IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Burninova Svetlana
May 2024

Outline


Executive Summary


Introduction


Methodology


Results


Conclusion


Appendix

Executive Summary

The methodologies used in the project: data collection, preprocessing, analysis, machine learning, and interactive visualization techniques. These methodologies encompassed a comprehensive approach to data analysis and decision-making in the context of SpaceX rocket launches, leveraging both traditional statistical techniques and modern machine learning algorithms, along with interactive visualization tools for enhanced exploration and understanding of the data.

The results of the project provide valuable insights into the factors influencing the success of Falcon 9 first stage landings, offering practical applications for decision-making in the aerospace industry. The combination of data-driven analysis, machine learning modeling, interactive visualization, and database management techniques demonstrates a holistic approach to solving complex challenges in rocket launch prediction and optimization.

Introduction

The project aims to predict the success of Falcon 9 first stage landings in SpaceX rocket launches. By analyzing historical data and leveraging machine learning techniques, the goal is to develop models that can forecast landing outcomes. This will enable stakeholders to make informed decisions about launch planning, cost estimation, and risk management. Key problems include launch success prediction, cost estimation, risk assessment, and decision support.

Key Problems to Address:

- Launch Success Prediction: Develop models to forecast Falcon 9 first stage landing outcomes.
- Cost Estimation: Estimate launch costs based on predicted landing success.
- Risk Assessment: Identify factors influencing launch success and assess associated risks.
- Decision Support: Provide tools and insights to support decision-making for SpaceX and stakeholders in the space industry.

Methodology

Section 1

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from API for SpaceX rocket launches, including historical records and details about the launch outcomes.
- Perform data wrangling
  - Row data was preprocessed for analysis by cleaning and transforming it.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Built, tuned and evaluated the models by preparing the data, selecting appropriate algorithms, and assessing their performance with metrics like accuracy and precision

Data Collection

To predict the successful landing of the Falcon 9 first stage, datasets were collected and parsed. 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, with much of the savings attributed to SpaceX's ability to reuse the first stage. The data can help determine the first stage's landing and launch cost. This information can be valuable if an alternative company wishes to bring to bid against SpaceX for a rocket launch. The data was collected and preprocessed into the correct format from an API.

Data Collection – SpaceX API

- The SpaceX launch data was utilized by a GET request to retrieve and parse it

To make the requested JSON results more consistent, we will use the following static response object for this project:

In [73]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/da
```

We should see that the request was successfull with the 200 status response code

In [74]:
```
response.status_code
```

Out[74]:
```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

In [75]:
```
# Use json_normalize meethod to convert the json result into a dataframe

data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

In [76]:
```
# Get the head of the dataframe
# Print the first 5 rows of the DataFrame
data.head()
```

Out[76]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	c
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	
1	None	NaN	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 301, 'altitude': 289, 'reason': 'harmonic oscillation leading to premature engine	Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown at T+7 min 30 s,	

Data Collection – Scraping

- Data was filtered by using BoosterVersion
- GitHub: https://github.com/Svetlanaburn/Applied-Data-Science-Capstone-IBM/blob/cf5dfd46fb6ff682b433856623538a78d43bb660/Data%20Science%20Capstone%20IBM/Space%20X%20-%20Data%20Collections%20API.ipynb

In [89]:
```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages/pandas/core/indexing.py:1773: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy
  self._setitem_single_column(ilocs[0], value, pi)
```

Out[89]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	
89	86	2020-09-03	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3
90	87	2020-10-06	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3
91	88	2020-10-18	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3
92	89	2020-10-24	Falcon 9	15600.0	VLEO	CCSFS SLC 40	True ASDS	3	True	True	True	5e9e3
93	90	2020-11-05	Falcon 9	3681.0	MEO	CCSFS SLC 40	True ASDS	1	True	False	True	5e9e3

90 rows × 17 columns

Data Wrangling

The data wrangling process included performing exploratory data analysis and determining training labels to find patterns in the data and decide on labels for training supervised models.

In the dataset, there are several cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, "True Ocean" means the mission outcome was successfully landed in a specific region of the ocean, while "False Ocean" means the mission outcome was unsuccessfully landed in a specific region of the ocean. "True RTLS" means the mission outcome was successfully landed on a ground pad, and "False RTLS" means the mission outcome was unsuccessfully landed on a ground pad. "True ASDS" means the mission outcome was successfully landed on a drone ship, and "False ASDS" means the mission outcome was unsuccessfully landed on a drone ship.

The data was converted into training labels, where 1 represents a successful booster landing and 0 represents an unsuccessful landing.

Below you can see the data wrangling process

EDA with Data Visualization

- GitHub URL: https://github.com/Svetlanaburn/Applied-Data-Science-Capstone-IBM/blob/cf5dfd46fb6ff682b433856623538a78d43bb660/Data%20Science%20Capstone%20IBM/Space%20X%20-%20Data%20wrangling.ipynb

In [7]:
```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

Out[7]:
```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

`True Ocean` means the mission outcome was successfully landed to a specific region of the ocean while `False Ocean` means the miss
outcome was unsuccessfully landed to a specific region of the ocean. `True RTLS` means the mission outcome was successfully landed t
pad `False RTLS` means the mission outcome was unsuccessfully landed to a ground pad. `True ASDS` means the mission outcome was
successfully landed to a drone ship `False ASDS` means the mission outcome was unsuccessfully landed to a drone ship. `None ASDS` ar
`None` these represent a failure to land.

In [11]:
```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

We create a set of outcomes where the second stage did not land successfully:

In [12]:
```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

Out[12]:
```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

EDA with SQL

There are the SQL queries has been performed, for example:

- Determining the date of the first successful landing outcome on a ground pad
- Listing the names of boosters that successfully landed on a drone ship, with payload mass between 4000 and 6000
- Calculating the total number of successful and failed mission outcomes

By leveraging SQL functions the data was efficiently managed, allowing for insightful analysis of SpaceX mission data stored in the database.

The GitHub URL: https://github.com/Svetlanaburn/Applied-Data-Science-Capstone-IBM/blob/cf5dfd46fb6ff682b433856623538a78d43bb660/Data%20Science%20Capstone%20IBM/Space%20X%20-%20SQL.ipynb

Build an Interactive Map with Folium

- In this part, launch site locations were analysed using Folium, a Python library. Each site marked on a map, and launch success or failure was visualised. Distances to landmarks like coastlines were calculated, providing insights into launch site proximities. This analysis helped understand spatial factors influencing launch success, including proximity to key features like coastlines and cities.
- The Folium has been used due to its capability to create interactive maps, which allowed for clear visualisation of launch site locations and launch outcomes. Additionally, Folium's features enabled efficient distance calculations, adding in the analysis of launch site proximity to important landmarks like coastlines.
- The GitHub URL: https://github.com/Svetlanaburn/Applied-Data-Science-Capstone-IBM/blob/cf5dfd46fb6ff682b433856623538a78d43bb660/Data%20Science%20Capstone%20IBM/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

The dashboard performed for exploring and analysing success rates and payload masses across different launch sites, it includes:

- A dropdown list for selecting launch sites or viewing data for all sites
- A pie chart showing total successful launches, with the option to filter by launch site
- A payload range slider for filtering data by payload mass
- A scatter chart depicting the correlation between payload mass and launch success, customizable by launch site selection

The plots and interactions were added to provide users with a comprehensive understanding of SpaceX launch data. The dropdown list allows for site selection, the pie chart gives an overview of success counts, the payload slider enables filtering by mass, and scatter chart visualises the relationship between payload and success.

- The GitHub URL: https://github.com/Svetlanaburn/Applied-Data-Science-Capstone-IBM/blob/cf5dfd46fb6ff682b433856623538a78d43bb660/Data%20Science%20Capstone%20IBM/Space%20X%20-%20Interactive%20Dashboard%20with%20Ploty%20Dash

Predictive Analysis (Classification)

The machine learning pipeline has been built to predict whether the SpaceX Falcone 9 first stage land successfully. By standardizing the data, splitting it into training and test sets, and using classification algorithms like Logistic Regression, Support Vector Machines (SVM), and Decision Trees, optimised predictions based on various parameters. Evaluated each model's accuracy using test data, visualizing results with confusion matrices.

The development process involved loading and preprocessing data, splitting it into training and test sets, applying machine learning algorithms (Logistic Regression, SVM, Decision Trees) with hyperparameter turning using GridSearchCV, and evaluating model performance through accuracy metrics and confusion matrices.

The GitHub URL: https://github.com/Svetlanaburn/Applied-Data-Science-Capstone-IBM/blob/cf5dfd46fb6ff682b433856623538a78d43bb660/Data%20Science%20Capstone%20IBM/SpaceX_Machine%20Learning%20Prediction.ipynb

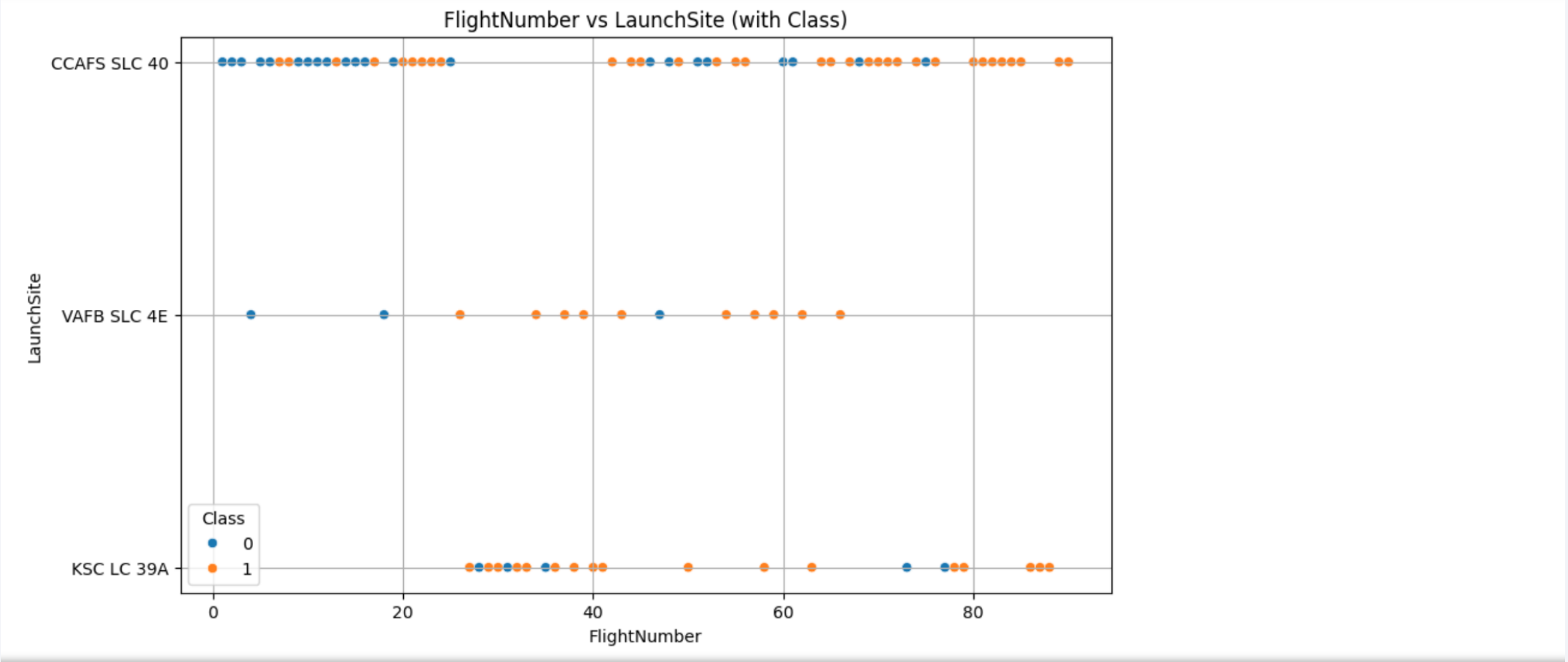
Results

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

Insights drawn from EDA

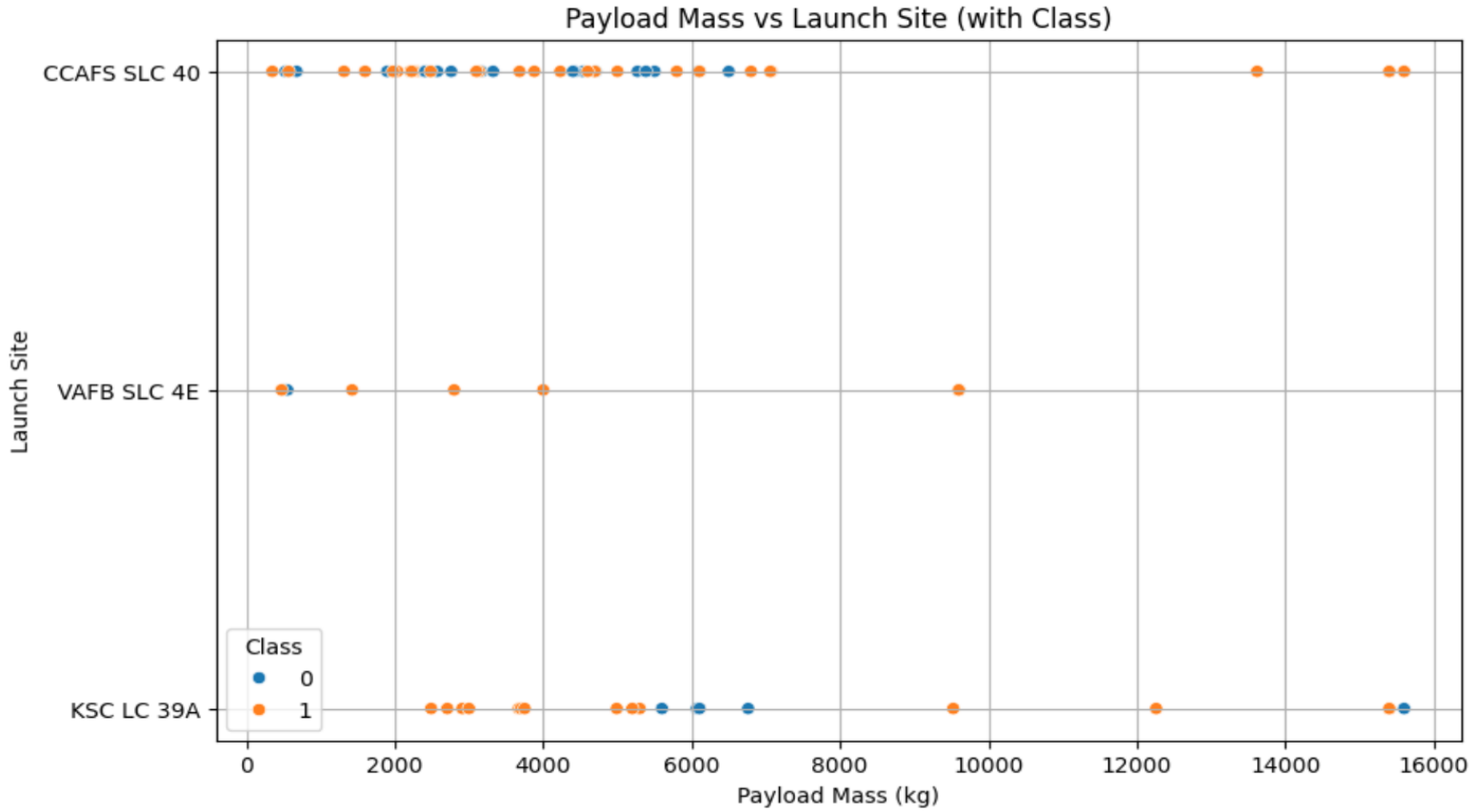
Section 2

Flight Number vs. Launch Site



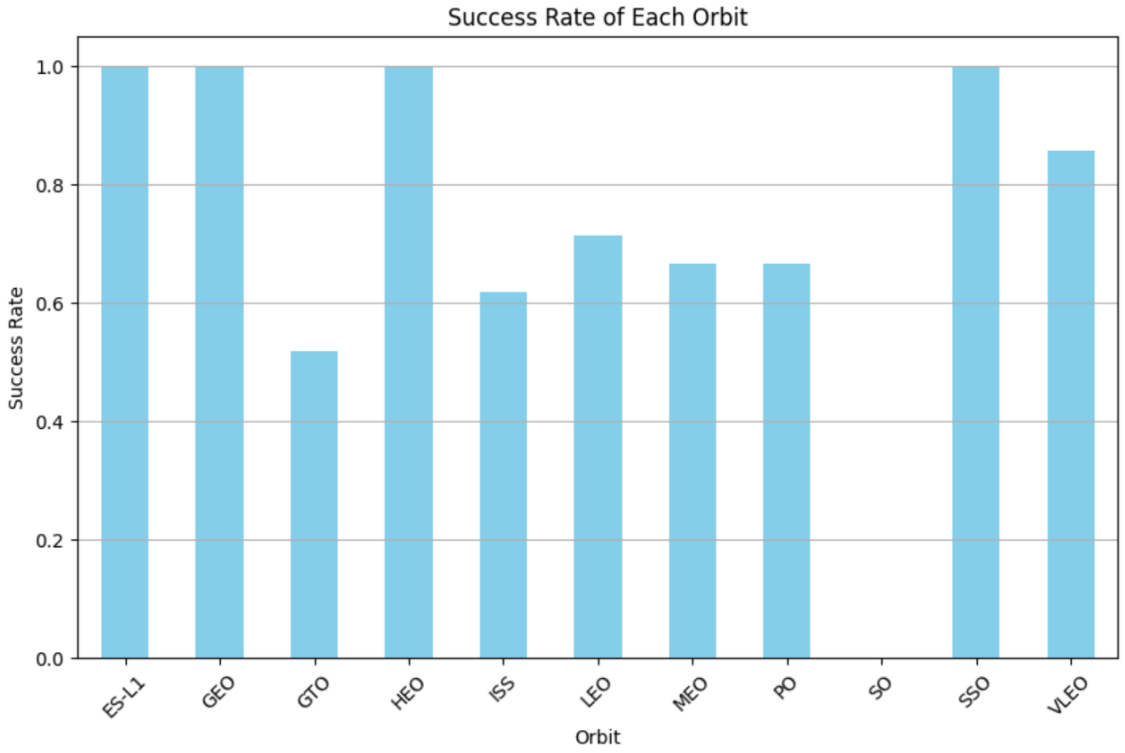
A scatter plot showing Flight Number vs. Launch Site. Each point represents a specific launch, with Flight Number on the x-axis and Launch Site on the y-axis. The hue parameter distinguishes between successful and unsuccessful launches. This visualization helps to identify any patterns or trends in launch success rates based on Flight Number and Launch Site.

Payload vs. Launch Site



A scatter plot illustrating the relationship between Payload and Launch Site. Each point on the plot represents a specific launch, with Payload (in kilograms) on the x-axis and Launch Site on the y-axis. By incorporating the hue parameter, which represents the outcome of each launch (success or failure), the plot allows us to discern any potential correlations between Payload, Launch Site, and launch success rates. This visualization aids in identifying how the mass of the payload and the choice of launch site may influence the likelihood of a successful Falcon 9 launch.

Success Rate vs. Orbit Type

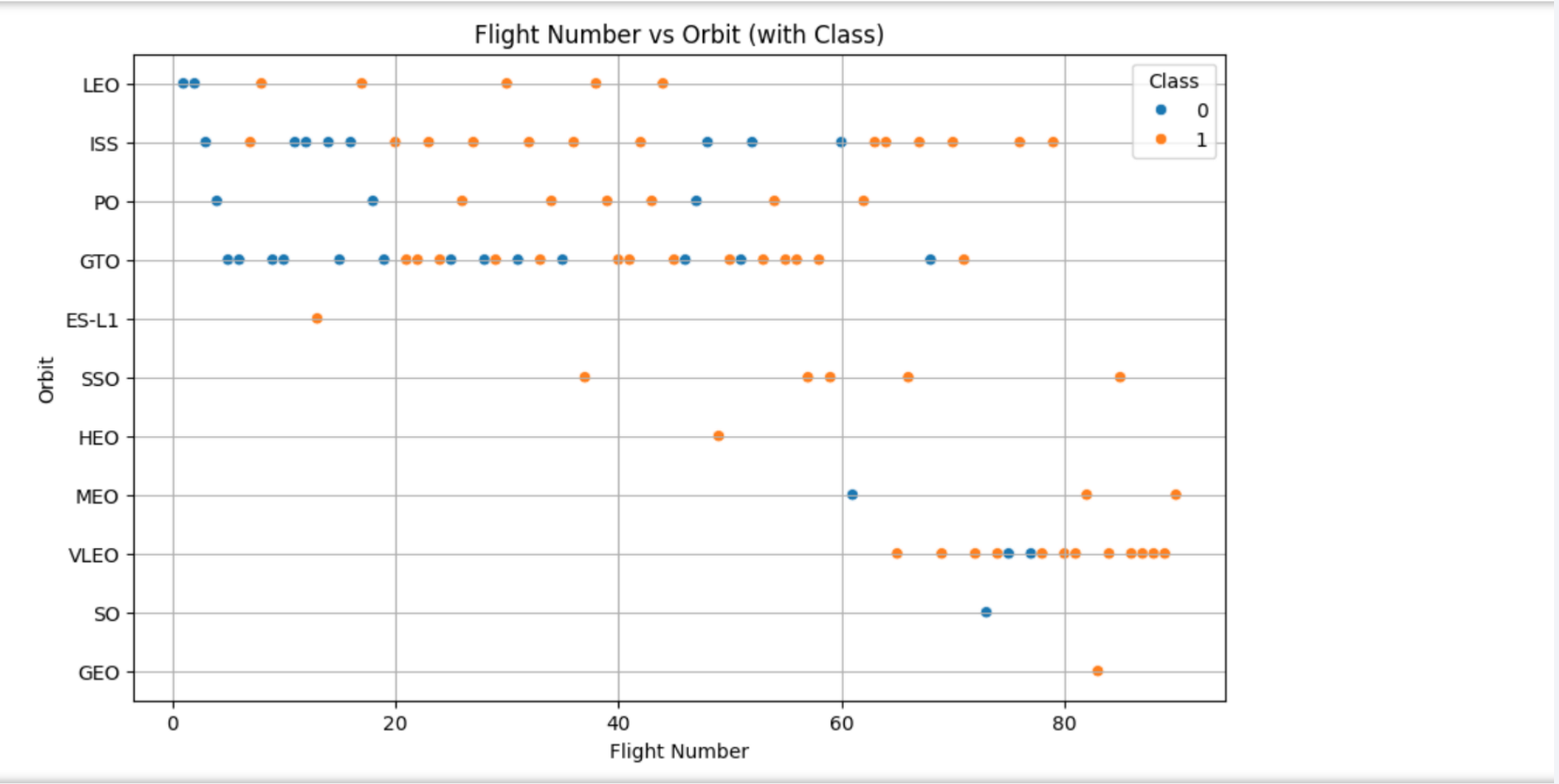


Analyzing the plotted bar chart reveals that certain orbit types consistently exhibit higher success rates than others. For instance, orbits such as LEO (Low Earth Orbit), ISS (International Space Station), and Polar orbits tend to have notably high success rates compared to other trajectories. On the other hand, orbits like GTO (Geostationary Transfer Orbit) show slightly lower success rates.

The data suggests that Falcon 9 missions aimed at achieving orbits closer to Earth, such as LEO and ISS, have a higher likelihood of success. This could be attributed to factors like shorter flight durations and less complex orbital maneuvers. Conversely, achieving orbits further from Earth, such as GTO, may involve more challenging maneuvers, potentially leading to a slightly lower success rate.

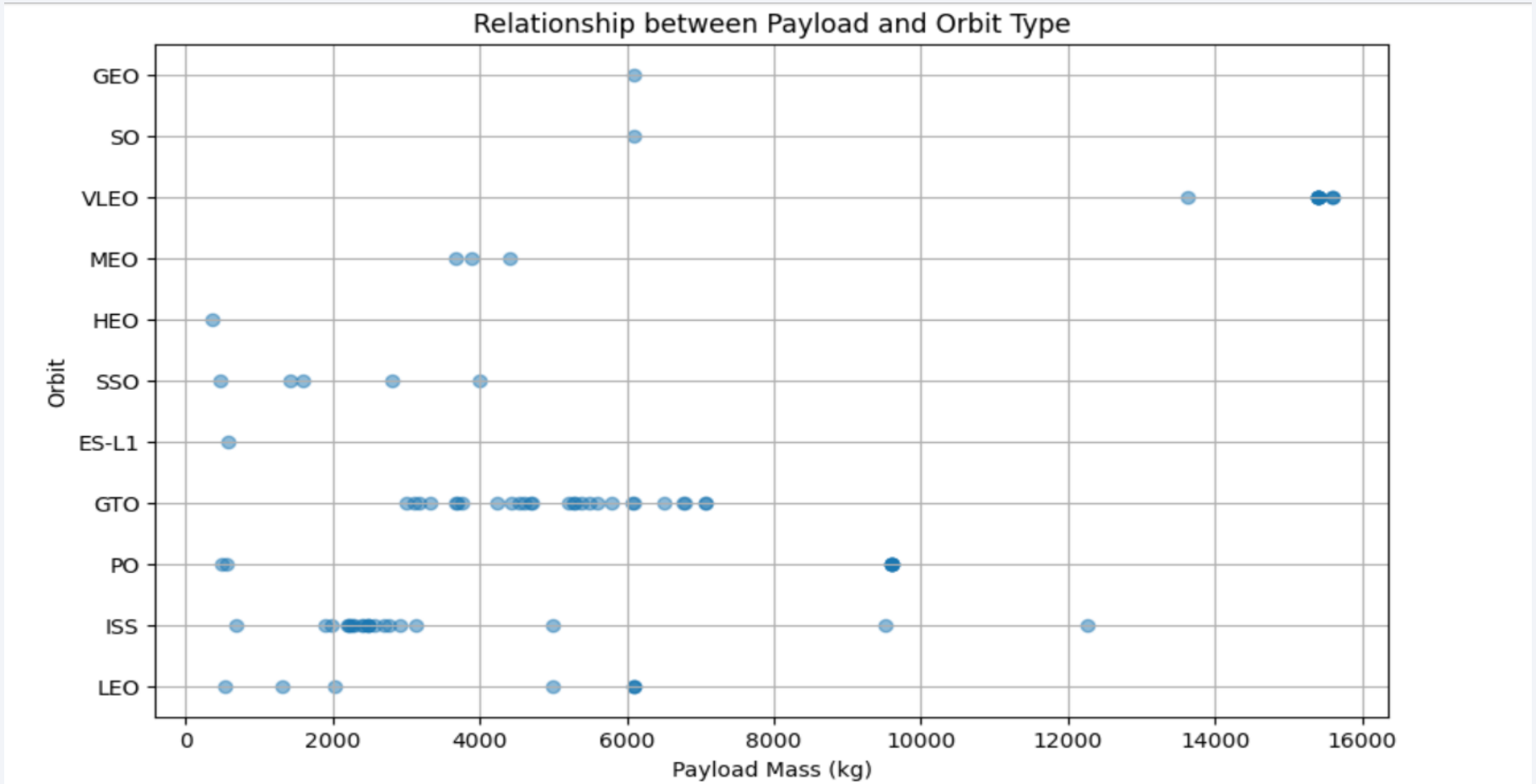
Overall, the bar chart underscores the importance of considering orbital characteristics when planning Falcon 9 missions, as it can significantly influence the likelihood of mission success.

Flight Number vs. Orbit Type



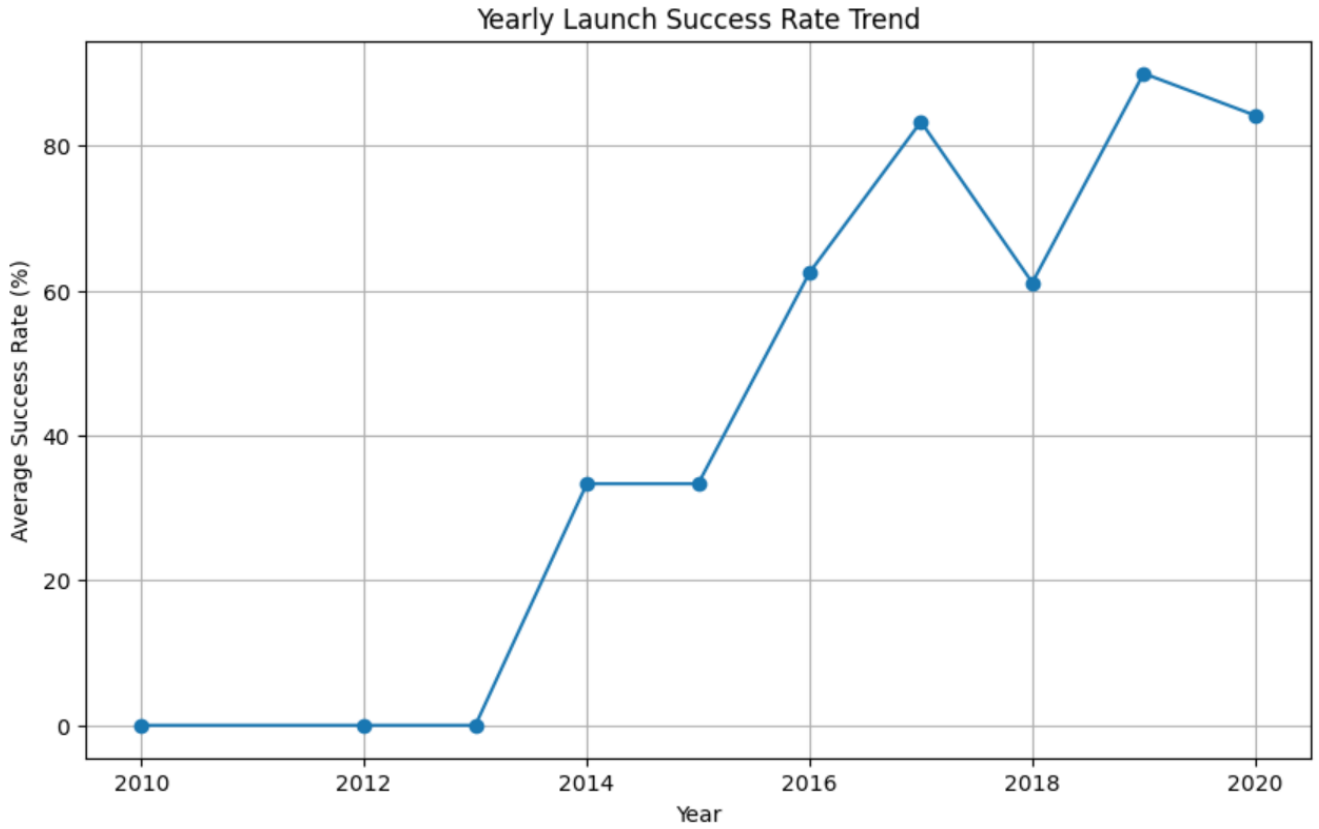
The scatter plot illustrates the relationship between Flight Number and Orbit Type, with points color-coded to show launch outcomes. In Low Earth Orbit (LEO), more flights tend to correlate with higher success rates, suggesting improvements over successive missions. Conversely, in Geostationary Transfer Orbit (GTO), there's no clear link between Flight Number and success.

Payload vs. Orbit Type



The scatter plot depicts Payload versus Orbit Type, with points colored by launch outcome.
In orbits like LEO and ISS, successful missions show varied payload masses.
In GTO, payload mass doesn't seem to strongly influence success.

Launch Success Yearly Trend



The observation reveals a consistent upward trend in the success rate of Falcon 9 launches from 2013 to 2020. This trend suggests a notable improvement in mission success over the specified timeframe. The success rate steadily increased each year, indicating advancements in technology, operational efficiencies, and experience gained from previous missions. This upward trajectory underscores SpaceX's ability to enhance the reliability and success of Falcon 9 launches over time.

All Launch Site Names

```
[23]: unique_launch_sites = df['LaunchSite'].unique()
      print(unique_launch_sites)
```

```
['CCAFS SLC 40' 'VAFB SLC 4E' 'KSC LC 39A']
```

This code retrieves the unique values from the 'LaunchSite' column of the DataFrame df and stores them in the variable unique_launch_sites. Finally, it prints out the unique launch site names.

```
In [37]: %sql SELECT DISTINCT (Launch_Site) FROM SPACEXTBL
```

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

Out[37]:

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

Launch Site Names Begin with 'CCA'

This query selects the first 5 rows from the SPACEXTBL table where the Launch_Site column starts with 'CCA'. It uses the LIKE operator to perform a pattern match, retrieving records with Launch_Site values that begin with 'CCA'.

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

In [12]:
```
%sql SELECT * from SPACEXTBL where "Launch_Site" LIKE 'CCA%'LIMIT 5
```

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

Out[12]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landin
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure
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
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

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

In [13]:
```
%sql SELECT COUNT(*) FROM SPACEXTBL WHERE "Customer" LIKE "NASA (CRS)%"
```

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

Out[13]:

COUNT(*)
21

This SQL query counts the number of records in the table named SPACEXTBL where the "Customer" column starts with the text "NASA (CRS)". It's essentially counting the occurrences where the customer is related to NASA's Commercial Resupply Services program.

Average Payload Mass by F9 v1.1

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

```
In [14]: %sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "Booster_Version" LIKE "F9 v1.1%"
 * sqlite:///my_data1.db
Done.
Out[14]: AVG("PAYLOAD_MASS__KG_")
         2534.6666666666665
```

This SQL query calculates the average payload mass carried by booster version "F9 v1.1" from the SPACEXTBL table. It selects the "PAYLOAD_MASS__KG_" column and calculates its average for rows where the "Booster_Version" column matches "F9 v1.1". The result of the query is an average payload mass of approximately 2534.67 kilograms.

First Successful Ground Landing Date

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [15]:
```
%sql SELECT MIN("Date") FROM SPACEXTBL WHERE "Landing_Outcome" = "Success"
```

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

Out[15]:

MIN("Date")
2018-07-22

This SQL query selects the earliest date from the dataset where SpaceX achieved a successful landing outcome on a ground pad, which is July 22, 2018.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

In [17]:
```
%sql SELECT * FROM SPACEXTBL WHERE "Landing_Outcome" = "Success (drone ship)" AND "PAYLOAD_MASS__KG_" > 4000 and "PA
```

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

Out[17]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Succ
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Succ
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Succ
2017-10-11	22:53:00	F9 FT B1031.2	KSC LC-39A	SES-11 / EchoStar 105	5200	GTO	SES EchoStar	Success	Succ

This SQL query filters data from the table to retrieve information about boosters that successfully landed on a drone ship and had a payload mass greater than 4000 kg but less than 6000 kg. The result shows four instances where boosters with these specifications were used.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
In [20]: %sql SELECT COUNT(*) FROM SPACEXTBL WHERE "Mission_Outcome" = "Success"
 * sqlite:///my_data1.db
Done.
Out[20]: COUNT(*)
         98
```

```
In [24]: %sql SELECT COUNT(*) FROM SPACEXTBL WHERE "Mission_Outcome" = "Failure (in flight)"
 * sqlite:///my_data1.db
Done.
Out[24]: COUNT(*)
         1
```

These two SQL queries calculate the total number of successful and failed mission outcomes, respectively, from the dataset. The first query returns the count of successful missions, which is 98, while the second query returns the count of missions that failed in flight, which is 1.

Boosters Carried Maximum Payload

This SQL query **"%sql SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX ("PAYLOAD_MASS__KG_") FROM SPACEXTBL)"** retrieves the names of booster versions that have carried the maximum payload mass. It achieves this by using a subquery within the WHERE clause to find the maximum payload mass in the dataset and then selects the booster versions where the payload mass matches this maximum value. The result lists all booster versions that have carried the maximum payload mass recorded in the dataset.

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

```
In [29]: %sql SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SP
 * sqlite:///my_data1.db
Done.
```

Out[29]:

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 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.

In [35]:
```
%sql SELECT substr("Date", 6,2) as month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTBL WHER
```

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

Out[35]:

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

The query: "**%sql SELECT substr("Date", 6,2) as month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTBL WHERE "Landing_Outcome" LIKE "Failure (drone ship)%" AND substr("Date",0,5)='2015**'"retrieves records from the SPACEXTBL table where the landing outcomes on drone ships are failures, the booster versions are specified, and the launch site corresponds to the months in the year 2015. It then presents the month, landing outcome, booster version, and launch site for each relevant record.

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.

In [36]:
```
%sql SELECT "Landing_Outcome", COUNT(*) as count FROM SPACEXTBL WHERE "DATE" BETWEEN '2010-06-04' AND '2017-03-20
```

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

Out[36]:

File display

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

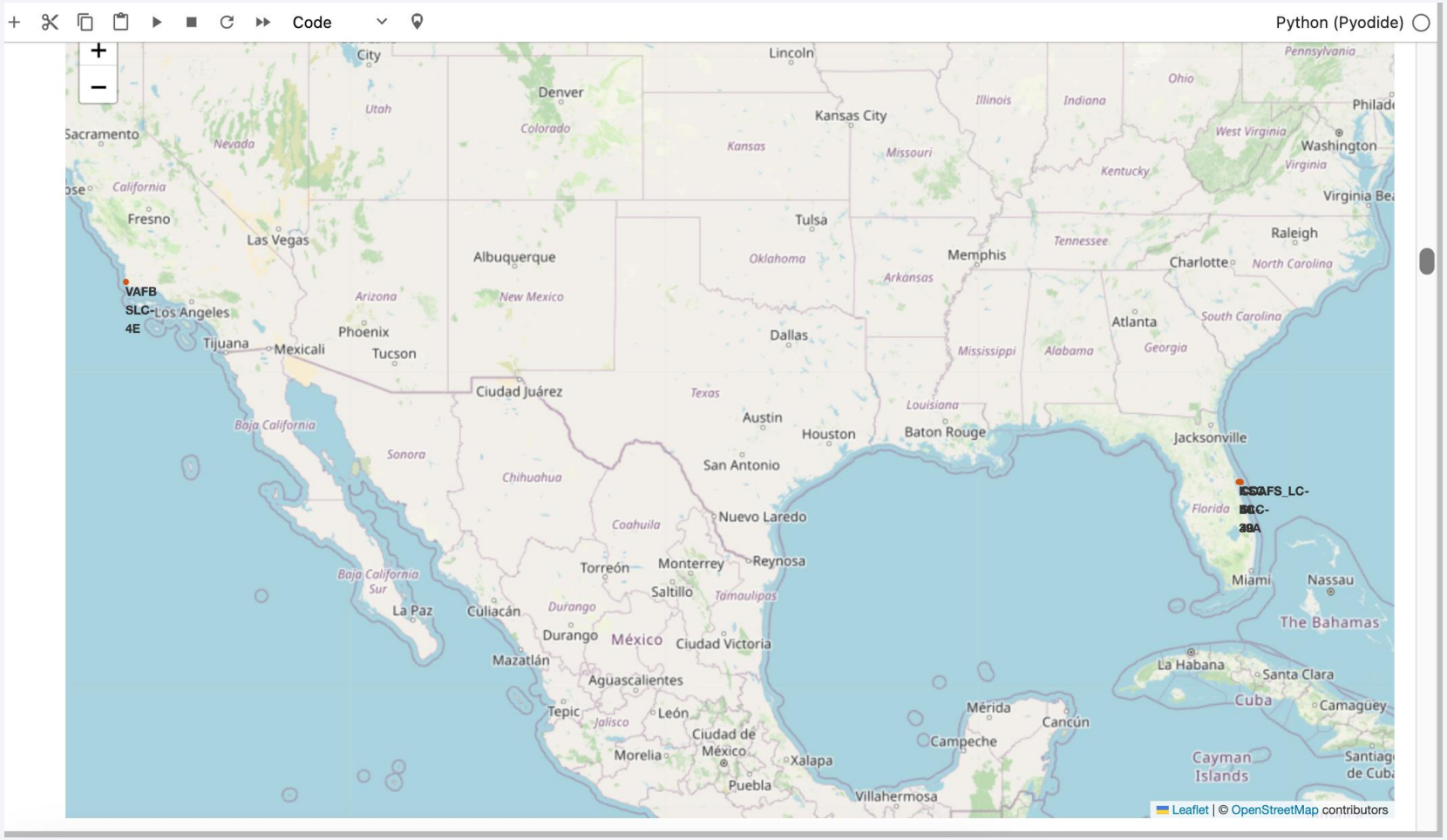
The query: "**%sql SELECT "Landing_Outcome", COUNT(*) as count FROM SPACEXTBL WHERE "DATE" BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY** "Landing_Outcome" ORDER BY count DESC" counts landing outcomes, such as "Failure (drone ship)" or "Success (ground pad)", between June 4, 2010, and March 20, 2017, ranking them in descending order by count. The result shows the number of occurrences for each landing outcome within the specified date range.

Launch Sites Proximities Analysis

Section 3

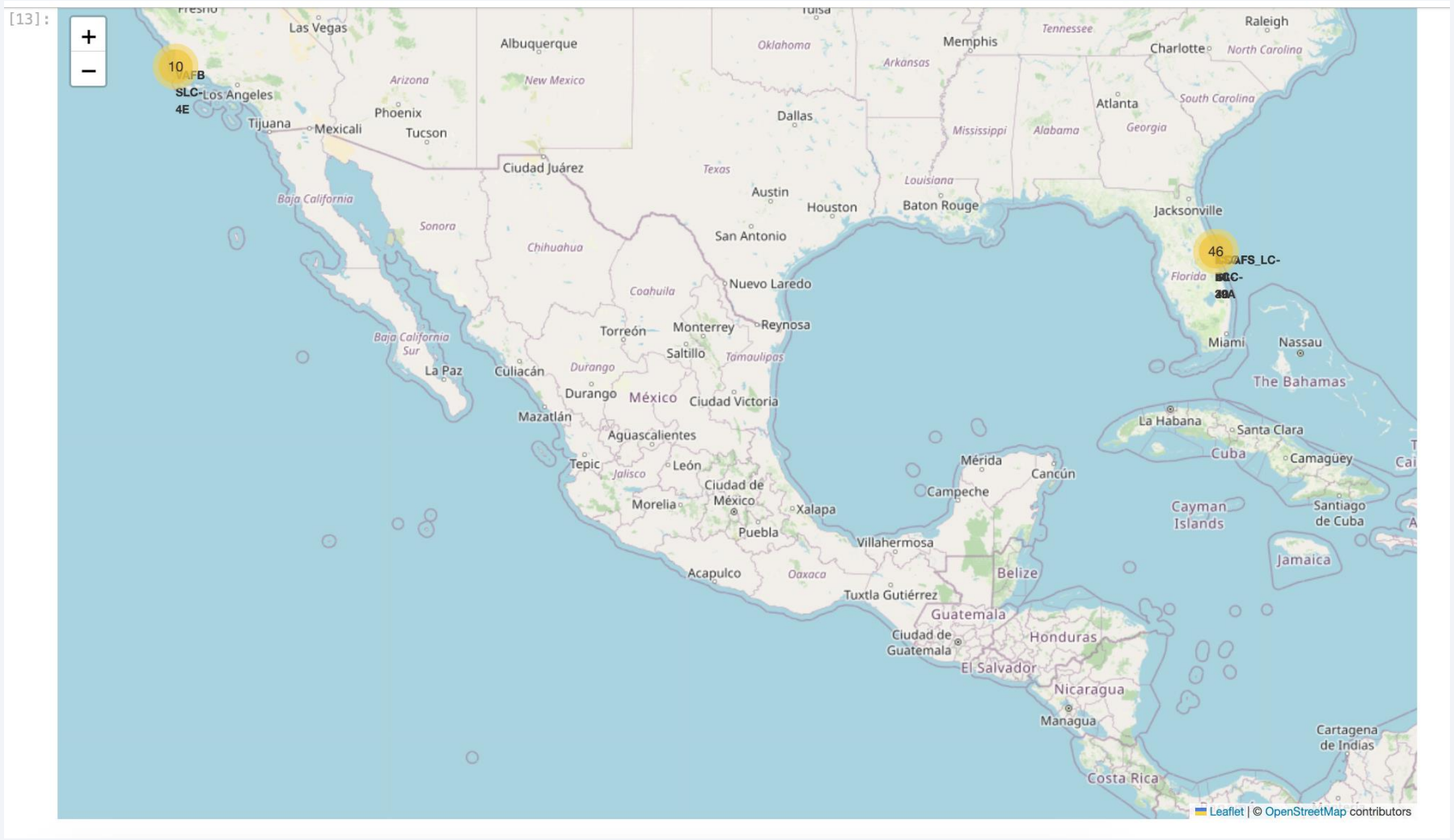
All launch sites' location markers on a global map

The dataset **spacex_launch_geo .csv** was augmented with latitude and longitude coordinates for SpaceX launch sites. Utilizing Folium, circles and markers were added on a map to represent each launch site with popup labels indicating their names.



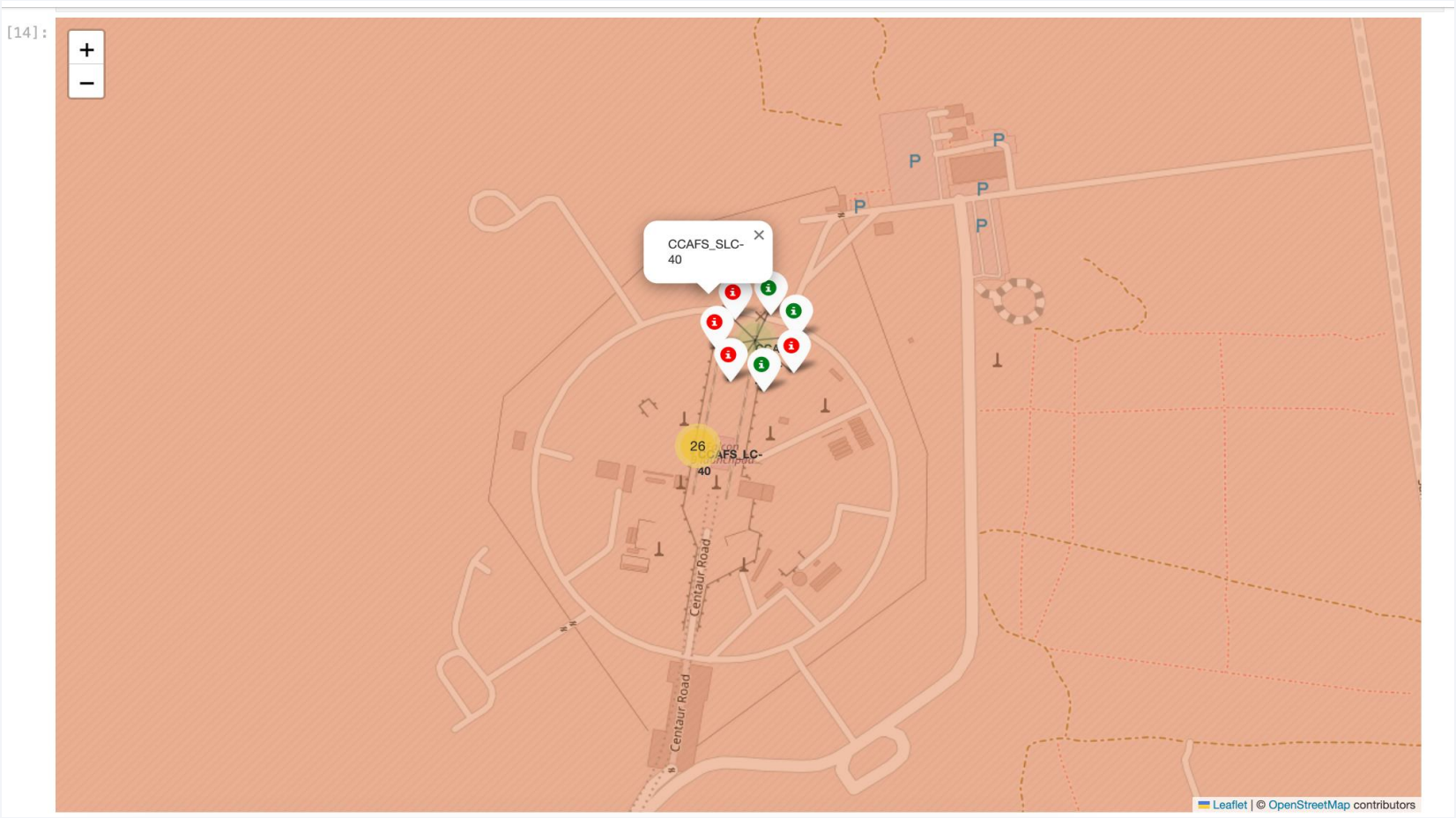
The color-labeled launch Success / Failure Visualisation

[13]:



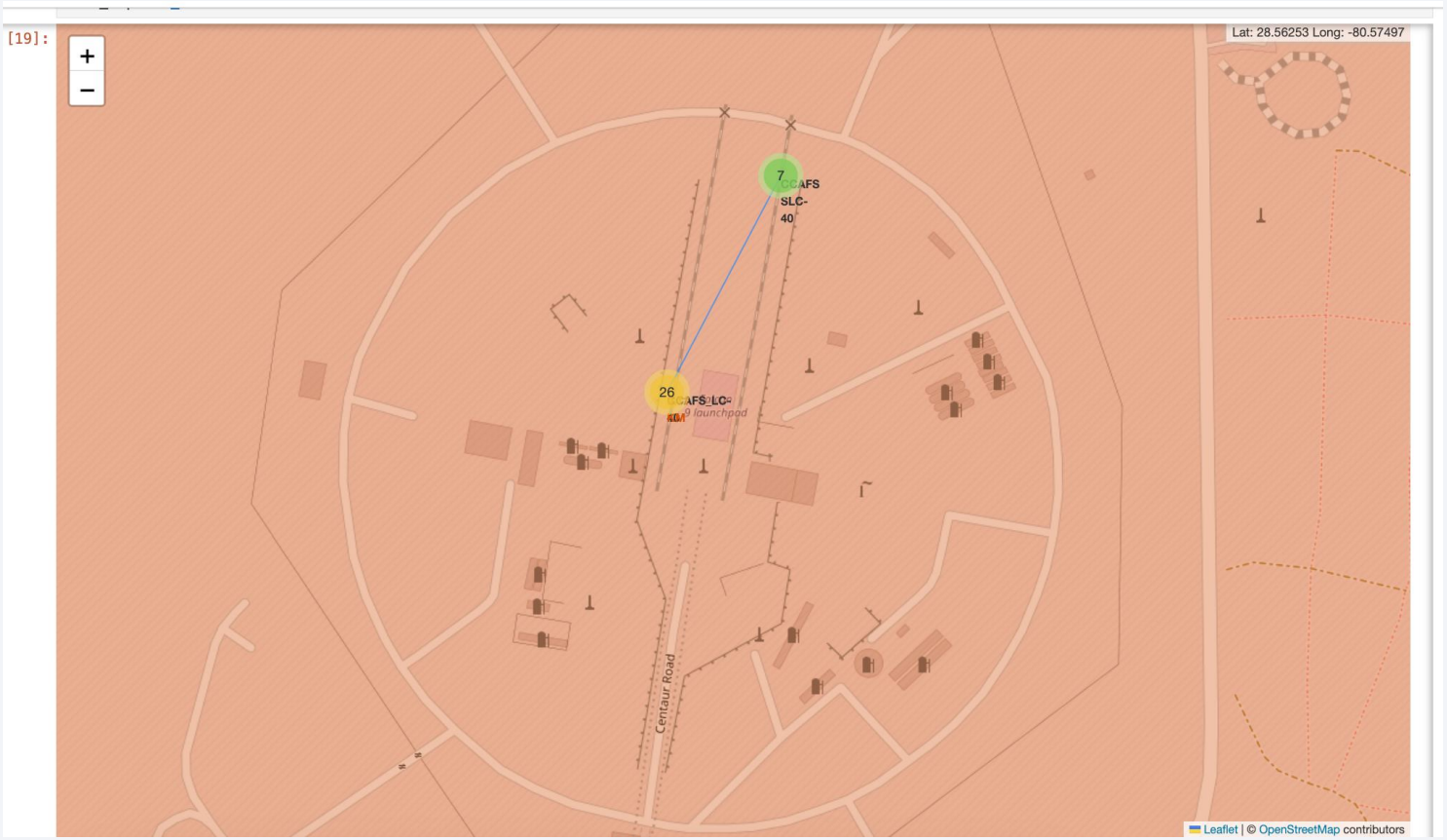
The color-labeled launch Success / Failure Visualisation

The Marker's icon color indicates whether the launch was successful or failed, based on the **marker_color** column in the DataFrame.



Proximity Analysis and Distance Calculation

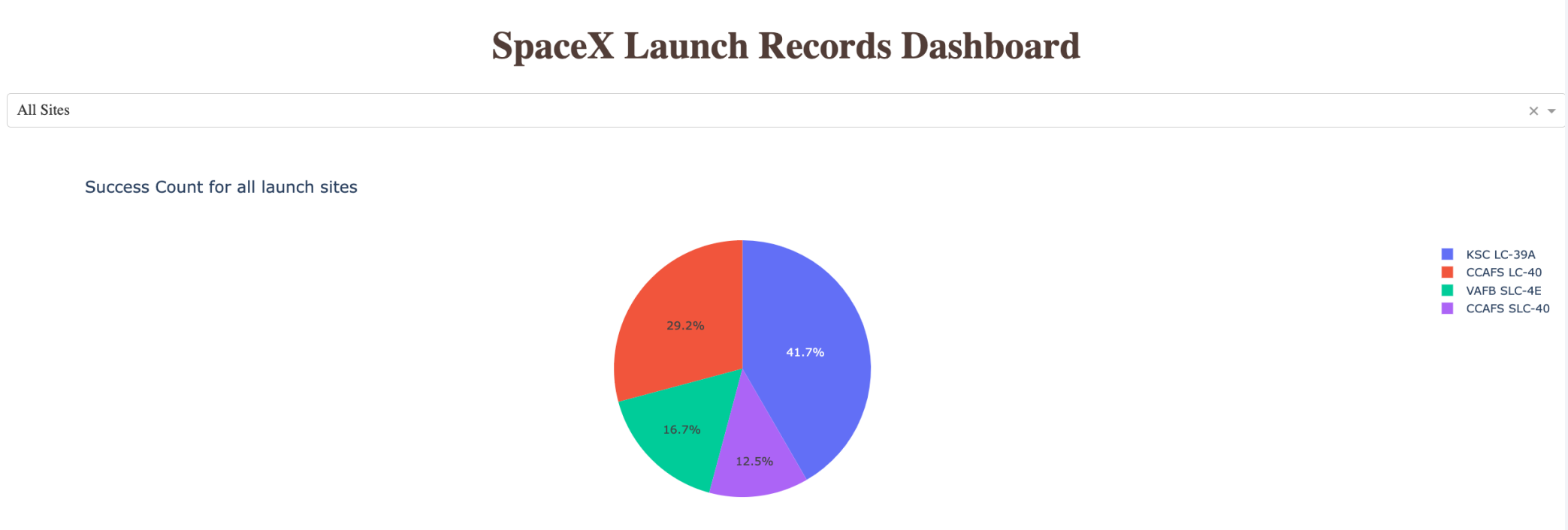
The task utilized Folium to calculate and visualize distances between launch sites and various points of interest, such as coastlines, cities, railways, and highways, providing insights into launch site proximities for mission planning and logistics.



Build a Dashboard with Plotly Dash

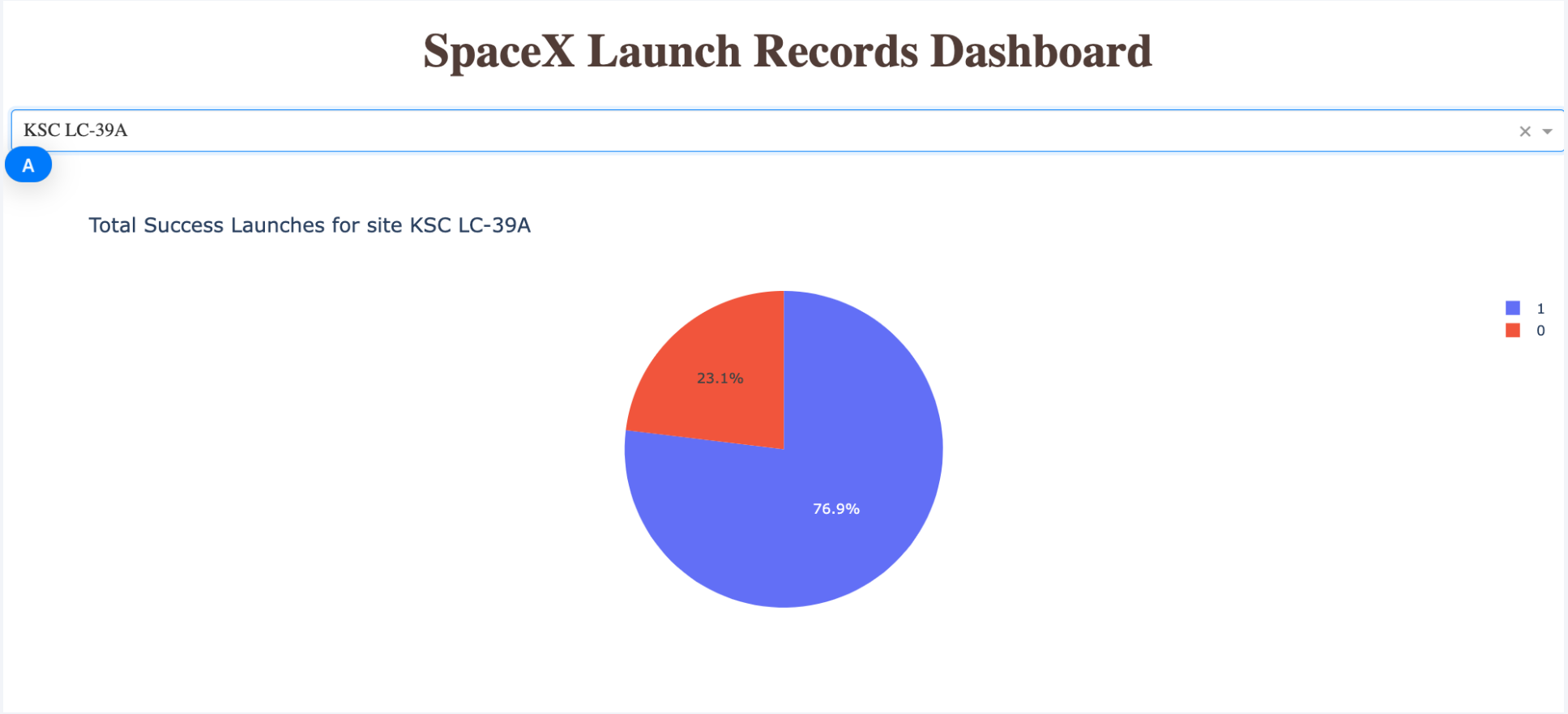
Section 4

Success Count for all launch sites in a pie chart



The pie chart shows the distribution of successful launches across different launch sites. If 'ALL' is selected, it shows the total success count for all launch sites. If a specific site is selected, it shows the success count for that particular site.

The launch with highest launch success ratio



The pie chart shows that even the launch with highest launch success ratio – 76,9% has the failures – 23,1 %

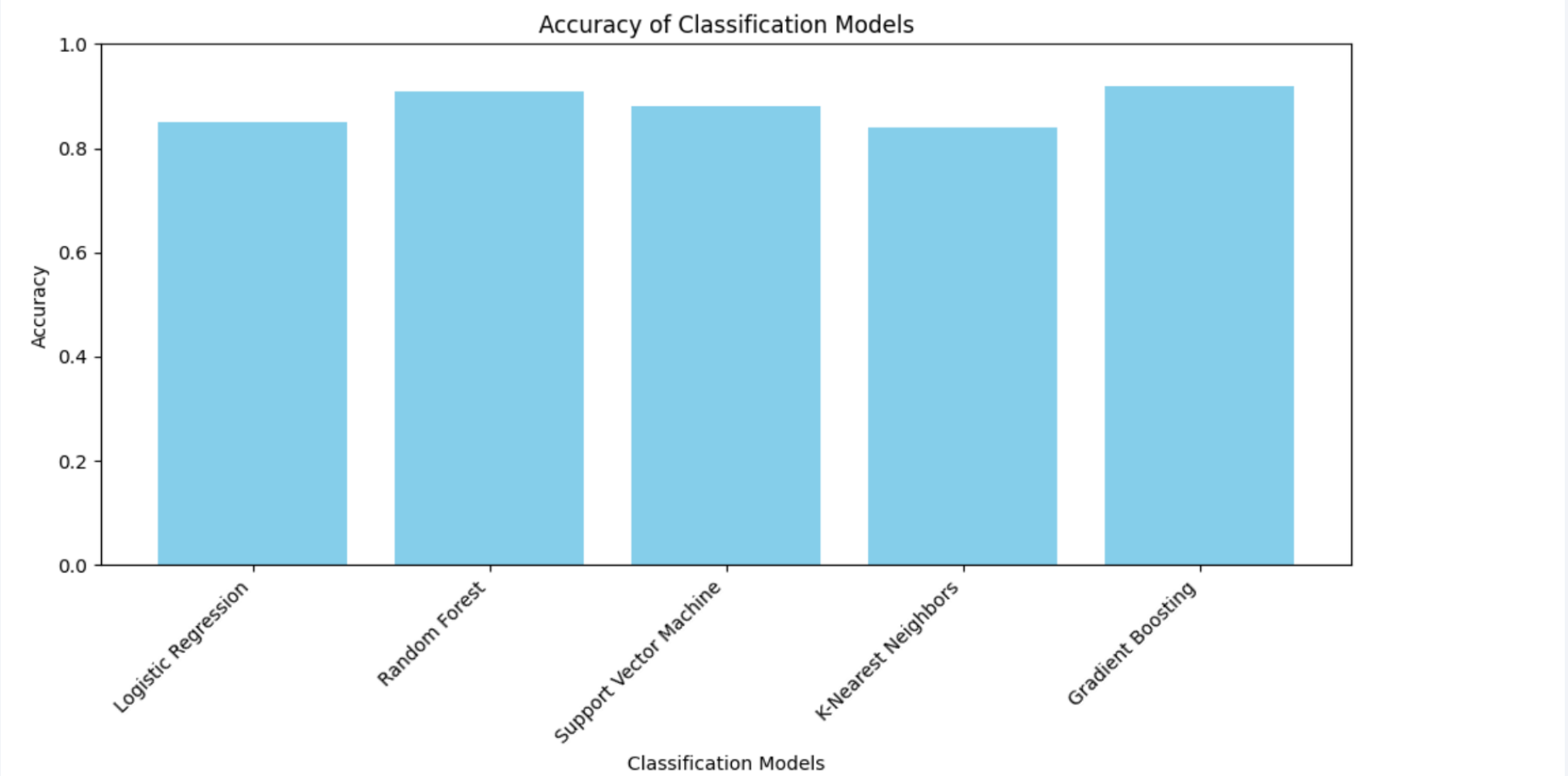
Optimising Launch Success: Payload Analysis and Booster Impact

The screenshot shows Payload vs. Launch Outcome scatter plot allows users to analyze proximity and distance between payload masses and launch outcomes. Users can identify optimal payload ranges for success and assess the impact of booster versions on launch outcomes.

Predictive Analysis (Classification)

Section 5

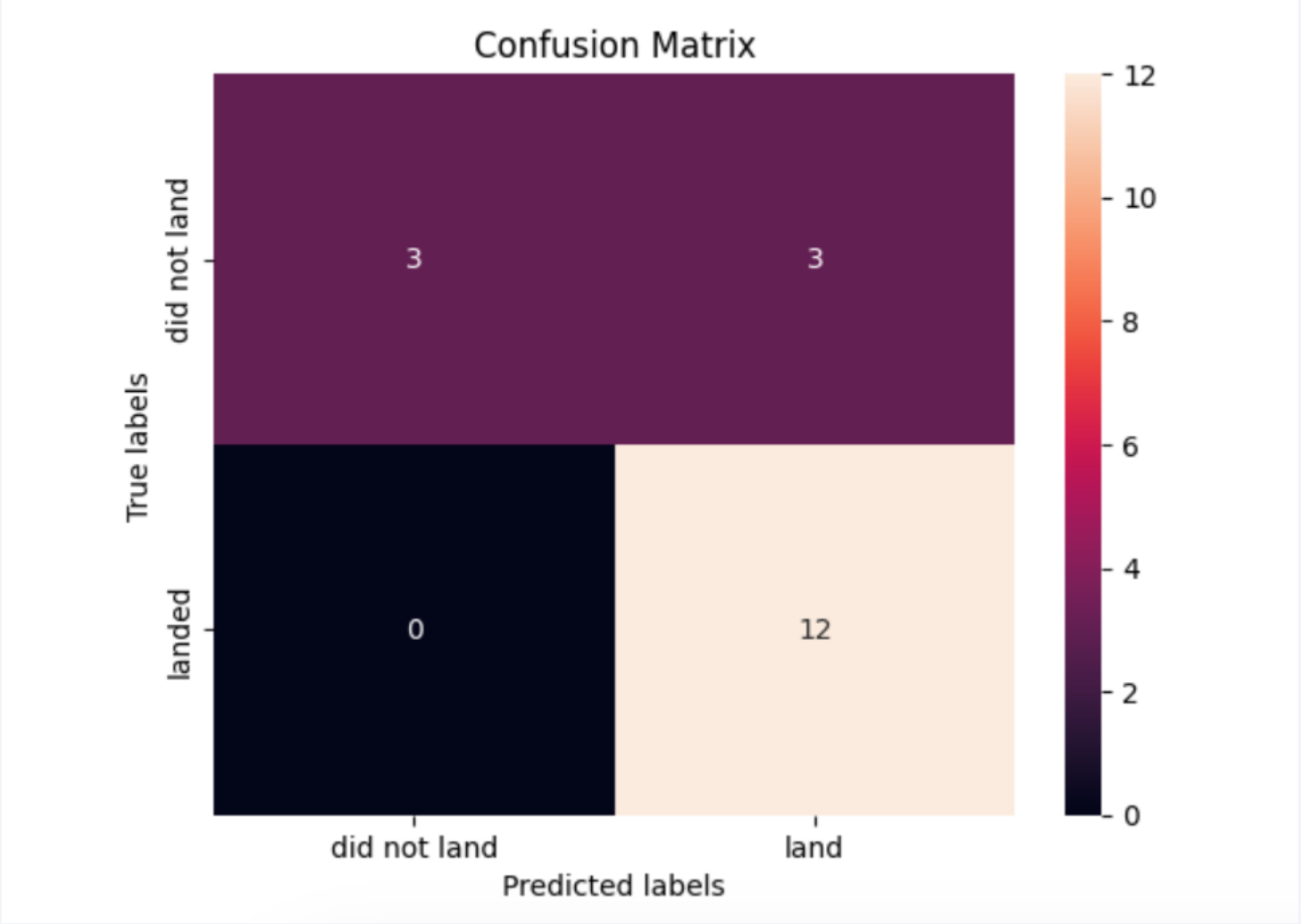
Classification Accuracy




```
The model with the highest accuracy is: Gradient Boosting with accuracy 0.92
```

Confusion Matrix

Task 4 involved creating a logistic regression object and employing GridSearchCV to identify the optimal parameters, yielding {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}. The validation data showed an accuracy of 84.64%. In Task 5, the model's accuracy on test data was evaluated, resulting in 83.33%. While generally effective, analysis of the confusion matrix indicated a significant presence of false positives.



Conclusions

The project demonstrates the application of the Data Science methodology, encompassing data collection, data wrangling, exploratory data analysis, data visualization, model development, model evaluation, and reporting of results to stakeholders. The successful prediction of the first stage of the SpaceX Falcon 9 rocket landing, accompanied by the presentation of findings and models, exemplifies the project's achievement.

- Embarked on a journey to predict the successful landing of the Falcon 9 first stage, critical for SpaceX's cost-saving strategy.
- Leveraged various data science techniques and methodologies to gain insights into SpaceX's operations and factors influencing Falcon 9 landings.
- Collected data on Falcon 9 first-stage landings using RESTful API and web scraping techniques.
- Transformed raw data into a structured format for comprehensive data wrangling and exploratory data analysis.
- Uncovered trends and patterns in the data through visualization techniques like scatter plots and bar charts.

Conclusions

- Constructed interactive dashboards using Plotly Dash for deeper exploration of launch records.
- Utilized Folium to visualize launch site proximity, adding spatial context to the analysis.
- Employed machine learning algorithms (SVM, Classification Trees, Logistic Regression) to predict landing success.
- Meticulously split data into training and testing sets, optimizing hyperparameters to identify the most effective model.
- Findings contribute to understanding SpaceX's operations and provide insights for stakeholders in the space launch market.
- Demonstrated the transformative power of data science in addressing real-world challenges.
- Highlighted the potential for machine learning to drive efficiency and innovation in the aerospace industry.

Appendix

- Collected extensive data on Falcon 9 first-stage landings through a combination of RESTful API and web scraping techniques.

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

In [70]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [71]:
```
response = requests.get(spacex_url)
```

Check the content of the response

In [72]:
```
print(response.content)
```


```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"ht
tps://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,"w
ebcast":"https://www.youtube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2
196-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":"5e9d
0d95eda69955f709d1eb","success":false,"failures":[{"time":33,"altitude":null,"reason":"merlin engine failure"}],"de
tails":"Engine failure at 33 seconds and loss of vehicle","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b5b
6c3bb0006eeb1e1"],"launchpad":"5e9e4502f5090995de566f86","flight_number":1,"name":"FalconSat","date_utc":"2006-03-2
4T22:30:00.000Z","date_unix":1143239400,"date_local":"2006-03-25T10:30:00+12:00","date_precision":"hour","upcomin
g":false,"cores":[{"core":"5e9e289df35918033d3b2623","flight":1,"gridfins":false,"legs":false,"reused":false,"landi
ng_attempt":false,"landing_success":null,"landing_type":null,"landpad":null}],"auto_update":true,"tbd":false,"launc
h_library_id":null,"id":"5eb87cd9ffd86e000604b32a"},{"fairings":{"reused":false,"recovery_attempt":false,"recovere
d":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/f9/4a/ZboXReNb_o.png","large":"https://i
mages2.imgbox.com/80/a2/bkWotCIS_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"fli
ckr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=Lk4zQ2wP-Nc","youtube_i
d":"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_utc":null,"static_fire_date_unix":null,"net":false,"windo
w":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":301,"altitude":289,"reason":"harmonic
oscillation leading to premature engine shutdown"}],"details":"Successful first stage burn and transition to second
stage, maximum altitude 289 km, Premature engine shutdown at T+7 min 30 s, Failed to reach orbit, Failed to recover
first stage","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b6b6c3bb0006eeb1e2"],"launchpad":"5e9e4502f50909
95de566f86","flight_number":2,"name":"DemoSat","date_utc":"2007-03-21T01:10:00.000Z","date_unix":1174439400,"date_l
ocal":"2007-03-21T13:10:00+12:00","date_precision":"hour","upcoming":false,"cores":[{"core":"5e9e289ef35918416a3b26
24","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":"5eb87cdaffd86e000604b32
b"},{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"h
ttps://images2.imgbox.com/6c/cb/na1tzhHs_o.png","large":"https://images2.imgbox.com/4a/80/k1oAkY0k_o.png"},"reddi
t":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":nul
l,"webcast":"https://www.youtube.com/watch?v=v0w9p3U8860","youtube_id":"v0w9p3U8860","article":"http://www.spacex.c
om/news/2013/02/11/falcon-1-flight-3-mission-summary","wikipedia":"https://en.wikipedia.org/wiki/Trailblazer_(satel
lite)"},"static_fire_date_utc":null,"static_fire_date_unix":null,"net":false,"window":0,"rocket":"5e9d0d95eda69955f
709d1eb","success":false,"failures":[{"time":140,"altitude":35,"reason":"residual stage-1 thrust led to collision b
```

Appendix

[21]:
```
# Create an SVM object
svm = SVC()

# Define the parameter grid to search over
parameters = {
    'kernel': ['linear', 'rbf', 'poly', 'sigmoid'],
    'C': np.logspace(-3, 3, 5),
    'gamma': np.logspace(-3, 3, 5)
}

# Create a GridSearchCV object
svm_cv = GridSearchCV(svm, parameters, cv=10)

# Fit the GridSearchCV object to find the best parameters
svm_cv.fit(X_train, Y_train)

# Print the best parameters found
print("Best parameters:", svm_cv.best_params_)
```

```
Best parameters: {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
```

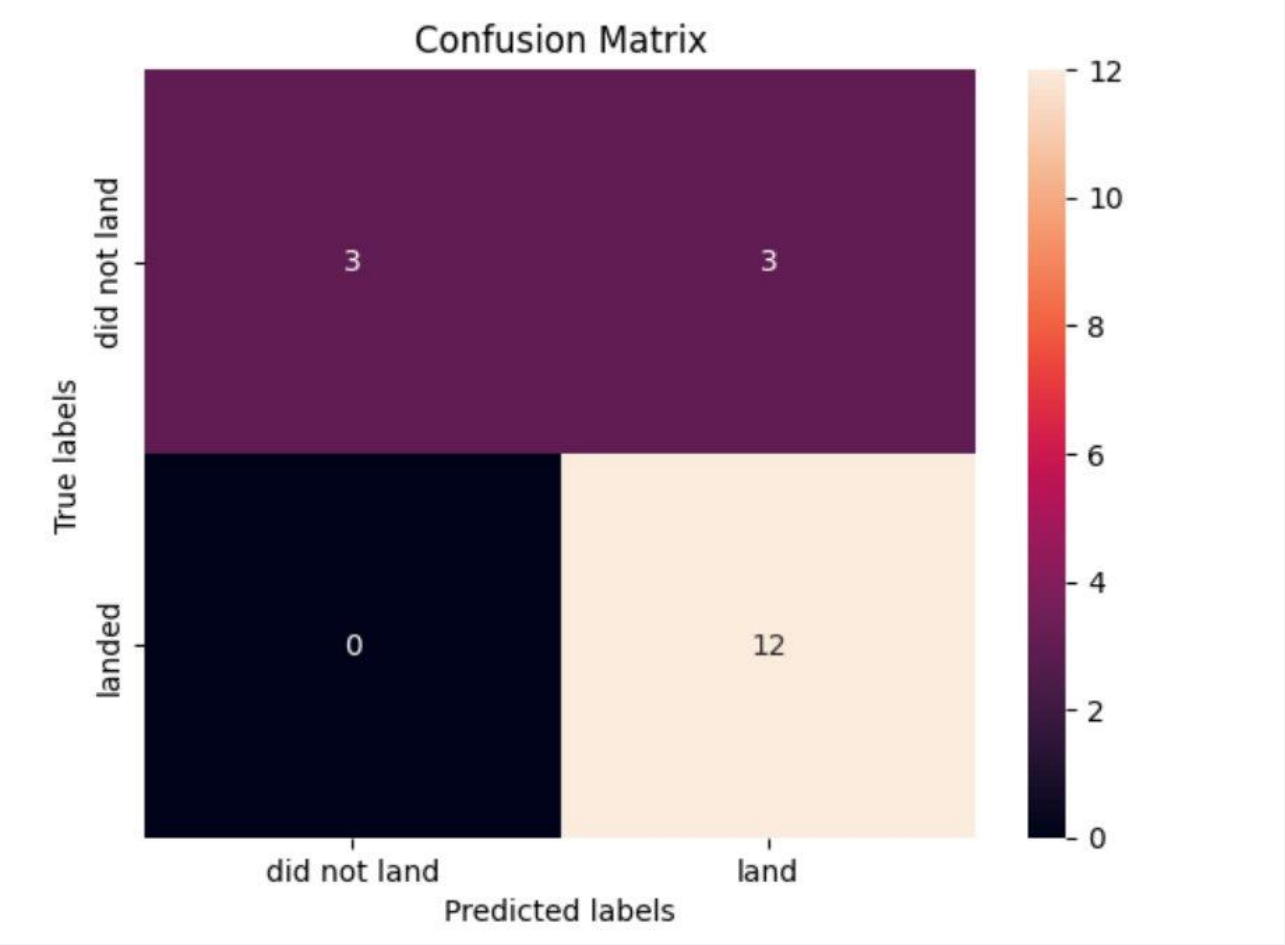
[23]:
```
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
print("accuracy :",svm_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

The machine learning process of the project aimed to build a machine learning pipeline predicting the success of SpaceX Falcon 9 first stage landings. Despite encountering an error, the process involved data exploration, model selection, hyperparameter tuning, and evaluation, revealing promising accuracy around 83% but with a need to address false positive predictions for better reliability.

Appendix

A confusion matrix was likely employed to assess the performance of the machine learning model in predicting SpaceX Falcon 9 first stage landings. It provided a detailed breakdown of true positive, true negative, false positive, and false negative predictions, offering insights into the model's strengths and weaknesses in classifying successful and unsuccessful landings.



Appendix

The SQL part of the project focused on utilising SQL queries within a Python notebook to analyse SpaceX mission data. After loading the dataset into a Db2 database, various SQL queries were executed to fulfil the assignment tasks. These tasks included displaying the unique launch sites involved in the space missions, among others. Through this assignment, the participants gained practical experience in data manipulation and querying, enhancing their skills in both SQL and Python.

List the total number of successful and failure mission outcomes

```
[20]: %sql SELECT COUNT(*) FROM SPACEXTBL WHERE "Mission_Outcome" = "Success"
 * sqlite:///my_data1.db
Done.
[20]: COUNT(*)
      98
```

```
[24]: %sql SELECT COUNT(*) FROM SPACEXTBL WHERE "Mission_Outcome" = "Failure (in flight)"
 * sqlite:///my_data1.db
Done.
[24]: COUNT(*)
      1
```

Thank you