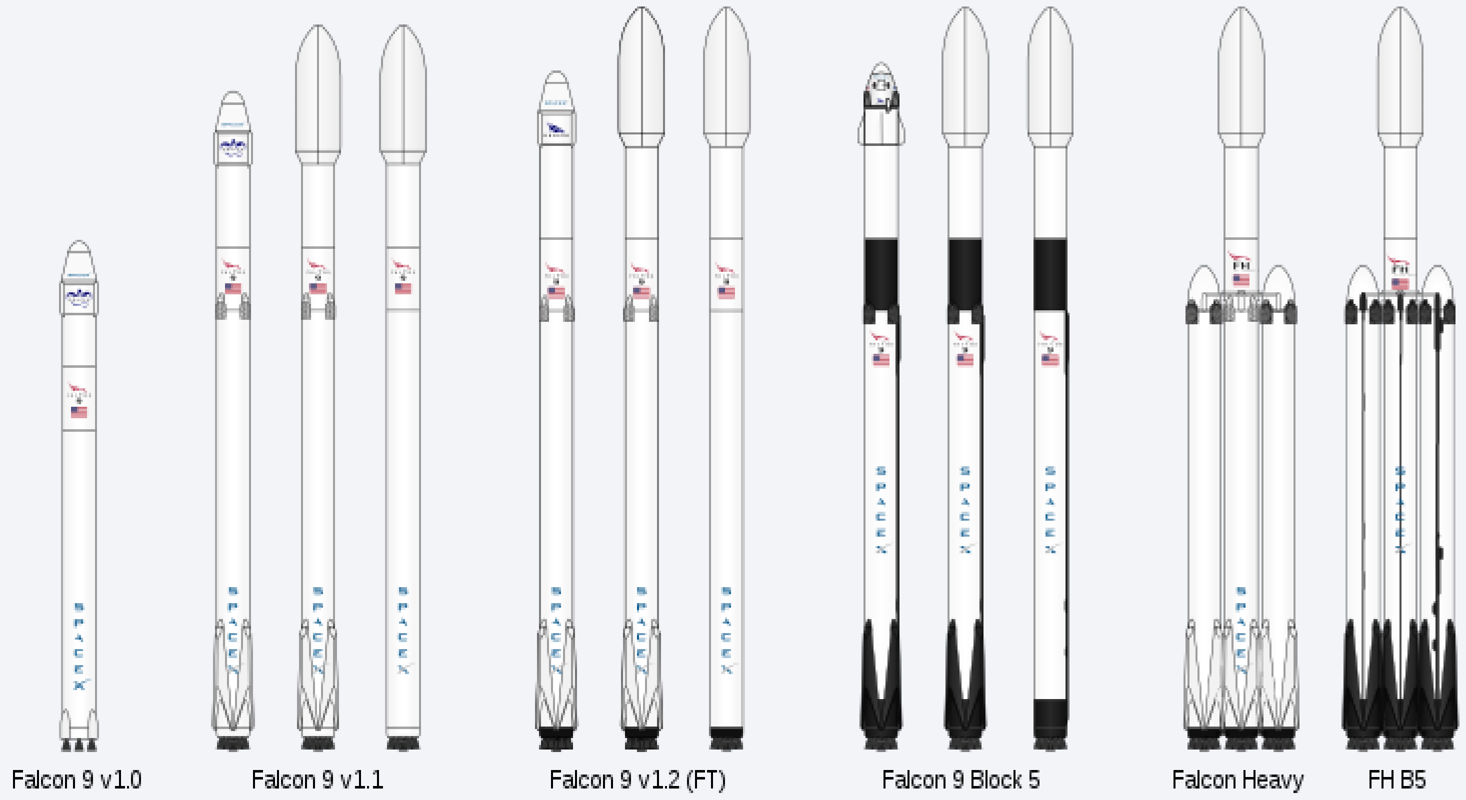
IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vinay Vashishtha
11th May 2024

Outline

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



Executive Summary

Summary of methodologies

- Data Collection through API
- Data Collection with Web Scraping
- Data Wrangling
- Exploratory Data Analysis with SQL
- Exploratory Data Analysis with Data Visualization
- Interactive Visual Analytics with Folium
- Machine Learning Prediction

Summary of all results

- Exploratory Data Analysis result
- Interactive analytics in screenshots
- Predictive Analytics result

Introduction

- **Project background and context**
  - Space X - Falcon 9 rocket launches cost 62 million dollars;
  - other providers cost upward of 165 million dollars each
  - Through machine learning model predict if the first stage will land successfully or not,
  - Determine the approximate cost of the whole mission
  - 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 determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing?
  - What operating conditions needs to be in place to ensure a successful landing program?

Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

Using SpaceX API

Web scraping from Wikipedia

Perform data wrangling

Basic EDA

Determined training labels

One-hot encoding was applied to 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

[GitHub Link for the project]

➢ **Requesting and parsing the SpaceX launch data using the GET request**

```
requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
```


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

➢ **Requesting the Falcon9 Launch Wiki page from its URL**

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

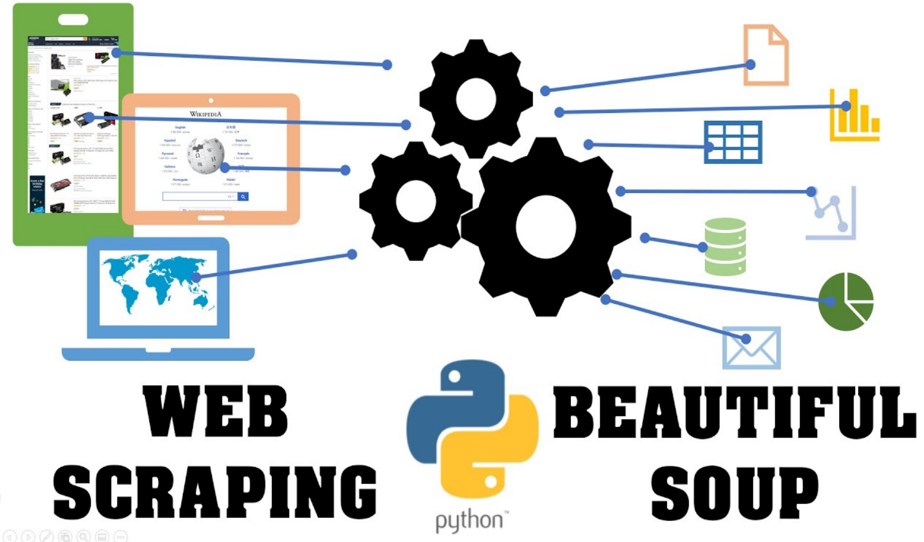
```
soup = BeautifulSoup(response.text, 'html.parser')
```

➢ **Extracting all column/variable names from the HTML table header**

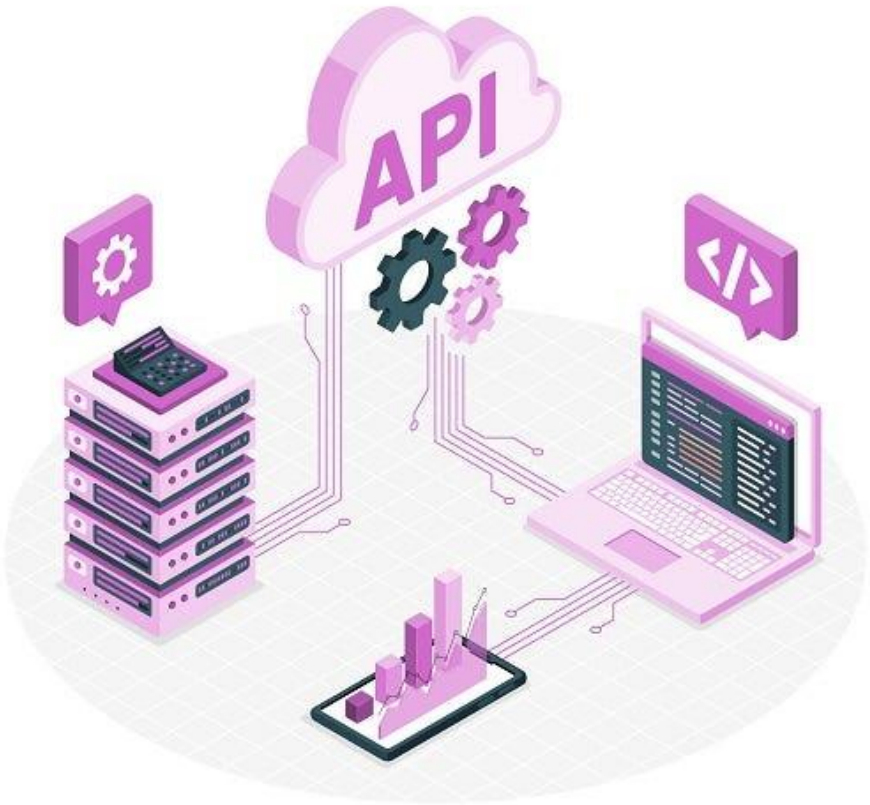
```
launch_dict= dict.fromkeys(column_names)
```

➢ **Creating a data frame by parsing the launch HTML tables**

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



Data Collection – SpaceX API



[GitHub Link for the project]

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}
```

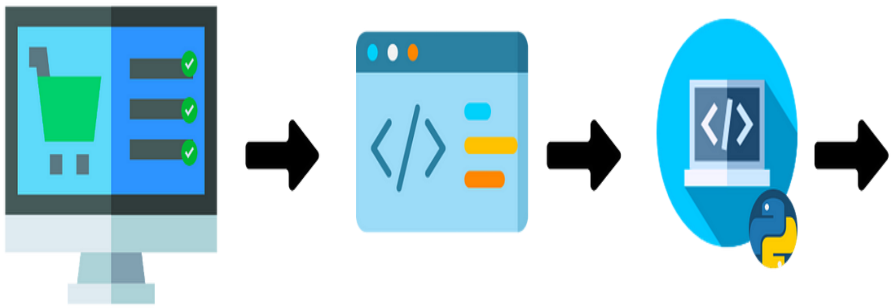
```
df = pd.DataFrame.from_dict(launch_dict)
```

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



Target a website and define what data must be collected

Extract and analyze the source code

Build a scraper in Python

Structured data

[GitHub Link for the project]

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)
```

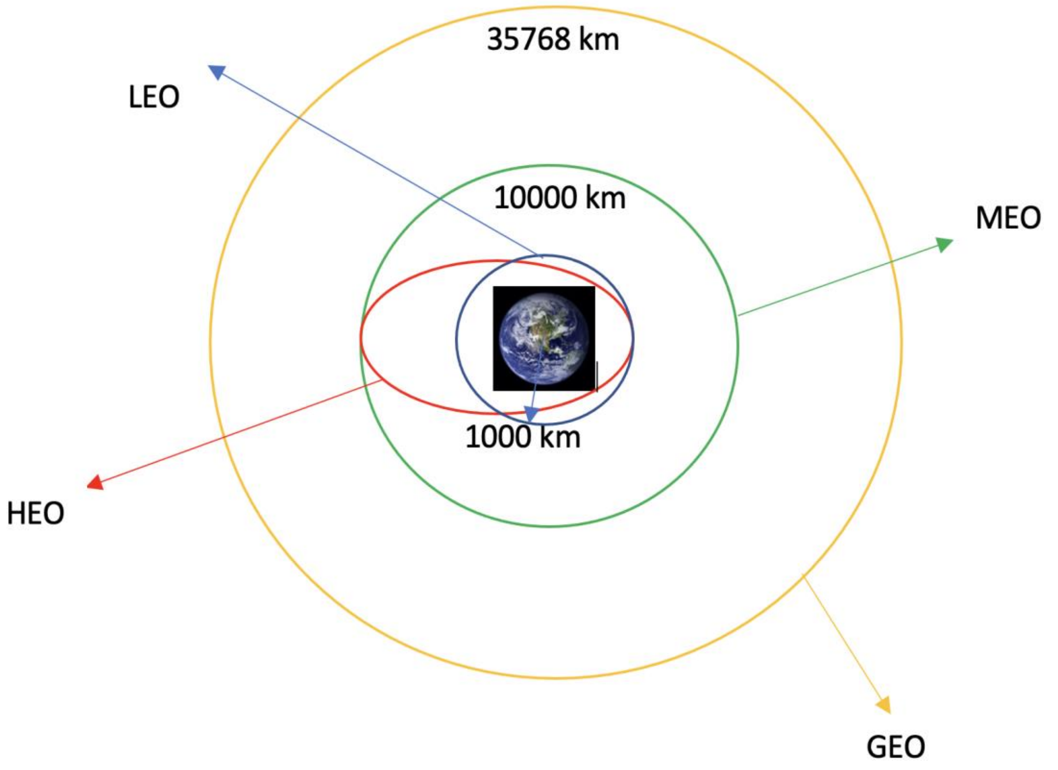
Data Wrangling

Exploratory Data Analysis

- **Number of launches on each site**
- **Number and occurrence of each orbit**
- **Number and occurrence of mission outcome per orbit type**

Determine Training Labels

- **Creating a landing outcome label from Outcome column**
- **Export dataset as .CSV**



GitHub Link for the project

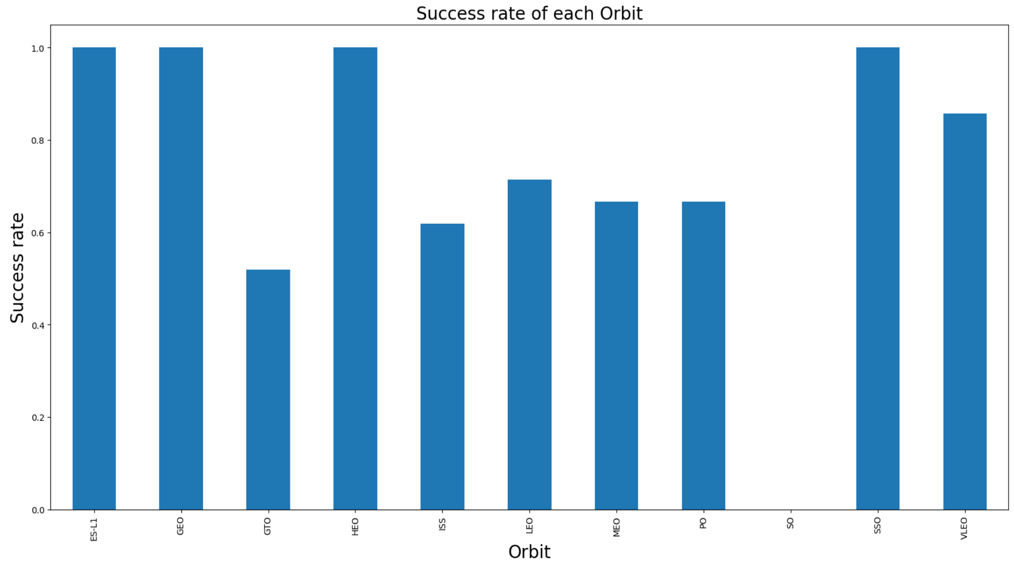
EDA with Data Visualization

GitHub Link for the project

Visualizing the relationship between

- Flight Number and Launch Site
- Payload and Launch Site
- success rate of each orbit type
- Flight Number and Orbit type
- Payload and Orbit type

Visualizing the launch success yearly trend



Scatter Graphs

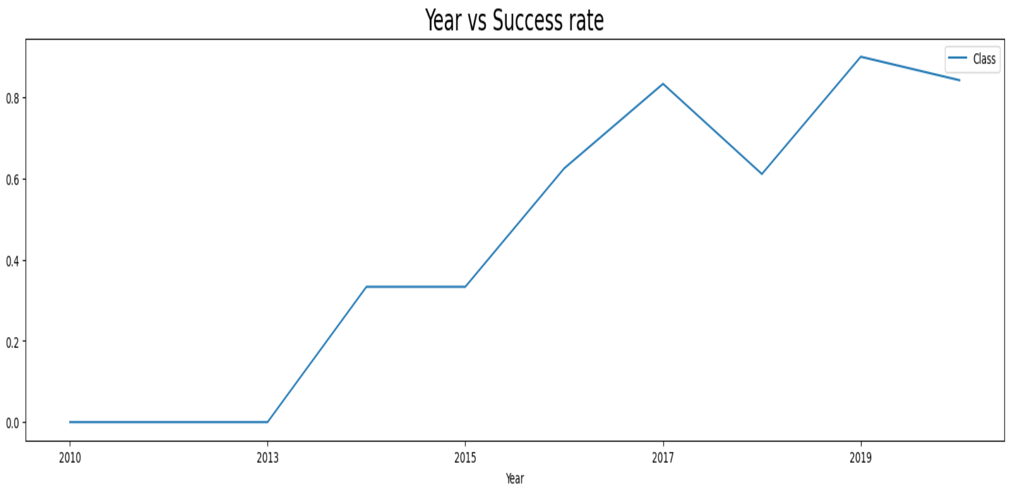
- Scatter plots show how much one variable is affected
- by another

Bar Graph

- A bar diagram makes it easy to compare sets of data between different groups at a glance

Line Graph

- Line graphs are useful in that they show data variables and trends



EDA with SQL

[GitHub Link for the project]

Loaded the dataset in IBM Db2 and connected to the database via api

Performed EDA with SQL

- unique launch site
- launch sites beginning with CCA
- total payload mass carried by boosters launched by NASA (CRS)
- average payload mass carried by booster version F9 v1.1
- date when the first successful landing outcome in ground pad was acheived
- The total no of successful and failed outcomes
- names of the booster_versions which have carried the maximum payload mass
- Rank of the count of landing outcomes between a specified date.

Build an Interactive Map with Folium

[GitHub Link for the project]

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

- Using Haversine's formula we calculated the distance from the Launch Site to various landmarks to find various trends

- We calculated the distances between a launch site to its proximities.

Build a Dashboard with Plotly Dash

[GitHub Link for the project]

Dashboard features

- Interactive
- dropdown for the site
- Range Slider for the Payload range

Plots

- plotted a pie chart to view the relative success and failures in launches for each site
- scatterplot between Payload and Class,

Outcome

- analyze the relation between payloads and launch sites,
- which helped to find the best site for launching Falcon 9

Predictive Analysis (Classification)

GitHub Link for the project

BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- 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 objects and train our dataset

EVALUATING MODEL

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

FINDING THE BEST PERFORMING CLASSIFICATION MODEL

- The model with the best accuracy score wins the best performing model

Results

[GitHub Link for the project]

Exploratory data analysis results

Newer rockets could carry more payload

Payloads over 8000kg have high success rate

the sucess rate since 2013 kept increasing till 2020

2010-2013 period had no success rate

Space X uses 4 different launch sites;

VLEO orbit has 14 launches and 85% success rate

The first successful landing outcome was in 2015, five years after the first launch.

With booster F9, almost every mission outcome was successful.

around 70 landing outcomes were successful, while there were 22 no attempts, and around 10 failed.

With time, the success increased mostly due to advancement in technology

Interactive analytics demo in screenshots

launch sites are close to the equator

Most launches happens at east cost launch sites

launch sites are in close proximity to railways

Launch sites are in close proximity to highways

Launch sites in close proximity to coastline

Launch sites keep certain distance away from cities

Predictive analysis results

All models, except KNN had almost the same accuracy for the test data

The best model is the Decision tree Classifier, having approximately 87.5% accuracy on training data and 88.9% test accuracy

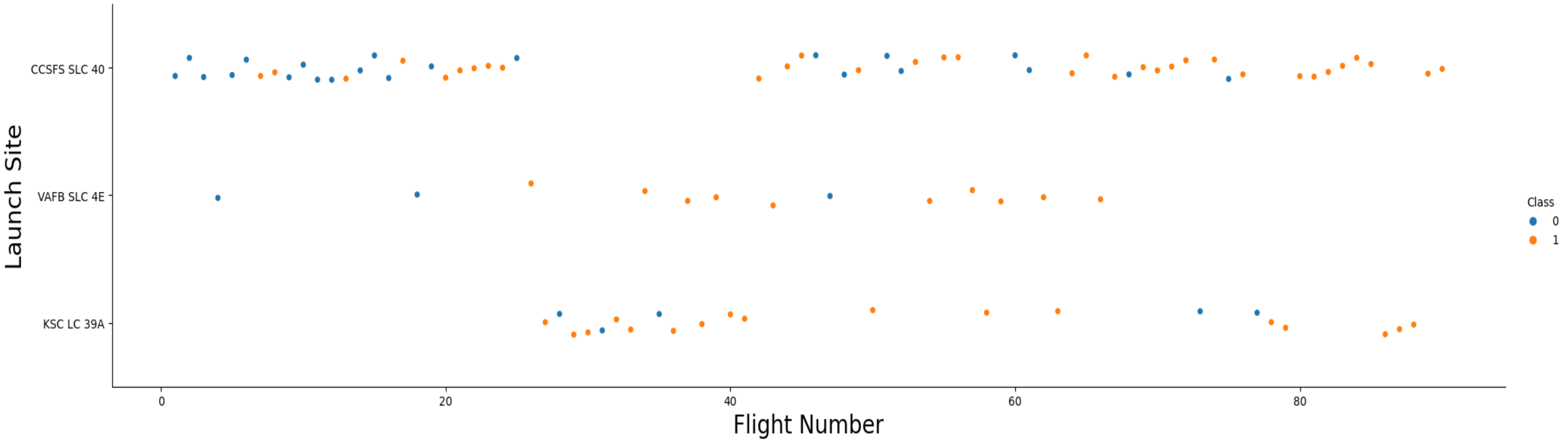
The worst model is the KNN, having approximately 80% mean accuracy

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

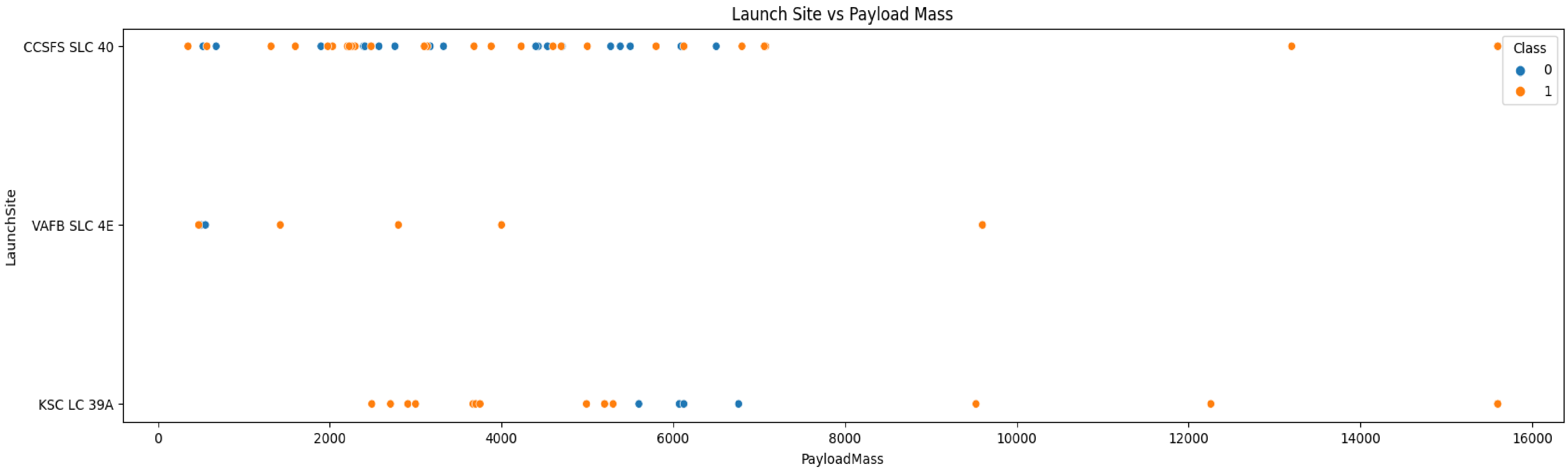
[GitHub Link for the project]



- CCAFS LC-40 has lower success rate than other two as it failed a lot during initial flights
- KSC LC-39A and VAFB SLC 4E have almost same success rate, and they have a relatively higher flight number so failure rate is low

Payload vs. Launch Site

GitHub Link for the project

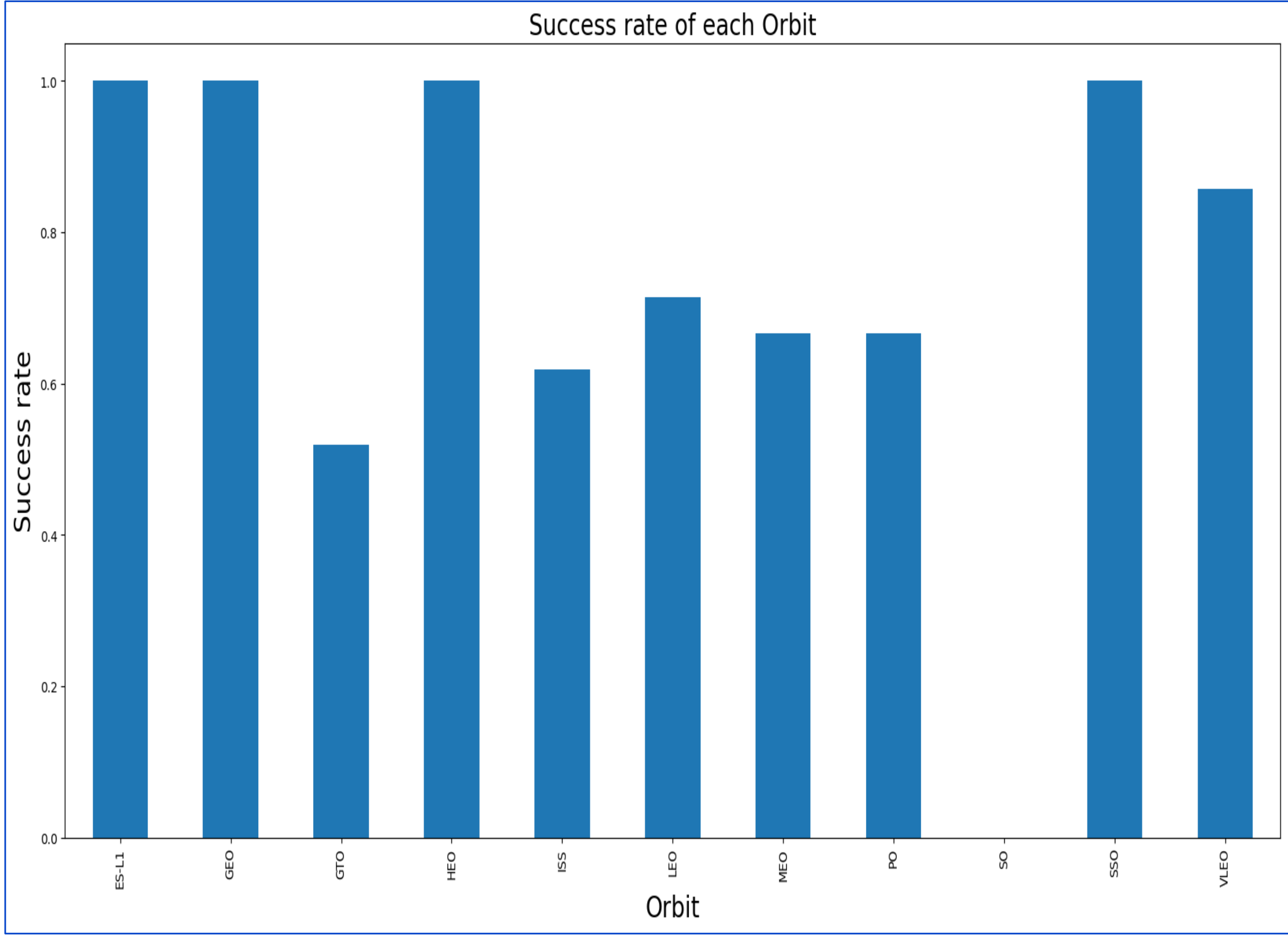


- ➢ There are no rockets launched for heavy payload mass(greater than 10000) for the VAFB-SLC launch site
- ➢ Payloads over 8000kg have high success rate
- ➢ Payloads less than 6000kg have high failure rate for the CCAFS LC-40 launch site

Success Rate vs. Orbit Type

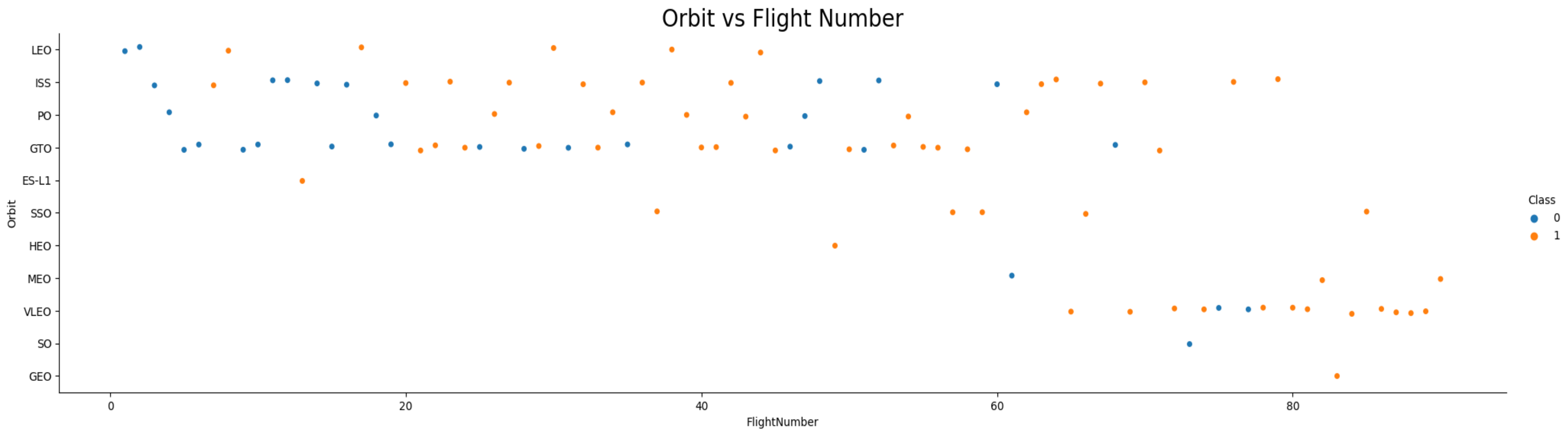
GitHub Link for the project

- ➢GEO, HEO, ES-L1, SSO have 1 launches and 100% success rate
- ➢SO has 1 launch and 0% succes rate
- ➢ISS has 21 launches 61% success rate
- ➢VLEO has 14 launches and 85% success rate



Flight Number vs. Orbit Type

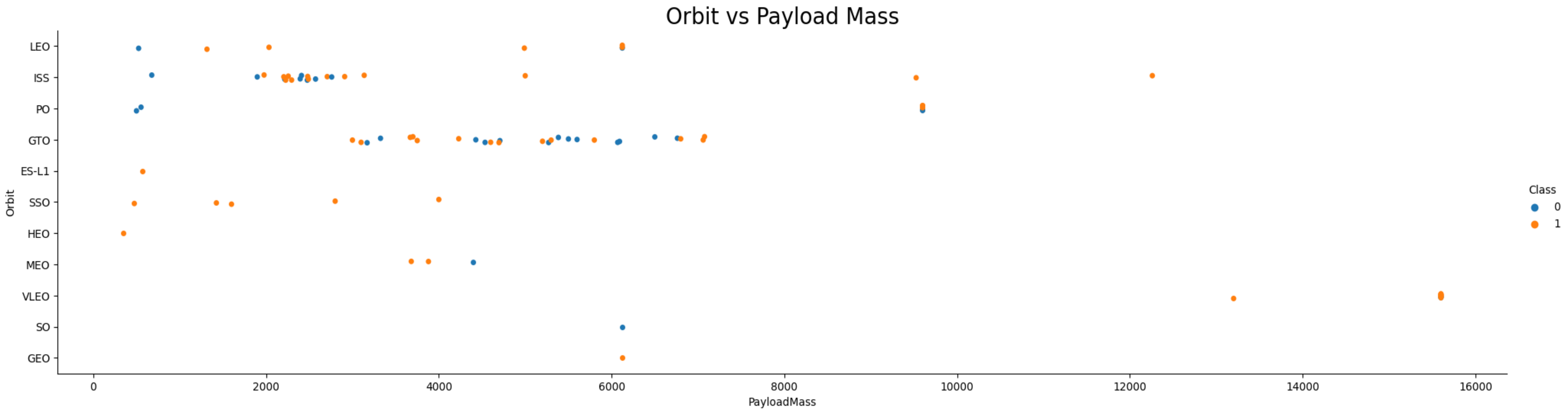
GitHub Link for the project



- We can 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.
- success rate is low for the first flight and it increases as the flight number increases

Payload vs. Orbit Type

[GitHub Link for the project]

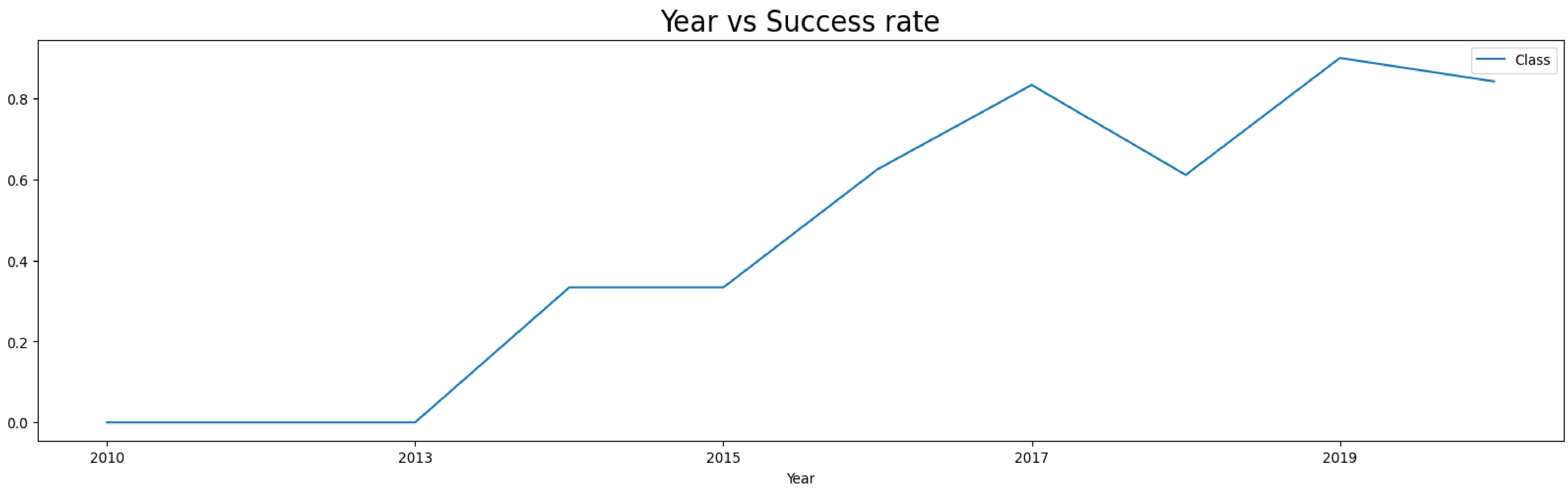


- With heavy payloads the successful landing or positive landing rate are more for PO, LEO and ISS orbits.

- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend

[GitHub Link for the project]



- ➢The sucess rate since 2013 kept increasing till 2020
- ➢2010-2013 had no success rate

All Launch Site Names

GitHub Link for the project

- QUERY EXPLAINATION
  - select DISTINCT Launch_Site from tblSpaceX
  - Using the word DISTINCT in the query means that it will only show Unique values in the Launch_Site column from tblSpaceX

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

Launch Site Names Begin with 'KSC'

GitHub Link for the project

- SQL QUERY
  - %sql select * from spacex where launch_site like 'CCA%' limit 5
- QUERY EXPLAINATION
  - Using the word TOP 5 in the query means that it will only show 5 records from tblSpaceX and LIKE keyword has a wild card with the words 'KSC%' the percentage in the end suggests that the Launch_Site name must start with KSC

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

GitHub Link for the project

- SQL QUERY

  - %sql select sum(payload_mass_kg_) as SUM from spacex where customer like 'NASA (CRS)'

SUM
45596

- QUERY EXPLAINATION

  - Using the function SUM summates the total in the column PAYLOAD_MASS_KG_
  - The WHERE clause filters the dataset to only perform calculations on Customer NASA (CRS)

Average Payload Mass by F9 v1.1

GitHub Link for the project

- SQL QUERY
  - %sql select avg(payload_mass_kg_) as AVG from spacex where booster_version = 'F9 v1.1'

AVG
2928

- QUERY EXPLAINATION
  - Using the function AVG works out the average in the column PAYLOAD_MASS_KG_
  - The WHERE clause filters the dataset to only perform calculations on Booster_version F9 v1.1

First Successful Ground Landing Date

GitHub Link for the project

- SQL QUERY
  - ➢ %sql select min(date) as DATE from spacex where landing_outcome like '%ground pad%'

DATE
2015-12-22

- QUERY EXPLAINATION
  - ➢ Using the function MIN works out the minimum date in the column Date
  - ➢ The WHERE clause filters the dataset to only perform calculations on Landing_Outcome Success (drone ship)

Successful Drone Ship Landing with Payload between 4000 and 6000

- SQL QUERY
  - %%sql select booster_version as name from spacex where landing_outcome like '%drone%' and payload_mass_kg_ > 4000 and payload_mass_kg_ < 6000

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

- QUERY EXPLAINATION
  - Selecting only Booster_Version
  - The WHERE clause filters the dataset to Landing_Outcome = Success (drone ship)
  - The AND clause specifies additional filter conditions Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000

GitHub Link for the project

Total Number of Successful and Failure Mission Outcomes

- SQL QUERY
  - %%sql select count(*) as successful_launches from spacex where mission_outcome like '%Success%';
  - %sql select count(*) as failed_launches from spacex where mission_outcome like '%Failure%'

successful_launches
100

failed_launches
1

- QUERY EXPLAINATION
  - The LIKE '%foo%' wildcard shows that in the record the foo phrase is in any part of the string in the records

GitHub Link for the project

Boosters Carried Maximum Payload

GitHub Link for the project

- SQL QUERY
  - %%sql select booster_version as MAX_PAYLOAD_BOOSTERS from spacex where payload_mass_kg_ = (select max(payload_mass_kg_) from spacex)
- QUERY EXPLAINATION
  - Using the word DISTINCT in the query means that it will only show Unique values in the Booster_Version column from tblSpaceX
  - GROUP BY puts the list in order set to a certain condition.
  - DESC means its arranging the dataset into descending order

max_payload_boosters
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

2015 Launch Records

GitHub Link for the project

- SQL QUERY
  - %%sql select booster_version, launch_site, landing_outcome from spacex where Year(date) = 2015 and landing_outcome like '%Failure%drone%'

booster_version	launch_site	landing_outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- QUERY EXPLAINATION
  - NVARCHAR the MONTH function returns name month.
  - The function CONVERT converts NVARCHAR to Date.
  - WHERE clause filters Year to be 2015

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

- SQL QUERY
  - %%sql select landing_outcome, count(landing_outcome) as COUNT from spacex group by landing_outcome order by count(landing_outcome) desc
- QUERY EXPLAINATION
  - Function COUNT counts records in column
  - WHERE filters data
  - LIKE (wildcard)
  - AND (conditions)
  - AND (conditions)

landing_outcome	COUNT
Success	38
No attempt	22
Success (drone ship)	14
Success (ground pad)	9
Controlled (ocean)	5
Failure (drone ship)	5
Failure	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

GitHub Link for the project

Section 3

Launch Sites Proximities Analysis

All launch sites with markers

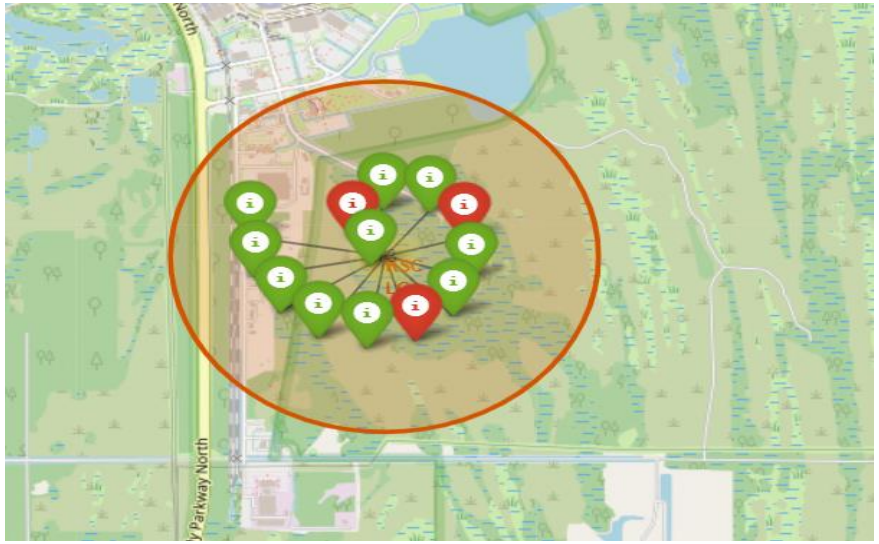
[GitHub Link for the project]

- Findings
  - We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California
  - all the sites are located close to the equator as it takes less fuel to launch a rocket from the equator

Colour Labelled Markers

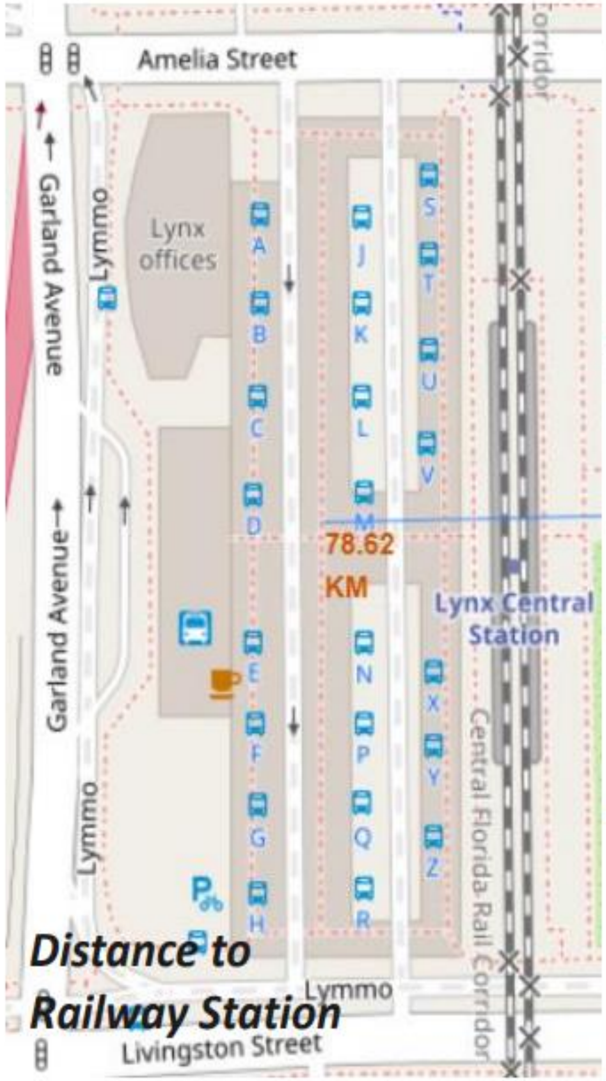
GitHub Link for the project

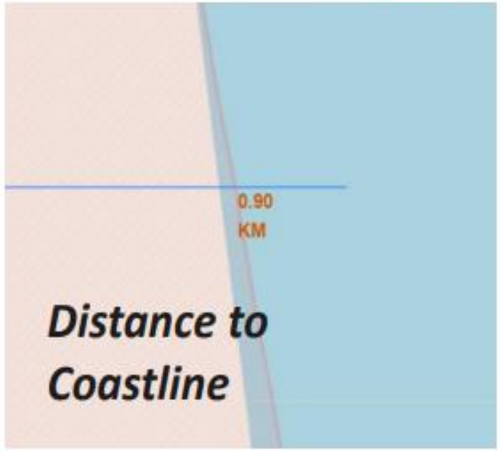
- Green Marker shows successful Launches and Red Marker shows Failures

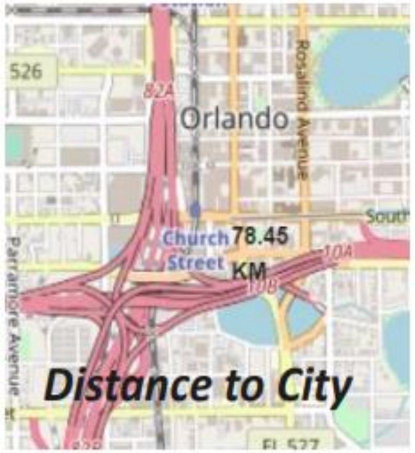


Distance to proximities

[GitHub Link for the project]







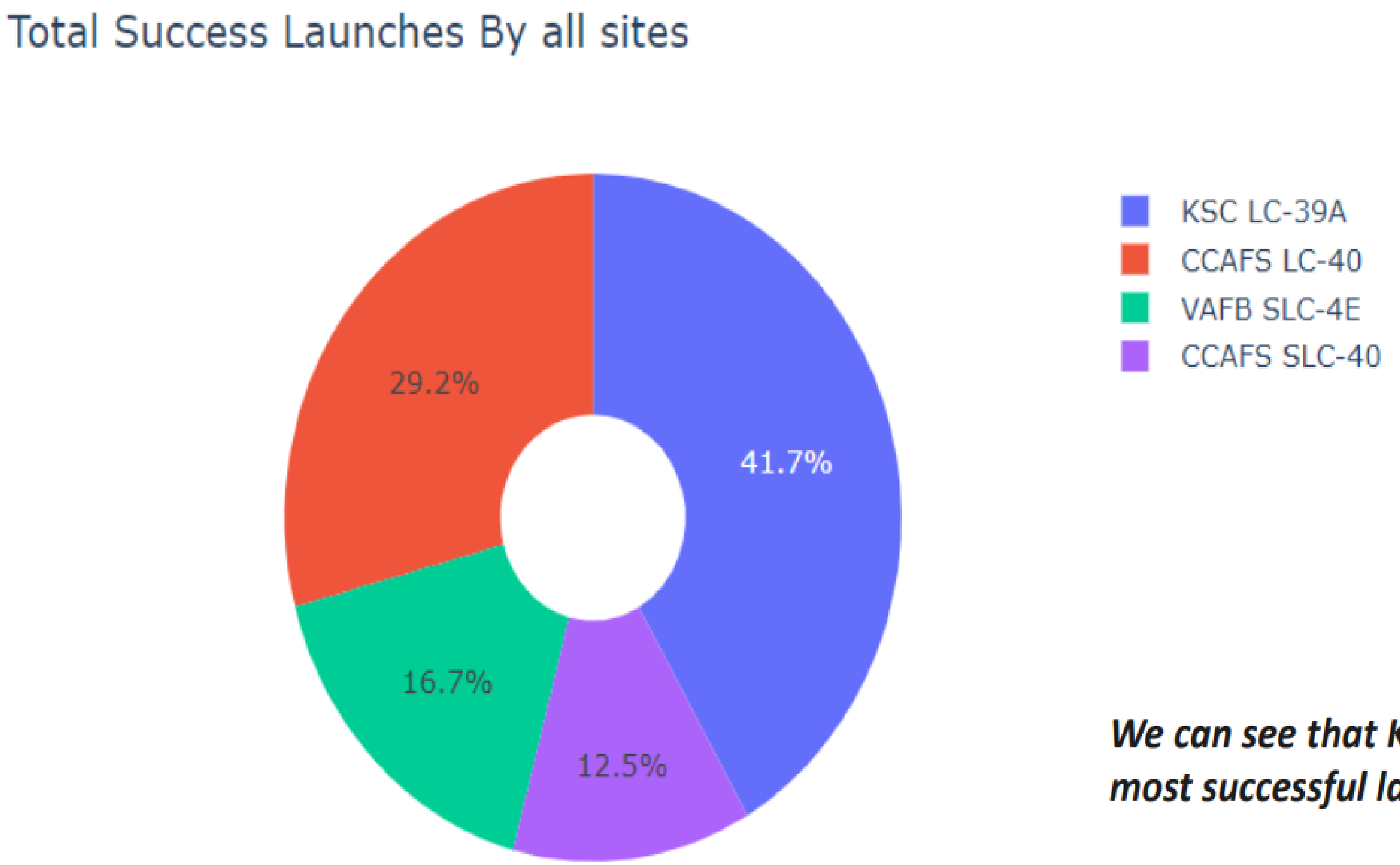
- •Are launch sites in close proximity to railways? No
- •Are launch sites in close proximity to highways? No
- •Are launch sites in close proximity to coastline? Yes
- •Do launch sites keep certain distance away from cities? Yes

Section 4

Build a Dashboard with Plotly Dash

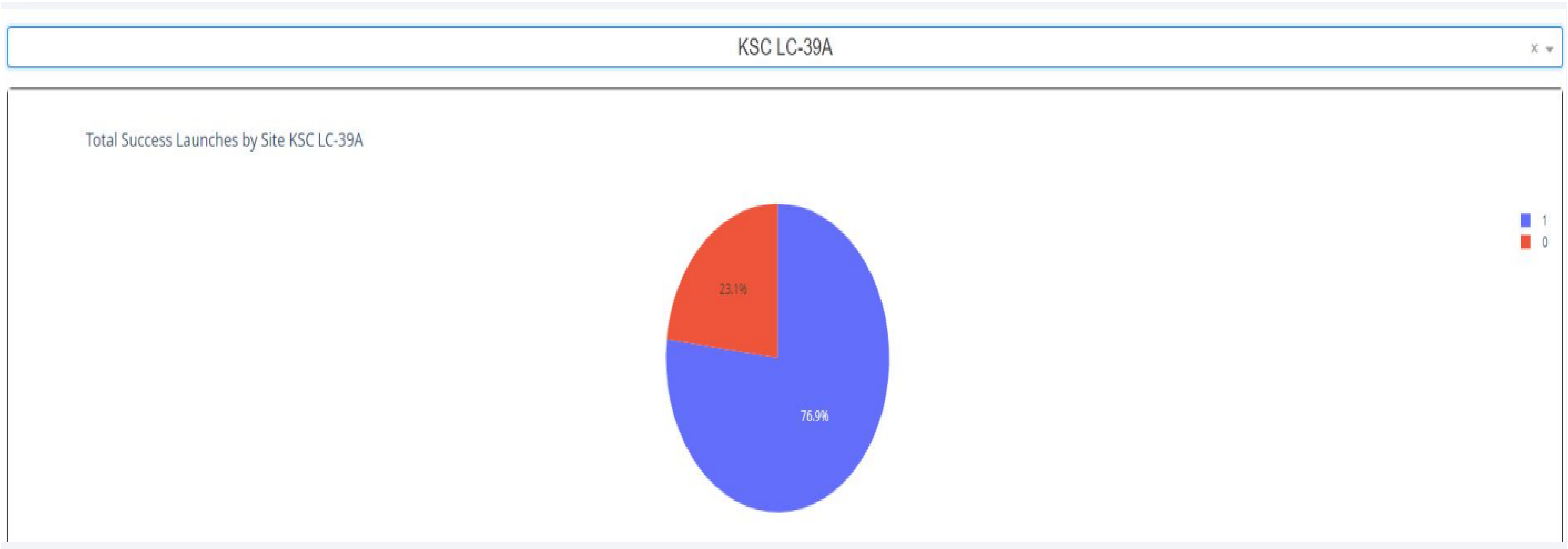
Launch Success Count Ratio For all site

[GitHub Link for the project]



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

Pie chart for the launch site with highest launch success ratio



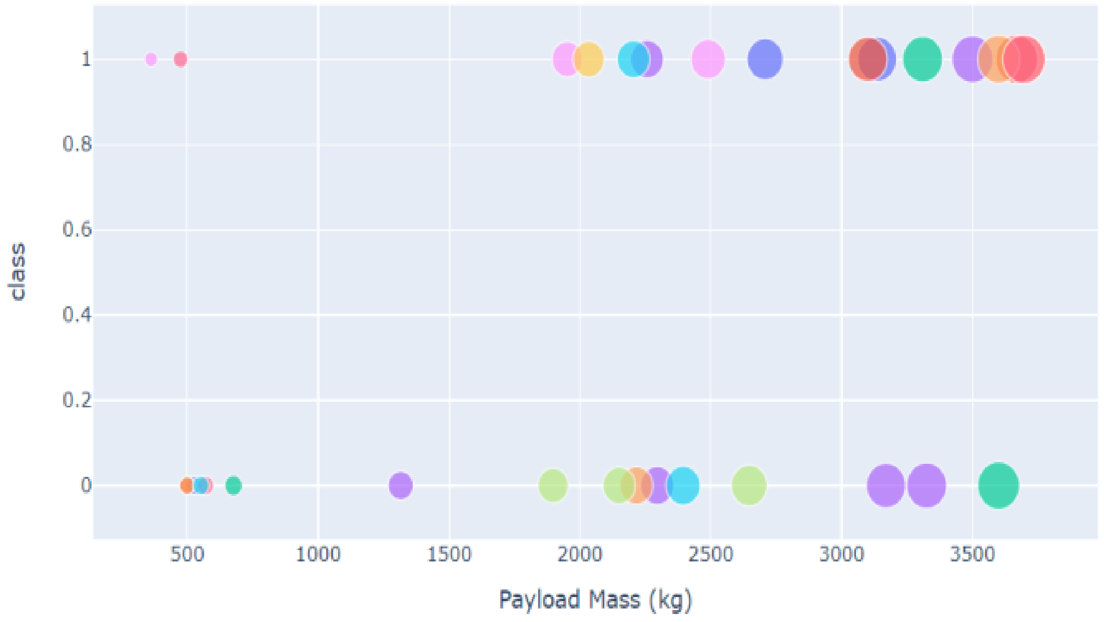
- The KSC LC-39A has the highest no of successful launches of all sites, and this pie chart shows the success and failure share.
- We can see that it has 76.9% success rate

[GitHub Link for the project]

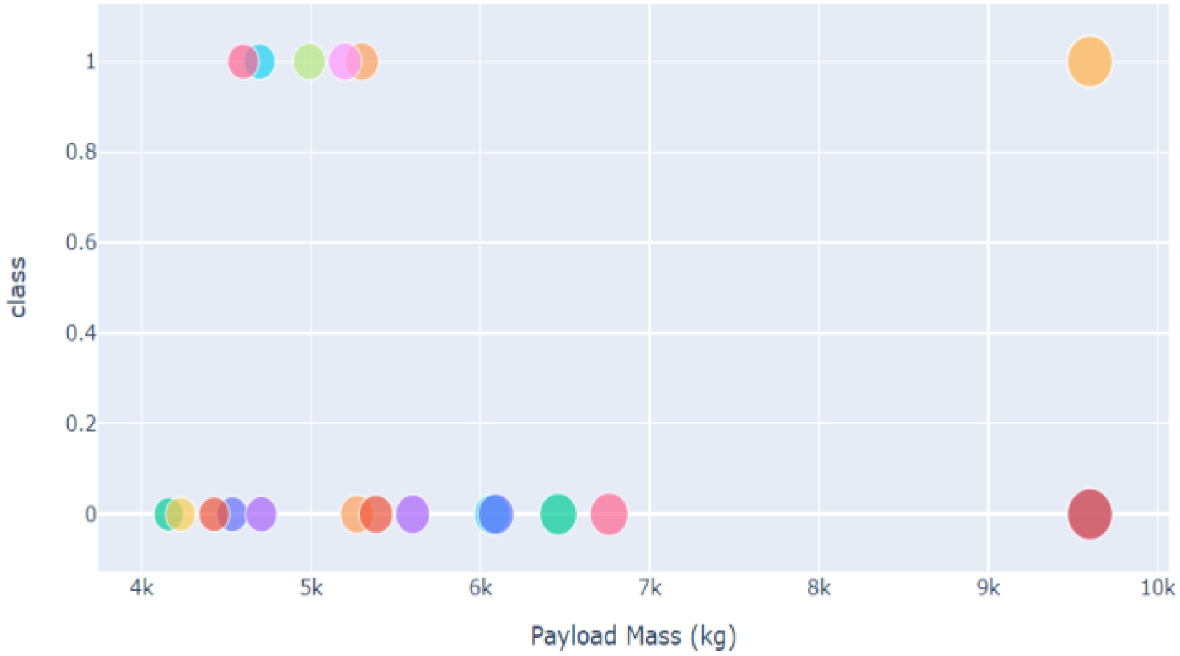
Payload vs. Launch Outcome

[GitHub Link for the project]

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg



We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

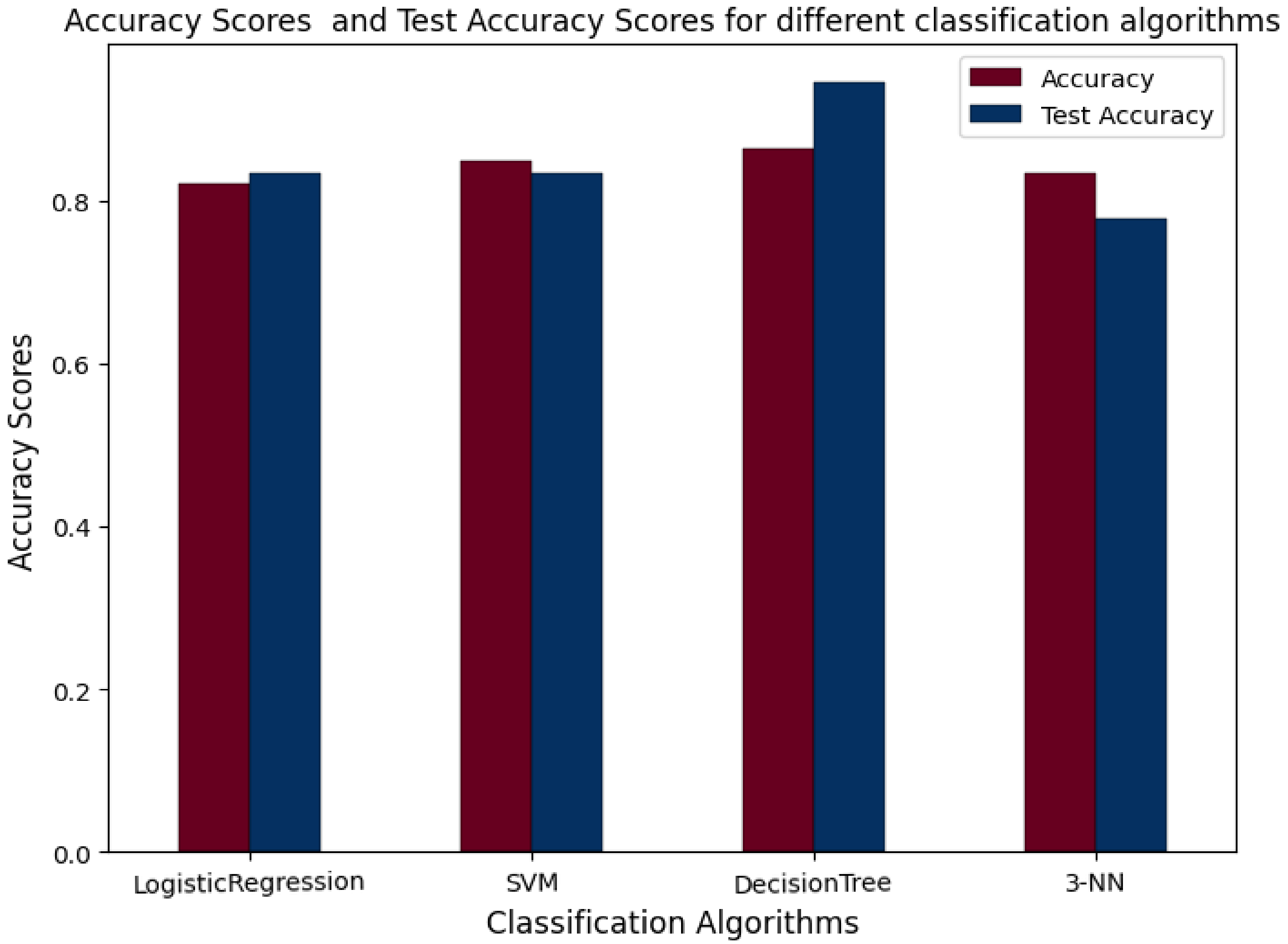
Section 5

Predictive Analysis (Classification)

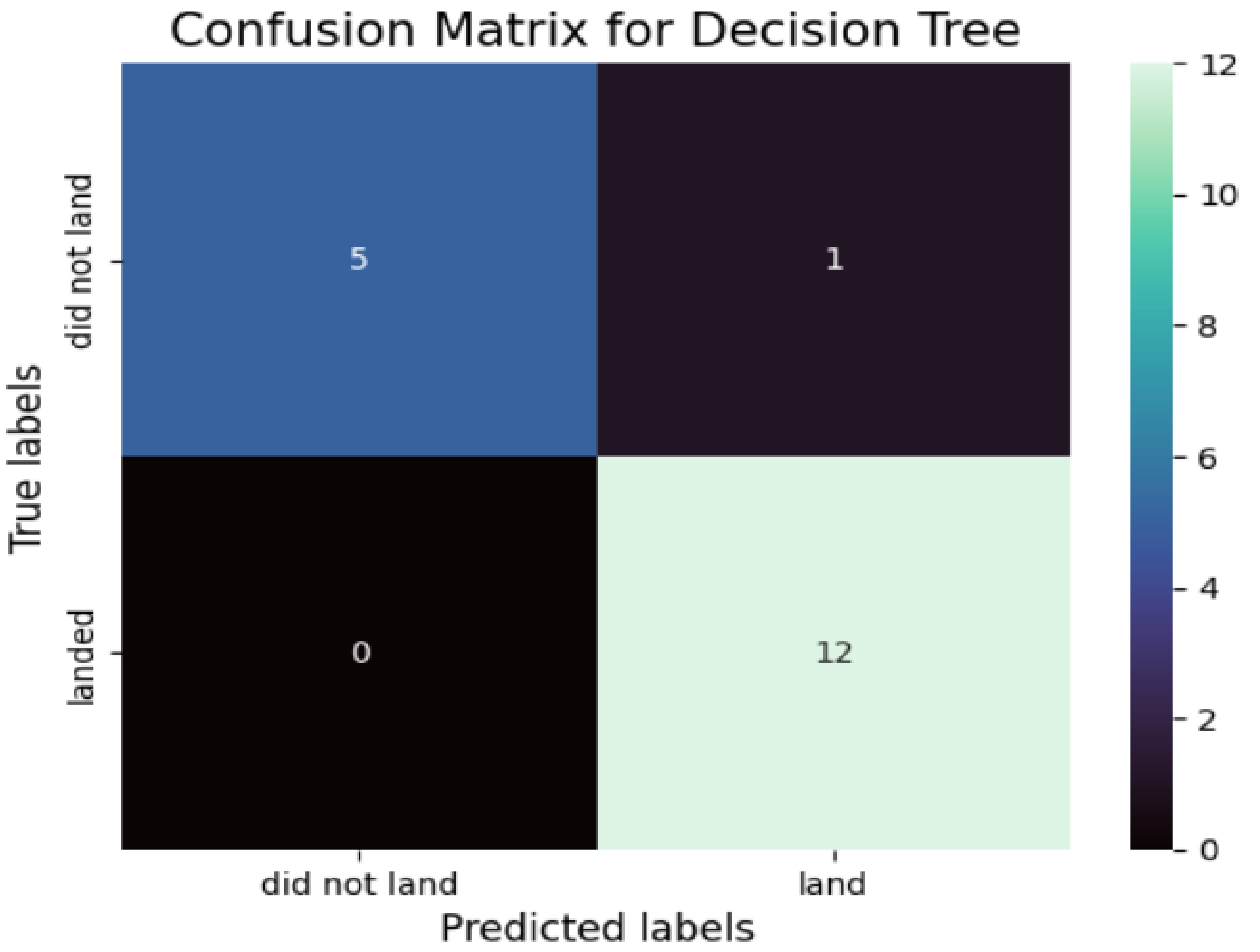
Classification Accuracy

- We can see that Decision Tree Classifier has the highest accuracy, while KNN with K=3 has the lowest accuracy score

[GitHub Link for the project]



Confusion Matrix



The Decision Tree correctly classified 17 test points and misclassified only 1 test data points

[GitHub Link for the project]

Conclusions

GitHub Link for the project

- ✓The Tree Classifier Algorithm is the best for Machine Learning for this dataset
- ✓Low weighted payloads perform better than the heavier payloads
- ✓The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- ✓We can see that KSC LC-39A had the most successful launches from all the sites
- ✓Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate.
- ✓Failure rate of new launches are low.
- ✓Launches with payloads over 8000kg have high Success Rate.

Thank you!