IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Mouammal Ziadah
04/01/2023

Outline

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

Executive Summary

- Summary of methodologies

1. Data Collection Data_Collection_API.ipynb Webscraping.ipynb
2. Data Wrangling Data_Wrangling.ipynb
3. Exploratory Data Analysis (EDA) EDA_Dataviz.ipynb EDA_SQL.ipynb
4. Interactive Visual Analytics IVA_Folium.ipynb spacex_dash_app.py
5. Predictive Analysis Machine_Learning_Prediction.ipynb

- Summary of all results

1. Results of EDA
2. Interactive analytics Demo
3. Results of Predictive Analysis

Introduction

- Project background and context :

- Unlike other rocket providers, SpaceX's Falcon 9 Can recover the first stage. Sometimes the first stage does not land. Sometimes it will crash as shown in this clip. Other times, Space X will sacrifice the first stage due to the mission parameters like payload, orbit, and customer.
- In this capstone, I will take the role of a data scientist working for a new rocket company. Space Y that would like to compete with SpaceX founded by Billionaire industrialist Allon Musk.

- Problems you want to find answers

1. Determine the price of each launch by gathering information about Space X and creating dashboards.
2. Determine if SpaceX will reuse the first stage by determining if the first stage will land successfully, train a machine learning model and use public information to predict if SpaceX will reuse the first stage.

Section 1

Methodology

Methodology

Executive Summary Github Link

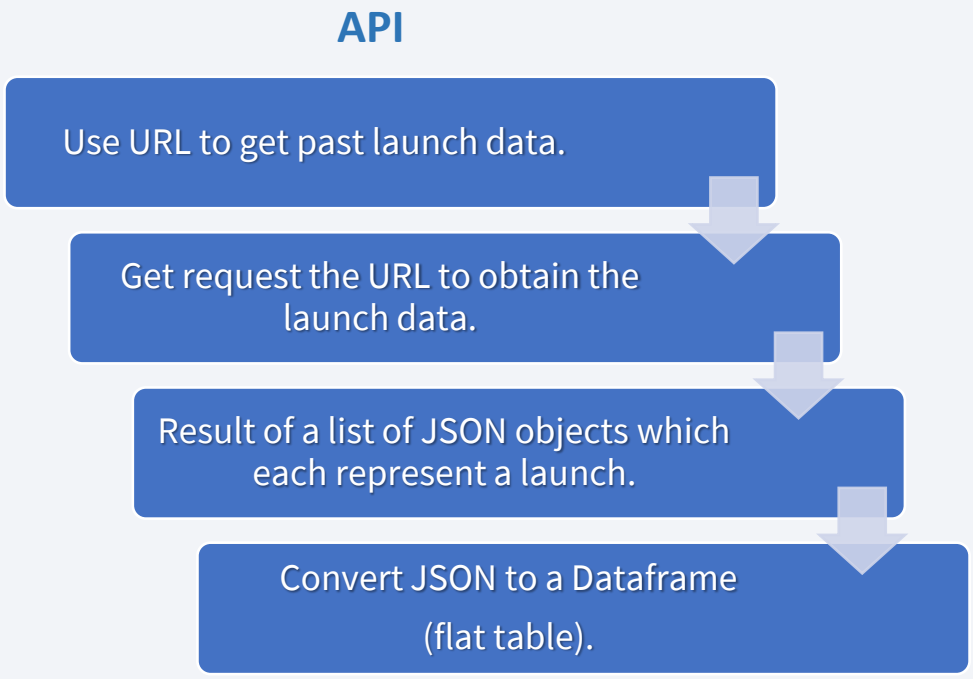
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

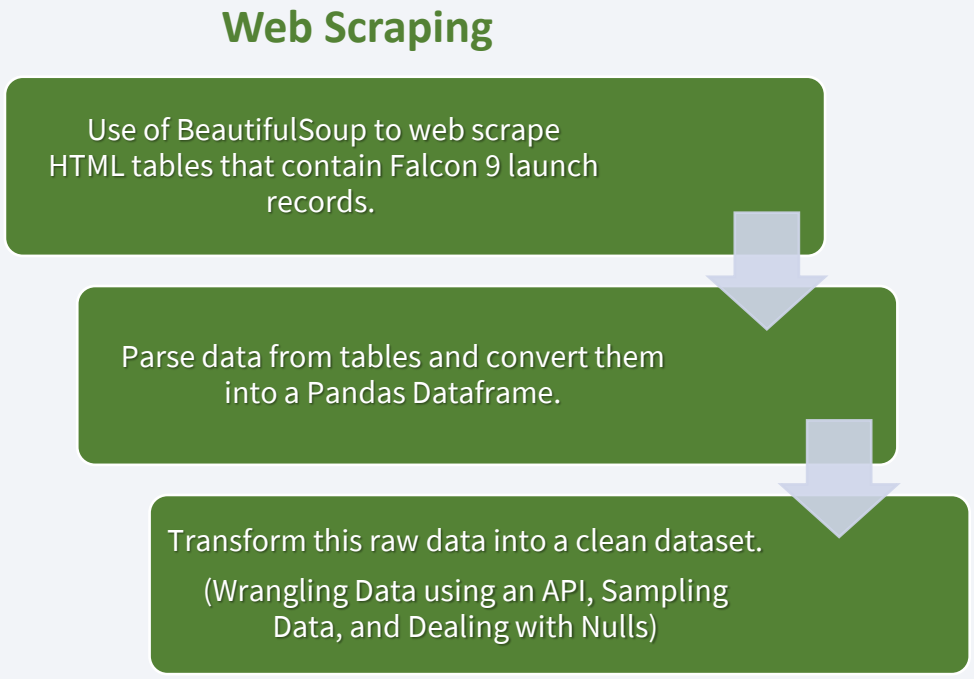
Data Collection

We used 2 methods to collect data : [Github_Link]

1. Collect SpaceX launch data that is gathered from **SpaceX REST API**.
2. Collect Falcon 9 Launch data with **Web Scraping** related Wiki pages.

→ GOAL = Use this data to predict whether SpaceX will attempt to land a rocket or not.





Data Collection – SpaceX API

Use URL to get past launch data.

Get request the URL to obtain the launch data.

Convert JSON with json_normalize method to a Dataframe (flat table).

Result of Dataframe of falcon 9 data.

[Github_Link]

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

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

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

```
data_falcon9.head(5)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	Lan
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	

Data Collection - Scraping

Use of BeautifulSoup to web scrape HTML tables that contain Falcon 9 launch records.

Parse data from tables and convert them into a Pandas Dataframe.

...

[Github_Link]

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

```
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])
```

Data Collection - Scraping

...

Transform this raw data into a clean dataset.

(Wrangling Data using an API, Sampling Data, and Dealing with Nulls)

...

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

```
df.head(5)
```

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Boost landir
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1	Failu
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1	Failu
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1	No attempt
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1	No attem
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1	No attempt

Github_Link

Data Wrangling

After collecting the data, we check the missing data, data types, clean and reorganize the data : Github_Link

1. Replace the missing data with one or mean value.
2. Change data type of the data.
3. Represent categorical data using integer or float, or dummy numbers (one hot encoding).

In the dataset, The column Outcome = first stage successfully landed. There are 8 of them, for example :

- True ASDS means the booster successfully landed to a drone ship.
- False ASDS means the mission outcome was unsuccessfully landed to a drone ship.

→ **GOAL** : We would like landing outcomes to be converted to Classes y (0 or 1).

Y will represent the classification variable that represents the outcome of each launch.)

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
```

```
df.isnull().sum()/df.shape[0]*100
```

```
FlightNumber      0.000000
Date              0.000000
BoosterVersion    0.000000
PayloadMass       0.000000
Orbit             0.000000
LaunchSite        0.000000
Outcome           0.000000
Flights           0.000000
GridFins          0.000000
Reused            0.000000
Legs              0.000000
LandingPad       28.888889
Block             0.000000
```

```
df.dtypes
```

```
FlightNumber      int64
Date             object
BoosterVersion   object
PayloadMass     float64
Orbit            object
LaunchSite       object
Outcome          object
Flights           int64
GridFins           bool
Reused             bool
Legs               bool
```

EDA with Data Visualization

After Data cleaning the we can proceed to Analyzing the data using visualization to get some insights of the launches with pandas, matplotlib and seaborn Github Link

6 Catplot chart

- FlightNumber and PayloadMass
- Flight Number and Launch Site
- Payload and Launch Site
- Outcome and Orbit
- Flight Number and Orbit
- PayLoad Mass (kg) and Orbit

4 Scatter point chart

- Flight Number and Launch Site
- PayLoad Mass (kg) and Launch Site
- Flight Number and Orbit
- PayLoad Mass (kg) and Orbit

1 Bar chart

- Orbit and Class

1 Line chart

- Year and Average Success Rate

EDA with SQL

Summary of the SQL queries performed :

- Names of the unique launch sites in the space mission.
- 5 records where launch sites begin with the string 'CCA'.
- Total payload mass carried by boosters launched by NASA (CRS).
- Average payload mass carried by booster version F9 v1.1.
- The date when the first successful landing outcome in ground pad was achieved.
- Names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
- Total number of successful and failure mission outcomes.
- names of the booster versions which have carried the maximum payload mass.
- Records which will display the month names, failure landing outcomes in drone ship ,booster versions, launch site for the months in year 2015.
- Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

Build an Interactive Map with Folium

Summary of map objects (markers, circles, lines) created and added to a folium map :

Github_Link

- **folium.Circle** = to add a highlighted circle area with a text label on a specific coordinate.

  → For visualizing launch sites on a map.

- **folium.Marker** = to add a marker on the location with anchor and location names.

  → For visualizing launch sites based on fail/success rate on a map with anchors.

- **MarkerCluster** = to simplify a map containing many markers having the same coordinate.

  → For a better visualization on a map full of markers.

- **folium.PolyLine** = Draw a line between 2 locations coordinates.

  → For analyzing proximities between launch sites or other location (coast…) on a map.

Build a Dashboard with Plotly Dash

Plotly Dash is Python library that makes it easier to create a dashboard for us as Data Scientist. Building a Plotly Dash application for users to perform interactive visual analytics on SpaceX launch data in real-time. Github Link

Summary of plots/graphs and interactions in the dashboard :

- Pie chart of Total success launches by sites and for all sites.

- Scatter plot of payload and launch outcome by sites and for all sites

- A slider of variable payload to see if mission outcome are correlated with payload.

Predictive Analysis (Classification)

Summary of built, evaluated, improved, and best performing classification model :

1. Load data
2. Normalize the data
3. Split the data in train dataset (80%) and test dataset (20%).
4. Select all parameters of a model and put it in a variable
5. Build Models (KNN, SVM, DECISION TREE, LOGISTIC REGRESSION)
6. Use Gridsearch to find best parameters of the models and train it (improve models)
7. Evaluate model with best parameters of Gridsearch (best performing)
8. Evaluate model with score method
9. Print a confusion matrix using the predict method to compare real label to predict label

Results

- Exploratory data analysis results

  → Dataframe with insights about how each important variable would affect the success rate, and one-hot encode all Dataframe to have only numbers.

- Interactive analytics demo in screenshots

  → Many interesting insights related to the launch sites' location using folium, in a very interactive way.

  → Most launches were form KSC PAD 39A since most of them were VLEO, GEO or ISS which makes it a good site to launch from.

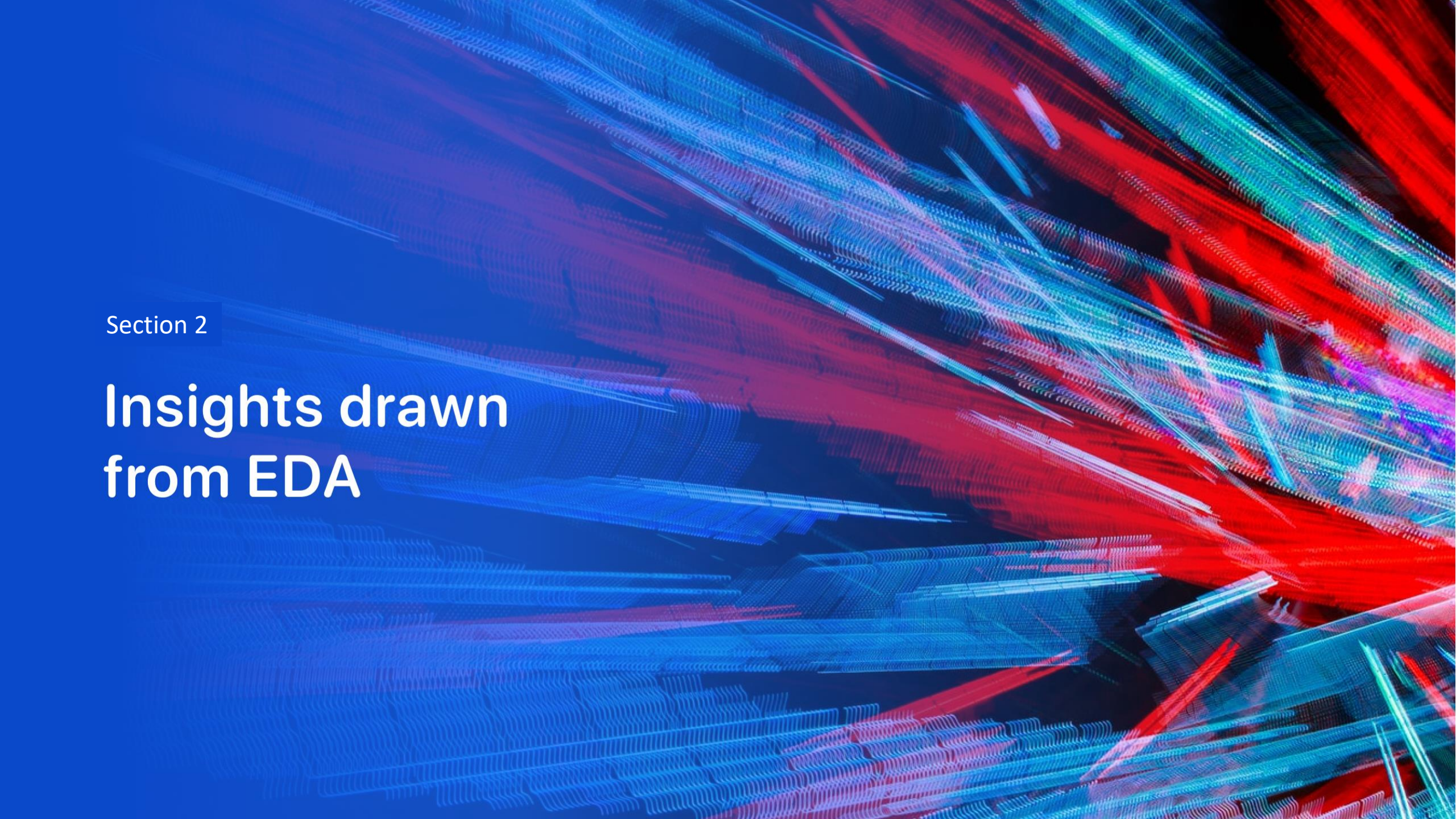
  → Falcon heavy launches mostly to full payload to maximize use of the falcon payload capacity.

  → Success rate increasing since 2013.

- Predictive analysis results

  → the most accurate model is the Tree model with best parameters (88%)

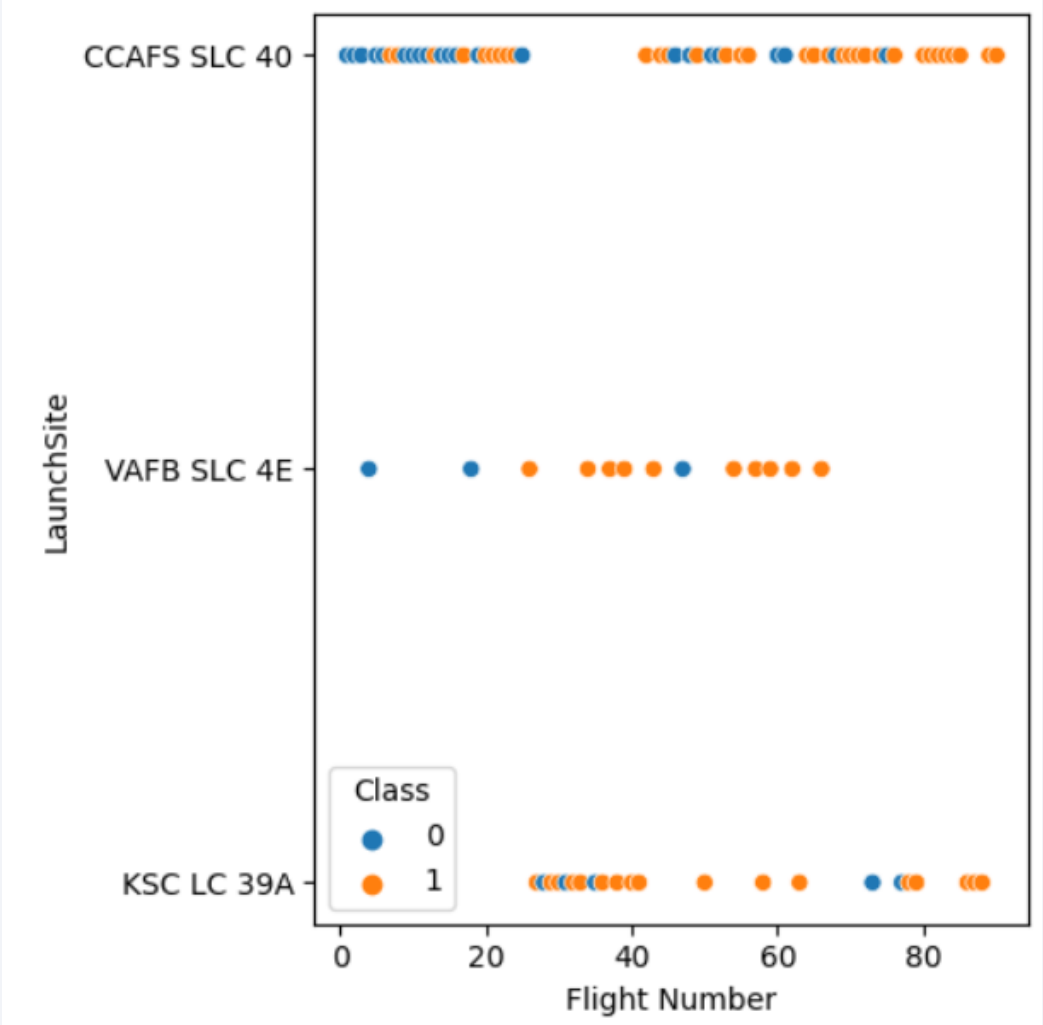
Github Link


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

Scatter plot of Flight Number vs. Launch Site

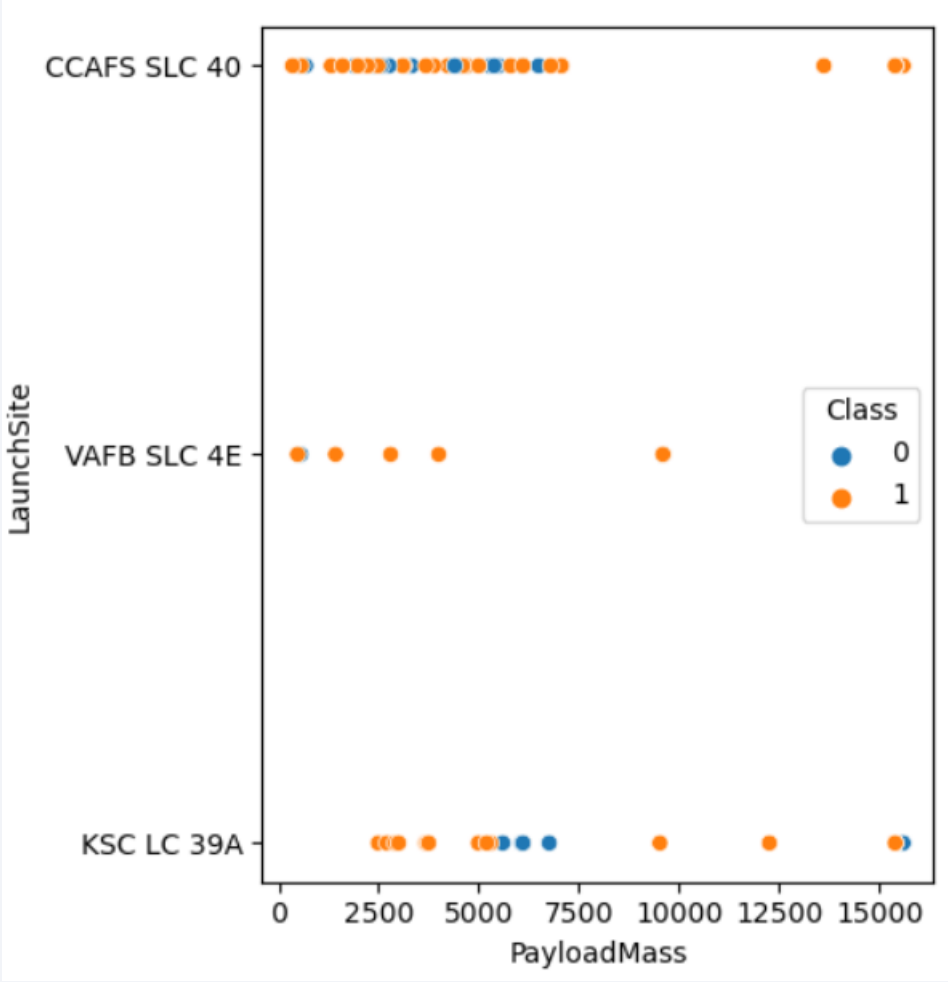
- Launch Sites are more likely to be successful when the flight number increases.
- After 80 rockets launched, KSC LC & CCAFS SLC launch sites are fully successful.



Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site

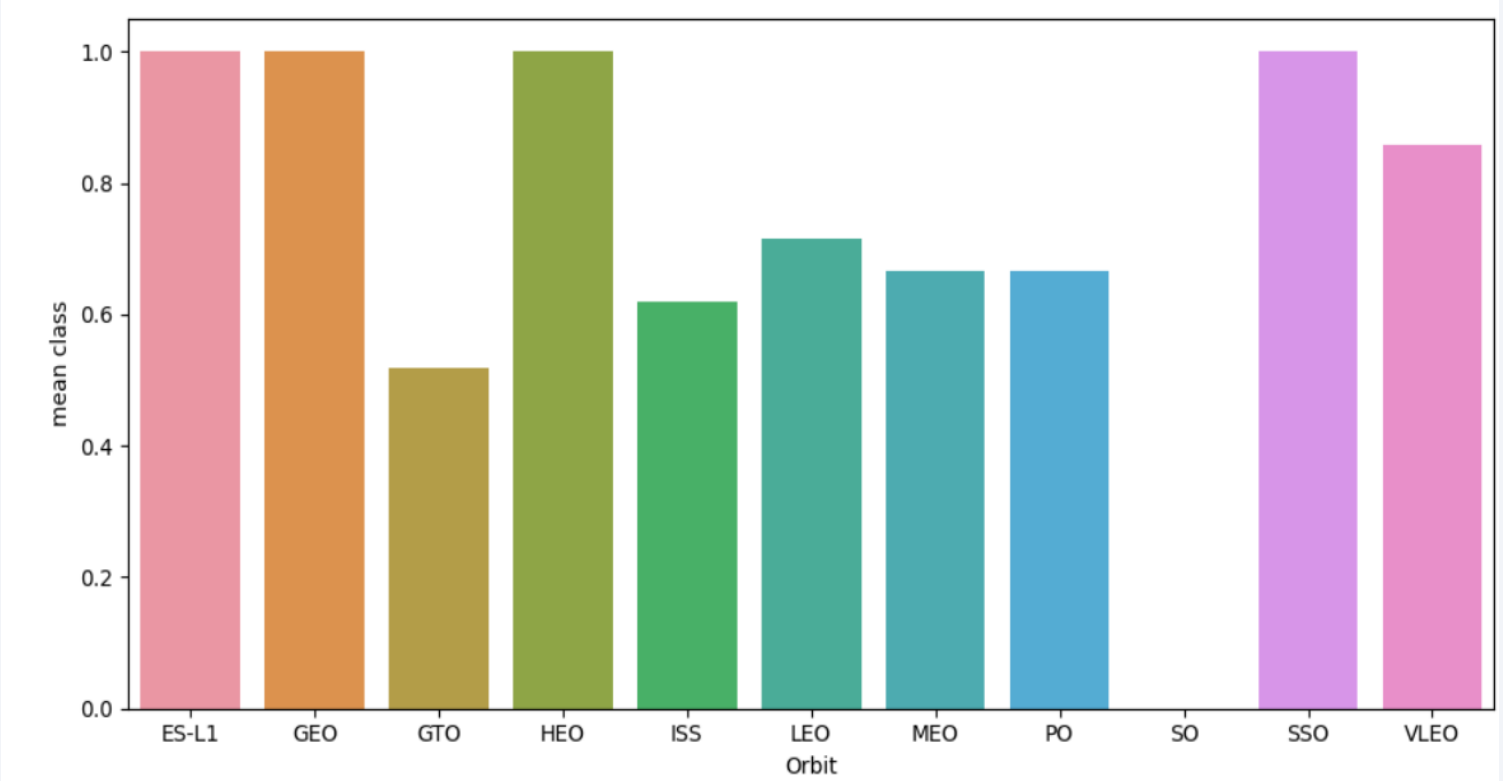
- VAFB-SLC launch site have NO rockets launched for heavy payload mass (greater than 10000).
- VAFB-SLC have successful landing for light payload.
- CCAFS SLC have successful landing for heavy payload (greater than 8000).



Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type

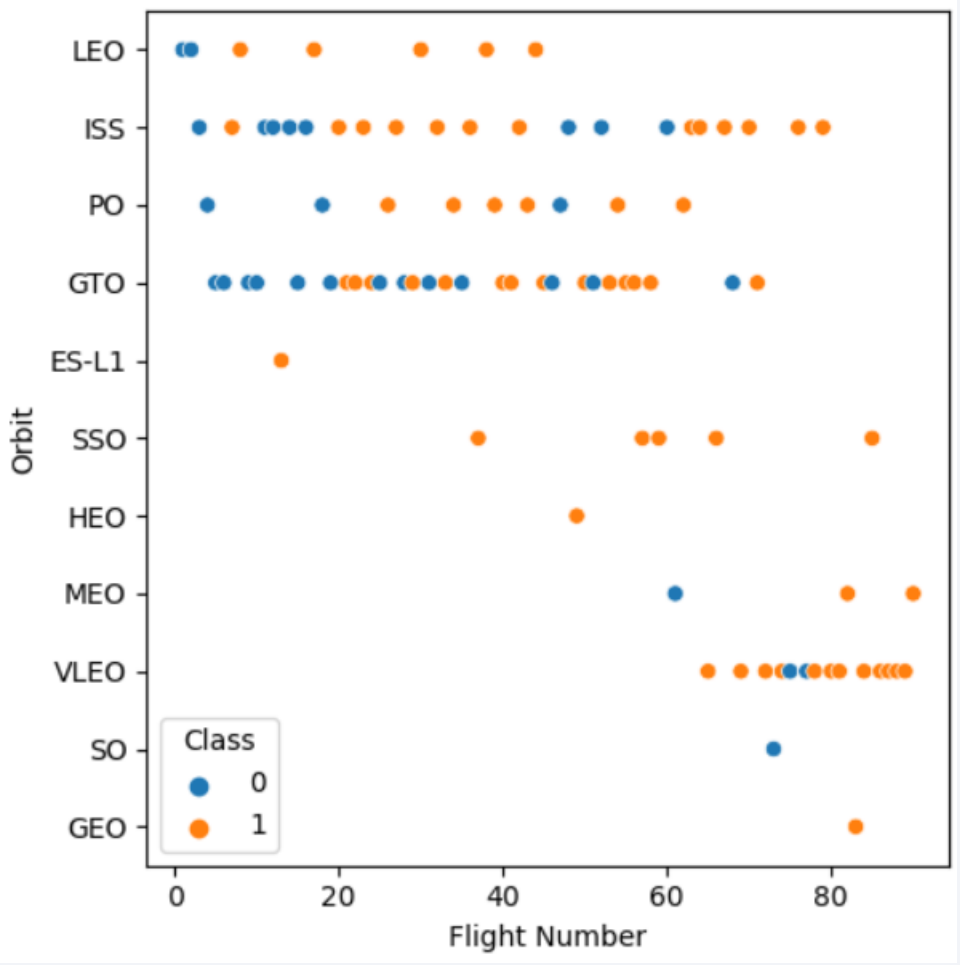
- ES-L1, GEO, HEO and SSO have high success rate.
- SO have zero success.



Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type

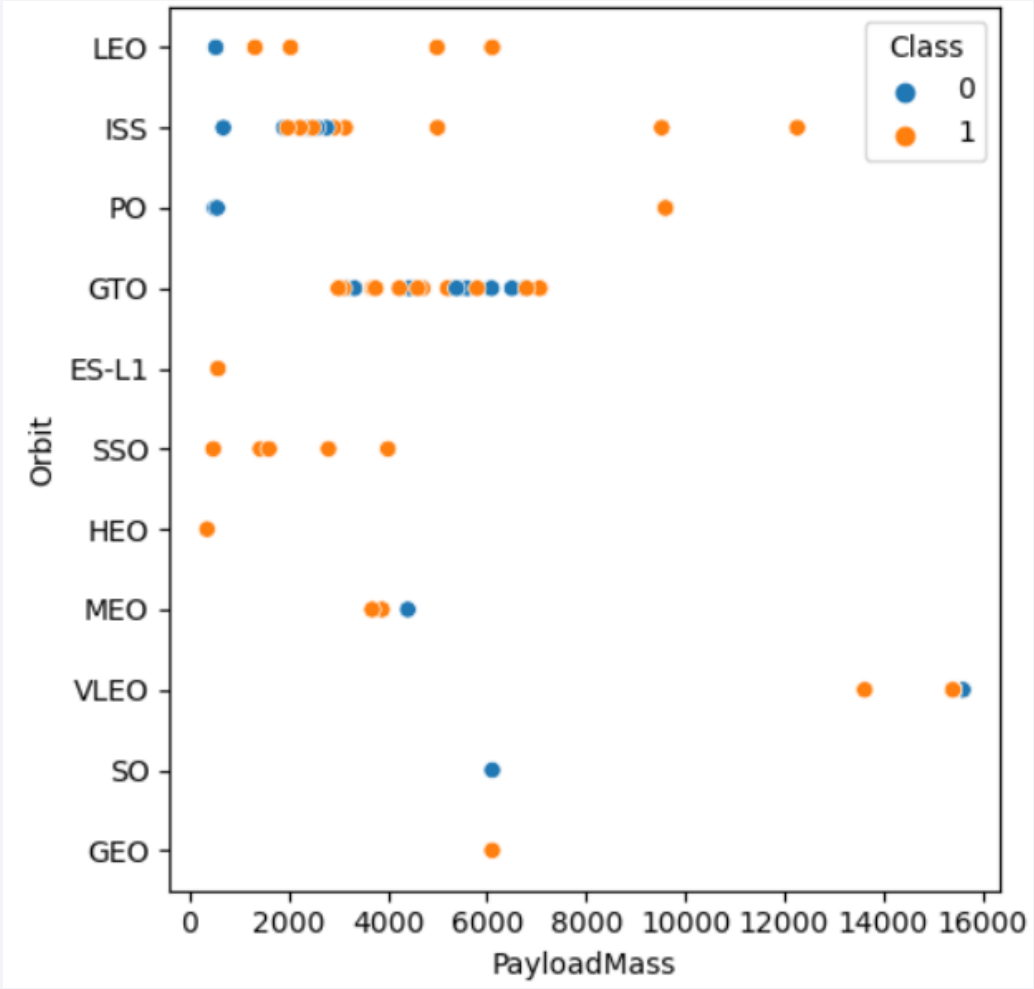
- For LEO orbit, Success appears to be related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.
- High success rate for ES, SSO, HEO, GEO orbit.
- Most failed for ISS and GTO orbit.



Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

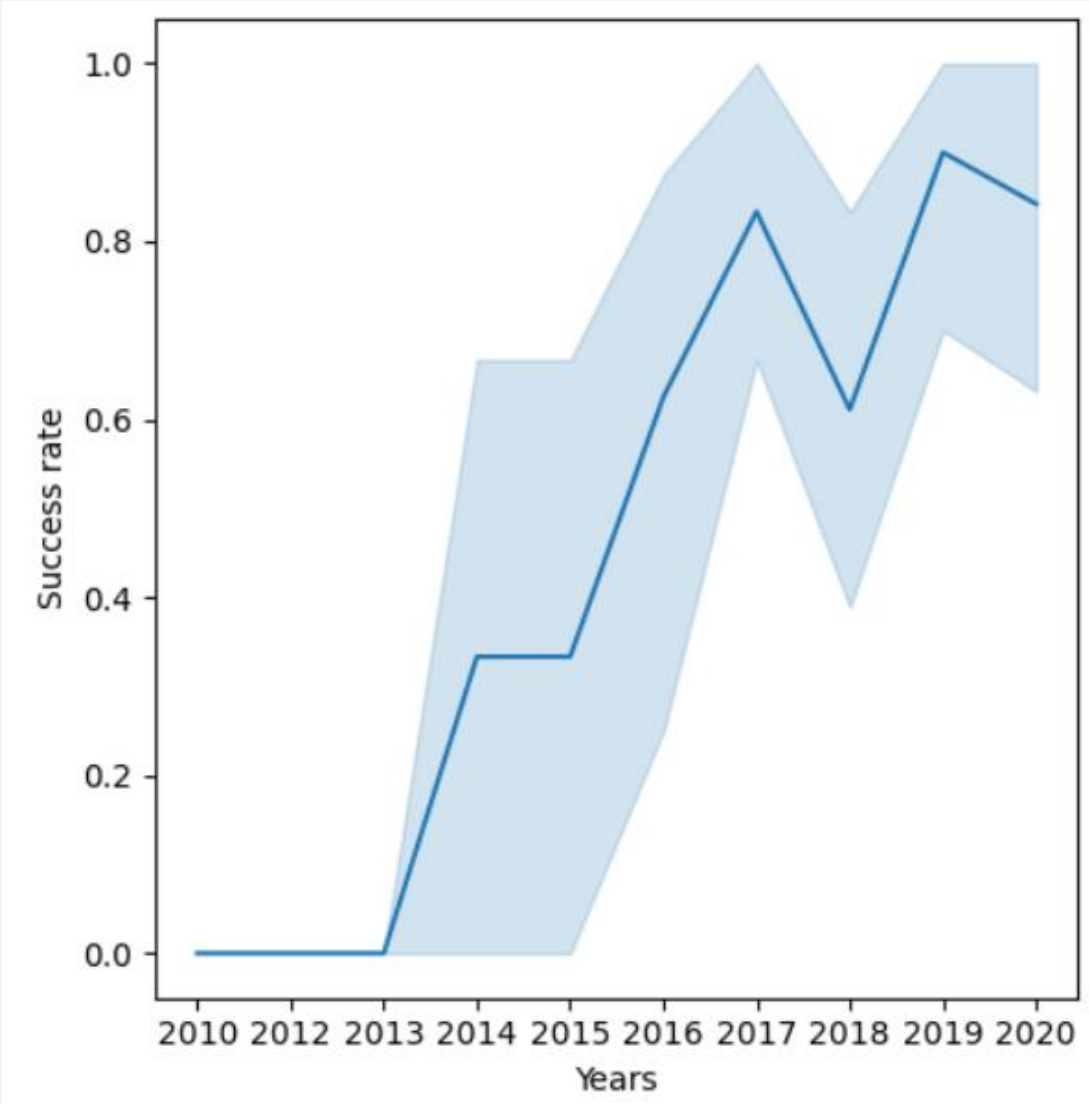
- Heavy payloads = Successful landing are more for PO, LEO and ISS.
- For GTO orbit is not distinguishable, positive landing rate + negative landing are both there.
- SSO have successful rate for low payload mass.



Launch Success Yearly Trend

- Show a line chart of yearly average success rate

- Success rate, since 2013 kept increasing till 2020



All Launch Site Names

- Find the names of the unique launch sites

```
%sql select DISTINCT Launch_Site from SPACEXTBL
```

```
 * 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'

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

```
%sql select * from SPACEXTBL 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
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	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

```
%sql select Customer, SUM(PAYLOAD_MASS__KG_) as total  from SPACEXTBL WHERE Customer = "NASA (CRS)"
```

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

Customer	total
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

```
%sql select Booster_Version, AVG(PAYLOAD_MASS__KG_) as avg from SPACEXTBL WHERE Booster_Version LIKE "F9 v1.1%"
```

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

Booster_Version	avg
F9 v1.1 B1003	2534.6666666666665

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
%sql select max("Date"), "Mission_Outcome", "Landing _Outcome" from SPACEXTBL where "Landing _Outcome" = "Success (ground pad)"
```

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

max("Date")	Mission_Outcome	Landing _Outcome
22-12-2015	Success	Success (ground pad)

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

```
%%sql select Booster_Version, PAYLOAD_MASS__KG_, "Landing _Outcome" from SPACEXTBL
where "Landing _Outcome" = "Success (drone ship)"
and PAYLOAD_MASS__KG_ between 4000 and 6000
```

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

Booster_Version	PAYLOAD_MASS__KG_	Landing _Outcome
F9 FT B1022	4696	Success (drone ship)
F9 FT B1026	4600	Success (drone ship)
F9 FT B1021.2	5300	Success (drone ship)
F9 FT B1031.2	5200	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
%sql select "Mission_Outcome", count("Mission_Outcome") as total from SPACEXTBL
```

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

Mission_Outcome	total
Success	101

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

```
%%sql select Booster_Version,
(select max(PAYLOAD_MASS__KG_) from SPACEXTBL) as max_payload from SPACEXTBL
```

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

Booster_Version	max_payload
F9 v1.0 B0003	15600
F9 v1.0 B0004	15600
F9 v1.0 B0005	15600
F9 v1.0 B0006	15600
F9 v1.0 B0007	15600
F9 v1.1 B1003	15600
F9 v1.1	15600
F9 v1.1	15600
F9 v1.1	15600
F9 v1.1	15600

2015 Launch Records

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

```
%%sql select substr(Date, 4, 2) as month, substr(Date,7,4) as year,
"Landing _Outcome", Booster_Version, Launch_Site from SPACEXTBL
where "Landing _Outcome" = 'Failure (drone ship)'
and year = "2015"
```

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

month	year	Landing _Outcome	Booster_Version	Launch_Site
01	2015	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	2015	Failure (drone ship)	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

```
%%sql select "Date", "Landing _Outcome", count(*) from SPACEXTBL where "Date" between "04-06-2010" and "20-03-2017"
and "Landing _Outcome" like "%Success%" group by "Landing _Outcome" order by "Date" DESC
```

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

Date	Landing _Outcome	count(*)
18-07-2016	Success (ground pad)	6
08-04-2016	Success (drone ship)	8
07-08-2018	Success	20

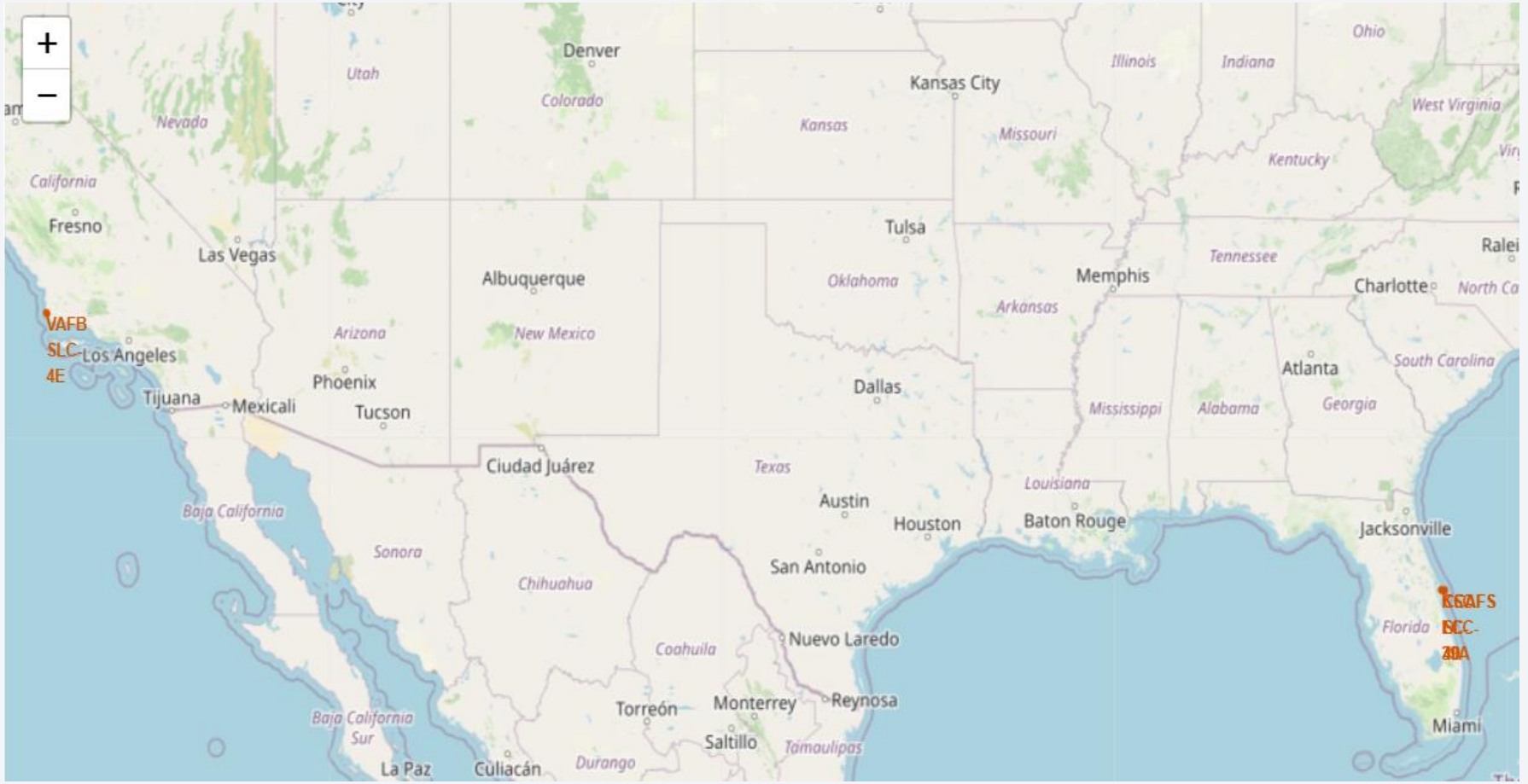
Section 3

Launch Sites Proximities Analysis

Folium Map all launch site's location markers

Folium map of all launch site's location markers.

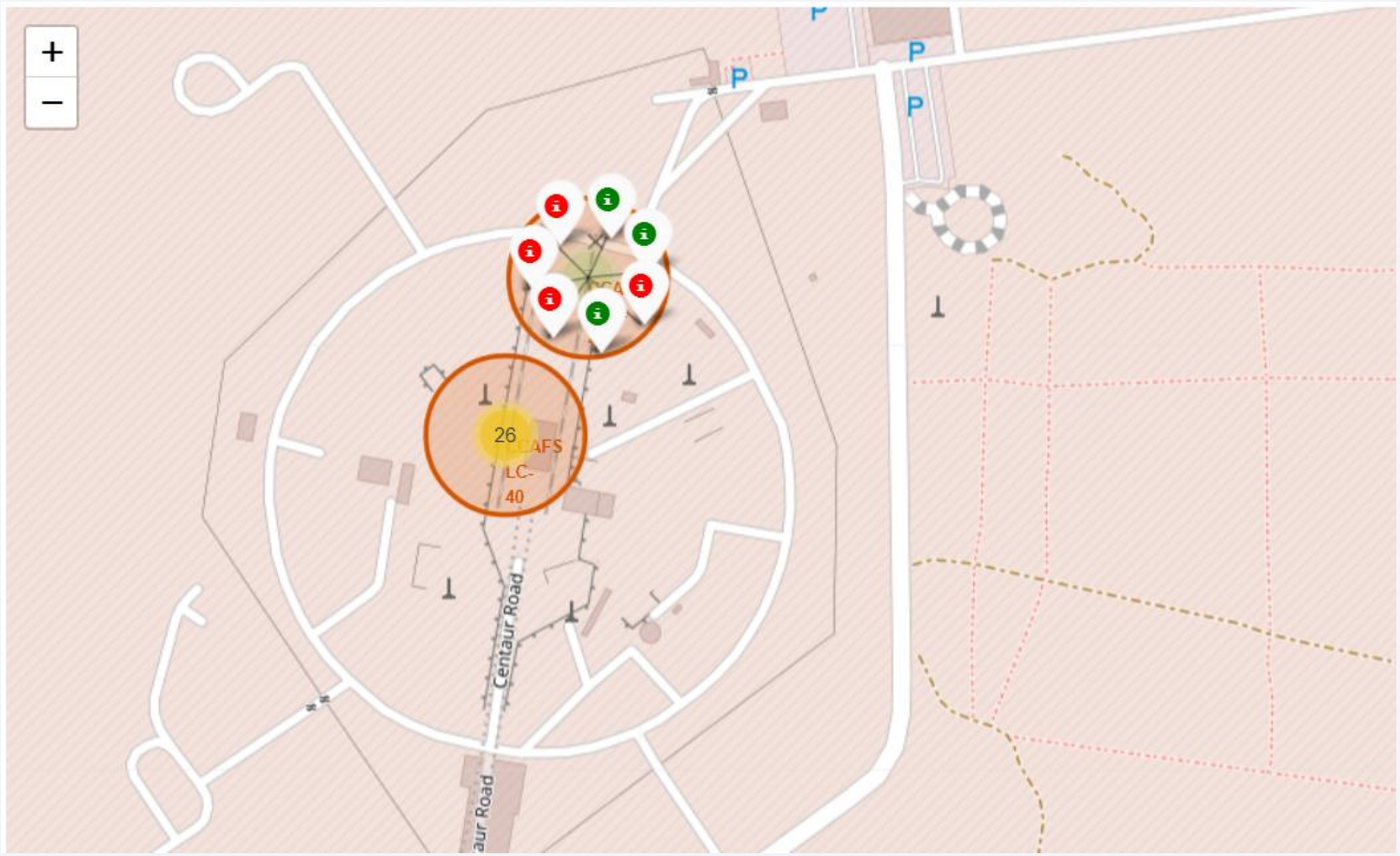
- All the launch sites are located near coasts (West and East)
- There are 3 separate launch sites displayed on the map
- Launch sites are not too far from Equator line to make launch easier thanks to the initial speed of the rocket.



Folium Map of a color-labeled launch outcomes

Folium map of a color-labeled launch outcomes on the map.

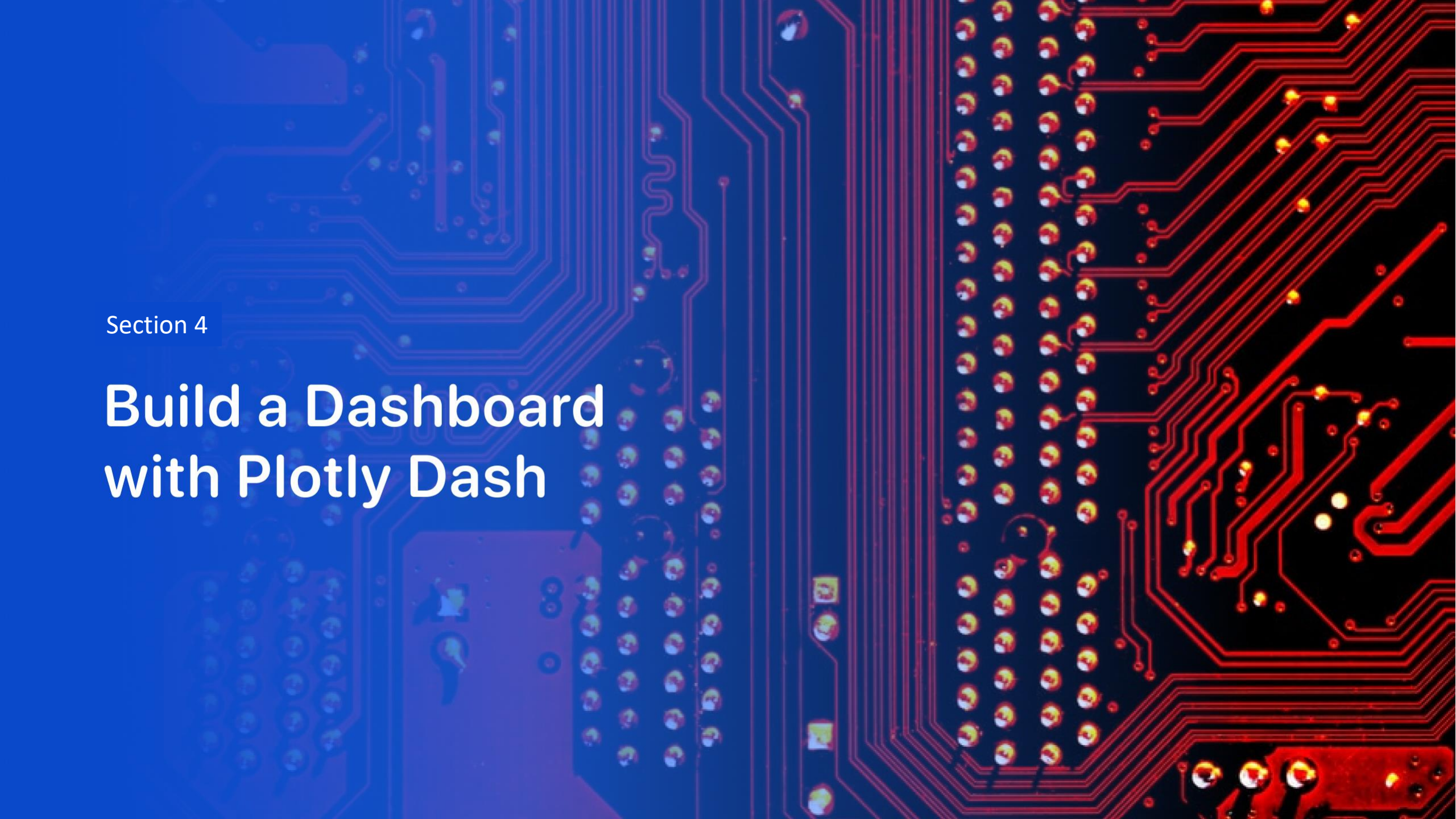
- More insights on the clusters of launch sites success/fail rate.
- CCAFS LC have more launch than the others.



Folium Map of a launch site proximities with a coastline

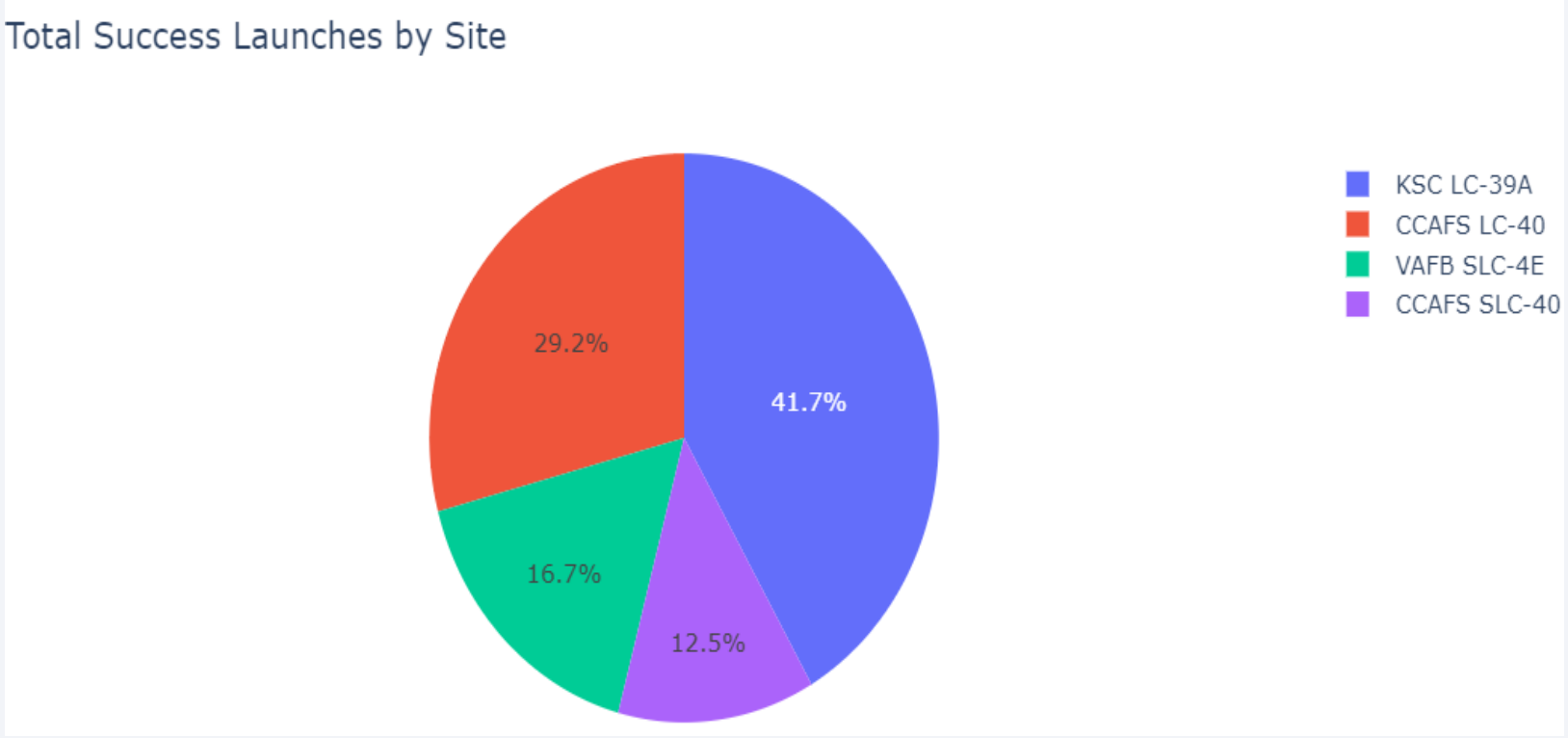
Folium map of a selected launch proximities with a coastline and a distance calculated and displayed.

- Launch sites are far from cities but close to coastline, highway and railway.


Section 4
Build a Dashboard with Plotly Dash

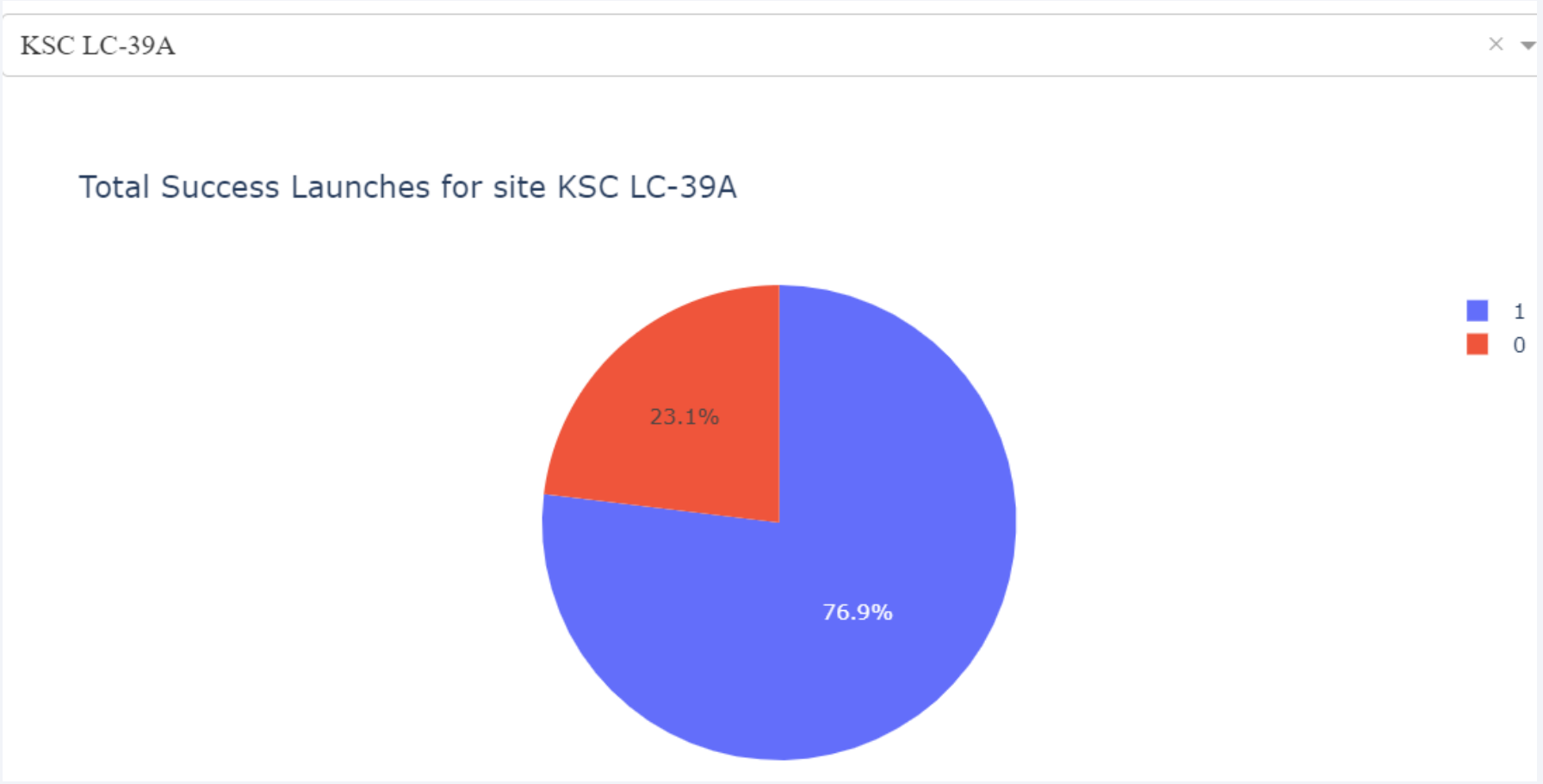
Dashboard Piechart of Launch success count for all sites

- KSC LC launch site have the higher success (almost 50%)



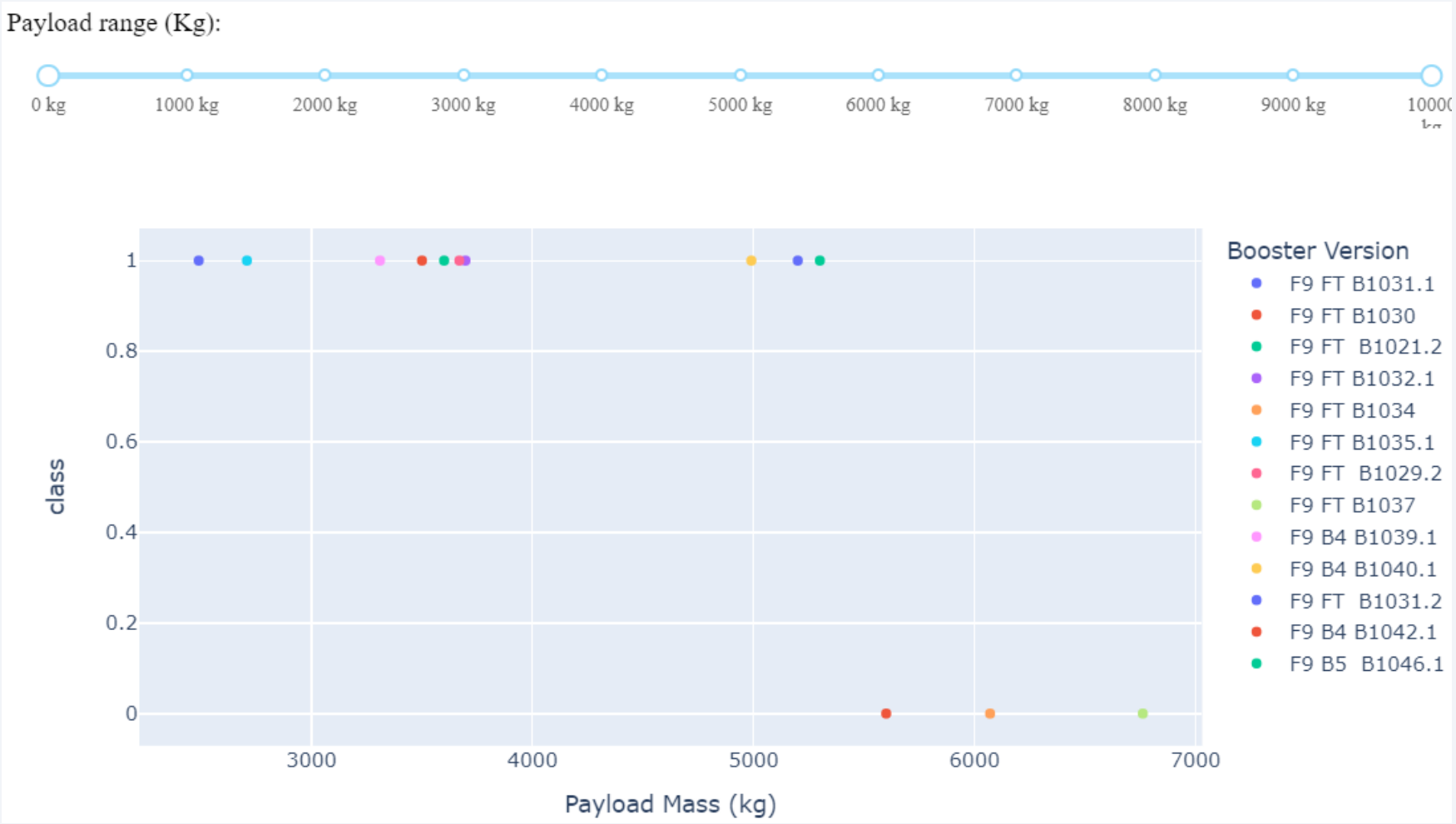
Dashboard Piechart for the launch site with highest launch success ratio

- KSC is the launch site with the highest success ratio
- KSC launch site have almost 80% of success rate and 20% of failure rate.



Dashboard Scatter plot of Payload vs Launch Outcome with slider

- Variable payload allow us to see the sites that succeed/failed for each booster versions.
- If we increase the payload, lowest success rate (greater than 5500).
- The payload range(s) with the highest launch success rate is between 0 and 5500.

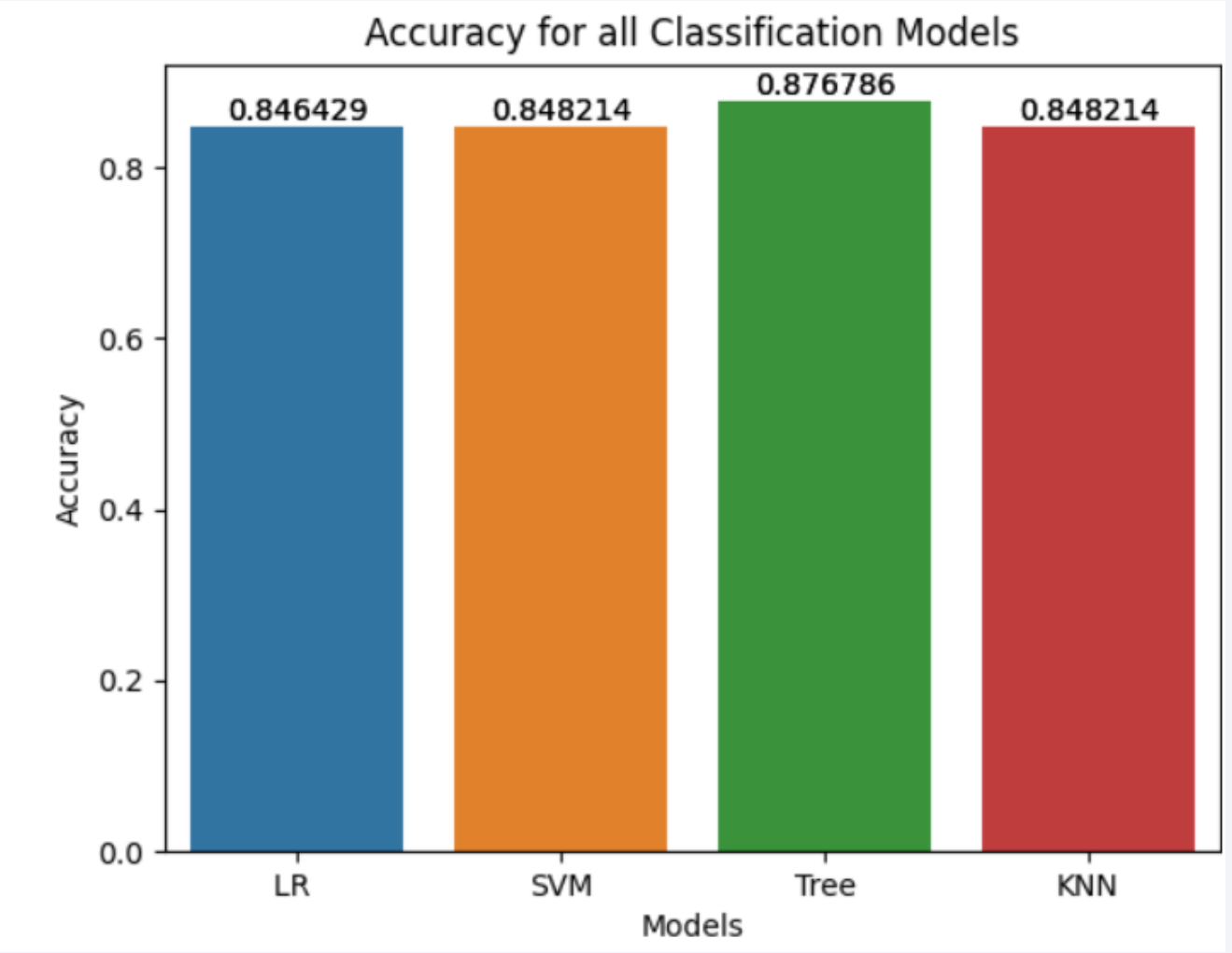


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Bar chart of accuracy for all models.
- The highest accuracy for best parameters is Tree Model with ~88%



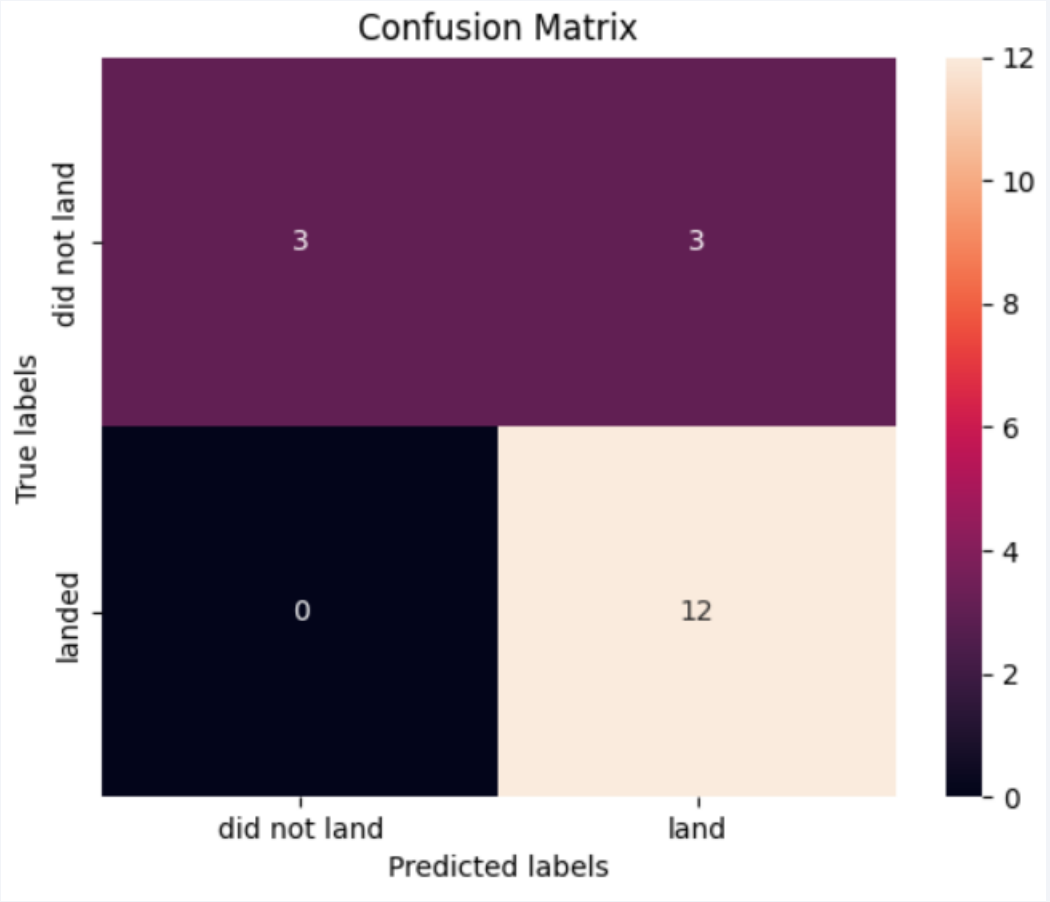
Confusion Matrix of tree_cv

Confusion matrix of the best performing model with an explanation

- The model distinguish between the different classes but failed with 3 labels (False Positive).
- Tree model have the best params ~ 88% on accuracy.

```
print("tuned hpyerparameters :(best param
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)
5, 'splitter': 'random'}
accuracy : 0.8767857142857143
```



Conclusions

- Using EDA in SpaceX data and other rocket companies can be the best way to reduce the cost of launches.
- Perform interactive visual analytics (Folium and Dash) is interesting for seeing insights related to the launch sites' location using folium, in a very interactive way.
- EDA + Interactive Visual Analytics make easier to determine the price of each launch by gathering information about Space X.
- Perform predictive analysis using classification models to determine if SpaceX will reuse the first stage by determining if the first stage will land successfully.
- Thanks to all this methodology, SpaceX and other rocket companies can be able to see the best way to reduce the cost of launches and grow.

Github Link

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!