IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Claudia Rodriguez
08-15-2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection with web scrapting and API
  - Data Wrangling
  - Data Analysis with SQL and visualization
  - Interactive visual analytics with Folium
  - Predictive Analysis
- Summary of all results
  - Visualization Data
  - Best Predictive Analysis

Introduction

- Project background and context

The project is about the success of the launch of Falcon 9 rocket. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards 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. Spaces X's Falcon 9 launch like regular rockets.

- Problems you want to find answers
  - With, what factors the rocket will land successfully?
  - The relationship between the variables.
  - Which is the best model for the success launch?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Rest API
  - Webscrapping from wikipedia
- Perform data wrangling
  - Dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
  - With bar chart and scatter graphs to know the relation between variables.
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Build and evaluate classification models

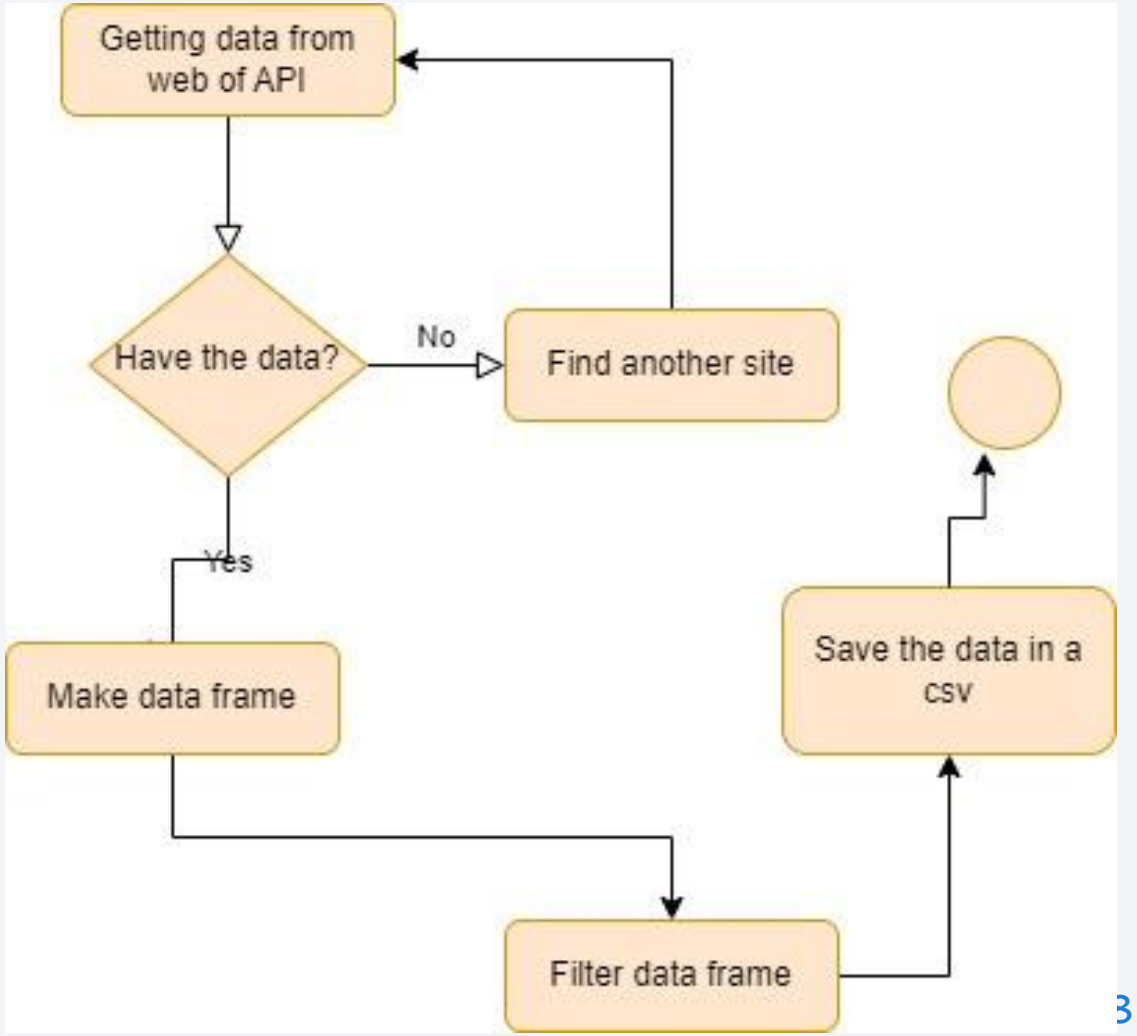
Data Collection

- The data collection is the process of getting the data, transform the data and clean the data. When we do that we can process the data into a model and obtain the prediction we want.
- The steps are:
  - Get the data from an API or a web page
  - Make a data frame
  - Filter the dataframe
  - Export to a CSV.

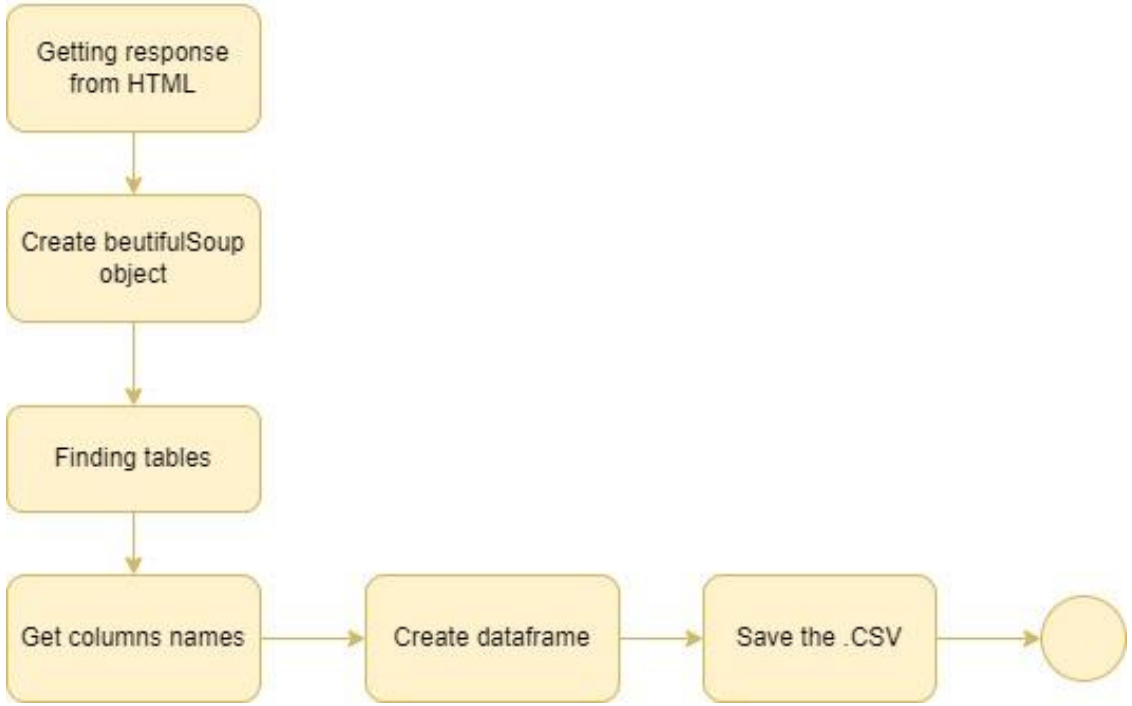
Data Collection – SpaceX API

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

```
FlightNumber        0
Date                0
BoosterVersion      0
PayloadMass         0
Orbit               0
LaunchSite          0
Outcome             0
Flights             0
GridFins            0
Reused              0
Legs                0
LandingPad         26
Block               0
ReusedCount         0
Serial              0
Longitude           0
Latitude            0
dtype: int64
```



https://github.com/claumary/spaceXProject/blob/main/Collecting%20the%20Data.ipynb




```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Version,<br/>Boost
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-12">[c]</a></sup>
</th>
<th scope="col">Payload mass
```

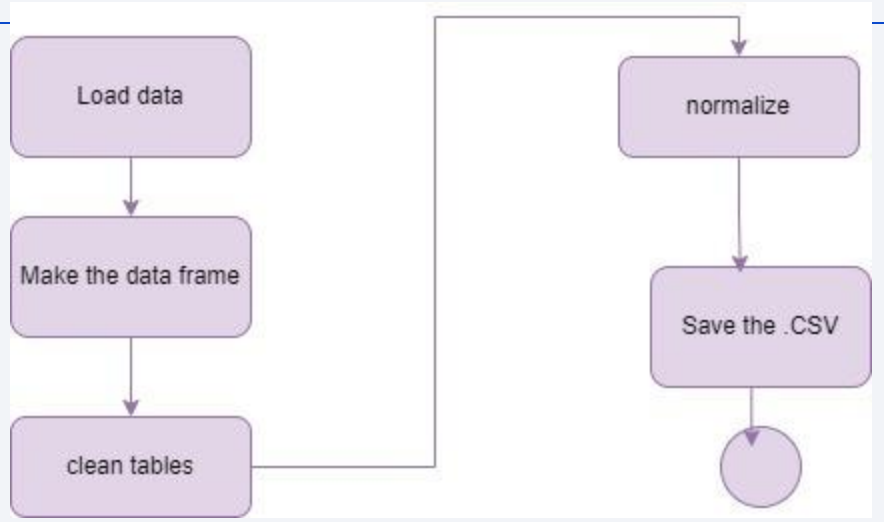
Data Collection - Scraping

https://github.com/claumary/spaceXProject/blob/main/Webscraping.ipynb

Data Wrangling

```
# Apply value_counts() on column LaunchSite
df.LaunchSite.value_counts()
```

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

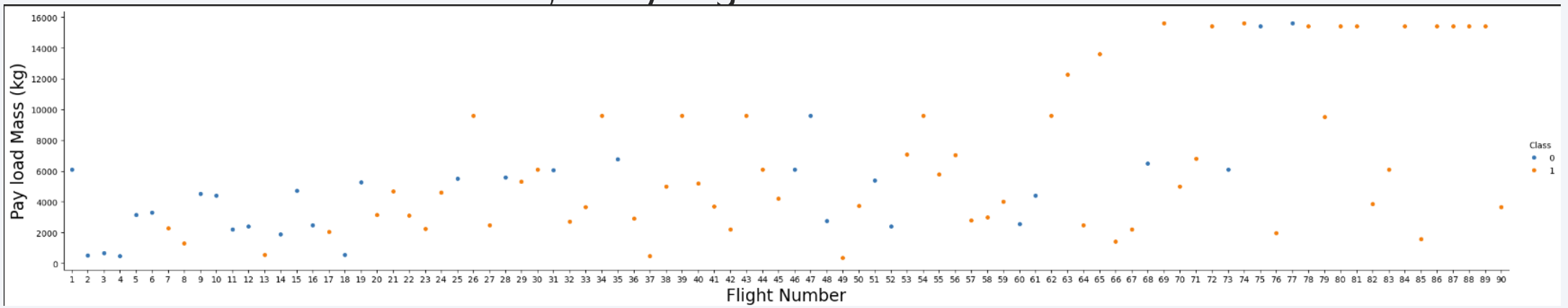


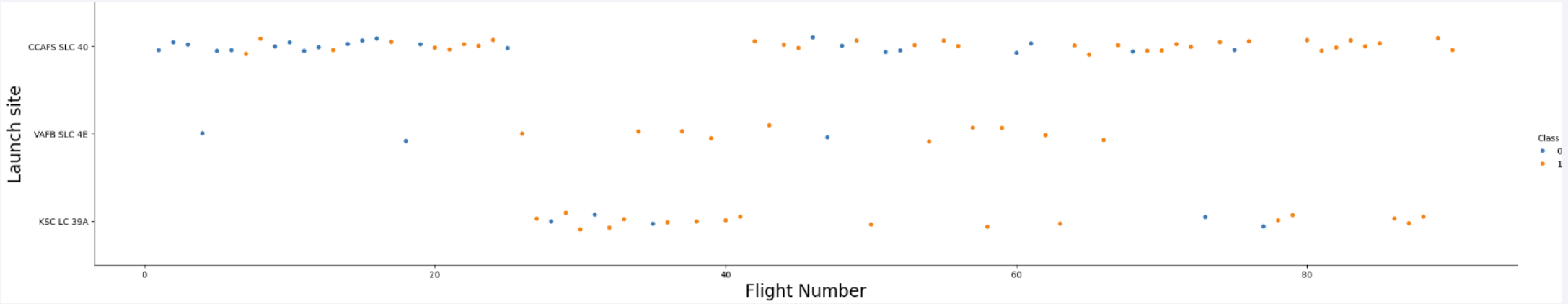
	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	C
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

https://github.com/claumary/spaceXProject/blob/main/Data%20wrangling.ipyn

EDA with Data Visualization

- Makes different data charts, analyzing with different variables

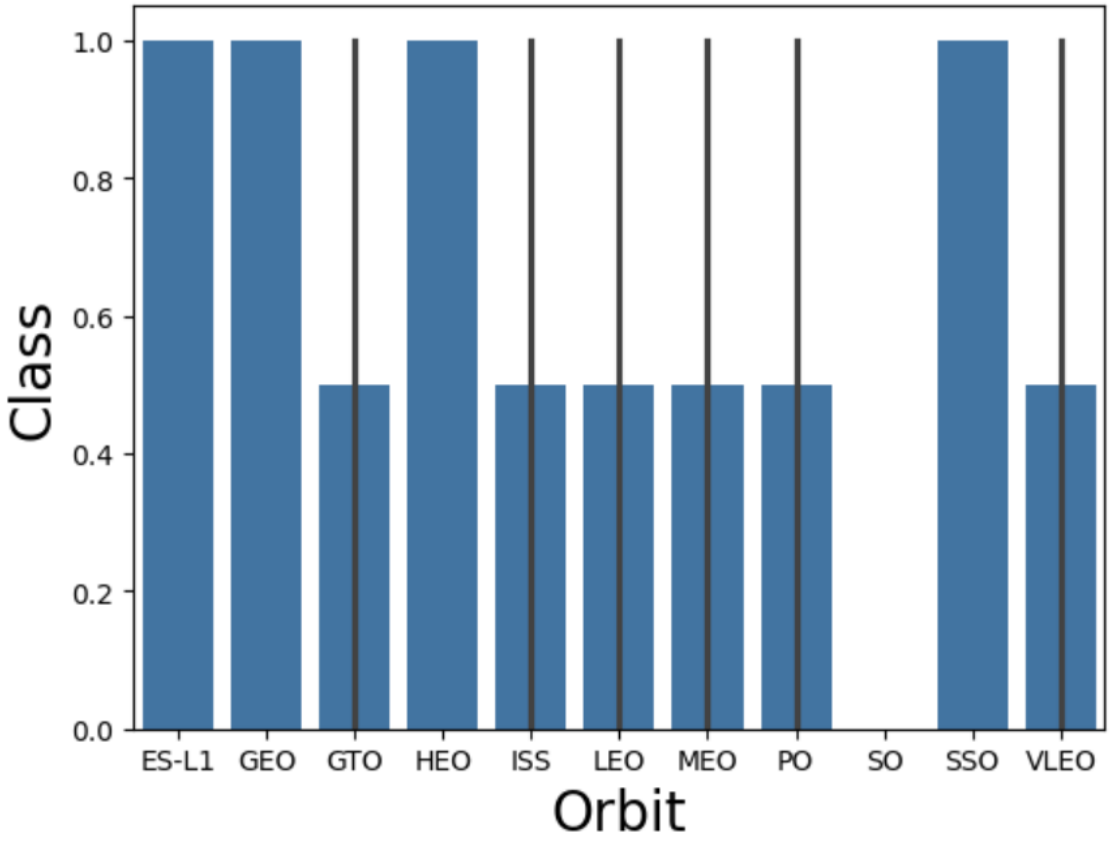


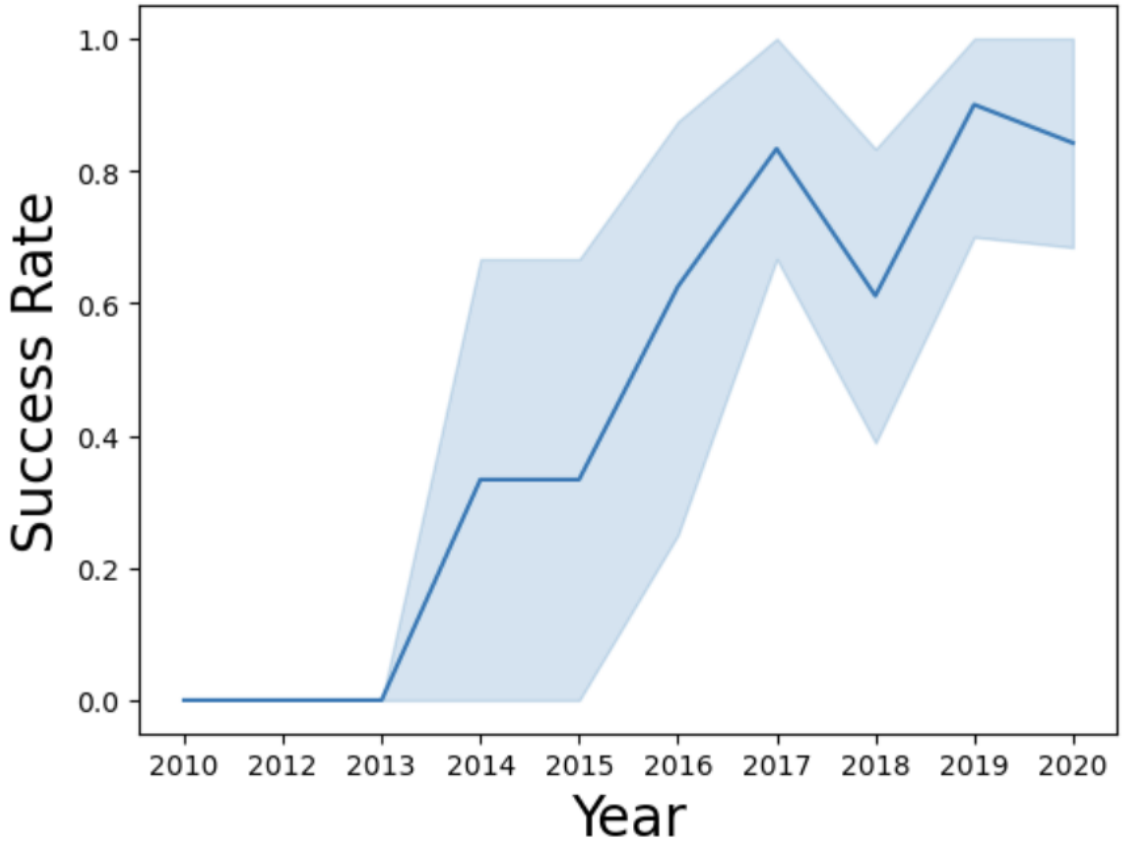


https://github.com/claumary/spaceXProject/blob/main/EDA%20with%20visualization.ipynb

with the bar chart we can see which is the better orbit, more success

The line chart show the trends of the success rate in the years





EDA with SQL

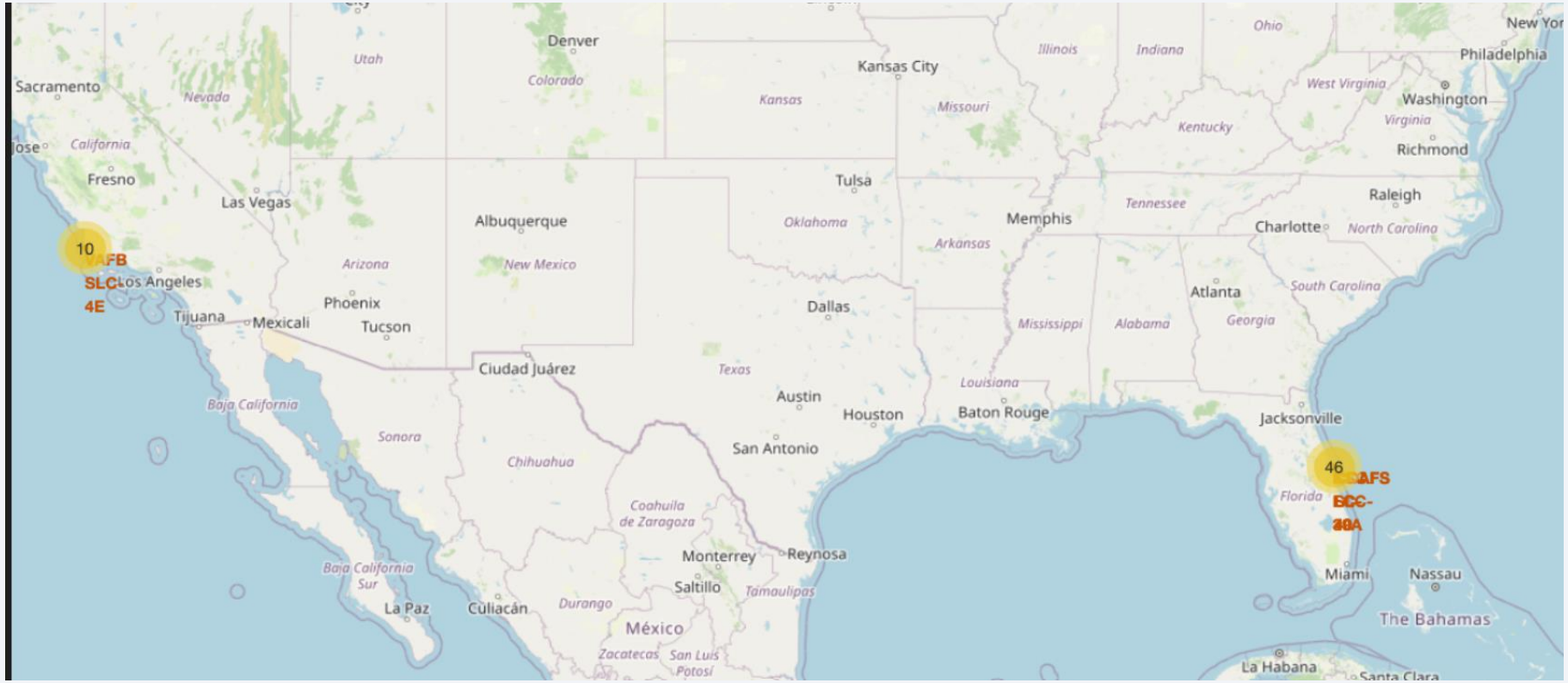
With SQL, we obtain information about the variable, this are the queries done:

- 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 succesful landing outcome in ground pad was acheived.
- 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. Use a subquery
- 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 landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

https://github.com/claumary/spaceXProject/blob/main/EDA-SQL-sqllite.ipynb

Build an Interactive Map with Folium

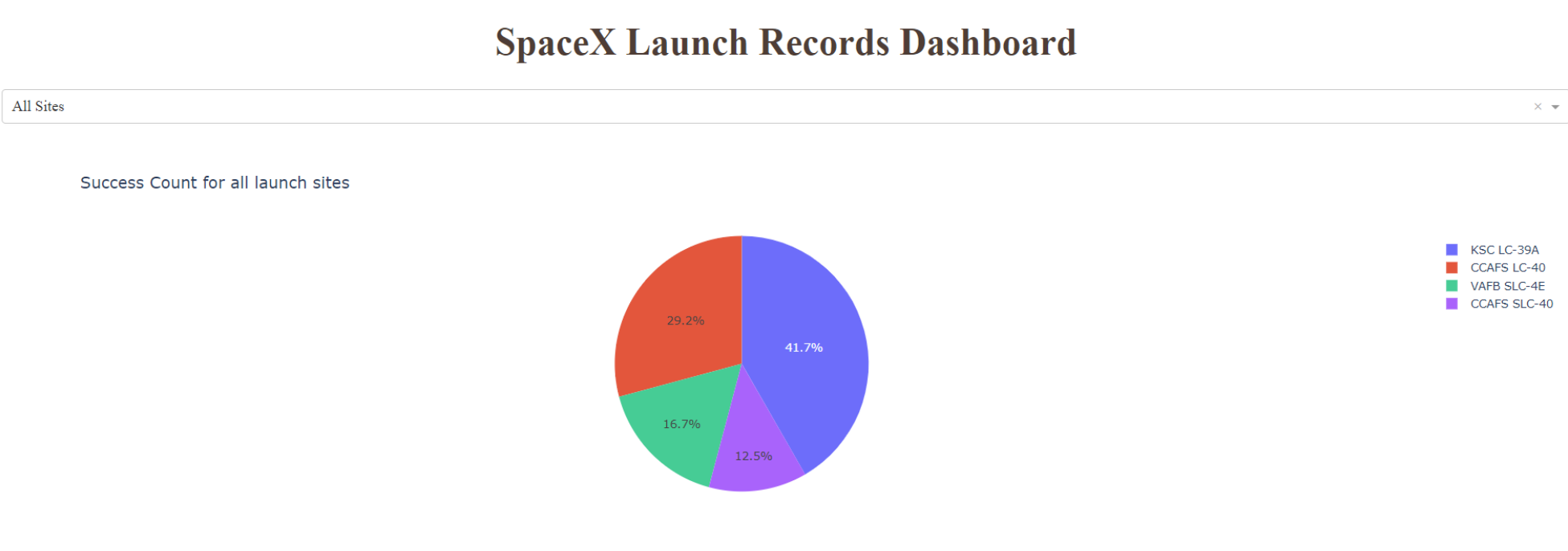
- Folium provide an easy way to analyze the data from the visualization maps.



https://github.com/claumary/spaceXProject/blob/main/Folium.ipynb

Build a Dashboard with Plotly Dash

With the plotly dash the user can dynamically interact with the data. It present a pie chart in which you can change the launch site and see what is the range of success, also it present the scatter plot graph showing the relation between the payload and success rate in differents scenarios. The dashboard permits change the inside for the graphics.



https://github.com/claumary/spaceXProject/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

The flow for this is:

- Building the model
- Evaluating the model
- Finding best performing classification model

For building the model used: KNN, Decision tree, logistic regression, and SNV.

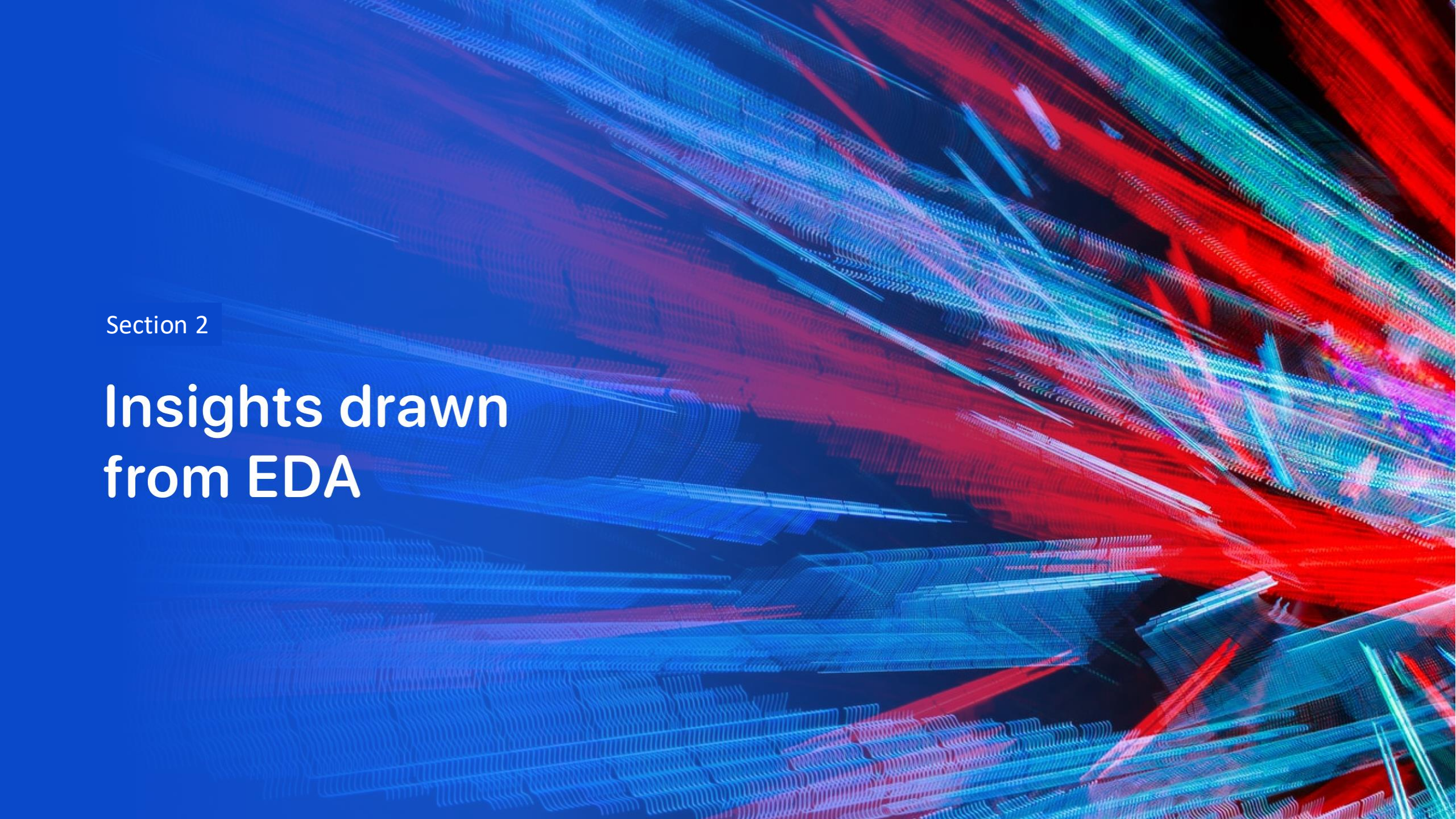
For evaluating the model used confusion matrix

For finding best performing calculate the accuracy score.

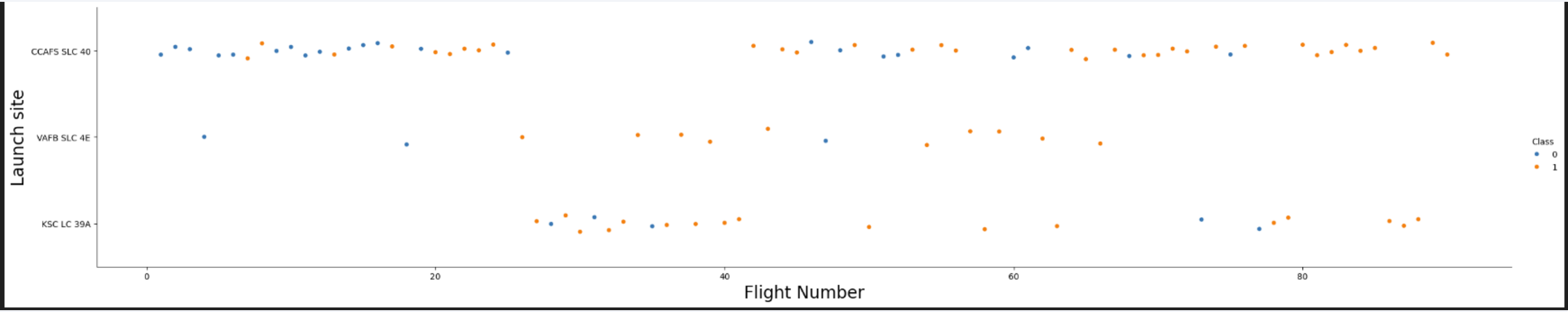
https://github.com/claumary/spaceXProject/blob/main/Machine%20Learning%20Prediction_Part_5.ipynb

Results

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

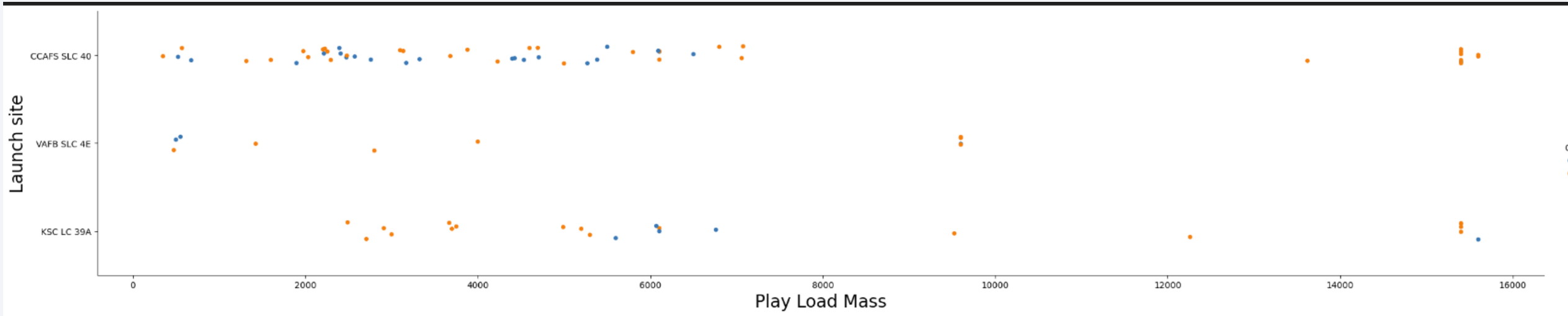

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



The higher flight number, the more rate success

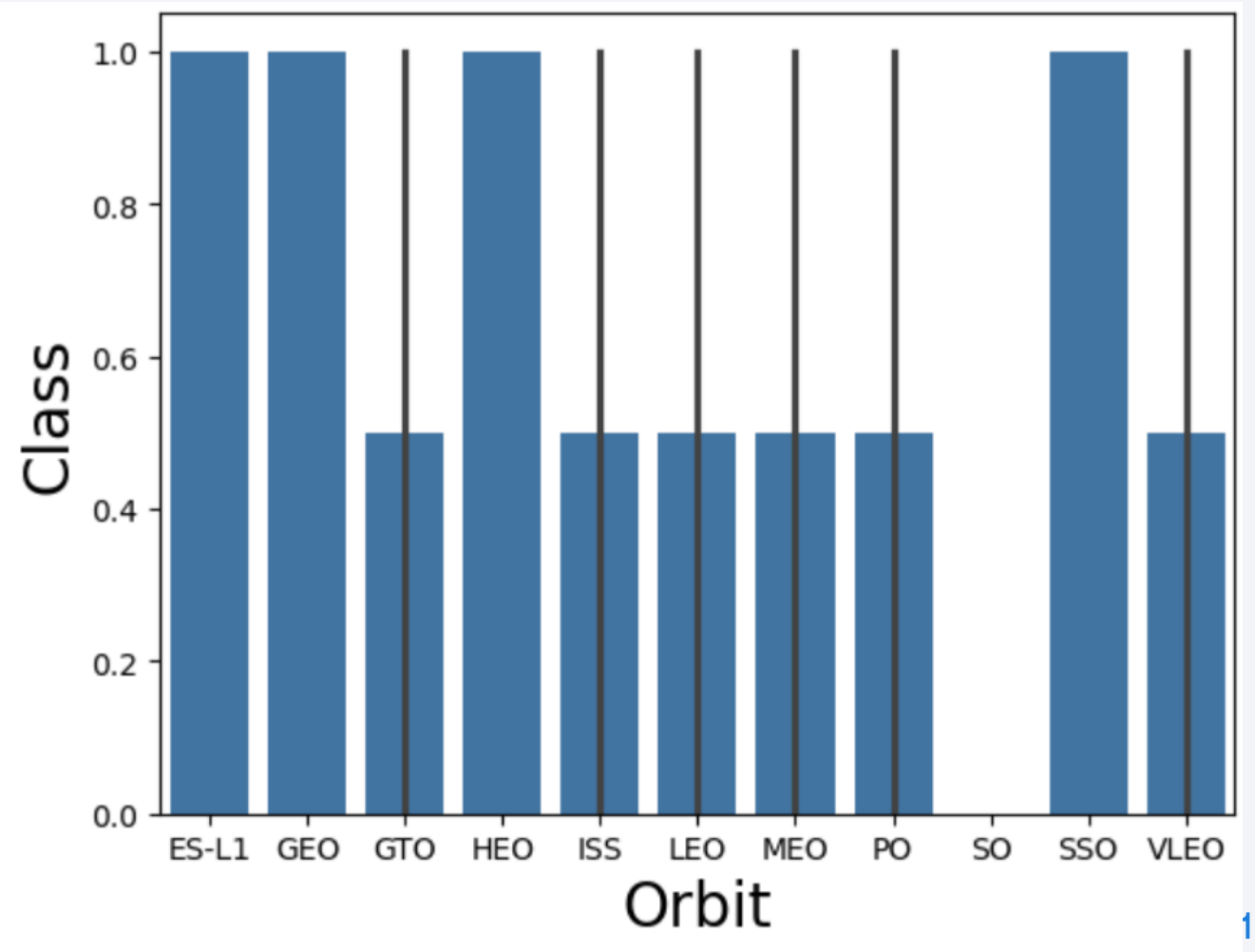
Payload vs. Launch Site



The greater load mass higher the success rate.

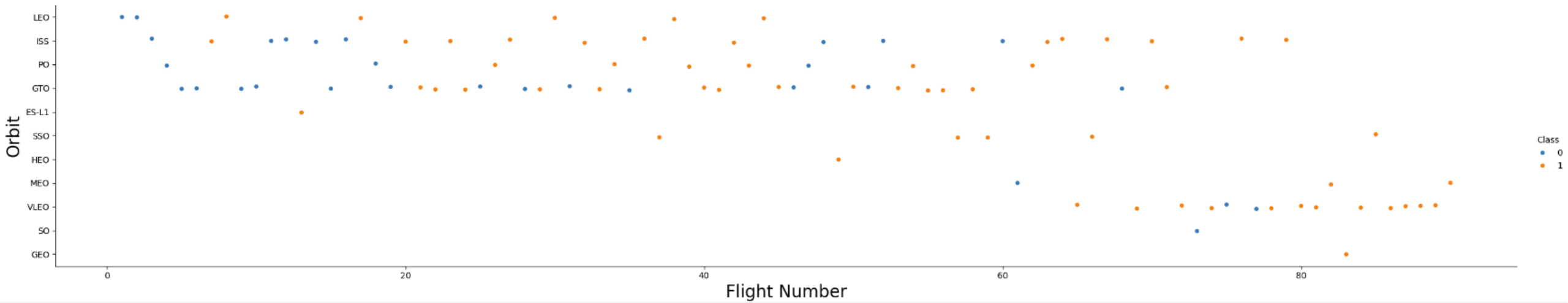
Success Rate vs. Orbit Type

- The highest rate of success are ES-L1, GEO, HEO, SSO



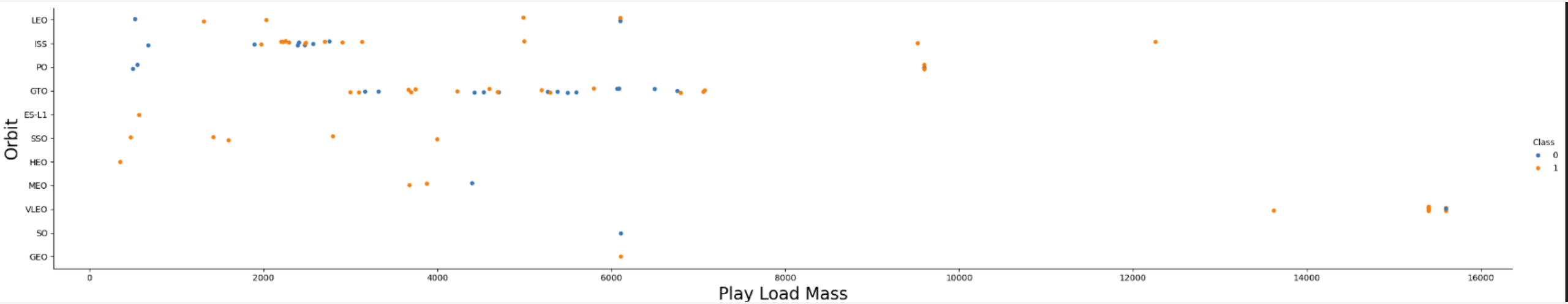
Flight Number vs. Orbit Type

There is no clear relationship between this two variables



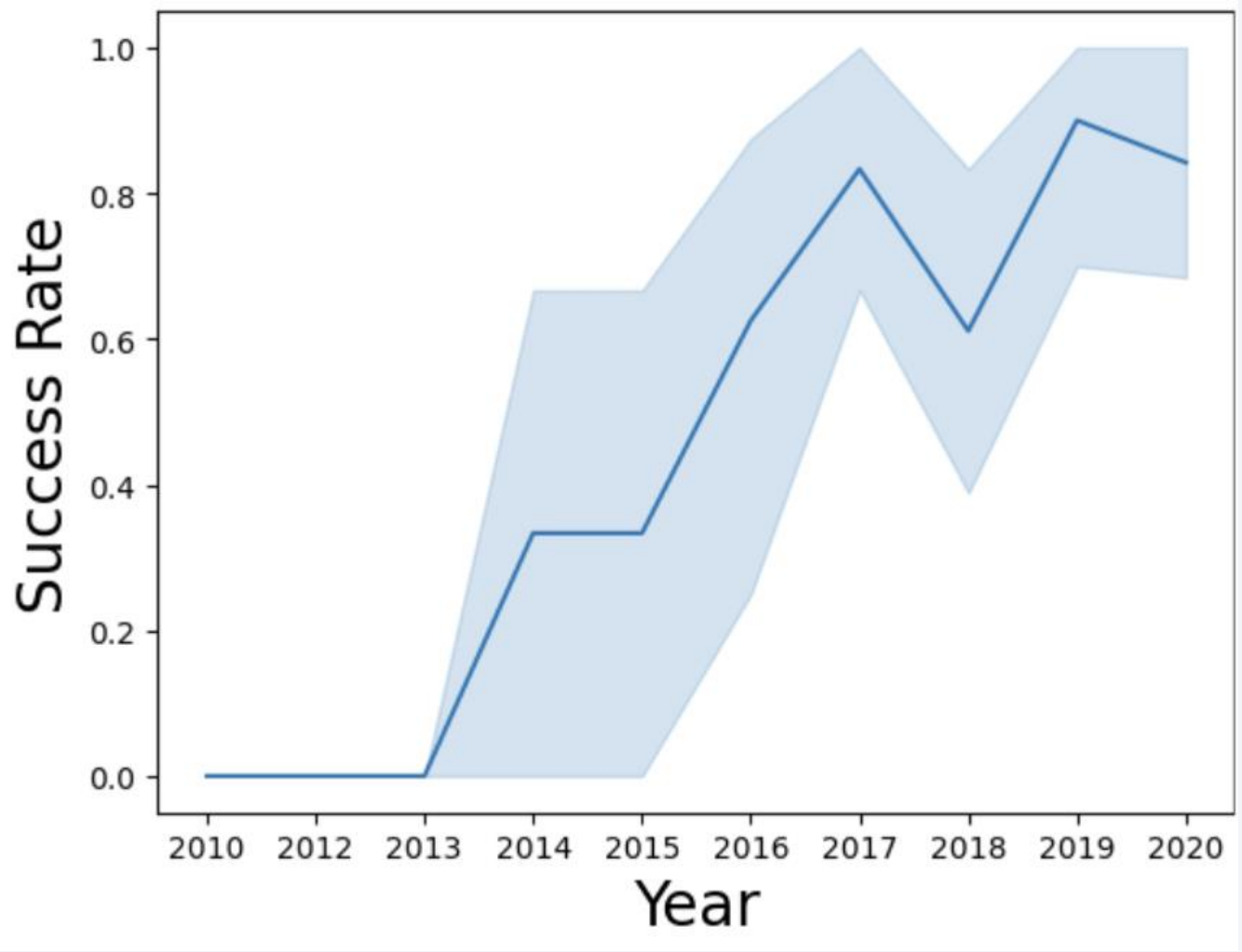
Payload vs. Orbit Type

- For some orbits the play load mass have a negative impact and for others have a positive impact



Launch Success Yearly Trend

- over the years the success rate has increased considerably.



All Launch Site Names

- The results of the consult of all launch site names

Display the names of the unique launch sites in the space mission

```
[12]: %sql select distinct(launch_site) from SPACEXTABLE
 * sqlite:///my_data1.db
Done.
```

[12]:

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

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with `CCA`
- Used Limit to obtain 5 records, and like to compare 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	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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- The reserve word SUM is used to summarized the payload mass and use where to limit to the customer with NASA

```
%sql select SUM(PAYLOAD_MASS__KG_) from SPACEXTABLE where CUSTOMER Like 'NASA (CRS)'
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- To calculate de average, used the reserve word avg for the variable PayloadMass, where the version is v1.1

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

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

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- For obtain the date used min for the variable date

```
[22]: %sql select min(DATE) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
 * sqlite:///my_data1.db
Done.
[22]: min(DATE)
      2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- For do the filter used the symbols <> with the numbers 4 and 6 k.

```
]: %sql select Booster_Version from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and 6000 > PAYLOAD_MASS__KG_ and PAYLOAD_MASS__KG_ > 4000
 * 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

- Calculate the total number of successful and failure mission outcomes
- To obtain the total number use count and group by.

```
[28]: %sql select Mission_Outcome, count(Mission_Outcome) from SPACEXTBL group by Mission_Outcome
 * sqlite:///my_data1.db
Done.
```

[28]:

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

Boosters Carried Maximum Payload

```
%sql select booster_version from SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (select MAX(PAYLOAD_MASS__KG_) from  SPACEXTBL)
```

```
 * 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

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

```
elect substr(Date, 6,2) as month, Booster_Version, Launch_site FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure%drone%' AND SUBSTR(Date,0,5) = '2015'
```

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

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

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

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.

```
%sql SELECT Landing_Outcome, COUNT(*) AS Numbers FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success%' AND Date BETWEEN '2010-06-04' AND '2017-03-20' GF
```

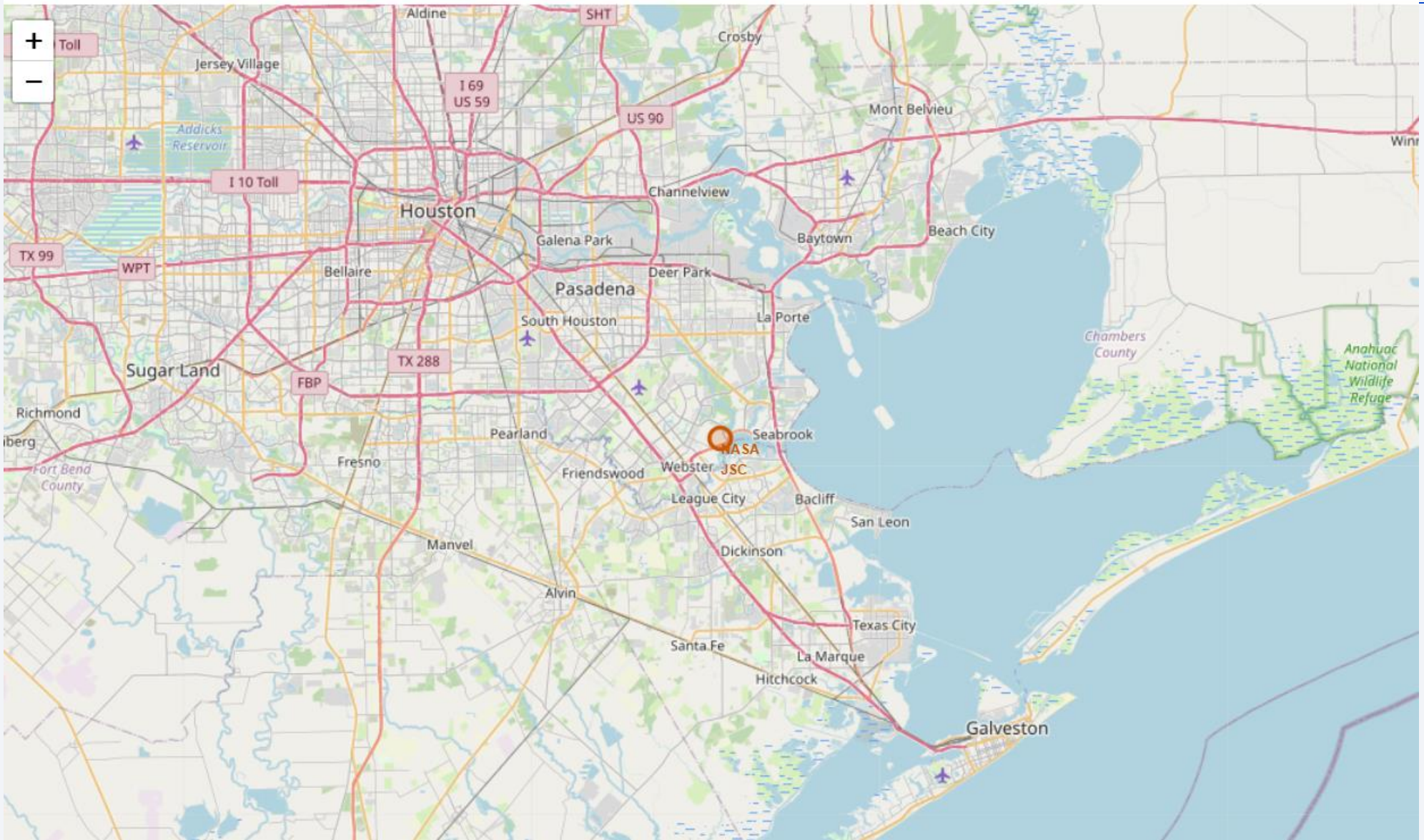
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	Numbers
Success (drone ship)	5
Success (ground pad)	3

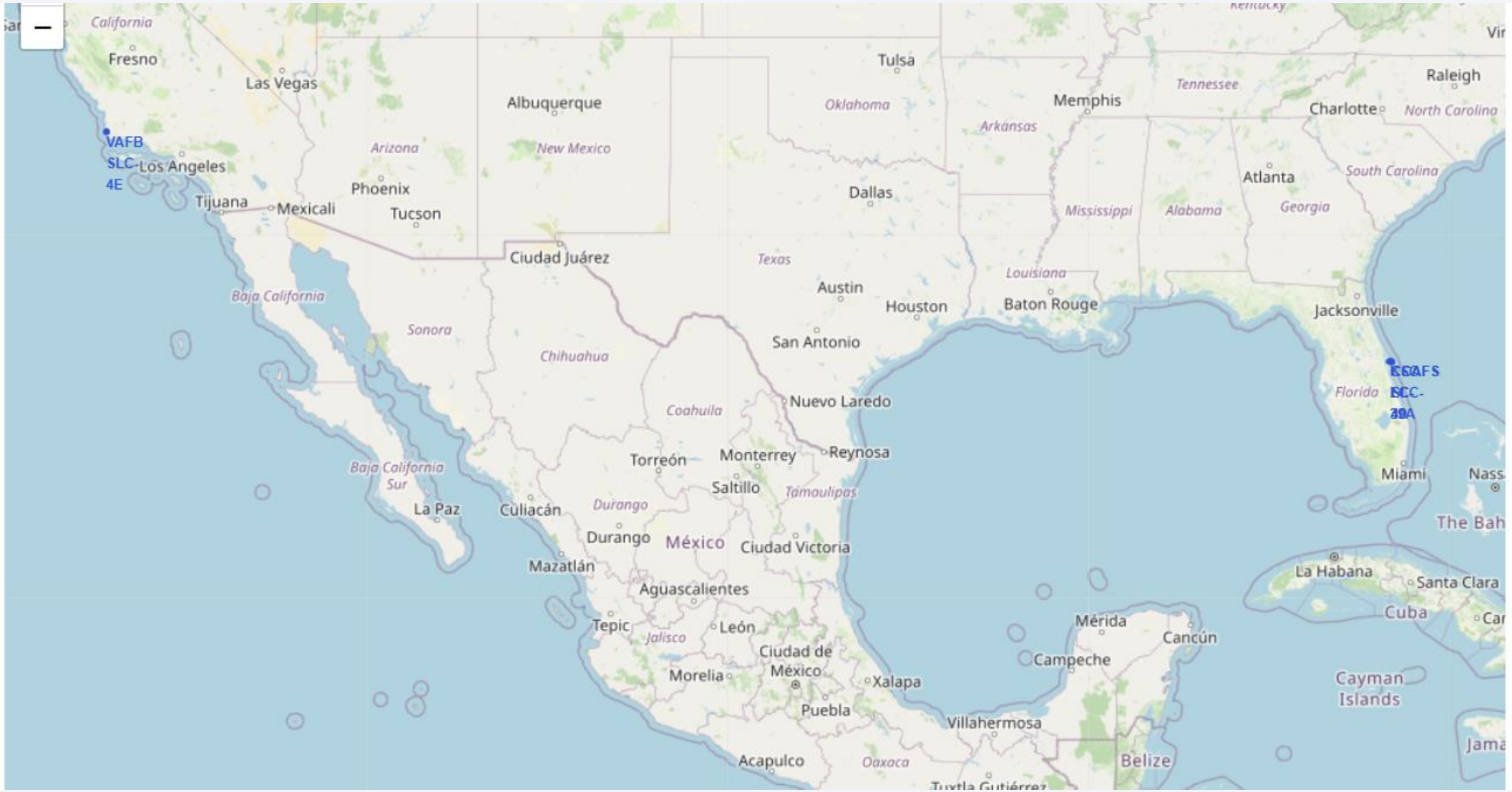
Section 3

Launch Sites Proximities Analysis

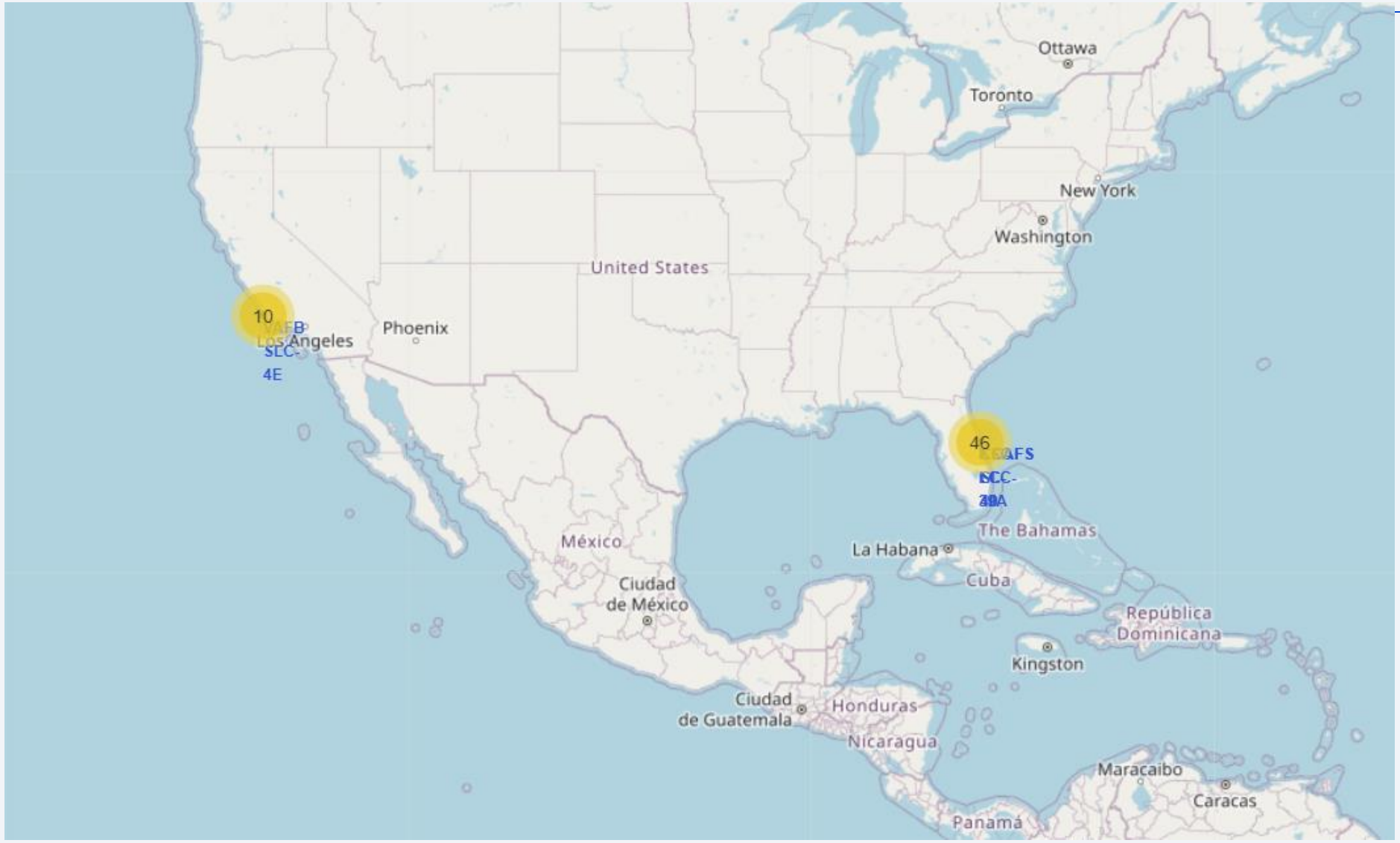
Launch sites

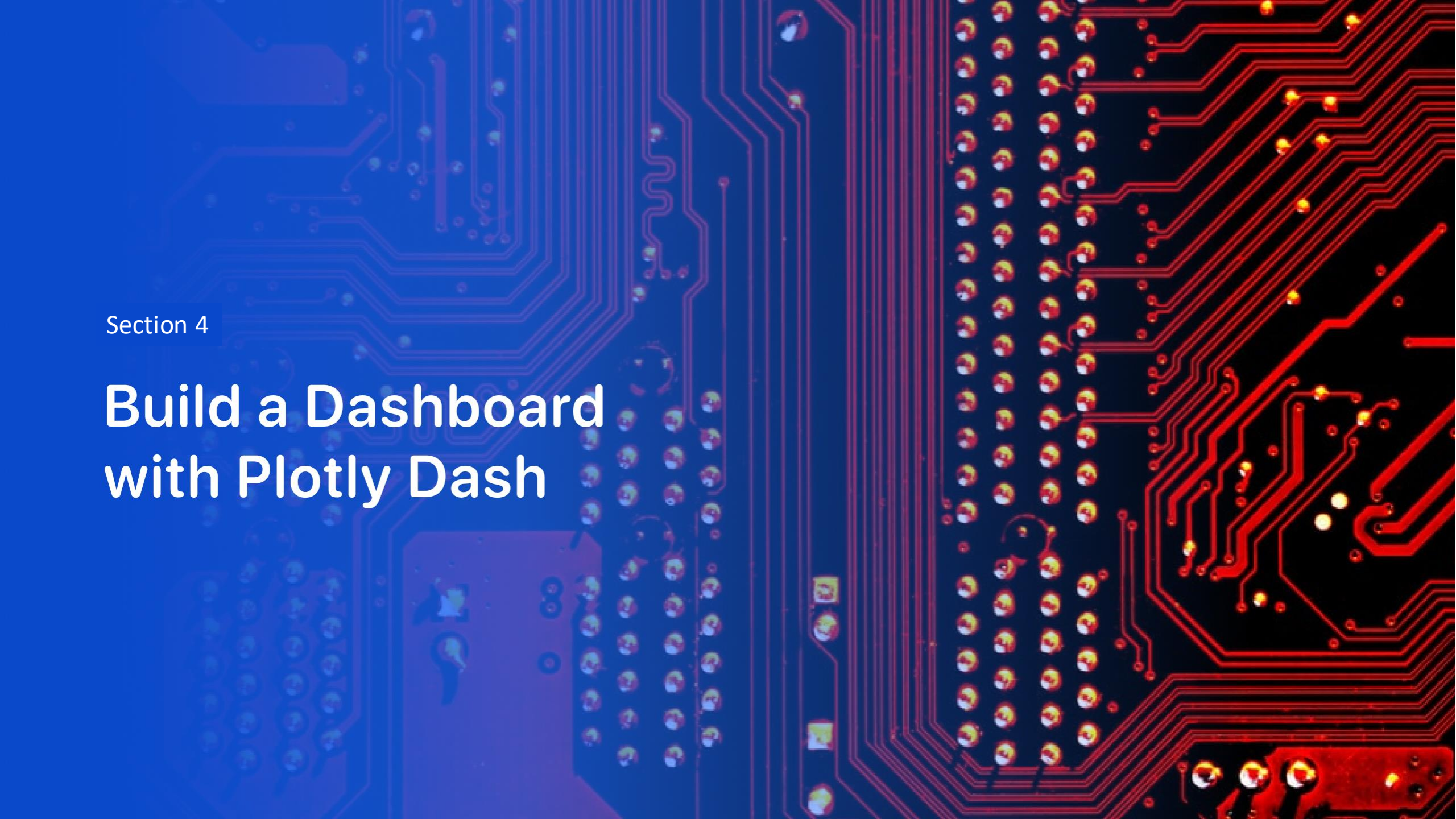


The flights in the map

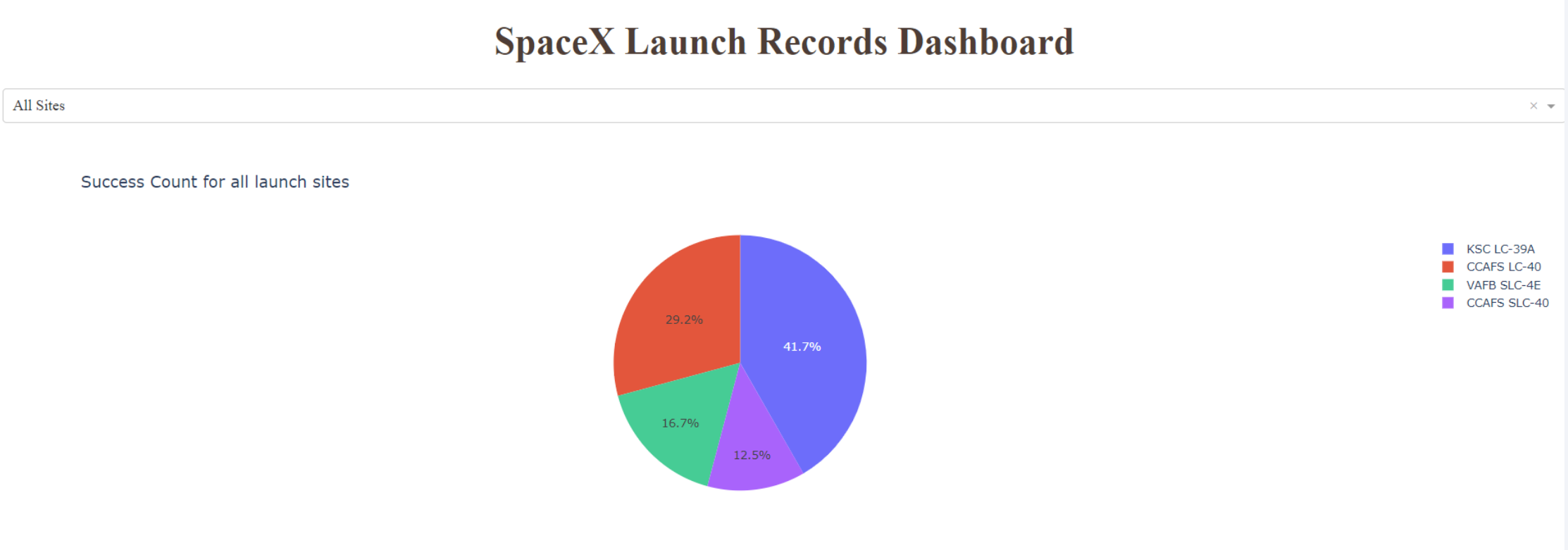


Color launch records



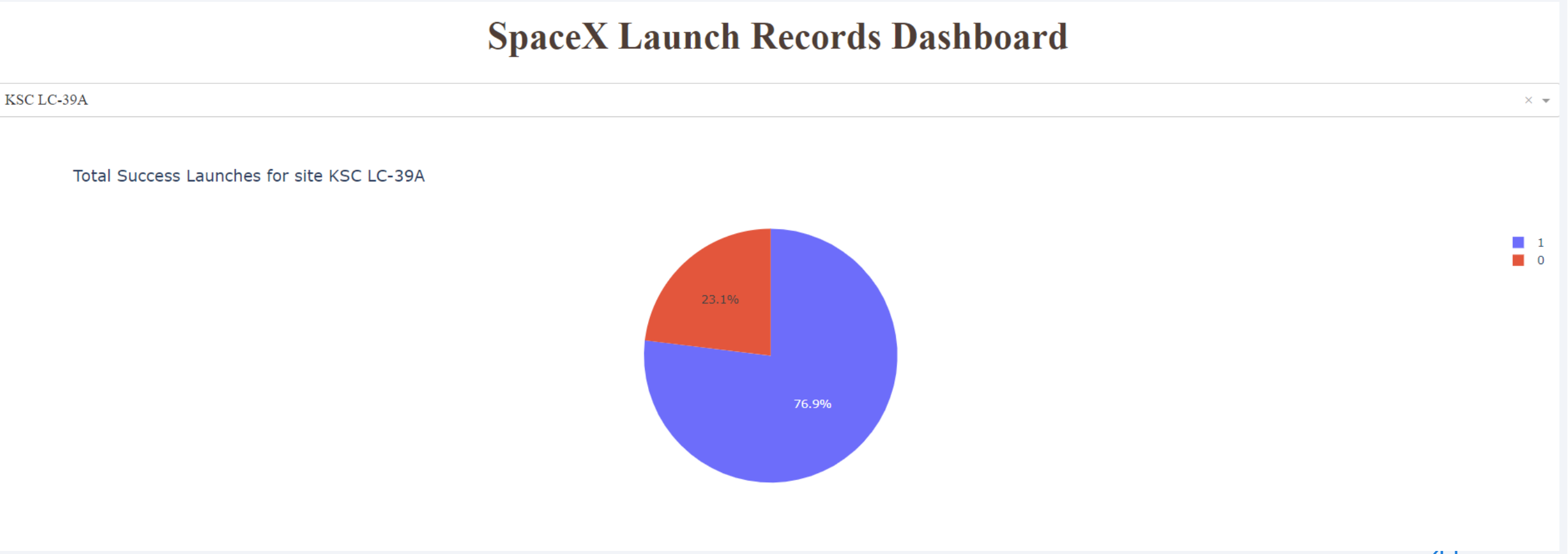

Section 4
Build a Dashboard with Plotly Dash

Success count for all launch sites



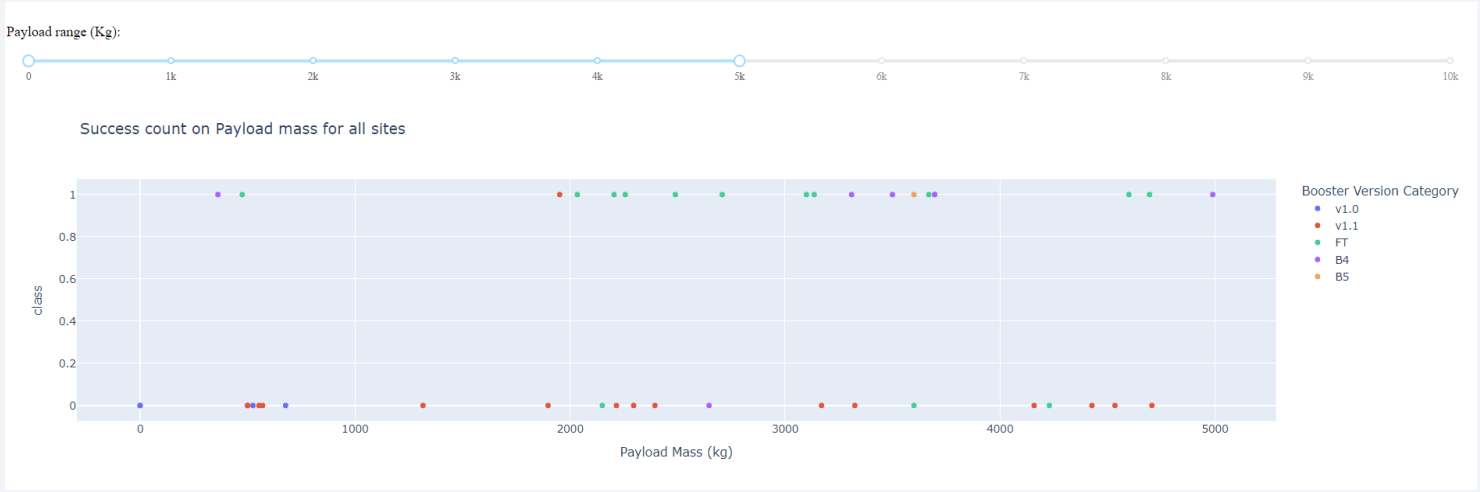
More success launches

We can see the success launches is for KSC LC



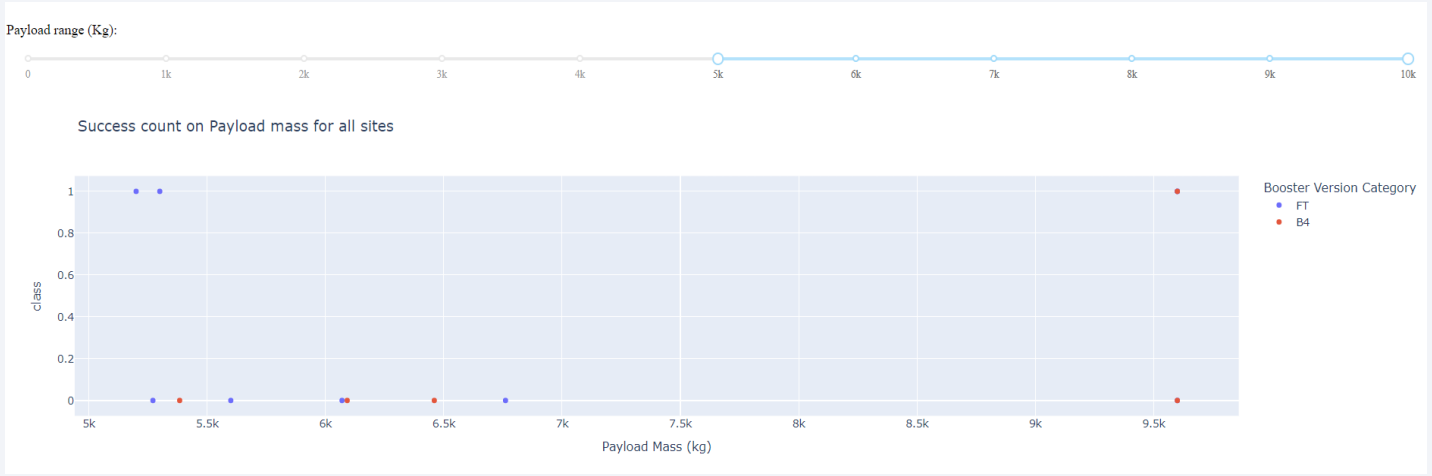
Mass vs Success rate

We can see the success rate is better in the low mass



0 to 5k

5k to 10k



Section 5

Predictive Analysis (Classification)

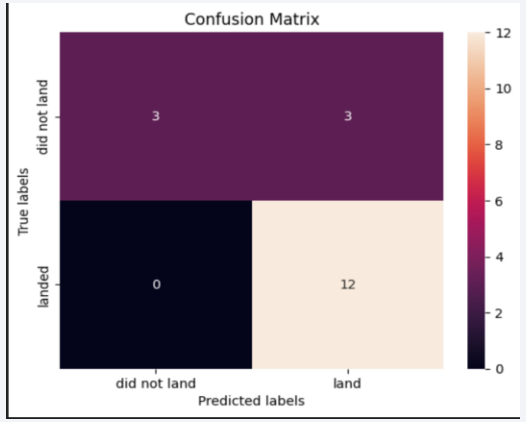
Classification Accuracy

- Calculate the accuracy for all the models, showing us the decision tree has the better accuracy

Algorithm	Accuracy
Logistic regression	0.8464285714285713
SVN	0.8482142857142856
Decision tree	0.875
KNN	0.8482142857142858

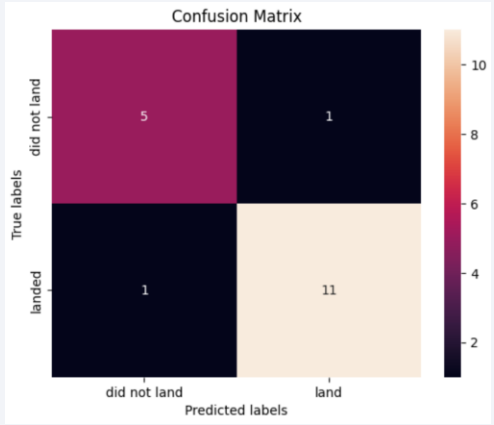
Confusion Matrix

Logistic regression

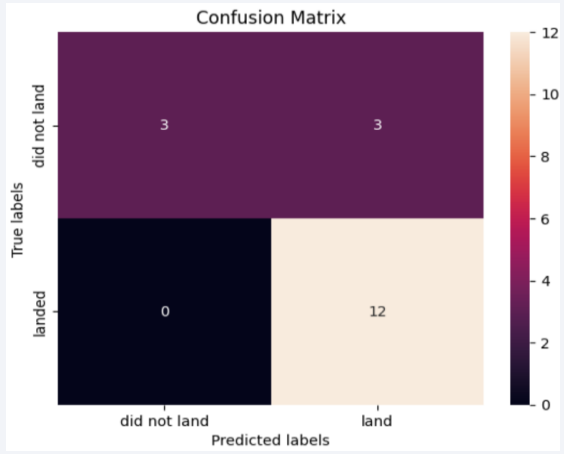


- The confusion matrix show us the result of the test sample.
- In this case almost the same confusion matrix is the result.

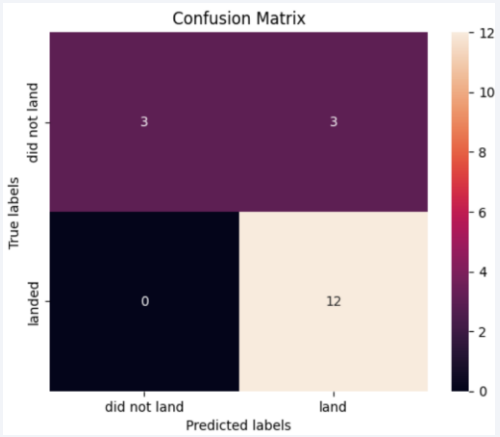
Decision tree



SVN



KNN



Conclusions

- Orbits ES-L1, GEO, HEO, SSO has the highest success rates
- Success rate has increased considerably over the time.
- The better classification model for this project is Decision tree

Appendix

This was a complete project, full of challenges and knowledge.

The Test and train set was one of the major parts of doing the prediction

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_test.shape,  Y_test.shape)
```

```
Train set: (72, 83) (72,)
Test set: (18, 83) (18,)
```

Thank you!