Winning Space Race with Data Science

Name: ABDULGHANI
Date: 7/15/2022
Github Repo: >Here<

Outline

i. Executive Summary

ii. Introduction

iii. Methodology

iv. Results

v. Conclusion

i. Executive Summary

- Collecting data from SpaceX API with Python, creating tables and columns for the successful landing, and exploring and visualize the data using SQL sci-learn and Pandas, and perform statistic operations on the data, and use Machine learning model to get the best predictions results.
- We will use Machine Learning Models for predicting better result like Logistic regression, Support Victor Machine, Destino Tree and K Nerats Nigbor, all these models able to predicts successful landing with accuracy up to 85%, if we add more data we can get more accurate results.

ii. Introduction

- **SpaceX** is an American space craft manufacture, space launch provider, and a satellite communications, corporation headquartered in Hawthorne, California, SpaceX was founded in 2002 by Elon Musk.
- SpaceX has best pricing ($62 Millions VS. $165 Millions)
- SpacY is another competitive company against SpaceX.

- The Problem: SpaceY tasked to train a Machine Learning Model to Predict the Accuracy Percentage of Successful landing.

Section 1

Methodology

iii. Methodology

Executive Summary

- Data collection methodology:
  - Collecting Data From SpaceX API, and Wikipedia website using Python
- Perform data wrangling
  - Trying to predicts the landing was successful or not
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Run models using GridSearchCV

Data Collection

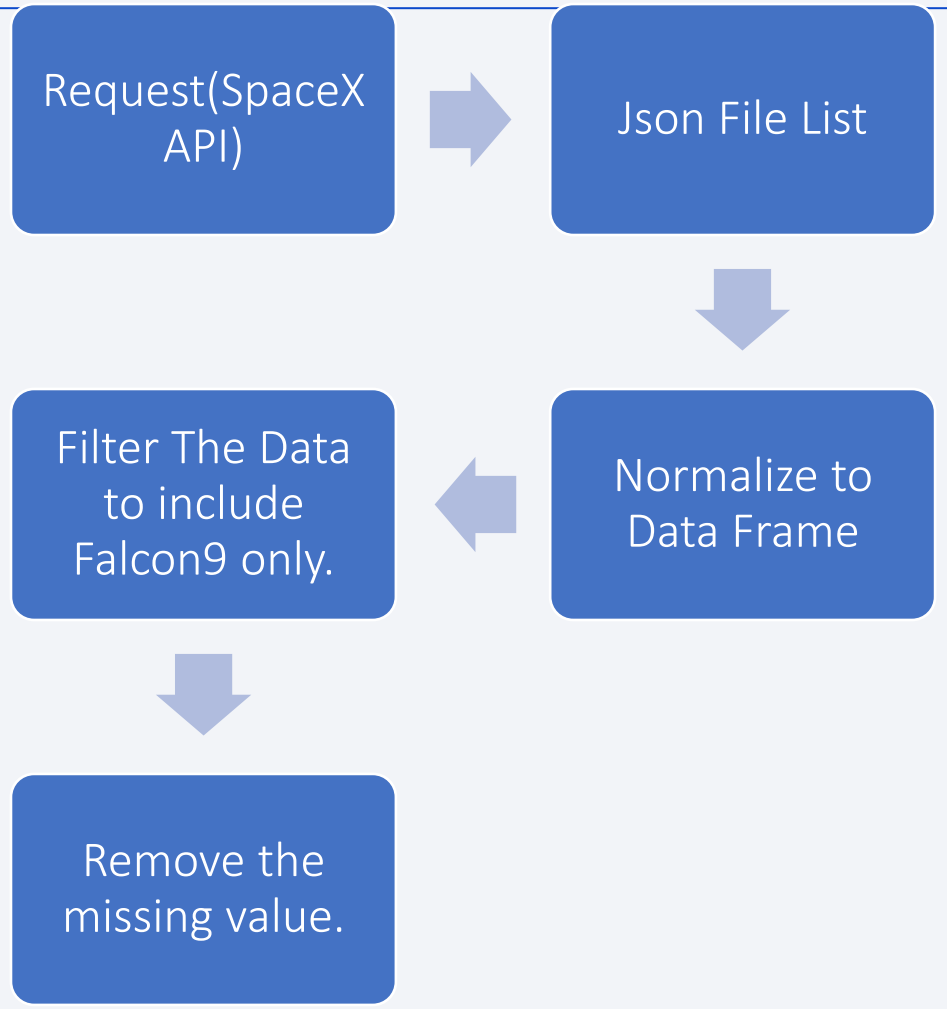
- We have to main method to collection the data

1. SpaceX API: Provide us with a different data consisted of rows and columns (FlightNumber, BoosterVersion, Date, PayloadMass, Orbit, LaunchSite, Outcome, Flight, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serail, Longitude, Latitude).
2. Wikipedia (web scraping): we can extract the data from wikipedia using Beatifulsoup Python package which contained (Flight No, Launch Site, Payload, PayloadMass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time).

Data Collection – SpaceX API

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

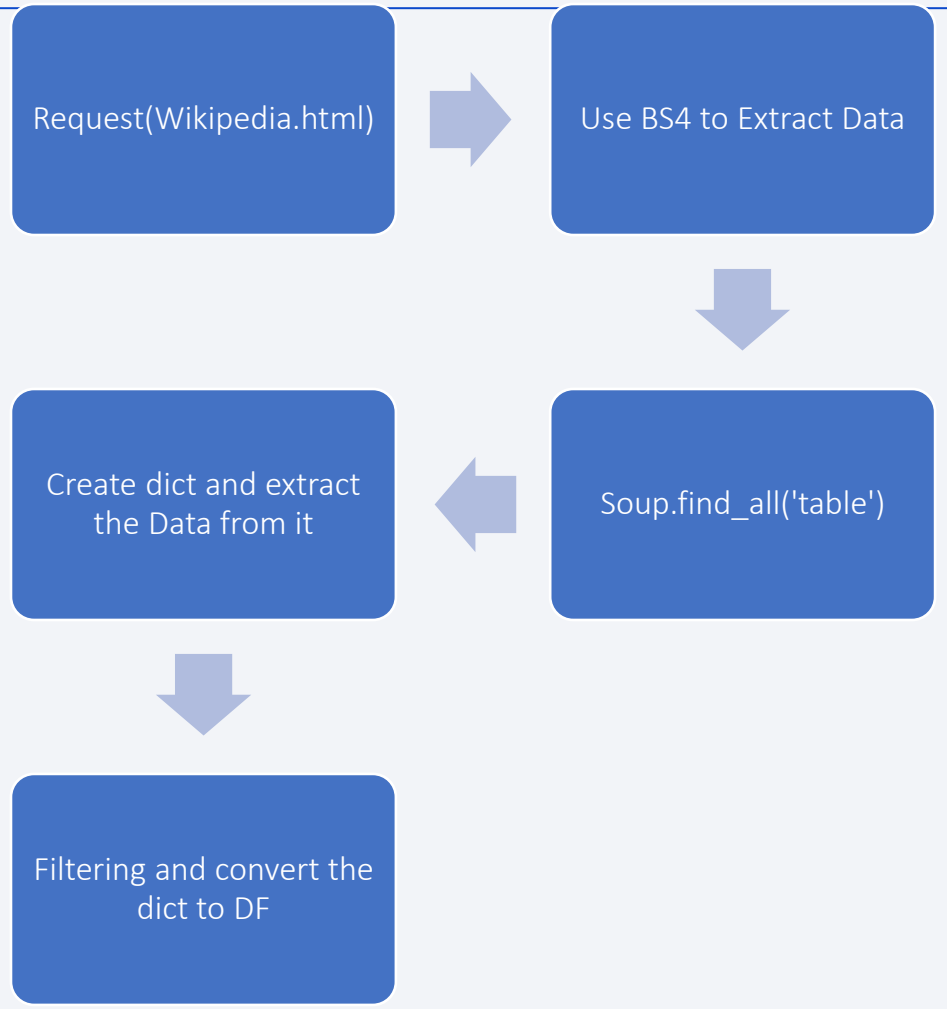
- GitHub: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_1_Introduction/1-jupter-lab-Falcon9.ipynb



Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- GitHub URL: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_1_Introduction/jupyter-labs-webscraping.ipynb



Data Wrangling

- We have two type of labels successful landing = 1, and unsuccessful=0, in the column Mission outcome.
- We add another two columns 'class' equal 1 if the mission outcome is successful, and 0 if it is unsuccessful, and another column color depend on the class column is the class =1 the color= Green, else 0 the color= Red.
- GitHub URL: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_1_Introduction/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- We plot Flight Number Vs. PayloadMass, Flight Number Vs. Launch Site, Orbit Vs Success Rate, Flight Number Vs. Orbit, Payload Vs. Orbit, and Success Yearly trends.
- We used scatter plots, line charts, and bar charts to see the relationship between variables, if there is any connection between any two variables they could be useful for building the model.
- GitHub URL: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_2_Exploratory_Data_Analysis_Using_SQL/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- We used SQL to analyze our data, by including the magic variable % we can use SQL queries with python.
- We review the dataset with select and other SQL queries.
- We could filter our data using where query and other useful SQL queries.
- GitHub URL: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_2_Exploratory_Data_Analysis_Using_SQL/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

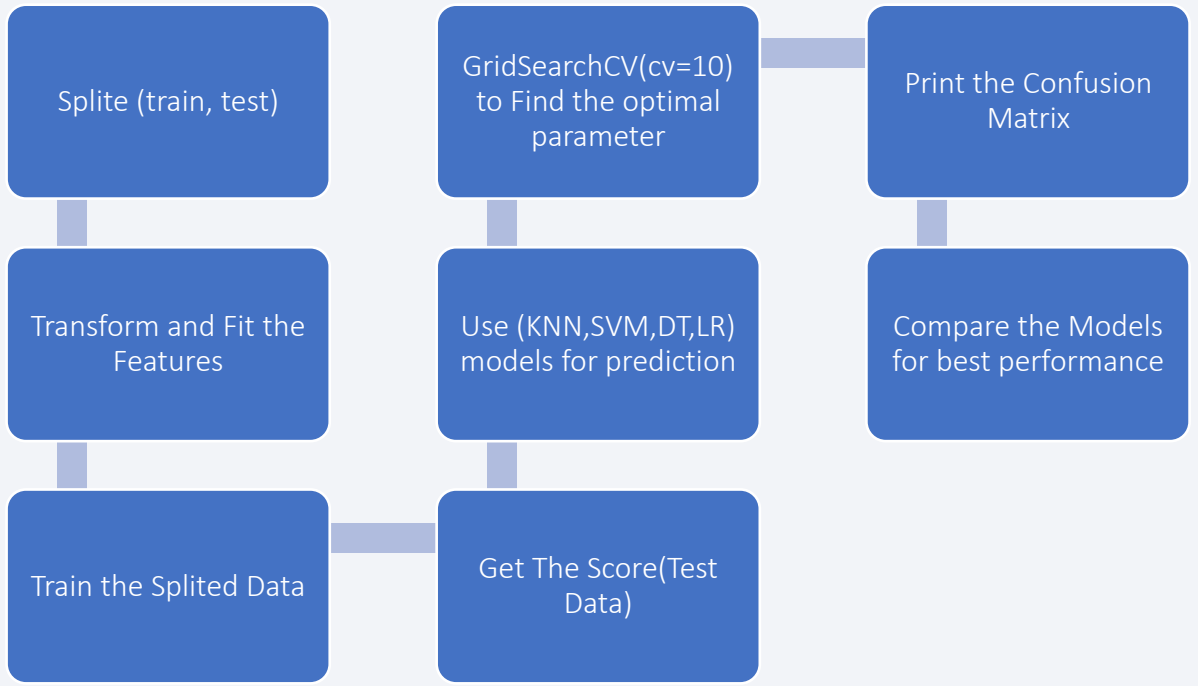
- Folium is a Python package that makes it easy for you to display any map for any country or region, and we use it to display the launching site on the map by using the latitude and longitude of the lunching site
- By reviewing the launching site on the map we will able to see where is the location of each site and how long the distance between each of them.
- GitHub URL: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_3_Interactive_Visual_Analytics_and_Dashboards/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

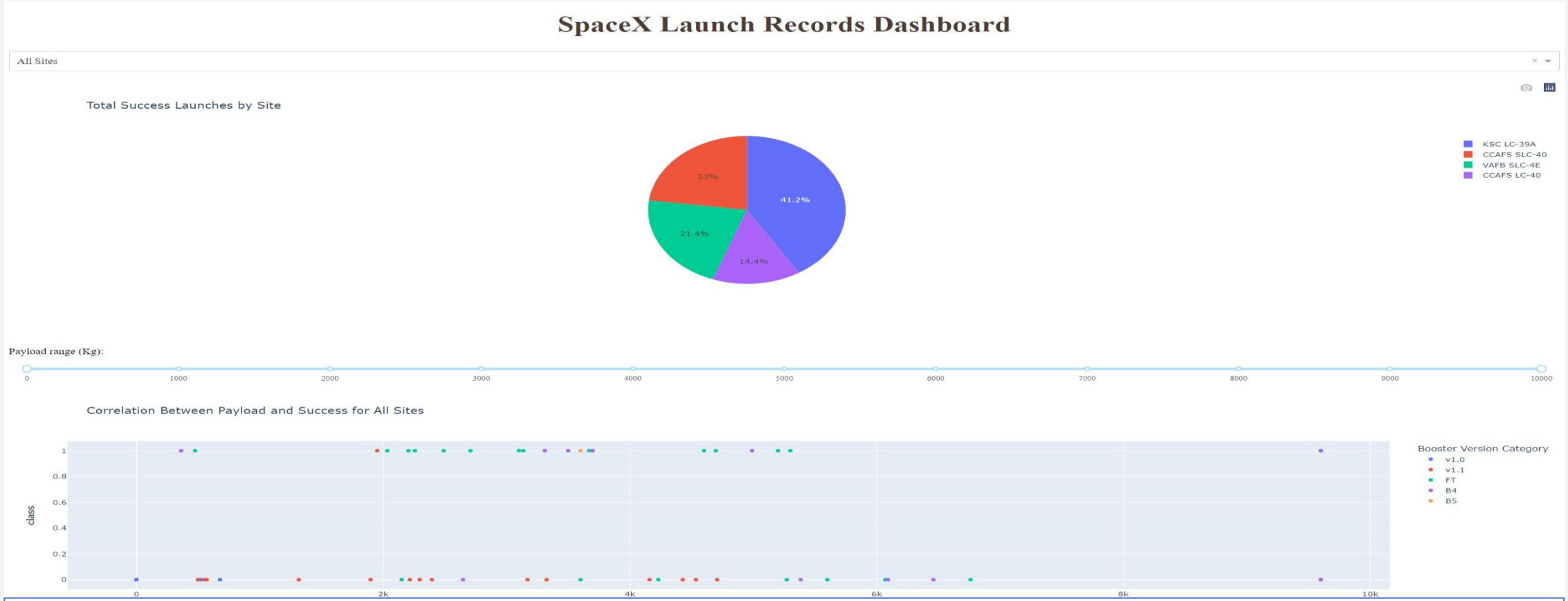
- Plotly Dash is another Python tool that let us display our data on a dashboard but this kind of live dashboard means you can select the data of any year from the dropdown menu and will show you the result directly, and you can specify the payload Mass KG for any range like between 5000-13000 KG it will show you the data related with the specific value of KG and the specific year, this is will help us to see how many success lunching for every year.
- GitHub URL of your completed Plotly Dash lab: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_3_Interactive_Visual_Analytics_and_Dashboards/spacex_dash_app.py

Predictive Analysis (Classification)

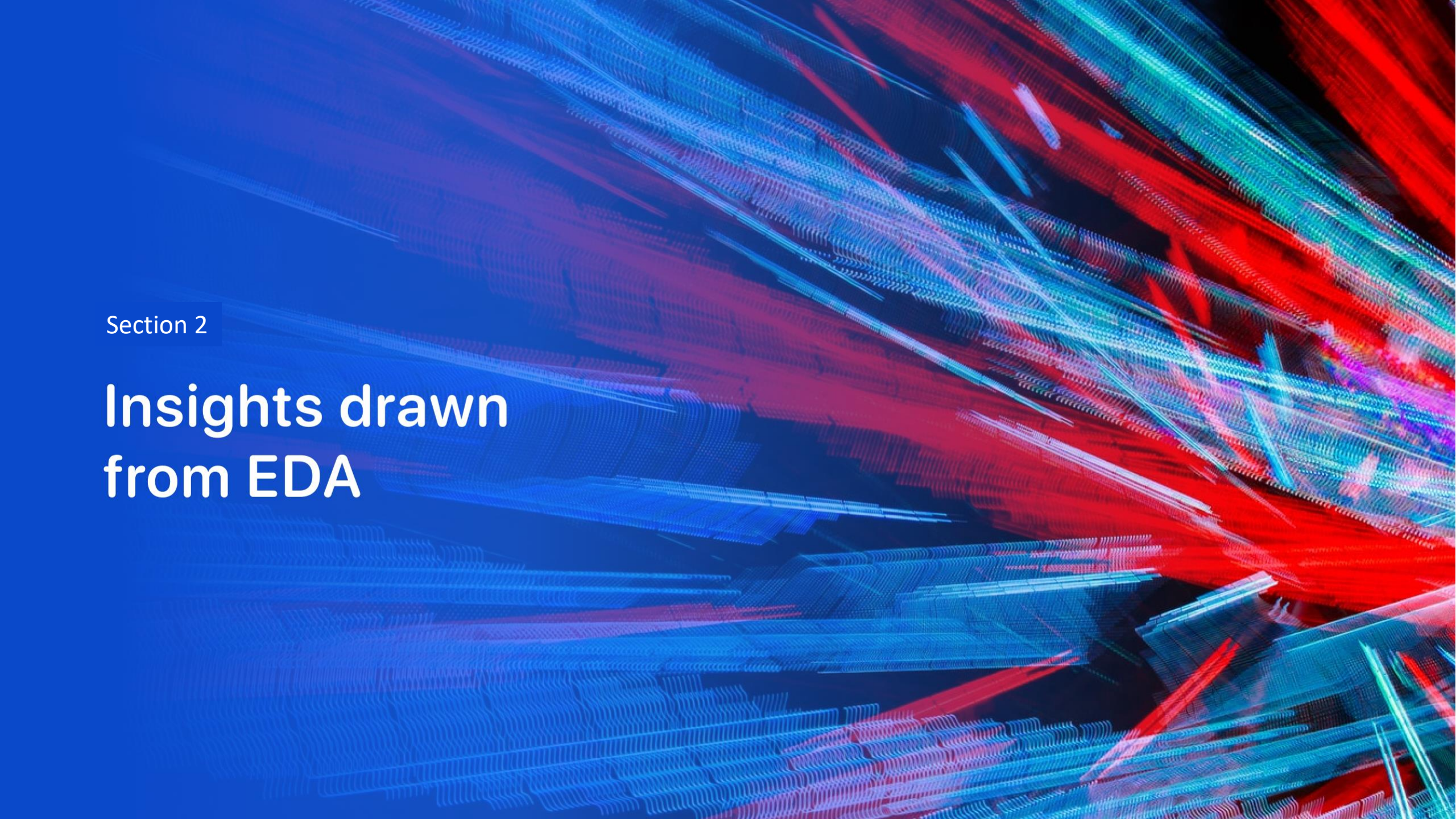
- Summarize how you built, evaluated, improved, and found the best performing classification model
- You need present your model development process using key phrases and a flowchart.
- GitHub URL of predictive analysis lab: https://github.com/abdulghani91/Applied-Data-Science-Capstone/blob/master/Week_4_Predictive_Analysis_Overview/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb



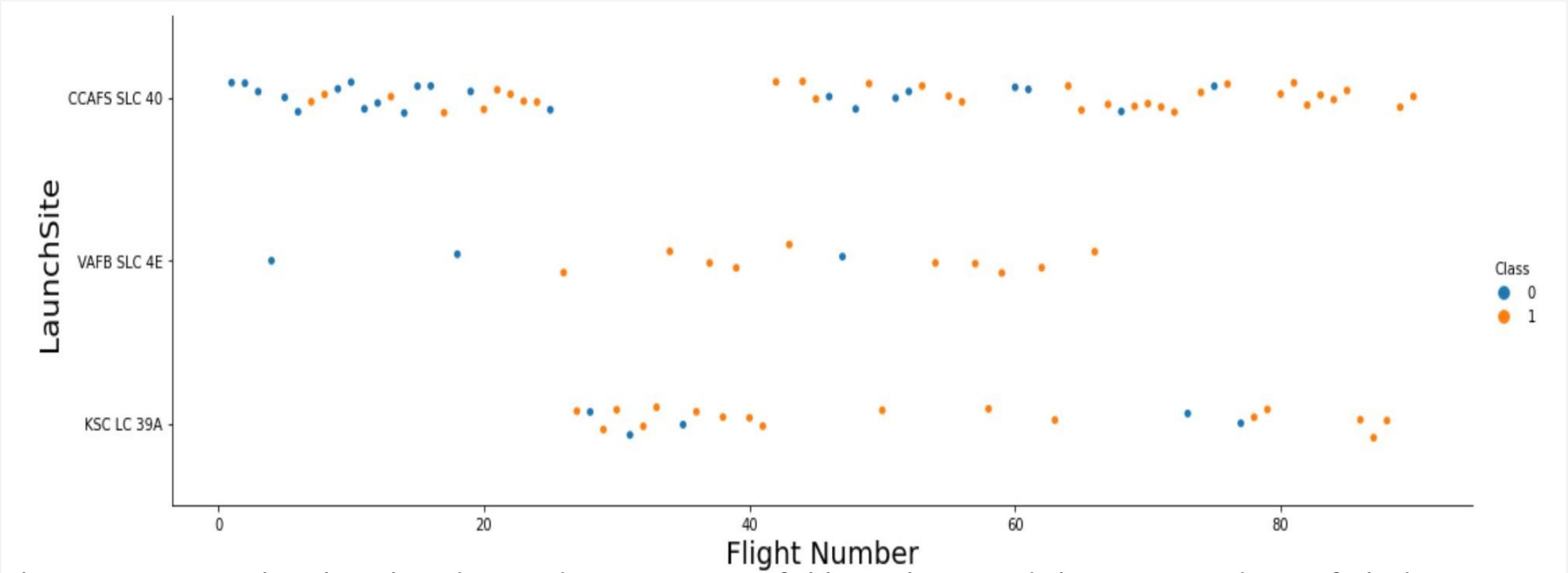
Results



This is a Plotly Dashboard show the result of EDA with visualization, EDA with SQL, Interactive Map with Folium and the result of our models

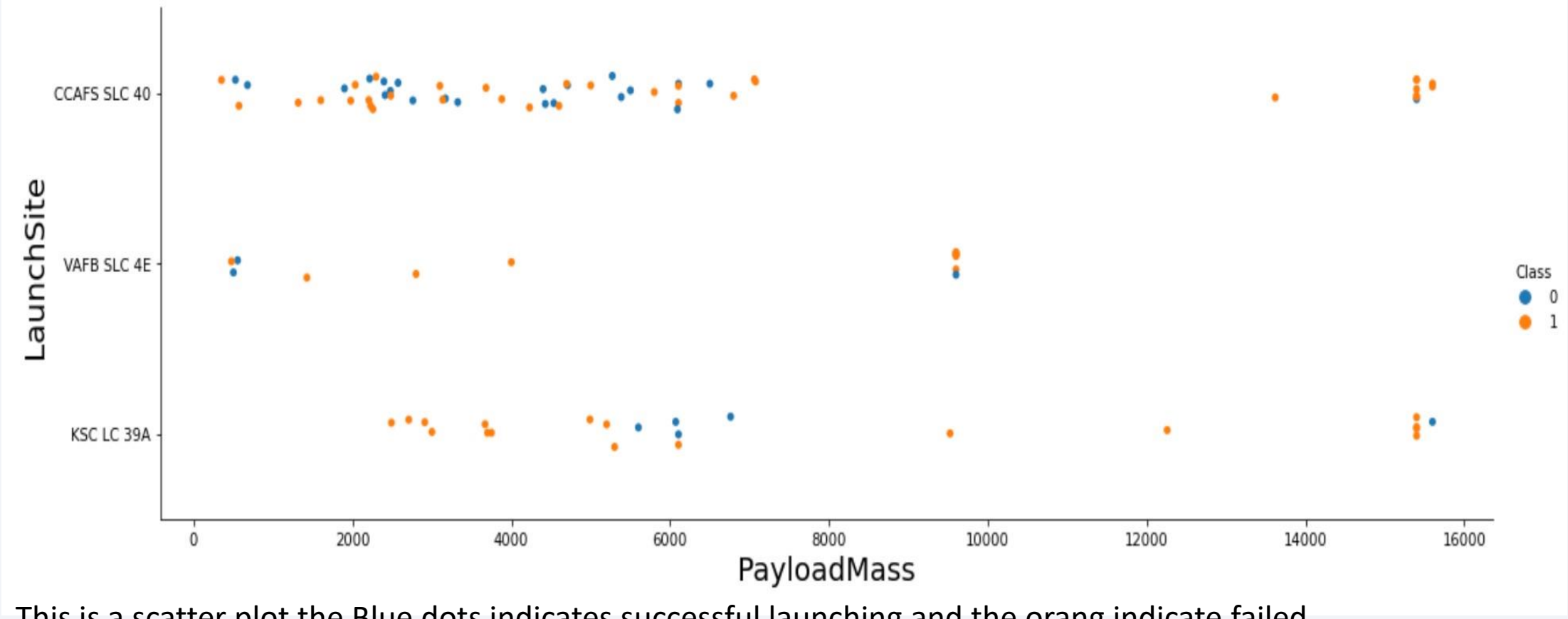

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



This is a scatter plot the Blue dots indicates successful launching and the orang indicate failed

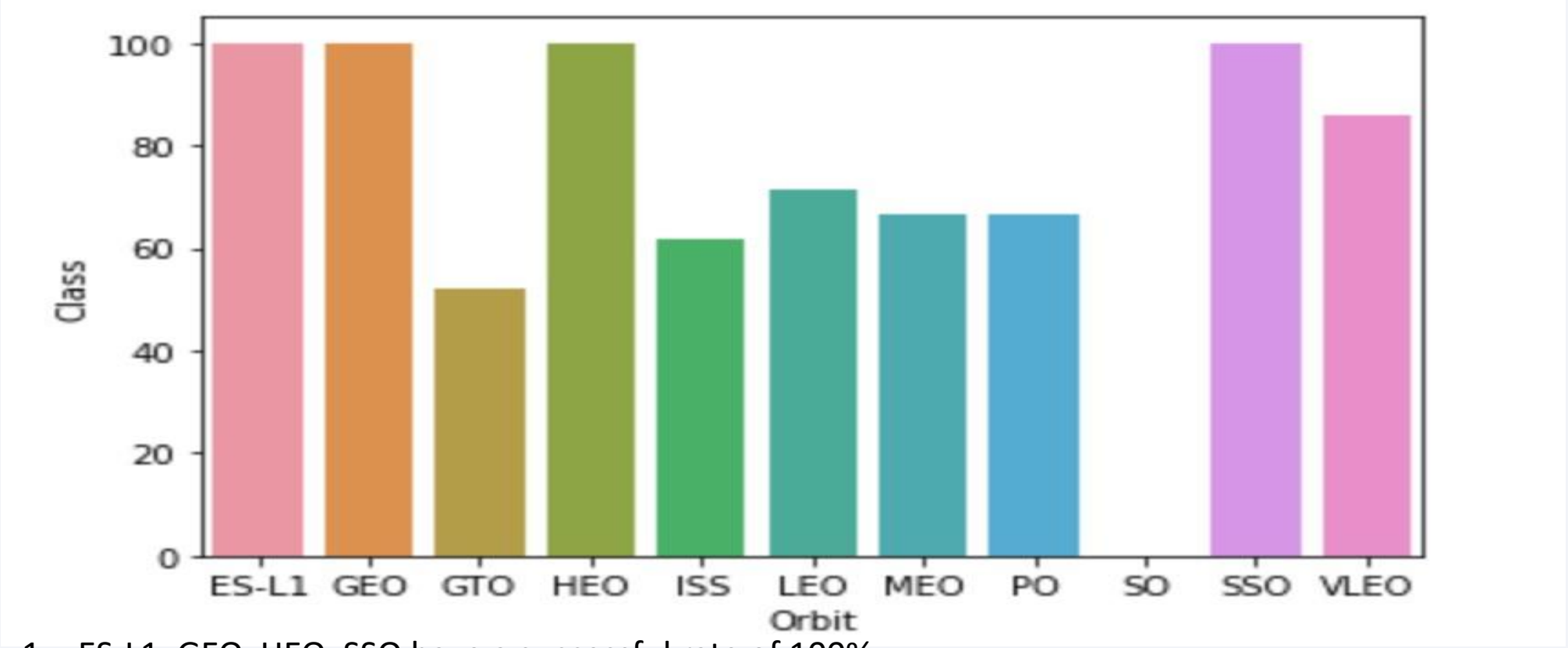
Payload vs. Launch Site



This is a scatter plot the Blue dots indicates successful launching and the orang indicate failed

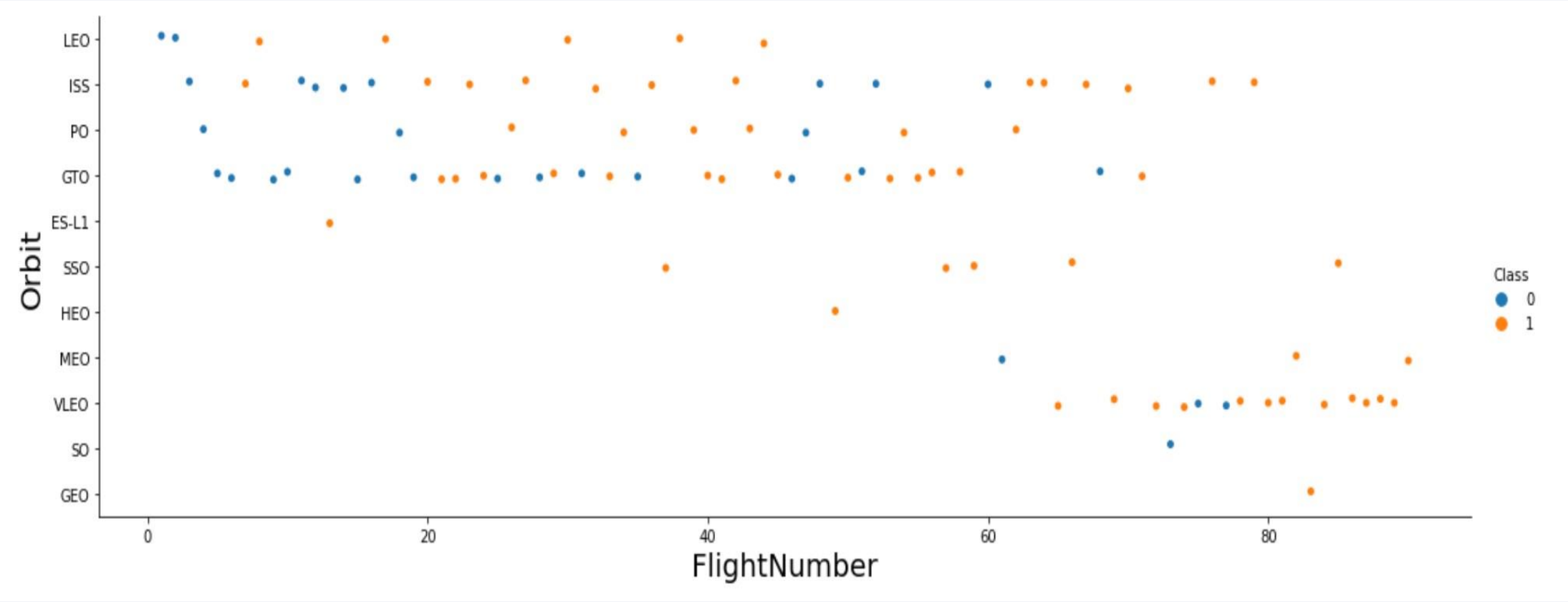
This graph show that almost 75% of the launching was for payloadmass between 0-8000

Success Rate vs. Orbit Type



1. ES-L1, GEO, HEO, SSO have a successful rate of 100%.
- GTO have a successful rate of 50%.
- ISS have a successful rate of 60%.
- LEO have a successful rate of 70%.
- MEO, PO have a successful rate of 65%.
- VLEO have a successful rate of 90%.

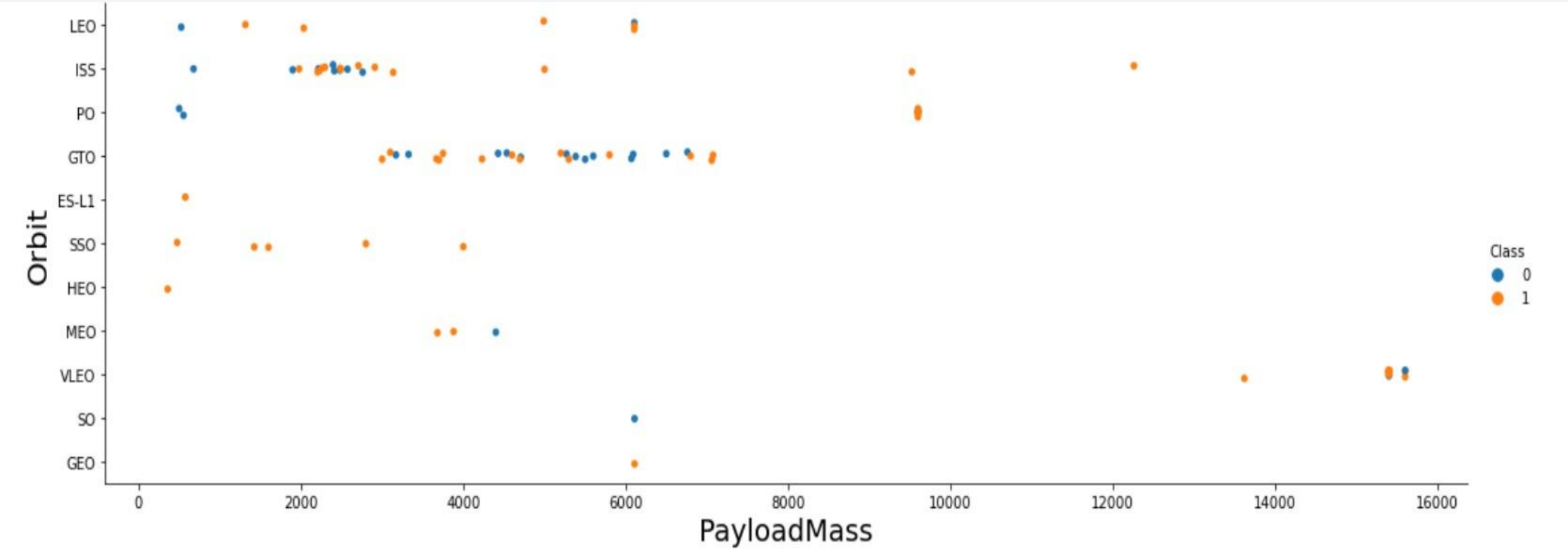
Flight Number vs. Orbit Type



This is a scatter plot the Blue dots indicates successful launching and the orang indicate failed

We can see that from flight number 0 to 60 the Flight done using orbit type LEO, ISS, PO, GTO
And from 60 to the last one for the others.

Payload vs. Orbit Type

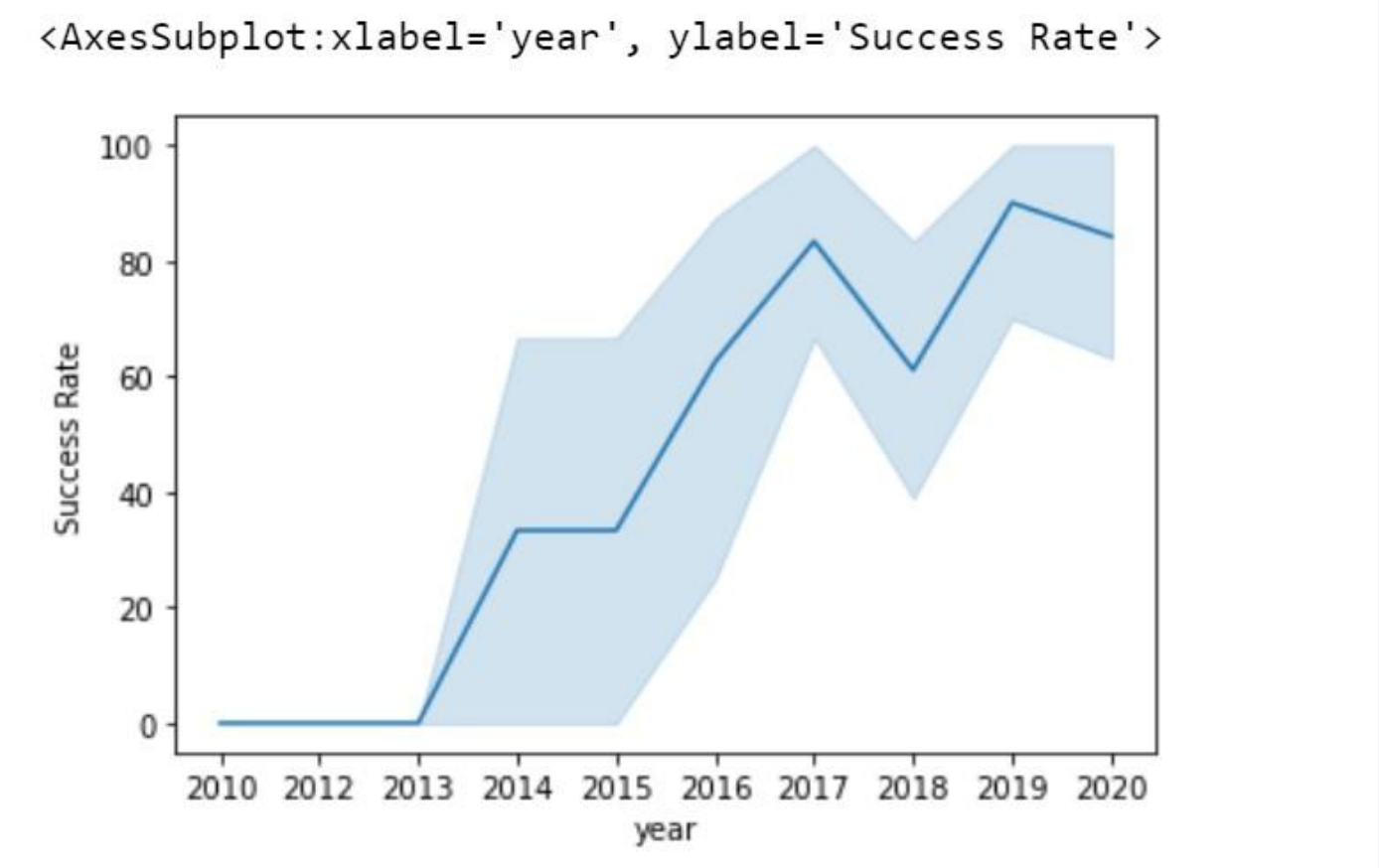


This is a scatter plot the Blue dots indicate successful launching and the orang indicates failed

- LEO, and ISS seem to have the most load from 0 to 3500 KG.
- GTO has the most load from 3500 to 8000.
- LEO, ISS GTO has a total of 75% of all the orbit types.

Launch Success Yearly Trend

- The chart indicates the increase in yearly success.
- The blue line indicates the increase in yearly success.
- The success rate star at 0 in the year 2010 and in 2013 started increasing until 2020.



EDA with SQL

- EXPLORATORY DATA ANALYSIS WITH SQL DB2
- INTEGRATED IN PYTHON WITH SQLALCHEMY

All Launch Site Names

- The names of Lancing Site: CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, CCAFS SLC-40.
- Using the Distinct query we will have a unique result without duplicated items.
- We use Distinct with the Lunch_site to get the name of every launching site.

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

```
%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'

- We used select to view the result and we specify the content using where and like.
- We use ‘CCA%’ in the like clause to specify we want to view any record start with CCA.

Display 5 records where launch sites begin with the string 'CCA'

```
In [35]: %sql select * from SPACEXTBL where "Launch_Site" like "CCA%" limit 5
```

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

Out[35]:

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

- We used the sum function which is a special SQL function to give us the summation of a specific column.
- We used sum function to sum all the values in PAYLOAD_MASS_KG columns if any values in customer columns = ‘NASA (CRS)’.
- The summation result was 45596.

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

```
In [36]: %%sql select sum("PAYLOAD_MASS__KG_") as sum
         from SPACEXTBL where Customer like 'NASA (CRS)'

          * sqlite:///my_data1.db
         Done.

Out[36]:  sum
          45596
```

Average Payload Mass by F9 v1.1

- We used AVG SQL special function to calculate the average of PAYLOAD_MASS_KG, for every Booster_Version equal to ‘F9 v1.1’.
- The Average value is stored in the new table named Average.
- The value of calculating the Average was 2928.4.

Display average payload mass carried by booster version F9 v1.1

```
In [38]: %%sql select avg("PAYLOAD_MASS__KG_") as Average
         from SPACEXTBL where "Booster_Version" like 'F9 v1.1'

          * sqlite:///my_data1.db
         Done.

Out[38]:
```

Average
2928.4

First Successful Ground Landing Date

- We use select to view the first successful landing, by specifying the value of column Mission_Outcome equal to Success.
- We used two different ways the first one with the min() function which gives us the mini value of the date, and the second one with the order by Ascending and limiting the result to 1.
- The date of the first successful landing on 01-03-2013.

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [63]:
```
%sql select min(Date) as DATE
from SPACEXTBL where "Mission_Outcome" like 'Success'

%sql select Date, Mission_Outcome from SPACEXTBL
where "Mission_Outcome" like 'Success' order by Date ASC limit 1
```

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

Out[63]:

DATE
01-03-2013

Successful Drone Ship Landing with Payload between 4000 and 6000

- the names of boosters (4000-6000): F9FTB1022, F9FTB1026, F9FTB10212, F9B10312.
- We used multiple conditions in where clause using AND, and like for specifying the values of the different columns.

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [72]:
```
%%sql select booster_version from SPACEXTBL
where (mission_outcome like 'Success') AND ("PAYLOAD_MASS__KG_" BETWEEN 4000 AND 6000)
AND ("Landing _Outcome" like 'Success (drone ship)')
```

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

Out[72]:

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- We have 99 Successes and 1 Failure.
- We use the count function with the group by to count how many success values and Failures we have, and we used group by to group the outcome into two classes success and Failure, and we store the counted values in new columns called Total_Number.

List the total number of successful and failure mission outcomes

```
In [80]: %sql select Mission_Outcome, count(*) as Total_Number
         from SPACEXTBL group by Mission_Outcome
```

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

Out[80]:

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

Boosters Carried Maximum Payload

- We used a subquery to review the result.
- We show the Booster_Version, Payload_Mass_KG, and our where clause contained a subquery for getting the maximum Payload_Mass_KG.

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [84]:
```
%sql select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTBL
where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

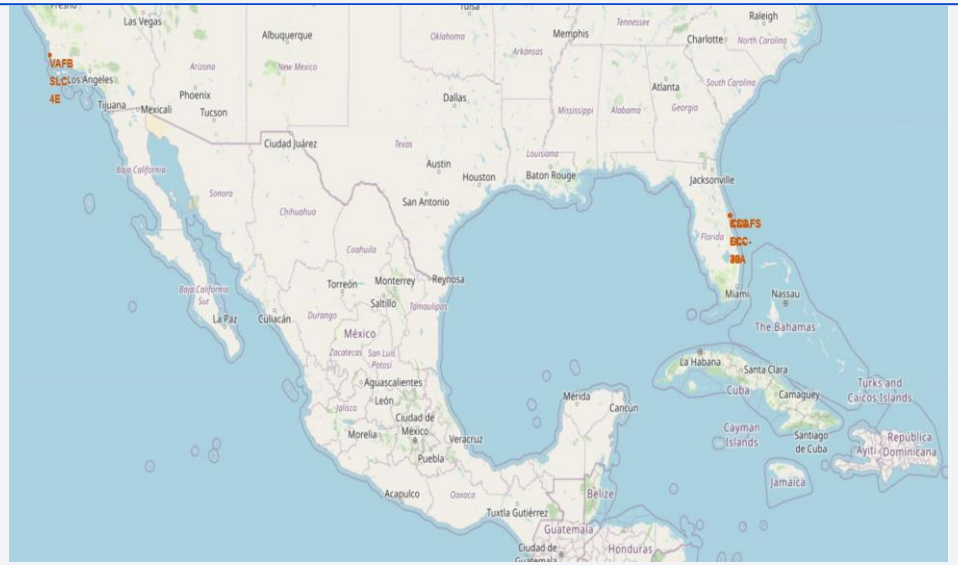
Out[84]:

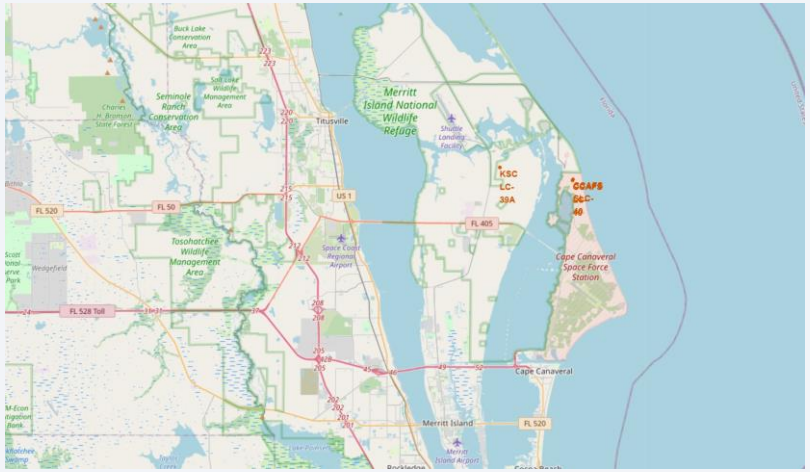
Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

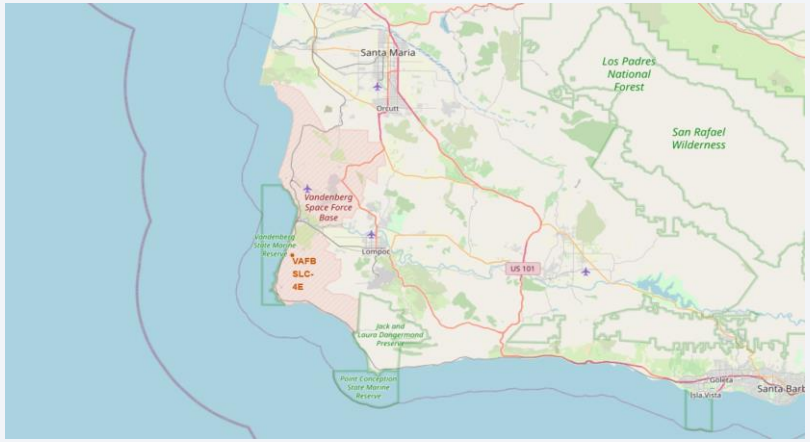
Section 3

Launch Sites Proximities Analysis

Launching Sites Locations



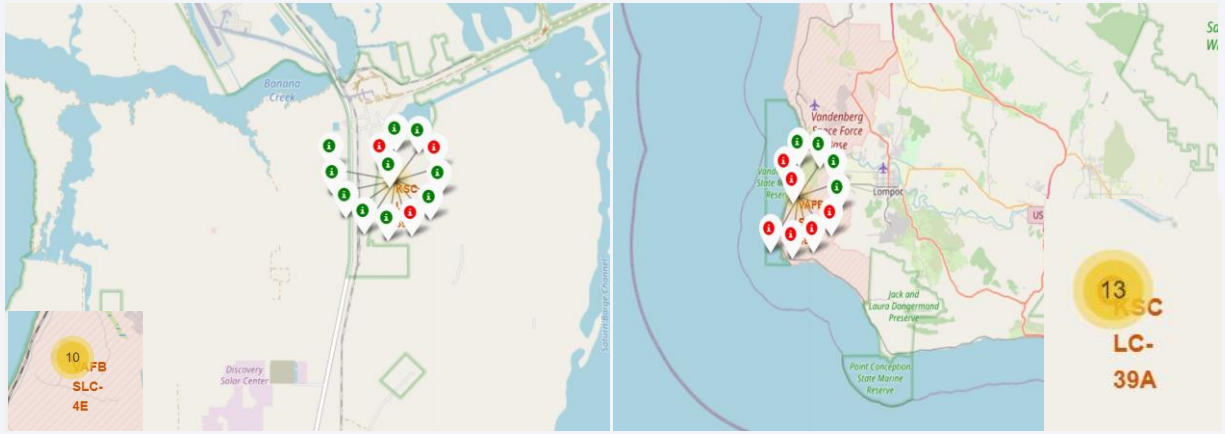


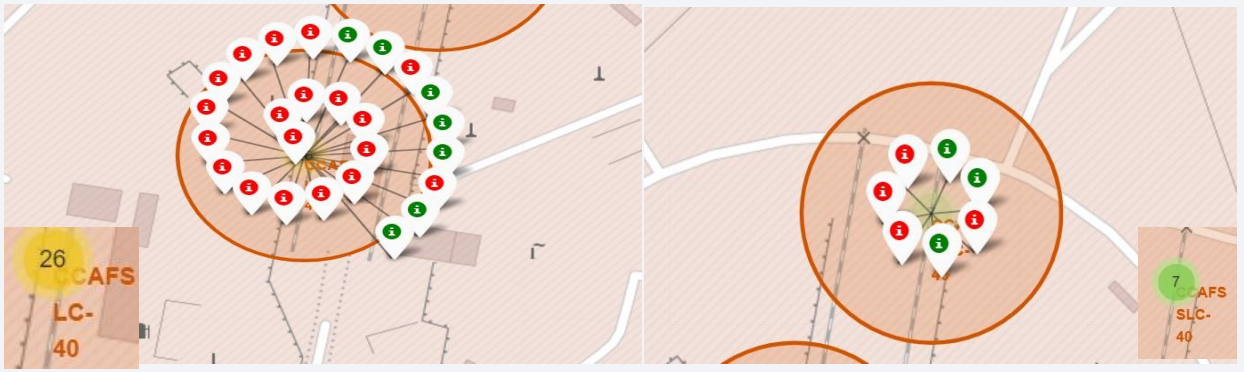


- The first photo on the left shows the site for each launching site.
- The right two photos show the location for each two nearest launching sites.
- The top right photo shows the two Florida launching sites that are very close to each other, and the bottom photo shows the Los Angles two launching sites that are close to each other too.

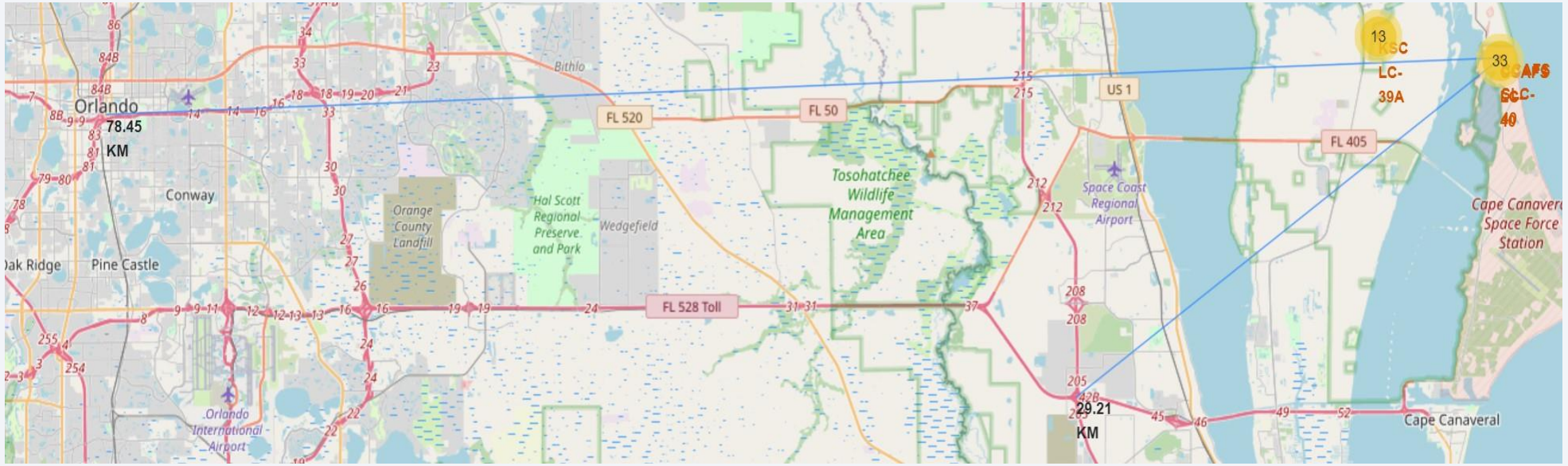
Total number of lunching with success or failed status

- In the photos we show the failed launching with red dots and the successful launching with green dots, for every launching site.
- The yellow area with the number, this number represents the total number of launches for every area.

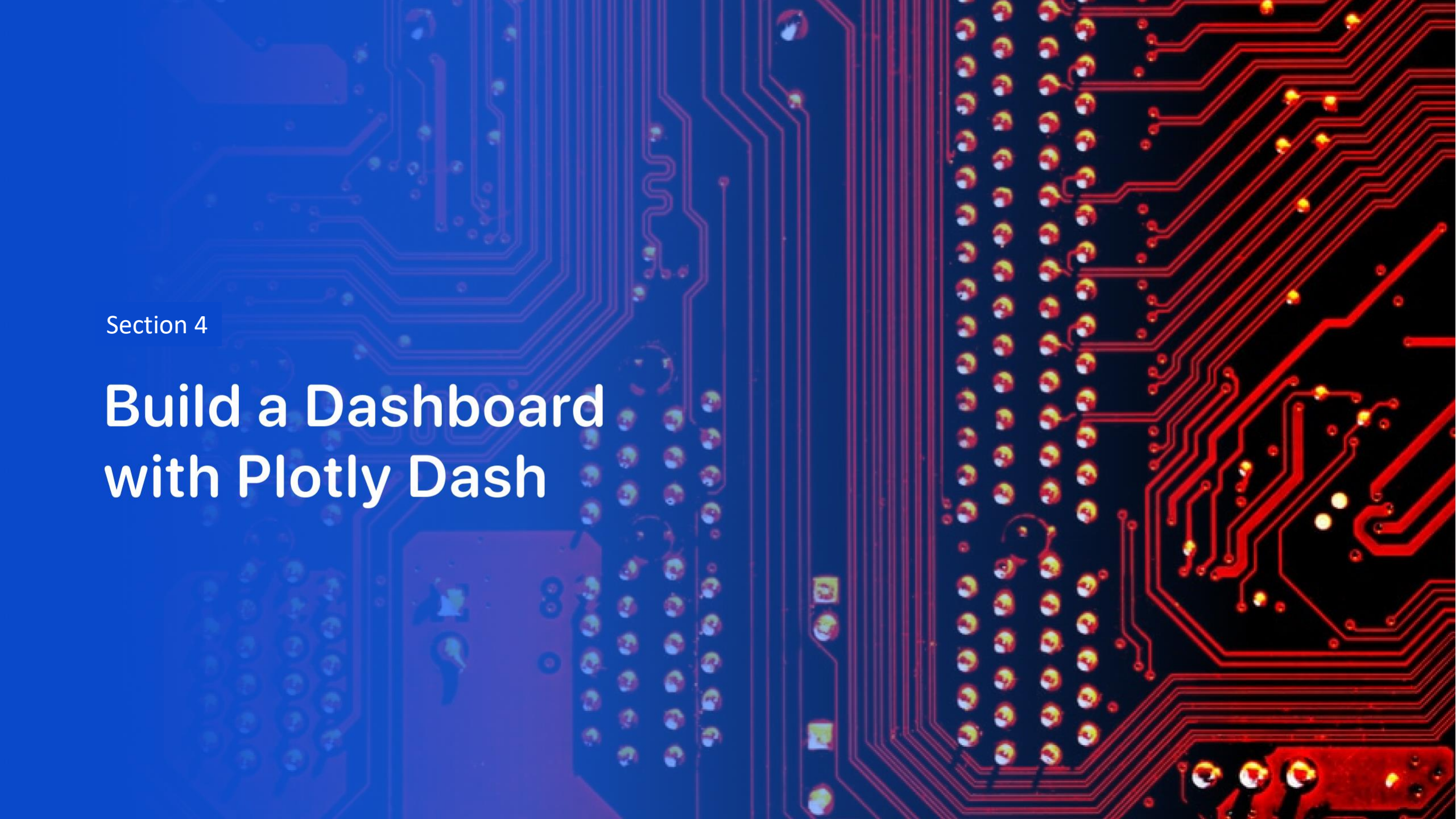




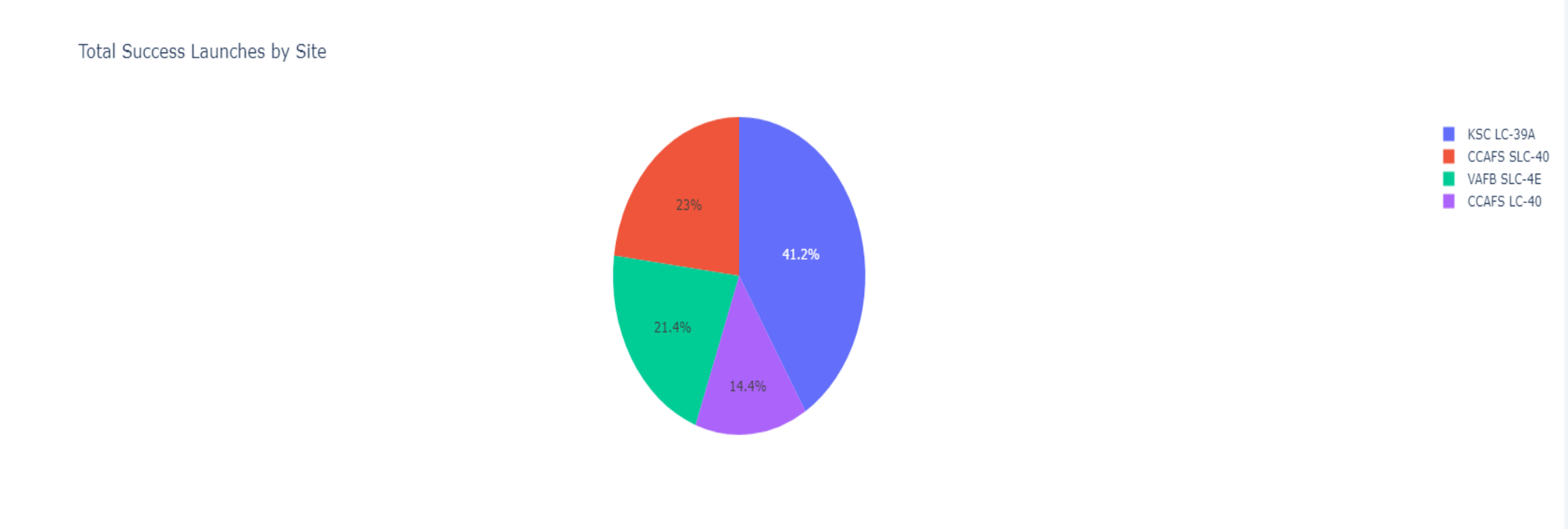
<Folium Map Screenshot 3>



- The two photos indicate the distance between the launching site and the nearest highway.
- From CCAFS SLC-40 to Beach Expression Way 29.21 KM
- From CCAFS SLC-40 to South Street 78.45 KM

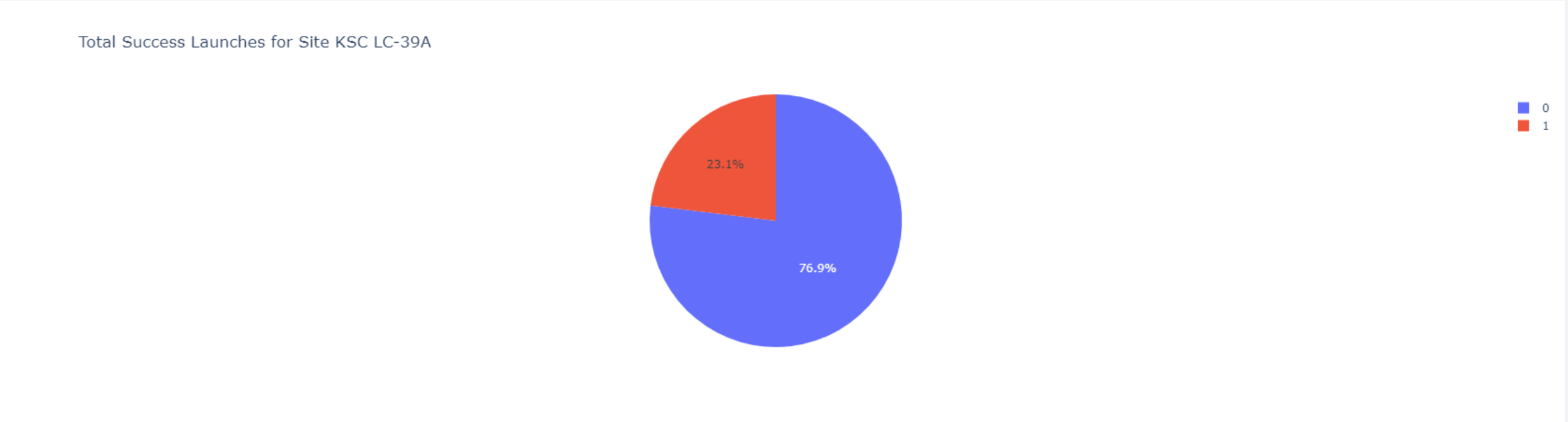

Section 4
Build a Dashboard with Plotly Dash

Total Success Landing Across Lunching Site



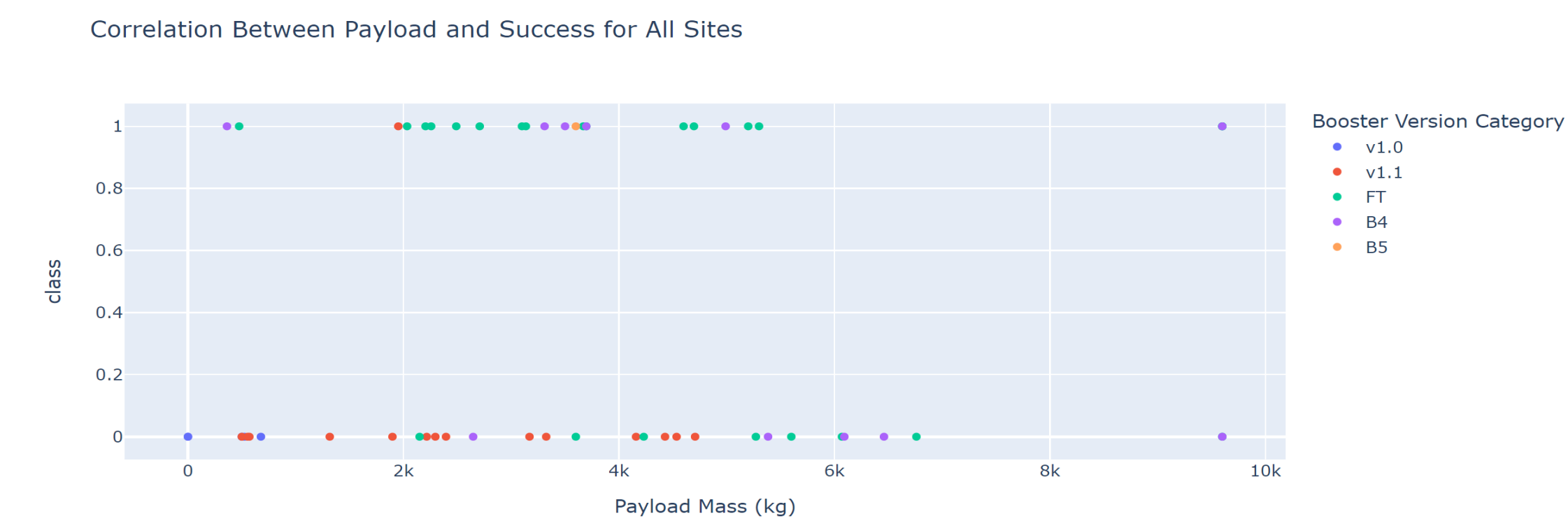
- This Pi chart review the total Success Landing for every launching site.
- KSC has the greatest amount of successful landings, in the second place CCAFS SLC-40.
- VAFB SLC-4E(21%), and CCAFS LC-40(23%) have the smallest amount of success.

The Highest Success Launching Site



- Pi chart review the highest success launching site KSC LC-39A.
- 1 viewed in blue color which indicates the success of landing, the success rate is 76.9, and 0 indicates the landing failed, the failure rate is 23.1%.

<Dashboard Screenshot 3>

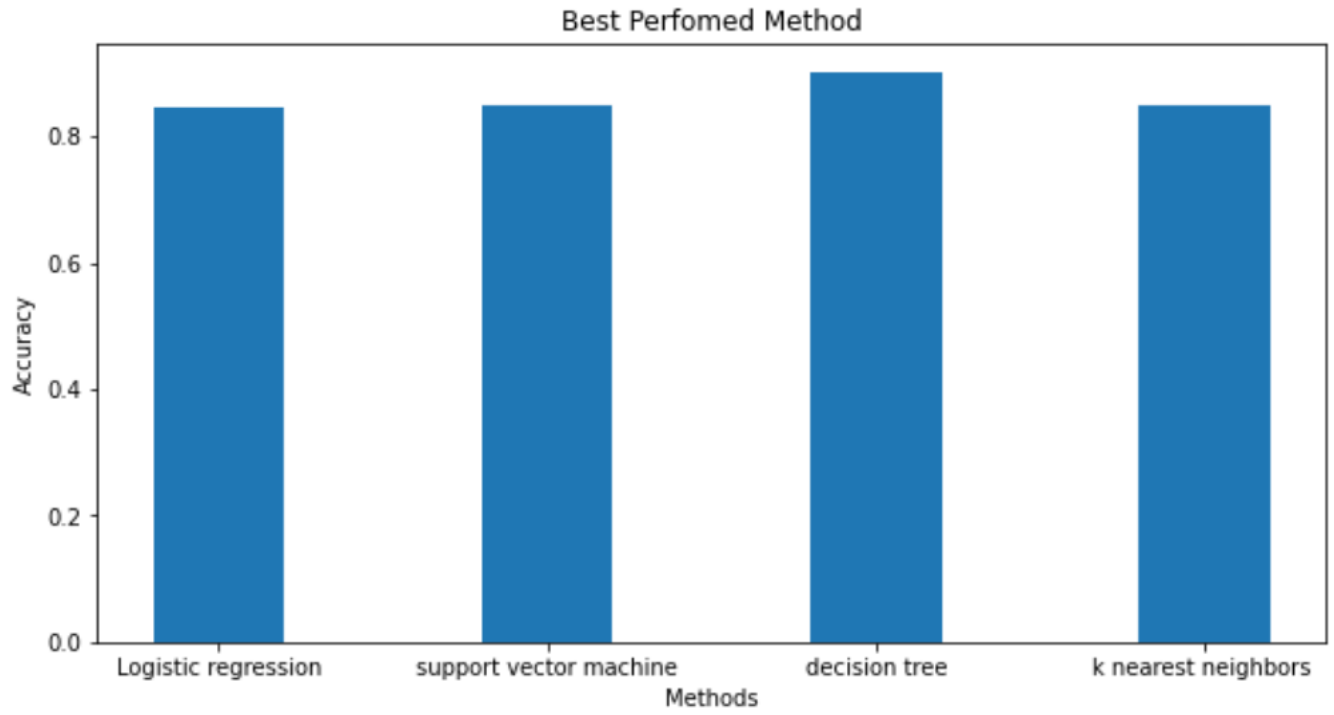


- This plot shows the success and faille of every booster with a different color.
- success landing take class 1 and faille 0, and every booster have a different amount of load (KG).

Section 5

Predictive Analysis (Classification)

Classification Accuracy



	LogReg	SVM	Tree	KNN
Jaccard_Score	0.833333	0.845070	0.776119	0.819444
F1_Score	0.909091	0.916031	0.873950	0.900763
Accuracy	0.866667	0.877778	0.833333	0.855556

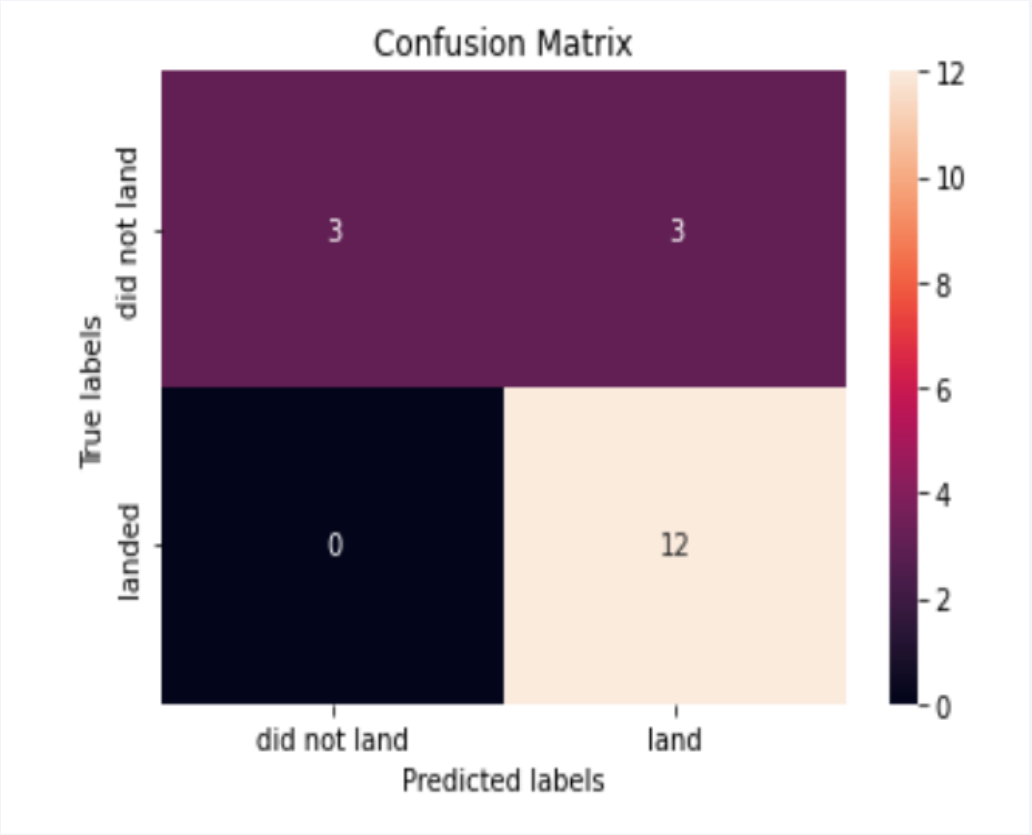
```
print("\n The Best Performs Method is The", max(scores), "Method")
```

```
The Best Performs Method is The Tree Method
```

- The bar chart displayed the accuracy for every model that we build.
- We can see that the Decision Tree model gets the best performance.

Confusion Matrix

- All the models have the same Convention Matrix because they give the same result with the test data.
- The model Predict 12 successful landings while it was labeled as landing.
- Predicts 3 unsuccessful landings while it's labeled unsuccessful landing.
- Predicts 3 successful landings while it's labeled unsuccessful landing.



Conclusions

- Our task: to develop a machine learning model for Space Y which wants to bid against SpaceX
- The goal of the model is to predict when Stage 1 will successfully land to save ~$100 million USD
- Used data from a public SpaceX API and web scraping SpaceX Wikipedia page
- Created data labels and stored data into a DB2 SQL database
- Created a dashboard for visualization
- We created a machine learning model with an accuracy of 83%
- Allon Mask of SpaceY can use this model to predict with relatively high accuracy whether a launch will have a successful Stage 1 landing before launch to determine whether the launch should be made or not
- If possible more data should be collected to better determine the best machine learning model and improve accuracy

Thank you!