IBM DS0720EN Data Science and Machine Learning Capstone Project – Space X

USHA V

28th November 2021

IBM Developer

SKILLS NETWORK

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Discussion
  - Findings & Implications
- Conclusion
- Appendix

EXECUTIVE SUMMARY

- **Summary of Methodologies Used**
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) with SQL
  - EDA with data Visualization
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive Analysis using Classification
- **Output Summary**
  - EDA results
  - Interactive Analysis Demo using screenshots
  - Predictive Analysis results

INTRODUCTION

Project Background:

In this capstone, we will predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward 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. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Issues to be Solved:

- Factors that influences the rocket launch
- Effects of these factors that determine the success of launch
- Identify the best Method to adopt for a successful rocket Launch for SpaceX

Section 1

Methodology

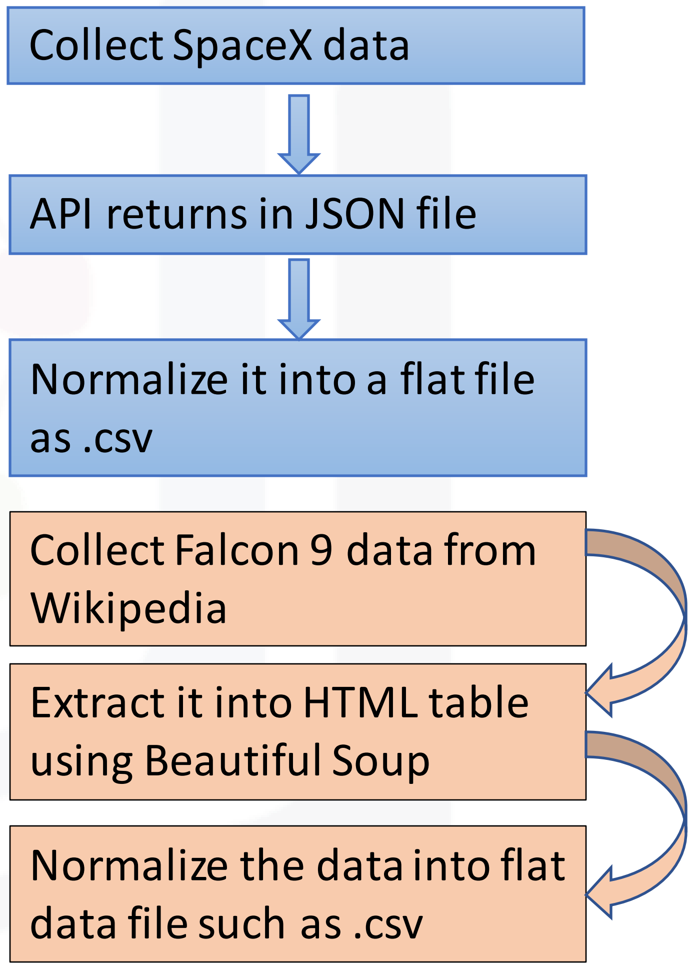
METHODOLOGY

Executive Summary

- Data collection methodology:
  - ➢ SpaceX REST API
  - ➢ Web Scraping from Wikipedia
- Perform data wrangling
  - ➢ Transform data using one hot encoding for Machine learning
  - ➢ Discard Irrevelant data
- Perform exploratory data analysis (EDA) using visualization and SQL
  - ➢ Plot Scatter charts and bar graphs to show the relationship between various data.
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - ➢ Apply all classification methods to identify the best model which gives good accuracy.

Data Collection

- SpaceX launch data is collected from SpaceX REST API.
- This data has information about the rocket used, payload, orbit, customer, mission outcome, landing outcome, etc.
- Our primary goal is to predict whether SpaceX can launch the rocket successively or not.
- SpaceX API url is https://api.spacexdata.com/v4/launches/past
- Another source for Web scraping the Falcon 9 rocket launch data is from Wikipedia with the url: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches



Data Collection – SpaceX API

GitHub Url to Notebook

Steps involved in the Data Collection:

1. Requesting rocket launch data from SpaceX API:

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

2. Request and parse the SpaceX launch data as a JSON file using the GET request and normalize it into a data frame

```
static_json_url='https://.../API_call_spacex_api.json'
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Apply functions to clean the data

```
getBoosterVersion(data), getLaunchSite(data),
getPayloadData(data), getCoreData(data)
```

Data Collection – SpaceX API contd...

4. Construct a dictionary with the required fields

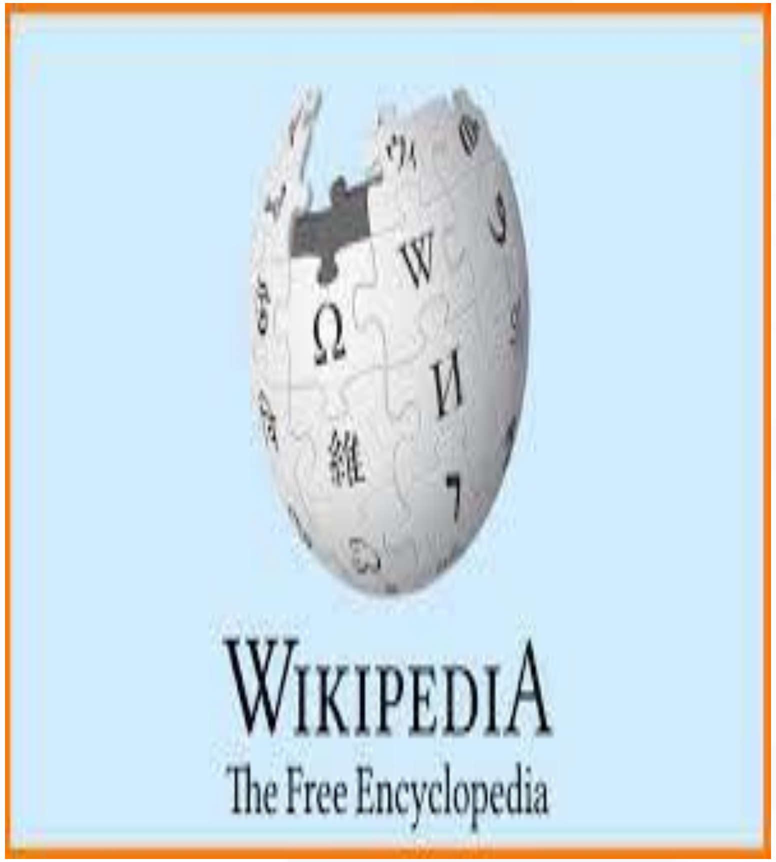
```
launch_dict = {'FlightNumber':list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,'PayloadMass':PayloadMass,
'Orbit':Orbit,'LaunchSite':LaunchSite,'Outcome':Outcome,
'Flights':Flights,'GridFins':GridFins,'Reused':Reused,
'Legs':Legs,'LandingPad':LandingPad,'Block':Block,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

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

5. Filter dataframe to include only Falcon 9 and export to .csv file

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection – Webscraping



1. Request the Falcon9 Launch Wiki page from its URL

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

2.Create a BeautifulSoup object

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

3.Find all tables and identify the launch table

```
html_tables = soup.find_all('table')
first_launch_table = html_tables[2]
```

4.Get the 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
```

Data Collection – Webscraping contd...

[GitHub Url to Notebook]

5. Create a dictionary by parsing the launch HTML tables

```
launch_dict= dict.fromkeys(column_names)
# Remove an irrelvant column
del launch_dict['Date and time ( )']
```

6.Fill up the launch_dict with launch records extracted from table rows.

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

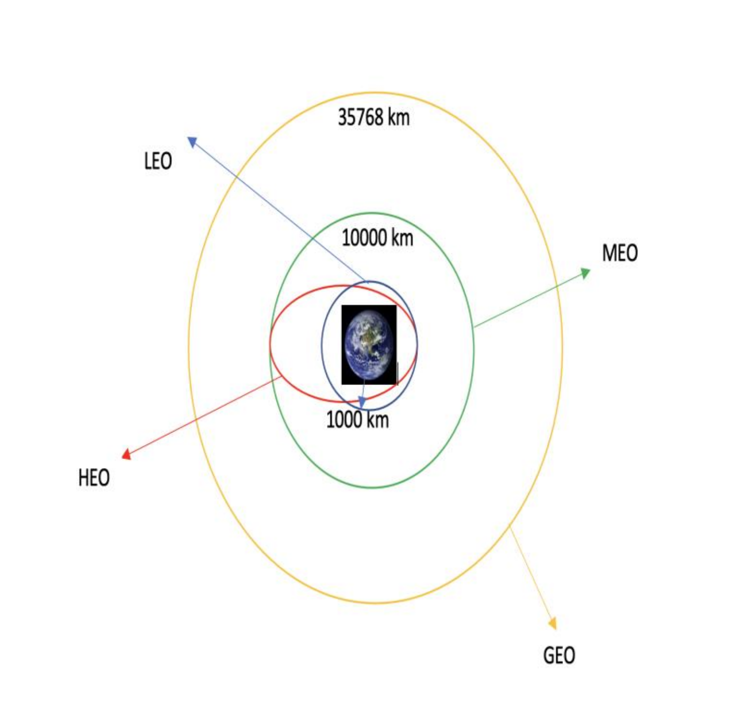
7.Convert dictionary to dataframe

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

8.Dataframe to .csv

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

Data Wrangling



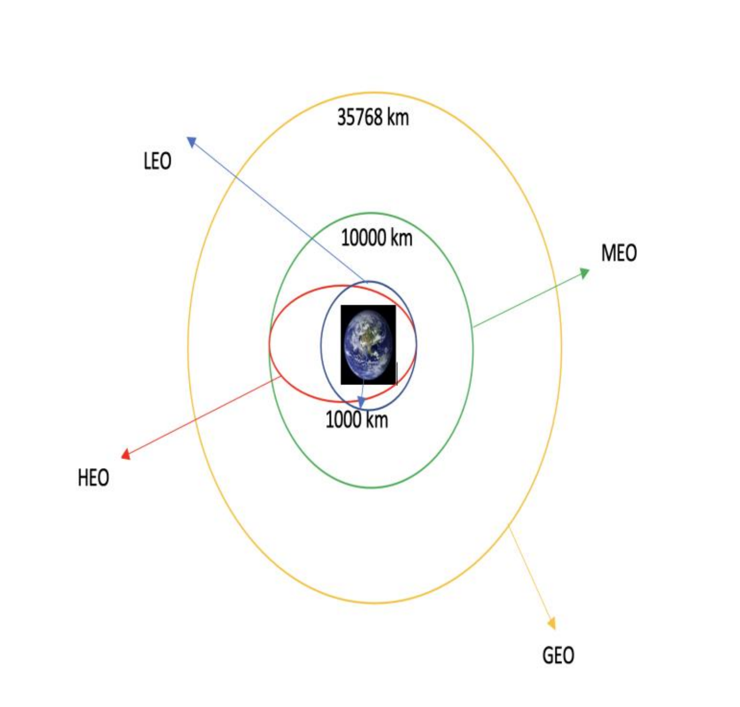
In this process, some Exploratory Data Analysis (EDA) is performed to find some patterns in the data and determine what would be the label for training supervised models.

Data Background:

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.

This step converts those outcomes into Training Labels with `1` means the booster successfully landed and `0` means it was unsuccessful.

Data Wrangling contd...



[GitHub Url to Notebook]

1. Load SpaceX dataset (.csv)

```
df=pd.read_csv(dataset_part_1.csv")
```

2. Identify and calculate the percentage of the missing values in each attribute

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

3. Calculate the number of launches on each site

```
df["LaunchSite"].value_counts()
```

4.Calculate the number and occurrence of each orbit

```
df["Orbit"].value_counts()
```

5.Calculate the number and occurence of mission outcome per orbit type

```
landing_outcomes = df["Outcome"].value_counts()
```

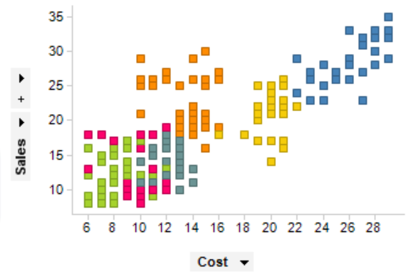
6. Create a landing outcome label from Outcome column and export it to .csv

```
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
      landing_class.append(0)
    else:
      landing_class.append(1)
df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

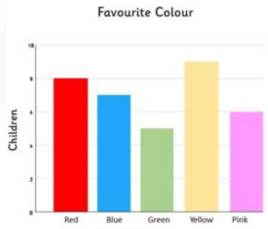
Loaded the dataset_part_2.csv file into a dataframe.
Scatter plots' primary uses are to observe and show relationships between two numeric variables. The dots in a scatter plot not only report the values of individual data points, but also patterns when the data are taken as a whole.
Following Scatter Plot graphs are drawn:

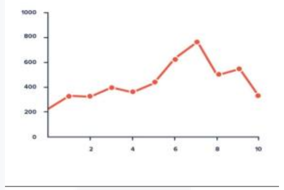
1. Flight Number Vs Payload Mass
2. Flight Number Vs Launchsite
3.Payload Mass Vs Launchsite
4. Flight Number Vs Orbit type



[GitHub Url to Notebook]

Bar Chart for **Success rate Vs Orbit** to study the success rate of each Orbit





Line graph to understand the **success** of launch in an ***yearly trend***

EDA with SQL

To Understand the SpaceX data set, the data is loaded into local MySql database and the following queries are answered

- *Display the names of the unique launch sites in the space mission*
- *Display 5 records where launch sites begin with the string 'KSC'*
- *Display the total payload mass carried by boosters launched by NASA (CRS)*
- *Display average payload mass carried by booster version F9 v1.1*
- *List the date where the first succesful landing outcome in drone ship was acheived.*
- *List the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000*
- *List the total number of successful and failure mission outcomes*
- *List the names of the booster_versions which have carried the maximum payload mass*
- *List the records which will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017*
- *Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.*

GitHub Url to Notebook

Build an Interactive Map with Folium

1. Mark all launch sites on a map

- Add each site's location on a map using site's latitude and longitude coordinates using folium Map object, with an initial center location to be NASA Johnson Space Center at Houston, Texas.
- Create and add `folium.Circle` and `folium.Marker` for each launch site on the site map.

2. Mark the success/failed launches for each site on the map

- created markers for all launch records. If a launch was successful `(class=1)`, then we use a green marker and if a launch was failed, we use a red marker `(class=0)`

3. Calculate the distances between a launch site to its proximities

- Using Haversine formula, the distance of the launch site from the nearest coastline, Railways, Highways and City is calculated.
- It is observed that the distance of the launchsite from the coastline is shorter than the other three proximities

GitHub Url to Notebook

Build a Dashboard with Plotly Dash

- Built a Plotly Dash application for users to perform interactive visual analytics on SpaceX launch data in real-time.
- The dashboard uses Flask and dash web Framework.
- This dashboard application contains input components such as a dropdown list and a range slider to interact with a **pie chart** and a **scatter point chart**.
- The drop down box is used to generate a pie chart interactively either for all launch site success or for a particular launch site.
  - Pie Chart is used to identify the site that has the largest successful launches
  - Launch that has the highest launch success rate
- The range slider is used to generate a scatter graph showing the relation between Outcome and PayLoad Mass for different Booster Versions. The range slider is used to change the PayLoad Mass interactively.
  - Scatter graph is used to find the Payload ranges that has the highest launch success rate
  - Lowest launch success rate too is found
  - Which Booster Version has the highest launch Success rate are identified.

GitHub Url to Notebook

Predictive Analysis (Classification)

Load the data into Pandas data Frame.

↓

Create a column for the Class field using Numpy array

↓

Standardize the data using transform function

↓

Split the data set into training and test data sets

↓

Apply logistic regression, SVM, decision tree and KNN methods on the training sample.

↓

Find the accuracy of the method using test samples

↓

Generate the confusion matrix for all the methods

↓

Identify the method which gives the best accuracy

GitHub Url to Notebook

Results

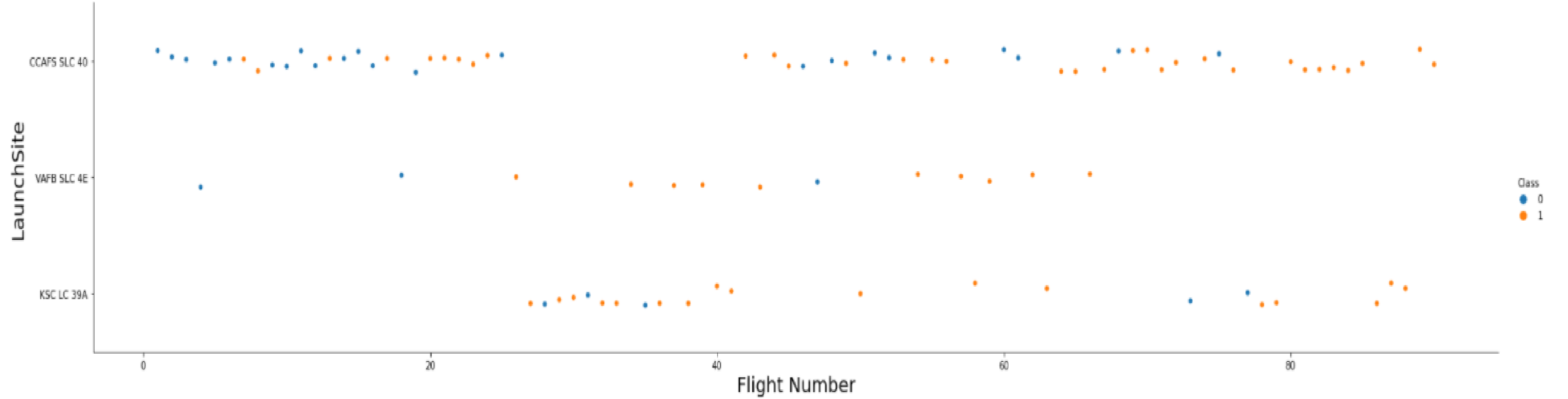
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number Vs Launch Site

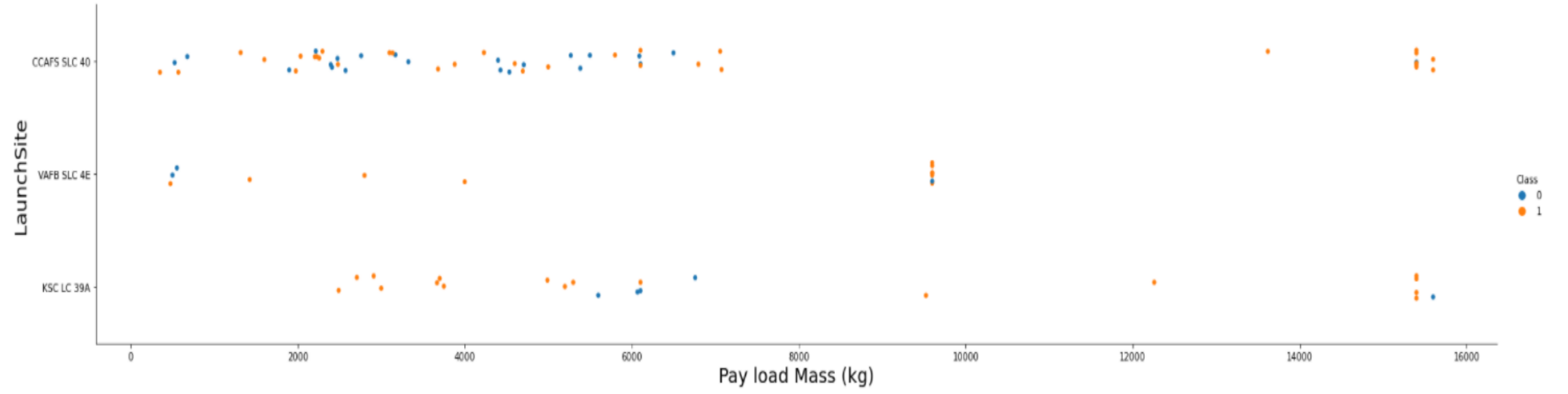
```python
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



From the above graph, it is apparent that CCAFS SLC 40 has more rocket launches than the other sites

Payload Vs Launch Site

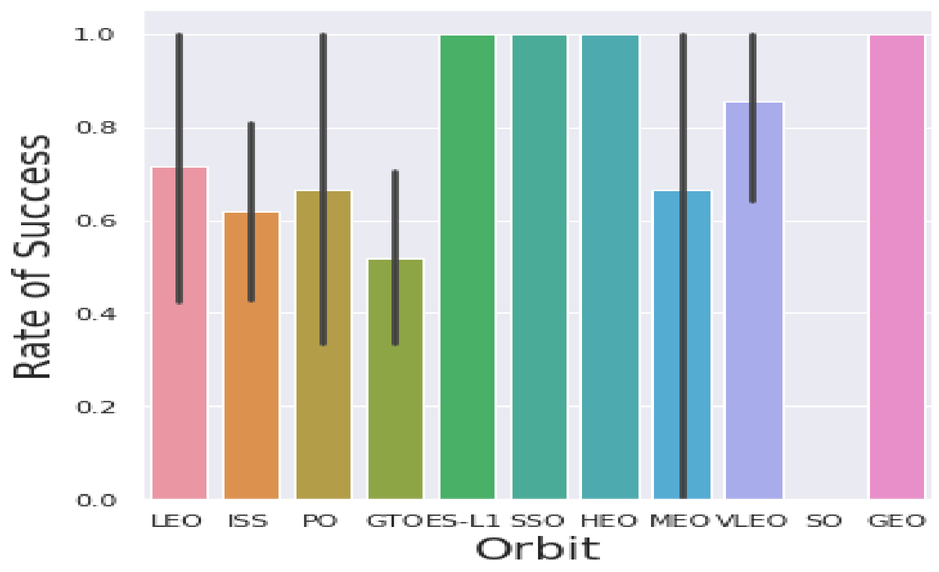
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class valu
e
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



It is observed that more rocket launches happens in the lighter Payload Mass than heavier Payload Mass. It is also observed that there are no rocket launches in the VAFB SLC site in the heavier Payload section.

Success Rate Vs Orbit Type

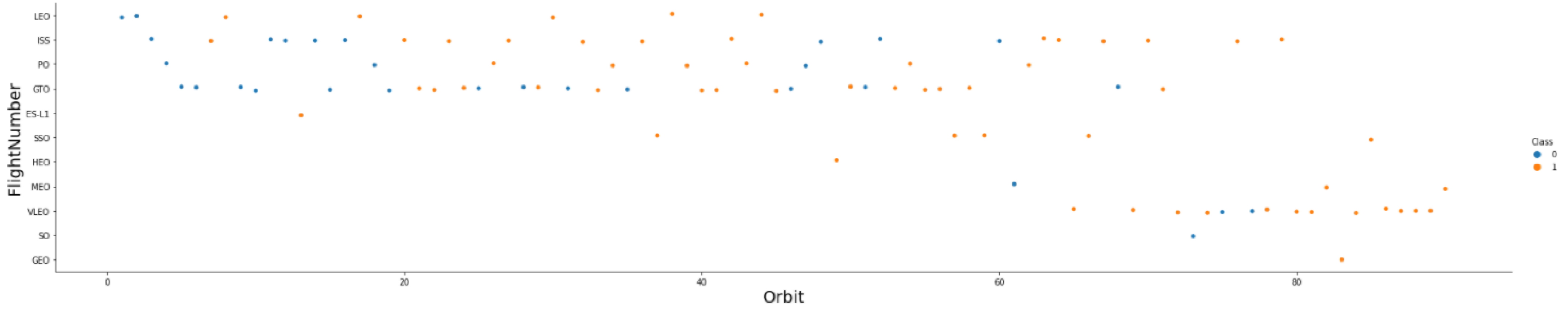
```
df.groupby(['Orbit']).mean()
sns.catplot(x="Orbit",y="Class",  kind="bar",data=df)
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Rate  of Success",fontsize=20)
plt.show()
```



Orbit ES-L1, SSO, HEO and GEO have better success rates than the other orbits.

Flight Number Vs Orbit Type

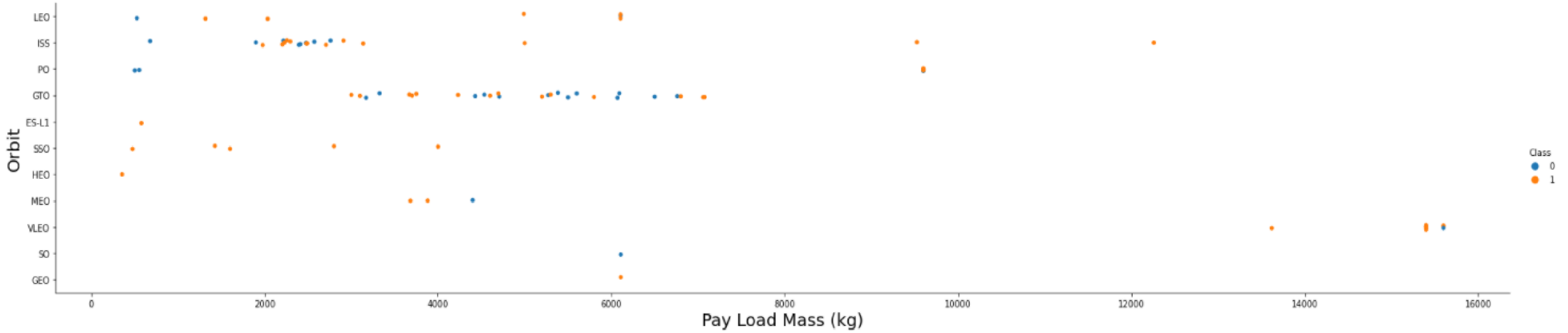
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("FlightNumber",fontsize=20)
plt.show()
```



It is clearly seen 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

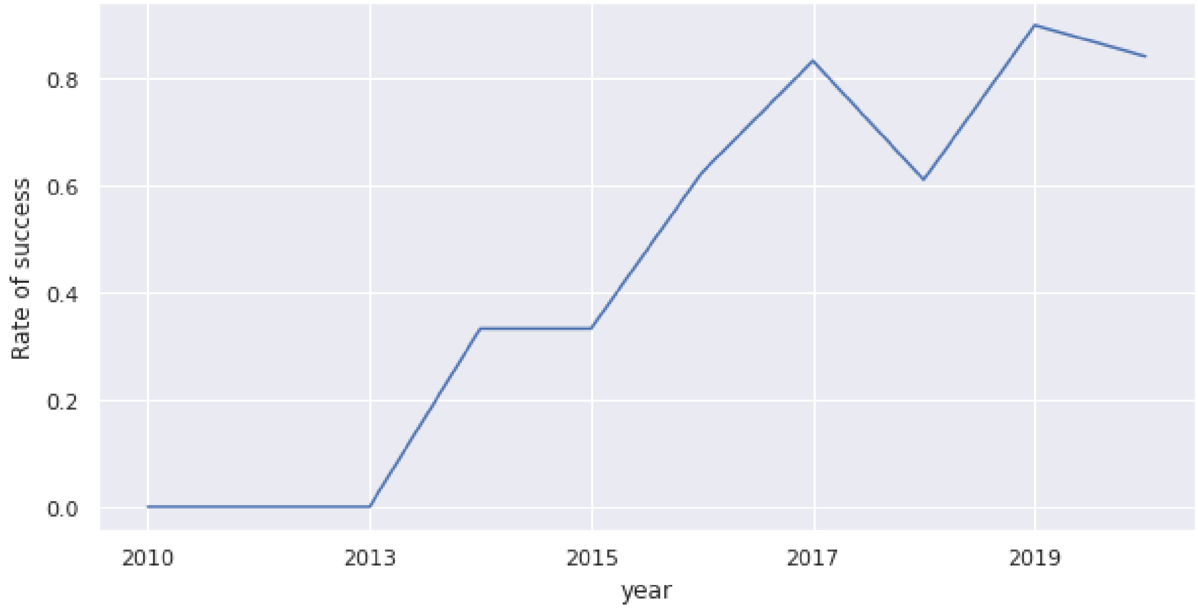
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay Load Mass (kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



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

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df1 = pd.DataFrame(Extract_year(df['Date']), columns = ['year'])
df1['Class'] = df['Class']
df1.groupby('year')['Class'].mean().plot(kind='line', figsize=(10, 5))
plt.xlabel('year')
plt.ylabel('Rate of success')
plt.show()
```



We can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

Query: select DISTINCT Launch_Site from spacex.spacextbl;

Output:

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

Explanation: The keyword 'DISTINCT' will fetch the unique launch sites from the table spacextbl

5 Records *where launch sites* Begin with 'KSC'

Query: select * from spacex.spacextbl WHERE Launch_Site LIKE 'KSC%' limit 5

Output:

```
Date       Time (UTC) Booster_Version      Launch_Site Payload             PAYLOAD_MASS Orbit      Customer   Mission Landing
                                                                                                            Outcome Outcome
19-02-2017 14:39:00   F9 FT B1031.1        KSC LC-39A SpaceX CRS-10          2490       LEO (ISS)  NASA (CRS) Success Success (ground
pad)
16-03-2017 06:00:00   F9 FT B1030          KSC LC-39A EchoStar 23            5600       GTO        EchoStar   Success    No attempt
30-03-2017 22:27:00   F9 FT  B1021.2       KSC LC-39A SES-10     5300        GTO        SES        Success    Success (drone ship)
01-05-2017 11:15:00   F9 FT B1032.1        KSC LC-39A NROL-76    5300        LEO        NRO        Success    Success (ground pad)
15-05-2017 23:21:00   F9 FT B1034          KSC LC-39A Inmarsat-5 F4          6070       GTO        Inmarsat   Success    No attempt
```

Explanation: The keyword 'LIKE' will fetch all the launch sites starting with KSC from the table spacextbl. The keyword LIMIT 5 will fetch the top 5 records in it.

Total Payload Mass *carried by boosters launched by NASA (CRS)*

Query: select SUM(PAYLOAD_MASS__KG_) as TotalPayloadMass from spacex.spacextbl where Customer = 'NASA (CRS)'

Output:
TotalPayloadMass
45596

Explanation: The keyword 'SUM' will calculate the sum of all PayLoad Mass for the Customer 'NASA (CRS) from the table spacextbl

Average Payload Mass by F9 v1.1

Query: `select AVG(spacex.spacextbl.PAYLOAD_MASS__KG_) as AveragePayloadMass from spacex.spacextbl where Booster_Version = 'F9 v1.1'`

Output:

```
AveragePayloadMass
2928.4
```

Explanation: The keyword 'AVG' will calculate the average of Payload Mass for the Booster Version 'F9 v1.1' from the table spacextbl

First Successful Landing in Drone Ship: Date

Query: select MIN(spacex.spacextbl.Date) from spacex.spacextbl where spacex.spacextbl.Landing_Outcome = 'Success (drone ship)'

Output:

```
MIN(spacex.spacextbl.Date)
08-04-2016
```

Explanation: The keyword 'MIN' will find the first date of successful landing in drone ship from the table spacextbl

Successful Ground Pad Landing with Payload between 4000 and 6000

Query: select Booster_Version from spacex.spacextbl where Landing_Outcome = 'Success (ground pad)' AND spacex.spacextbl.PAYLOAD_MASS__KG_ > 4000 AND spacex.spacextbl.PAYLOAD_MASS__KG_ < 6000

Output:

```
Booster_Version
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1
```

Explanation: This query will fetch all booster versions which have successful ground pad landing and payload between 4000 and 6000 kg from the table spacextbl

Total Number of Successful and Failure Mission Outcomes

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

Output:

Successful_Mission_Outcomes	Failure_Mission_Outcomes
100	1

Explanation: This query uses the keyword COUNT to count all successful Mission Outcomes as well as Failure Mission Outcomes from the table spacextbl

Boosters Carried Maximum Payload

Query: select BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);

Output:

```
boosterversion
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
```

Explanation: This query uses the sub query to find the maximum PayLoad Mass from the table spacextbl. This maximum value is used to find the Booster Versions which has the maximum Payload Mass

2017 Launch Records

Query: SELECT monthname(DATE). landing_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL where YEAR(DATE)='2017' and landing_Outcome='Success (ground pad)'

Output:

monthname(DATE)	landing_OUTCOME	BOOSTER_VERSION	LAUNCH_SITE
February	Success (ground pad)	F9 FT B1031.1	KSC LC-39A
May	Success (ground pad)	F9 FT B1032.1	KSC LC-39A
June	Success (ground pad)	F9 FT B1035.1	KSC LC-39A
August	Success (ground pad)	F9 B4 B1039.1	KSC LC-39A
September	Success (ground pad)	F9 B4 B1040.1	KSC LC-39A
December	Success (ground pad)	F9 FT B1035.2	CCAFS SLC-40

Explanation: This query uses the function month() to fetch the month names and year() function to fetch the year 2017 from the table spacextbl

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

Query: SELECT date,LANDING_OUTCOME,row_number() over (order by date desc) as Ranks FROM SPACEXTBL WHERE landing_outcome like '%Success%' and DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC;

Output:

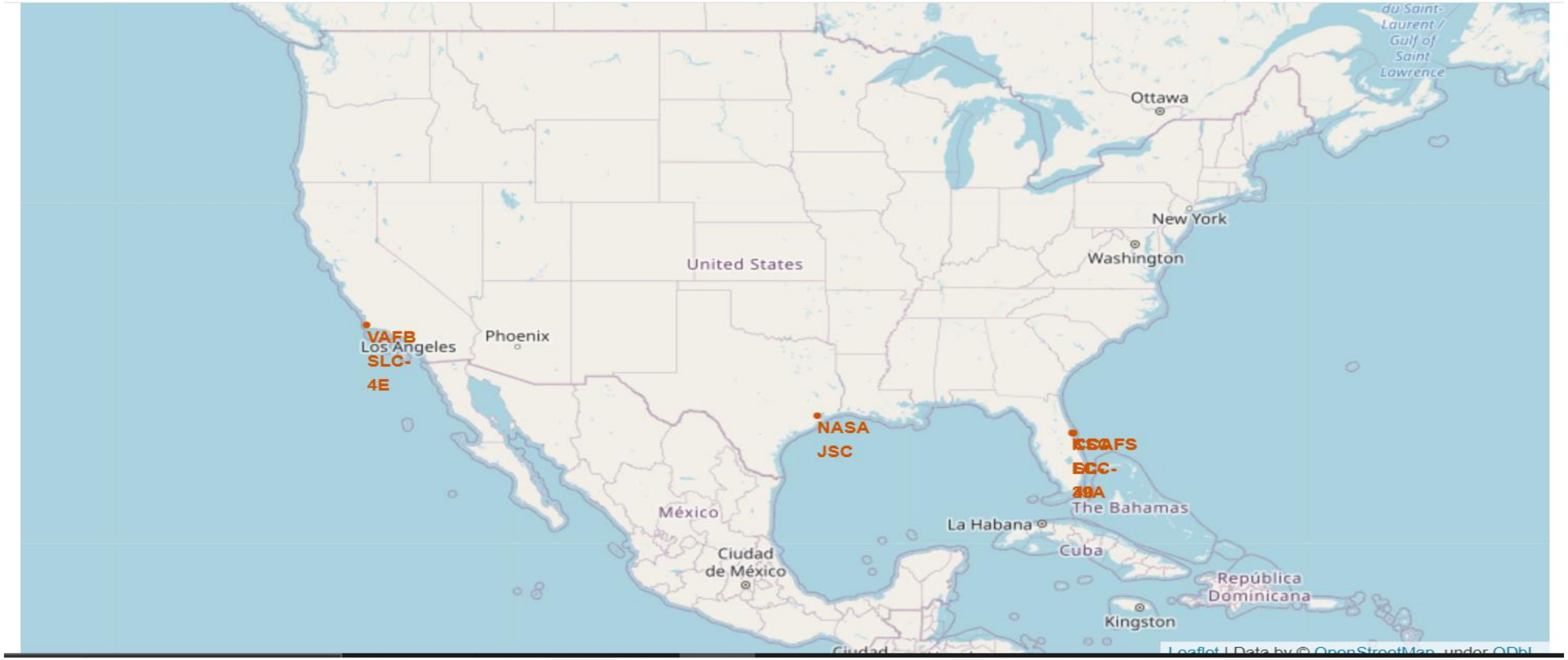
date	LANDING_OUTCOME	Ranks
19-02-2017	Success (ground pad)	1
14-01-2017	Success (drone ship)	2
14-08-2016	Success (drone ship)	3
18-07-2016	Success (ground pad)	4
27-05-2016	Success (drone ship)	5
06-05-2016	Success (drone ship)	6
08-04-2016	Success (drone ship)	7
22-12-2015	Success (ground pad)	8

Explanation: This query uses the function rownumber() to generate ranks

Section 4

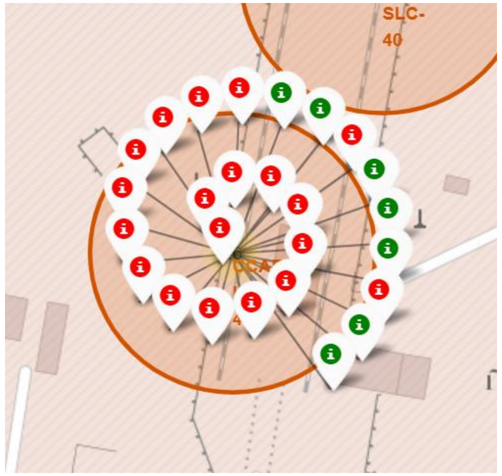
Launch Sites Proximities Analysis

All Launch Sites on a Map

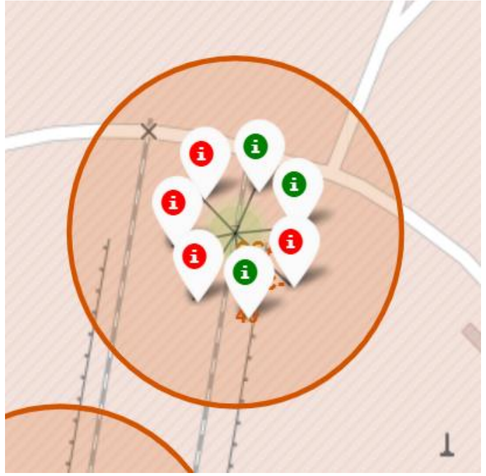


All launch Sites are in California and Florida with Nasa in Houston as Center

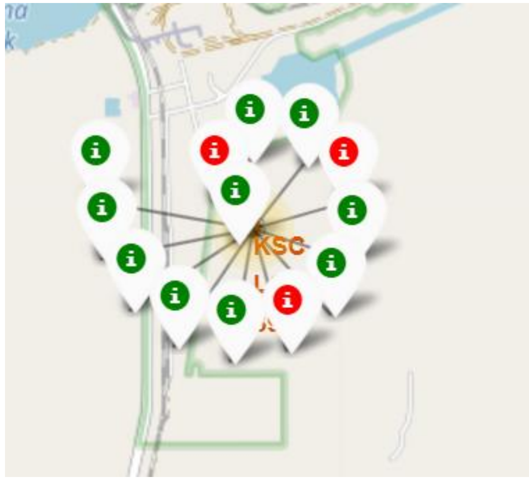
Mark the success/failed launches for each site on the map



CCAFS LC-40



CCAFS SLC-40



KSC LC-39A

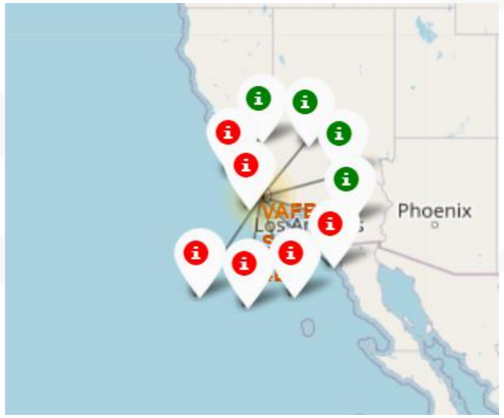
Florida

Success Launch

Failure Launch

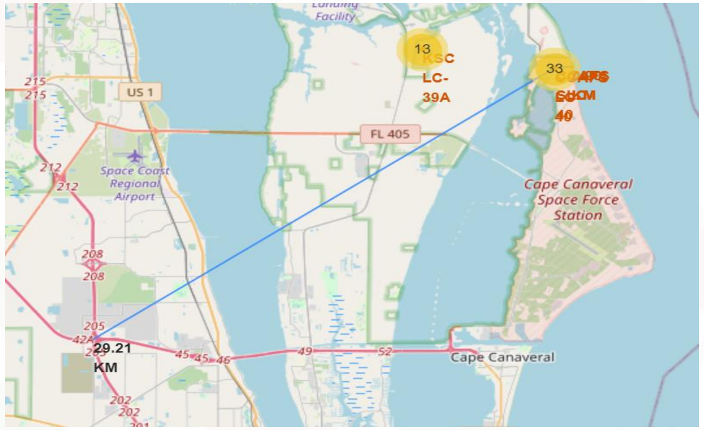
VAFB SLC-4E

California

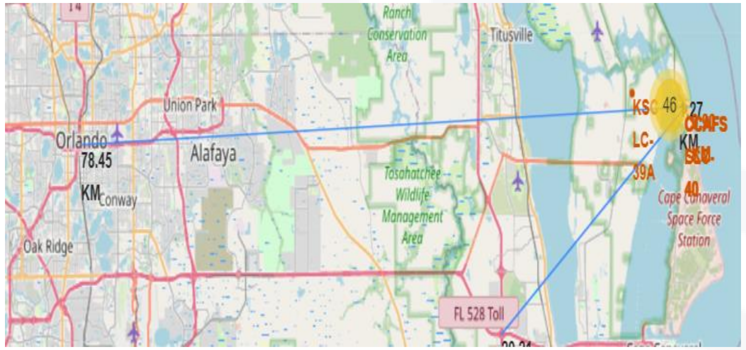


Launch Site CCAFS SLC-40 to its Proximities

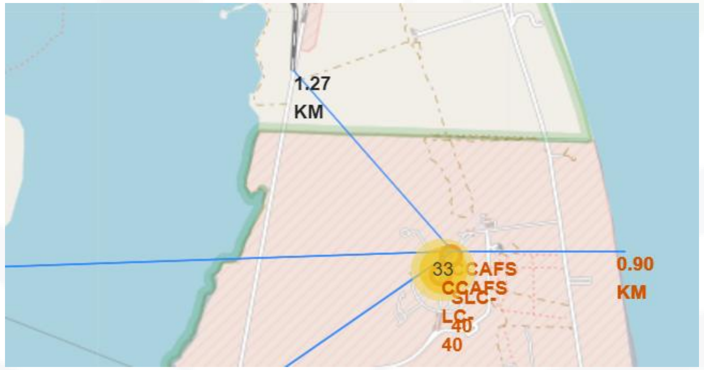
Coastline – 0.9 km



Highways – 29.21 km



Nearest City Florida – 78.45 km

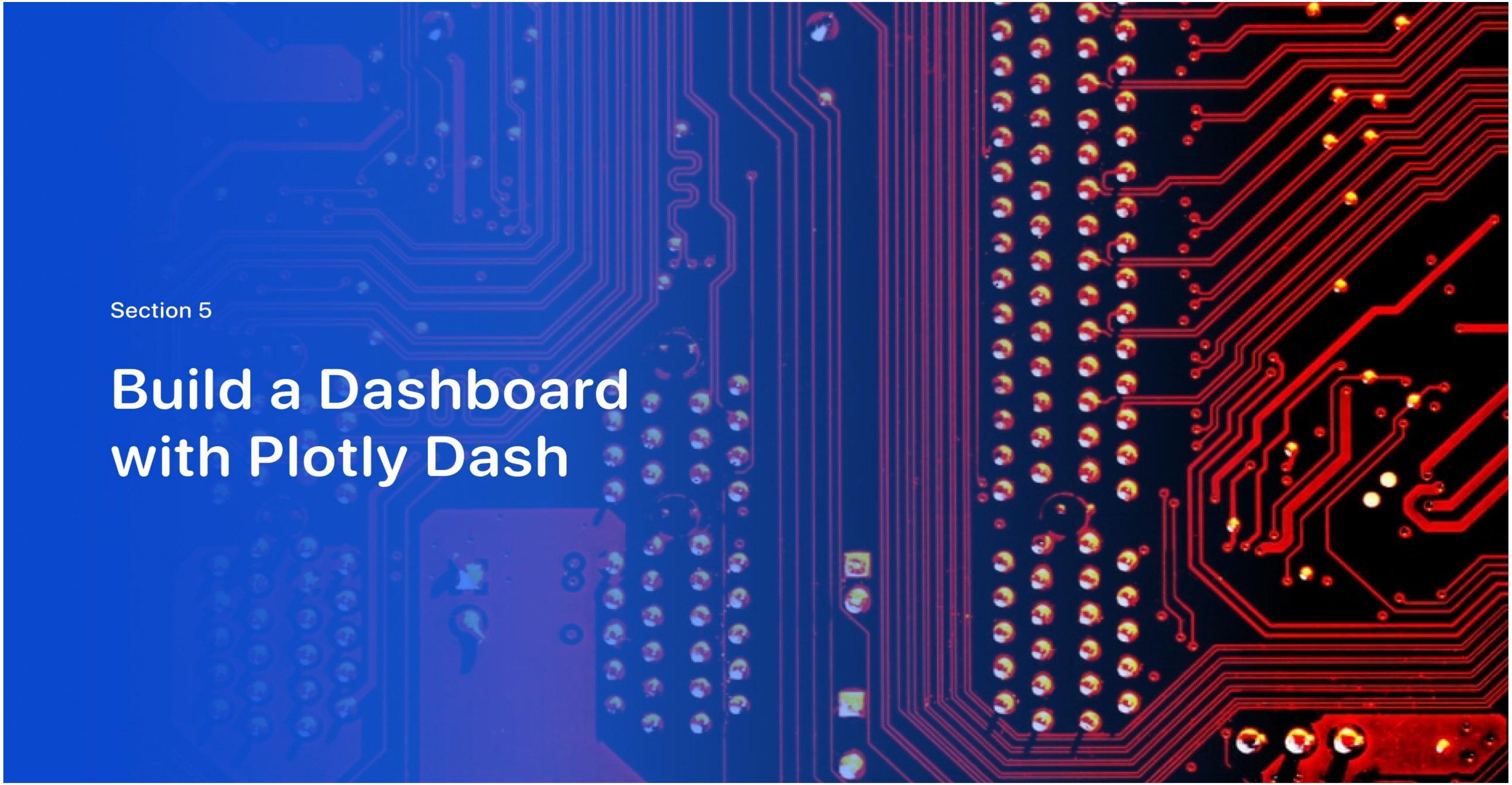


Railways – 1.27 km

Launch Sites are in close proximity to Coastline and Railways compared to Highways and Nearest City.

Section 5

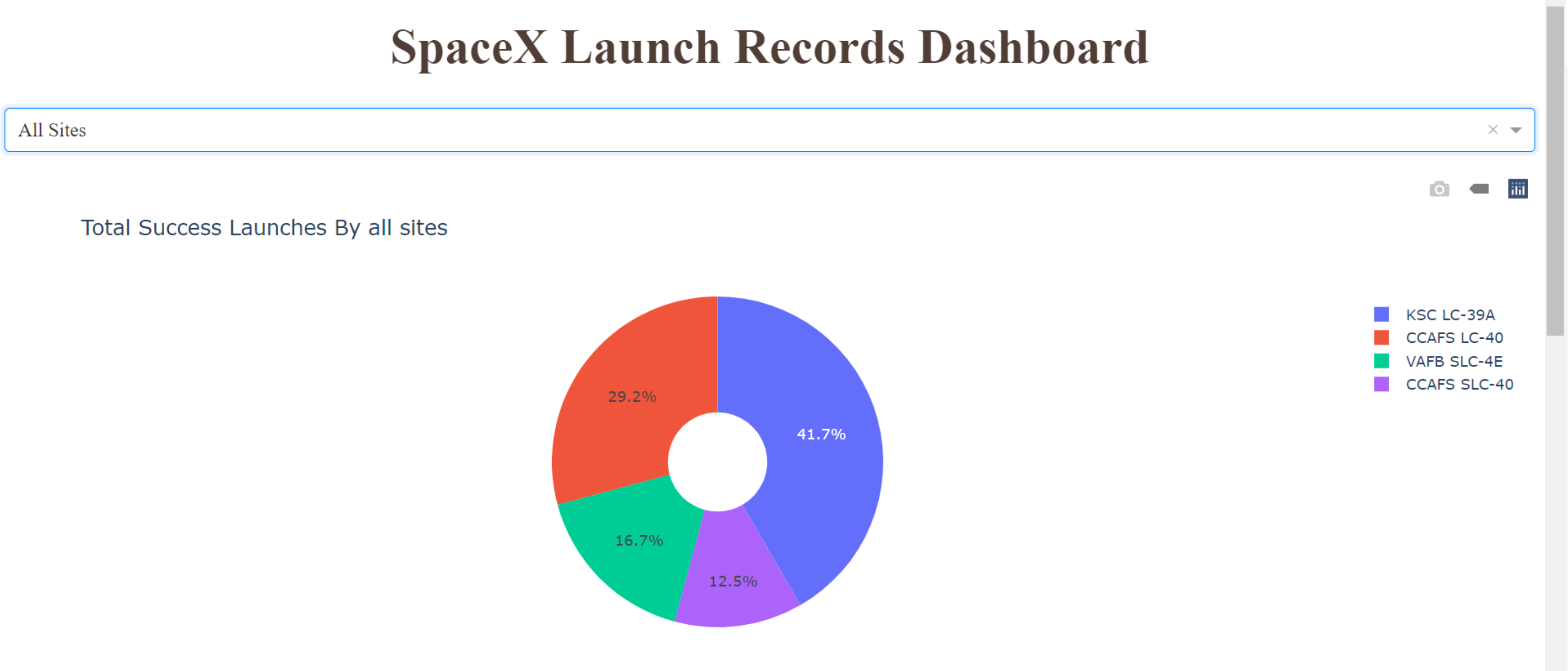
Build a Dashboard with Plotly Dash



IBM Developer

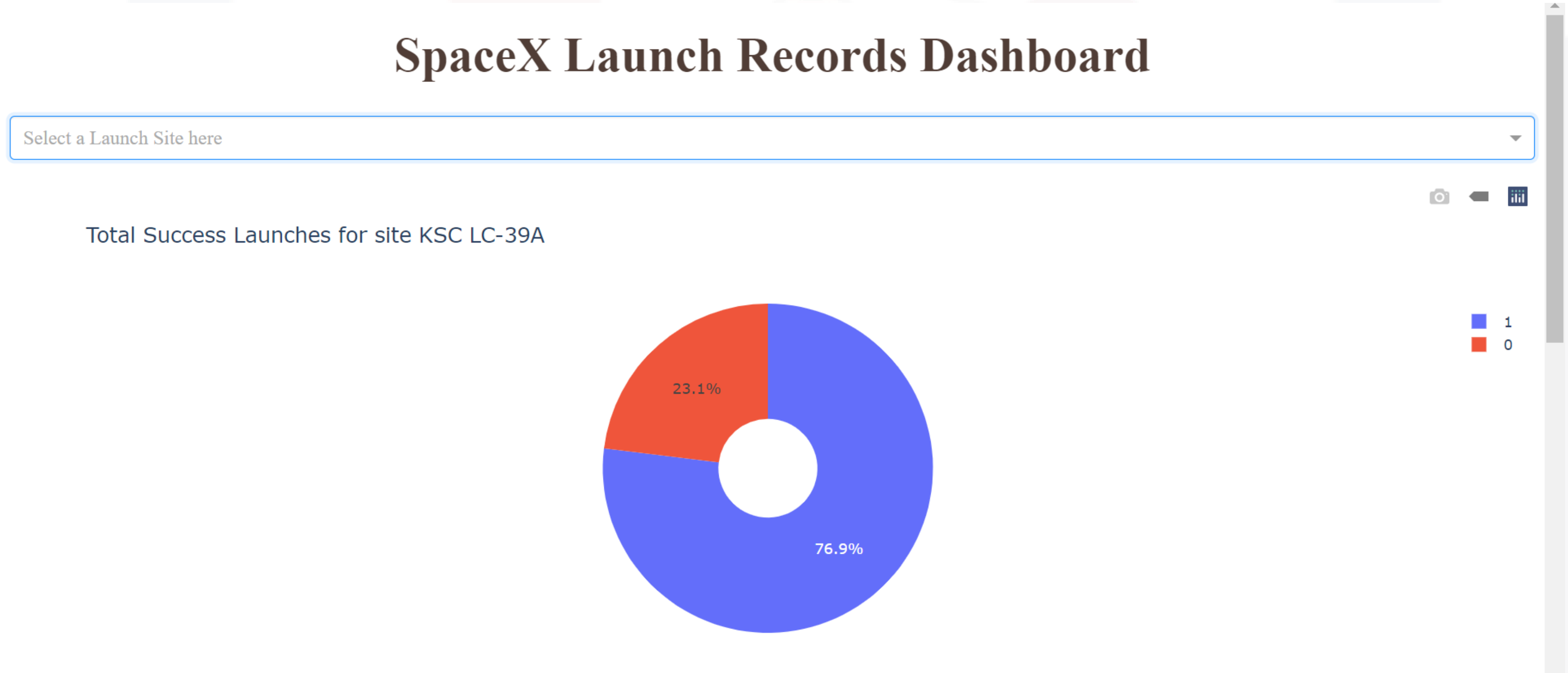
SKILLS NETWORK

Total Success Launches By All Sites



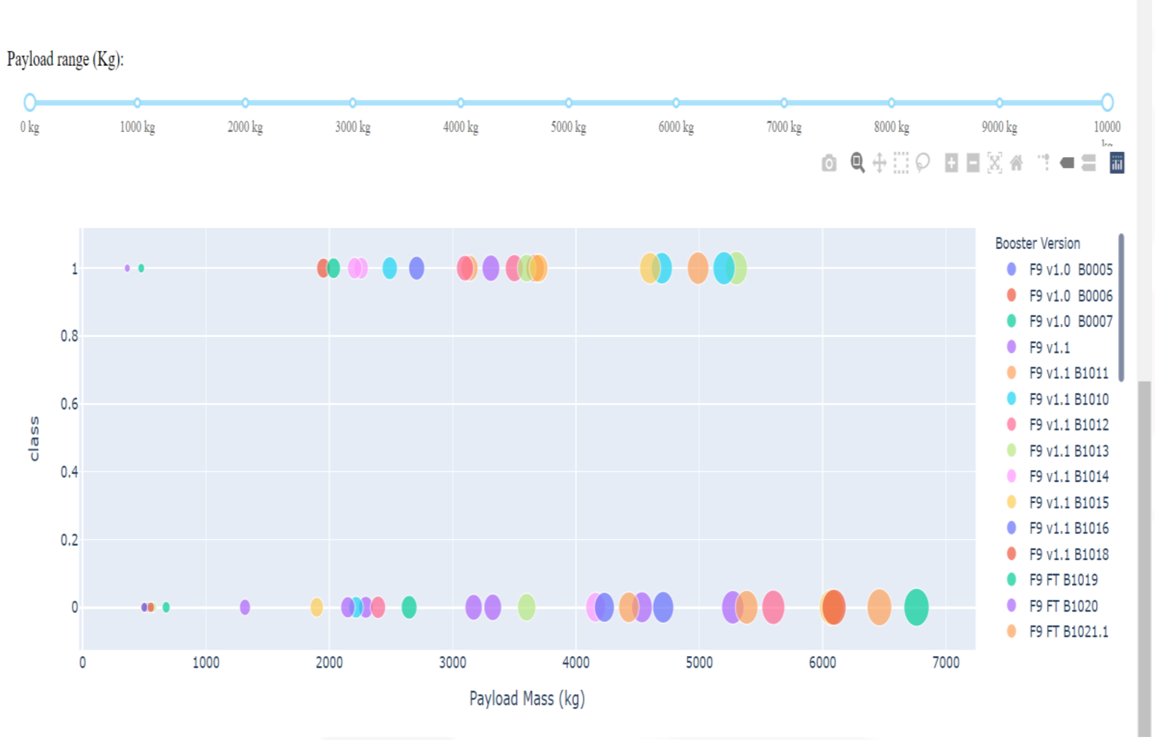
It is observed that KSC LC-39A has more successful launches than other Sites

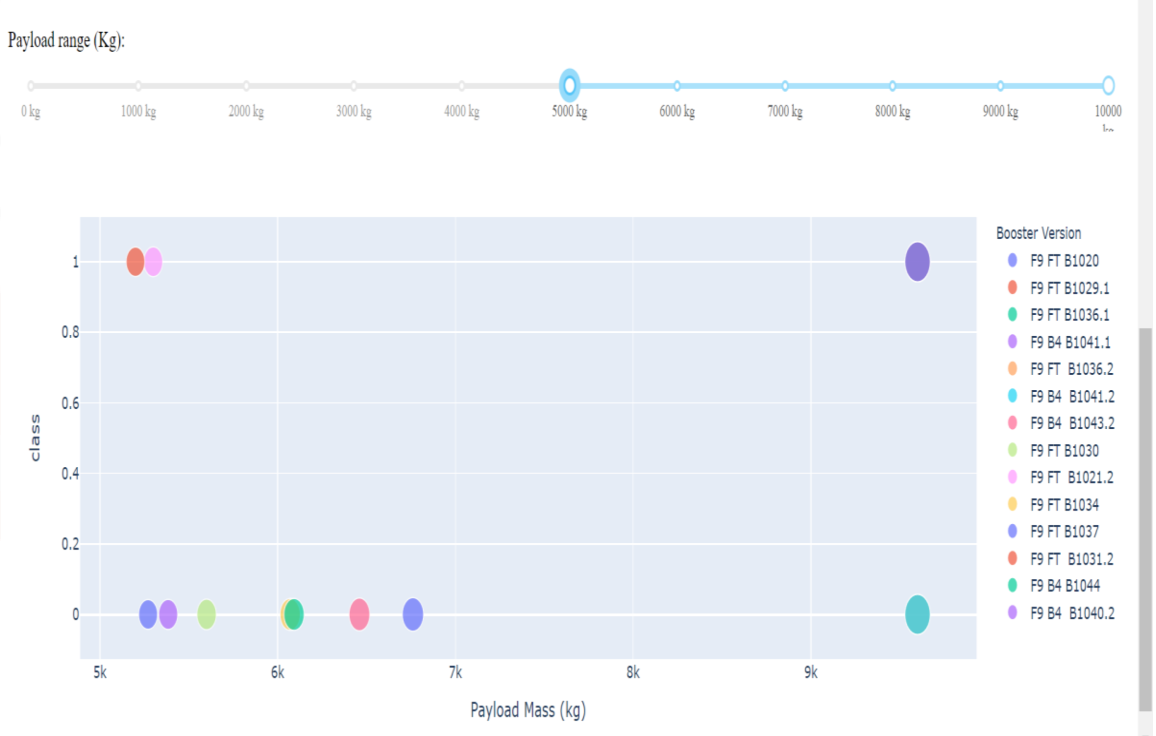
Site with Highest Launch Success Ratio



The success rate of KSC LC-39A is 76.9% while the failure rate is below 25% which means 3 out of 4 rocket launches leads to a success

Payload Vs Launch Outcome Dashboard





It is apparent that the number of Successful launches is more with less Payload Mass (less than 5000 kg) compared to more Payload (5000 to 10000 kg)

Section 6

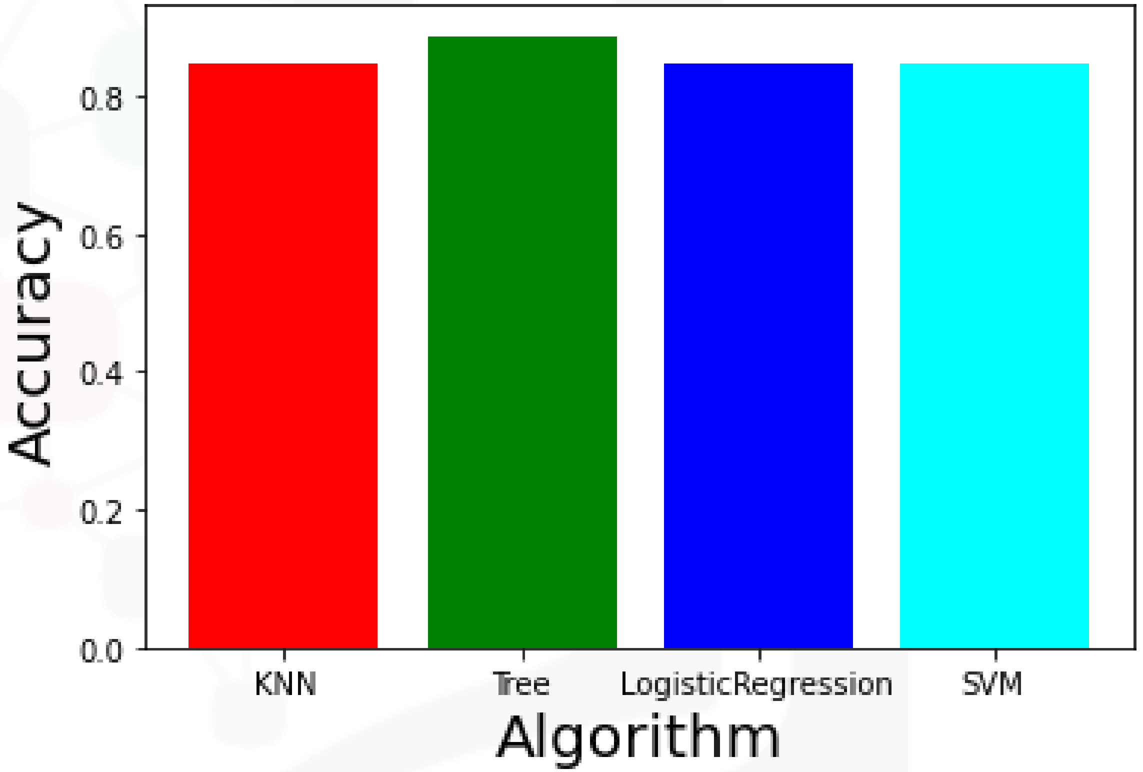
Predictive Analysis (Classification)

Classification Accuracy

From the bar graph it is observed that Decision Tree Classification Algorithm gives the best accuracy when applied on the training data set although other algorithms accuracy' varies by a little percentage.

The accuracy of Decision Tree Algorithm on the test data is

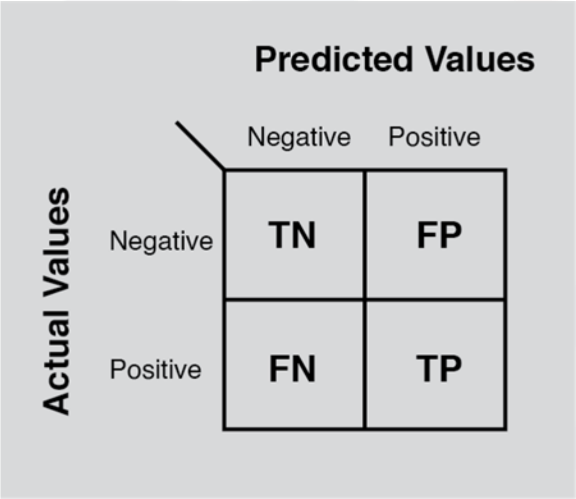
Accuracy= 0.8333333333333334

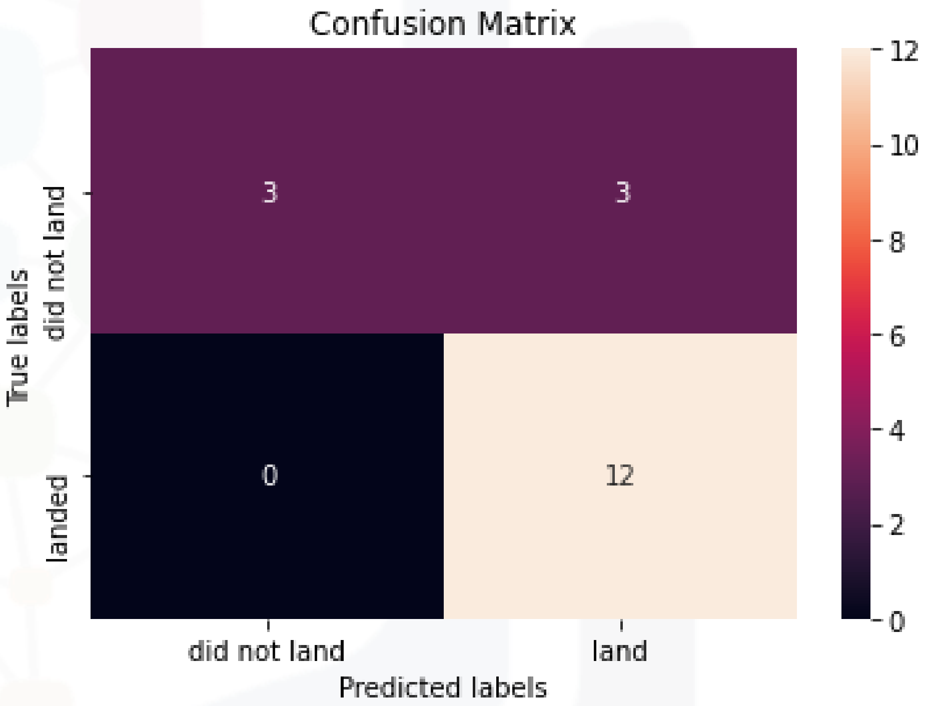



```
Best Algorithm is Tree with a score of 0.8875000000000002
Best Params is : {'criterion': 'entropy', 'max_depth': 2, 'max_features': 'sqrt','min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'best'}
```

Confusion Matrix for Decision Tree

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





Conclusions

- **CCAFS SLC 40** has more rocket launches. But the success ratio is high in **KSC LC-39A** launch site.
- Number of Successful launches is more with less Payload Mass (less than 5000 kg) compared to more Payload (5000 to 10000 kg).
- Orbit ES-L1, SSO, HEO and GEO have better success rates than the other orbits.
- The success rate increases year on year since 2013.
- Launch Sites are in close proximity to Coastline and Railways compared to Highways and Nearest City (Florida).
- Decision Tree Algorithm gives better accuracy for this training data set compared to Logistic Regression, SVM and KNN algorithms.

Appendix

Haversine formula:

The **haversine formula** determines the great-circle distance between two points on a sphere given their longitudes and latitudes, that is, the shortest distance over the earth's surface – giving an 'as-the-crow-flies' distance between the points

Formula:

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi 1 \cdot \cos\varphi 2 \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right)$$

$$c = 2 \cdot \operatorname{atan2}\left(\sqrt{a}, \sqrt{(1 - a)}\right)$$

$$d = R \cdot c$$

Where ϕ is latitude, λ is longitude, R *is earth's radius (mean radius = 6,371km); note that angles need to be in radians to pass to trig functions!*

Used above formula to calculate proximities with the launch site


Thank you!