IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Liping Tseng
6/6/2024

Outline

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

Executive Summary

- We will predict if the SpaceX Falcon 9 first stage will land successfully using several machine learning classification algorithms.
- The main steps in this project include: • Datacollection,wrangling,andformatting
  • Exploratorydataanalysis
  • Interactivedatavisualization • Machinelearningprediction
- Our graphs show that some features of the rocket launches have a correlation with the outcome of the launches, i.e., success or failure.
- It is also concluded that decision tree may be the best machine learning algorithm to predict if the Falcon 9 first stage will land successfully.

Introduction

- In this capstone, we will predict if the Falcon 9 first stage will land successfully. SpaceX 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 SpaceX 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 SpaceX for a rocket launch.
- The main question that we are trying to answer is, for a given set of features about a Falcon 9 rocket launch which include its payload mass, orbit type, launch site, and so on, will the first stage of the rocket land successfully?

Section 1

Methodology

Methodology

Executive Summary

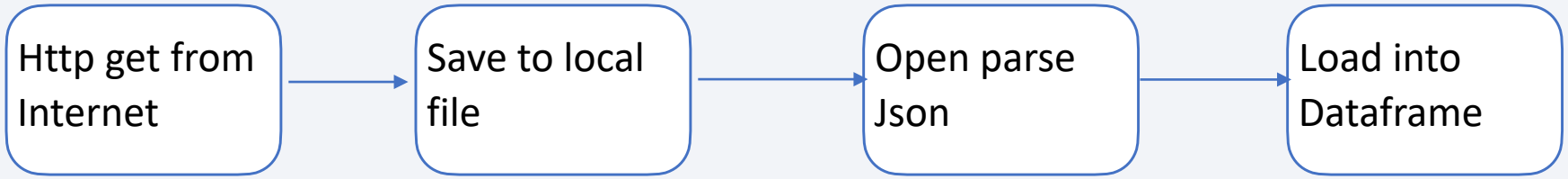
- Data collection methodology:
  - Data API (SpaceX)
  - Web scraping
- Perform data wrangling
  - Remove / replace missing. Parsing using BeautifulSoup for table
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Using Pandas and numpy together with SQL query
- Perform interactive visual analytics using Folium and Plotly Dash
  - Using Dash/ Plotly on webserver/browser, and Folium in Jupyterlab env.
- Perform predictive analysis using classification models
  - Built 4 modules including Logistic regression, Support Vector Machine, Decision Tree and K-nearest neighbors

Data Collection

- Describe how data sets were collected.

  Source file from internet, need to get and convert it into Pandas data frame

- You need to present your data collection process use key phrases and flowcharts



Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook (https://github.com/lipingt/hello-world/blob/master/ibm/jupyter-labs-spacex-data-collection-api.ipynb, as an external reference and peer-review purpose

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

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

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

Check the content of the response

```
print(response.content)
```

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwo
```

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

```
response.status_code
```

```
200
```

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

```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

Data Collection - Scraping

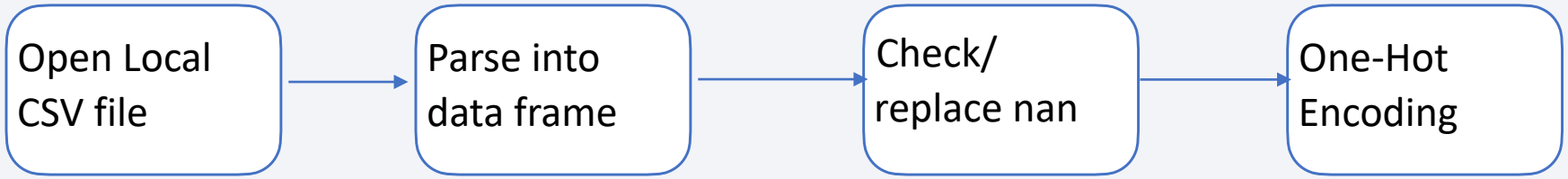
- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook, https://github.com/lipingt/hello-world/blob/master/ibm/jupyter-labs-webscraping.ipynb

```
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()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            launch_dict['Date'].append(date)
            #print(date)

            # Time value
            # TODO: Append the time into launch_dict with key `Time`
            time = datatimelist[1]
            launch_dict['Time'].append(time)
            #print(time)

            # Booster version
            # TODO: Append the bv into launch_dict with key `Version Booster`
            bv=booster_version(row[1])
            if not(bv):
                bv=row[1].a.string
            launch_dict['Version Booster'].append(bv)
            print(bv)

            # Launch Site
            # TODO: Append the bv into launch_dict with key `Launch Site`
            launch_site = row[2].a.string
            launch_dict['Launch site'].append(launch_site)
            #print(launch_site)

            # Payload
            # TODO: Append the payload into launch_dict with key `Payload`
            payload = row[3].a.string
            launch_dict['Payload'].append(payload)
            #print(payload)
```

Data Wrangling

- Check, replace nan value and one-hot encoding categorical features
- Ref: https://github.com/lipingt/hello-world/blob/master/ibm/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- Scatter charts to see relation on multiple features pairs.
- Bar chart to see grouped result for Orbit
- Line chart to see trend on Date and Class
- Ref: https://github.com/lipingt/hello-world/blob/master/ibm/edadataviz.ipynb

EDA with SQL

- Check unique Launch Sites
- Check CCA prefixed Launch Sites detail
- Query Payload_mass summation against multiple features
- Query min/max date with Landing Outcome condition
- Identify Fail/successful result
- Rank the Landing Outcomes result
- Ref: https://github.com/lipingt/hello-world/blob/master/ibm/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Added markers, circles, lines
- To local launch sites, and the distance to the boarder
- ref: https://github.com/lipingt/hello-world/blob/master/ibm/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

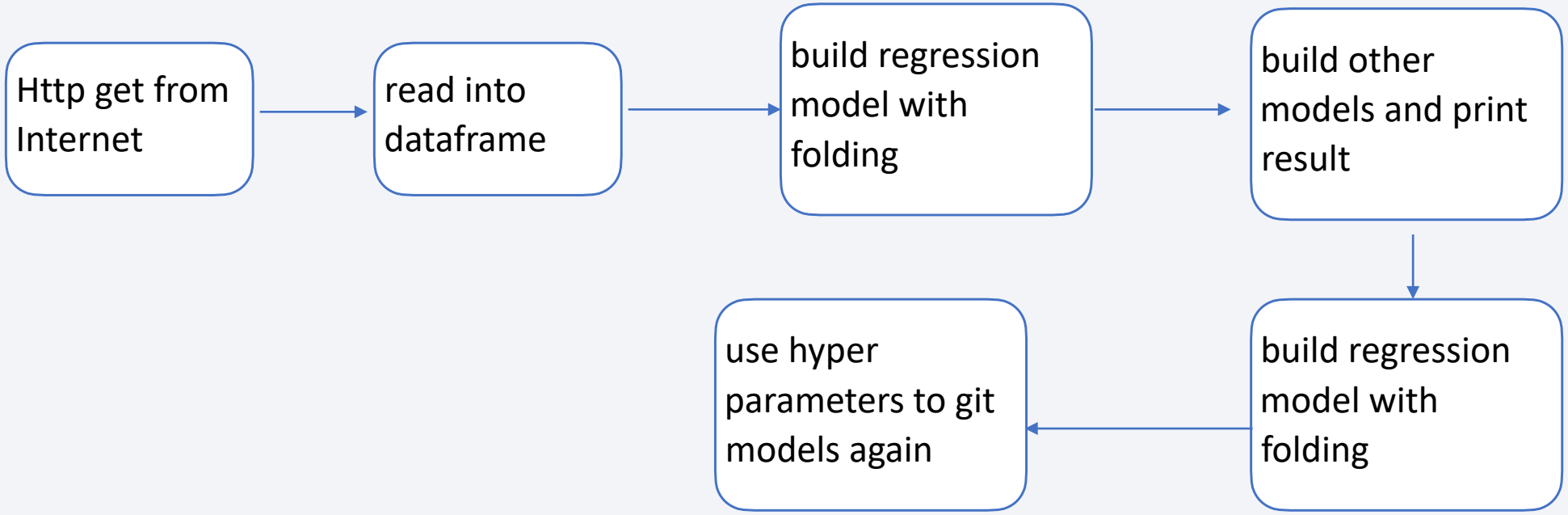
Functions from Das hare used to generate an interactive site where we can toggle the input

using a dropdown menu and a range slider.
• The total success launches from each launch site

•The correlation between payload mass and mission outcome for each launch site

- ref: https://github.com/lipingt/hello-world/blob/master/ibm/spacex_dash_app.py

Predictive Analysis (Classification)



- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

Results

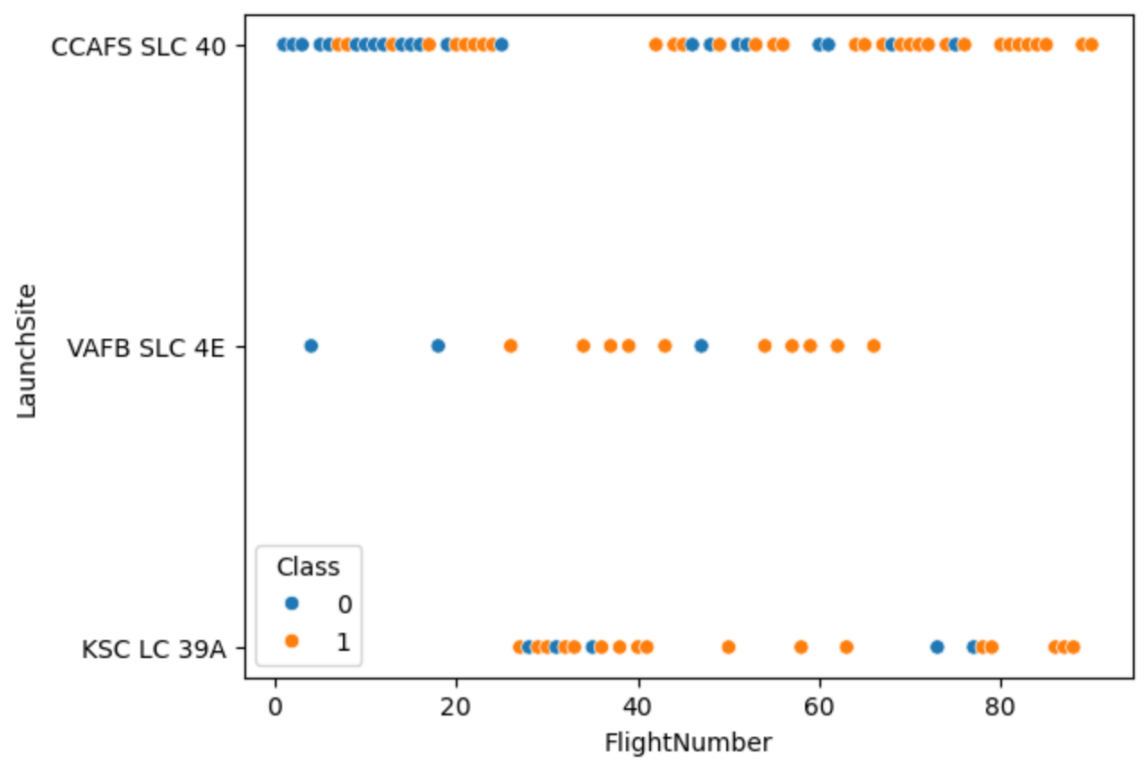
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

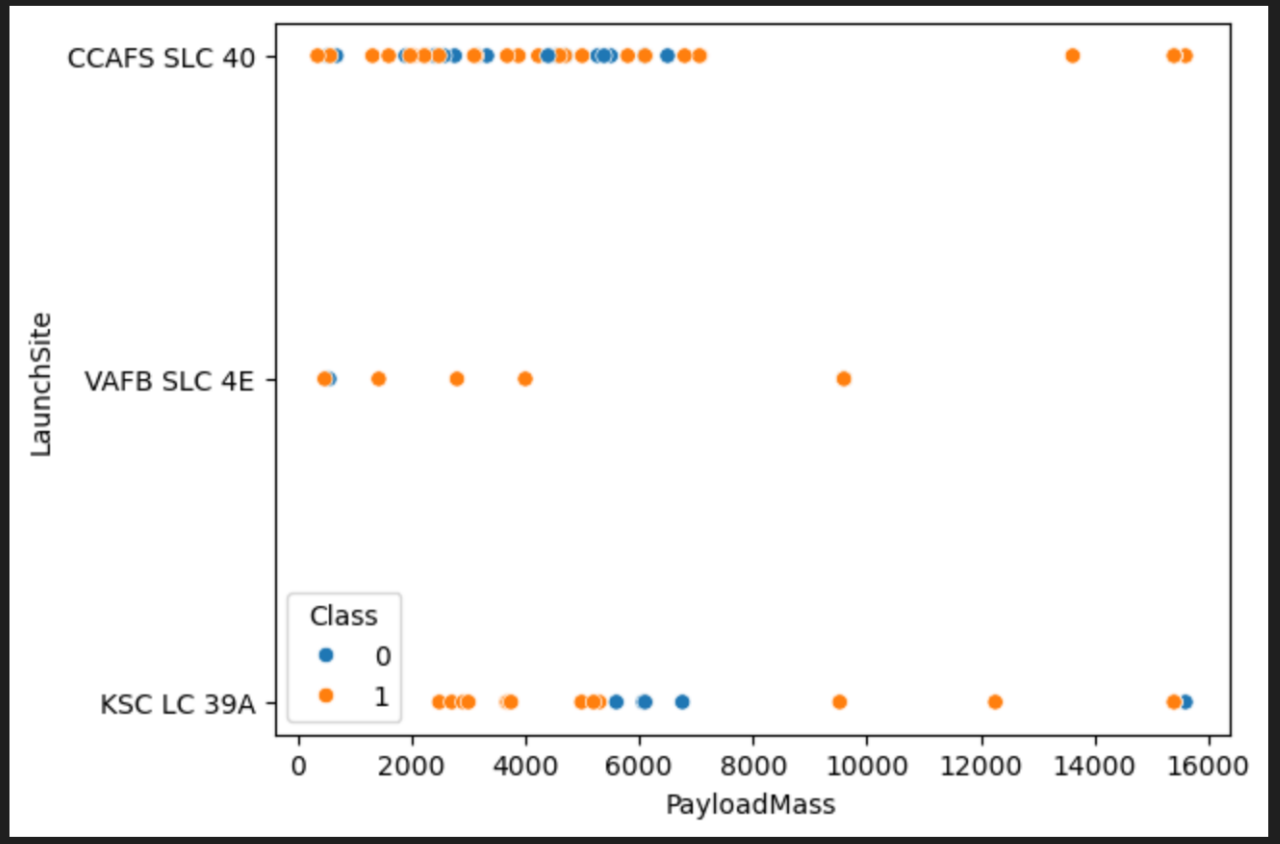
```
plt.figure(figsize=(14,8))
sns.scatterplot(x="FlightNumber", y="LaunchSite", hue="Class", data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Payload vs. Launch Site

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the laun
sns.scatterplot(x="PayloadMass",y="LaunchSite", hue ="Class",data=df)
```

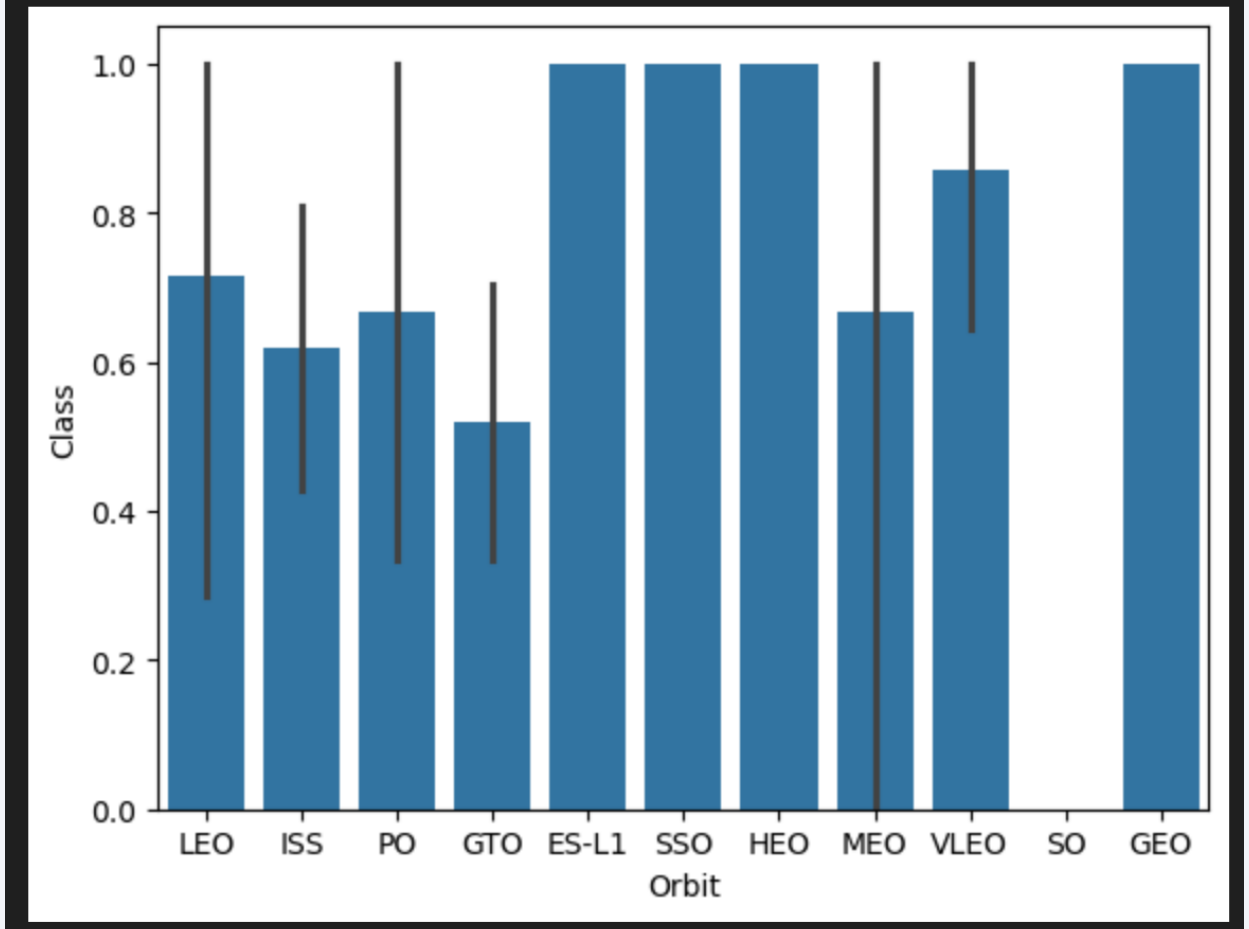
```
<AxesSubplot:xlabel='PayloadMass', ylabel='LaunchSite'>
```



Success Rate vs. Orbit Type

```
sns.barplot(data=df, x="Orbit",y="Class")
```

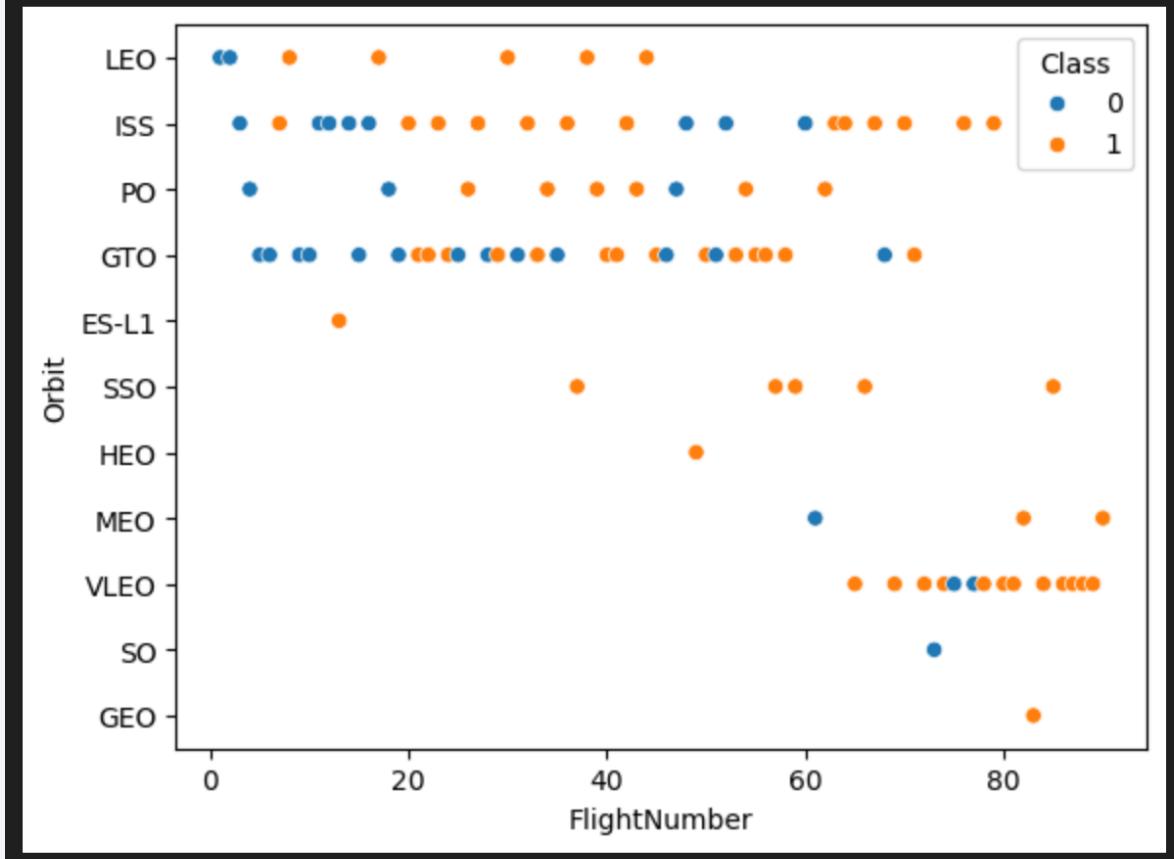
```
<AxesSubplot:xlabel='Orbit', ylabel='Class'>
```



Flight Number vs. Orbit Type

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the
sns.scatterplot(data = df, x="FlightNumber",y="Orbit",hue="Class")
```

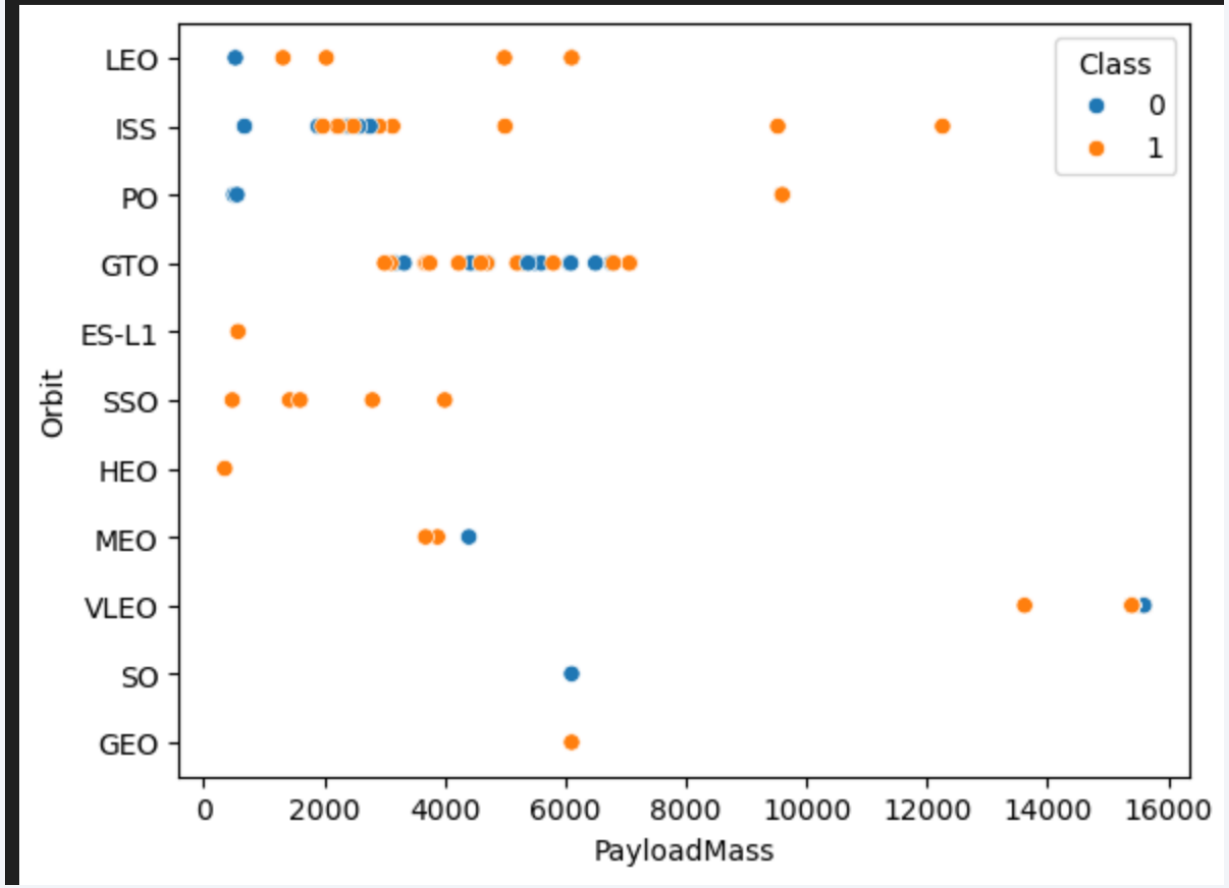
```
<AxesSubplot:xlabel='FlightNumber', ylabel='Orbit'>
```



Payload vs. Orbit Type

```
sns.scatterplot(data = df, x="PayloadMass",y="Orbit",hue="Class")
```

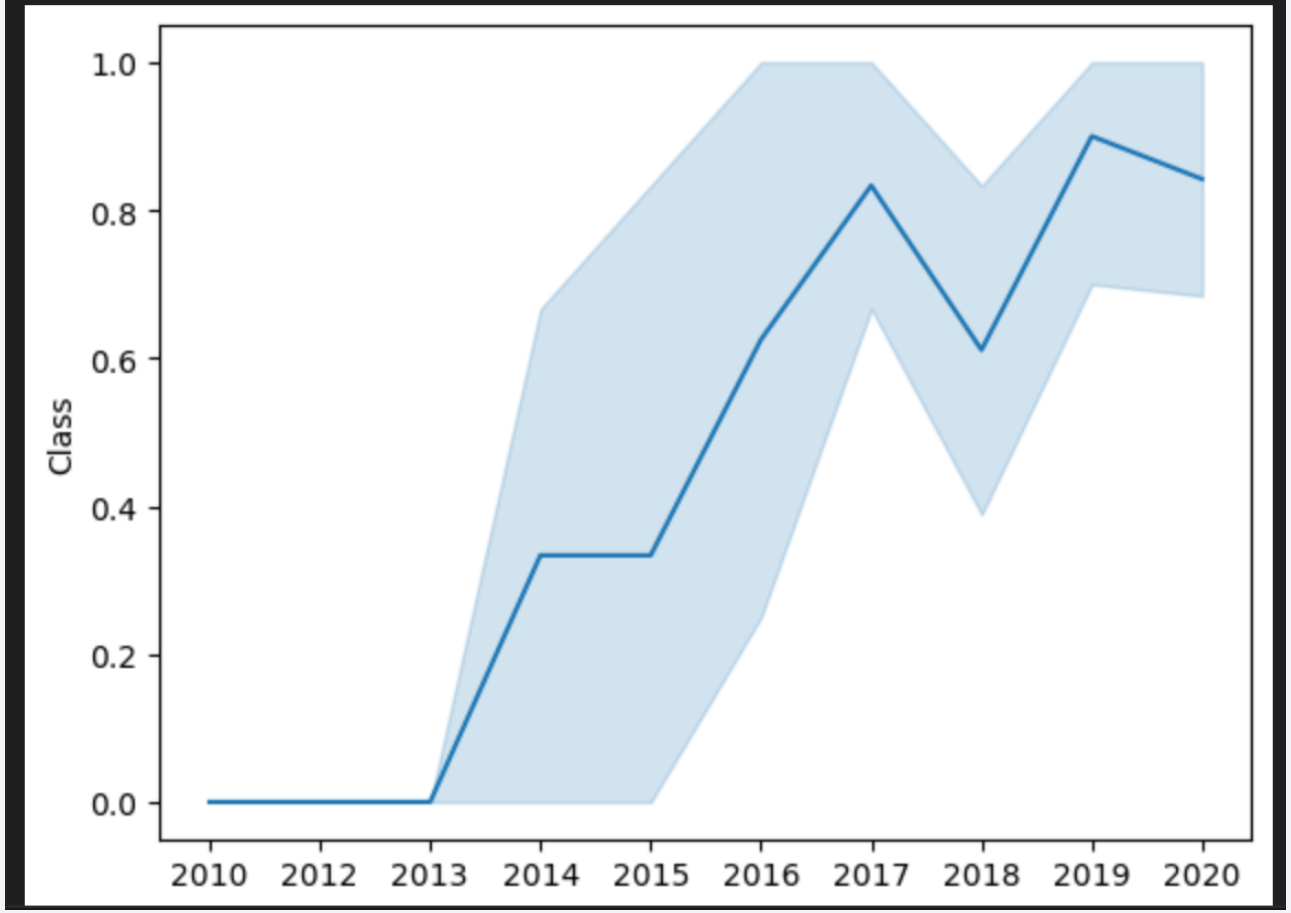
```
<AxesSubplot:xlabel='PayloadMass', ylabel='Orbit'>
```



Launch Success Yearly Trend

```
sns.lineplot(data=df, x="Date", y="Class" )
```

```
<AxesSubplot:xlabel='Date', ylabel='Class'>
```



All Launch Site Names

```
%sql select distinct(Launch_Site) from SPACEXTABLE;
```

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

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

Launch Site Names Begin with 'CCA'

```
%sql select * from SPACEXTABLE 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
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success

Total Payload Mass

Display the total payload mass carried by boosters launched by NASA (CRS)

```
]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer = 'NASA (CRS)';
```

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

]:

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version like 'F9 v1.1%';
```

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

avg(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

```
%sql select min(Date) from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)';
```

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

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select Booster_Version from SPACEXTABLE 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

Total Number of Successful and Failure Mission Outcomes

```
%sql select substr(Mission_Outcome, 1, 7), count(*) from SPACEXTABLE group by substr(Mission_Outcome, 1, 7);
```

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

substr(Mission_Outcome, 1, 7)	count(*)
Failure	1
Success	100

Boosters Carried Maximum Payload

```
%sql select Booster_Version from SPACEXTABLE where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTABLE);
```

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

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

2015 Launch Records

```
%sql select substr(Date, 6,2) as month, Booster_Version, Launch_Site from SPACEXTABLE
  where substr(Date,0,5)='2015' and Landing_Outcome = 'Failure (drone ship)';
```

```
 * 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

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

```
%sql select Landing_Outcome, count(*) as num from SPACEXTABLE
  where Date >= '2010-06-04' and Date <= '2017-03-20' group by Landing_Outcome order by num desc;
```

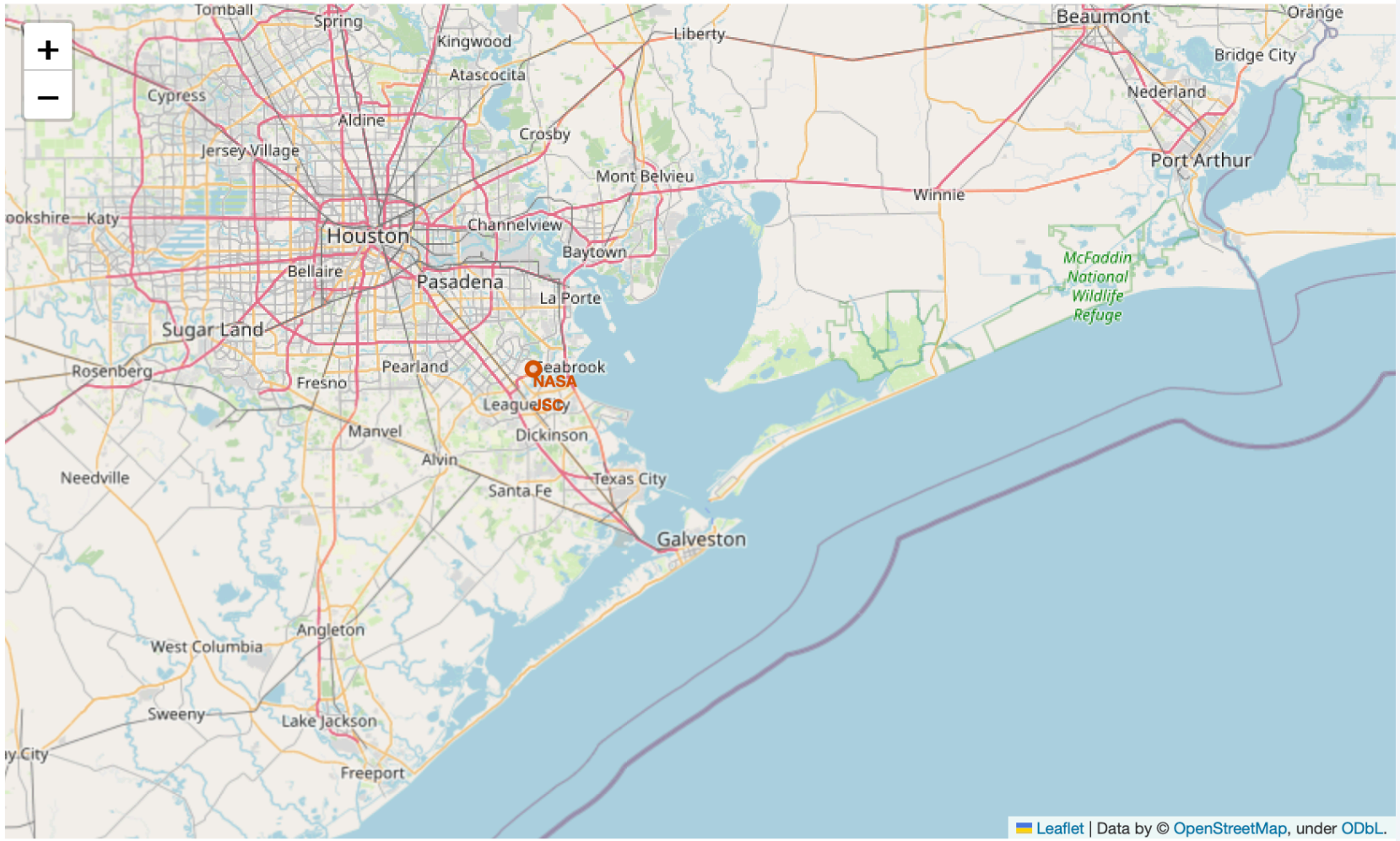
```
 * sqlite:///my_data1.db
Done.
```

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

Section 3

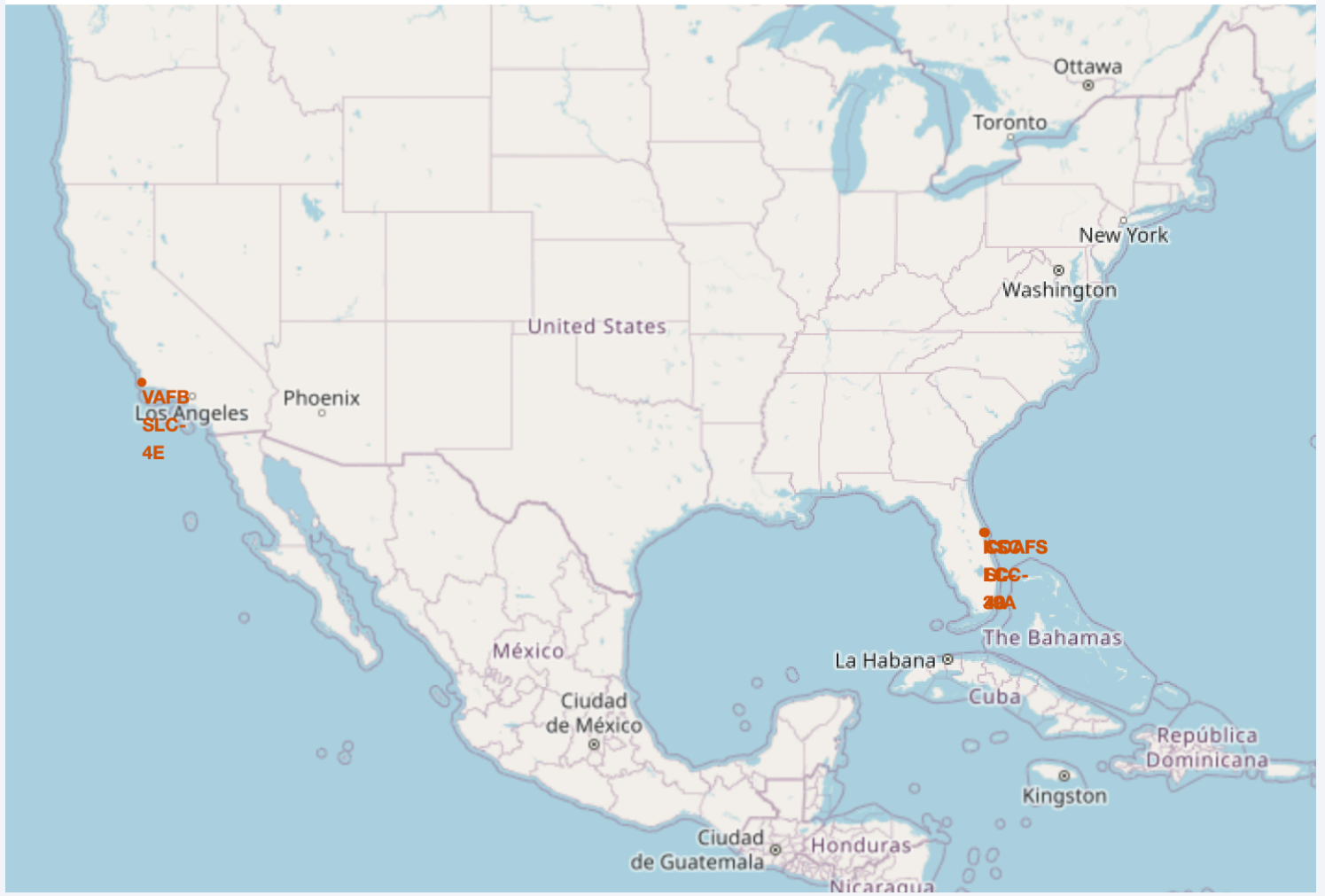
Launch Sites Proximities Analysis

Houston launch site

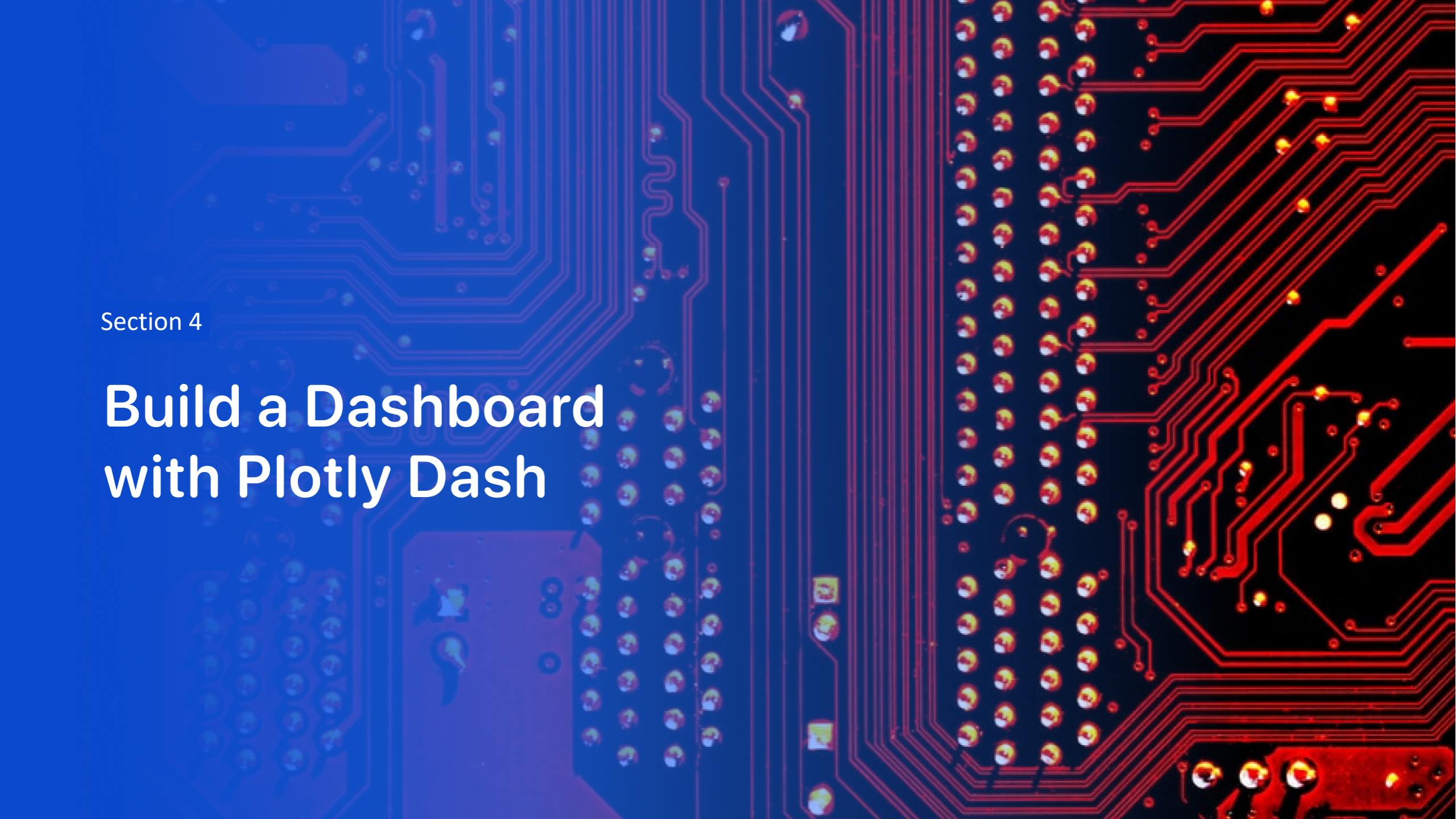


and you should find a small yellow circle near the city of Houston and you can zoom-in to see a larger circle.

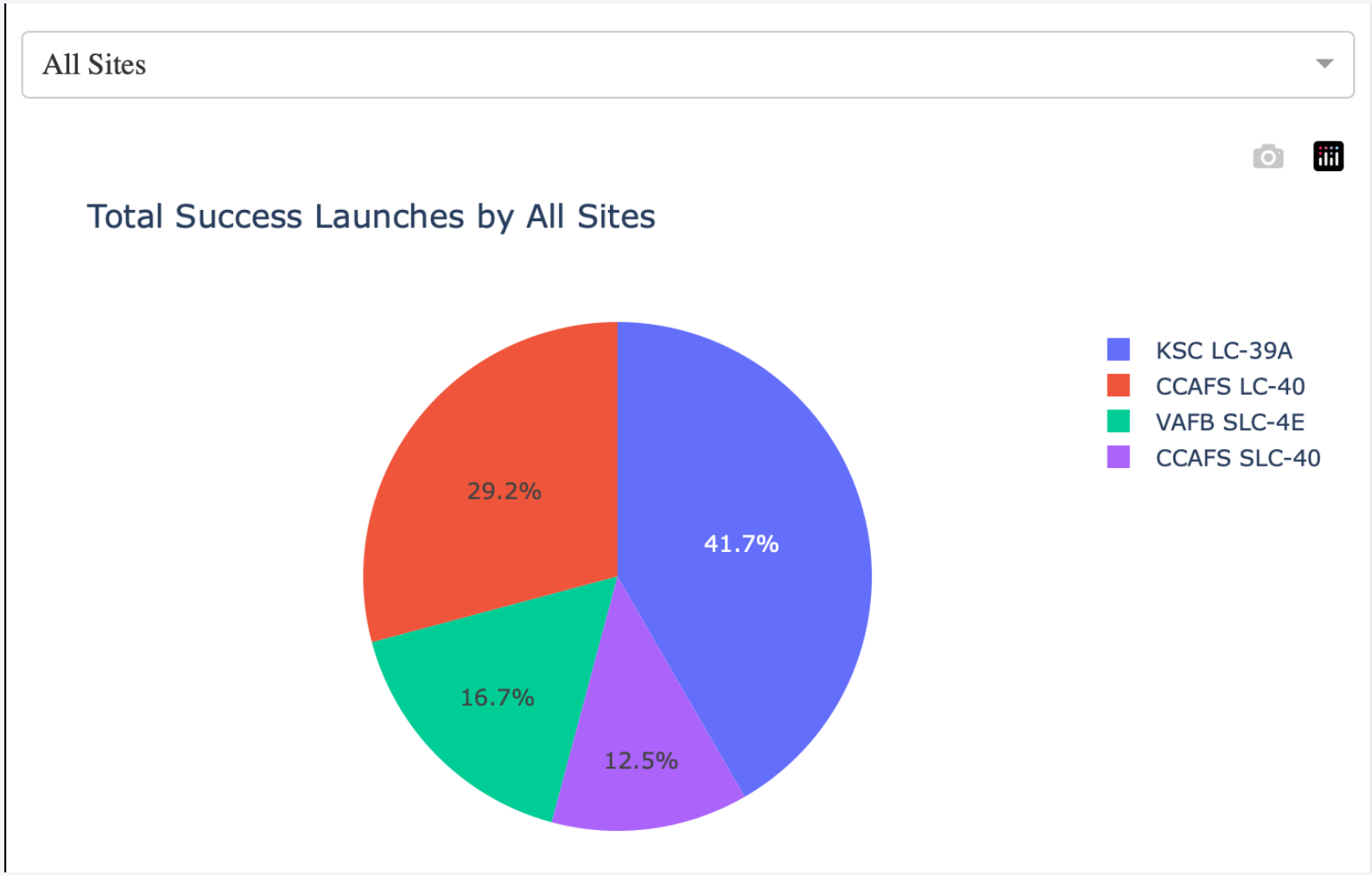
Circle Marker Launch Site



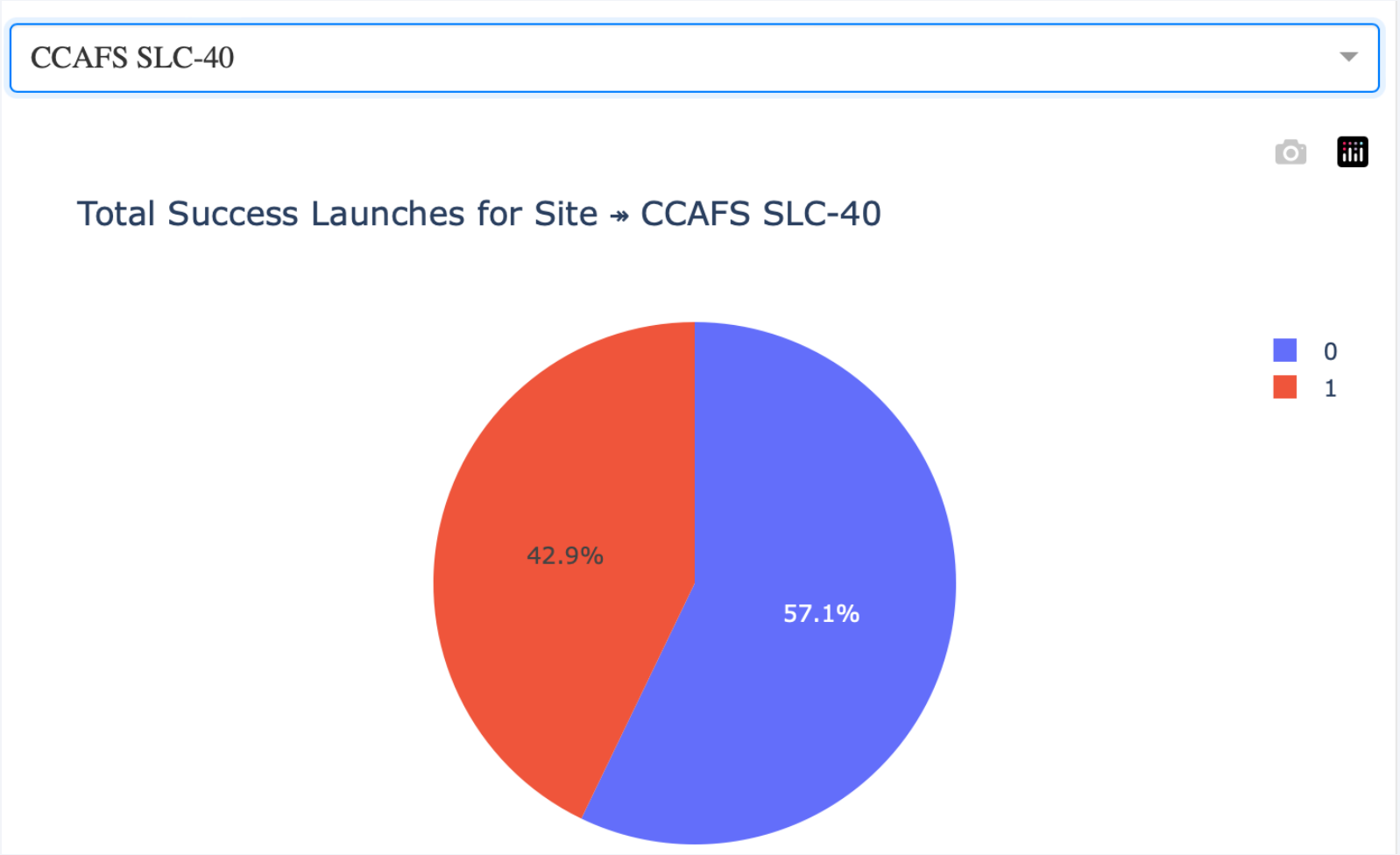
Color Launch site


Section 4
Build a Dashboard with Plotly Dash

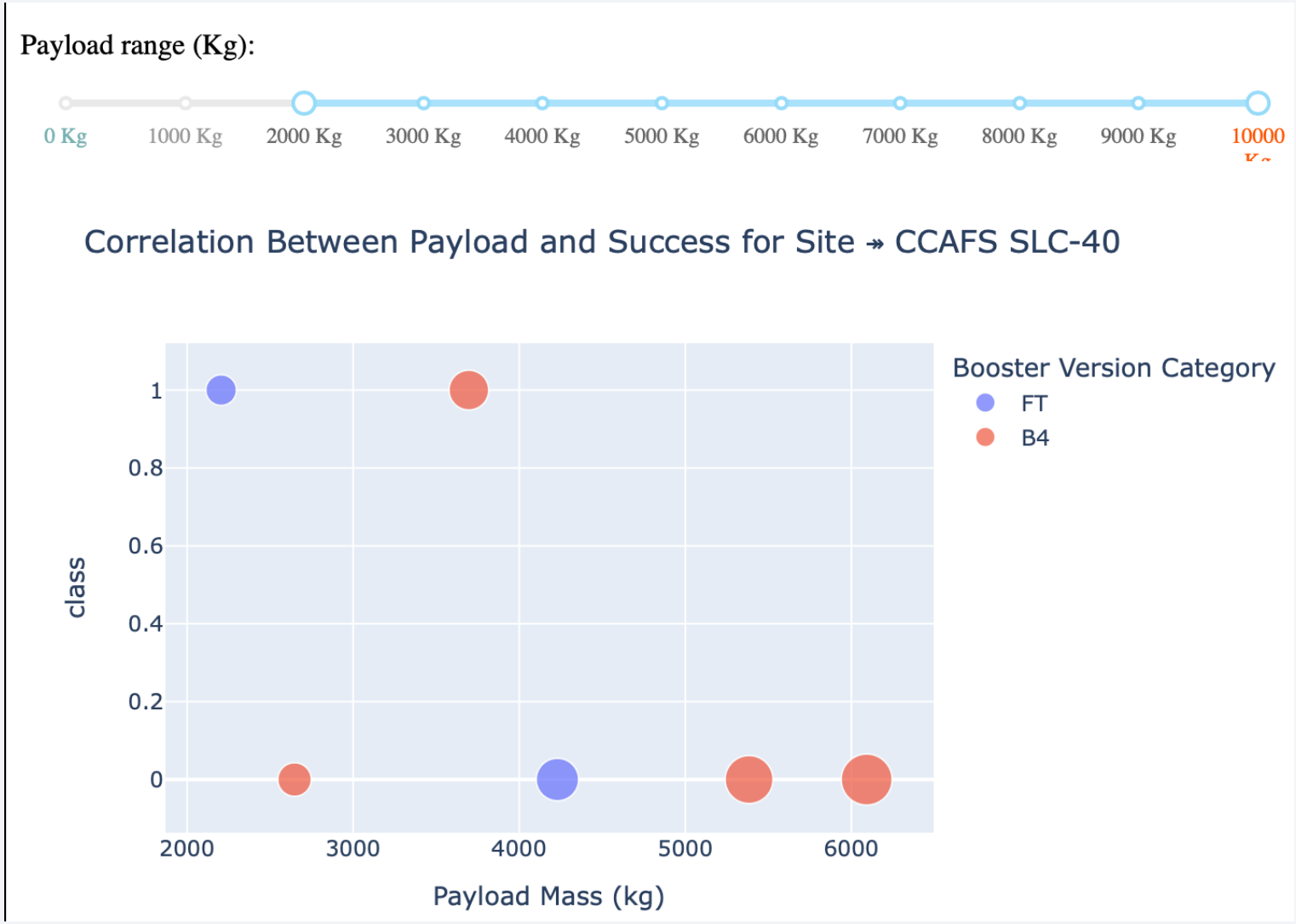
All Sites Success Launches



Highest Success Site



Correlation Payload-Launch Outcome



Section 5

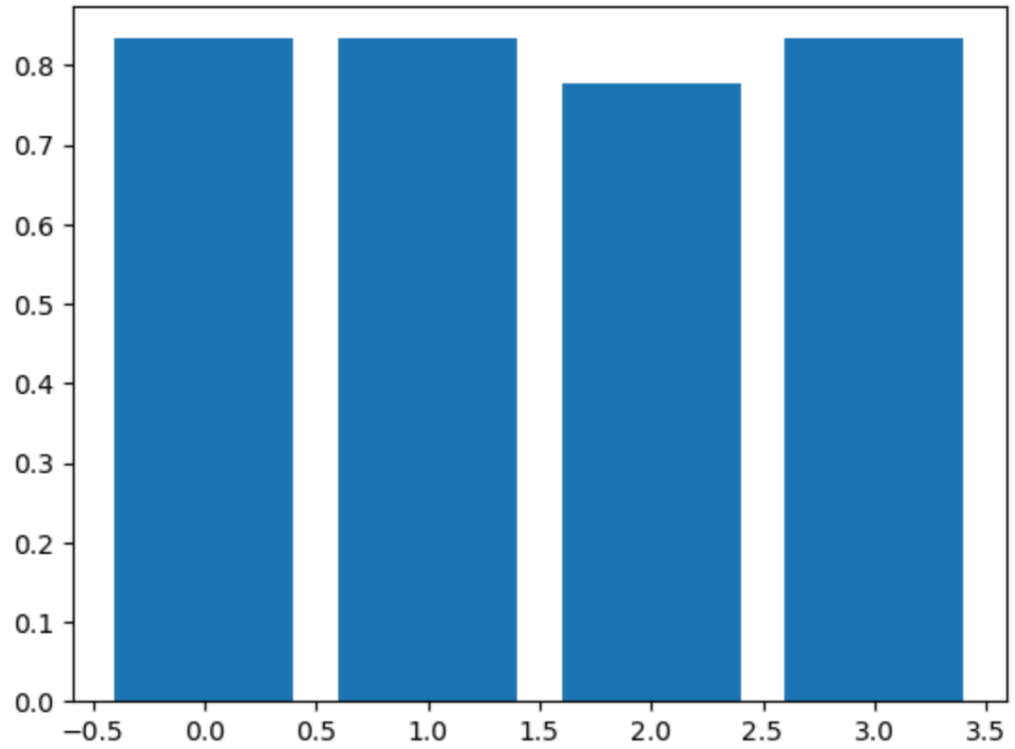
Predictive Analysis (Classification)

Classification Accuracy

```
4]: {'logistic regression': 0.8333333333333334,
     'svc': 0.8333333333333334,
     'decision tree': 0.7777777777777778,
     'knn': 0.8333333333333334}
```

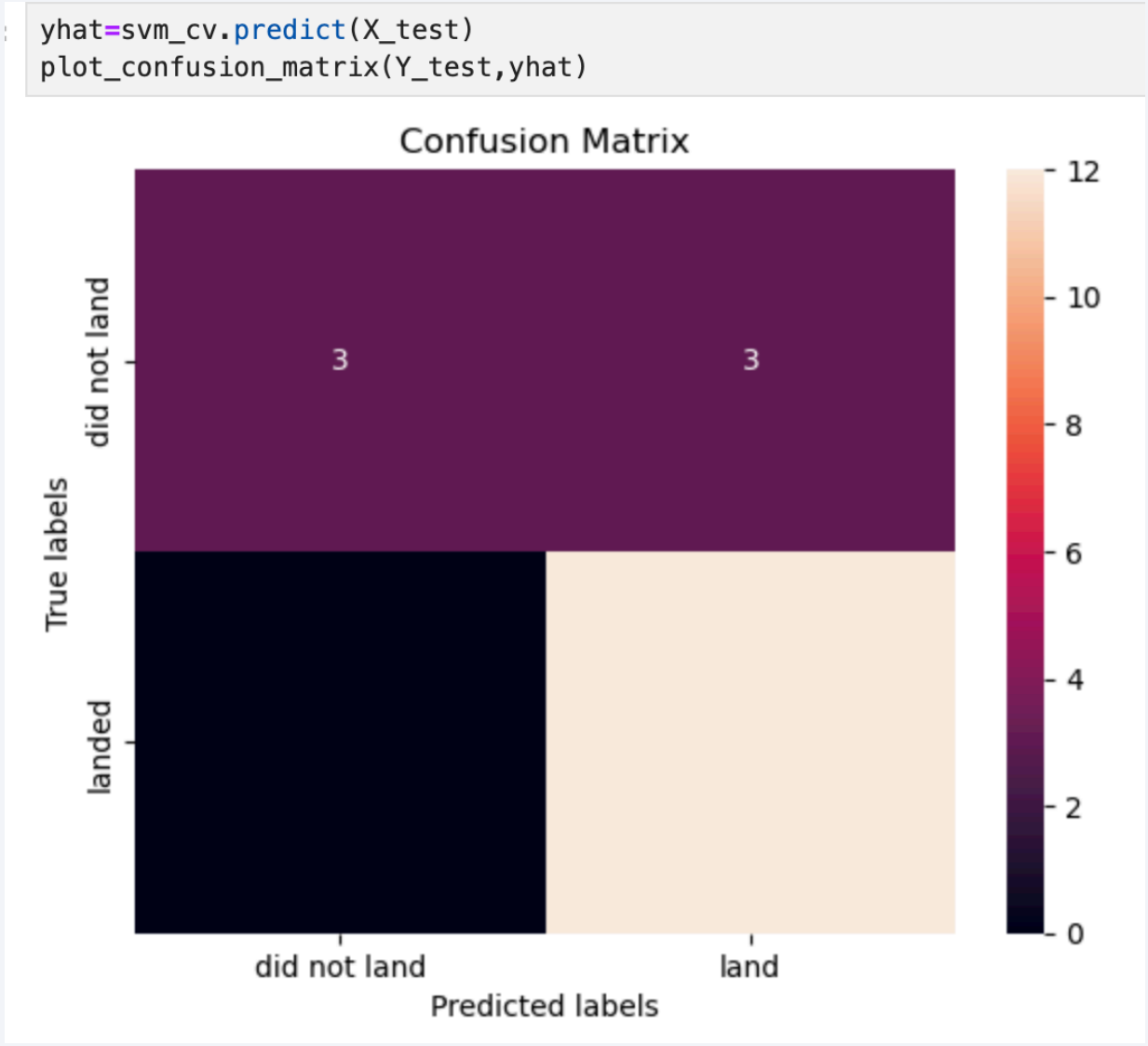
```
2]: # all the same for the test data
    plt.bar(range(len(scores)), list(scores.values()), align='center')
```

```
2]: <BarContainer object of 4 artists>
```



Confusion Matrix

```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- In this project, we try to predict if the first stage of a given Falcon 9 launch will land in order to determine the cost of a launch.
- Each feature of a Falcon 9 launch, such as its payload mass or orbit type, may affect the mission outcome in a certain way.
- Several machine learning algorithms are employed to learn the patterns of past Falcon 9 launch data to produce predictive models that can be used to predict the outcome of a Falcon 9 launch.
- The predictive model produced by decision tree algorithm performed the best among the 4 machine learning algorithms employed.

Appendix

- Data Collection API https://github.com/lipingt/hello-world/blob/master/ibm/jupyter-labs-spacex-data-collection-api.ipynb
- Webscraping: https://github.com/lipingt/hello-world/blob/master/ibm/jupyter-labs-webscraping.ipynb
- EDA SQ: https://github.com/lipingt/hello-world/blob/master/ibm/jupyter-labs-eda-sql-coursera_sqllite.ipynb
- Data Wranging: https://github.com/lipingt/hello-world/blob/master/ibm/labs-jupyter-spacex-Data%20wrangling.ipynb
- EDA Visualization: https://github.com/lipingt/hello-world/blob/master/ibm/edadataviz.ipynb

Thank you!