IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Melvin Gabriel
2 Jun 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Maps with Folium
  - Interactive Dashboard with Plotly
  - Predictive analysis
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo using screenshots
  - Predictive analysis results

Introduction

Project background and context

The goal of this project is to 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.

Problems you want to find answers

- What are the factors that determine if the rocket will land successfully?
- The relationship between various factors that determine the success rate of a successful landing.
- What is the ideal operating condition that should be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web-scrapping from Wikipedia
- Perform data wrangling
  - One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

- Data collection was done from the SpaceX API using get request
- The response was decoded as a Json using json() function, and converted to a pandas dataframe using json_normalize
- Then, we cleaned the data by checking for missing values and filling in appropriate values where necessary
- We extracted Falcon 9 launch data from Wikipedia via web scraping using Beautiful Soup. The launch data was extracted as HTML table and converted to a pandas dataframe for further analysis.

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data, filtered the data to include only Falcon 9 launches and created a flat file.
- GitHub URL: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/1.%20jupyter-labs-spacex-data-collection-api.ipynb

1. Get response from API

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

response = requests.get(spacex_url)
```

2. Use json_normalize to convert response to dataframe

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

3. Obtain more data, construct a dictionary and then convert to dataframe

```
getBoosterVersion(data)

getLaunchSite(data)

getPayloadData(data)

getCoreData(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}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
# Create a data from launch_dict
data2 = pd.DataFrame.from_dict(launch_dict)
```

4. Filter the dataframe to only include Falcon 9 launches

```
data_falcon9 = data2[data2['BoosterVersion'] == 'Falcon 9']
```

Data Collection - Scraping

- We extracted Falcon 9 launch data from Wikipedia via web scraping using Beautiful Soup. The launch data was extracted as HTML table and converted to a pandas dataframe for further analysis.
- GitHub URL: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/2.%20jupyter-labs-webscraping.ipynb

1. Get response from API

```
response = requests.get(static_url)
# assign the response to a object
html = response.content
```

2. Create beautiful soup object

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

3. Find tables and extract column names

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

```
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
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

4. Create an empty dictionary with keys from the extracted column names and fill in the launch record values into it

4. Convert the dictionary to dataframe

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

Data Wrangling

- Identified the percentage of the missing values in each attribute.
- Calculated the number of launches on each site.
- Calculated the number and occurrence of each orbit.
- Calculated the number and occurence of mission outcome per orbit type.
- Created a landing outcome label from Outcome column and exported the results to csv.
- GitHub URL: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/3.%20labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Visualized the relationship between the following using scatterplots to understand the co-relation between the variables.
  - Flight Number and Launch Site
  - Payload and Launch Site
  - FlightNumber and Orbit type
  - Payload and Orbit type
- Visualized the relationship between success rate of each orbit type using bar chart.
- Visualized the launch success yearly trend using Line graph. Chose line graph here because trends are very clearly represented by them
- Github URL: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/5.%20jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- Loaded the dataset into Db2 instance and used SQL Magic to perform the below queries
  - 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 successful landing outcome in drone ship was achieved.
  - 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, successful 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: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/4.%20jupyter-labs-eda-sql-coursera.ipynb

Build an Interactive Map with Folium

- Marked all launch sites on the map.
- Added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.•
- Using the color labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances 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 URL: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/6.%20lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

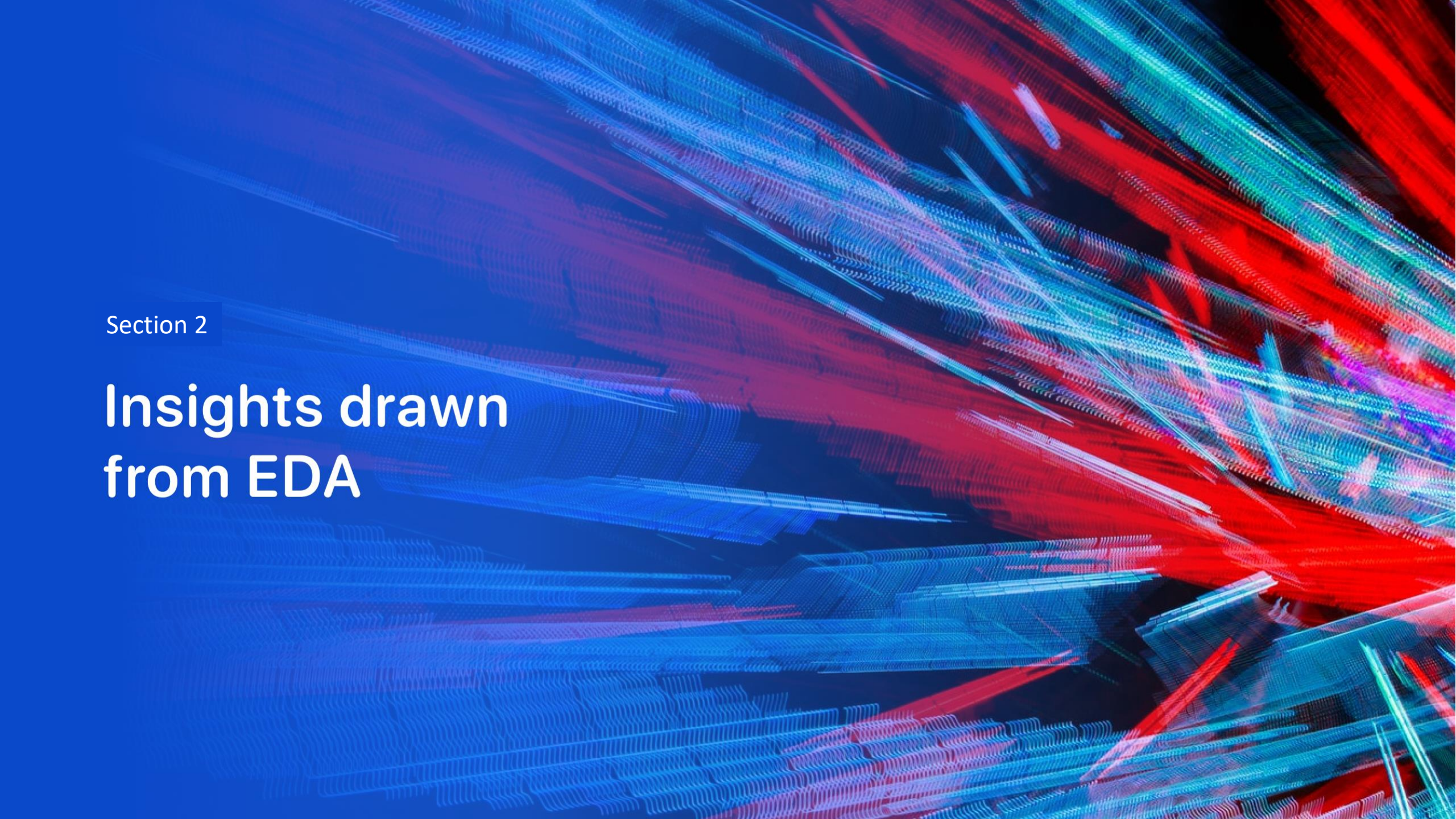
- Built an interactive dashboard with Plotly dash
- Plotted pie charts showing the total launches by a certain sites
- Plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- GitHub URL: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/7.%20Interactive%20Dashboard%20with%20Ploty%20Dash.ipynb

Predictive Analysis (Classification)

- Loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- Built different machine learning models and tune different hyperparameters using GridSearchCV
- Improved the accuracy of the model using feature engineering and algorithm tuning and found the best performing classification model.
- GitHub URL: https://github.com/MelG05/IBM-Data-Science-Final-Project/blob/master/8.%20SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

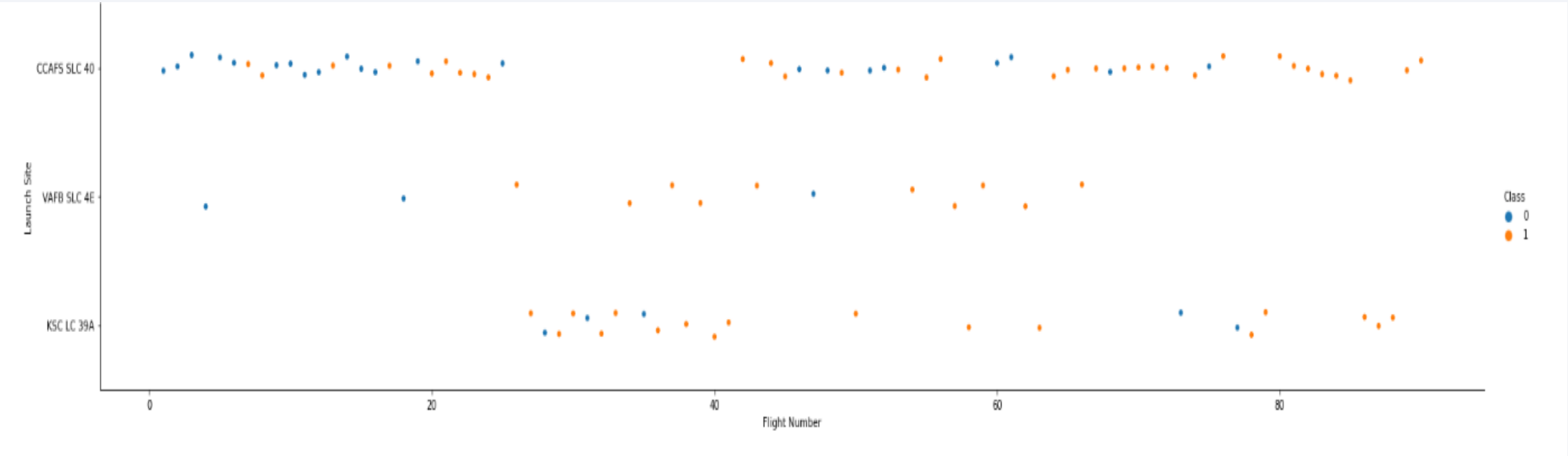
Results

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


Section 2
Insights drawn from EDA

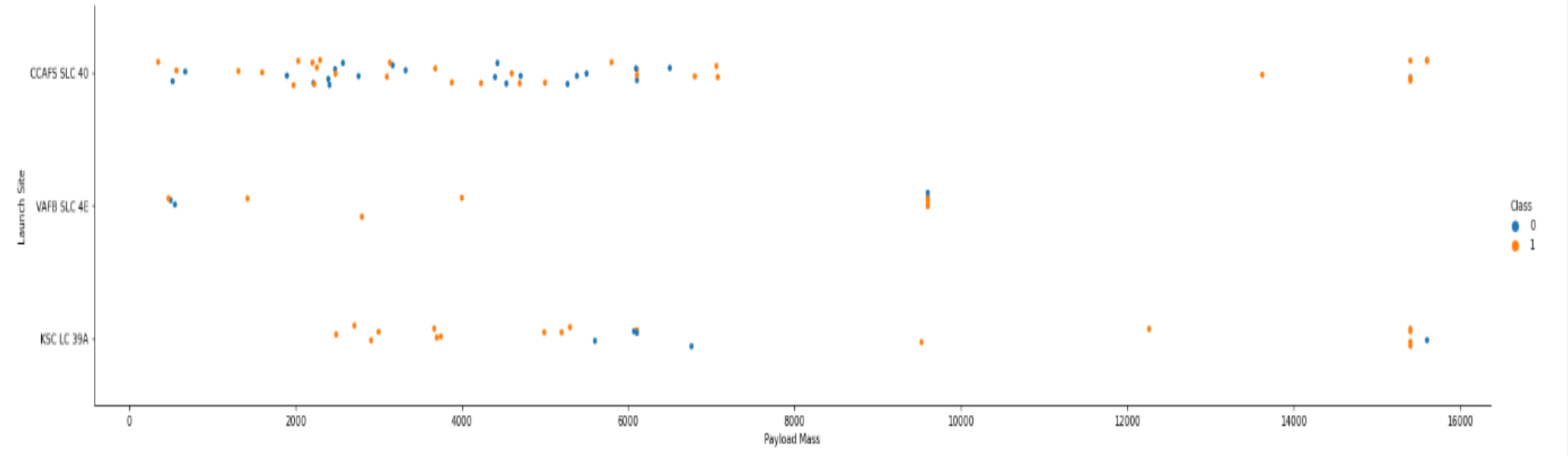
Flight Number vs. Launch Site

The higher the number of flights at a launch site the greater the success rate at a launch site.



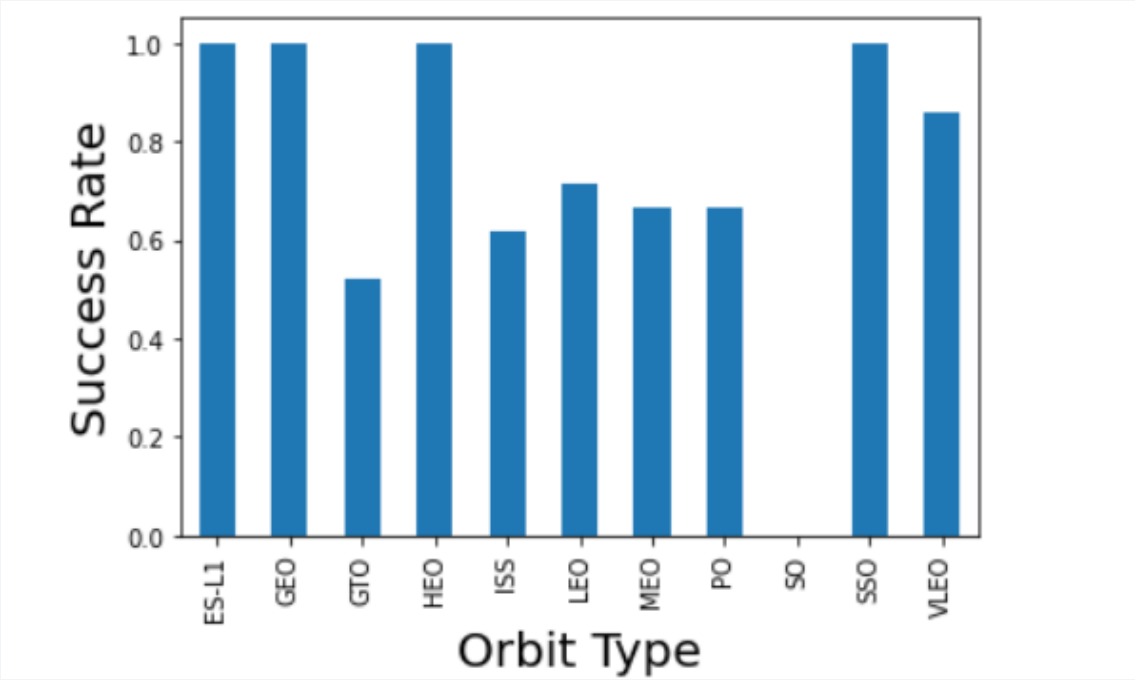
Payload vs. Launch Site

The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket.



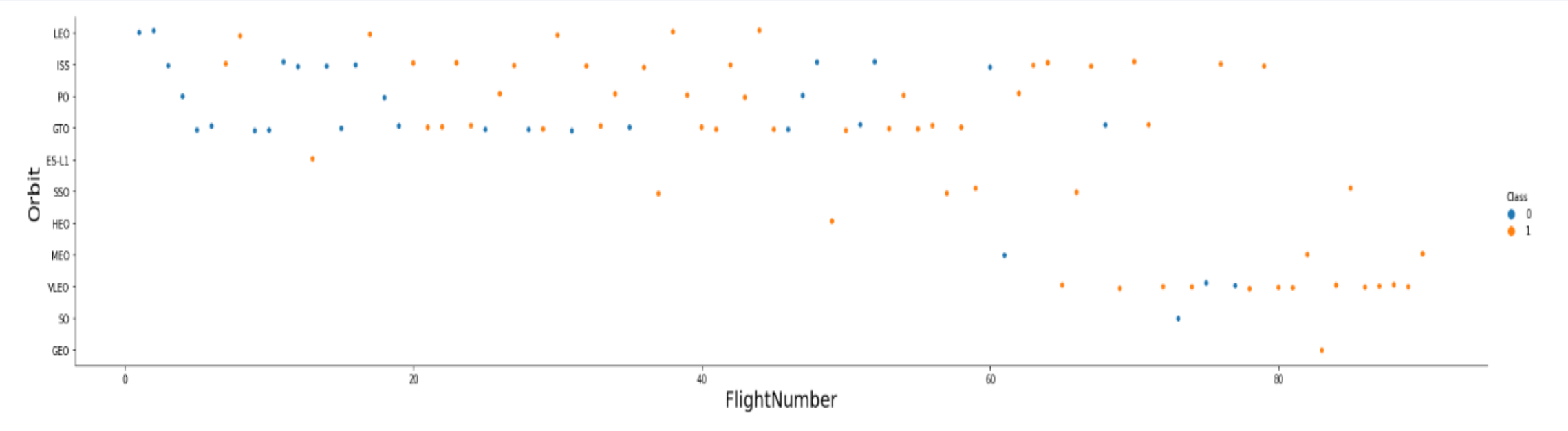
Success Rate vs. Orbit Type

From the graph, we can see that ES L1, GEO, HEO, SSO, VLEO had the most success rate..



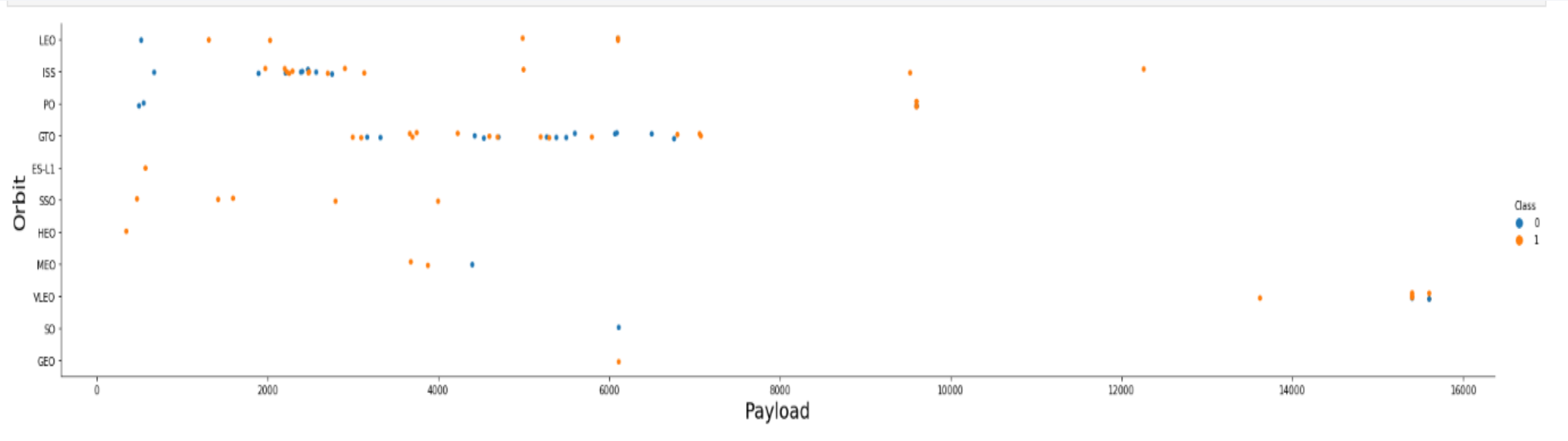
Flight Number vs. Orbit Type

We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



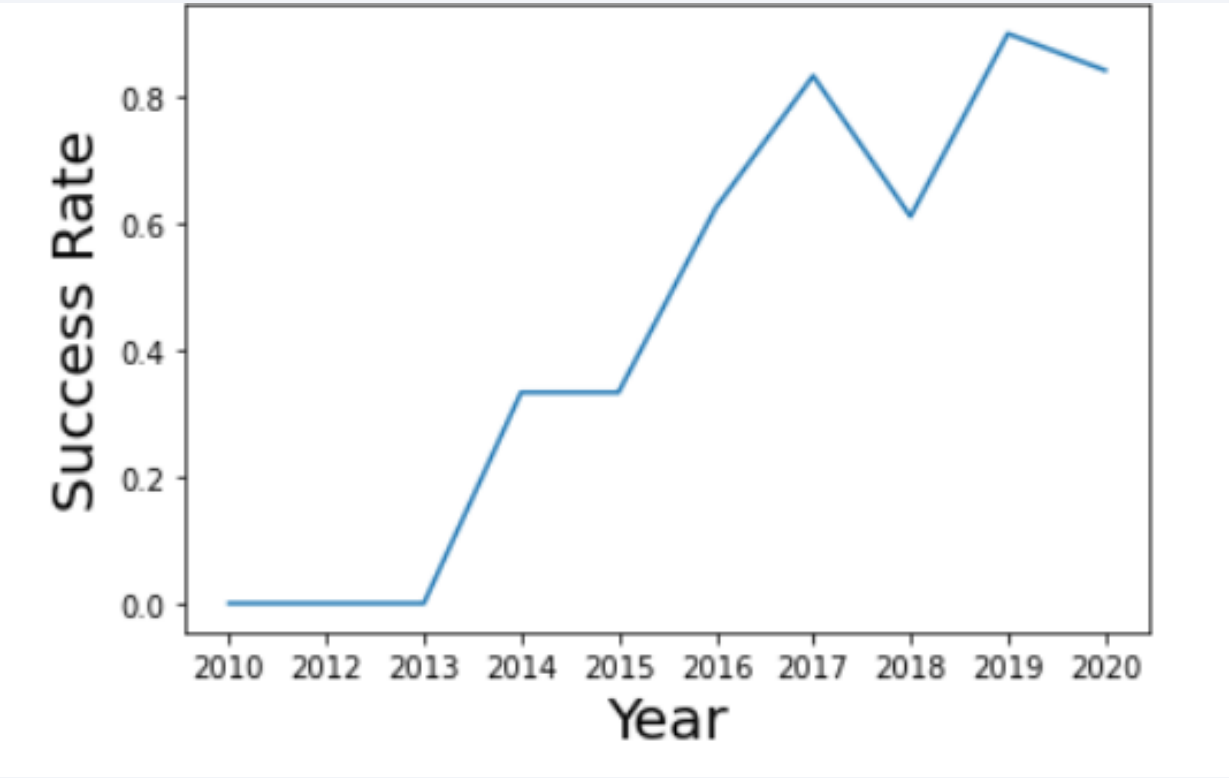
Payload vs. Orbit Type

Heavy payloads have a negative influence on GTO orbits and positive impact on GTO and Polar LEO (ISS) orbits.



Launch Success Yearly Trend

It can be observed that the success rate has been increasing since 2013



All Launch Site Names

The key word DISTINCT in the query is used to fetch only unique launch sites from the SpaceX data.

In [6]:
```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL;
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1oc
Done.
```

Out[6]:

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

Launch Site Names Begin with 'CCA'

- LIMIT keyword in the query ensures that the top 5 rows are fetched

In [7]:

```
%%sql
SELECT LAUNCH_SITE
FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE 'CCA%'
LIMIT 5;
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9
Done.
```

Out[7]:

launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- SUM keyword in the query is used to calculate the total payload mass. Result is 45596

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Customer = 'NASA (CRS)';
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f
Done.
```

1
45596

Average Payload Mass by F9 v1.1

- AVG keyword in the query calculates the average payload mass. Result is 2928

```
In [17]:
%%sql
SELECT AVG(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Booster_Version = 'F9 v1.1';
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l
Done.
```

Out[17]:

1
2928

First Successful Ground Landing Date

MIN keyword is used to get the first successful ground landing date. Result 22/12/2015

```
%%sql
SELECT MIN(Date)
FROM SPACEXTBL
WHERE Landing__Outcome = 'Success (ground pad)';
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs
Done.
```

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000

32]:
```
%%sql
SELECT Booster_Version
FROM SPACEXTBL
WHERE Landing__Outcome = 'Success (drone ship)'
AND PAYLOAD_MASS__KG_ > 4000
AND  PAYLOAD_MASS__KG_ <6000;
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8lcg.dat
Done.
```

32]:

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

Total Number of Successful and Failure Mission Outcomes

- We found the count of Mission outcomes and grouped the values by Mission outcome

[35]:
```
%%sql
SELECT Mission_Outcome, COUNT(Mission_Outcome) AS TOTAL
FROM SPACEXTBL
GROUP BY Mission_Outcome;
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08k
Done.
```

[35]:

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

Boosters Carried Maximum Payload

We used a subquery to fetch the max payload mass and then fetched Booster versions corresponding to that

```
36]:
%%sql
SELECT Booster_Version
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8lcg.da
Done.
```

36]:

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

We used the “Where” clause to fetch records that had Landing outcome = Failed and year 2015

```
%%sql
SELECT Booster_Version, Launch_Site, Landing__Outcome
FROM SPACEXTBL
WHERE Landing__Outcome = 'Failure (drone ship)'
AND YEAR(DATE) = 2015
```

```
 * ibm_db_sa://lrz28718:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io
Done.
```

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

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

We used the “BETWEEN” keyword to get the records between 4/6/2010 and 20/3/2017 and then grouped the output by Landing outcome and ordered them based on total count

```
]: %%sql
   SELECT Landing__Outcome, COUNT(Landing__Outcome) as Total
   from SPACEXTBL
   WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
   GROUP BY Landing__Outcome
   Order by total desc
```

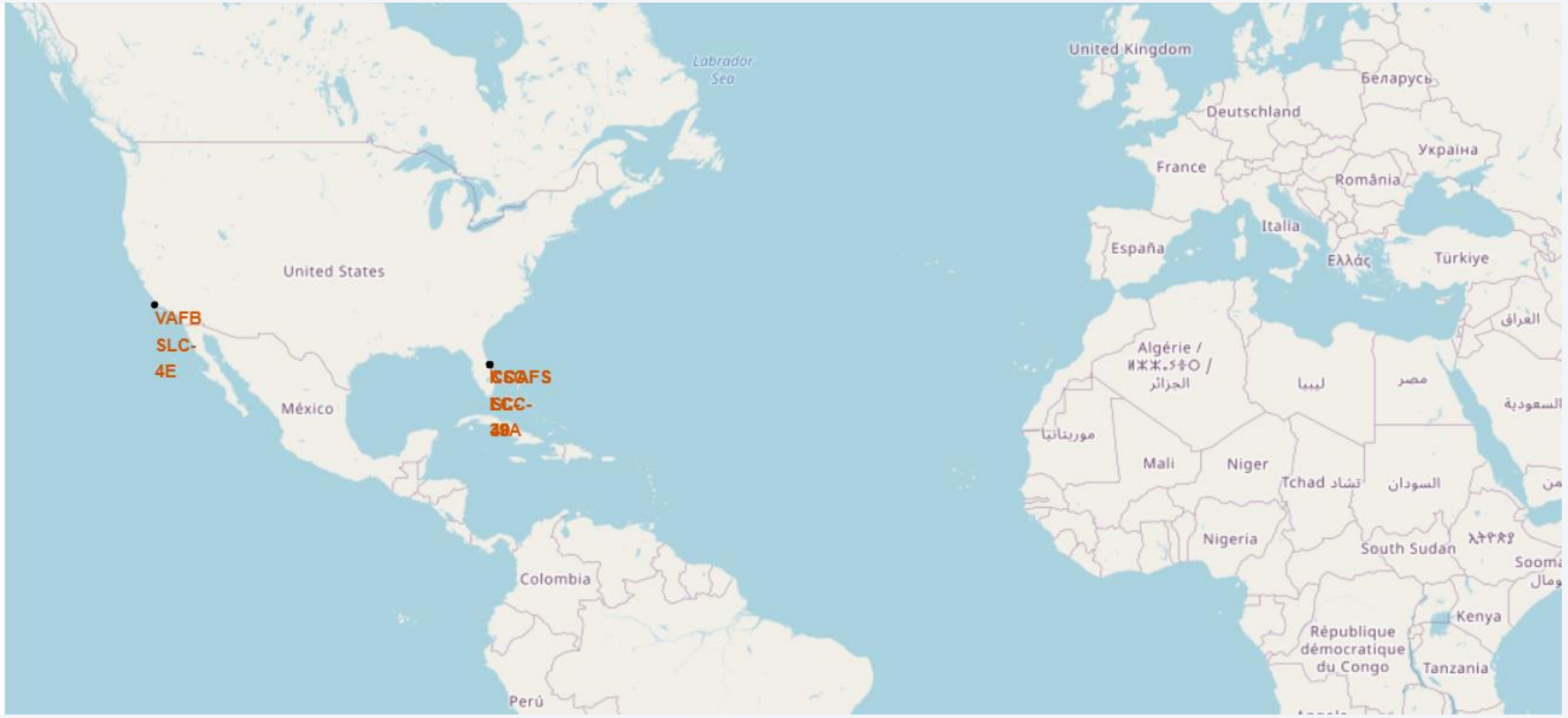
landing_outcome	total
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

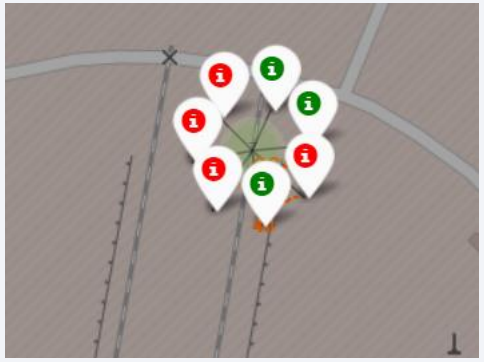
All launch sites in global map

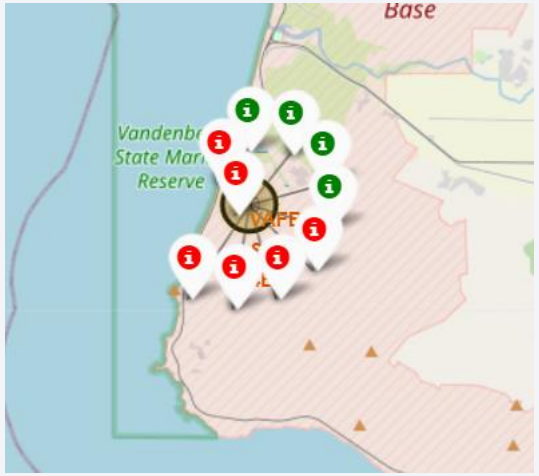
All launch sites are in the United states of America

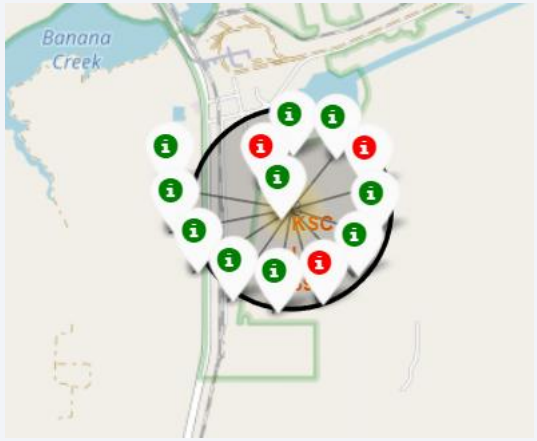


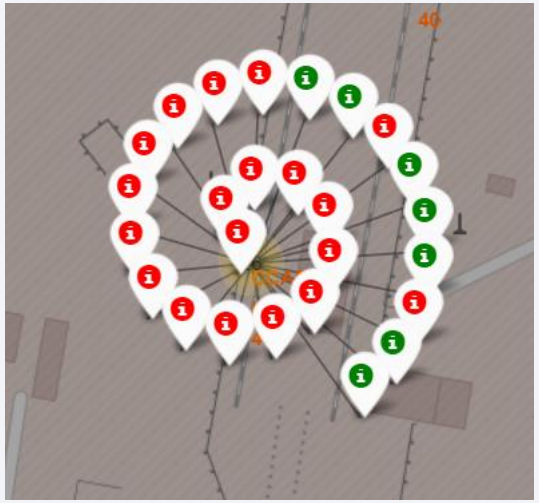
Success/Failed launches for each site shown in a map

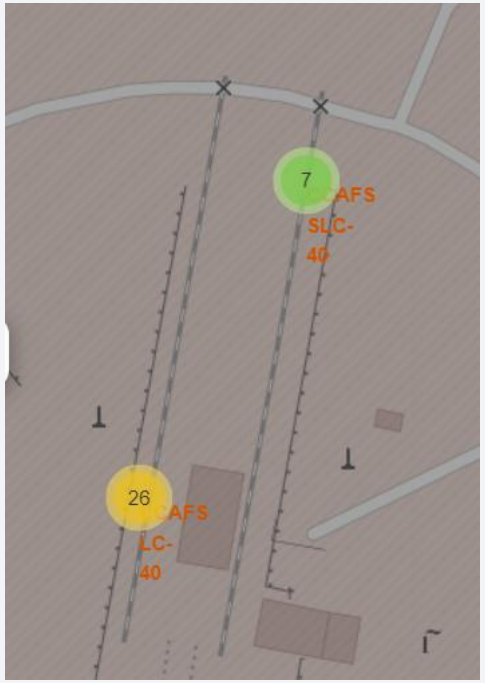
Successful launches are represented in Green. Failed launches are represented in Red.





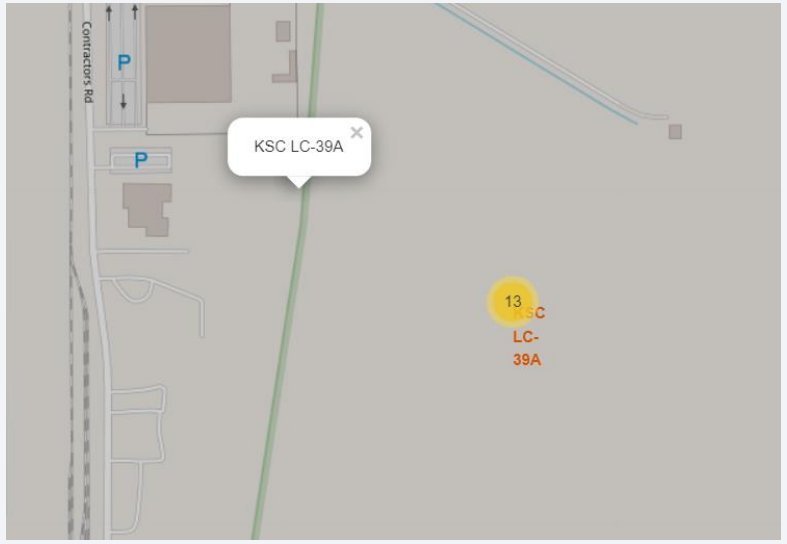


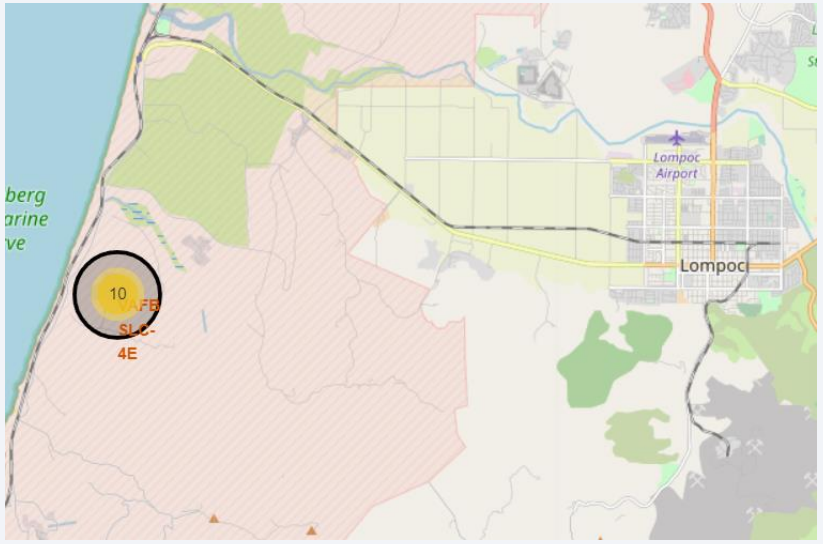


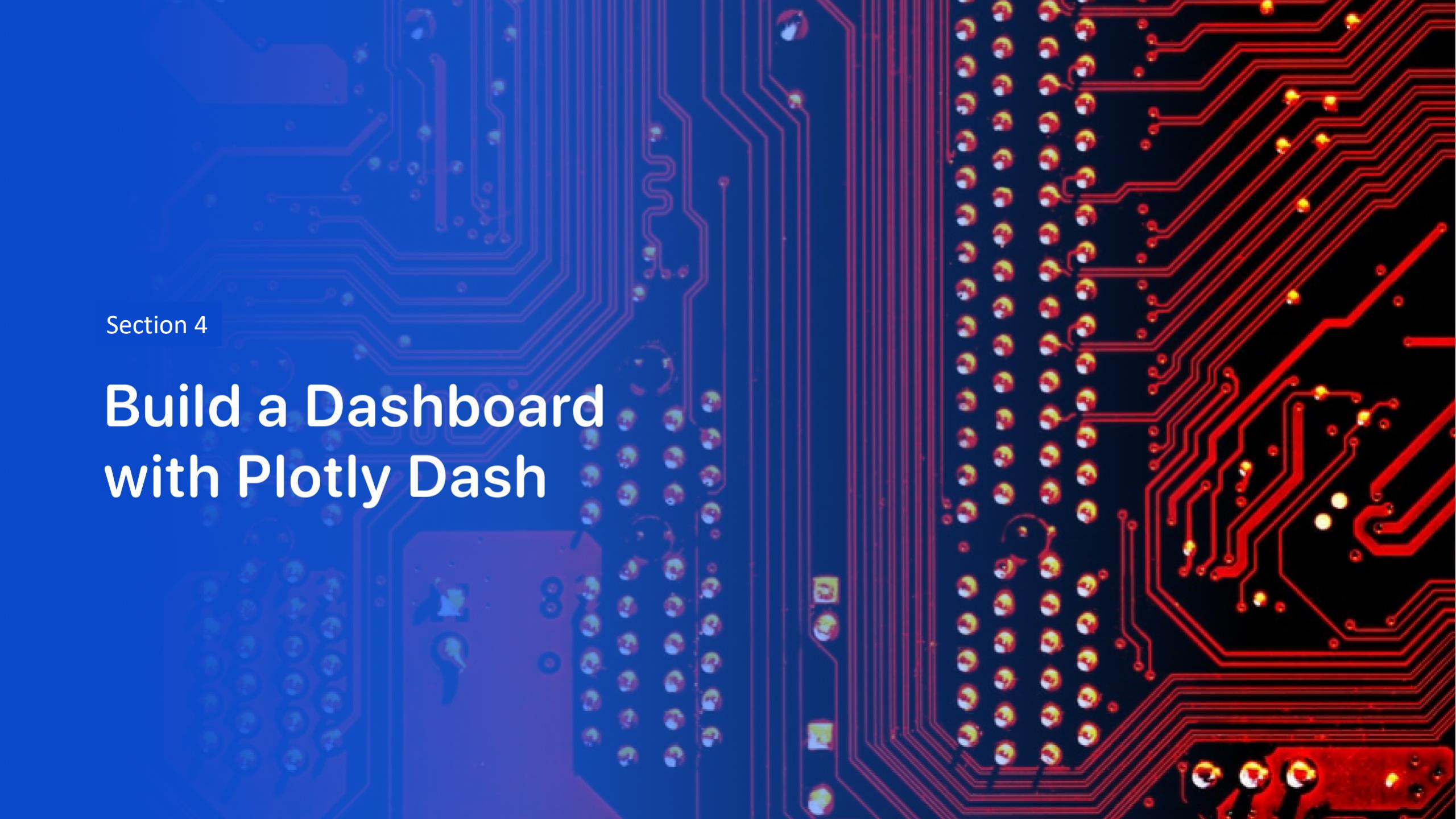


Launch site proximity to landmarks

- 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

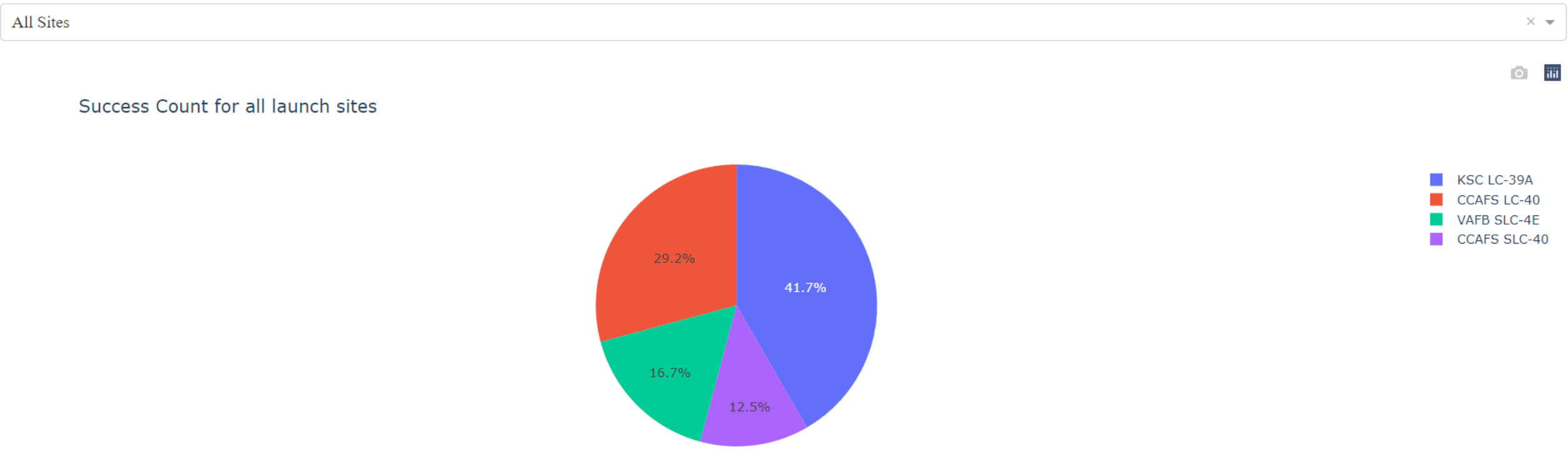




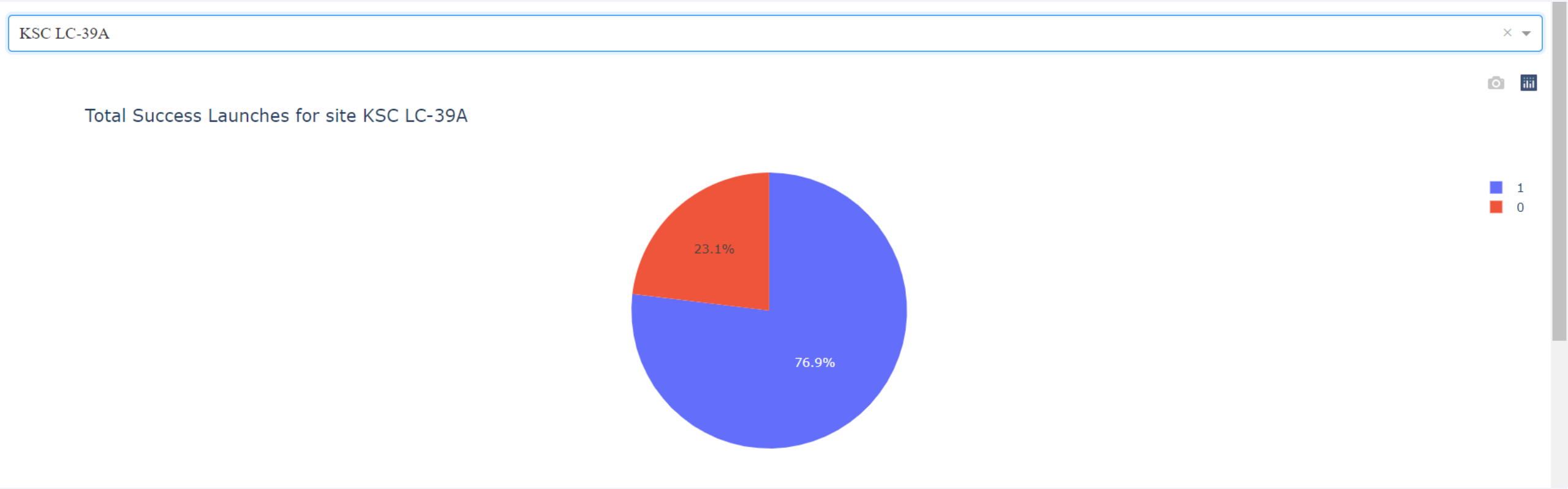

Section 4
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site

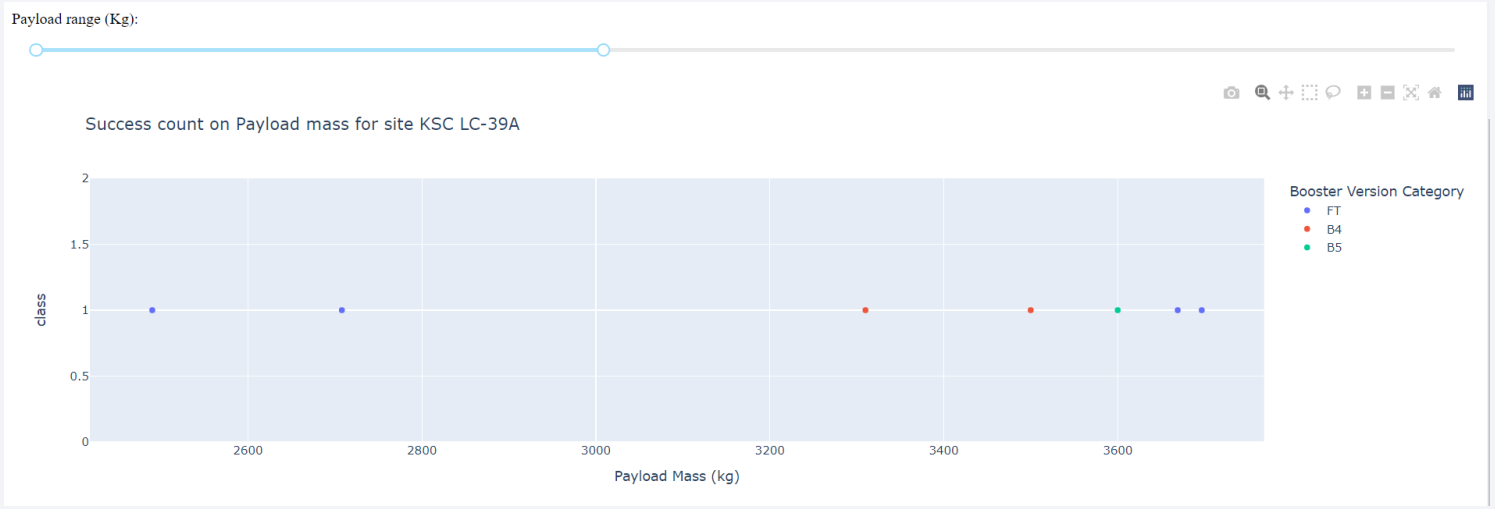
KSC LC-39 A has the highest success percentage followed by CCAFS LC-40



Piechart for the launch site with highest launch success ratio



Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

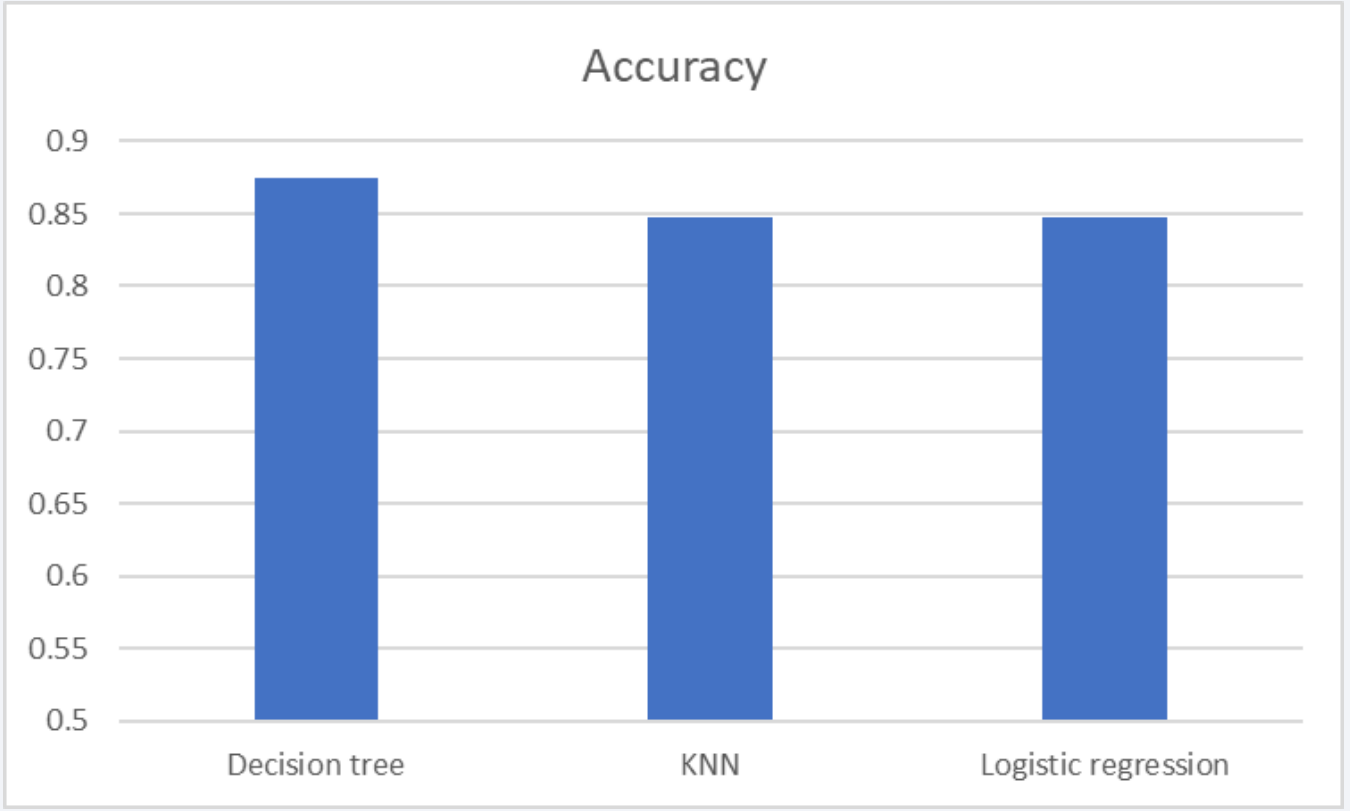


Section 5

Predictive Analysis (Classification)

Classification Accuracy

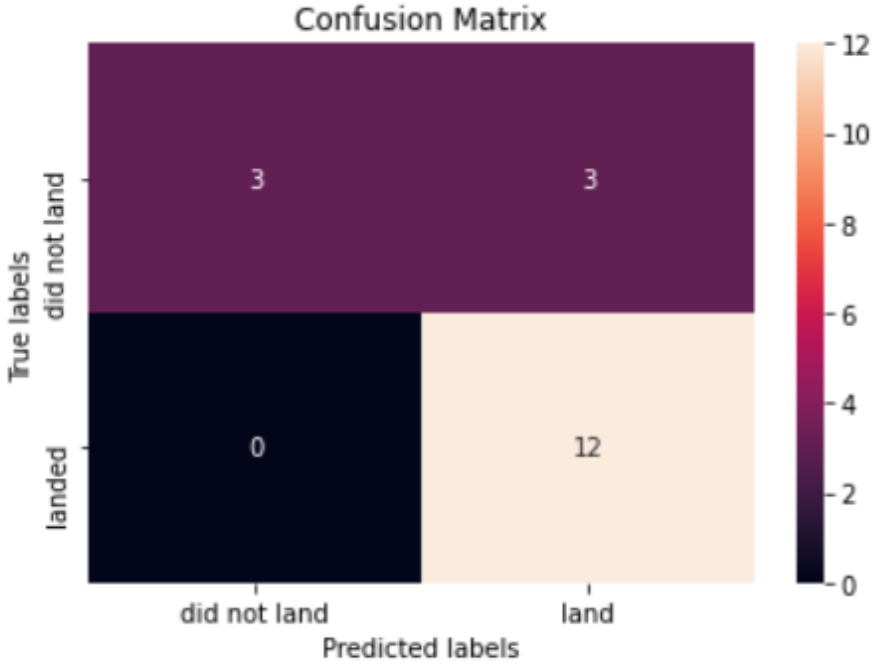
- Almost all models have similar accuracy.
- Decision tree has a slightly better accuracy than the rest



Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.

```
yhat = svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- 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 machine learning algorithm for this task.

Thank you!