IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Fernando Camargo>
<09-22-2024>

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context
  - 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. This goal of the project is to create a
  - machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers
  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
  - We use SQL to select and sort data

Methodology

Executive Summary

- Perform interactive visual analytics using Folium and Plotly Dash
  - Built an interactive dashboard containing pie charts and scatter plots to analyze data using the Plotly Dash Python library. Calculated distances on an interactive map by writing Python code with the Folium library.
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.
- The link to the notebook is https://github.com/FernandoC31/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/1.%20Data%20Colection%20API.ipynb

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

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

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork
```

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

```
response.status_code
```

```
200
```

```
json_data =response.json()
```

```
data = pd.json_normalize(json_data)
```

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is https://github.com/FernandoC31/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/1.%20Data%20Colection%20API.ipynb

```
url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy
```

```
response = requests.get(url)
response.status_code
```

```
200
```

```
html_content = response.text
soup = BeautifulSoup(html_content, 'html5lib')
```

```
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

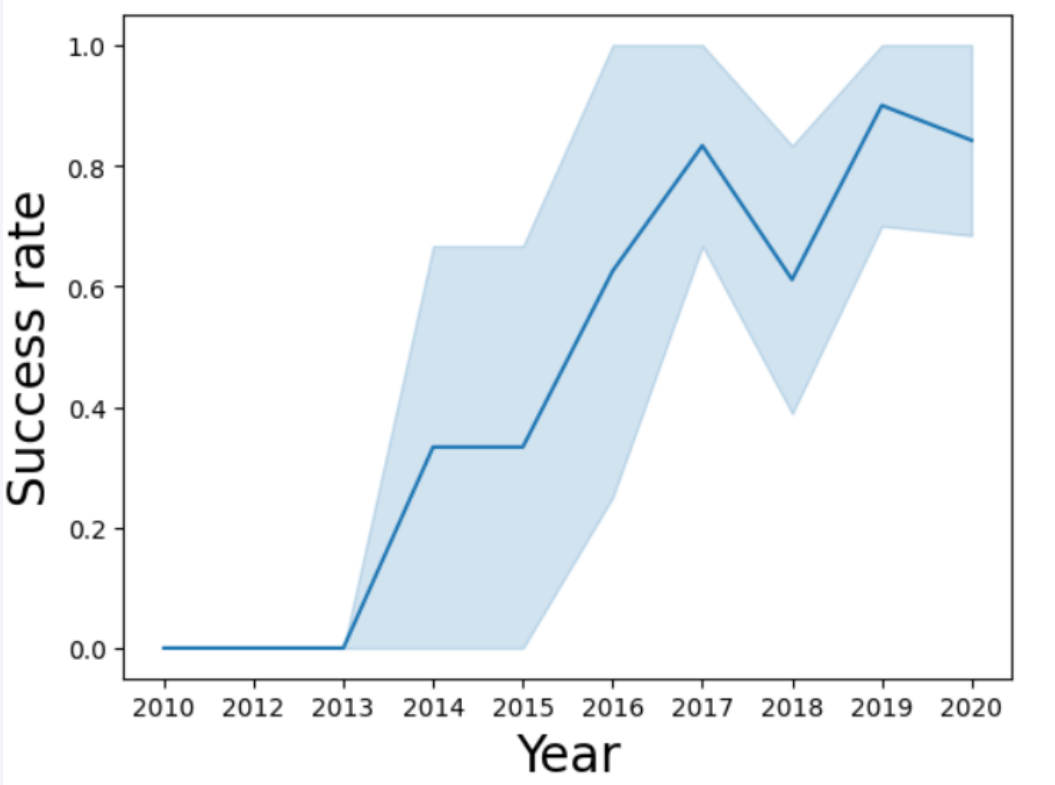
```
for col in first_launch_table.find_all('th'):
    names = extract_column_from_header(col)
    if names is not None and len(names)>0:
            column_names.append(names)
```

Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/FernandoC31/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/2.%20Exploratory%20Analysis%20SQL.ipynb

EDA with Data Visualization

- We explored the data by visualizing the relationship between flight
- number and launch Site, payload and launch site, success rate of each orbit
- type, flight number and orbit type, the launch success yearly trend.
- The link to the notebook is https://github.com/FernandoC31/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/3.%20Exploratory%20Analysis%20Matplotlib.ipynb



EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook is https://github.com/FernandoC31/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/2.%20Exploratory%20Analysis%20SQL.ipynb

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities.

Build a Dashboard with Plotly Dash

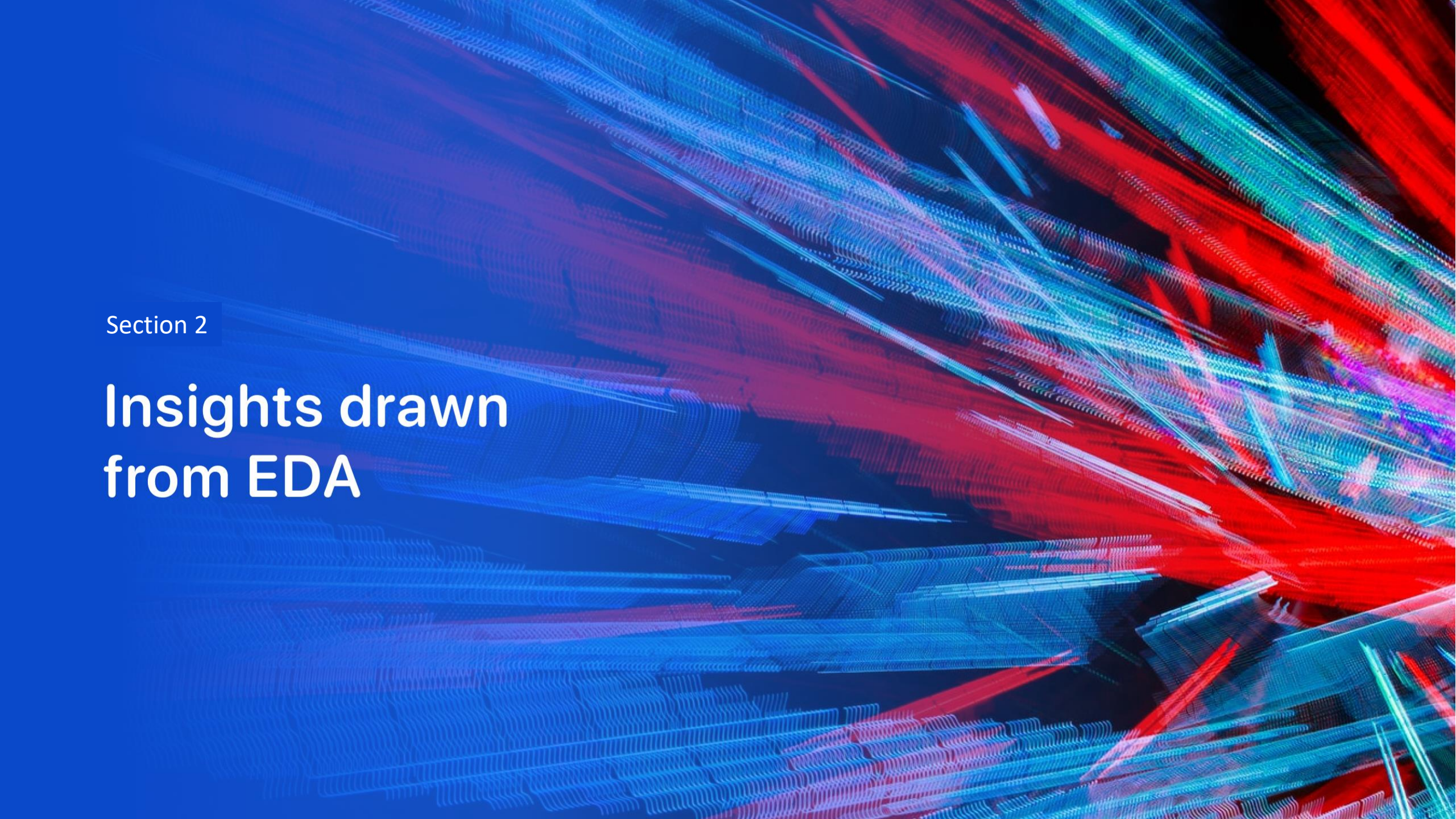
- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The link to the notebook is https://github.com/FernandoC31/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/5.%20Dashboard.ipynb

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is https://github.com/FernandoC31/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/6.%20ML%20Predictions.ipynb

Results

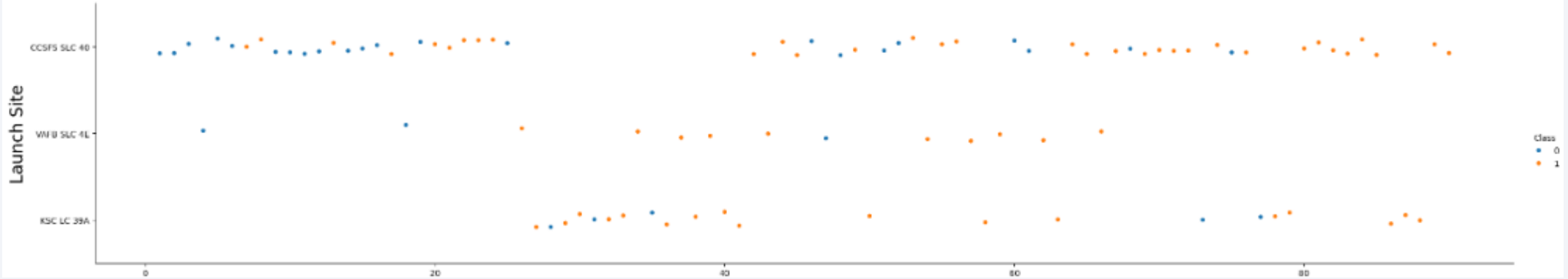
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.

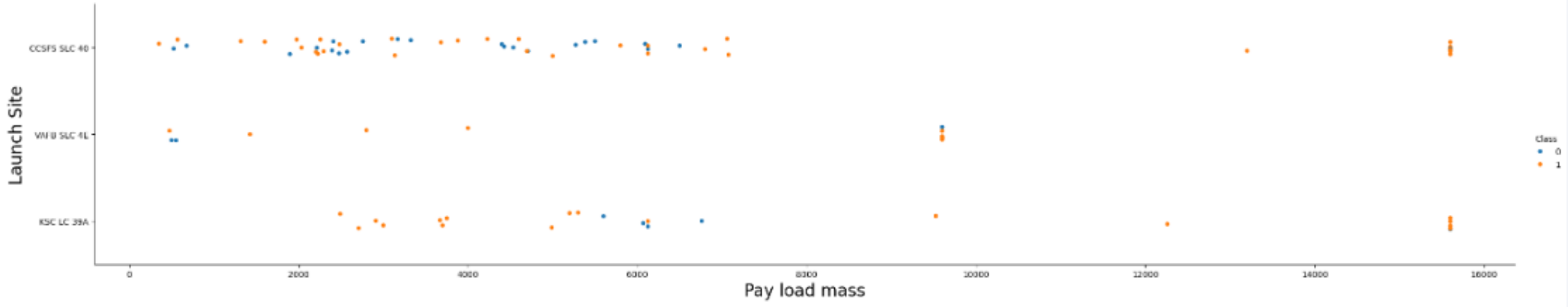
```
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Payload vs. Launch Site

- The greater payload mass for launch is CCAFS SLC 40

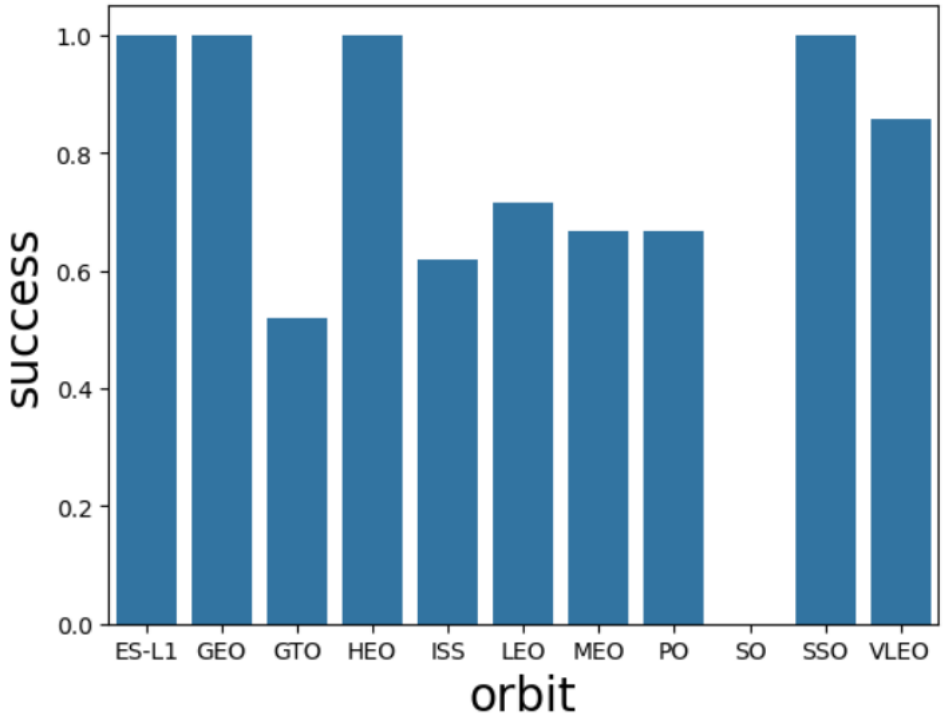
```
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load mass",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.

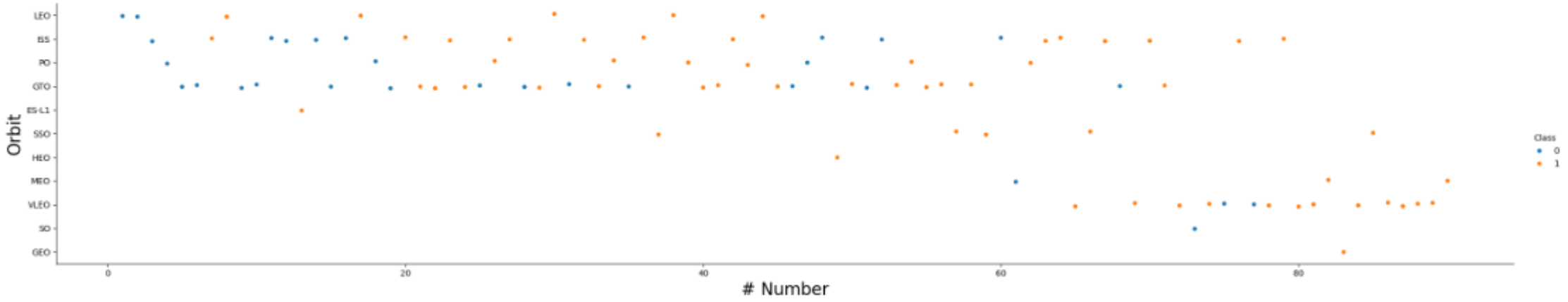
```
sns.barplot(y="Class", x="Orbit", data=plot_df)
plt.xlabel("orbit",fontsize=20)
plt.ylabel("success",fontsize=20)
plt.show()
```



Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

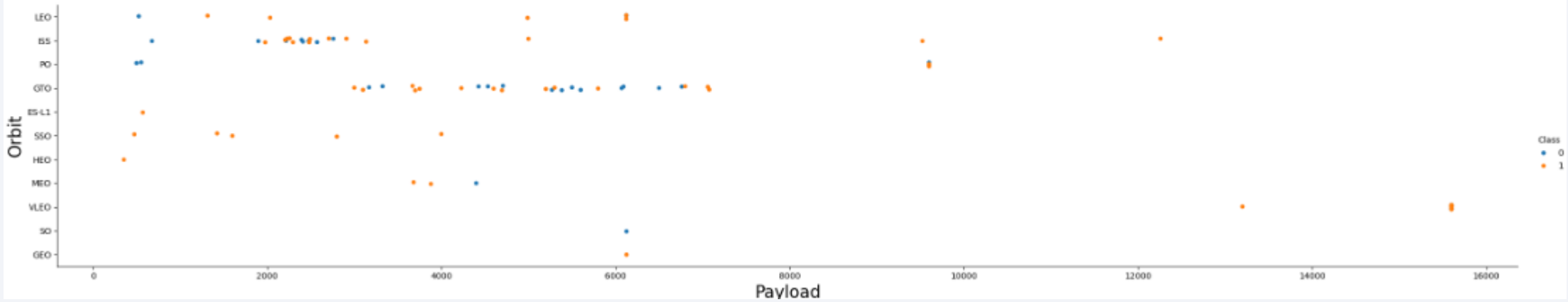
```
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("# Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.

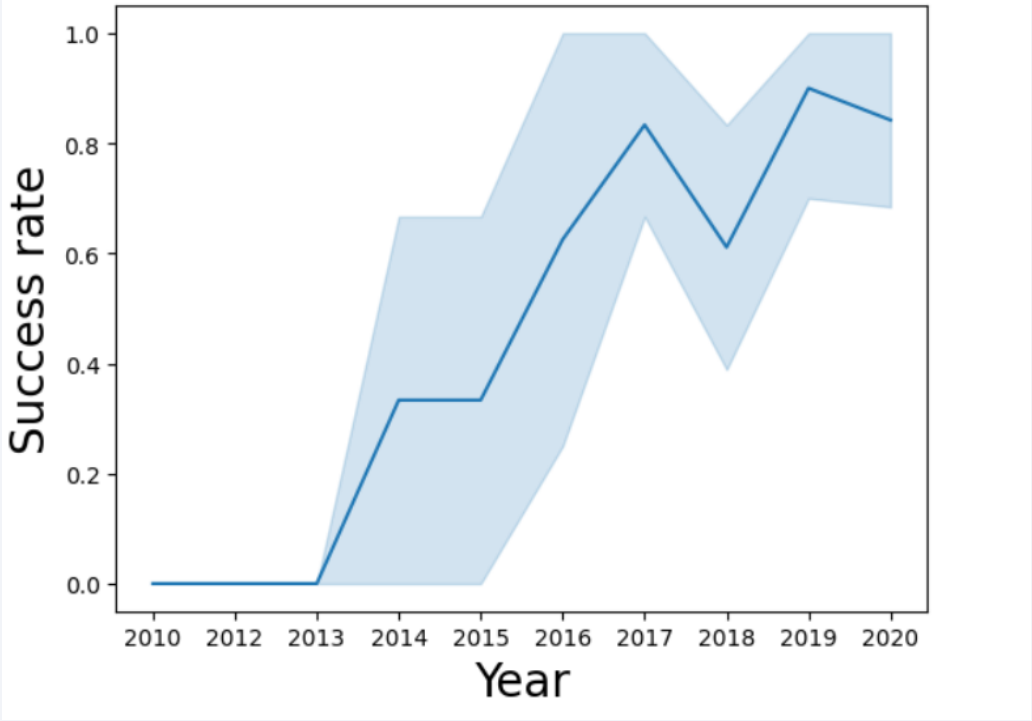
```
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.

```
sns.lineplot(y="Class", x="Date", data=df)
plt.xlabel("Year",fontsize=20)
plt.ylabel("Success rate",fontsize=20)
plt.show()
```



All Launch Site Names

- We used the key word group by to show launch sites from the SpaceX data.

```
%%sql
SELECT LAUNCH_SITE FROM SPACEXTBL GROUP BY LAUNCH_SITE
```

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

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

Launch Site Names Begin with 'CCA'

- We used the query above to display 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
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

We calculated the total payload carried by boosters from NASA as 48213 using the query below

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL
WHERE CUSTOMER LIKE '%CRS%'
```

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

SUM(PAYLOAD_MASS__KG_)
48213

Average Payload Mass by F9 v1.1

We calculated the average payload mass carried by booster version F9 v1.1 as 2534.66

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL
WHERE Booster_Version LIKE '%F9 v1.1%'
```

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

AVG(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

```
%%sql
SELECT MIN(DATE) FROM SPACEXTBL
WHERE Landing_Outcome LIKE '%Success%'
```

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

MIN(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000

```
%%sql
SELECT Booster_Version FROM SPACEXTBL
WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

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

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

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like ‘%’ to filter for WHERE MissionOutcome was a success or a failure.

```
%%sql
SELECT
    COUNT(CASE WHEN Landing_Outcome LIKE '%Failure%' THEN 1 END) AS Failure_Count,
    COUNT(CASE WHEN Landing_Outcome LIKE '%Success%' THEN 1 END) AS Success_Count
FROM SPACEXTBL;
```

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

Failure_Count	Success_Count
10	61

Boosters Carried Maximum Payload

- We determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.

```
%%sql
SELECT Booster_Version FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL
)
```

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

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

2015 Launch Records

- We used a combinations of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

```
%%sql
SELECT SUBSTR(Date, 6,2) AS MONTH, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL
WHERE Landing_Outcome LIKE '%drone ship%'
AND  SUBSTR(Date,0,5) = '2015'
```

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

MONTH	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40
06	Precluded (drone ship)	F9 v1.1 B1018	CCAFS LC-40

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

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20.
- We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order

```
%%sql
SELECT Landing_Outcome, COUNT(Landing_Outcome) FROM SPACEXTBL
WHERE DATE > '2010-06-04' AND DATE <'2017-03-20'
GROUP BY Landing_Outcome
ORDER BY COUNT(Landing_Outcome) DESC
```

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

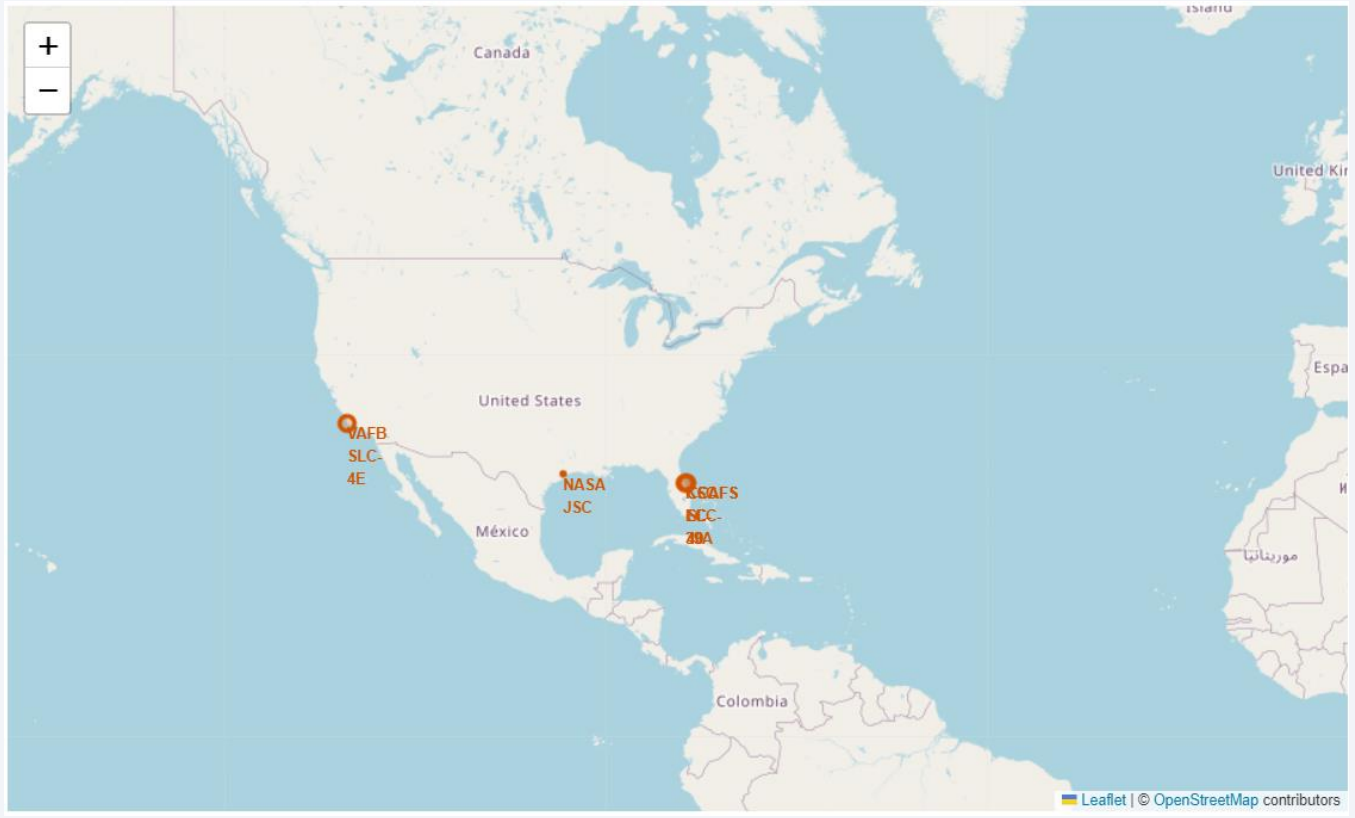
Landing_Outcome	COUNT(Landing_Outcome)
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Section 3

Launch Sites Proximities Analysis

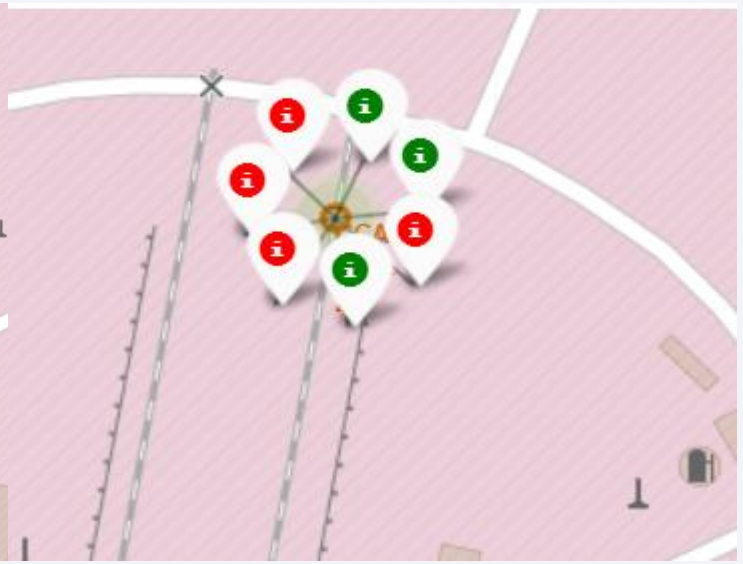
<Folium Map Screenshot 1>

- We can see al the launch sites are in the USA near the coast of Florida and California



<Folium Map Screenshot 2>

- In this section we can see all the failure(red) and success(green) launches

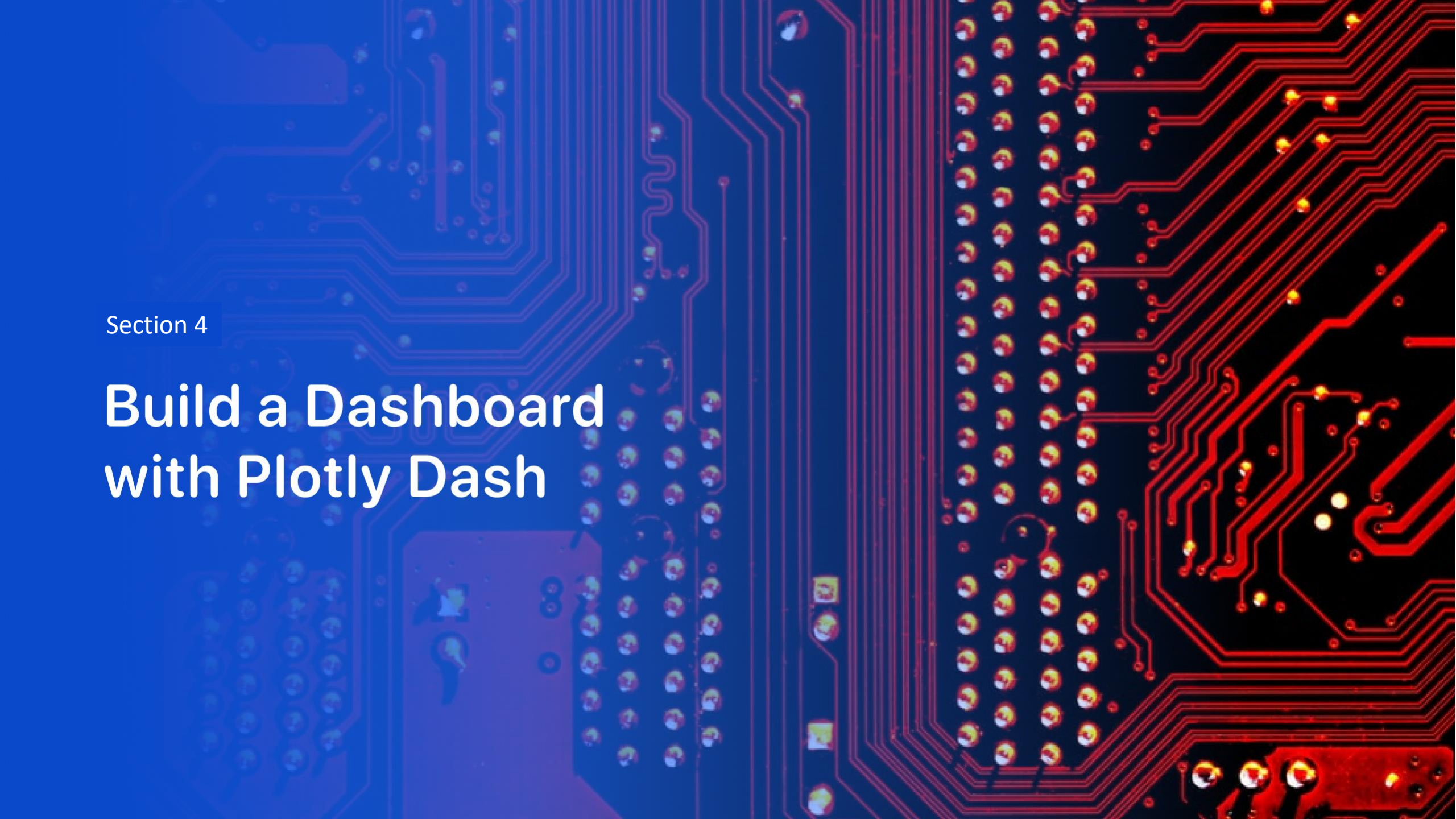


<Folium Map Screenshot 3>

```
from math import sin, cos, sqrt, atan2, radians

def calculate_distance(lat1, lon1, lat2, lon2):

    R = 6373.0

    lat1 = radians(lat1)
    lon1 = radians(lon1)
    lat2 = radians(lat2)
    lon2 = radians(lon2)

    dlon = lon2 - lon1
    dlat = lat2 - lat1

    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    distance = R * c
    return distance
```

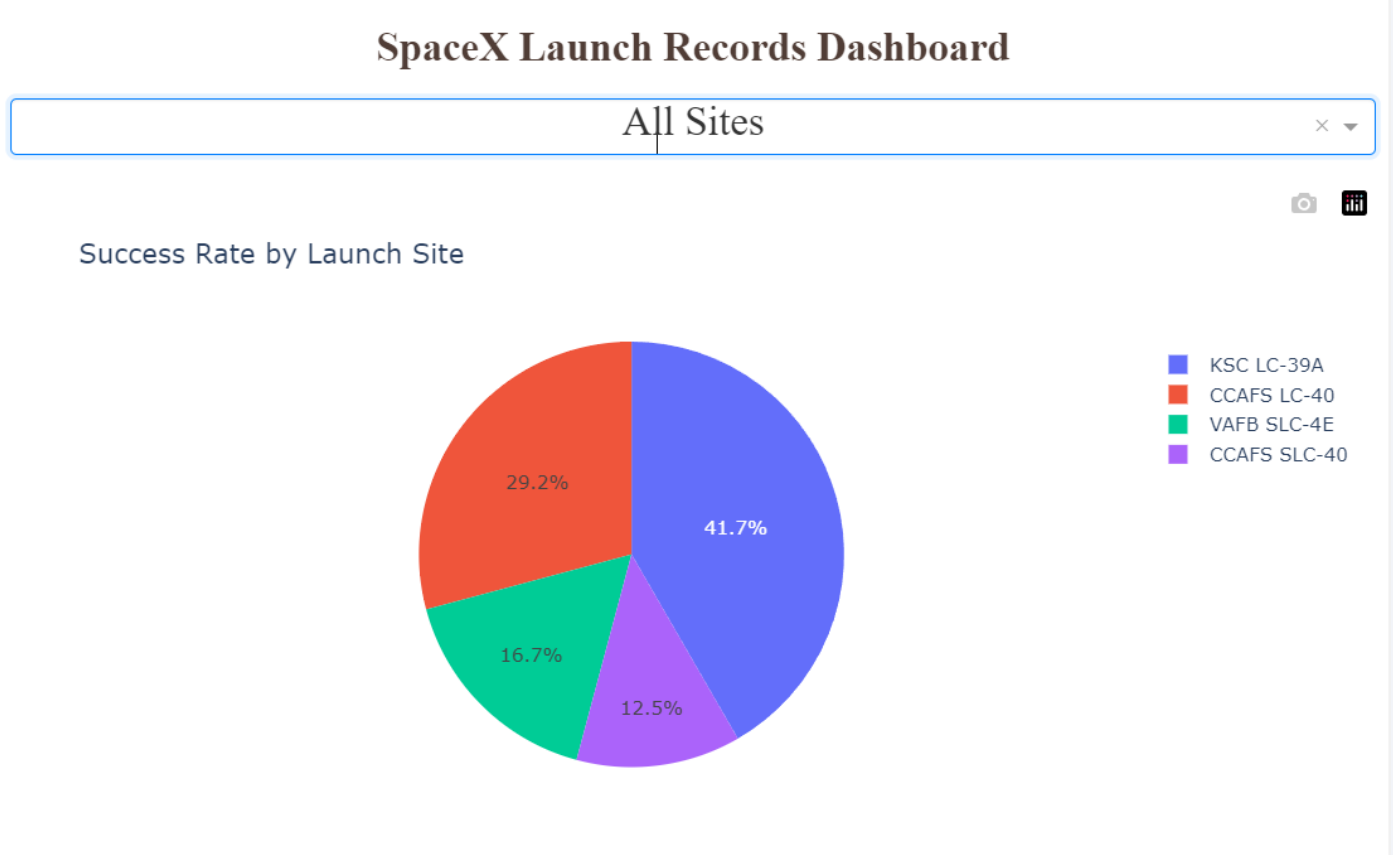
```
distance_coastline = calculate_distance(28.573255, 80.646895, 28.56276, -80.5)
distance_coastline
```

```
13359.059343152654
```


Section 4
Build a Dashboard with Plotly Dash

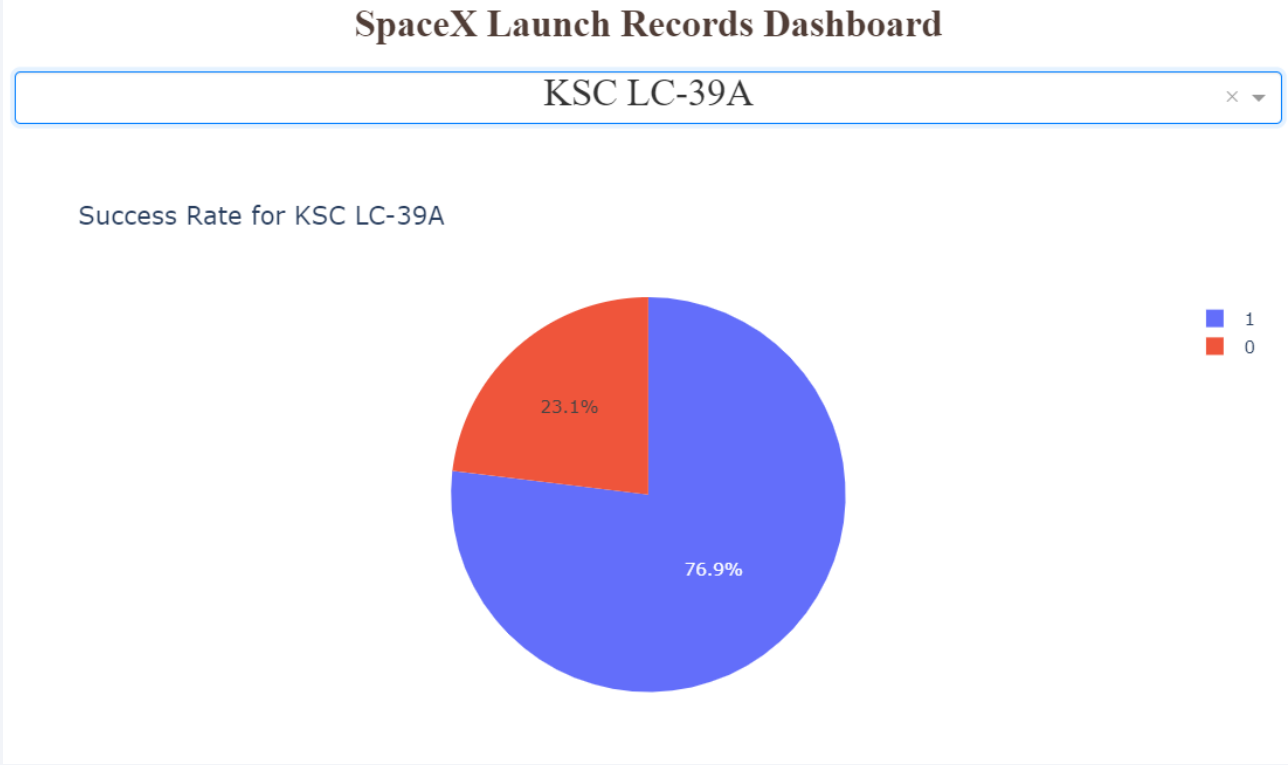
<Dashboard Screenshot 1>

- In the graphic, we can see that KSC LC 39A had the most successful launches from all the sites



<Dashboard Screenshot 2>

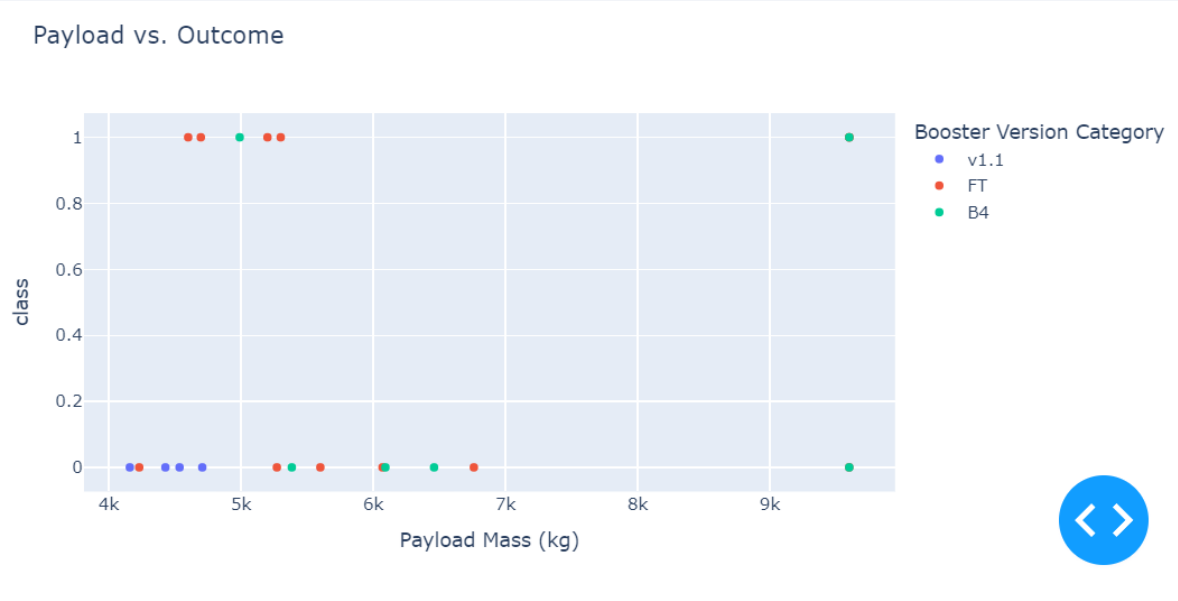
- The KSC LC 39A has 76.9% success rate that is pretty impresive



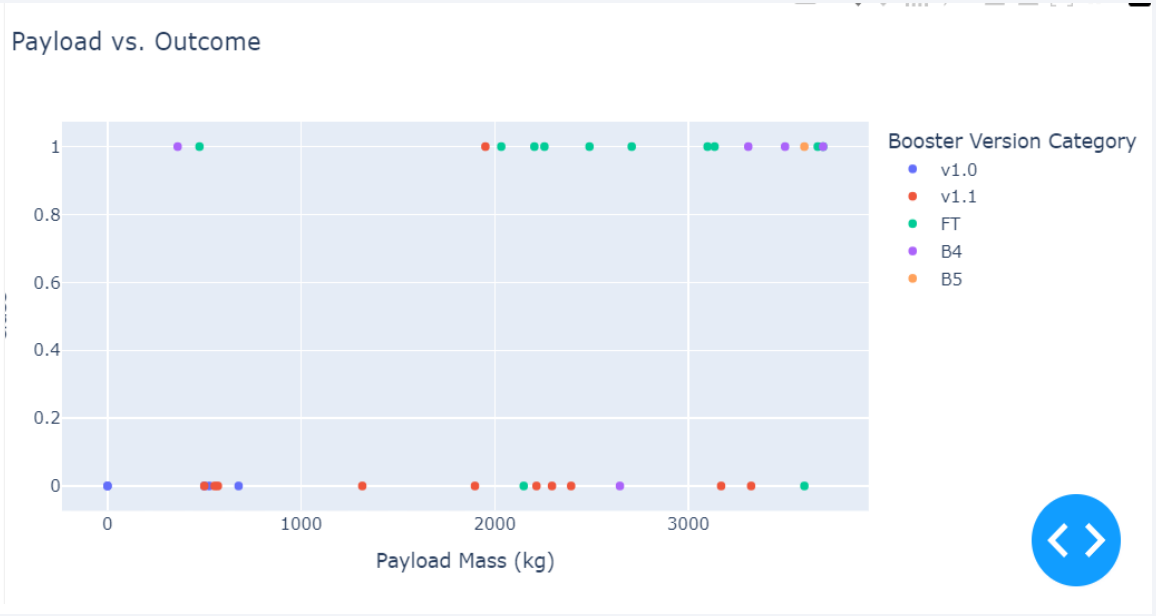
<Dashboard Screenshot 3>

The importance of have a dashboard is that you can select diferente types of range or class to analyse the data

Heavy Weight Payload Mass



Low Weight Payload Mass



Section 5

Predictive Analysis (Classification)

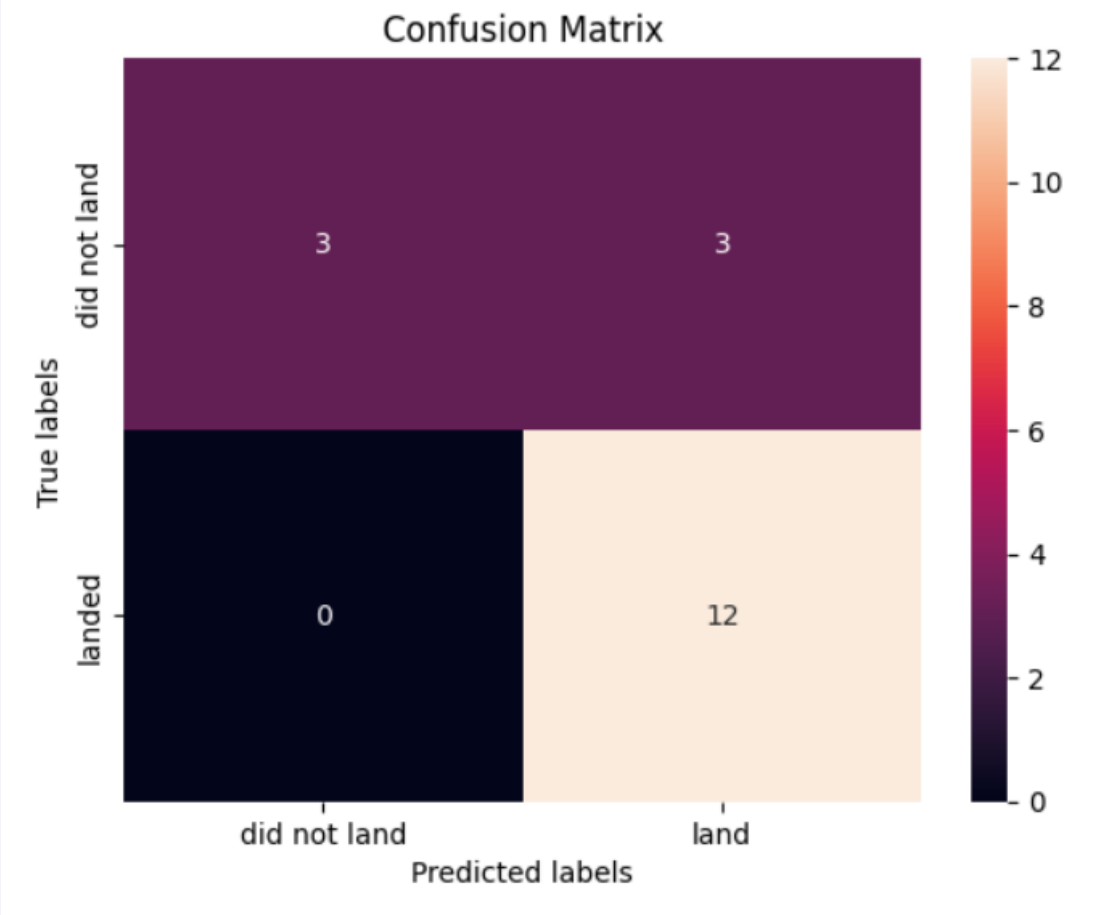
Classification Accuracy

```
logreg_accuracy = test_accuracy
svm_accuracy = test_accuracy_svm
tree_accuracy = test_accuracy_tree
knn_accuracy = test_accuracy_knn
model_accuracies = {
    'Logistic Regression': logreg_accuracy,
    'SVM': svm_accuracy,
    'Decision Tree': tree_accuracy,
    'KNN': knn_accuracy
}

best_model = max(model_accuracies, key=model_accuracies.get)
best_accuracy = model_accuracies[best_model]
print(f"The best model is {best_model} with an accuracy of {best_accuracy:.4f}.")
```

```
The best model is Logistic Regression with an accuracy of 0.8333.
```

Confusion Matrix

The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier



Conclusions

- We can conclude that:
- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Logistic regression classifier is the best machine learning algorithm for this task.

Thank you!