IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

VS
02.03.2023.

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - Exploratory data analysis (EDA) via SQL
  - Data visualization, EDA
  - Deployment of an interactive launch site map via Folium
  - Deployment of an interactive dashboard via Plotly
  - Classification model selection
- Summary of all results
  - Exploratory data analysis
  - Classification prediction results

Introduction

- SpaceX advertises Falcon 9 rocket launches 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.
- Project goal is to predict whether the Falcon 9 first stage will land successfully to be reused again.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Interacting with the SpaceX REST API via GET requests
  - Parsing data from Wikipedia web-page
- Perform data wrangling
  - Removing the unnecessary data, filling the missing data, converting the categorical variables to indicator variables
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Building and evaluating classifier models: log regression, KNN, decision tree and SVM

Data Collection

- A launch data JSON file was obtained via GET request to the SpaceX REST API
- A 2020 launch record table was scraped from the wikipedia page via BeatifulSoup

Data Collection – SpaceX API

1. Obtain the response from the SpaceX API

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

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

2. Convert the response to a JSON file

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

3. Use the pre-defined functions to excract information from the JSON file

```
getLaunchSite(data)    getPayloadData(data)    getCoreData(data)
```

4. Convert the obtained data to a dataframe

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

5. Select only the 'Falcon 9' entries

```
data_falcon9 = launch_df.loc[launch_df['BoosterVersion'] != 'Falcon 1']
```

6. Replace the missing data

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

7. Load the dataframe into a CSV file

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

https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

1. Obtain response from the wikipedia webpage

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

2. Parse its contents into a BeautifulSoup object

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

3. Locate the data of interest in a HTML table

```
html_tables = soup.find_all('table')
first_launch_table = html_tables[2]
```

4. Get the column names from the table

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

5. Create a dictionary where keys are the column names

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

6. Iterate through the contents of the table and update the dictionary with values

7. Convert the dictionary to a dataframe and store it as a CSV file

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


https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/jupyter-labs-webscraping.ipynb

Data Wrangling

1. Check the number of missing data

2. Check the datatypes stored in each column

3. Calculate the number of launches on each site

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

4. Calculate the number and occurrence of each orbit

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

5. Calculate the number and occurence of mission outcome per orbit type

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

6. Create a landing outcome label from Outcome column

```
# landing_class = 0 if bad_outcome
landing_class = []
# landing_class = 1 otherwise
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
landing_class
```

7. Assign the label to the target variable column

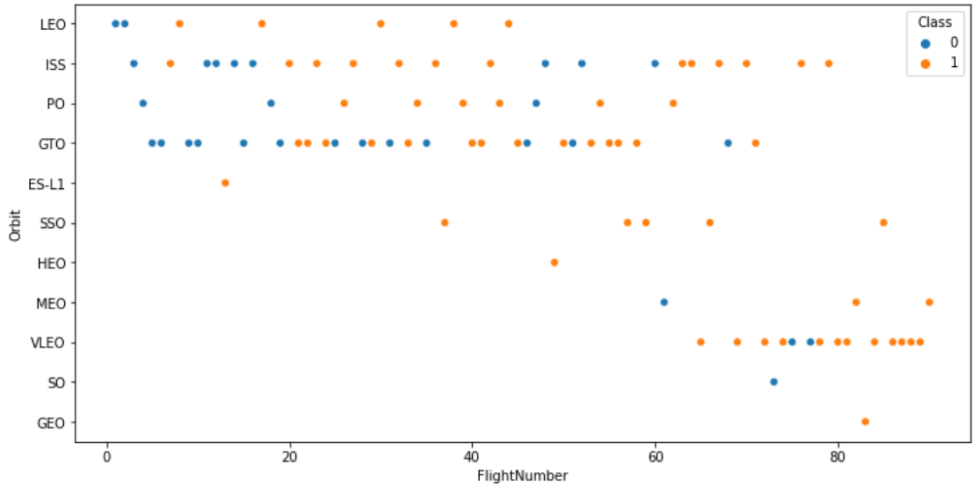
```
df['Class']=landing_class
```

8. Store the dataframe

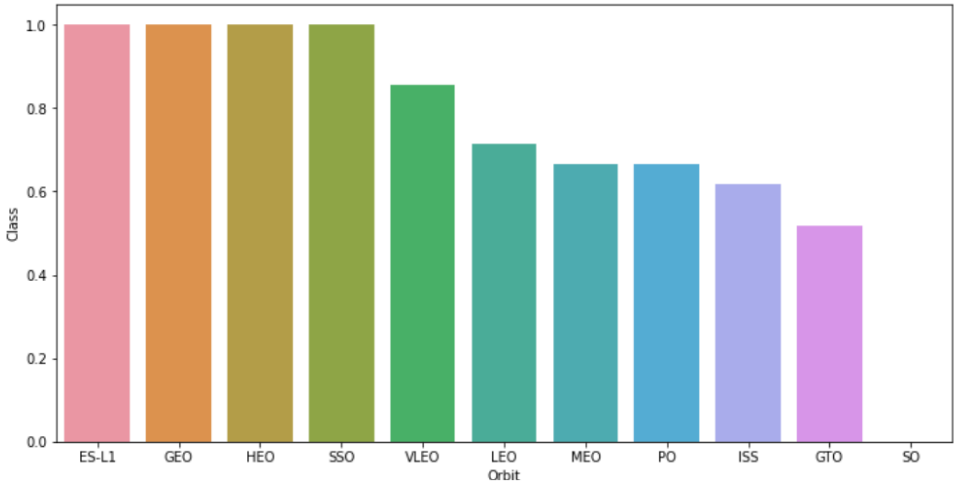
```
df.to_csv("dataset_part_2.csv", index=False)
```


https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/labs-jupyter-spacex-Data%20wrangling.ipynb

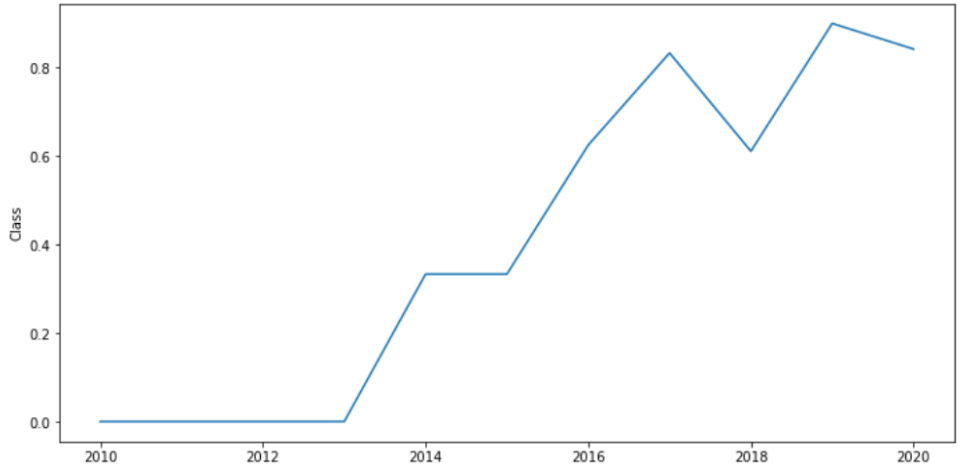
EDA with Data Visualization



1. Scatterplots were created to check for possible connection between 2 variables



2. This barplot shows the success rate of rocket landings depending on the orbit



3. This lineplot shows the overall landing success rate per year

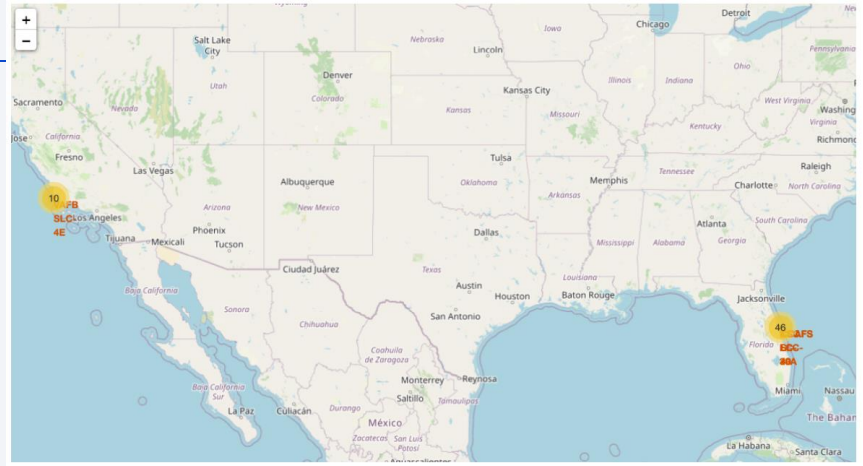
https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- SELECT DISTINCT LaunchSite FROM SPACEXDATASET
- SELECT * FROM SPACEXDATASET WHERE LaunchSite LIKE 'CCA%' LIMIT 5
- SELECT SUM(PayloadMass) AS total_payload FROM SPACEXDATASET WHERE Customer LIKE 'NASA (CRS)'
- SELECT AVG(PayloadMass) AS average_payload FROM SPACEXDATASET WHERE BoosterVersion LIKE 'F9 v1.1'
- SELECT MIN(Date) FROM SPACEXDATASET WHERE LandingOutcome LIKE 'Success (ground pad)'
- SELECT BoosterVersion FROM SPACEXDATASET WHERE LandingOutcome = 'Success (drone ship)'
- AND PayloadMassKG > 4000 AND PayloadMassKG < 6000
- SELECT COUNT(MissionOutcome) FROM SPACEXDATASET WHERE MissionOutcome LIKE 'Success%';
- SELECT COUNT(MissionOutcome) FROM SPACEXDATASET WHERE MissionOutcome LIKE 'Failure%';
- SELECT BoosterVersion, PayloadMassKG FROM SPACEXDATASET
- WHERE PayloadMassKG = (SELECT MAX(PayloadMassKG) FROM SPACEXDATASET) ORDER BY BoosterVersion
- SELECT BoosterVersion, LaunchSite, LandingOutcome, substr(Date, 4, 2)
- FROM SPACEXDATASET WHERE LandingOutcome LIKE 'Failure (drone ship)' AND substr(Date,7,4)='2015'
- SELECT LandingOutcome, COUNT(LandingOutcome) FROM SPACEXDATASET WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LandingOutcome ORDER BY COUNT(LandingOutcome) DESC

https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium



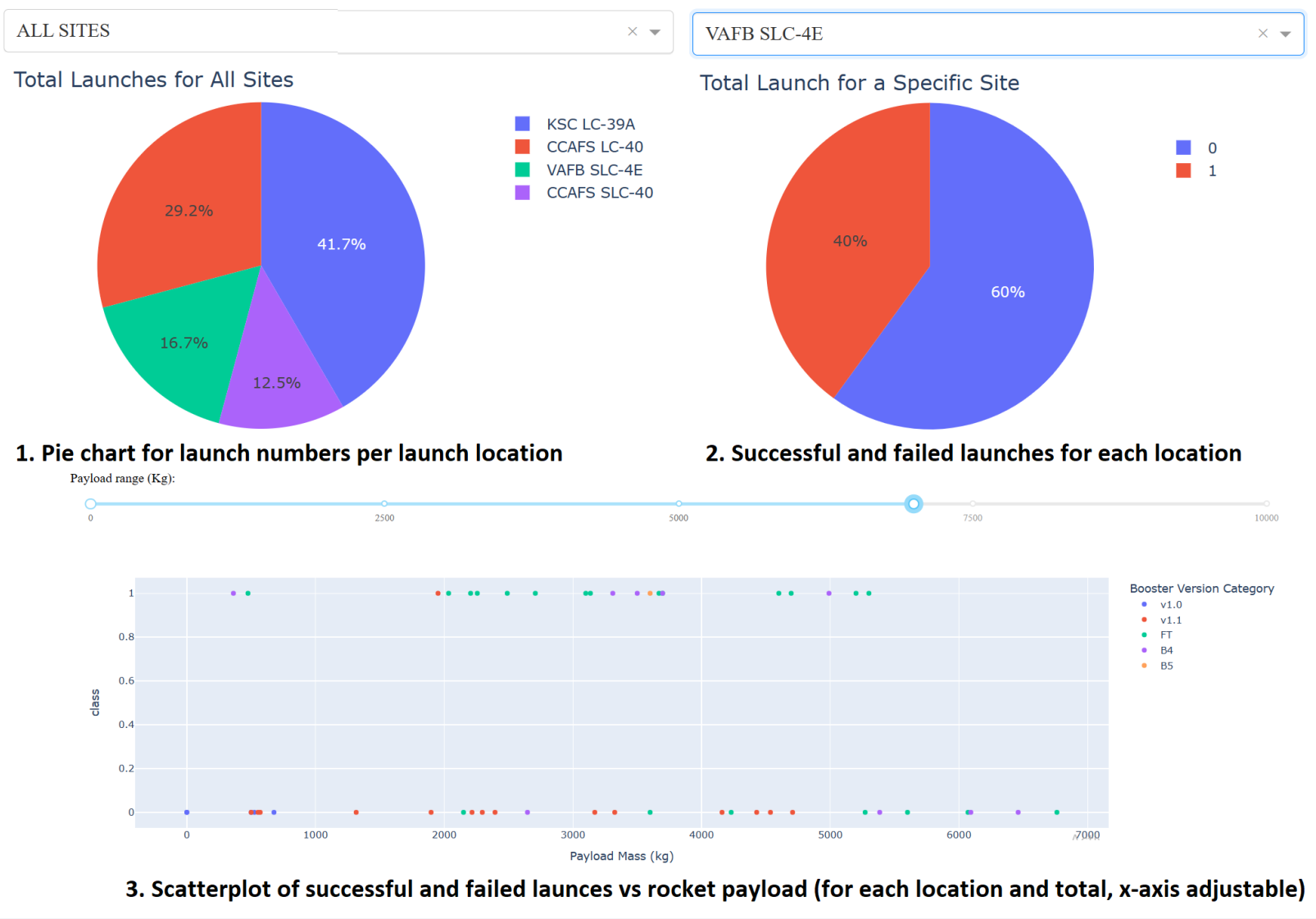
1. Launch locations were plotted, marker clusters created

2. Green and red Markers for successful and failed landings

3. A function was created to connect a launch location with the mouse cursor position with a straigth line

https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash



https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/SpaceX%20Plotly%20Dashboard.py

Predictive Analysis (Classification)

1. Create the feature X dataframe and the target variable Y array

```
X = pd.read_csv('dataset_part_3.csv')
Y = data['Class'].to_numpy()
```

2. Scale the numeric values in X using Standard Scaler

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

3. Split the dataset 80 : 20 as train : test

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

4. Using GridSearchCV, fit a Logistic Regression and calculate prediction accuracy

```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

5. Using GridSearchCV, fit a SupportVectorMachine. Calculate prediction accuracy

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

6. Using GridSearchCV, fit a Decision Tree and calculate prediction accuracy

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 16, 'max_features': 'sqrt',
'min_samples_leaf': 4, 'min_samples_split': 5, 'splitter': 'random'}
accuracy : 0.9
```

7. Using GridSearchCV, fit a KNN model and calculate prediction accuracy

```
tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

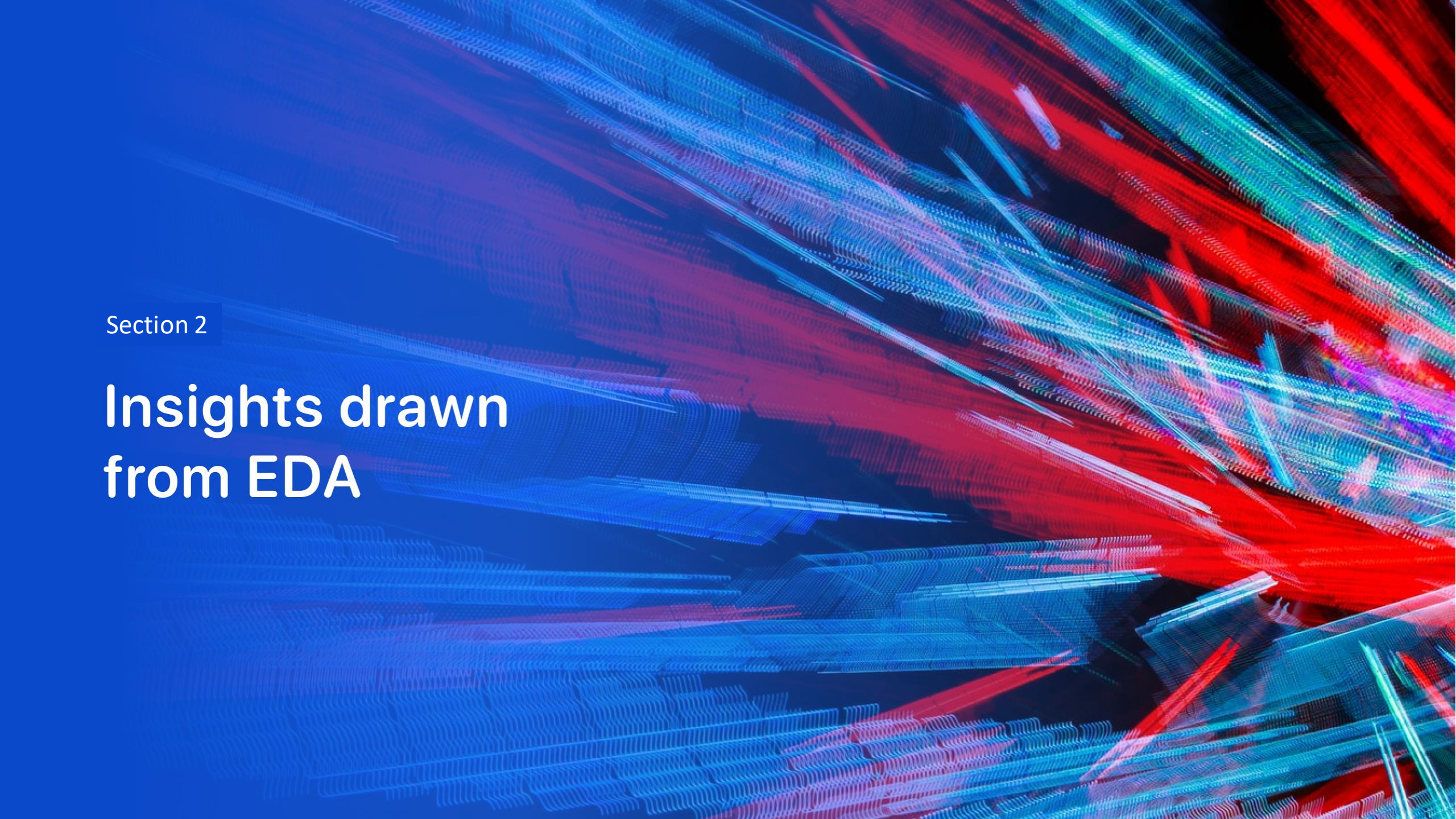

https://github.com/vladislav-sitchihin/IBM/blob/Module-10---Capstone-SpaceX/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

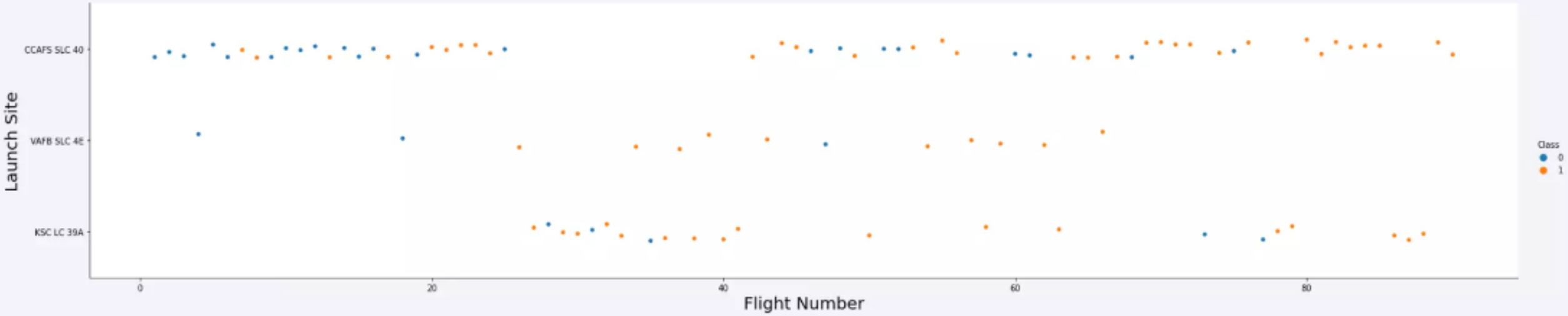
- Successful landings become more frequent with the time (most of the failed landings were planned though).
- As shown by the dashboard, rockets with lower payload mass landed successfully more often.
- KSC LC 39A has the most successful launches.
- All classifier models performed well with accuracy score between 0.83 and 0.85.

Section 2

Insights drawn from EDA

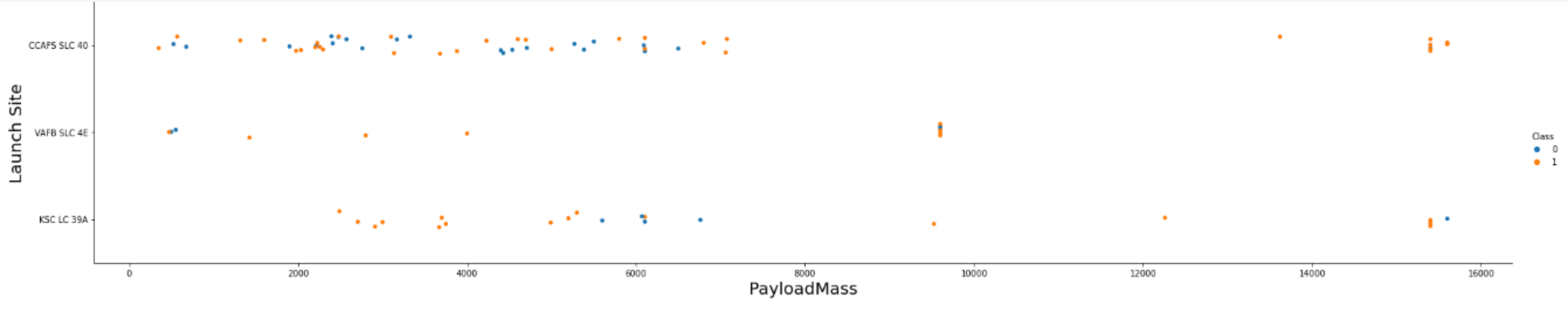


Flight Number vs. Launch Site



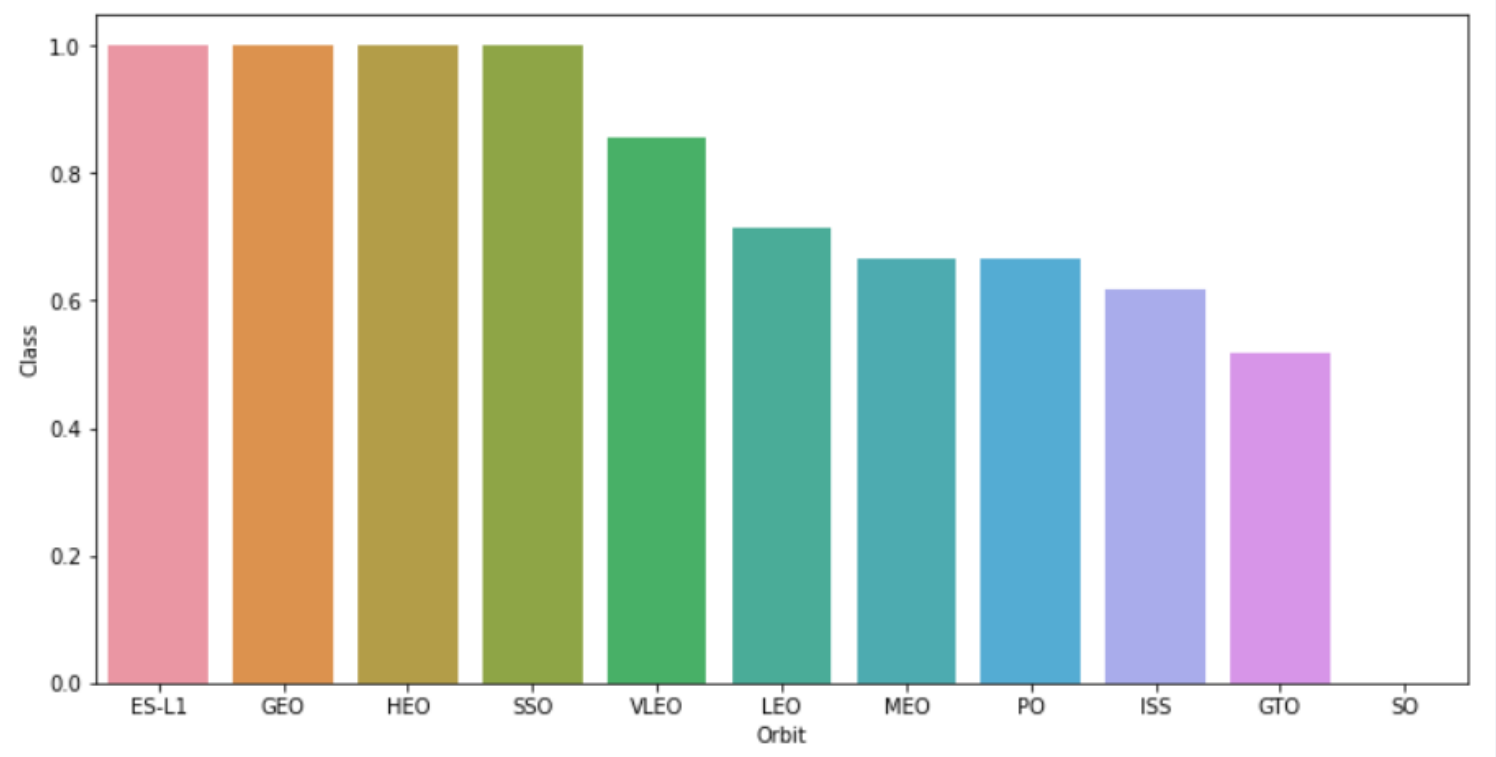
- Launches from CCAFS SLC 40 are more frequent (dense on the x-axis)

Payload vs. Launch Site



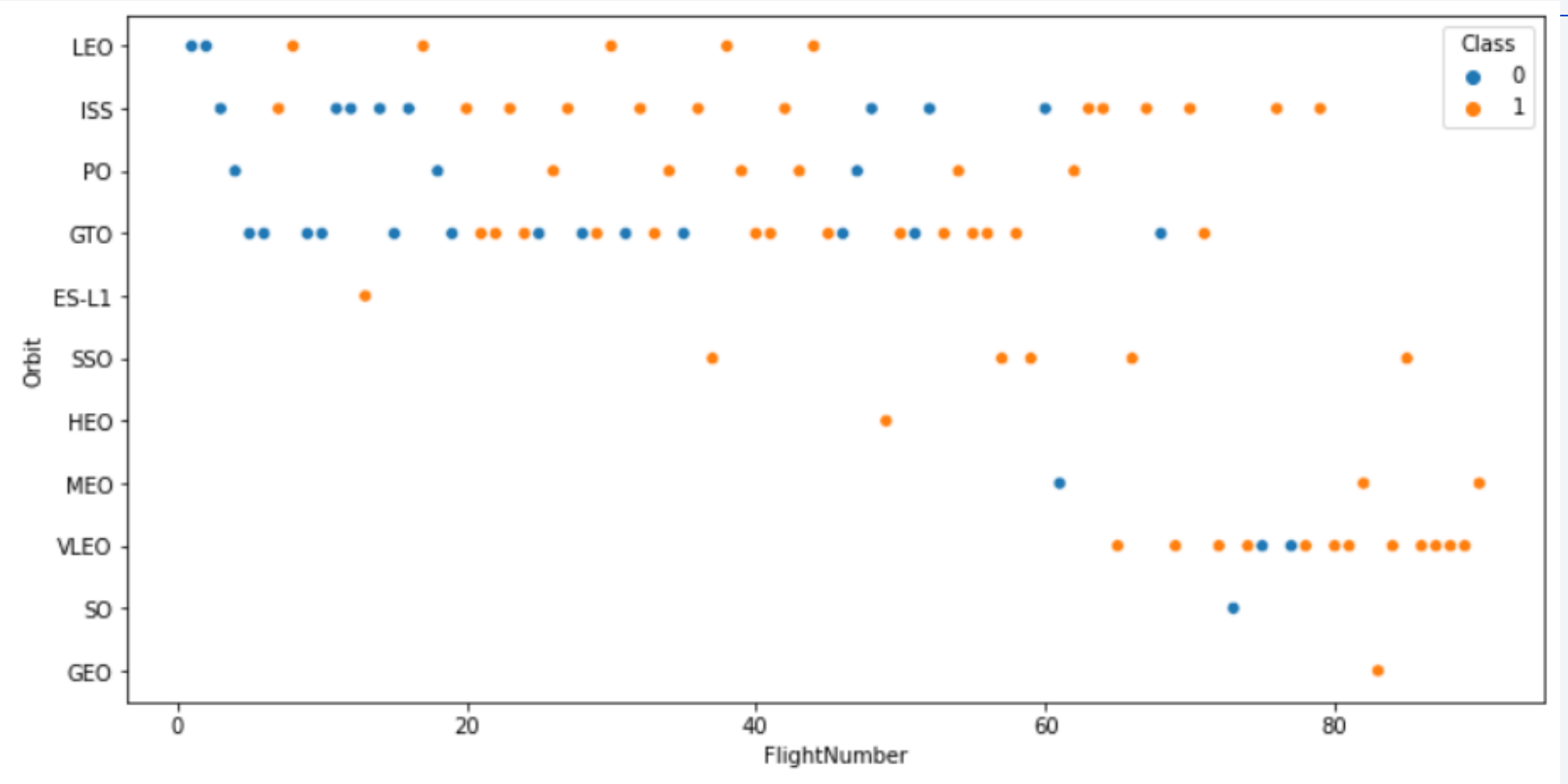
- The VAFB launch site was used to launch lighter rockets

Success Rate vs. Orbit Type



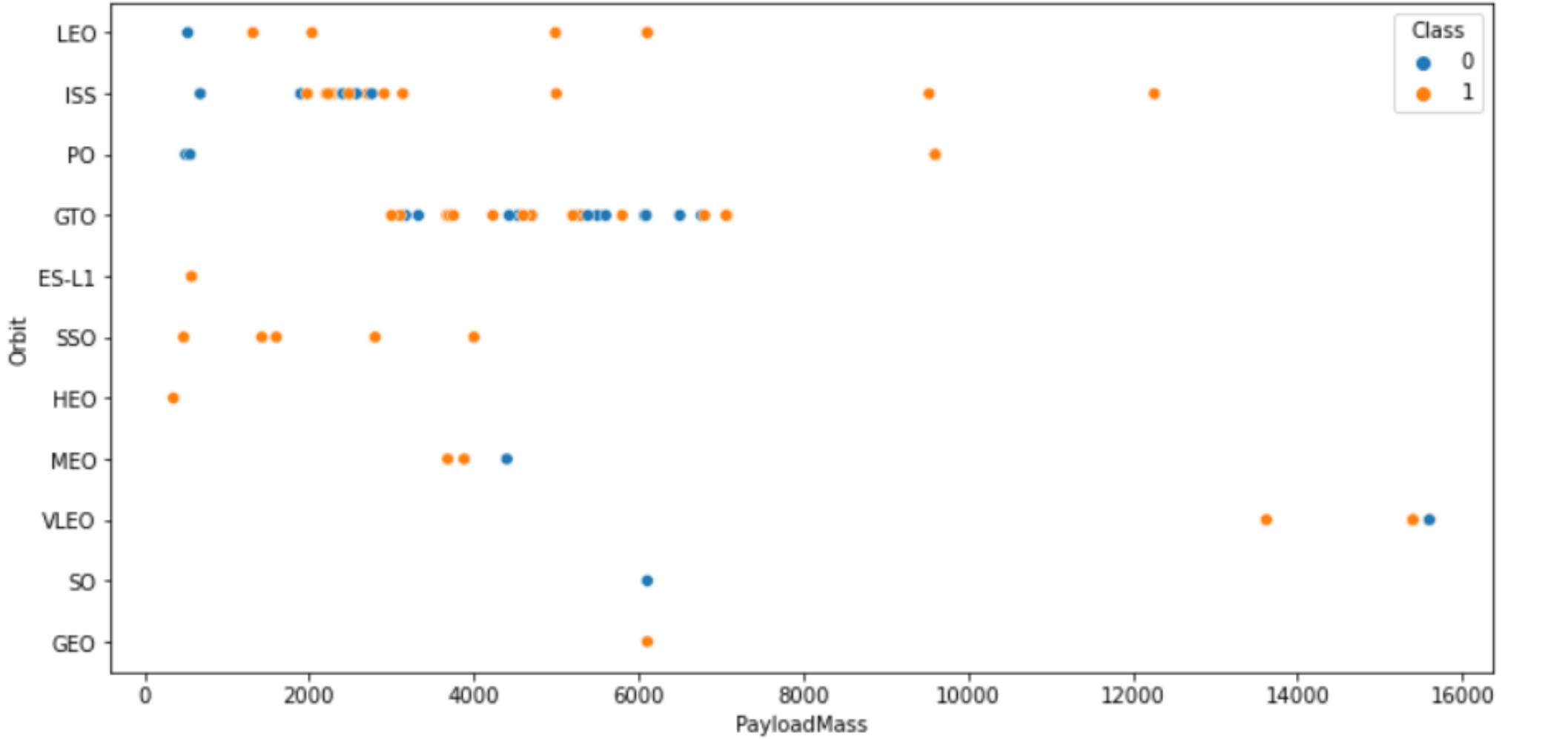
- A bar chart for the success rate of each orbit type

Flight Number vs. Orbit Type



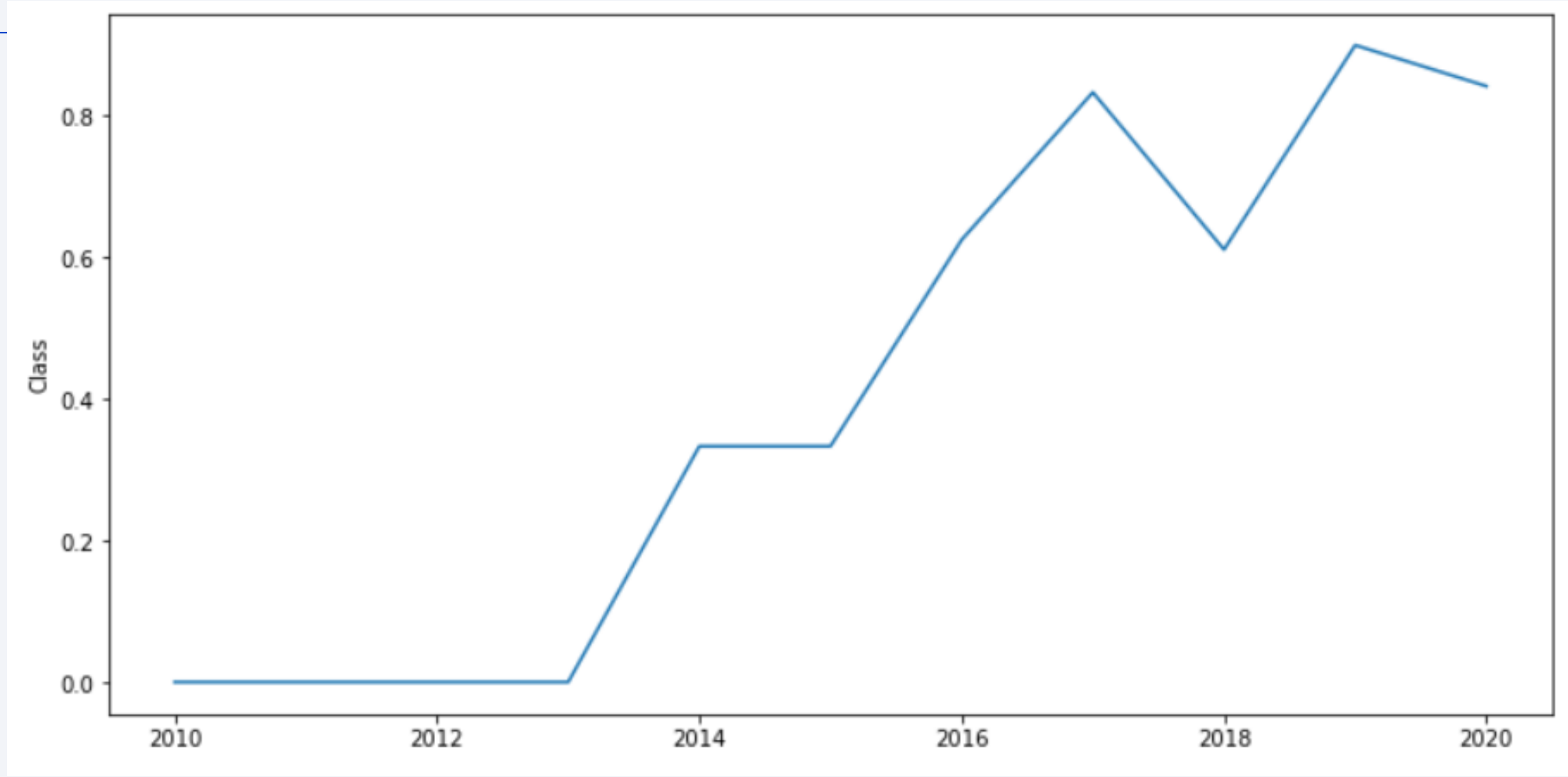
- A scatter point of Flight number vs. Orbit type

Payload vs. Orbit Type



- A scatter plot of payload vs. orbit type

Launch Success Yearly Trend



- A line chart of yearly average success rate

All Launch Site Names

- The names of the unique launch sites
- `SELECT DISTINCT LaunchSite FROM SPACEXDATASET`

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

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with `CCA`
- `SELECT * FROM SPACEXDATASET WHERE LaunchSite LIKE 'CCA%' LIMIT 5`

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

- total payload carried by boosters from NASA is 45596
- `SELECT SUM(PayloadMass) AS total_payload FROM SPACEXDATASET WHERE Customer LIKE 'NASA (CRS)'`

Average Payload Mass by F9 v1.1

- average payload mass carried by booster version F9 v1.1 is 2928.4
- `SELECT AVG(PayloadMass) AS average_payload FROM SPACEXDATASET WHERE BoosterVersion LIKE 'F9 v1.1'`

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad is

  2015-12-22

- ```
  SELECT MIN(Date) FROM SPACEXDATASET
  WHERE LandingOutcome LIKE 'Success (ground pad)'
  ```
  ```

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
-
```
SELECT BoosterVersion FROM SPACEXDATASET
WHERE LandingOutcome = 'Success (drone ship)'
AND PayloadMassKG > 4000 AND PayloadMassKG < 6000
```

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

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failure mission outcomes is

  100

- ```
  SELECT COUNT(MissionOutcome) FROM SPACEXDATASET
  WHERE MissionOutcome LIKE 'Success%';
  SELECT COUNT(MissionOutcome) FROM SPACEXDATASET
  WHERE MissionOutcome LIKE 'Failure%';
  ```
  ```

Boosters Carried Maximum Payload

- The names of the booster which have carried the maximum payload mass
- SELECT BoosterVersion, PayloadMassKG FROM SPACEXDATASET WHERE PayloadMassKG = (SELECT MAX(PayloadMassKG) FROM SPACEXDATASET) ORDER BY BoosterVersion

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

- The failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- SELECT BoosterVersion, LaunchSite, LandingOutcome, substr(Date, 4, 2)
  FROM SPACEXDATASET WHERE LandingOutcome LIKE 'Failure (drone ship)' AND substr(Date,7,4)='2015'

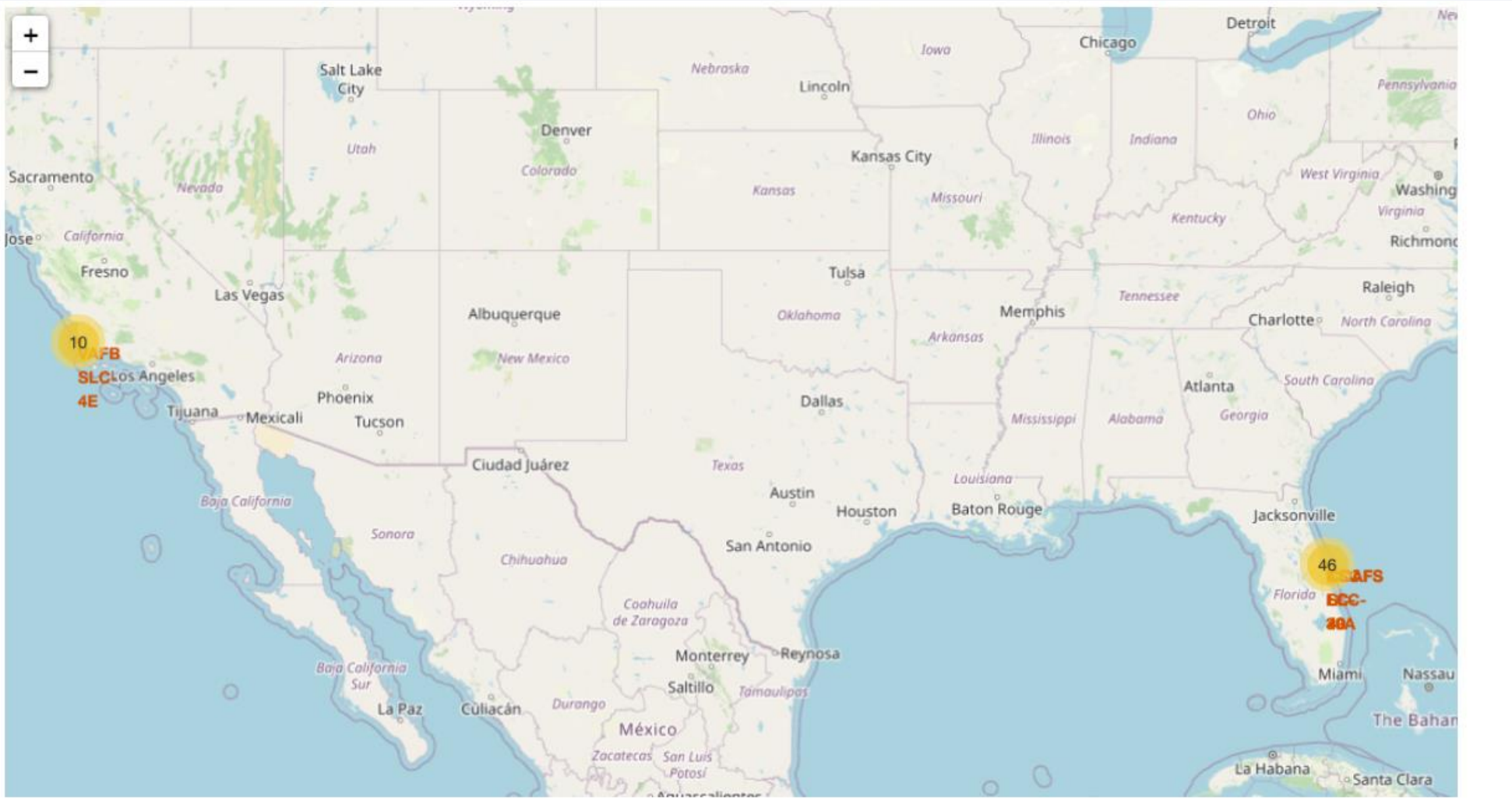
booster_version	launch_site	landing__outcome
F9 FT B1031.1	KSC LC-39A	Success (ground pad)
F9 FT B1029.1	VAFB SLC-4E	Success (drone ship)
F9 FT B1026	CCAFS LC-40	Success (drone ship)
F9 FT B1025.1	CCAFS LC-40	Success (ground pad)
F9 FT B1023.1	CCAFS LC-40	Success (drone ship)

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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
-
```
SELECT LandingOutcome, COUNT(LandingOutcome)
FROM SPACEXDATASET WHERE Date BETWEEN '2010-06-04' AND
'2017-03-20' GROUP BY LandingOutcome ORDER
BY COUNT(LandingOutcome) DESC
```

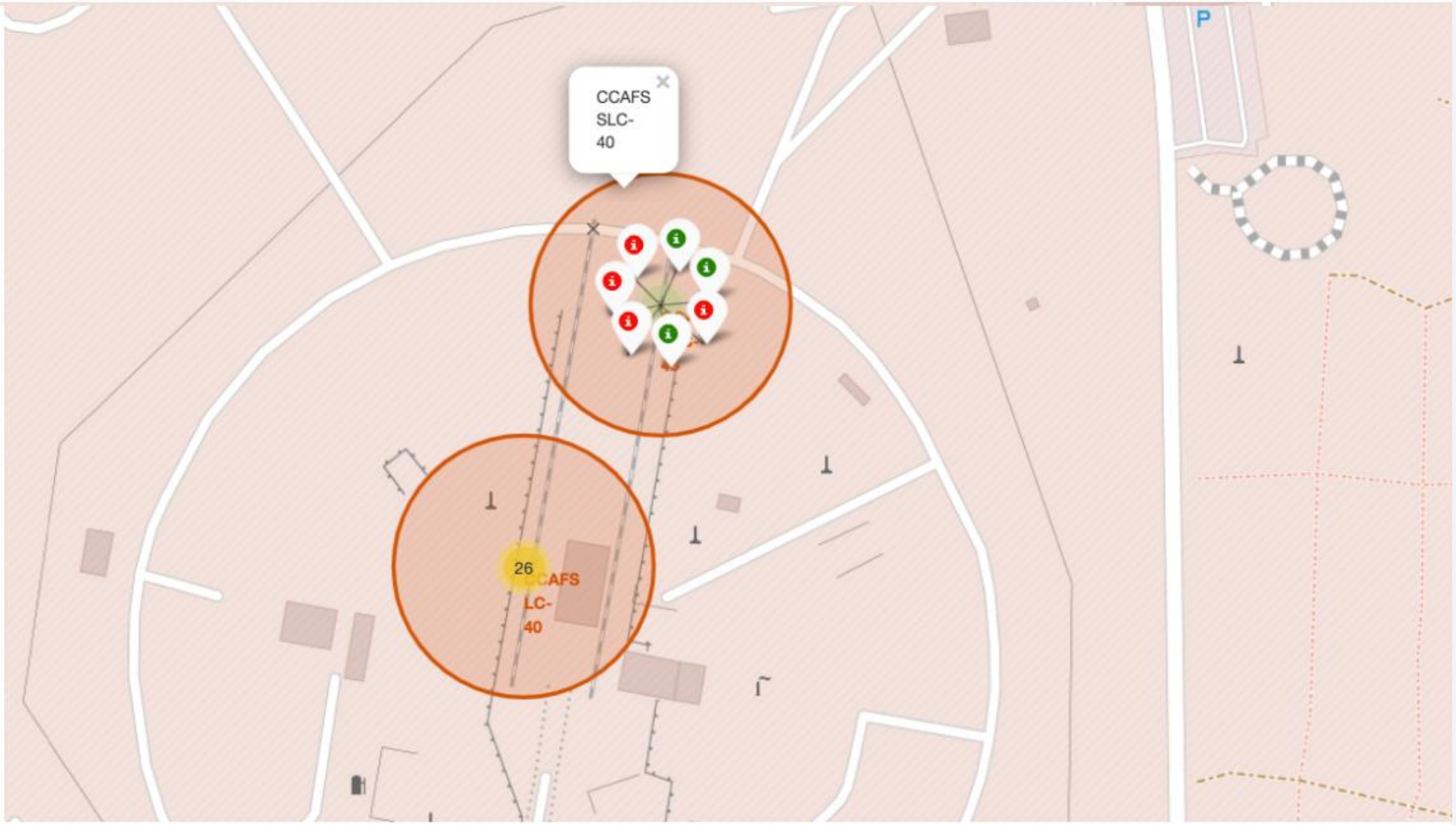

Section 3
Launch Sites
Proximities Analysis

All launch locations



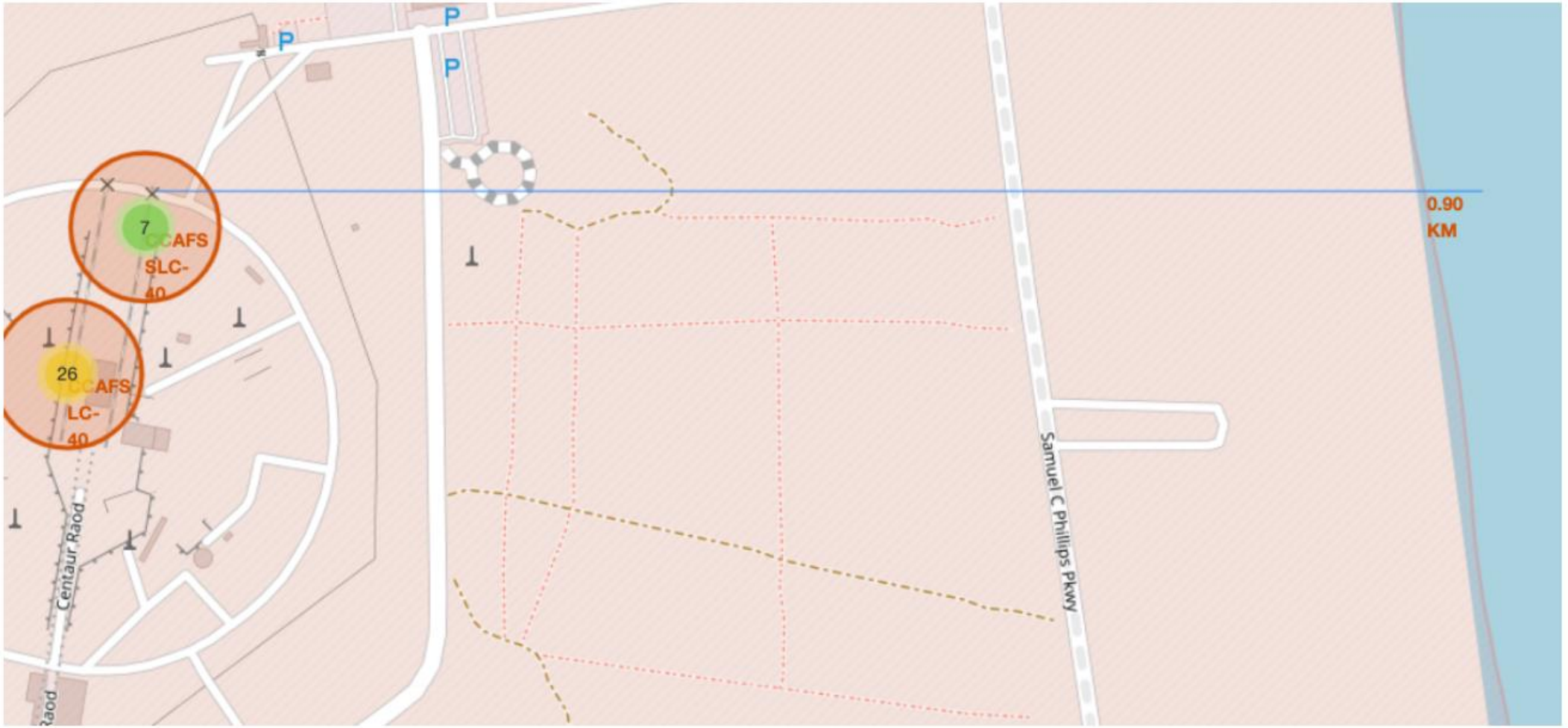
1. Launch locations were plotted, marker clusters created

Launch outcomes at a location

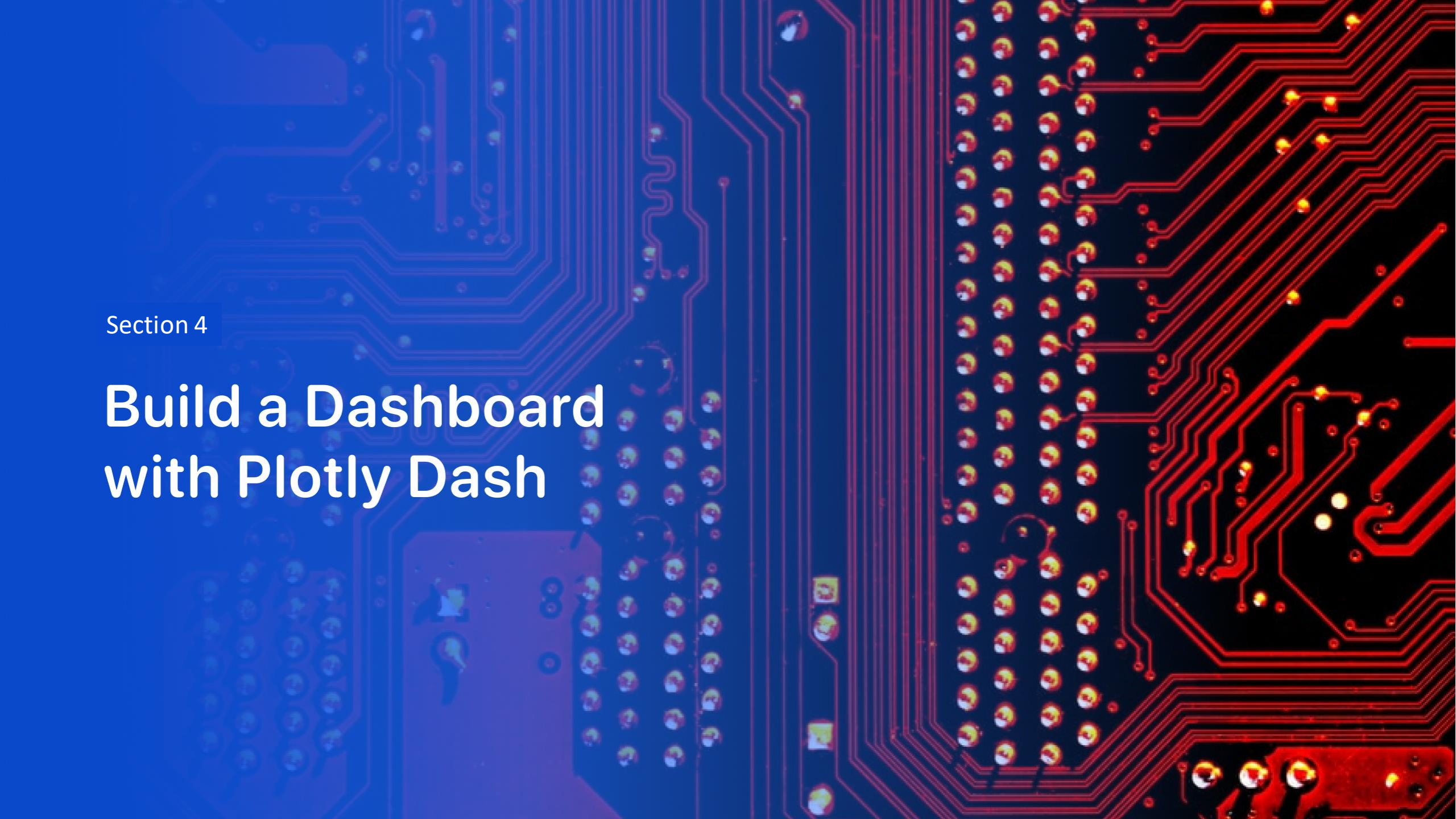


2. Green and red Markers for successful and failed landings

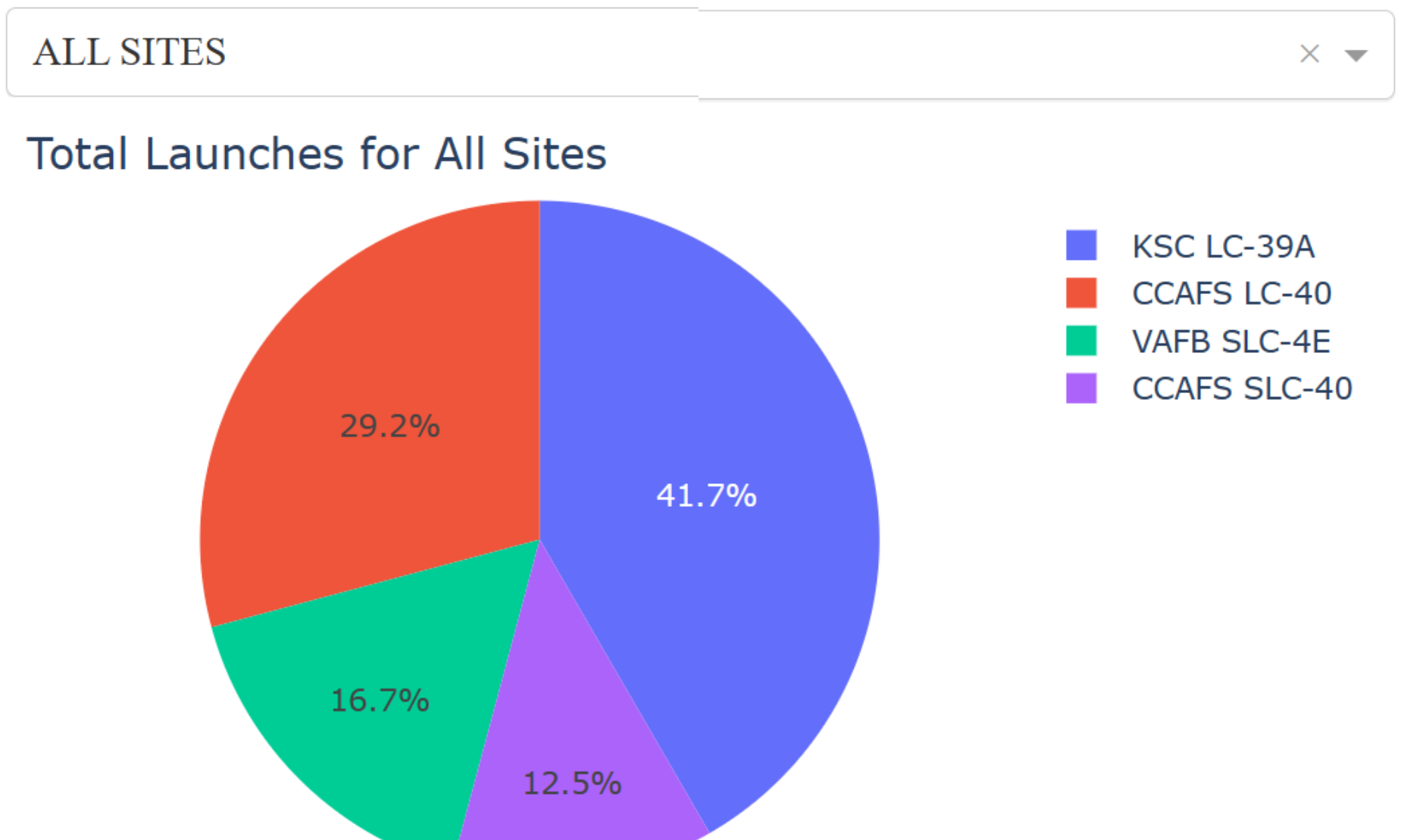
Distance from a launch site to its nearest coastline



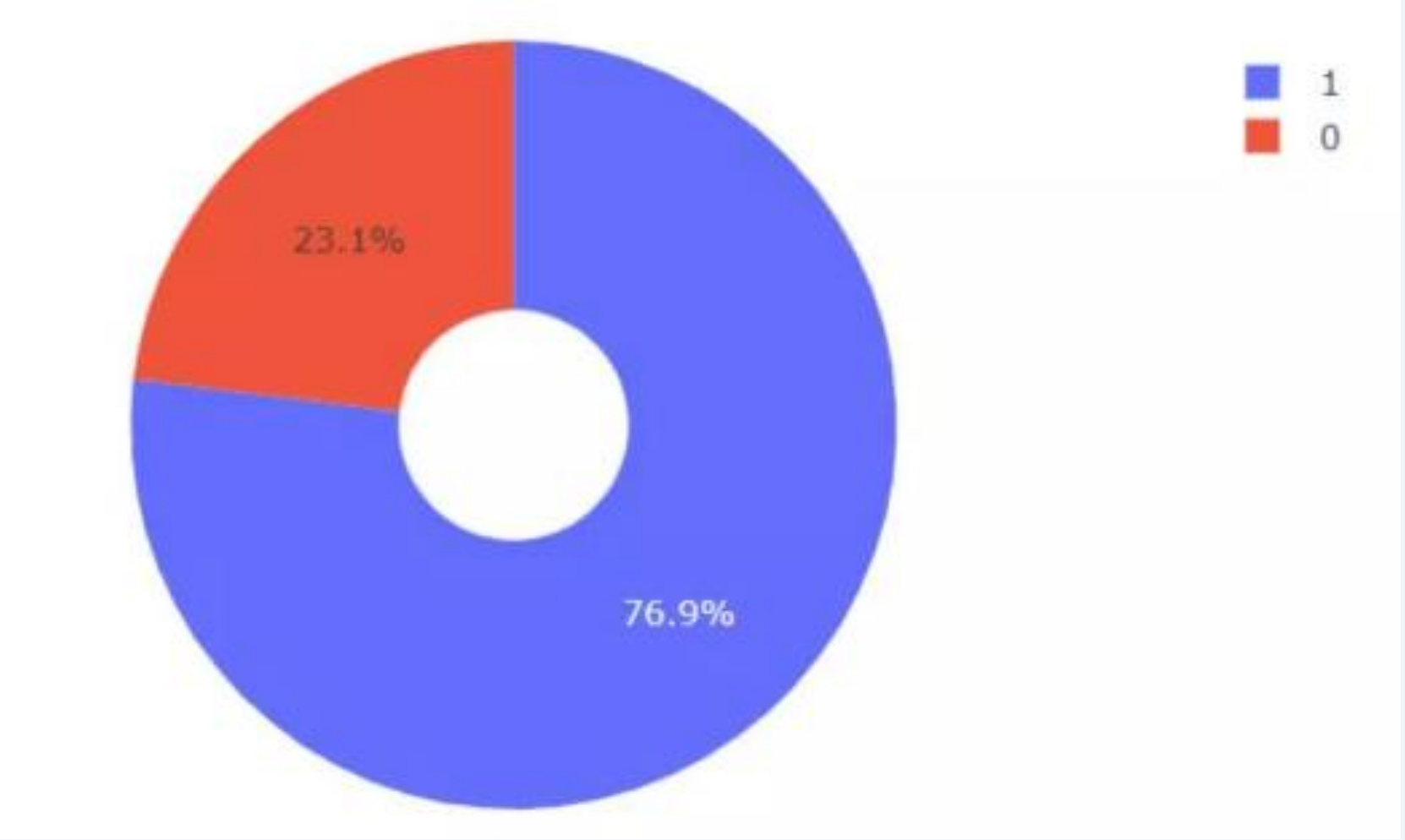
3. A function was created to connect a launch location with the mouse cursor position with a straigth line


Section 4
Build a Dashboard with Plotly Dash

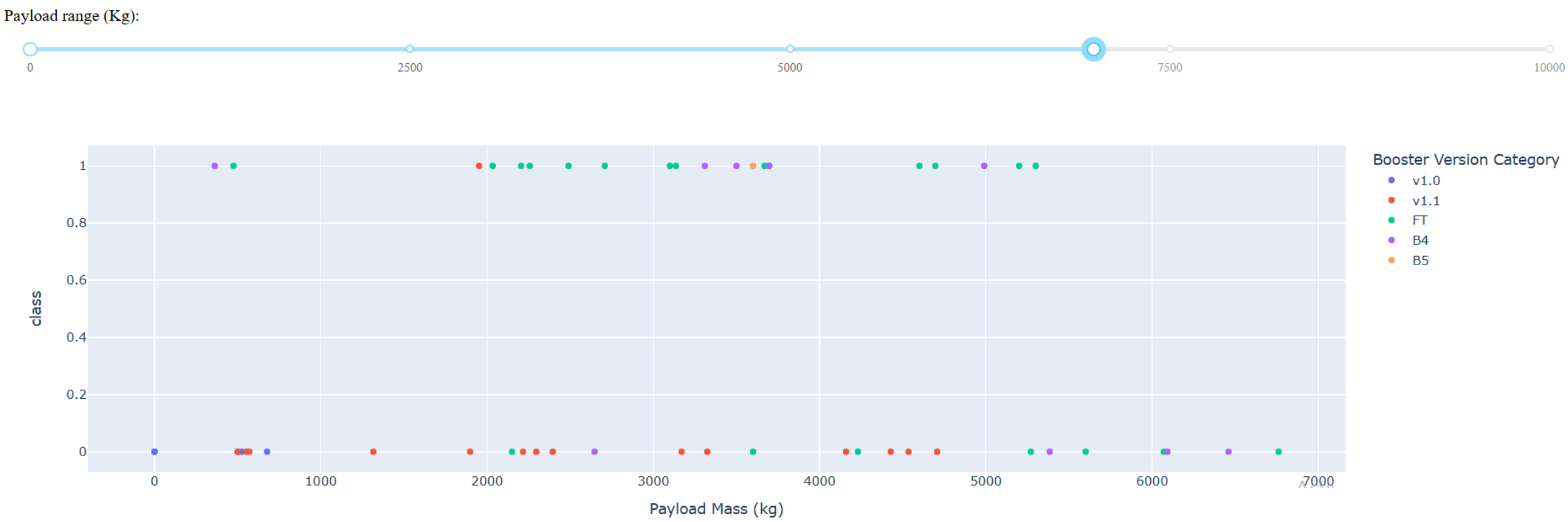
Total launches for all sites



The launch site with highest launch success ratio



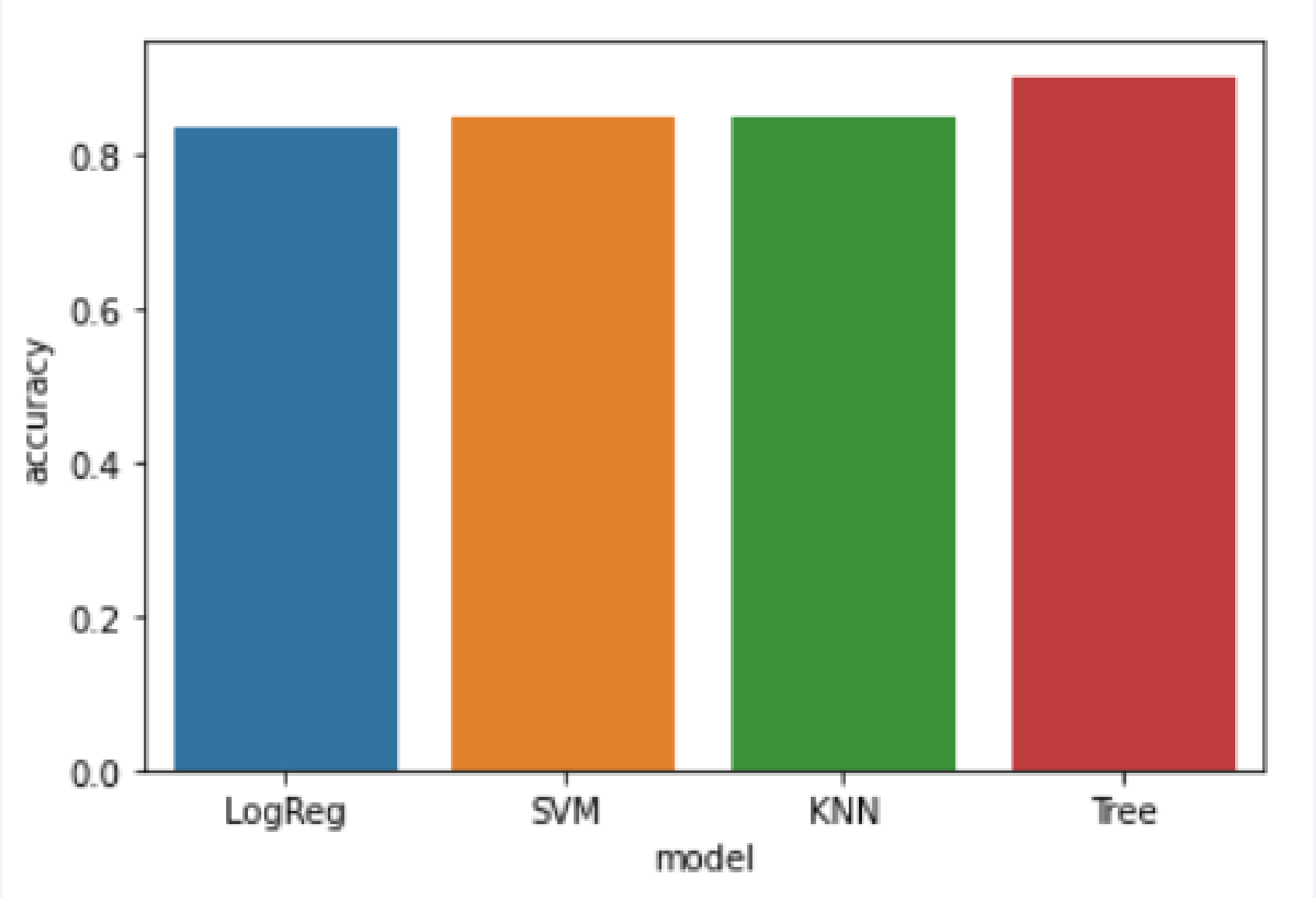
Payload vs. Launch Outcome scatter plot for all sites



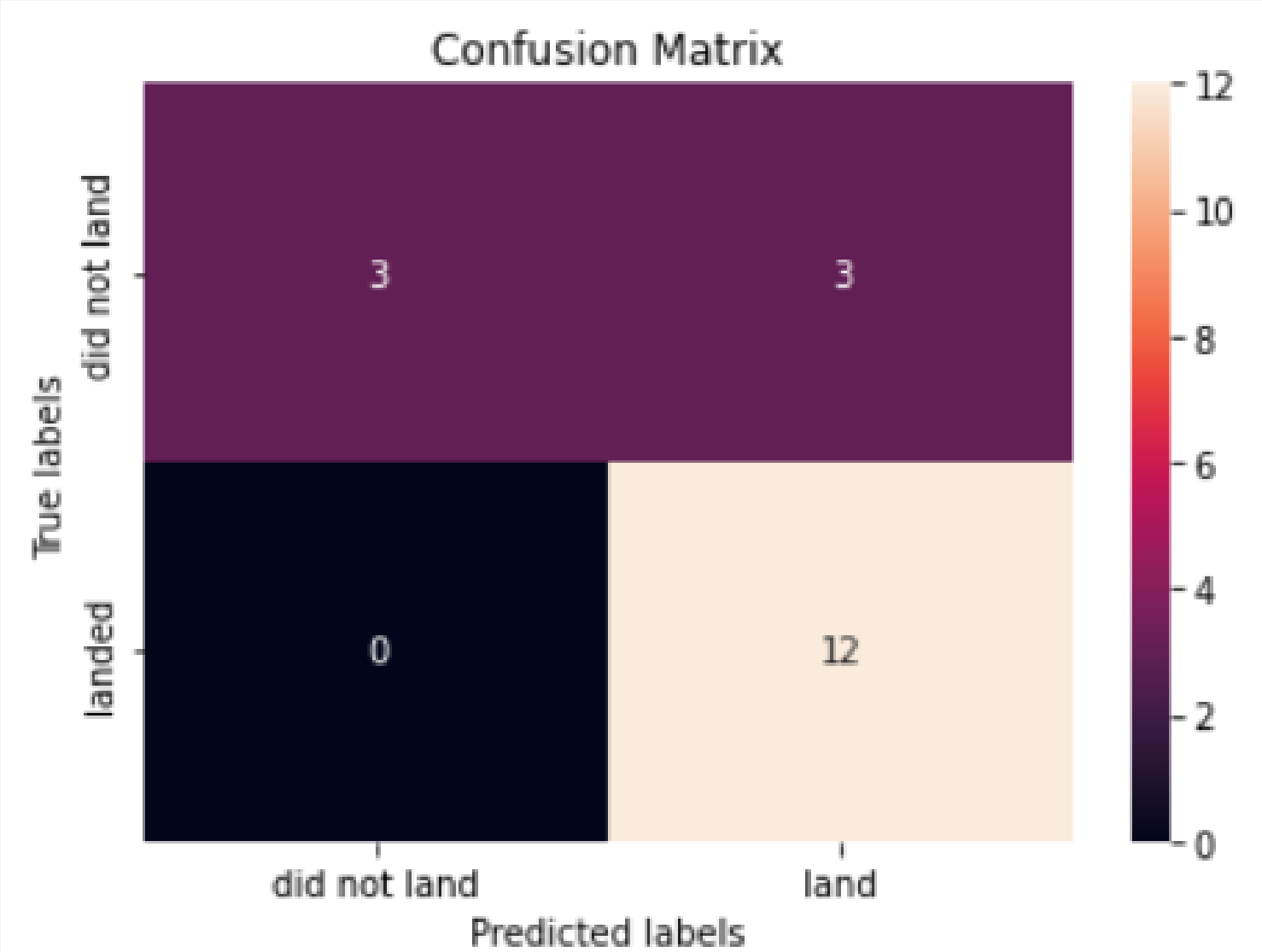
Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix of the Decision Tree predictions



Conclusions

- Decision tree classifier allows to predict the outcome of rocket landings with accuracy of 0.9 on the testing sub-set of the data
- Other observations:
  - Successful landings become more frequent with the time (most of the failed landings were planned though).
  - As shown by the dashboard, rockets with lower payload mass landed successfully more often.
  - KSC LC 39A has the most successful launches.

Thank you!