IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Akerke Absalykova
24/10/2023

Outline

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

Executive Summary

Summary of Methodologies:
In this project, I employed a series of well-defined methodologies to analyze data related to SpaceX's landing outcomes.

- **Data Collection:** I gathered data from two primary sources - the SpaceX API and the SpaceX Wikipedia page. This data collection process enabled me to compile a comprehensive dataset, and I created a 'class' column to categorize successful landings.
- **Data Exploration:** I explored the dataset using a range of methodologies, including SQL queries, data visualization, Folium maps, and the creation of interactive dashboards. This extensive exploration allowed me to gain a deep understanding of the data's characteristics.
- **Feature Selection:** I meticulously identified and selected relevant columns to serve as features for the subsequent machine learning models.
- **Data Preprocessing:** To prepare the dataset for modeling, I transformed categorical variables into binary format through one-hot encoding. This transformation was crucial to make the data suitable for machine learning.
- **Standardization and Model Optimization:** The data was standardized to ensure uniformity for modeling. Subsequently, I employed GridSearchCV to identify the best hyperparameters for the machine learning models, fine-tuning their performance.
- **Machine Learning Models:** The result of these efforts was the development of four machine learning models
- - Logistic Regression, Support Vector Machine, Decision Tree Classifier, and K Nearest Neighbors.

Executive Summary

Summary of Results:
The application of these methodologies led to several key findings:

- **Model Performance**: All four machine learning models produced remarkably similar results, achieving an accuracy rate of approximately 83.33%. This level of consistency across different models is noteworthy.
- **Over-prediction of Successful Landings**: One notable observation was that all models demonstrated a tendency to over-predict successful landings. While the models performed well, this tendency indicates a possible imbalance or limitation in the dataset.
- **Need for Additional Data**: To enhance the accuracy and precision of these models, it is clear that further data collection is warranted. Additional data points can contribute to refining the models and making them more reliable.

In conclusion, the application of rigorous methodologies in data collection, exploration, and machine learning yielded valuable insights into SpaceX's landing outcomes. The results, while promising, emphasize the importance of expanding the dataset to further improve the models' predictive capabilities.

Introduction

Project Background and Context:

In the dynamic landscape of space exploration, SpaceX has emerged as a pioneering force, revolutionizing rocket launches with its cost-effective Falcon 9 program. At a mere $62 million per launch, SpaceX has disrupted the traditional industry, where prices often exceed $165 million. A substantial part of this cost efficiency is attributable to SpaceX's groundbreaking ability to recover and reuse the first stage of its Falcon 9 rockets.

In this context, I embark on a crucial project with a distinctive purpose: to predict the successful landing of the SpaceX Falcon 9 first stage. The outcome of this prediction holds the key to estimating launch costs, a vital consideration for both SpaceX and other players in the field.

Introduction

Problems I Seek to Address:

1. Cost Estimation: I aim to provide accurate cost estimates for Falcon 9 rocket launches. These estimates are crucial not only for SpaceX but also for potential competitors vying for rocket launch contracts. By understanding the landing outcomes, I can offer valuable insights into the financial implications of space missions.

2. Competitive Assessment: I seek to answer the question of whether alternate companies should bid against SpaceX for rocket launches. In doing so, I can contribute to informed decision-making within the space launch industry.

3. Optimizing Space Exploration: My project ultimately addresses the broader challenge of optimizing space exploration by harnessing data-driven methodologies to forecast the success of rocket landings. The impact of such predictions extends beyond cost considerations, influencing the strategies and innovations that drive space exploration into the future.

Section
1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Combined data from SpaceX public API and SpaceX Wikipedia page
- Perform data wrangling
  - Classifying true landings as successful and unsuccessful otherwise
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Tuned models using GridSearchCV

Data Collection

The data collection process involved a two-pronged approach, combining API requests from SpaceX's public API and web scraping data from a table within SpaceX's Wikipedia entry. This method allowed us to compile a comprehensive dataset for analysis.

SpaceX API Data Columns:

- FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude

Data Collection

Wikipedia Web Scraped Data Columns:

- Flight Number
- Launch Site
- Payload
- Payload Mass
- Orbit
- Customer
- Launch Outcome
- Version Booster
- Booster Landing
- Date
- Time

The following slides will illustrate the data collection flowcharts, showcasing the steps for both the API data retrieval and the web scraping process. This multi-faceted data collection approach has empowered us with a rich dataset for thorough analysis and model development.

Data Collection – SpaceX API

1. Request (SpaceX APIs)
2. JSON file + Lists(Launch Site, Booster Version, Payload Data)
3. Json_normalize to DataFrame data from JSON
4. Filter data to only include Falcon 9 launches
5. Cast dictionary to a DataFrame
6. Dictionary relevant data
7. Replace missing PayloadMass values with mean

Data Collection - Scraping

Link to Github Data Collection Project

Request Wikipedia html

BeautifulSoup html5lib Parser

Find launch info html table

Create dictionary

Iterate through table cells to extract data to dictionary

Cast dictionary to DataFrame

Data Wrangling

Creating a Training Label

I've undertaken a critical data wrangling task to prepare our dataset for machine learning. The key focus of this task is to generate a training label that distinguishes landing outcomes as either 'successful' (1) or 'failure' (0).

Understanding Outcome Components:

Our Outcome column is composed of two essential components: 'Mission Outcome' and 'Landing Location.'

Introducing the 'class' Label: I've introduced a novel training label named 'class,' where a value of 1 signifies a 'Mission Outcome' marked as 'True,' indicating a successful landing. Conversely, a value of 0 is assigned when 'Mission Outcome' is marked as 'False' or 'None,' signifying a landing failure.

Value Mapping for the 'class' Label:

- 'True' values in 'Mission Outcome,' particularly 'ASDS,' 'RTLS,' and 'Ocean,' are set to '1,' denoting a successful landing.
- 'False' values and 'None' values in 'Mission Outcome,' including 'ASDS,' 'Ocean,' and 'RTLS,' are all set to '0,' indicating a landing failure.

This data wrangling effort is a pivotal step in shaping our dataset for machine learning, enabling us to train our models to classify landing outcomes based on 'Mission Outcome.' Here is a link to my project in Github

EDA with Data Visualization

Summary of Plotted Charts and Rationale:

Flight Number vs. Payload Mass: This scatter plot was used to visualize the relationship between Flight Number and Payload Mass. It allowed us to assess if there were any patterns or trends in the payload mass over time.
Flight Number vs. Launch Site: Another scatter plot was created to examine the connection between Flight Number and Launch Site. This chart provided insights into the distribution of launches across different sites.
Payload Mass vs. Launch Site: A bar plot was employed to compare Payload Mass across different Launch Sites. This helped in identifying any site-specific patterns in payload mass.
Orbit vs. Success Rate: A bar plot was used to visualize the relationship between the Orbit and the Success Rate. This chart provided an overview of the success rates for different orbits.
Flight Number vs. Orbit: A line chart was employed to track the changes in the Orbit type over time, based on Flight Number. This helped identify any long-term trends in orbit choices.
Payload vs. Orbit: A scatter plot was utilized to understand the relationship between Payload Mass and the type of Orbit. This chart aided in identifying patterns or preferences in payload destinations.
Success Yearly Trend: A line chart was plotted to depict the yearly trend in mission success. This chart offered insights into the overall mission success rate over time.
Link to the project in Github

EDA with SQL

Summary of SQL Queries:

- Data Set Loading: I successfully loaded our dataset into an IBM DB2 Database, ensuring the data was accessible and well-organized for further analysis.
- Launch Site Names: To gain insights into the launch sites, I executed SQL queries to retrieve information about launch site names, allowing us to understand where missions originated.
- Mission Outcomes: Queries were conducted to extract details regarding mission outcomes, enabling a comprehensive view of mission success and failure.
- Customer Payload Sizes: I leveraged SQL queries to collect data on various payload sizes of customers. This information was essential for understanding the diversity in payload requirements.
- Booster Versions: Queries were executed to retrieve details about booster versions used in missions, providing insights into the evolution of technology.
- Landing Outcomes: To assess the success of landings, SQL queries were applied to access information about landing outcomes, offering a comprehensive understanding of the landing phase.

These SQL queries were instrumental in exploring and comprehending different facets of our dataset, enriching our analysis and decision-making for the project.

Link to the project in Github

Build an Interactive Map with Folium

Summary of Folium Map Objects:

- **Launch Site Markers:** I added markers to the map to represent the launch sites. These markers served as visual cues pinpointing the exact locations from which missions originate.
- **Successful and Unsuccessful Landing Markers:** Green markers were used to represent successful landings, while red markers signified unsuccessful ones. This color-coded distinction made it easy to assess landing outcomes.
- **Proximity Circles:** Proximity circles were added to key locations such as Railway, Highway, Coast, and City. These circles illustrated the proximity of launch sites to significant infrastructure and urban centers. The goal was to understand the strategic location of launch sites relative to these features.

Build an Interactive Map with Folium

Explanation for Inclusion:

- **Launch Sites**: Including launch site markers was crucial for providing a visual reference to the points of origin for space missions. It helped in comprehending the geographic distribution of launches.
- **Successful and Unsuccessful Landing Markers**: The differentiation between successful and unsuccessful landing markers offered a clear visual representation of mission outcomes. This visual aid simplified the understanding of where missions excelled and where improvements might be necessary.
- **Proximity Circles**: Proximity circles played a key role in analyzing the strategic placement of launch sites in relation to key locations. By visualizing the distances to railways, highways, coasts, and cities, I gained insights into the rationale behind launch site selection and assessed their practicality.

Creating the Folium map contributed to a comprehensive understanding of the spatial dynamics of launch sites, their proximity to essential infrastructure, and the relative success of missions based on their geographic locations.

Link to the project in Github

Build a Dashboard with Plotly Dash

Summary of Dashboard Elements:

Pie Chart: The dashboard features a pie chart, allowing users to select between two views. It can display the distribution of successful landings across all launch sites or provide a breakdown of individual launch site success rates.

Scatter Plot: The scatter plot is a dynamic component that accepts two inputs. Users can choose to view data for all launch sites or for an individual site. Additionally, they can adjust the payload mass using a slider, which ranges from 0 to 10,000 kg.

Build a Dashboard with Plotly Dash

Explanation for Inclusion:

- **Pie Chart:** The pie chart serves as a dynamic visualization tool that allows users to explore launch site success rates. It offers an overview of how successful landings are distributed across different sites. The option to select individual launch site success rates provides a more granular understanding of site-specific performance.
- **Scatter Plot:** The scatter plot is a versatile element that provides insights into the relationship between multiple variables. By allowing users to choose between all sites and specific sites, it caters to varied analytical needs. The payload mass slider enables users to investigate how success rates vary with payload mass, an important factor in space missions. Furthermore, the inclusion of booster version categories provides a multi-dimensional view of success trends.

These interactive elements within the dashboard empower users to explore and analyze launch site success rates, understand how success varies across launch sites, payload masses, and booster versions, and make data-driven decisions based on their specific analytical objectives. Link to the project in Github

Predictive Analysis (Classification)

1. Data Preparation:

- Split the 'Class' column from the dataset, designating it as the label for classification.
- Features are standardized using the StandardScaler, ensuring uniform scaling for modeling.
- Data is divided into training and testing sets through the train-test split.

2. Model Hyperparameter Optimization:

- Employ GridSearchCV with 10-fold cross-validation to explore and identify the optimal hyperparameters for four classification models:
    - Logistic Regression
    - Support Vector Machine (SVM)
    - Decision Tree
    - K-Nearest Neighbors (KNN)

Predictive Analysis (Classification)

3. Model Scoring and Evaluation:

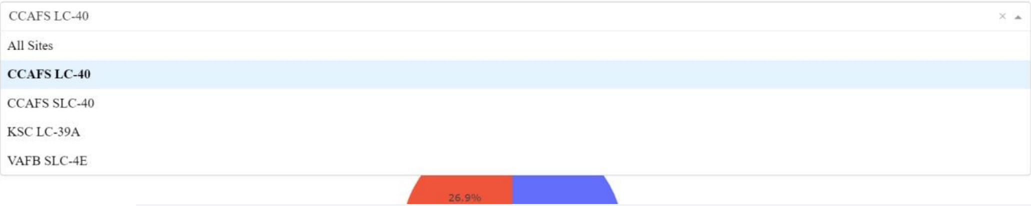
- Evaluate the performance of each model on the test set, calculating accuracy scores.
- Generate confusion matrices to gain insights into each model's classification performance, including true positives, true negatives, false positives, and false negatives.

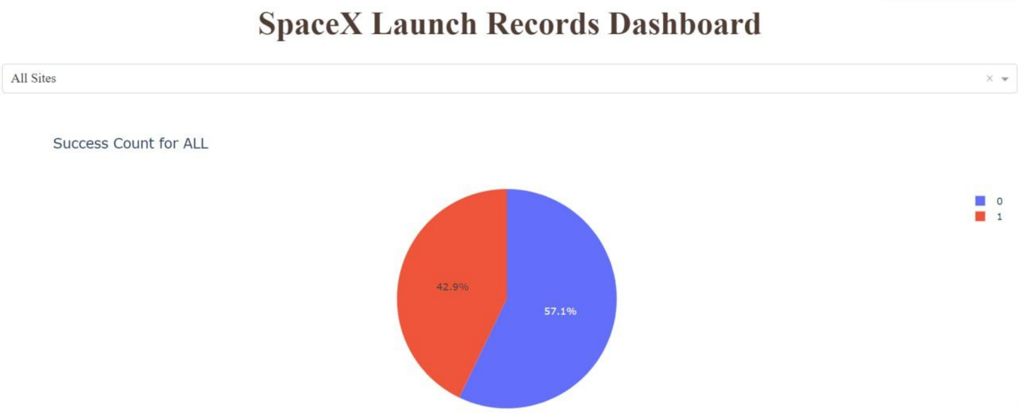
4. Model Comparison:

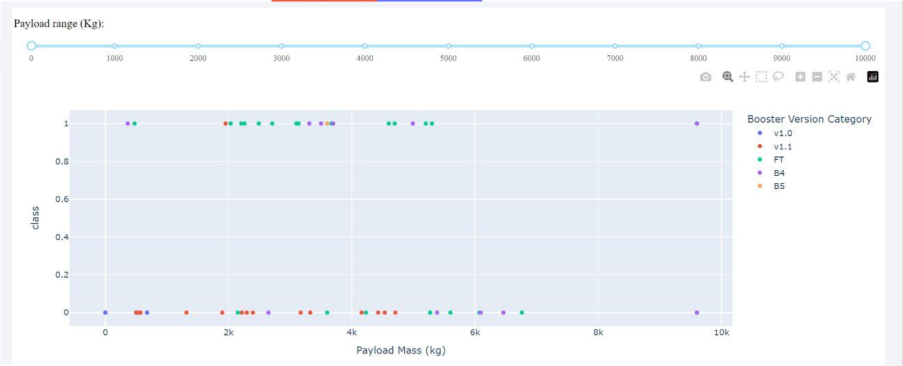
- Create a bar plot to visually compare the accuracy scores of all models. This enables an effective assessment of each model's performance.

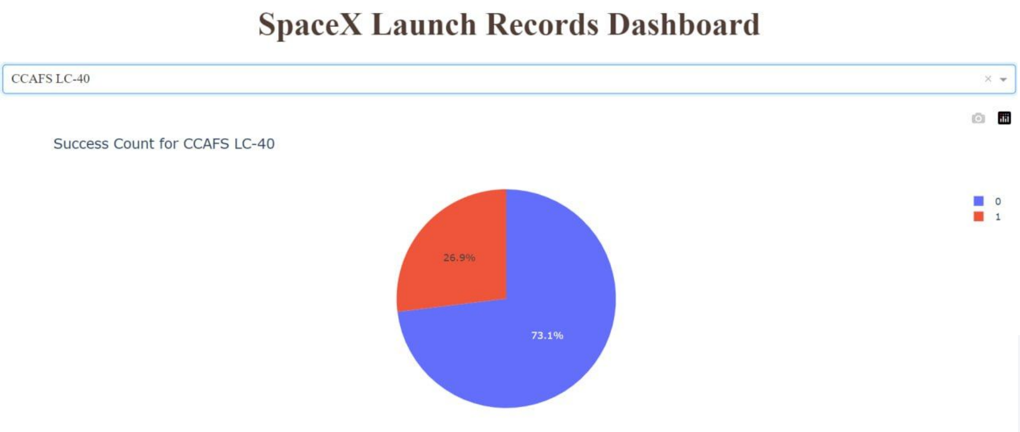
Link to the project in Github

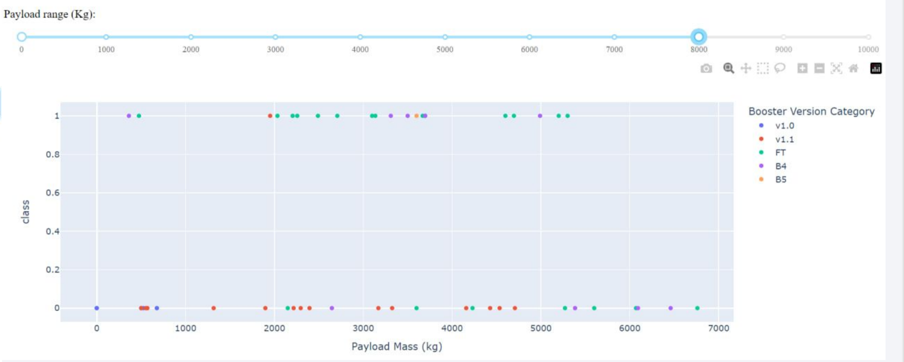
Results











SpaceX Dashboard Preview with Dash

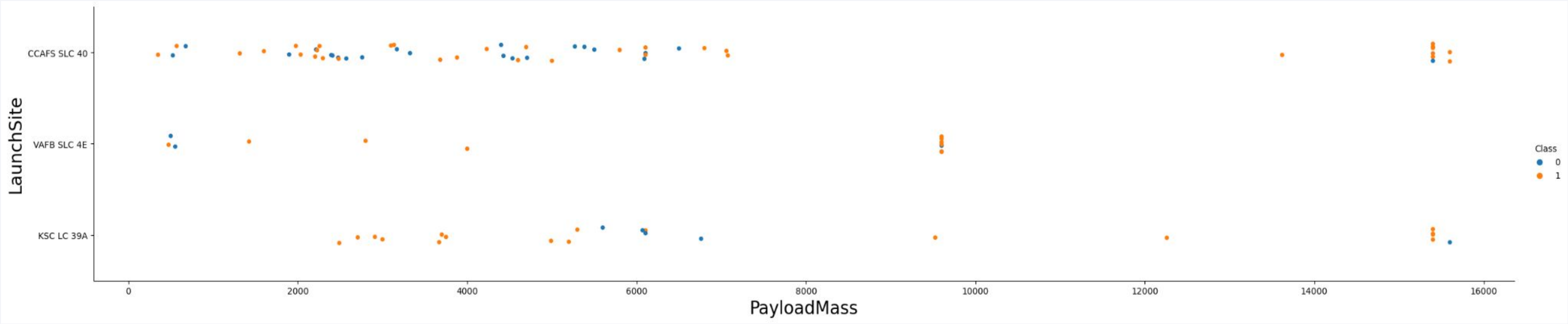
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

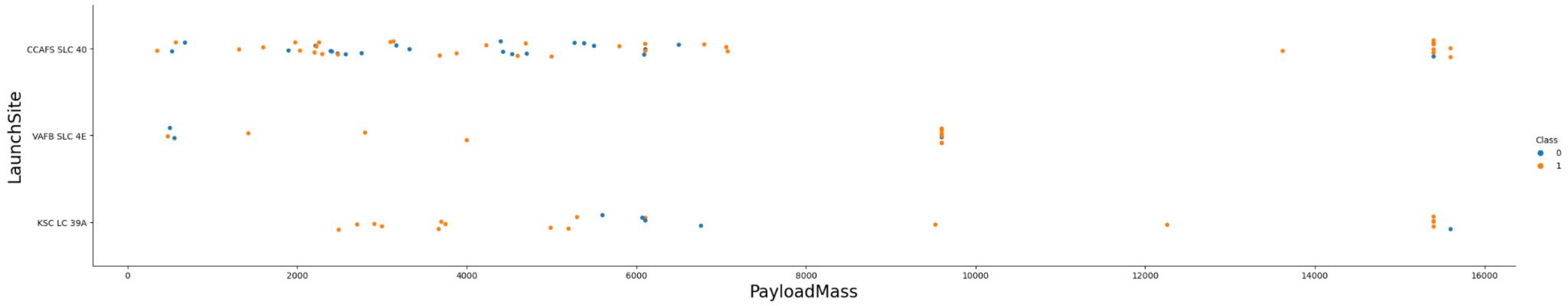
- Scatter plot of Flight Number vs. Launch Site

The graph strongly indicates a prominent upward trend in the success rate over time, as denoted by the Flight Number. Notably, around the 20th flight, there appears to be a pivotal moment that led to a substantial increase in the success rate. Furthermore, the visuals underscore Cape Canaveral Air Force Station (CCAFS) as the primary launch site, given its highest launch volume.



Payload vs. Launch Site

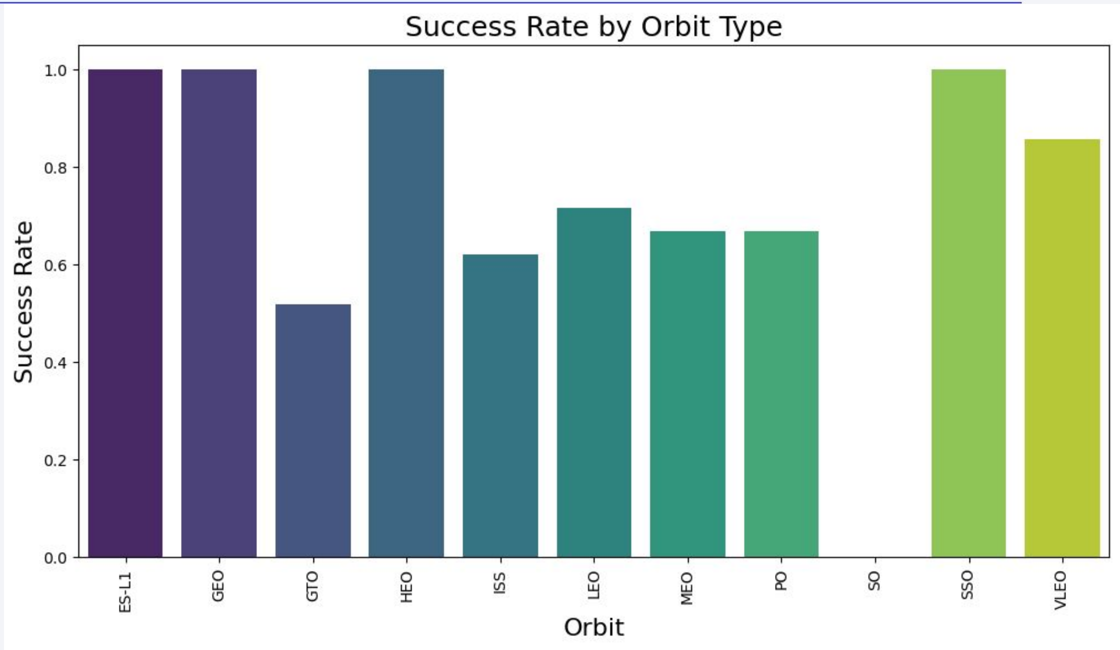
- Payload vs. Launch Site scatter point chart shows that for the VAFB-SLC launchsite there are no rockets launched for heavy payload mass(greater than 10000).



Success Rate vs. Orbit Type

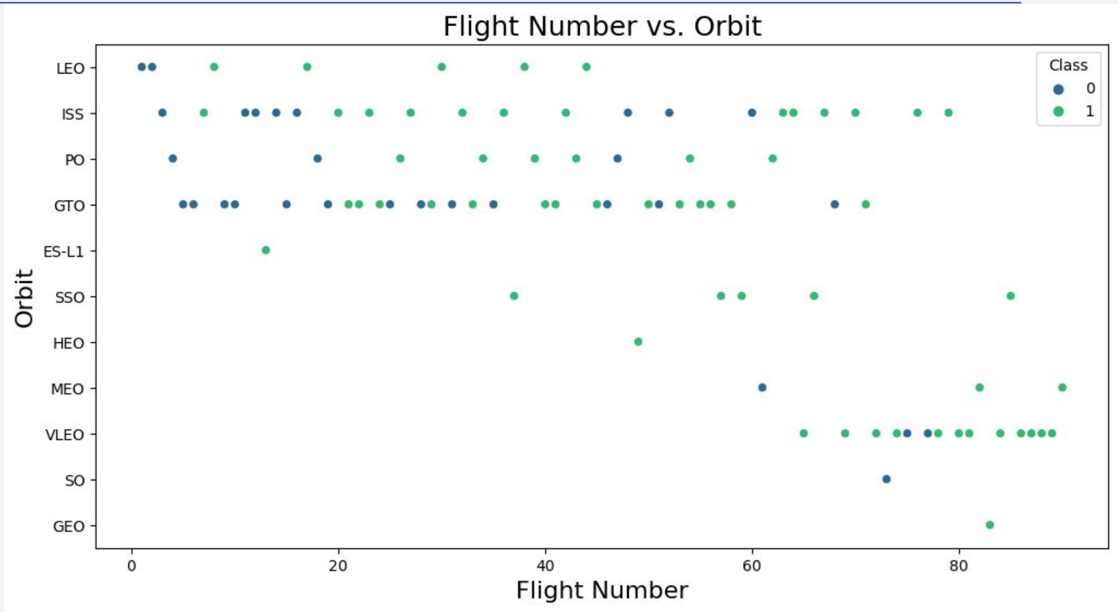
ES-L1 (1), GEO (1), and HEO (1) exhibit a flawless 100% success rate, as indicated by their respective sample sizes in parentheses. Additionally, the SSO category (sample size of 5) maintains a perfect 100% success rate.

Furthermore, VLEO (sample size of 14) demonstrates a commendable success rate. Conversely, the SO category (with a sample size of 1) hasn't achieved any successful landings, while GTO (with a sample size of 27) boasts a success rate hovering around 50%, making it the category with the largest sample size.



Flight Number vs. Orbit Type

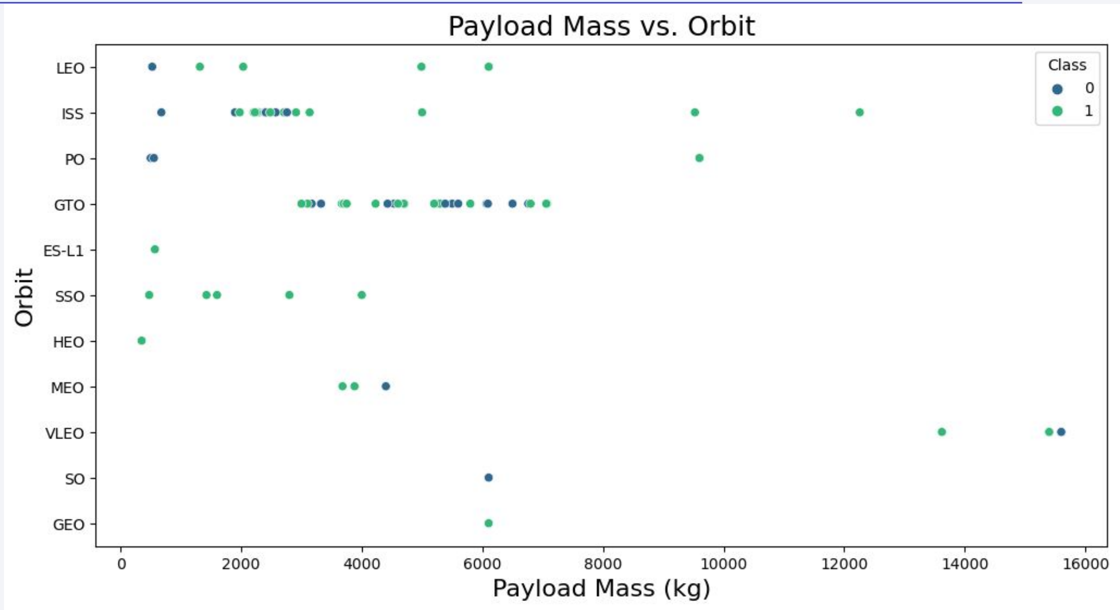
In the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



Payload vs. Orbit Type

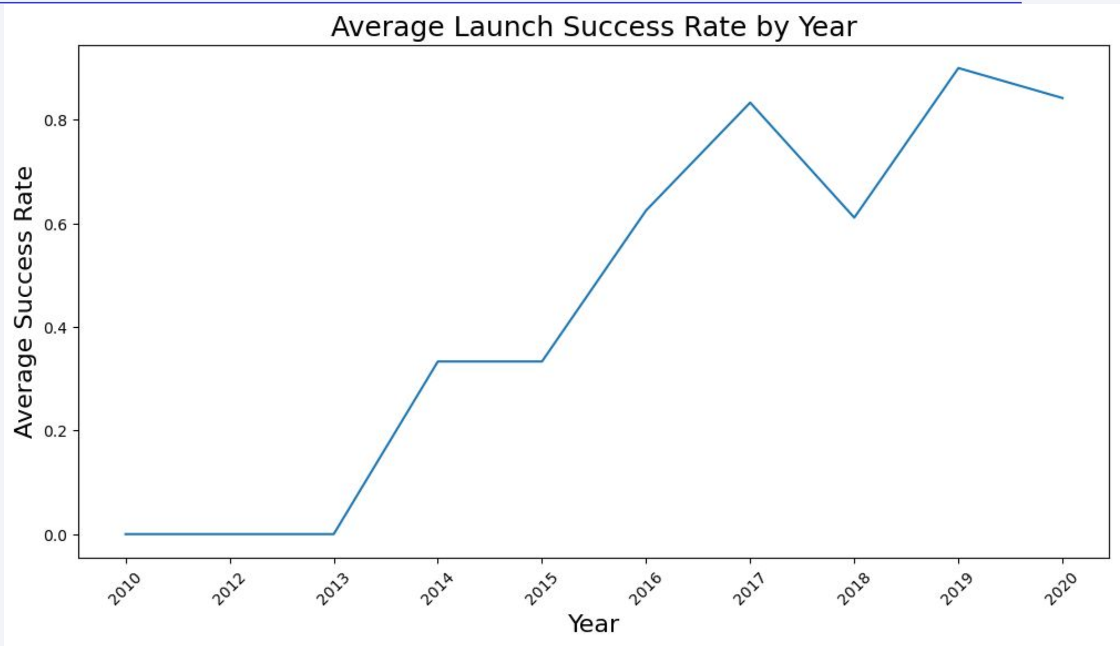
With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.



Launch Success Yearly Trend

The success rate since 2013 kept increasing till 2020



All Launch Site Names

```
In [16]: %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;

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

Out[16]:

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

When querying unique launch site names from the database, it's apparent that there may be data entry errors where "CCAFS SLC-40" and "CCAFS SLC-40" might represent the same launch site. The previous name for this site was "CCAFS LC-40." In essence, there are likely only three unique launch site values to consider: "CCAFS SLC-40," "KSC LC-39A," and "VAFB SLC-4E."

Launch Site Names Begin with 'CCA'

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

In [17]:
```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

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

Out[17]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

LIMIT 5 fetches only 5 records, and the LIKE keyword is used with the wild card ‘CCA%’ to retrieve string values beginning with ‘CCA’.

Total Payload Mass

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

In [18]:
```
%sql SELECT SUM("PAYLOAD_MASS__KG_") AS "Total Payload Mass (kg)" FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)';
```

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

Out[18]:

Total Payload Mass (kg)
45596

The SUM keyword serves to compute the cumulative total of the LAUNCH column. Additionally, the SUM keyword, in conjunction with an associated condition, narrows down the results to include only boosters that are affiliated with NASA's CRS program.

Average Payload Mass by F9 v1.1

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

In [19]:
```
%sql SELECT AVG("PAYLOAD_MASS__KG_") AS "Average Payload Mass (kg)" FROM SPACEXTABLE WHERE "Booster_Version" LIKE 'F9 v1.1%
```

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

Out[19]:

Average Payload Mass (kg)
2534.6666666666665

The AVG keyword is employed to compute the mean of the PAYLOAD_MASS__KG_ column. Simultaneously, the WHERE keyword, along with its associated condition, refines the results to encompass only instances related to the F9 v1.1 booster version.

First Successful Ground Landing Date

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

Hint:Use min function

In [20]:
```
%sql SELECT MIN("Date") AS "First succesful landing outcome in ground pad" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Succe
```

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

Out[20]:

First succesful landing outcome in ground pad
2015-12-22

The MIN keyword is applied to determine the minimum date in the DATE column, representing the earliest date. Simultaneously, the WHERE keyword, in conjunction with its associated condition, refines the results to include only instances of successful ground pad landings.

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 [26]:
```
%sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" > 4(
```

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

Out[26]:

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

The WHERE keyword is utilized to selectively include results that meet both conditions specified within the brackets, considering the simultaneous use of the AND keyword. Moreover, the BETWEEN keyword facilitates the selection of values falling within the range of 4000 < x < 6000.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [38]:
```
%sql SELECT "Mission_Outcome", COUNT(*) AS "Total Count" FROM SPACEXTABLE WHERE "Mission_Outcome" LIKE '%Failure%' OR "Miss:
```

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

Out[38]:

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

The COUNT keyword is employed to determine the overall count of mission outcomes. Additionally, the GROUPBY keyword is utilized to categorize these results based on the type of mission outcome.

Boosters Carried Maximum Payload

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

```
In [44]: %sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE INNER JOIN (SELECT MAX("PAYLOAD_MASS__KG_") AS max_payload FROM SPAC
```

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

Out[44]:

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

The SELECT statement enclosed within the brackets identifies the maximum payload, and this value is subsequently integrated into the WHERE condition. The DISTINCT keyword is then utilized to fetch only unique and distinct booster versions.

2015 Launch Records

In [45]:

```
%%sql

SELECT
    strftime('%m', Date) AS Month,
    CASE
        WHEN Landing_Outcome LIKE 'Failure%' AND Landing_Outcome LIKE '%drone ship%' THEN 'Failure (drone ship)'
        ELSE NULL
    END AS Landing_Outcome,
    Booster_Version,
    Launch_Site
FROM SPACEXTBL
WHERE strftime('%Y', Date) = '2015'
    AND Landing_Outcome LIKE 'Failure%'
    AND Landing_Outcome LIKE '%drone ship%'
ORDER BY Month, Landing_Outcome;
```

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

Out[45]:

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

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

In [46]:
```
%%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;
```

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

Out[46]:

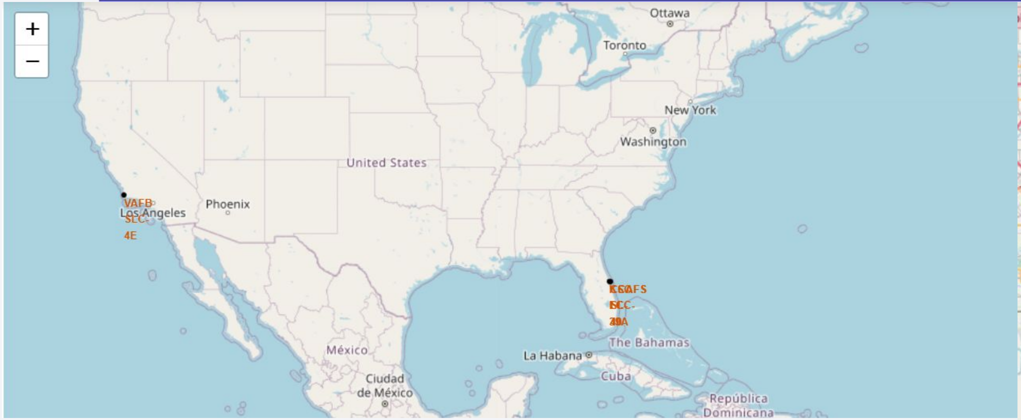
Landing_Outcome	Count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

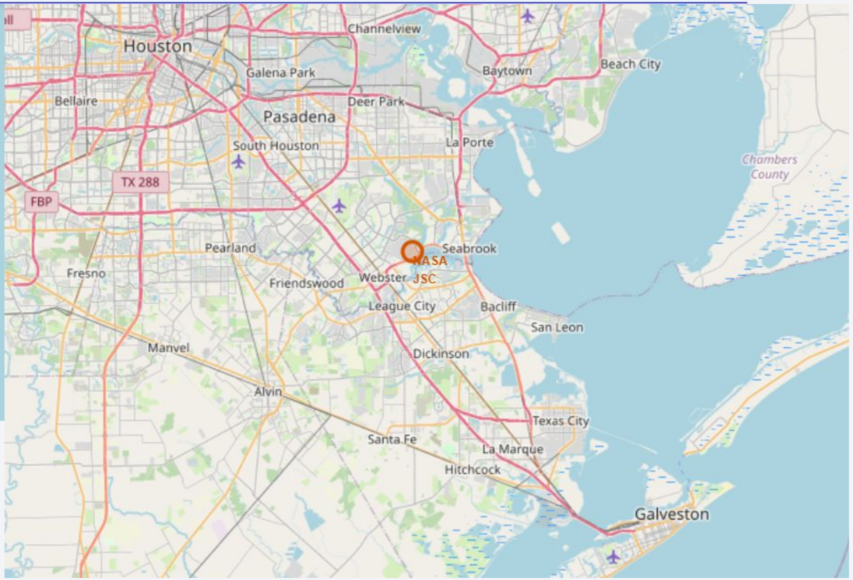
The WHERE keyword is used with the BETWEEN keyword to filter the results to dates only within those specified. The results are then grouped and ordered, using the keywords GROUP BY and ORDER BY, respectively, where DESC is used to specify the descending order.

Section
3

Launch Sites Proximities Analysis

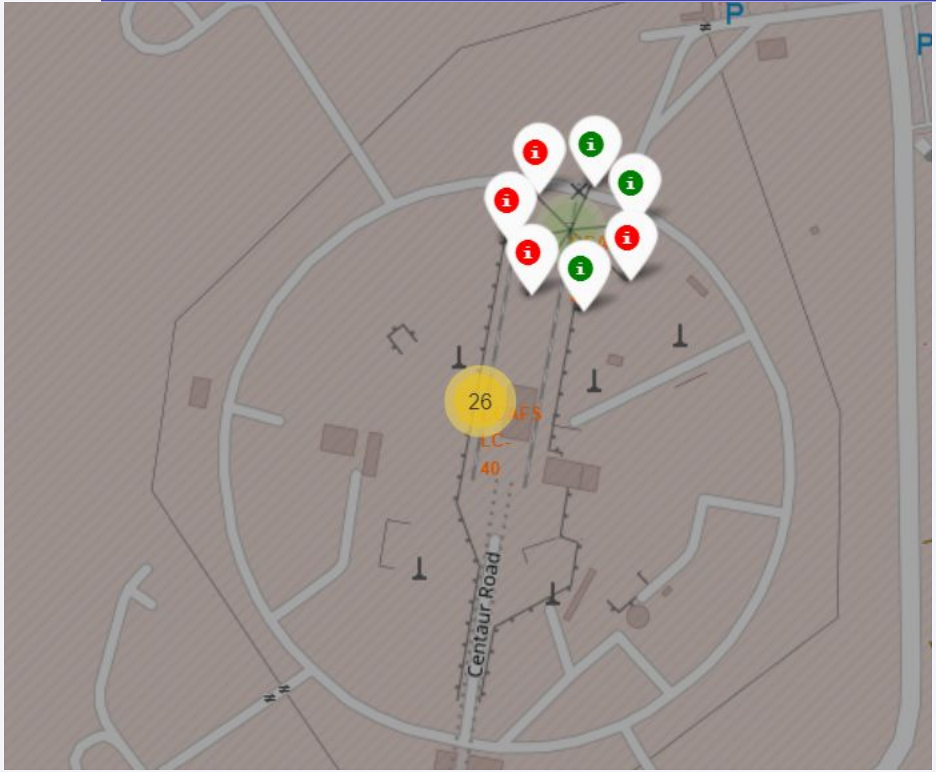
Launch Site Locations





All SpaceX launch sites are located along the coastlines of the United States, specifically in the states of Florida and California.

Success/failed launches for each site on the map



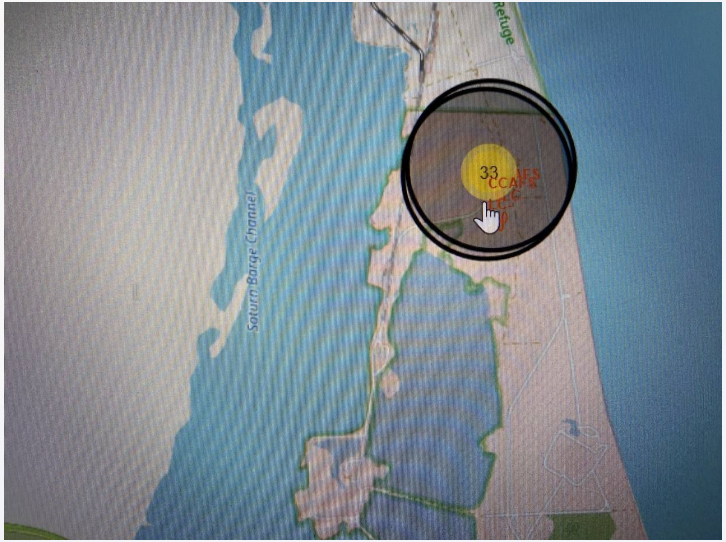
Successful launch, green marker
Failed launch, red marker

MousePosition

`MousePosition` added on the map to get coordinate for a mouse over a point on the map

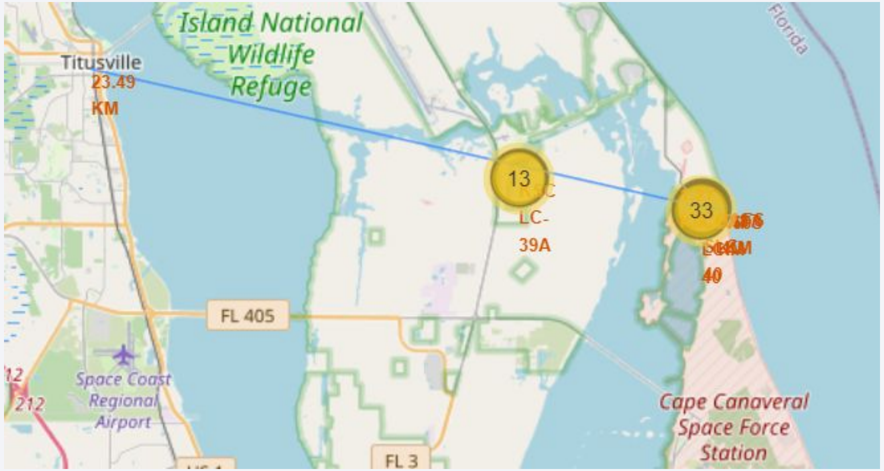
```
# Add Mouse Position to get the coordinate (Lat, Long) for a mouse over on the map
formatter = "function(num) {return L.Util.formatNum(num, 5);};"
mouse_position = MousePosition(
    position='topright',
    separator=' Long: ',
    empty_string='NaN',
    lng_first=False,
    num_digits=20,
    prefix='Lat:',
    lat_formatter=formatter,
    lng_formatter=formatter,
)

site_map.add_child(mouse_position)
site_map
```



The distances between a launch site to its proximities

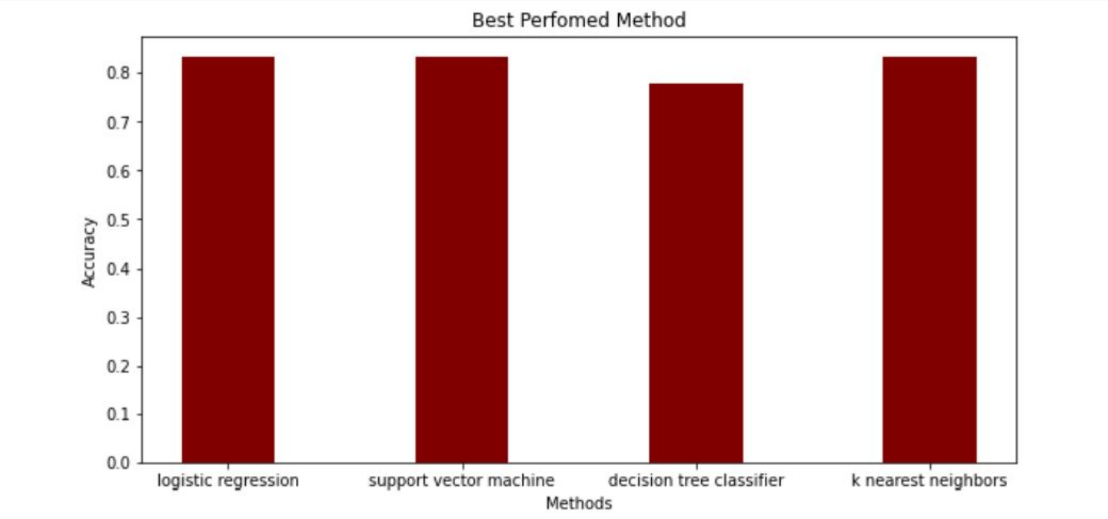
Proximity to see if you can easily find any railway, highway, coastline, etc.



Section
5

Predictive Analysis (Classification)

Classification Accuracy



All models consistently demonstrated nearly identical accuracy levels on the test set, achieving an impressive 83.33% accuracy. It's worth noting that the test set is relatively small, with a sample size of only 18. This limited sample size can lead to considerable variance in accuracy results, as exemplified by the Decision Tree Classifier model, which achieved 66% accuracy.

It's important to recognize that the small sample size may not provide a comprehensive representation of the models' true performance. Therefore, to determine the best model with more confidence, acquiring additional data is likely necessary.

Confusion Matrix

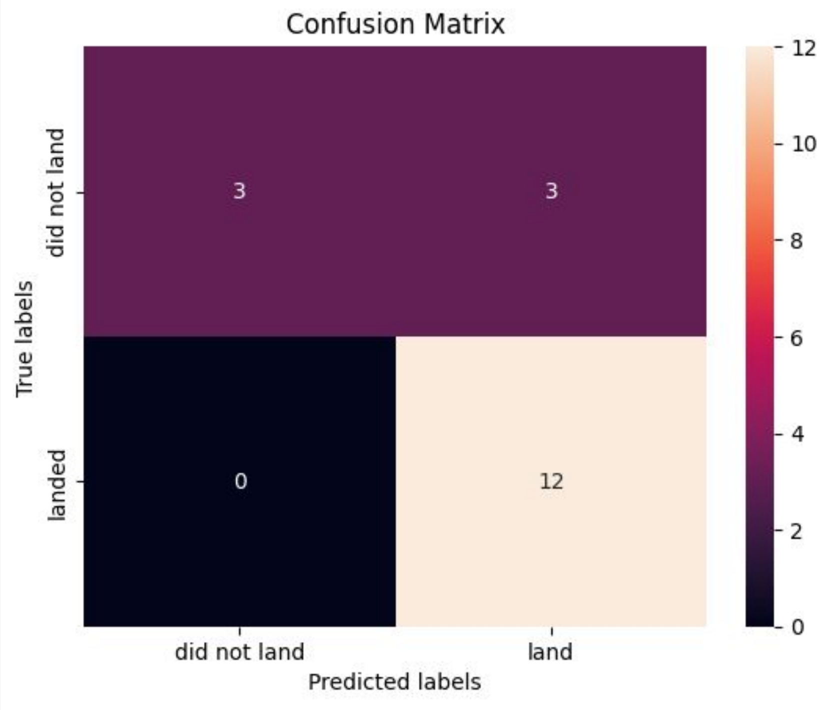
Given that all models, with the exception of the Decision Tree, exhibited identical performance on the test set, the confusion matrix remains consistent across these models. It's important to note that all models made consistent predictions:

- The models accurately predicted 12 successful landings when the true label indicated a successful landing.
- Likewise, the models correctly predicted 3 unsuccessful landings when the true label indicated an unsuccessful landing.

However, there is a noteworthy pattern:

- The models consistently predicted 3 successful landings when the actual label indicated unsuccessful landings, resulting in false positives.

This recurrent pattern across models reveals a tendency to overpredict successful landings, which calls for further analysis and refinement in future iterations.



Conclusions

- Our mission: To equip Space Y with a competitive edge against SpaceX through the development of a machine learning model.
- The model's objective: To accurately predict the successful landing of Stage 1, potentially saving approximately $100 million USD.
- Data sources: I sourced data from a publicly accessible SpaceX API and conducted web scraping on the SpaceX Wikipedia page.
- Data organization: I meticulously engineered data labels and securely stored the dataset within an IBM DB2 SQL database.
- Data visualization: I created an interactive and insightful dashboard for data visualization.
- Machine learning success: The machine learning model I developed achieved an accuracy rate of 83%, setting the stage for precise predictions.
- Quest for improvement: To enhance the model's accuracy and effectiveness, there's a strong recommendation for the collection of additional data.
- Top performer: The Decision Tree model emerged as the best-performing classification model, boasting an impressive accuracy of 94.44%.

This presentation encapsulates my journey in developing a robust machine learning solution for Space Y.

Appendix

- Github link: https://github.com/akerkeabs/IBM_data_science_projects

Thank you!