IBM DATA SCIENCE CAPSTONE PROJECT

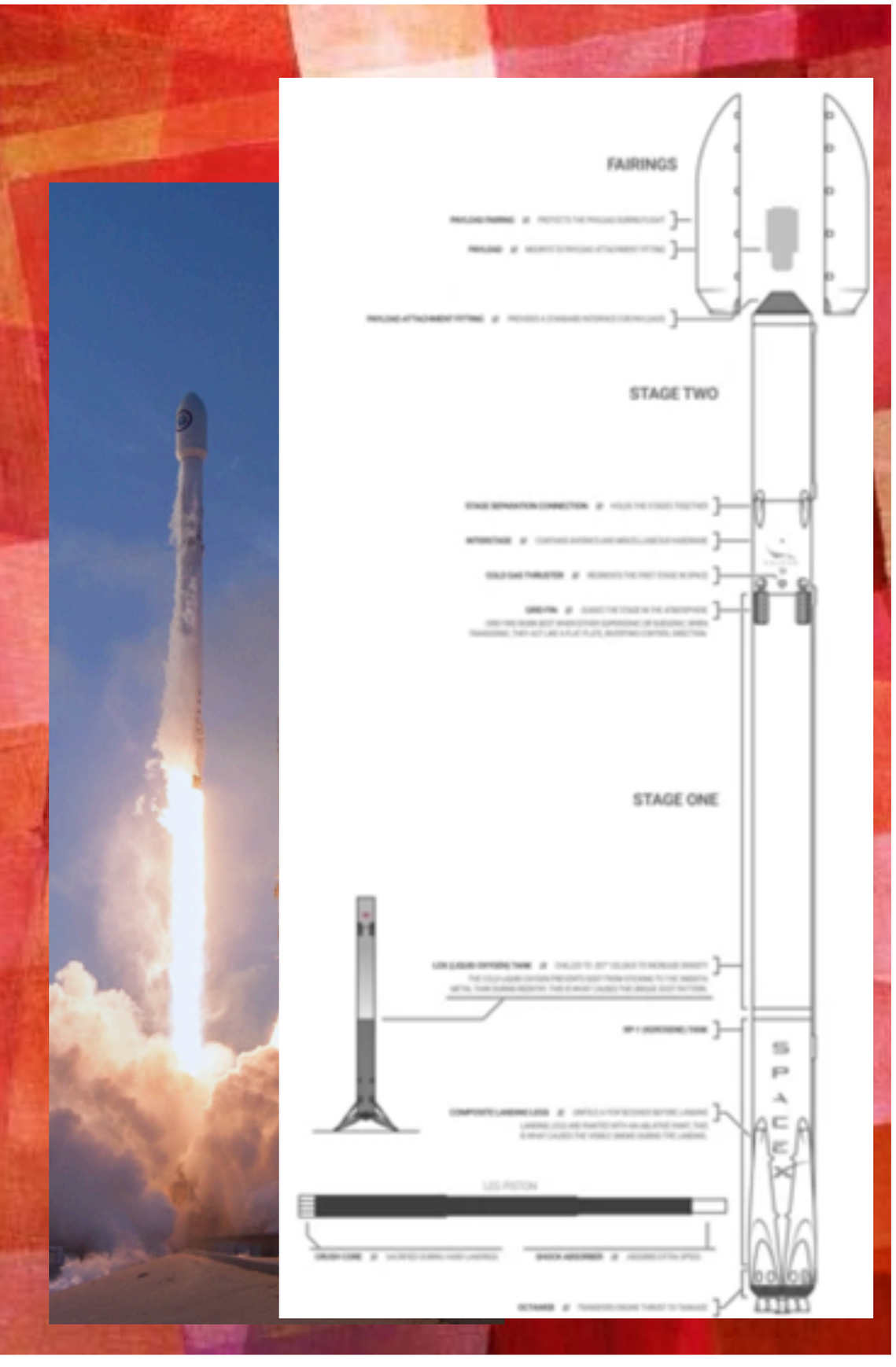
WINNING SPACE RACE WITH DATA SCIENCE

Rima Hinnawi

March 2022

OUTLINE

- Introduction
- Executive Summary
- Methodology
- Results
- Conclusion



INTRODUCTION

Project Background and Context

- SpaceX rocket launches are less expensive than its competitors in space travel.
- SpaceX advertises Falcon 9 rocket launches on its website for 62 million dollars while other rocket launches cost is more than 165 million dollars each.
- SpaceX can save because it can recover and reuse the first stage of launch.
- If we can determine if the first stage will land(which means it can be reused), we can determine the cost of a launch.
- SpaceX´s 'Falcon 9' launch like regular rockets. To understand the scale of 'Falcon 9', we can look at the diagrams of the different components of the 'Falcon 9'. The first stage is the largest.
- If we can determine if the first stage will land, we can determine the cost of a launch. That information can be used for SpaceY to be able to bid against SpaceX.

Problem we are trying to answer

- Will gather publicly available information about SpaceX and Data Science methodologies determine if the first stage will land.
- Determine if Rocket will land successfully
- Understand what needs to happen to achieve best results
- Analysis can be used by company SpaceY that would like to compete with well known company SpaceX

EXECUTIVE SUMMARY

➤ Summary of Methodologies

➤ Data Collection

- Used SpaceX Rest API to get data about launches including info about rocket used, payload used, launch specifications, landing specifications and landing outcome.

- Used “Web Scraping” with “BeautifulSoup” from Wikipedia.

➤ Data Wrangling

Dealt with missing values, identified categorical and numerical values and simplified launch outcome to 1 (for successful) or 0 (for unsuccessful).

➤ Exploratory data analysis (EDA)

- Used Python with Pandas, SQL, Matplotlib and Seaborn

- Interactive visual analytics using Folium and Plotly Dash

➤ Predictive analysis

➤ Summary of Results

METHODOLGY

METHODOLOGY

- Data Collection
  - By using SpaceX Rest API. This API gives us data about launches including info about rocket used, payload used, launch specifications, landing specifications and landing outcome.
  - By using “Web Scraping” with “BeautifulSoup” from Wikipedia
- Data Wrangling
  - Dealt with missing values, identified categorical and numerical values and simplified launch outcome to 1 (for successful) or 0 (for unsuccessful)
- Exploratory data analysis (EDA)
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis

DATA COLLECTION DETAILS- GENERAL

1.Get Data from SpaceX API
Or
Through Webscraping.

2. API returns json file.
Save result into a Dataframe
so that we can work with it.

3. Filter Data so we get
only information we need.
In this case, only FALCON 9 Data is needed.

4. Save results
into a CSV file

DATA COLLECTION DETAILS- SPACEX API

[Code details on GitHub]

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

1. Make request to SpaceX API and save response

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

2. API returns response with .json file
Save data into dataframe using json_normalize

3. Use custom functions to clean Data

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

4. construct dataset using the collected data. We

5. Filter Data so we only get Falcon 9 Data

```
data_falcon9= data.loc[data['BoosterVersion']!='Falcon 1']
data_falcon9
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount
4	6	2010-06-04	Falcon 9	None	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1	0
5	8	2012-05-22	Falcon 9	525	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1	0
6	10	2013-03-01	Falcon 9	677	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1	0
7	11	2013-09-29	Falcon 9	500	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1	0
8	12	2013-12-03	Falcon 9	3170	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1	0

6. Save results into a CSV file

DATA COLLECTION DETAILS- WEBSCRAPING

Request the Falcon9 Launch Wiki page from its URL and save response

```
# use requests.get() method with the provided static_url

response = requests.get(static_url)

# assign the response to a object

response_text= response.content
```

[Code Details on GitHub]

2. Create a BeautifulSoup object from the HTML response

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

3. Find html tables

4. Extract column/variable names from the HTML table header

5. Create Dictionary of Column names

6. Convert Dictionary to dataframe

7. Save dataframe results into a CSV file

DATA WRANGLING

In the next step, I will work on finding some patterns in the data to determine the label for training supervised models.

The dataset shows that there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident. For example, **True Ocean** means the mission outcome was successfully landed to a specific region of the ocean while **False Ocean** means the mission outcome was unsuccessfully landed to a specific region of the ocean. **True RTLS** means the mission outcome was successfully landed to a ground pad **False RTLS** means the mission outcome was unsuccessfully landed to a ground pad. **True ASDS** means the mission outcome was successfully landed on a drone ship **False ASDS** means the mission outcome was unsuccessfully landed on a drone ship.

After exploring the data, in this step, I will convert the above outcomes into Training Labels:
1 will mean the booster successfully landed and 0 means it was unsuccessful.

DATA WRANGLING

[Code Details on GitHub]

```
In [3]: df.isnull().sum()/df.count()*100
Out[3]: FlightNumber      0.000
        Date              0.000
        BoosterVersion    0.000
        PayloadMass       0.000
        Orbit             0.000
        LaunchSite        0.000
        Outcome           0.000
        Flights           0.000
        GridFins          0.000
        Reused            0.000
        Legs              0.000
        LandingPad       40.625
        Block             0.000
        ReusedCount       0.000
        Serial            0.000
        Longitude         0.000
        Latitude          0.000
        dtype: float64
```

1.Identify missing values

2.Identify numerical and categorical columns

```
In [4]: df.dtypes
Out[4]: FlightNumber      int64
        Date             object
        BoosterVersion   object
        PayloadMass     float64
        Orbit            object
        LaunchSite       object
        Outcome          object
        Flights           int64
        GridFins           bool
        Reused             bool
        Legs               bool
        LandingPad       object
        Block           float64
        ReusedCount       int64
        Serial           object
        Longitude       float64
        Latitude        float64
        dtype: object
```

3. Calculate number of launch sites and number of launches at each site. This information is found in the column LaunchSite

```
In [5]: # Apply value_counts() on column LaunchSite
        df.LaunchSite.value_counts()
Out[5]: CCAFS SLC 40    55
        KSC LC 39A      22
        VAFB SLC 4E     13
        Name: LaunchSite, dtype: int64
```

4. Calculate of times each orbit is used. This information is found in the column Orbit

```
[6]: # Apply value_counts on Orbit column
     df.Orbit.value_counts()
Out[6]: GTO      27
        ISS      21
        VLEO     14
        PO        9
        LEO       7
        SSO       5
        MEO       3
        ES-L1     1
        GEO       1
        HEO       1
        SO        1
        Name: Orbit, dtype: int64
```

5.Calculate the number of landings by orbit types.This information is found in the column Outcome

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

Out[9]: 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
```

6.Create a lists for bad outcomes(not landed) and good outcomes(landed)

```
In [10]: for i,outcome in enumerate(landing_outcomes.keys()):
             print(i,outcome)

0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

We create a set of outcomes where the second stage did not land successfully:

```
In [11]: bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
         bad_outcomes
Out[11]: {'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

7. Simplify result to either 1 (for not landed) or 0 (for landed) and add new column to dataframe

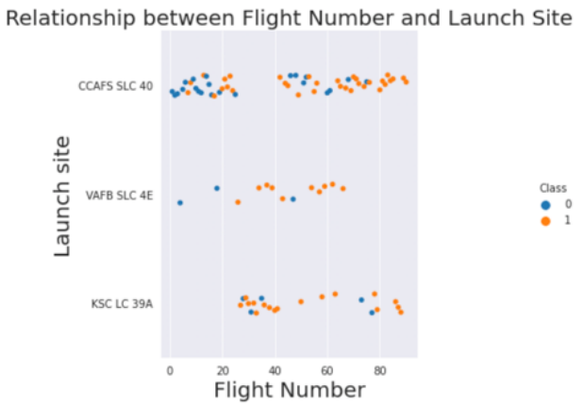
8. Save Result in CSV File

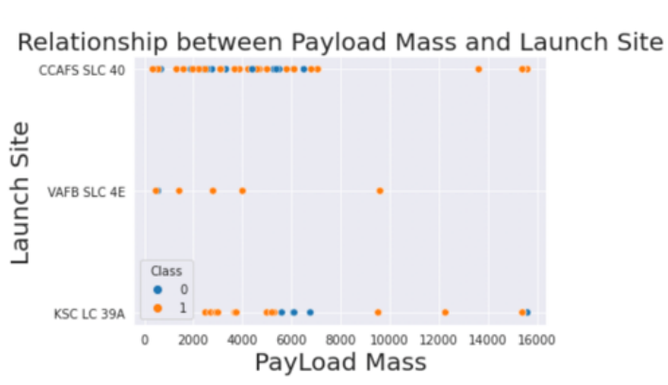
EDA- DATA VISUALIZATION

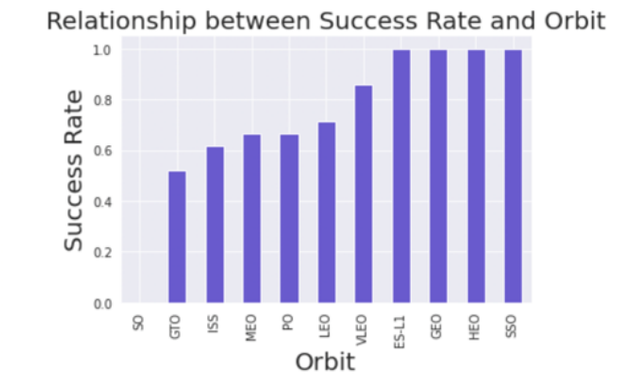
Code Details on GitHub

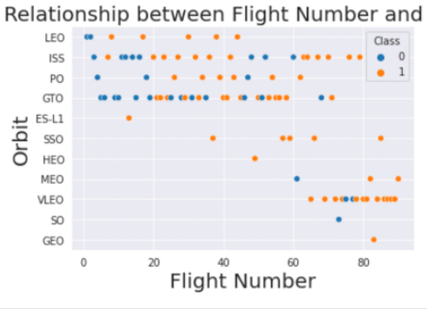
Below are the different Visualizations that were performed. Details can be found in Github repository (link above):

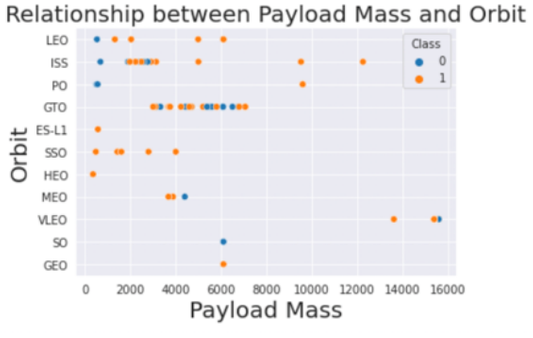
1. Scatter Plot to show relationship between Flight Number and Launch Site
2. Scatter Plot to show relationship between Payload Mass and Launch Site
3. Scatter Plot to show relationship between Flight Number and Orbit
4. Scatter Plot to show relationship between Payload Mass and Orbit
5. Bar Plot to show relationship between Success Rate and Orbit and sorted by Success Rate.
6. Line Plot to show success rate by year

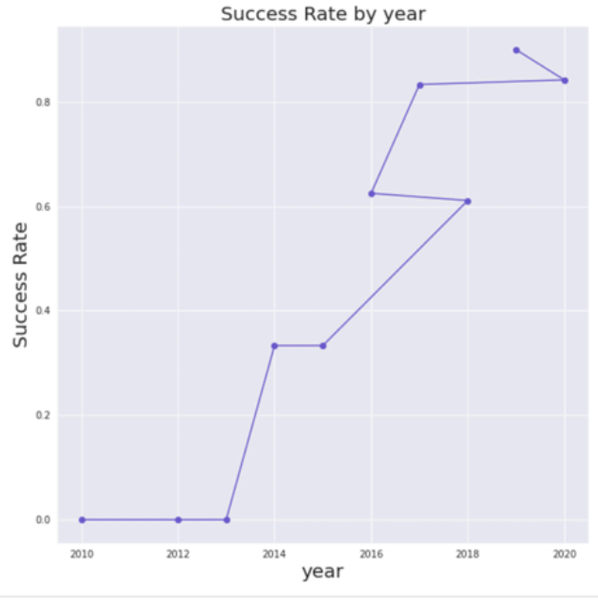








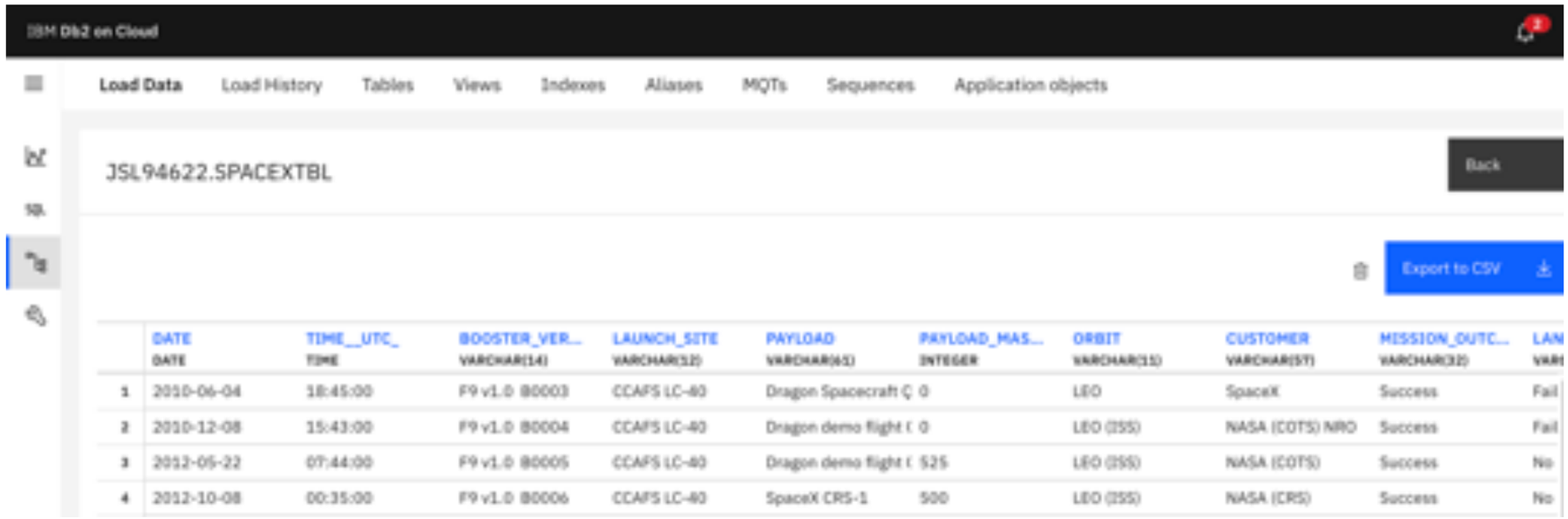




EDA- SQL

[Code Details on GitHub]

- Used IBM DB2 database and the load tool to upload SPACEX.csv into database table
- Below are results in table SPACEXTBL



	DATE	TIME_UTC_	BOOSTER_VER...	LAUNCH_SITE	PAYLOAD	PAYLOAD_MAS...	ORBIT	CUSTOMER	MISSION_OUTC...	LAN
	DATE	TIME	VARCHAR(14)	VARCHAR(32)	VARCHAR(61)	INTEGER	VARCHAR(15)	VARCHAR(57)	VARCHAR(32)	VAR
1	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Q	0	LEO	SpaceX	Success	Fail
2	2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C	0	LEO (ISS)	NASA (COTS) NRO	Success	Fail
3	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C	525	LEO (ISS)	NASA (COTS)	Success	No
4	2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No

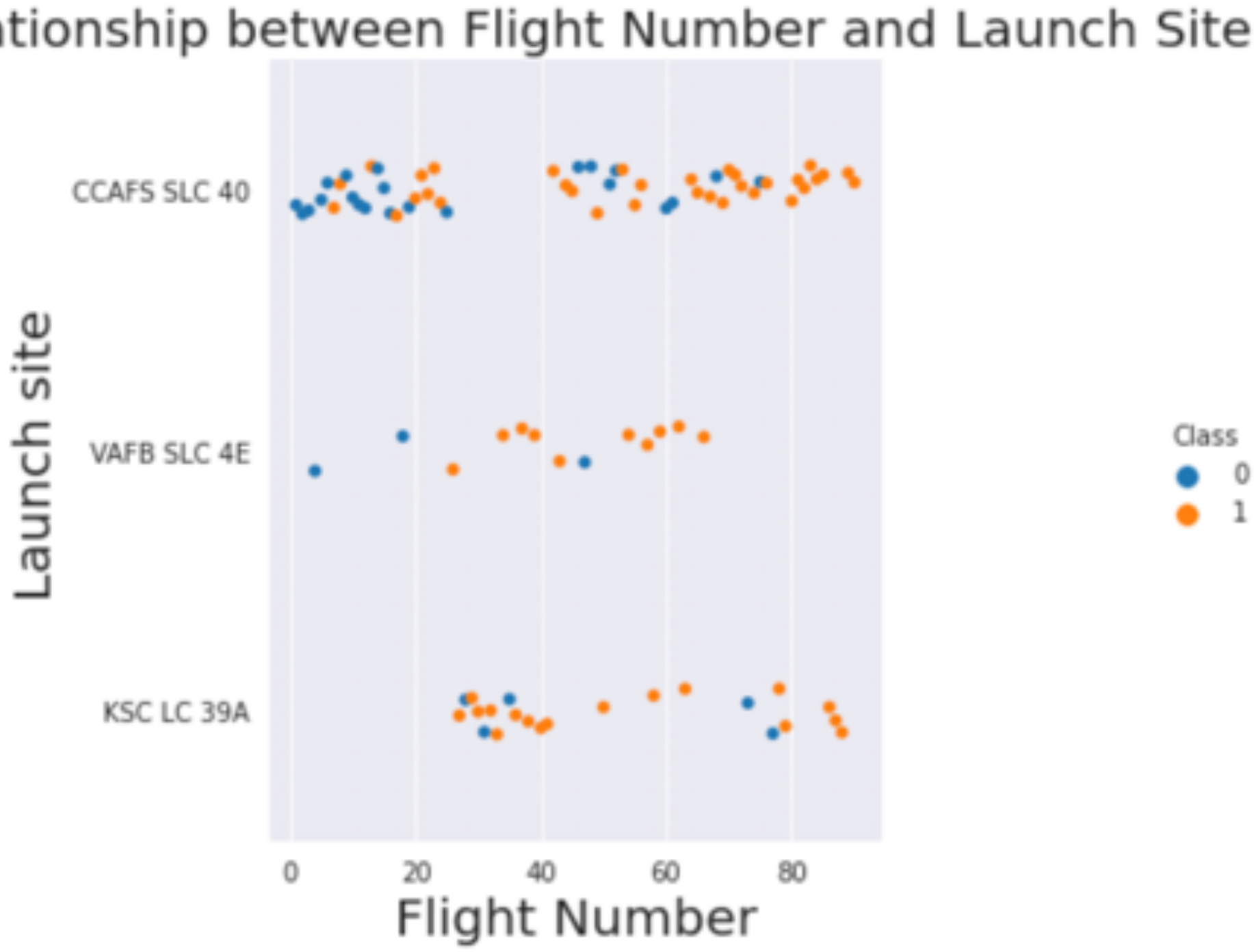
Below are the different queries that were performed. Details can be found in Github repository (link above):

1. Displayed the names of the unique launch sites in the space mission.
2. Displayed 5 records where launch sites begin with the string ‘CCA’.
3. Displayed the total payload mass carried by boosters launched by NASA (CRS).
4. Displayed the average payload mass carried by booster version F9 v1.1.
5. Calculated the date when the first successful landing outcome in ground pad was achieved.
6. Listed the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
7. Calculated the total number of successful and failure mission outcomes.
8. Listed the names of the booster_versions which have carried the maximum payload mass.
9. Listed the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015.
10. Listed, in descending order, the count of landing outcomes between 2010-06-04 and 2017-03-20.

RESULTS

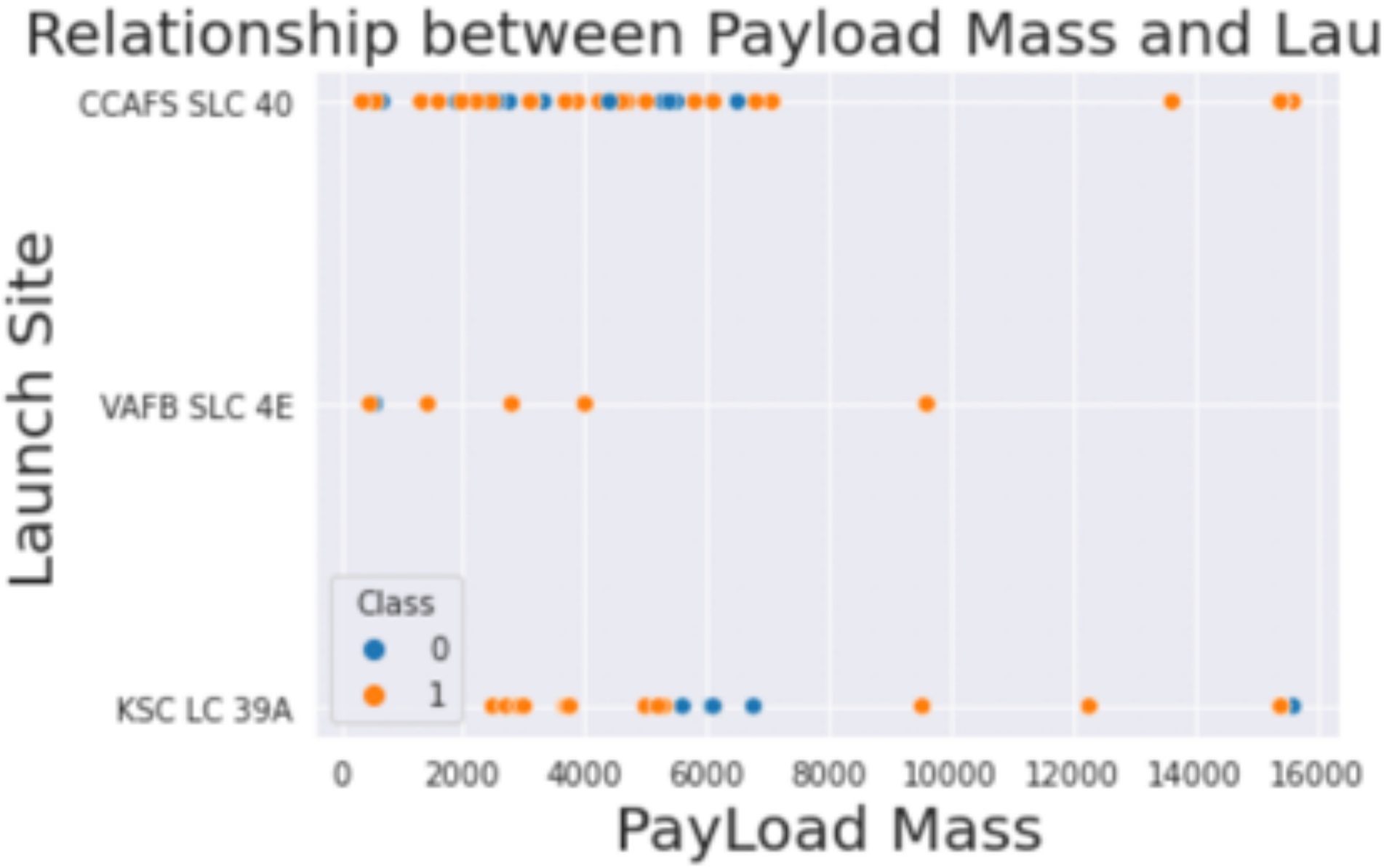
VISUALIZATION RESULTS– FLIGHT NUMBER AND LAUNCH SITE

[Code Details on GitHub]



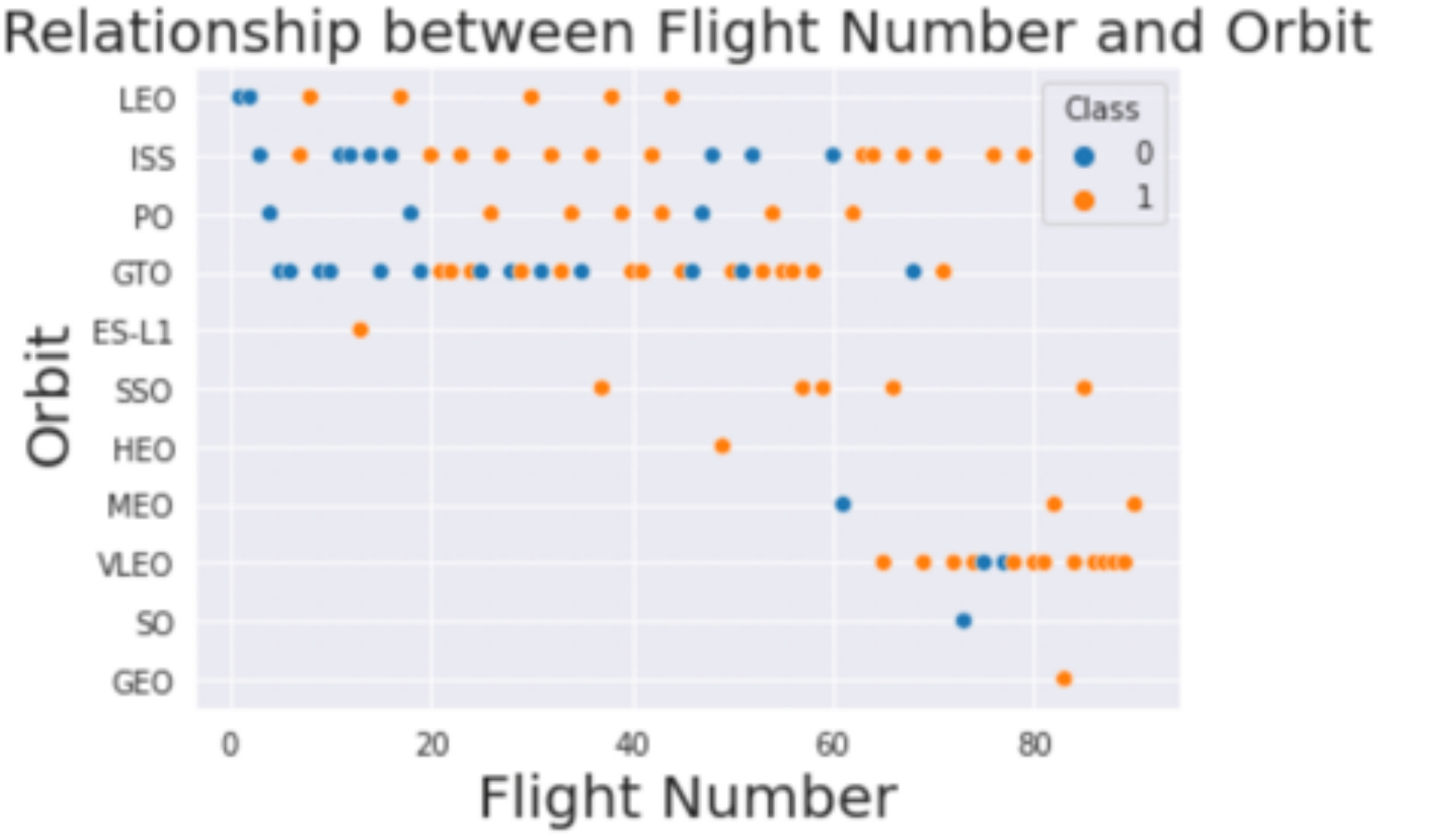
VISUALIZATION RESULTS- PAYLOAD MASS AND LAUNCH SITE

Code Details on GitHub



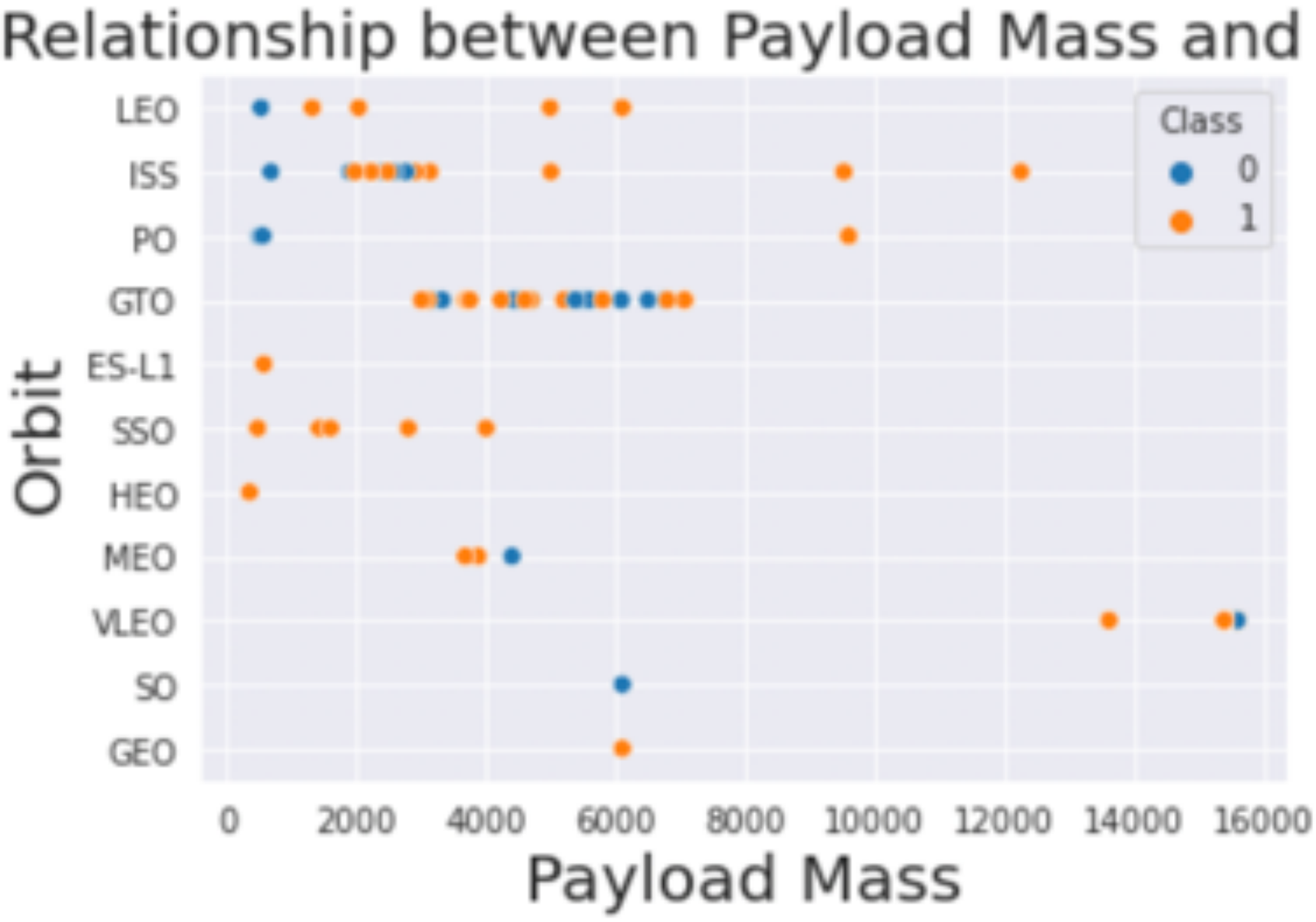
VISUALIZATION RESULTS– FLIGHT NUMBER AND ORBIT

[Code Details on GitHub]



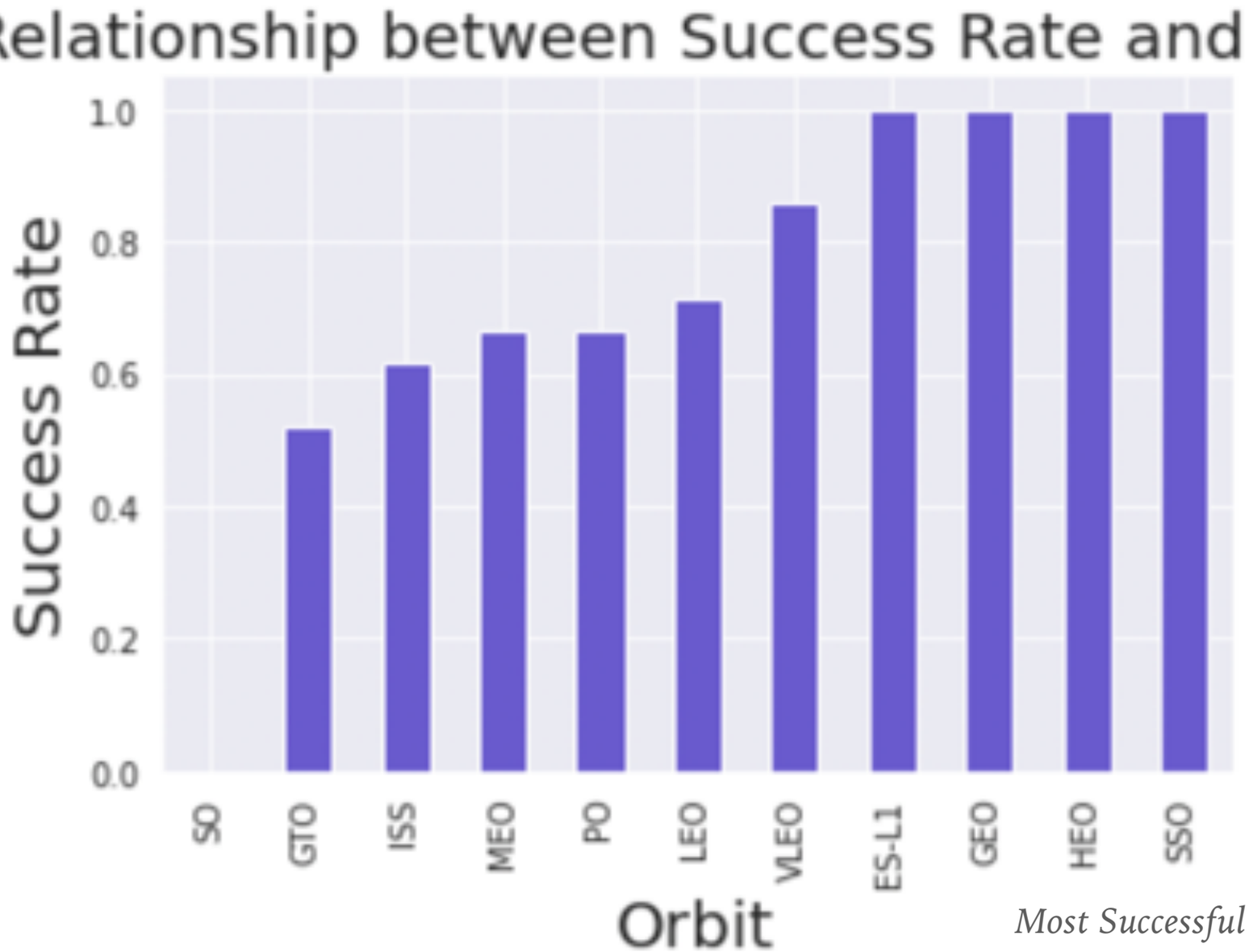
VISUALIZATION RESULTS- PAYLOAD MASS AND ORBIT

Code Details on GitHub



VISUALIZATION RESULTS– SUCCESS RATE AND ORBIT

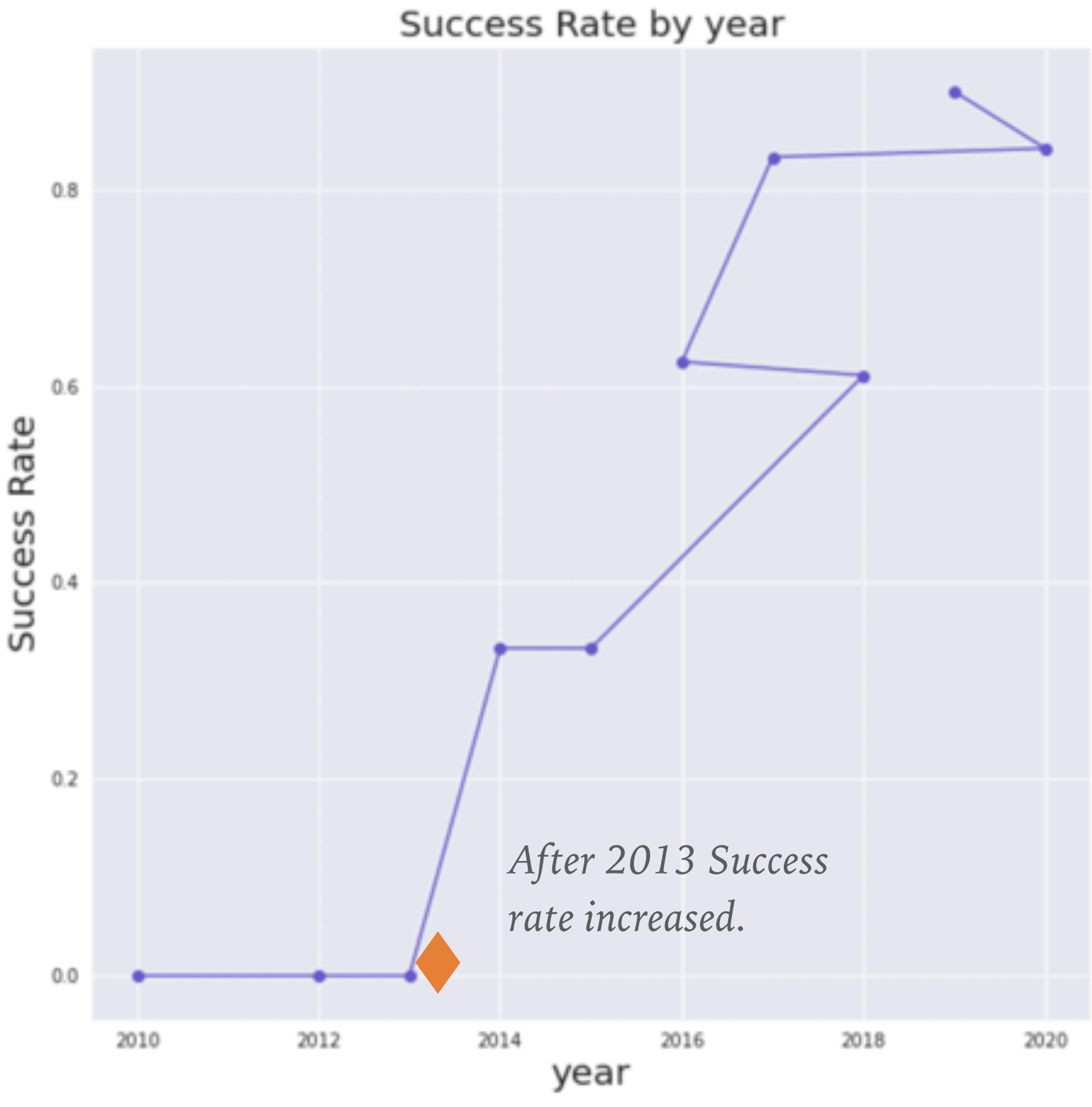
Code Details on GitHub



Most Successful Orbits

VISUALIZATION RESULTS– SUCCESS RATE BY YEAR

[Code Details on GitHub]



SQL RESULTS–LAUNCH SITES

Code Details on GitHub

In [30]:
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

```
 * ibm_db_sa://jsl94622:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/BLUDB
Done.
```

Out[30]:

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

First, looked up all unique launch sites.

Then, looked up launch sites starting with 'CCA'

In [55]:
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://jsl94622:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/BLUDB
Done.
```

Out[55]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

SQL RESULTS–MISSION OUTCOMES

[Code Details on GitHub]

Looks like there were 100 successful outcomes and 1 unsuccessful.

In [47]:
```
%sql SELECT count(mission_outcome) AS SUCCESSFUL_MISSION_OUTCOME FROM SPACEXTBL WHERE mission_outcome LIKE 'Success%'
```

```
 * ibm_db_sa://jal94622:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/BLUDB
Done.
```

Out[47]:

successful_mission_outcome
100

In [51]:
```
%sql SELECT count(mission_outcome) AS UNSUCCESSFUL_MISSION_OUTCOME FROM SPACEXTBL WHERE mission_outcome Like 'Failure%'
```

```
 * ibm_db_sa://jal94622:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/BLUDB
Done.
```

Out[51]:

unsuccessful_mission_outcome
1

SQL RESULTS-RANK OUTCOMES

Code Details on GitHub

Rank the landing outcomes.

In [89]:
```
%sql SELECT landing__outcome, count(landing__outcome) as total from SPACEXTBL \
where DATE BETWEEN '2010-06-04' and '2017-03-20' \
group by landing__outcome \
order by COUNT(landing__outcome) DESC
```

```
 * ibm_db_sa://js194622:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/BLUDB
Done.
```

Out[89]:

landing__outcome	total
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

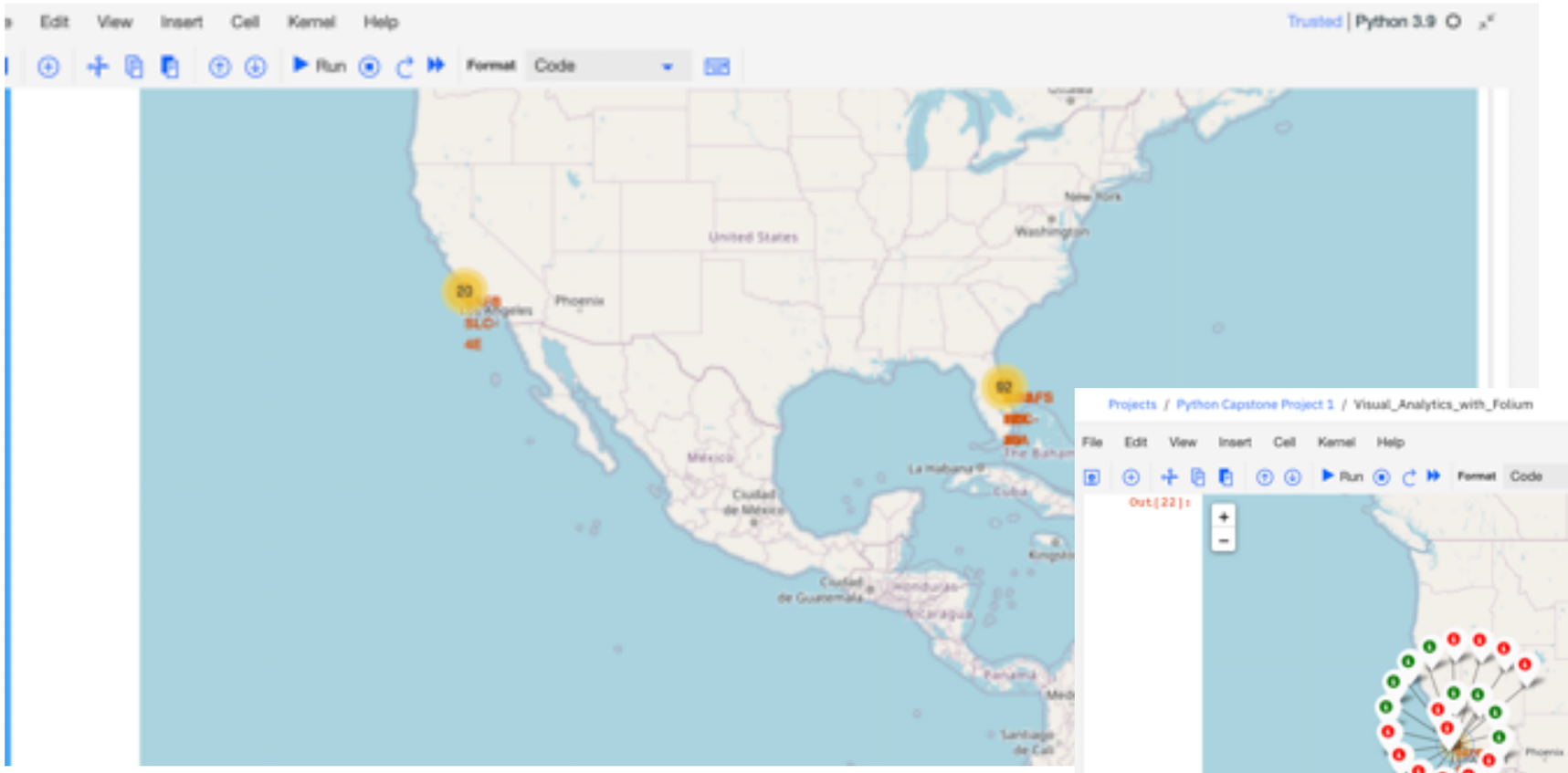
LAUNCH SITES ANALYSIS

INTERACTIVE MAP WITH FOLIUM

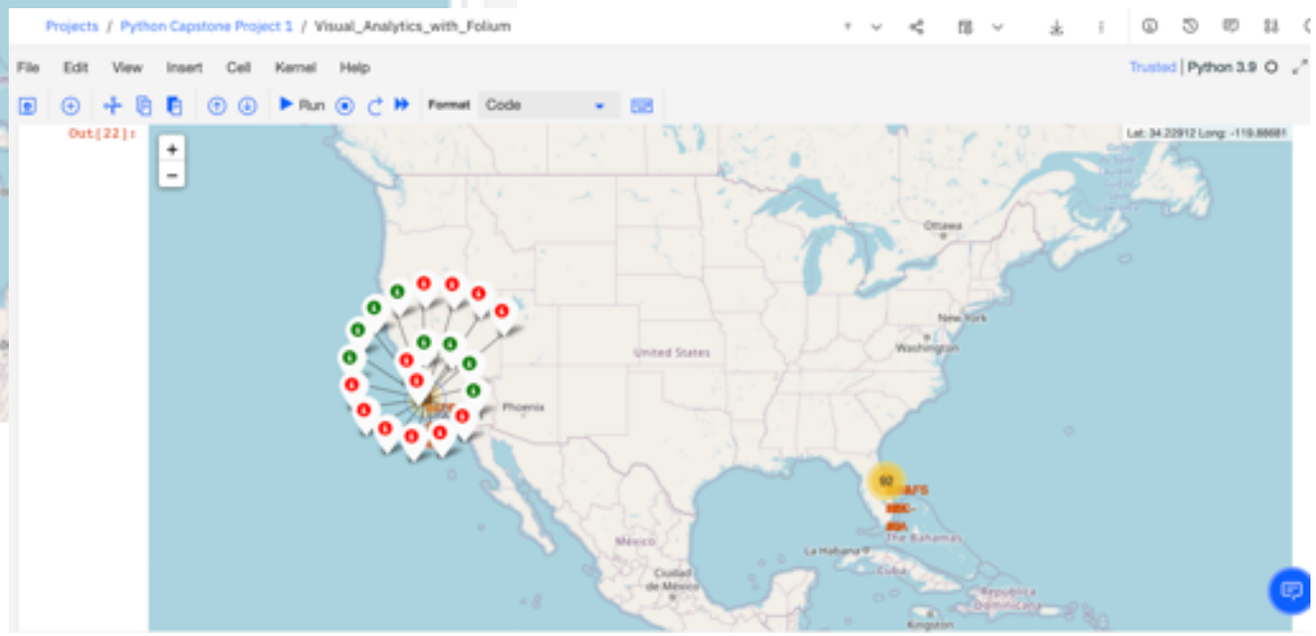
[Code Details on GitHub]

Used Folium to create interactive maps and explore the following:

- Created Map of all launch sites
- Created Map showing successful and unsuccessful Launches. Successful launch were marked in green and unsuccessful launches are parked in red. When zoomed out, can see overview of different launch sites and when zoomed in, can see different launches at each launch site
- Calculated distances between launch site and coastline, railroad, highway and marked line showing distance.



	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610746

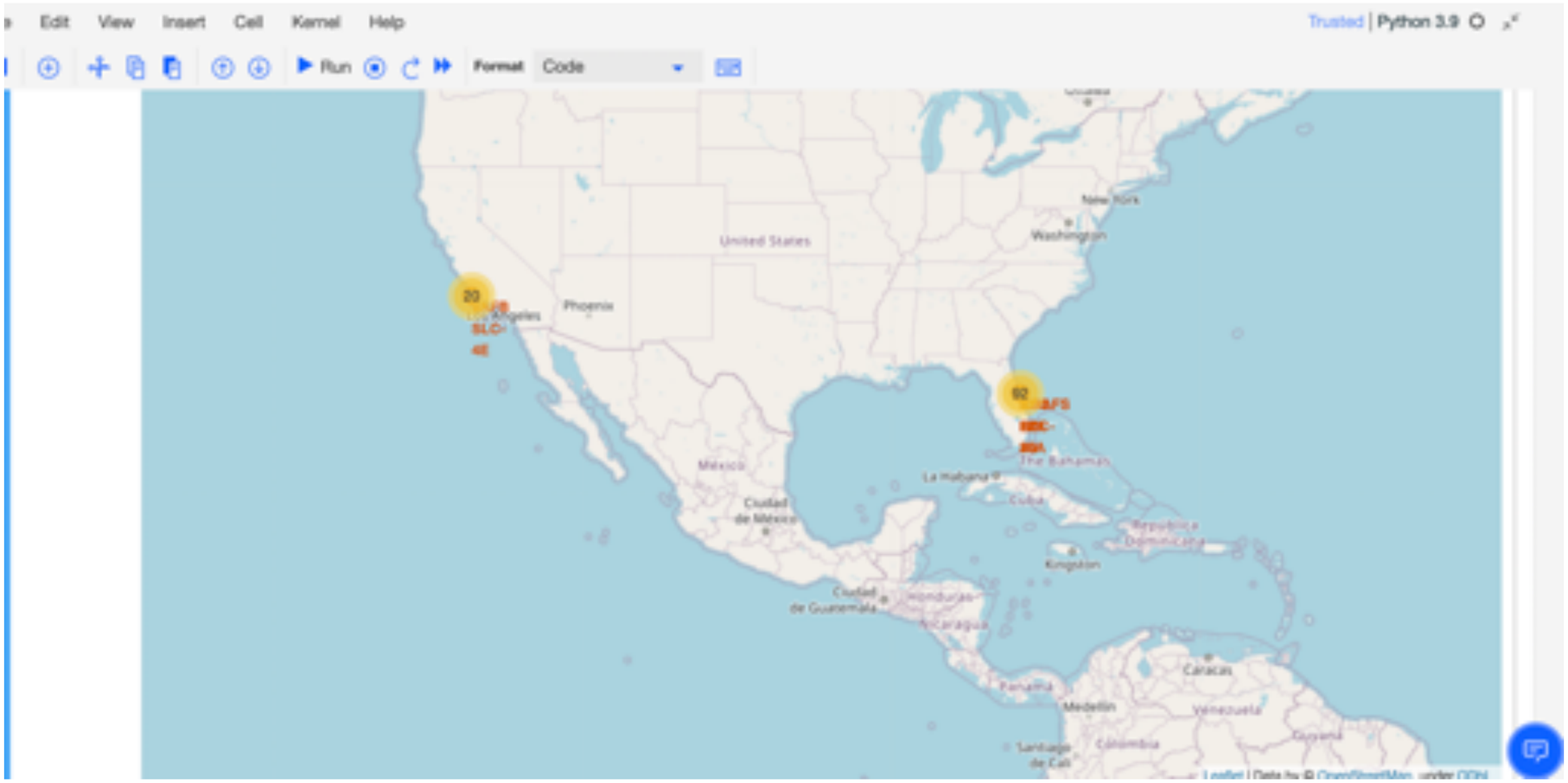


INTERACTIVE MAP ANALYSIS WITH FOLIUM

- All launch sites with coordinates:

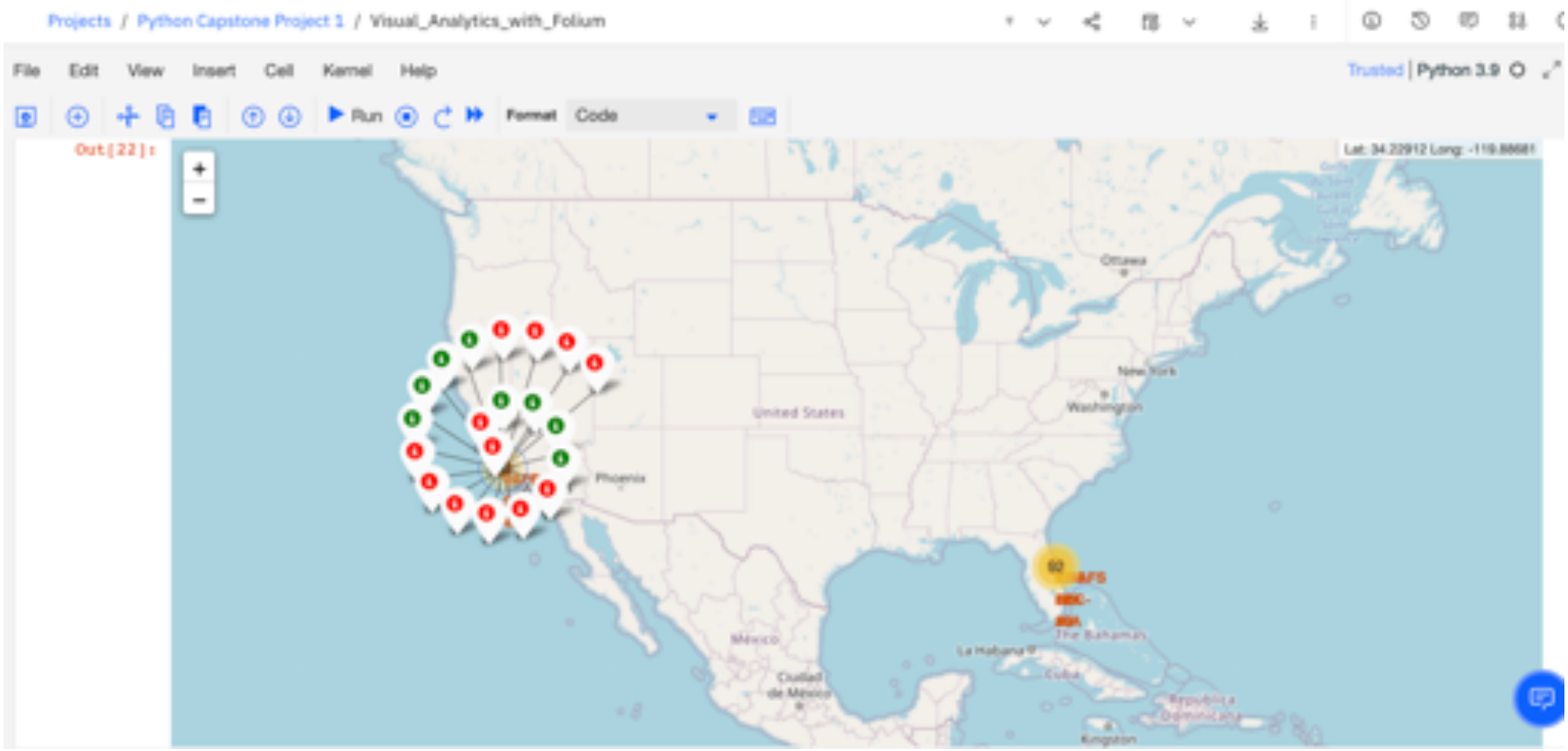
	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610746

- Map of Launch sites (1 on west coast and 3 on east coast):



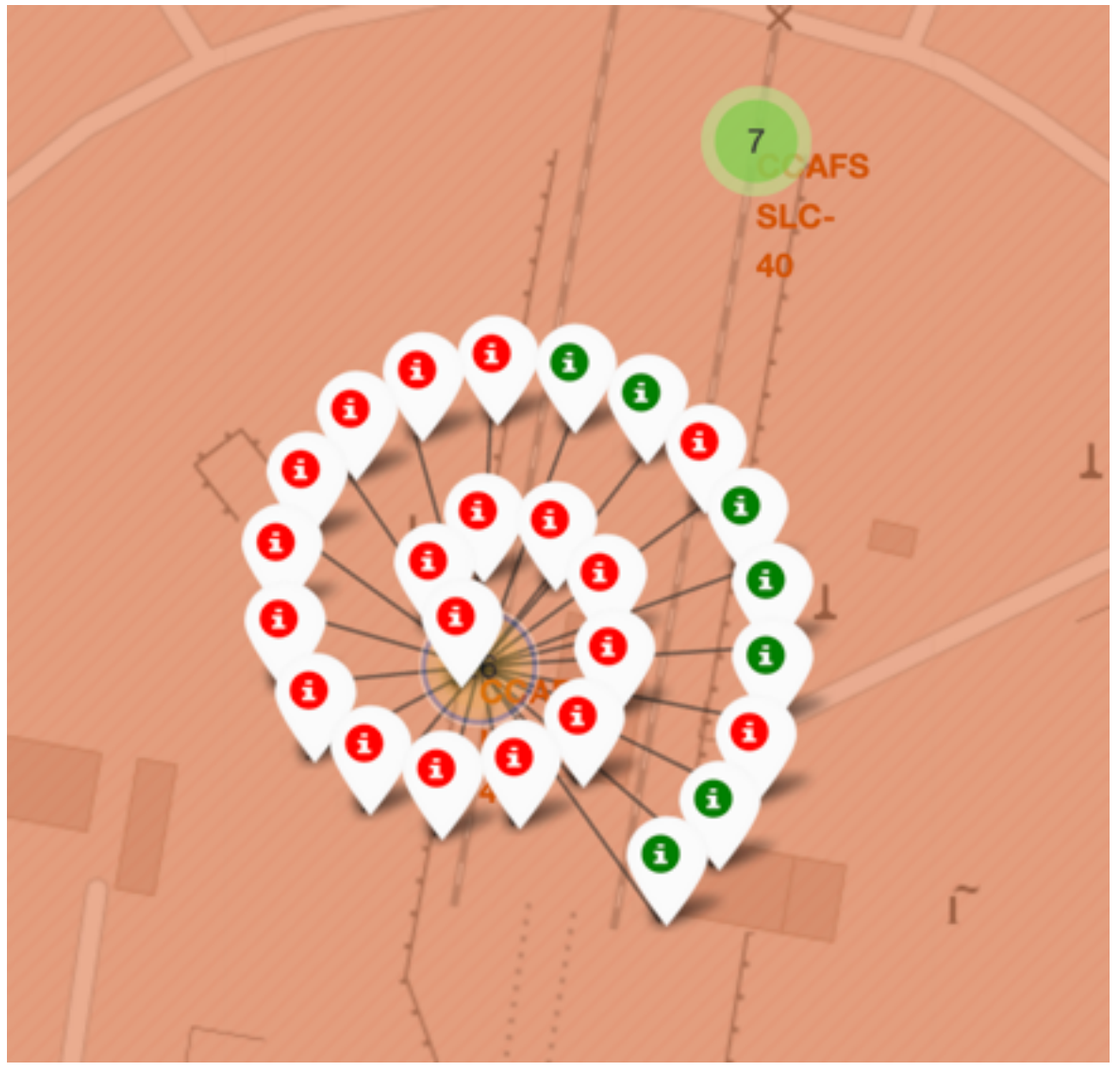
INTERACTIVE MAP ANALYSIS WITH FOLIUM

- **Launches at launch site 'VAFB SLC-4E':** Green markers are successful launches and red markers are unsuccessful launches. Looks like there were 12 unsuccessful launches and 8 successful launches.



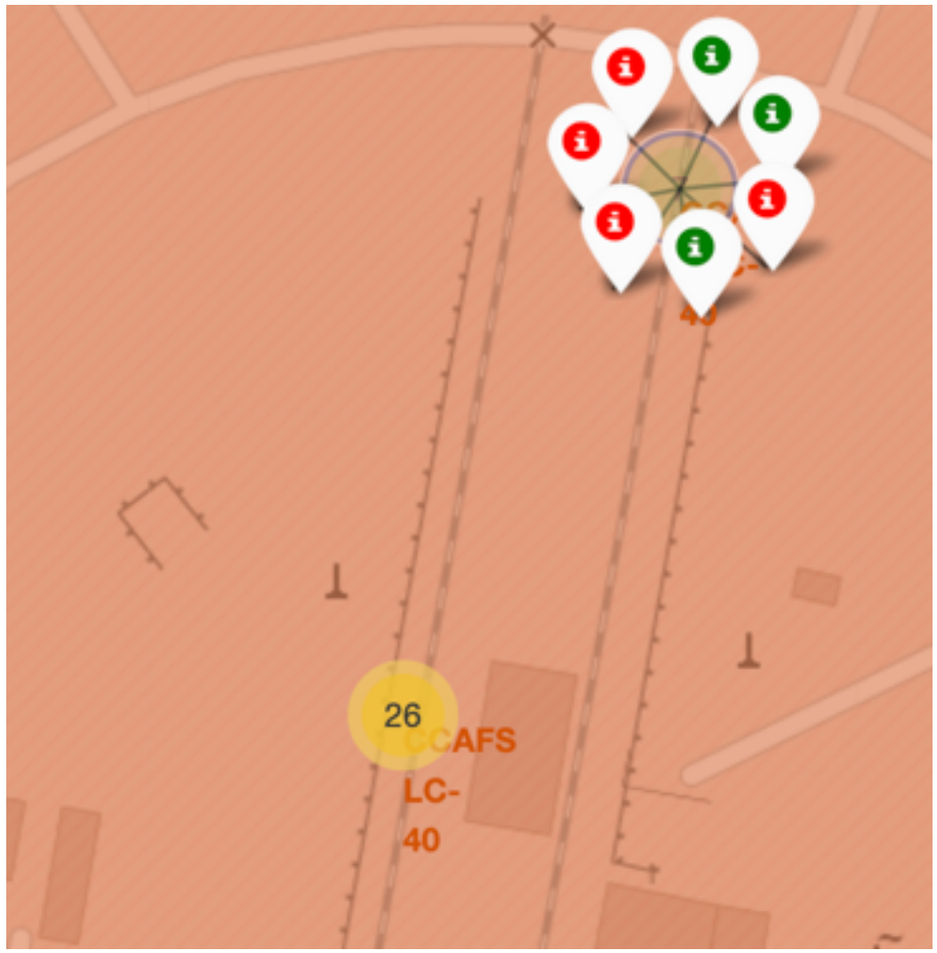
INTERACTIVE MAP ANALYSIS WITH FOLIUM

- **Launches at launch site ‘CCAFS LC-40’:** Green markers are successful launches and red markers are unsuccessful launches. Looks like there were 19 unsuccessful launches and 7 successful launches.



INTERACTIVE MAP ANALYSIS WITH FOLIUM

- **Launches at launch site 'CCAFS SLC-40':** Green markers are successful launches and red markers are unsuccessful launches. Looks like there were 4 unsuccessful launches and 3 successful launches. This launch site has the least number of launches.

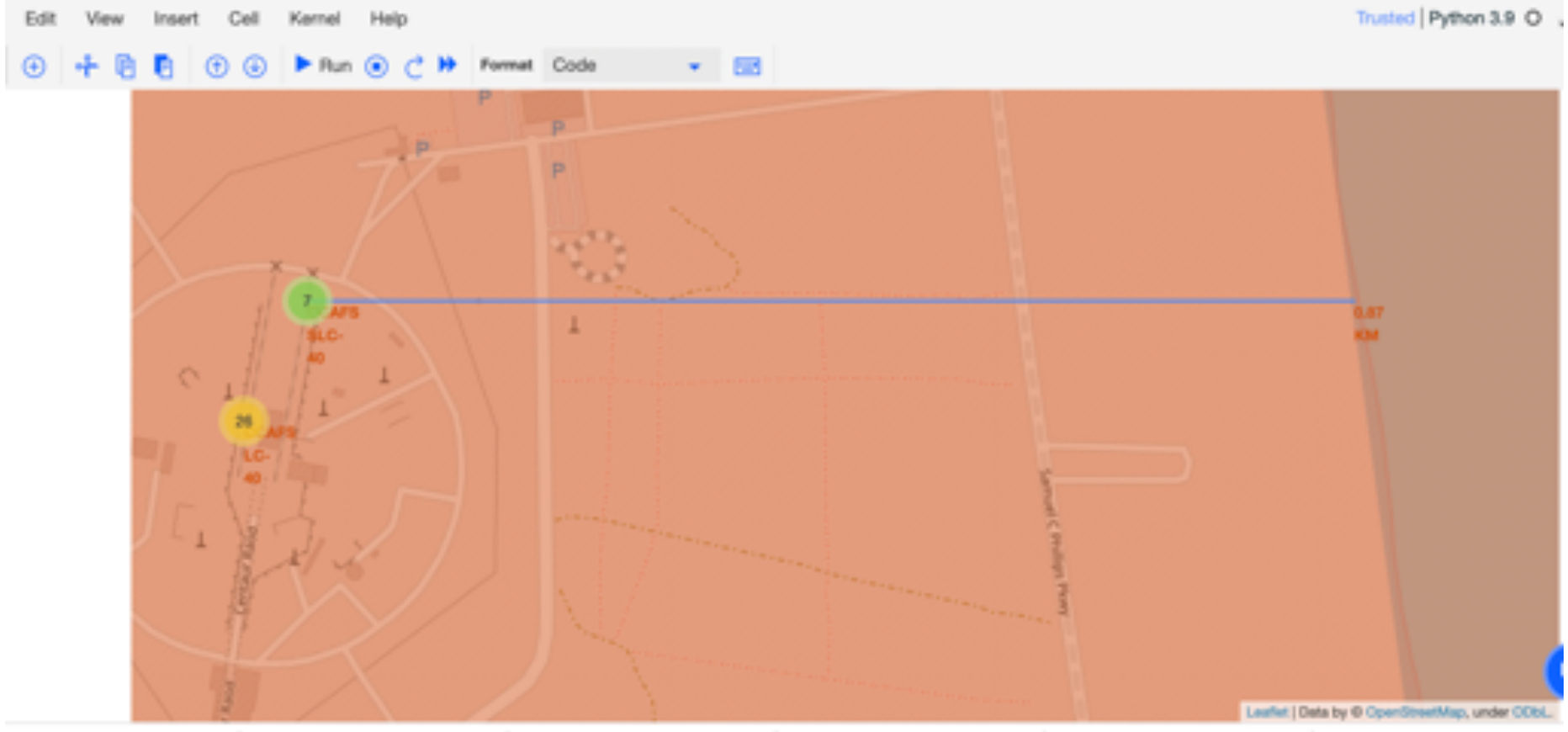


INTERACTIVE MAP ANALYSIS WITH FOLIUM

- **Launches at launch site 'KSC LC-39A':** Green markers are successful launches and red markers are unsuccessful launches. Looks like there were 3 unsuccessful launches and 10 successful launches. Looks like this launch site has the highest rate of successful launches.

INTERACTIVE MAP ANALYSIS WITH FOLIUM

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610746



DASHBOARD WITH PLOTLY

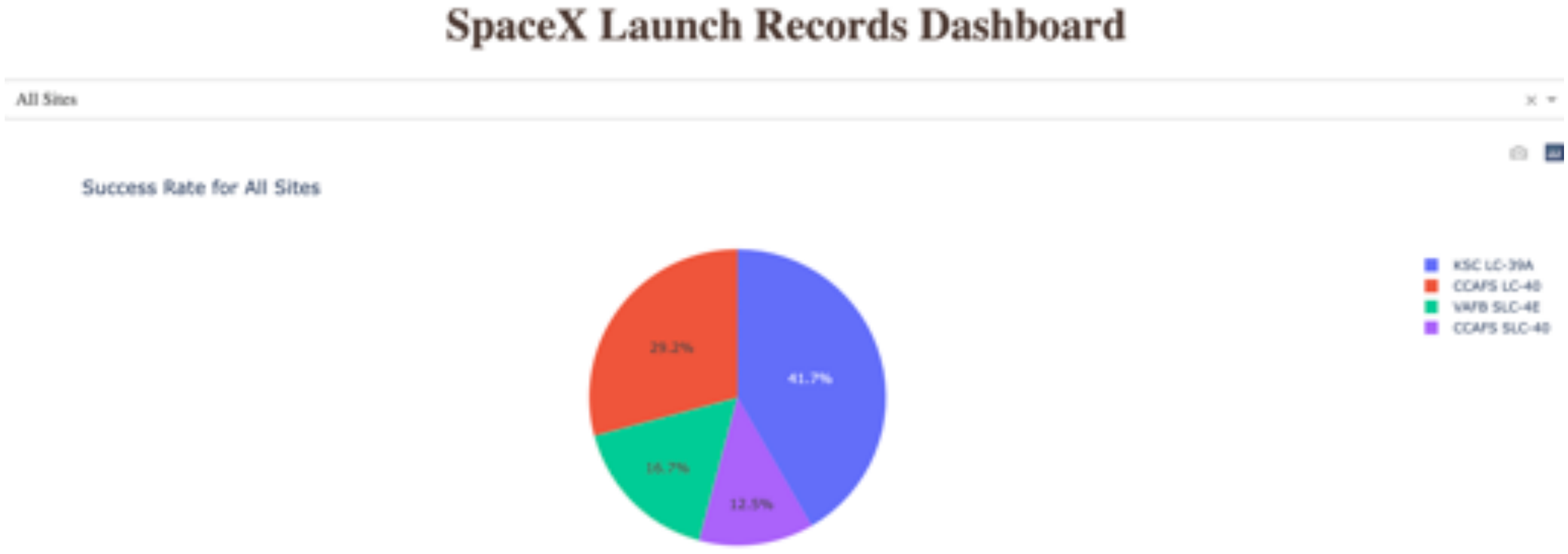
PLOTLY DASH

Code Details on GitHub

- Plotly is a Python library that is used to enable users to perform visualize and analyze data interactively.
- Building a dashboard using Plotly was useful to analyze SpaceX launch data in real-time.
- Created The SpaceX dashboard application with a dropdown list(see below) containing all the launch sites to interact with a pie chart based on each launch site info (next slides).

PLOTLY DASH-PIE CHART FOR ALL SITES

[Code Details on GitHub]

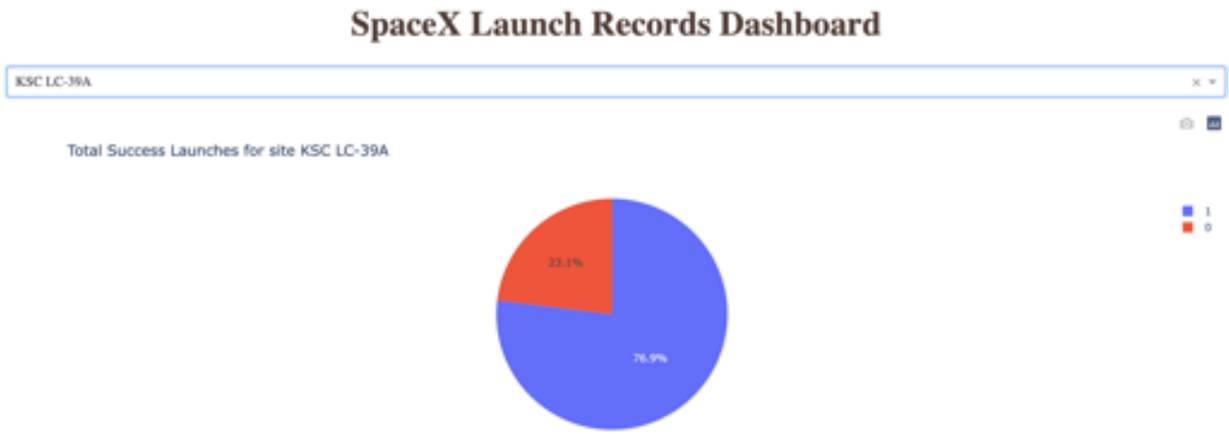


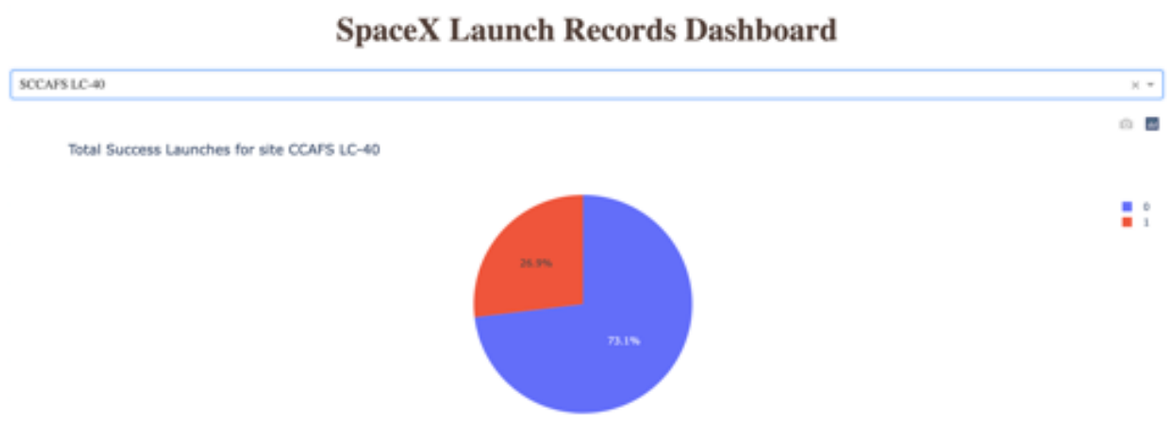
Launch site:´KSC LC -39A´ has the most successful launches.

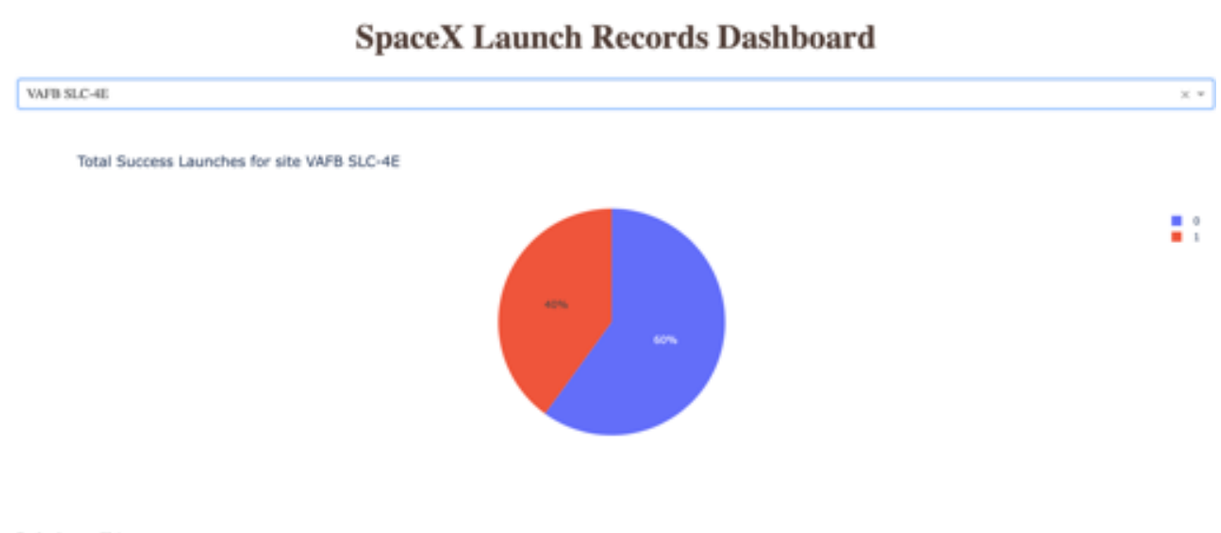
The result in pie chart is consistent with other results in analysis.

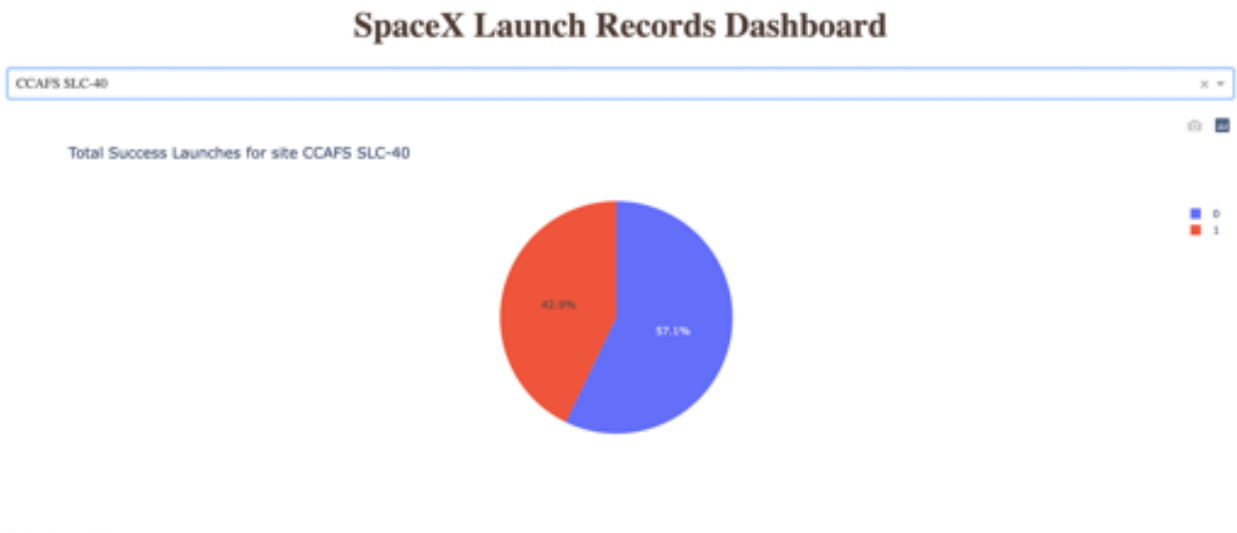
PLOTLY DASH-PIE CHART RESULTS BASED ON SELECTION

[Code Details on GitHub]



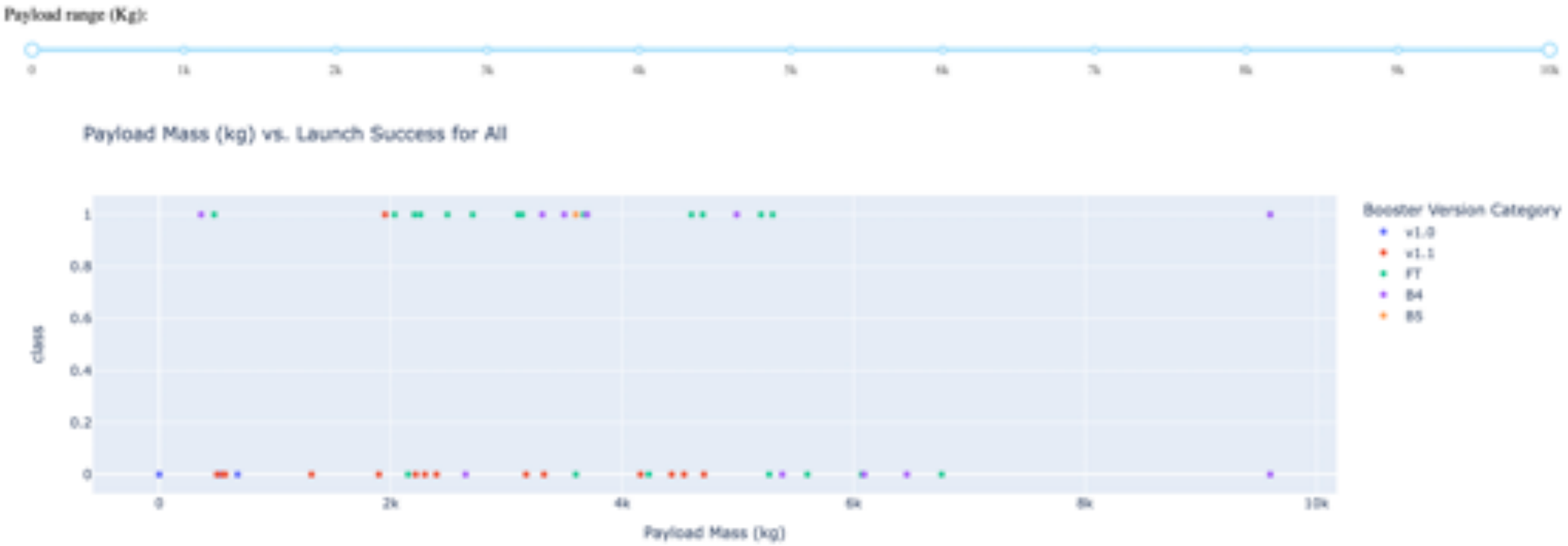






PLOTLY DASH-SCATTER PLOT RESULTS FOR ALL SITES

Code Details on GitHub

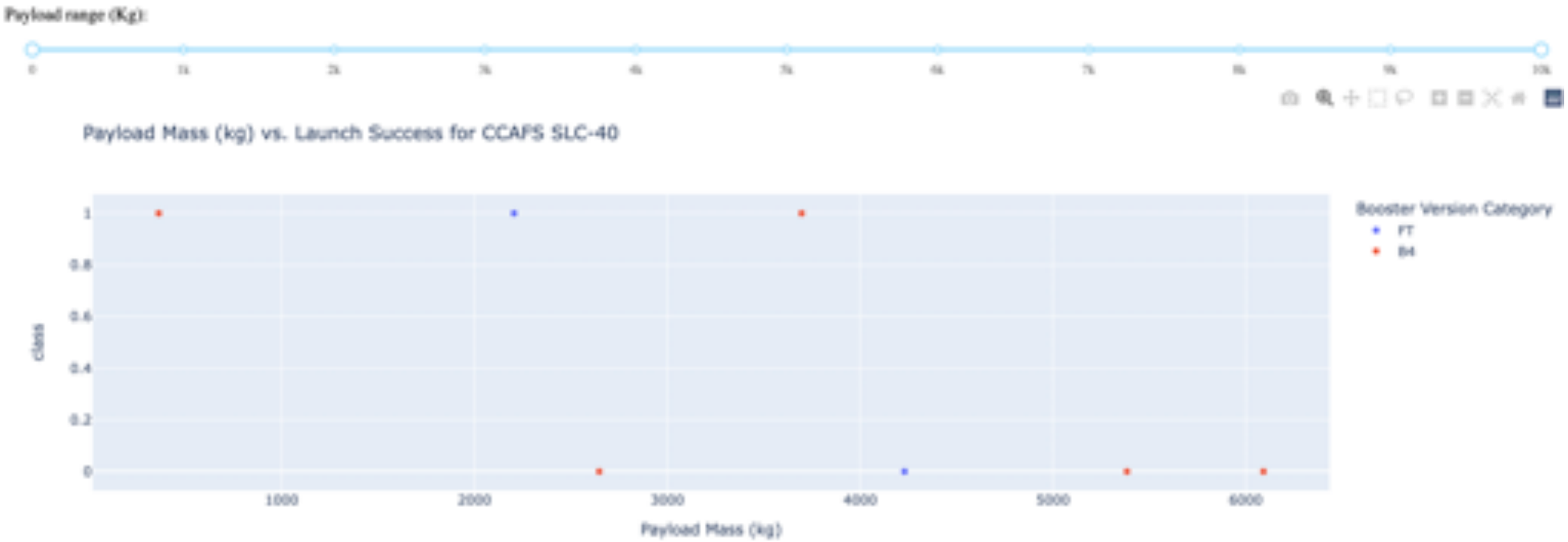


PLOTLY DASH-SCATTER PLOT RESULTS BASED ON SELECTION

Code Details on GitHub

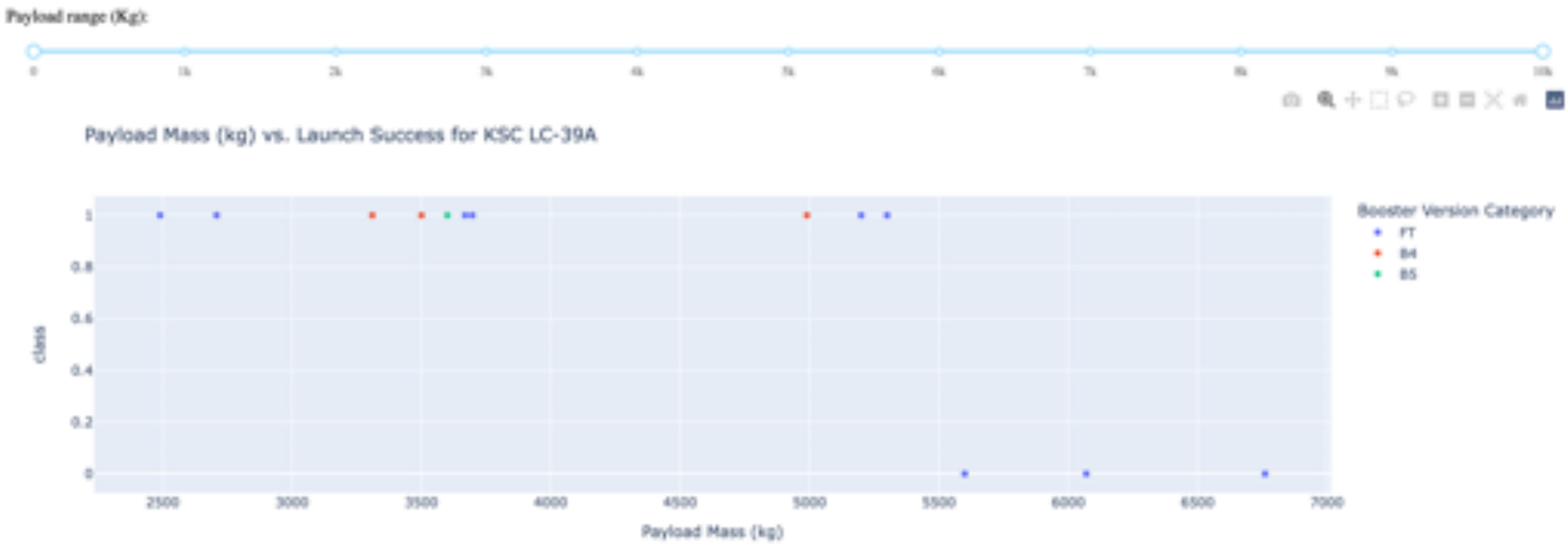
PLOTLY DASH-SCATTER PLOT RESULTS BASED ON SELECTION

[Code Details on GitHub]



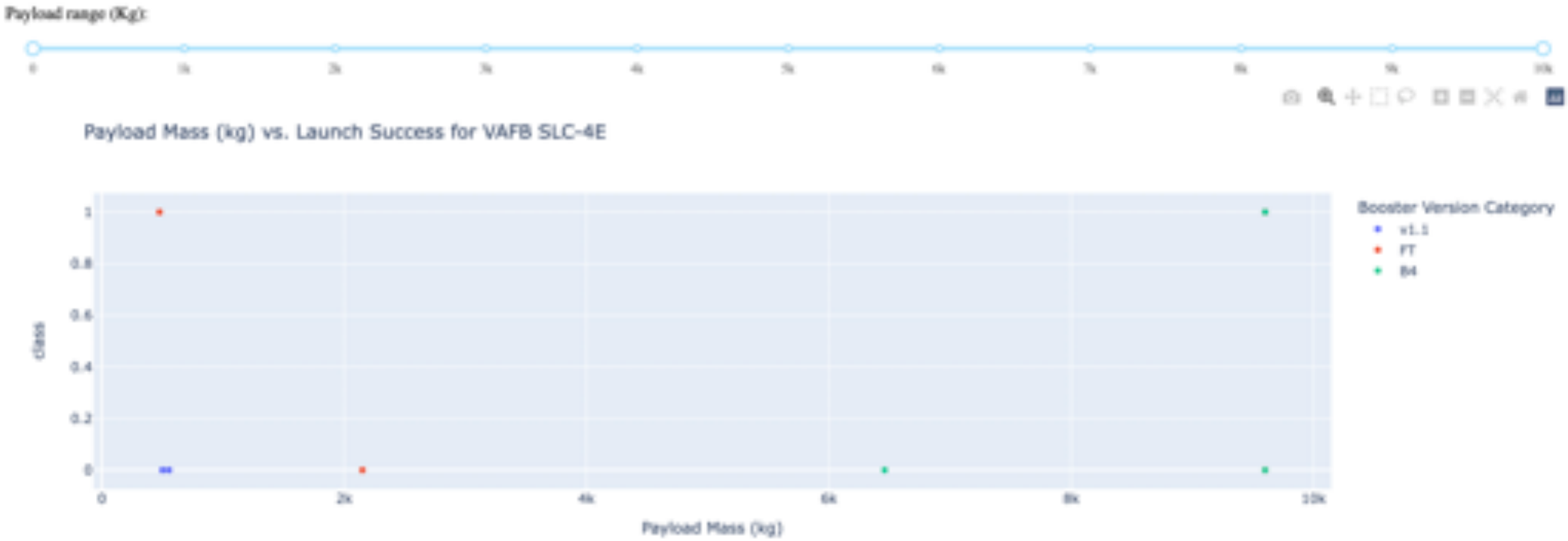
PLOTLY DASH-SCATTER PLOT RESULTS BASED ON SELECTION

[Code Details on GitHub]



PLOTLY DASH-SCATTER PLOT RESULTS BASED ON SELECTION

[Code Details on GitHub]



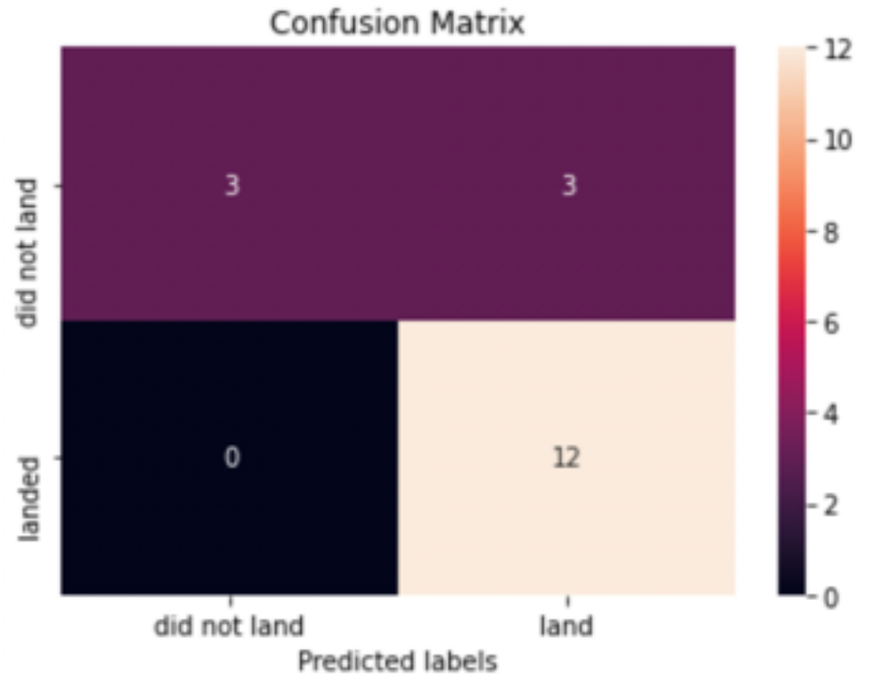
PREDICTIVE ANALYSIS

PREDICTIVE ANALYSIS USING MACHINE LEARNING

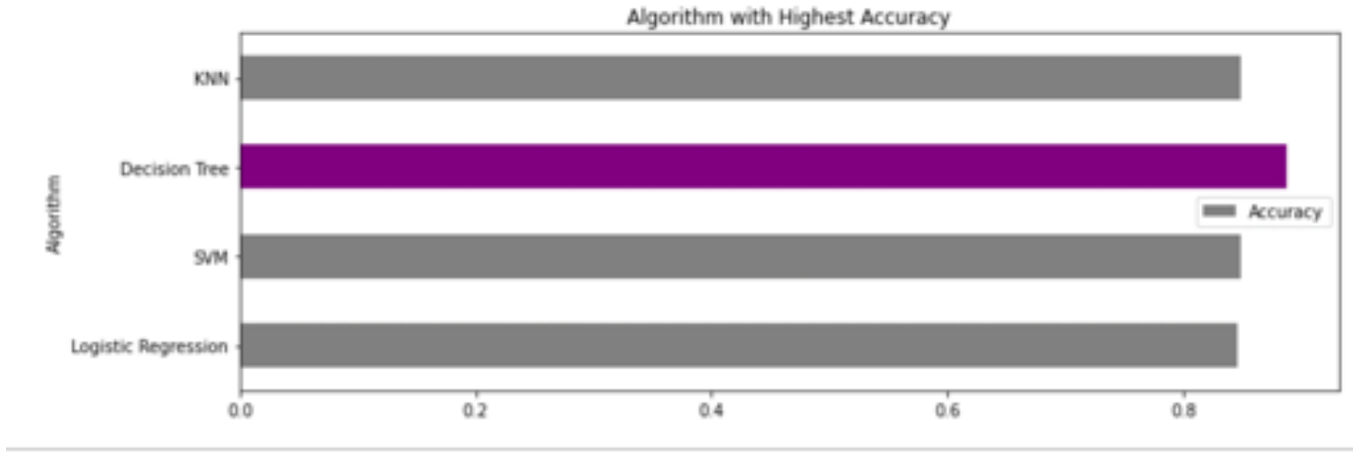
- Perform Exploratory Data Analysis and determine Training Labels
- Find best Hyperparameter for SVM, Classification Trees and Logistic Regression

Results:

- Decision Tree Algorithm has the highest accuracy (but not by a lot).
- Confusion Matrix shows that there are a lot of False positives (same confusion matrix with every algorithm). This is a major problem.



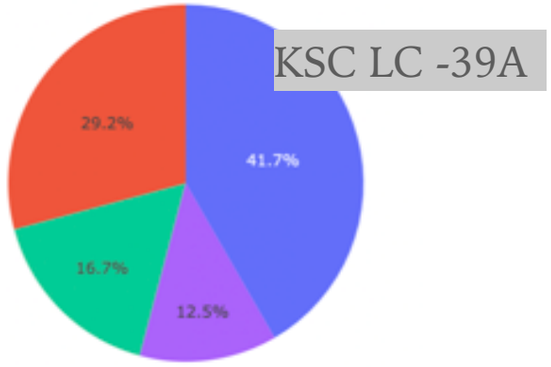
	Algorithm	Accuracy
0	Logistic Regression	0.846429
1	SVM	0.848214
2	Decision Tree	0.887500
3	KNN	0.848214

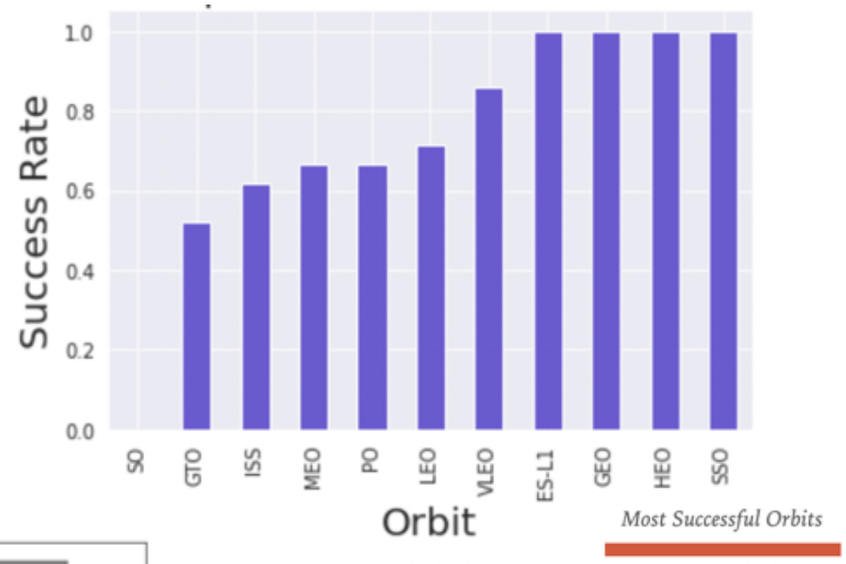


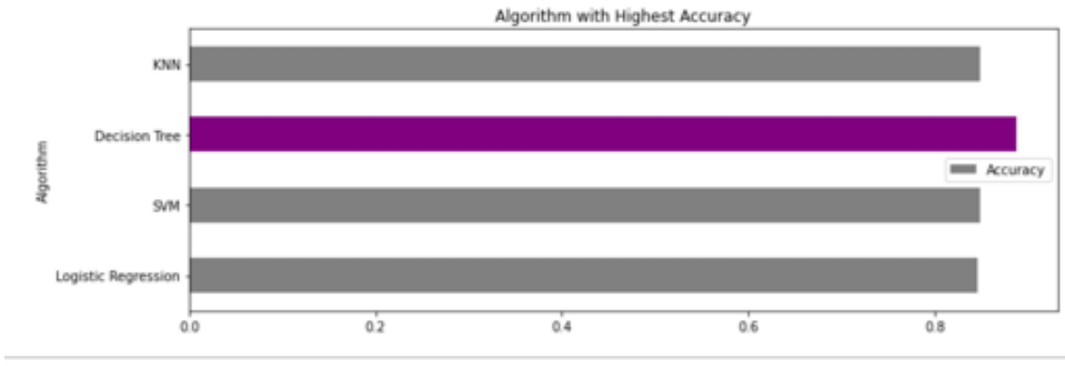
CONCLUSION

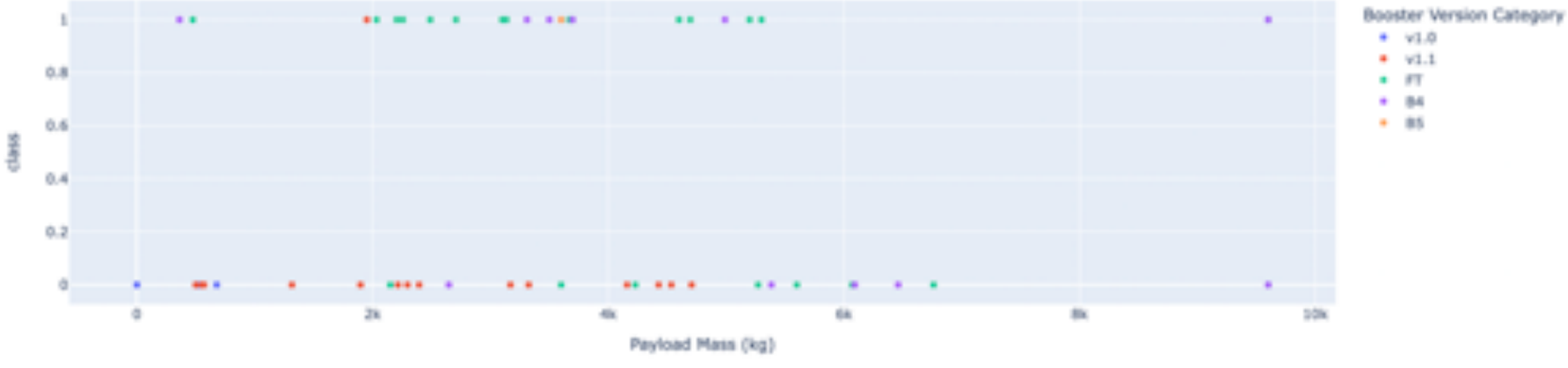
CONCLUSION

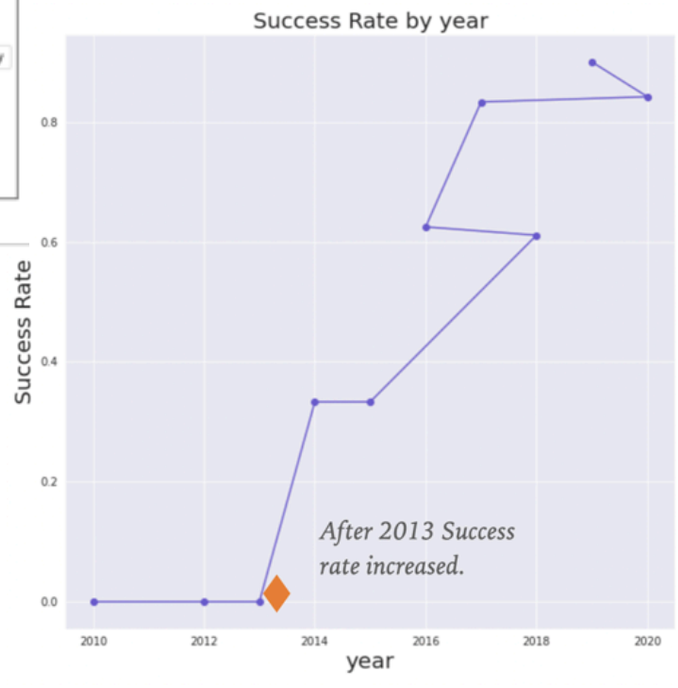
- Launch site **'KSC LC-39A'** has the most successful launches.
- There has been an increase in successful launches since 2013.
- Most Successful Orbits are: ES-L1, GEO, HEO, SSO.
- Booster Version 'FT' has most successful launches and most successes are between 2000-6000kg Payload Mass.
- Decision Tree Algorithm has highest Accuracy to predict future launch success.











THANK YOU!