IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Quy Thi
19th Jan 2024

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- In this capstone, we will predict if the Falcon 9 first stage will land successfully. SpaceX 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 SpaceX can reuse the first stage.

- Therefore if we can determine if the first stage will land, we can determine the cost of a launch.

Section 1

Methodology

Methodology

Executive Summary

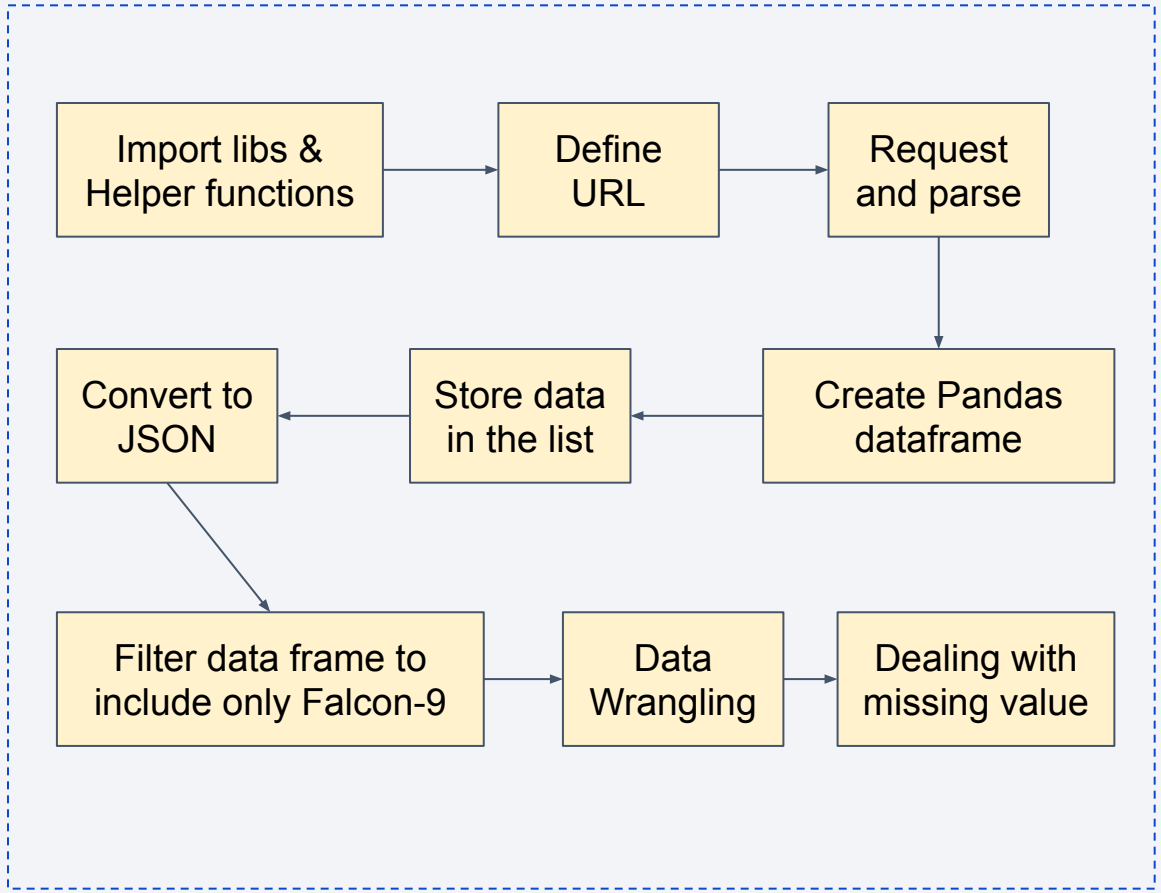
- Data collection methodology:
  - Request to the SpaceX API
  - Extract a Falcon 9 launch records HTML table from Wikipedia
- Perform data wrangling
  - Here are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident. So I convert those outcomes into Training Labels with `1` means the booster successfully landed `0` means it was unsuccessful.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Find best Hyperparameter for SVM, Classification Trees and Logistic Regression

Data Collection

- Request to the SpaceX API
- Extract a Falcon 9 launch records HTML table from Wikipedia

Data Collection – SpaceX API

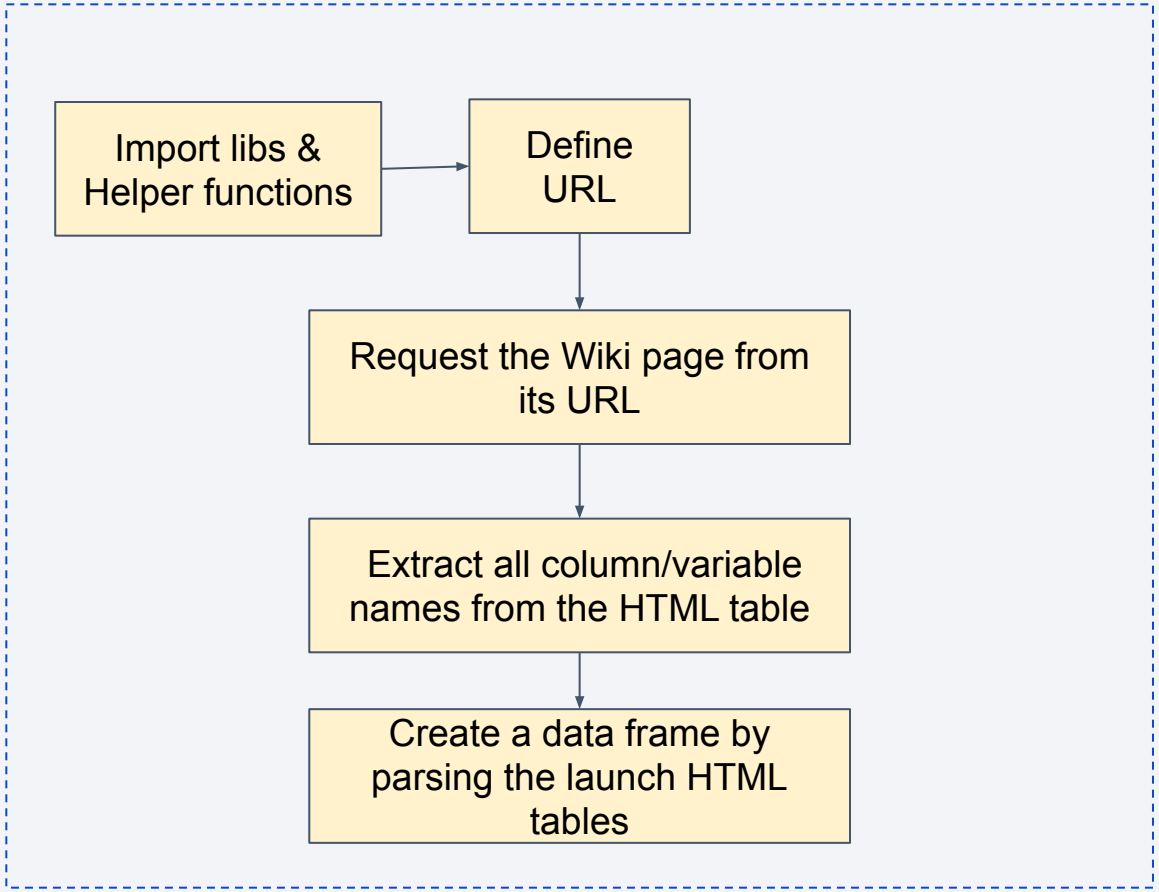
- Present your data collection with SpaceX REST calls using key phrases and flowcharts
- https://github.com/quy-thi/DS-Final/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

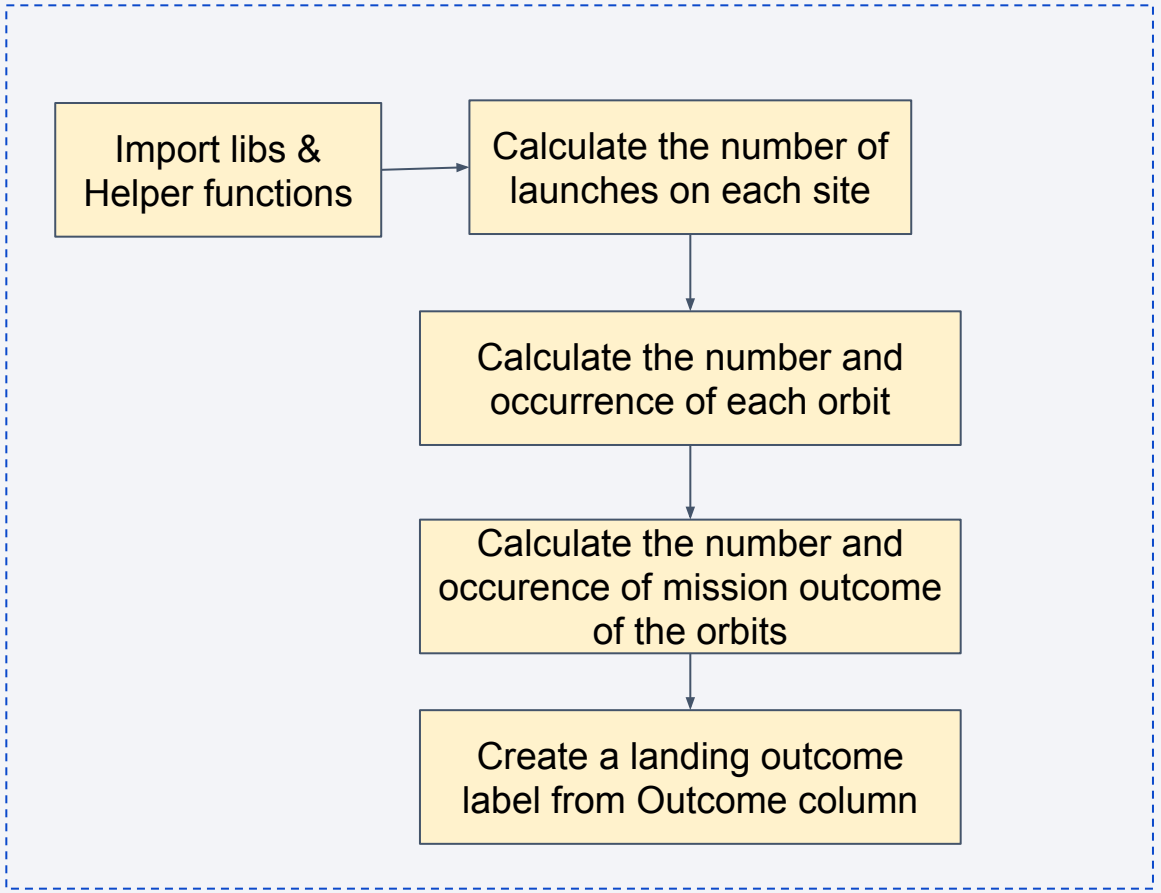
- https://github.com/quy-thi/DS-Final/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

- Mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

- https://github.com/quy-thi/DS-Final/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- EDA with SQL
- Build an Interactive Map with Folium
- Build a Dashboard with Plotly Dash

EDA with SQL

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass
- 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.
- 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

- https://github.com/quy-thi/DS-Final/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Visualize the relationship between Flight Number and Launch Site using catplot
- Visualize the relationship between Payload and Launch Site using scatterplot
- Visualize the relationship between success rate of each orbit type using barplot
- Visualize the relationship between FlightNumber and Orbit type using scatterplot
- Visualize the relationship between Payload and Orbit type using catplot
- Visualize the launch success yearly trend using lineplot

- https://github.com/quy-thi/DS-Final/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

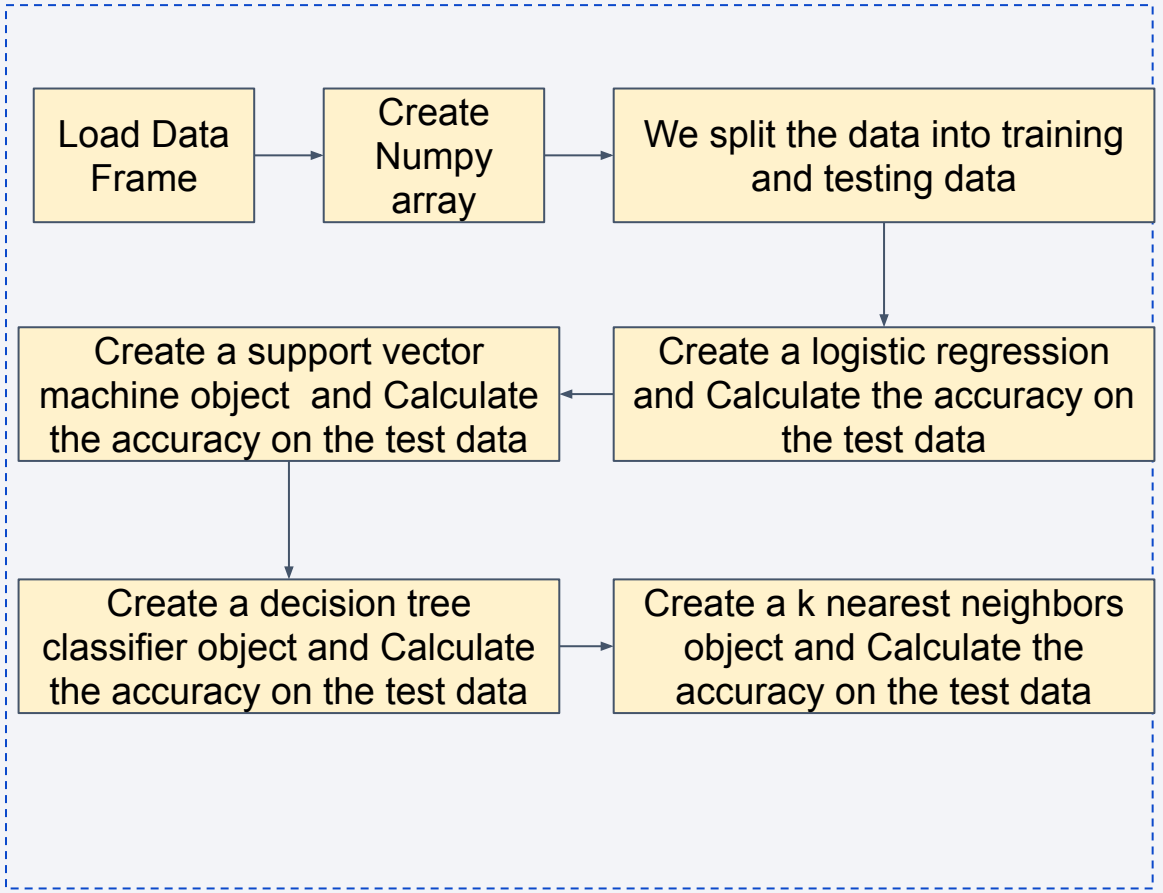
Build a Dashboard with Plotly Dash

- Folium Map object, with an initial center location to be NASA Johnson Space Center at Houston, Texas.
- Start location is NASA Johnson Space Center
- We can answer these questions
    - Are all launch sites in proximity to the Equator line?
    - Are all launch sites in very close proximity to the coast?
- Calculate the distances between a launch site to its proximities
- Draw a PolyLine between a launch site to the selected coastline point
- Similarly, you can draw a line betwee a launch site to its closest city, railway, highway

- https://github.com/quy-thi/DS-Final/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

Predictive Analysis (Classification)

- The best performance is KNN but when we tuned hpyer parameters then the accuracy of decision tree classifier is better
- https://github.com/quy-thi/DS-Final/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

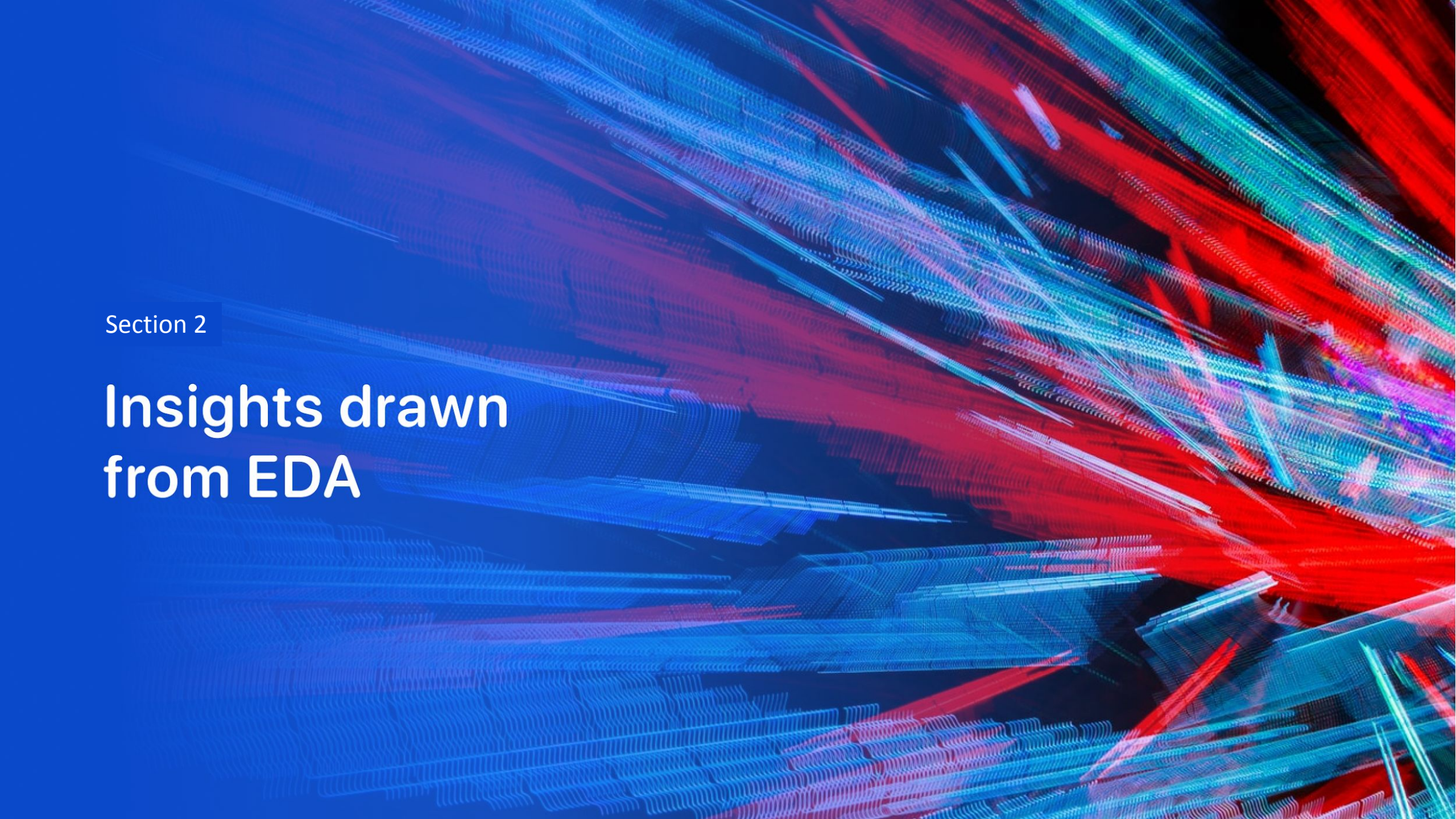


Results

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

Section 2

Insights drawn from EDA



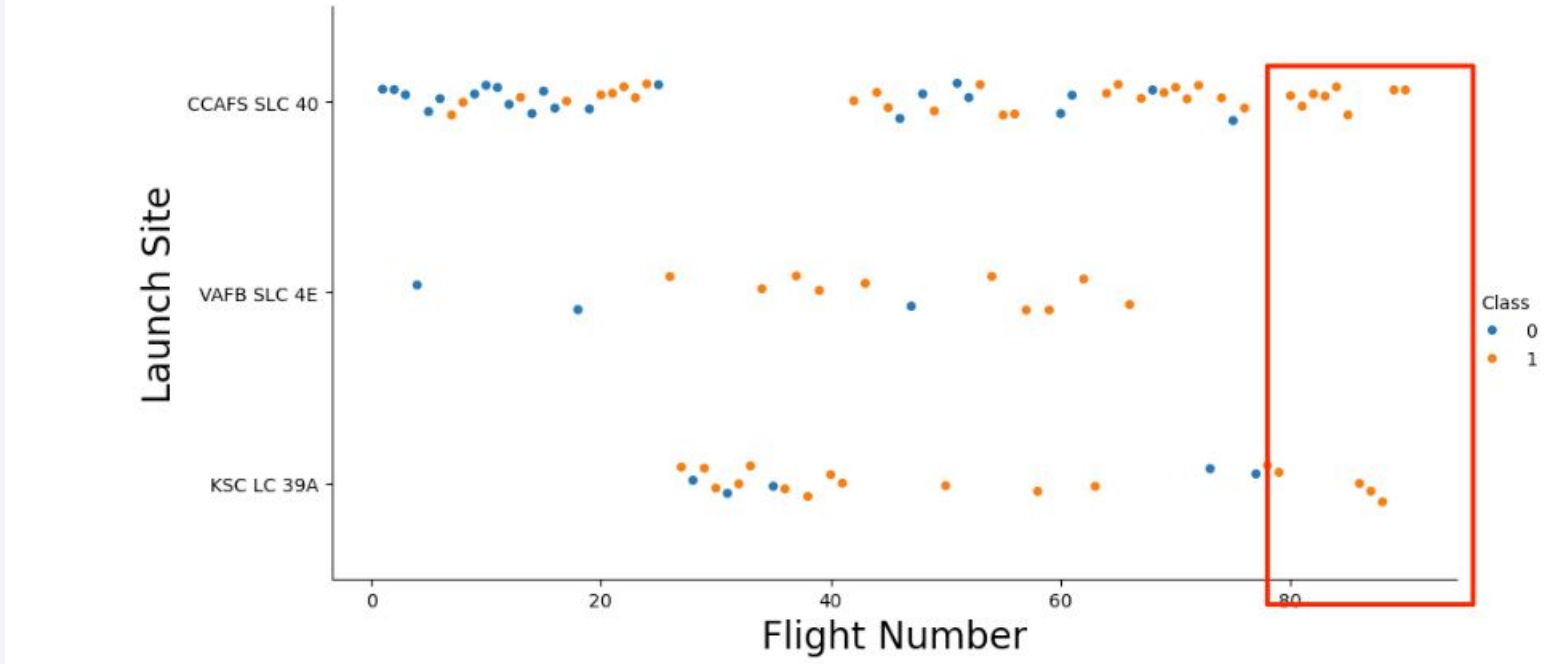
Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site

- If the Flight number is > 80 then the success rate is 100%

In [6]:

```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect=2)
plt.xlabel("Flight Number", fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```

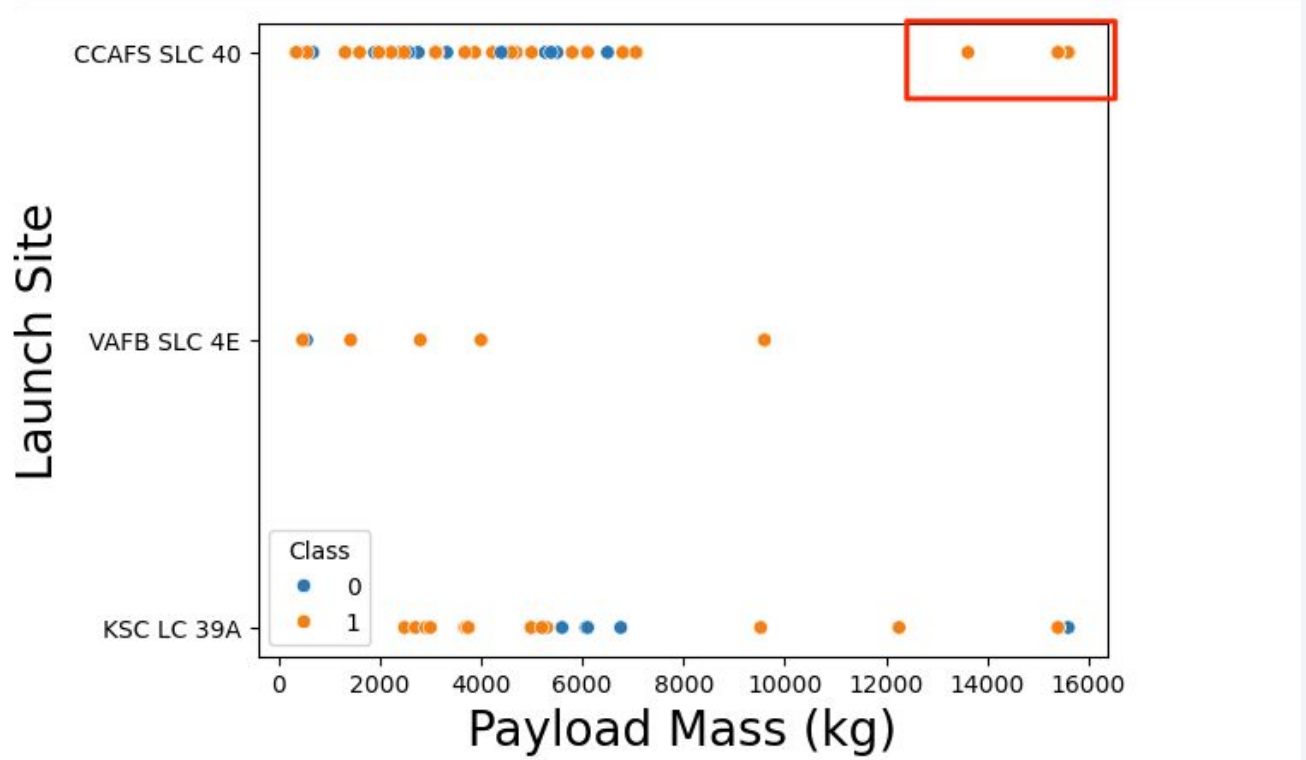


Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site

- At CCAFS SLC 40 if the payload mass > 10000kg then the success rate = 100%

```
### TASK 2: Visualize the relationship between Payload and Launch Site
sns.scatterplot(x="PayloadMass", y="LaunchSite", hue="Class", data=df)
plt.xlabel("Payload Mass (kg)", fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```

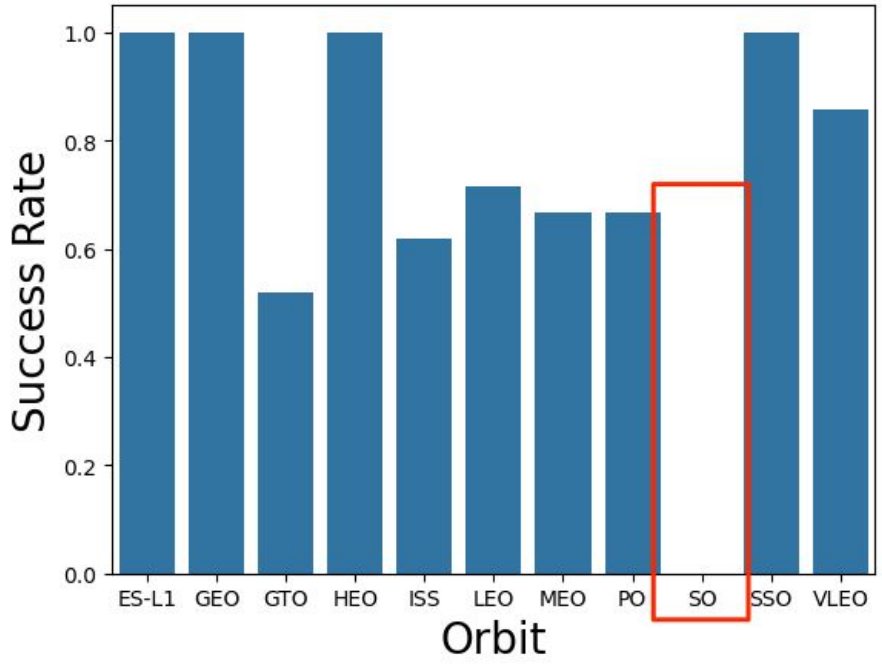


Success Rate vs. Orbit Type

- If Orbit = SO then success rate = 0

In [8]:

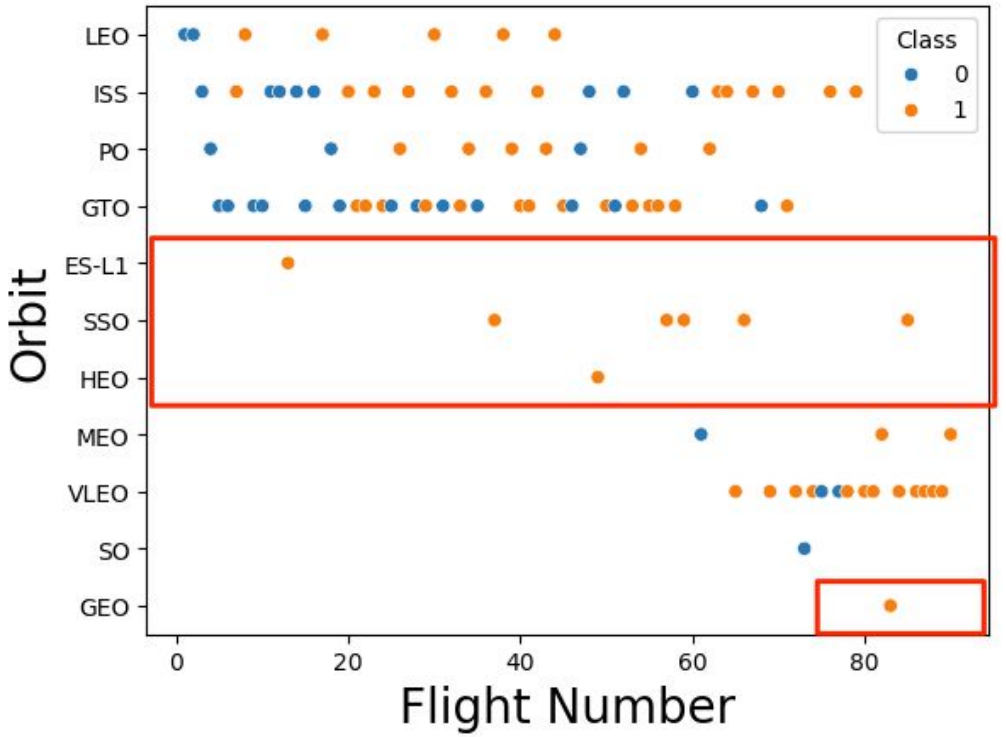
```
### TASK  3: Visualize the relationship between success rate of each orbit type
orbit_success_rate = df.groupby('Orbit')['Class'].mean().reset_index()
sns.barplot(x="Orbit", y="Class", data=orbit_success_rate)
plt.xlabel("Orbit", fontsize=20)
plt.ylabel("Success Rate", fontsize=20)
plt.show()
```



Flight Number vs. Orbit Type

- If orbit in GEO, SSO, HEO, ES-L1 have success rate = 100%

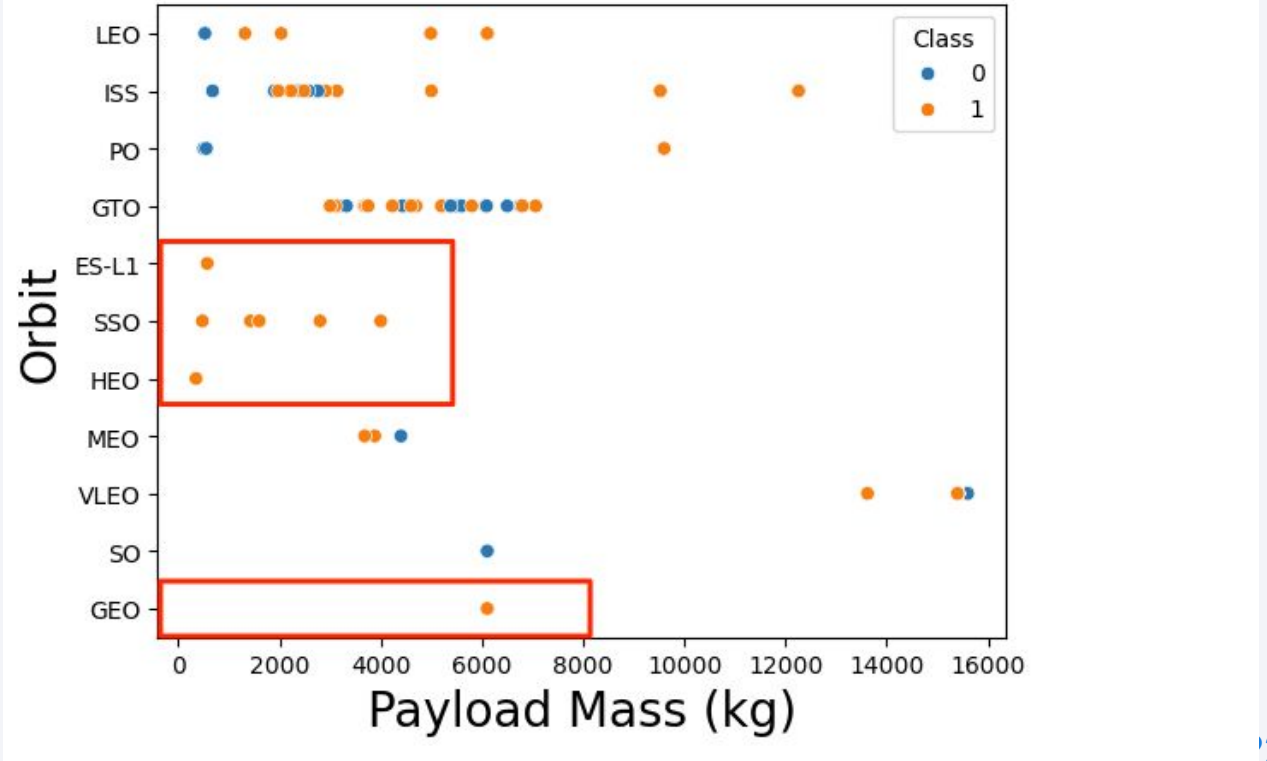
```
In [9]:
### TASK  4: Visualize the relationship between FlightNumber and Orbit type
sns.scatterplot(x="FlightNumber", y="Orbit", hue="Class", data=df)
plt.xlabel("Flight Number", fontsize=20)
plt.ylabel("Orbit", fontsize=20)
plt.show()
```



Payload vs. Orbit Type

- If Orbit in (ES-L1, SSO, HEO, GEO) and Payload Mass < 8000kg. Then success rate = 100%

```
### TASK  5: Visualize the relationship between Payload and Orbit type
sns.scatterplot(x="PayloadMass", y="Orbit", hue="Class", data=df)
plt.xlabel("Payload Mass (kg)", fontsize=20)
plt.ylabel("Orbit", fontsize=20)
plt.show()
```

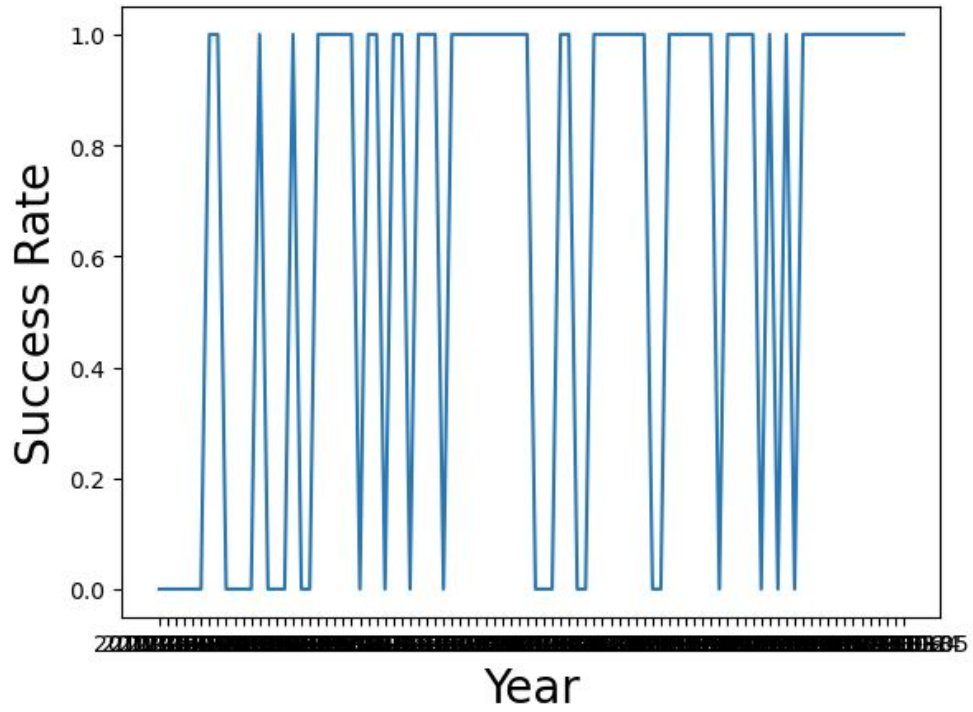


Launch Success Yearly Trend

- From here very hard to have any explanation

In [11]:

```
### TASK  6: Visualize the launch success yearly trend
# Assuming year extraction and Date column modification is already done as per your description
yearly_success_rate = df.groupby('Date')['Class'].mean().reset_index()
sns.lineplot(x="Date", y="Class", data=yearly_success_rate)
plt.xlabel("Year", fontsize=20)
plt.ylabel("Success Rate", fontsize=20)
plt.show()
```



All Launch Site Names

- Find the names of the unique launch sites
- 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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;

```
In [22]: %sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
 * sqlite:///my_data1.db
Done.
```

Out[22]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (p
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 (p
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	N
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	N
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	N

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- SELECT SUM("Payload_Mass__KG_") FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)';

```
In [12]: %sql SELECT SUM("Payload_Mass__KG_") FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)';
 * sqlite:///my_data1.db
Done.
Out[12]: SUM("Payload_Mass__KG_")
                            45596
```

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- SELECT AVG("Payload_Mass__KG_") FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1';

```
In [13]: %sql SELECT AVG("Payload_Mass__KG_") FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1';
 * sqlite:///my_data1.db
Done.
Out[13]: AVG("Payload_Mass__KG_")
         2928.4
```

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)'

```
In [24]: %sql SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)'
 * sqlite:///my_data1.db
Done.
Out[24]: 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
- SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "Payload_Mass__KG_" > 4000 AND "Payload_Mass__KG_" < 6000

```
In [25]: %sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "Payload_Mass
 * sqlite:///my_data1.db
Done.
```

Out[25]:

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- SELECT "Landing_Outcome", COUNT(*) FROM SPACEXTABLE GROUP BY "Landing__Outcome";

```
In [26]: %sql SELECT "Landing_Outcome", COUNT(*) FROM SPACEXTABLE GROUP BY "Landing__Outcome";
 * sqlite:///my_data1.db
Done.
```

Out[26]:

Landing_Outcome	COUNT(*)
Failure (parachute)	101

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Payload_Mass__KG_" = (SELECT MAX("Payload_Mass__KG_") FROM SPACEXT

```
In [27]: %sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Payload_Mass__KG_" = (SELECT MAX("Payload_Mass__KG_") FRO
 * sqlite:///my_data1.db
Done.
```

Out[27]:

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

- SELECT substr("Date", 6, 2) AS month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE WHERE "Landing_Outcome" LIKE 'Failure (drone ship)' AND substr("Date", 0, 5) = '2015';

```
In [29]: %sql SELECT substr("Date", 6, 2) AS month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE W
 * sqlite:///my_data1.db
Done.
```

Out[29]:

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

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
- SELECT "Landing_Outcome", COUNT(*) as count FROM SPACEXTABLE WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY "Landing_Outcome" ORDER BY count DESC;

```
In [30]: %sql SELECT "Landing_Outcome", COUNT(*) as count FROM SPACEXTABLE WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-
 * sqlite:///my_data1.db
Done.
```

Out[30]:

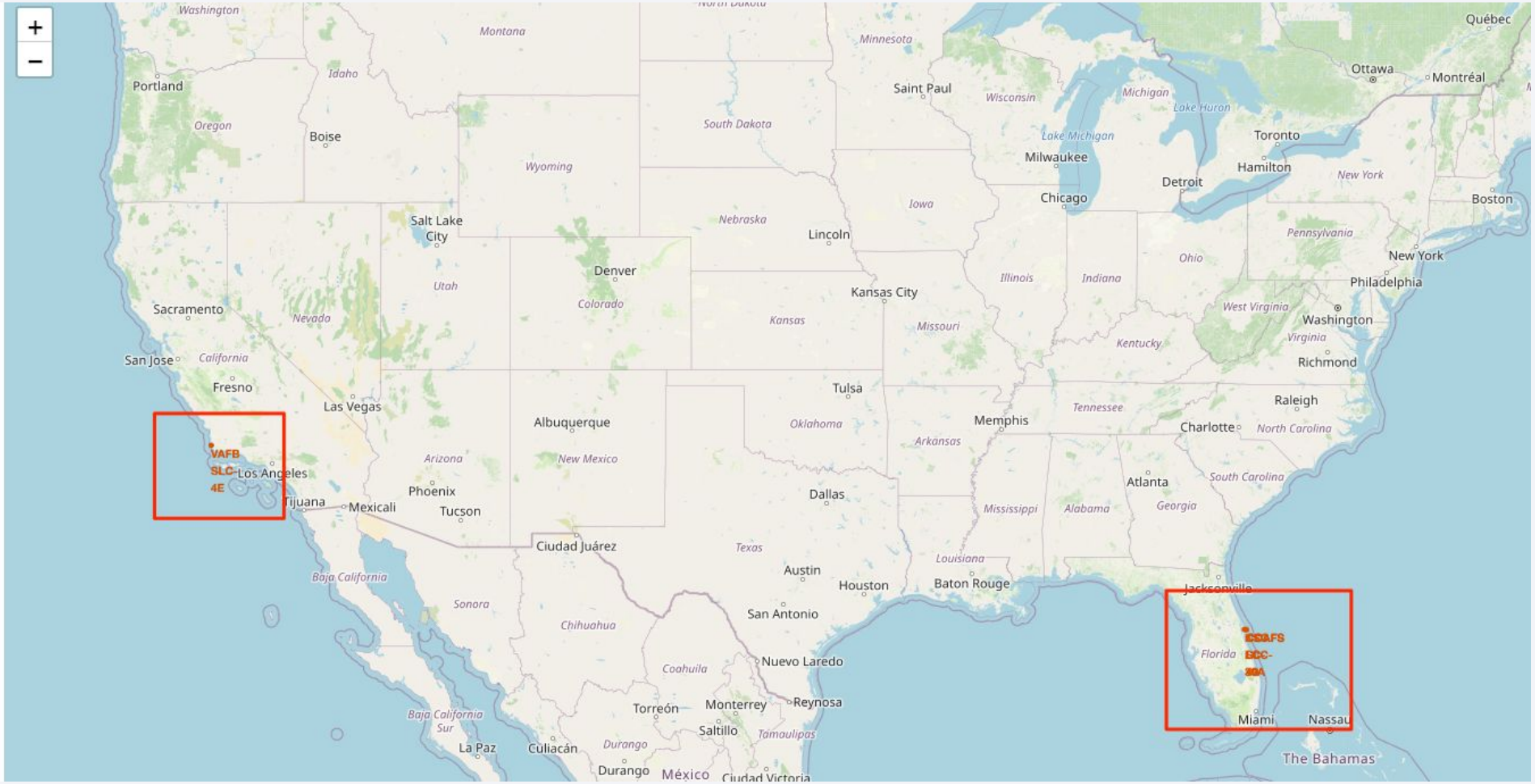
Landing_Outcome	count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1


Section 3
Launch Sites
Proximities Analysis

The generated map with marked launch sites

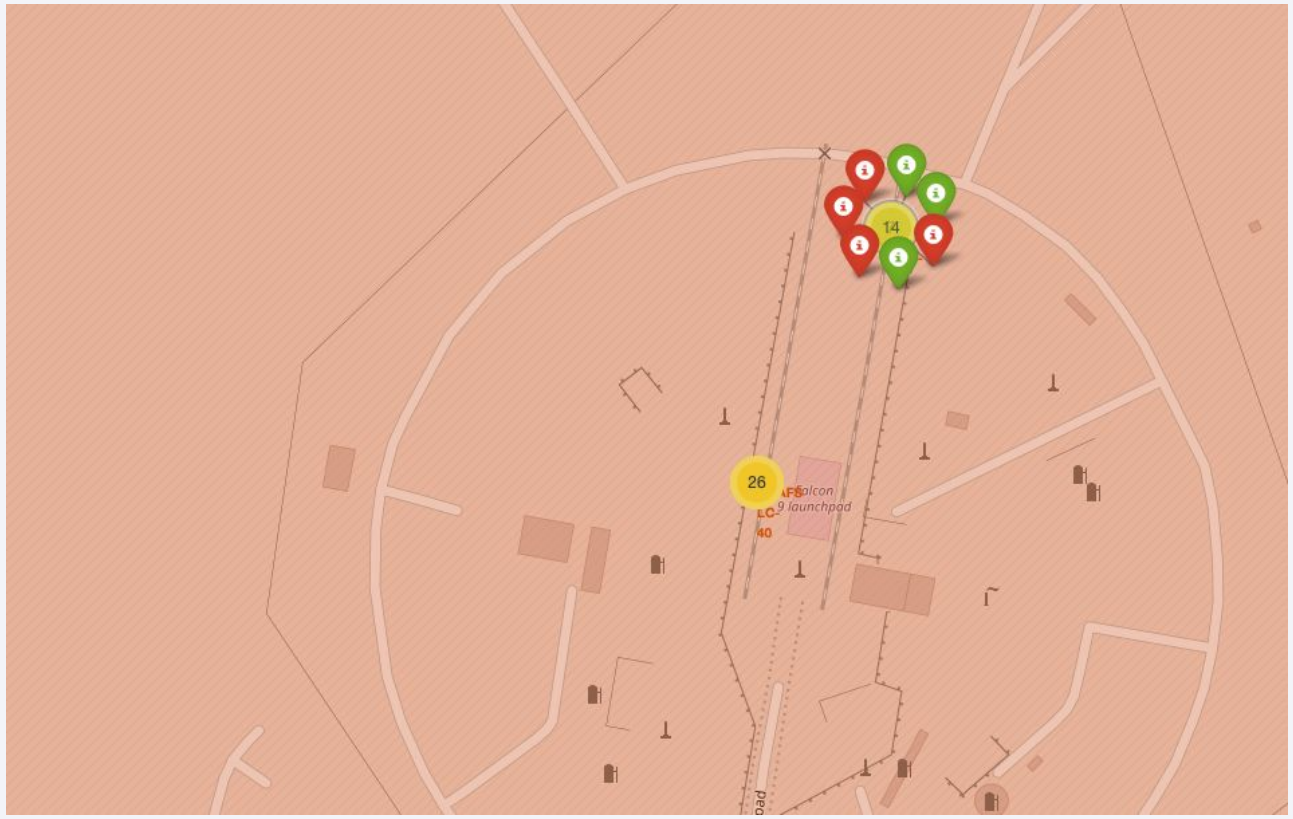
- Explain the important elements and findings on the screenshot

[9]:



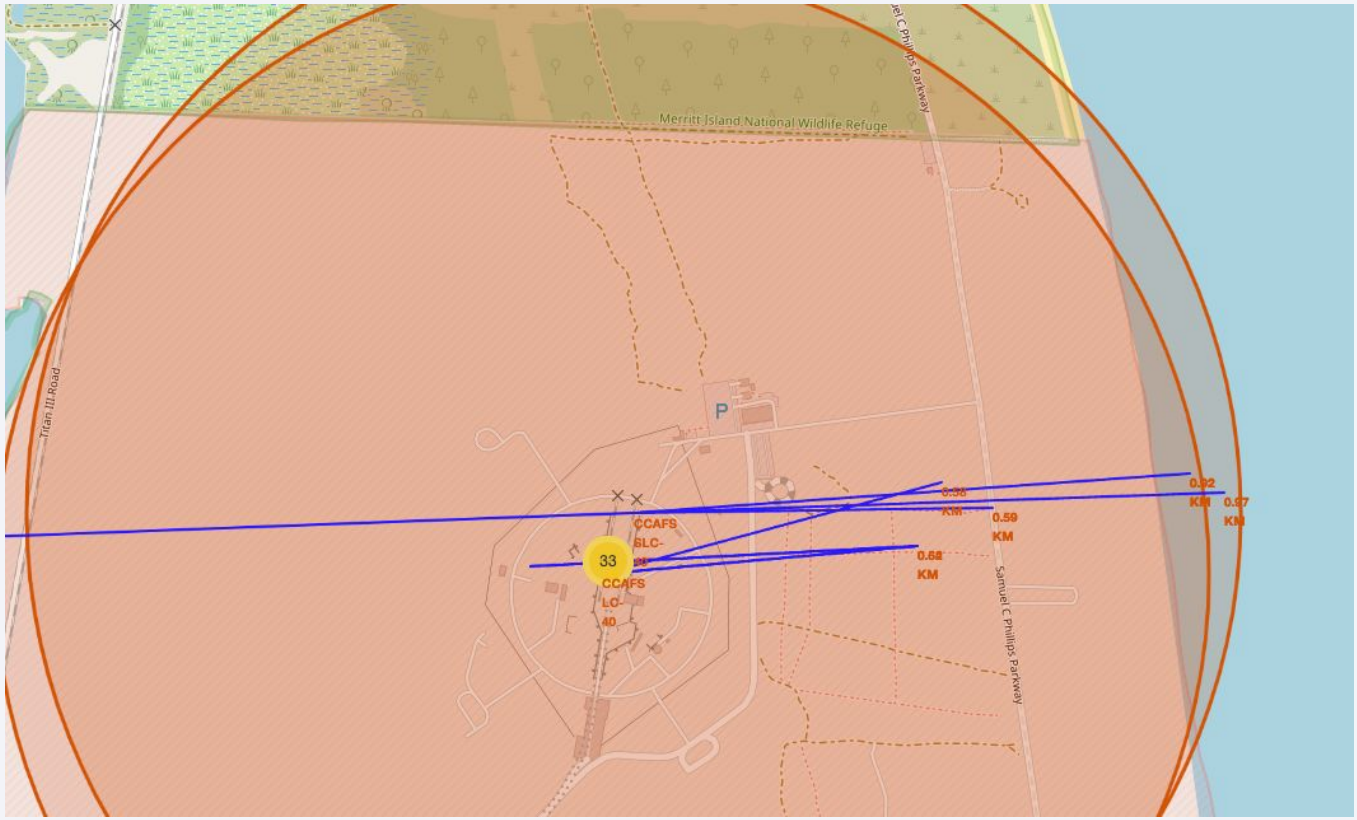
Success rate per launch site

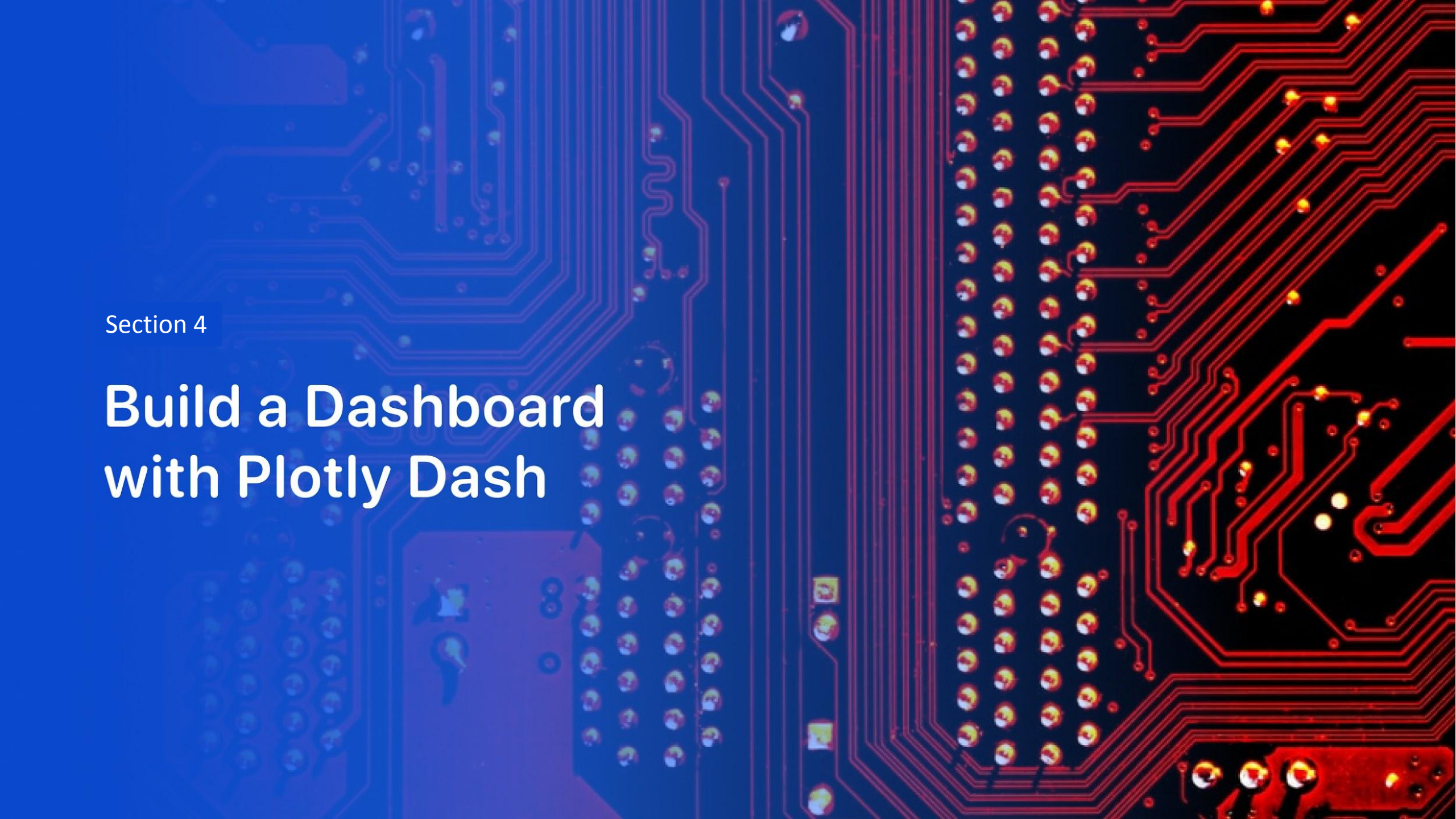
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map



Launch site distances

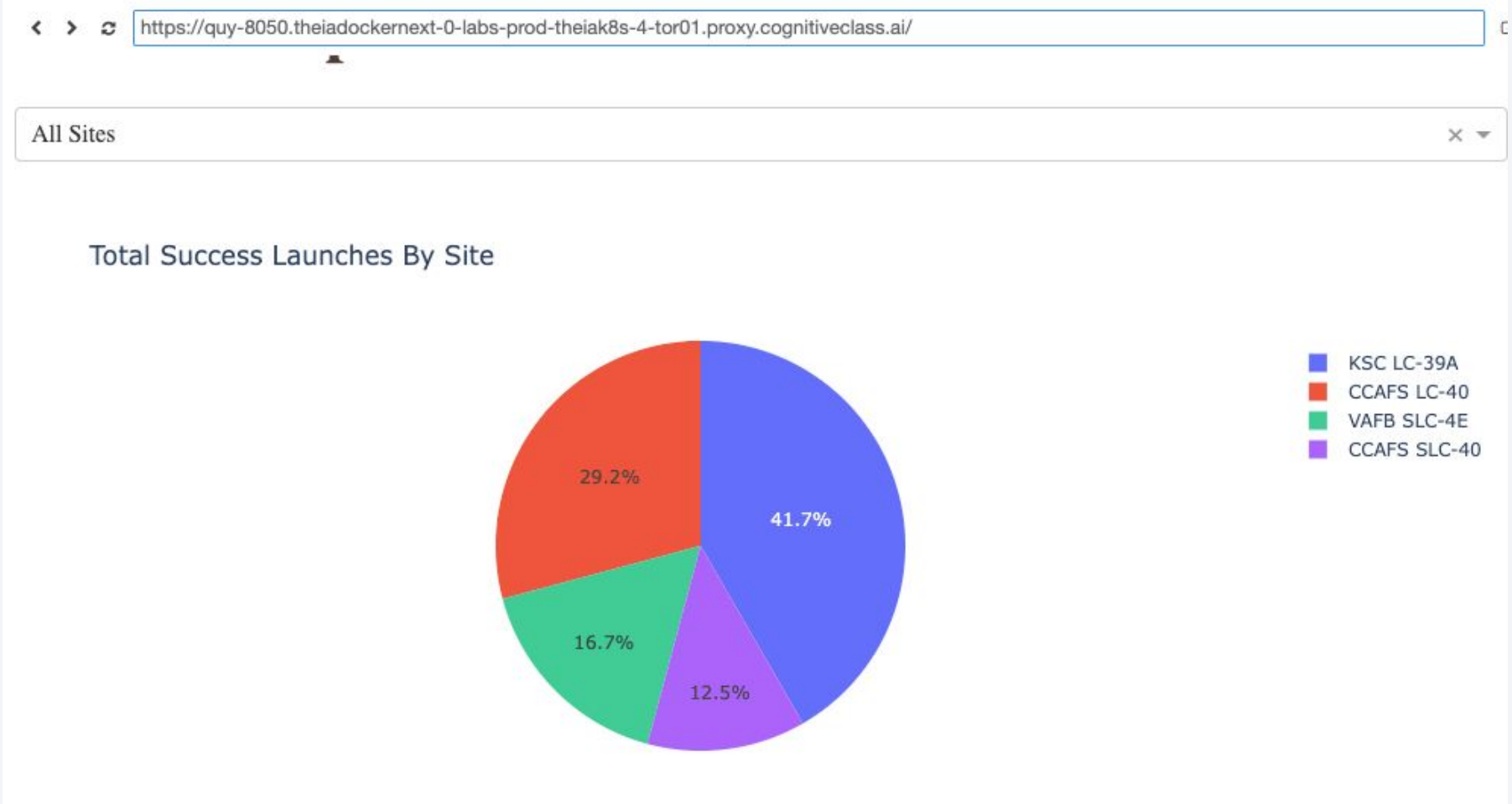
- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed




Section 4
Build a Dashboard with Plotly Dash

Launch success for all sites

- Show the screenshot of launch success count for all sites, in a piechart

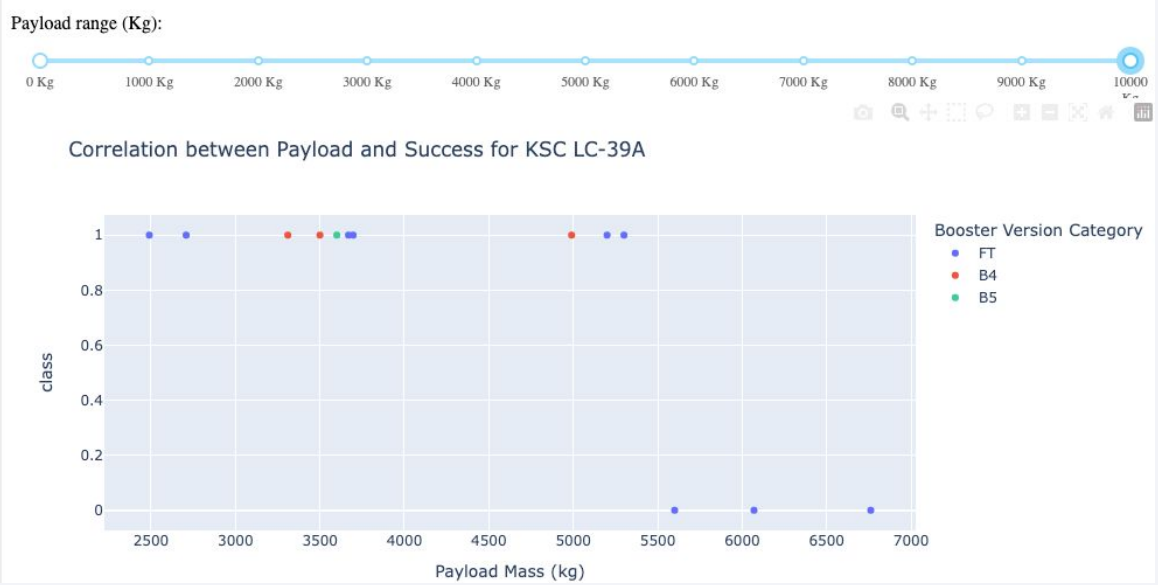


Highest launch success ratio

- Show the screenshot of the piechart for the launch site with highest launch success ratio

Payload vs Launch Outcome

- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider



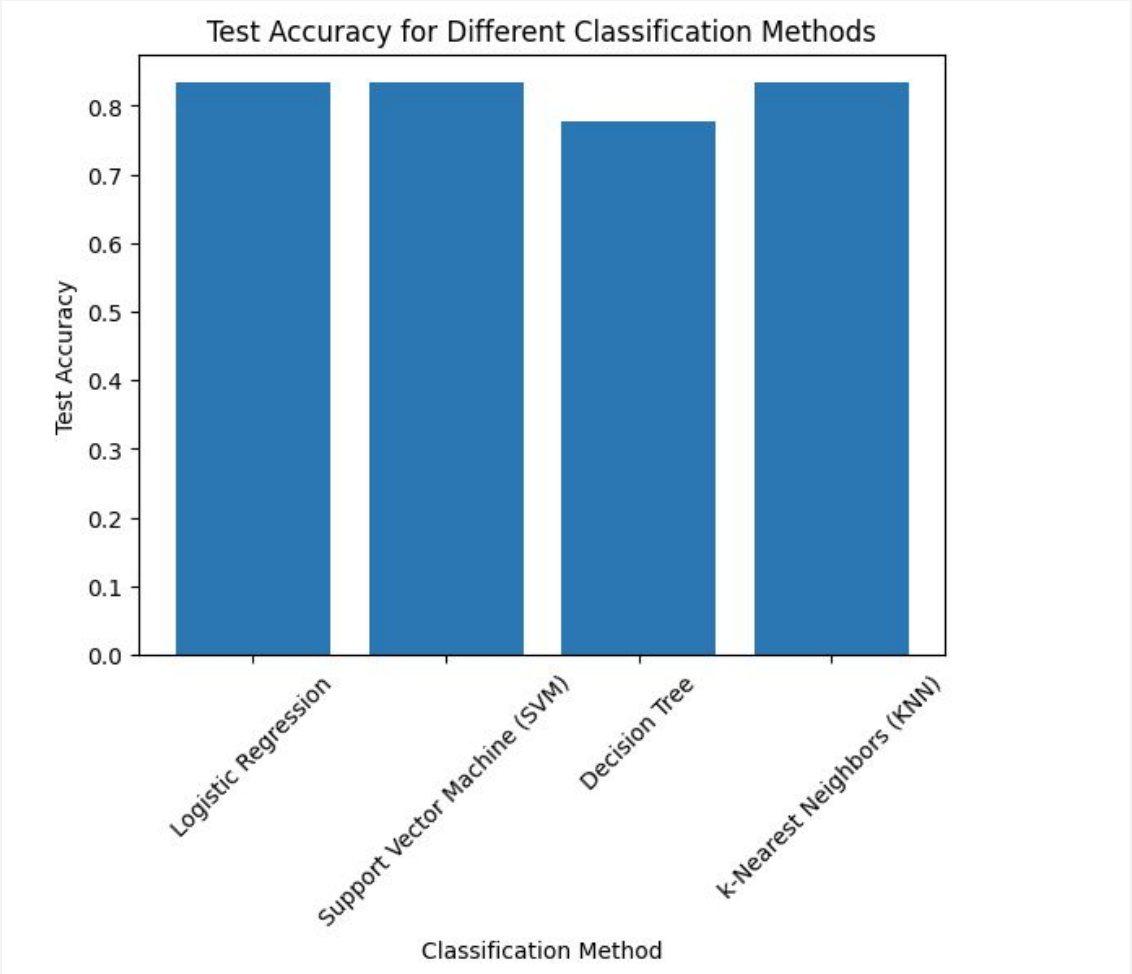
Section 5

Predictive Analysis (Classification)

Classification Accuracy

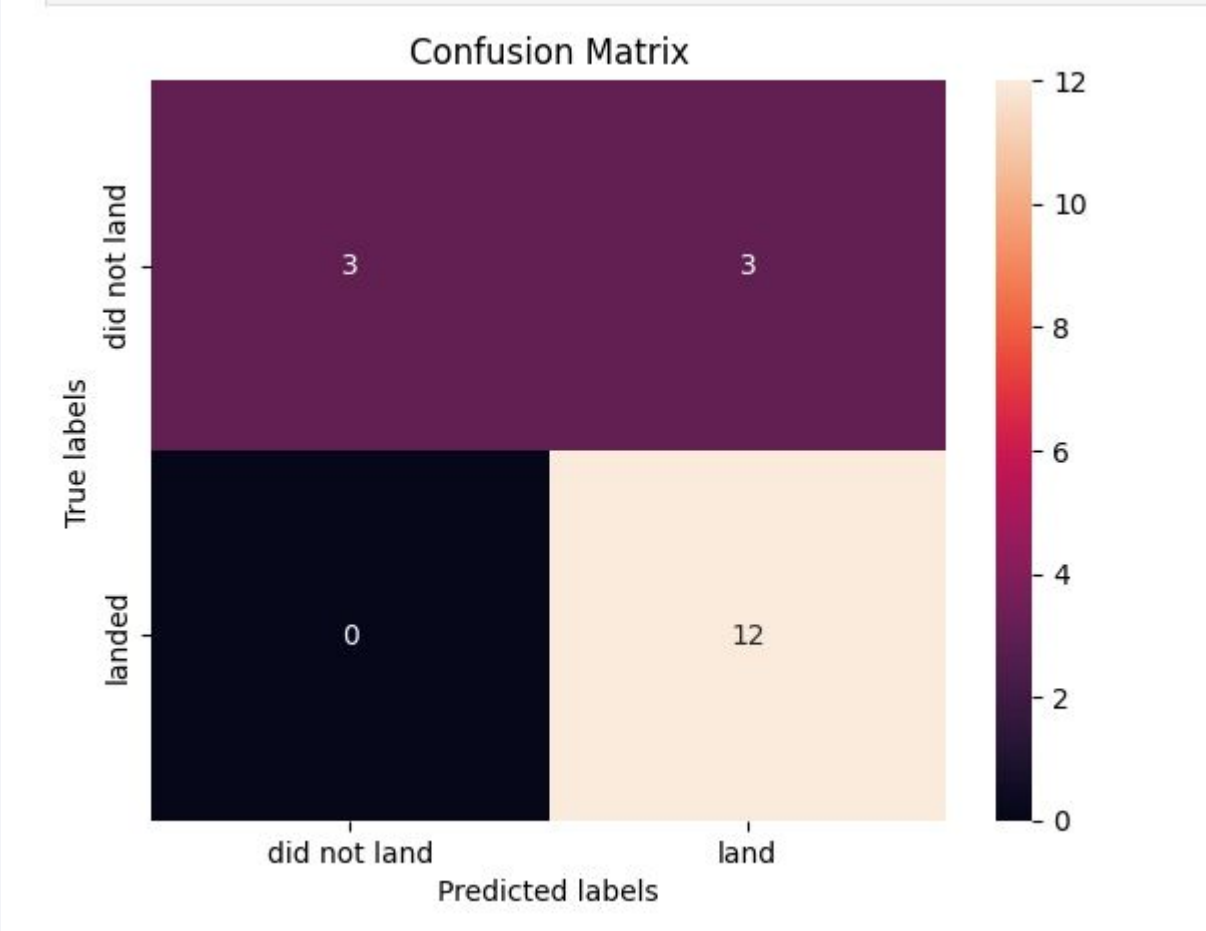
- Visualize the built model accuracy for all built classification models, in a bar chart

- Find which model has the highest classification accuracy



Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



Conclusions

- The best performance is KNN but when we tuned hpyer parameters then the accuracy of decision tree classifier is better
- Best accuracy is 83%

Appendix

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

Thank you!