IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Junyoung JONG

18th March 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection with API
  - Data Collection with Web Scraping techniques
  - Data Wrangling (Pre-processing)
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Python Libraries (Pandas, Seaborn, etc)
  - Data visualization with Folium and Plotly (Dashboard)
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis results
  - Interactive Analysis with screenshots
  - Predictive Analysis results from Machine Learning Techniques

Introduction

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. In this project, we conduct data analysis and machine learning on the given data of Space X's Falcon 9 rocket and use machine learning to predict the landing outcome of the first stage of launch.

Problems you want to find answers

1) Identifying all variables that influence the landing success rate
2) Relationship between each variables (Correlation)
3) The best model that gives maximum success rate

Section 1

Methodology

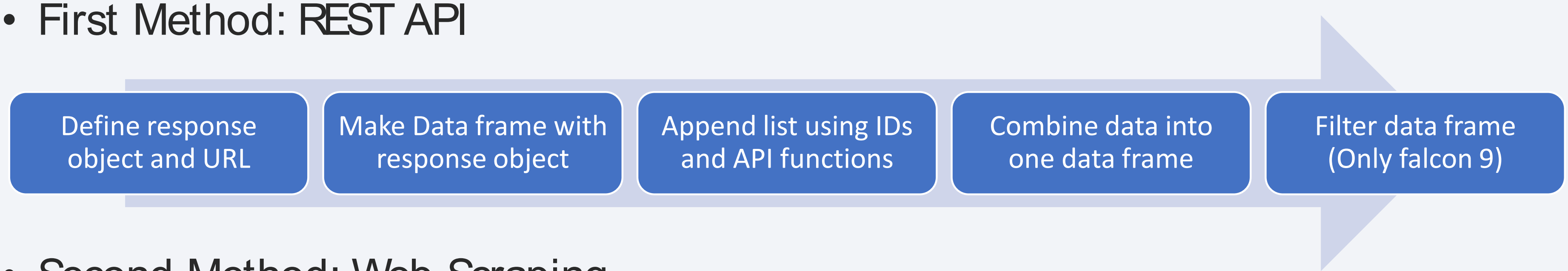
Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SPACE X's API (JSON) and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Experiment models with machine learning techniques such as SVM and Decision Trees

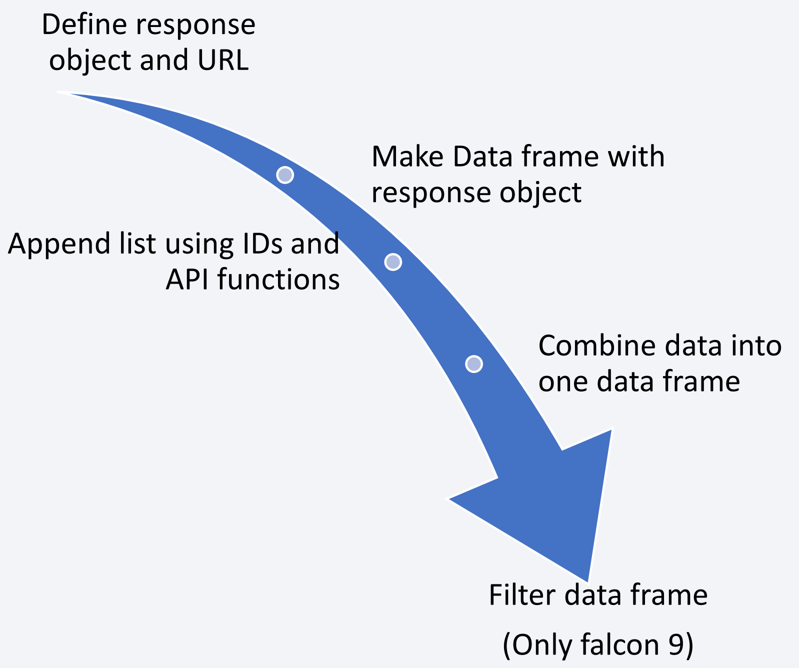
Data Collection

- Data collection was done with 2 different methods. First, the REST API of Space X and the second was using web scraping techniques.
- First Method: REST API



- Second Method: Web Scraping

Data Collection – SpaceX API




```
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())
```

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

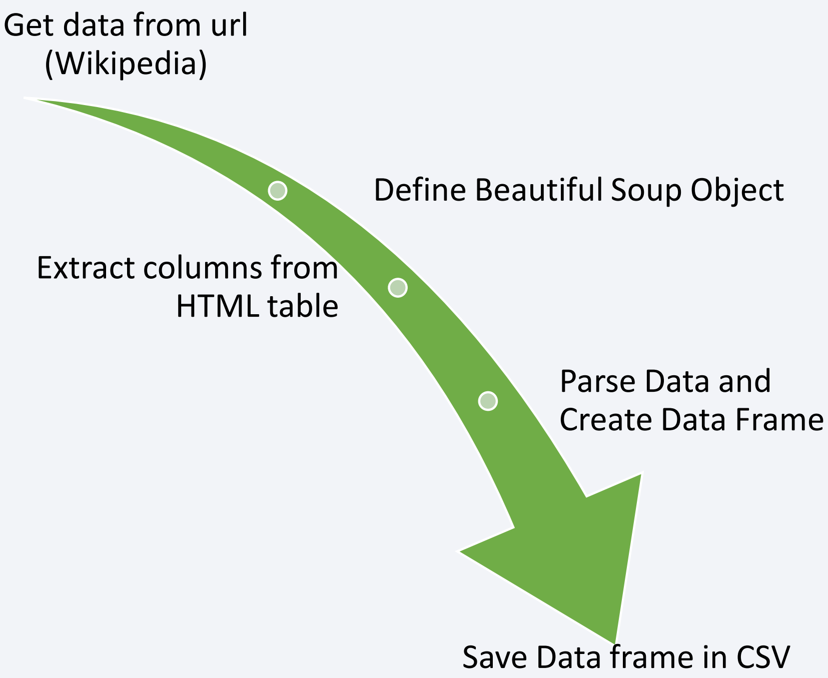
```
# Call getCoreData
getCoreData(data)
```

Finally lets construct our dataset using the data we have obtained. We we combine the columns into a dictionary.

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

https://github.com/shanis345/IBM_Capstone-2024-/blob/main/1.%20jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping



TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP

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

200

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text)
```

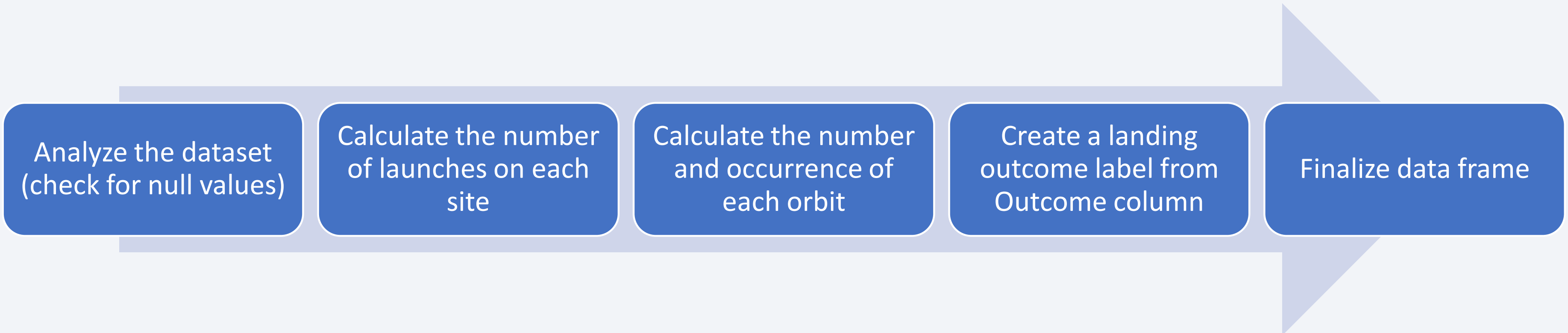
```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders cc
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a num
        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])
```

https://github.com/shanis345/IBM_Capstone-2024-/blob/main/2.%20jupyter-labs-webscraping.ipynb

Data Wrangling

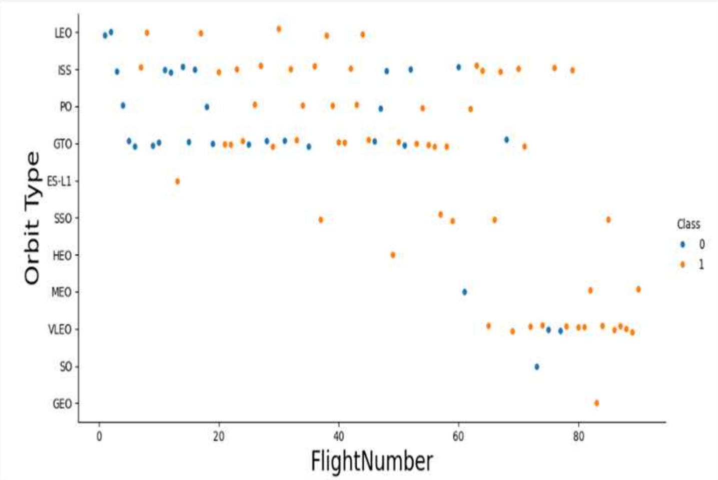
- Data wrangling is the process of cleansing and purify the data frame. In other words, it is called data preprocessing. Here we deal with null values, drop useless data and also add necessary columns. Such columns may include classification dummies and date columns.

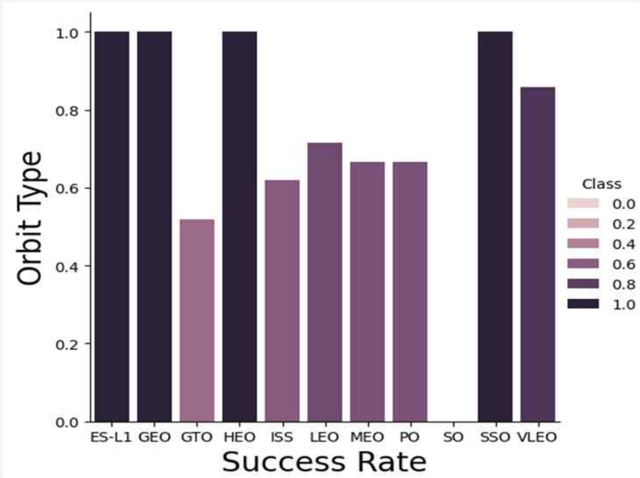


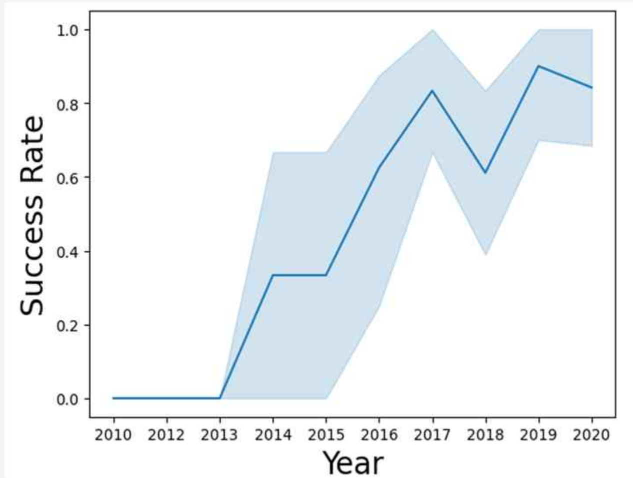
https://github.com/shanis345/IBM_Capstone-2024-/blob/main/3.%20labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Here we used scatterplots, bar charts and line charts for data visualization. The chief goal was to identify the change of success rate depending on the relation between variables. The main variables were, ‘Orbit’, ‘Launch Site’, ‘Flight Number’, ‘Payload Mass’ and ‘Year’







https://github.com/shanis345/IBM_Capstone-2024-/blob/main/5.%20jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

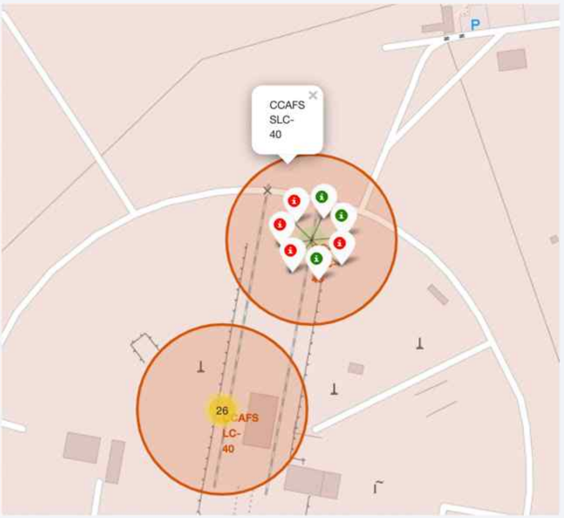
- The purpose of EDA with SQL was to understand the data set in more details. We used multiple queries to understand how our data looks like.

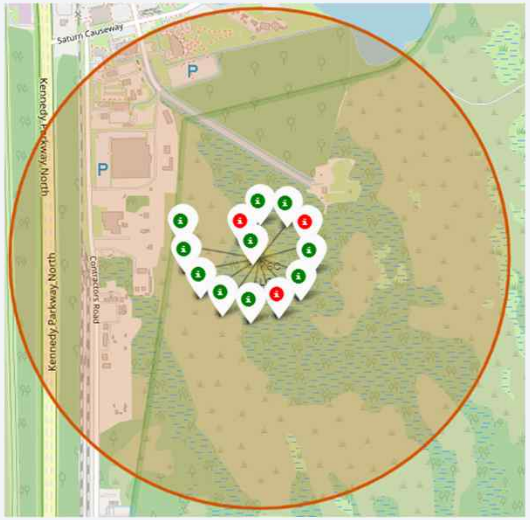
1) Display the names of the unique launch sites in the space mission
2) Display 5 records where launch sites begin with the string 'CCA'
3) Display the total payload mass carried by boosters launched by NASA (CRS)
4) Display average payload mass carried by booster version F9v1.1
5) List the date when the first succesful landing outcome in ground pad was acheived.
6) List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7) List the total number of successful and failure mission outcomes
8) List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
9) 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.
10) 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.

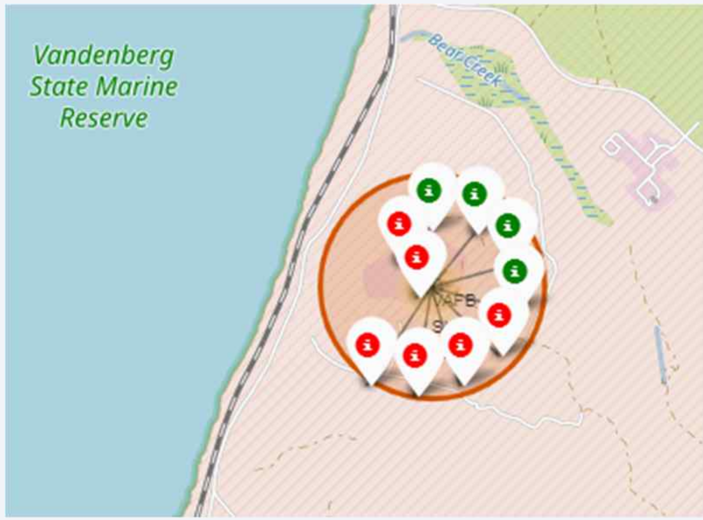
https://github.com/shanis345/IBM_Capstone-2024-/blob/main/4.%20jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- The main purpose using Folium was to visualize the launch data into an interactive map. Therefore, we needed the latitude and longitude coordinates of each launch site and added circle markers around each site. Also, color markers were used to identify failed launches (red) and successful launches (green)



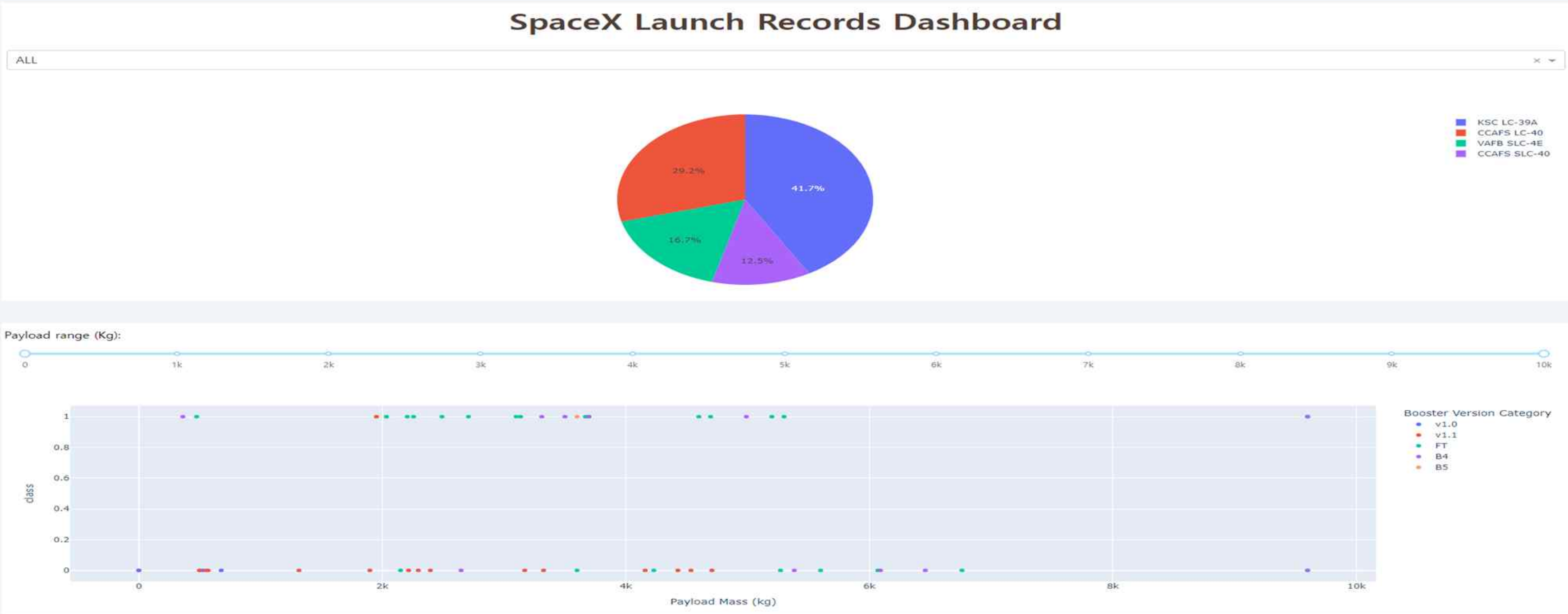




https://github.com/shanis345/IBM_Capstone-2024-/blob/main/6.%20lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

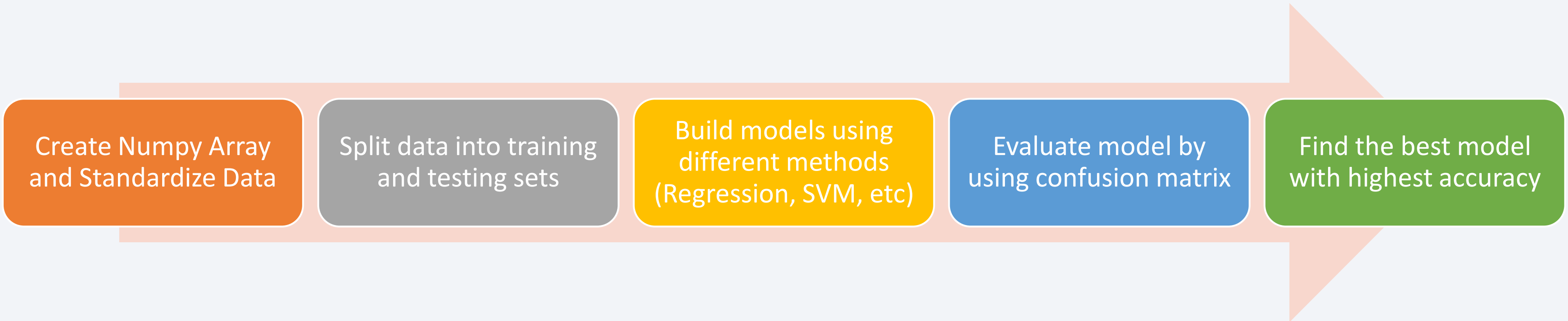
- Pie charts and scatter charts were used for creating a dashboard. The main purpose of plotting it on a dashboard was to identify the relation between multiple variables such as outcome, payload mass and launch site.



https://github.com/shanis345/IBM_Capstone-2024-/blob/main/7.%20Dashboard_SpaceX_.ipynb

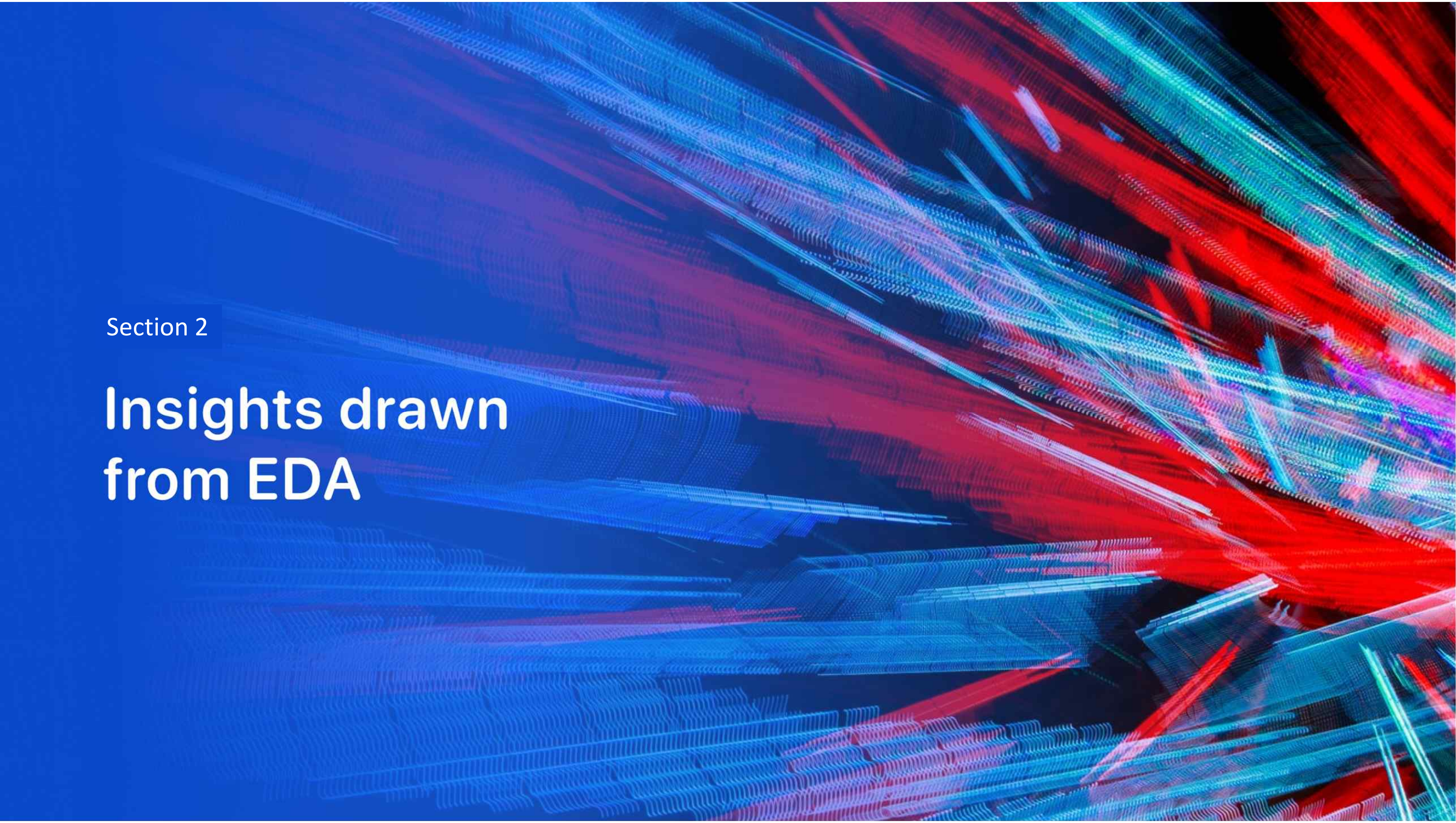
Predictive Analysis (Classification)

- Predictive analysis was conducted using multiple machine learning methods. The data set was split into training data and testing data, and then experimented the data using different methods, such as SVM and Decision Tree. Finally, the best option was chosen based on the experiment results.



Results

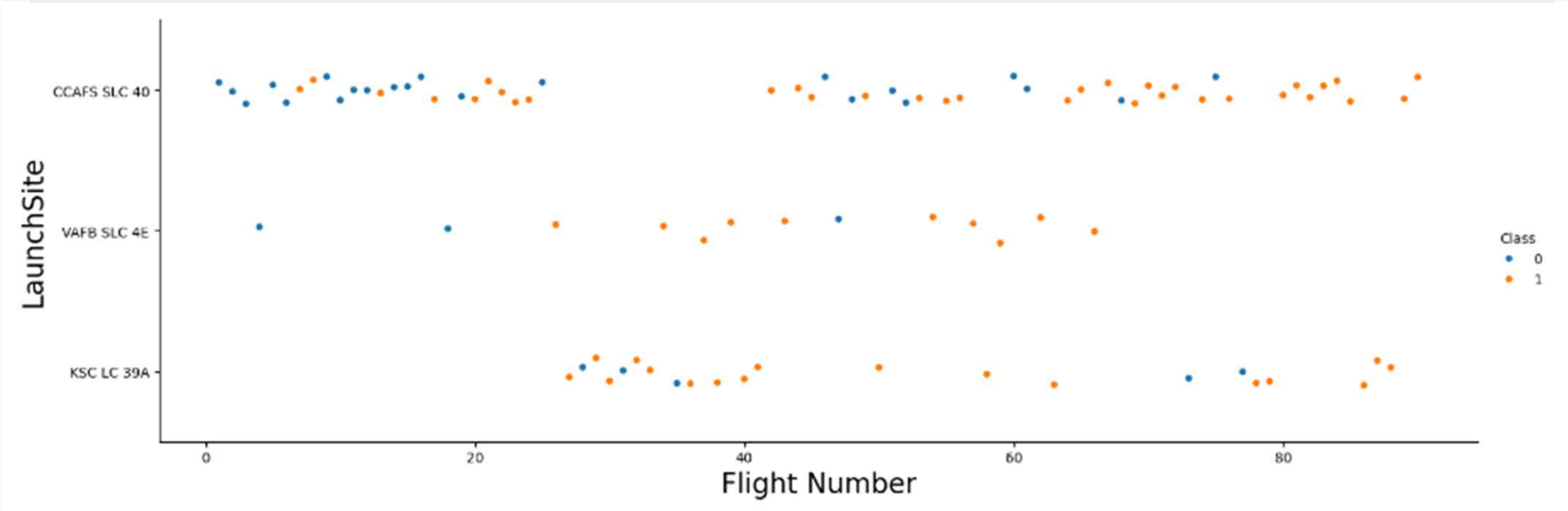
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

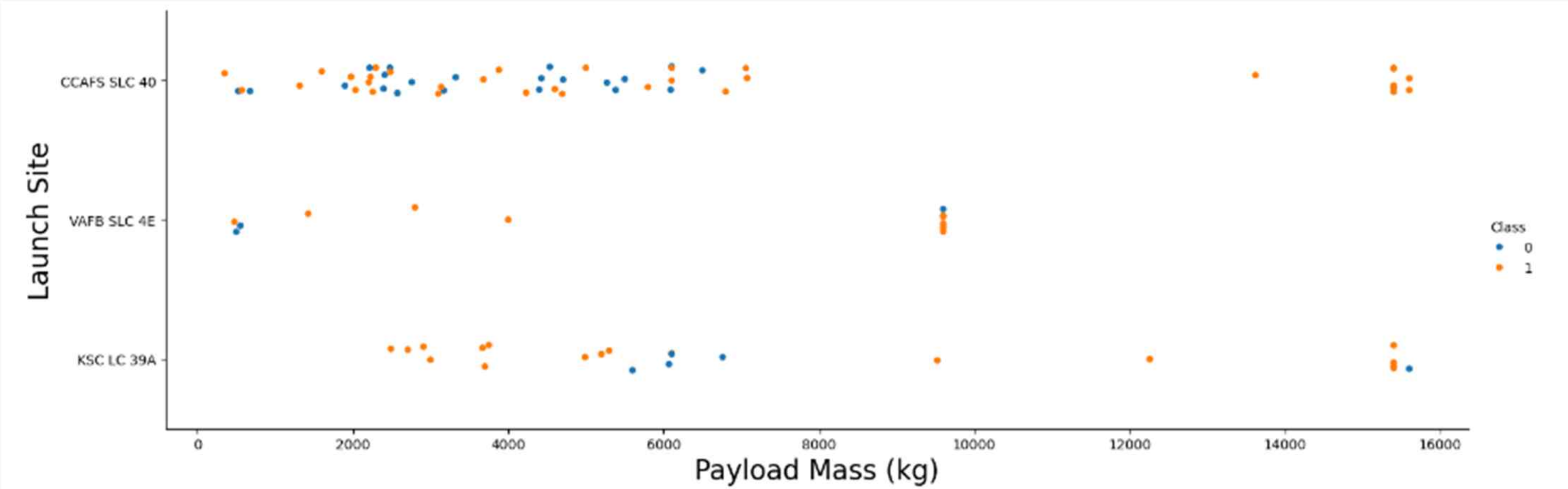
Insights drawn from EDA

Flight Number vs. Launch Site



- This plot shows that for the cases with higher flight numbers seems to have a higher success rate. It also shows that the launch site 'CCAFS SLC 40' has the most cases.

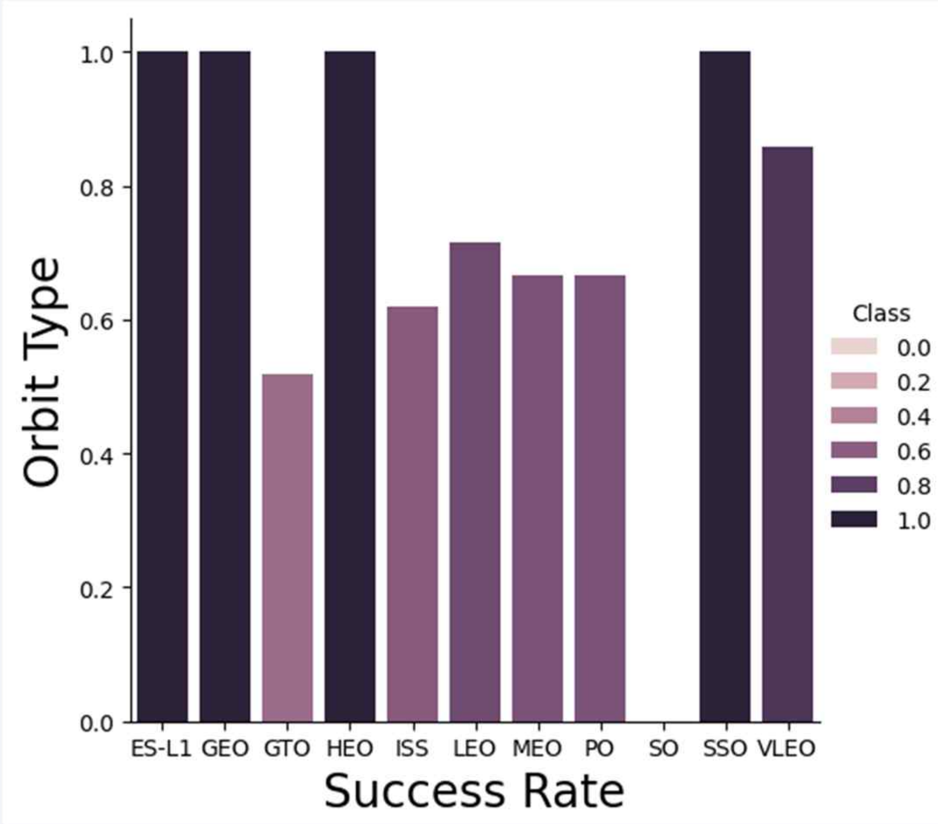
Payload vs. Launch Site



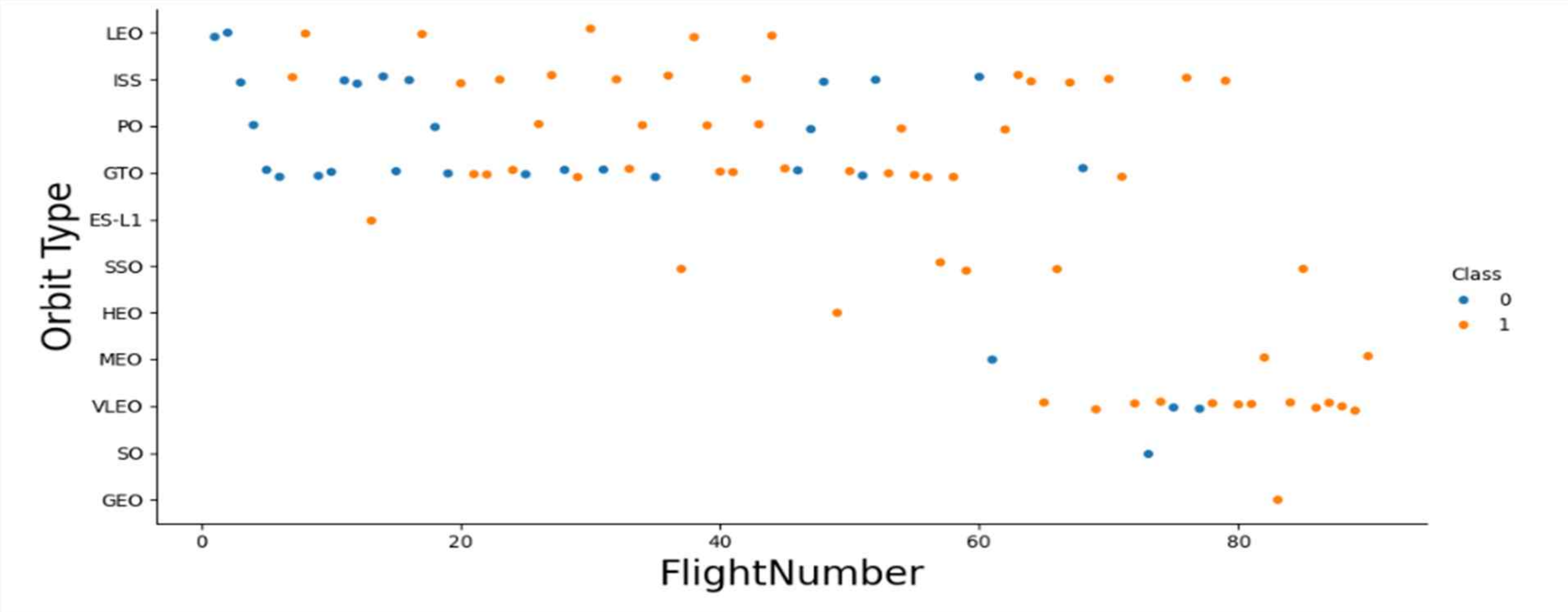
- This plot shows that for the cases with heavier payload mass, the success rate increases. Especially, once the payload mass is greater than 7000 kg, almost all launches have been proved to be successful

Success Rate vs. Orbit Type

- This plot shows that the success rate for different orbit types vary. For such models as ES-L1 and GEO, the success rate was 100%. However, for models such as SO (0%) and GTO (50%), the success rate was very low.

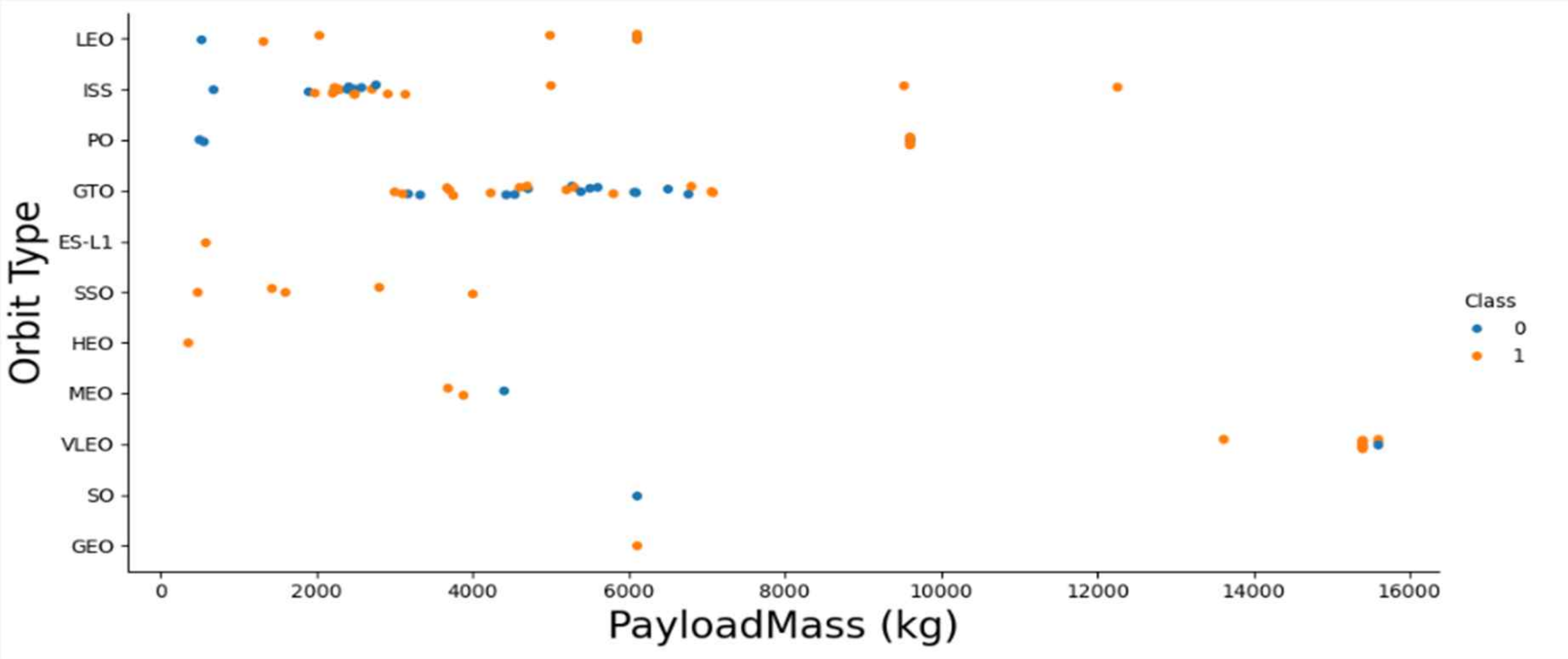


Flight Number vs. Orbit Type



- This plot shows that for the cases with higher flight numbers seems to have a higher success rate for all orbit types. In general, we can say there is a correlation between the two variables.

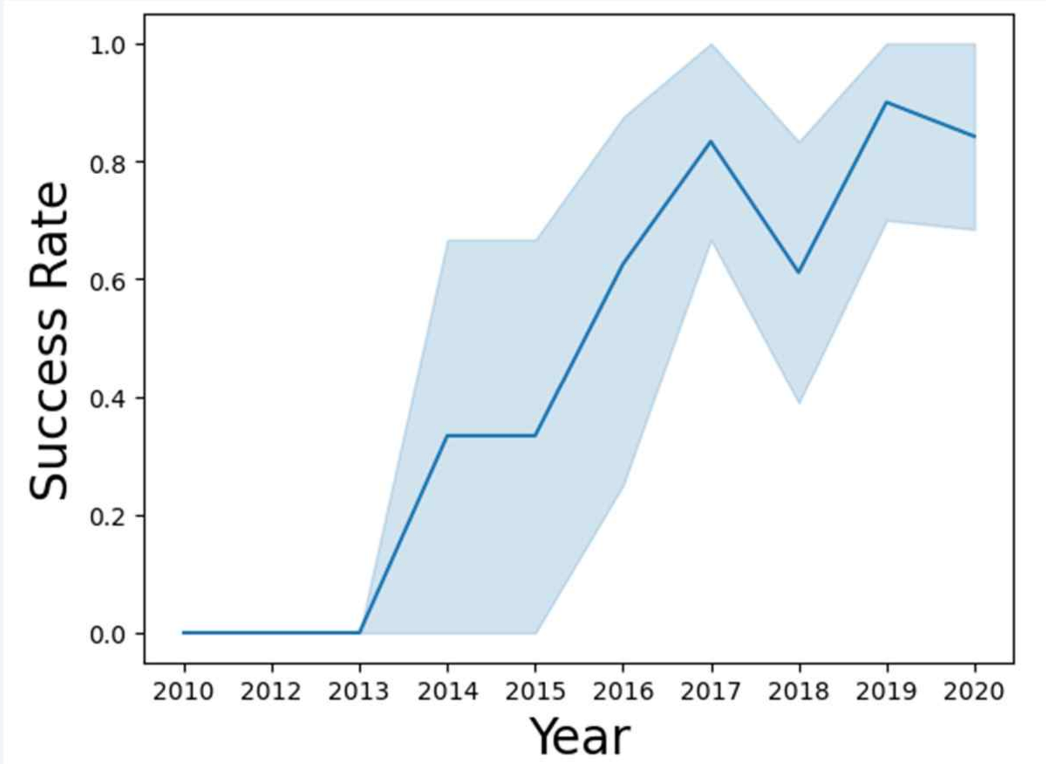
Payload vs. Orbit Type



- This plot shows that for some orbit types such as PO, LEO and ISS, cases with heavier payloads seems to be more successful. However, for GTO, it shows that the payload mass doesn’t really influence the success rate.

Launch Success Yearly Trend

- This figure shows that the success rate is increasing since 2013. Especially since 2019, the success rate reached over 80%. Therefore, we can say that the success rate for futural launches would be higher than old cases.



All Launch Site Names

- This query lists the unique names of launch sites in the space mission. We use ‘distinct’ query to find all unique values from a typical column (LAUNCH_SITE)

Task 1

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

In [14]:
```
%sql select distinct LAUNCH_SITE from SPACEXTBL;
```

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

Out[14]:

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

Launch Site Names Begin with 'CCA'

- This query displays the top 5 records where launch sites begin with the string 'CCA'
- We used 'like' query.

Task 2

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

In [15]:
```
%sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5;
```

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

Out[15]:

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 attemp
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attemp
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attemp

Total Payload Mass

Task 3

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

```
%sql select Customer, sum(PAYLOAD_MASS__KG_) as Total_NASA_CRS_payloadmass from SPACEXTBL where Customer = "NASA (CRS)";
```

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

Customer	Total_NASA_CRS_payloadmass
NASA (CRS)	45596

- This query displays the total payload mass carried by boosters launched by NASA. The SUM query was used to do this task.

Average Payload Mass by F9 v1.1

Task 4

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

```
%sql select Booster_Version, avg(PAYLOAD_MASS__KG_) as Avg_Booster_version_F9v1_1 from SPACEXTBL where Booster_Version = "F9 v
```

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

Booster_Version	Avg_Booster_version_F9v1_1
F9 v1.1	2928.4

- This query displays the average payload mass carried by booster version F9 v1.1. the avg query was used in this task.

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

```
%sql select min(Date), Landing_Outcome as First_Succesful_Landing from SPACEXTBL where Landing_Outcome = "Success (ground pad]
```

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

min(Date)	First_Succesful_Landing
2015-12-22	Success (ground pad)

- This query displays date when the first successful landing outcome in ground pad was achieved. Here we used ‘min’ query. However, ‘order by’ query will work well too.

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

```
%sql select distinct Booster_Version, Landing_Outcome, PAYLOAD_MASS__KG_ from SPACEXTBL where Landing_Outcome = "Success (dror
```

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

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

- This query displays the successful drone ship landing with payload between 4000 and 6000. Here the ‘between’ command was used. However, using ‘and’ command between two phrases will work as well.

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
%sql select Mission_Outcome, count(Mission_Outcome) as 'Count of Mission Outcomes' from SPACEXTBL group by Mission_Outcome;
```

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

Mission_Outcome	Count of Mission Outcomes
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- This query counts the number of success and failure missions. Here the ‘count’ query was used. Total 100 cases were successful and 1 case was a failure.

Boosters Carried Maximum Payload

- This query displays the booster versions which have carried the maximum payload mass.
- We used 'max' query inside a sub query of the select query.

Task 8

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

```
%sql select Booster_Version,Landing_Outcome, PAYLOAD_MASS__KG_ from SPACEXTBL where PAYLOAD_MASS__KG_ in (select max(PAYLOAD_M
```

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

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

2015 Launch Records

Task 9

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%sql select  Date, Booster_Version, Launch_Site, Landing_Outcome from SPACEXTBL where Landing_Outcome= 'Failure (drone ship)'
```

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

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

- This query lists the records of the launches in 2015. There were 2 launches, and both were a failure. The launch site was CCAFS LC - 40 and the booster version was F9 v1.1

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

Task 10

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(Landing_Outcome) as "Total Count"
from SPACEXTBL
where Landing_Outcome = "Failure (drone ship)" or Landing_Outcome = "Success (ground pad)" and
Date between "2010-06-04" and "2017-03-20"
group  Landing_Outcome
order by Landing_Outcome desc;
```

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

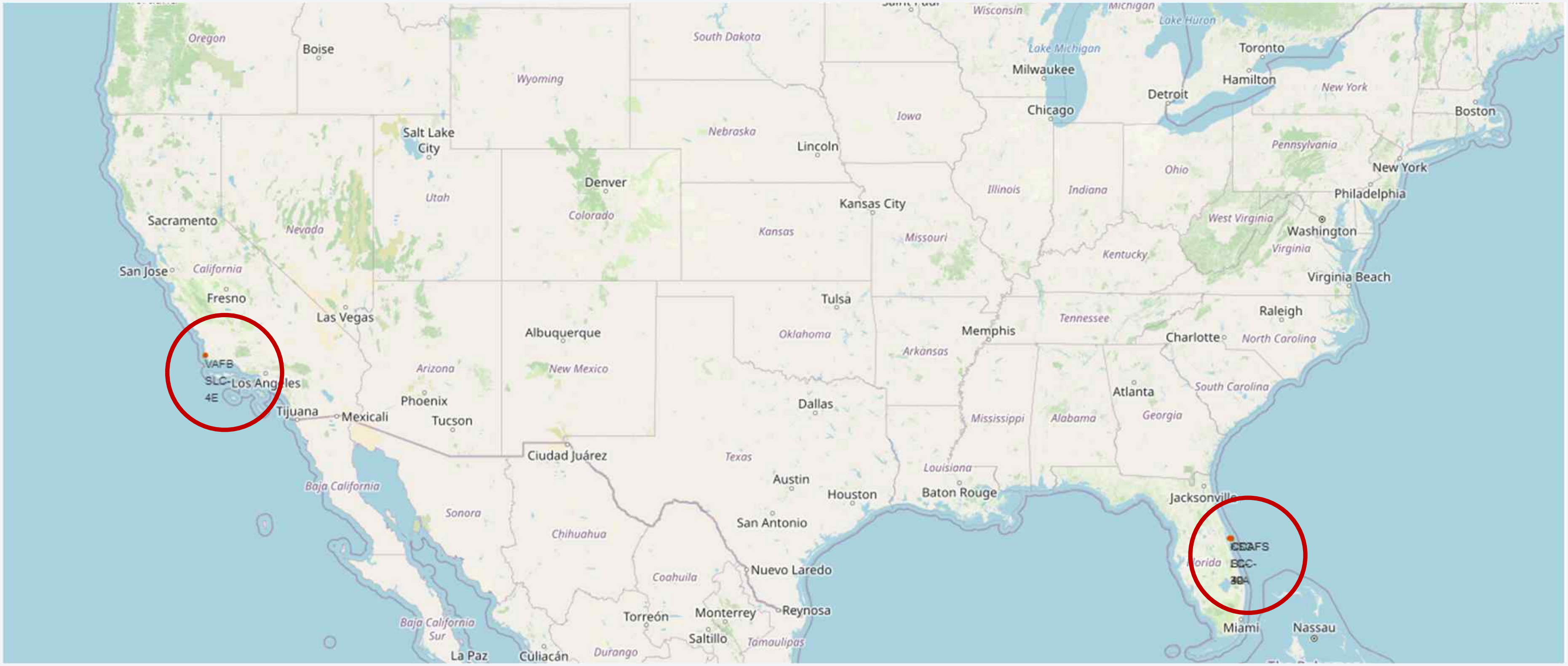
Landing_Outcome	Total Count
Success (ground pad)	3
Failure (drone ship)	5

- This query ranks the count of landing outcomes between the date 2010-06-04 and 2017-03-20 in a descending order. Here ‘select’, ‘where’, ‘between’, ‘group by’, ‘order by’ queries were used.

Section 3

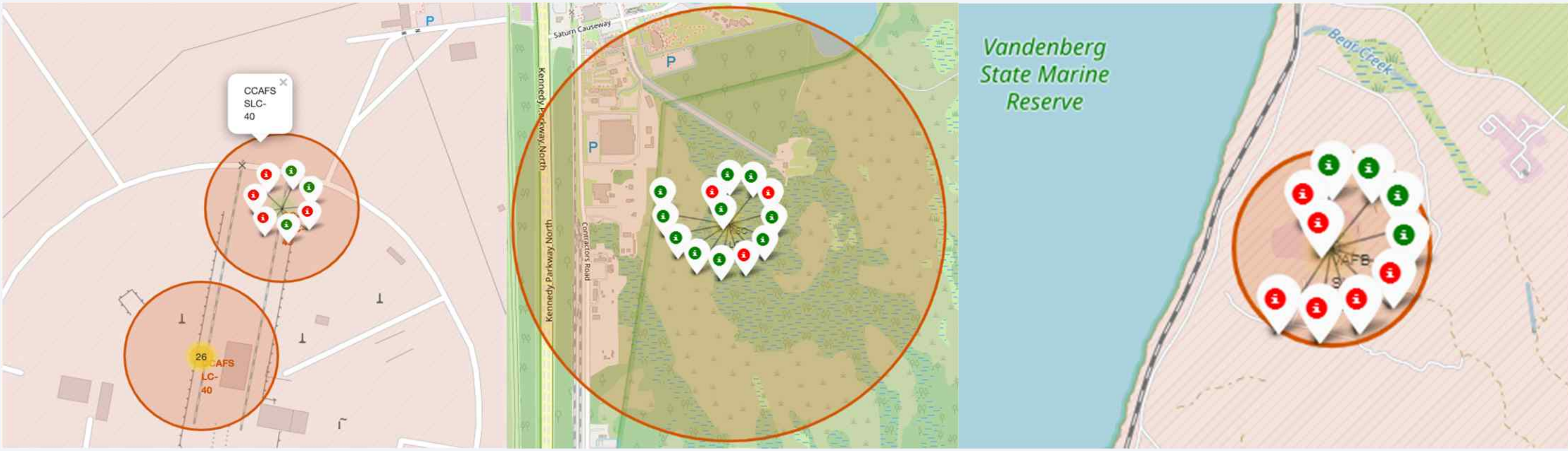
Launch Sites Proximities Analysis

Location of all launch sites



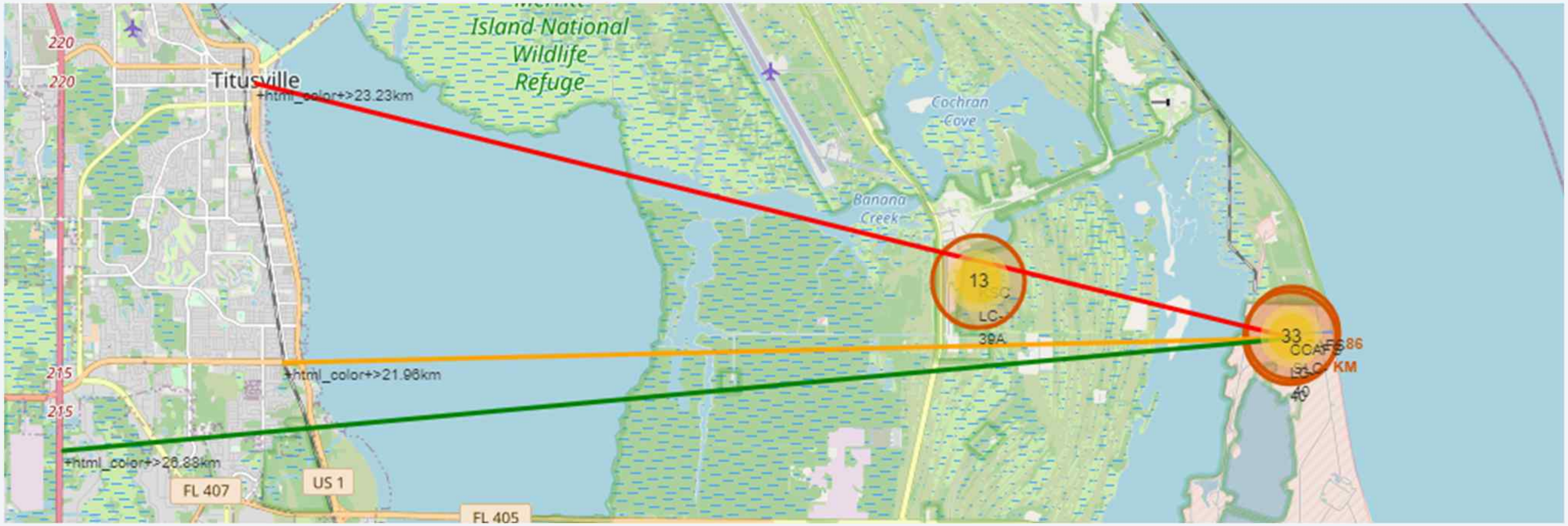
- The location of the launch sites are marked on the map as red dots.

Success and Failure cases for launch sites

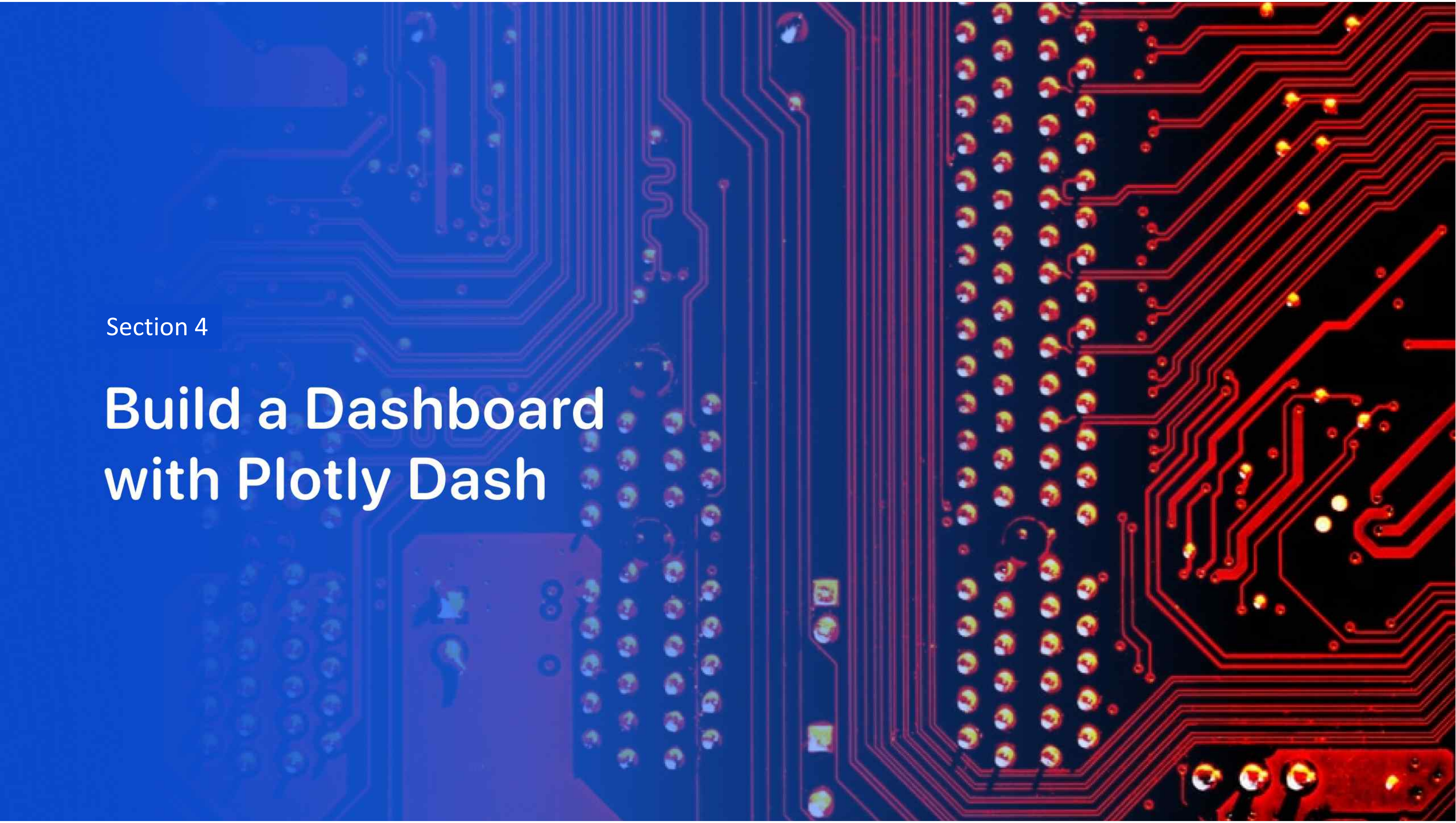


- The green markers show the launch cases which were successful, and the red markers were those that ended as a failure. We have results coming from 3 different launch sites.

Distance to important landmarks



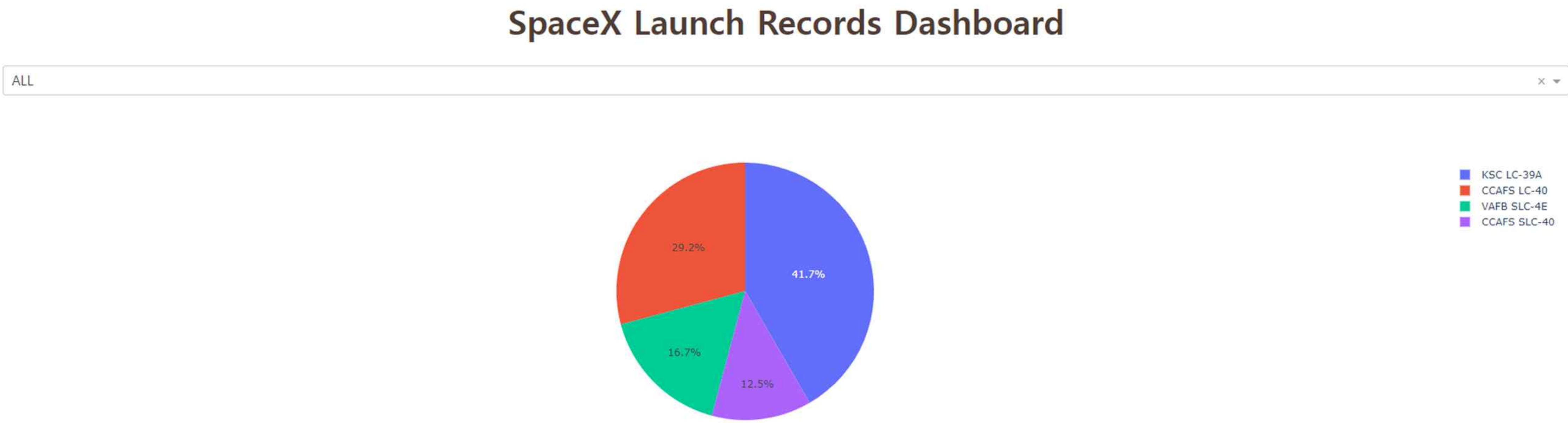
- Each line represents the distance from important landmarks. The green is the closest highway, red is closest city, yellow is closest railway and blue is closest coastline. It shows that the launch site is far away from important ground facilities and closer to the coastline.



Section 4

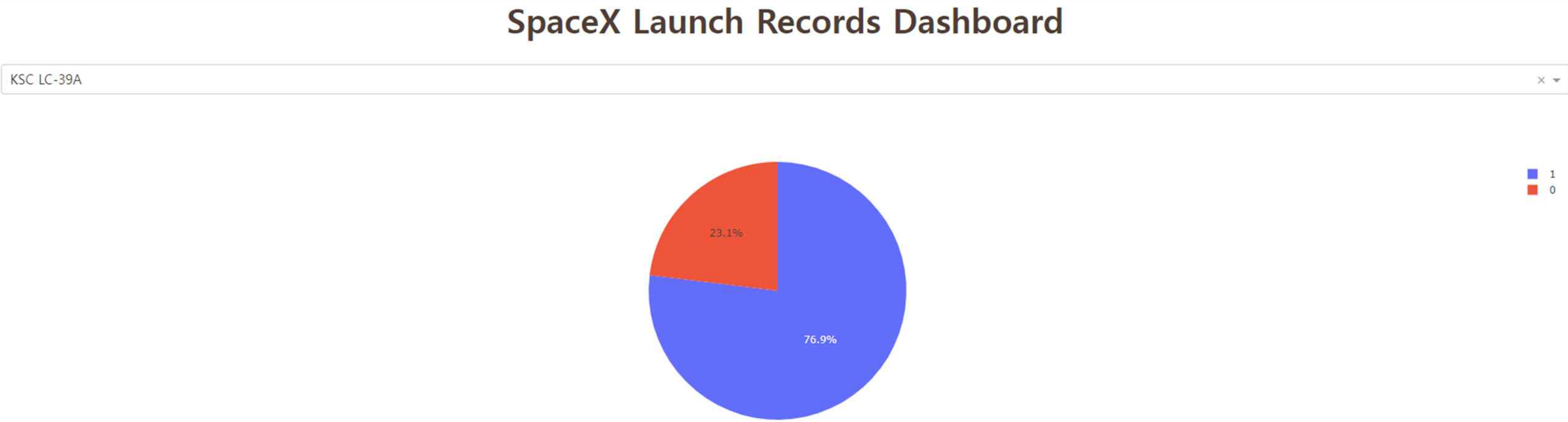
Build a Dashboard with Plotly Dash

Successful launch counts for all sites



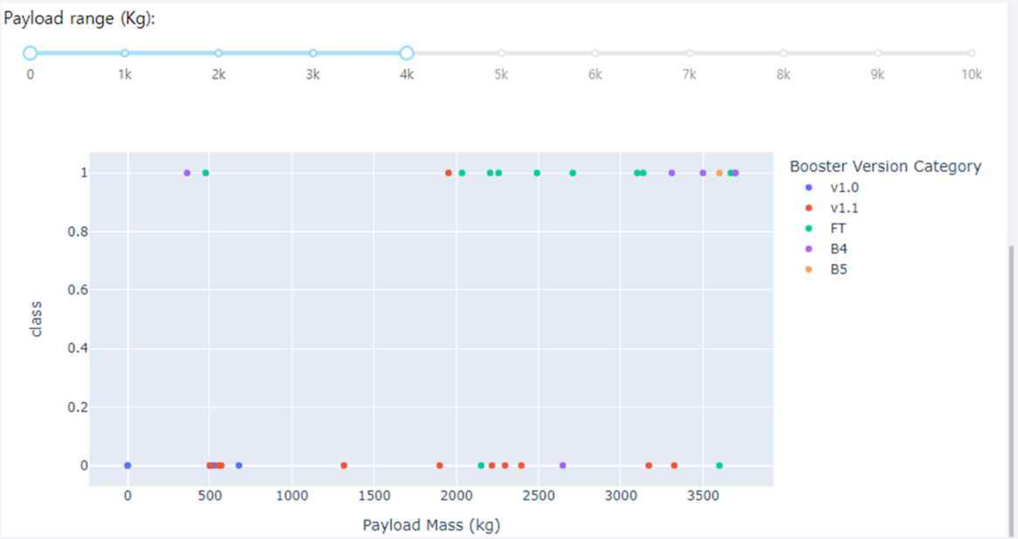
- This pie chart shows the success launch counts for each site. We can notice the KSC LC-39A had the most successful launches from all sites and CCAFS SLC-40 was the site least successful

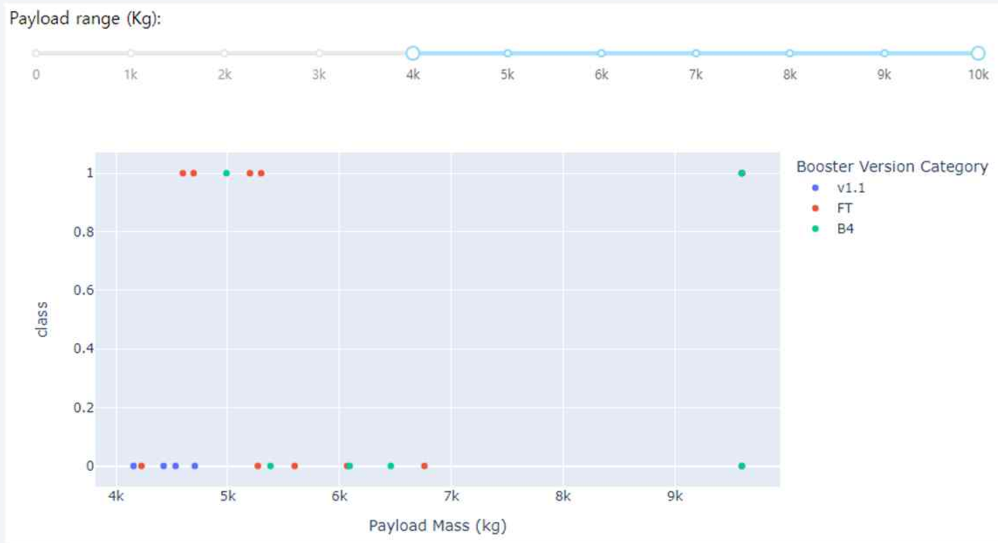
Highest Success Ratio: KSC LC-39A



- This pie chart shows the success rate of the model that shows the highest success ratio. The launch site ‘KSC LC-39A’ had 76.9% success ratio.

Success rate depending on payload





- This scatter chart shows 2 different ranges. The first one shows success rate of all booster versions which have payload between 0 to 4000kg, and the second one shows those that are higher than 4000kg. It shows that cases with higher payloads seems to be more unsuccessful

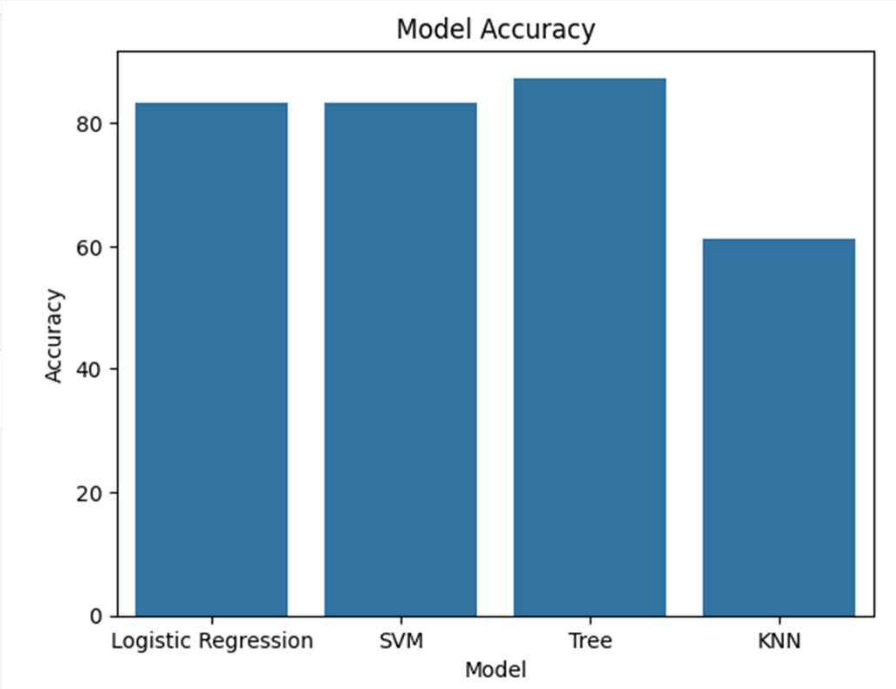
Section 5

Predictive Analysis (Classification)

Classification Accuracy per Model

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

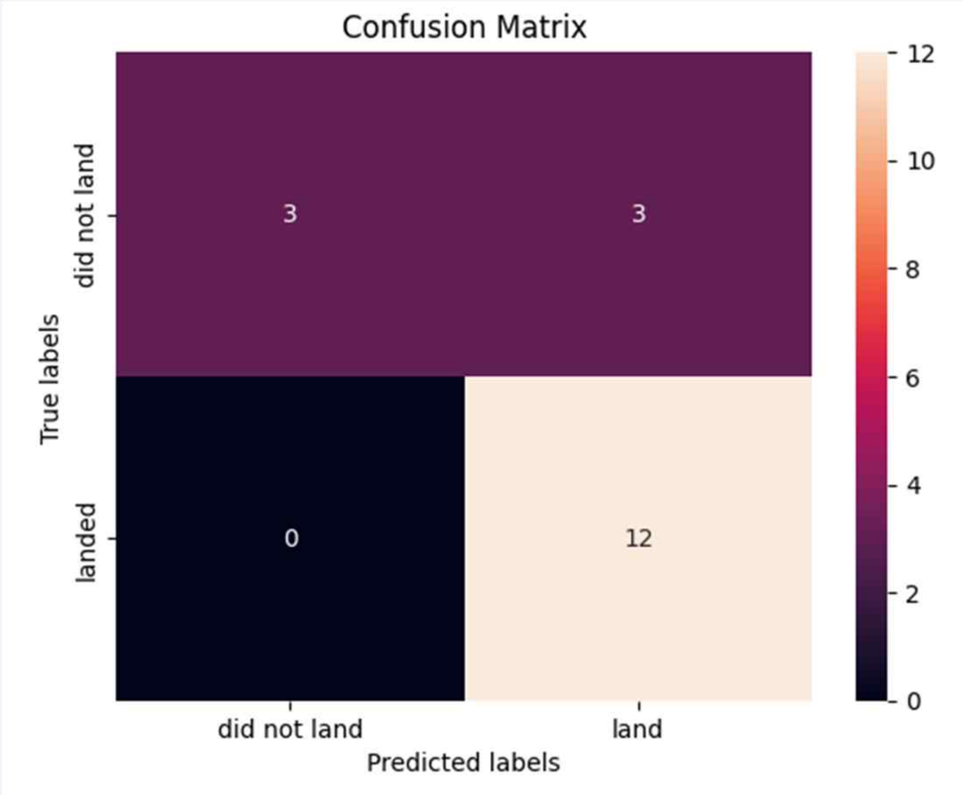
```
Best Algorithm is Tree with a score of 0.8732142857142857
Best Params is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
```



- This bar chart shows that the model using 'Tree' shows the highest accuracy. The accuracy score was 0.8732.

Confusion Matrix of Best Model

- This is the confusion matrix of the best model (Tree). It shows that True negative is 3, True positive is 12, False positive is 3 and false negative is 0.



Conclusions

- Launch sites are carefully chosen by considering the location, distance from major facilities and coastline.
- The success rate of launches from different launch sites vary. The launch site with the highest success rate was KSC LC-39A
- The overall success rates seems to increase year by year. Recently, the success rate is over 80%
- For some orbit types, the success rate was very high (100% for SSO)
- The low weighted payloads showed better success rate
- The Tree Classifier Algorithm showed best accuracy in prediction


Thank you!