IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sujitha Surendran
Mar 29 2022

Outline

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

Executive Summary

- The main intention of this project is to analyze if Falcon 9 will reuse its stage 1 by landing successfully. Space x launches Falcon 9 rocket with a cost of $62M whereas other competitors cost upward $165M each. This is mainly due to the reuse of stage 1. So, we must understand if stage 1 will land successfully to launch the rocket with minimal cost. Data was collected from Space x REST API URL and performed data wrangling using Beautiful Soup to obtain a readable data set. Exploratory Data Analysis was done to determine features impacting the result. To visualize, scatter plots, line plots, Folium were utilized, and ML models were implemented to get the prediction on outcome.
- The results are predicted based on the ML models and also how the success rate of each launch is affected by features selected.

Introduction

- We are on the verge of building commercial space travel. There are major companies in this space race like blue origin, virgin galactic, and space x. The current leader in this race seems to be space x, and the reason behind that is the reusability of their stage 1. Which reduces the cost of launch from a minimum of 165 million to around 62 million per launch.
- The problem that we are trying to answer is that how can we predict the launch price of falcon 9, so that we can use this data for companies that want to compete with space x. Predicting that whether stage 1 will land successfully or not, plays a crucial role in predicting the launch price. As that stage can be reused again with different payloads, thus reducing the cost by more than half of original.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using REST API https://api.spacexdata.com/v4/launches/past and from Wikipedia.
  - Beautiful Soup framework was used to convert json to data frame.
- Perform data wrangling
  - Data was processed using one Hot encoding for categorical values in orbit, launch site, landing pad and serial. Also, irrelevant columns were removed.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - ML models like logistic regression, SVM, K-nearest neighbor, decision tree classifier were used.

Data Collection

- Data gathered through Spacex REST API

https://api.spacexdata.com/v4/launches/past

Data Collection – SpaceX API

1. Get request for capture content from API link

```
In [71]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [72]: response = requests.get(spacex_url)
```

2. Convert json content to dataframe using normalize method

```
In [76]: # Use json_normalize meethod to convert the json result into a dataframe
         space_data = json.loads(response.text)
         space_df = pd.json_normalize(space_data)
```

3..

3. Data wrangling by removing missing values with mean

```
In [101]: # Calculate the mean value of PayloadMass column
          mean_plm = data_falcon9['PayloadMass'].mean()

          # Replace the np.nan values with its mean value
          data_falcon9['PayloadMass'].replace(np.nan, mean_plm)
          data_falcon9.to_csv('dataset_part_1.csv', index= False)
```

- Github Link

Data Collection - Scraping

1. Get text content from webpage using BeautifulSoup object

```
In [7]: # use requests.get() method with the provided static_url
        # assign the response to a object
        resp = requests.get(static_url)
        resp.status_code

Out[7]: 200
```

Create a `BeautifulSoup` object from the HTML `response`

```
In [8]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
        soup = BeautifulSoup(resp.text,'html.parser')
```

2. Extract list of column names using soup object

Next, we just need to iterate through the `<th>` elements and apply the provided `extract_column_from_header()` to extract column nam

```
In [20]:
# 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
column_names = []
element = first_launch_table.find_all('th')
for i in range(len(element)):
    name = extract_column_from_header(element[i])
    if(name is not None and len(name)>0):
        column_names.append(name)
    else:
        continue
```

- Github Link

3. Create pandas dataframe with extracted column names and empty dictionary keys

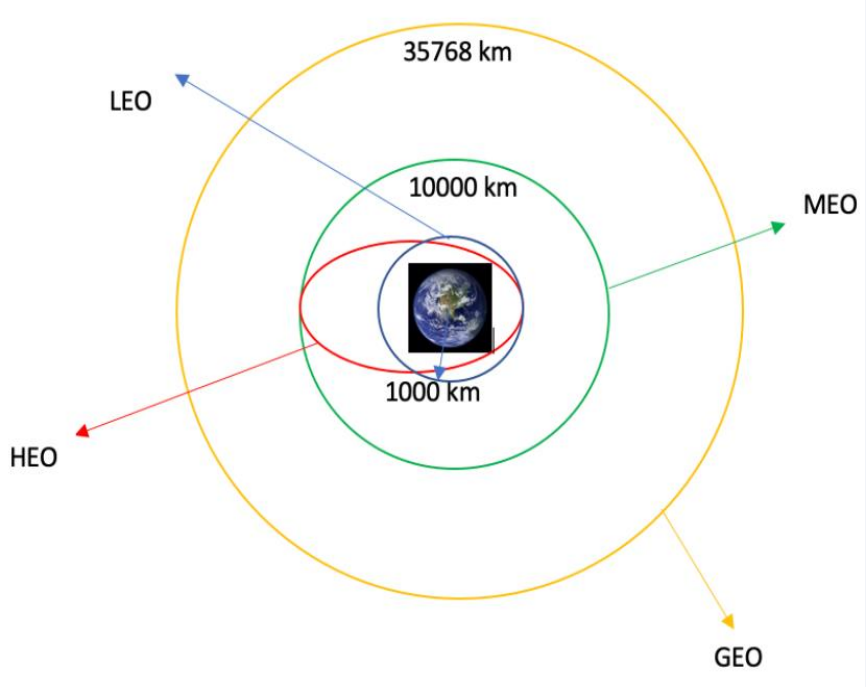
```
In [22]: launch_dict= dict.fromkeys(column_names)

# 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'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

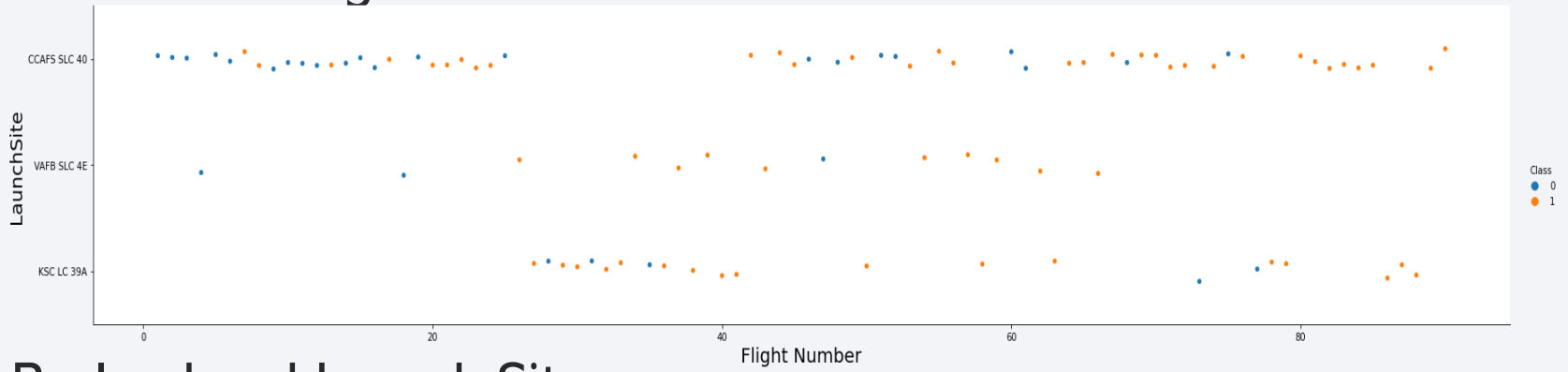
Data Wrangling

- Determining the missing values in each of the column
- Calculated number of launches in each site
- Calculated number and occurrences of each orbit
- Calculated number and occurrences of mission outcome per orbit type
- Create landing outcome label
- [Github Link]

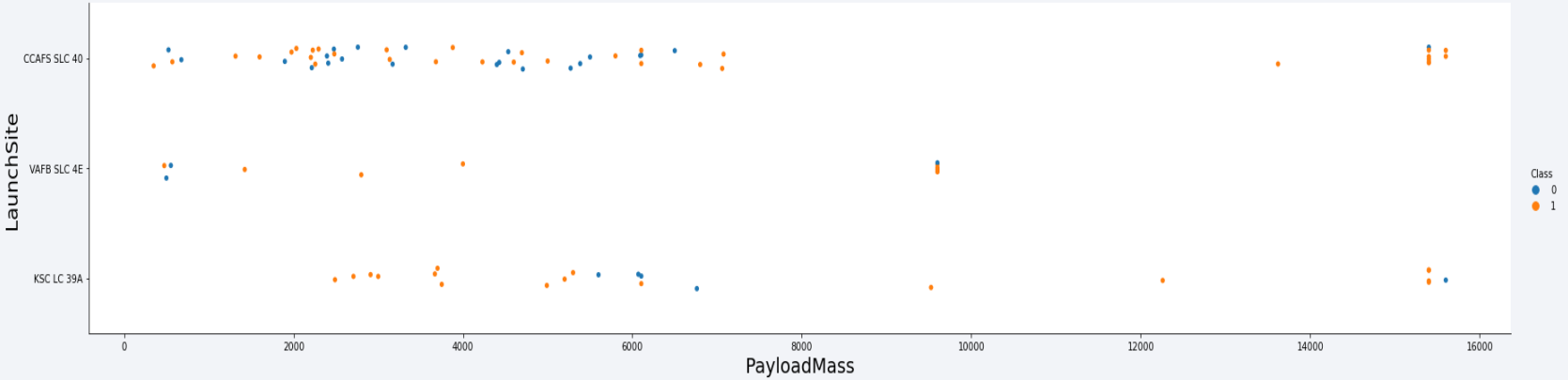


EDA with Data Visualization

- Performed Exploratory Data Analysis to understand the relationship between Flight Number and Launch site



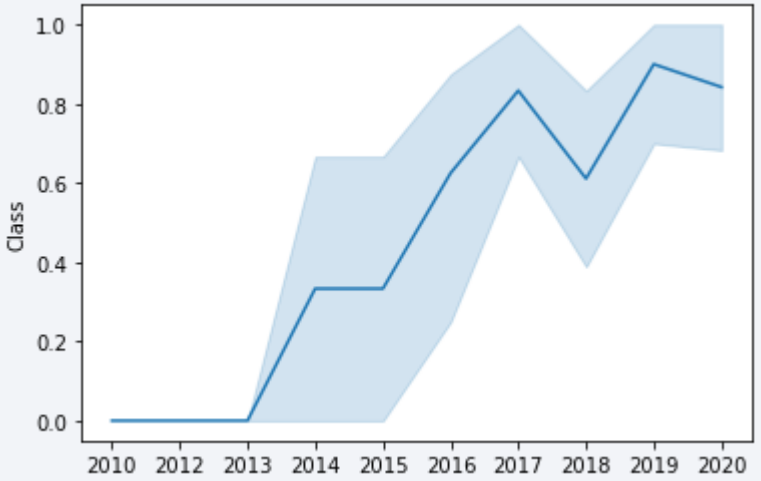
Payload and Launch Site



- [Github Link]

EDA with Data Visualization

- Determining the success rate of each year – 2013 has a steady raise in success rate till 2020



- Feature engineering using the features determined through visualization process

```
In [47]: features = df[['FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', 'GridFins', 'Reused', 'Legs', 'LandingPad', 'Block', 'ReusedCount', 'Serial']]
         features.head()
```


- Github Link

EDA with SQL

Following queries been performed in jupyter notebook.

- The names of unique launch sites in the space mission.
- The total payload mass carried by boosters launched by NASA (CRS)
- The average payload mass carried by booster version F9 v1.1
- Names of booster versions with maximum payload capacity
- The total number of successful and failure mission outcomes
- The failed landing outcomes in drone ship, their booster version and launch site names.

- Github Link

Build an Interactive Map with Folium

- Marked launch sites and added map objects like markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Assigned feature launch outcomes (failure or success) to class 0 and 1 i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, identified launch sites that have relatively high success rate.
- Calculated the distance between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.

- Github Link

Build a Dashboard with Plotly Dash

- Built an interactive dashboard with Plotly dash.
- Plotted pie charts with total launches by each site
- Plotted scatter graph with the relationship between landing Outcome and Payload Mass (Kg) for various booster version.

- Github Link

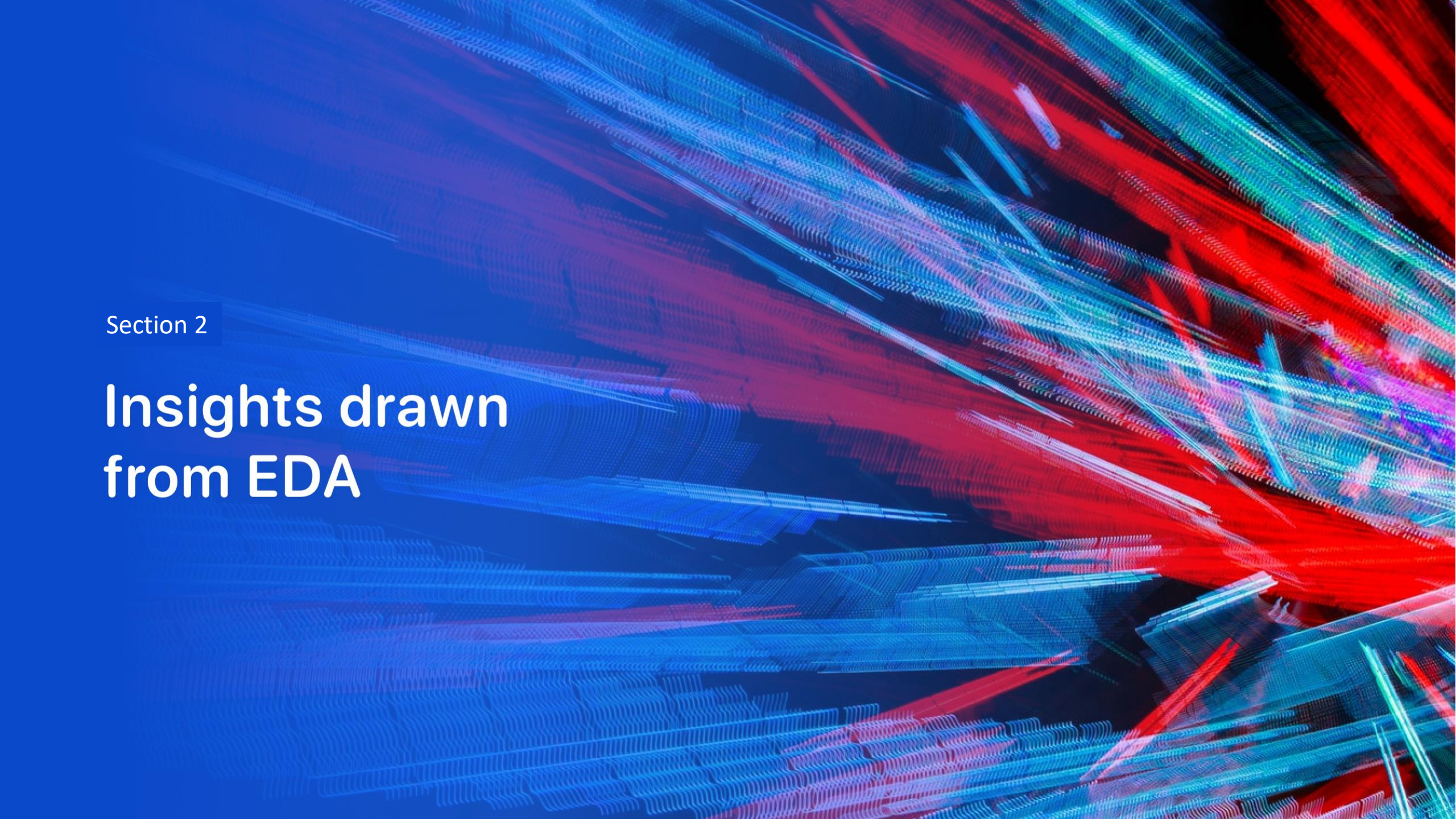
Predictive Analysis (Classification)

- Loaded the dataframe using pandas.
- Using Numpy, determined the Y variable and standardized/transformed X variable.
- Data is split into training and testing with random state 2.
- Built various machine learning models and tune different hyperparameters using GridSearchCV.
- Determined accuracy for each of the model, improved the model using feature engineering and algorithm tuning.
- Identified the best performing classification model.

- Github Link

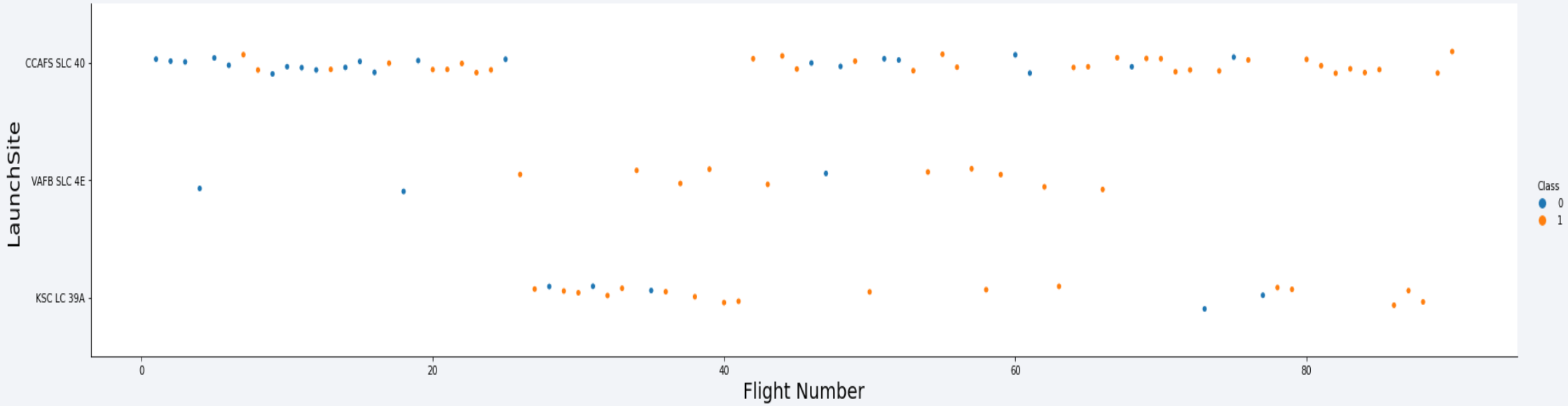
Results

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


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

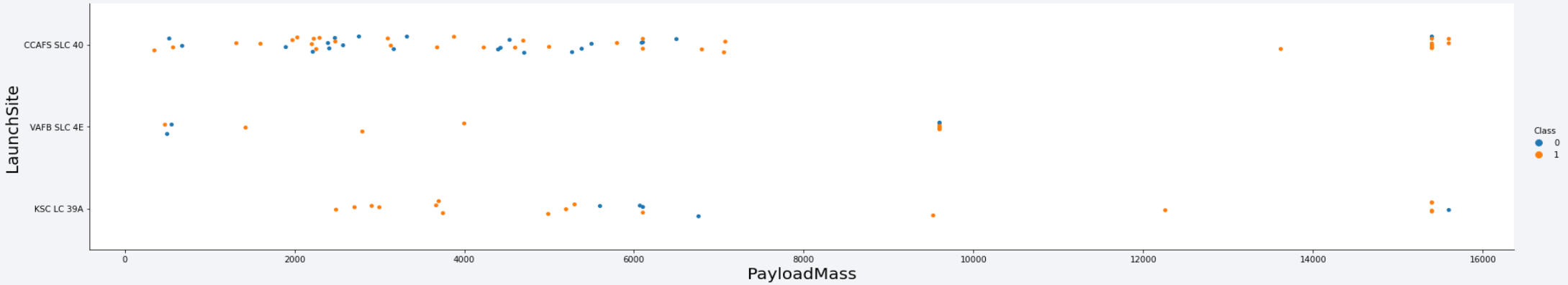
- Scatter plot of Flight Number vs. Launch Site



- The higher amount of flights at a launch site has greater success rate

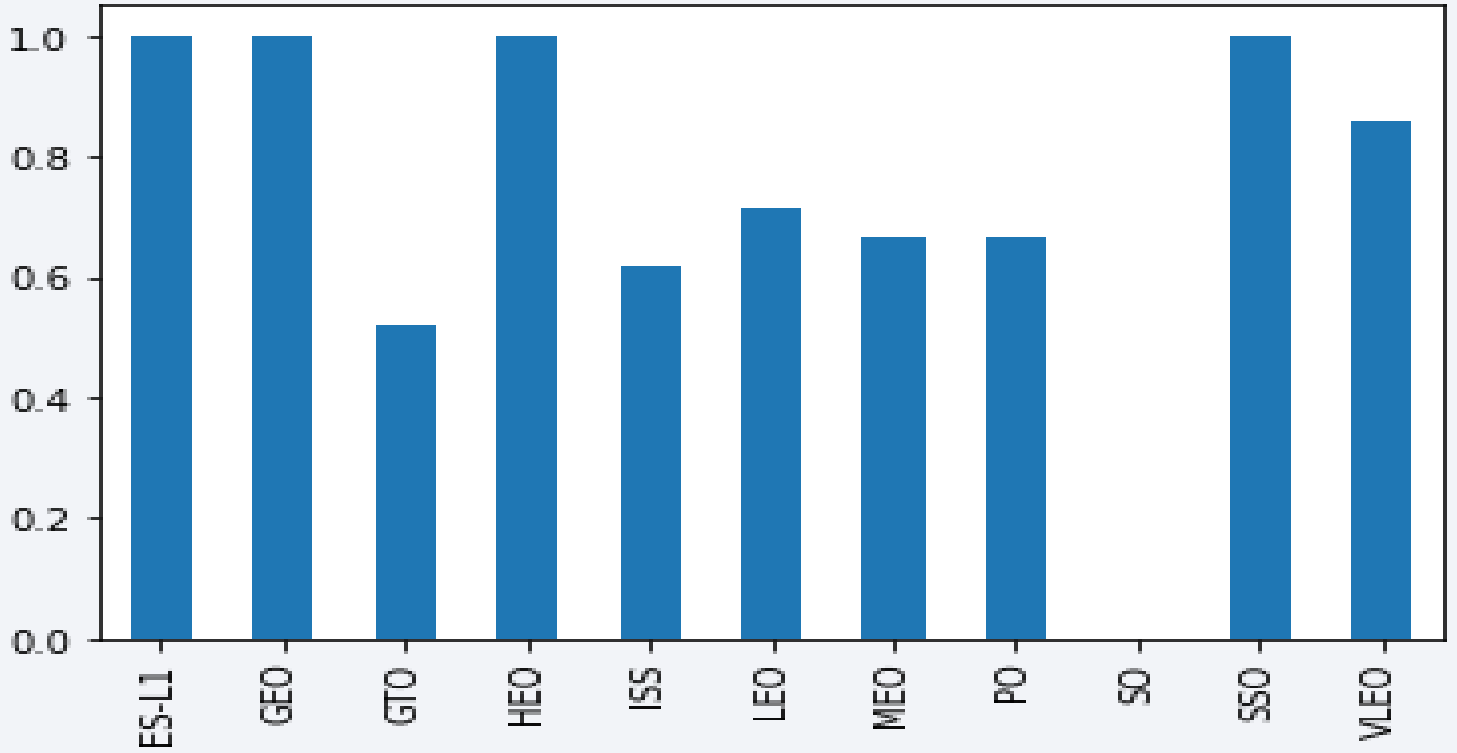
Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site



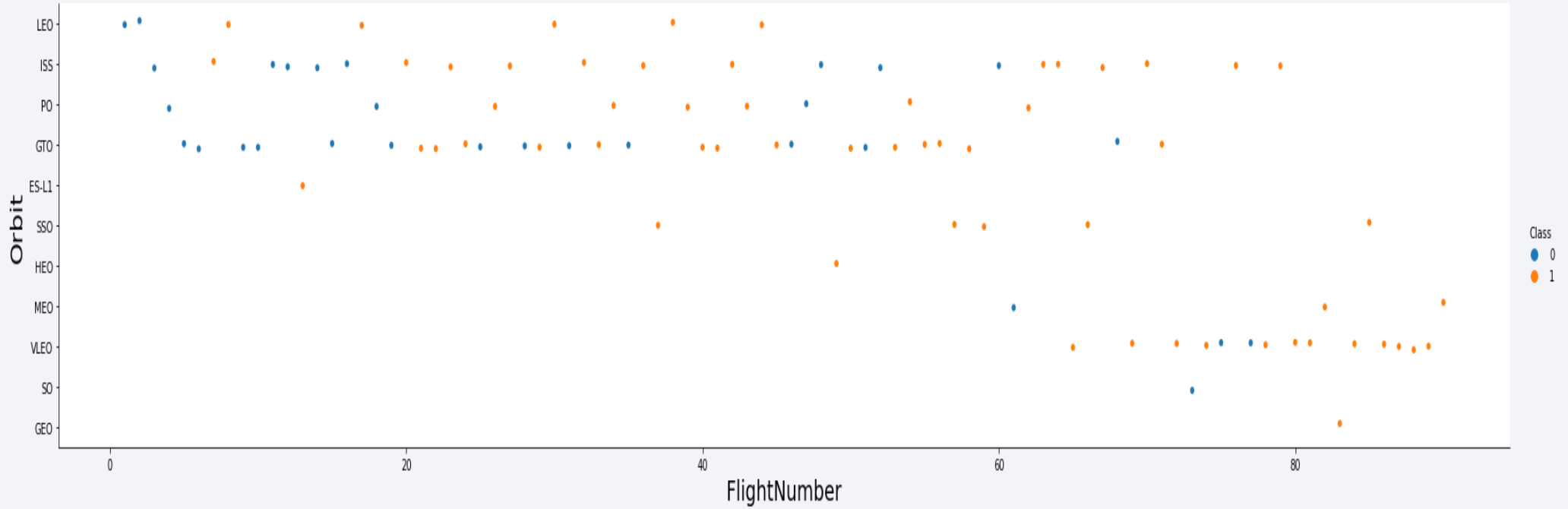
- Lesser payload mass faced many failures than the higher payload mass.

Success Rate vs. Orbit Type



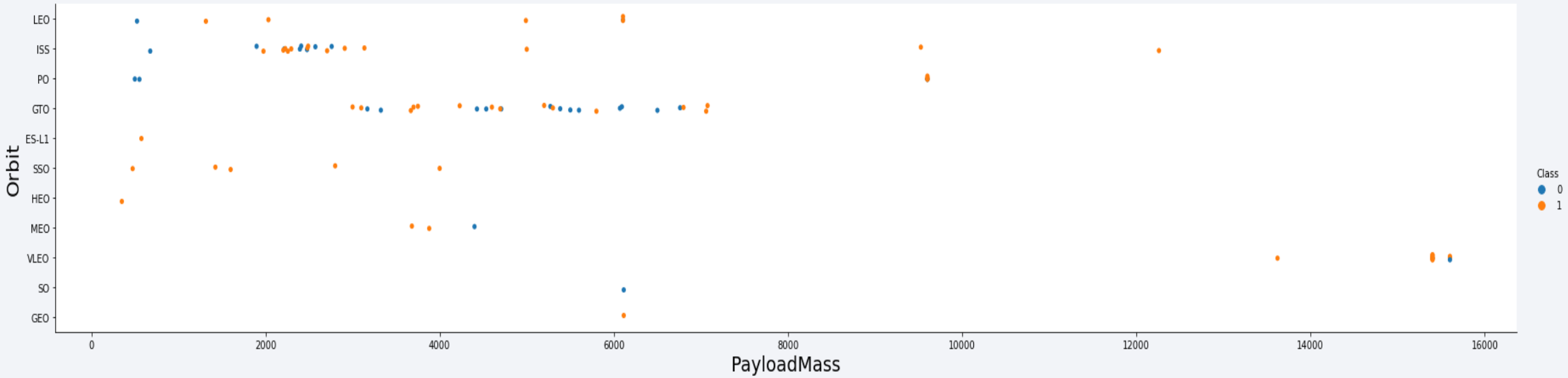
- Per bar diagram, ES-L1, GEO, HEO, SSO has the higher success rate than the other Orbit types

Flight Number vs. Orbit Type



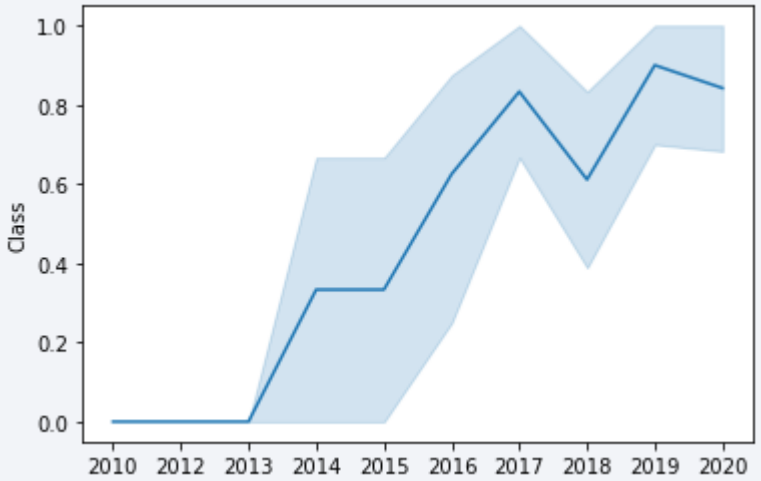
- In GTO orbit, success rate is not related to Flight Number
- In LEO orbit, higher the flight number better the success rate

Payload vs. Orbit Type



- Higher payload mass has reduced the success rate in GTO orbit
- Higher payload has also increased the success rate in ISS, LEO orbits

Launch Success Yearly Trend



- There is a steady hike in success rate from the year 2013

All Launch Site Names

- Used Distint keyword to fetch unique launch site names from the table

```
In [13]: %sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL
```

Out[13]:

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

Launch Site Names Begin with 'CCA'

- Used Like keyword in launch_site column name to start with ‘CCA’
- To limit to 5 records used Limit 5

In [17]:
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

Out[17]:

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

Total Payload Mass

- Used Sum() function to calculate total payload mas launched by NASA

Display the total payload mass carried by boosters launched by NASA (CRS)

In [22]:
```
%sql select sum(payload_mass__kg_) from spacextbl where customer =  'NASA (CRS)'
```

Out[22]:

1
45596

Average Payload Mass by F9 v1.1

- Determined the average payload mass taken by Falcon 9 using avg() function

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

In [23]:
```
%sql select avg(payload_mass__kg_) from spacextbl where booster_version like 'F9 v1.1%'
```

Out[23]:

1
2534

First Successful Ground Landing Date

- Found the date of the first successful landing outcome on ground pad using min() function on DATE column

```
In [27]: %sql select min(DATE) from spacextbl where mission_outcome = 'Success'
```

Out[27]:

1
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

- Retrieved the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 using ‘Between’ keyword

```
In [15]: %sql select booster_version from spacextbl where payload_mass__kg_ between 4000 and 6000 and mission_outcome = 'Success'
```

Out[15]:

booster_version
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1021.2
F9 FT B1032.1
F9 B4 B1040.1
F9 FT B1031.2
F9 FT B1032.2
F9 B4 B1040.2
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5 B1046.3
F9 B5B1054
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5B1060.1
F9 B5 B1058.2
F9 B5B1062.1

Total Number of Successful and Failure Mission Outcomes

- Calculated the total number of successful and failure mission outcomes

```
%sql select mission_outcome, count(mission_outcome) from spacextbl group by mission_outcome
```

```
 * ibm_db_sa://nmm70478:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- Listed the names of the booster which have carried the maximum payload mass

```
%sql select booster_version from spacextbl where payload_mass__kg_ in (select MAX(payload_mass__kg_) from spacextbl)
```

```
 * ibm_db_sa://nmm70478:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

1]:

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

2015 Launch Records

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

```
%sql select booster_version, launch_site from spacextbl where mission_outcome LIKE  'Failure%' and year(DATE)='2015'
```

```
 * ibm_db_sa://nmm70478:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1018	CCAFS LC-40

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

- Ranked 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 select mission_outcome, count(mission_outcome) from spacextbl where date between '2010-06-04' and '2017-03-20' group by mission_outcome
```

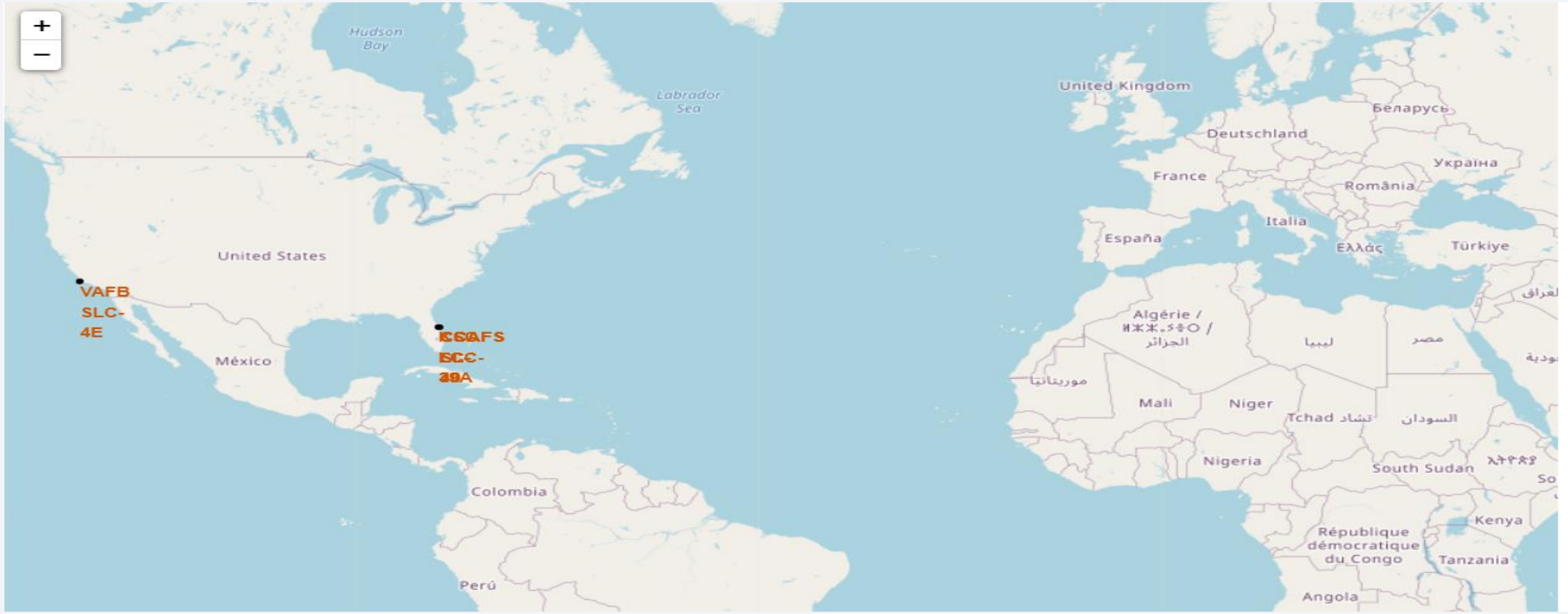
```
 * ibm_db_sa://nmm70478:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

mission_outcome	2
Failure (in flight)	1
Success	30

Section 3

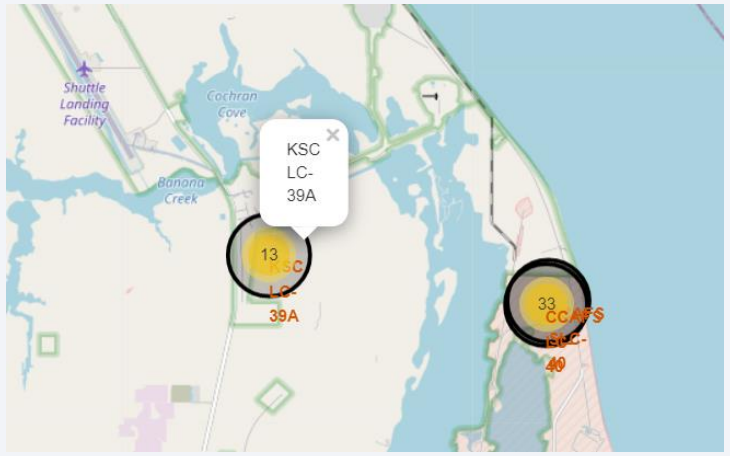
Launch Sites Proximities Analysis

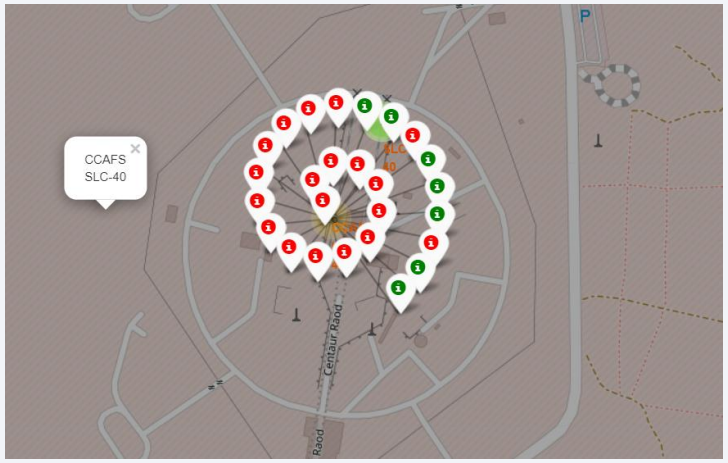
Launch sites Global markers

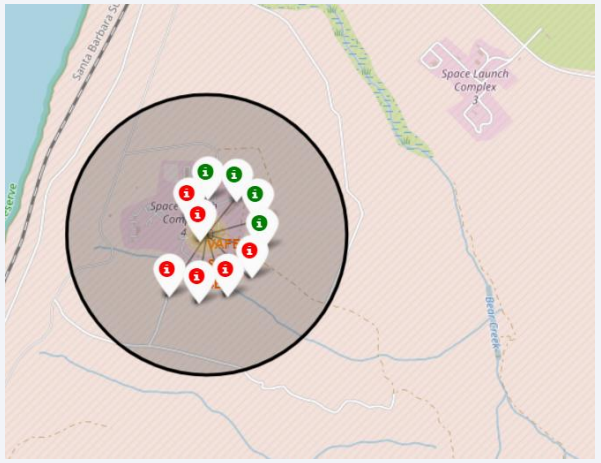


- The launch sites are located at United States coasts. Florida and California

Markers with Success/failure Color labels

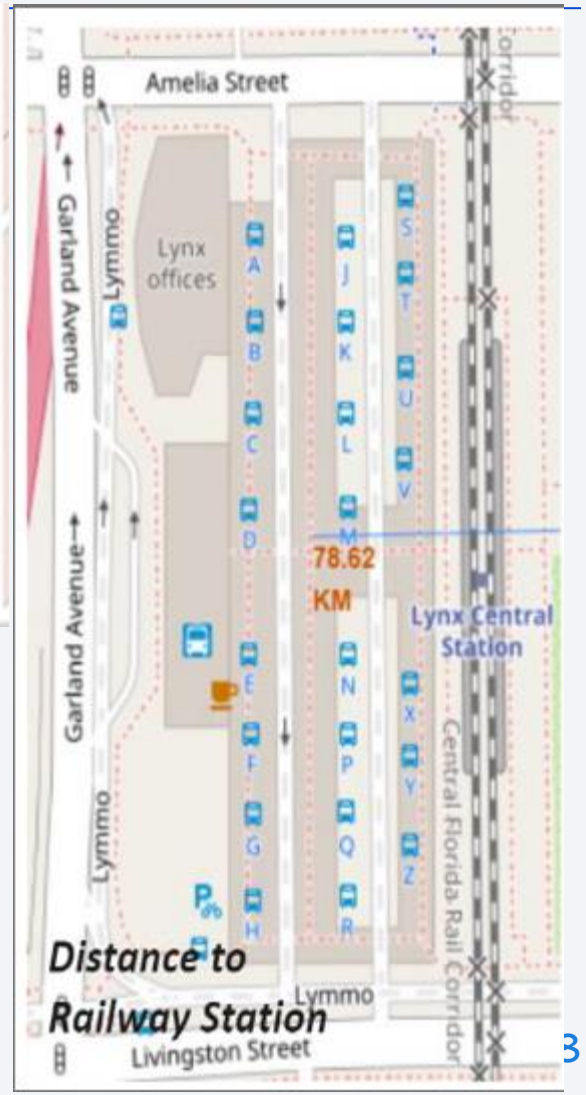




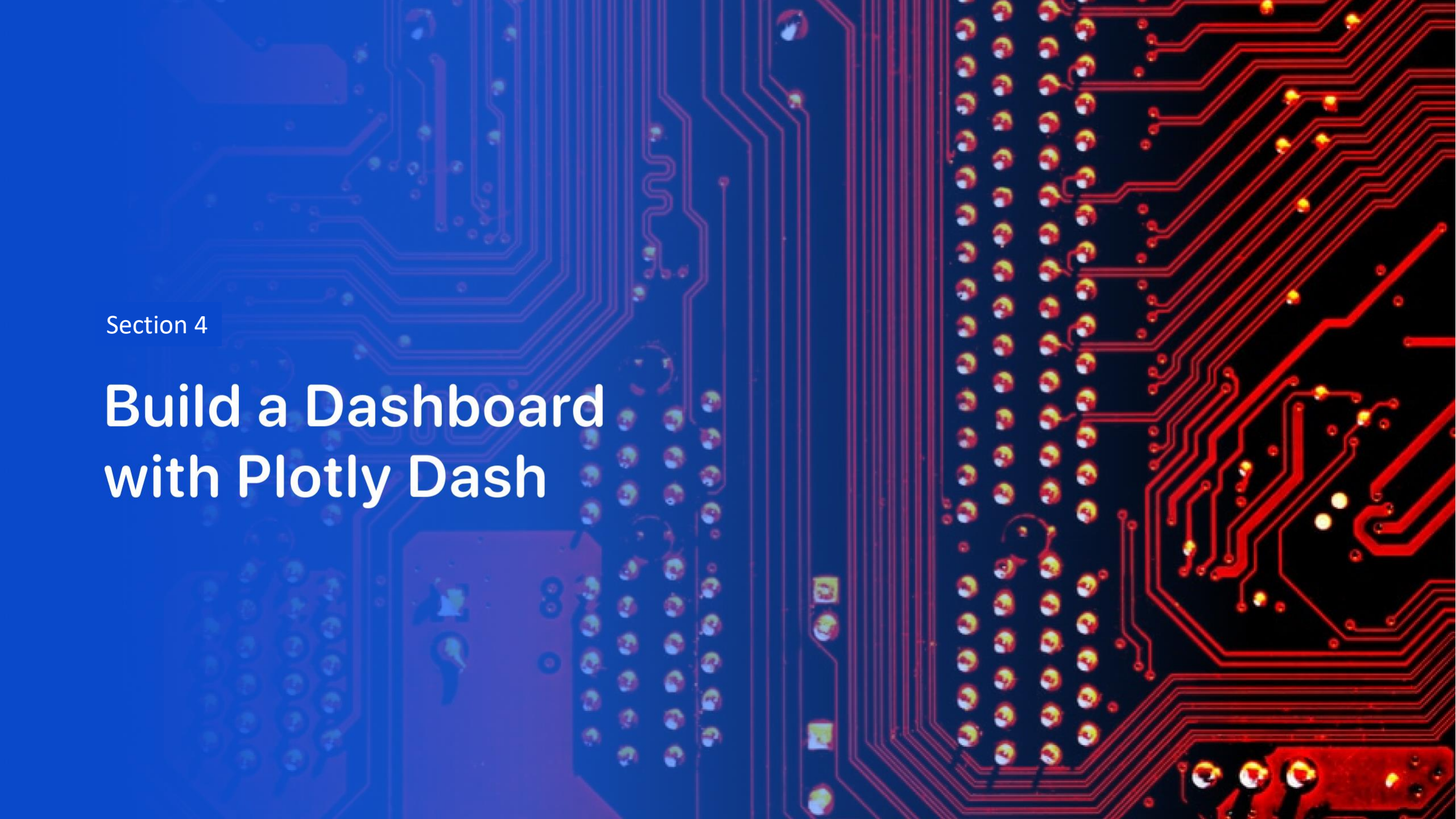


- Florida launch Site
- Green markers show successful launches and Red markers show the failure ones.

Distance between landmarks and Launchsites

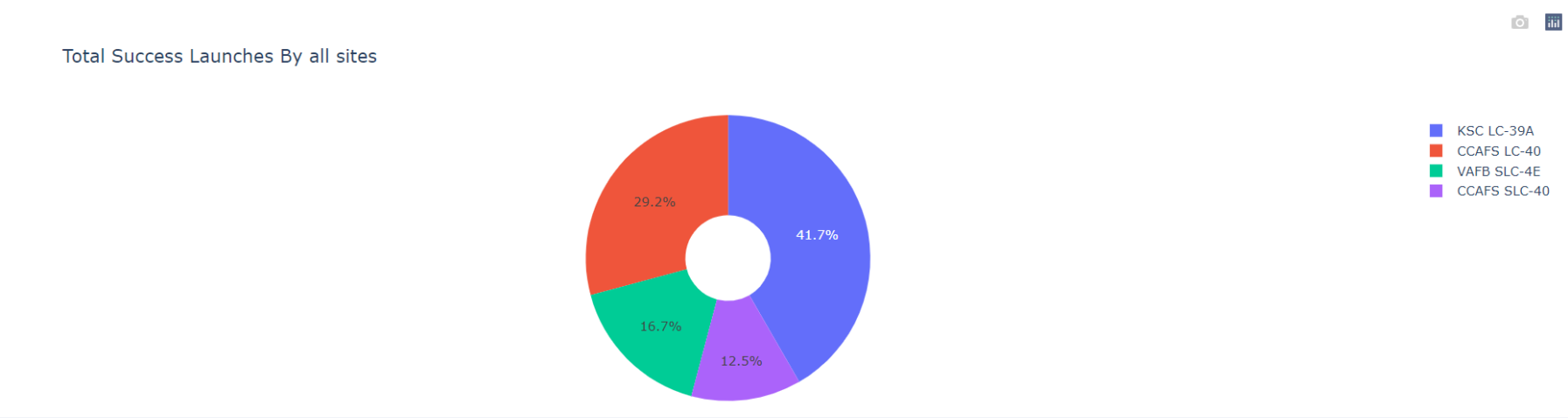


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

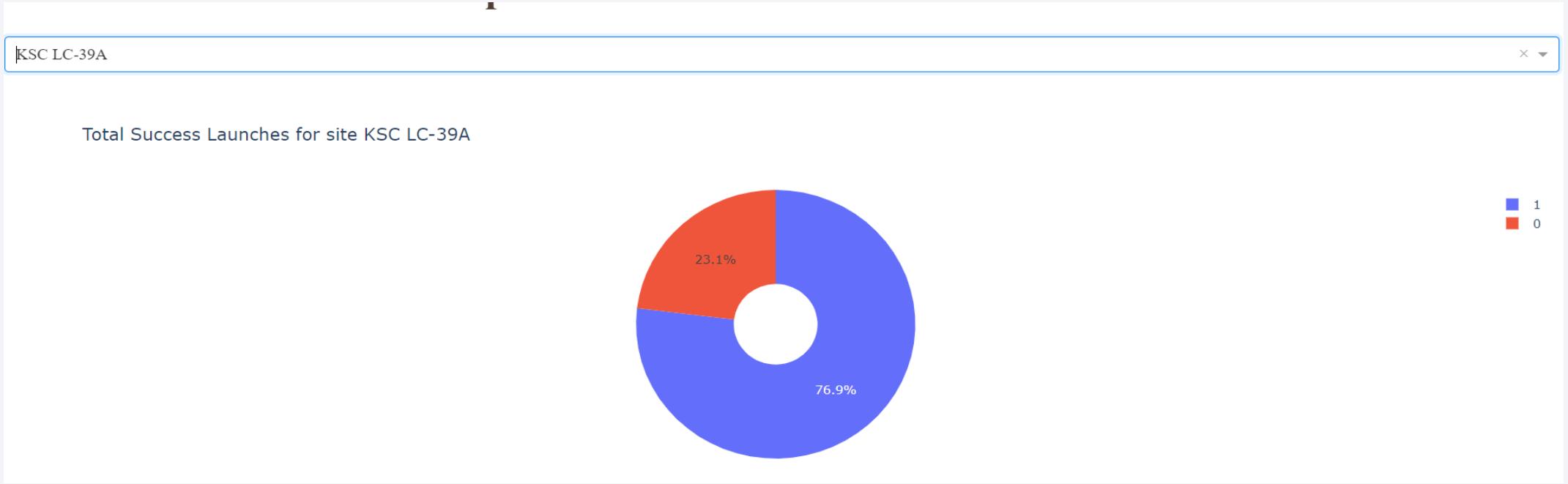

Section 4
Build a Dashboard with Plotly Dash

Dashboard showing success rate of each Launch site

- Pie chart depicts that launch Site KSL LC-39A has the highest success rate



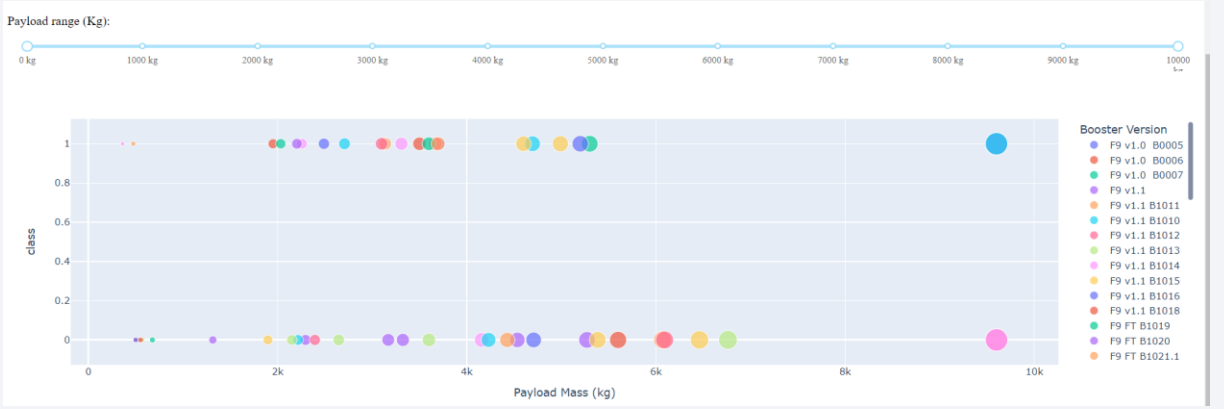
Launch Site with highest success ratio



- KSC LC-39A has 76.9% success and 23.1% failure rate

Payload Vs Launch outcome scatter plot

- Payload weight from 0kg to 10k kg



- Payload weight from 5000kg to 8000kg

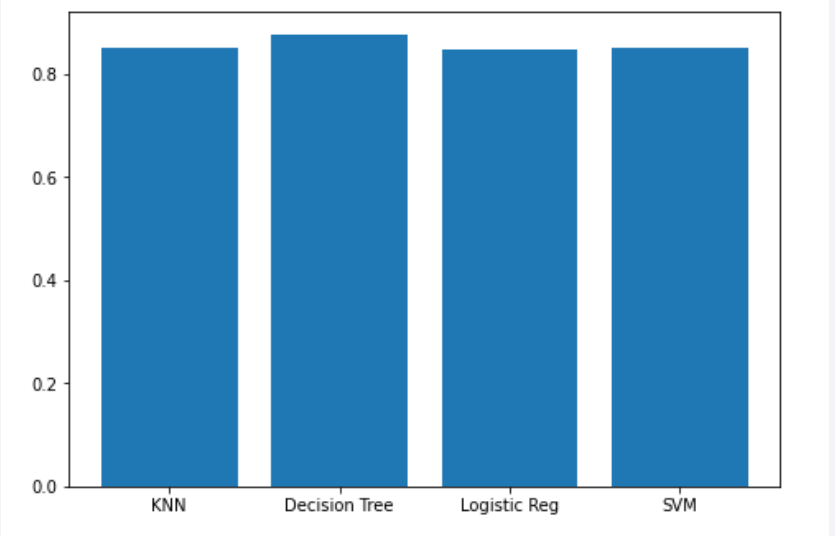
We see the lesser the Payload weight higher the success rate

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Decision Tree Classification model has the highest accuracy among all algorithms




```
In [33]: models = {'KNeighbors':knn_cv.best_score_,
                  'Decision Tree':tree_cv.best_score_,
                  'Logistic Regression':logreg_cv.best_score_,
                  'SupportVector': svm_cv.best_score_}

         best = max(models, key=models.get)
         print('Best model is', best, 'with a score of ', models[best])
         if(best=='Decision Tree'):
             print('Best parameter is', tree_cv.best_params_)
         if(best=='KNeighbors'):
             print('Best parameter is', knn_cv.best_params_)
         if(best=='SupportVector'):
             print('Best parameter is', svm_cv.best_params_)
         if(best=='Logistic Regression'):
             print('Best parameter is', logreg_cv.best_params_)

         Best model is Decision Tree with a score of  0.8767857142857143
         Best parameter is {'criterion': 'entropy', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'random'}
```

Confusion Matrix

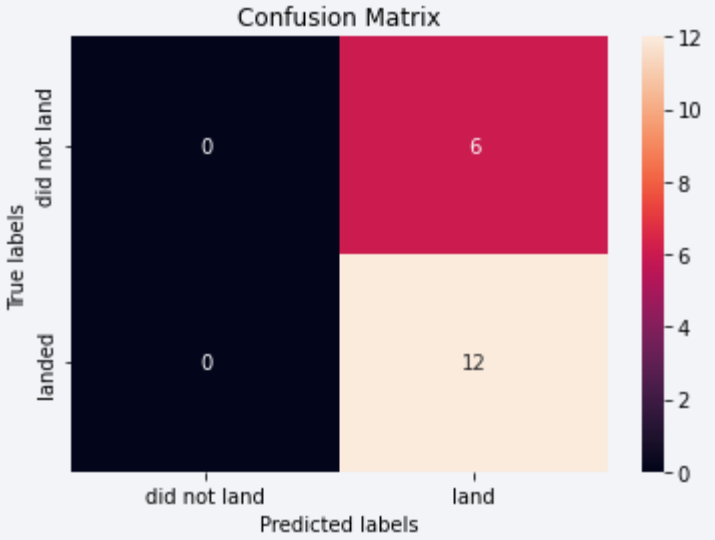
- Major issue with this result is false positives.

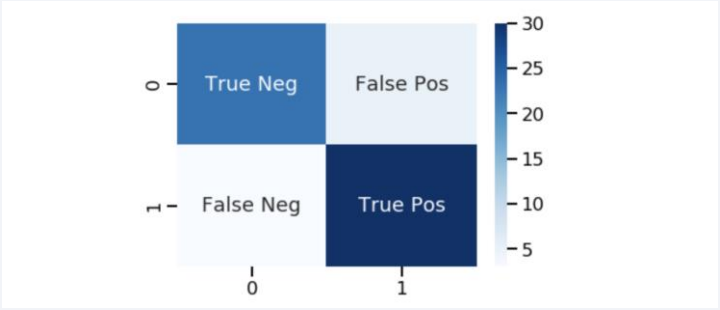
Unsuccessful landing marked as success

in the findings

- Classifier is clearly distinguishing between

various classes(Success/failure)





Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Payload mass help in success rate. The lesser the payload, the higher the success rate.
- Launch success rate started to increase in 2013 till 2020 and it will hike through the upcoming years.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best suited machine learning algorithm.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!