IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Jacinto Diaz
October 09 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 Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics result
  - Predictive Analysis result

Introduction

- Project background and context
  - 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.

- Problems you want to find answers
  - Predict if the first stage if of the SpaceX Flacon 9 rocket will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API using api.spacexdata.com/v4/.
  - Web scrapping from Wikipedia
- Perform data wrangling
  - One Hot Encoding data fields for Machine Learning and data cleaning of null values and irrelevant column.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models.
  - LR, KNN, SVM, DT models have been built and evaluated for the best classifier

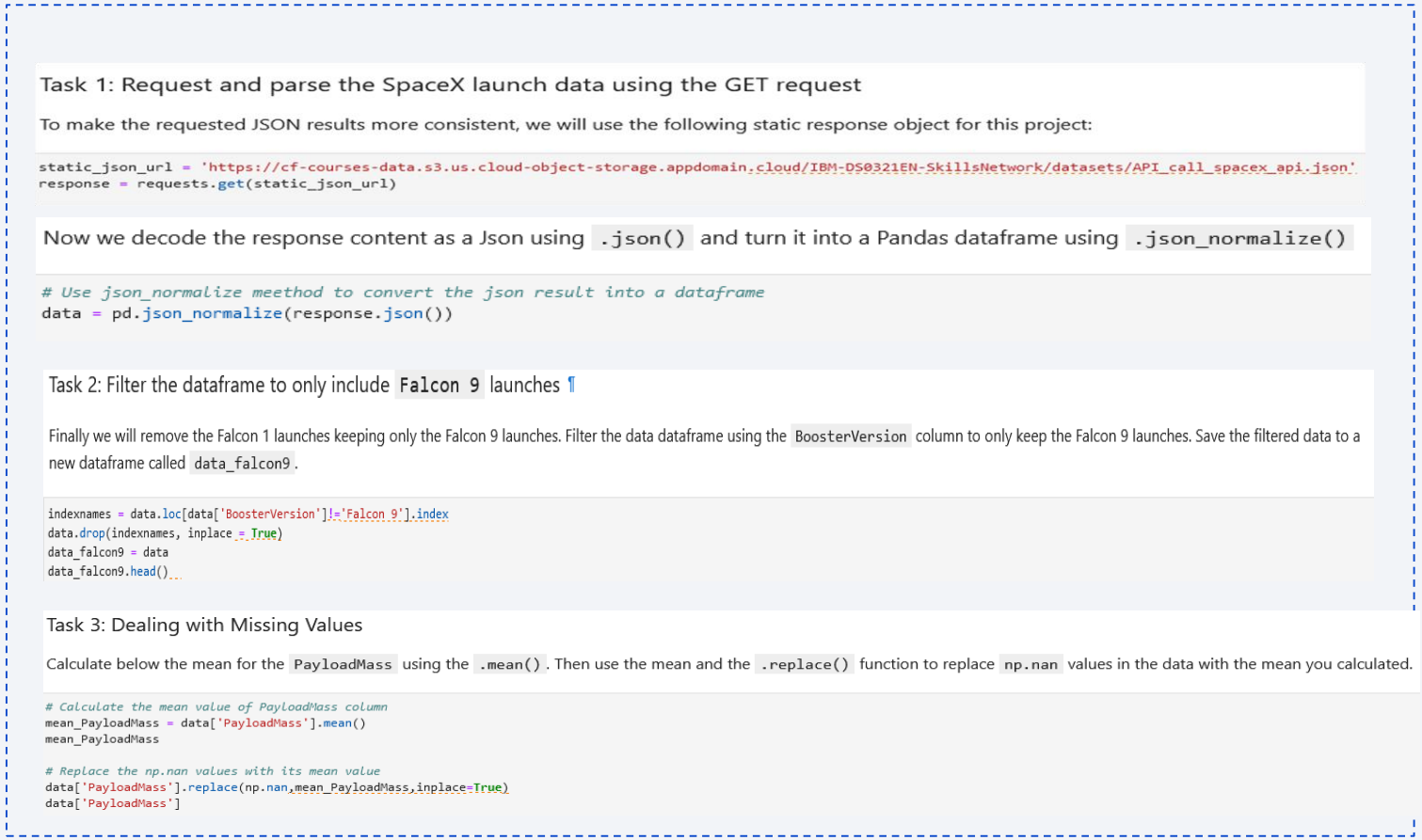
Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas data frame using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas data frame for future analysis.

Data Collection – SpaceX API

- We sent a get request to the SpaceX API, converted it into a Pandas dataframe. Filtered the data for Falcon9 launches and handled the missing values.
- GitHub URL: GitHub URL https://github.com/jacint61/data-project/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

..



Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
static_json_url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
response = requests.get(static_json_url)
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

Task 2: Filter the dataframe to only include `Falcon 9` launches

Finally we will remove the Falcon 1 launches keeping only the Falcon 9 launches. Filter the data dataframe using the `BoosterVersion` column to only keep the Falcon 9 launches. Save the filtered data to a new dataframe called `data_falcon9`.

```
indexnames = data.loc[data['BoosterVersion']!='Falcon 9'].index
data.drop(indexnames, inplace = True)
data_falcon9 = data
data_falcon9.head()
```

Task 3: Dealing with Missing Values

Calculate below the mean for the `PayloadMass` using the `.mean()`. Then use the mean and the `.replace()` function to replace `np.nan` values in the data with the mean you calculated.

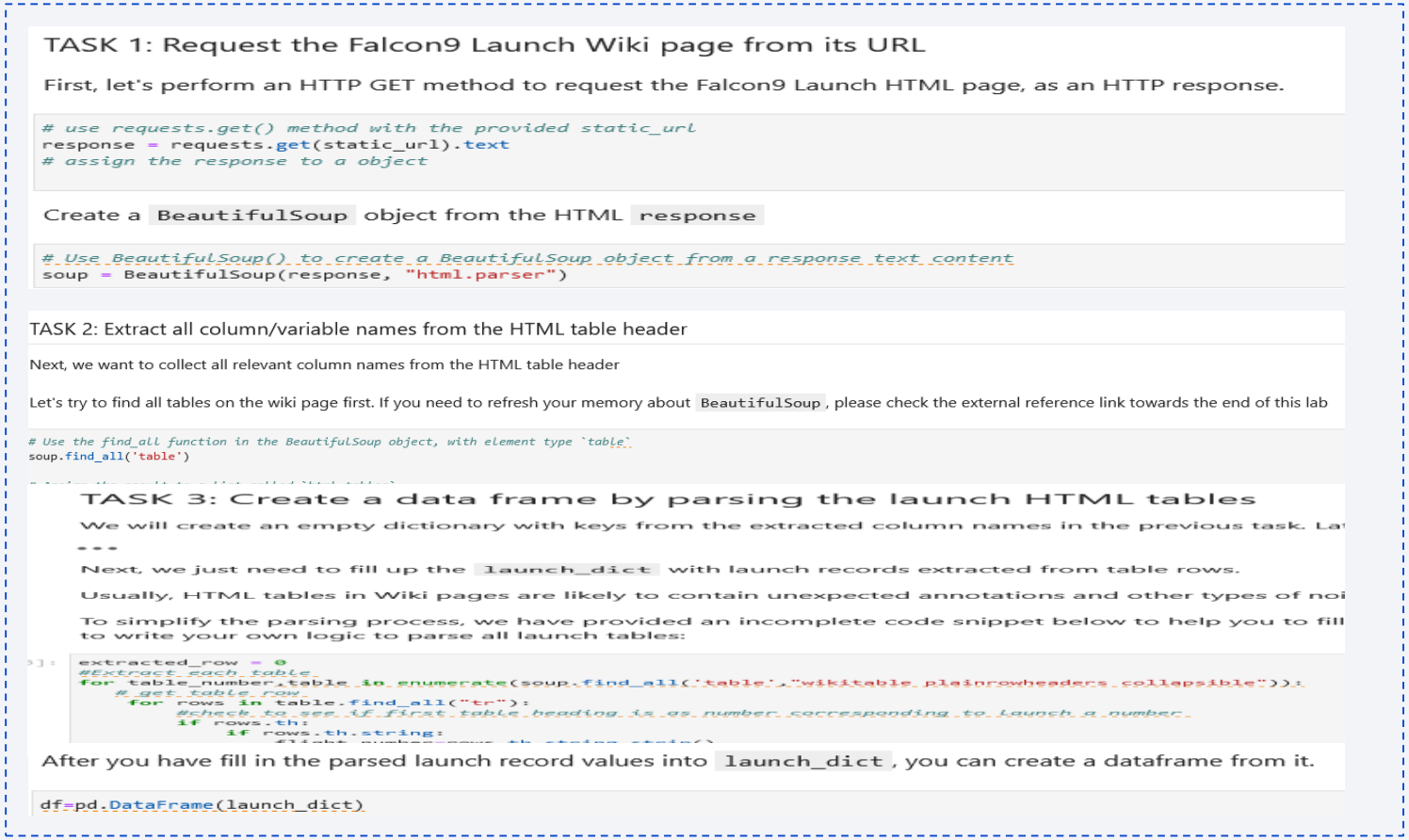
```
# Calculate the mean value of PayloadMass column
mean_PayloadMass = data['PayloadMass'].mean()
mean_PayloadMass

# Replace the np.nan values with its mean value
data['PayloadMass'].replace(np.nan,mean_PayloadMass,inplace=True)
data['PayloadMass']
```

Data Collection - Scraping

- We used BeautifulSoup to web scrap, extracted the column names and converted the data into a Pandas Dataframe.
- GitHub URL of the web scraping notebook:

https://github.com/jacint61/data-project/blob/main/Lab%202%20-%20jupyter-labs-webscraping.ipynb



TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

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

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

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

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
soup.find_all('table')
```

TASK 3: Create a data frame by parsing the launch HTML tables

We will create an empty dictionary with keys from the extracted column names in the previous task. La

---

Next, we just need to fill up the `launch_dict` with launch records extracted from table rows.

Usually, HTML tables in Wiki pages are likely to contain unexpected annotations and other types of noi

To simplify the parsing process, we have provided an incomplete code snippet below to help you to fill to write your own logic to parse all launch tables:

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
```

After you have fill in the parsed launch record values into `launch_dict`, you can create a dataframe from it.

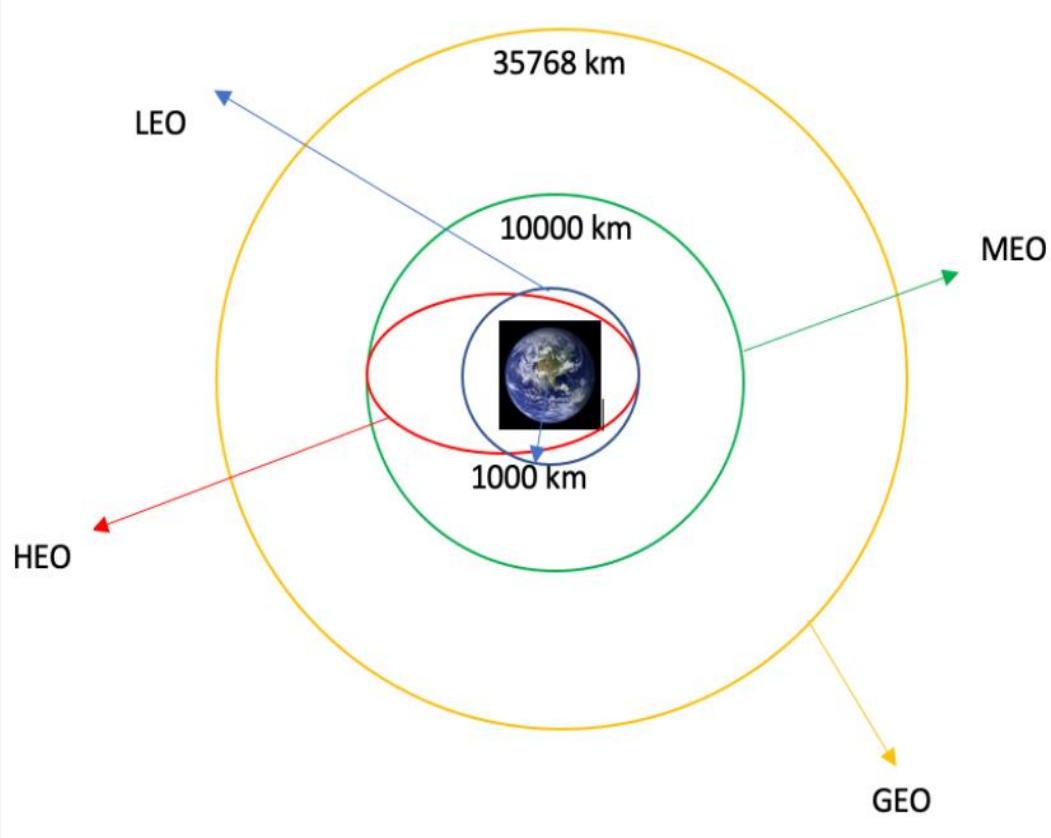
```
df=pd.DataFrame(launch_dict)
```

Data Wrangling

- Use the method value_counts() on the columns LaunchSite, Orbit and Outcomes to determine the occurrences of each.
- Created a bad outcome set to separate bad outcomes from the successful ones.
- Created “Class” column to show the outcome.

  We provide Class value =0 for bad outcomes.

  We provide Class value =1 for successful outcomes.
- GitHub URL : https://github.com/jacint61/data-project/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- We plotted Scatter Plots for the following :

  Flight Number v/s Launch Sites

  Payload v/s Launch Sites

  Flight Number v/s Orbits

  Payload v/s Orbits

We used scatter plots to see the relation between these two features for each.

- We plotted Bar Chart for Orbits to show the success rate of each.
- We plotted Line Chart to see how average success ratio changed with years.
- GitHub URL : https://github.com/jacint61/data-project/blob/main/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- We performed EDA with SQL queries to get insights from data. The queries were:

  Display names of each unique launch site in the space mission.

  Display 5 records where launch sites begin with the string 'CCA'

  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 when the first successful landing outcome in ground pad was acheived

  List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

  List the total number of successful and failed outcomes

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

  List the records which will display the month names, failure landing outcomes in drone ship ,booster versions, launch site for the months in year 2015.

  Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order

GitHub URL : https://github.com/jacint61/data-project/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We marked all launch sites and added map markers to mark the success and failure of each launch from the different sites.
- We assigned class label 0 and 1 for failure and success outcome respectively.
- Calculated distances between the launch sites and its proximities like coastline, highway and city.

  To see if launch sites keep away from cities?

  To see if they are close to coastlines?

  How are they connected with highways and railways?
- GitHub URL :

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly Dash.
- A dropdown menu with options to either select all or specific launch sites
- A pie chart showing total successful launches for all sites or to show a pie chart between successful and failure ratio from specific sites.
- We added a payload range slider to select a range of payload mass.
- We plotted a scatter plot between outcome and payload mass picked from slider.
- GitHub URL:

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas. Transformed the data using StandardScaler().
- We split the data into train and test data with test size 0.2
- We built different ML models and found best hyperparameters using GrisdSearchCV.
- We found the best performing model.
- GitHub URL :

```
transform = preprocessing.StandardScaler()
X=transform.fit_transform(X)
```

↓

```
X_train, X_test, Y_train, Y_test=train_test_split(X,Y,test_size=0.2,random_state=2)
```

↓

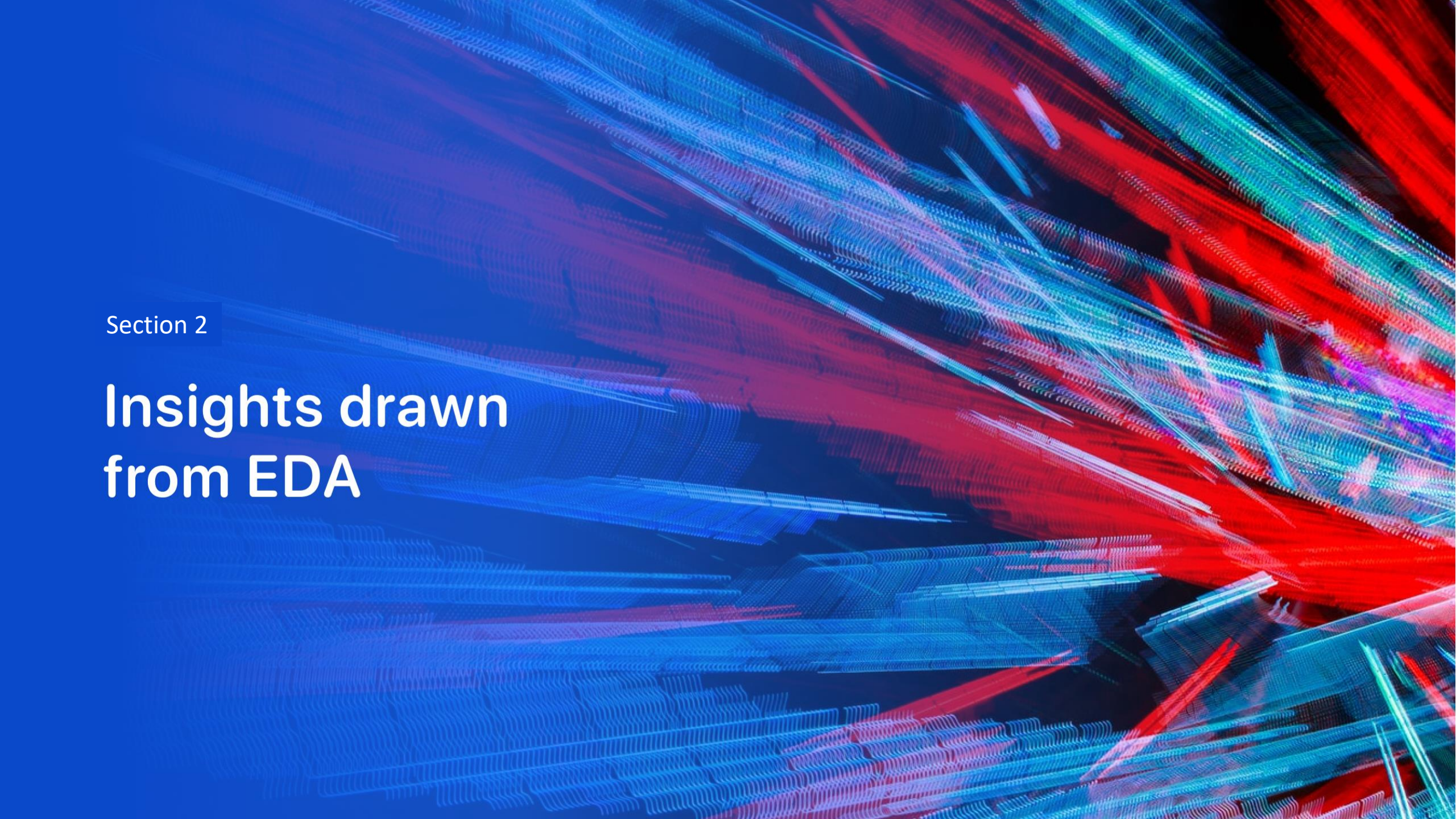
```
#Building a model
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}
tree = DecisionTreeClassifier()
tree_cv_search=GridSearchCV(estimator=tree, param_grid = parameters, scoring = 'accuracy', cv = 10)
tree_cv=tree_cv_search.fit(X,Y)
tree_cv.score(X_test, Y_test)
```

Results

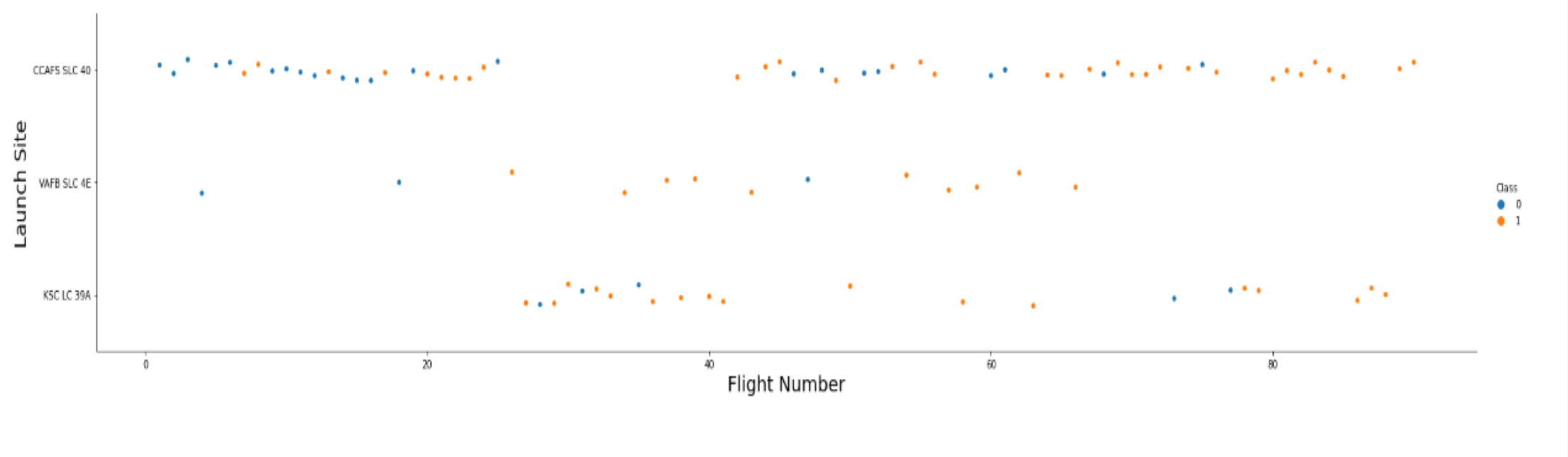
- Exploratory data analysis results
  - Space X uses 4 different launch sites;
  - The first launches were done to Space X itself and NASA;
  - The average payload of F9 v1.1 booster is 2,928 kg;
  - The first success landing outcome happened in 2015 fiver year after the first launch;
  - Many Falcon 9 booster versions were successful at landing in drone ships having payload above the average;
  - Almost 100% of mission outcomes were successful;
  - Two booster versions failed at landing in drone ships in 2015: F9 v1.1 B1012 and F9 v1.1 B1015;
  - The number of landing outcomes became as better as years passed.
- Interactive analytics demo in screenshots
- Predictive analysis results
  - Predictive analysis showed that Decision Tree Classifier is the best classification model for our data with accuracy of 90%.
  - Only 1 case each of False Positive and False Negative.

Results

- Interactive analytics demo in screenshots
  - Using interactive analytics was possible to identify that launch sites use to be in safety places, near sea, for example and have a good logistic infrastructure around.
  - Most launches happen at east coast line.

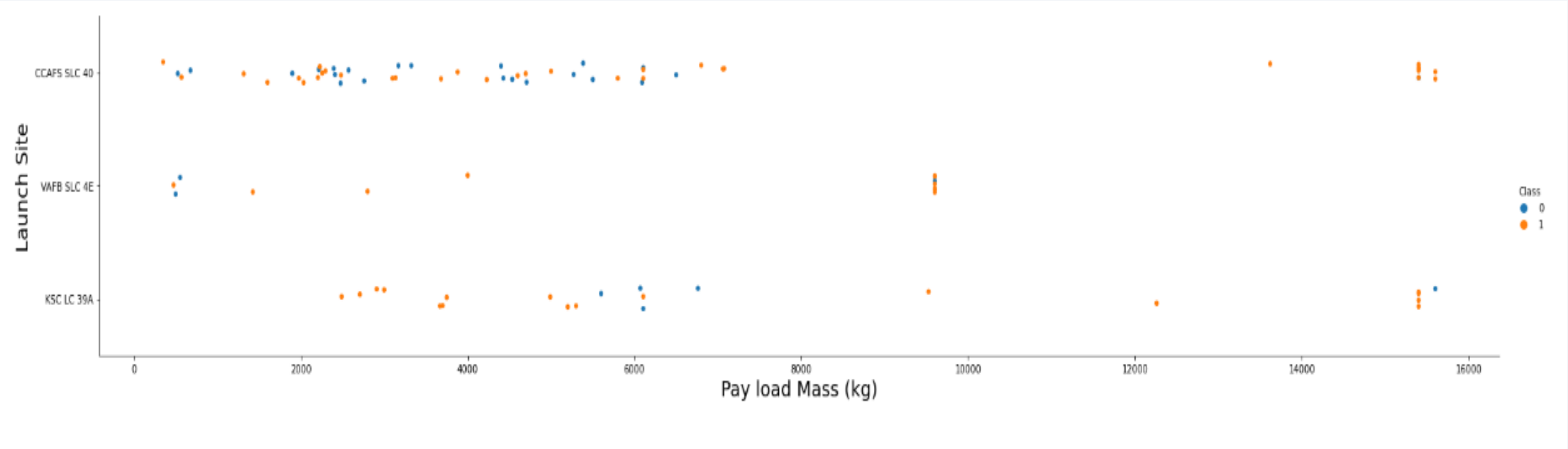

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



The larger the flight amount at a launch site, the greater the success rate at a launch site.

Payload vs. Launch Site

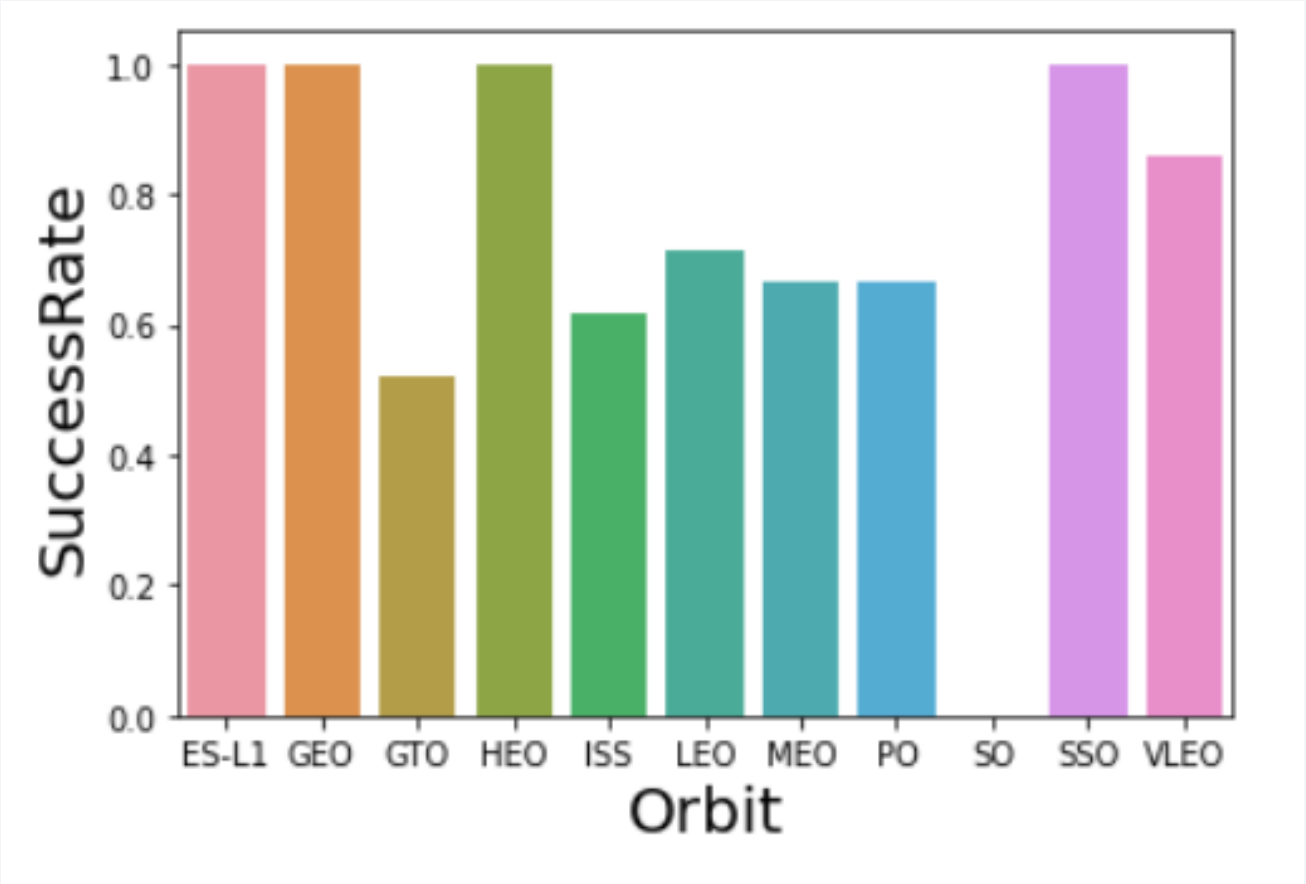


We see that with high payload mass we have more points with data label 1.

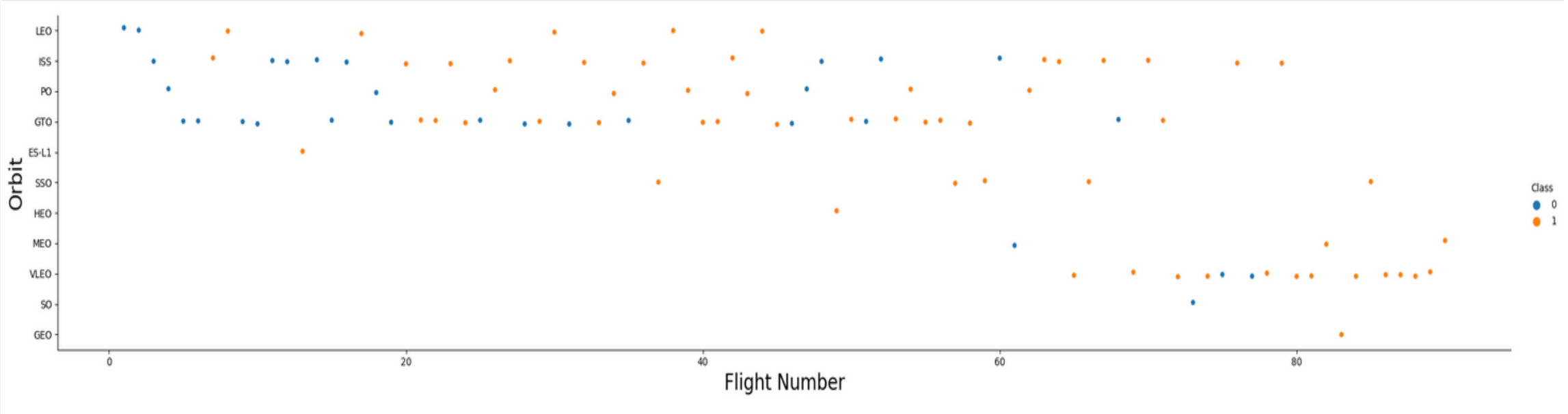
This shows that launches with high payload mass have more chances for successful outcome than a low payload mass launch.

Success Rate vs. Orbit Type

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

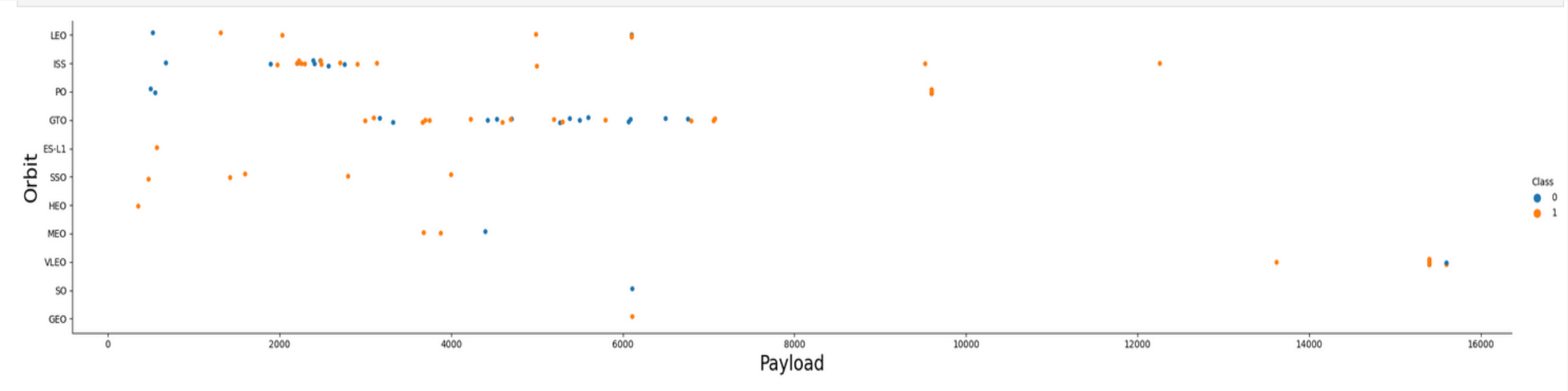


Flight Number vs. Orbit Type



- The plot below shows the 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



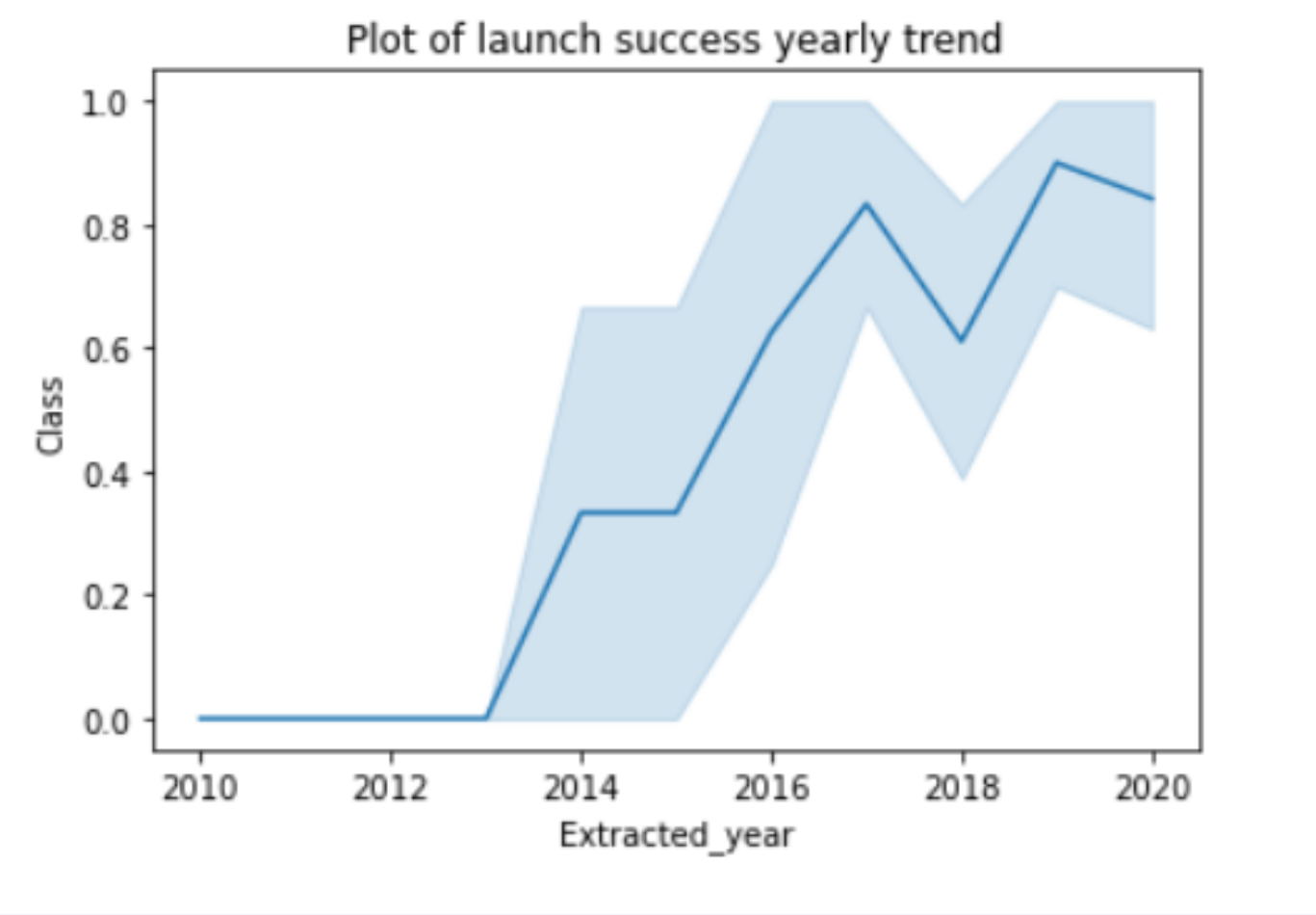
With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.

However for GTO we cannot distinguish this very well as both of the positive landing rate and negative landing(unsuccessful mission) are there.

Launch Success Yearly Trend

We see that success rate has been on a rise since year 2013.

However, there was a downfall for success rate in the year 2018.



All Launch Site Names

```
%sql SELECT DISTINCT(Launch_Site) FROM  SPACEXTBL
```

```
 * sqlite:///my_data1.db
```

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

We select DISTINCT launch sites from the table.

We get these 4 outcomes as 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

We display first 5 record where the Launch Sites begin with CAA using LIKE operation.

Total Payload Mass

```
%sql select customer, sum(payload_mass__kg_)as total_payload_mass from SPACEXTBL Where  customer = "NASA (CRS)"
```

Customer	total_payload_mass
NASA (CRS)	45596

We calculate the total payload mass (kg) carried by boosters launched by NASA and get the value of 45 596 KG.

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

We calculate the average payload mass (kg) carried by booster version F9 v1.1 and get the value of 2928 KG.

First Successful Ground Landing Date

```
%sql SELECT MIN(Date) FROM SPACEXTBL where "Landing _Outcome" = "Success (ground pad)"
```

MIN(Date)
01-05-2017

We calculate the earliest date for a successful landing on a ground pad and get the date as 2017-05-01.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT BOOSTER_VERSION, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

We list the booster versions which have a successful landing on a drone ship and the Payload is between 4000-6000 KG.

Total Number of Successful and Failure Mission Outcomes

```
%%sql
SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) FROM SPACEXTBL GROUP BY MISSION_OUTCOME;
```

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

We calculate total successful and failure mission outcomes and get 100 Successful mission and 1 Failure mission.

We do so by using GROUP BY operation in our query.

Boosters Carried Maximum Payload

```
%%sql
SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_=(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

We display the boosters that carried maximum payload weight and get these values.

We do so by using a SUB QUERY in our query and use MAX operation in that sub query.

2015 Launch Records

```
%%sql
SELECT BOOSTER_VERSION, LAUNCH_SITE, LANDING_OUTCOME FROM SPACEXTBL WHERE LANDING_OUTCOME LIKE 'Failure (drone ship)' AND Date B
ETWEEN '2015-01-01' AND '2015-12-31';
```

We display booster version, launch site and the landing outcome for failed landing outcome in 2015. We get the result as shown.

We do so by using BETWEEN in our query.

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;
```

landing_outcome	2
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

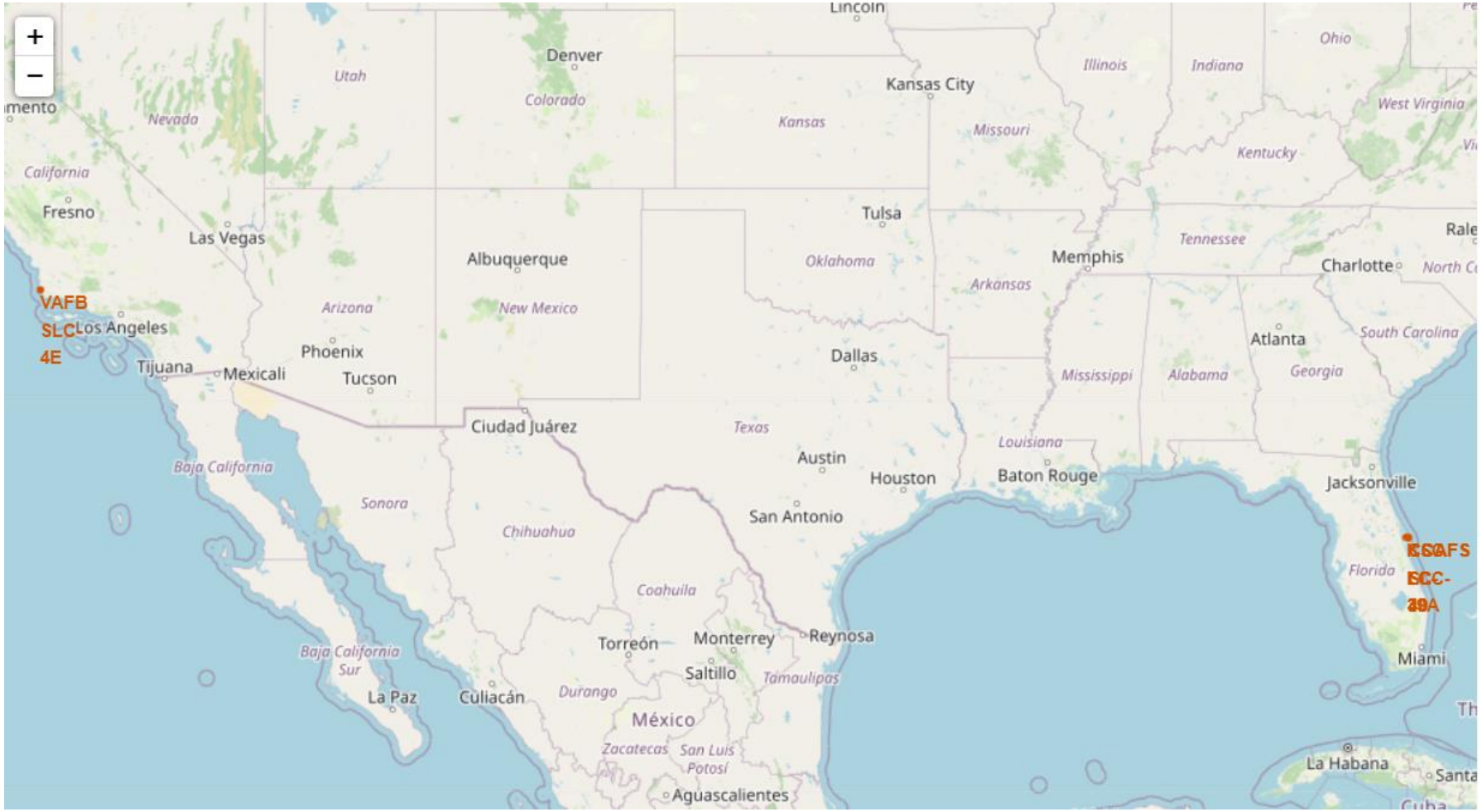
We landing outcome, and their occurrences between specified dates. We get this output as shown.

We do so by using COUNT BETWEEN ORDER BY & GROUP BY in our query.

Section 3

Launch Sites Proximities Analysis

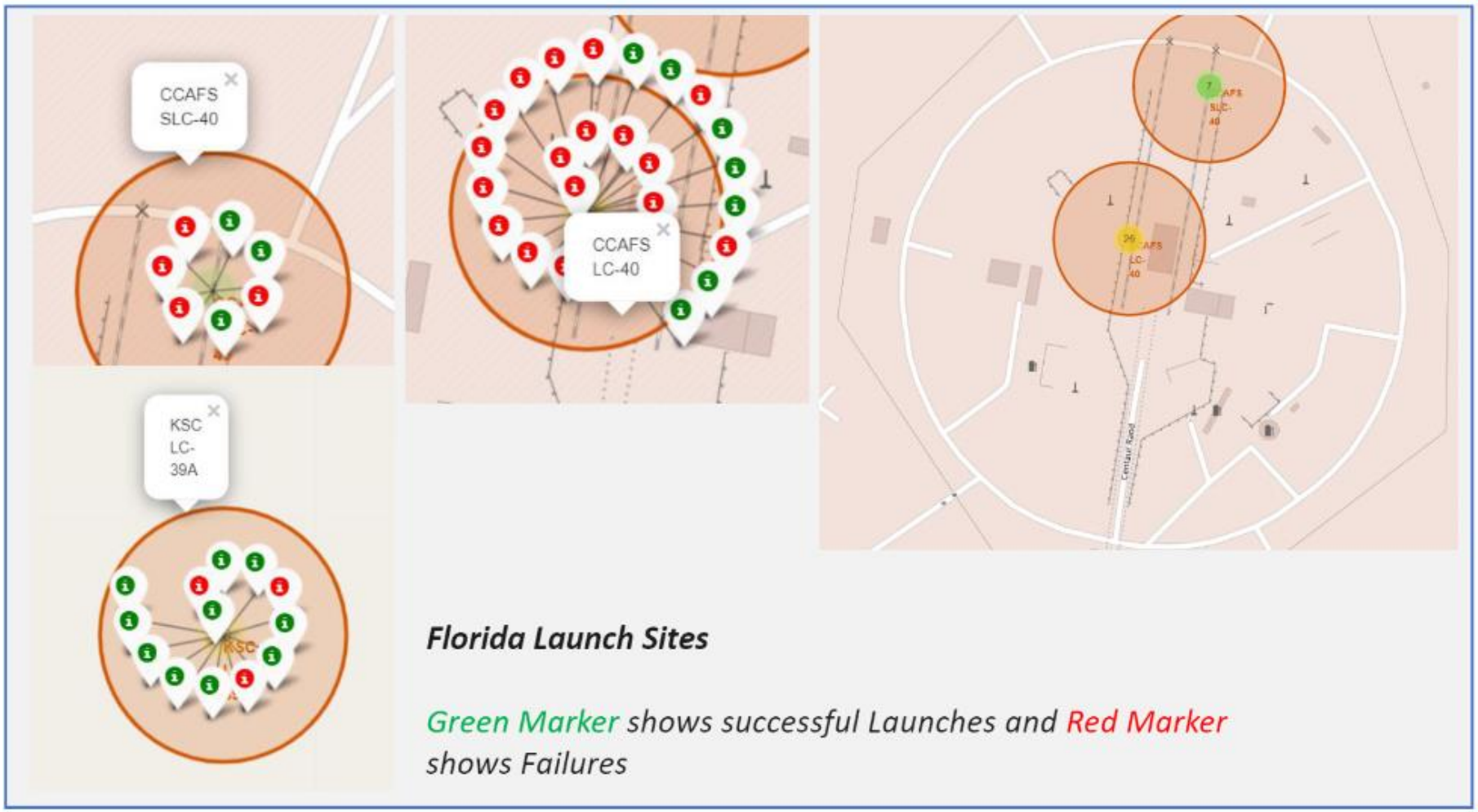
Launch Sites on Map of USA.



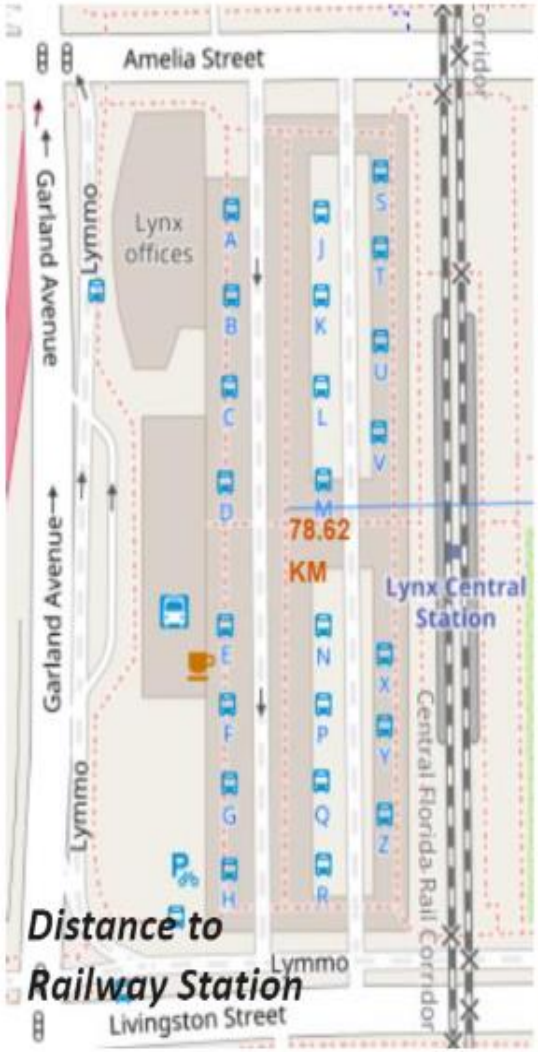
We see that the SpaceX's Launch sites are in United States of America.

All launch sites are in coastal region.

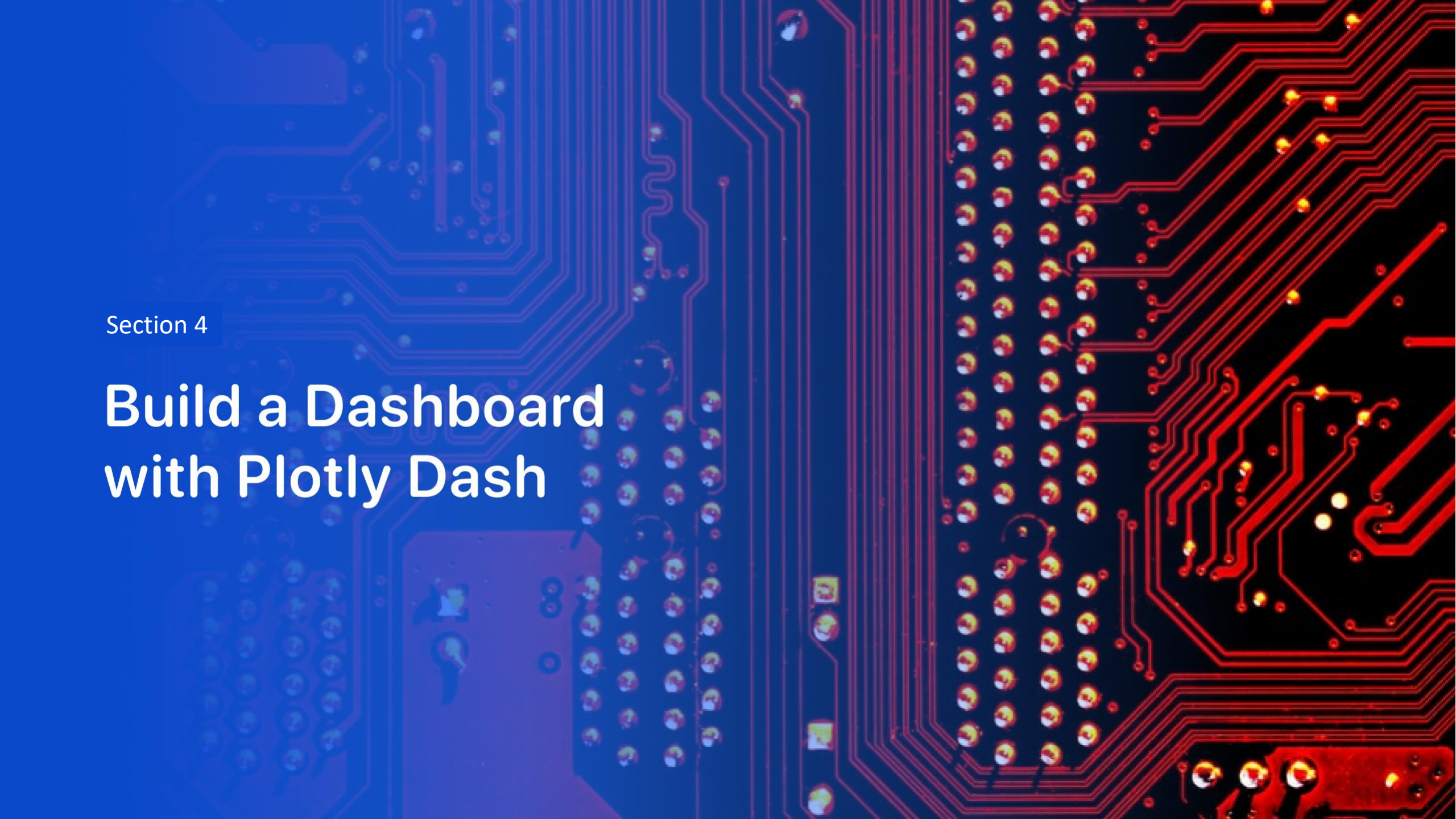
Markers showing launch sites with color labels



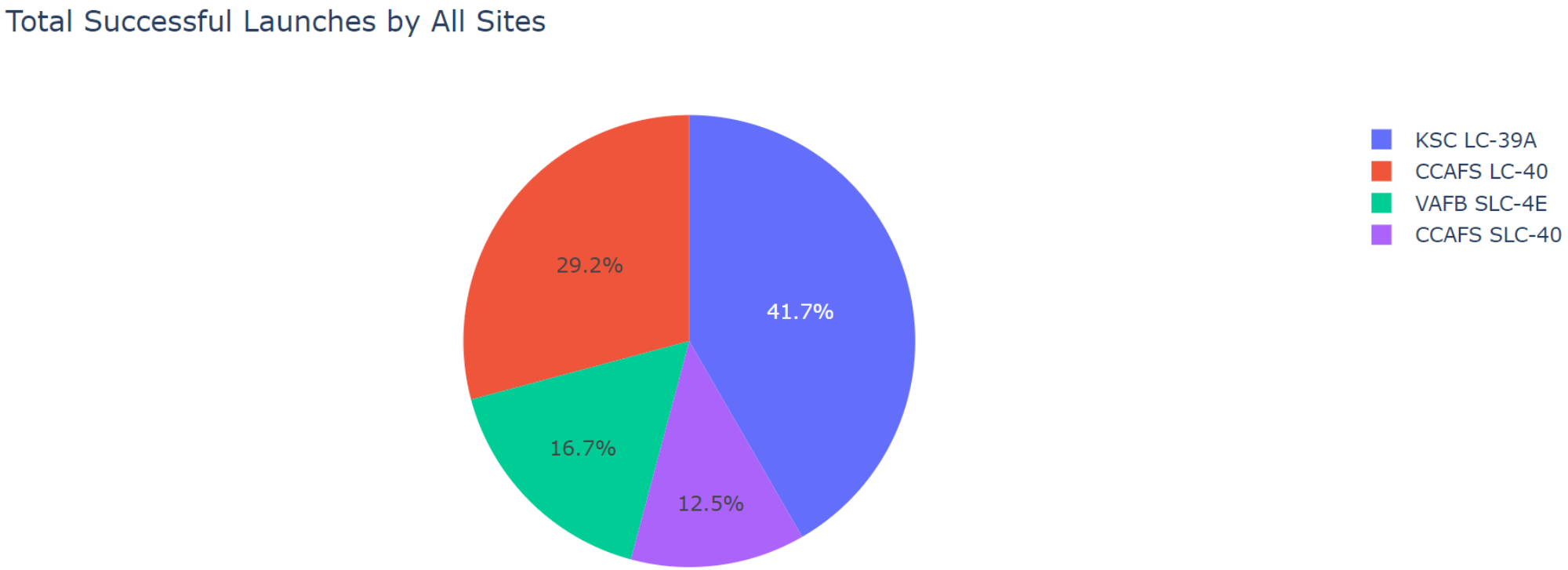
Launch Site distance to landmarks



- 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

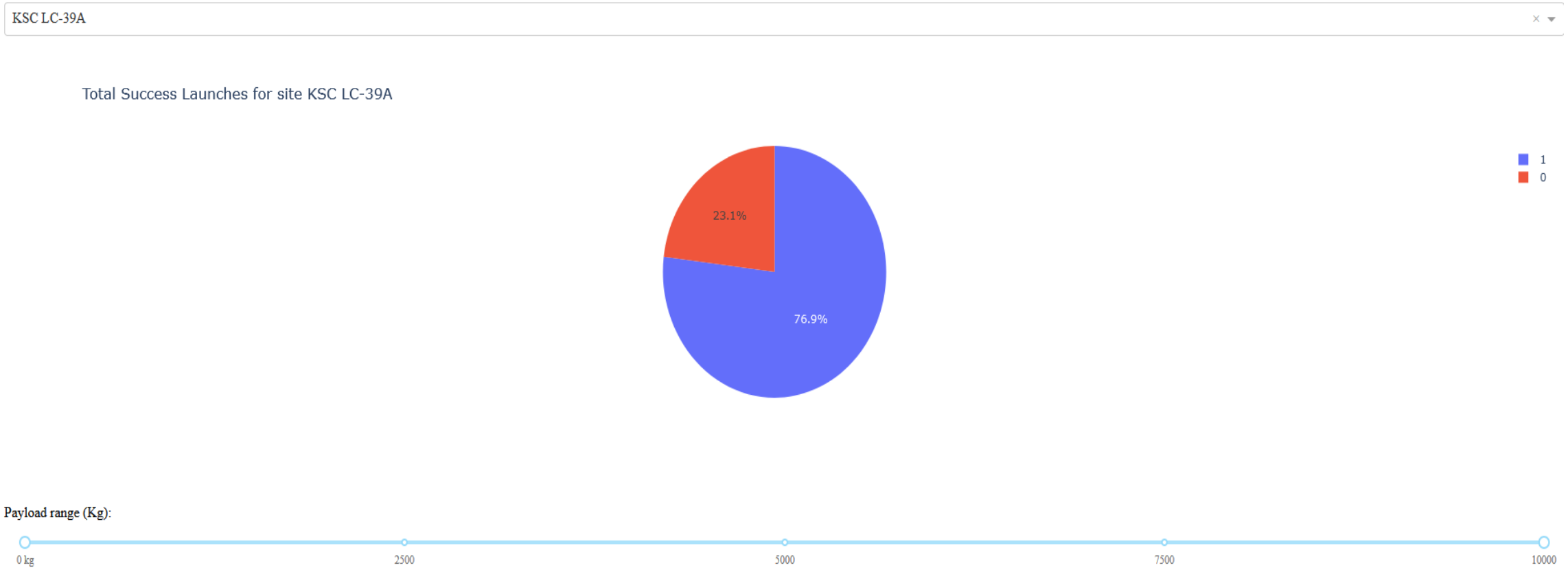

Section 4
Build a Dashboard with Plotly Dash

Pie Chart for Successful Launches for All Sites



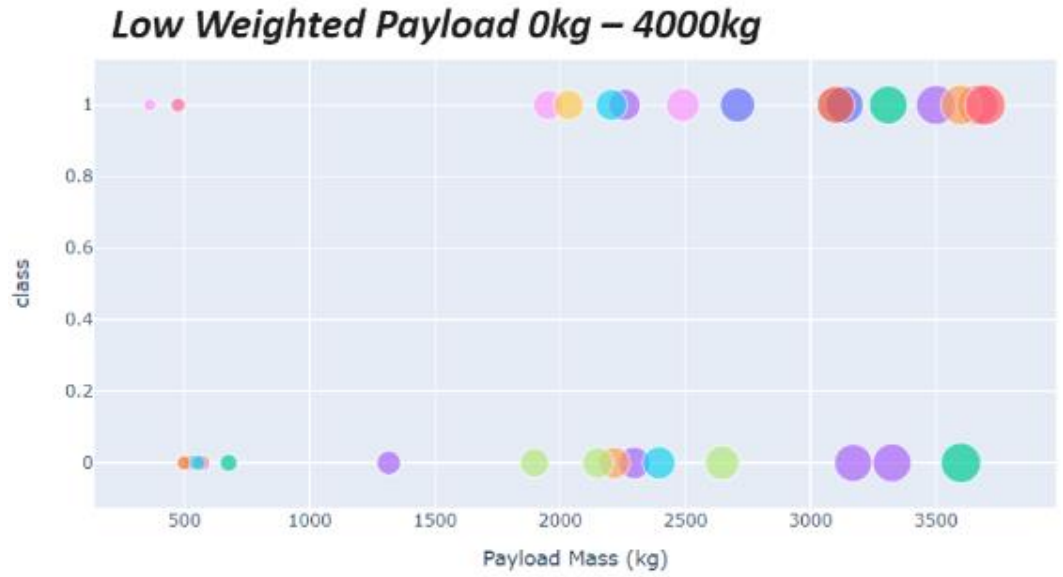
We can see that KSC LC-39A has highest number of successful launches. We also see that CCAFS SLC-40 has lowest number.

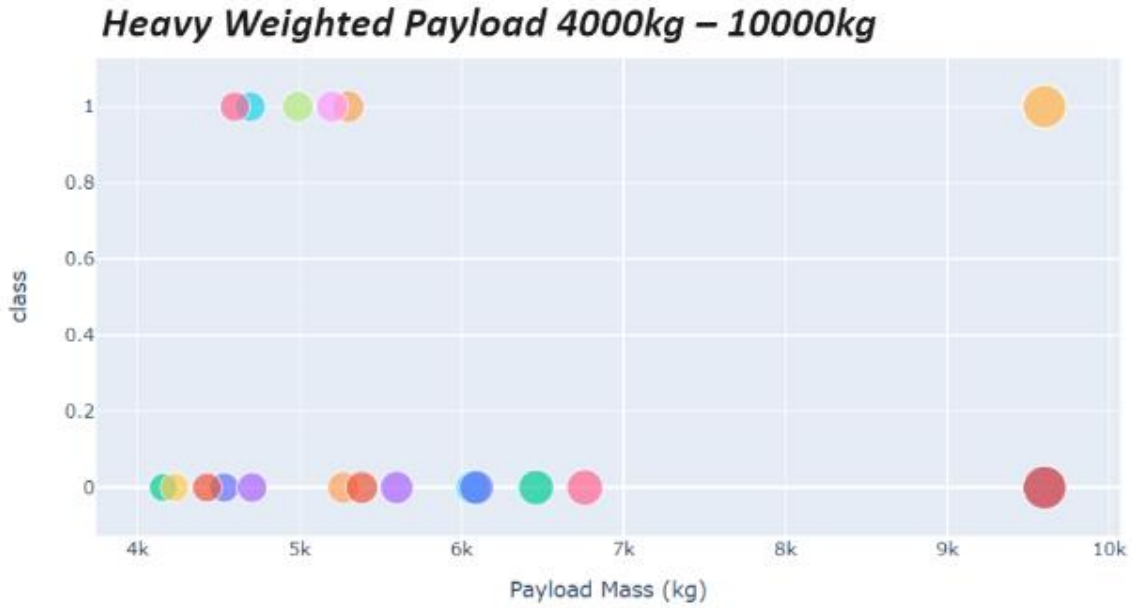
Pie Chart for Success ratio of KSC LC-39A



We plot the pie chart for KSC LC-39A and observe that it has success rate of 76.9 %

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





We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

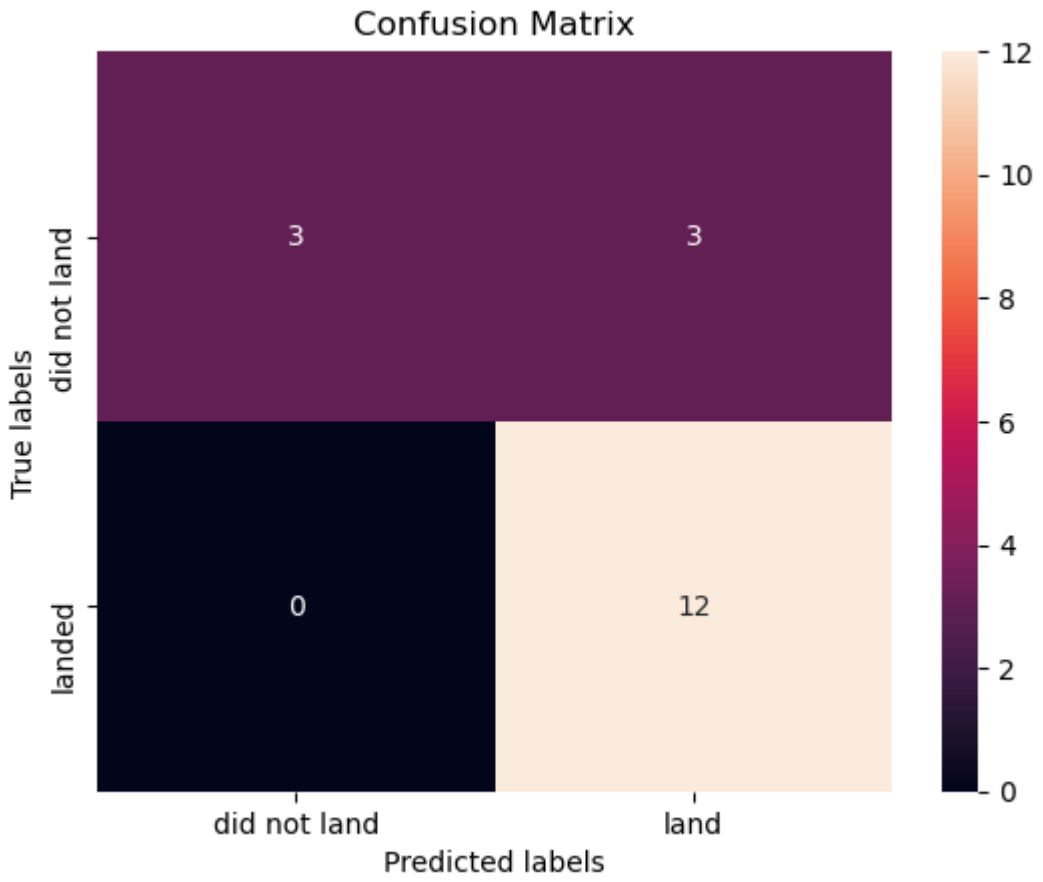
The decision tree classifier is the model with the highest classification accuracy

```
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 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix



Confusion Matrix for Decision Tree Classifier shows it has 12 True-Positive and 3 True-Negative.

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.
- Launch site KSC LC-39A has highest success rate and highest number of successful launches.
- Decision Tree Classifier is the best model for classification whether launch is successful or not.

Thank you!