IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Anagha Harikrishnan
14-03-2023

Outline

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

Executive Summary

Summary of methodologies

- Data Collection
- Data Wrangling
- Exploratory Data Analysis
  - Using SQL
  - Visualization (using pandas and matplotlib)
- Data Visualization using Folium
- Interactive Dashboard with Plotly

Summary of all results

- Exploratory data analysis results
- Interactive analysis demo
- Predictive analysis result

Introduction

Project background and context

- Space X promotes Falcon 9 rocket launches on its website for 62 million dollars; other suppliers charge upwards of 165 million dollars for each launch.
- A large portion of the savings is due to Space X's ability to reuse the first stage.
- Hence, if we can figure out if the first stage will land, we can figure out how much a launch will cost.

Problems you want to find answers

- Obtaining best features from the dataset for prediction process.
- Predicting if the first stage will land successfully based on the best score features.

Section 1

Methodology

Methodology

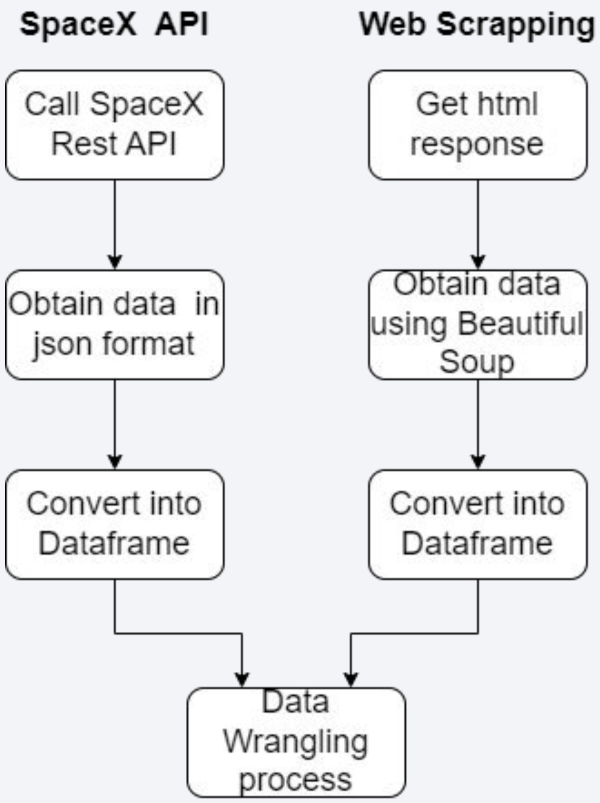
Executive Summary

- ❖ Data collection methodology:
  - ➢ Data was collected via Spacex Rest API and Web Scrapping from wikipedia.
- ❖ Perform data wrangling
  - ➢ EDA to find some patterns in data and labels for training supervised models
- ❖ Perform exploratory data analysis (EDA) using visualization and SQL
- ❖ Perform interactive visual analytics using Folium and Plotly Dash
- ❖ Perform predictive analysis using classification models
  - ➢ Built LR, KNN, SVM DT models for classification and evaluated them to find the best model.

Data Collection

Data were collected in two ways:

- ❖ SpaceX Rest API
  - ➢ Once the SpaceX Rest API was called, it gave us the information about launch sites, launch pads, payload, rocket specifications, landing outcome etc. in a json format.
  - ➢ Further this json format was converted to a dataframe for better analysis.
- ❖ Web Scrapping
  - ➢ Here, data was scrapped from wikipedia using Beautiful Soup library.



Data Collection Flowchart

Data Collection – SpaceX API

- ❖ Data Collection using SpaceX Rest API.

- ❖ GitHub URL of the completed SpaceX API calls notebook https://github.com/ahk99/IBM-CapestoneProject/blob/e6161b260045f57d745140e66e9dbff86c173a7b/data%20collction.ipynb

1) Response from SpaceX Rest API.

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

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

2) Converting json results to a dataframe.

```
# Use json_normalize meethod to convert the json result into a dataframe
data=pd.json_normalize(response.json())
```

3) Functions to process data.

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

4) Constructing a dataset using the data.

```
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}
```

5) Creating a Pandas dataframe from the dictionary.

```
# Create a data from launch_dict
df=pd.DataFrame([launch_dict])
```

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- GitHub URL of the web scraping notebook https://github.com/ahk99/IBM-CapestoneProject/blob/810fc8697136f29951c73117f3fe27ad51c39824/data%20collection%20web%20scrapping.ipynb

1) Response from Html and object for beautiful soup

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

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

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

2) Finding tables

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables=soup.find_all('table')
```

3) Iterating through each table and getting data

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
item=soup.find_all('th')
for i in range (len(item)):
    try:
        name=extract_column_from_header(item[i])
        if (name is not None and len(name)>0):
            column_names.append(name)
    except:
        pass
```

4) Creating an empty dict and add the data.

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
```

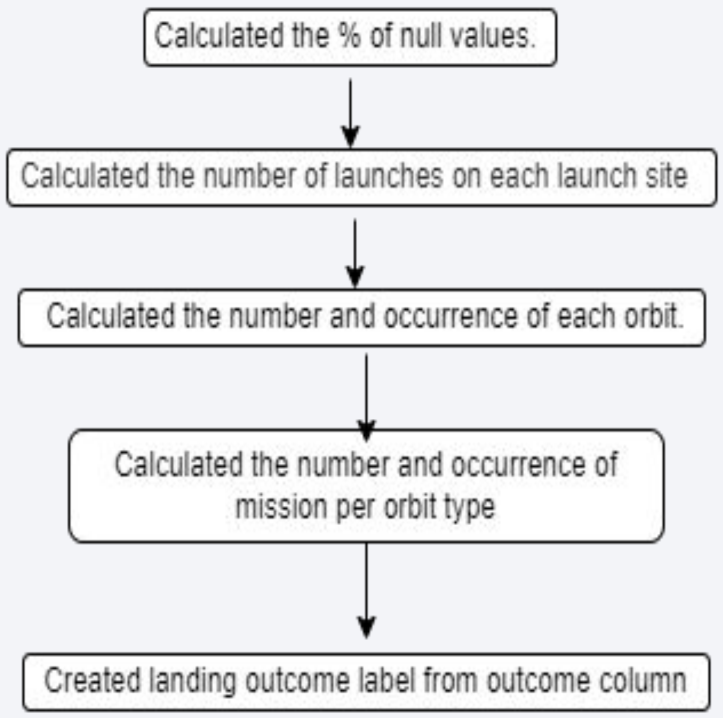
5) Converting to dataframe.

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

Data Wrangling

- ❖ Performed EDA to find some patterns in the data and to determine the label for training supervised model.
- ❖ Calculated the number of launches on each launch site.
- ❖ Calculated the number and occurrence of each orbit.
- ❖ Calculated the number and occurrence of mission outcome per orbit type.
- ❖ Created a landing outcome label from outcome column.
- ❖ GitHub URL of data wrangling:

  https://github.com/ahk99/IBM-CapestoneProject/blob/39f17f12e994d7685d03c4005e06e84559f8787f/EDA.ipynb



Data Wrangling Flowchart

EDA with Data Visualization

- ❖ Three different charts were used to graphically represent the relationship between various features in the dataset.
- ❖ These charts were catplot, bar chart and line plot from the seaborn library imported as sns.
- ❖ Catplot was used to show the relation between payload mass vs flight number, flight number vs launch site, payload mass vs launch site, flight number vs orbit type and payload mass vs orbit type.
- ❖ Barchart was used to show the relation between orbit and class.
- ❖ Line plot was used to show the relation between year and class.

- ❖ GitHub URL of EDA with data visualization:

  https://github.com/ahk99/IBM-CapestoneProject/blob/5b356ea1d615137356ca512e4fba02bf121b6379/eda%20viualization.ipynb

EDA with SQL

- SQL queries performed are:
  - Displaying the names of the unique launch sites in the space mission.
  - Displaying five records where launch sites begin with the string ‘CCA’.
  - Displaying the total payload mass carried by boosters launched by NASA (CRS).
  - Displaying average payload mass carried by booster version F9 v1.1.
  - Listing the date when the first successful landing outcome in ground pad was achieved.
  - Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - Listing the total number of successful and failure mission outcome.
  - Listing the names of the booster versions which have carried the maximum payload mass.
  - Listing the records which will display the month names, failure landing outcomes in drone ship, booster versions, launch sites for the months in the year 2015.
  - Ranking the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

- GitHub URL of EDA with SQL:

  https://github.com/ahk99/IBM-CapestoneProject/blob/12c65066f7335169e8e277aa69d491a6bcb1e422/jupyter-labs-eda-sql-coursera_sqllite%20(1).ipynb

Build an Interactive Map with Folium

- Markers as circles are created in folium map to define each launch sites
- Marker clusters is use to mark the success/ failed launches for each site on the map where the marker shows green if it is a successful launch and red if it is a failed one.
- Distance between two points on the map is shown by latitude and longitude and also using falium.marker function.

- GitHub URL of interactive map with Folium map:

  https://github.com/ahk99/IBM-CapestoneProject/blob/1cca68e32cd4551708ca03895b5e95578dd20769/Interactive%20visualization.ipynb

Build a Dashboard with Plotly Dash

- First graph is a pie chart that shows the total success launches and can be filtered using a drop down for each launch sites or all of them.
- Second graph is a catplot that determines the success or failure of a rocket booster version based on its payload mass.
- GitHub URL of Plotly Dash lab:

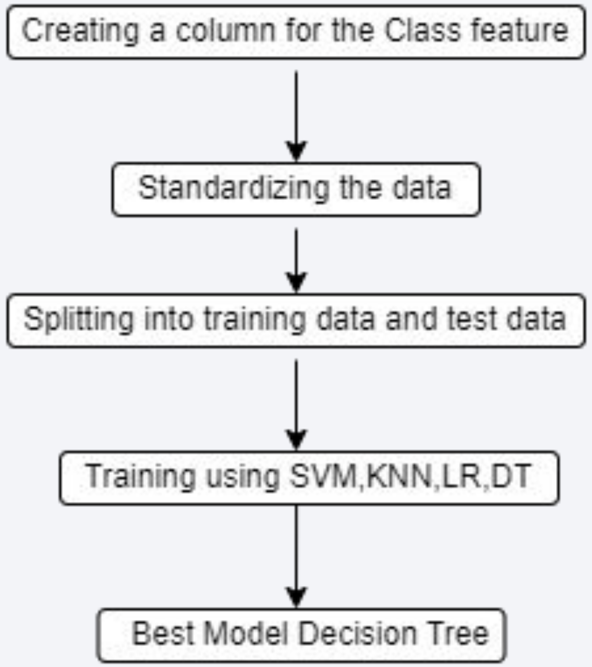
  https://github.com/ahk99/IBM-CapestoneProject/blob/3b28d17829afd7e6ab2904f53a2853fdf8539334/dashboard.py

Predictive Analysis (Classification)

- Here , four classification algorithms are used namely Logistic Regression, K Nearest Neighbours, Decision Tree and Support Vector Machine.
- The data are standardized and split into training and testing data.
- Best features are obtained from the dataset and they are used as the feature set for the models to train on.
- Best score from each model defines the accuracy of these features for the training models.

- GitHub URL of predictive analysis:

  https://github.com/ahk99/IBM-CapestoneProject/blob/3e55c45ab39d2734541b0245e28783088c63a728/Machine%20learning.ipynb



Predictive Analysis Flowchart

Results

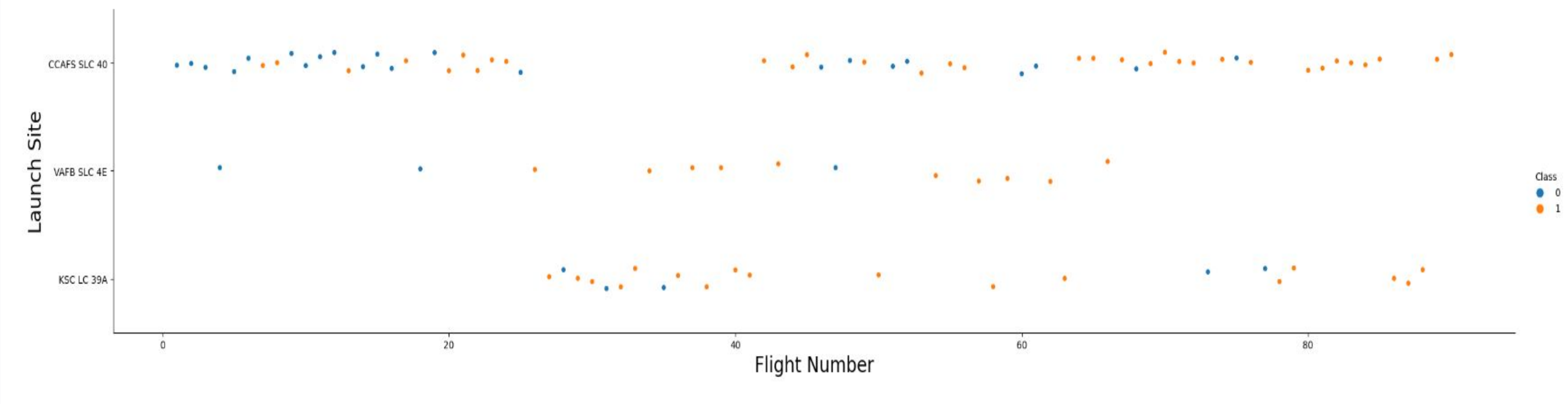
- The decision tree model shows the highest accuracy for this dataset.
- Success rate for payload is directly proportional to thee time in years they eventually perfect the launches.
- Low weighted payloads perform better than heavier payloads.
- Orbit GEO,HEO,SSO,ES L1 has the ebay Success rate.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

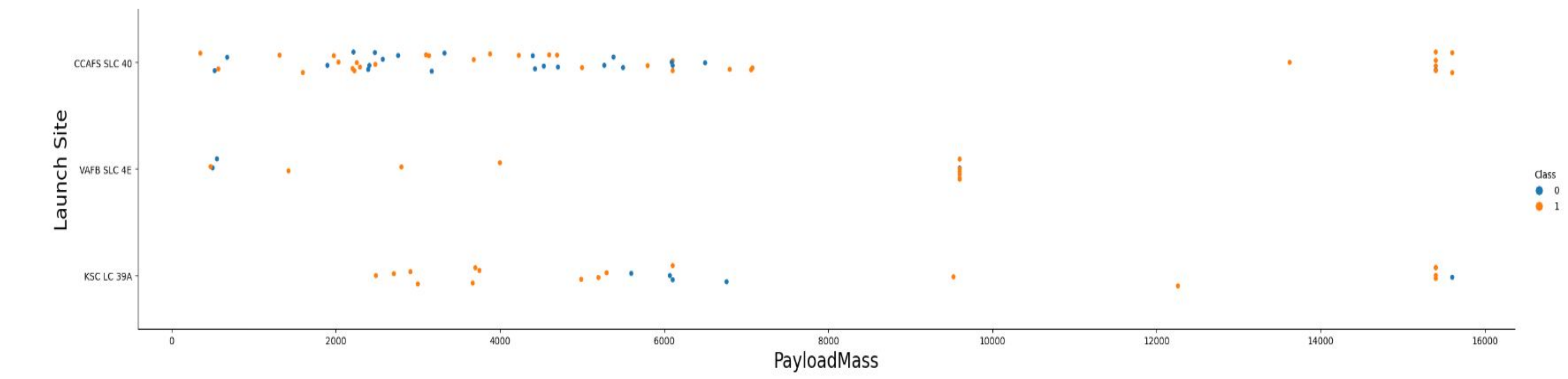
Scatter plot of Flight Number vs. Launch Site



Launches from site CCAFS LC 40 is higher than launches from other sites

Payload vs. Launch Site

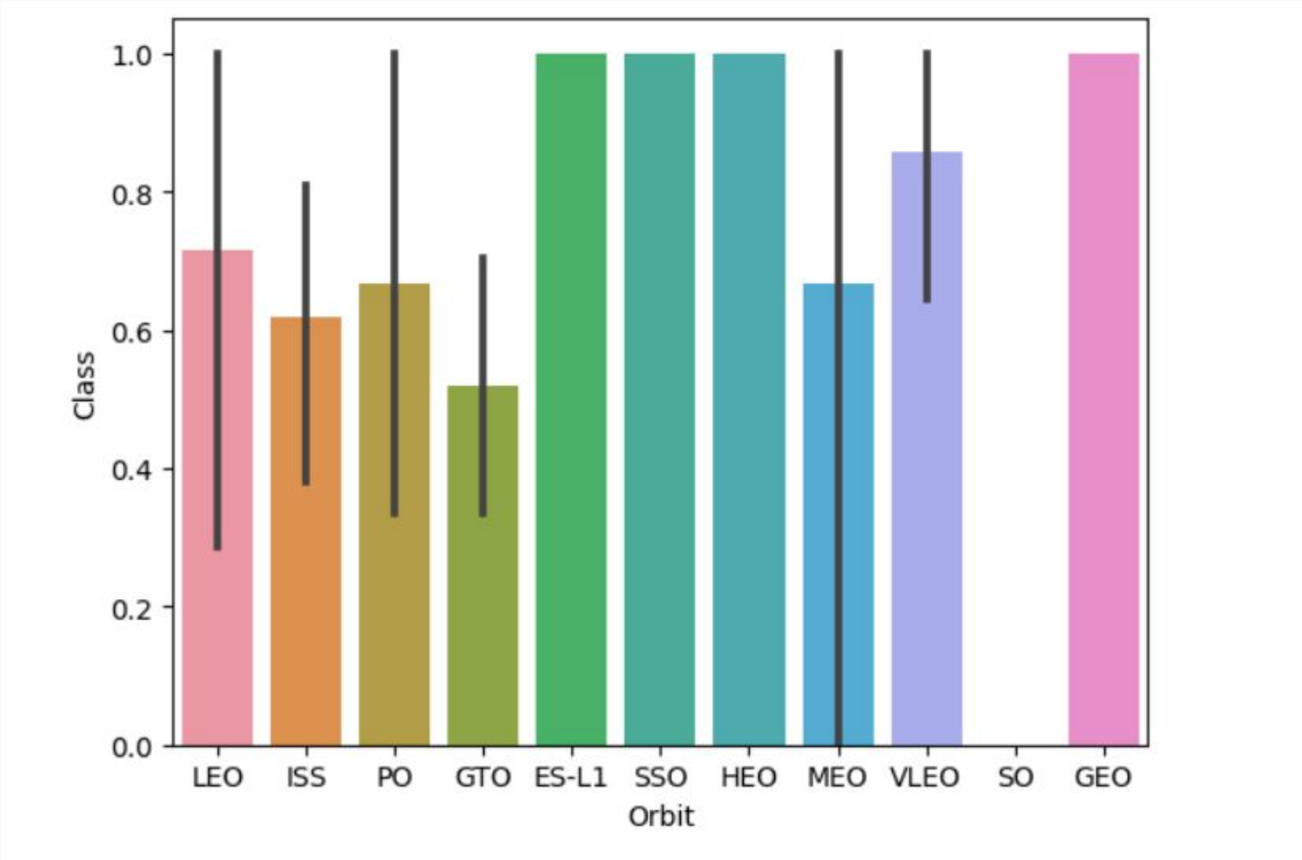
Show a scatter plot of Payload vs. Launch Site



Majority of lower mass payload rockets have been launched from CCAFS SLC 40

Success Rate vs. Orbit Type

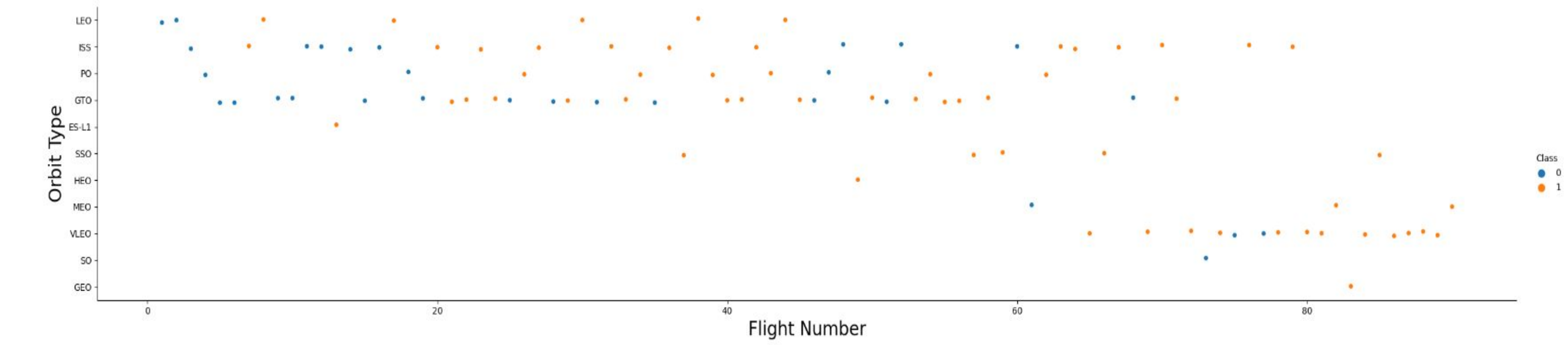
Show a bar chart for the success rate of each orbit type



The orbits having highest success rates are GEO HEO SSO and ES-L1.

Flight Number vs. Orbit Type

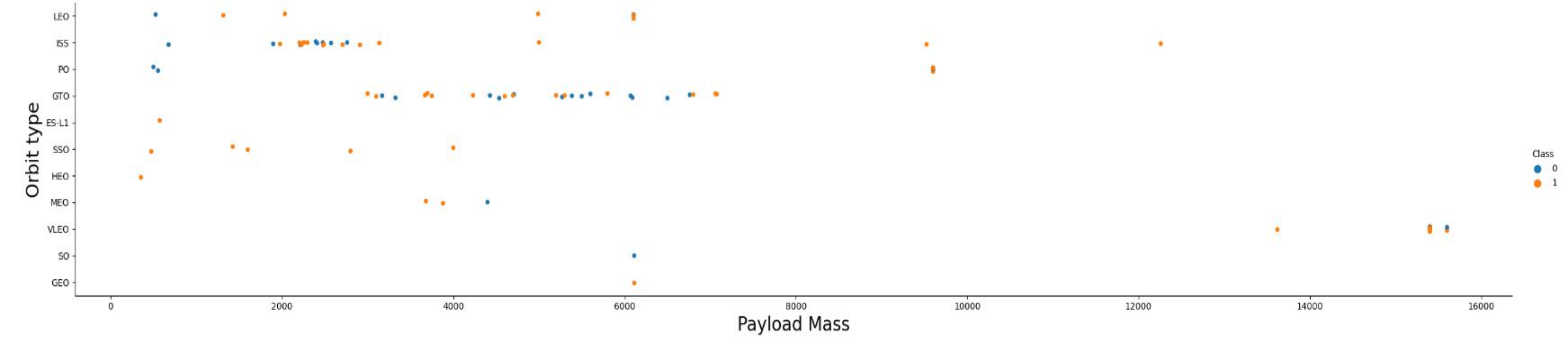
Show a scatter point of Flight number vs. Orbit type



In recent years the orbit have been changed to VLEO.

Payload vs. Orbit Type

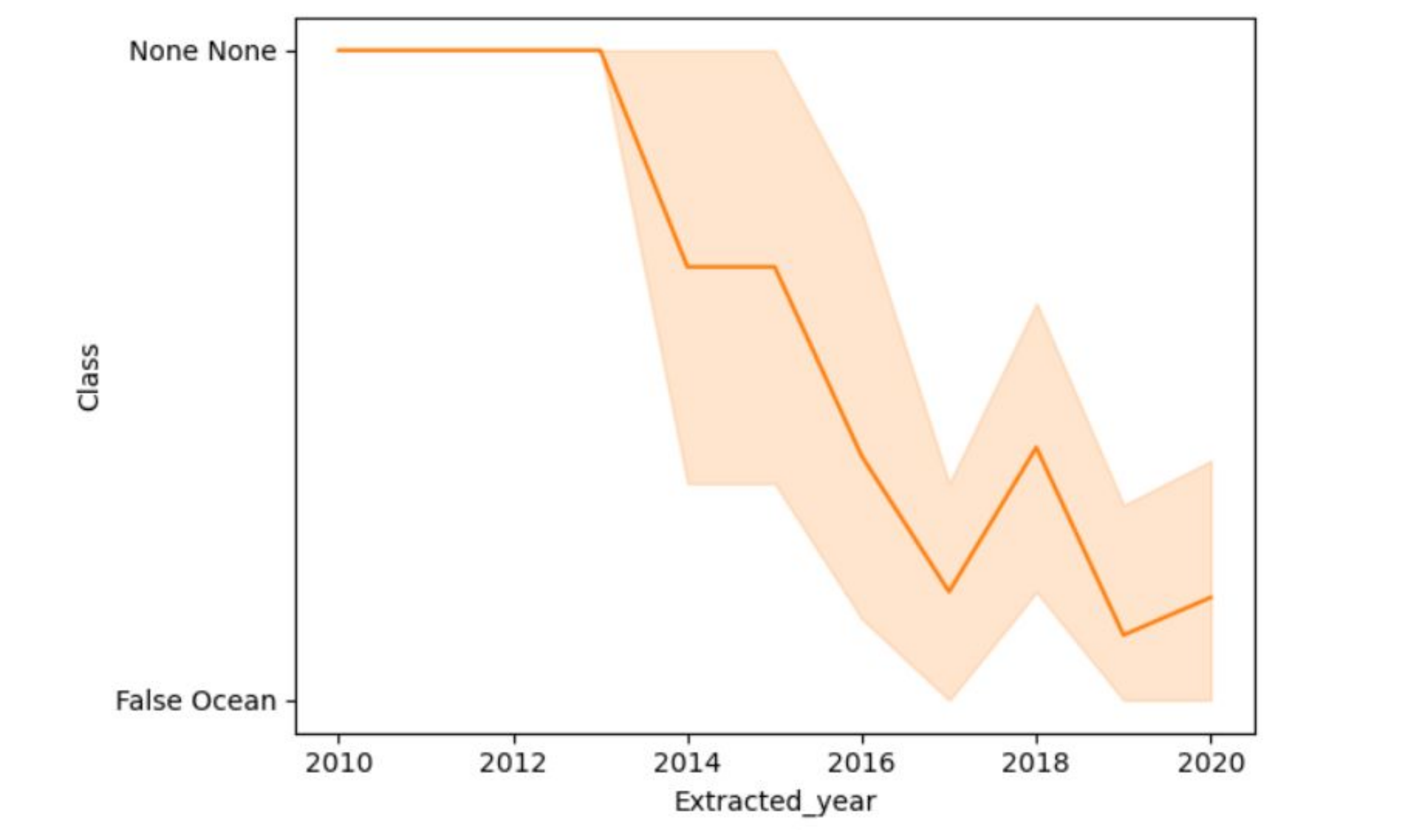
Show a scatter point of payload vs. orbit type



There is a strong correlation between ISS and Payload around 2000 as well as between GTO at the range of 4000-8000

Launch Success Yearly Trend

Show a line chart of yearly average success rate



Launch failure rate has been decreased from 2013 which means the success rate is increasing drastically..

All Launch Site Names

%sql SELECT DISTINCT Launch_Site FROM SPACEXTBL

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

DISTINCT keyword is used to show unique launch sites.

Launch Site Names Begin with 'CCA'

%sql SELECT * from SPACEXTBL where Launch_Site like 'CCA%' limit 5

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Used the query to display five record where launch site begins with ‘CCA’

Total Payload Mass

%sql SELECT SUM(PAYLOAD_MASS__KG_) as Total_payload_mass from SPACEXTBL where Customer='NASA (CRS)'

Total_payload_mass
45596

Calculated total payload mass from NASA as 45596 uusing the above query.

Average Payload Mass by F9 v1.1

%sql SELECT AVG(PAYLOAD_MASS__KG_) as Average_payload_mass from SPACEXTBL where Booster_Version='F9 v1.1'

Average_payload_mass
2928.4

Calculated average payload mass for F9 v1.1 as 2925.4 using the above query.

First Successful Ground Landing Date

%sql SELECT MIN(Date) AS FirstSuccessfull_landing_date FROM SPACEXTBL WHERE Landing__Outcome LIKE 'Success (ground pad)'

	firstsuccessfull_landing_date
0	2015-12-22

Found that the first successful landing on ground pad was on 22nd December 2015.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql  SELECT "Booster_Version" FROM SPACEXTBL  where
'Landing_Outcome'='Success (drone ship)'  and
PAYLOAD_MASS__KG_ >4000  and PAYLOAD_MASS__KG_<6000
```

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Used the WHERE clause to determine successful landing on drone ship and used AND clause to obtained the bosterversion having payload mass between 4000 and 6000 Kg.

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT COUNT(Mission_Outcome) AS SuccessOutcome FROM
SPACEXTBL WHERE Mission_Outcome LIKE 'Success%';
```

SuccessOutcome
100

```
%sql SELECT COUNT(Mission_Outcome) AS FailureOutcome FROM
SPACEXTBL WHERE Mission_Outcome LIKE 'Failure%';
```

FailureOutcome
1

Boosters Carried Maximum Payload

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL ORDER BY Booster_Version
```

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

2015 Launch Records

%sql SELECT Booster_Version, Launch_Site, Landing_Outcome FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure (drone ship)' AND Date BETWEEN '2015-01-01' AND '2015-12-31'

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

%sql SELECT Landing_Outcome, COUNT(Landing_Outcome) FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY COUNT(Landing_Outcome) DESC

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

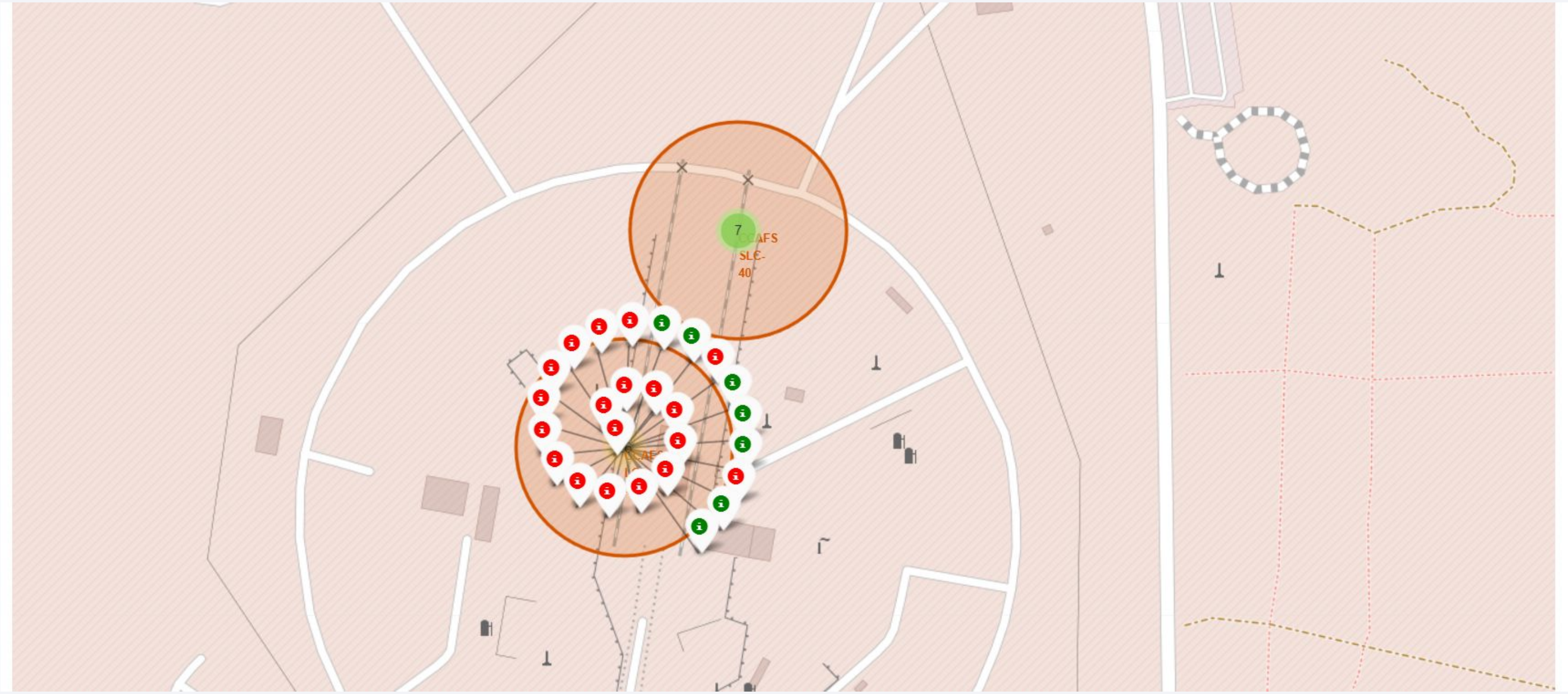
Section 3

Launch Sites Proximities Analysis

All Launch Sites

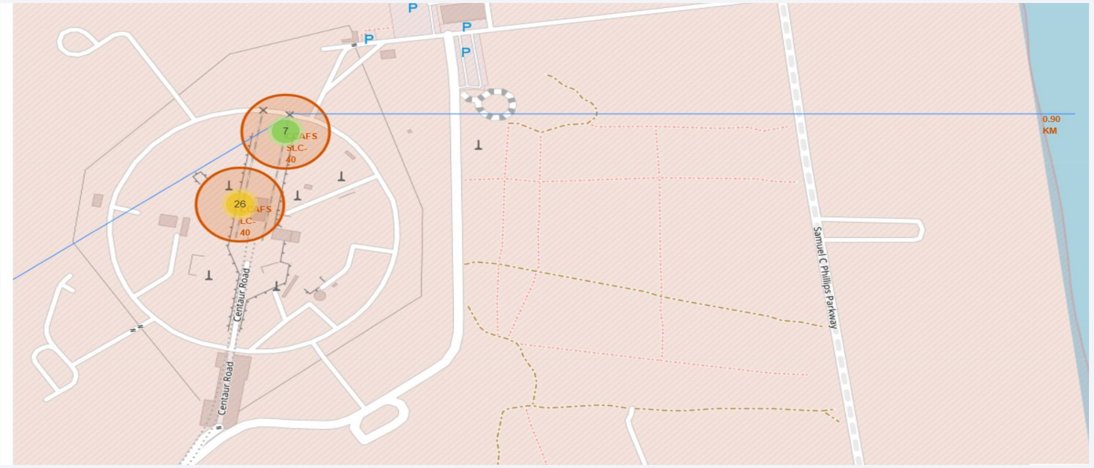
The SpaceX launch sites are in USA coasts.

Launch sites with markers showing outcome.

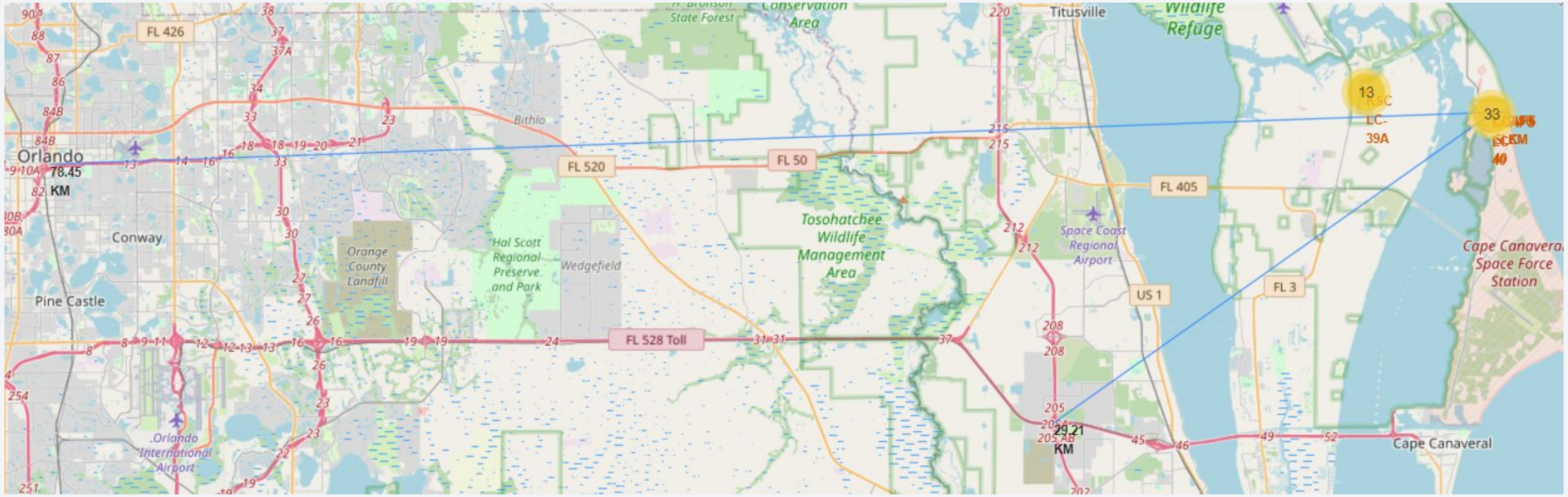


Green marker show successful launch site and red marker show failure launch sites.

Launch Site distance to coastline and railway station



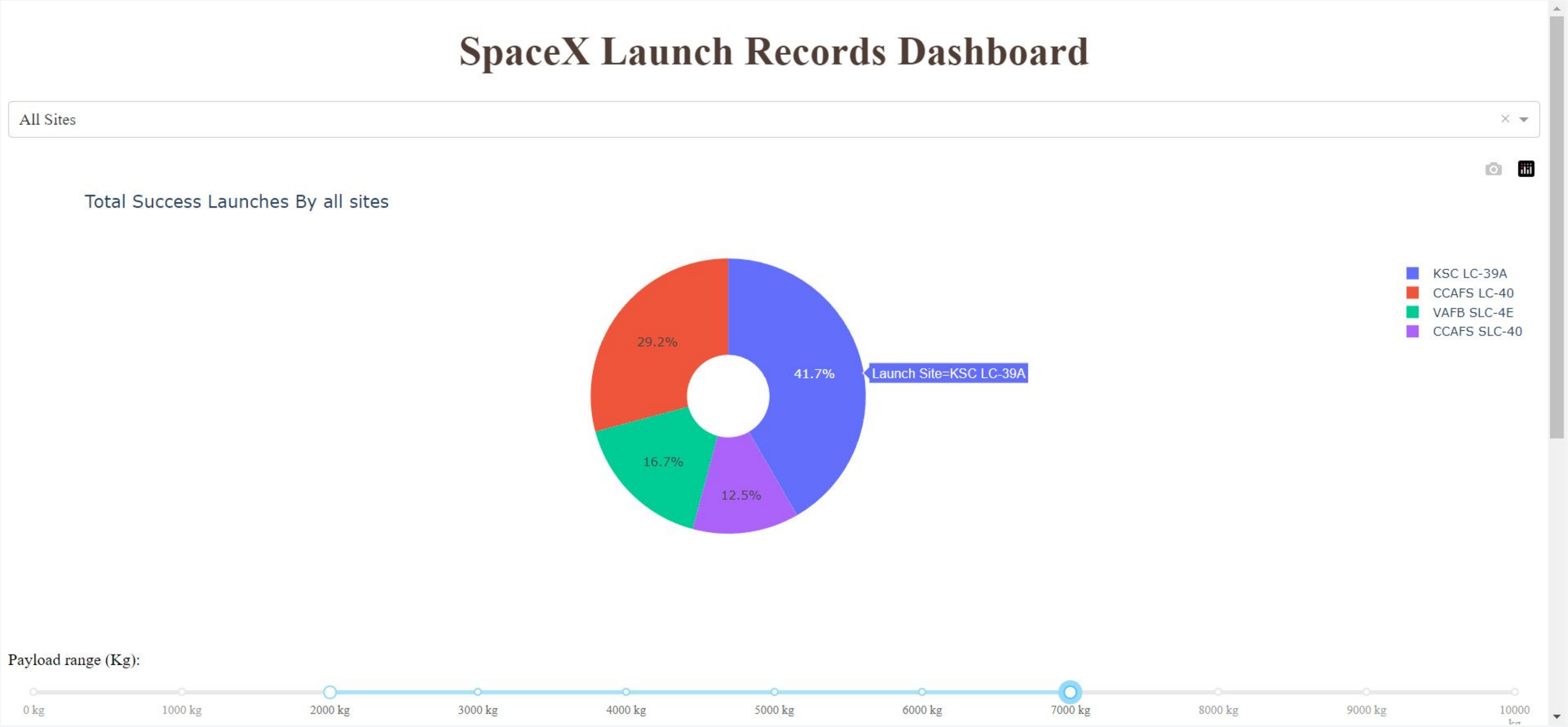
Launch site distance to coastline and railway station is specified.



Section 4

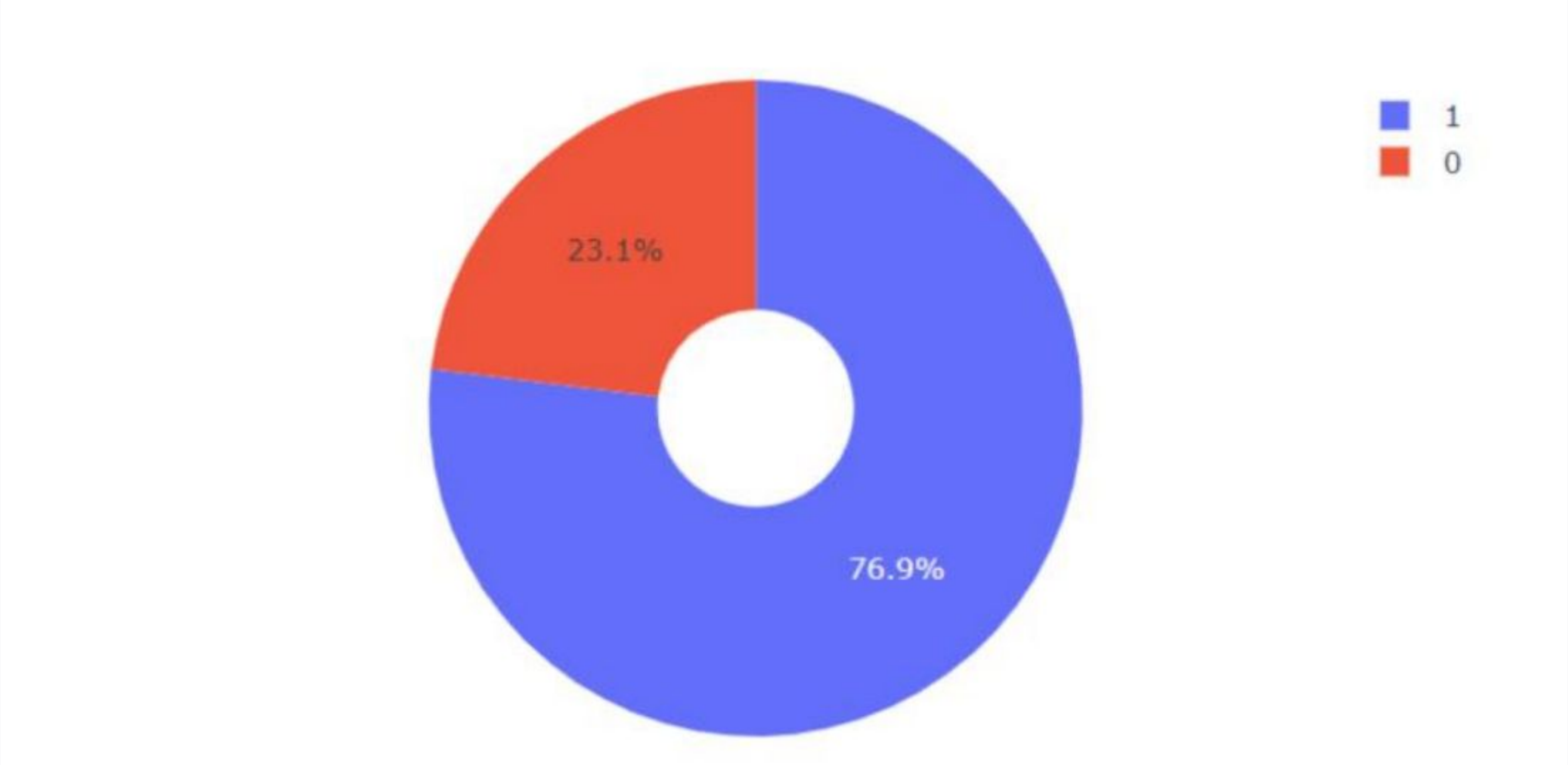
Build a Dashboard with Plotly Dash

Launch site success percentage.



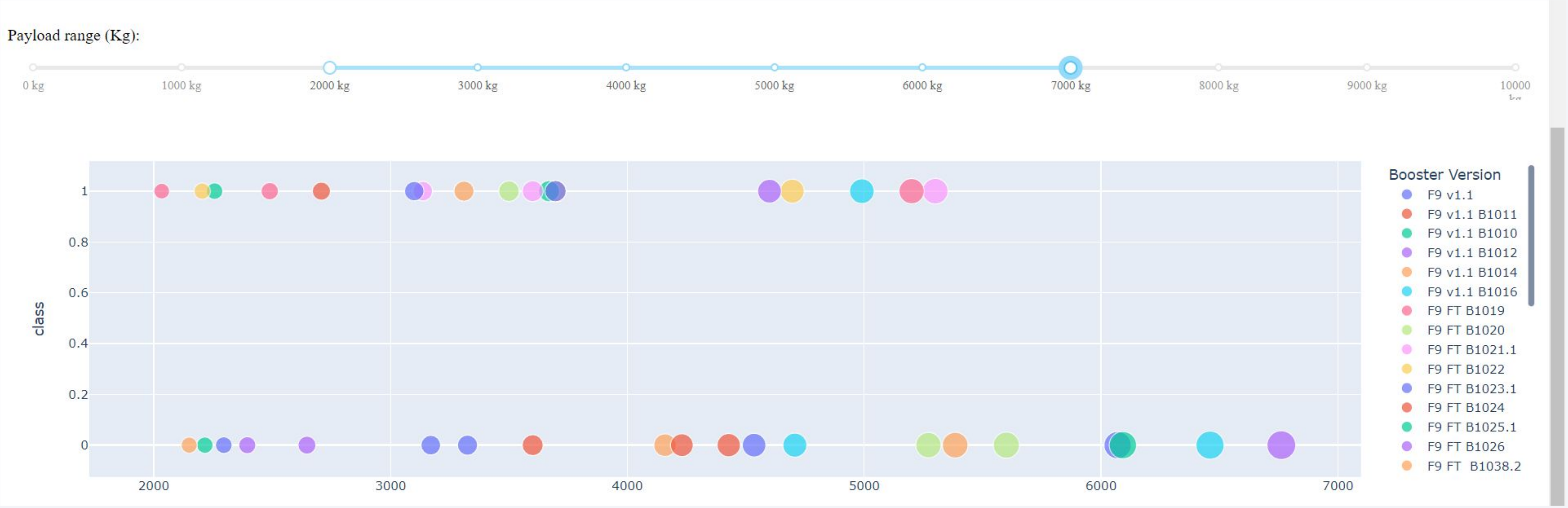
KSC LC 39A has the most successful launches from all sites.

Launch site with highest success rate.



KSC LC 39A achieved 76.9% success rate and a failure rate of only 23.1%

Payload vs Launch outcome with a payload range slider



Success rate of low weighted payload is greater than heavy weighted payload.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Decision Tree classifier is the model that has the highest accuracy with an accuracy rate of 89%

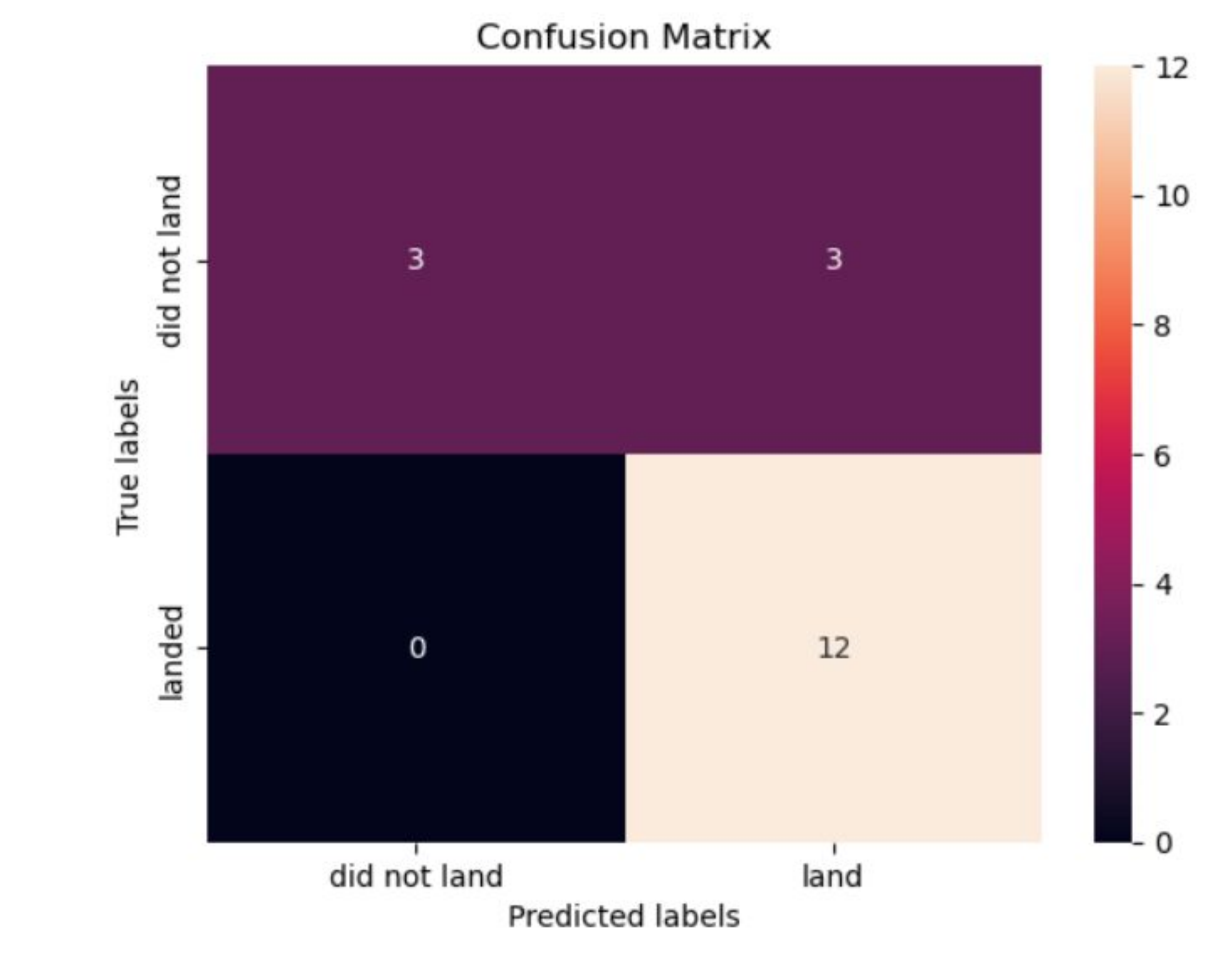
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8892857142857142
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 10, 'splitter': 'best'}
```

Confusion Matrix

The confusion matrix for decision tree classifier is shown here.
There is a problem of false positives which denotes the unsuccessful landings represented as successful ones.



Conclusions

- ❖ Success rate has started to increase in 2013 till 2020.
- ❖ Orbits ES-L1,GEO,HEO,SSO,VLEO had the most success rate.
- ❖ KSC-LC 39A had most successful launch rate.
- ❖ Launch site having larger flight number have more success rate.
- ❖ The decision tree classifier is the best model to predict the success rate, with an accuracy of 89%

Appendix

Github link for the final project

https://github.com/ahk99/IBM-CapestoneProject.git

Thank you!