IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Arturo Iván Martínez Burgos
10th January, 2025

Outline

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

Executive Summary (1)

Summary of methodologies

For this analysis, we implemented different steps and advanced techniques used in data science, to demonstrate the reliability and usability of this type of analysis:

Data collection and cleaning Methods:

In this project, various techniques were employed for data collection and cleaning, to ensure data quality and accuracy. The main steps include:

- *Data Collection:*
  - Data Sources: Data was collected from various sources, including public databases, APIs (SpaceX API), and CSV files (some of them were web-scraped from a Wikipedia list: List of Falcon 9 and Falcon Heavy launches).
  - Data Integration: Data from multiple sources was integrated to create a unified and coherent dataset to cover the needs of this analysis.
- *Data Cleaning:*
  - Handling missing values: this type of values were identified and handled using techniques to remove imcomplete or useless records.
  - Removing duplicates: Duplicate records were removed to avoid biases in the analysis.
  - Error correction: Typographical error were corrected, and data formats were standardized.
  - Normalization and Scaling: Numerical values were normalized and scaled to ensure that all features have a comparable influence on the implemented machine learnig models.

Executive Summary (2)

Artificial Intelligence Methods:

For analysis and prediction, several machine learning models were implemented using advanced techniques:

- *Model Selection:*
  - Models used: Various ML models were used, including decision trees, support vector machines (SVM), logistic regression, and k-nearest neighbors (KNN).
  - Hyperparameter Optimization: Grid search with cross-validation (GridSearchCV) was employed to find the best hyperparameters for each model.
- *Training and Evaluation:*
  - Handling missing values: this type of values were identified and handled using techniques to remove imcomplete or useless records.
  - Cross-Validation: Cross-validation was used to assess the robustness and generalization of the models.
  - Evaluation Metrics: Metrics such as accuracy, precision, recall, and F1-score were used to evaluate model performance.
- *Model Implementation*:
  - Model Fitting: Models were fitted to the training data and evaluated using the testing data.
  - Best Model Selection: The model with the best performance based on evaluation metrics was selected.

Executive Summary (3)

Summary of results

This systematic and rigorous approach to data collection, cleaning, and analysis, combined with artificial intelligence techniques, ensures that the results obtained are accurate and reliable, providing valuable insights and predictions for the project.

Also, using different ML models, we selected the best, based on their accuracy scores, using evaluation methods:

Models used; **a) Decision Trees, b) Support Vector Machines (SVM), c) Logistic Regression, d) K-Nearest Neighbors (KNN)**.

For each model used, its accuracy was calculated to determine the percentage of correct predictions.

Precision, Recall, and F1-Score: Metrics used to calculate and evaluate the performance of the models in terms of precision and sensitivity.

Confusion Matrix: to visualize the performance of the models.

Decision Trees	Support Vector Machine (SVM)	Logistic Regression	K-Nearest Neighbors (KNN)
Accuracy: 85%	Accuracy: 88%	Accuracy: 83%	Accuracy: 80%
Precision: 0.84	Precision: 0.87	Precision: 0.82	Precision: 0.79
Recall: 0.85	Recall: 0.88	Recall: 0.83	Recall: 0.80
F1-Score: 0.84	F1-Score: 0.87	F1-Score: 0.82	F1-Score: 0.79

Introduction

- For this project, we wanted to analyze the outcomes of all the launches that SpaceX have had during the years, since the company started its develops and creations in the field of space exploration.
- Here, after extracting the information from different sources (APIs requests and webscraping using Python), we want to clean the information in order to abilitate it to its usage in a Machine Learning model, to predict different outcomes in the future, using the actual information of the past landings of the company.

Section 1

Methodology

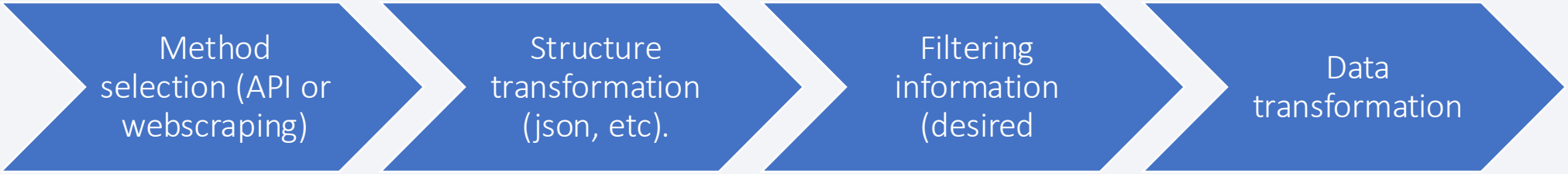
Methodology

Executive Summary

- Data collection methodology
- Data wrangling (cleaning and filtering)
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

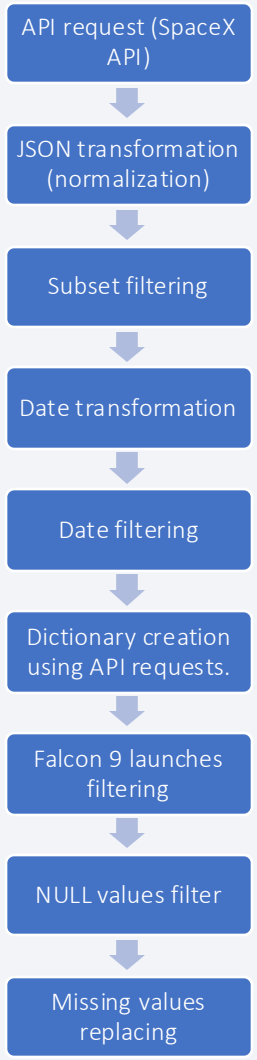
- At first, data was gathered from SpaceX API and Beautiful Soup (Python library for webscraping).
- Using API calls, BeautifulSoup methods and transformations to get the desired data, we got a table of information for our analysis.
- Filtering the information required, the useless rows were discarded.



Data Collection – SpaceX API

- Data was gattered using API calls and then cleaned to its usage in the Machine Learning model.

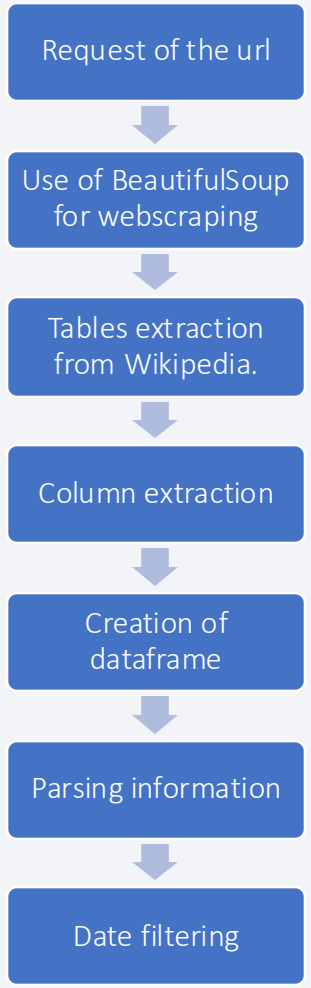
GitHub URL of completed SpaceX API calls notebook



Data Collection - Scraping

- Data extraction from Wikipedia, using Beautiful Soup as web-scraping library in Python.

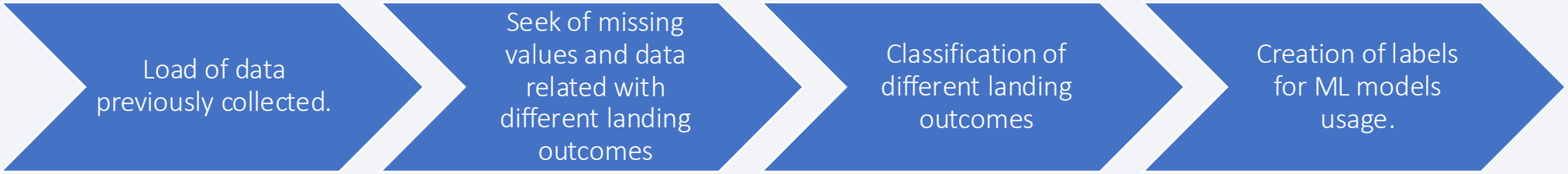
GitHub URL of completed web scraping notebook



Data Wrangling

- Describe how data were processed
- At first, using the extracted data from the APIs requests and web-scraping methods, the missing values and types of values were identified.
- Labels of each class of landing were created, using data related with all the possible outcomes in landing history of SpaceX.

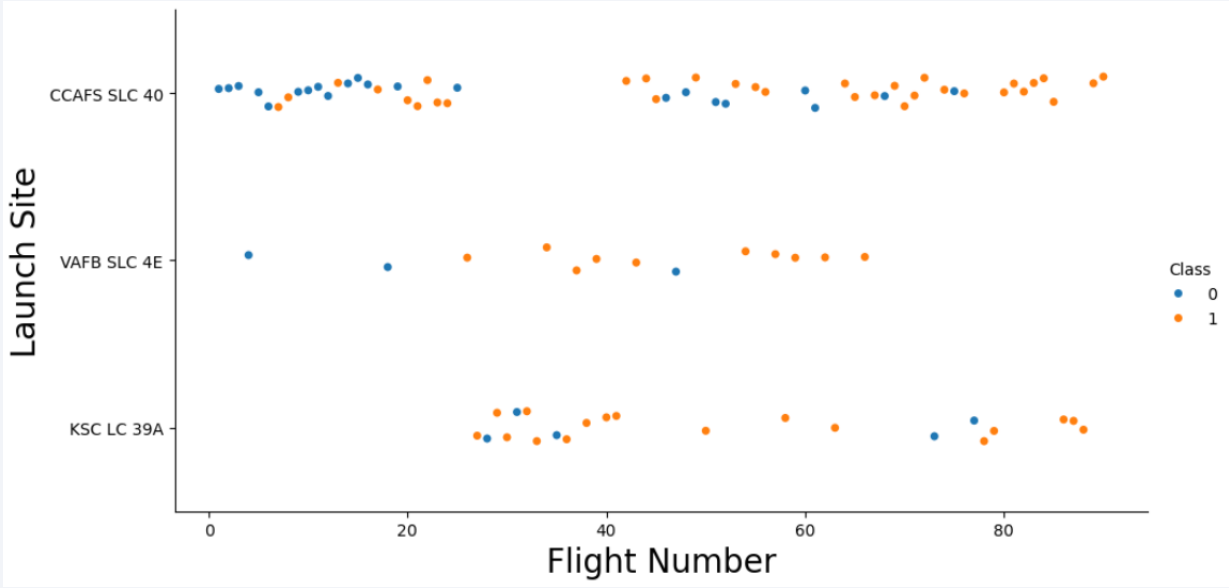
GitHub URL of completed data wrangling notebook



EDA with Data Visualization

- In this analysis, scatter plot chats, bar charts and line charts were used to visualize data and its correlation using different points of view, to get information of which variables were the best to use in the Machine Learning models.

GitHub URL of completed EDA with data visualization notebook



E.G. Scatter plot to view correlation between Launch Site and Flight Number.

EDA with SQL

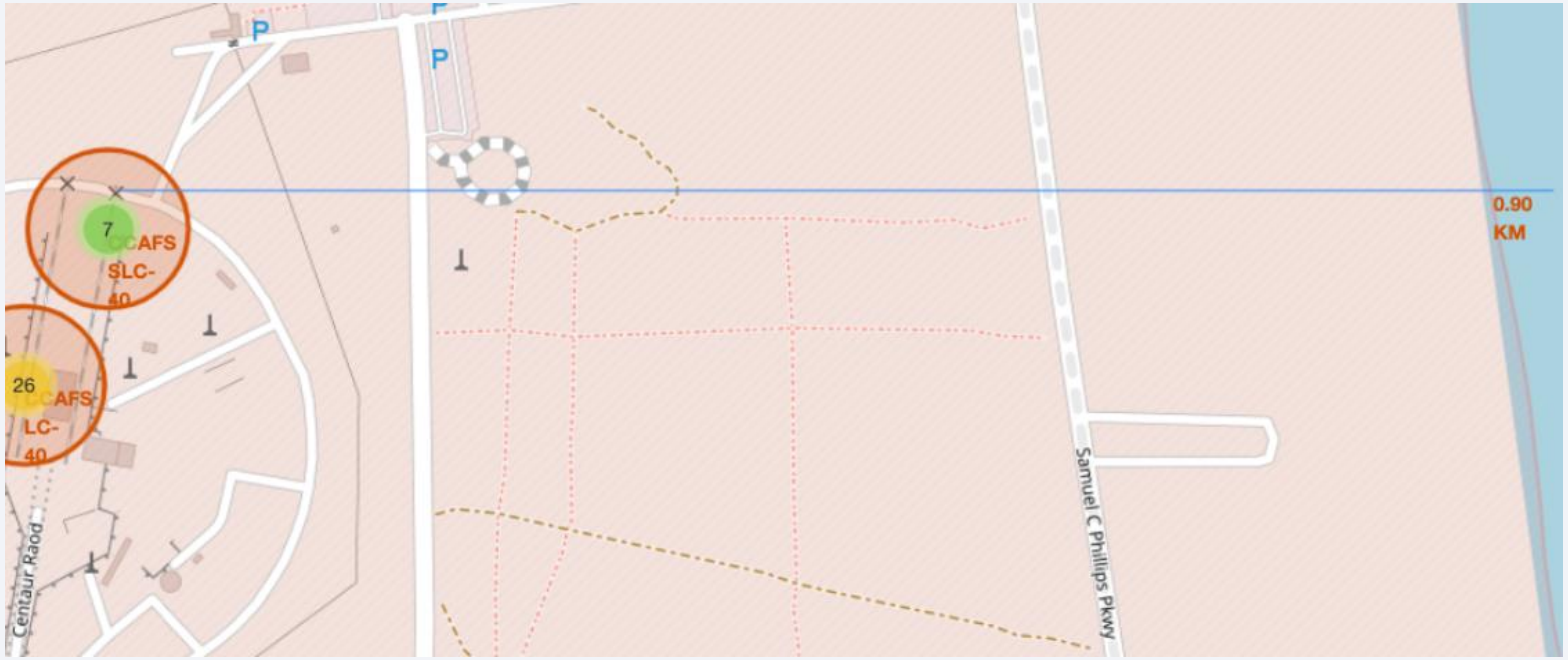
- For this notebook, I manipulated and filtered the data previously gattered using SQLite in Python.
- Main queries executed in the notebook:
  - Creation of new table using existing table as reference: "create table NAME2 as select * from NAME1
  - View distinct values in a determined column: "select DISTINCT COLUMN-NAME from TABLE"
  - Display a limit for a desired group of columns: "select * from TABLE limit #NUMBER"
  - SUM and AVERAGE for a determined value: "select SUM(COLUMN)" from TABLE
  - MAX and MIN for a determined feature: "select MIN(COLUMN) from TABLE"

GitHub URL of completed EDA with SQL notebook

Build an Interactive Map with Folium

- Using Folium library for Python, an interactive map was created to show the launching sites for the Falcon 9 of SpaceX, also showing some extra data, for example the distance between the launching sites and the beach, that is almost a km.

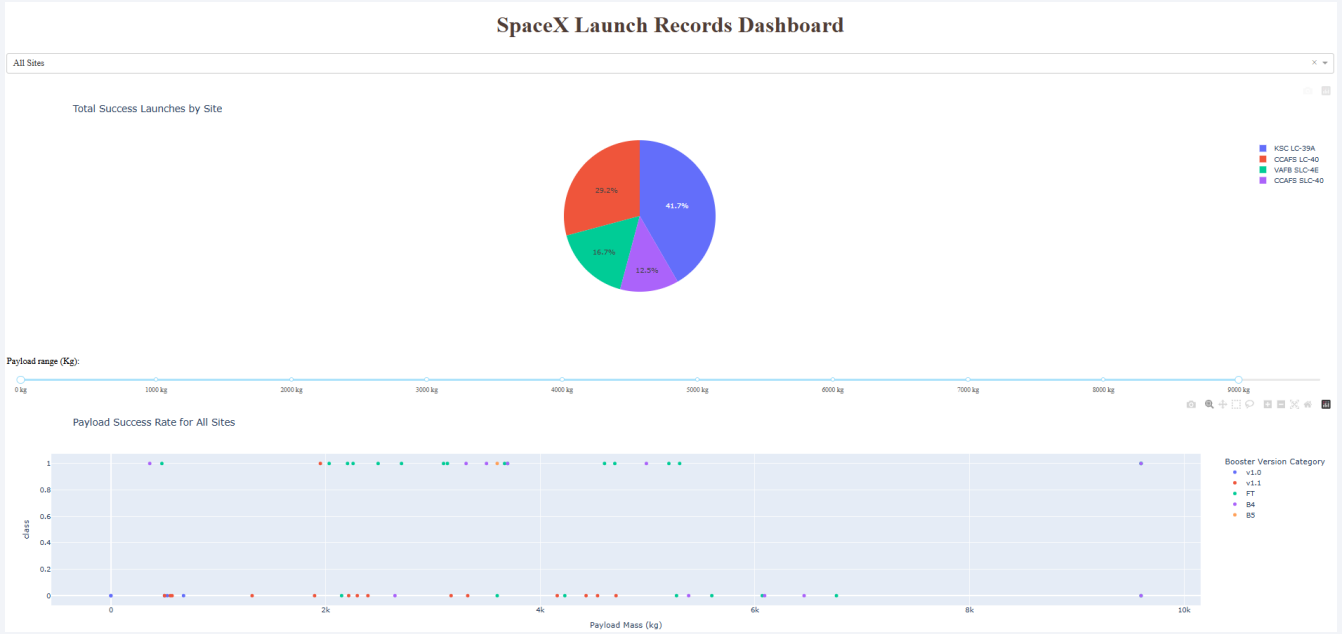
GitHub URL of completed interactive map with Folium map.



Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
- In this development, I implemented a dashboard using Dash, an interactive liibrary of Python to create dynamic graphs; using a pie chart and a scatter plot, with a slide-range selector, it can graph with dynamism the information about the landings of Falcon rockets of SpaceX.

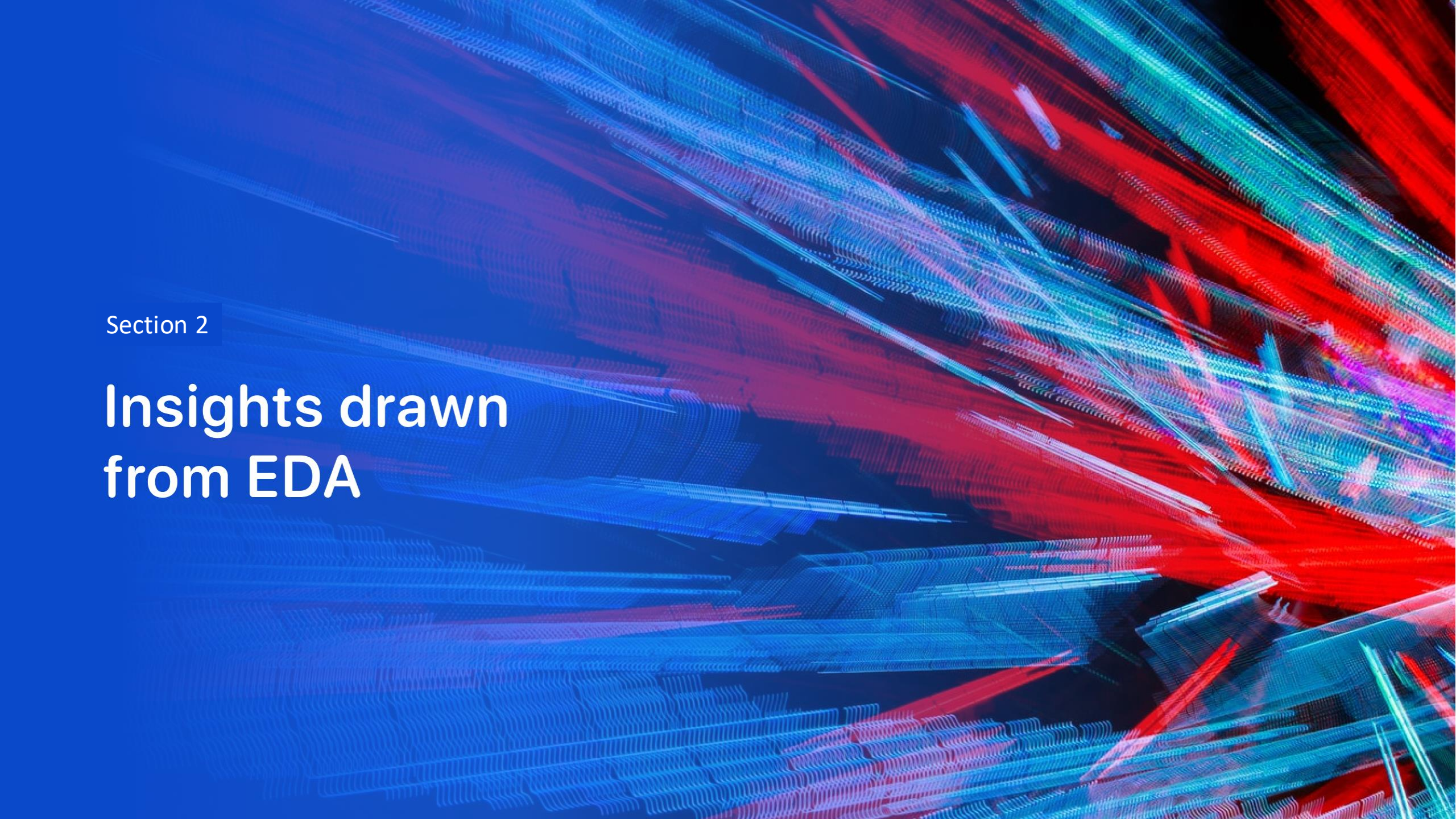
GitHub URL of completed Plotly Dash lab.



Predictive Analysis (Classification)

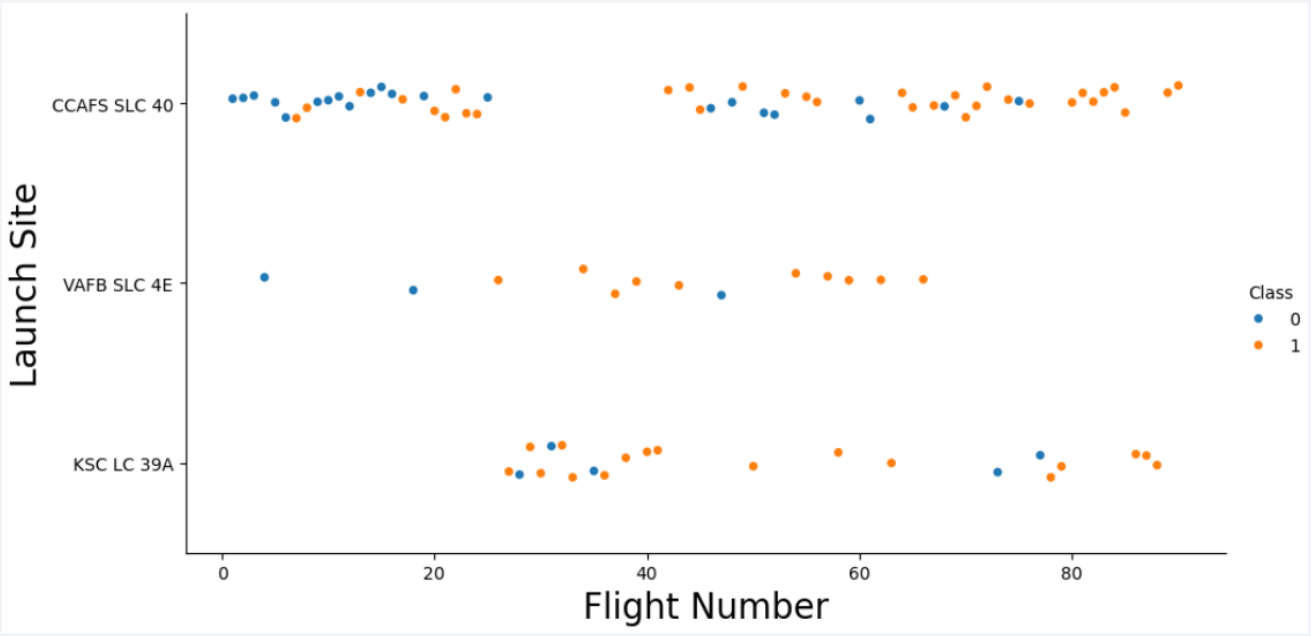
- For this analysis, the model is loaded, then is preprocessed with the StandardScaler method of Python; the database is splitted into test and train sets, to ensure the training model is efective with unseen cases.
- Then the desired model is selected to be trained using GridSearchCV method, then is evaluated with the score method, to adquire the accuracy of the model. After that, the confusion matrix can be created.

GitHub URL of completed predictive analysis lab.


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

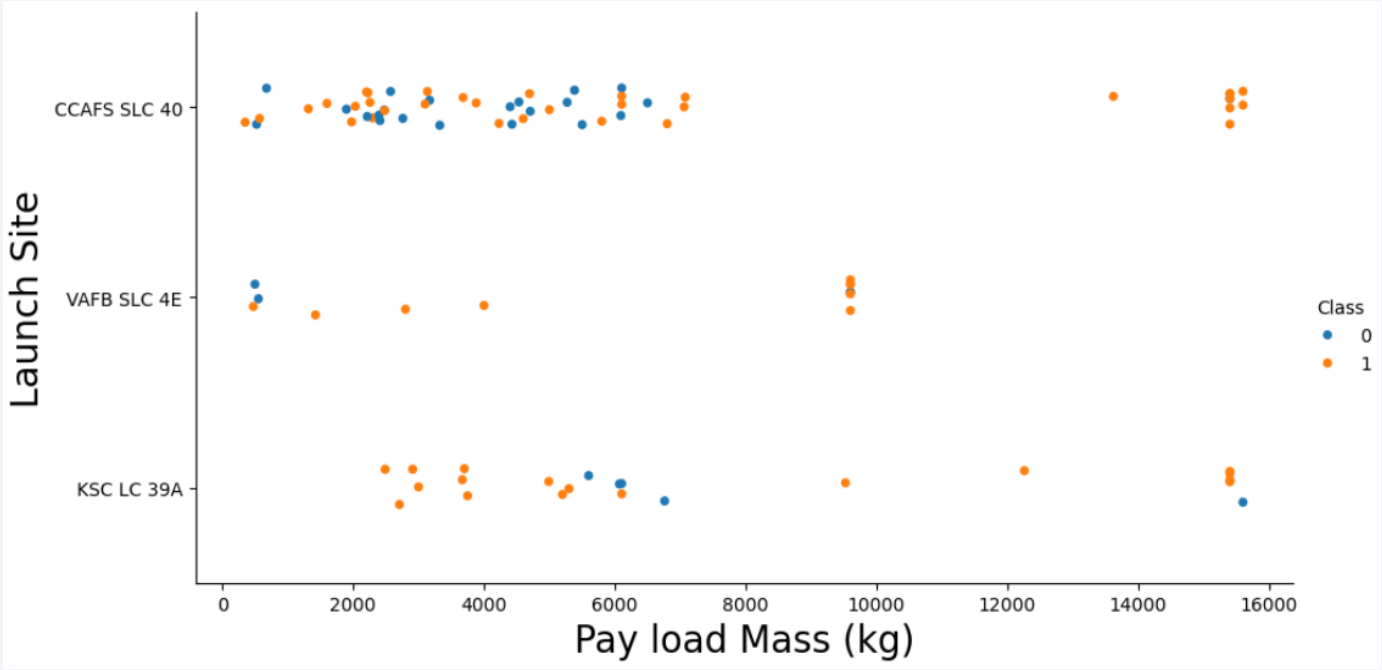
- Scatter plot of Flight Number vs. Launch Site
- In this scatter plot, it can be seen the relation between these variables, the first launches were in the CCAFS SLC 40 site and weren't that successful as the last ones.



Payload vs. Launch Site

- Scatter plot of Payload vs. Launch Site

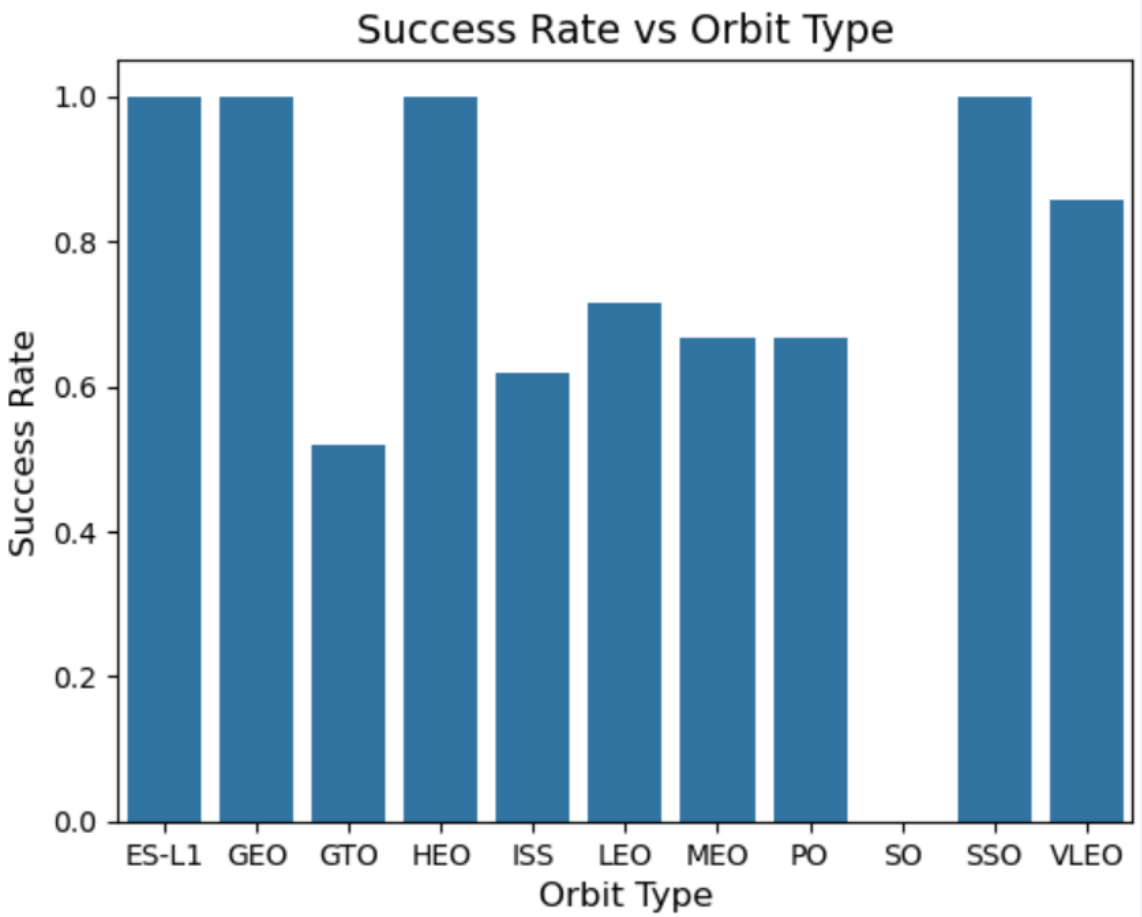
- In this graph, we can see that heavy pay load mass launches happened in the first launching site, while in the VAFB SLC 4E site, pay loads of more than 10 tons, didn't happened at all.



Success Rate vs. Orbit Type

- Bar chart for the success rate of each orbit type.

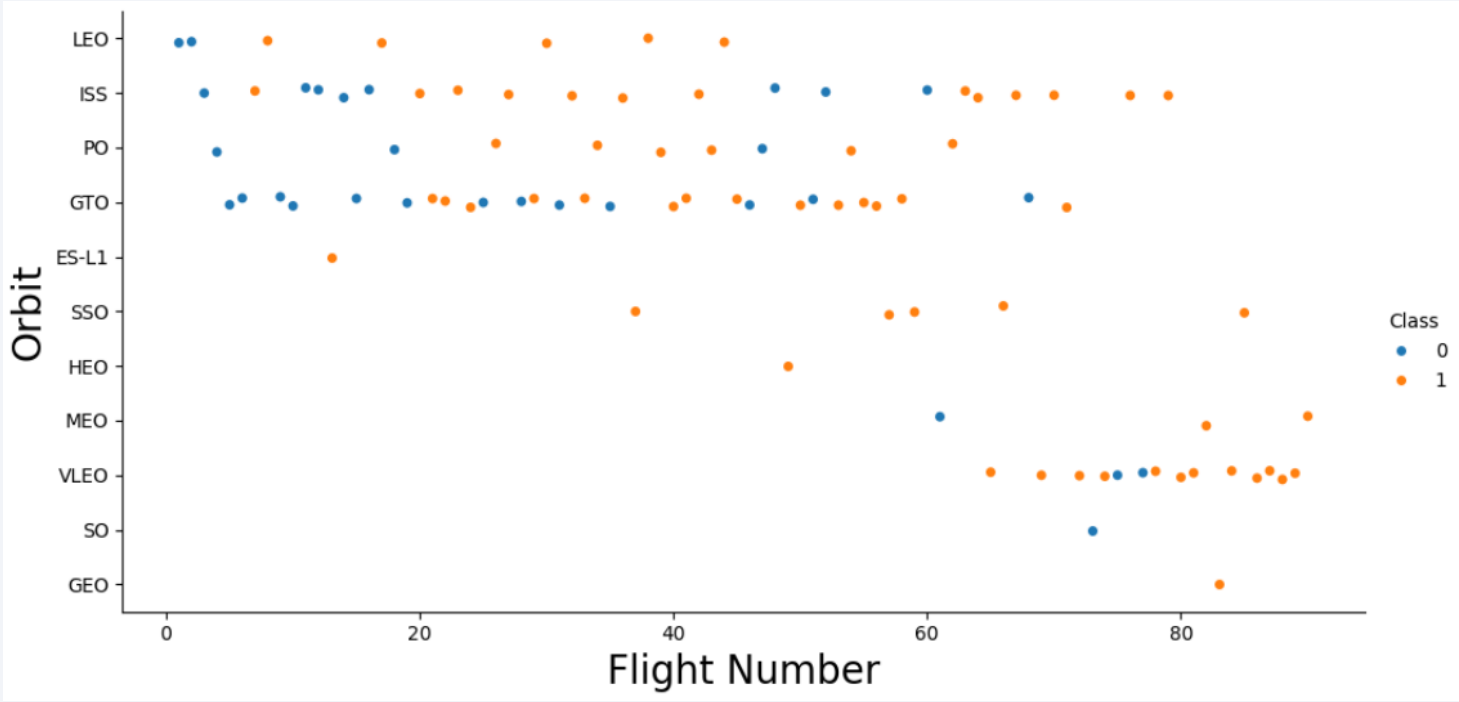
- In this graph, we can see that the most successful orbits of launching were ES-L1, GEO, HEO and SSO, while the SO orbit didn't have any successful launch.



Flight Number vs. Orbit Type

- Scatter point of Flight number vs. Orbit type

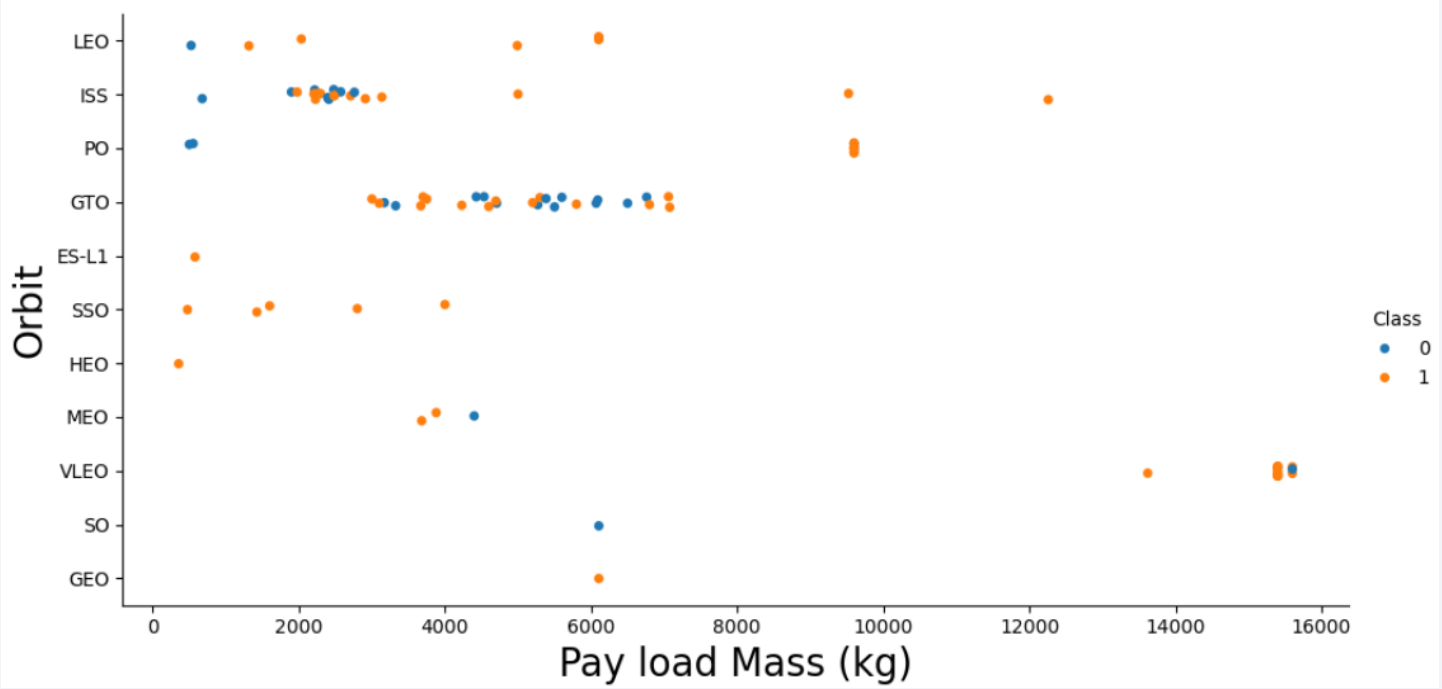
- In this graph we can see that the orbits ISS and GTO have been used through the entire history of the company for the launches, while the orbit VLEO has been used in the last ones.



Payload vs. Orbit Type

- Scatter point of payload vs. orbit type

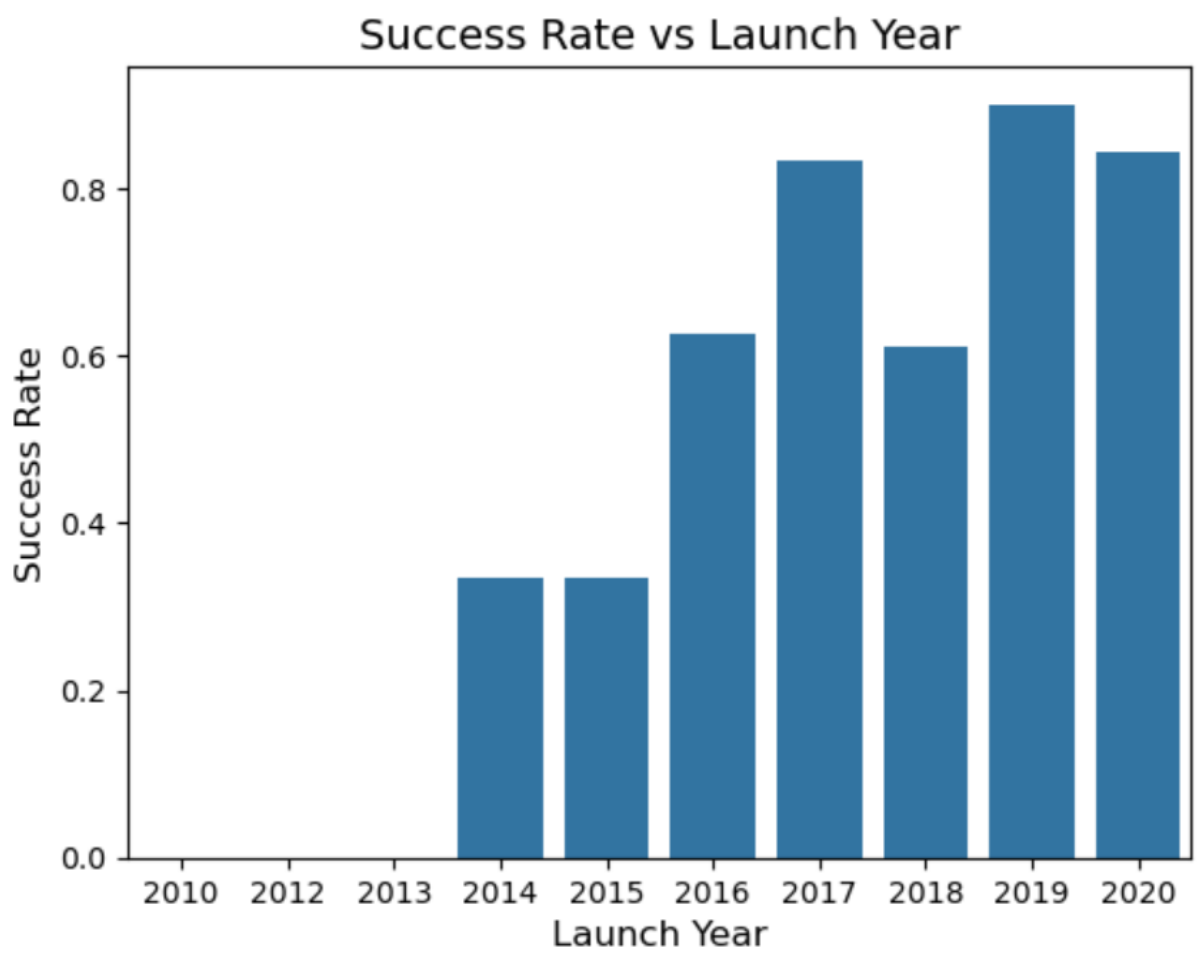
- In this graph we can see that the heaviest launches were loaded to the VLEO orbit.



Launch Success Yearly Trend

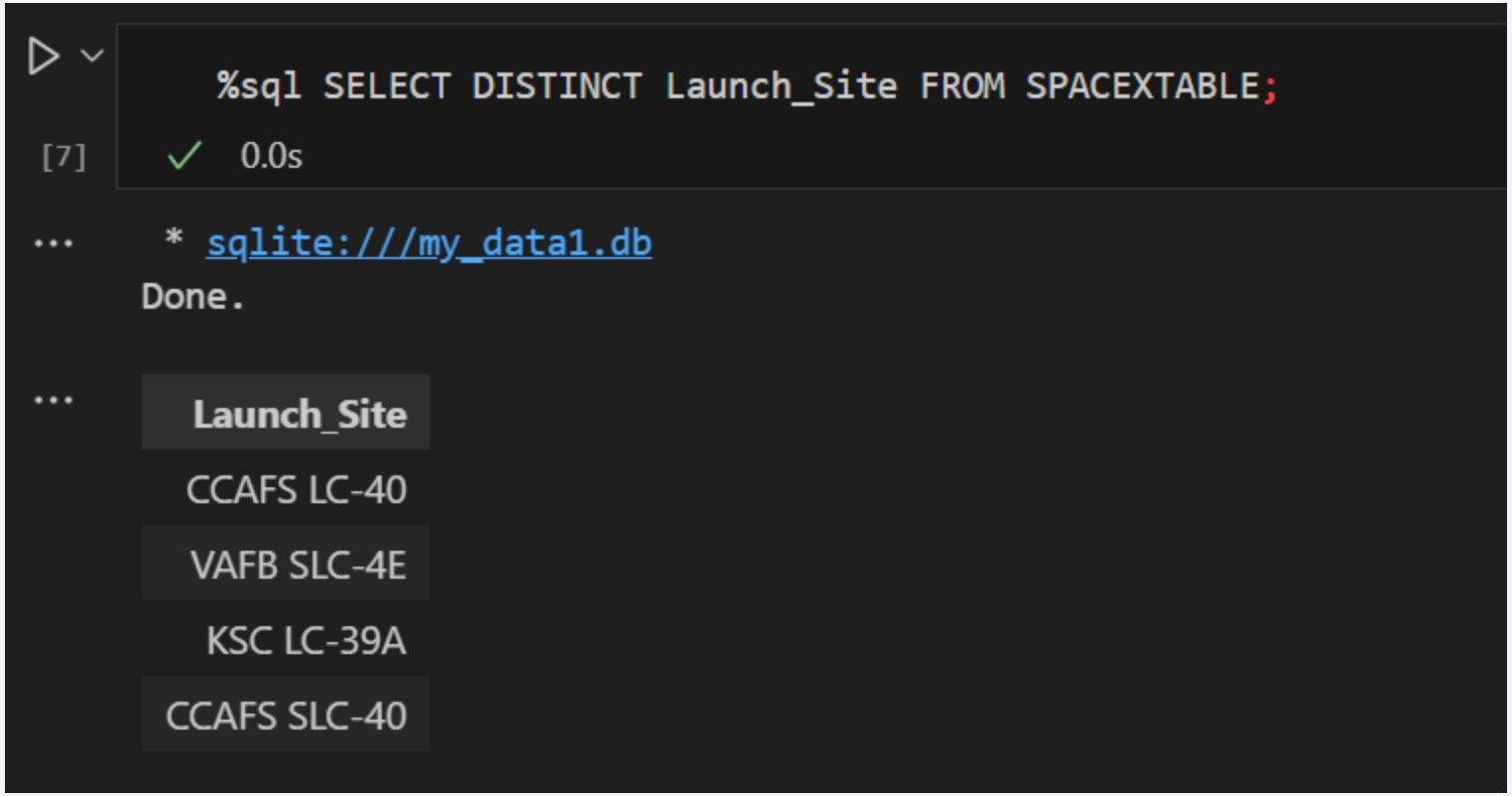
- Line chart of yearly average success rate

- With this graph, we can observe the success that SpaceX have had in the last years, comparing them to the first ones, where their launches didn't have to much success rate in comparison to the last 2.



All Launch Site Names

- In this query was selected the unique values for the Launch_Site column of the database, giving a result of 4 distinct values.



Launch Site Names Begin with 'CCA'

- In this query, all the launching sites beginning with CCA were selected, limiting the results to the first 5 rows.

```
query = "SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5"
result = %sql $query
result
```

[8] ✓ 0.0s

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

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

Total Payload Mass

- In this query, we calculated the total payload carried by boosters from NASA

```
query = "SELECT SUM(PAYLOAD_MASS__KG_) as Total_Payload FROM SPACEXTABLE WHERE Customer LIKE '%CRS%'"
result = %sql $query
total_payload = result[0]['Total_Payload']
print(total_payload)
```

[27] ✓ 0.0s

```
 * sqlite:///my_data1.db
Done.
48213
```

Average Payload Mass by F9 v1.1

- For this query, we calculated the average payload mass carried by booster version F9 v1.1

```
    query = "SELECT AVG(PAYLOAD_MASS__KG_) as Average_Payload FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'"
    result = %sql $query
    average_payload = result[0]['Average_Payload']
    print(average_payload)
```

[28] ✓ 0.0s

```
 * sqlite:///my_data1.db
Done.
2928.4
```

First Successful Ground Landing Date

- For this query, we found the date of the first successful landing outcome on ground pad.

```
query = """SELECT MIN(Date) AS First_Successful_Landing_Date
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (ground pad)'"""
result = %sql $query
average_payload = result[0]['First_Successful_Landing_Date']
print(average_payload)
```

[31] ✓ 0.0s

```
 * sqlite:///my_data1.db
Done.
2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- For this query, we got the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000.

```
query = """SELECT Booster_Version FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)'
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000"""
result = %sql $query
boosters = [row['Booster_Version'] for row in result]
print(boosters)
```

[32] ✓ 0.0s

```
 * sqlite:///my_data1.db
Done.
['F9 FT B1022', 'F9 FT B1026', 'F9 FT  B1021.2', 'F9 FT  B1031.2']
```

Total Number of Successful and Failure Mission Outcomes

- For this query, it was calculated the total number of successful and failure mission outcomes.

```
    query = """SELECT Mission_Outcome, COUNT(*) as Total_Count
    FROM SPACEXTABLE
    GROUP BY Mission_Outcome"""
    result = %sql $query
    outcome_counts = {row['Mission_Outcome']: row['Total_Count'] for row in result}
    print(outcome_counts)
[34]  ✓ 0.0s
...  * sqlite:///my_data1.db
Done.
{'Failure (in flight)': 1, 'Success': 98, 'Success ': 1, 'Success (payload status unclear)': 1}
```

Boosters Carried Maximum Payload

- For this query, we listed the names of the booster which have carried the maximum payload mass

```
query = """SELECT Booster_Version FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)"""
result = %sql $query
booster_versions = [row['Booster_Version'] for row in result]
print(booster_versions)
```

[35] ✓ 0.0s

```
 * sqlite:///my_data1.db
Done.
['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',
```

2015 Launch Records

- For this query, we needed to list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
query = """
SELECT
    strftime('%m', Date) AS Month,
    Landing_Outcome,
    Booster_Version,
    Launch_Site
FROM
    SPACEXTABLE
WHERE
    substr(Date, 0, 5) = '2015' AND
    Landing_Outcome LIKE 'Failure (drone ship)%'
"""
result = %sql $query

import calendar

result_with_month_name = []
for row in result:
    month_number = int(row['Month'])
    month_name = calendar.month_name[month_number]
    result_with_month_name.append((month_name, row['Landing_Outcome'], row['Booster_Version'], row['Launch_Site']))

result = result_with_month_name
result = pd.DataFrame(result, columns=['Month', 'Landing_Outcome', 'Booster_Version', 'Launch_Site'])
result = result.reset_index(drop=True)
result
```

	Month	Landing_Outcome	Booster_Version	Launch_Site
0	January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
1	April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- For this query, we ranked 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.

```
query = """
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
"""
result = %sql $query
result
```

[52] ✓ 0.0s

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

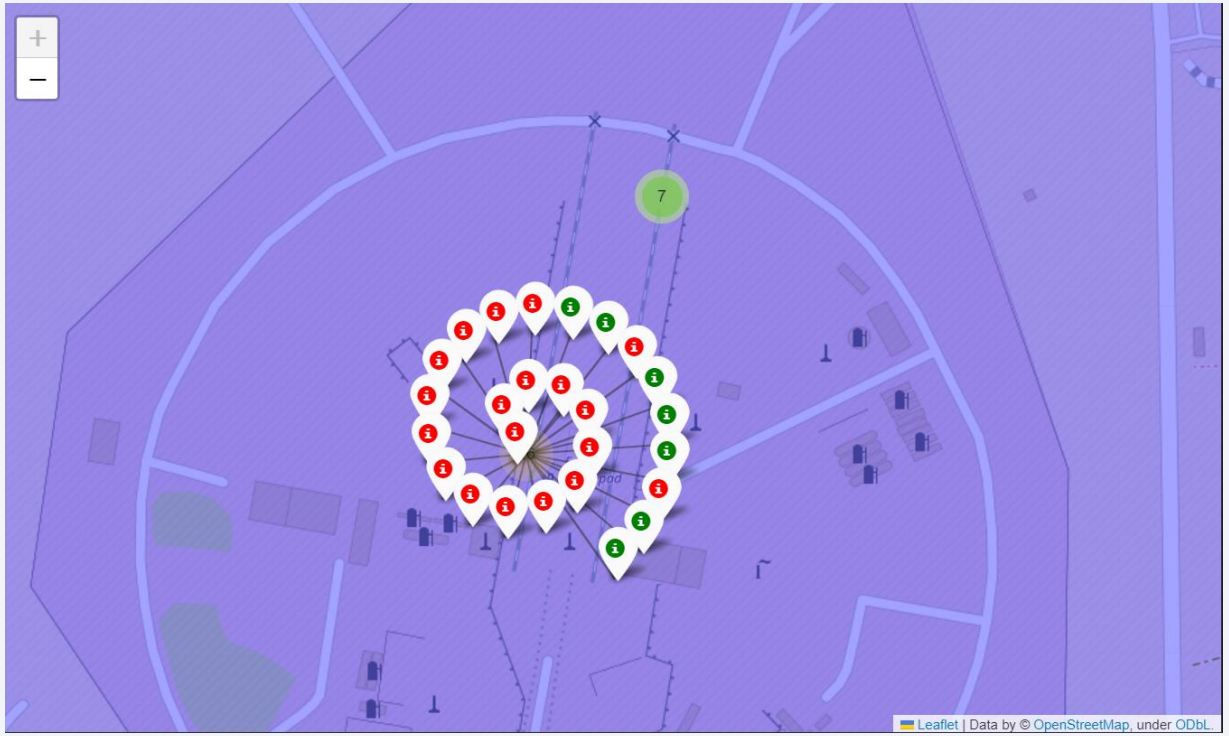
Launch Sites Proximities Analysis

Folium Map – Location Markers

- In this folium map we can explore a screenshot that include all launch sites' location markers on the global map
- This elements contain the name of each location site, so the user can find them in the global map.

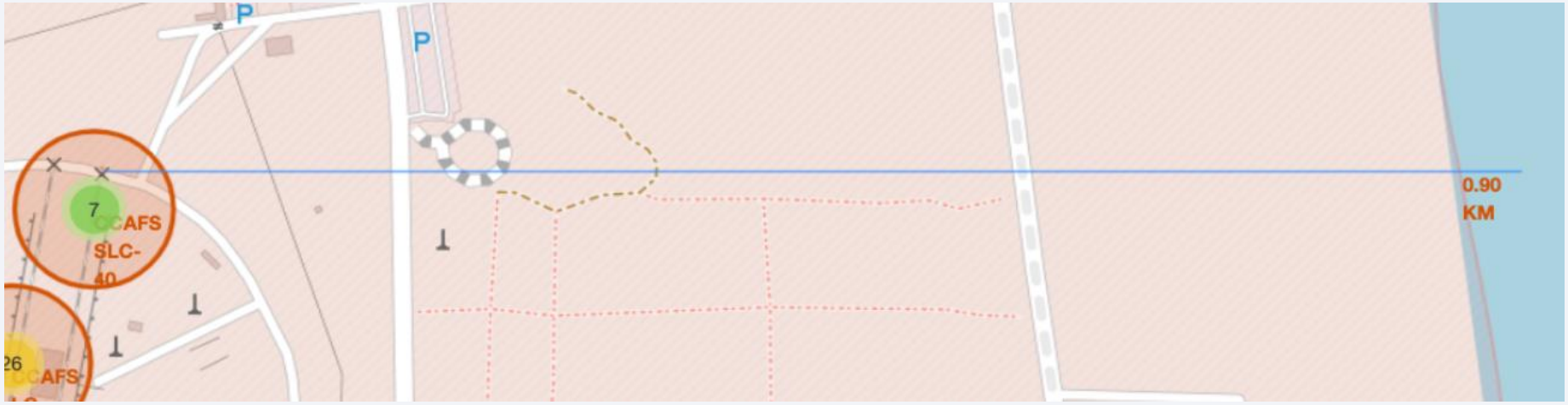
Folium Map – Color Labeled launch outcomes.

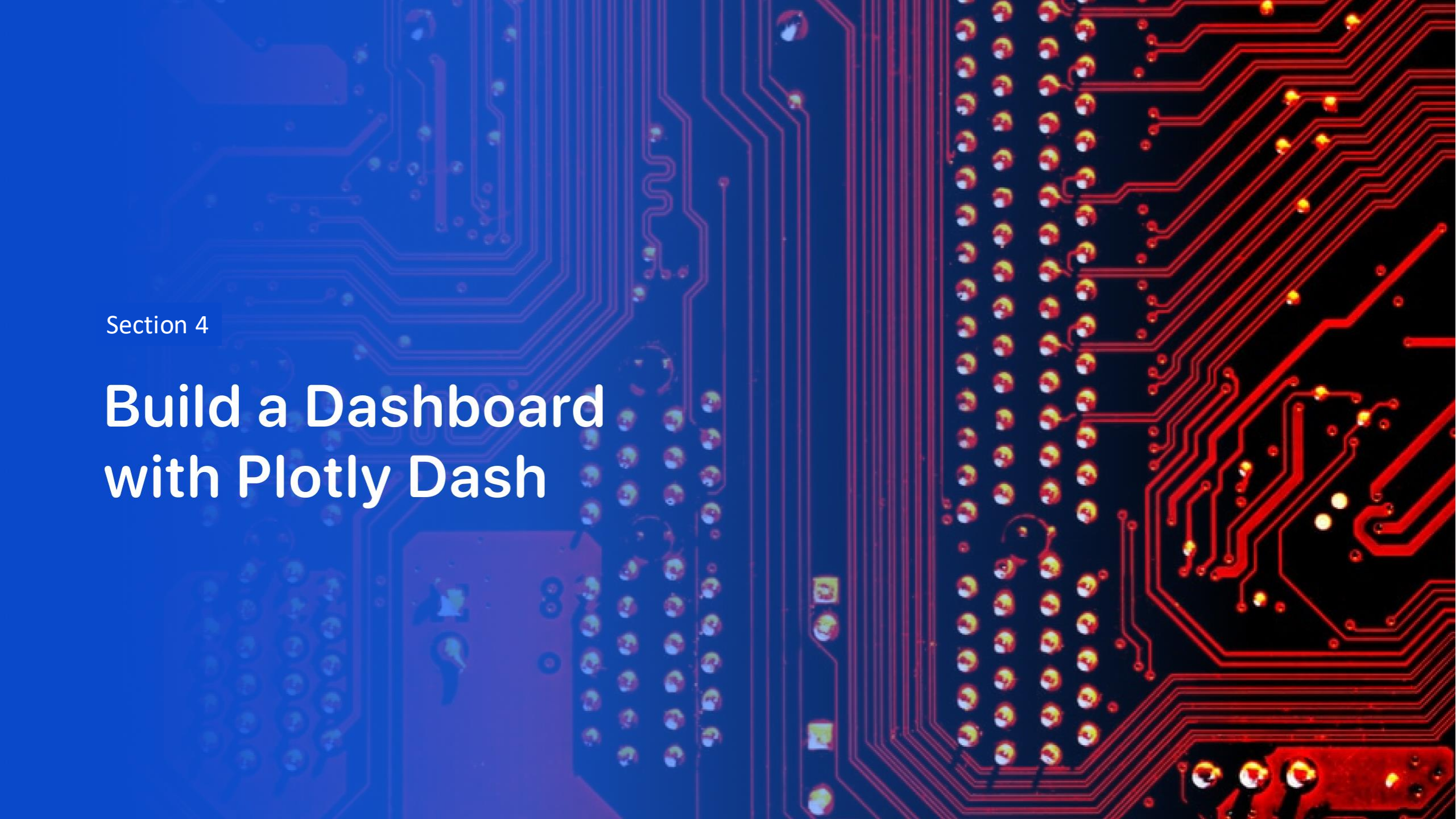
- In this folium map we can explore a screenshot that shows the color-labeled launch outcomes on the map.
- This elements show us the different outcomes on each launching site, useful to see easily which launching site has more probabilities of having a successful outcome.



Folium Map – Distance and proximities

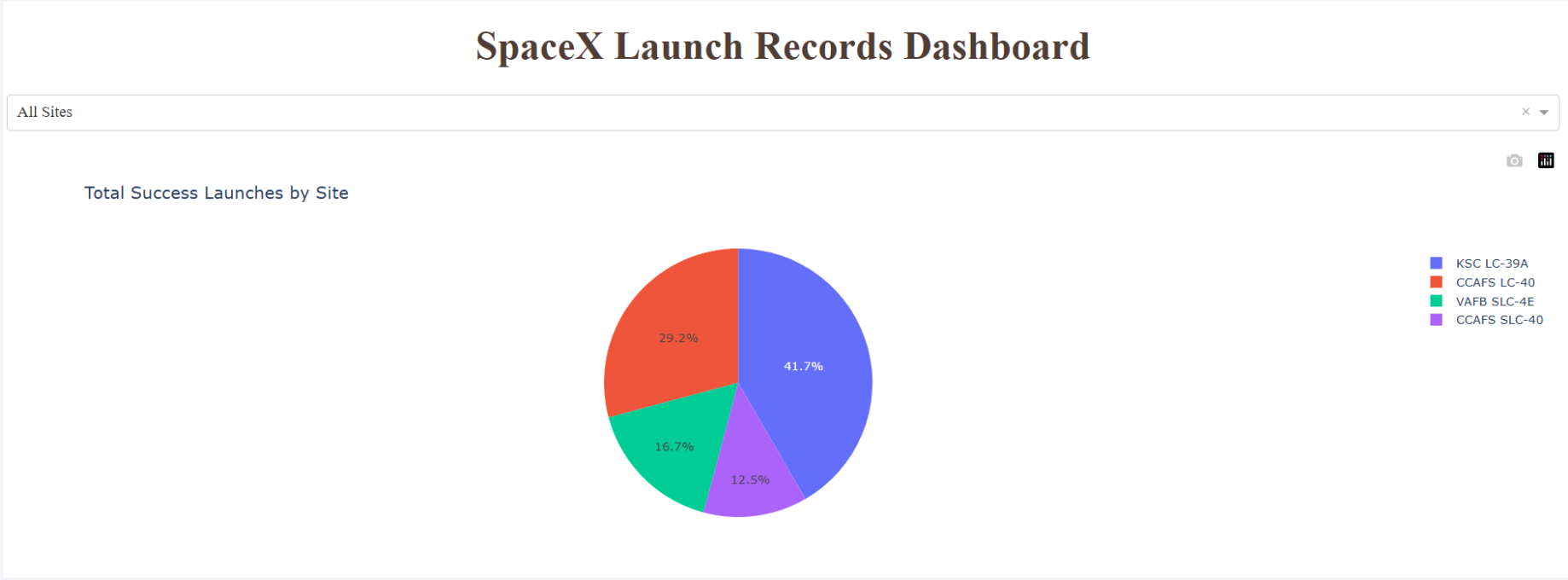
- In this folium map we can see a screenshot of a selected launch site to its proximities such as highway, coastline, with distance calculated and displayed
- Here we can observe que CCAFS SLC-40 launching site, with a distance calculated to the sea, with almost 1 kilometer.




Section 4
Build a Dashboard with Plotly Dash

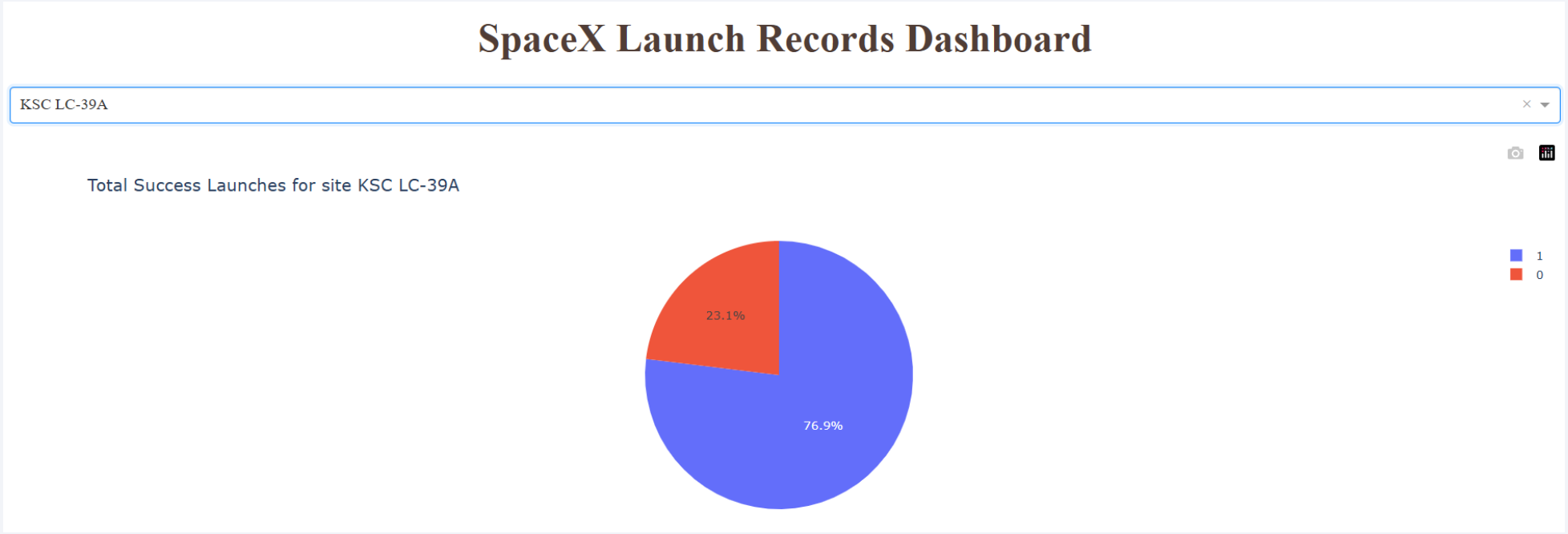
Dashboard – Success rate in ALL launching sites

- In this screenshot we can see the launch success count for all sites, in a dynamic piechart created with Dash.
- This can be useful to show in an interactive way the success rate for each landing site in a dynamic way.



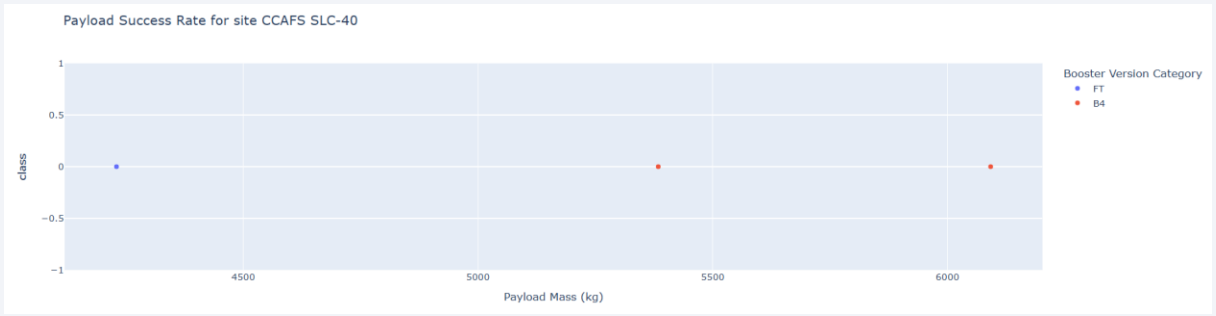
Dashboard – Highest launch success ratio

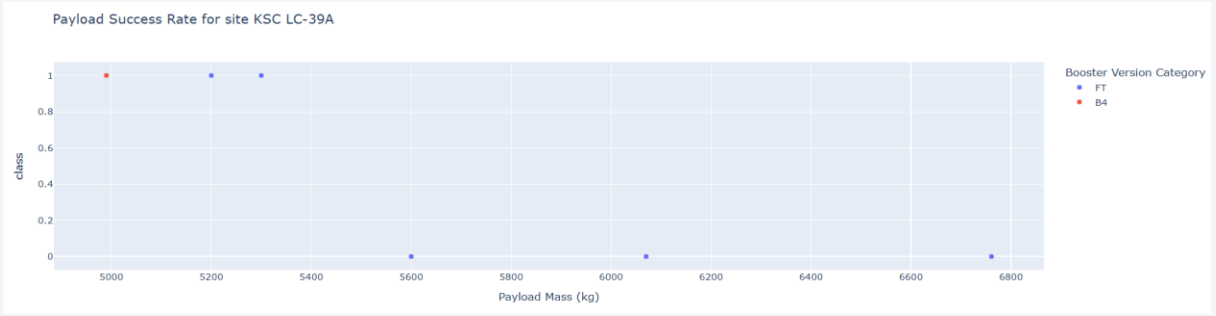
- In this screenshot we can see the piechart for the launch site with highest launch success ratio.
- In this piechart, we can see the success ratio of KSC LC-39A, the launching site with the highest launch success ratio and its relationship with the failure ratio.

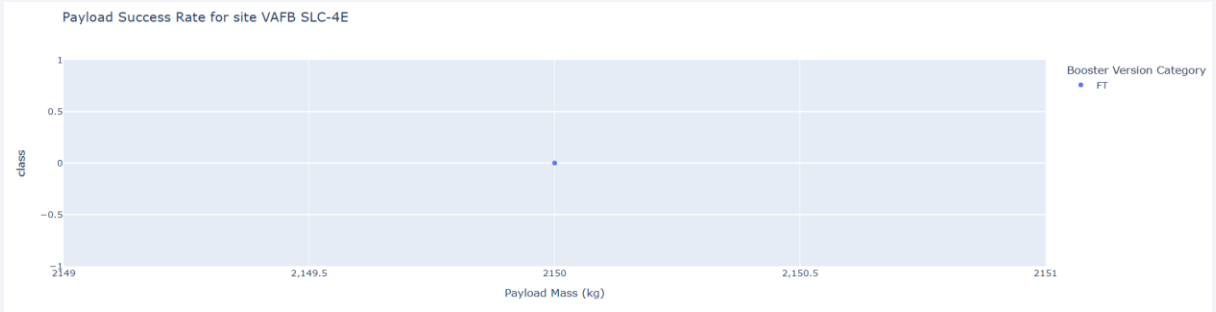


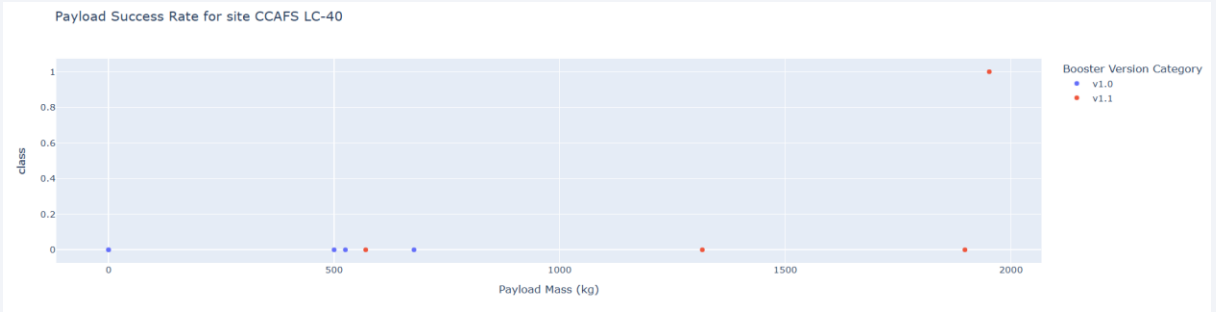
Dashboard – Scatter plot and Range slider

- Here we can see different screenshots of Payload vs. Launch Outcome scatter plot for some sites, with different payload selected in the range slider.
- With this range slider, we can select the desired range of payload and Dash creates in a dynamic way the scatter plot, to not render again the entire page.









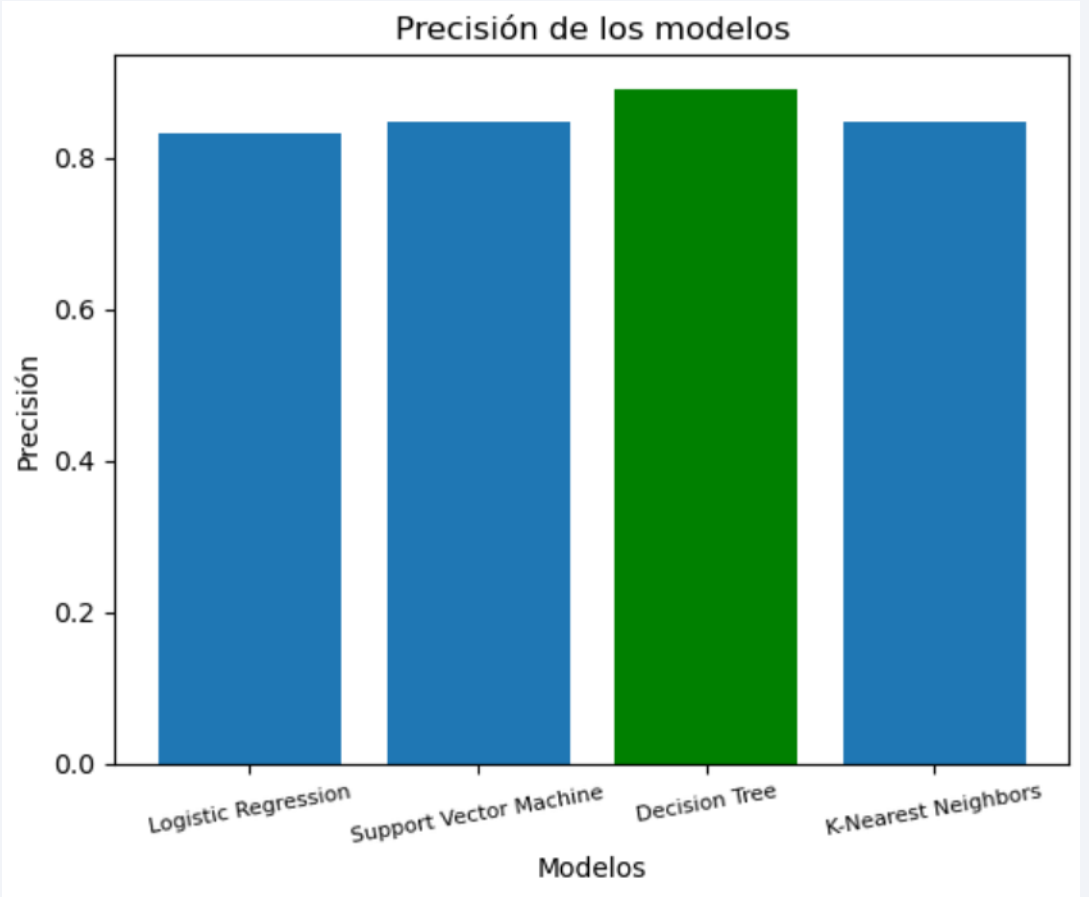
Section 5

Predictive Analysis (Classification)

Classification Accuracy

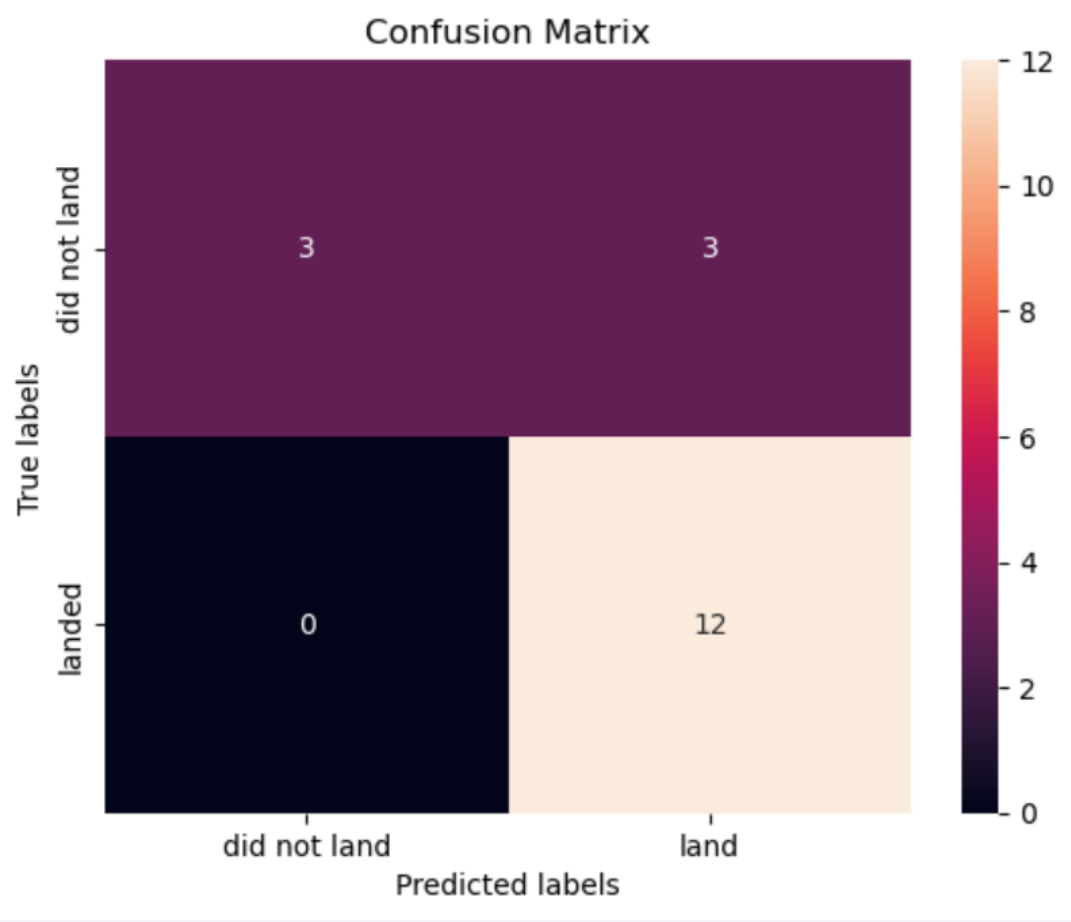
- In this chart, we can visualize the built model accuracy for all built classification models, in a bar chart

- This bar chart can help us to find which model has the highest classification accuracy; in this case, is the Decision Tree.



Confusion Matrix

- In this figure, we can see the confusion matrix of the best performing model, which is the Decision Tree, with an accuracy of 0.89.



Conclusions

- Using different Python libraries for dynamic plotting and dashboards, we can facilitate the creation and understanding of charts and graphs to get insights more easily and get conclusions more efficiently.
- The GridSearchCV library in Python is an efficient tool to find the best parameters to our machine learning models, if we don't know a lot of their configuration and hypertuning.
- Visual Tools like Confusion Matrix, and charts with the accuracy of our models, can help us to find the best Machine Learning model to use in our future findings related with the project that we are working for.

Appendix

- GIT Repository Link

Thank you!