IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

EDWIN ISAAC SOTO COSSIO

15/11/23

CONTENTS OF THIS TEMPLATE

EXECUTIVE SUMMARY

Summary of methodologies

- Data Collection through API:Obtaining essential data through the SpaceX Application Programming Interface (API).
- Data Collection with Web Scraping:Additional extraction of relevant information using web scraping techniques to ensure a comprehensive database.
- Data Wrangling:Thorough processing and cleaning of data to ensure coherence and reliability in the analysis.
- Exploratory Data Analysis (EDA) with SQL:Utilizing SQL queries to explore structured data and gain fundamental insights.
- Exploratory Data Analysis (EDA) with Data Visualization:Employing data visualization tools to graphically represent patterns, trends, and relationships in the data.
- Interactive Visual Analytics with Folium:Developing interactive visualizations using Folium to map launch locations and enhance spatial understanding.
- Machine Learning Prediction:Training machine learning models to predict the success of the first-stage booster landing.

Summary of all results

- Exploratory Data Analysis result: Identifying significant patterns and trends in SpaceX launch strategies through in-depth exploratory data analysis.
- Interactive analytics: Providing visual representation through screenshots to interactively illustrate key data and findings.
- Predictive Analytics result: Describing the results obtained through the application of predictive models, highlighting the accuracy in predicting the success of the first-stage booster landing.

Introduction

In the dynamic realm of commercial space exploration, emerging competitors drive a race for affordable travel. This project focuses on SpaceY, a newcomer challenging SpaceX, a key player. SpaceX, led by Elon Musk, not only achieved milestones like sending spacecraft to the International Space Station but also revolutionized cost efficiency through reusable rocket boosters.

The primary project challenge is determining SpaceY's rocket launch pricing strategy. SpaceX's innovative approach involves reserving fuel for first-stage booster landings, reducing launch costs. Understanding crucial factors for booster landing success is vital for SpaceY to competitively bid against SpaceX.

The project aims to optimize SpaceY's launch costs by drawing inspiration from SpaceX's approach. Additionally, it seeks to enhance booster landing success predictions, identifying key parameters and developing a precise predictive model. Finally, the project aims to bolster SpaceY's competitive position against SpaceX by providing insights for informed bidding, using booster landing predictions to estimate launch costs effectively in the dynamic commercial space exploration landscape.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

The data collection process involved sourcing data from diverse channels, leveraging both public APIs and web scraping techniques. Additionally, datasets from reputable space-related sources were acquired to ensure comprehensive coverage.

- Perform data wrangling

Data wrangling was executed to enhance data quality and coherence. This process involved handling missing values, addressing outliers, and transforming variables to align with analytical requirements.

- Perform exploratory data analysis (EDA) using visualization and SQL

Exploratory Data Analysis was conducted using a combination of visualization tools and SQL queries. This dual approach provided both graphical representations and structured queries to uncover patterns, trends, and relationships within the dataset.

- Perform interactive visual analytics using Folium and Plotly Dash

Interactive visual analytics were implemented through tools such as Folium and Plotly Dash. Folium facilitated dynamic map visualizations, while Plotly Dash enabled interactive dashboards, enhancing the user's ability to explore and understand complex spatial and temporal patterns.

Methodology

Executive Summary

- Perform predictive analysis using classification models
    - The process of building classification models involved splitting the dataset into training and testing sets, training the models on the training set, and fine-tuning hyperparameters through techniques like cross-validation. Evaluation metrics such as accuracy, precision, recall, and F1 score were employed to assess the models' performance, ensuring their effectiveness in making accurate predictions.

Data Collection

1. SpaceX API Utilization:

- The primary method involved making GET requests to the SpaceX API.
- The response content, initially in JSON format, was decoded using the .json() function.
- The structured JSON data was then normalized into a Pandas dataframe using .json_normalize().

2. Data Cleaning:

- A thorough data cleaning process was initiated to ensure data integrity.
- Missing values were identified, and appropriate fill-in strategies were implemented where necessary.

3. Web Scraping from Wikipedia:

- Web scraping was employed to gather Falcon 9 launch records from Wikipedia.
- BeautifulSoup was utilized to target and extract launch records presented in HTML tables.
- These tables were parsed and converted into Pandas dataframes, laying the groundwork for future analyses.

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook (https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-spacex-data-collection-api.ipynb), as an external reference and peer-review purpose

Now let's start requesting rocket launch data from SpaceX API with the following URL:

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [8]:
```
response = requests.get(spacex_url)
```

Check the content of the response

In [9]:
```
print(response.content)
```

In [9]:
```
print(response.content)
```


b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":"2006-03-17T00:00:00.000Z","static_fire_date_unix":1142553600,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":33,"altitude":null,"reason":"merlin engine failure"}],"details":"Engine failure at 33 seconds and loss of vehicle","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b5b6c3bb0006eeb1e1"],"launchpad":"5e9e4502f5090995de566f86","flight_number":1,"name":"FalconSat","date_utc":"2006-03-24T22:30:00.000Z","date_unix":1143239400,"date_local":"2006-03-25T10:30:00+12:00","date_precision":"hour","upcoming":false,"cores":[{"core":"5e9e289df35918033d3b2623","flight":1,"gridfins":false,"legs":false,"reused":false,"landing_attempt":false,"landing_success":null,"landing_type":null,"landpad":null}],"auto_update":true,"tbd":false,"launch_library_id":null,"id":"5eb87cd9ffd86e000604b32a"},{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/f9/4a/ZboXReNb_o.png","large":"https://images2.imgbox.com/80/a2/bkWotCIS_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=Lk4zQ2wP-Nc","youtube_id":"Lk4zQ2wP-Nc","article":"https://www.space.com/3590-spacex-falcon-1-rocket-fails-reach-orbit.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_u

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook(https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-webscraping-edwin.ipynb), as an external reference and peer-review purpose

In [27]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

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

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

In [28]:
```
data  = requests.get(static_url).text
```

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

In [30]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data,'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [33]:
```
# Use soup.title attribute

print(soup.title)
```

```
El titulo es: None
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

In [47]:
```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
df
```

Out[47]:

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA (COTS)\nNRO	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43

Data Wrangling

The data underwent a systematic processing phase to derive meaningful insights and prepare it for further analyses. The following steps outline the data processing procedures:

1. **Exploratory Data Analysis (EDA):**

- Conducted EDA to explore and understand the dataset.
- Determined the labels for training supervised models based on landing outcomes.

2. **Calculation of Launch Metrics:**

- Calculated the number of launches at each launch site to gain insights into site-specific activities.
- Quantified the occurrences of each orbit type to understand the distribution of missions.

3. **Creation of Training Labels:**

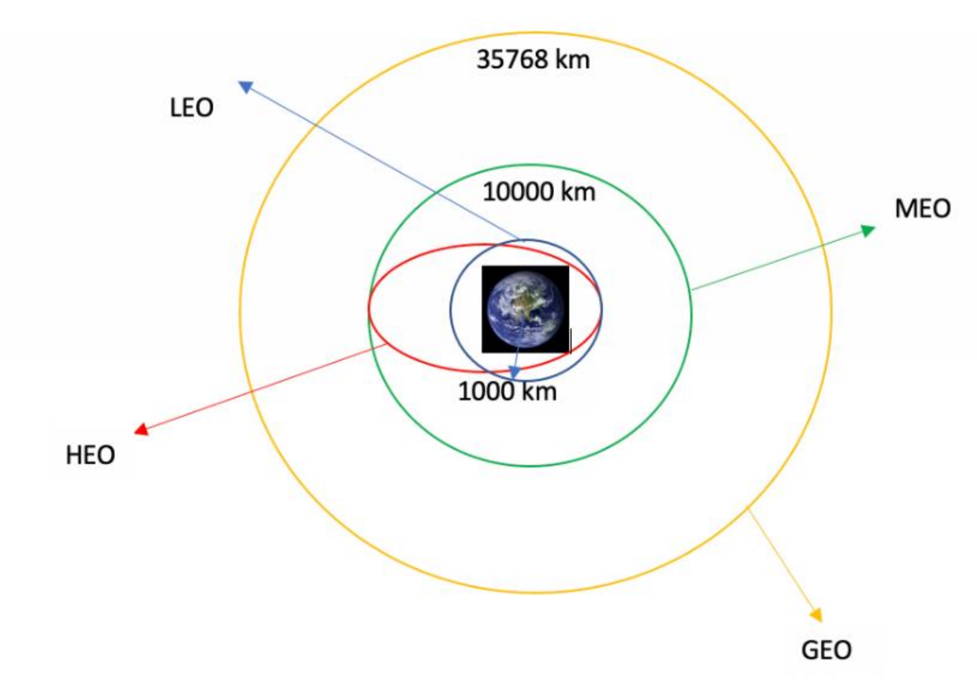
- Developed training labels, denoted as 'Class,' to categorize first-stage booster landing outcomes.
- The 'Class' variable was assigned as follows:

Class = 0: First stage booster did not land successfully, considering various landing scenarios.

Class = 1: First stage booster landed successfully, encompassing different successful landing scenarios.

4. **Exporting Results to CSV:**

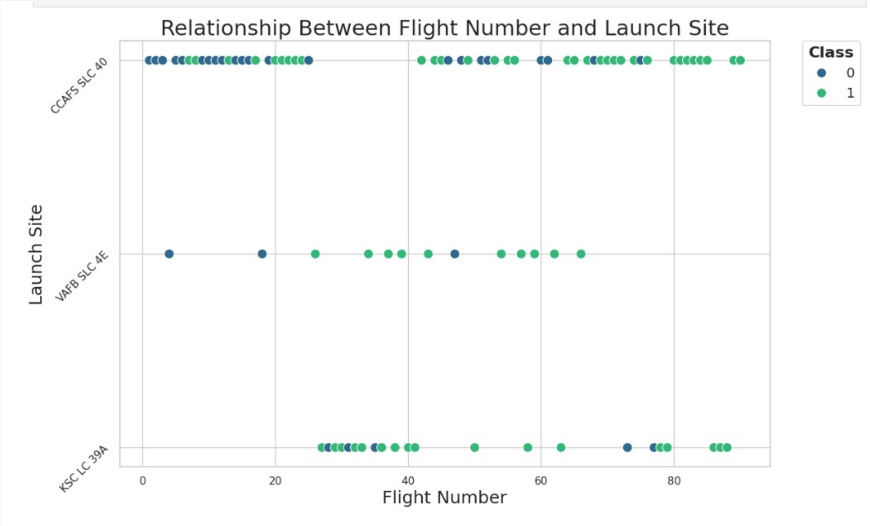
- Created a landing outcome label derived from the 'Outcome' column.
- Exported the processed results, including calculated launch metrics and training labels, to a CSV file for documentation and potential use in subsequent analyses.



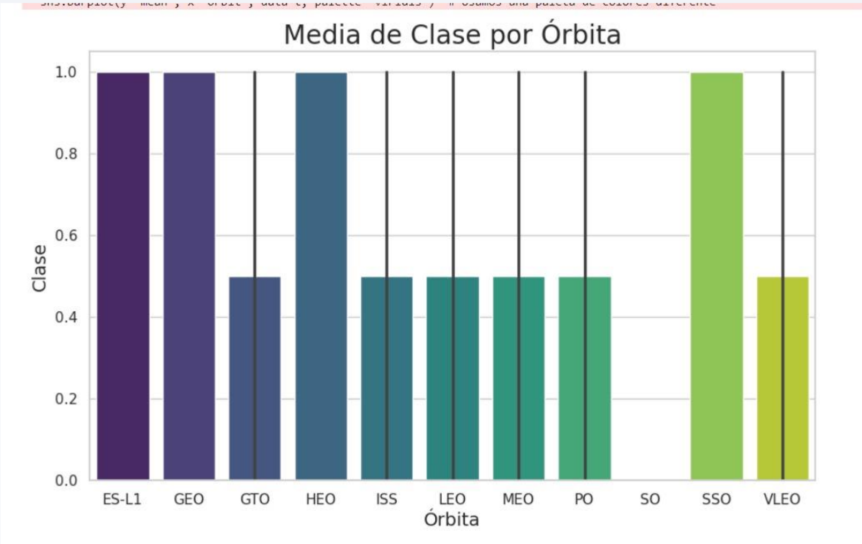
https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/labs-jupyter-spacex-Data%20wrangling%20(1).ipynb

EDA with Data Visualization

- During the Exploratory Data Analysis (EDA) phase, various charts were plotted to gain insights into different aspects of the dataset. The choice of each chart was driven by the need to explore specific relationships and patterns within the data. Throughout the Exploratory Data Analysis (EDA) phase, diverse visualizations were employed, including bar charts, line charts, and scatter plots.



https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb



EDA with SQL

- Executed SQL queries to retrieve information about launch sites, providing insights into the distribution and characteristics of different launch locations.
- Utilized SQL to query payload masses, allowing for an exploration of the range and distribution of payload masses across various launches.
- Employed SQL queries to gather details about booster versions, offering a comprehensive understanding of the different versions used in the launches.
- Ran SQL queries to investigate mission outcomes, providing an overview of the success or failure of each mission.
- Utilized SQL to query booster landings, extracting information about the success or failure of the first-stage booster landings in each launch.Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

Task 1

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

[19]:
```
%sql SELECT * FROM sqlite_master WHERE type='table' AND name='SPACEXTABLE';
```

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

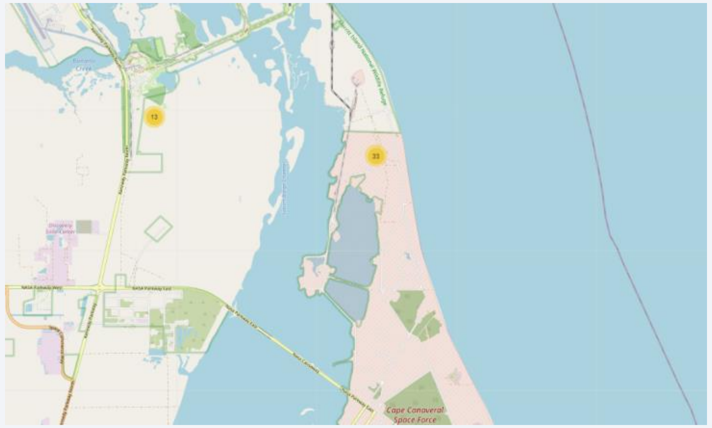
[19]:

type	name	tbl_name	rootpage	sql
table	SPACEXTABLE	SPACEXTABLE	6	CREATE TABLE SPACEXTABLE(Date TEXT, "Time (UTC)" TEXT, Booster_Version TEXT, Launch_Site TEXT, Payload TEXT, PAYLOAD_MASS__KG_ INT, Orbit TEXT, Customer TEXT, Mission_Outcome TEXT, Landing_Outcome TEXT)

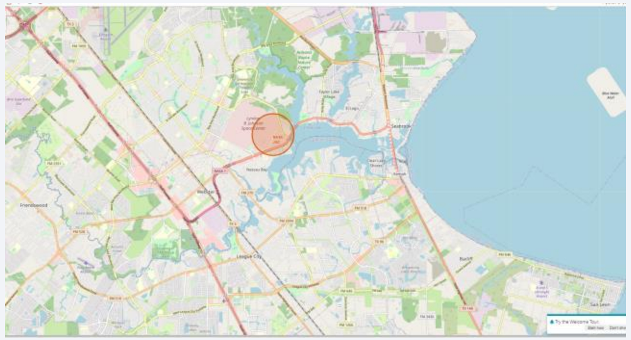
https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

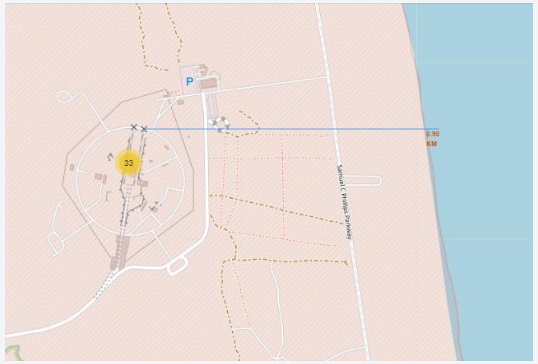
Build an Interactive Map with Folium

- In enhancing the folium map, several notable modifications were implemented. The introduction of color-coded markers and circles served to delineate areas and visually represent the success or failure of launches at each site. The utilization of varying colors in marker clusters not only added aesthetic appeal but also facilitated the identification of launch sites with notable success rates. The incorporation of radius adjustments to specific locations further enriched the map's visual representation, offering a nuanced perspective on the spatial distribution of launch outcomes.
- Furthermore, an exploration into the geographical aspects was conducted by calculating distances between launch sites and their surroundings. This analysis delved into pertinent questions, such as the proximity of launch sites to railways, highways, coastlines, and urban areas. The comprehensive utilization of map objects, along with the thoughtful integration of color and radius adjustments, enabled a multifaceted exploration of spatial relationships and launch outcomes within the folium map framework.



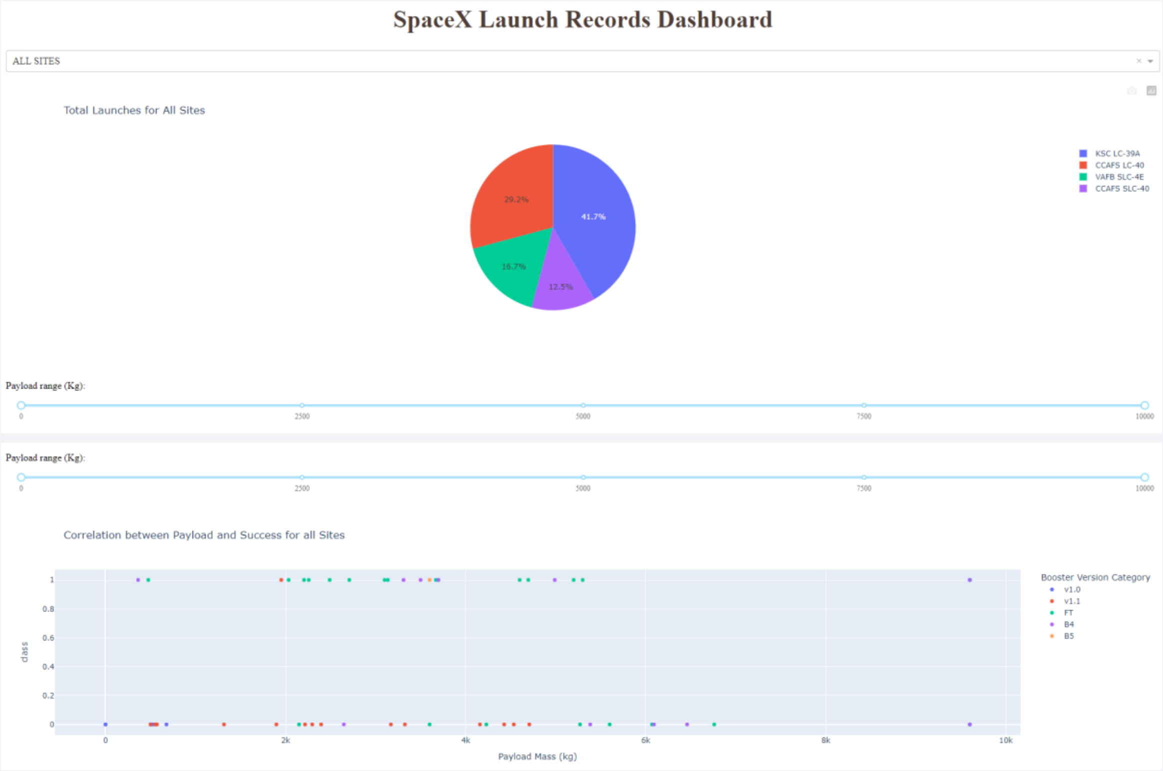
https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb





Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboardIn the creation of the Launch Records Dashboard using Plotly Dash, several interactive and visually informative elements were incorporated.
- A pie chart was integrated to visually represent the success rate, color-coded by launch site, providing stakeholders with an at-a-glance overview of launch outcomes.
- Additionally, a scatter chart displayed the relationship between payload mass and landing outcome, with color coding based on booster version.
- The inclusion of a range slider allowed users to dynamically limit payload amounts for a more focused analysis.
- Furthermore, a drop-down menu provided the flexibility to switch between viewing data for all launch sites or individual launch sites, enhancing the dashboard's user-friendly interface.



https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/spacex_dash_app%20(1).py

Predictive Analysis (Classification)

The model development process can be visualized as follows:

1. **Data Loading and Transformation:**
Utilized NumPy and Pandas to load and transform the dataset.
2. **Data Splitting:**
Split the data into training and testing datasets.
3. **Model Fitting:**
Employed various classification models (Logistic Regression, SVM, Decision Tree, KNN) to fit the training data.
4. **Hyperparameter Tuning:**
Conducted a cross-validated grid search to optimize hyperparameters for each model.
5. **Model Evaluation:**
Evaluated model accuracy using the test data.
6. **Model Improvement:**
Iteratively improved model performance through feature engineering and algorithm tuning.
7. **Best Model Identification:**
Identified the best-performing classification model based on accuracy.

Load the dataframe

Load the data

```
from js import fetch
import io

URL1 = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv"
resp1 = await fetch(URL1)
text1 = io.BytesIO((await resp1.arrayBuffer()).to_py())
data = pd.read_csv(text1)
```

`X_train, X_test, Y_train, Y_test`

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,test_size=0.2,random_state=2)
```

Create a logistic regression object then create a GridSearchCV object `logreg_cv` with cv = 10. Fit the object to find the best parameters from the dictionary `parameters`.

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
logreg_cv = GridSearchCV(lr, parameters, cv=10)
logreg_cv.fit(X_train, Y_train)
```

```
GridSearchCV
estimator: LogisticRegression
LogisticRegression
```

```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

```
# Identificar el mejor modelo según una métrica específica (por ejemplo, la precisión)
best_model = results_df.loc[results_df['Precision'].idxmax(), 'Model']
print(f"The best model based on Precision is: {best_model}")
```

	Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
0	KNN	0.833333	0.800000	1.000000	0.888889	0.750
1	SVM	0.833333	0.800000	1.000000	0.888889	0.750
2	Logistic Regression	0.833333	0.800000	1.000000	0.888889	0.750
3	Decision Tree	0.888889	0.916667	0.916667	0.916667	0.875

The best model based on Precision is: Decision Tree

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

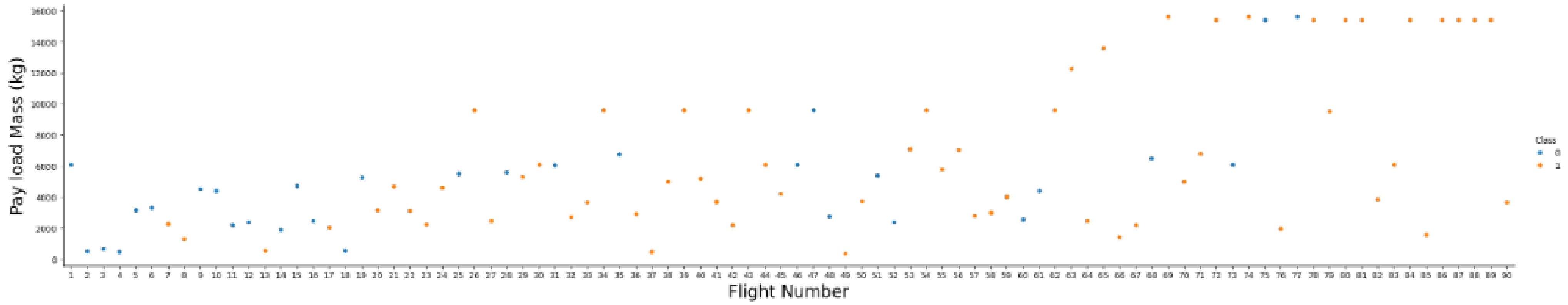
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

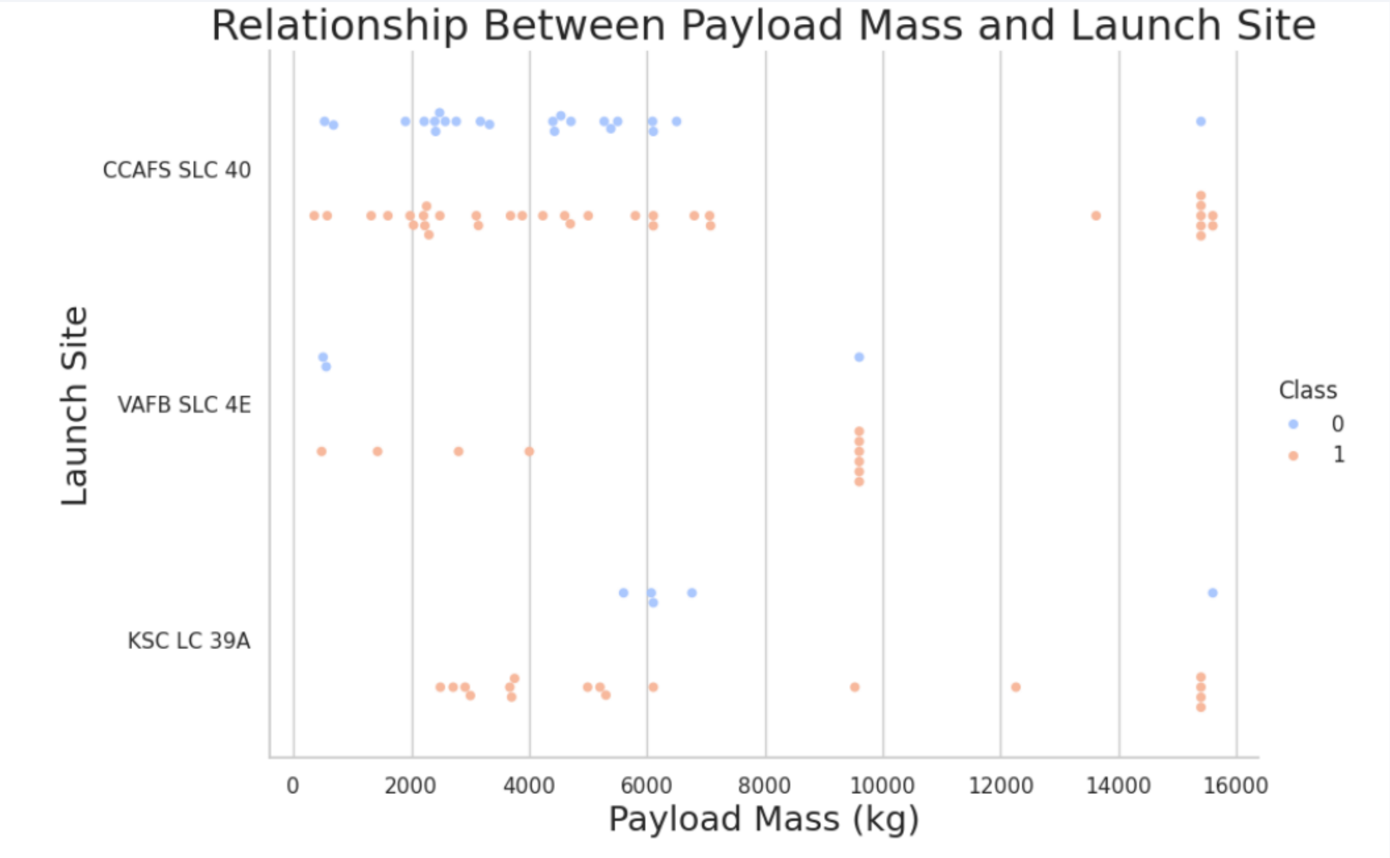
The analysis of FlightNumber in relation to PayloadMass and 1st stage landing success reveals significant correlations that offer valuable insights for SpaceX's launch operations. The positive correlation between 1st stage landing success and continuous launch attempts implies that frequent launches contribute to a higher success rate.



We see that different launch sites have different success rates. `CCAFS LC-40`, has a success rate of 60 %, while `KSC LC-39A` and `VAFB SLC 4E` has a success rate of 77%.

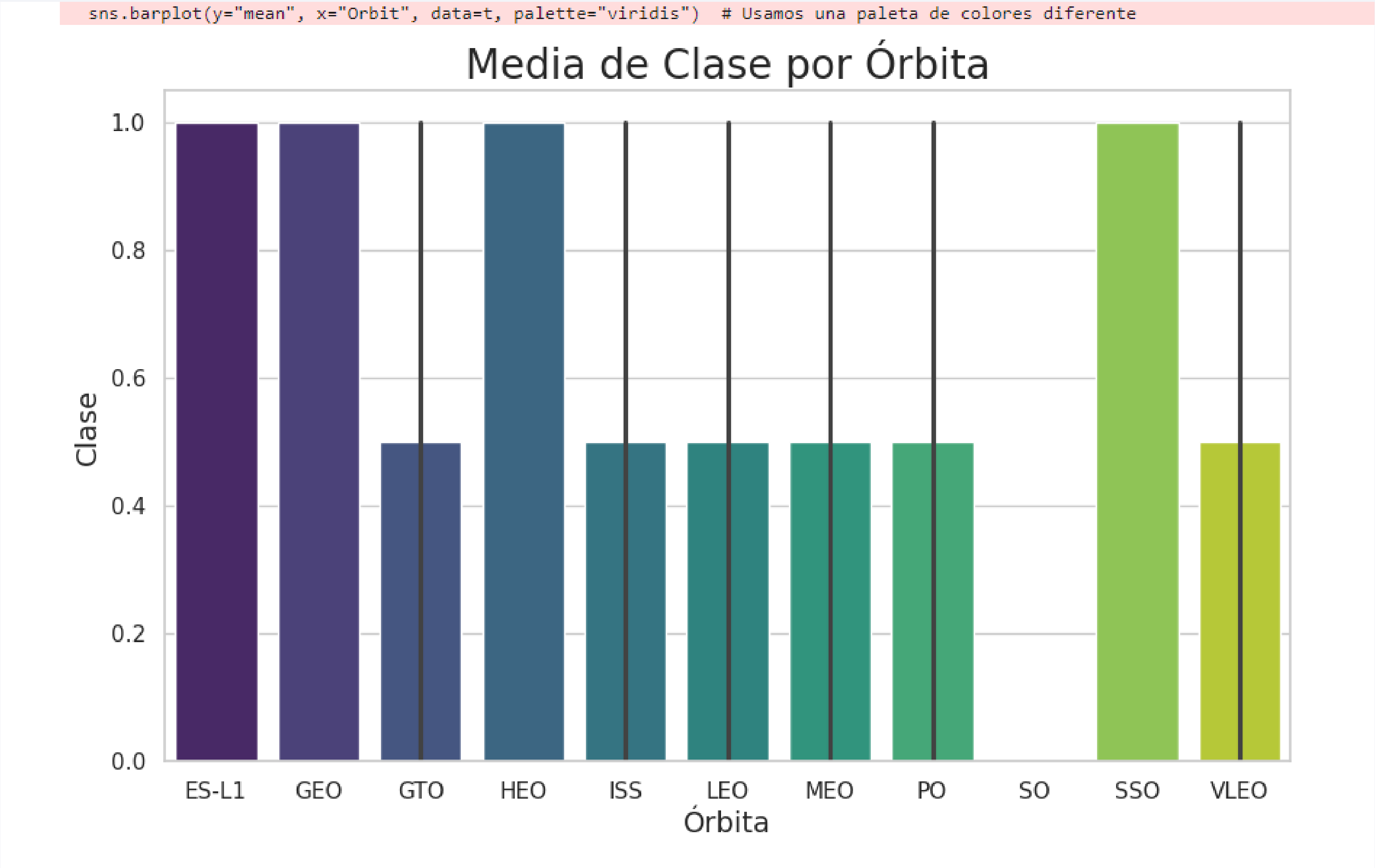
Payload vs. Launch Site

- Notably, CCAFS SLC 40 was identified as the launch site where most of the early first-stage landing failures occurred. This suggests that there may have been challenges or issues specific to this launch site during the initial stages.



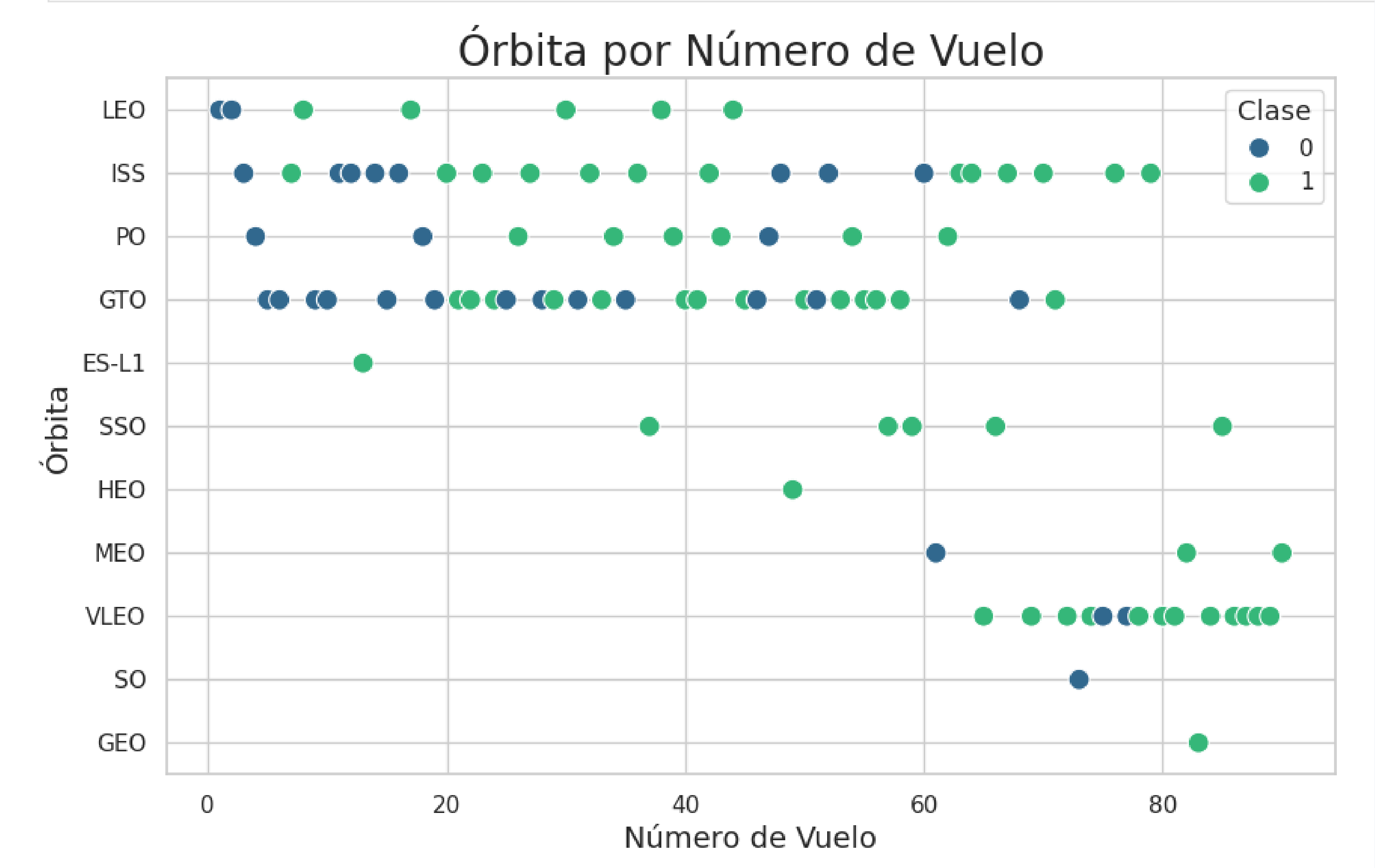
Success Rate vs. Orbit Type

The analysis of Orbit type in relation to success rate indicates a notable trend where all orbit types, except 'SO,' have recorded successful 1st stage landings. Notably, orbits such as ES-L1, GEO, HEO, SSO, and VLEO stand out with the highest success rates. This suggests that SpaceX has demonstrated consistent proficiency in achieving successful landings across a variety of orbital trajectories. The success in diverse orbit types showcases the versatility and reliability of SpaceX's first-stage landing capabilities, contributing to the overall success and effectiveness of their space launch operations.



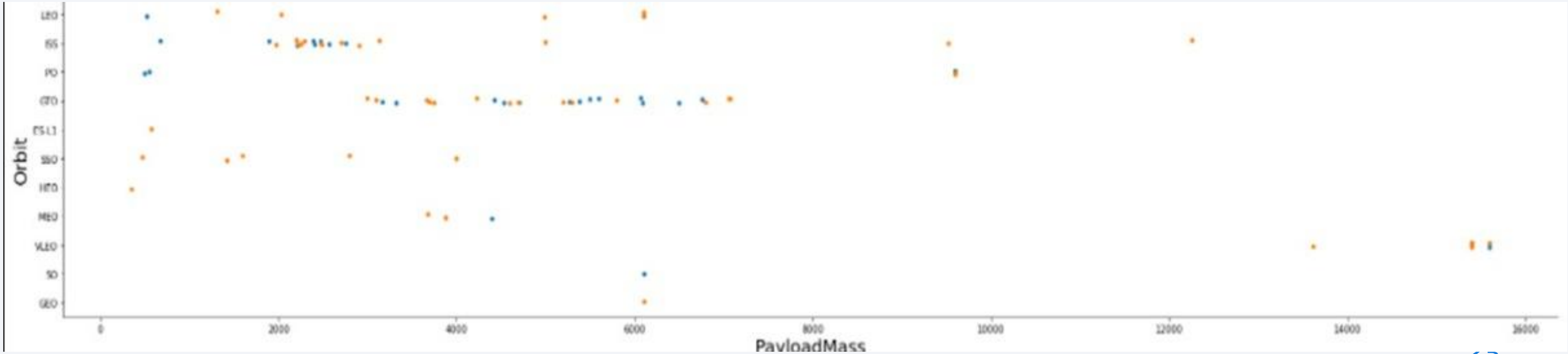
Flight Number vs. Orbit Type

The analysis of Flight Number in relation to Orbit type reveals interesting patterns. Specifically, in the Low Earth Orbit (LEO), there is a positive correlation between the number of flights and the success of 1st stage recovery. This suggests that as the flight frequency increases in the LEO orbit, the likelihood of successful 1st stage recovery also tends to rise. On the other hand, in the Geostationary Transfer Orbit (GTO), there appears to be no discernible relationship between flight number and the success of the orbit. This insight emphasizes the nuanced dynamics of SpaceX's operations, where the impact of flight frequency on recovery success varies across different orbital trajectories.



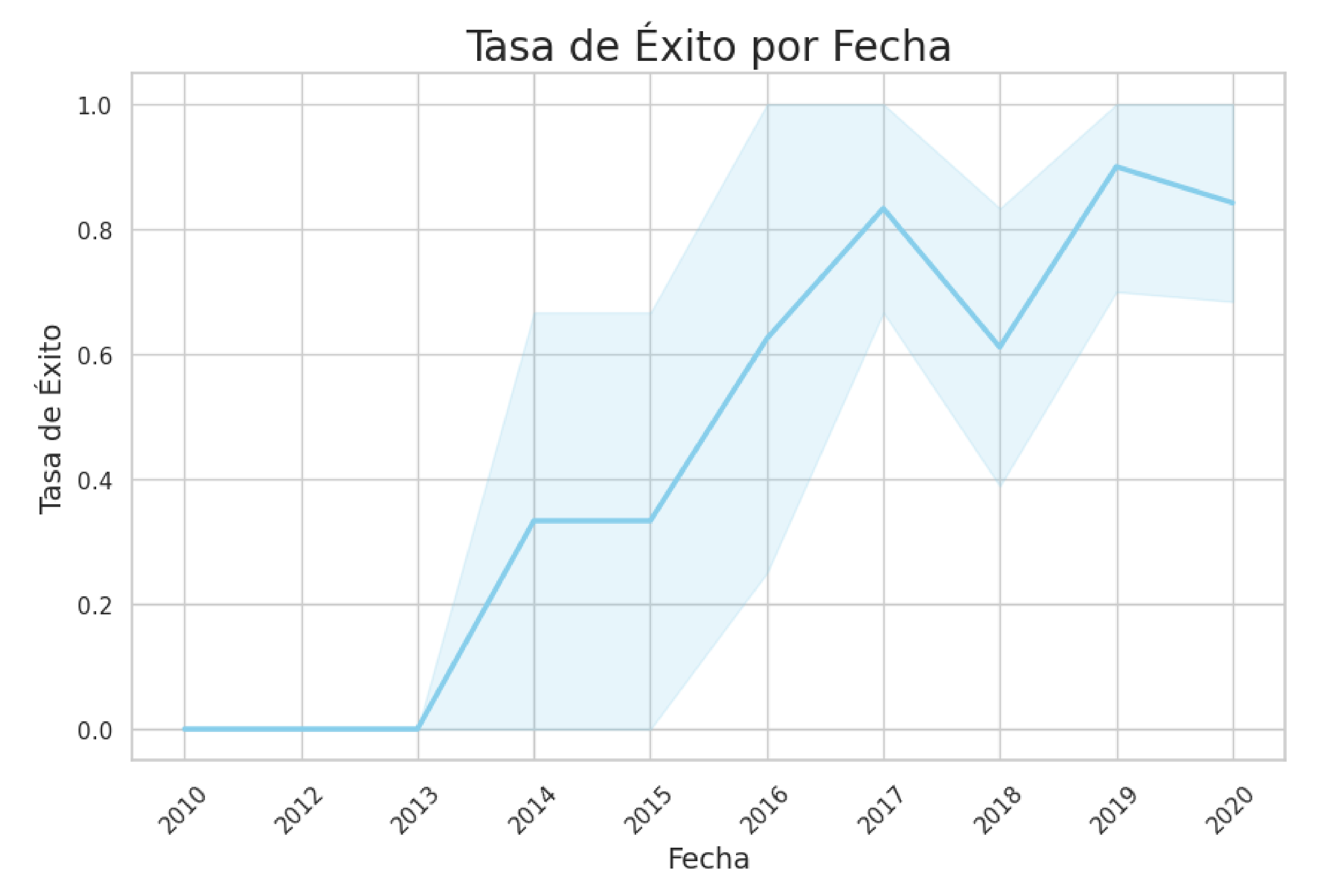
Payload vs. Orbit Type

The analysis of PayloadMass in relation to Orbit type uncovers notable trends. Heavier payloads exhibit a negative influence on Geostationary Transfer Orbits (GTO) but a positive influence on Polar Orbits (PO), Low Earth Orbits (LEO), and International Space Station (ISS) orbits. This observation suggests that, when dealing with heavy payloads, the success of landings is more prevalent in orbits with polar, low Earth, and ISS trajectories. This insight provides valuable considerations for payload management and successful landings across different orbital types within SpaceX's operations.



Launch Success Yearly Trend

The examination of the success rate over the years reveals a consistent upward trend since 2013, reaching its peak in 2020. This positive trajectory suggests an overall improvement in the success of SpaceX launches annually. The upward trend may indicate advancements in technology, operational efficiency, and experience, contributing to a more reliable and successful spaceflight program over the years.



All Launch Site Names

In [20]:

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL;
```

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

Out[20]:

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

We utilized the **DISTINCT** keyword to display exclusively the distinct launch sites present in the SpaceX dataset.

Launch Site Names Begin with 'CCA'

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

[23]:
```
%%sql
SELECT LAUNCH_SITE
FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE 'CCA%'
LIMIT 5;
```

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

[23]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

In the provided SQL code, the LIKE clause is used with the % symbol as a wildcard. The % is used to represent any set of characters. In this case, the condition WHERE LAUNCH_SITE LIKE 'CCA%' looks for rows where the value in the LAUNCH_SITE column begins with 'CCA'. The % indicates that there can be any number of characters after 'CCA'. The query is limited to show only the first 5 results with the LIMIT 5 clause. In summary, the query searches and displays launch sites that start with 'CCA' followed by any set of characters.

Total Payload Mass

Task 3

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

```
]: %%sql SELECT SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Customer = 'NASA (CRS)';
```

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

]:

SUM(PAYLOAD_MASS__KG_)
45596

Whe utilized the SUM(PAYLOAD_MASS_KG) function is used to calculate the total sum of the payload mass across all rows that meet the specified conditions in the query.

Average Payload Mass by F9 v1.1

Task 4

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

[25]:
```
%%sql SELECT AVG(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Booster_Version LIKE 'F9 v1.0%';
```

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

[25]:

AVG(PAYLOAD_MASS__KG_)
340.4

We calculated the average using the AVG() function, IN ADDITION TO USING THE % FOR INFORMATION FILTERING THANKS TO THE WHERE clause

First Successful Ground Landing Date

[26]:
```
%%sql
SELECT MIN(Date)
FROM SPACEXTBL
WHERE Landing_Outcome = 'Success (ground pad)';
```

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

[26]:

MIN(Date)
2015-12-22

We filtered the minimum date using the MIN() function, and also utilized the WHERE clause.

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING_OUTCOME = 'Success (drone ship)'
    AND 4000 < PAYLOAD_MASS__KG_ < 6000;
```

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

Booster_Version
F9 FT B1021.1
F9 FT B1022
F9 FT B1023.1
F9 FT B1026
F9 FT B1029.1
F9 FT B1021.2
F9 FT B1029.2
F9 FT B1036.1
F9 FT B1038.1
F9 B4 B1041.1
F9 FT B1031.2
F9 B4 B1042.1
F9 B4 B1045.1
F9 B5 B1046.1

We applied conditional filtering using the WHERE clause, specifically employing the AND operator and the greater than (>) and less than (<) signs

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

9]:
```
%%sql
SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_NUMBER
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME;
```

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

9]:

Mission_Outcome	TOTAL_NUMBER
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

In this SQL query, we are counting the total number of occurrences for each unique value in the "MISSION OUTCOME" column from the SPACEXTBL table. The result is a summary of the different mission outcomes along with their respective counts.

Boosters Carried Maximum Payload

Task 8

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

30]:
```
%%sql
SELECT DISTINCT BOOSTER_VERSION
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL);
```

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

30]:

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

In this SQL query, we are selecting unique booster versions from the SPACEXTBL table. The selection is based on a condition specified in the WHERE clause, where we are filtering rows where the payload mass is equal to the maximum payload mass in the entire table. The DISTINCT keyword ensures that only unique booster versions meeting this condition are included in the result set.

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.

[35]:
```
%%sql
SELECT LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE Landing_Outcome = 'Failure (drone ship)'
    AND strftime('%Y', DATE) = '2015';
```

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

[35]:

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

In this SQL query, we are selecting data from the SPACEXTBL table. We are interested in the columns "LANDING OUTCOME," "BOOSTER_VERSION," and "LAUNCH_SITE." The query includes a condition specified in the WHERE clause, where we filter rows based on two criteria: the landing outcome should be 'Failure (drone ship)' and the year extracted from the 'DATE' column should be '2015'. The strftime('%Y', DATE) function is used to extract the year from the 'DATE' column.

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.

[37]:

```
%%sql
SELECT LANDING_OUTCOME, COUNT(LANDING_OUTCOME) AS TOTAL_NUMBER
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING_OUTCOME
ORDER BY TOTAL_NUMBER DESC;
```

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

[37]:

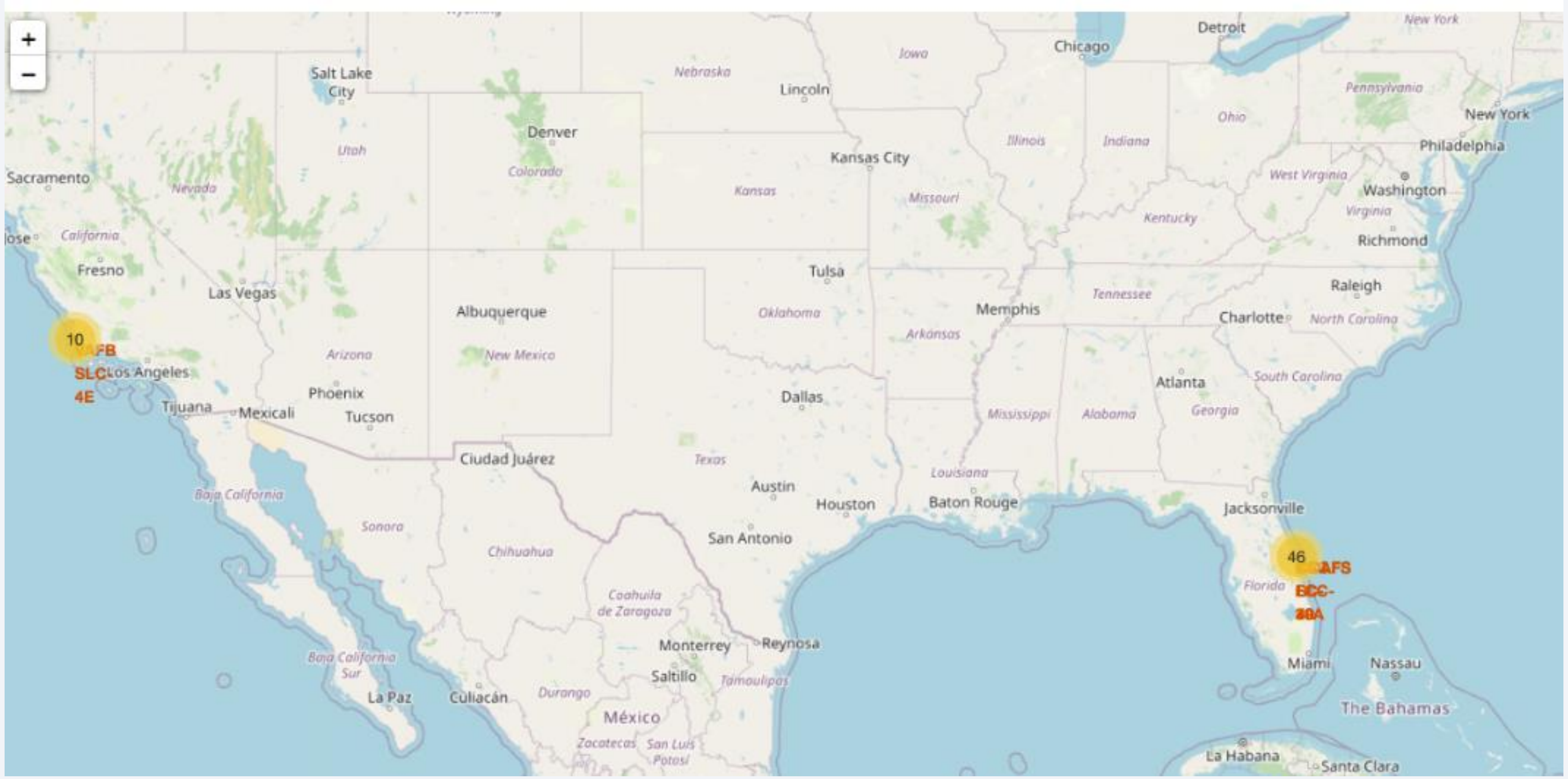
Landing_Outcome	TOTAL_NUMBER
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

In this SQL query, we are selecting data from the SPACEXTBL table. We are interested in the columns "LANDING OUTCOME" and the count of each outcome, named as "TOTAL NUMBER." The query includes a condition specified in the WHERE clause, where we filter rows based on the 'DATE' column. We only select records where the date falls between June 4, 2010, and March 20, 2017. The results are then grouped by "LANDING OUTCOME," and the total count for each outcome is calculated. Finally, the results are ordered in descending order based on the total count.

Section 3

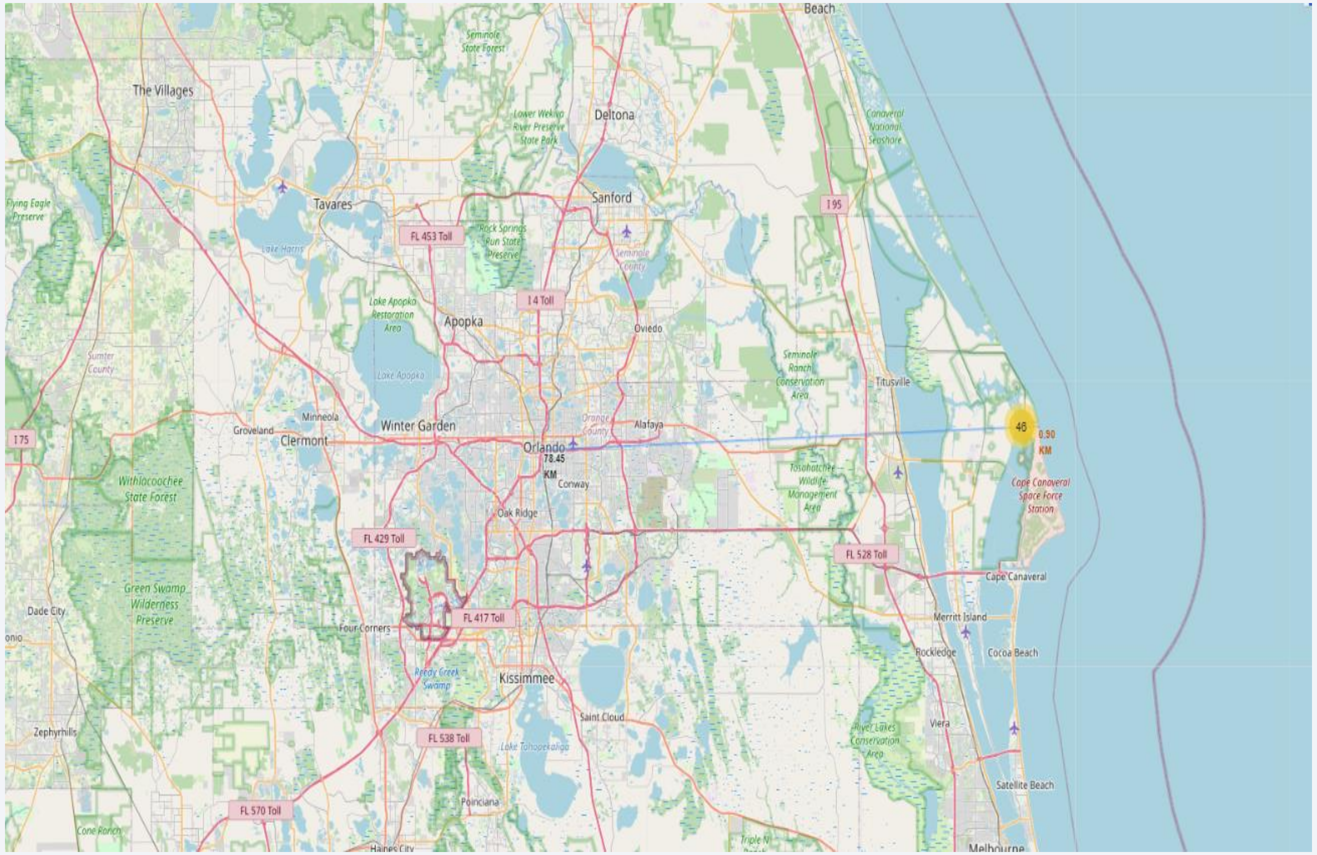
Launch Sites Proximities Analysis

All launch sites in the world map maker



Mapping the launch sites provides a visual representation that underscores the significance of their proximity to both the coastline and the equator.

Distance of the launch site from notable landmark

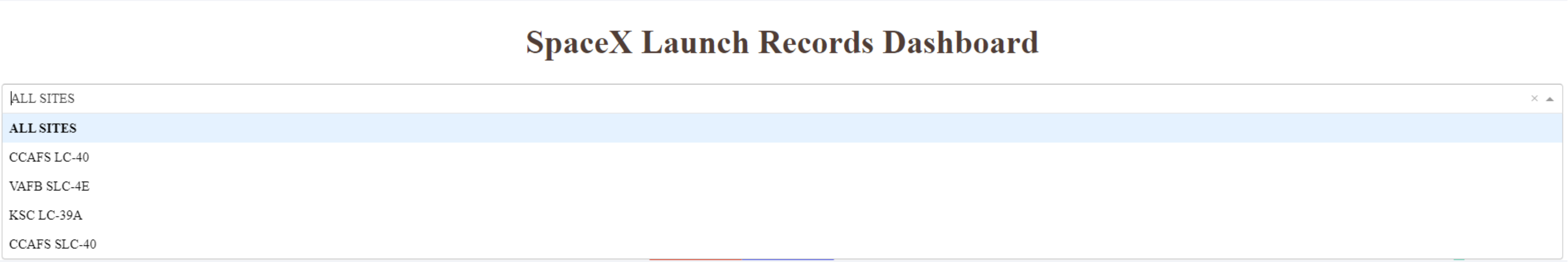


The analysis of the launch site's distance from notable landmarks emphasizes the significance of the geographical location for space launches. Proximity to specific landmarks, such as coastlines, equators, or other key geographical features, plays a crucial role in determining optimal launch sites. This information is valuable for strategic decision-making in the selection of launch sites, considering factors that can impact the success and efficiency of space missions.

Section 4

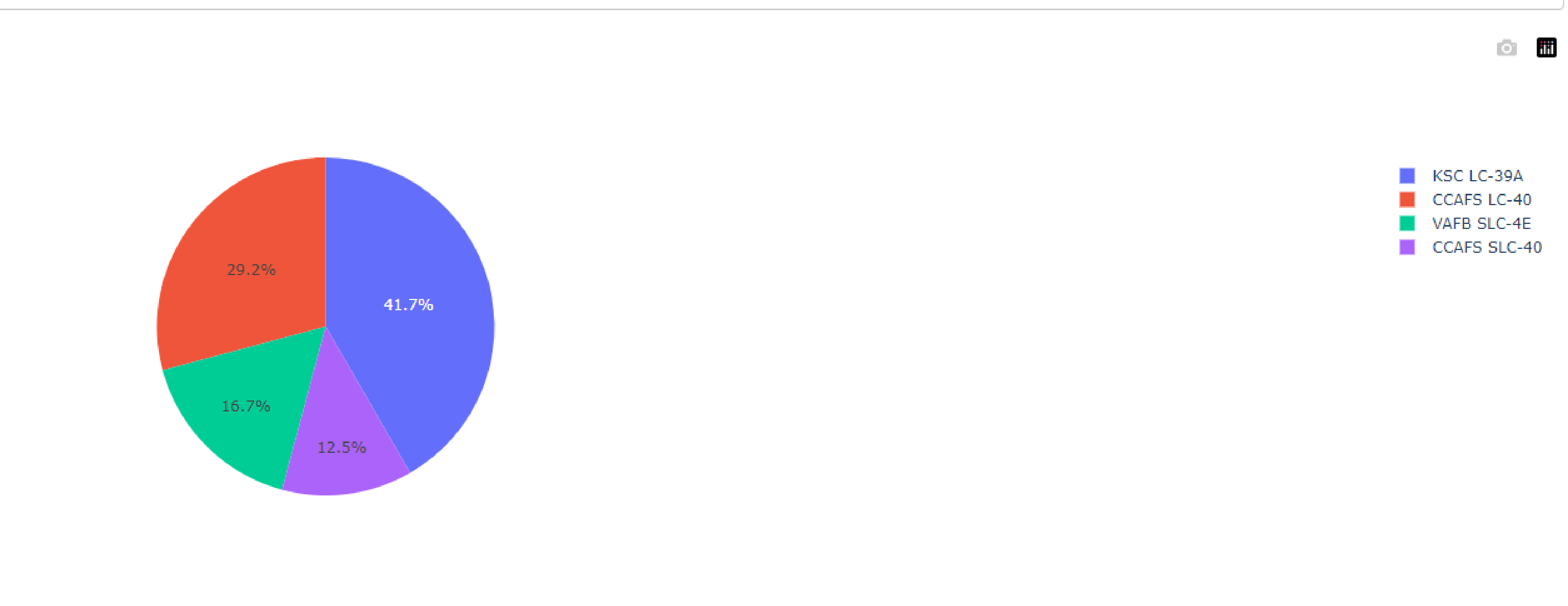
Build a Dashboard with Plotly Dash

Tittle and dropdown option



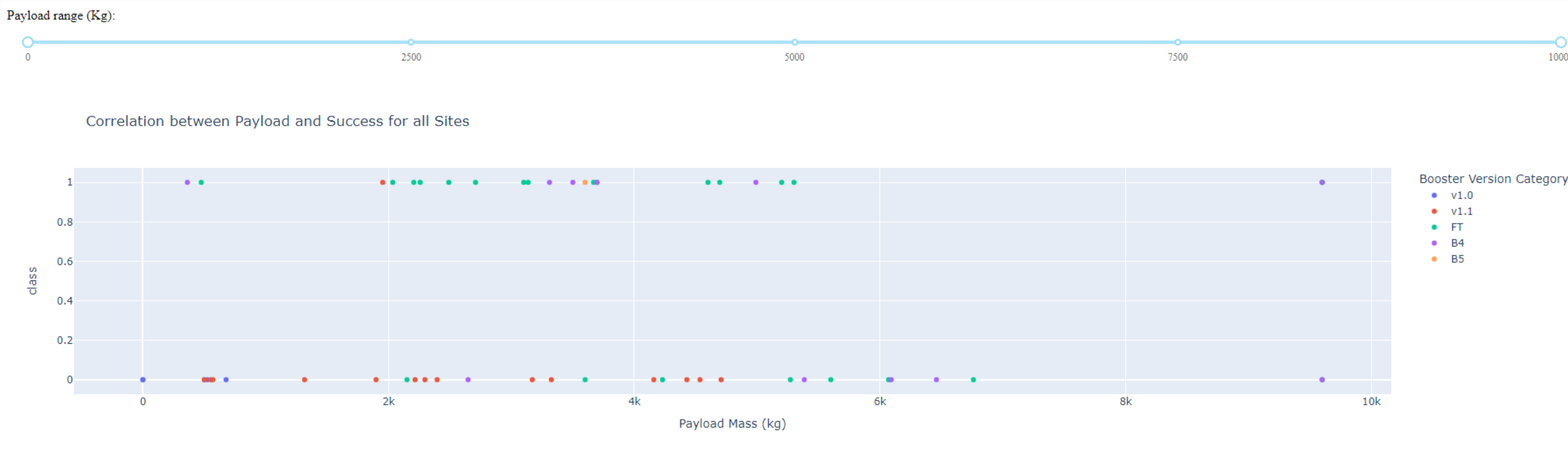
In the following image, the title and the dropdown menu options are displayed for performing the search.

PIE CHART FOR TOTAL LAUNCHE ALL SITES



In the following image, a pie chart is displayed, showing the percentages based on the number of launches per site.

PAYLOAD RANGE AND SCATTERPLOT



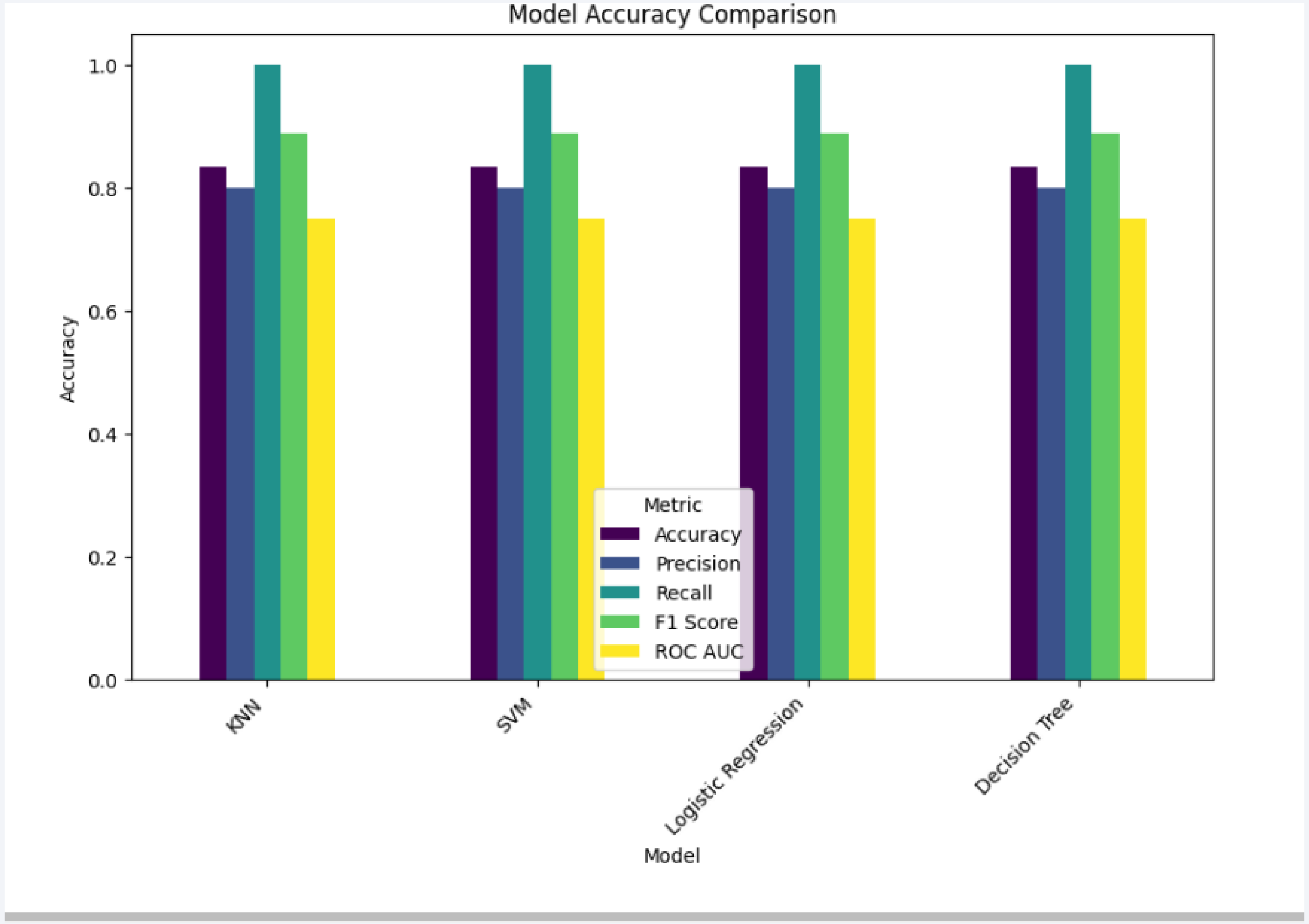
In the following image, a range adjustment is shown on the Payload range in kilograms. This will modify the scatter plot (class vs Payload mass) to include values within that range.

Section 5

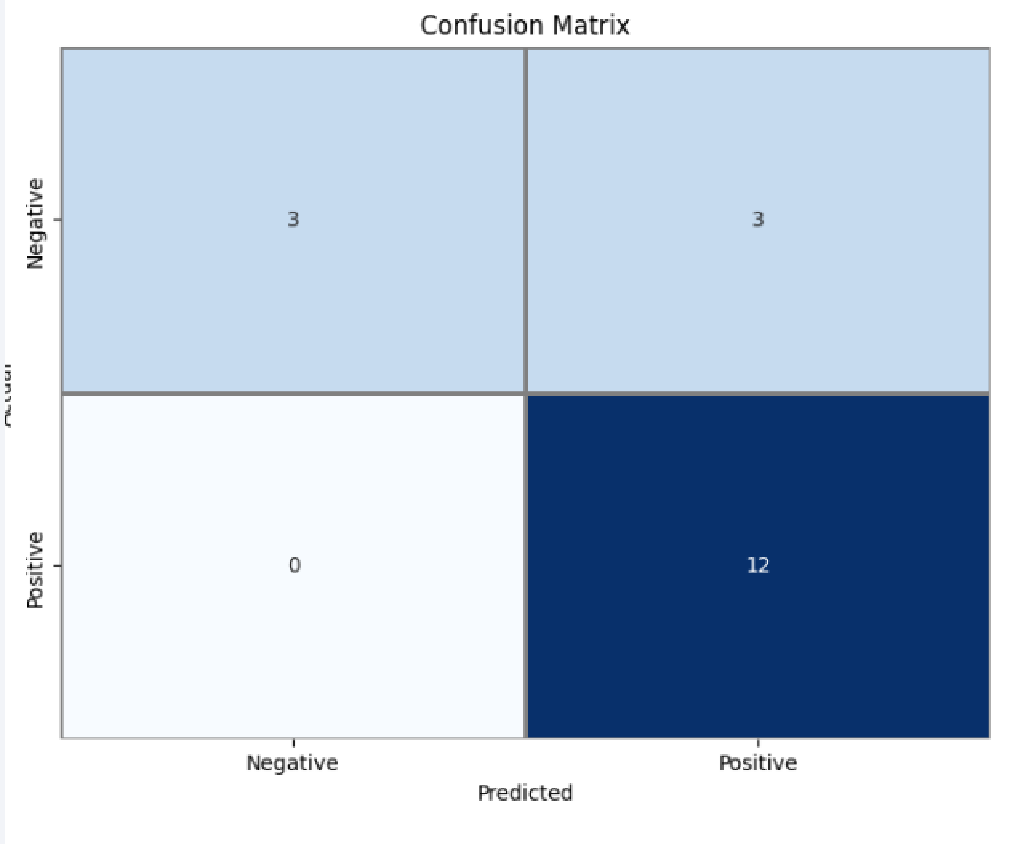
Predictive Analysis (Classification)

Classification Accuracy

There is no model with better performance because they have the same precision



Confusion Matrix



The confusion matrices for the top-performing models (which are tied) exhibit identical patterns. The primary issue lies in false positives, where the models incorrectly predict the 1st stage booster's successful landing in 3 out of 18 samples within the test set.

Conclusions

- The success rate of launches at a site appears to be positively correlated with the total number of flights conducted at that site, suggesting that more experienced launch sites tend to achieve higher success rates.

- The analysis indicates a positive trend in launch success rates from 2013 to 2020, reflecting an overall improvement in the efficiency and reliability of space missions during this period.

- Certain orbit types, such as ES-L1, GEO, HEO, SSO, and VLEO, consistently demonstrate higher success rates, providing valuable insights into the preferred orbits for successful landings.

- The Decision Tree classifier emerges as the most effective machine learning algorithm for predicting the success of first-stage booster landings, offering SpaceY a reliable tool with an 83.3% accuracy rate.

Appendix

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/jupyter-labs-webscraping-edwin.ipynb

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/labs-jupyter-spacex-Data%20wrangling%20(1).ipynb

https://github.com/EdwinSotto12311/Data_Science_Capstone_proyect/blob/main/spacex_dash_app%20(1).py

Thank you!