IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Samet Akdag
06.10.2024

Outline

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

Executive Summary

- The project focused on predicting whether the first stage of SpaceX's Falcon 9 rockets would successfully land. Data collection was conducted using the SpaceX API and web scraping techniques, followed by data analysis using various machine learning algorithms. The prediction models achieved an accuracy rate of 83.3%.
- The models predicted the success rate of first-stage rocket landings with 83.3% accuracy. These predictions enable SpaceY to present more strategic offers against SpaceX, providing a competitive advantage in the commercial space industry.

Introduction

- SpaceY is a rapidly growing startup in the commercial space travel industry. It aims to compete with SpaceX's Falcon 9 rockets. SpaceX provides a cost advantage with its reusable rocket technology. The goal of the project is to develop a prediction model that can help SpaceY make more strategic decisions in this field.
- The main question this project seeks to answer is how to predict whether SpaceX will successfully recover the first stage of a rocket. This prediction is crucial for understanding launch costs and gaining an advantage in competing bids.

Section 1

Methodology

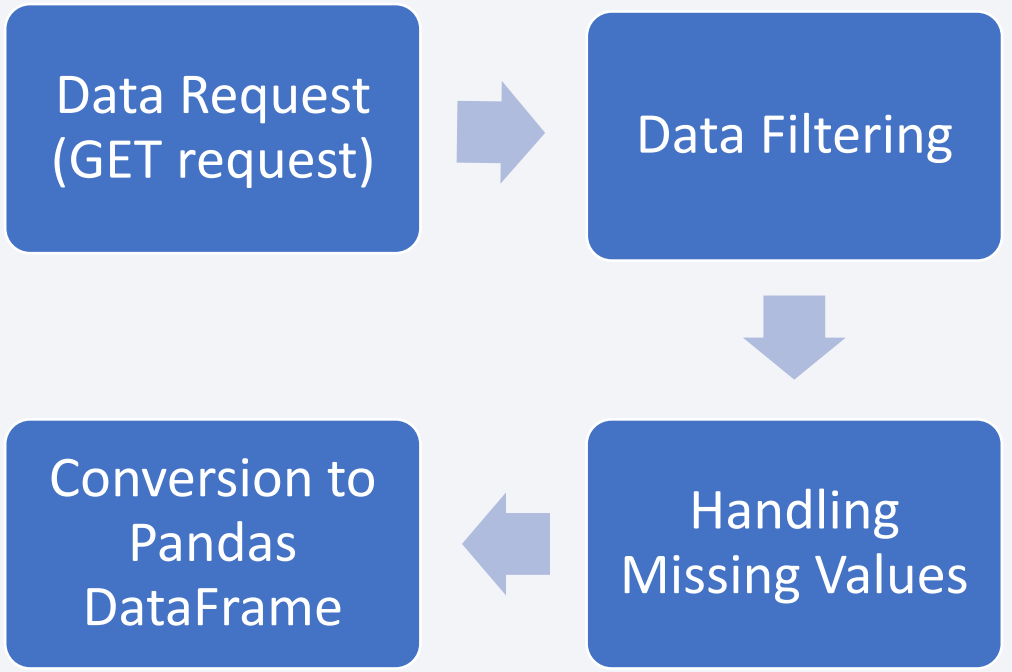
Methodology

Executive Summary

- **Data Collection:** The data collection process was carried out using Python to gather data related to SpaceX's Falcon 9 launches. Data was collected using API and web scraping techniques.
- **Data Wrangling**: The collected data was cleaned, and missing values were handled appropriately.
- **Data Processing:** The data was processed using Python libraries and made suitable for analysis.
- **EDA Using Visualization and SQL**: Exploratory data analysis was performed on the data, and the data was visualized using Matplotlib, Seaborn, and SQL.
- **Interactive Visual Analytics:** Folium was used for maps, and Plotly Dash was used for interactive charts. The map analysis shows launch sites and successful landing rates.
- **Predictive Analysis:** Classification models such as Logistic Regression, SVM, Decision Tree, and KNN were trained, and hyperparameter tuning was performed. The models were tested with an accuracy of 83.3%.

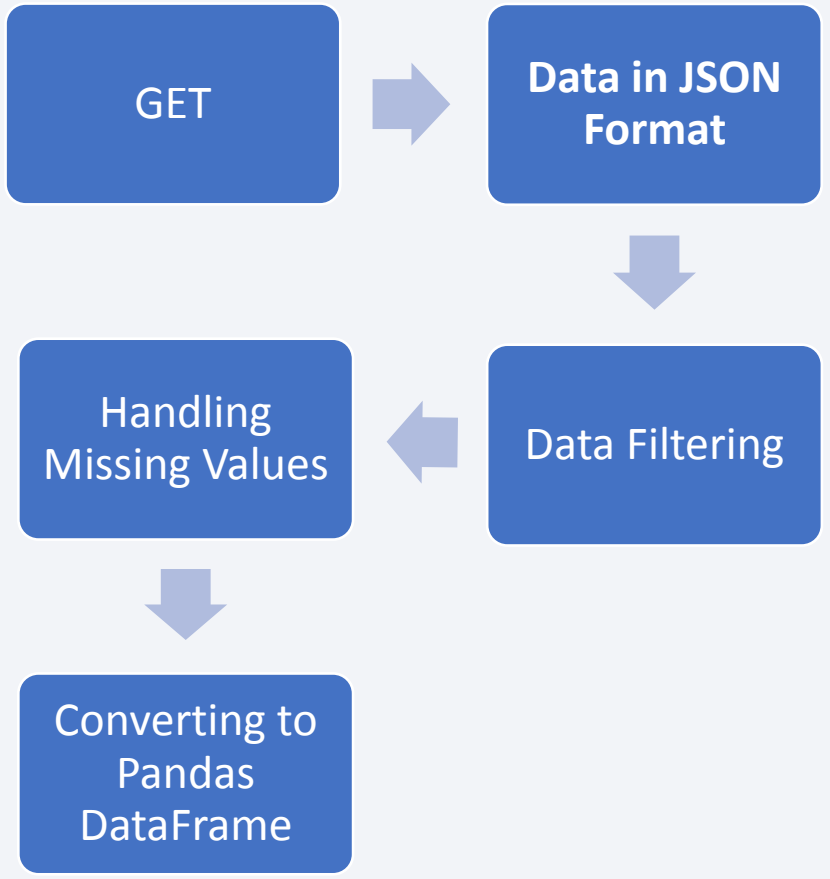
Data Collection

- **Data Sources:** Data was collected using web scraping techniques from the SpaceX API and Wikipedia. The data from these two sources contains historical information about Falcon 9 rocket launches.
- **Using SpaceX API:** Data on Falcon 9 rockets was retrieved from the SpaceX API using a GET request. The data obtained from the API was filtered specifically to analyze landing successes.
- **Web Scraping:** Additional information regarding Falcon 9 and Falcon Heavy launches was gathered from the Wikipedia page. The HTML table structure was parsed, and the data was converted into a pandas dataframe.
- **Missing Values:** Missing data was filled with the average value of the respective column to ensure data integrity.



Data Collection – SpaceX API

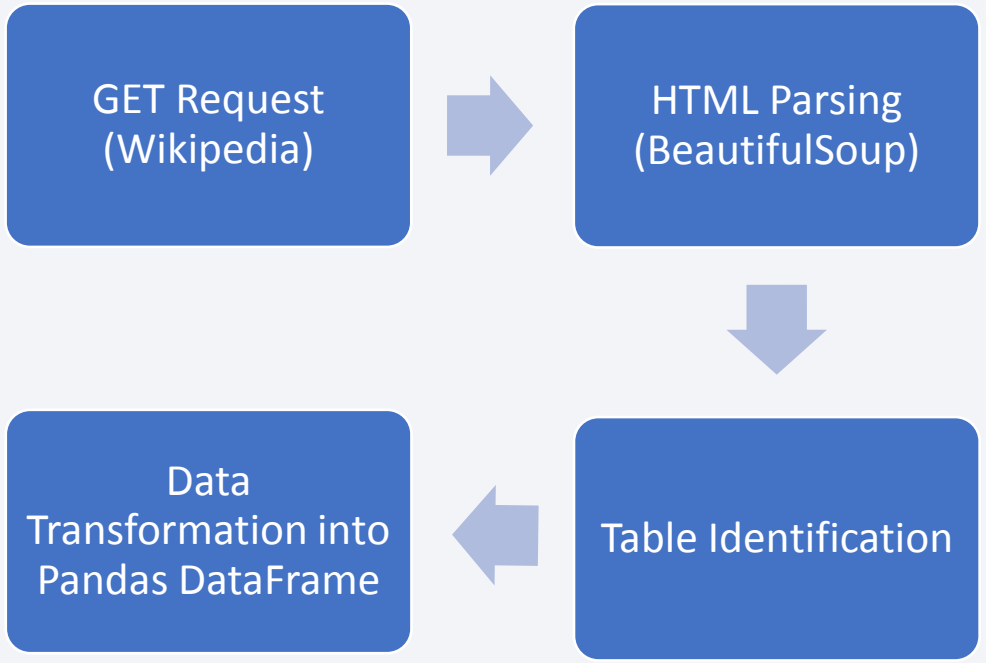
- **Using REST API:** Raw data on rocket launches was obtained using SpaceX's publicly available REST API. These API calls returned data in JSON format.
- **GET Request:** Information about Falcon 9 launches was collected through a GET request to the SpaceX API.
- **Filtering:** Only Falcon 9 launches were selected to create the relevant dataset.
- **Missing Values:** Missing values in the launch data were filled with the mean, and the data was cleaned prior to analysis.
- **GitHub URL:** https://github.com/AKDG-SMT/Data-Collection-API-Lab.git



Data Collection - Scraping

Web Scraping Process: Data related to the Falcon 9 and Falcon Heavy launch list on Wikipedia was scraped using Python's *Requests* and *BeautifulSoup* libraries.

- **Request to Wikipedia**: A request was sent to the relevant Wikipedia page.
- **HTML Parsing**: The Wikipedia page was retrieved in HTML format and parsed using BeautifulSoup.
- **Data Extraction**: HTML table structures were identified, and data was extracted along with the table headers.
- **DataFrame Transformation**: The extracted data was transformed into a pandas DataFrame, making it ready for analysis.



Data Wrangling

Data Processing Steps: After data collection, missing or erroneous values were handled, and the data was cleaned and organized for analysis.

- **Missing Value Handling**: Missing data was filled with the averages of the respective columns.
- **Outlier Detection**: Outliers were identified using outlier analysis and handled appropriately.
- **Data Standardization**: The data was normalized for model training.
- **Feature Engineering**: Necessary features were created, and the training set was prepared for the classification model.

EDA with Data Visualization

- **FlightNumber vs PayloadMass**: A scatter plot was used to examine the relationship between flight numbers and payload mass. This helps to identify any trends in payload mass over the different launches.
- **FlightNumber vs LaunchSite**: A bar chart was used to show the frequency of launches from different sites. This visualized launch site performance across multiple flights.
- **PayloadMass vs LaunchSite**: A scatter plot was used to analyze how payload mass varies by launch site, giving insights into how different sites handle different payload sizes.
- **Orbit Type vs Success Rate**: A bar chart was plotted to display the success rates of launches across different orbit types. This helped identify which orbits are more likely to have successful launches.
- **Yearly Success Trend**: A line plot was used to show the trend of launch success rates over the years, revealing any improvements or regressions over time.

GitHub: https://github.com/AKDG-SMT/Data-Collection-API-Lab/blob/main/jupyter-labs-eda-dataviz%20(1).ipynb

EDA with SQL

- **Launch Sites Query**: Retrieved data for all the launch sites used in Falcon 9 launches.
- **Payload Mass Query**: Queried the database to get the payload mass associated with each launch.
- **Booster Version Query**: Retrieved information about the versions of the boosters used in the launches.
- **Mission Outcome Query**: Listed the outcomes (success or failure) of each mission.
- **Booster Landing Query**: Displayed information on whether the booster successfully landed or not after the launch.

GitHub: https://github.com/AKDG-SMT/Data-Collection-API-Lab/blob/main/jupyter-labs-eda-sql-coursera_sqllite%20(1).ipynb

Build an Interactive Map with Folium

Map Objects Added

- **Markers**: Added markers to indicate the location of each launch site. These markers help in identifying the geographical spread of the launch sites.
- **Colored Circles**: Different colored circles were added around the markers to represent the success rate of launches from each site, giving a quick visual indication of the performance.
- **Lines**: Lines were used to display proximity to important infrastructure such as highways, railways, and coastlines. This was done to analyze how these factors may impact the choice of launch sites and their success rates.

Why Added

- **Markers**: To clearly identify the exact geographic locations of all launch sites.
- **Colored Circles**: To visually distinguish between successful and failed launches at a glance.
- **Lines**: To assess the strategic advantages of each launch site based on proximity to infrastructure, which might influence operational efficiency.

GitHub: https://github.com/AKDG-SMT/Data-Collection-API-Lab/blob/main/lab-jupyter-launch-site-location-v2%20(1).ipynb

Build a Dashboard with Plotly Dash

Plots/Graphs Added

- **Pie Chart**: A pie chart was included to display the success rates of launches by launch site, color-coded for clarity.
- **Scatter Plot**: A scatter plot was added to show the relationship between payload mass and landing outcomes, helping to visualize which payloads were more likely to result in a successful landing.
- **Range Slider**: A range slider was used to filter payload mass dynamically. Users could adjust the range to view data specific to payload masses they are interested in.
- **Dropdown Menu**: A dropdown was added to allow users to select individual launch sites or view data for all sites simultaneously.

Build a Dashboard with Plotly Dash

Why Added:

- **Pie Chart**: To give a clear overview of the success distribution across different launch sites, helping stakeholders quickly assess which sites perform better.
- **Scatter Plot**: To visualize how payload mass affects the likelihood of a successful booster landing, providing insights into optimal payload conditions.
- **Range Slider**: To make the dashboard interactive, allowing users to filter data based on payload mass ranges, making it easier to explore specific subsets of data.
- **Dropdown Menu**: To offer flexibility in analyzing data either globally across all launch sites or at a specific site level, catering to different stakeholder needs.

Predictive Analysis (Classification)

Model Building: Several classification models were built, including:

- **Logistic Regression**
- **Support Vector Machine (SVM)**
- **Decision Tree Classifier**
- **K Nearest Neighbors (KNN)**

Data Preparation:The dataset was split into training and test sets.

- Feature engineering was conducted, including creating a "Class" column representing the first-stage booster landing outcome.
- Data was standardized to ensure all features contributed equally.

Predictive Analysis (Classification)

Model Evaluation:

- Accuracy scores were calculated for each model using the test data. Confusion matrices were generated to identify misclassifications.
- The models were evaluated based on accuracy and the confusion matrix results, highlighting false positives and negatives.

Model Improvement:

- Hyperparameter tuning was performed using **GridSearchCV** to find the best parameter combinations for each model.

Best Model:

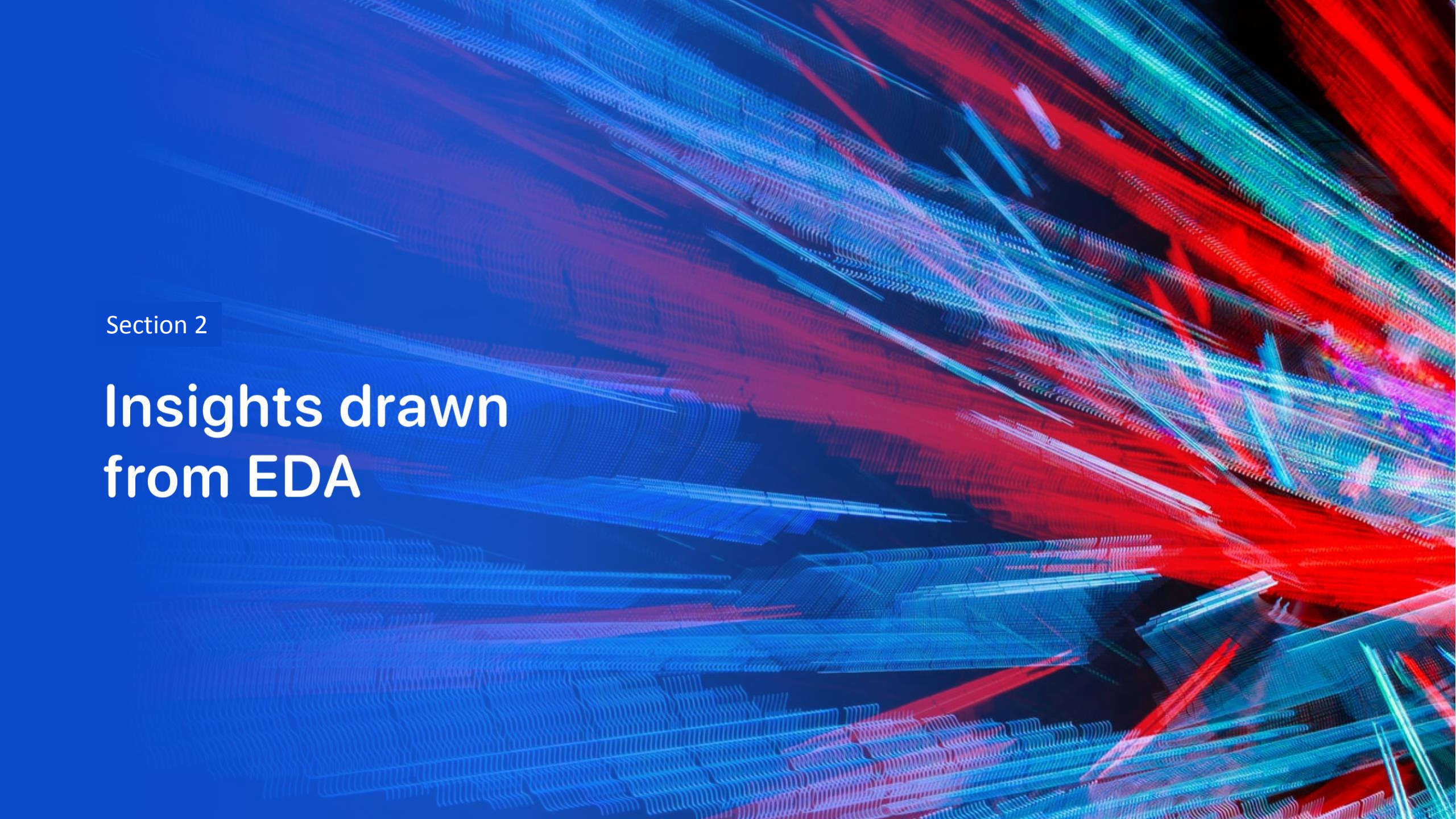
- After tuning, the **Logistic Regression** model achieved the highest accuracy and was selected as the best-performing model, with an accuracy of **83.33%.**

GitHub: https://github.com/AKDG-SMT/Data-Collection-API-Lab/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1%20(1).ipynb

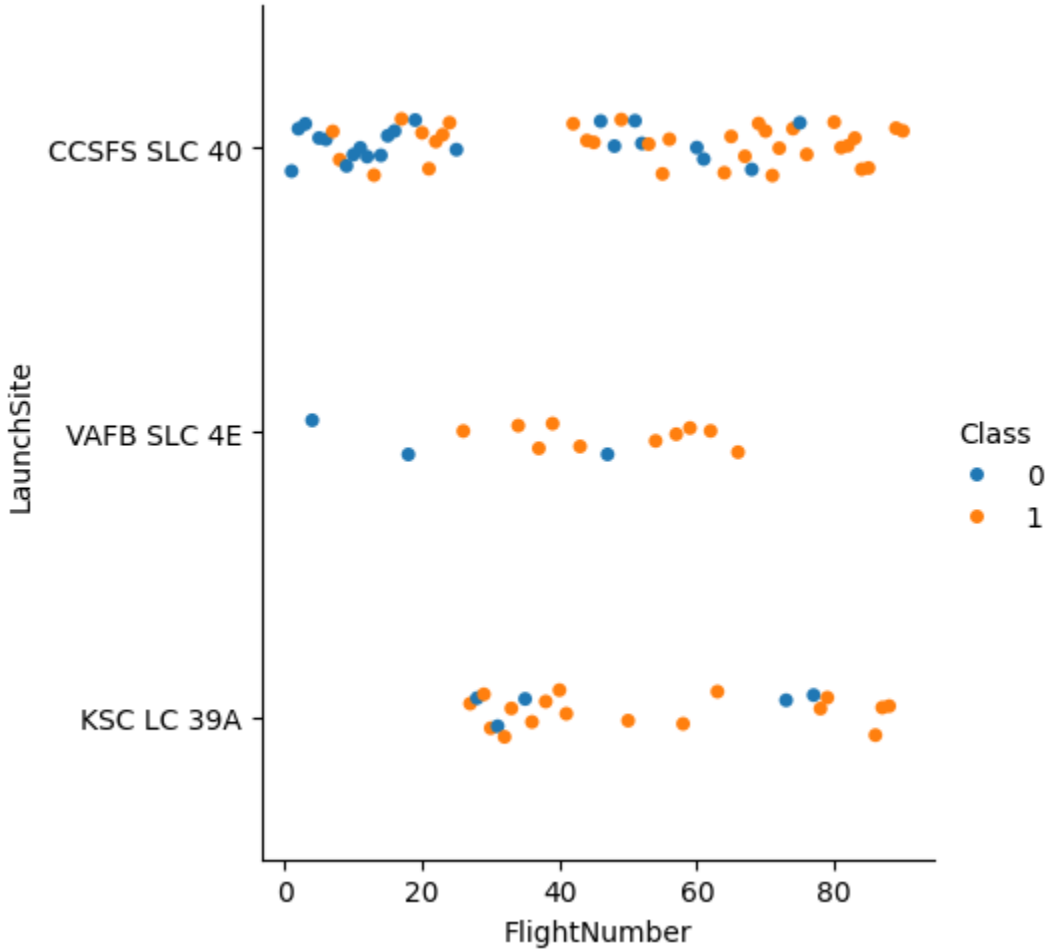
Results

- Visualizations revealed that payload mass and orbit type are strong indicators of launch success. Certain launch sites had significantly higher success rates compared to others, such as KSC LC-39A.

- The final Logistic Regression model achieved an accuracy of 83.33% in predicting the success of a first-stage booster landing. Confusion matrix analysis revealed some false positives, but the model provided valuable predictive insights for SpaceY’s competitive strategy against SpaceX.

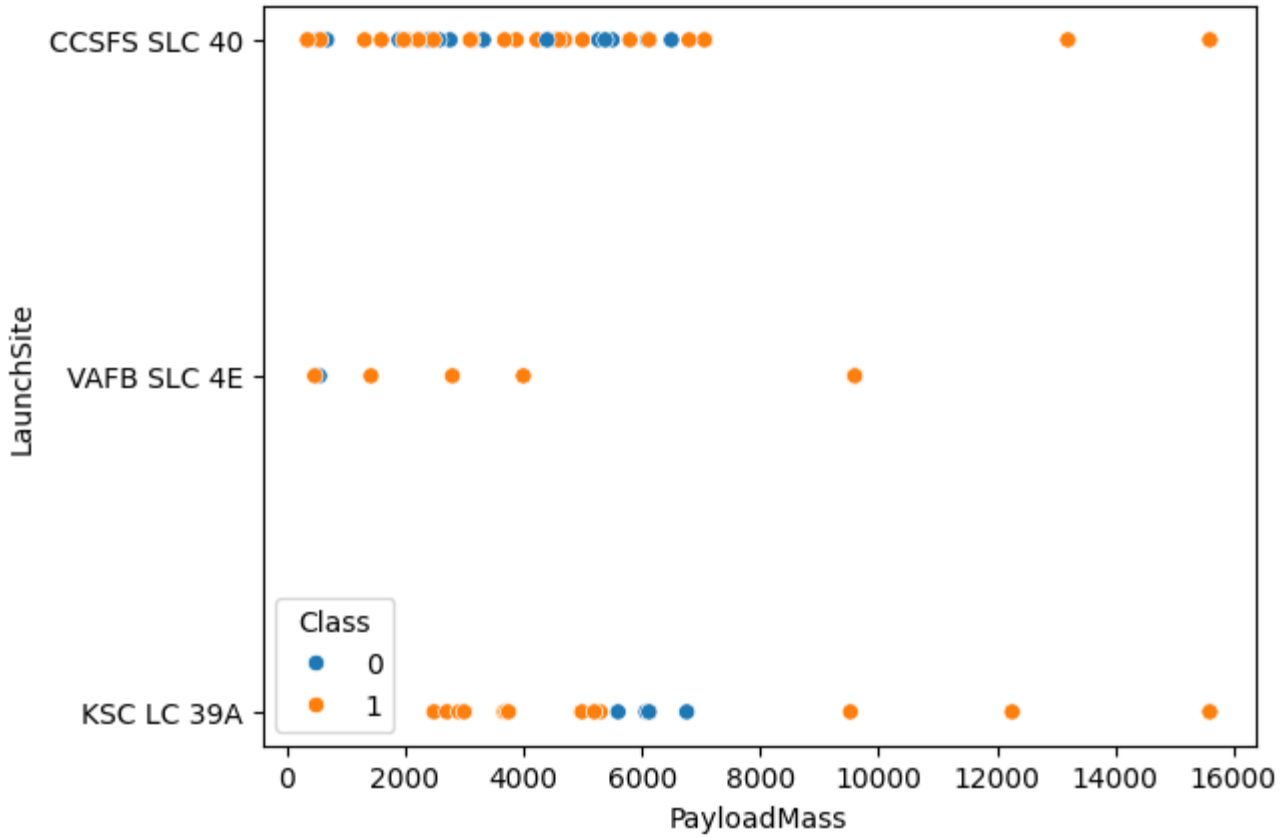

Section 2
Insights drawn from EDA

Results – EDA: Visualize the relationship between Flight Number and Launch Site



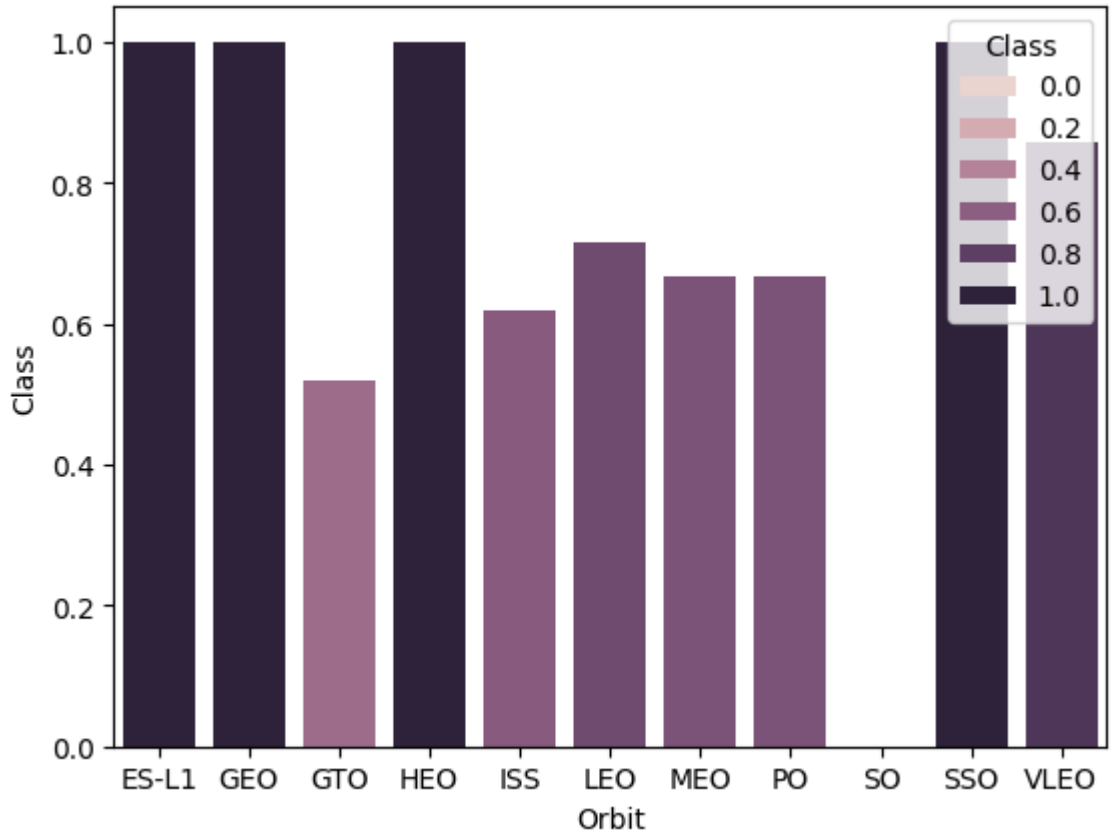
Insight: VAFB SLC 4E and KSC LC 39A Launch Sites have high success rates

Results – EDA:
Visualize the relationship between Payload and Launch Site



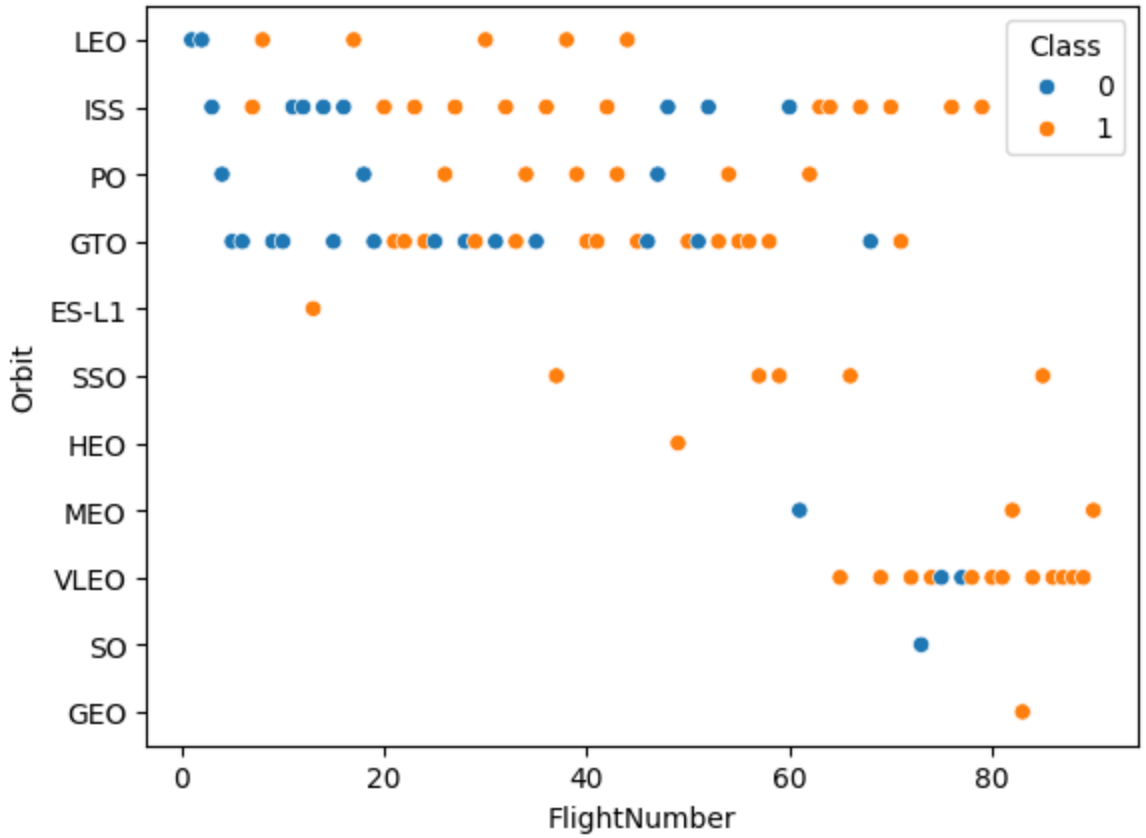
Insight: Very high success rate when Payload Mass is between [2000,5000]. Payload Mass above 8000 also has high success rate, but low statistical significance.

Results – EDA:
Visualize the relationship between success rate of each orbit type



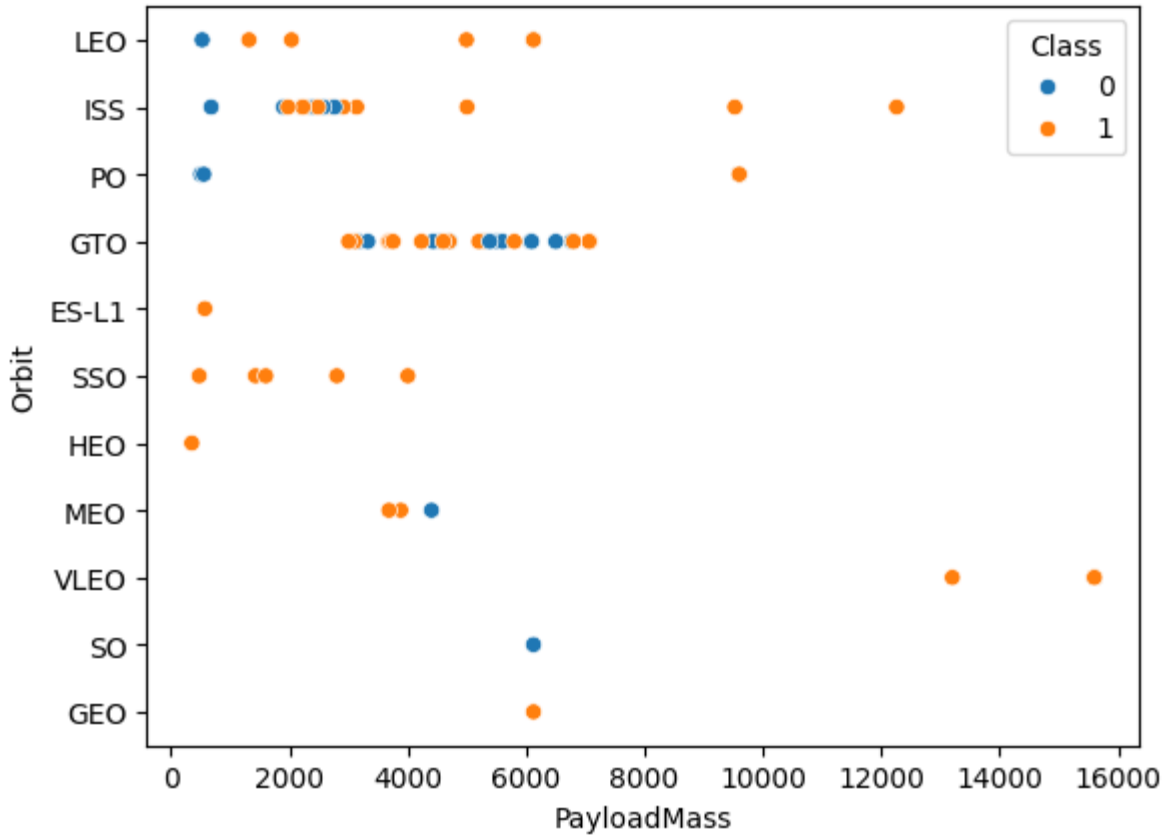
Insight: ES-L1, GEO, HEO, SSO, VLEO Orbits has high success rate. But this has not considered statistical significance.

Results – EDA:
Visualize the relationship between Flight Number and Orbit type



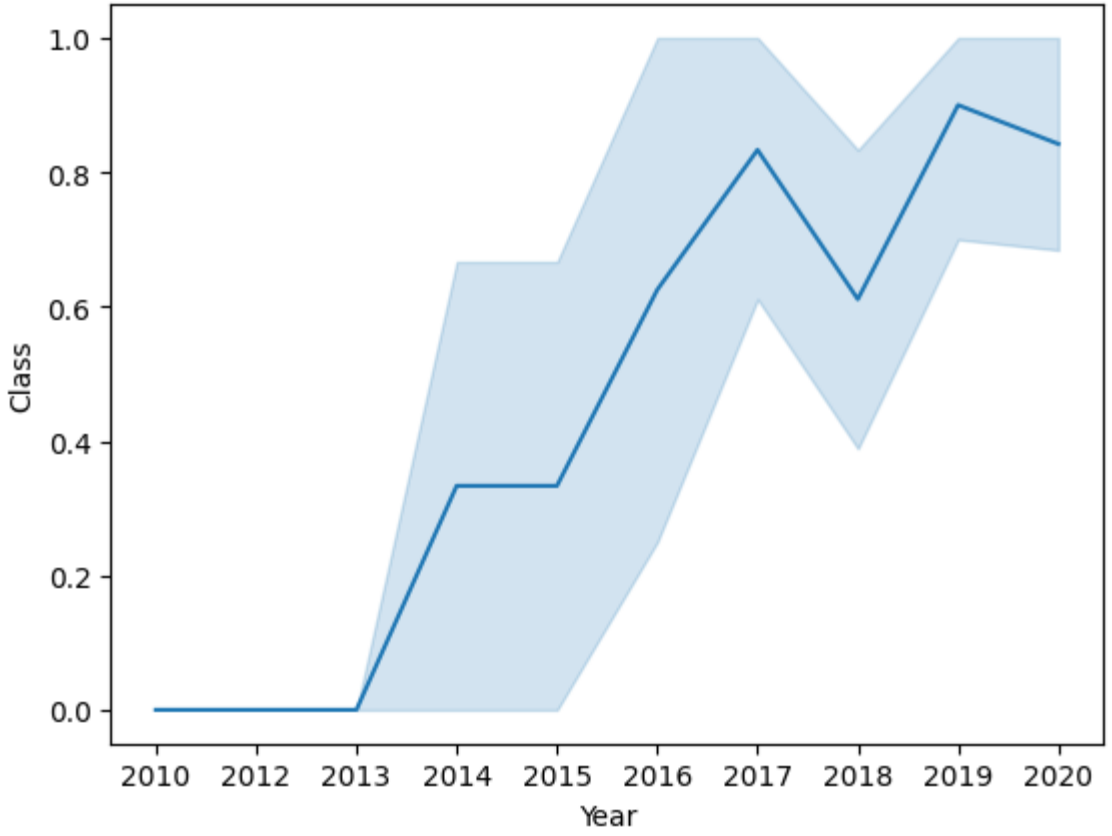
Insight: LEO Orbit option has improved success rate over time.

Results – EDA:
Visualize the relationship between Payload and Orbit type



Insight: In the case of heavy payloads, the successful landing or positive landing rate is more favourable for PO, LEO and ISS. However, for GTO, it is not possible to distinguish between the positive landing rate and the negative landing (unsuccessful mission) with the same degree of clarity.

Results – EDA:
Visualize the launch success yearly trend



Insight: The success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.

All Launch Site Names

Task 1

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

```
%sql Select distinct Launch_Site from SPACEXTBL
```

[12] Python

```
 * 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'

Task 2

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

```
[17]  %sql select * from spacextbl where Launch_Site like "CCA%" limit 5
                                                                                Python
```

```
 * 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

Task 3

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

```
[18]  %sql select sum(PAYLOAD_MASS__KG_) from spacextbl where Customer = "NASA (CRS)"
                                                                                Python
```

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

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

Task 4

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

```
[26]  %sql select avg(PAYLOAD_MASS__KG_) from spacextbl where Booster_Version = "F9 v1.1"
                                                                                    Python
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

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

```
[41]  %sql select min(Date) from spacextbl where Landing_Outcome = "Success (ground pad)"
                                                                                    Python
```

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

min(Date)
2015-12-22

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

```
[46] %sql select Payload from spacextbl where Landing_Outcome = "Success (drone ship)" and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

Python

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

Payload
JCSAT-14
JCSAT-16
SES-10
SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
[47]  %sql select count(*) from spacextbl where Landing_Outcome like "Succ%" or Landing_Outcome like "Fail%"
                                                                                                          Python
```

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

count(*)
71

Boosters Carried Maximum Payload

Task 8

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

```
%sql select Booster_Version from spacextbl where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacextbl)
```

[53] Python

```
 * 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

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.

```
[69]
%sql select substr(Date,6,2) as Month,Booster_Version,Launch_Site, Landing_Outcome from spacextbl where substr(Date,0,5)='2015' and Landing_Outcome = "Failure (drone ship)"
```

Python

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

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

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.

[81]

```
%sql select Landing_Outcome, count(*) as Count from spacextbl where Date between "2010-06-04" and "2017-03-20" group by Landing_Outcome order by Date desc
```

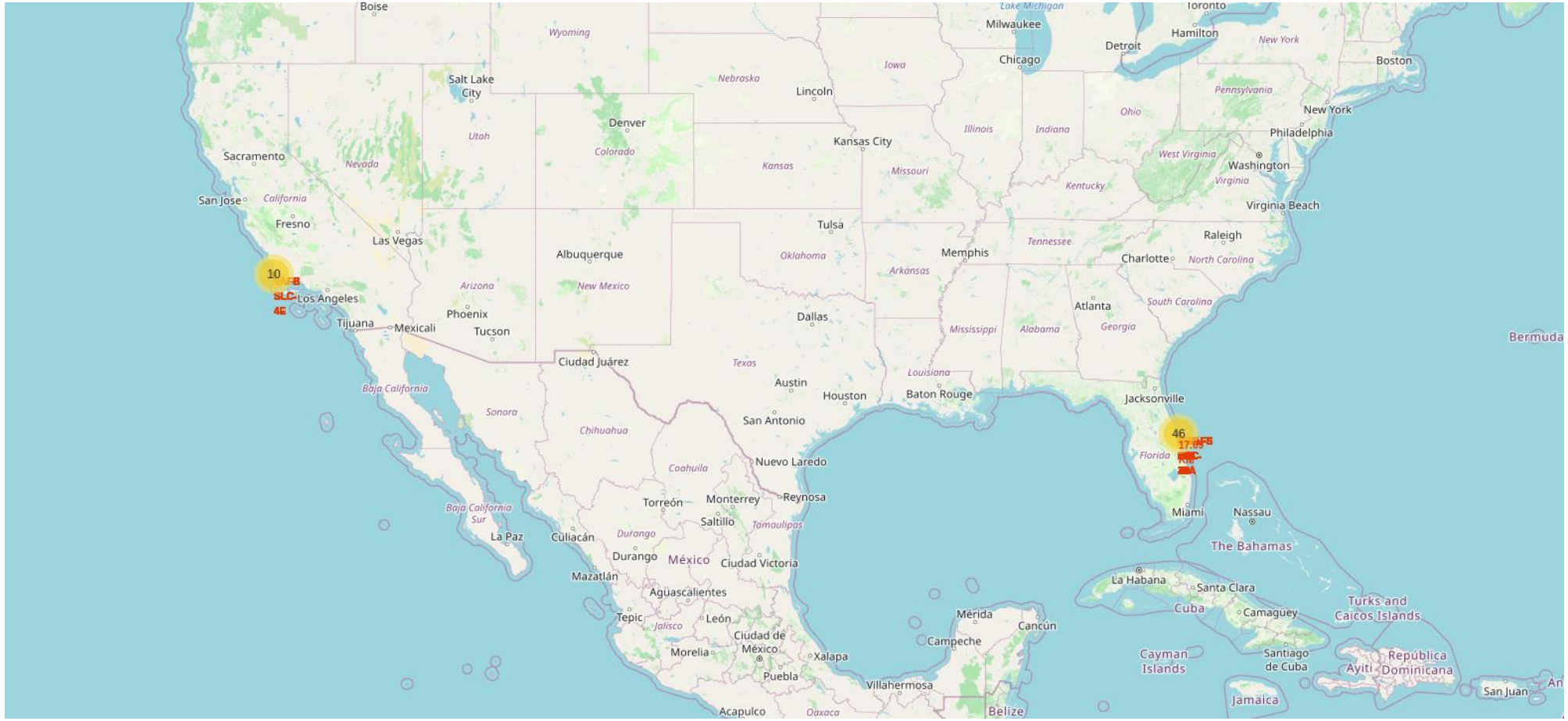
Python

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

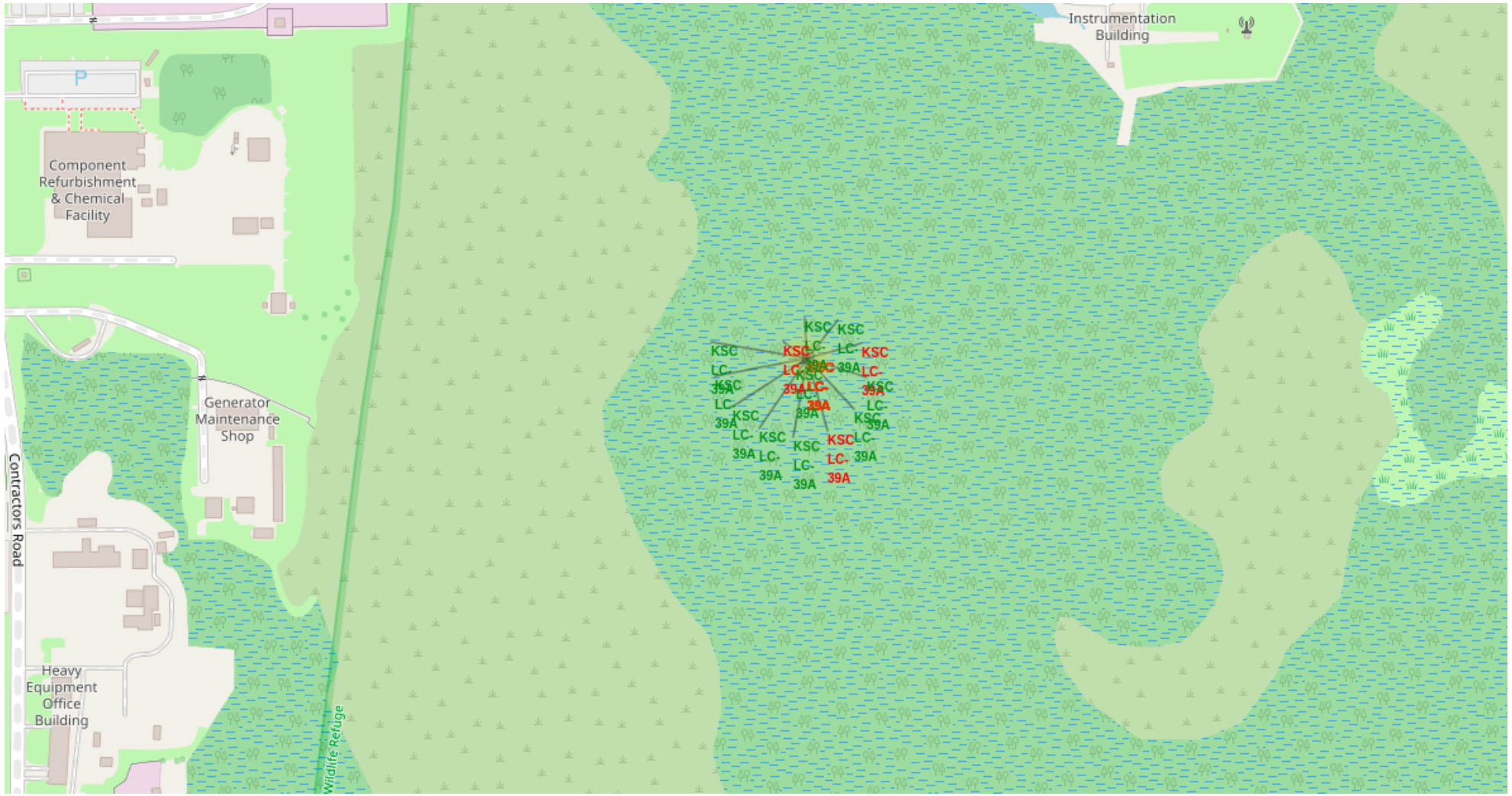
Landing_Outcome	Count
Success (drone ship)	5
Success (ground pad)	3
Precluded (drone ship)	1
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
No attempt	10
Failure (parachute)	2


Section 3
Launch Sites
Proximities Analysis

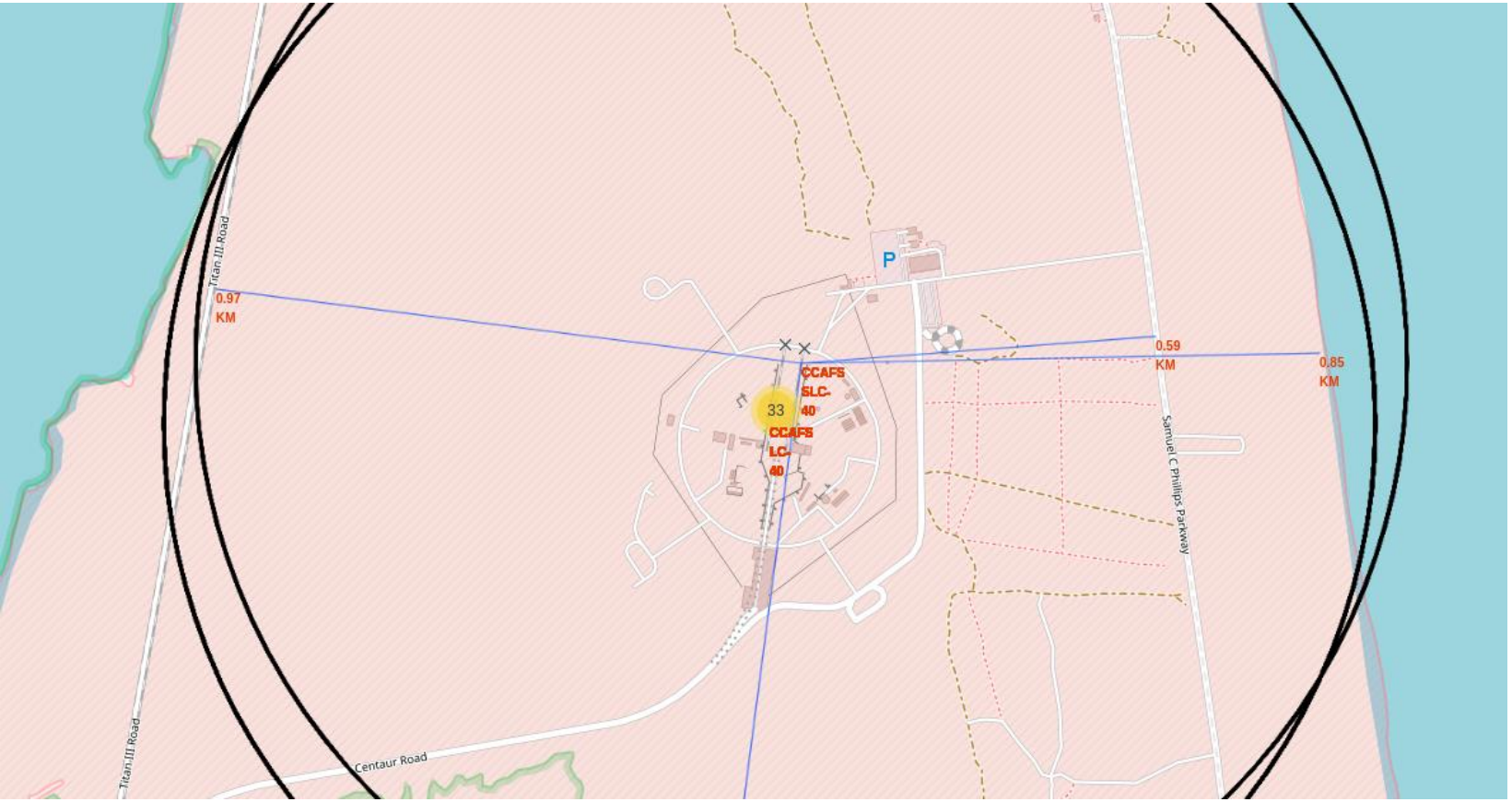
Result – Folium Map: All launch sites



Result – Folium Map: Launch Outcomes

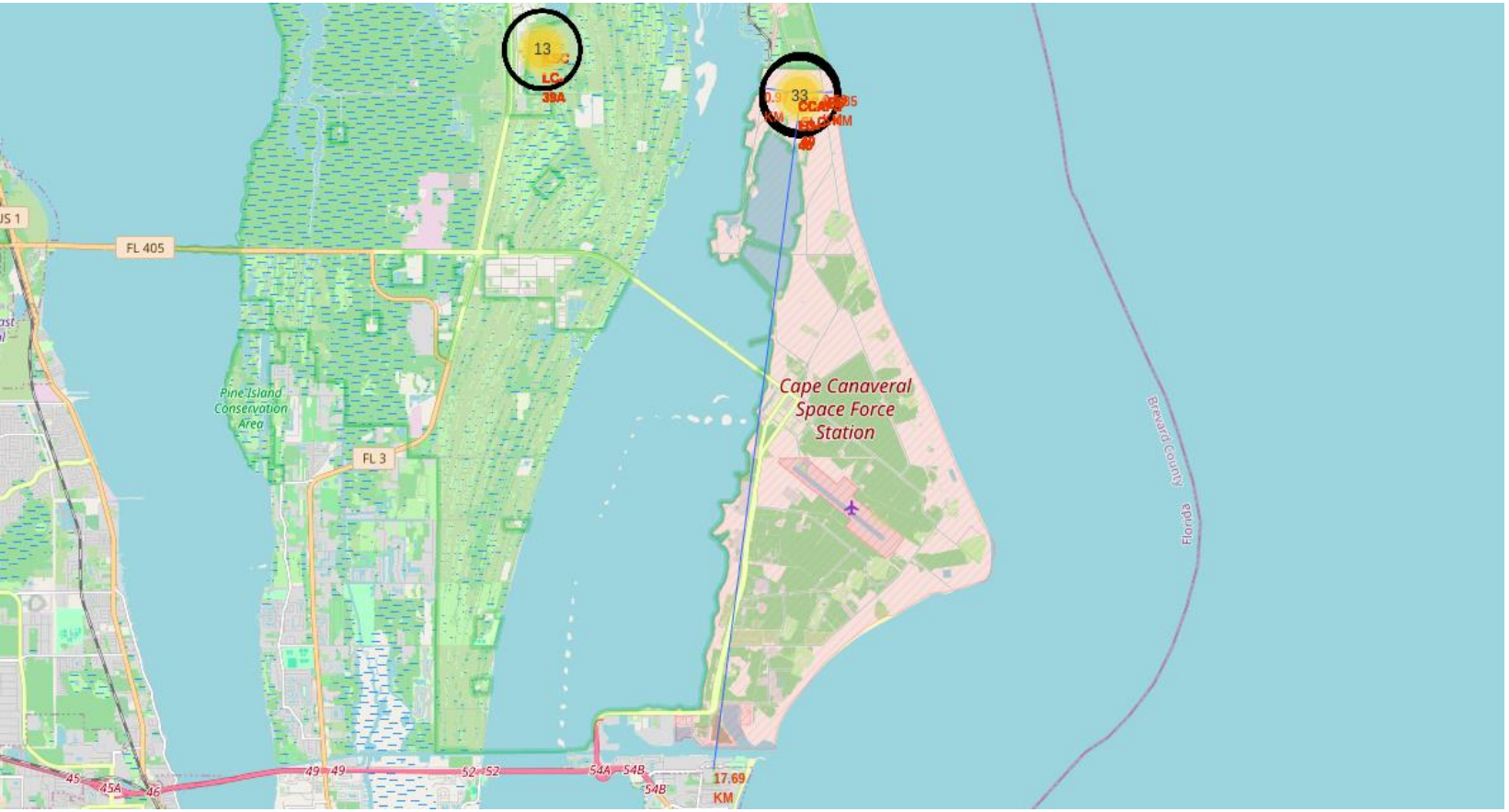


Result – Folium Map: Distance from proximities

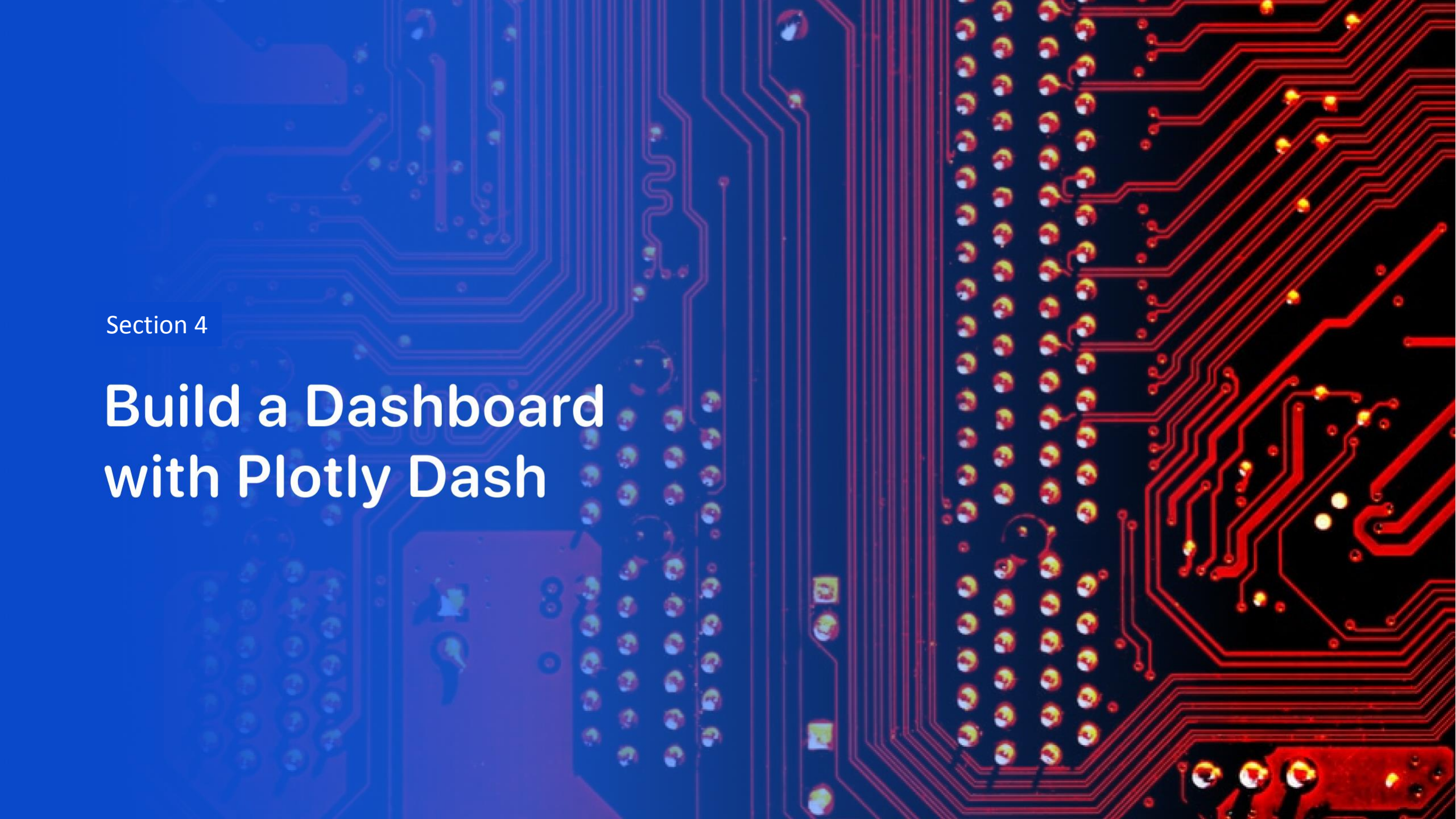


- Distance from nearest railway: 0.97KM.
- Distance from nearest highway: 0.59KM.
- Distance from nearest coastline: 0.85 KM.

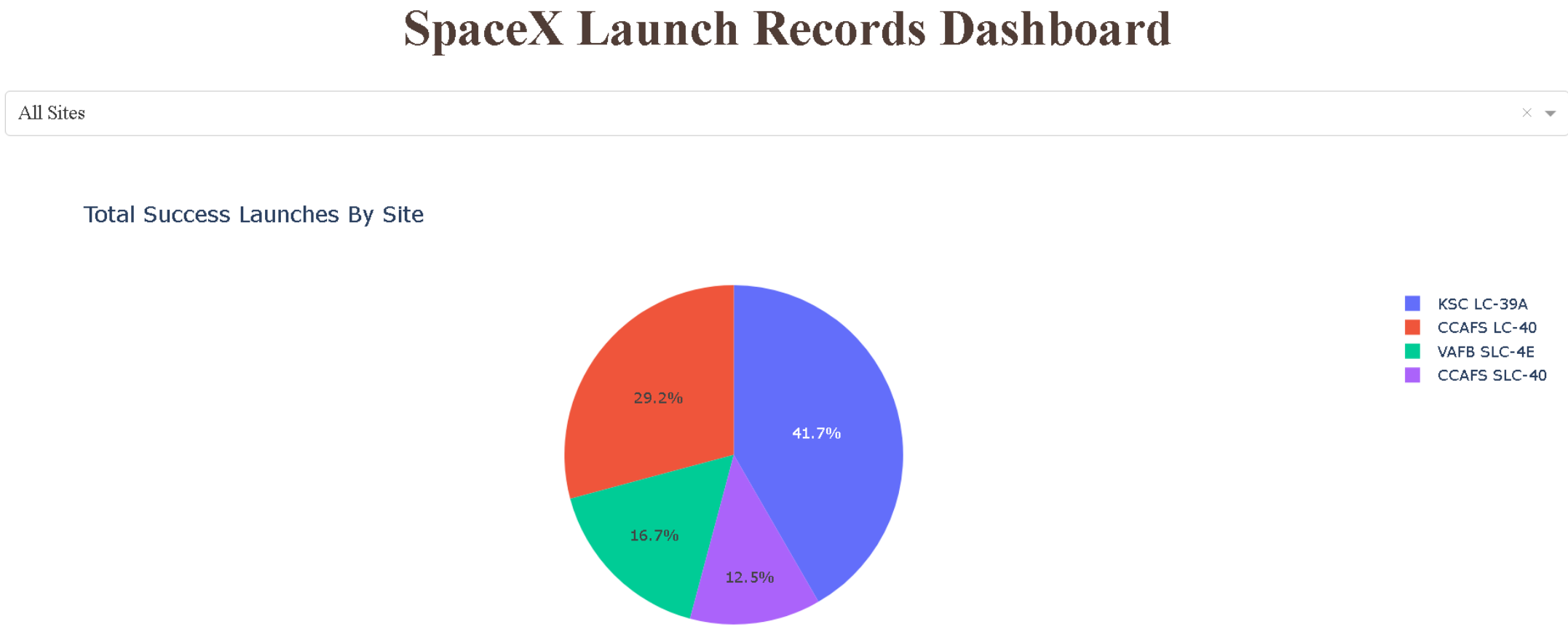
Result – Folium Map: Distance from proximities



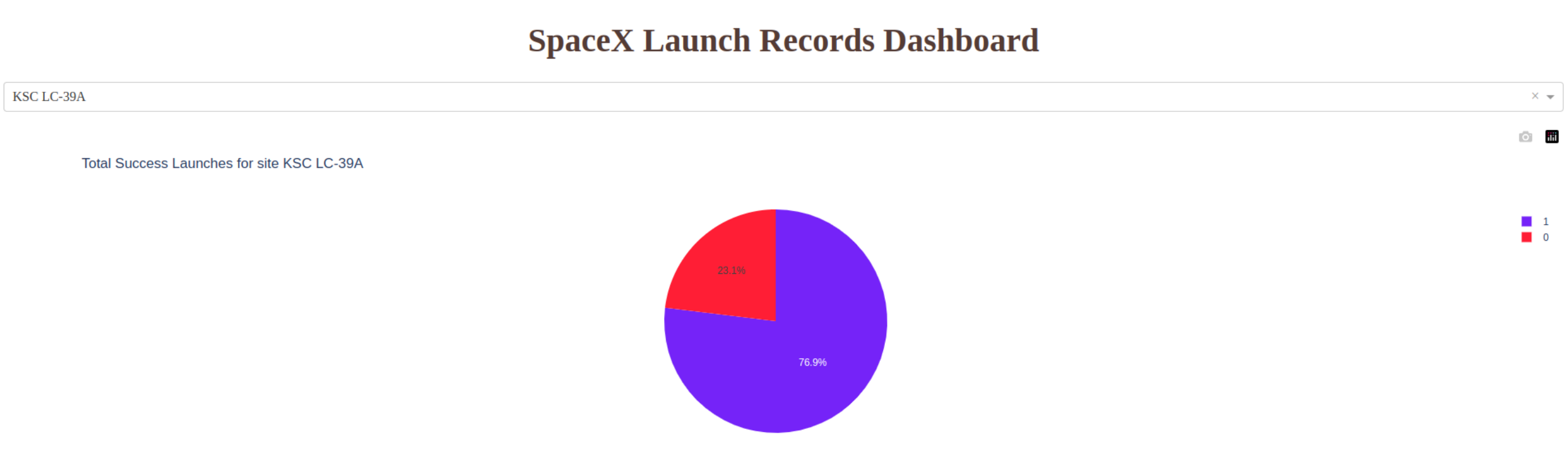
- Distance from nearest city: 17.69 KM.


Section 4
Build a Dashboard with Plotly Dash

Results – Interactive: Pie Chart



Results – Interactive: Highest success rate site pie chart



Results – Interactive: Slider & Scatter Plot

Section 5

Predictive Analysis (Classification)

Results – Predictive Analysis

In-Sample (80%): 10-Fold Cross Validation
Out-of-Sample (20%): Retest using best hyperparameters

Logistic Regression	Support Vector Classifier	Decision Tree Classifier	K Neighbors Classifier
• Train Score: 0.8464 • Test Score: 0.8333 • TP: 12 • FN: 0 • FP: 3 • TN: 3	• Train Score: 0.8482 • Test Score: 0.8333 • TP: 12 • FN: 0 • FP: 3 • TN: 3	• Train Score: 0.8768 • Test Score: 0.6666 • TP: 9 • FN: 3 • FP: 3 • TN: 3	• Train Score: 0.8482 • Test Score: 0.8333 • TP: 12 • FN: 0 • FP: 3 • TN: 3

Conclusions

- **Launch Success**: The probability of success is contingent upon the specific launch site and the optimal payload range, which is defined as 2,000 to 5,000 kilograms.
- **Orbit Types**: It can be observed that specific orbital paths, such as ES-L1 and GEO, demonstrate a higher degree of success.
- **Predictive Analysis:** Three out of four machine learning models have been demonstrated to effectively predict SpaceX's first stage reuse, thereby illustrating the potential value of such forecasting techniques.

The insights gained from this analysis can assist SpaceY in optimising its launch strategies, with a particular focus on identifying more successful launch sites and payload configurations. By employing analogous machine learning models, SpaceY could enhance its predictive capabilities, thereby optimising decision-making and operational efficiency.

Appendix

- The Decision Tree Classifier produces disparate scores when the Jupyter notebook is executed anew.

- In addition to the Decision Tree Classifier, all other models are equally suitable for predicting landing success with a test precision of 0.8333.

Thank you!