IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Victor Eduardo Cauvilla de Oliveira
10-03-2023

# Outline

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

# Executive Summary

- Summary methodologies
  - Data collection with SpaceX API
  - Data collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) using SQL
  - EDA using Data Visualization
  - Interactive Visual Analytics with Folium and Dash
  - Predictive Analysis with Machine L:earning
- Summary of results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

# Introduction

- Project background and context

  Space X 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 Space X 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 space X for a rocket launch. The objective of this project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Data was obtained using SpaceX API and web-scraping from Wikipedia
- Perform data wrangling
  - One-hot encoding was done on categorical features.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic Regression, SVM, Decision Tree and KNN models were used
  - Hyperparameters were turned using GridSearch cross-validation
  - Accuracy on test data was calculated using score method and confusion matrix was plotted.

# Data Collection

- Following steps were involved in Data Collection:
  - SpaceX API:
    - Data was collected by sending GET request to SpaceX API
    - The data was decoded as json using .json() and turned into a DataFrame using .json_normalize()
    - The data was then cleaned, missing values were identified and filled where necessary.
  - Web-Scraping:
    - Falcon 9 launch records were also collected by web-scraping from Wikipedia
    - A HTTP GET request was sent to Falcon 9 launch HTML page
    - The data was parsed and stored as a DataFrame using BeatifulSoup

# Data Collection – SpaceX API

- A GET request was sent to the SpaceX API.
- The data was decoded using .json() and converted to DataFrame using .json_normalize()
- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/jupyter-labs-spacex-data-collection-api.ipynb

## 1 – GET Request to SpaceX API

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

response = requests.get(spacex_url)
```

## 2 – Decoding using .json() and converted to a DataFrame using .json_normalize()

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage

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

## 3 – Data Cleaning and Dealing with Missing Values

```
# Calculate the mean value of PayloadMass column
PayLoadMassMean = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan, PayLoadMassMean, inplac
```

# Data Collection - Scraping

- HTTP GET request was sent to Falcon 9 Launch HTML Page
- Data was parsed and stored as DataFrama using BeautifulSoup
- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/jupyter-labs-webscraping.ipynb

## 1 – HTTP GET request to SpaceX Falcon 9 Launch HTML Page

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_
```

```
data  = requests.get(static_url)
```

## 2 – BeautifulSoup Object

```
soup = BeautifulSoup(data, 'html5lib')
```

## 3 – Extract all column/variable names from the HTML table header

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

# Data Wrangling

- Calculated the number of launches on each site, the number and occurrence of each orbit, and types and respective number of landing outcomes
- Crated a landing outcome label from the Outcome column
- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/labs-jupyter-spacex-Data%20wrangling.ipynb

## 1 – Number of launches on each site

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

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
```

## 2 – Number and Occurrence of each orbit

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

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
HEO       1
SO        1
ES-L1     1
GEO       1
```

## 3 – Types and number of landing outcomes

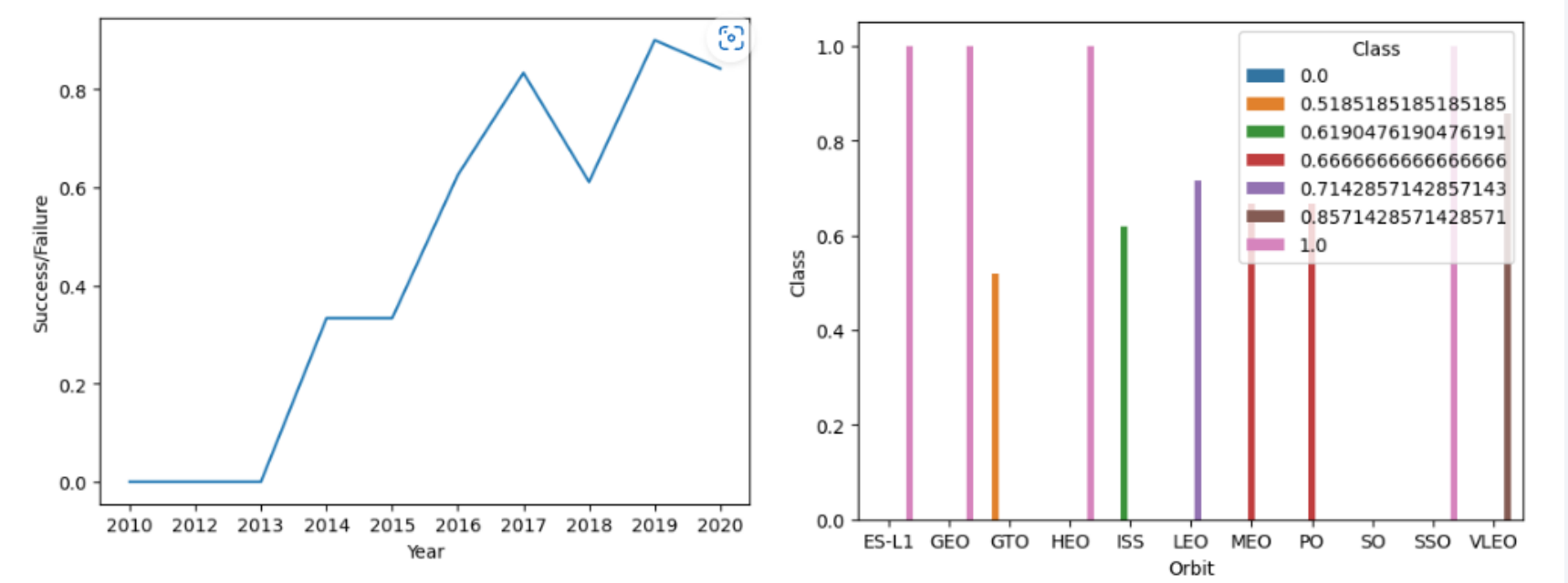
```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

## 4 – Landing column label from Outcome Column

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

# EDA with Data Visualization

- We visualized the relationship between flight number and launch site, payload and launch site, success rate and orbit, number of flights and orbit, payload mass and orbit, and launch success yearly trend




- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/jupyter-labs-eda-dataviz.ipynb

# EDA with SQL

- SQL table was loaded in Jupyter and following SQL queries were performed:
  - 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
  - The total number of successful and failure mission outcomes
  - The failed lading outcomes in drone ship, their booster version and launch site names

- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/jupyter-labs-eda-sql-coursera.ipynb

# Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the 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, we identified which launch sites have relatively high success rate.
- We calculated the distances between the launch site and its proximities like railways, highways, and cities.
- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/lab_jupyter_launch_site_location.ipynb

# Build a Dashboard with Plotly Dash

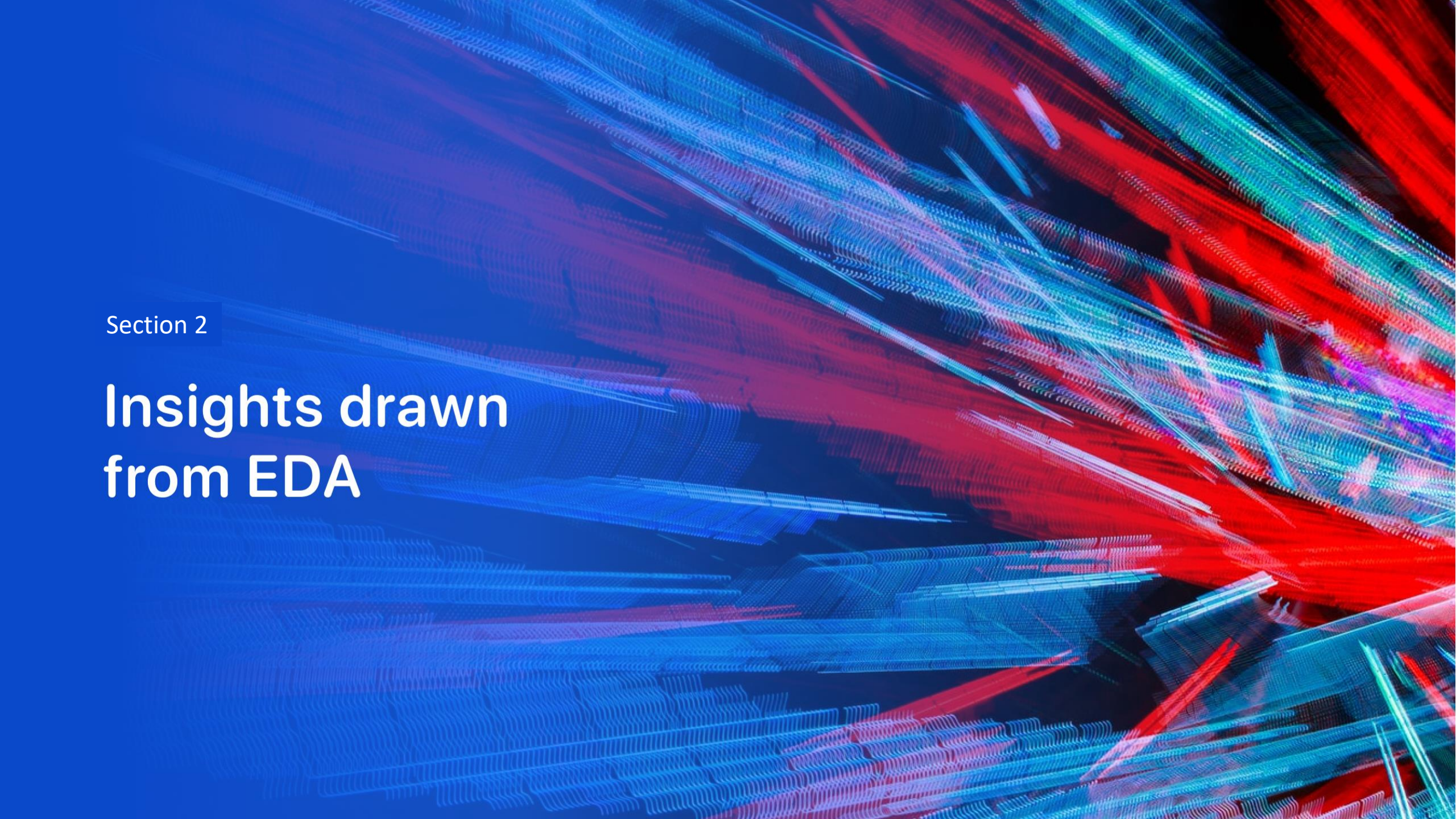
- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/spacex_dash_app.py

# Predictive Analysis (Classification)

- We split the dataset into training and testing set.
- We built different machine learning models and tuned different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model and improved the model using feature engineering and hyperparameter tuning.
- We found the best performing classification model.
- GitHub Link: https://github.com/Falkenus/IBM-DataScienceFiles/blob/main/Applied%20Data%20Science%20Capstone/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

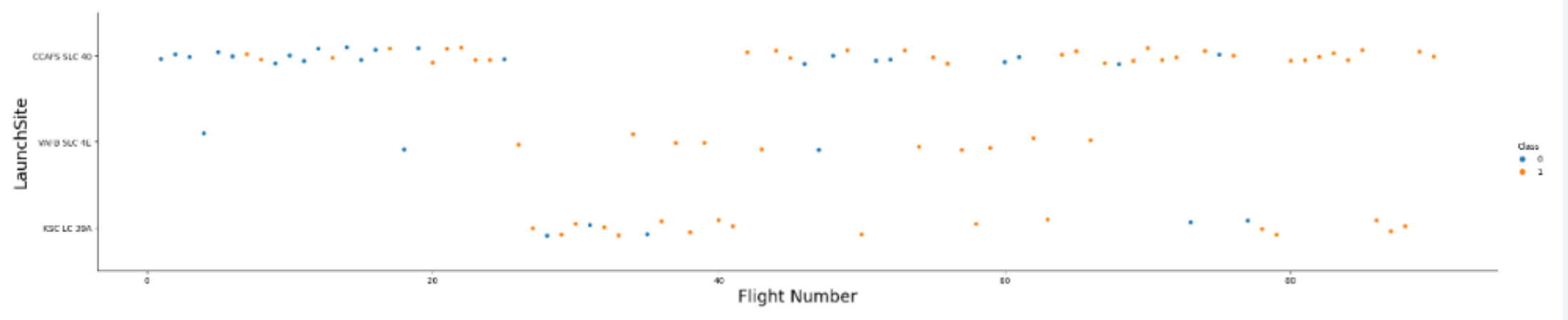
# Results

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


Section 2
Insights drawn from EDA

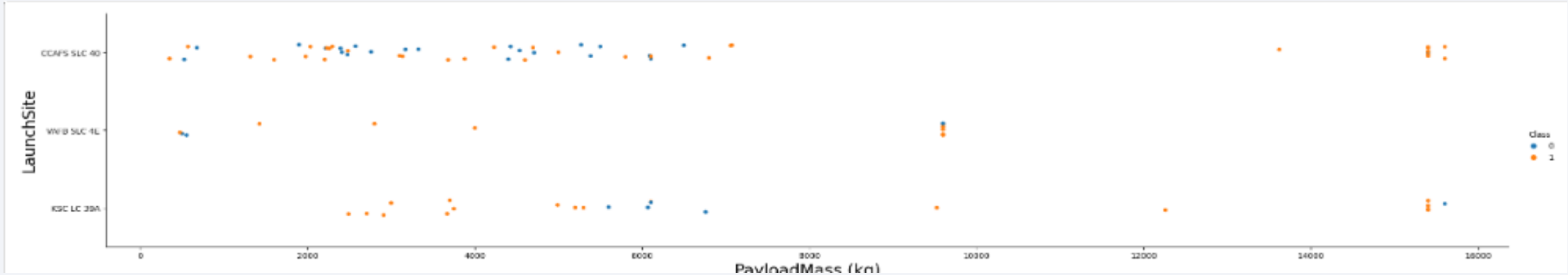
# Flight Number vs. Launch Site

- From the scatter plot of Flight Number vs. Launch Site we see that greater the number of flights at a launch site, greater is the success rate at the launch site.



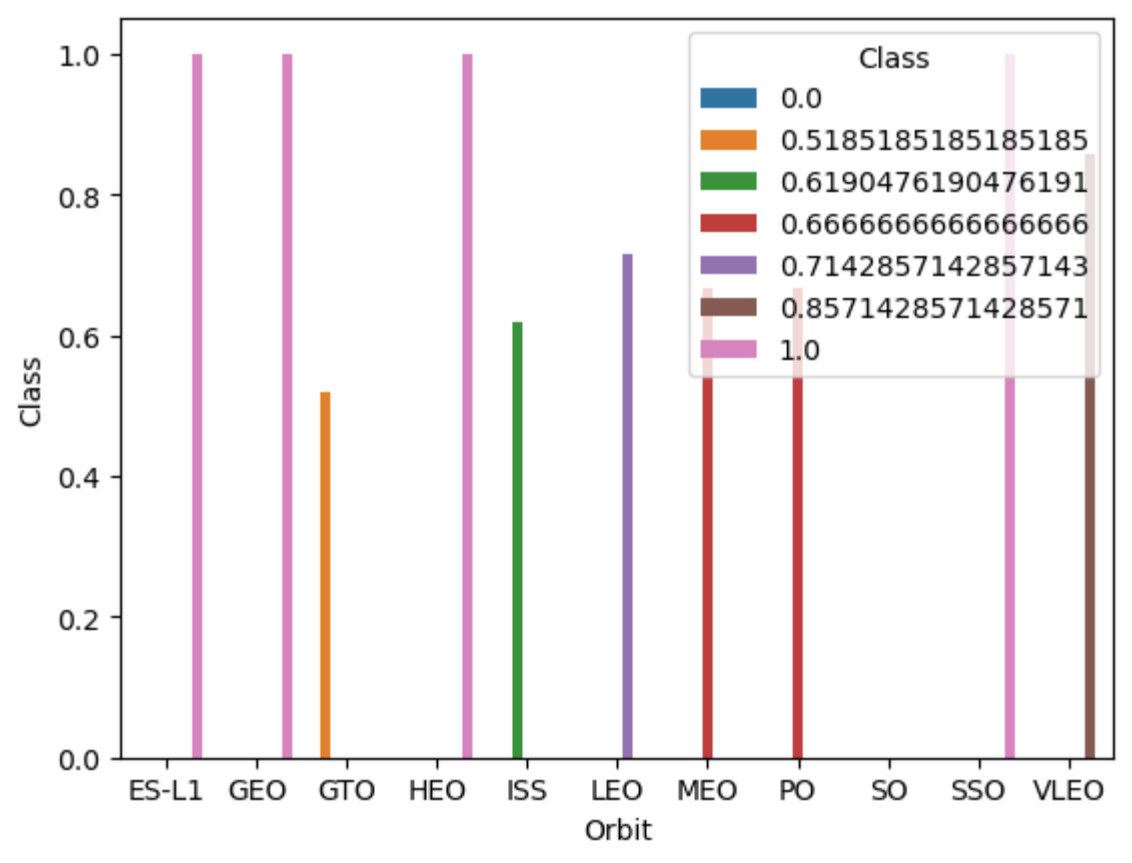
# Payload vs. Launch Site

- For CCSFS SLC 40, as payload mass increase the first stage is more likely to land.
- For VAFB-SLC launch site there are no rockets launched for payload > 10,000 kg
- For KDC LC 39A launch site there are no rockets launched for payload < 2,000 kg



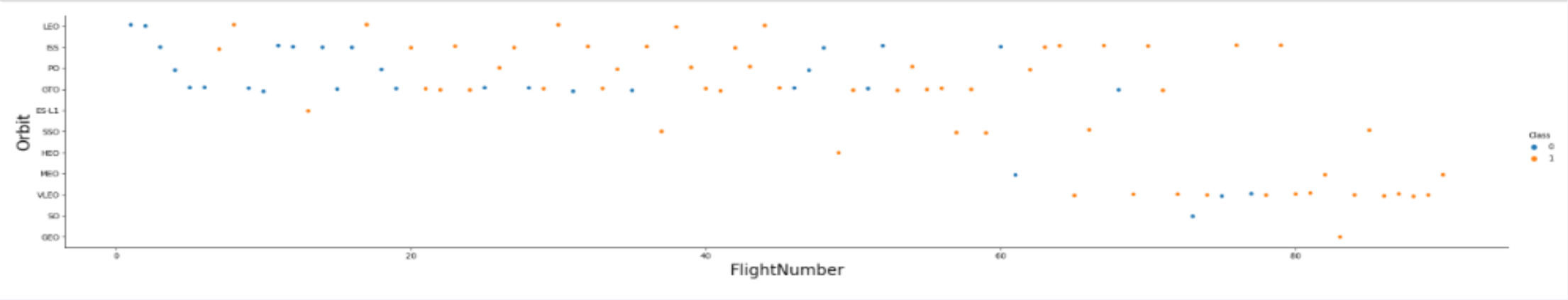
# Success Rate vs. Orbit Type

- From the bar chart, we see that ES-L1, GEO, HEO, and SSO orbits have the highest success rate.



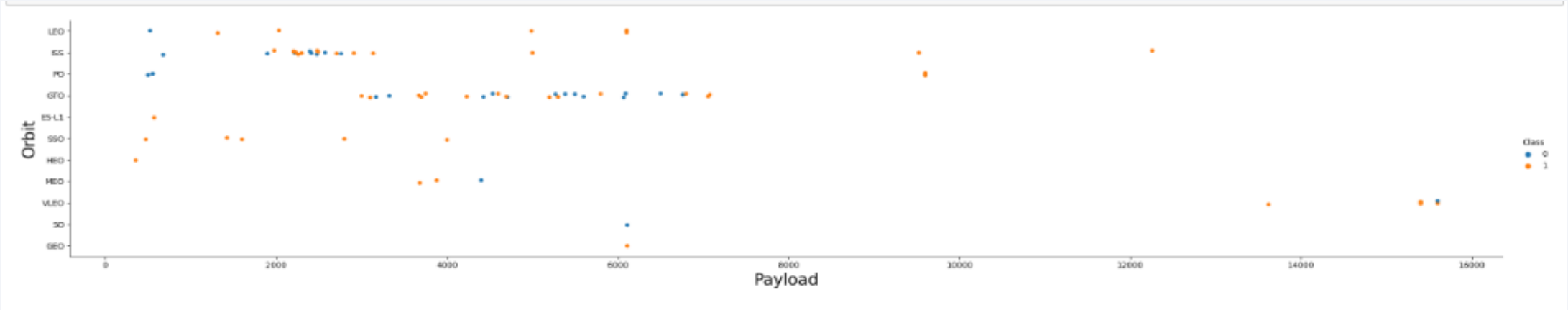
# Flight Number vs. Orbit Type

- From the scatter plot of Flight number vs. Orbit type, we see that LEO's success rate is related to flight number, whereas GEO's success rate seems to be independent of flight number.



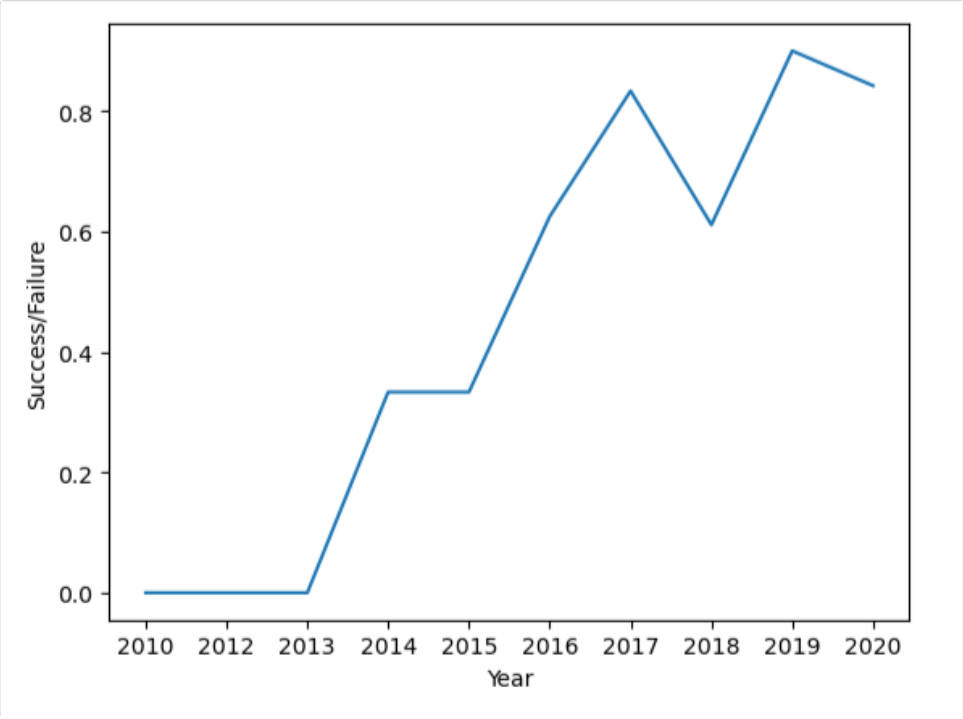
# Payload vs. Orbit Type

- We see that for heavier payloads, success rate increases for LEO, ISS and Polar orbits
- There seems to be no observable relation between GTO orbit and payload mass



# Launch Success Yearly Trend

- The Line Chart shows that the success rate increased from 2013 to 2020 with minor fluctuations throughout the period.
- We see a major plunge in the success rate for the year 2018.



# All Launch Site Names

- We use the keyword DISTINCT to select unique launch sites from the table.

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL;
```

```
 * ibm_db_sa://xcg80731:***@b
Done.
```

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

# Launch Site Names Begin with 'CCA'

- The keyword LIKE is used to specify that the launch site name begins with CCA.
- The limit keyword is used to query only 5 records.

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

```
 * ibm_db_sa://xcg80731:***@ba
Done.
```

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

# Total Payload Mass

- The total payload mass for customer NASA (CSR) is 45,596 kg.
- The function SUM is used to obtain the sum of payload mass.
- The WHERE clause is used to specify that the customer should be NASA (CRS)

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

```
 * ibm_db_sa://xcg80731:***@ba9
Done.
```

| 1 |
|---|
| 45596 |

# Average Payload Mass by F9 v1.0

- • The average payload mass for rockets with booster version F9 v1.0 is 340 kg.
- The function AVG is used to calculate the average of payload mass.
- The WHERE clause is used to specify that the booster version should be F9 v1.0.

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Booster_Version LIKE 'F9 v1.0%';
```

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d5
Done.
```

| 1 |
| --- |
| 340 |

# First Successful Ground Landing Date

- The first successful ground landing occurred on 22 December 2015.
- The function MIN is used to find the earliest date.
- The WHERE clause is used to specify that the landing outcome is a successful ground pad landing.

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

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8f
Done.
```

| 1 |
| --- |
| 2015-12-22 |

# Successful Drone Ship Landing with Payload between 4000 and 6000

- The names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 – F9 FT B1022, F9 FT B1026, F9 FT B1021.2, F9 FT B1031.2
- The WHERE clause specifies that the landing was a success drone ship landing.
- The AND clause adds another condition that the payload mass is between 4000 and 6000.

```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (drone ship)'
    AND 4000 < PAYLOAD_MASS__KG_ < 6000;
```

| booster_version |
|---|
| F9 FT B1021.1 |
| F9 FT B1023.1 |
| F9 FT B1029.2 |
| F9 FT B1038.1 |
| F9 B4 B1042.1 |
| F9 B4 B1045.1 |
| F9 B5 B1046.1 |

# Total Number of Successful and Failure Mission Outcomes

- Total Number of Successful Mission Outcomes – 99
- Total Number of Failure Mission Outcomes - 1
- The COUNT function gives the number of landings.
- The LIKE clause is used to specify that the outcome is a success or a failure.

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

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08
Done.
```

| mission_outcome | total_number |
|---|---|
| Failure (in flight) | 1 |
| Success | 99 |
| Success (payload status unclear) | 1 |

# Boosters Carried Maximum Payload

- The WHERE clause is used to specify that the booster version with the maximum payload mass is to be selected
- The MAX function gives the maximum payload mass.

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

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

# 2015 Launch Records

- The SUBSTR function selects the year out of the date.
- The WHERE clause specifies that the date should be 2015 and the landing outcome should be a drone ship failure.

```
%%sql
SELECT LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE Landing__Outcome = 'Failure (drone ship)'
    AND YEAR(DATE) = 2015;
```

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6
Done.
```

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

# 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
- The GROUP BY clause was used to group the landing outcomes and the order by clause was used to sort the grouped landing outcome in descending order

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

| landing__outcome | total_number |
|---|---|
| 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 |

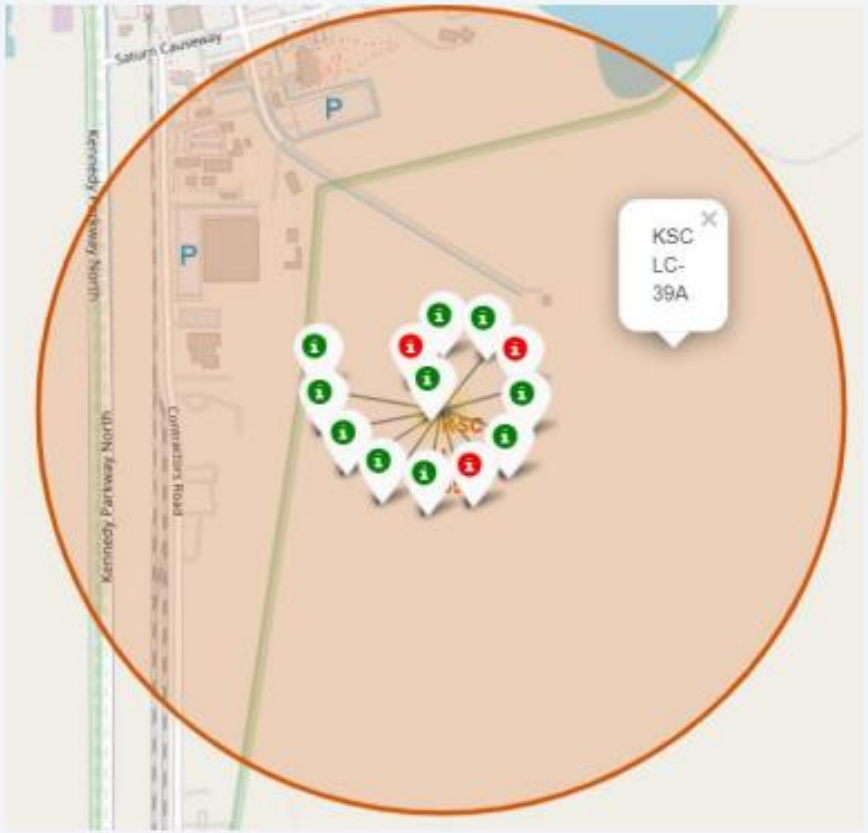
Section 3

# Launch Sites Proximities Analysis

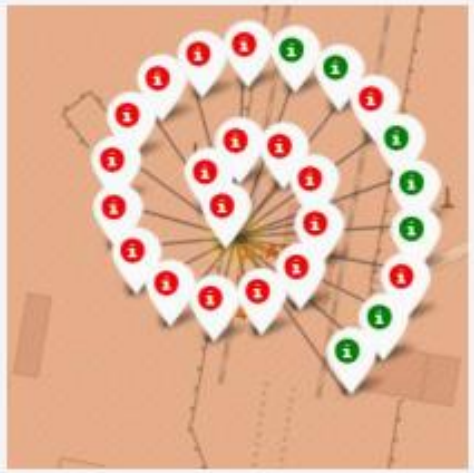
# SpaceX Launch Sites on the Global Map


- The SpaceX Launch Sites are on the coasts Florida and California in America
- Only the launch site VAFB SLC – 4E is in California; the rest are in Florida

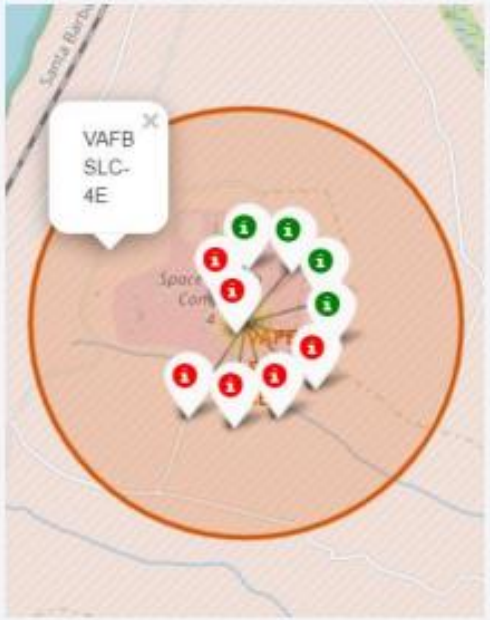
# Success/Failed Launches for Each Launch Site




Florida Launch Sites

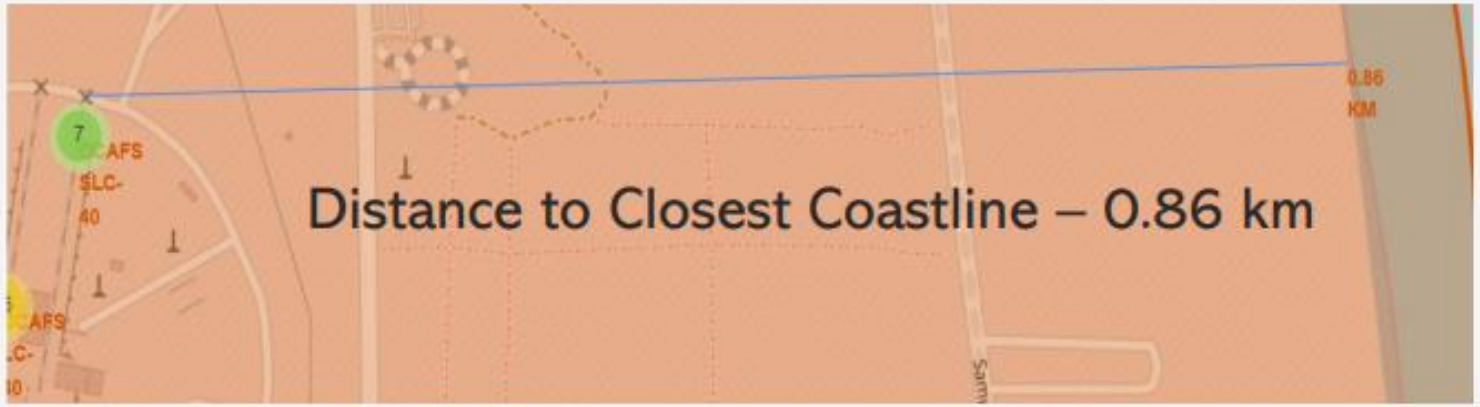
CCAFS LC-40



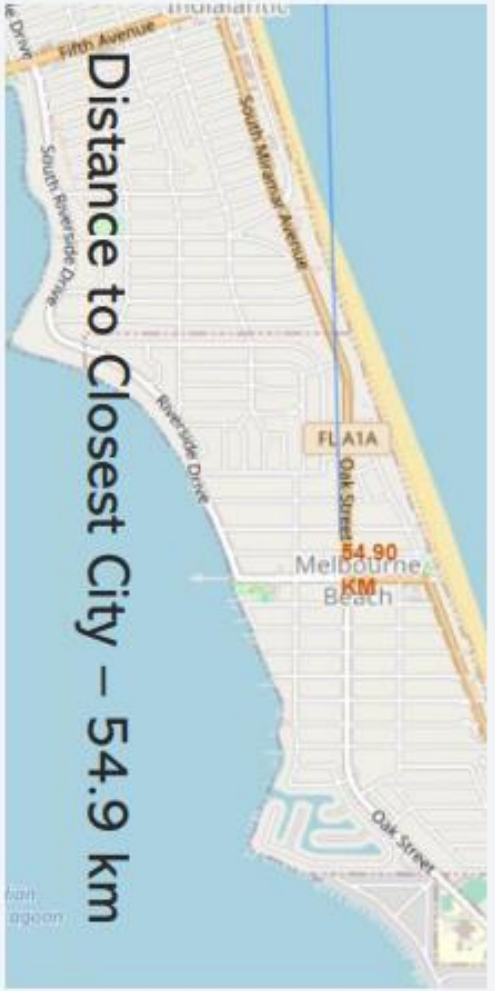

California Launch Site

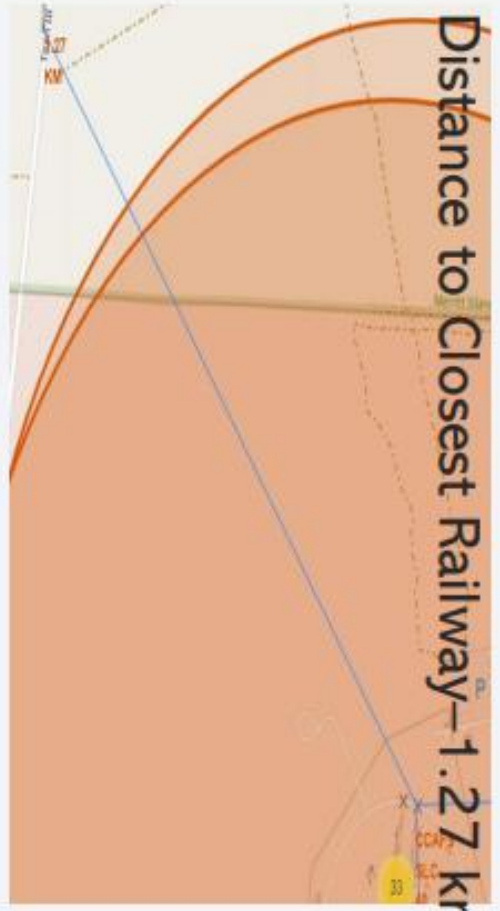
Red Markers – Failure; Green Markers - Success

# Launch Site Proximity From Landmarks

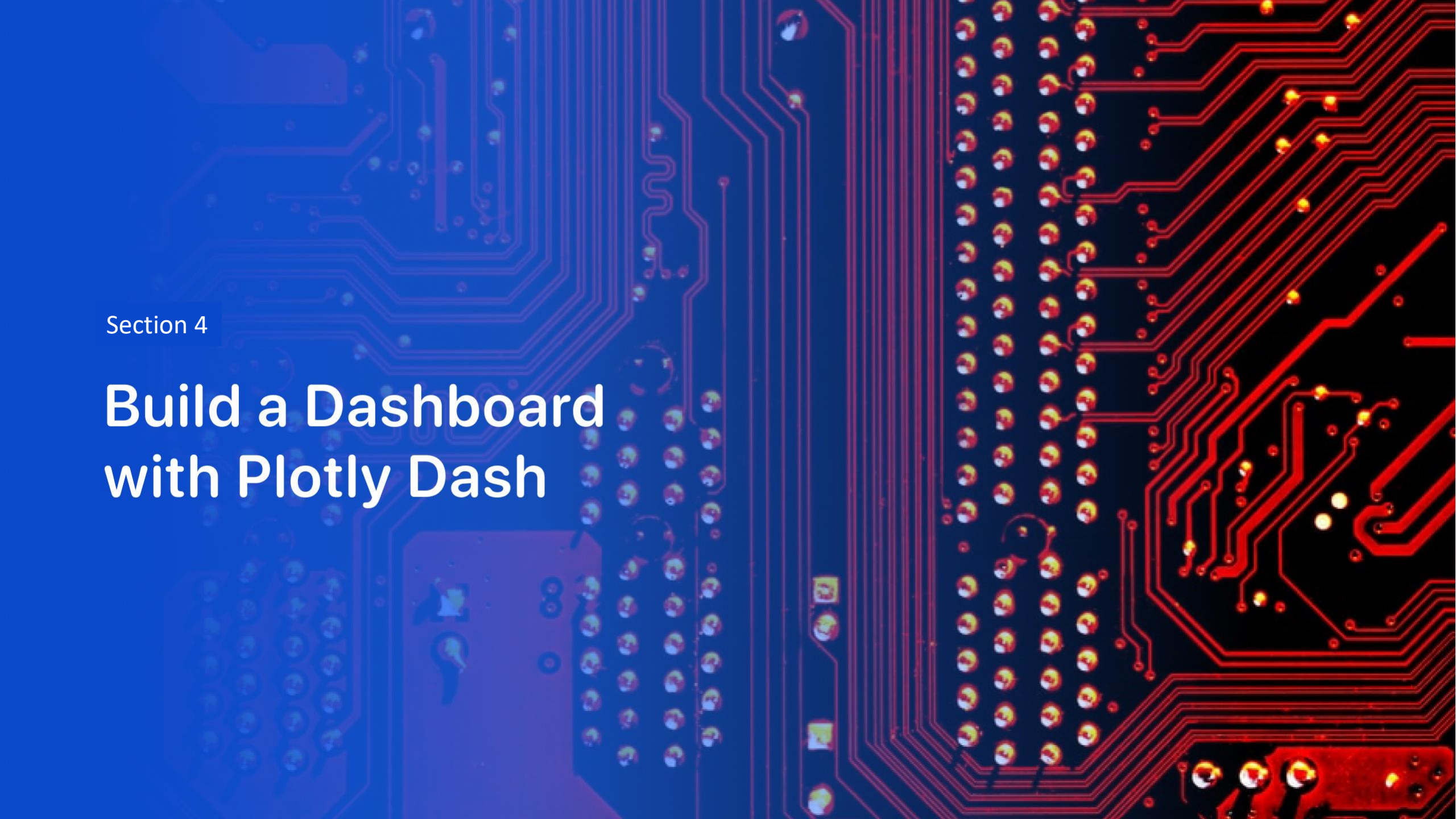




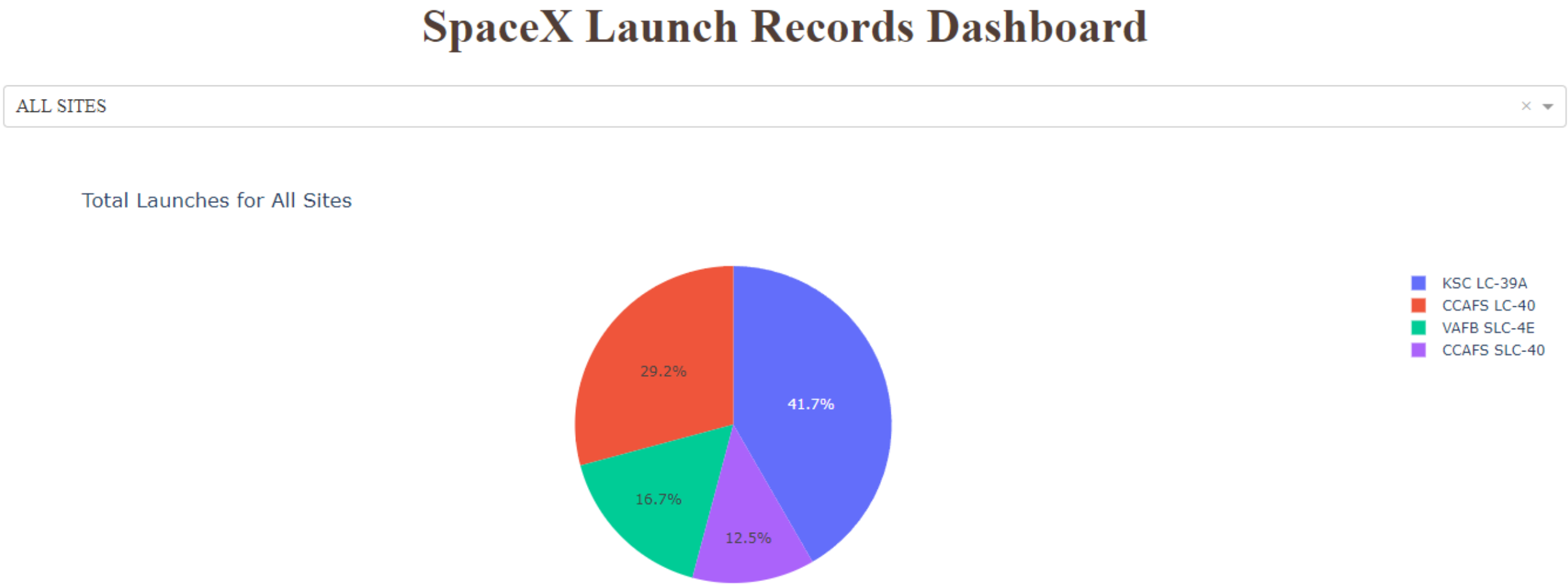



Are launch sites in close proximity to railways? – Yes (1.27 km)
Are launch sites in close proximity to highways? Yes (0.58 km)
Are launch sites in close proximity to coastline? – Yes (0.86 km)
Do launch sites keep certain distance away from cities? – Yes (54.9 km from Melbourne Beach)

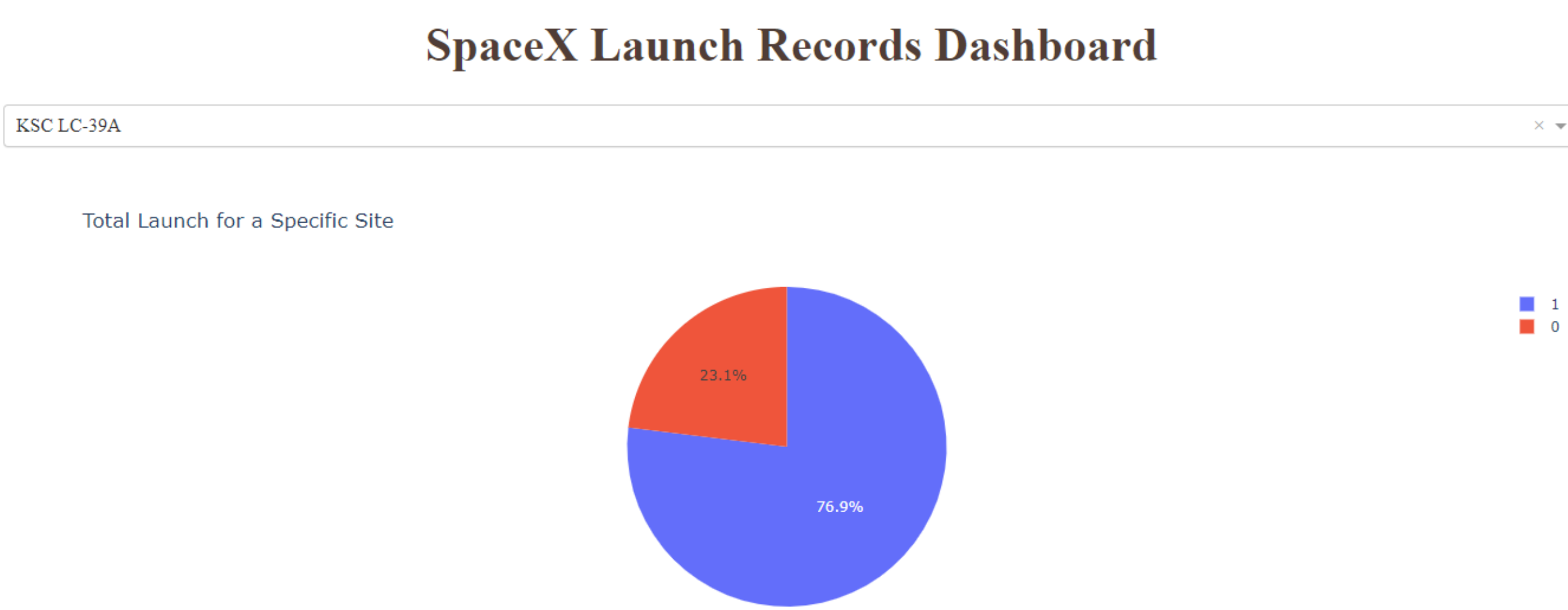

Section 4
Build a Dashboard with Plotly Dash

# Pie Chart of Total Success Launches by all Site



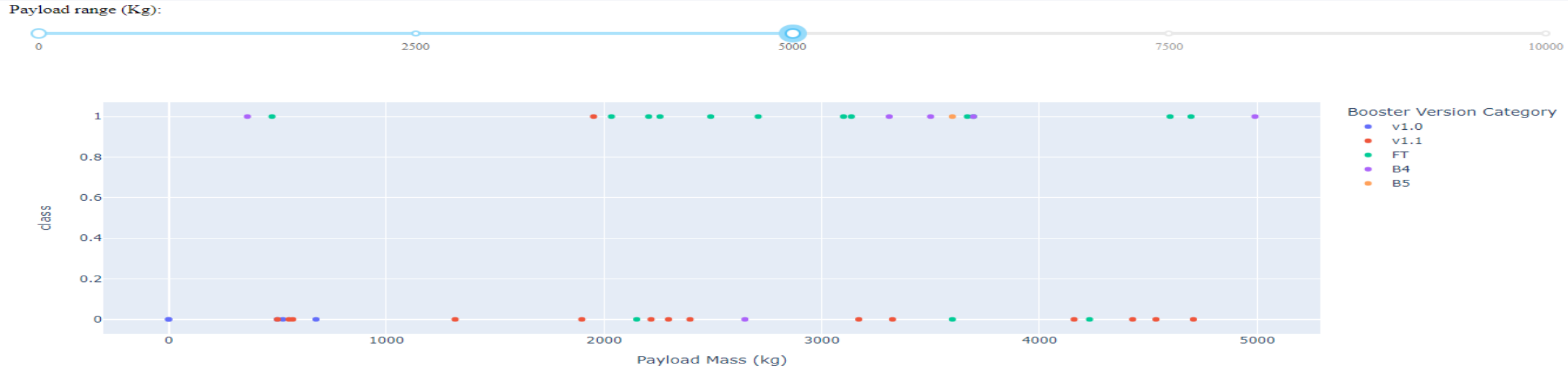

Launch Site KDC LC-39 A has the Greatest Launch Success Rate

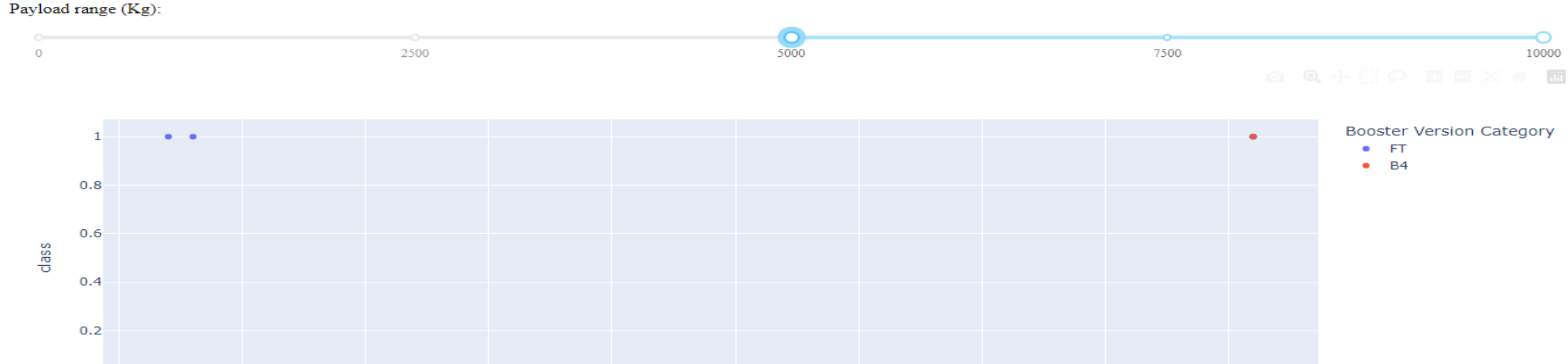
# Pie Chart of Success/Failure Launch Rate of KDC LC-39 A




We see that the launch site KSC LC-39A achieved a launch success rate of 76.9%

# Payload comparation between 0-5000 KG and 5000-10000 kg



Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

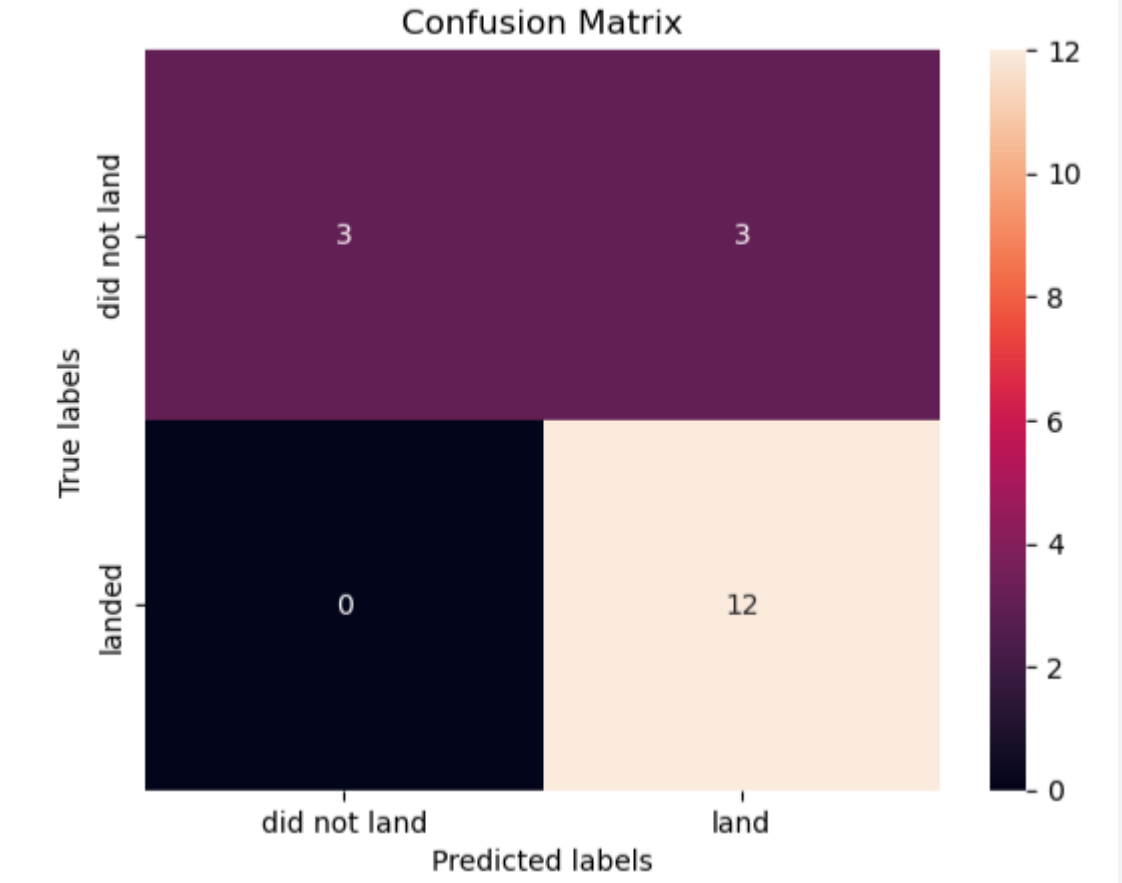
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}
bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score 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 0.875
Best params is : {'criterion': 'gini', 'max_depth': 12, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10,
'splitter': 'best'}
```

We can see that the decision tree classifier has the highest classification accuracy of 0.875

# Confusion Matrix

- Best Model – Decision Tree Classifier
- The confusion matrix for the Decision Tree Classifier shows that the classifiers gives a high number of false positives, i.e., the rocket did not actually land successfully but the classifiers predicts that it landed successfully.



# Conclusions

- The greater the number of flights at a launch site, greater is the success rate at the launch site.
- ES-L1, GEO, HEO, and SSO orbits have the highest success rate.
- For heavier payloads, success rate increases for LEO, ISS and Polar orbits.
- Launch success rate increased from 2013 to 2020 with minor fluctuations throughout the period.
- Launch Site KDC LC-39 A has the Greatest Launch Success Rate
- The Decision Tree Classifier is the best machine learning algorithm to predict SpaceX Falcon 9 rocket's first stage will land successfully.

# Appendix

- GitHub Link from the repository utilized on this project with all code and notebooks developed: IBM-DataScienceFiles/Applied Data Science Capstone at main · Falkenus/IBM-DataScienceFiles (github.com)

# Thank you!