IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kohei Kurano
1/3/2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection via API, Web Scraping
  - Exploratory Data Analysis with Data Visualization
  - EDA with SQL
  - Interactive Map with Folium
  - Dashboard with Plotly Dash
  - Predictive Analysis
- Summary of all results
  - Exploratory Data Analysis results
  - Interactive maps and dashboard
  - Predictive results

Introduction

- Project background and context
  - The aim of this project is to predict if the Falcon 9 first stage will successfully land. SpaceX says on its website that the Falcon 9 rocket launch cost 62 million dollars. Other providers cost upward of 165 million dollars each. The price difference is explained by the fact that SpaceX can refuse the first stage. By determining if the stage will land, we can determine the cost of a launch. This information is interesting for another company if it wants to compete with SpaceX for a rocket launch.
- Problems you want to find answers
  - 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 allow SpaceX to achieve the best landing success rate?

Section 1

Methodology

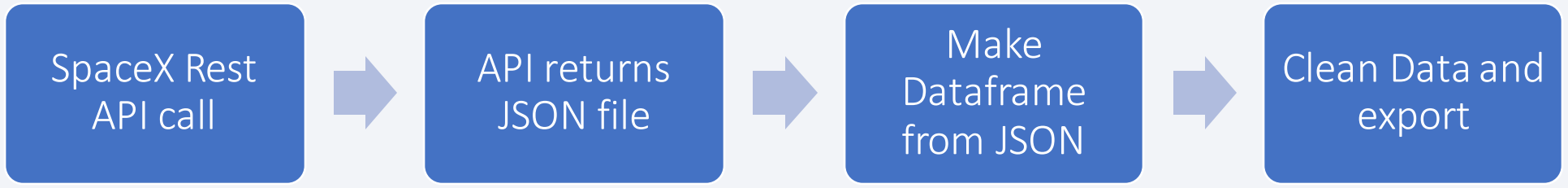
Methodology

Executive Summary

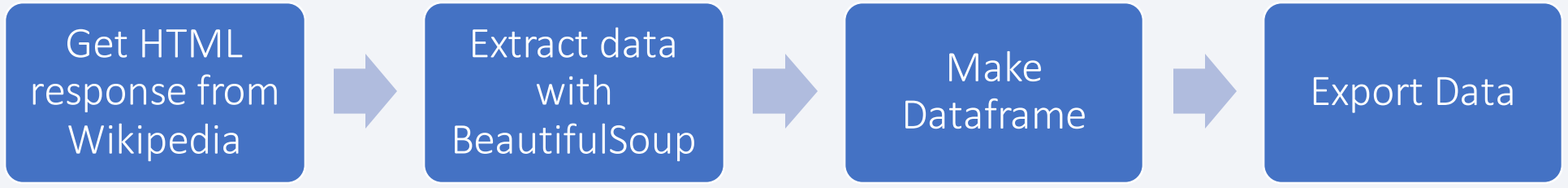
- Data collection methodology:
  - Space X REST API
  - Web Scraping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using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

Data Collection

- Datasets are collected from Rest SpaceX API and web scaraping Wikipedia
  - The information obtained by the API are rocket, launches, payload information
    - The SpaceX REST API URL



  - The information obtained by web scraping of Wikipedia are launches, landing, payload information
    - Wikipedia URL



Data Collection – SpaceX API

Link to code

1. Getting Response from API

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

response = requests.get(spacex_url)
```

2. Convert Response to JSON File

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

3. Tranfrom data

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

4.Create dictionary with data

```
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}
```

5. Create dataframe

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

6. Filter dataframe

```
data_falcon9 = df[df.BoosterVersion == 'Falcon 9']
```

7. Export to file

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

Data Collection - Scraping

Link to code

1. Getting Response from HTML

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

2. Create BeautifulSoup Object

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

3. Find all tables

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

4. Get 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)
```

5. 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']=[]
```

6. Add data to keys

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

7. Create dataframe from dictionary

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

8. Export to file

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

Data Wrangling

Link to code

- In the dataset, there are several cases where the booster did not land succesfully
  - True Ocean, True RTLS, True ASDS means the mission has been successful
  - False Ocean, False RTLS, False ASDS means the mission was failure
- We need to transform string variables into categorical variables where 1 means the mission has been successful and 0 means the mission was failure

1. Calculate launches number for the each site

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

2. Calculate the number and occurrence of each orbit

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

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

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

4. Create a landing outcome 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)

df['Class']=landing_class
```

5. Export to file

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

EDA with Data Visualization

Link to code

- Scatter Graphs
  - 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 Graph
  - Success rate vs Orbit
- Line Graph
  - Success rate vs Year

EDA with SQL

Link to code

- We performed SQL queries to gather and understand data from dataset:
  - Display 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 version, 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

Build an Interactive Map with Folium

Link to code

- Folium map object is a map centered on NASA Johnson Space Center at Houston, Texas
  - Red circle at NASA Johnson Space Center's coordinate with label showing its name
  - Red circle at each launch site coordinates with label showing launch site name
  - The grouping of points in a cluster to display multiple and different information for the same coordinates
  - Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing
  - Markers to show distance between launch site to key locations
- 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

Build a Dashboard with Plotly Dash

Link to code

- Dashboard has dropdown, pie chart, rangeslider and scatter plot components
  - Dropdown allows a user to choose the launch site or all launch sites
  - Pie chart shows the total success and the total failure for the launch site chosen with the dropdown component
  - Rangeslider allows a user to select a payload mass in a fixed range
  - Scatter chart shows the relationship between two variables, in particular Success vs Payload Mass

Predictive Analysis (Classification)

Link to code

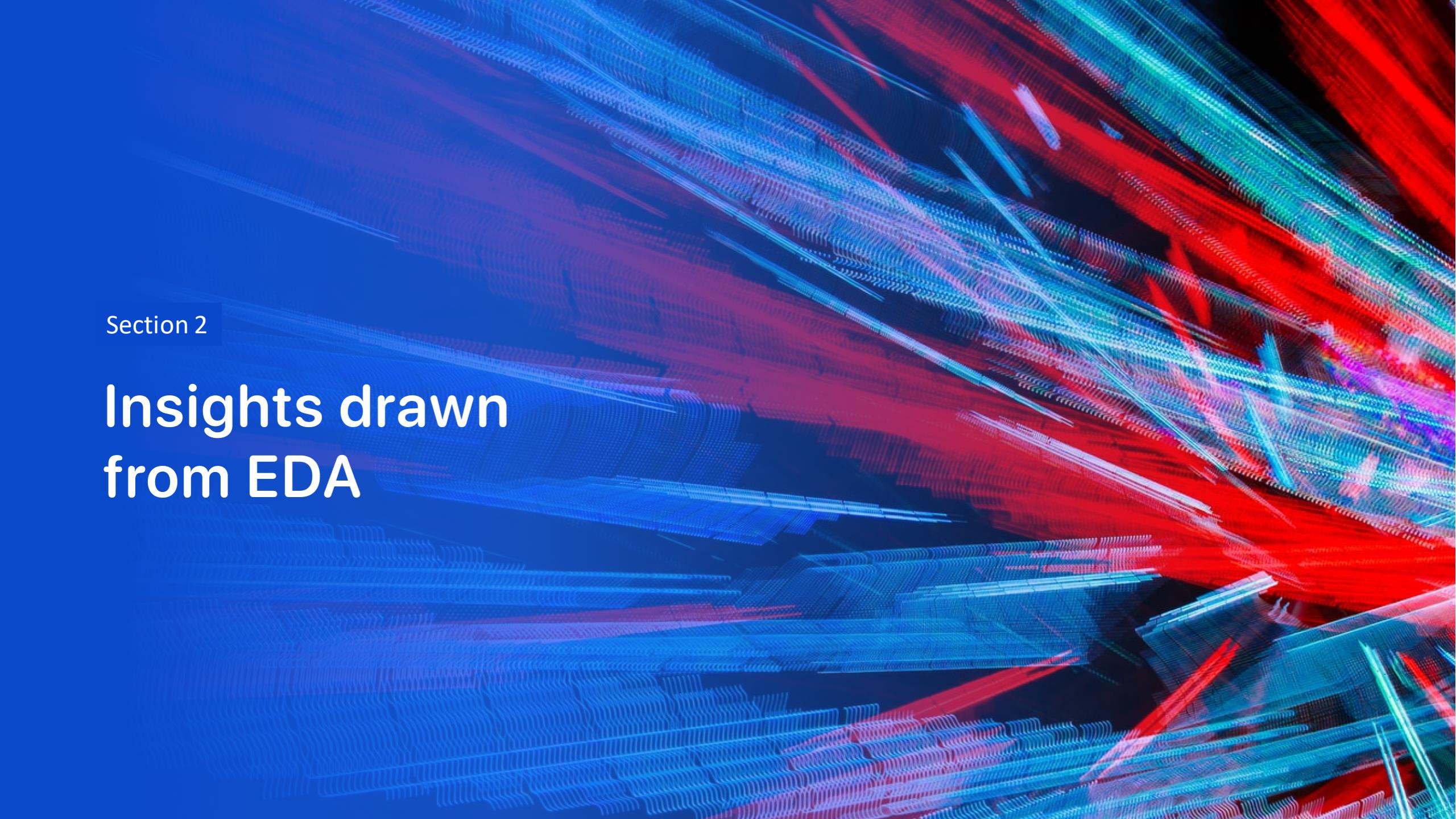
- Data preparation
  - Load dataset
  - Normalize data
  - Split data into training and test sets
- Model preparation
  - Selection of machine learning algorithms
  - Set parameters for each algorithm to GridSearchCV
  - Training GridSearchModel models with training dataset
- Model evaluation
  - Get 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

Results

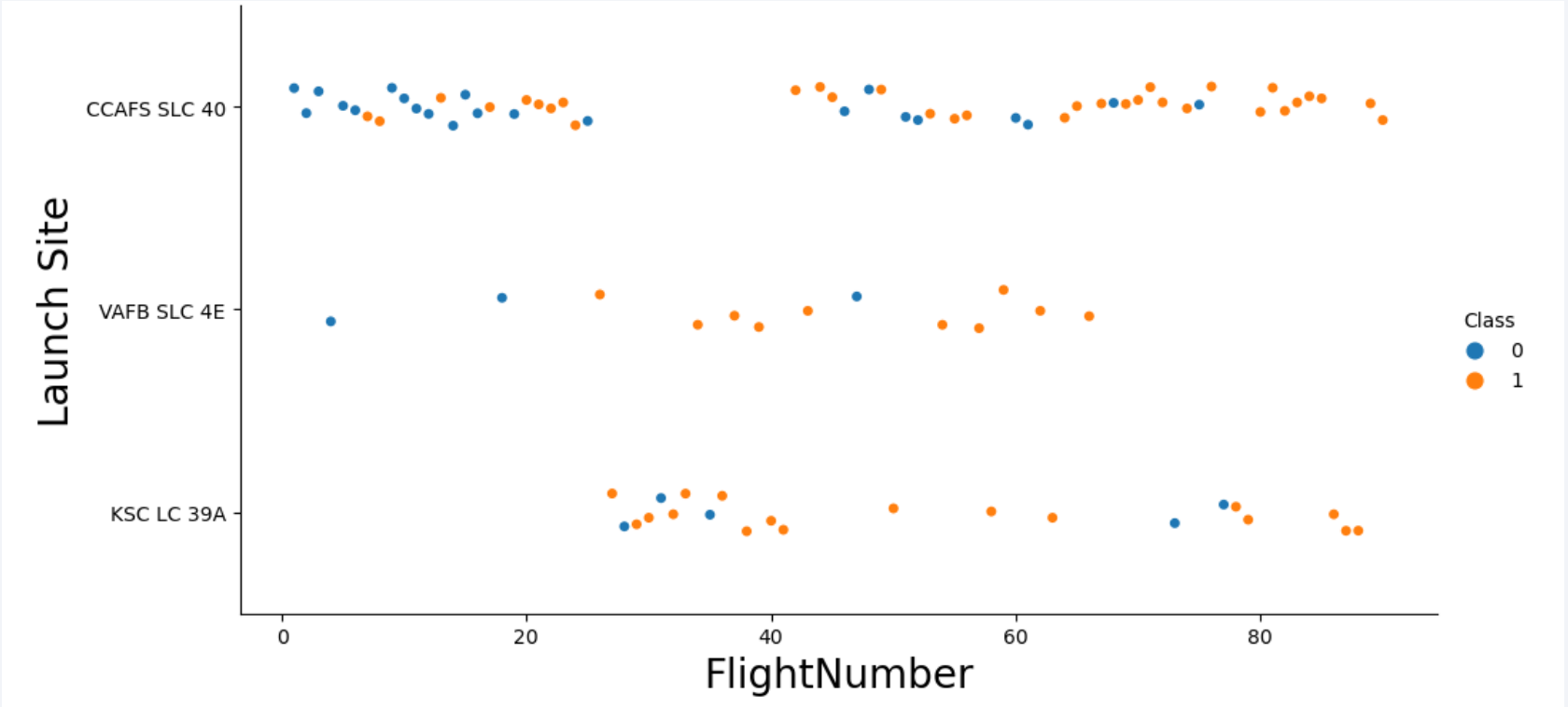
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

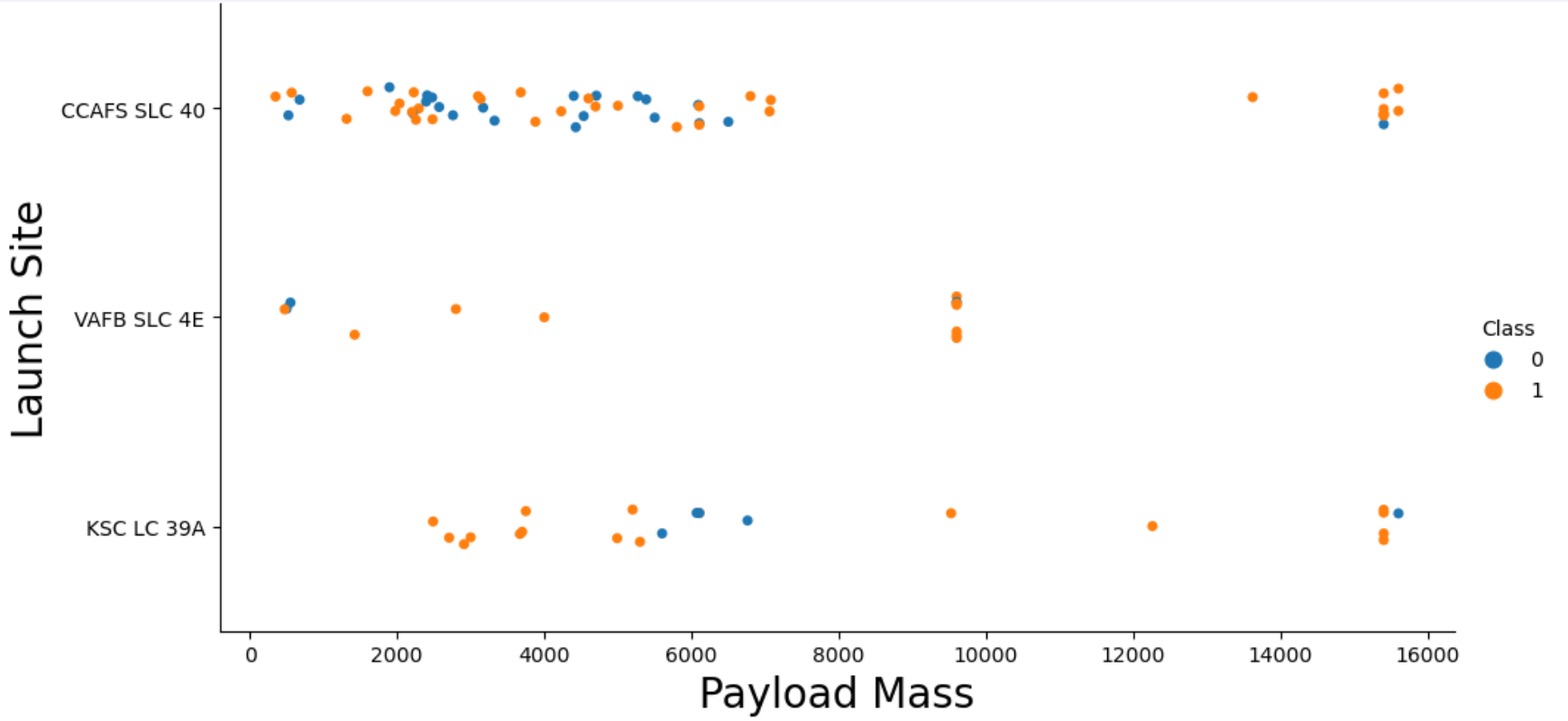


Flight Number vs. Launch Site



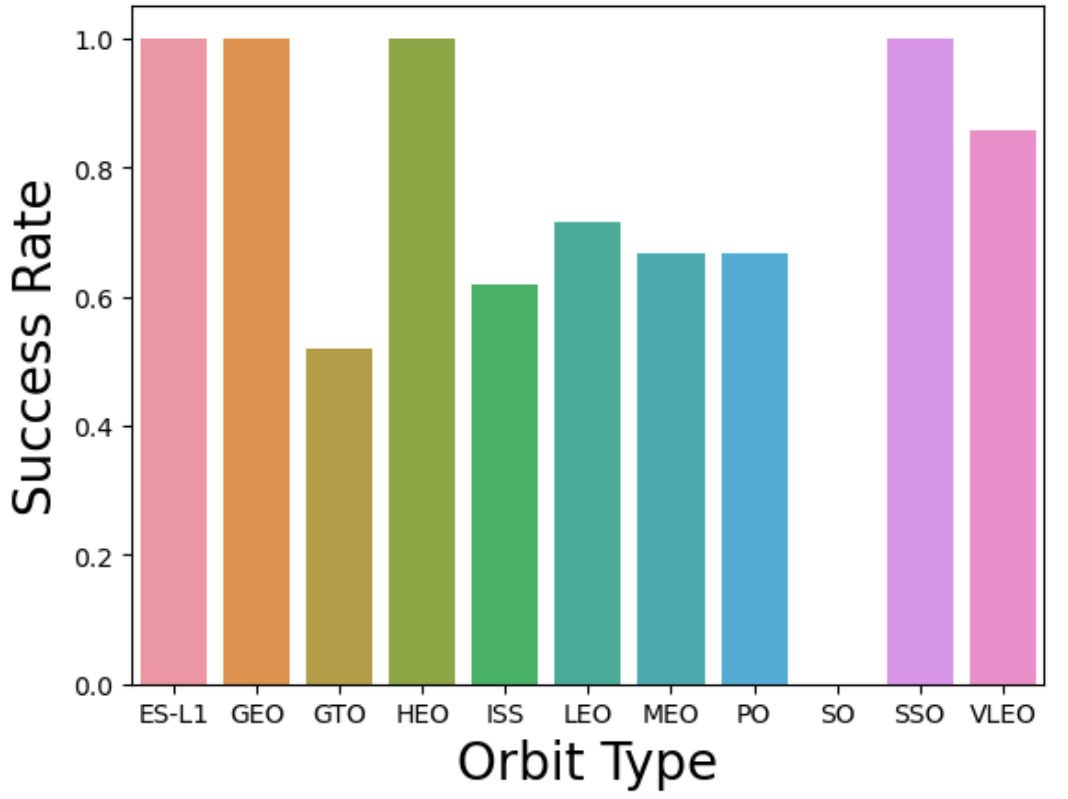
The success rate is increasing for each site

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

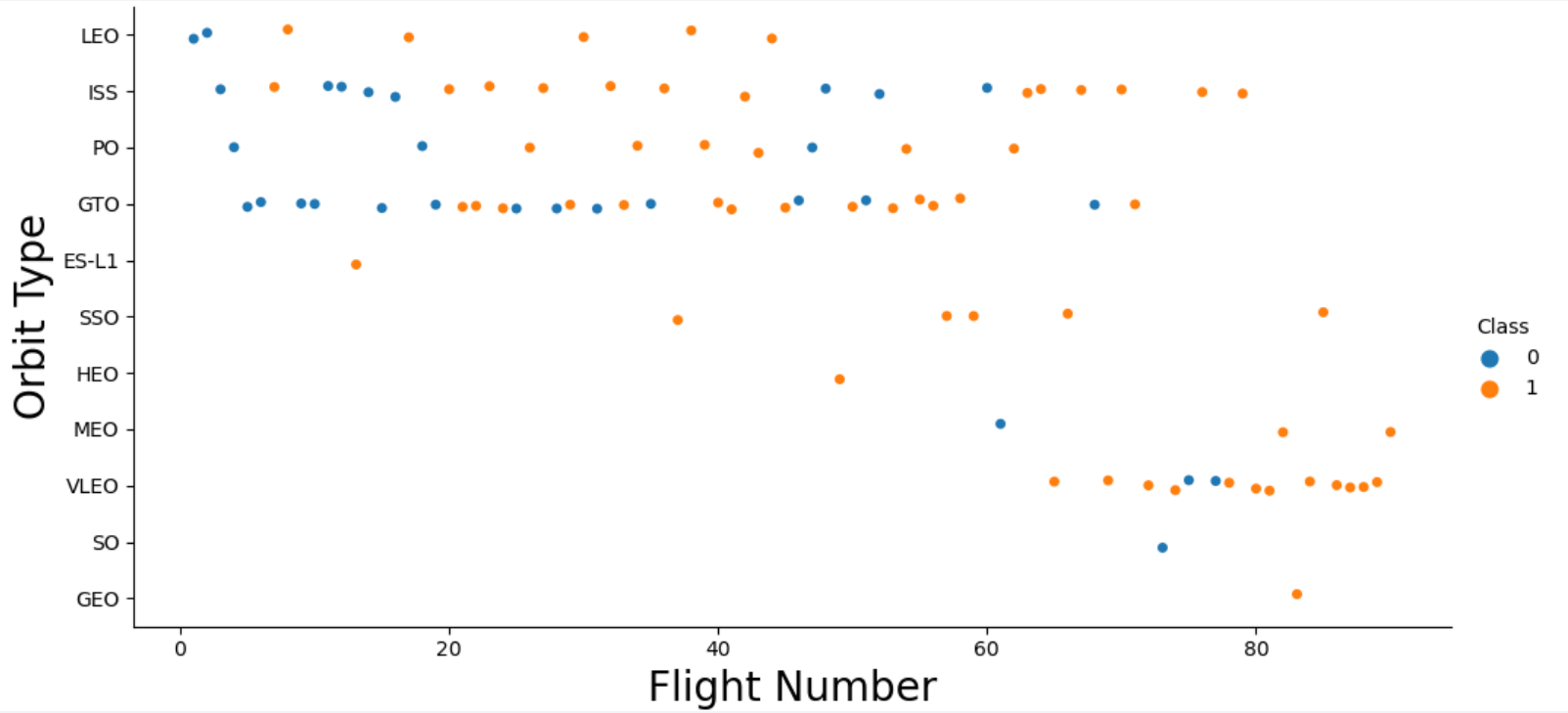
Success Rate vs. Orbit Type



With this plot, we can see success rate for different type of Orbit.

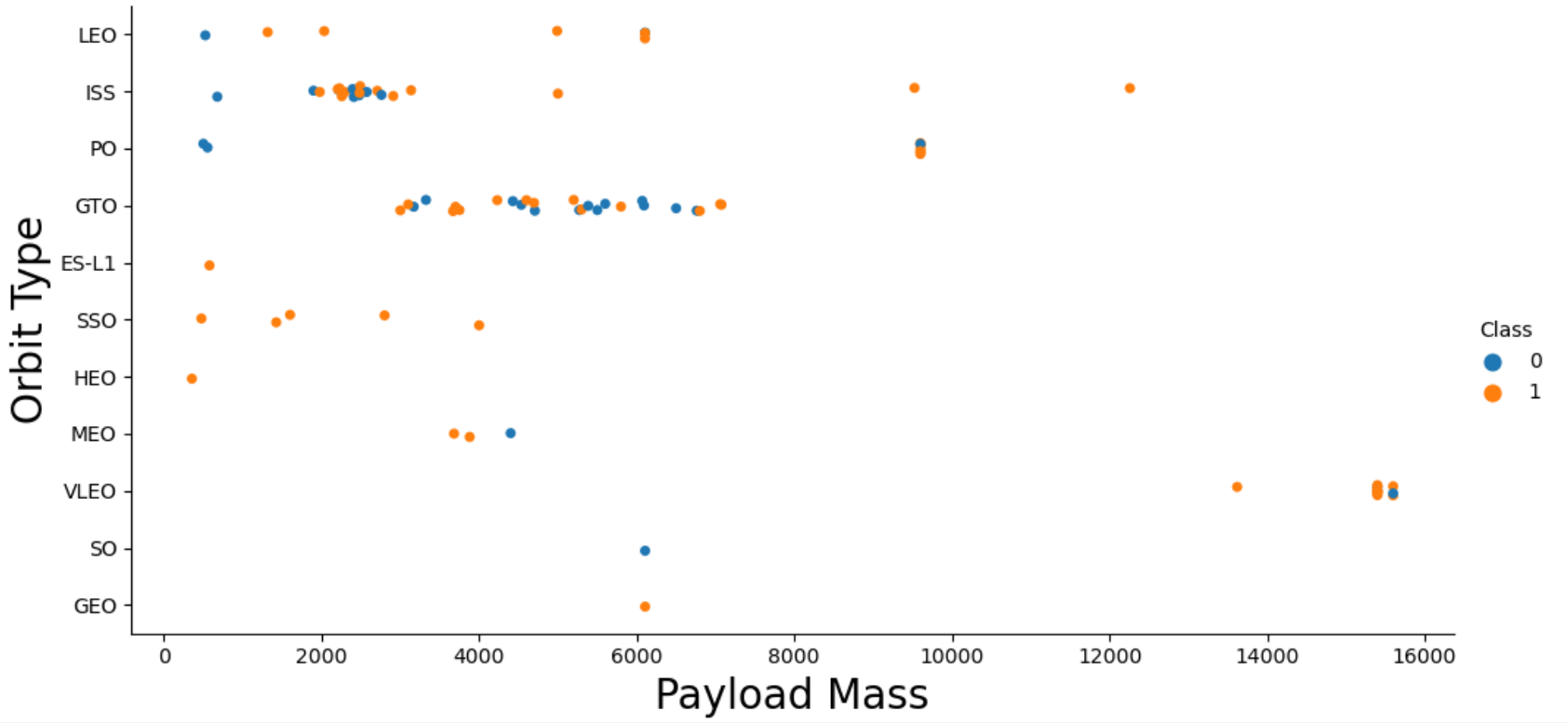
ES-L1, GEO, HEO, SSO have the best success rate

Flight Number vs. Orbit Type



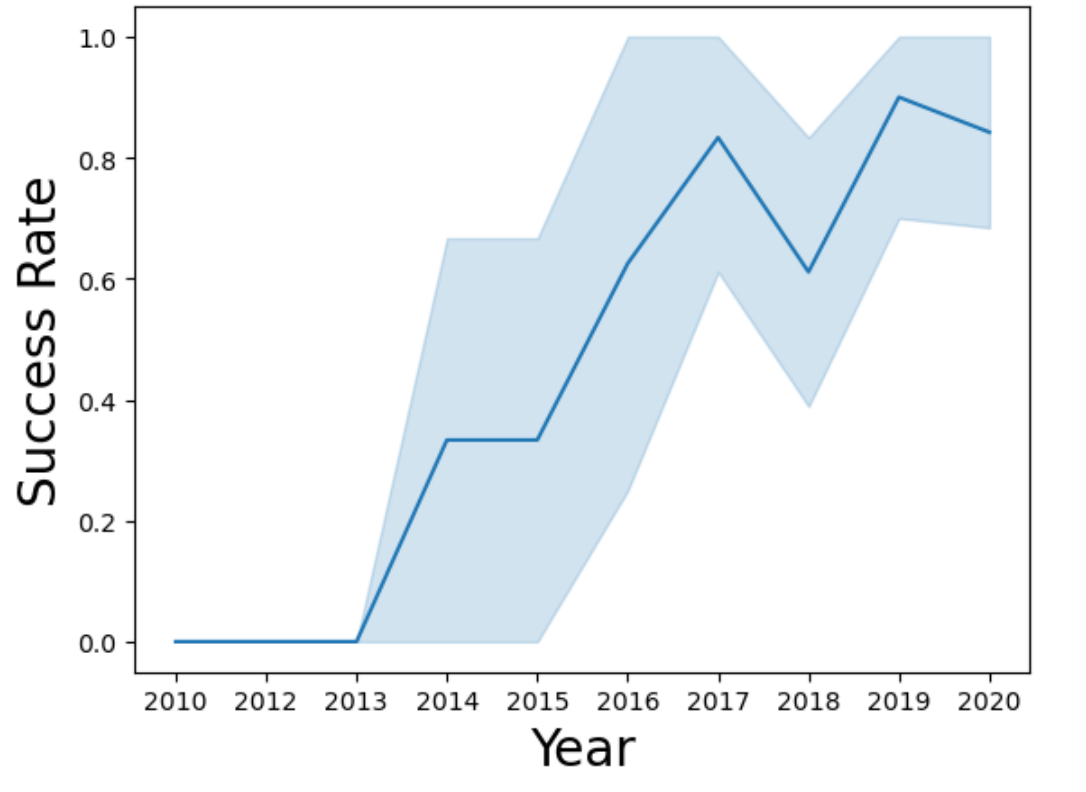
We notice that the success rate increases with the number of flights for the LEO orbit. For some orbits like GTO, there is no relation between the rate and the number of flights. However we can suppose that the high success rate of some orbits like SSO or HEO is due to the knowledge learned during former 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. 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



Since 2013, we can see an increase in the SpaceX Rocket success rate

All Launch Site Names

```
%sql select Unique(LAUNCH_SITE) from SPACEX
```

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

The use of UNIQUE in the query allows to remove duplicate LAUNCH_SITE

Launch Site Names Begin with 'CCA'

```
%sql select LAUNCH_SITE from SPACEX where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5
```

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

WHERE clause followed by LIKE clause filters LAUNCH_SITE that contain the substring 'CCA'.
LIMIT 5 shows 5 records from filtering

Total Payload Mass

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEX where (CUSTOMER) = 'NASA (CRS)'
```

1
45596

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

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEX where (BOOSTER_VERSION) LIKE '%F9 v1.1%'
```

1
2534

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 SPACEX where (LANDING__OUTCOME) LIKE '%Success%'
```

1
2015-12-22

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

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select BOOSTER_VERSION from SPACEX where LANDING__OUTCOME='Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000
```

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 and 6000 kg. The WHERE and AND clauses filter the dataset

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT MISSION_OUTCOME, COUNT(*) FROM SPACEX GROUP BY MISSION_OUTCOME
```

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

With the COUNT query, we select both success and failure outcome. Then with GROUP BY query, it shows the number of each records

Boosters Carried Maximum Payload

```
%sql select BOOSTER_VERSION from SPACEX where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEX)
```

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

We use a subquery to filter data by returning only te heaviest payload mass with MAX function. The main query uses subquery results and return unique booster version with the heaviest payload mass

2015 Launch Records

```
%sql select BOOSTER_VERSION, LAUNCH_SITE from SPACEX where LANDING__OUTCOME = 'Failure (drone ship)' and DATE LIKE '2015%'
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

This query returns booster version and launch site where landing was failure in 2015.

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

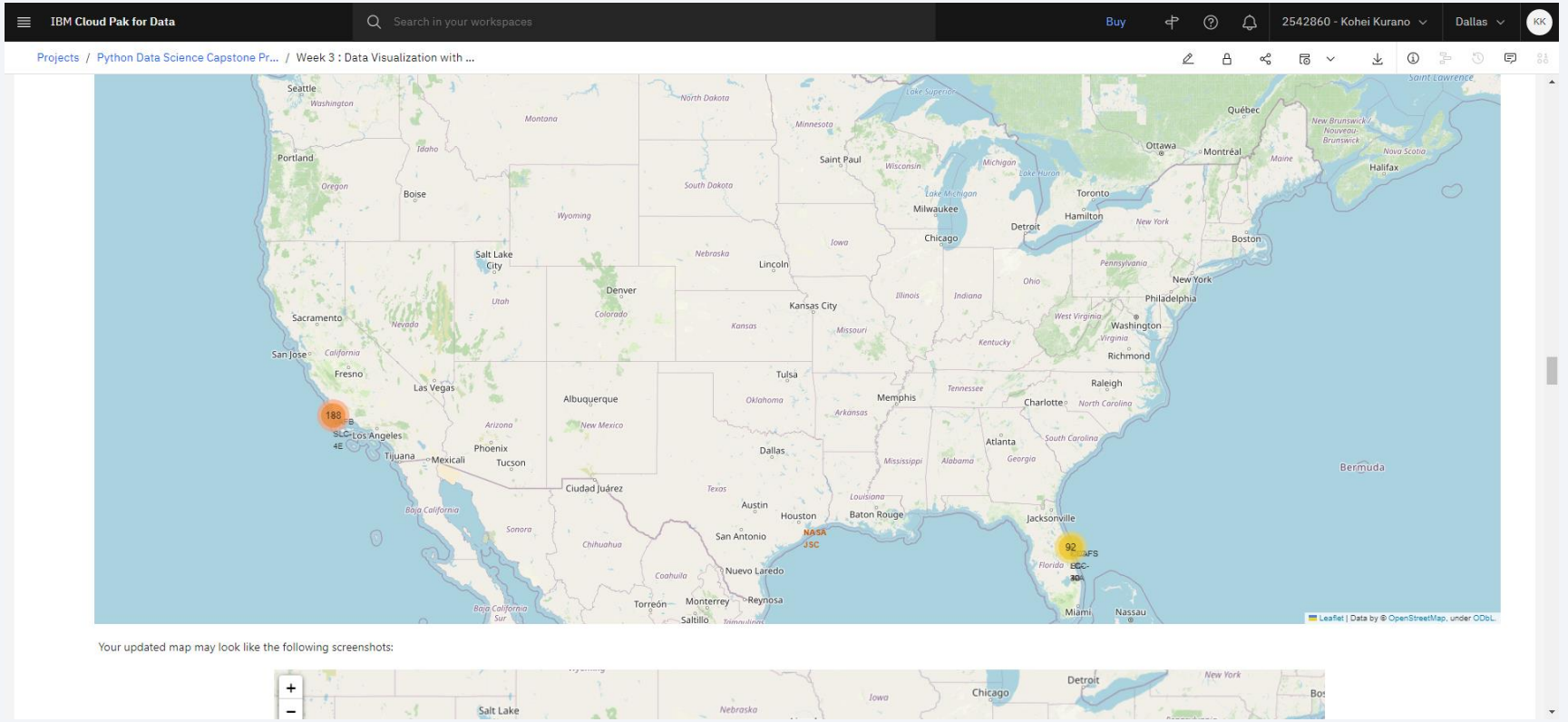
```
%sql select LANDING__OUTCOME, COUNT(*) from SPACEX where DATE between '2010-06-04' and '2017-03-20' group by LANDING__OUTCOME order by COUNT(*) DESC
```

landing__outcome	2
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

This query returns landing outcomes and their count in the time between 2010-06-04 and 2017-03-20. The GROUPBY clause groups results by landing outcome and ORDERBY COUNT DESC shows the results in descending order

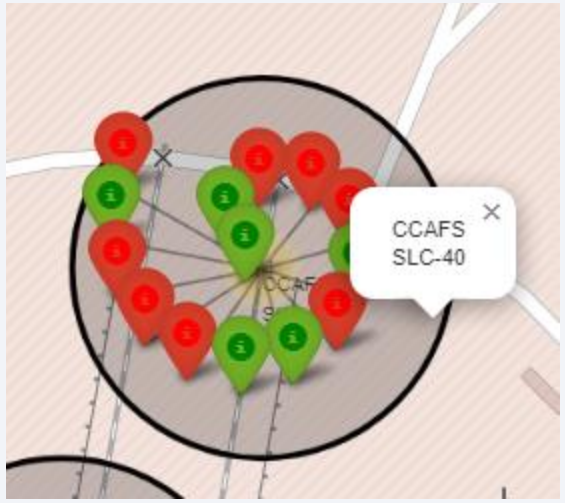

Section 3
Launch Sites
Proximities Analysis

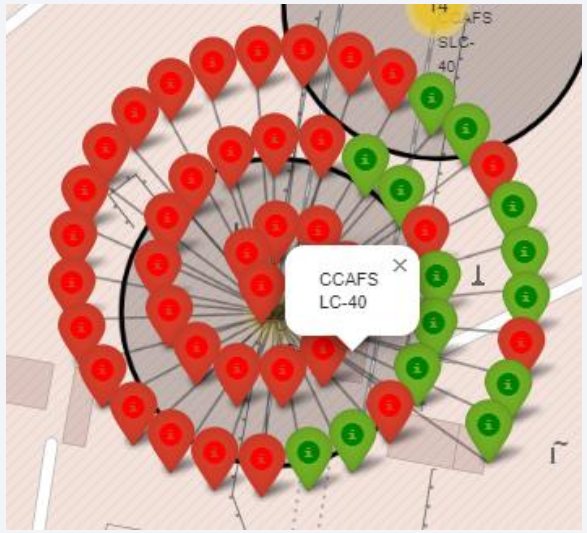
Folium Map – Ground stations

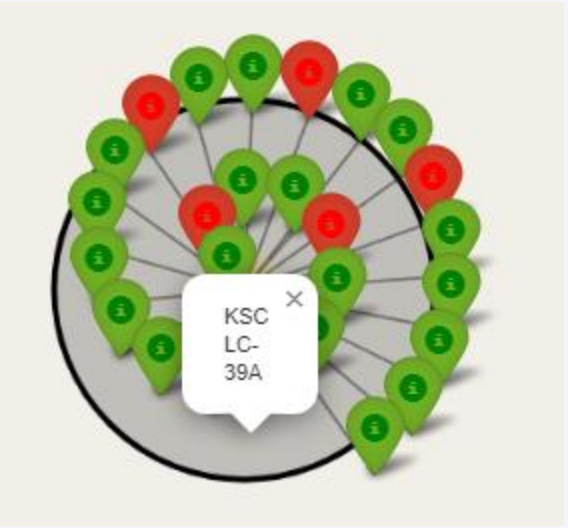


SpaceX launch sites are located on the coast of the United States

Folium Map – Color Labeled Markers

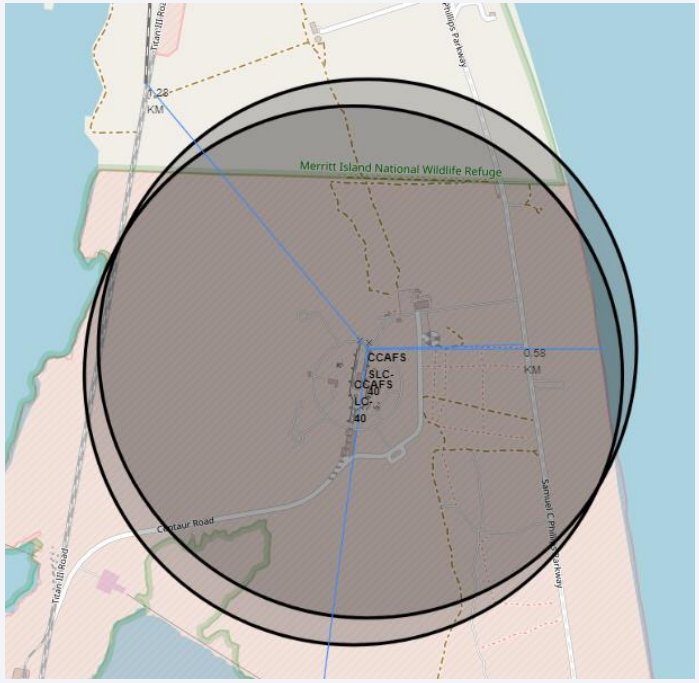




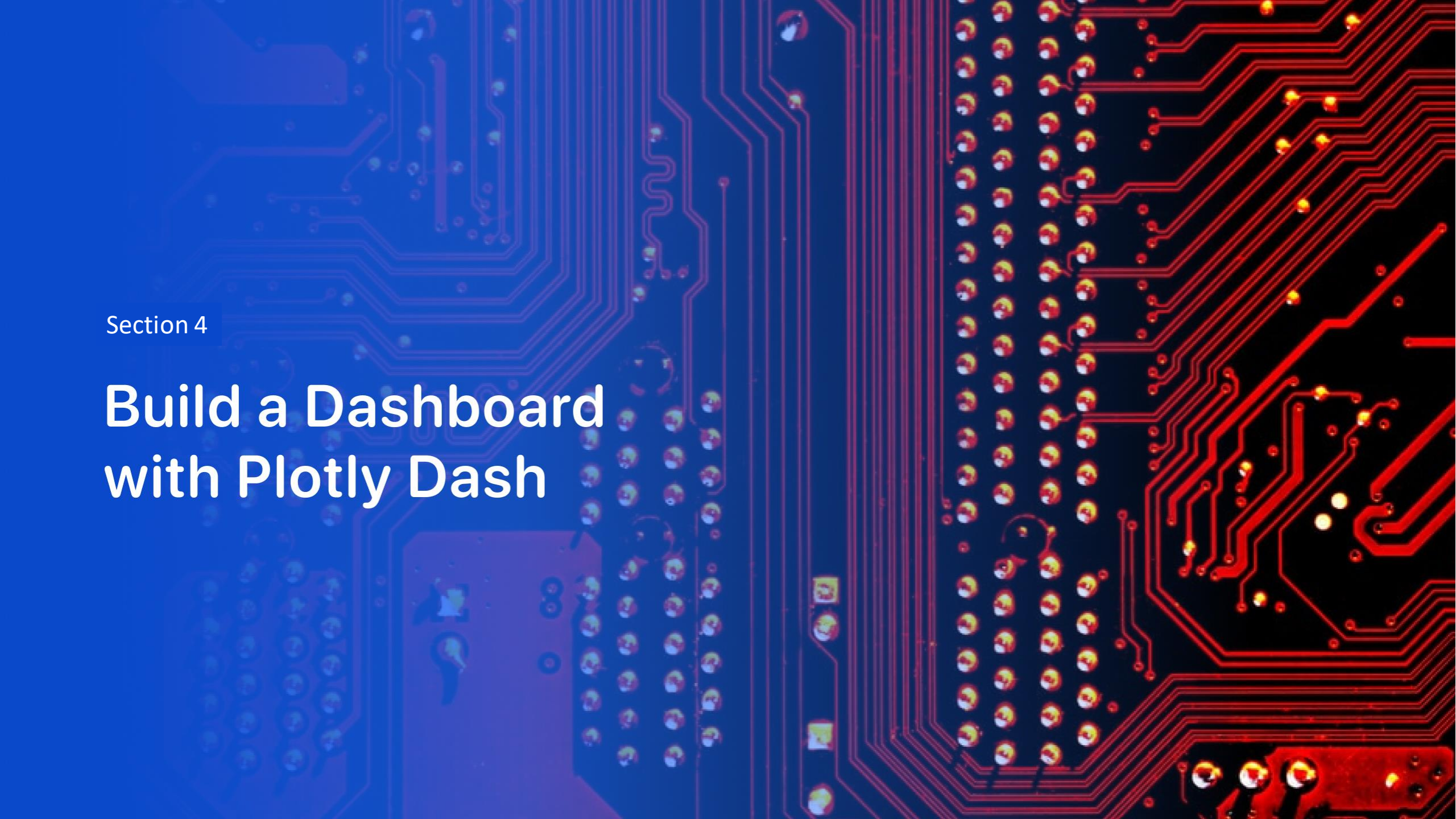


Green marker represents successful launches. Red marker represents unsuccessful launches.
We note that KSC LC-39A has a higher launch success rate

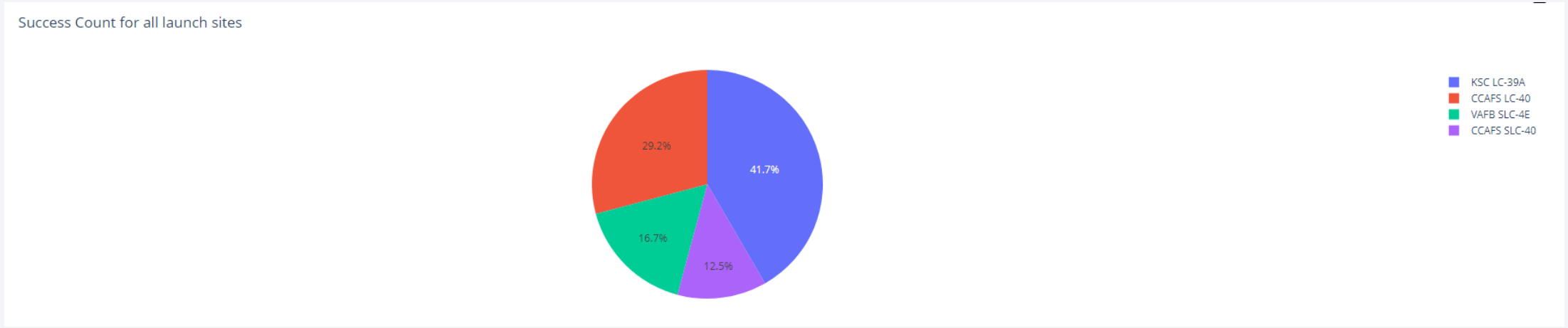
Folium Map – Launch site and its proximities



CCAFS SLC-40 is in close proximity to railways, highways, and coastline

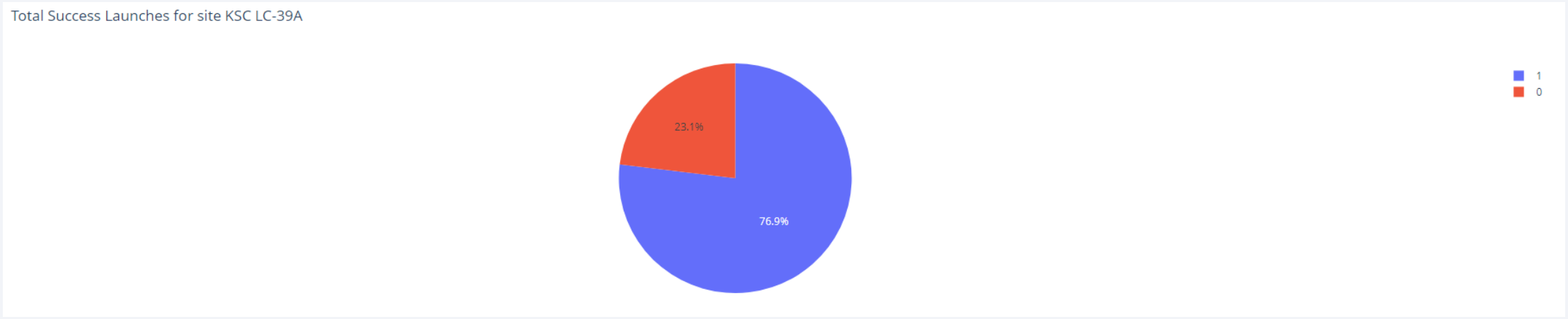

Section 4
Build a Dashboard with Plotly Dash

Dashboard – Total success by Site



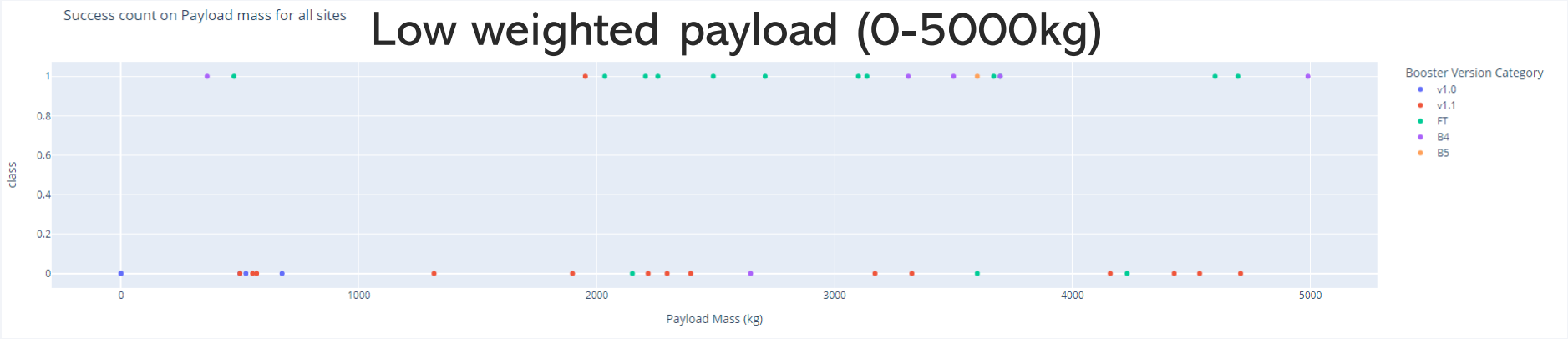
We note that KSC LC-39A has the best success rate of launches

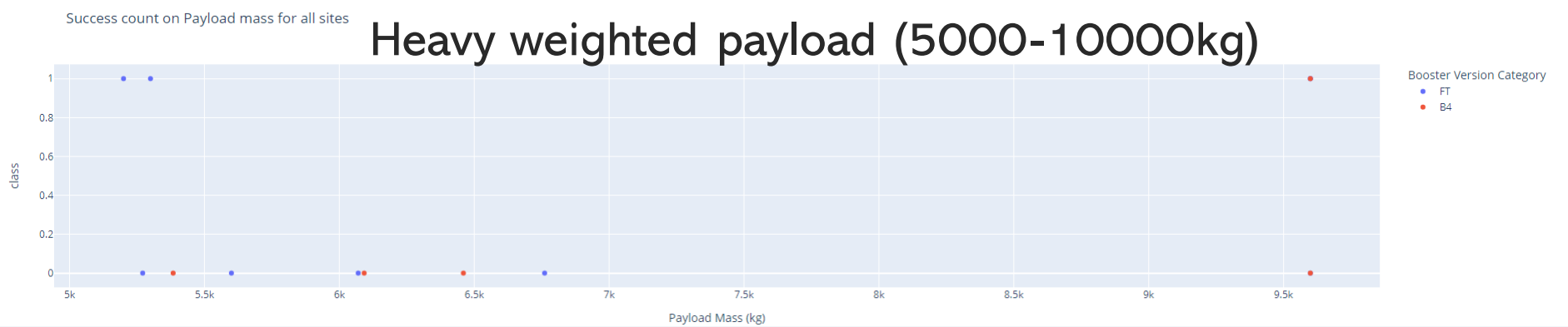
Dashboard – Total success launches for KSC LC-39A



We note that KSC LC-39A has achieved 76.9% success rate while getting 23.1% failure rate

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





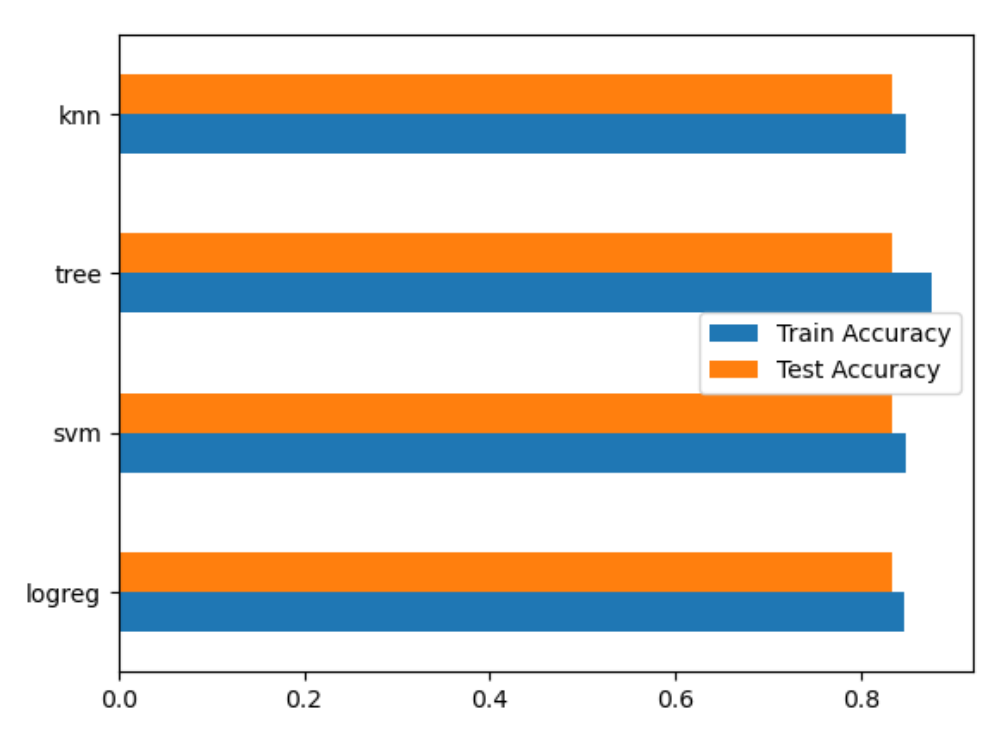
Low weighted payloads have higher success rate than heavy weighted payloads

Section 5

Predictive Analysis (Classification)

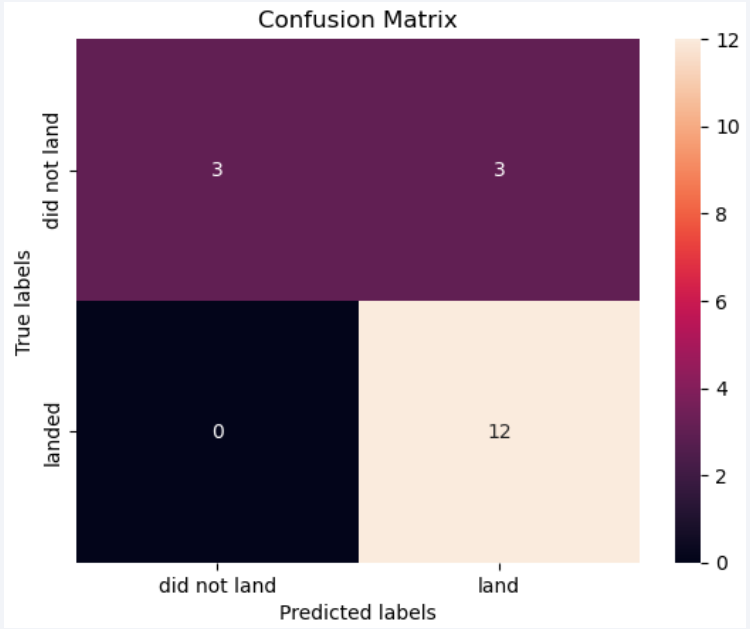
Classification Accuracy

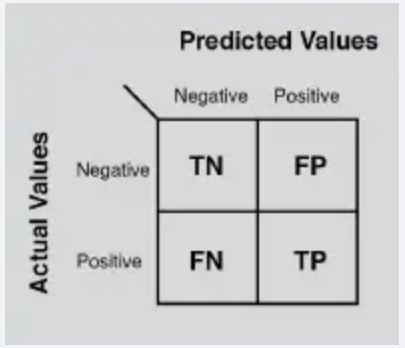
	Train Accuracy	Test Accuracy
logreg	0.846429	0.833333
svm	0.848214	0.833333
tree	0.876786	0.833333
knn	0.848214	0.833333



All methods performed similar in test accuracy. Base on train accuracy score, I would choose decision tree to be the best one

Confusion Matrix





As the test accuracy are all equal, the confusion matrix are also identical. 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.
- The orbits with the best success rates are GEO, HEO, SSO, ES-L1
- Depending on the orbits, the payload mass can be criterion to take into account for the success of a mission.
- With the current data, we cannot explain why some launch sites are better than others. 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.

Thank you!