IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ezgi Tanyeli
11/29/2022

Outline

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

Executive Summary

Summary of methodologies

- Data Collection with API and Web Scraping
- Exploratory Data Analysis with Data Visualization
- Exploratory Data Analysis with SQL
- Interactive Map with Folium
- Dashboards with Plotly Dash
- Predictive Analysis

Summary of all results

- Exploratory Data Analysis results
- Interactive maps and dashboard
- Predictive results

Introduction

Project Background and Context

This project aims to predict whether the Falcon 9 first stage will have a successful landing.

The below questions have been discussed in this Project with further analysis.

What are the main characteristics of a successful or failed landing?	What are the effects of each relationship of the rocket variables on the success or failure of a landing?	What are the conditions which will allow SpaceX to achieve the best landing success rate?

Section 1

Methodology

Methodology

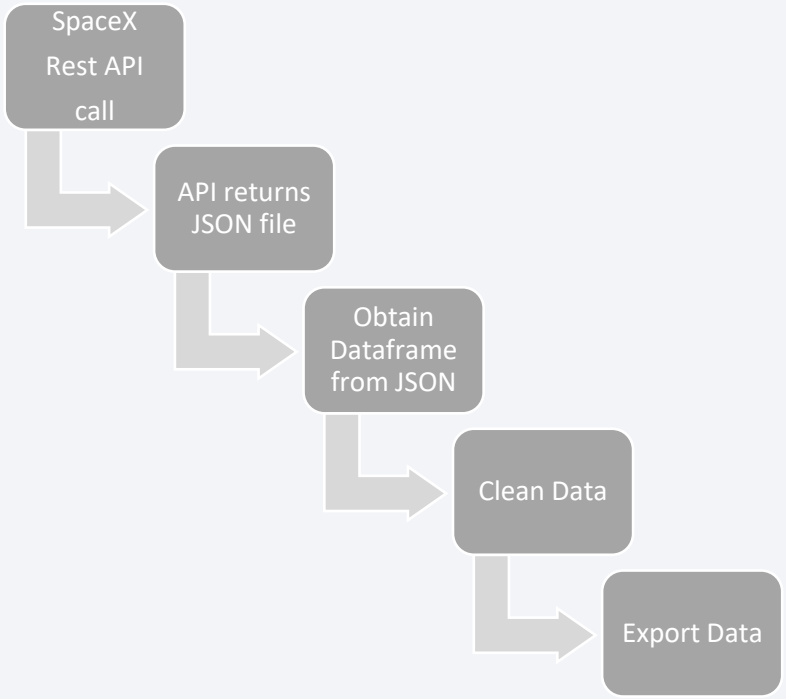
Executive Summary

- Data collection methodology:
  - The data is collected from Web Scrapping from Wikipedia and SpaceX REST API.
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding to get dummy variables 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

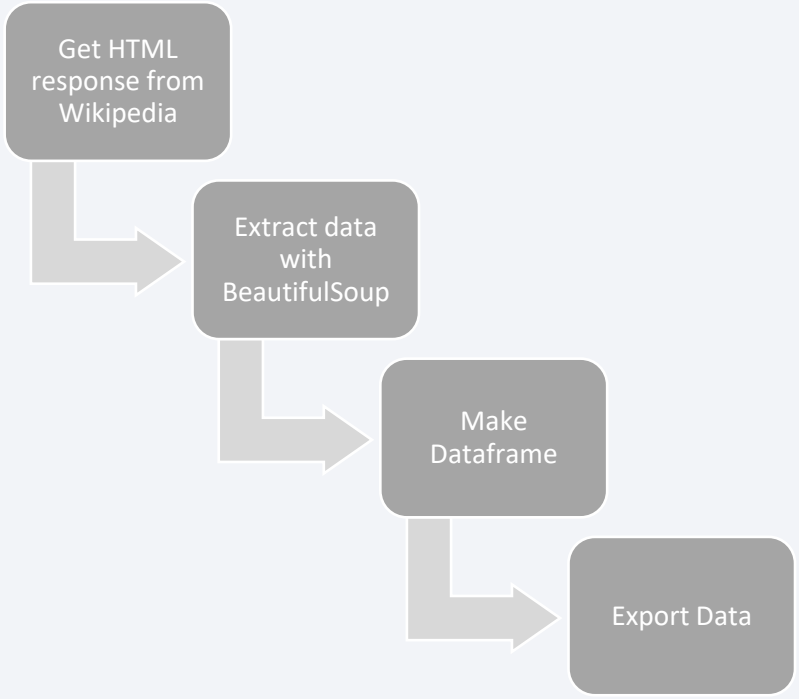
Data Collection

- The data is collected from Web Scrapping from Wikipedia** and SpaceX REST API*.

The information obtained by the API are rocket, launches, payload information.



The information obtained by the webscrapping of Wikipedia are launches, landing, payload information.



The Space XREST API URL is api.spacexdata.com/v4/

***The List of Falcon 9 Wikipedia data URL is https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922*

Data Collection – SpaceX API

Requesting rocket launch data from SpaceX API

1 Decode the response content as a Json and turn it into a Pandas dataframe

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

2 Combine the columns into a dictionary

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

3 *Create a data from launch_dict*

```
# Create a data from launch_dict
data = pd.DataFrame({key:pd.Series(value) for key, value in launch_dict.items()})
```

4 Filter dataframe and save it to a new dataframe

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

Export it to a CSV

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

https://github.com/etnyl/capstoneproject/blob/master/Data_Collection_API.ipynb

Data Collection - Scraping

Getting Response from HTML

1 Create BeautifulSoup Object

```
soup = BeautifulSoup(response.text, "html5lib")
```

2 Find all tables

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

3 Get column names

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

4 Create dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

5 Add data to keys

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is a
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.stri
                flag=flight_number.isdigit()
```

6 Create dataframe from dictionary

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

Export it to a CSV

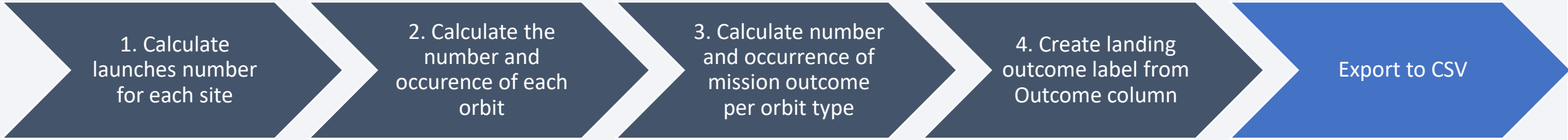
https://github.com/etnyl/capstoneproject/blob/master/DataCollection_Webscraping.ipynb

Data Wrangling

In the dataset, there are several cases where the booster did not land successully.

- True Ocean, True RTLS, True ASDS means the mission has been successful.
- False Ocean, False RTLS, False ASDS means the mission was a failure.

- We need to transform string variables into categorical variables where 1 means the mission has been successful and 0 means the mission was a failure.



https://github.com/etnyl/capstoneproject/blob/master/Data_Wrangling.ipynb

EDA with Data Visualization

While doing EDA some of chart types have been used for better understand the data.

Scatter plots are an essential type of data visualization that shows relationships between variables.

- Flight Number vs. Payload Mass
- Flight Number vs. Launch Site
- Payload vs. Launch Site
- Orbit vs. Flight Number
- Payload vs. Orbit Type
- Orbit vs. Payload Mass

Bar Charts show the frequency counts of values for the different levels of a categorical or nominal variable.

- Success rate vs. Orbit

Line graphs are used to track changes over different periods of time.

- Success rate vs. Year

https://github.com/etnyl/capstoneproject/blob/master/EDA_with_Visualization.ipynb

EDA with SQL

We performed SQL queries to gather and understand data from dataset:

- Displaying the names of the unique lauunch 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 date04-06-2010 and 20-03-2017in descending order.

https://github.com/etnyl/capstoneproject/blob/master/EDA_With_SQL.ipynb

Build an Interactive Map with Folium

Folium map object is a map centered on NASA Johnson Space Center at Houson, Texas

- Red circle at NASA Johnson Space Center's coordinate with label showing its name (folium.Circle, folium.map.Marker).
- Red circles at each launch site coordinates with label showing launch site name (folium.Circle, folium.map.Marker, folium.features.DivIcon).
- The grouping of points in a cluster to display multiple and different information for the same coordinates (folium.plugins.MarkerCluster).
- Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing. (folium.map.Marker, folium.Icon).
- Markers to show distance between launch site to key locations (railway, highway, coastway, city) and plot a line between them. (folium.map.Marker, folium.PolyLine, folium.features.DivIcon)

These objects are created in order to understand better the problem and the data. We can show easily all launch sites, their surroundings and the number of successful and unsuccessful landings.

https://github.com/etnyl/capstoneproject/blob/master/InteractiveVisualAnalytics_and_Dashboard.ipynb

Build a Dashboard with Plotly Dash

Dashboard has dropdown,pie chart,rangeslider and scatter plot components

- Dropdownallows a user to choose the launch site or all launch sites

  dash_core_components.Dropdown

- Pie chart shows the total success and the total failure for the launch site chosen with the dropdown component

  plotly.express.pie

- Rangeslider allows a user to select a payload mass in a fixed range

  dash_core_components.RangeSlider

- Scatter chart showsthe relationship between two variables, in particular Success vs Payload Mass

  plotly.express.scatter

Predictive Analysis (Classification)

Data Preparation

- Load dataset
- Normalize data
- Split data into training and test sets.

Model Preparation

- Selection of machine learning algorithms
- Set parameters for each algorithm to GridSearchCV
- Training GridSearchModel models with training dataset

Model Evaluation

- Get best hyperparameters for each type of model
- Compute accuracy for each model with test dataset
- Plot Confusion Matrix

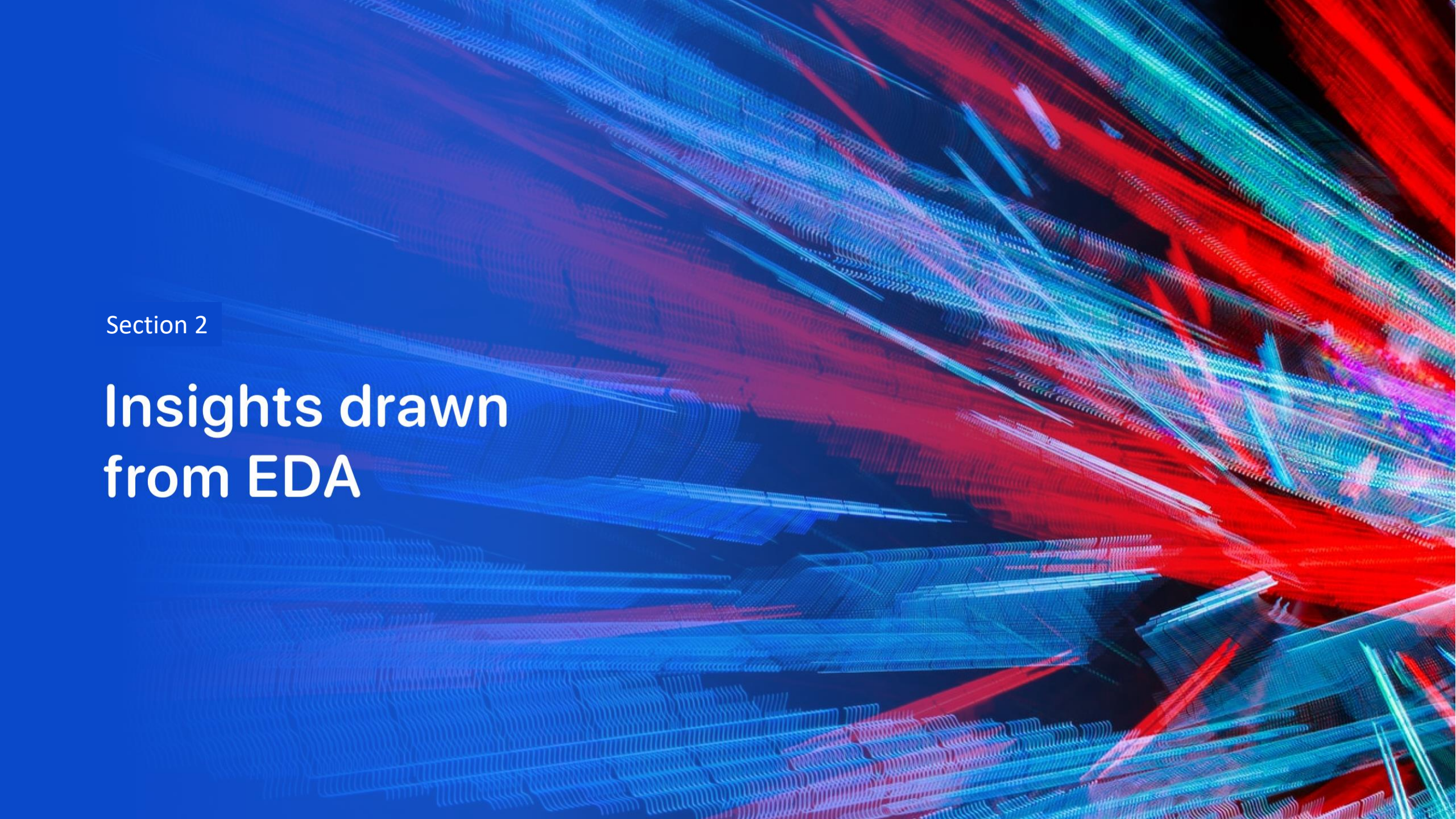
Model Comparison

- Comparison of models according to their accuracy
- The model with the best accuracy will be chosen

https://github.com/etnyl/capstoneproject/blob/master/Predictive_Analysis.ipynb

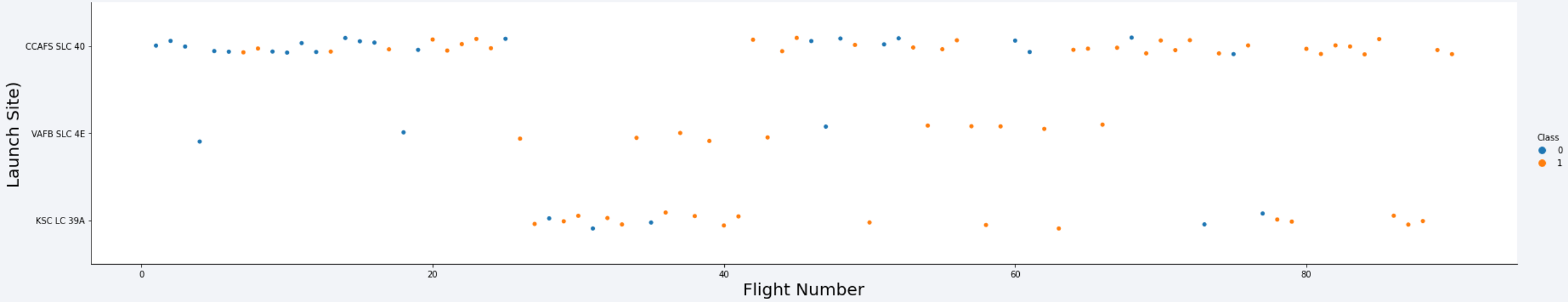
Results

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


Section 2
Insights drawn from EDA

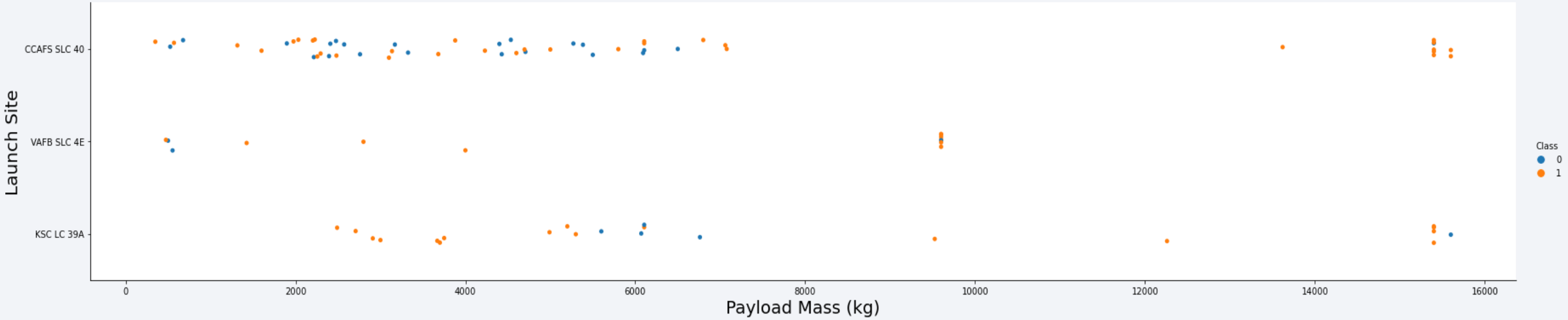
Flight Number vs. Launch Site

Taking the graph consideration for each site, the success rate is increasing.



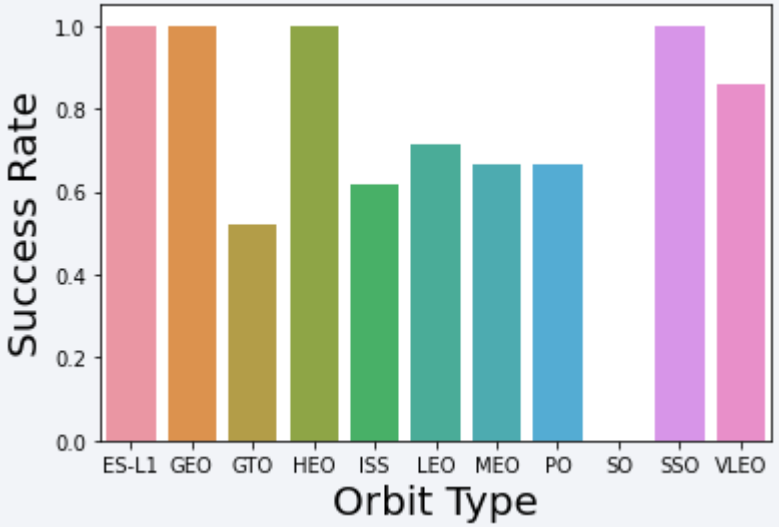
Payload vs. Launch Site

Depending on the launch site, a heavier payload may be a consideration for a successful landing. On the other hand, a too heavy payload can make a landing fail.



Success Rate vs. Orbit Type

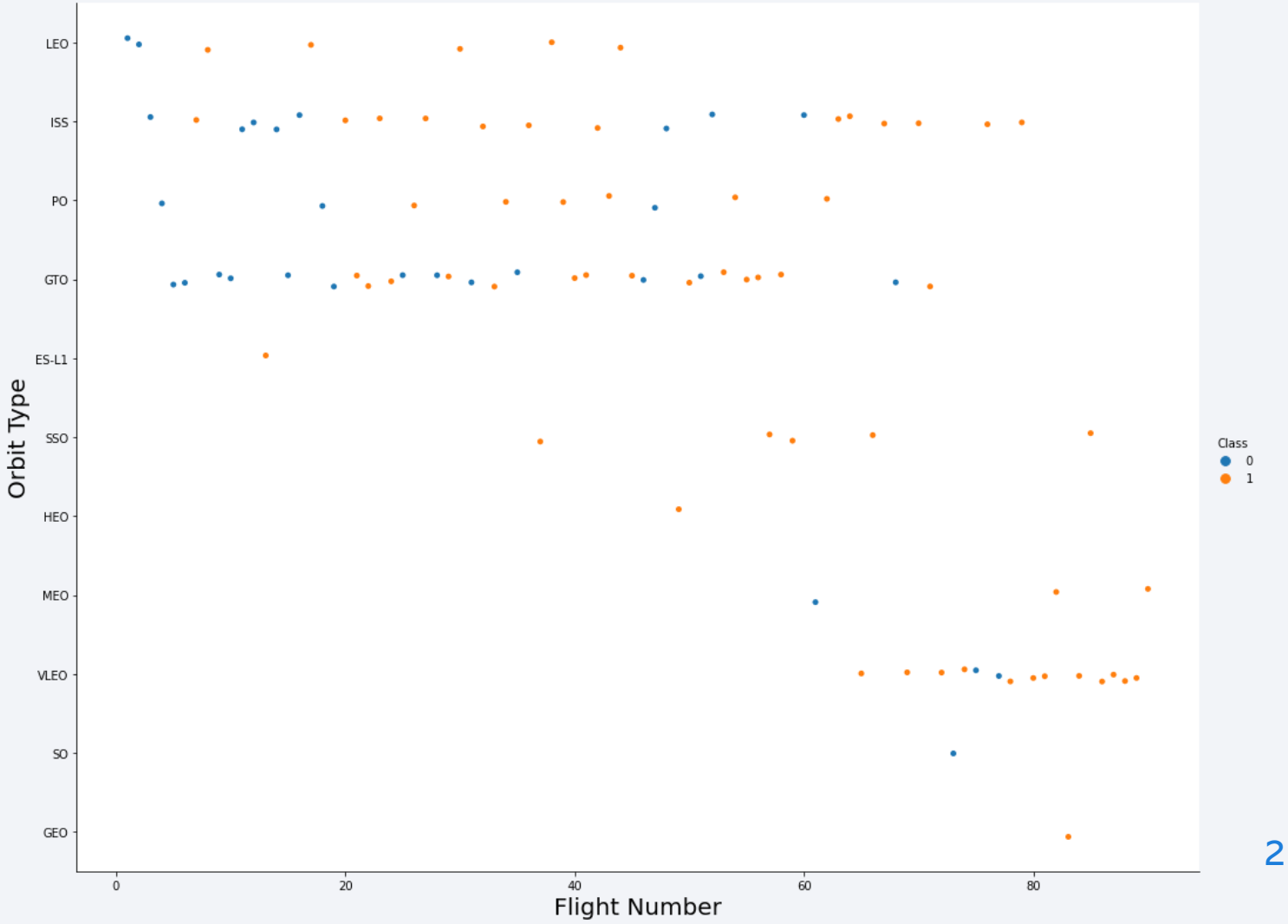
With this plot, we can see success rate for different orbit types. We note that ES-L1, GEO, HEO, SSO have the best success rate.



Flight Number vs. Orbit Type

We notice 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.

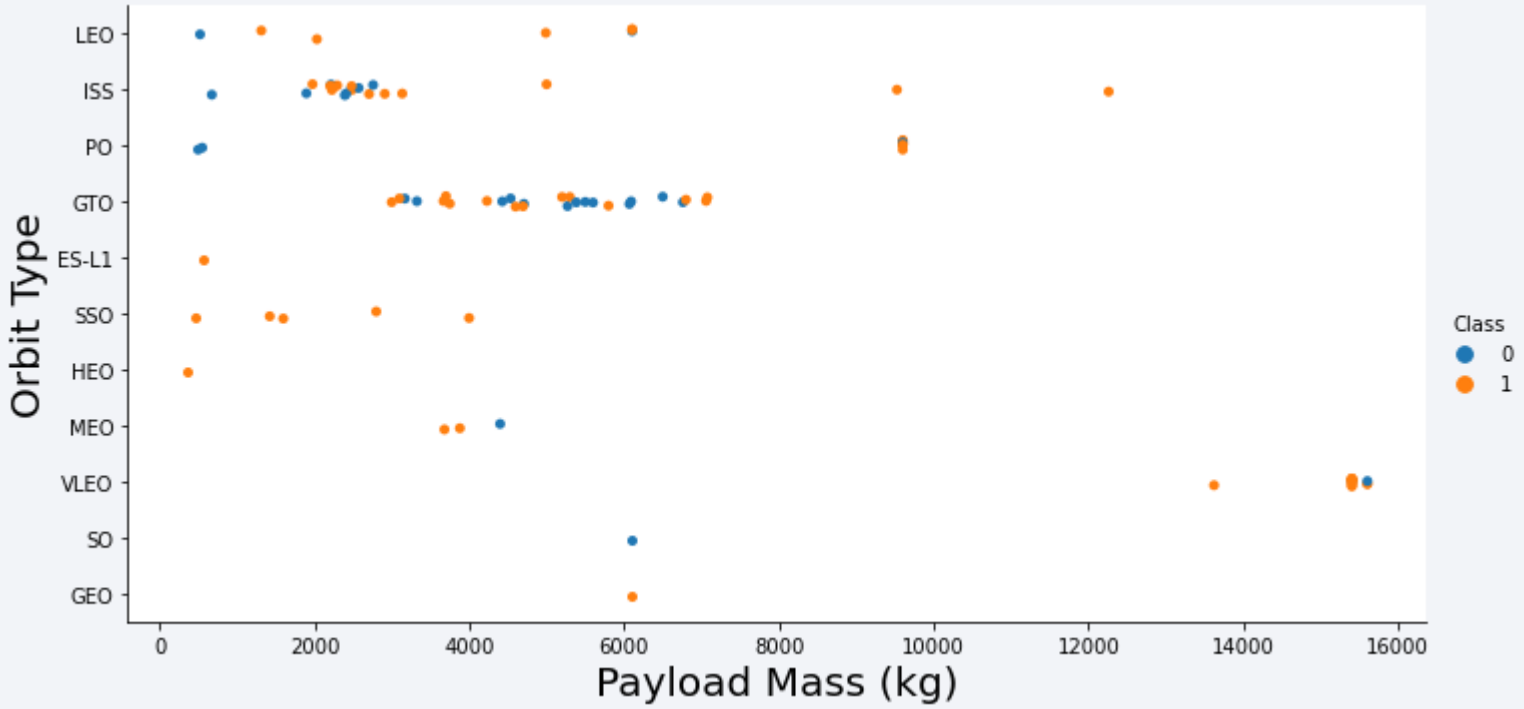
But we can suppose that the high success rate of some orbits like SSO or HEO is due to the knowledge learned during former launches for other orbits.



Payload vs. Orbit Type

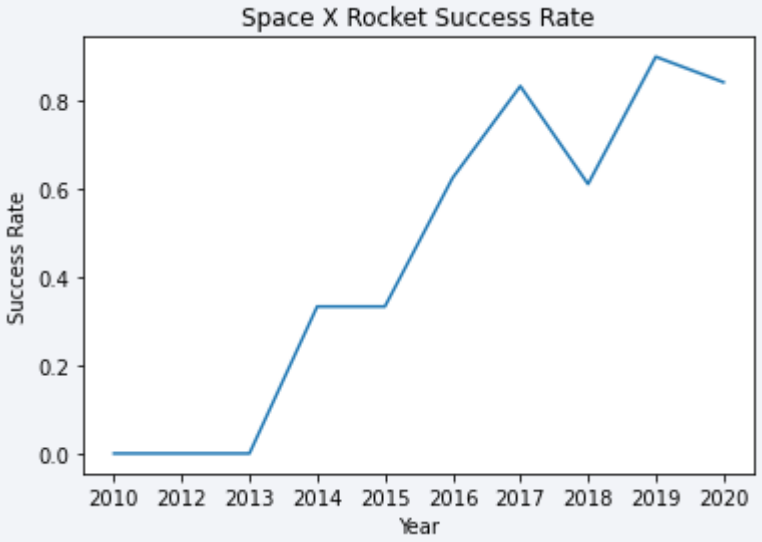
The weight of the payloads can have a great influence on the success rate of the launches in certain orbits. For example, heavier payloads improve the success rate for the LEO orbit.

Another finding is that decreasing the payload weight for a GTO orbit improves the success of a launch.



Launch Success Yearly Trend

You can observe that the success rate since 2013 kept increasing till 2020.



All Launch Site Names

- The names of the unique launch sites:

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

- SQL Query and Explanation:

The use of DISTINCT in the query allows to remove duplicate LAUNCH_SITE.

```
q = pd.read_sql('select distinct Launch_Site from spacexdata', conn)
q
```

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with `CCA`:

	index	Date	Time_(UTC)	Booster_Version	Launch_Site
0	0	2010-06-04 00:00:00	18:45:00	F9 v1.0 B0003	CCAFS LC-40
1	1	2010-12-08 00:00:00	15:43:00	F9 v1.0 B0004	CCAFS LC-40
2	2	2012-05-22 00:00:00	07:44:00	F9 v1.0 B0005	CCAFS LC-40
3	3	2012-10-08 00:00:00	00:35:00	F9 v1.0 B0006	CCAFS LC-40
4	4	2013-03-01 00:00:00	15:10:00	F9 v1.0 B0007	CCAFS LC-40

- SQL Query and Explanation:

The WHERE clause followed by LIKE clause filters launch sitesthat contain the substring CCA. LIMIT 5 shows 5 records from filtering.

```
q = pd.read_sql("select * from spacexdata where Launch_Site like 'CCA%' limit 5", conn)
q
```

Total Payload Mass

- The total payload carried by boosters from NASA is **45596.**

- SQL Query and Explanation:

This query returns the sum of all payload masses where the customer is NASA (CRS).

```
q = pd.read_sql("select sum(PAYLOAD_MASS__KG_) from spacexdata where Customer='NASA (CRS)'", conn)
q
```

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is **2928.4 kgs.**

- SQL Query and Explanation:

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

```
q = pd.read_sql("select avg(PAYLOAD_MASS__KG_) from spacexdata where Booster_Version='F9 v1.1'", conn)
q
```

First Successful Ground Landing Date

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

2015-12-22

- SQL Query and Explanation:

With this query, we select the oldest successful landing. The WHERE clause filters dataset in order to keep only records where landing was successful. With the MIN function, we select the record with the oldest date.

```
q = pd.read_sql("select min(Date) from spacexdata where Landing__Outcome='Success (ground pad)'", conn)
q
```

Successful Drone Ship Landing with Payload between 4000 and 6000

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

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

- SQL Query and Explanation:

```
q = pd.read_sql("select distinct Booster_Version from spacexdata where Landing__Outcome='Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000", conn) q
```

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

Total Number of Successful and Failure Mission Outcomes

- List the total number of successful and failure mission outcomes

Mission_Outcome	count(*)
Failure	1
Success	100

- SQL Query and Explanation:

```
q = pd.read_sql("select substr(Mission_Outcome,1,7) as Mission_Outcome, count(*) from spacexdata group by 1", conn)
```

With the first SELECT, we show the subqueries that return results. The first subquery counts the successful mission. The second subquery counts the unsuccessful mission. The WHERE clause followed by LIKE clause filters mission outcome . The COUNT function counts records filtered.

Boosters Carried Maximum Payload

- List the names of the booster_versions which have carried the maximum payload mass.

	Booster_Version
0	F9 B5 B1048.4
1	F9 B5 B1049.4
2	F9 B5 B1051.3
3	F9 B5 B1056.4
4	F9 B5 B1048.5
5	F9 B5 B1051.4
6	F9 B5 B1049.5
7	F9 B5 B1060.2
8	F9 B5 B1058.3
9	F9 B5 B1051.6
10	F9 B5 B1060.3
11	F9 B5 B1049.7

- SQL Query and Explanation:

```
q = pd.read_sql("select distinct Booster_Version from spacexdata where
PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacexdata)",
conn)
```

We used a subquery to filter data by returning only the heaviest payload mass with MAX function. The main query uses subquery results and returns unique booster version (SELECT DISTINCT) with the heaviest payload mass.

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015.

	Landing__Outcome	Booster_Version	Launch_Site
0	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
1	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40
2	Failure (drone ship)	F9 v1.1 B1017	VAFB SLC-4E
3	Failure (drone ship)	F9 FT B1020	CCAFS LC-40
4	Failure (drone ship)	F9 FT B1024	CCAFS LC-40

- SQL Query and Explanation:

```
q = pd.read_sql("select distinct Landing__Outcome, Booster_Version, Launch_Site from spacexdata where Landing__Outcome='Failure (drone ship)'", conn)
```

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.

	Landing__Outcome	count(*)
0	No attempt	10
1	Success (drone ship)	5
2	Failure (drone ship)	5
3	Success (ground pad)	3
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1

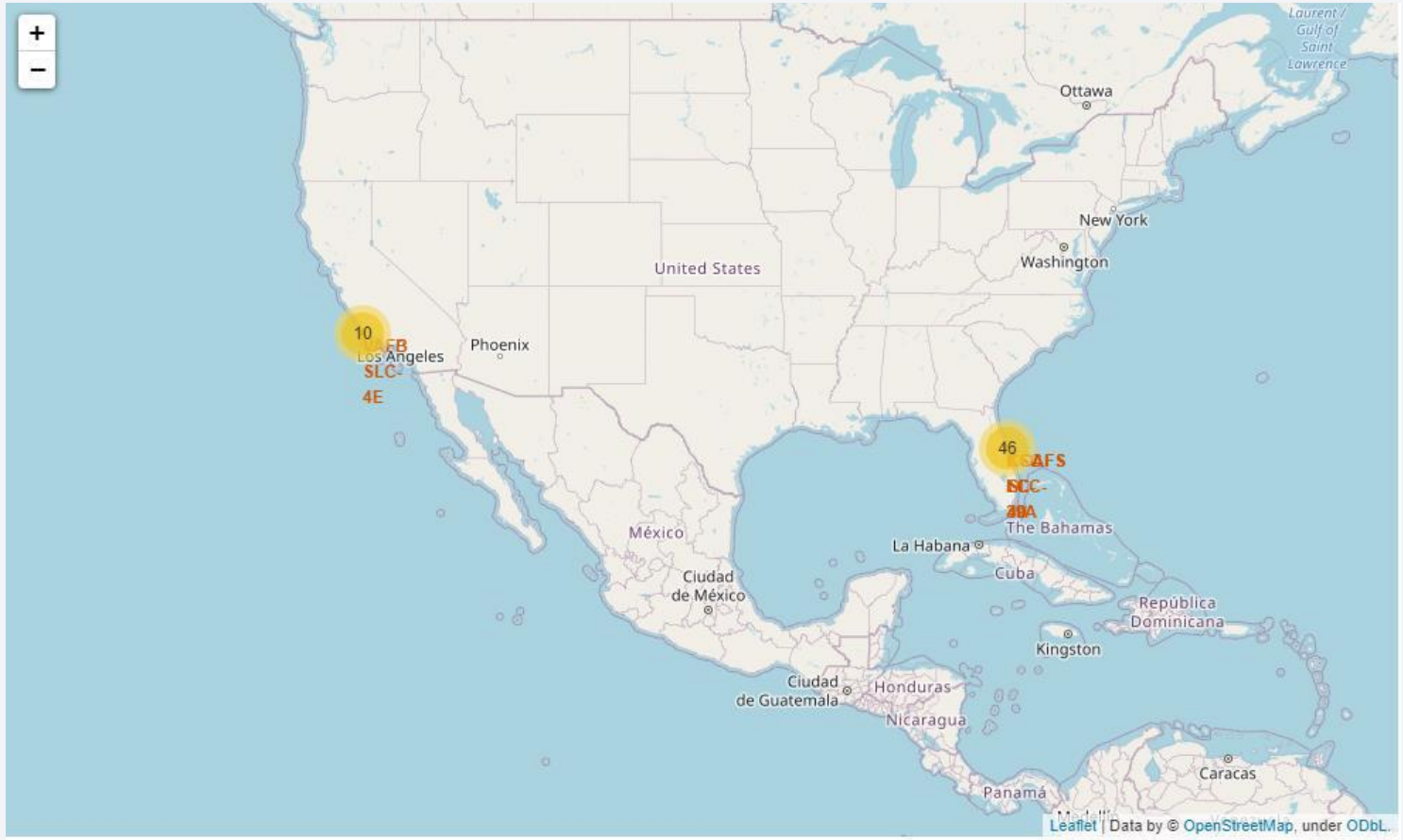
- SQL Query and Explanation:

```
q = pd.read_sql("select Landing__Outcome, count(*) from spacexdata where Date between '2011-06-04' and '2017-03-20' group by Landing__Outcome order by 2 desc"
```

This query returns landing outcomes and their countwhere 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 2 DESC shows results in decreasing order.

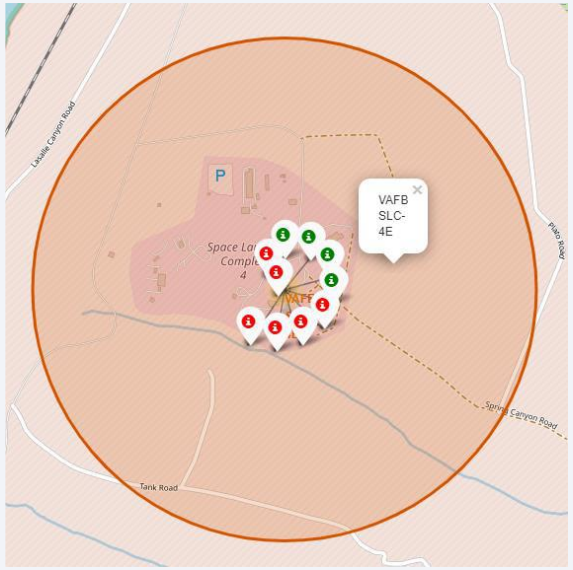

Section 3
Launch Sites
Proximities Analysis

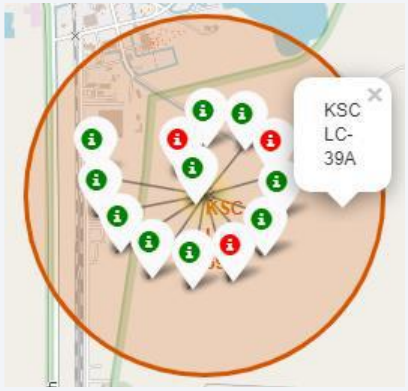
Folium map - Ground stations

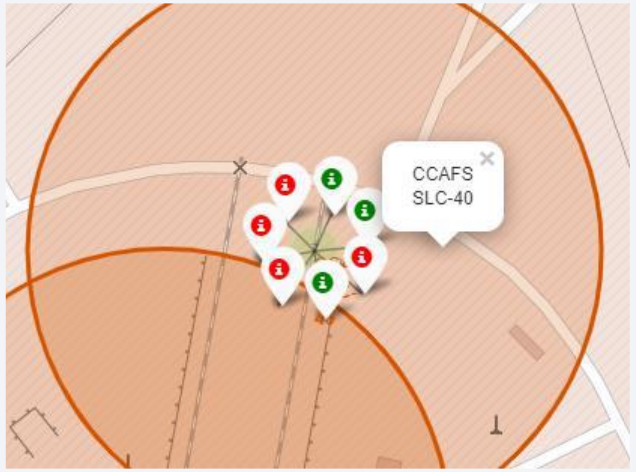


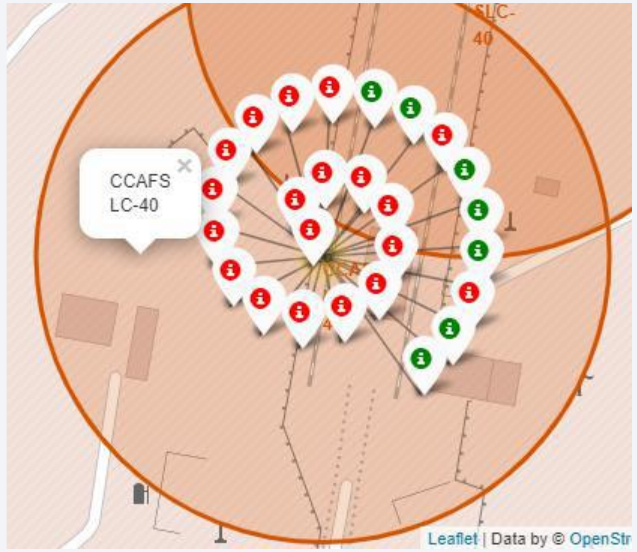
It can be seen from the map, Space X launch sites are located on the coast of the United States.

Folium map – Color Labeled Markers







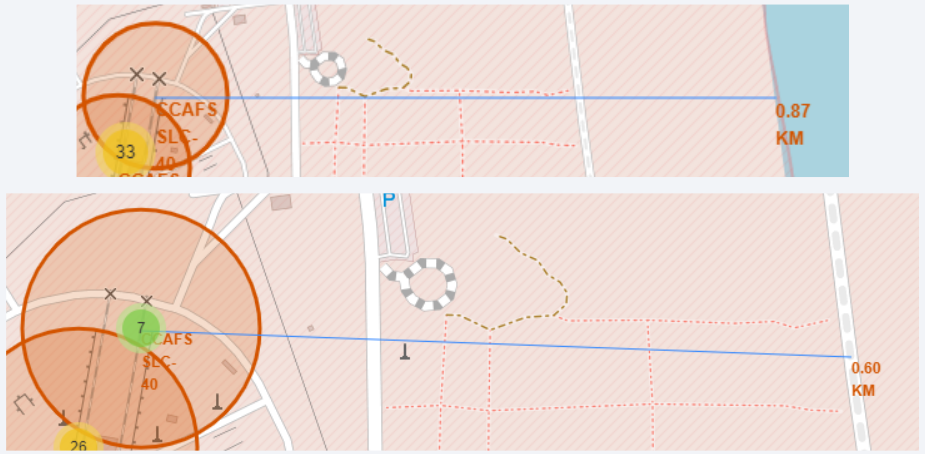


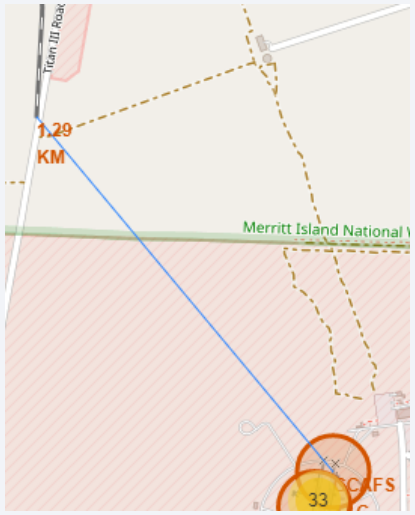
Greenmarker represents successful launches.

Red marker represents unsuccessful launches.

We note that KSC LC-39A has a higher launch success rate.

Folium Map –Distances between CCAFS SLC-40 and its proximities



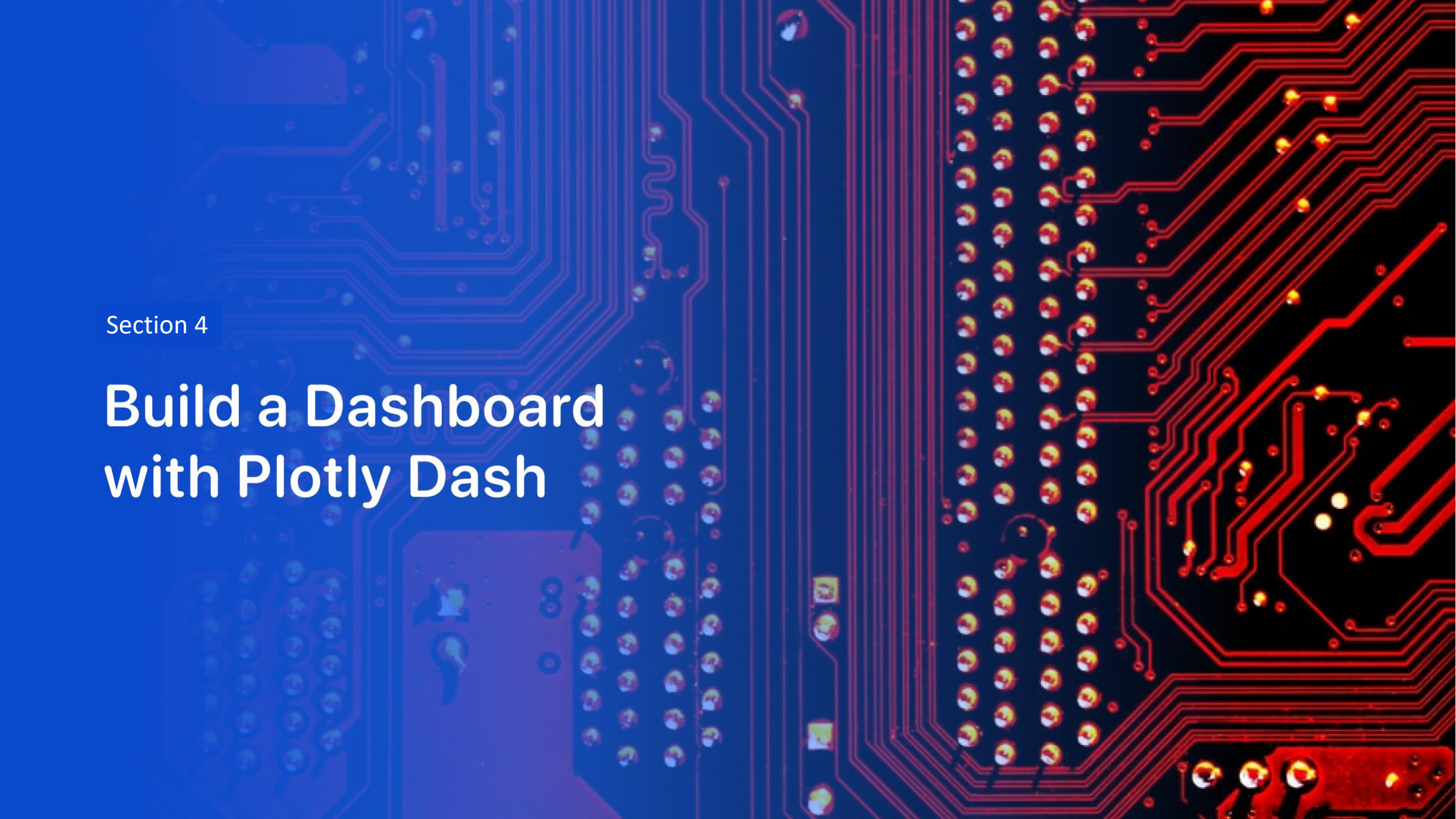


Is CCAFS SLC-40 in close proximity to railways ? **Yes**

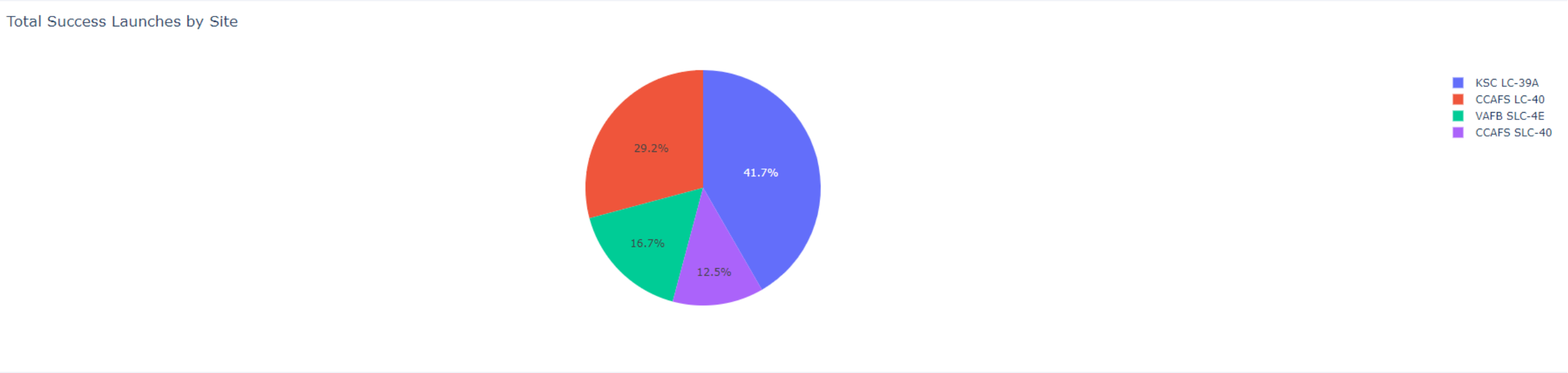
Is CCAFS SLC-40in close proximity to highways ? **Yes**

Is CCAFS SLC-40in close proximity to coastline ? **Yes**

Do CCAFS SLC-40keeps certain distance away from cities ? **No**

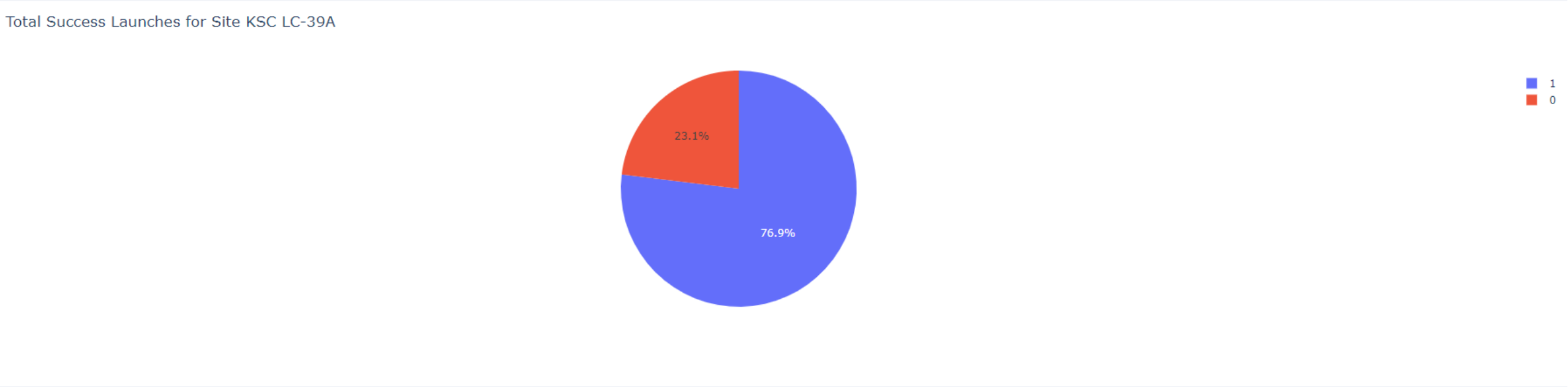

Section 4
Build a Dashboard with Plotly Dash

Dashboard –Total success by Site



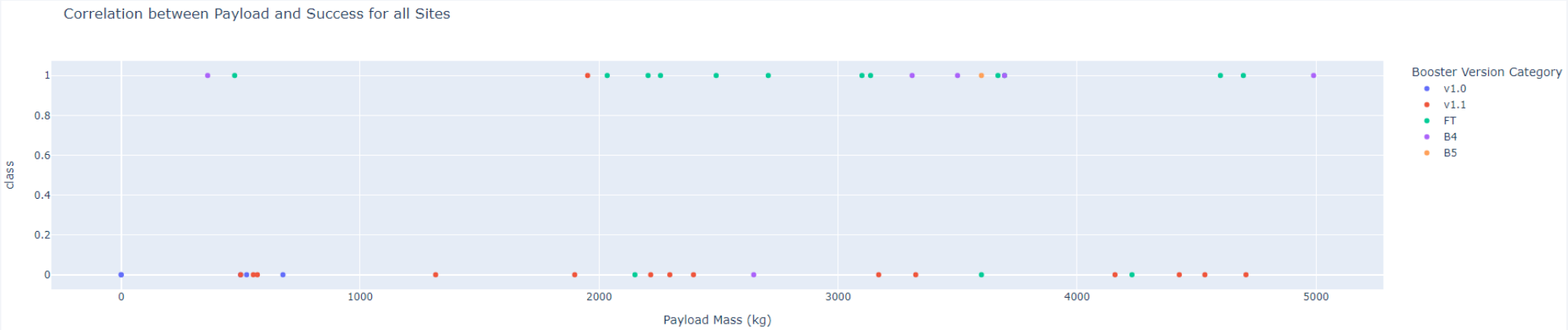
KSC LC-39A has the most success rate of launches with 41.7%.

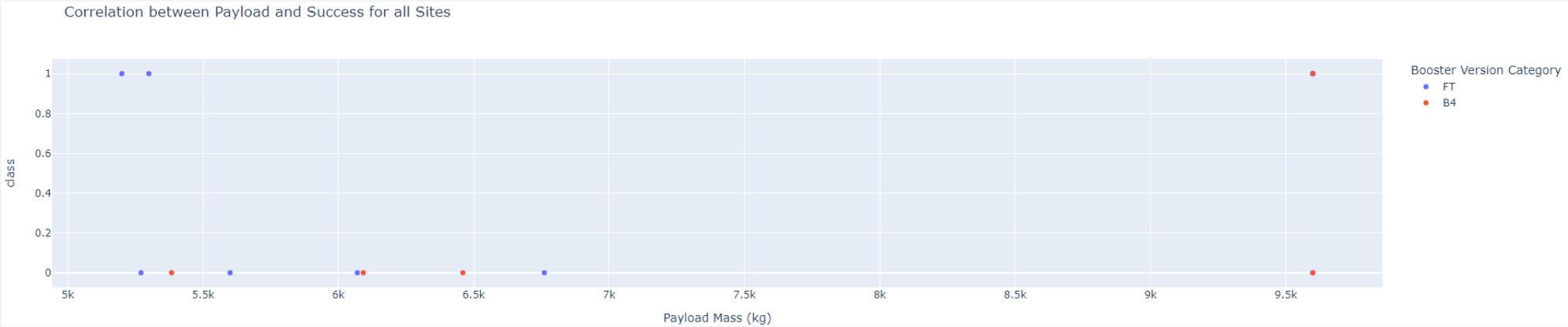
Dashboard –Total success launches for Site KSC LC-39A



KSC LC-39A has achieved a 76.9% success rate while getting a 23.1% failure rate.

Dashboard- Payload mass vs Outcome for all sites with different payload mass selected





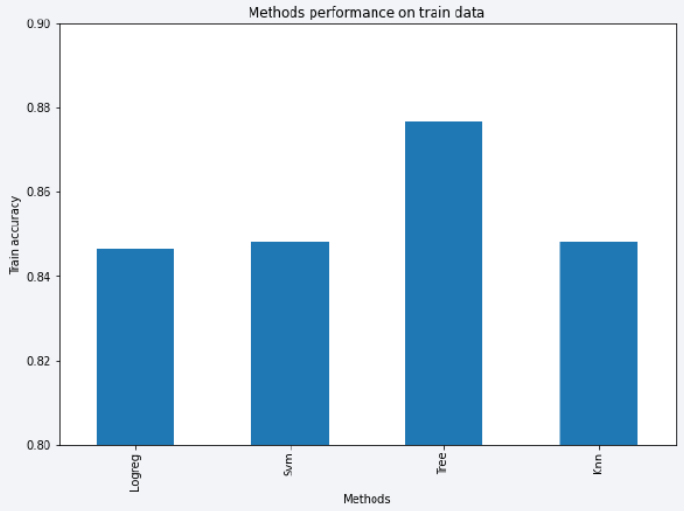
Low weighted pay loads have a better success rate than the heavy weighted payloads.

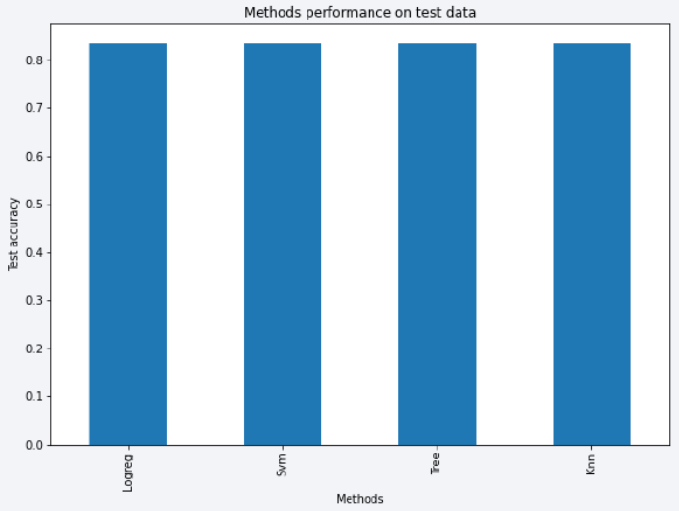
Section 5

Predictive Analysis (Classification)

Classification Accuracy

Model	Accuracy Train	Accuracy Test
Tree	0.876786	0.888889
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333





The difference between the all methods is not important when assessing the accuracy.

But if we really need to choose one, this would be the decision tree by 88.9% accuracy rate.

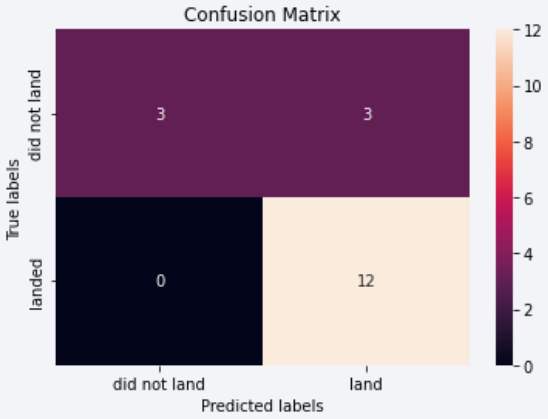
Confusion Matrix

Logistic Regression

Decision Tree

KNN

SVM



The confusion matrixs of all models have the same result. The main problem of these models are false positives.

Conclusions

- The success of a mission can be explained by several factors such as the launch site, the orbit and especially the number of previous launches. Indeed, we can assume that there has been a gain in knowledge between launches that allowed to go from a launch failure to a success.
- The orbits with the best success rates are GEO, HEO, SSO, ES-L1.
- Depending on the orbits, the payload mass can be a criterion to take into account for the success of a mission. Some orbits require a light or heavy payload mass. But generallyl ow weighted payloads perform better than the heavy weighted payloads.
- With the current data, we cannot explain why some launch sites are better than others (KSC LC-39A is the best launch site). To get an answer to this problem, we could obtain atmospheric or other relevant data.
- For this dataset, we choose the Decision Tree Algorithm as the best model even if the test accuracy between all the models used is identical. We choose Decision Tree Algorithm because it has a better train accuracy.

Appendix

- For more information about Jupyter Notebook Outputs, Python codes, SQL queries see the below GitHub link

https://github.com/etnyl/capstoneproject

Thank you!