



IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Smita Tiwari>

<25-04-2022>



Outline

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

Executive Summary

- **Summary of methodologies**

- Data Collection from SpaceX launch data using an API
- Data Collection using web scrapping related to Wiki pages
- Data wrangling using API to transform raw data into a clean data.
- EDA using SQL
- EDA using Pandas and Matplotlib
- Building an attractive dashboard using plotly
- Predictive Analysis using various Machine Learning Classifier and their comparative analysis for SpaceX Falcon 9 launch of rocket.

- **Summary of all results**

- The data was successfully collected from various public sources about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome. The extracted raw data was successfully cleaned using data wrangling.
- The correlation between the features were identified and visualized using EDA. The best features were selected to provide input to the predictive analysis using various classification algorithms.
- The predictive analysis was done using various ML algorithms. Based on the comparative analysis, the best classifier was obtained to predict the successful launch of SpaceX Falcon satellite.

Introduction

- **Project background and context**

The objective is to determine the cost of a Spaces X's Falcon 9 launch during the first stage of landing of rockets. The objective can be achieved by training a machine learning model and using public information to predict if SpaceX will reuse the first stage or not. The trained model can be used if an alternate company wants to bid against SpaceX for a rocket launch. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars and other providers cost upwards of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

- **Problems you want to find answers**

- Parameters to decide successful landing of rockets.
- Relationship between various independent variable and dependent variables for the outcome.
- Features responsible for achieving best classifiers for predicting successful launch of the SpaceX falcon 9 rocket and the total cost of the launches.
- Find the best place to make the launches.

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**

Data Collection from SpaceX launch data was done using two approaches :

- Using API .
- Using web scrapping related Wiki pages

- **Perform data wrangling**

Data wrangling using API was done to transform the raw data obtained from data collection phase into a clean dataset to obtain meaningful data. This step involves dealing with Null values and one hot encoding and label encoding of the sting values in the columns of the tables.

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL

The EDA was done matplotlib and pandas library. SQL was used to query the data. The correlation between the features and patterns between the features were identified using scatter and bar graphs

- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

The predictive analysis was done using various ML algorithms. Based on the comparative analysis, the best classifier was obtained to predict the successful launch of SpaceX Falcon satellite.

Data Collection

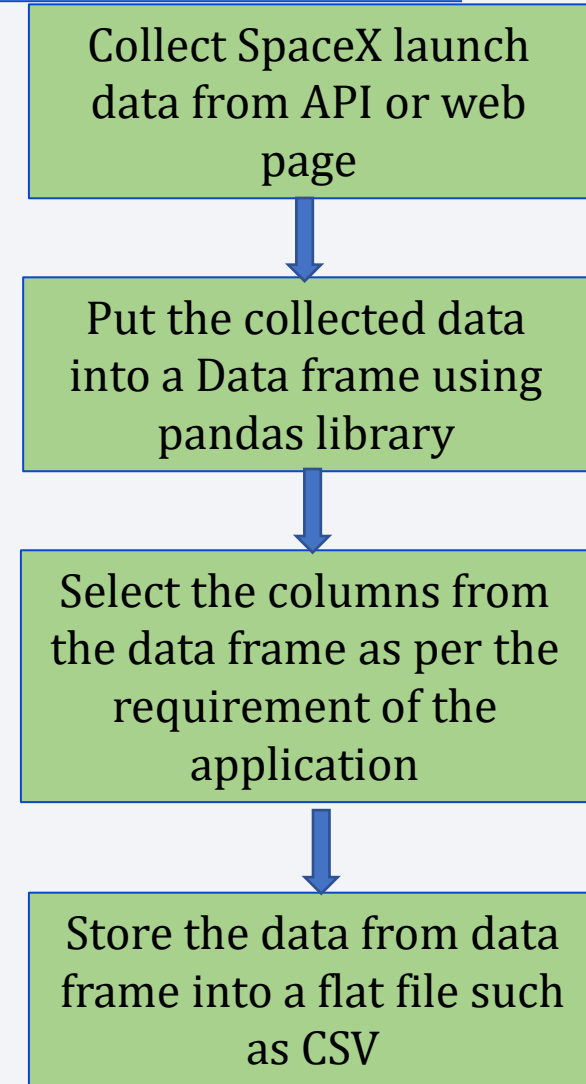
Data Collection from SpaceX launch data was done using two approaches :

- **Using API**

The API used for data collection was SpaceX REST API. The collected data was about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome. These data were used to make the prediction using Machine Learning classifier whether the SpaceX will attempt to land a rocket or not. Space X API link (<https://api.spacexdata.com/v4/rockets/>)

- **Using web scrapping related**

The data for SpaceX Falcon 9 Launch was collected using web scraping related Wiki pages. The Python BeautifulSoup package was to web scrape some HTML tables that contain valuable Falcon 9 launch records. Then it was required to parse the data from those tables and convert them into a Pandas data frame for further visualization and analysis.



Data Collection – SpaceX API

The GitHub URL of the completed SpaceX API
https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX_DataCollection_usingAPI%20.ipynb

1. Obtain the rocket launch data from SpaceX API

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

2. Request and parse the SpaceX launch data using the GET request and obtain the response

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

3. Convert the response content as a JSON using .json() and turn it into a Pandas data frame using .json_normalize()

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

4. Clean the data by applying custom functions

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

5. Assign the list to a dictionary

```
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}
```

6. Create a data frame of the dictionary

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

7. Filter the data frame

```
data_falcon9 = data_dic_pd[data_dic_pd.BoosterVersion == 'Falcon 9']
```

8. Perform Data Wrangling: deal with null values and missing values

```
data_falcon9.isnull().sum()
mean=data_falcon9.PayloadMass.mean()
data_falcon9['PayloadMass']=data_falcon9['PayloadMass'].replace(np.nan,mean)
```

9. Export the data to a flat file (.CSV)

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

Data Collection – Web Scraping

The GitHub URL of the completed SpaceX Web Scrapping https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX_DataCollection_usingWebscraping.ipynb

1. Assign the URL of the wiki pages to a variable
2. Request the HTML page from the above URL and get a response object
3. Create a BeautifulSoup object from the HTML response
4. Find all tables on the wiki page
5. Get the column names from all the tables
6. Create an empty dictionary with keys from the extracted column names
7. Create a data frame
8. Export it to a CSV

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

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

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

```
html_tables=soup.find_all('table')
```

```
column_names = []  
data=first_launch_table.find_all('th')  
for i in data :  
    name=extract_column_from_header(i)  
    if name is not None and len(name)>0:  
        column_names.append(name)
```

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

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

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

Data Wrangling

The GitHub URL of the completed SpaceX Web Data Wrangling
https://github.com/Smitatiwari/Applied-Data-Science-CapstoneProject/blob/main/SpaceX_Data_Wrangling.ipynb

- The process of cleaning and unifying messy and complex data sets for easy access and analysis is known as data wrangling.

Steps for data wrangling

Initially, perform some Exploratory Data Analysis (EDA) on dataset

Calculate the number of launches at each site

Calculate the number and occurrence of each orbit

Calculate the number and occurrence of mission outcome per orbit type

Create a landing outcome label from Outcome column

Export dataset as .CSV

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

```
CCAFS SLC 40    55  
KSC LC 39A     22  
VAFB SLC 4E     13  
Name: LaunchSite, dtype: int64
```

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

```
GTO      27  
ISS      21  
VLEO     14  
PO        9  
LEO        7  
SSO        5  
MEO        3  
ES-L1      1  
HEO         1  
SO          1  
GEO         1  
Name: Orbit, dtype: int64
```

```
landing_outcomes=df['Outcome'].value_counts()  
print(landing_outcomes)
```

```
True ASDS      41  
None None      19  
True RTLS      14  
False ASDS      6  
True Ocean      5  
False Ocean     2  
None ASDS       2  
False RTLS      1  
Name: Outcome, dtype: int64
```

```
df.to_csv("dataset_part_2.csv",index=False)
```

```
landing_class = df['Outcome'].apply(lambda landing_class: 0 if landing_class in bad_outcomes else 1)
```

```
df.head(5)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004

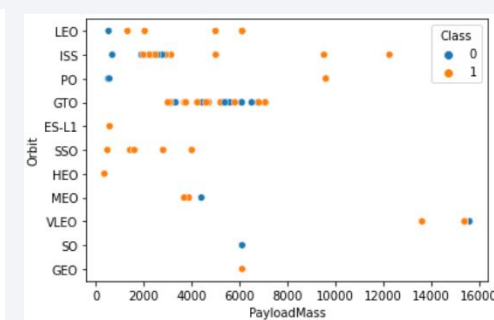
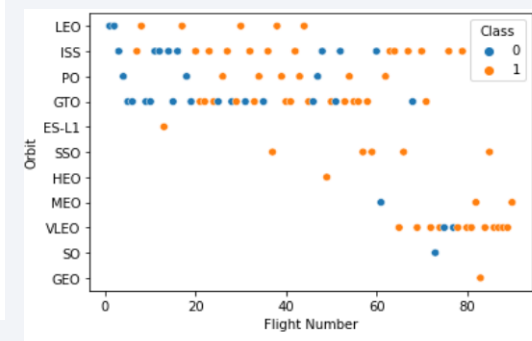
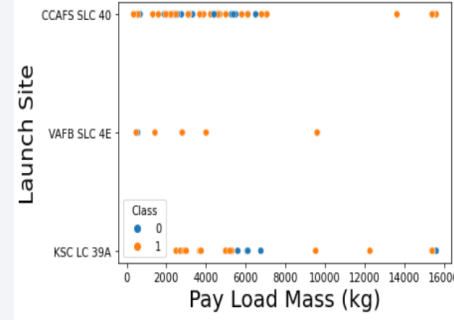
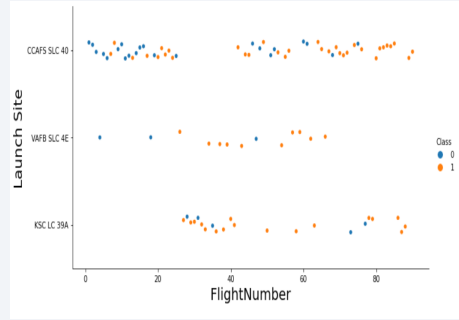
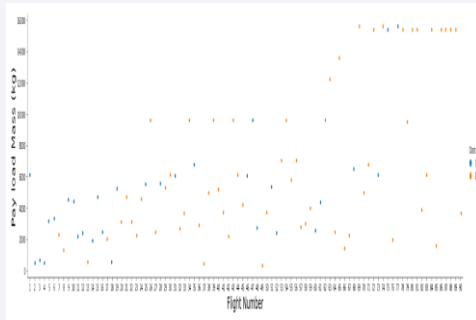
EDA with Data Visualization

Github link: https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX_EDA%20using%20Pandas%20and%20Matplotlib.ipynb

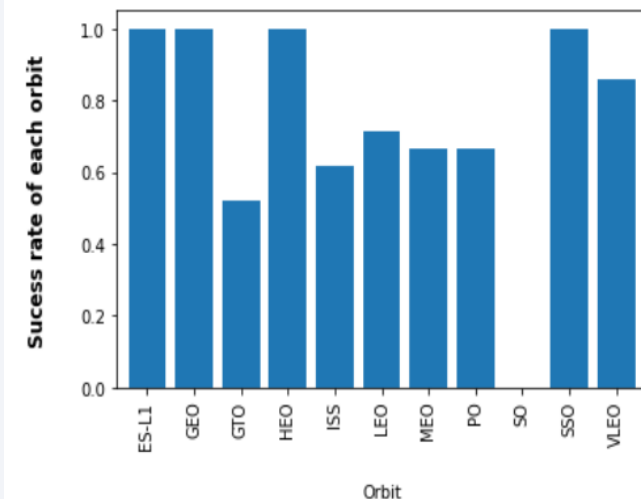
The data was explored using scatterplot and bar plots and relationship between pair of features were identified using visualization.

Scatter plots between pair of features such as follows:

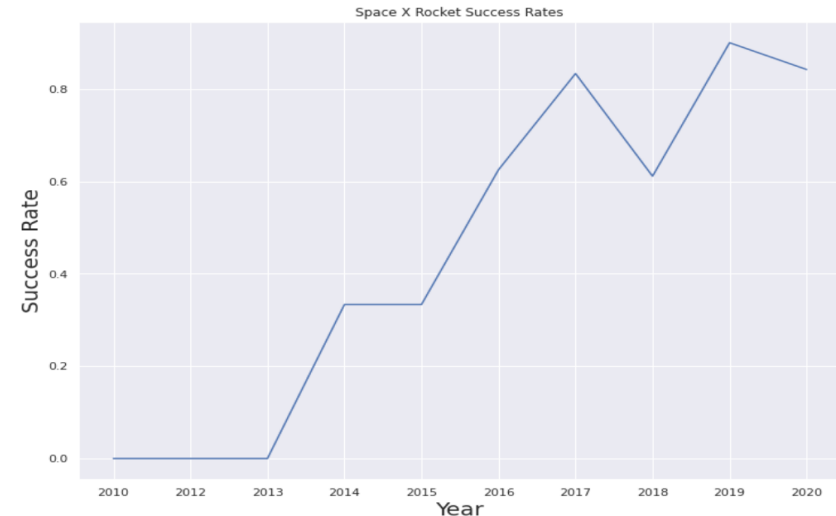
Flight Number VS. Payload Mass , Flight Number VS. Launch Site , Payload VS. Launch Site , Orbit VS. Flight Number , Payload Vs. Orbit Type



Bar graph to visualize the relationship between success rate of each orbit type



Visualization of the yearly trend of launch success



EDA with SQL

Github Link: [https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX EDA%20using%20SQL.ipynb](https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX%20EDA%20using%20SQL.ipynb)

The following SQL queries were performed:

1. Names of the unique launch sites in the space mission; `sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL ORDER BY 1;`
2. Top 5 launch sites whose name begin with the string 'CCA'; `sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;`
3. Total payload mass carried by boosters launched by NASA (CRS); `sql SELECT SUM(PAYLOAD_MASS_KG_) AS TOTAL_PAYLOAD FROM SPACEXTBL WHERE PAYLOAD LIKE '%CRS%';`

4. Average payload mass carried by booster version F9 v1.1;

```
sql SELECT AVG(PAYLOAD_MASS_KG_) AS AVG_PAYLOAD FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';
```

5. Date when the first successful landing outcome in ground pad was achieved;

```
sql SELECT MIN(DATE) AS FIRST_SUCCESS_GP FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)';
```

6. Names of the boosters which have success in drone ship and have payload mass between 4000 and 6000 kg;

```
sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS_KG_ BETWEEN 4000 AND 6000 AND LANDING_OUTCOME = 'Success (drone ship)';
```

7. Total number of successful and failure mission outcomes;

```
sql SELECT MISSION_OUTCOME, COUNT(*) AS QTY FROM SPACEXTBL GROUP BY MISSION_OUTCOME ORDER BY MISSION_OUTCOME;
```

8. Names of the booster versions which have carried the maximum payload mass;

```
sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS_KG_ = (SELECT MAX(PAYLOAD_MASS_KG_) FROM SPACEXTBL) ORDER BY BOOSTER_VERSION;
```

9. Failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015; and

```
sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Failure (drone ship)' AND DATE_PART('YEAR', DATE) = 2015;
```

10. Rank of the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and

```
2017-03-20. sql SELECT LANDING_OUTCOME, COUNT(*) AS QTY FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING_OUTCOME ORDER BY QTY DESC;
```


Build an Interactive Map with Folium

The Folium Maps used the markers, circles, lines and marker clusters to visualize the launch data into an interactive map.

Markers indicate points like launch sites, circles indicate highlighted areas around specific coordinates, marker clusters indicates groups of events in each coordinate, like launches in a launch site and lines are used to indicate distances between two coordinates.

Visualization of the Launch Data into an interactive map.

The Latitude and Longitude Coordinates at each launch site were taken and added a Circle Marker around each launch site with a label of the name of the launch site.

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

The distance was calculated Using Haversine's formula from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns.

Lines are drawn on the map to measure distance to landmarks

GitHub link: https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX_Visual%20Analytics%20using%20Folium.ipynb

Build a Dashboard with Plotly Dash

- The dashboard is built with Plotly Dash.
- The relationship with Outcome and Payload Mass (Kg) for the different Booster Versions was shown using scatter plots.
- The relationship between two variables i.e between payloads and launch sites, identified the best place to launch according to payloads.
- A non linear pattern was shown in best manner.
- The range of data flow, i.e. maximum and minimum value, can be determined.
- Github link: https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

Machine learning classification models such as logistic regression, support vector machine, decision tree and k nearest neighbors were used for predictive analysis and among these the best model was identified for SpaceX dataset.

BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data into NumPy arrays
- Standardize and transform the 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 the 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
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

Github link: [https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX ML%20Prediction.ipynb](https://github.com/Smitatiwari/Applied-Data-Science-Capstone-Project/blob/main/SpaceX%20ML%20Prediction.ipynb)

Results

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

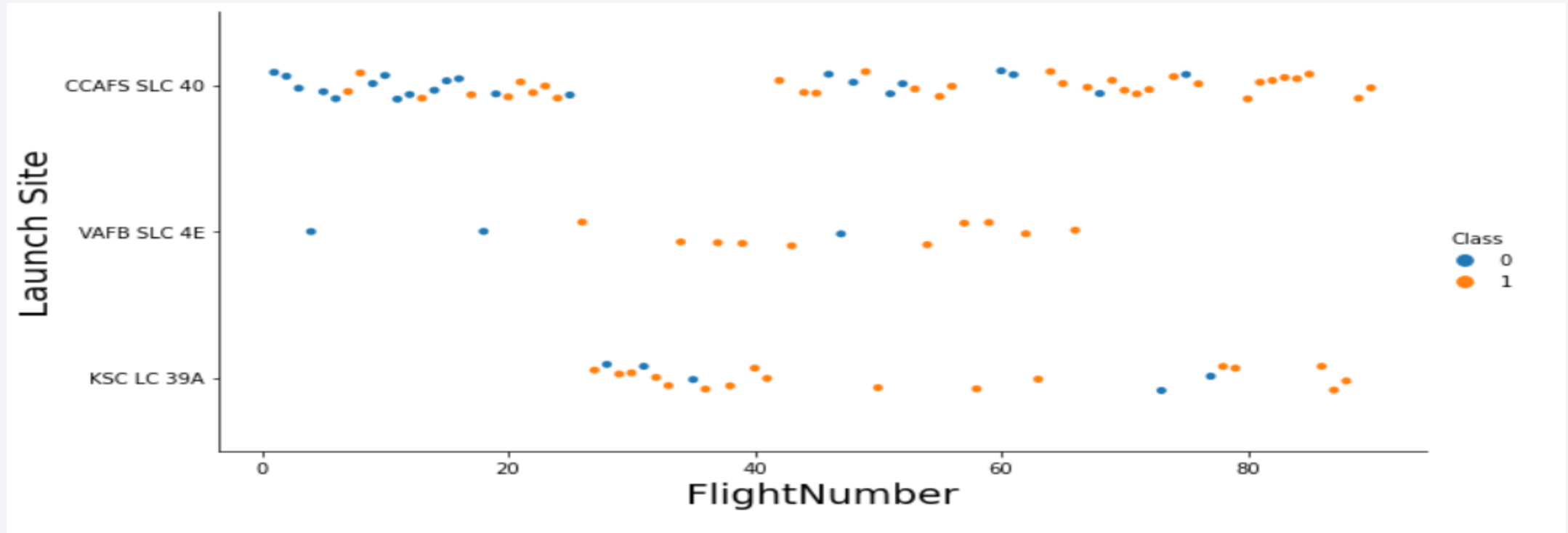
The background of the slide is an abstract composition. It features a dark blue base color. Overlaid on this are numerous diagonal streaks in shades of red and cyan. A faint, light blue grid pattern is also visible, particularly in the lower half of the image. The overall effect is dynamic and technological.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

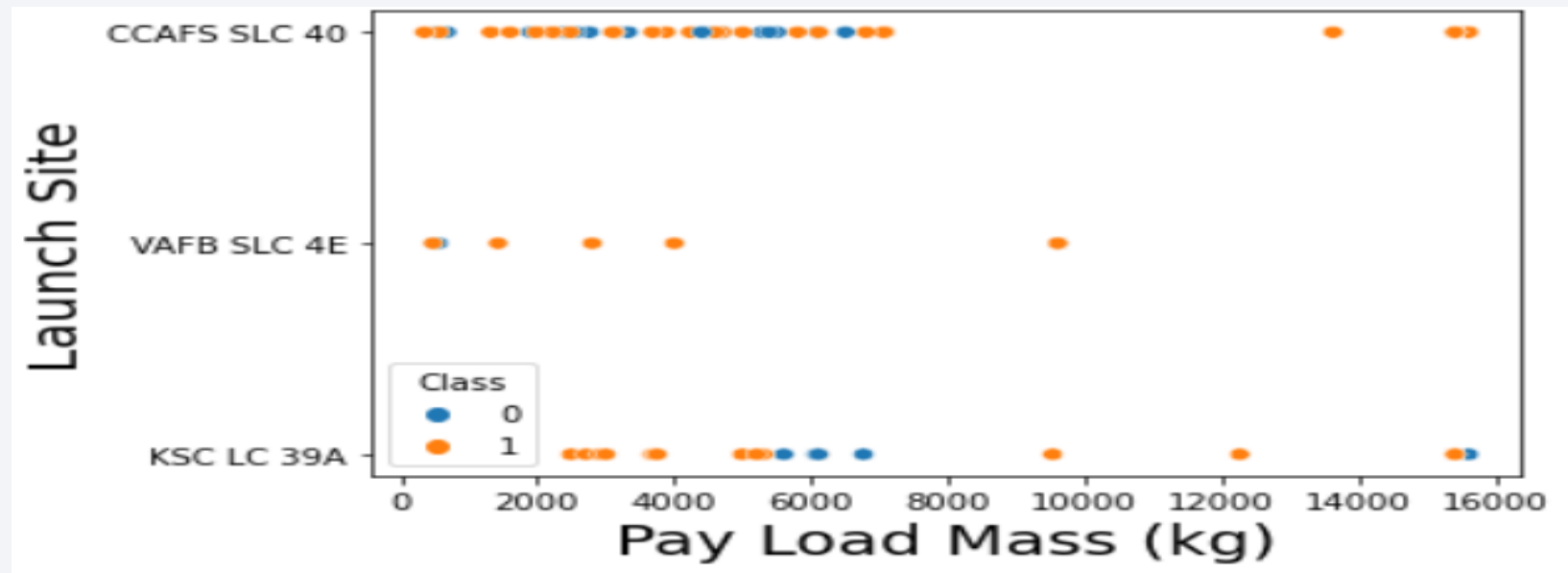
- Scatter plot of Flight Number vs. Launch Site



- The above scatter plot provides insight about the best launch site. The plot shows the best launch site is CCAFS SLC 40, where most of recent launches were successful. Then the next better launch site is VAFB SLC 4E and KSC LC 39A is next good launch site.
- The scatter plot also provides insight about the success rate which is improved over time.

Payload vs. Launch Site

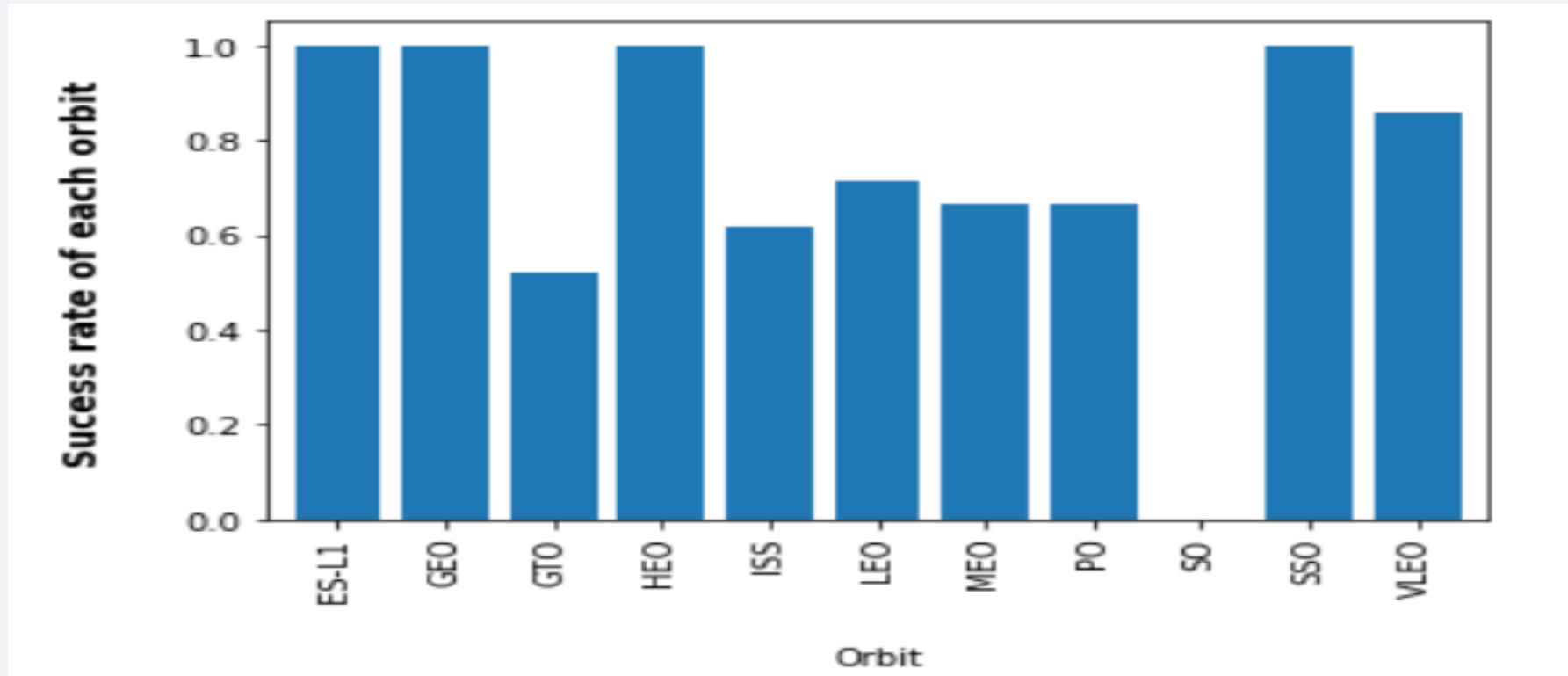
Scatter plot of Payload vs. Launch Site



Scatter plot provided insight about payloads on launch site. It is shown in the above plot that payload over 9,000kg have excellent success rate, then payloads over 12,000kg seems to be possible only on CCAFS SLC 40 and KSC LC 39A launch sites.

Success Rate vs. Orbit Type

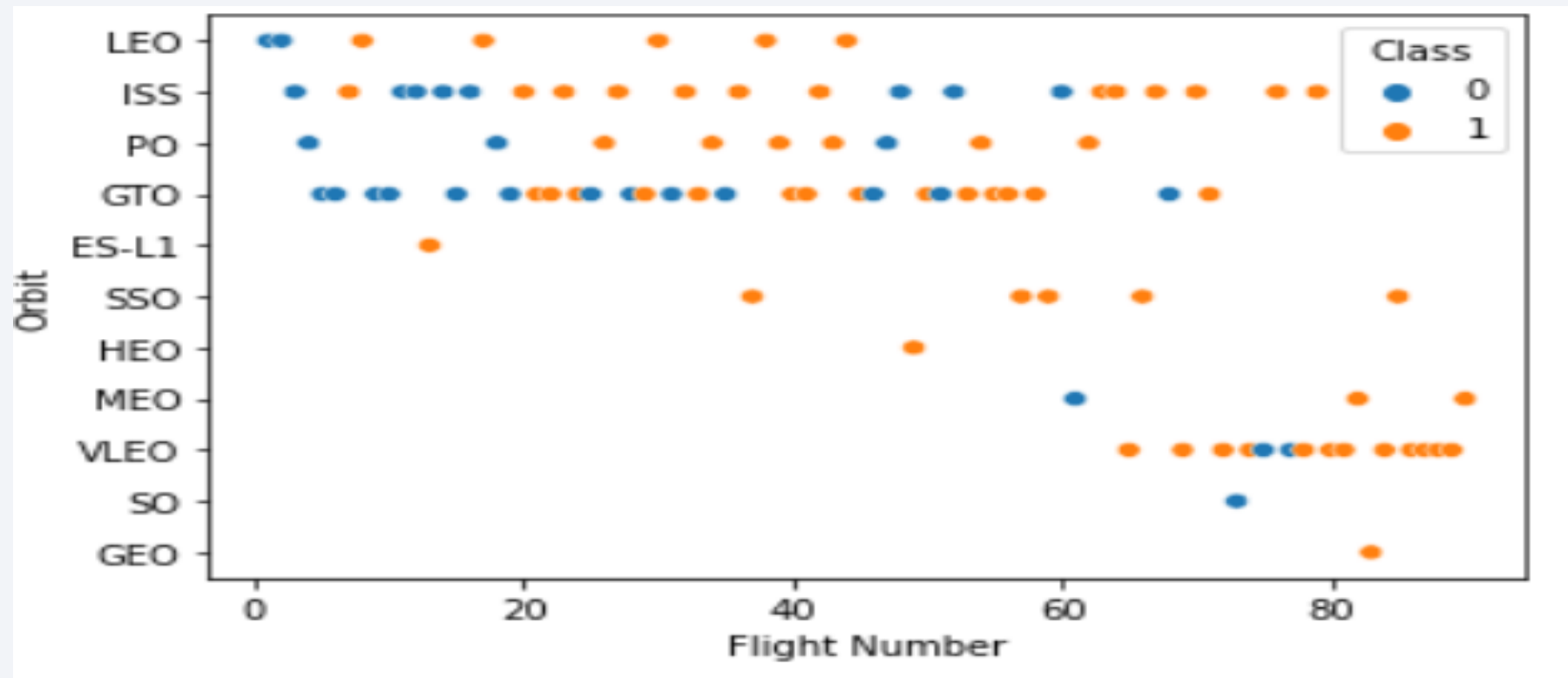
- Visualization of bar chart for the success rate of each orbit type



The biggest success rates was obtained to orbits ES-L1, GEO, HEO, and SSO.

Flight Number vs. Orbit Type

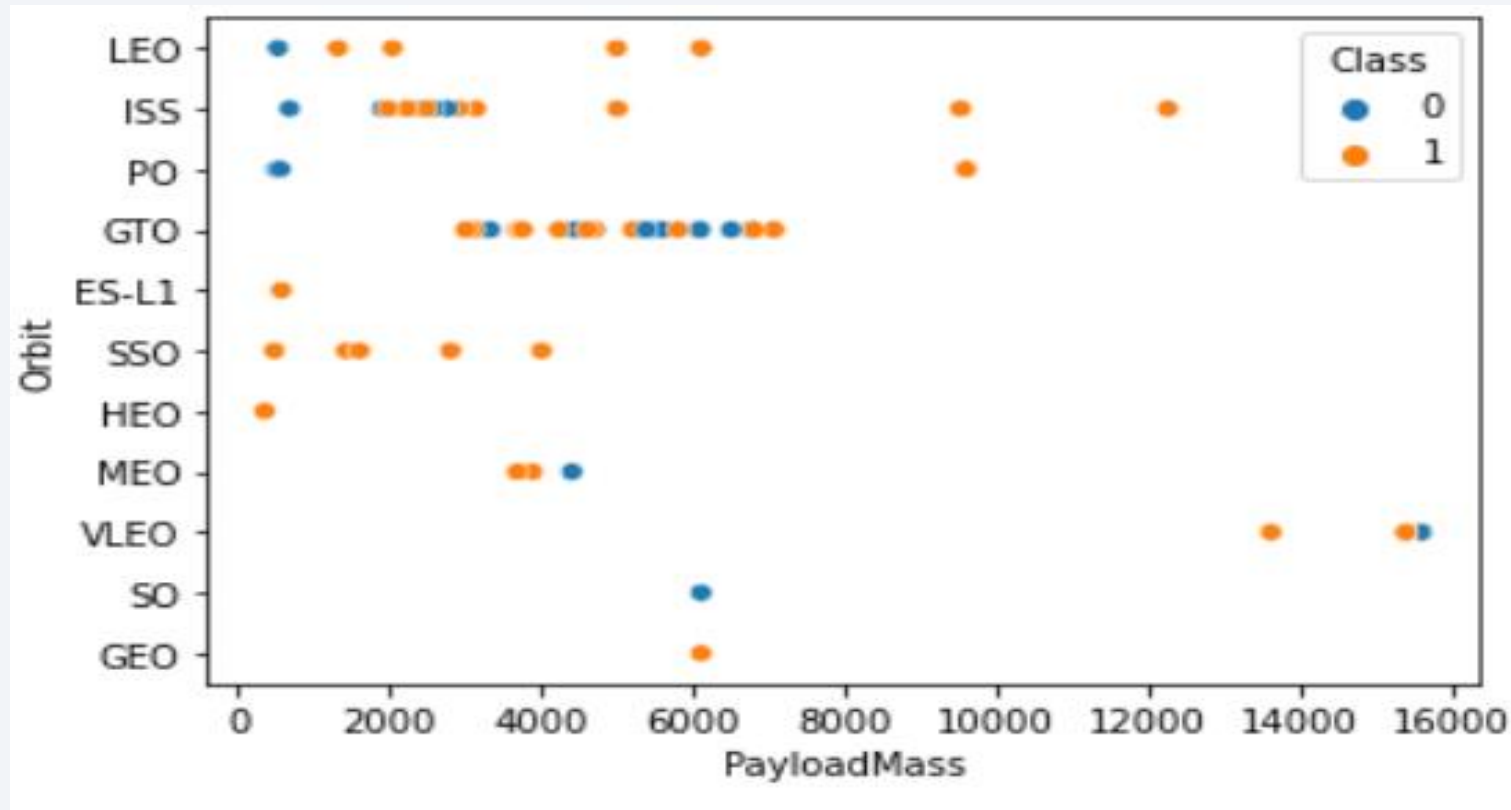
- Visualization using scatter plot of Flight number vs. Orbit type



Scatter plot of flight number vs orbit provide insight about success rate that improved over time to all orbits. The frequency of VLEO orbit seems a new business opportunity, due to recent increase of its frequency.

Payload vs. Orbit Type

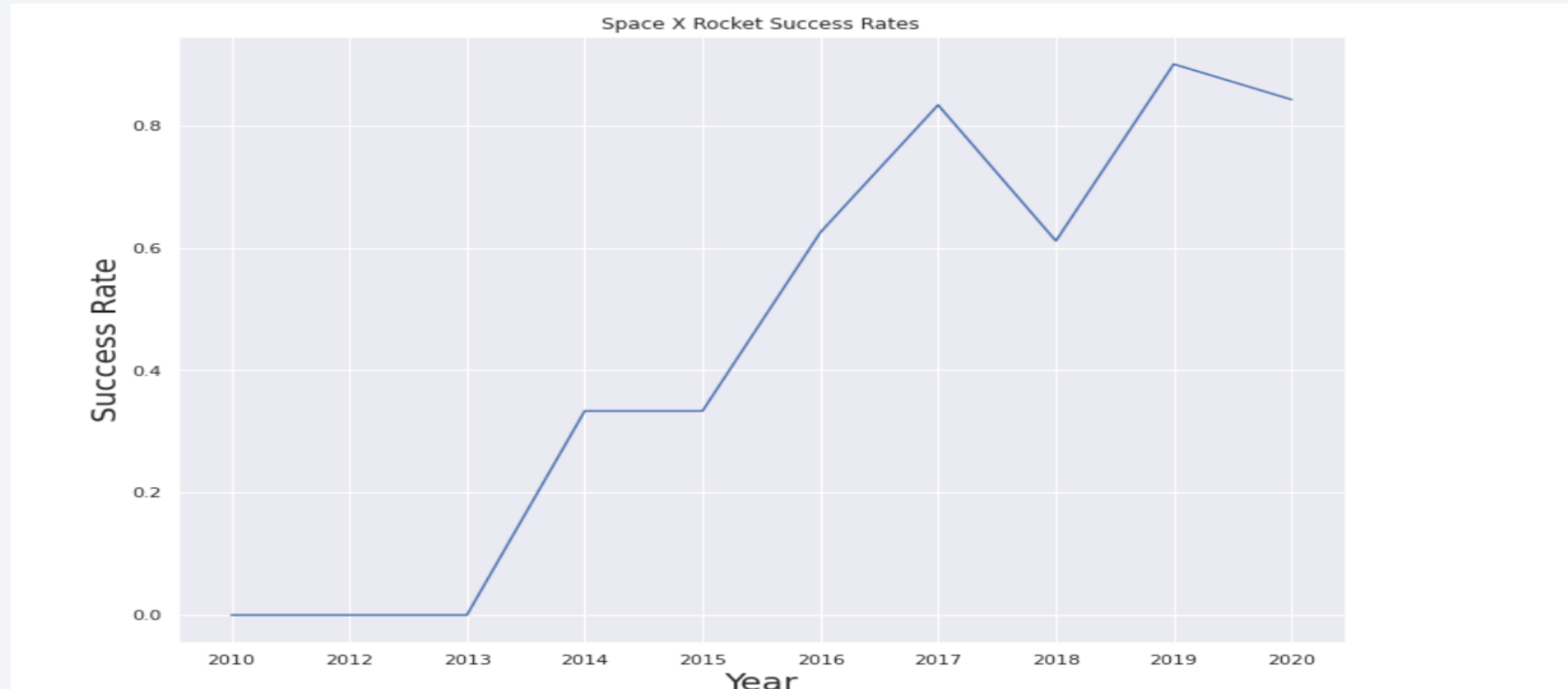
Scatter plot of payload vs. orbit type



The graph shows there is no relation between payload and success rate to orbit GTO. ISS orbit has the widest range of payload and a good rate of success. There are few launches to the orbits [23](#) SO and GEO.

Launch Success Yearly Trend

A line chart of yearly average success rate



Line chart shows that the success rate started increasing in 2013 and keep on increasing until 2020. The yearly trend showed that the first three years were a period of adjustment and improvement of technology. 24

All Launch Site Names

- The names of the unique launch sites

SQL Query:

```
SELECT DISTINCT LAUNCH_SITE  
FROM SPACEXTBL ORDER BY 1;
```

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

The word **DISTINCT** in the query will only show unique values in the **LAUNCH_SITE** column from **SPACEXTBL**.

Launch Site Names Begin with 'CCA'

- Five records where launch sites begin with 'CCA'

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

```
SELECT *
```

```
FROM SPACEXTBL
```

```
WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

The keyword 'LIMIT 5' is used to fetch five records from the `SPACEXTBL` table. The condition is applied using wildcard 'CCA%'. The "%" sign in 'CCA%' ensures that Launch_Site name starts with CCA.

Total Payload Mass

The total payload carried by boosters from NASA

SQL Query

```
SELECT SUM(PAYLOAD_MASS_KG_) AS TOTAL_PAYLOAD  
FROM SPACEXTBL  
WHERE PAYLOAD LIKE '%CRS%';
```

total_payload

111268

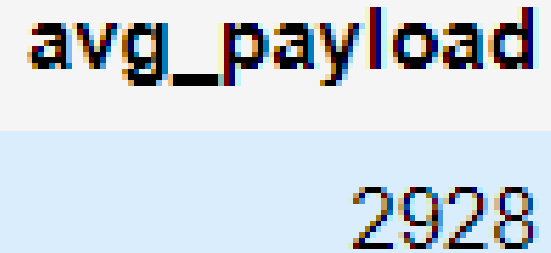
The function ***SUM*** summates the total in the column ***PAYLOAD_MASS_KG_***. The ***WHERE*** clause filters the dataset to only perform calculations on ***payload using wildcard LIKE '%CRS%'***.

Average Payload Mass by F9 v1.1

The average payload mass carried by booster version F9 v1.1

SQL Query

```
SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD  
  
FROM SPACEXTBL  
  
WHERE BOOSTER_VERSION = 'F9 v1.1';
```



avg_payload
2928

The function **AVG** is used to find 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

Find the dates of the first successful landing outcome on ground pad

SQL Query

```
SELECT MIN(Date) AS FIRST_SUCCESS_GP  
FROM SPACEXTBL  
WHERE LANDING_OUTCOME = 'Success (ground pad)';
```

first_success_gp
2015-12-22

The function **MIN** is used to find 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

List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

SQL Query

```
SELECT DISTINCT BOOSTER_VERSION
```

```
FROM SPACEXTBL
```

```
WHERE PAYLOAD_MASS_KG_ BETWEEN 4000 AND 6000 AND LANDING_OUTCOME ='Success (drone ship)';
```

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

Booster_Version is selected. 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**.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

SQL Query

```
SELECT MISSION_OUTCOME, COUNT(*) AS QTY  
FROM SPACEXTBL  
GROUP BY MISSION_OUTCOME  
ORDER BY MISSION_OUTCOME;
```

mission_outcome	qty
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

COUNT function is used to find total number mission outcomes. **GROUP BY** function is used for grouping mission outcomes , **ORDER BY** is used for ordering the mission outcomes and counting records for each group provided the summary above.

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

SQL Query

```
SELECT DISTINCT BOOSTER_VERSION  
FROM SPACEXTBL  
WHERE PAYLOAD_MASS_KG_ =  
      (SELECT MAX(PAYLOAD_MASS_KG_)  
        FROM SPACEXTBL)  
ORDER BY BOOSTER_VERSION;
```

The MAX function is used to fetch the maximum payload in the column PAYLOAD_MASS_KG_ in the sub query. The Booster Version which had maximum payload is filtered by WHERE clause.

booster_version
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

SQL Query

```
SELECT BOOSTER_VERSION, LAUNCH_SITE  
FROM SPACEXTBL  
WHERE LANDING_OUTCOME = 'Failure (drone ship)'  
AND DATE_PART('YEAR', DATE) = 2015;
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

BOOSTER_VERSION, LAUNCH_SITE are selected in SELECT clause by putting filters in WHERE clause LANDING_OUTCOME equals to 'Failure (drone ship)' AND DATE_PART('YEAR', DATE) as 2015.

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

- 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

SQL Query

```
SELECT LANDING_OUTCOME, COUNT(*) AS QTY
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING_OUTCOME
ORDER BY QTY DESC;
```

landing__outcome	qty
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

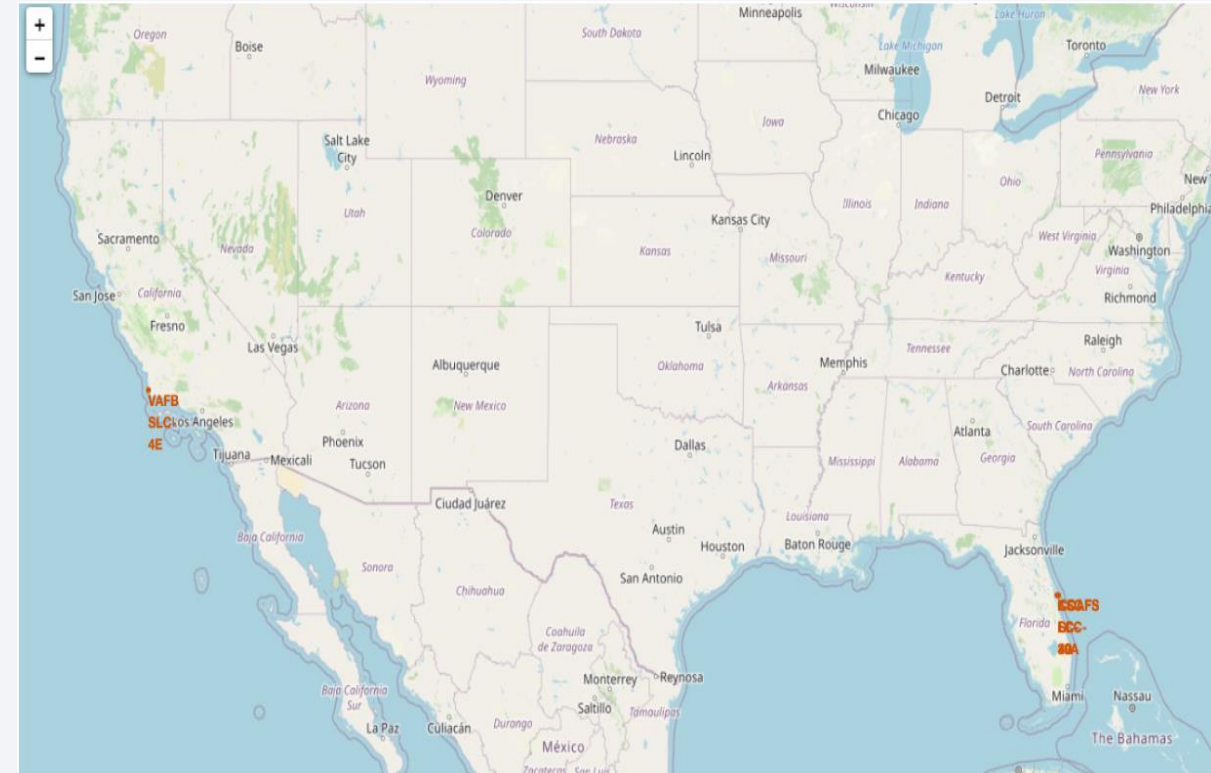
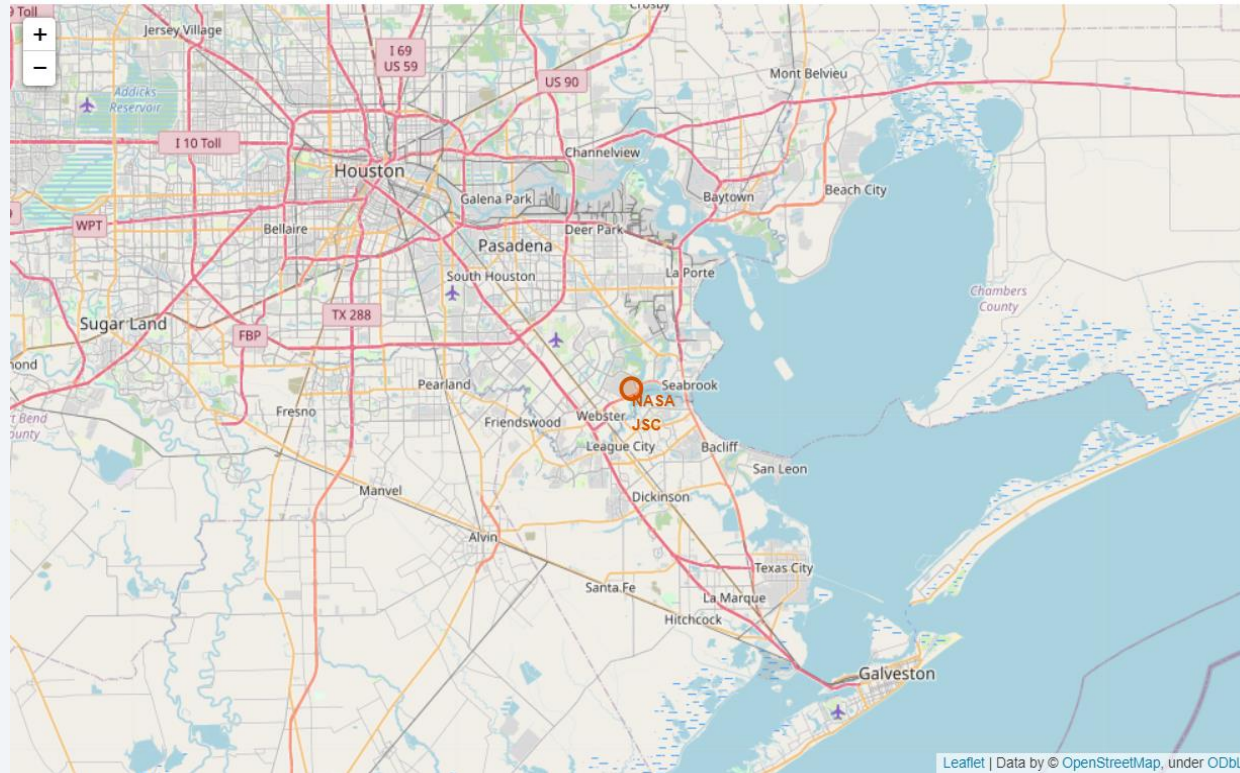
LANDING_OUTCOME is selected as "Landing Outcome" in **SELECT** clause by putting filters in **WHERE** clause as **"DATE BETWEEN '2010-06-04 AND '2017-03-20'"**. **GROUP BY** clause used for grouping data according LANDING_OUTCOME and ORDER BY for ordering in descending order.

A satellite view of Earth from space, showing the curvature of the planet and city lights at night. The background is a deep blue gradient.

Section 3

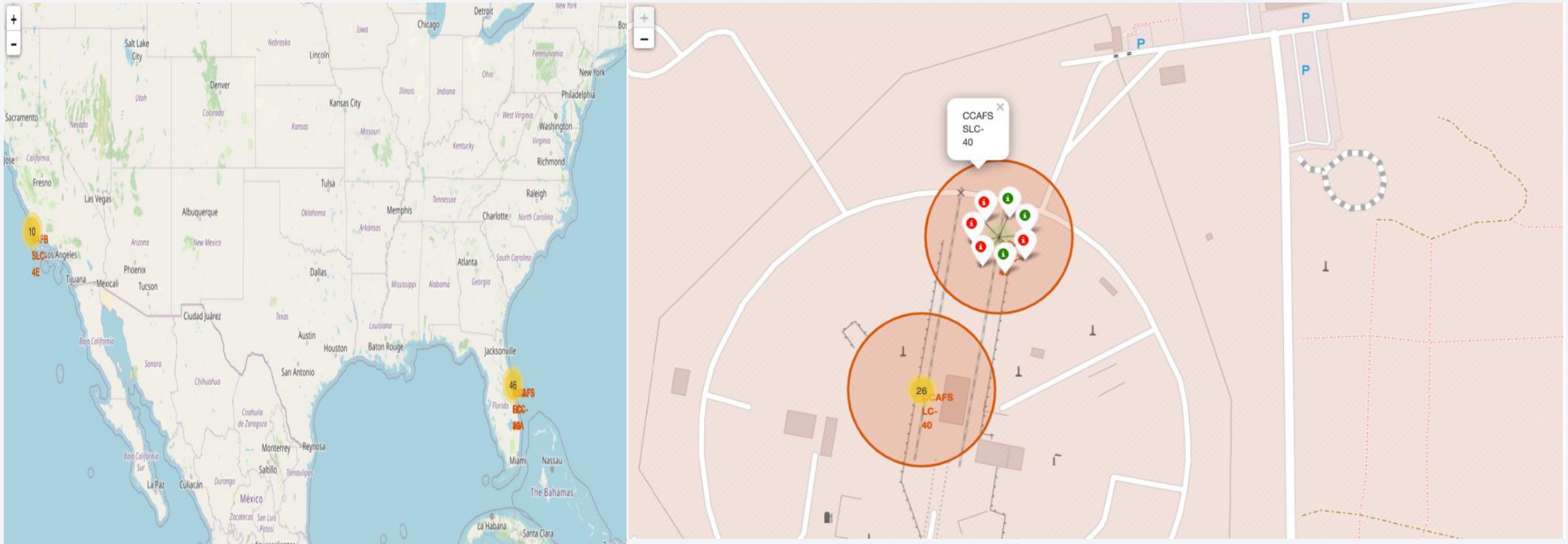
Launch Sites Proximities Analysis

Mark all launch sites on a map



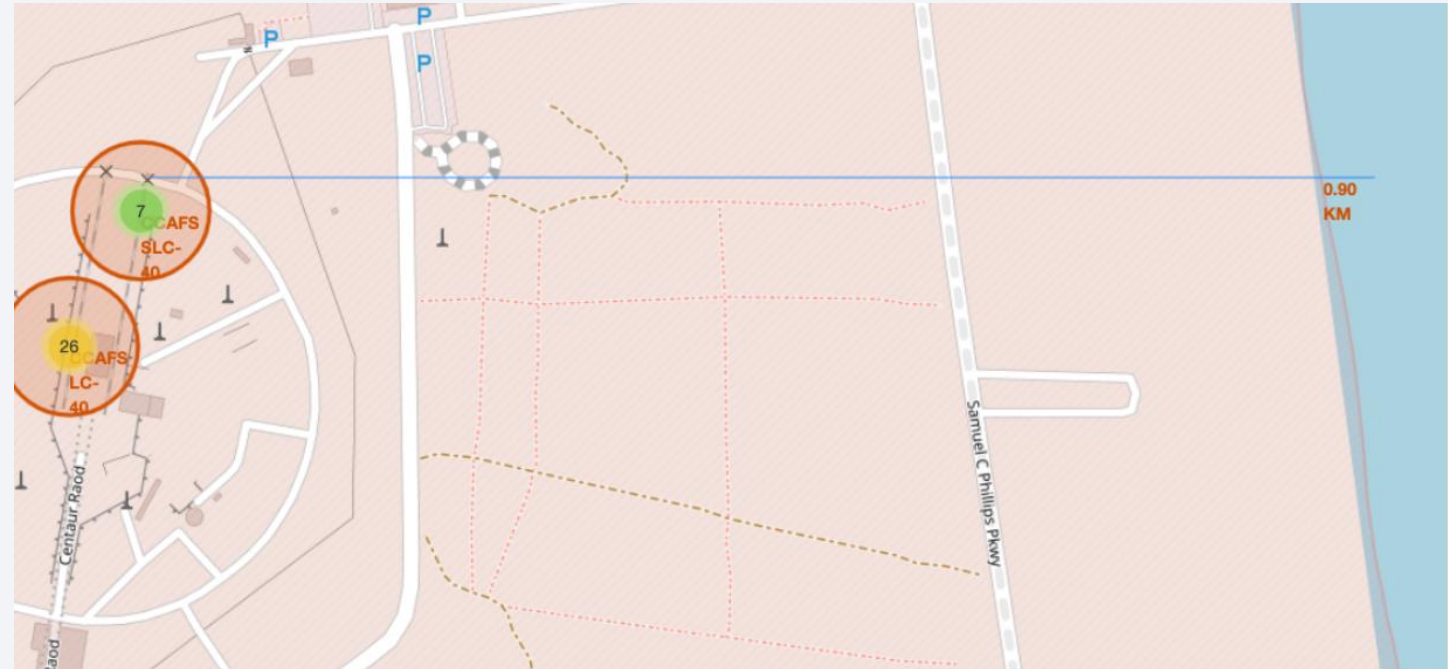
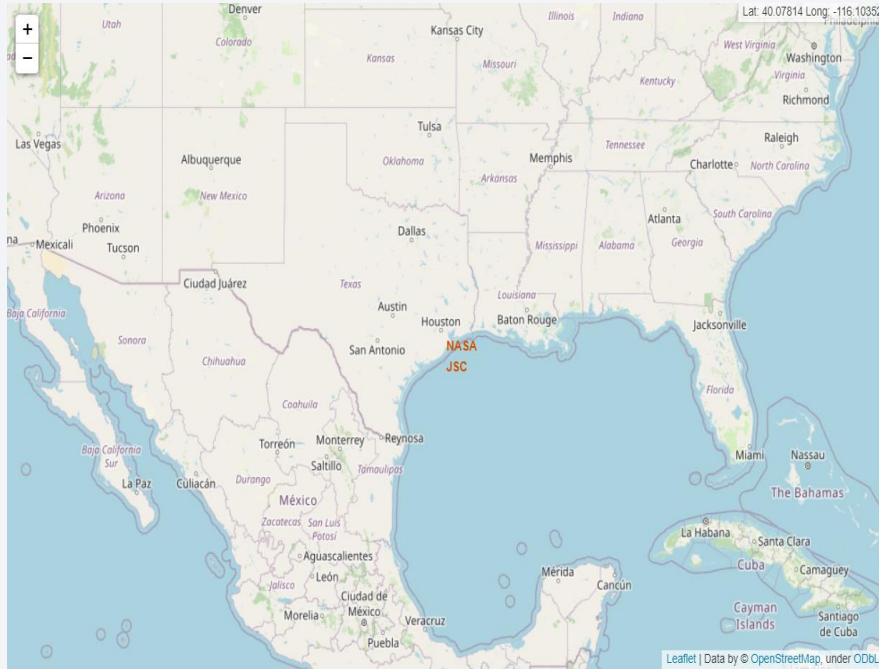
All the launch sites are in very close proximity to the coastal are in the above Folium Map Screenshot

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



The successful Launches are shown by Green Marker and Failures are shown in Red Marker in the above Folium map

Calculate the distances between a launch site to its proximities



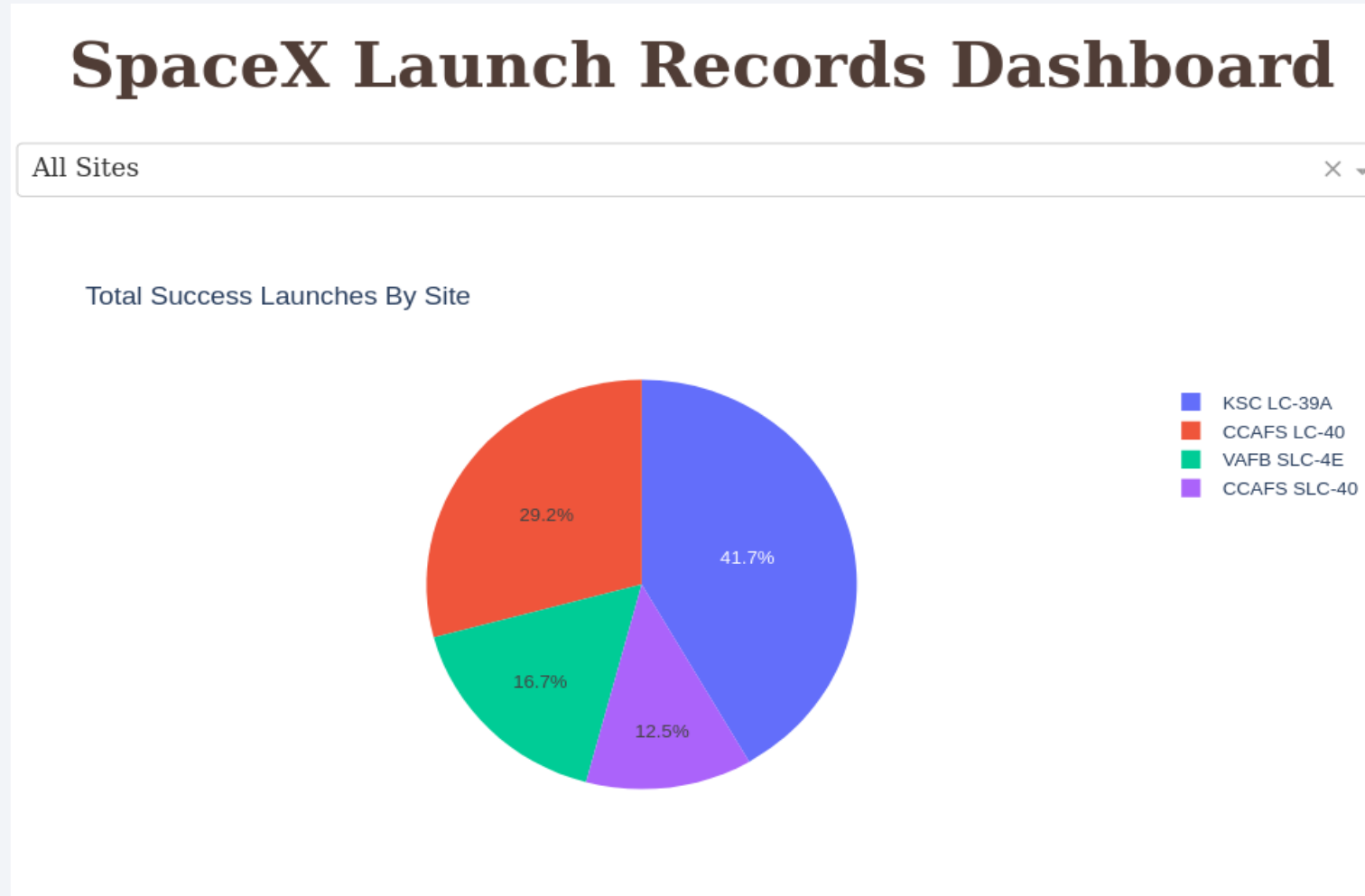
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

Build a Dashboard with Plotly Dash

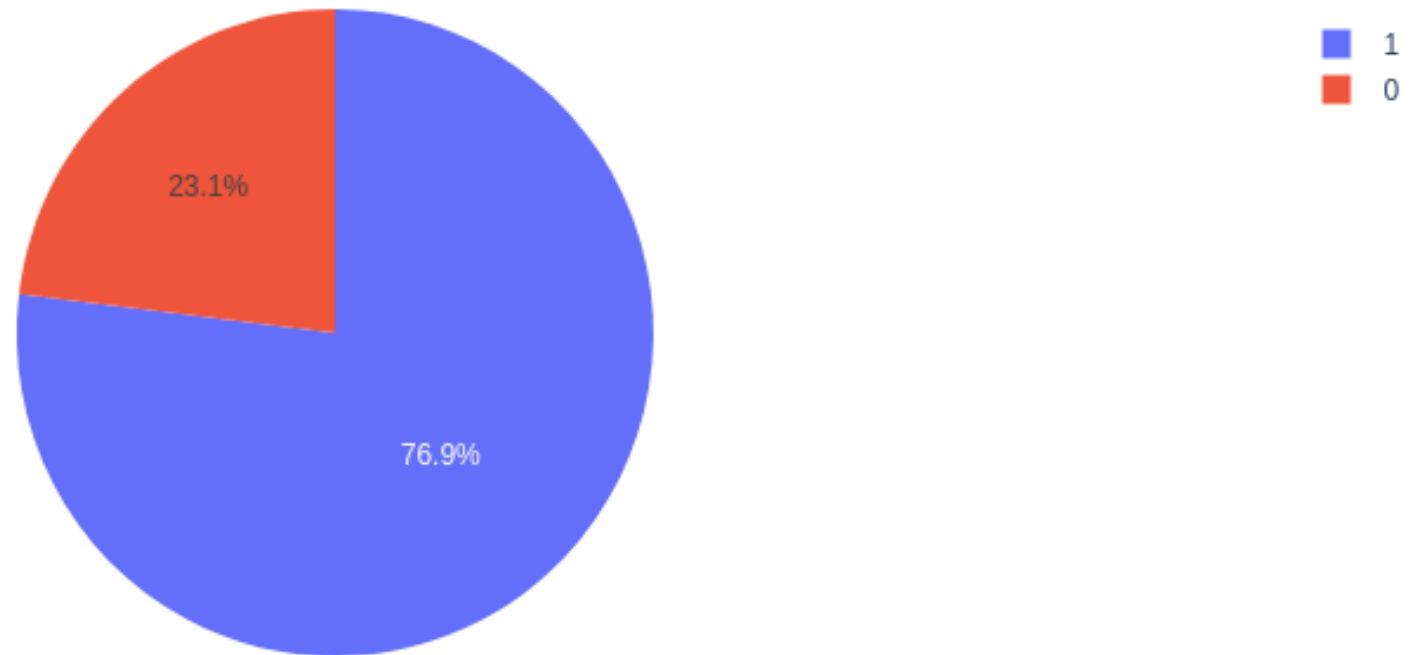
Screenshot of Dashboard for successful launches by each site



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

Launch Success Ratio for KSC LC-39A

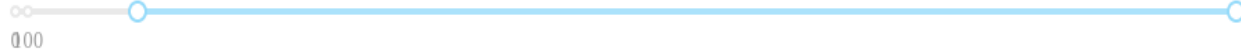
Total Launches for site KSC LC-39A



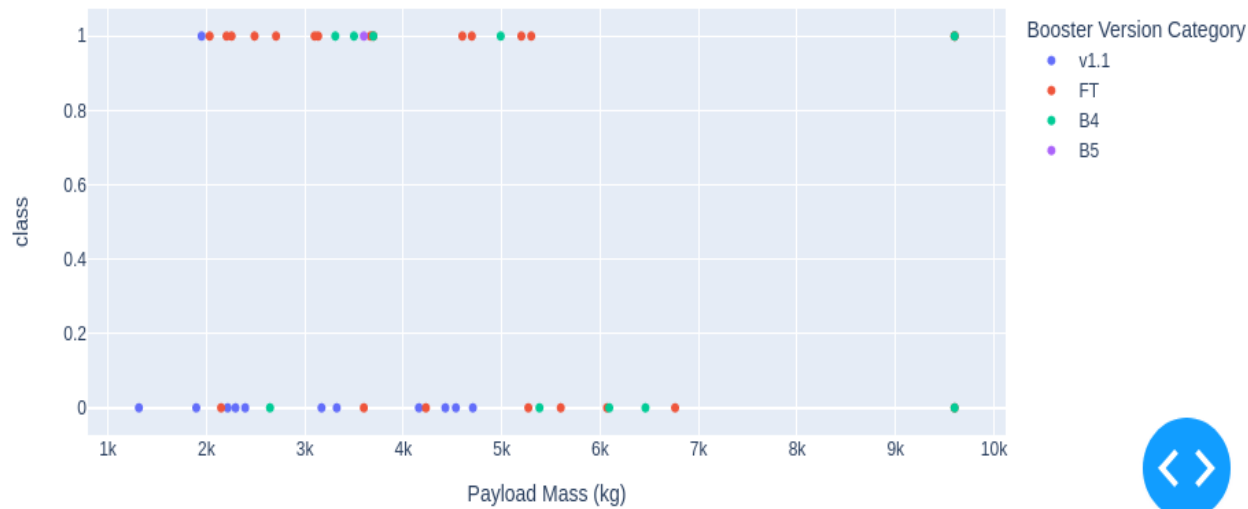
76.9% of launches are successful in KSC LC-39A site

Scatter plot for Payload vs. Launch Outcome for all site

Payload range (Kg):



All sites - payload mass between 1,000kg and 10,000kg

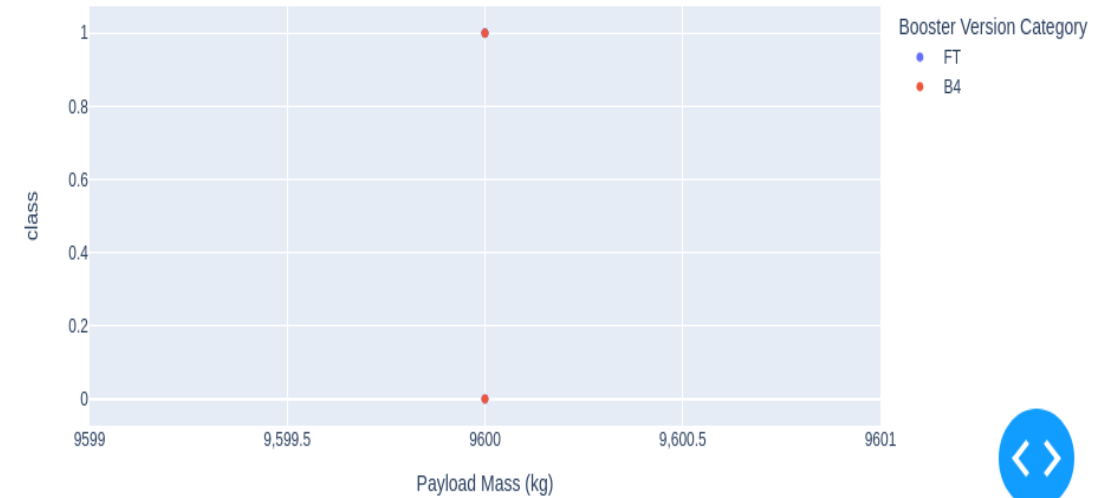


Combination of Payloads under 6,000kg and FT boosters

Payload range (Kg):



All sites - payload mass between 7,000kg and 10,000kg



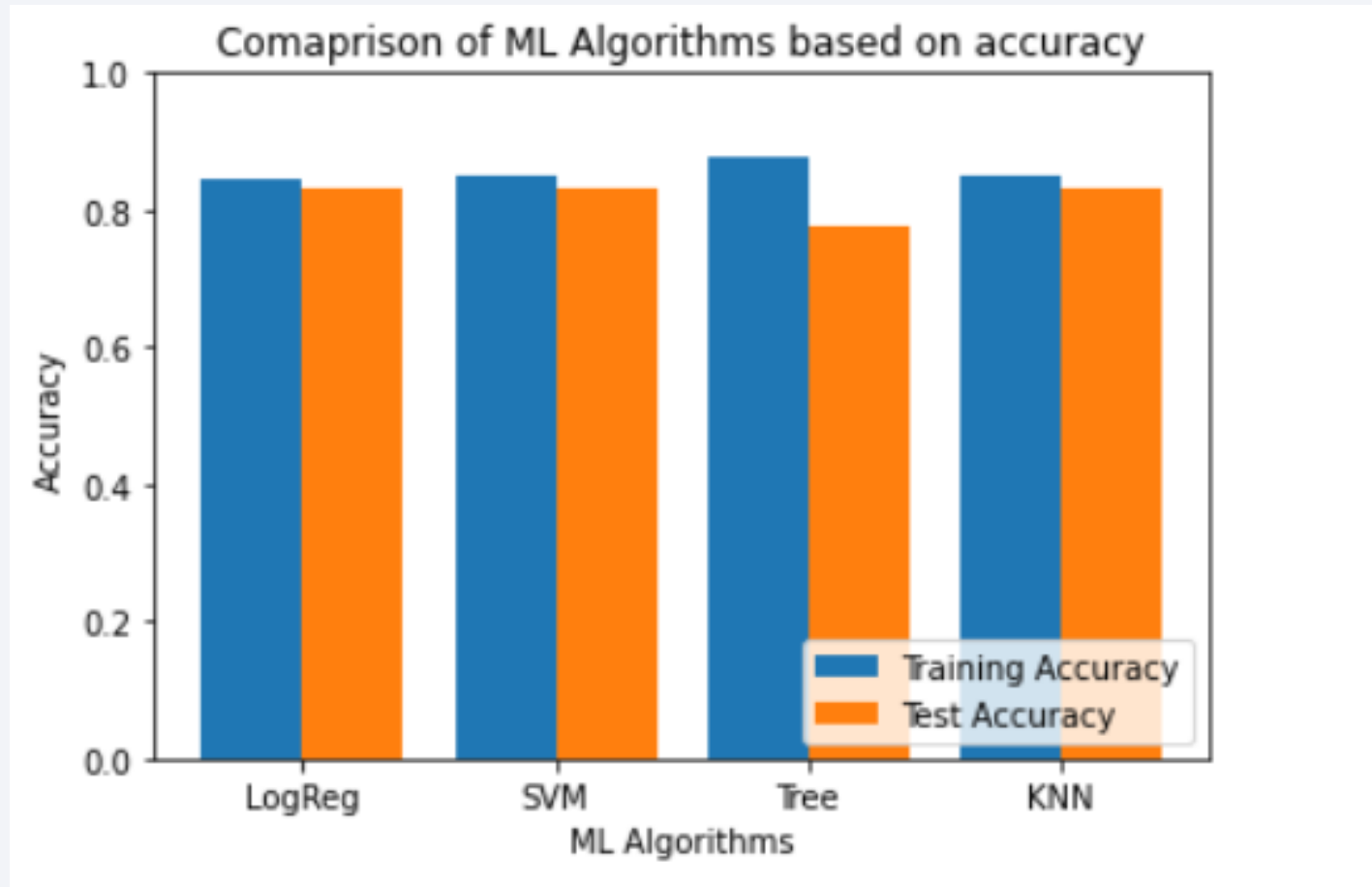
Launches on Payloads over 7,000kg



Section 5

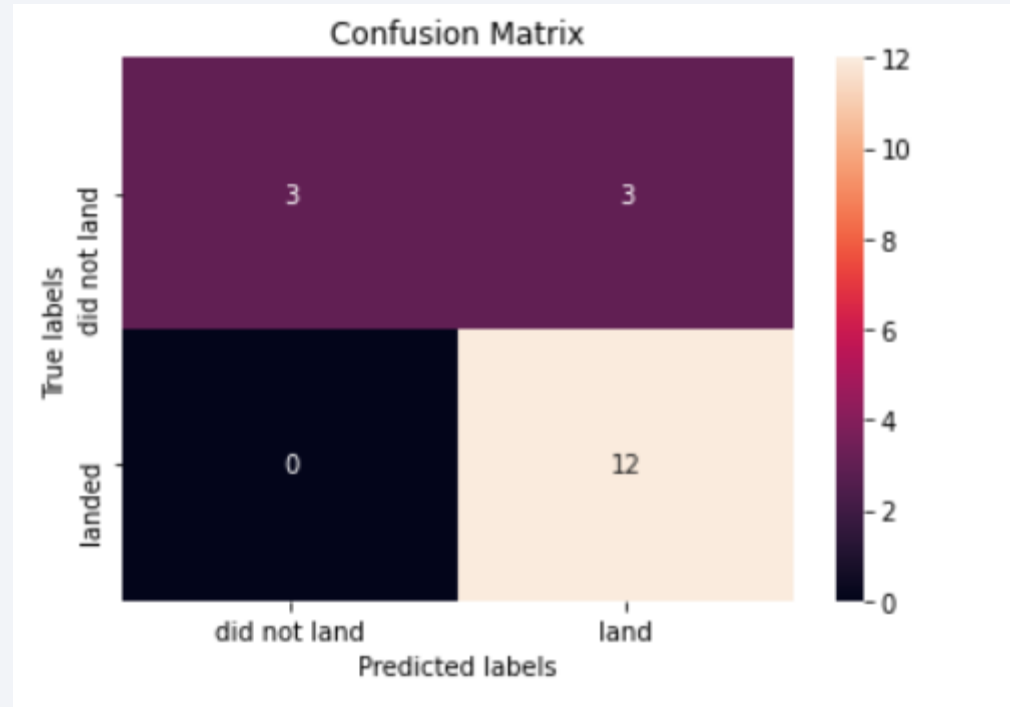
Predictive Analysis (Classification)

Classification Accuracy



The implemented classification models were four in number such as Logistic Regression, SVM, Decision Tree, and KNN. Among these the Decision Tree model achieved highest ⁴⁴ classification accuracy which can also be concluded from the above bar chart

Confusion Matrix



Decision Tree Classifier Confusion matrix obtained large values of true positive and true negative as compared to the false ones.

Conclusions

- The best launch site is KSC LC-39A;
- Launches above 7,000kg are less risky;
- Although most of mission outcomes are successful, successful landing outcomes seem to improve over time, according the evolution of processes and rockets;
- Decision Tree Classifier is best to predict successful landings of such kind of dataset.
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Appendix

- Module sqlserver
- PythonAnywhere 24/7 dashboard

Thank you!

