IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

NITIN BHOYATE
21.09.2024

Outline

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

Executive Summary

- **In this Project we used Following Methodologies**
  - Data collection using API and Web Scrapping
  - Data wrangling to review data attributes
  - EDA with SQL, Pandas and Matplotlib
  - Interactive Visual Analytics
    - Building an interactive map with Folium
    - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- **Results Achieved**
  - Data Collected Using APIs and Web Scrapping
  - Exploratory data analysis results
  - Interactive analytics Done and Dashboard created
  - Machine Learning Prediction results

Introduction

- Project background and context

  We predicted whether the Falcon 9's first stage will land successfully. SpaceX offers Falcon 9 launches at $62 million, much cheaper than competitors charging $165 million, largely due to reusable first stages. Predicting successful landings helps estimate launch costs, useful for other companies bidding against SpaceX.

- Problems you want to find answers
  - Predict if the Falcon 9 first stage will land successfully or Not?
  - What factors influence a rocket's landing success?
  - How do different variables impact the success rate?
  - What conditions must SpaceX meet for optimal landing success?
  - Launch Costs.

Section 1

Methodology

Methodology

Executive Summary

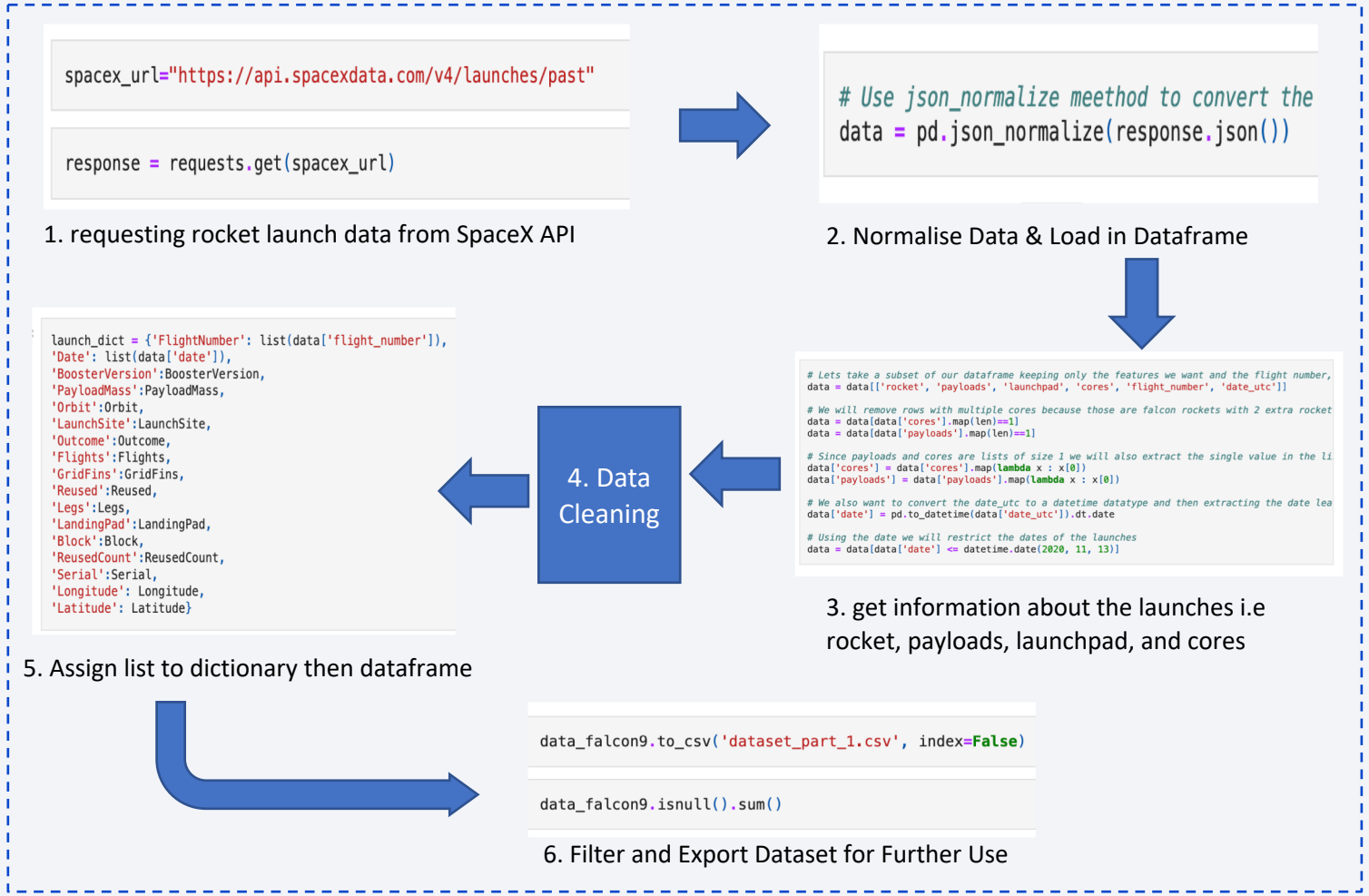
- Data collection methodology:
  - Using SpaceX Api & Web Scraping of Wikipedia Page
- Perform data wrangling
  - Used One Hot Encoding
- Performed exploratory data analysis (EDA) using visualization and SQL Used Scatter Graphs and Bar Graphs
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Used Hyperparameter for SVM, Classification Trees, and Logistic Regression

Data Collection

- We used Two Methods for Data collection for This Project
  - Using SpaceX Rest API
  - Web Scrapping of Wikipedia Page
- Following Endpoints were used to Collect Data
  - Spacex API : https://api.spacexdata.com/v4/
  - Wikipedia : https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- Detailed Data collection process & flowcharts are Explained in Next Slides

Data Collection – SpaceX API

- We Done data collection with SpaceX REST calls
- We collected Data for Rockets, Launchpads, Payloads, Cores and past Launch Data
- After Cleaning of Data We constructed data and Combined Required Columns into Dictionary for Further Processing
- GitHub URL of the completed SpaceX API calls notebook - https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/1-1_jupyter-labs-spacex-data-collection-api.ipynb



Flow Chart of API call

Data Collection - Scraping

- We used Wikipedia page titled List of Falcon 9 and Falcon Heavy launches for Web Scrapping
- We used BeautifulSoup html parser for HTML parsing
- GitHub URL of the completed web scraping notebook : https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/1-2_jupyter-labs-webscraping.ipynb

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

1. HTTP GET method to request the Falcon9 Launch HTML page

```
# Use BeautifulSoup() to create a BeautifulSoup
soup = BeautifulSoup(page.text, 'html.parser')
```

2. BeautifulSoup object from the HTML response

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

3. Extract all column/variable names from the HTML table header

```
launch_dict= dict.fromkeys(column_names

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with ea
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

4. data frame by parsing the launch HTML tables

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

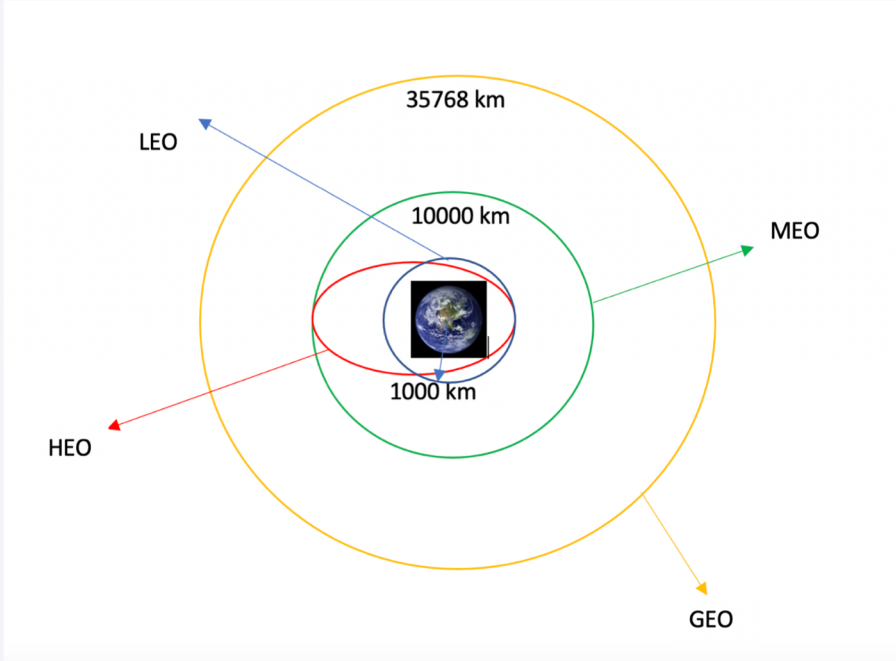
5. fill in the parsed launch record values into launch_dict

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

6. Export Data to CSV for Further Use

Data Wrangling

- In Data wrangling We Performed exploratory Data Analysis and determined Training Labels
- Steps / Flowchart Followed
  - Step 1 : Identify and calculate the percentage of the missing values in each attribute
  - Step 2 : Identify which columns are numerical and categorical
  - Step 3 : Calculate the number of launches on each site
  - Step 4 : Calculate the number and occurrence of each orbit
  - Step 5: Calculate the number and occurrence of mission outcome of the orbits
  - Step 6 : Create a landing outcome label from Outcome column
  - Step 7 : Export Data to a CSV for Further use



GitHub URL of completed data wrangling :
https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/1-3_labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Following Charts were used for EDA with Data Visualization
  - Scatterplot
  - Bar Chart
  - Line chart
- Objective was Exploratory Data Analysis & Preparing Data Feature Engineering
- GitHub URL of completed EDA with data visualization notebook : https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/2-2edadataviz.ipynb

EDA with SQL

- We used various SQL queries to perform EDA with SQL
  - DROP TABLE – Drop if Table Exist
  - CREATE TABLE – Create New Table
  - SELECT DISTINCT – To get Unique Data
  - WHERE condition
  - LIKE , MAX, MEAN, SUM, AVG, MIN operators to get Data
  - We also used SUBQUERIES

- GitHub URL of completed EDA with SQL notebook : https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/2-1-jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We used folium map to Build Interactive Map which Shows Launch Sites over MAP
- We took the Latitude and Longitude Coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.
- We assigned the dataframe launch_outcomes(failures, successes) to classes 0 and 1 with Green and Red markers on the map
- Using Haversine's formula we calculated the distance from the Launch Site to various landmarks
- GitHub URL of completed interactive map with Folium map : https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/3-1lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- We use Plotly express to plots/graphs and interactions and Created dashboard
- The dashboard is built with Flask and Dash web framework.
- Graphs Used
  - Pie Chart showing the total launches by a certain site/all sites
  - Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions
- GitHub URL of completed Plotly Dash lab : https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/3-2spacex_dash_app.py

Predictive Analysis (Classification)

BUILDING MODEL FLOWCHART & STEPS

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Fit our datasets into the GridSearchCV objects and train our dataset.

MODEL EVALUATION

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

MODEL IMPROVEMENT

- Feature Engineering
- Algorithm Tuning

PERFORMANCE ANALYSIS

- Calculate Accuracy Score
- Compare Using Graph

GitHub URL of completed predictive analysis lab :

https://github.com/nitinbhoyate/applied-data-science-capstone/blob/main/4-1SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

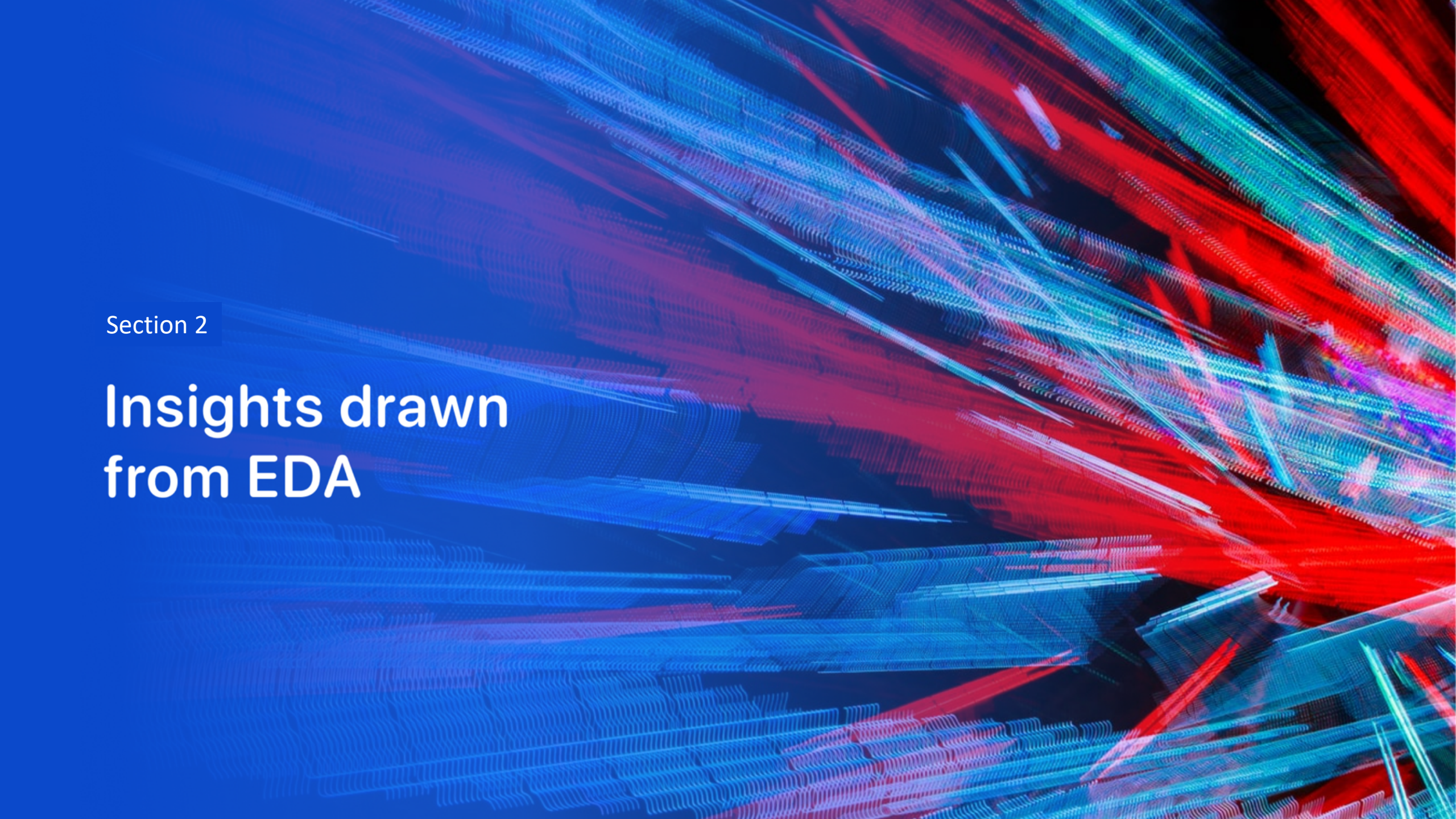
Results

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

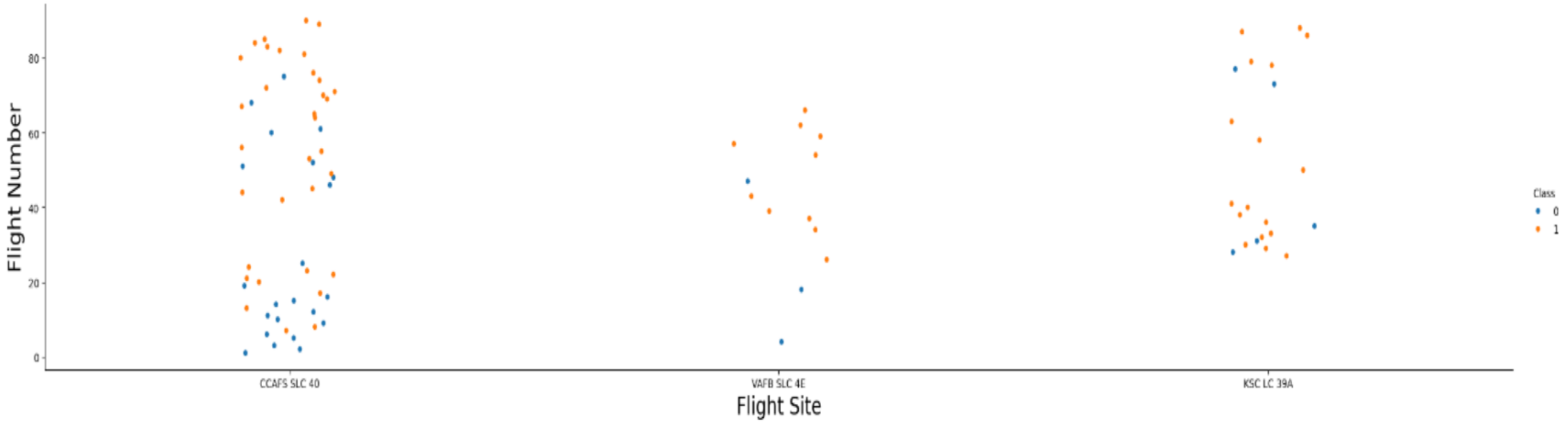
ALL RESULTS ARE SHOWN IN NEXT SLIDES

Section 2

Insights drawn from EDA

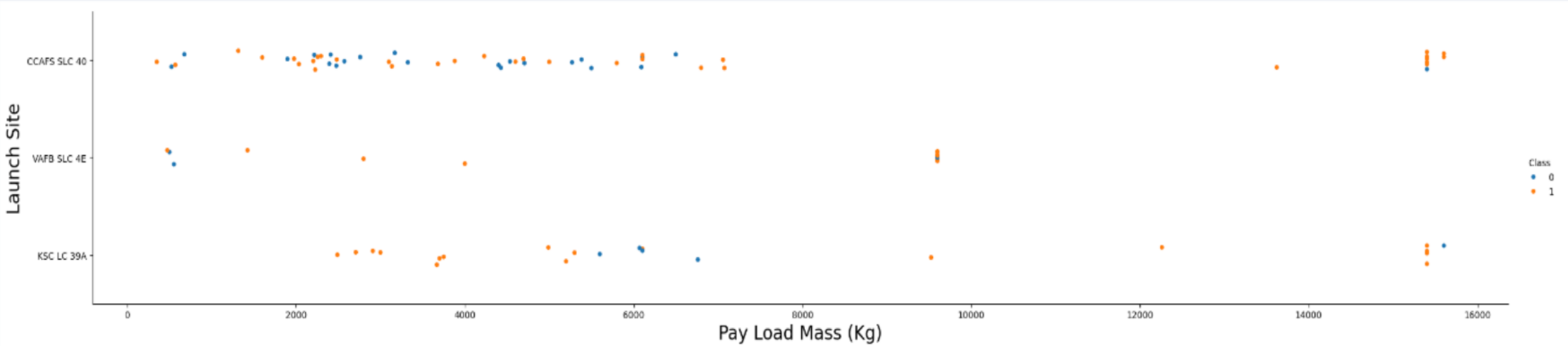


Flight Number vs. Launch Site



The more amount of flights at a launch site the greater the success rate at a launch site.

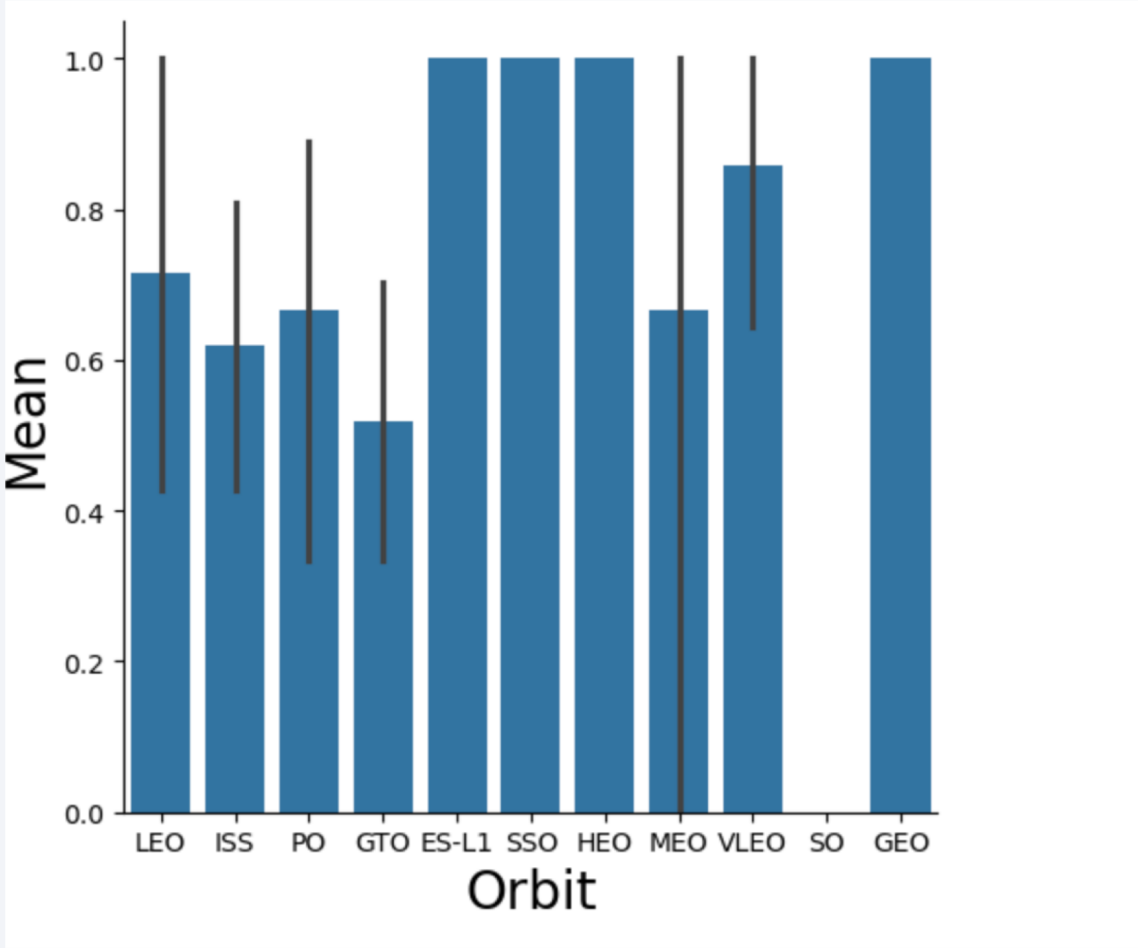
Payload vs. Launch Site



Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

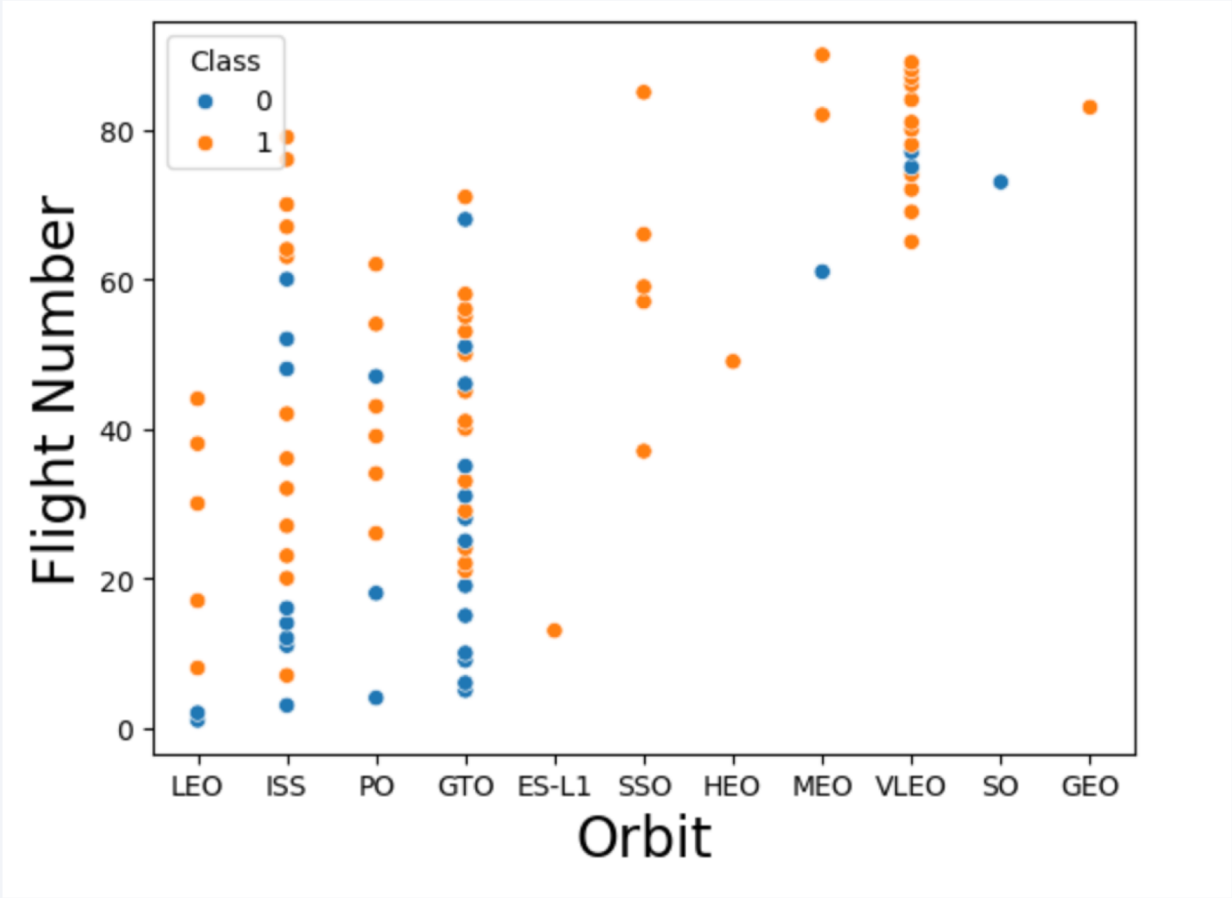
Success Rate vs. Orbit Type

- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate



Flight Number vs. Orbit Type

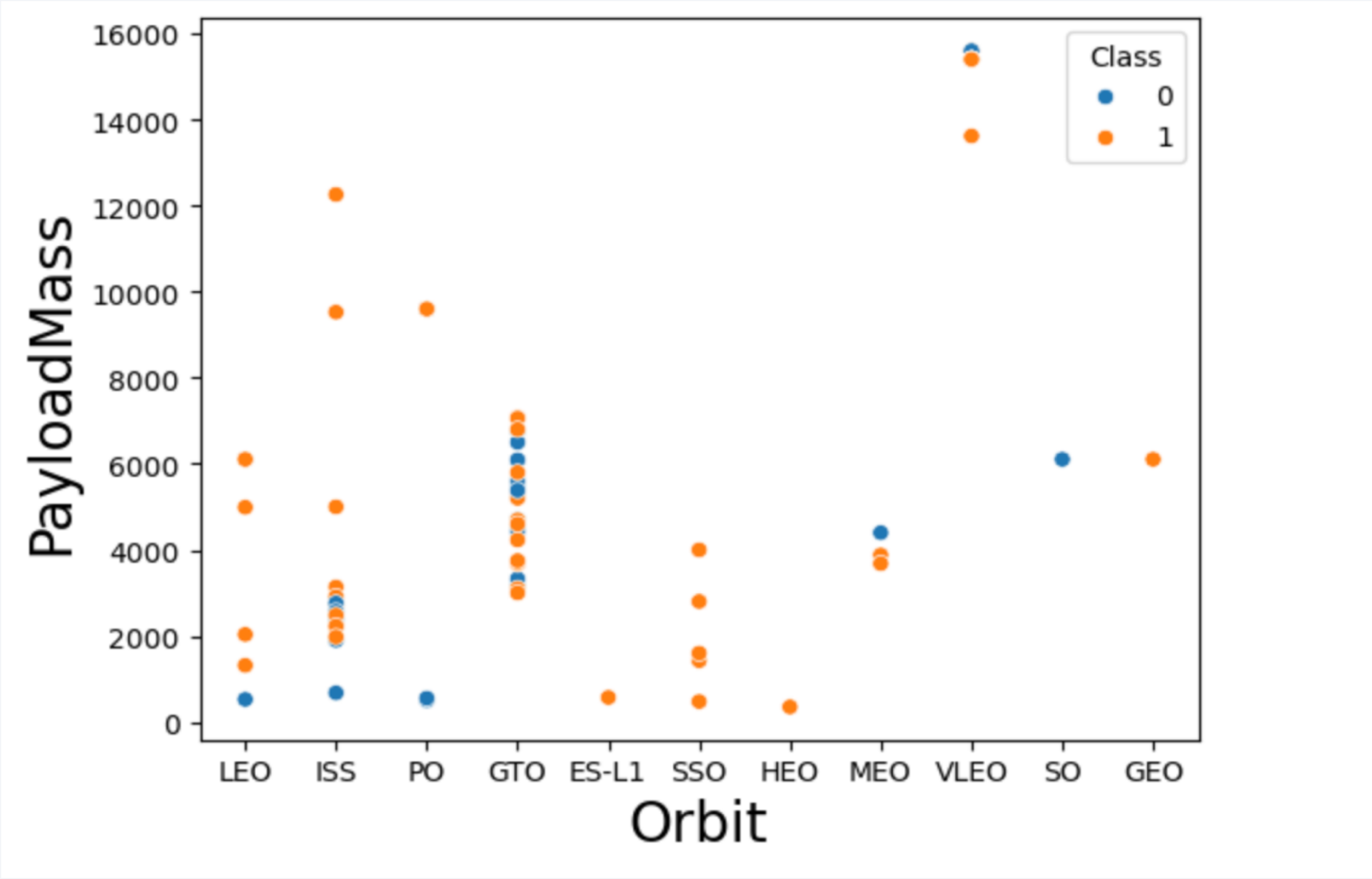
- You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.



Payload vs. Orbit Type

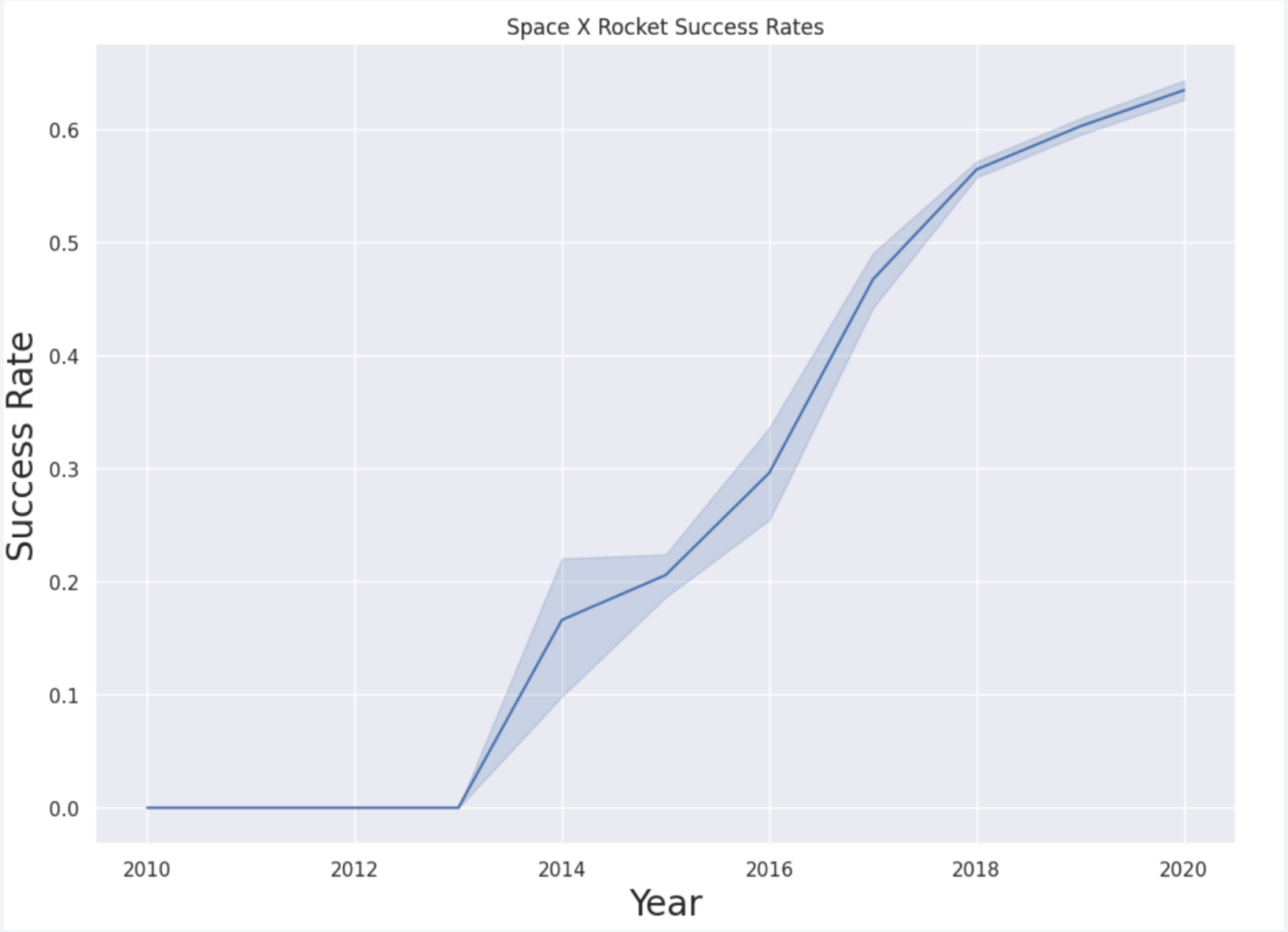
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

- However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.



Launch Success Yearly Trend

- you can observe that the sucess rate since 2013 kept increasing till 2020



All Launch Site Names

- Find the names of the unique launch sites
- We are Using DISTINCT Operator to Display the names of the unique launch sites in the space mission
- SQL Query : select DISTINCT Launch_Site from SPACEXTABLE

```
%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

Unique launch sites
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`
- We Used LIKE clause and LIMIT 5 to Display 5 records where launch sites begin with the string 'CCA'
- Query used :

select * from SPACEXTABLE WHERE Launch_Site LIKE "CCA%" LIMIT 5

```
%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
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
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
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- We Used SUM() Arithmatic Function to 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

- Calculate the average payload mass carried by booster version F9 v1.1
- We used AVG Function Display average payload mass carried by booster version F9 v1.1

```
%sql select AVG(PAYLOAD_MASS__KG_) from SPACEXTABLE 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
- We used MIN function to get minimum Date and WHERE Clause get succesful landing outcome in ground pad was achieved.

```
%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

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

- We Used AND operator to add multiple conditions
- SQL Query :

```
select Booster_Version from SPACEXTABLE
WHERE Landing_Outcome="Success (ground pad)"
AND PAYLOAD_MASS__KG_ >4000
AND PAYLOAD_MASS__KG_ < 6000
```

Booster_Version
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- User GROUP BY Clause and COUNT to List the total number of successful and failure mission outcomes

```
%sql select Mission_Outcome, COUNT(*) AS total from SPACEXTABLE GROUP BY Mission_Outcome
```

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

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

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- We Used SUBQUERY to List the names of the booster_versions which have carried the maximum payload mass.
- SQL QUERY

```
SELECT BOOSTER_VERSION FROM SPACEXTBL
WHERE
PAYLOAD_MASS__KG_ = (SELECT
MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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
- We used Substr() date function
- SQLLite does not support monthnames. So need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

SQL Query

```
%sql SELECT substr(Date,6,2) as month, DATE,BOOSTER_VERSION, LAUNCH_SITE, Landing_Outcome FROM SPACEXTBL where Landing_Outcome = 'Failure (drone ship)' and substr(Date,0,5)='2015';
```

month	Date	Booster_Version	Launch_Site	Landing_Outcome
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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
- We Used DESC to get counts in Descending Order
- SQL QUERY

```
SELECT Landing_Outcome, count(*) as count_outcomes FROM SPACEXTBL WHERE DATE between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by count_outcomes DESC;
```

Landing_Outcome	count_outcomes
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

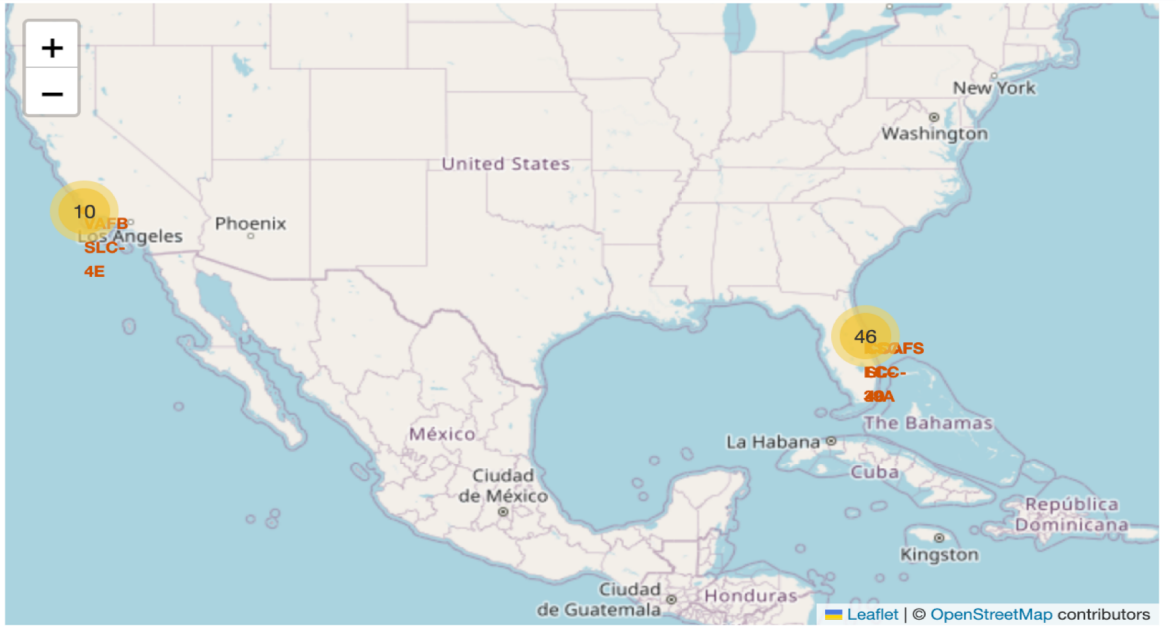
Section 3

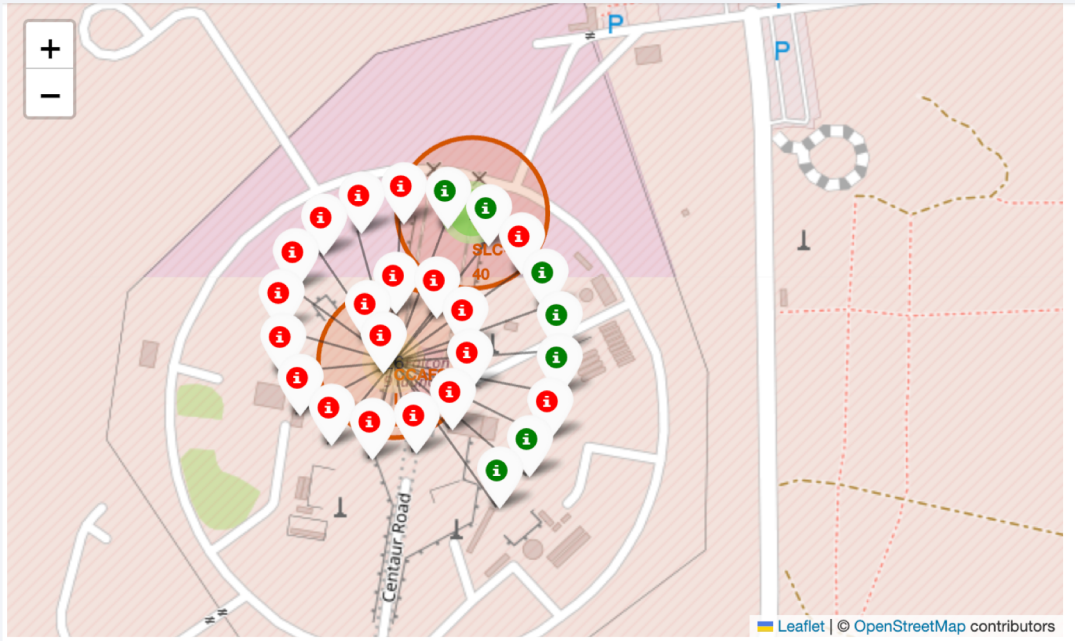
Launch Sites Proximities Analysis

All Launch Site - Locations

SpaceX launch sites are in Florida and California (USA)

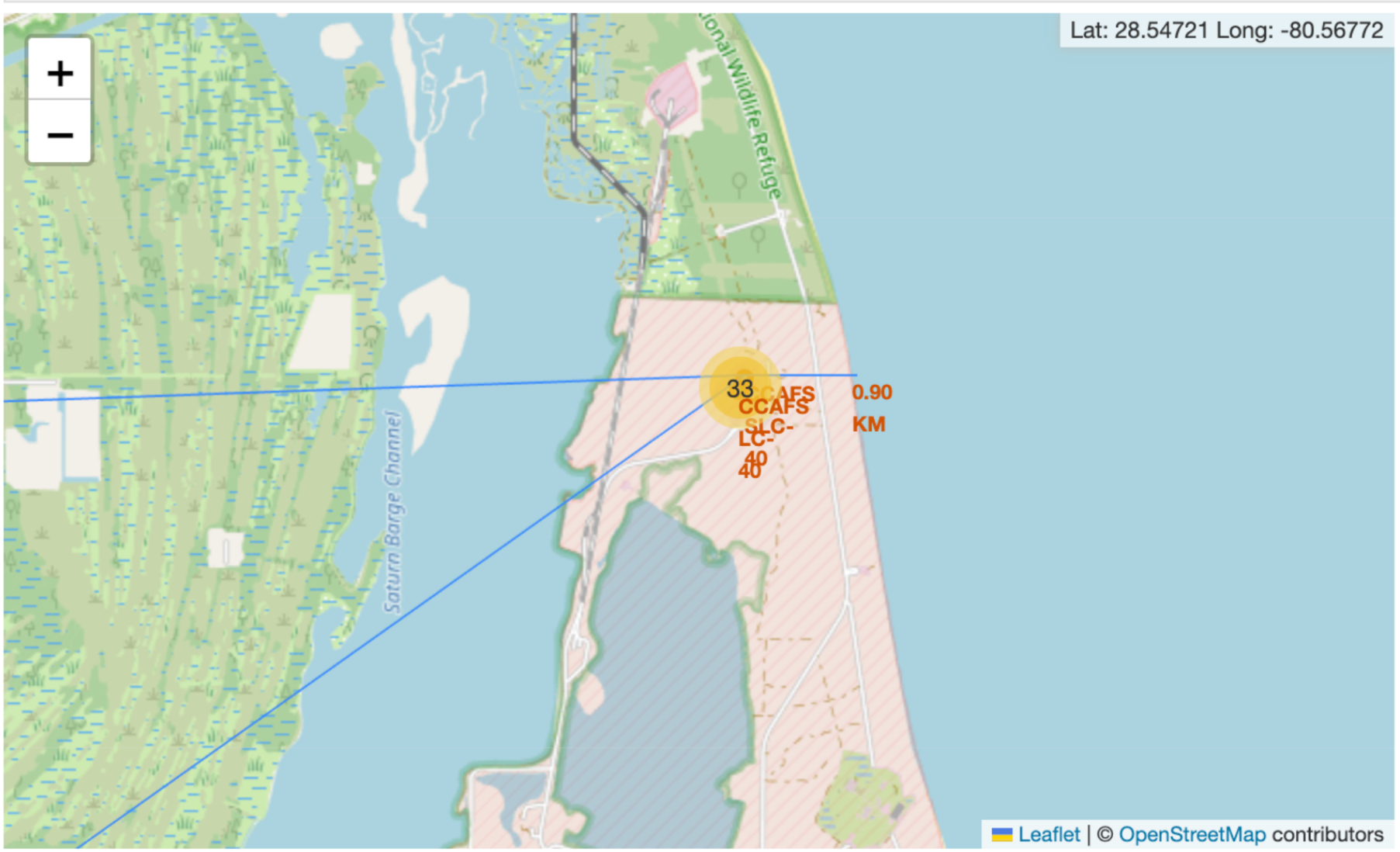
All Launch Site – Locations With Markers



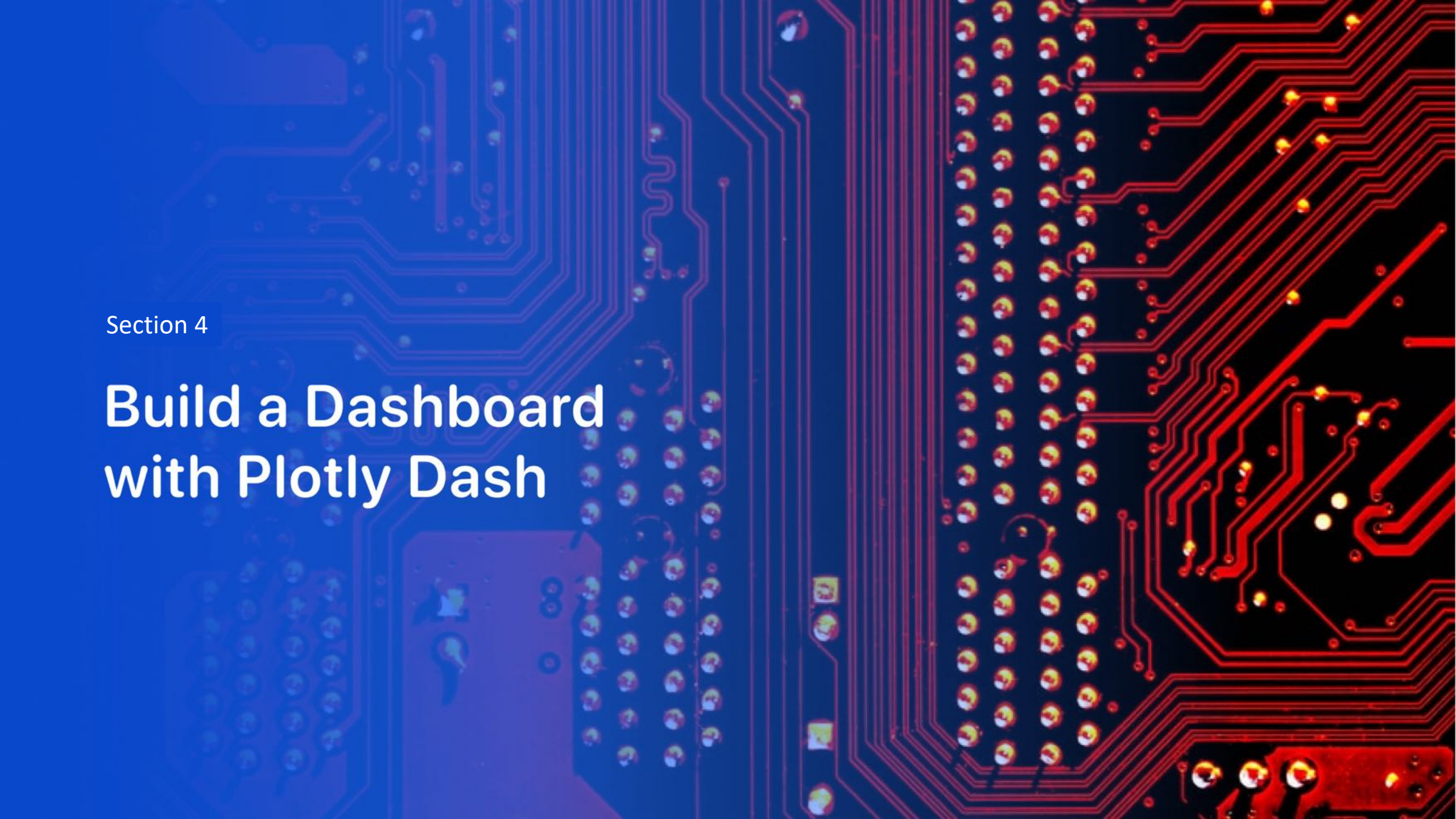


Successful launches are in Green and Failed Launches are in Red

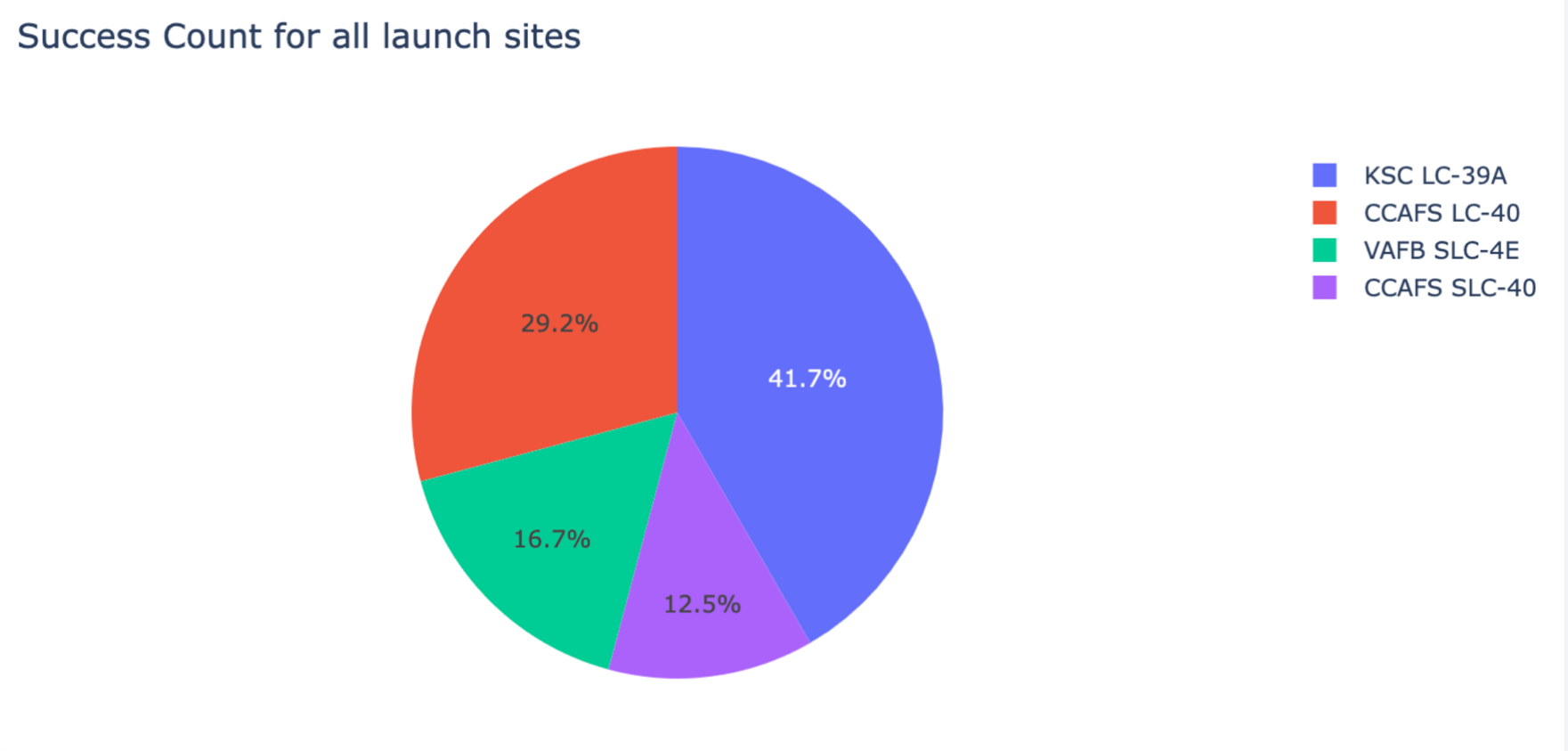
CCAFS SLC 40 Nearby Locations



- CCAFS SLC 40 us Near to Coastal Line only 0.90 KM

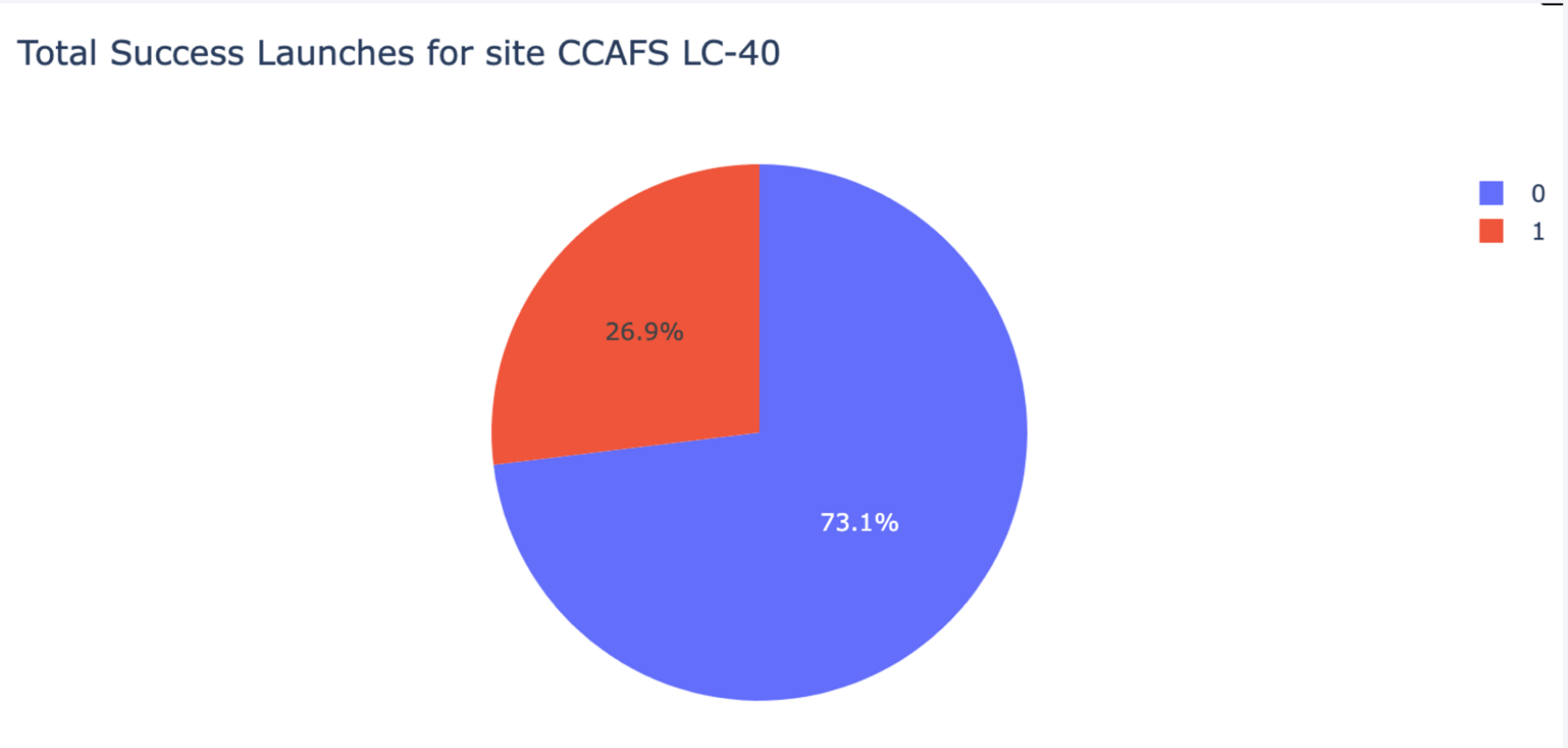

Section 4
Build a Dashboard with Plotly Dash

SpaceX Launch Records – Success Count for All Sites



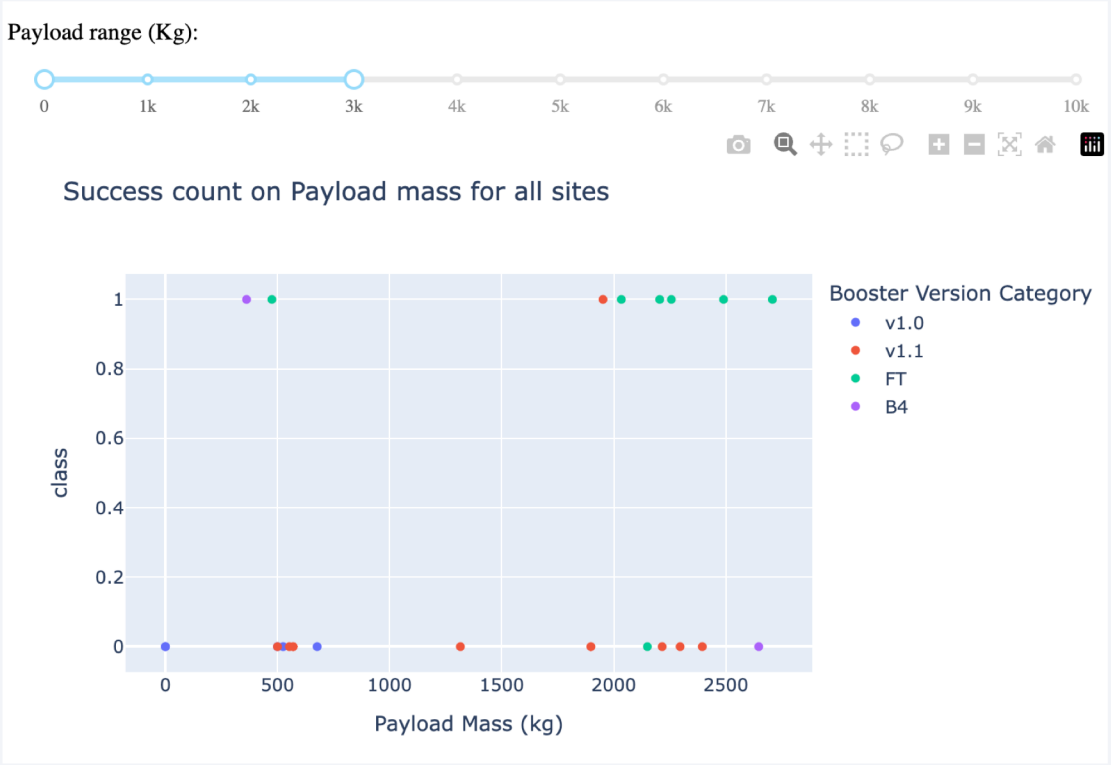
KSC LC 39A has Highest Success Ratio and VAFB SLC-4E has Lowest Success Ratio

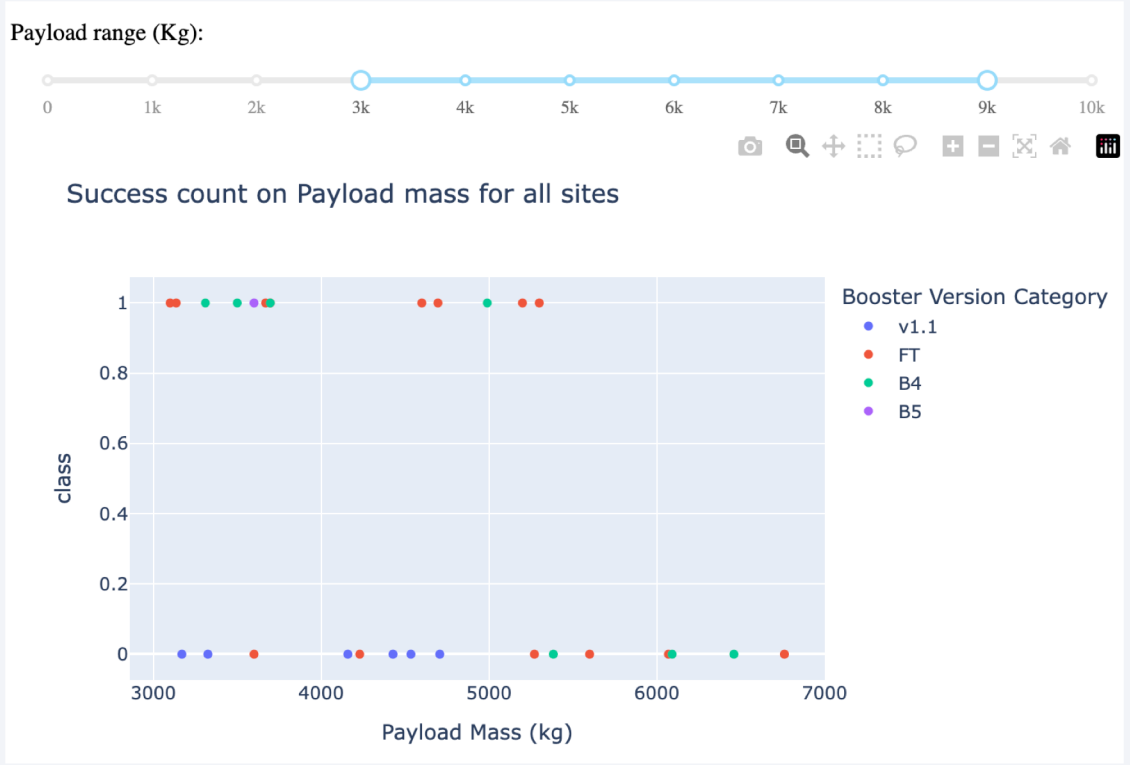
SpaceX Launch Records – Highest Launch Success Ratio



CCAFS LC40 Launch site has 73.1% success ratio

Payload vs. Launch Outcome - all sites



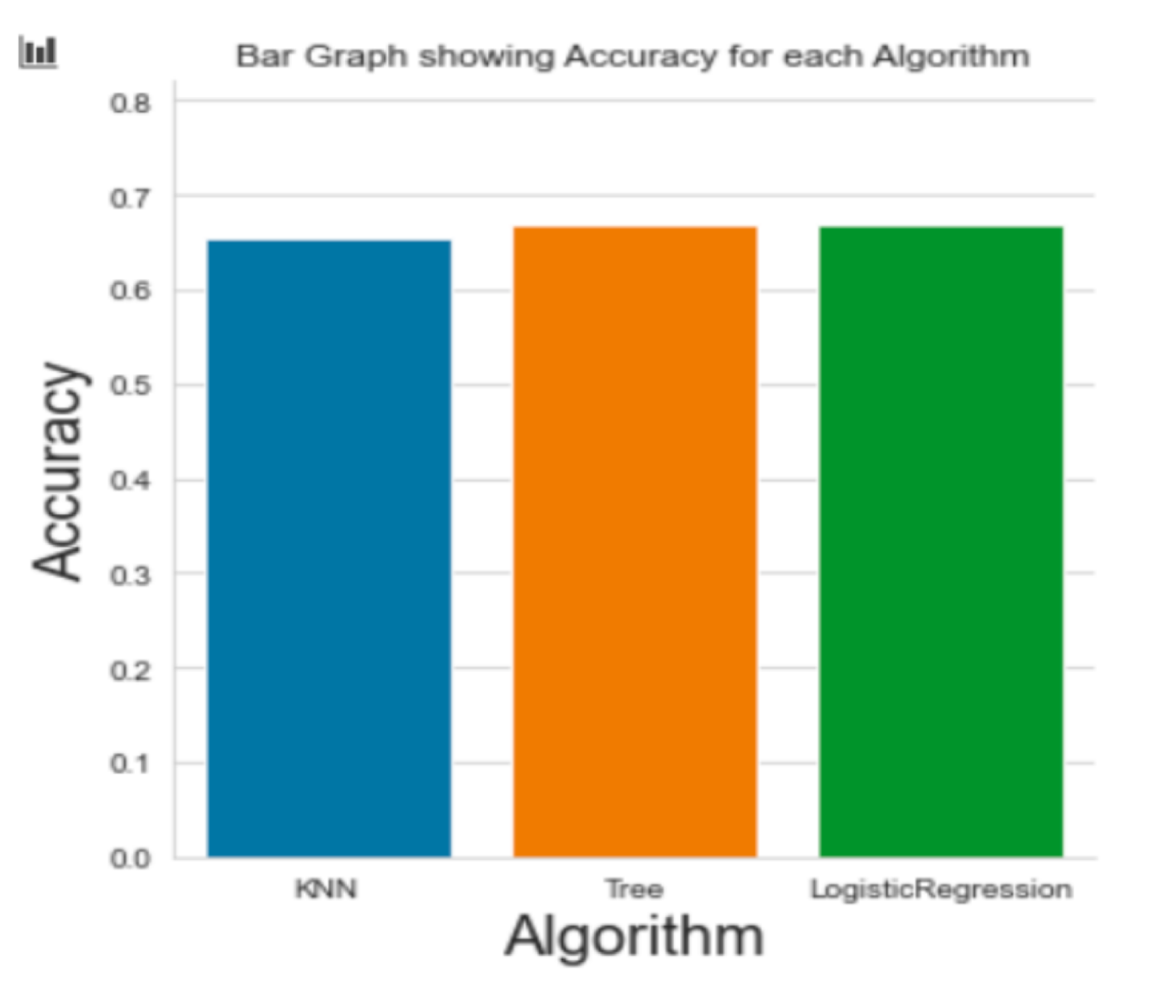


- We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

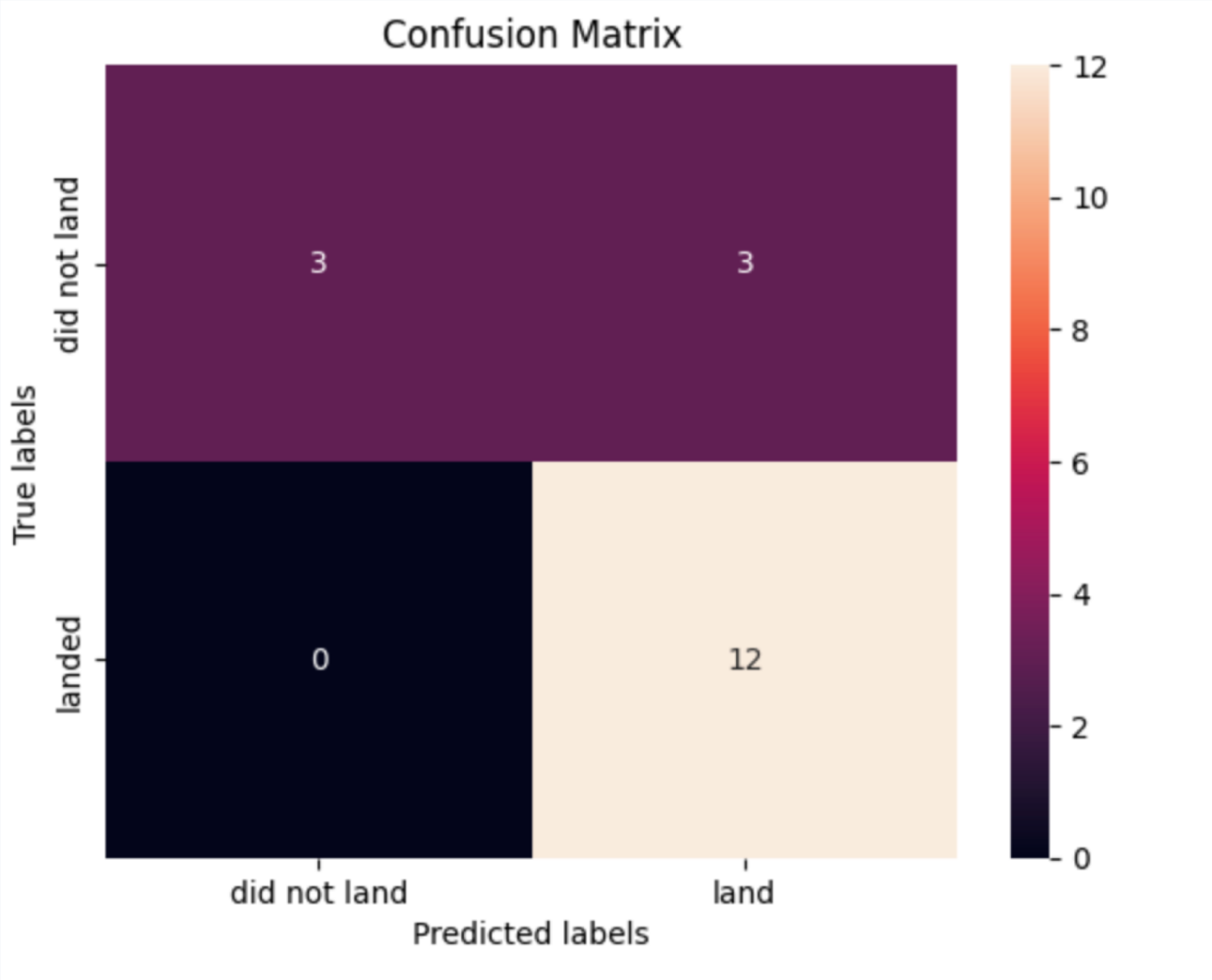
Predictive Analysis (Classification)

Classification Accuracy



- Accuracy is extremely close to All Models
- After selecting the best hyperparameters for the decision tree classifier using the validation data, we achieved 83.33% accuracy on the test data.

Confusion Matrix



- Examining the confusion matrix, we see that logistic regression can distinguish between the different classes.
- We see that the problem is false positives.

Conclusions

- All Model test set accuracy is 0.8333333333333334
- After Tuning hyperparameter accuracy is 0.8732142857142857
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- We can see that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Appendix

- All Source Codes - <link>
- Data Set Links –
  - Spacex API : https://api.spacexdata.com/v4/
  - Wikipedia : https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- Matplotlib - https://matplotlib.org/
- Seaborn - https://seaborn.pydata.org/
- SciKit learn - https://scikit-learn.org/stable/
- Plotly - https://plotly.com/
- Folium - https://python-visualization.github.io/folium/latest/

Thank you!