IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

NALAN BESLI
NOVEMBER 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 in screenshots
  - Predictive Analytics result

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, and much of the savings is because Space X can reuse the first stage. In this project, we will be to determine the price of each launch which depends on whether the first stage would land successfully to be reused. Using machine learning models and public data on Falcon 9, we will predict if SpaceX will reuse the first stage.

- Problems you want to find answers
  - What are the main factors that determine if the rocket will land successfully?
  - What are the effects of each relationship of the rocket variables on the success or failure of a landing?
  - What operating conditions need to be in place to achieve the best landing success rate?

Section 1

Methodology

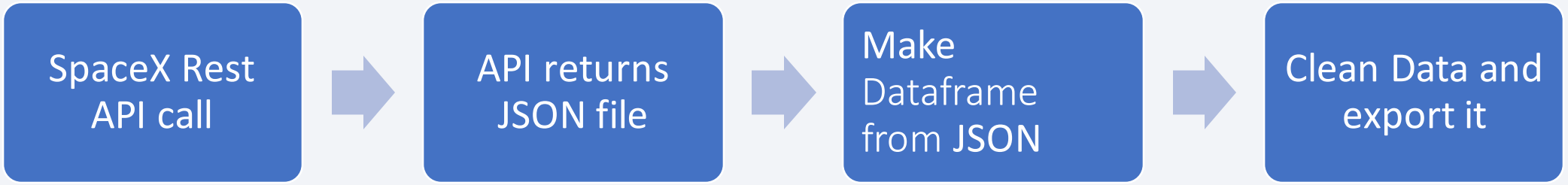
Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- 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 dataframe 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 dataframe for future analysis.

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_a
```

We should see that the request was successfull with the 200 status response code

In [10]:
```
response.status_code
```

Out[10]: 200

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

In [11]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Now let's start requesting rocket launch data from SpaceX API with the following URL:

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

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.

In [28]:
```
# Calculate the mean value of PayloadMass column
PayloadMass_mean = data_falcon9.PayloadMass.mean()
# Replace the np.nan values with its mean value
data_falcon9["PayloadMass"] = data_falcon9["PayloadMass"].replace(np.nan, PayloadMass_mean)

data_falcon9.isnull().sum()
```

Code link

Data Collection - Scraping

- We scraped Falcon 9 launch data from Wikipedia with BeautifulSoup.
- We parsed the table and converted it into pandas dataframe.

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

In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url).text
```

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

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [14]:
```
# Use soup.title attribute
print(soup.title)
```

```
List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

Extract all column/variable names from the HTML table header

```
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

temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

Create a data frame by parsing the launch HTML tables

[Code link]

Data Wrangling

- We calculated the number of launches at each site and the number and occurrence of each orbits.
- We transformed variables into categorical variables 1 and 0 and exported the results to cvs.

```
df.LaunchSite.value_counts()
```

```
CCAFS SLC 40     55
KSC LC 39A       22
VAFB SLC 4E      13
Name: LaunchSite, dtype: int64
```

```
df.Orbit.value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

```
landing_outcomes = df.Outcome.value_counts()
landing_outcomes
```

```
True ASDS       41
None None       19
True RTLS       14
False ASDS       6
True Ocean       5
False Ocean      2
None ASDS        2
False RTLS       1
Name: Outcome, dtype: int64
```

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

```
landing_class = []

for key, value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

Code link

EDA with Data Visualization

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend with scatter plots, bar graphs and line graphs.
- **Scatter plots** show relationship between variables. This relationship is called the correlation.
- **Bar graphs** show the relationship between numeric and categoric variables.
- **Line graphs** show data variables and their trends. Line graphs can help to show global behavior and make prediction for unseen data.

Code link

EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data.
- **We wrote queries for:**

- Displaying the names of the unique launch sites 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 achieved.
- 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 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, 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. Link to code

Code link

Build an Interactive Map with Folium

- 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, 0 for failure, and 1 for success.
- Using the color-labeled markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing, we can see which launch sites have relatively high success rate.
- We calculated the distances between a launch site to key locations like railway, highway, coastway, city, and plot a line between them.

Code link

Build a Dashboard with Plotly Dash

- Built an interactive dashboard with Plotly dash.
- Pie charts showing the total launches by a certain sites
- Scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

Code link

Predictive Analysis (Classification)

- Data preparation

  Load the data using numpy and pandas, normalized the data, split data into training set and testing set.

- Model preparation

  Select machine learning algorithms, set parameters for each algorithm to GridSearchCV than train GridSearchModel models with training dataset.

- Model evaluation

  Get best hyperparameters for each type of model than compute accuracy for each model with test dataset than plot confusion matrix.

- Model comparison

  Compare models according to their accuracy then choose the model with the best accuracy.

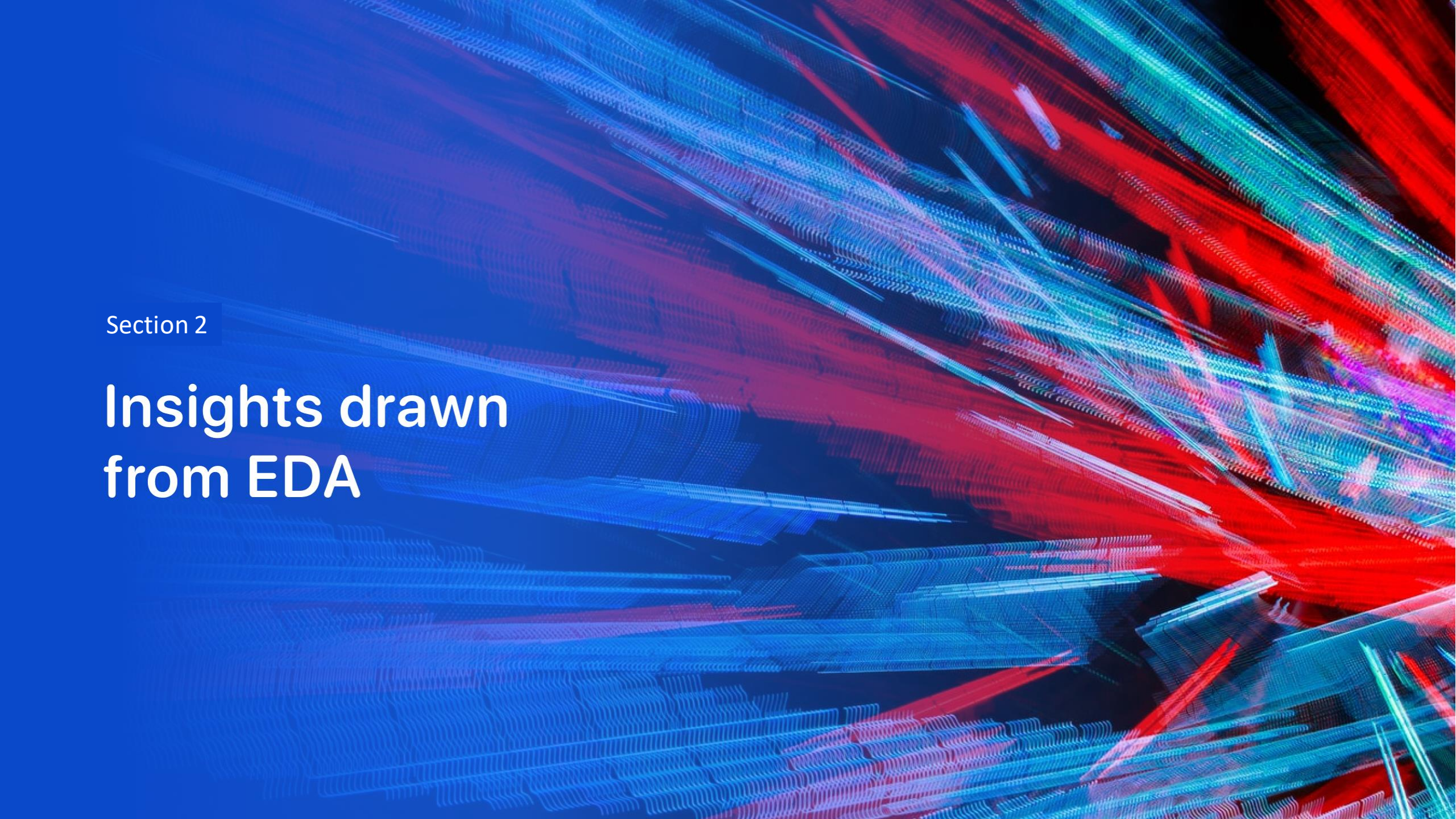
Code link

Results

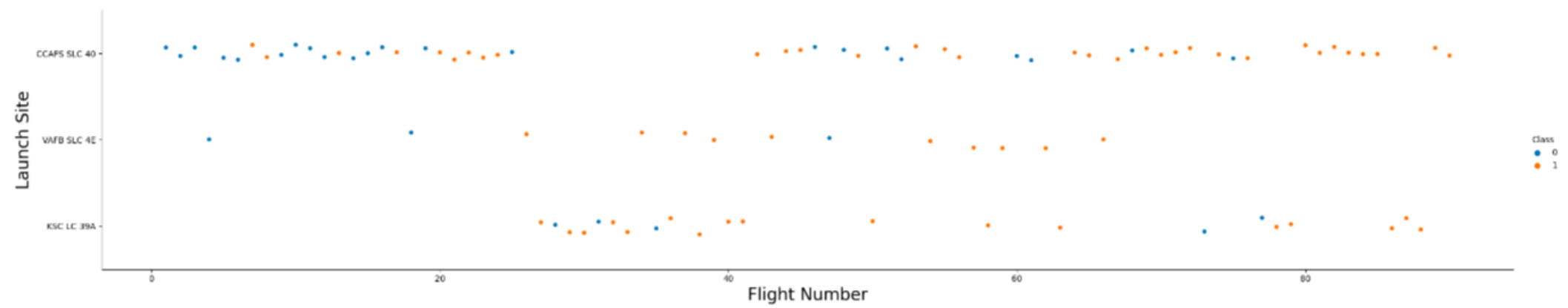
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

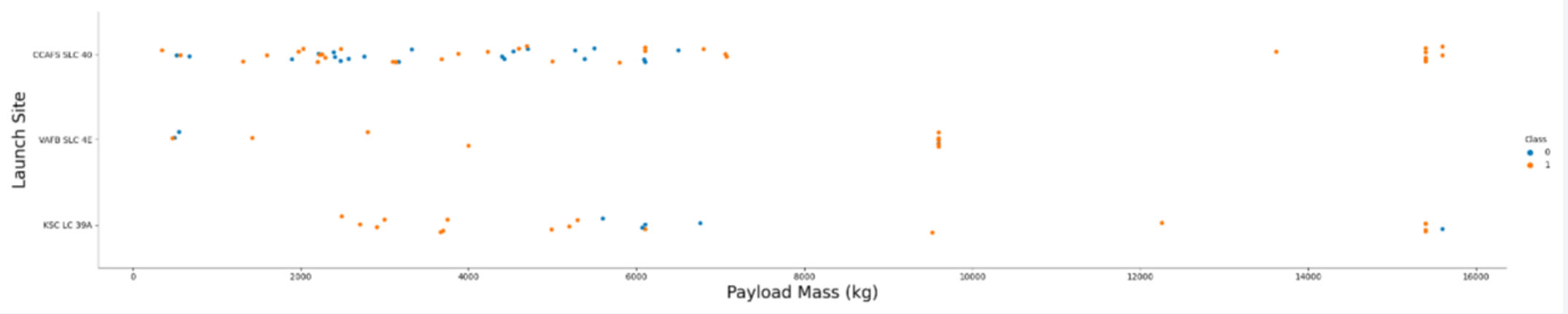


Flight Number vs. Launch Site



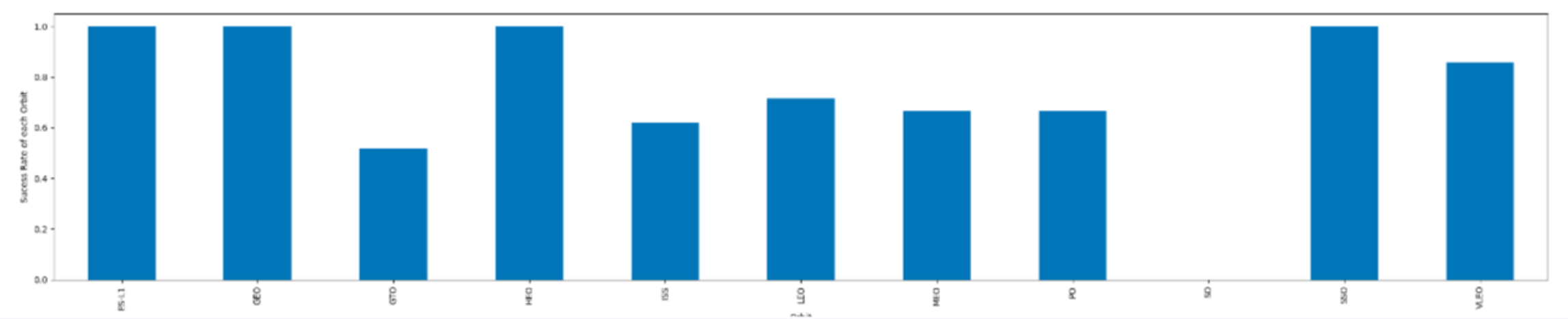
- From the plot, we can see that the larger the flight number at a launch site, the greater the success rate.

Payload vs. Launch Site



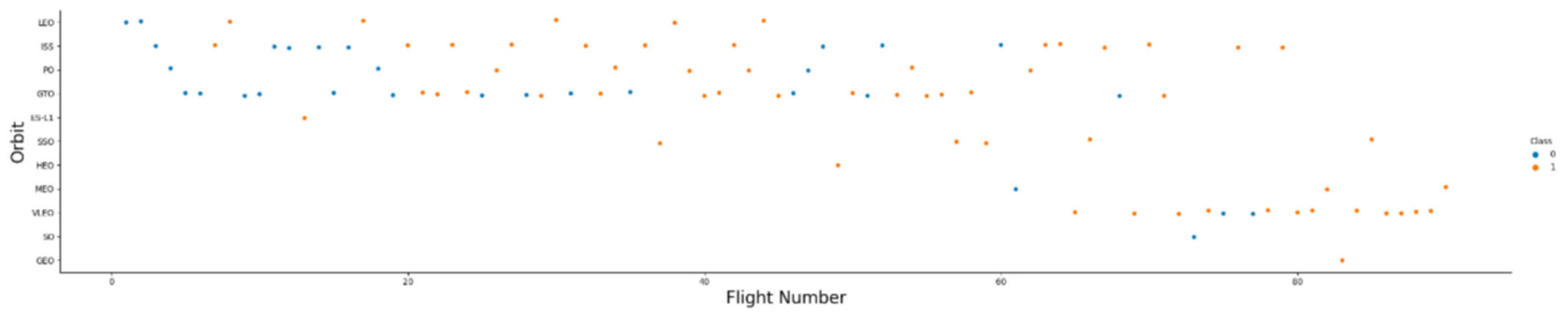
- Depending on the launch site, a heavier payload may be a consideration for a successful landing.

Success Rate vs. Orbit Type



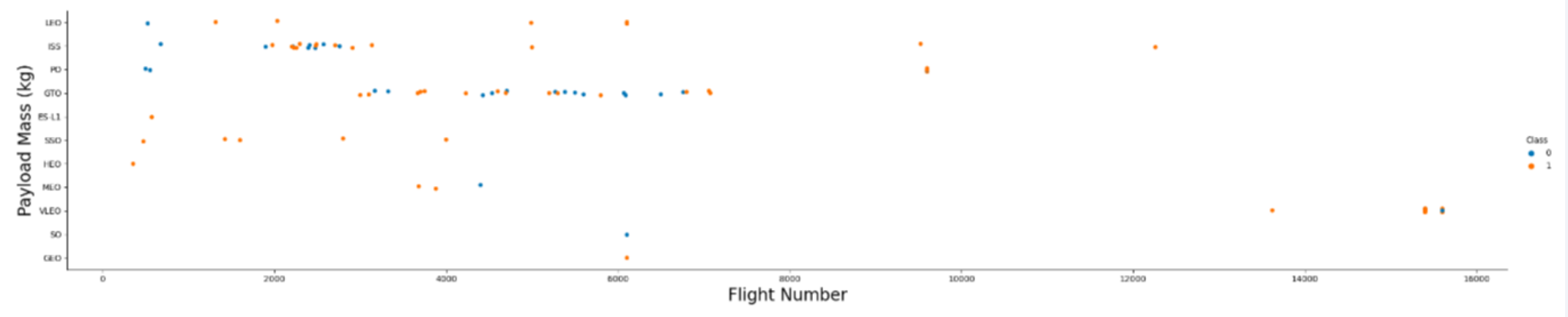
- This plot shows the success rate for different orbit types. We can see ES-L1, GEO, HEO, SSO have the best success rate.

Flight Number vs. Orbit Type



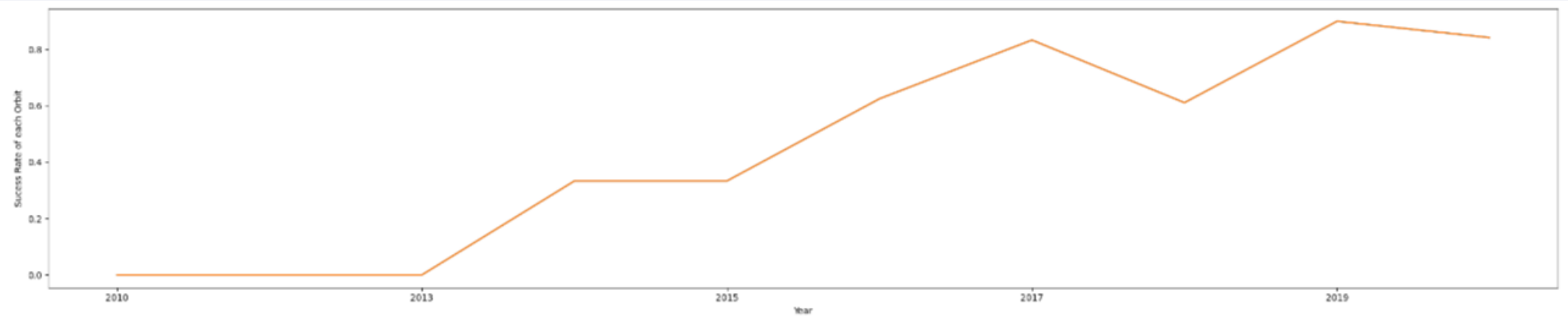
- We can see that the success rate increases with the number of flights for the LEO orbit. For some orbits like GTO, there is no relation between the success rate and the number of flights.

Payload vs. Orbit Type



- The weight of the payloads can have a great influence on the success rate of the launches in certain orbits. We can see that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.

Launch Success Yearly Trend



- Since 2013, we can see an increase in the Space X Rocket success rate.

All Launch Site Names

```
%sql SELECT DISTINCT(launch_site) FROM spacextbl
```

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

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

- The use of the key word **DISTINCT** is showing only unique launch sites from the SpaceX data.

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM spacextbl WHERE launch_site LIKE "CCA%" LIMIT 5
```

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

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

- The query above displays 5 records where launch sites begin with `CCA`.

Total Payload Mass

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM spacextbl WHERE customer = "NASA (CRS)"
```

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

SUM(PAYLOAD_MASS__KG_)
45596

- The total payload carried by boosters from NASA is shown using the query above.

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM spacextbl WHERE Booster_Version = "F9 v1.1"
```

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

AVG(PAYLOAD_MASS__KG_)
2928.4

- This query returns the **average** of all payload masses where the booster version contains the substring F9 v1.1.

First Successful Ground Landing Date

```
%sql SELECT MIN(Date) FROM spacextbl WHERE "Landing _Outcome" = 'Success (ground pad)'
```

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

MIN(Date)
01-05-2017

- With this query, we select the oldest successful landing with **WHERE** clause and **MIN** function .

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT Booster_Version FROM spacextbl WHERE "Landing _Outcome" = 'Success (drone ship)' \
and PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000
```

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

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

- This query returns the booster version where landing was successful and payload mass is **between** 4000 - 6000 kg. The **WHERE** and **AND** clauses filter the dataset.

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) FROM spacextbl GROUP BY Mission_Outcome
```

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

Mission_Outcome	COUNT(Mission_Outcome)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- Sorted results with **GROUP BY** if the Mission Outcome was a success or a failure. The **COUNT** function counts records filtered.

Boosters Carried Maximum Payload

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM spacextbl WHERE PAYLOAD_MASS__KG_ = \
(SELECT MAX(PAYLOAD_MASS__KG_) FROM spacextbl)
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

- Used subquery to filter data by returning only the heaviest payload mass with **MAX** function.

2015 Launch Records

```
%sql SELECT substr("DATE", 4, 2) AS MONTH, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL\
WHERE "LANDING _OUTCOME" = 'Failure (drone ship)' and substr("DATE",7,4) = '2015'
```

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

MONTH	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

- This query returns month, booster version, launch site where landing was unsuccessful and landing date took place in 2015. Substr function process date in order to take month or year. Substr(DATE, 4, 2) shows **month**. Substr(DATE,7, 4) shows **year**.

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

```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE '%Success%'\
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING _OUTCOME") DESC
```

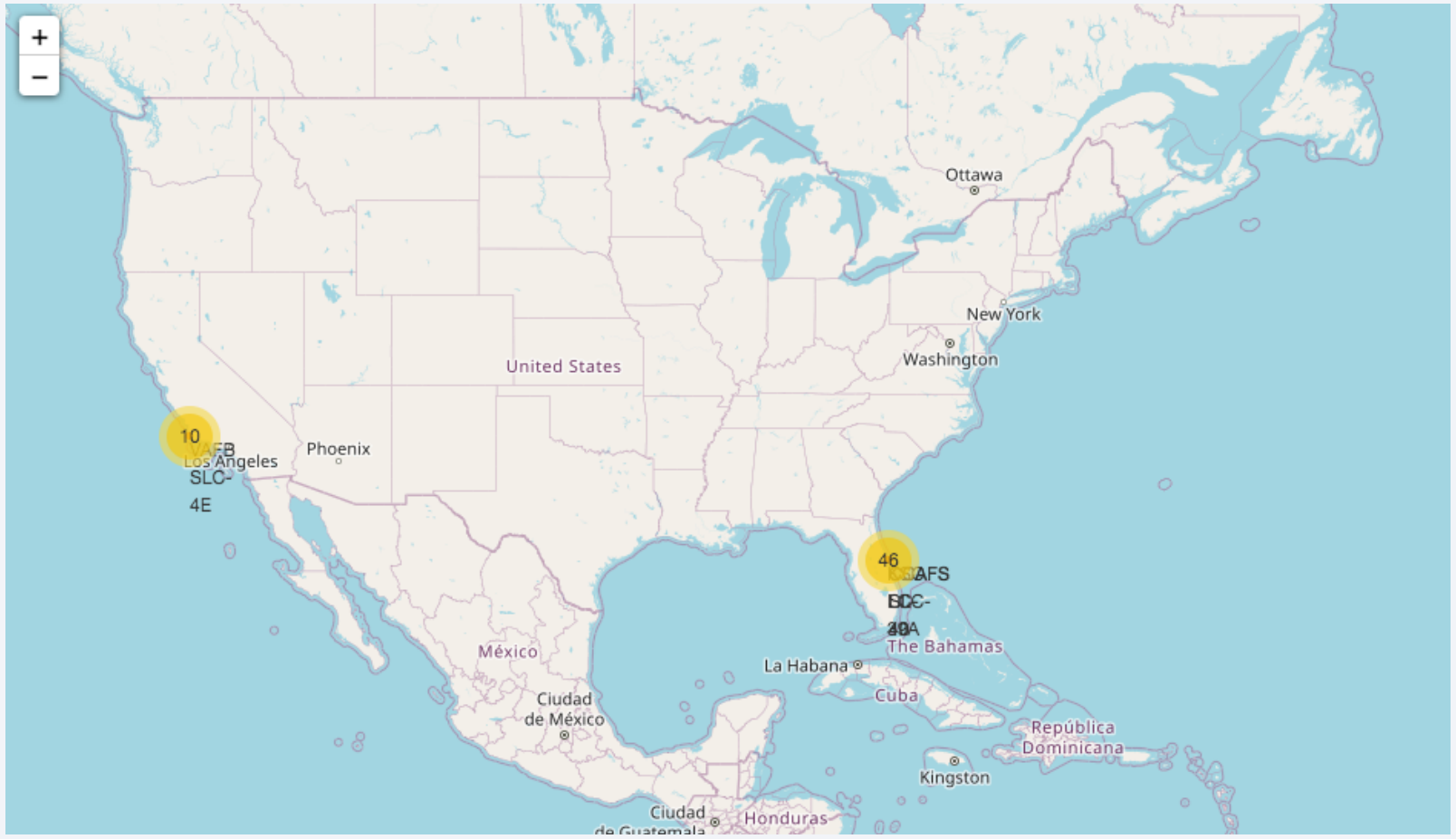
```
 * sqlite:///my_data1.db
Done.
```

Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

- This query returns landing outcomes and their count where mission was successful and date is between 04/06/2010 and 20/03/2017. The **GROUP BY** clause groups results by landing outcome and **ORDER BY COUNT DESC** shows results in decreasing order.

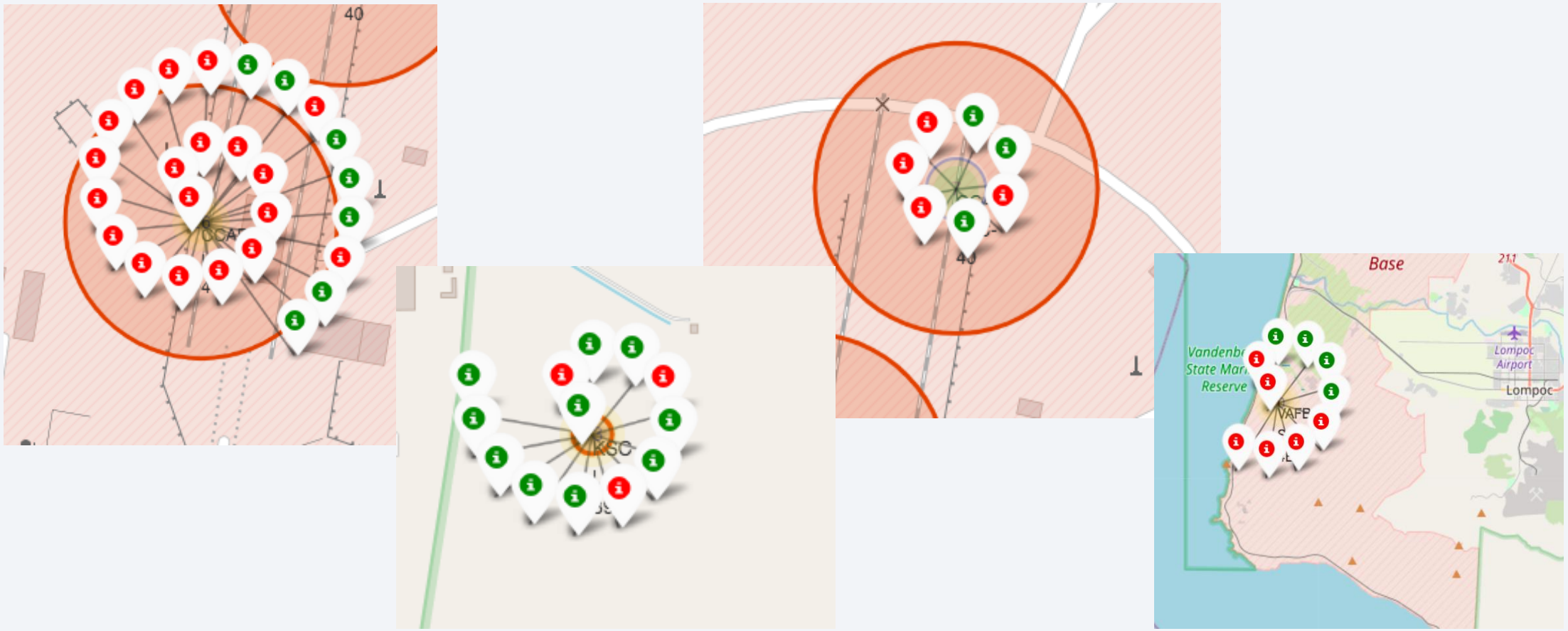

Section 3
Launch Sites
Proximities Analysis

All launch sites global map markers



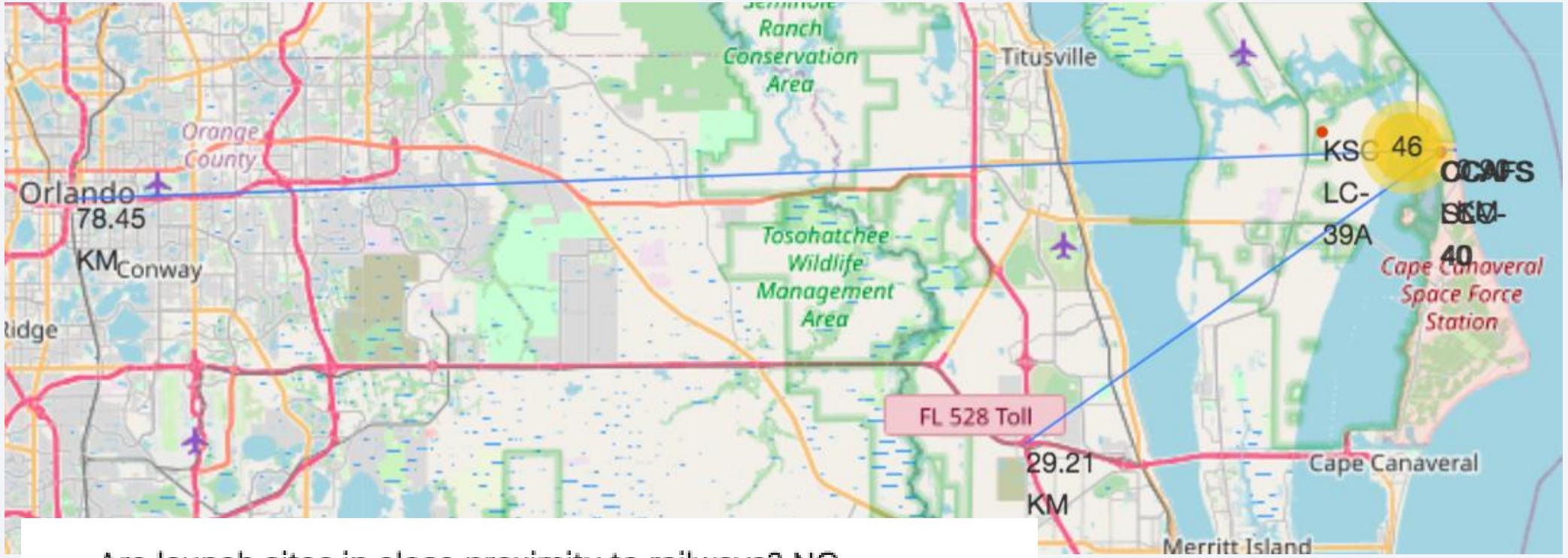
- We can see that Space X launch sites are located on the coast of the United States.

Markers showing launch sites with color labels

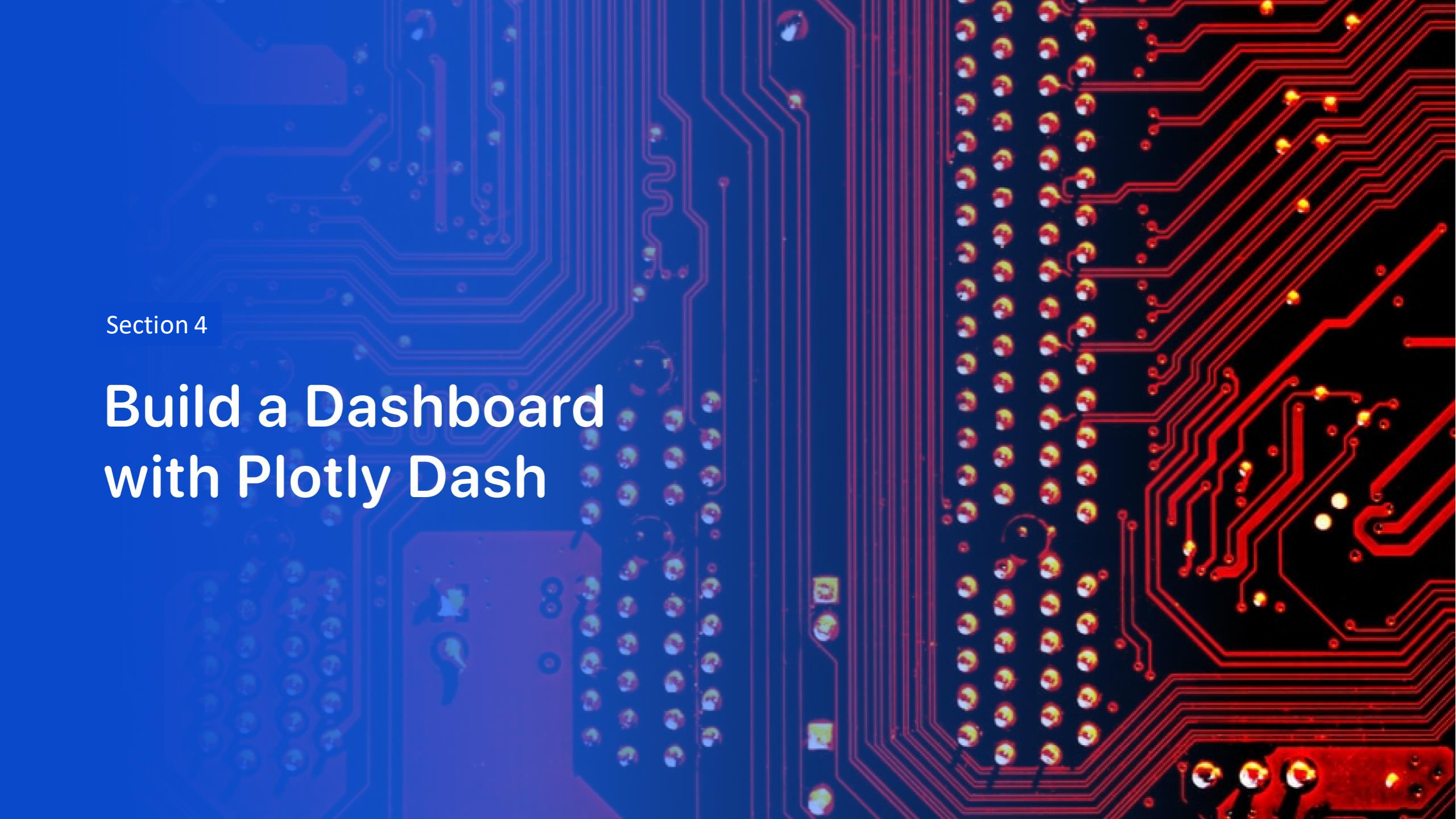


- Green marker represents successful launches. Red marker represents unsuccessful launches.

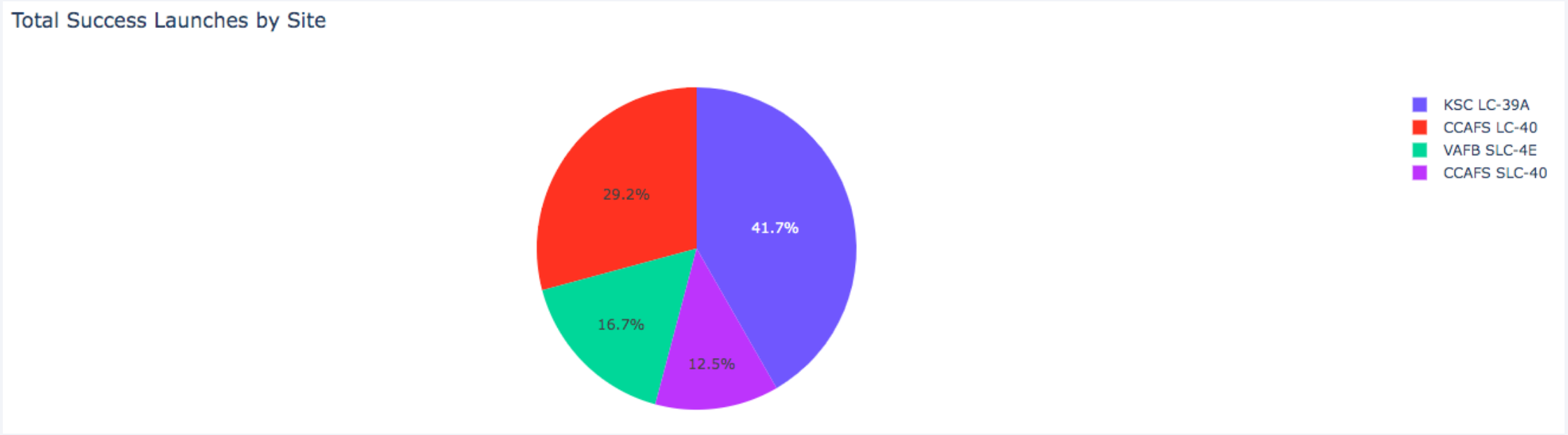
Launch Site distance to landmarks



- 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

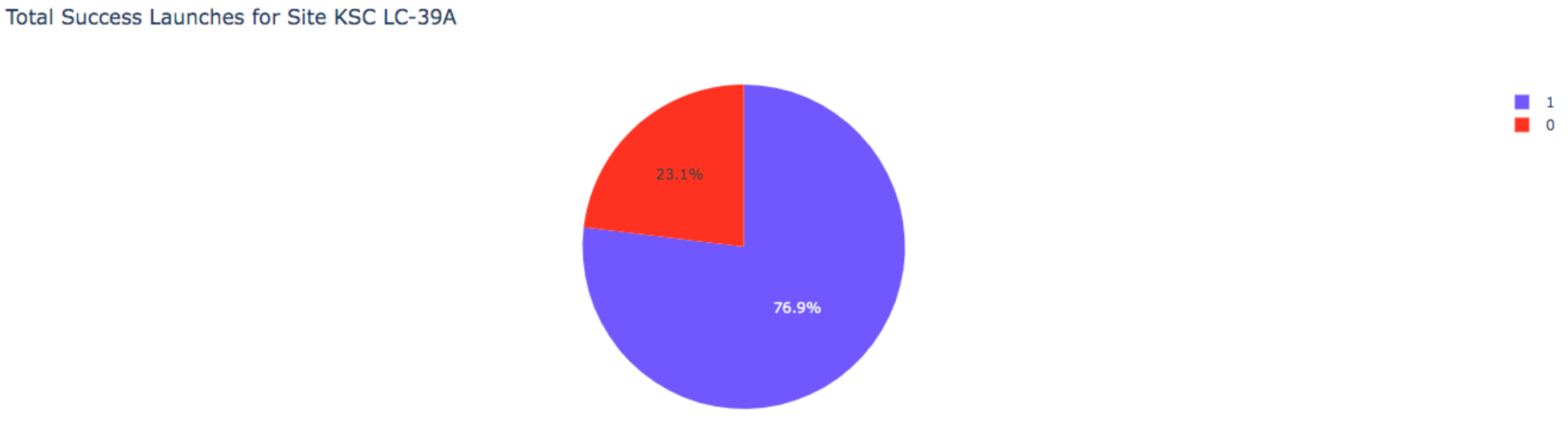

Section 4
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage by each launch site



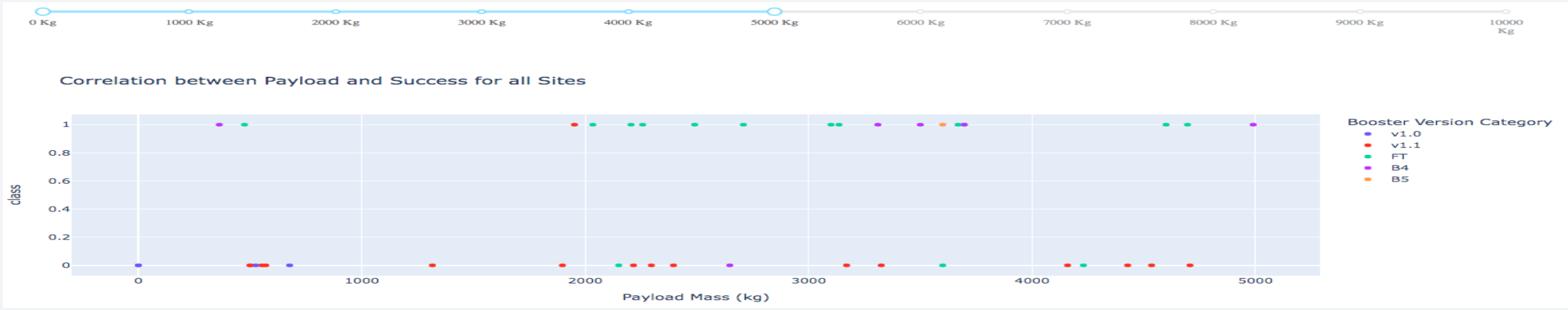
- We can see that KSC LC-39A has the best success rate.

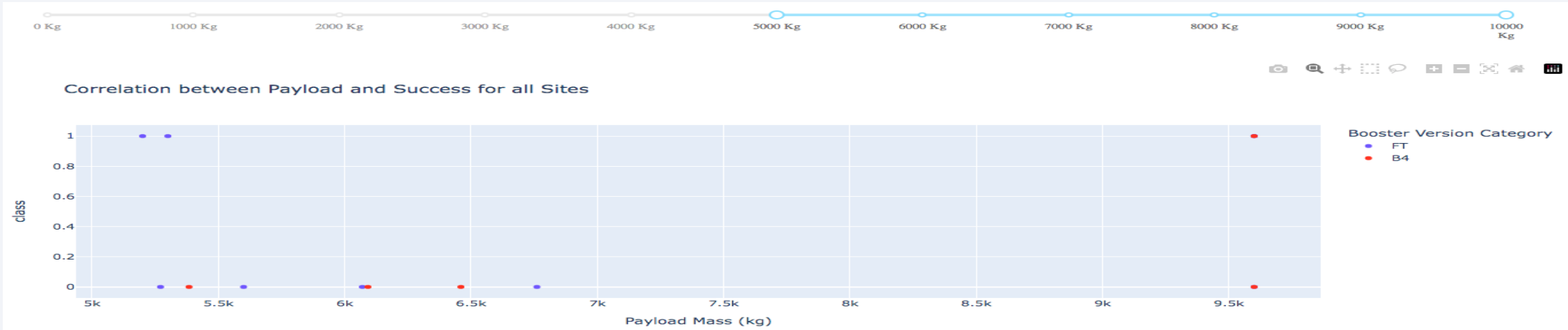
Pie chart showing the Launch site with the highest launch success ratio



- We can see that KSC LC-39A has 76.9% success rate and 23.1% failure rate.

Scatter plot of Payload vs Launch Outcome for all sites





- Low weighted payloads have a better success rate than the heavy weighted payloads.

Section 5

Predictive Analysis (Classification)

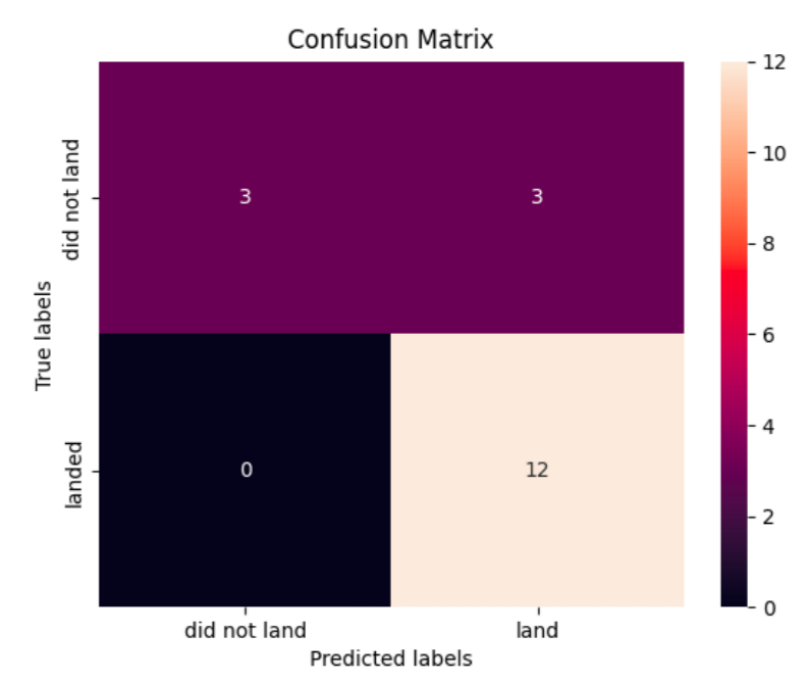
Classification Accuracy

```
methods = ['Logreg','Svm','Tree','Knn']
accs_train = [logreg_cv.best_score_, svm_cv.best_score_, tree_cv.best_score_, knn_cv.best_score_]
accs_test = [accuracy_lr_test_data, accuracy_svm_test_data, accuracy_tree_test_data, accuracy_knn_test_data]

dict_meth_accs = {}

for i in range(len(methods)):
    dict_meth_accs[methods[i]] = [accs_train[i], accs_test[i]]

df = pd.DataFrame.from_dict(dict_meth_accs, orient='index')
df.rename(columns={0: 'Accuracy Train', 1: 'Accuracy Test'}, inplace = True)

df_sorted_train = df.sort_values(by = ['Accuracy Train'], ascending=False)
df_sorted_train
```

	Accuracy Train	Accuracy Test
Tree	0.885714	0.888889
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333

- For accuracy test, all methods performed similar. We could get more test data to decide between them. But the decision tree classifier is the model with the highest classification accuracy

Confusion Matrix



- This is the confusion matrix for the decision tree classifier but the confusion matrices for all methods were identical .
- The main problem of these models are false positives (unsuccessful landing marked as successful landing).

Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Depending on the launch site, a heavier payload may consideration for a successful landing.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- Since 2013, we can see an increase in the Space X Rocket success rate.
- All methods performed similar but the decision tree classifier is the best machine learning algorithm for this task.

Thank you!