IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Franz Monzales
07/26/2024

Outline

Executive Summary

Summary of methodologies

Data Collection

- Utilized the SpaceX API to gather comprehensive data on all SpaceX flights..
- Performing a Web Scraping to collect Falcon 9 historical launch from a Wikipedia page Titled “*List of Falcon 9 and Falcon Heavy launches*”
- Collected detailed information on launch dates, rocket types, payloads, mission outcomes, and more.

Data Wrangling

- Perform Data Cleaning for finding and removing duplicated and some irrelevant data points
- Determine Training Labels convert those outcomes into Training Labels with `1` means the booster successfully landed `0` means it was unsuccessful

Executive Summary

Summary of methodologies

Exploratory Data Analysis (EDA) with SQL

- Data Retrieval using Data Query Language (DQL) SELECT Statement to extract relevant data from the Database
- Analyze Columns like Launch_Site, Payload, PAYLOAD_MASS__KG_, Orbit Then Mission and Landing Outcome
- Employed Subqueries for complex calculation and analysis for further insights

Data Visualization

- By using Python as Programing Language for creating Data Visualization needed
- Import Libraries needed like Pandas, Matplotlib and Seaborn
- Visualize the Relationship for some columns like between Flight Number and Launch Site or Payload and Launch Site using Scatter Point Chart

Executive Summary

Summary of methodologies

Interactive Visual Analytics

- Using Folium Library in Python as Map Library to determine all launch sites on a map
- Mark the success/failed launches for each site on the map
- Calculate the distance between a launch site to its important proximities

Predictive Analysis

- Determine the Training Label Dataset created from Data Wrangling to Standardize the data
- Create a Column for the classes using for training data and test data
- Find best Hyperparameter for SYM, Classification Tree and Logistic Regression

Introduction

Project background and context

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch.

Problems you want to find answers

- To predict if the first stage will land given the data from the preceding labs.
- Determine the best method for Machine Learning use in predicting the outcome of Space X rockets

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Combines detailed launch metrics from the SpaceX API with historical content and additional insights obtained though web scraping of Wikipedia to create a comprehensive dataset
- Perform data wrangling
  - The process on Data that making sure to be made ready for EDA, for Visualization and for Modeling in ML we do Data Cleaning, Handling Missing Value, Standardization and Integration

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
  - By Approach in Visualization using Scatter Point for determine relationship among the Columns in the entire Database
- Perform interactive visual analytics using Folium and Plotly Dashboard
  - By using Folium to create interactive maps to visualiza grographical data and launch location
  - In Plotly Dashboard, for develop interactive dashboard for real-time data analysis and visualization
- Perform predictive analysis using classification models
  - Identification of the most accurate classification model to enhance prediction of landing success and inform decision-making by choose between valuate Logistic Regression, SVM, Decision Tree, and K-Nearest Neighbors (KNN) for predictive accuracy.

Data Collection

Describe how data sets were collected.

1. SpaceX API
    - Purpose: Access detailed launch data (example Mission profile, Outcomes, Payloads)
    - Method: By using Request library in Python
2. Wikipedia Web Scraping
    - Purpose Supplement API data with historical context and operational details.
    - Method: Using BeautifulSoup Library in Python to easily get the data needed

Data Collection – SpaceX API

Fetch Data from SpaceX API

- Set GET request to static_json_url
- Load JSON response into DataFrame

Select Relevant Features

- Extract columns: rocket, payloads, launchpad, cores, flight_number, date_utc

Clean Data

- Remove rows with multiple cores or payloads
- Extract single values from lists for cores and payloads

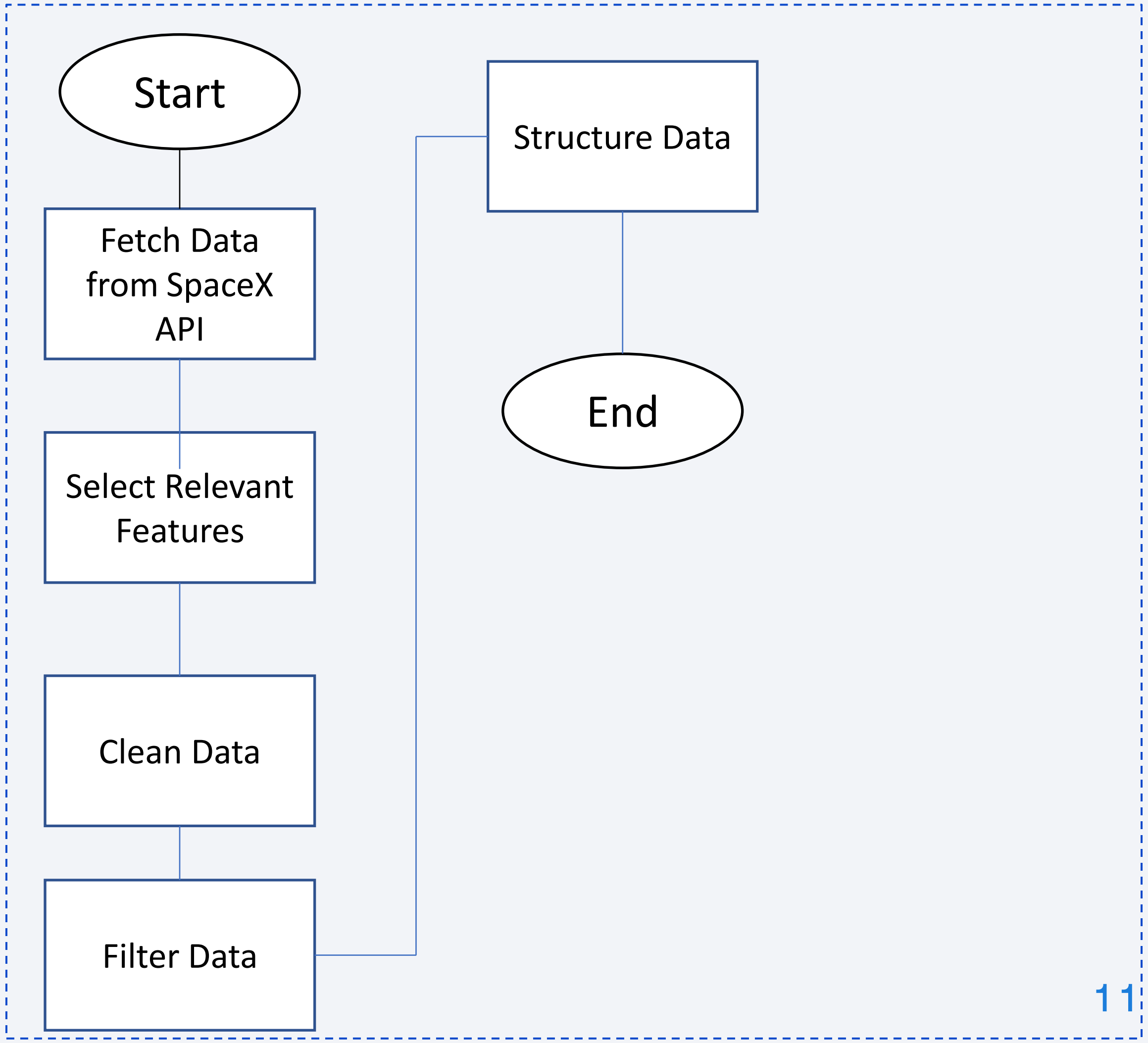
Filter Data

- Restrict dates to ≤ November 13, 2020

Structure Data

- Create a dictionary with key features
- Convert dictionary to DataFrame

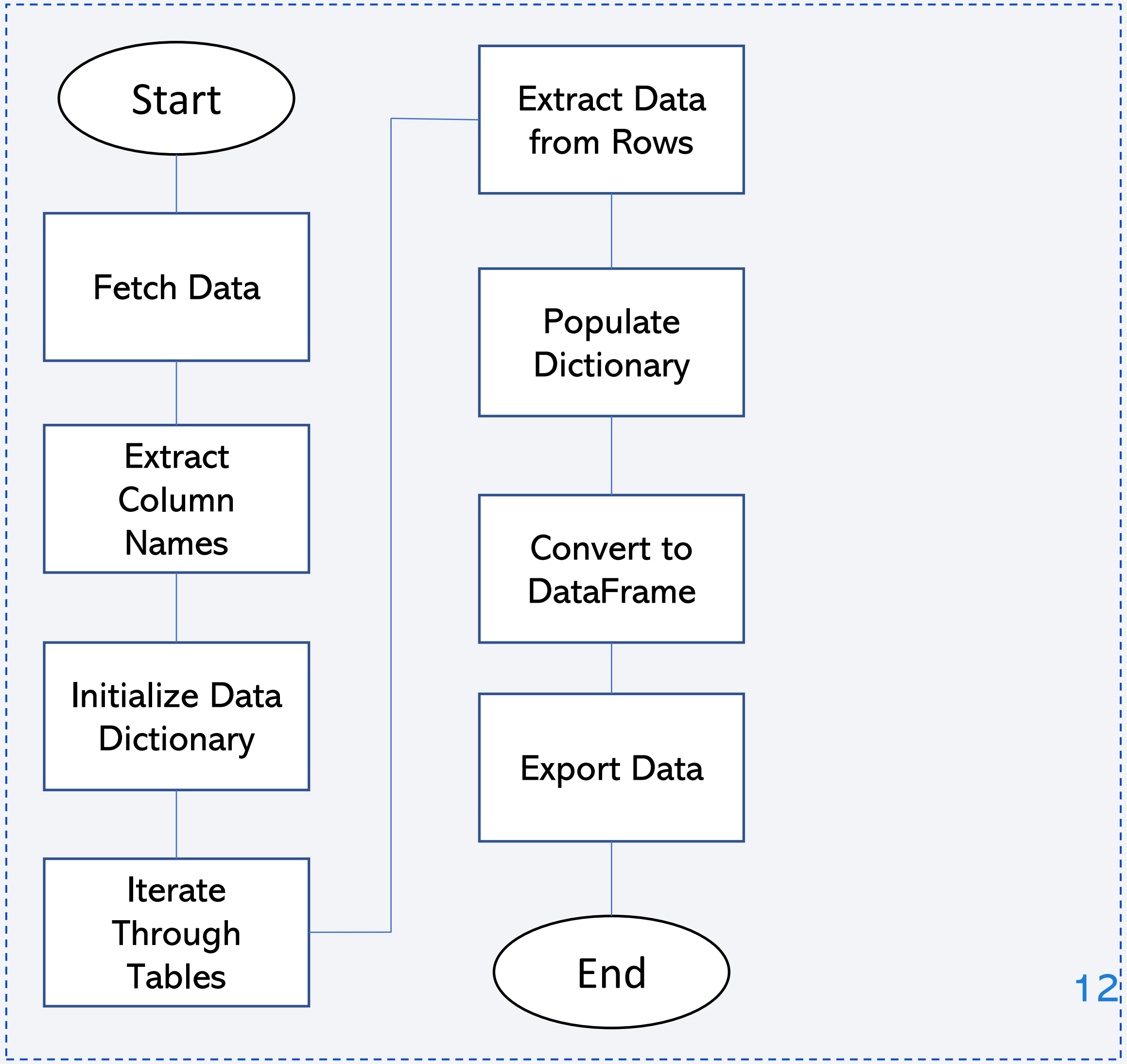
Github URL:https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%201%20(Data%20Gathering%20and%20Wragling)/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- **Fetch Data**: Retrieve HTML content from Wikipedia using requests.
- **Parse HTML**: Use BeautifulSoup to parse the HTML content.
- **Extract Column Names**: Identify and extract column names from table headers.
- **Initialize Data Dictionary**: Create and initialize a dictionary to store launch data.
- **Iterate Through Tables**: Loop through tables to process each row.
- **Extract Data from Rows**: Extract specific details (flight number, date, time, etc.) using defined functions.
- **Populate Dictionary**: Append extracted data to the dictionary.
- **Convert to DataFrame**: Convert to DataFrame
- **Export Data**: Save the DataFrame as a CSV file (spacex_web_scraped.csv).

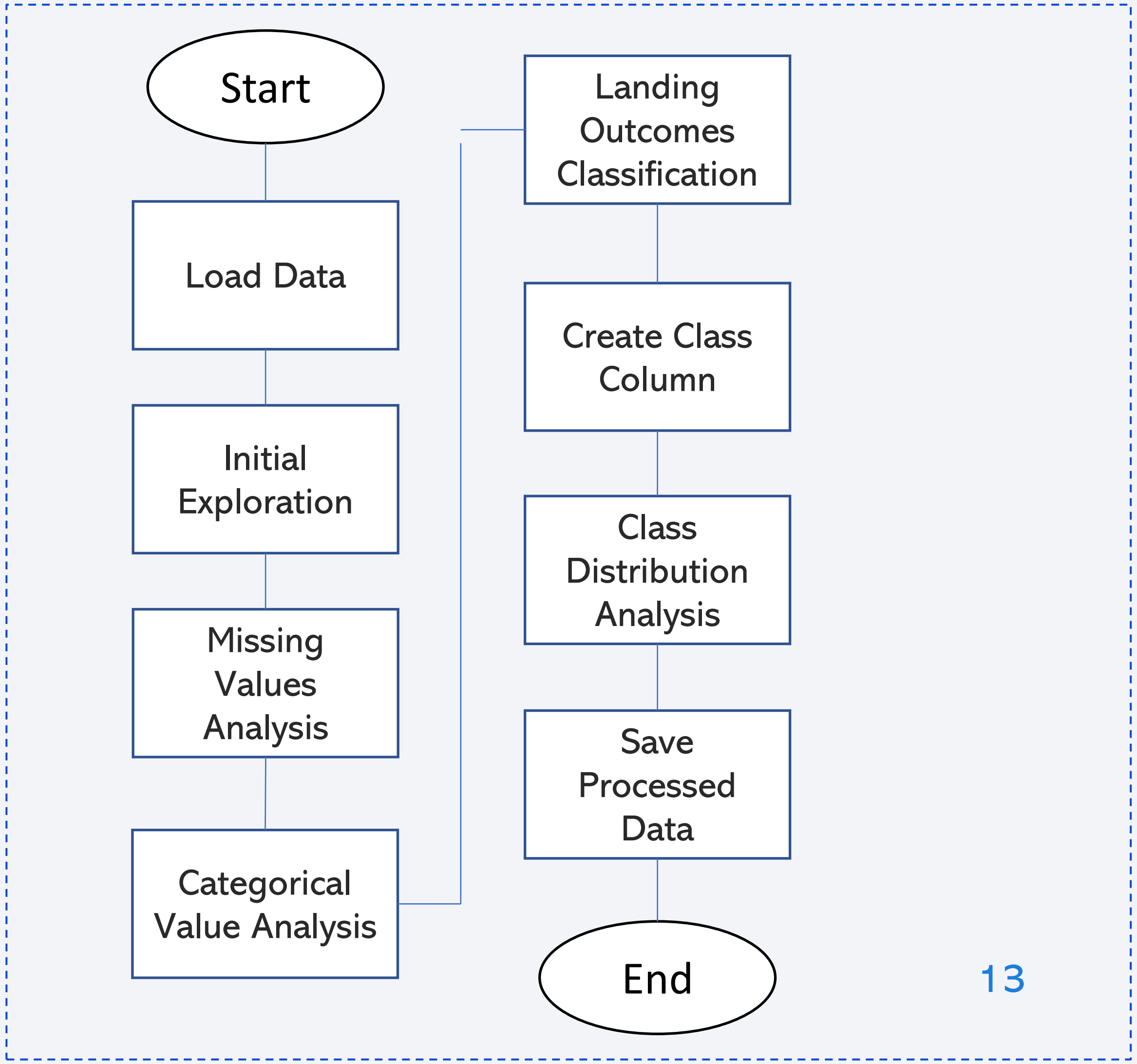
Github URL:https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%201%20(Data%20Gathering%20and%20Wragling)/jupyter-labs-webscraping.ipynb



Data Wrangling

- Load Data: Load dataset from a CSV file into a Pandas DataFrame.
- Initial Exploration: Display the first 10 rows of the dataset.
- Missing Values Analysis: Calculate the percentage of missing values for each column.
- Categorical Value Analysis: Analyze the distribution of values in LaunchSite and Orbit columns.
- Landing Outcomes Classification: Classify landing outcomes into good (1) and bad (0).
- Create Class Column: Add a binary Class column to the DataFrame.
- Class Distribution Analysis: Review the distribution and mean of the Class column.
- Save Processed Data: Export the cleaned and processed DataFrame to a new CSV file

Github URL:https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%201%20(Data%20Gathering%20and%20Wragling)/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

Summary on what Charts were usage and why

- Visualize the Relationship Between Flight Number and Launch Site Using: Catplot for analyze categorical variables and observe the relationship between flight numbers and launch sites.
- Visualize the Relationship Between Payload and Launch Site Using Scatterplot To determine the relationship between payload mass and launch site locations.
- Visualize the Relationship Between Success Rate of Each Orbit Type Using Bar Chart for compare the success rates across different orbit types and identify patterns or trends.
- Visualize the Relationship Between Flight Number and Orbit Type using scatterplot for To observe the distribution and relationship between flight numbers and orbit types.
- Visualize the Launch Success Yearly Trend using Line Chart for analyze the trend of launch successes over the years.
- Feature Engineering: To convert categorical data into a format suitable for machine learning models, enhancing predictive analysis.

Github URL: https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%202%20(Exploratory%20Data%20Analysis)/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

Summary of SQL Queries

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

- Query: SELECT DISTINCT Launch_Site FROM SPACEXTABLE
- Explanation: retrieves a unique list of all launch sites from the SPACEXTABLE. The DISTINCT keyword ensures that only distinct (unique) launch sites are returned, removing any duplicates.

Display 5 records where launch sites begin with the string ‘CCA’

- Query: SELECT * FROM SPACEXTABLE WHERE Launch_Site Like 'CCA%' LIMIT 5
- Explanation: Retrieve all columns from SPACEXTABLE where the Launch_Site starts with 'CCA'. The LIKE 'CCA%' clause filters the results, and LIMIT 5 restricts the output to the first 5 matching records.

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

- Query: SELECT SUM(PAYLOAD_MASS__KG_) as Total_payload_mass_kg FROM SPACEXTABLE WHERE customer = 'NASA (CRS)'
- Explanation: Calculates the total payload mass in kilograms for all launches where the customer is 'NASA (CRS)'. The SUM(PAYLOAD_MASS__KG_) function adds up the payload masses, and the result is labeled as Total_payload_mass_kg..

EDA with SQL

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

- Query: SELECT AVG(PAYLOAD_MASS__KG_) as AVG_PAYLOAD_MASS_KG FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'
- Explanation: Calculates the average payload mass in kilograms for all launches that used the 'F9 v1.1' booster version. The AVG(PAYLOAD_MASS__KG_) function computes the average, and the result is labeled as AVG_PAYLOAD_MASS_KG..

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

- Query: SELECT * FROM SPACEXTABLE ORDER BY Date asc limit 1
- Explanation:. Retrieves the earliest launch record from SPACEXTABLE. The ORDER BY Date ASC clause sorts the records by date in ascending order, and LIMIT 1 ensures only the first (earliest) record is returned.

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

- Query: SELECT * FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ >= 4000 AND PAYLOAD_MASS__KG_ < 6000
- Explanation: Retrieve all columns from SPACEXTABLE where the landing outcome was a successful drone ship landing and the payload mass is between 4000 and 5999 kilograms (inclusive of 4000 but exclusive of 6000).

EDA with SQL

List the total number of successful and failure mission outcomes.

- Query: SELECT Mission_Outcome, count(Mission_Outcome) FROM SPACEXTABLE Group By Mission_Outcome
- Explanation: Counts the number of occurrences of each mission outcome in SPACEXTABLE. The GROUP BY Mission_Outcome clause groups the records by the Mission_Outcome field, and the COUNT(Mission_Outcome) function provides the count for each group..

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

- Query: SELECT * FROM (SELECT Booster_Version, max(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
- Explanation :Retrieve all columns from the result of a subquery that finds the maximum payload mass for each booster version in SPACEXTABLE. The MAX(PAYLOAD_MASS__KG_) function identifies the highest payload mass.

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.

- Query: SELECT substr(Date, 6,2) as Month, * FROM SPACEXTABLE WHERE substr(Date, 0,5) = '2015’
- Explanation: Retrieve all columns from SPACEXTABLE along with the month extracted from the Date column for records from the year 2015. The SUBSTR(Date, 6, 2) function extracts the month part of the date, and SUBSTR(Date, 1, 4) = '2015' filters the results to include only those from the year 2015

EDA with SQL

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..

- Query SELECT * FROM (SELECT * FROM SPACEXTABLE WHERE DATE(date) BETWEEN '2010-06-04' AND '2017-03-20') GROUP BY Landing_Outcome ORDER BY date DESC
- Explanation: Retrieve all records from SPACEXTABLE where the date falls between June 4, 2010, and March 20, 2017. It then groups the results by Landing_Outcome and sorts the grouped results by date in descending order.

Github Link: https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%202%20(Exploratory%20Data%20Analysis)/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

1. Data Preparation

   - Downloaded SpaceX launch data from a CSV file.
   - Filtered and grouped data to obtain launch sites with their coordinates.

2. Map Creation

   - Downloaded SpaceX launch data from a CSV file. Initialized the map centered on NASA Johnson Space Center.
   - Added a marker and circle for NASA Johnson Space Center..

3. Launch Site Visualization

   - Created a base map centered on the average coordinates of all SpaceX launches
   - Used MarkerCluster to group nearby launch events.
   - Added markers for each launch site, color-coded based on launch success (green for successful, red for failed).

4. Additional Features

   - Included MousePosition to display latitude and longitude
   - Calculated and displayed distances between the launch site and various points of interest (e.g., coastline, railroad, highway, city).
   - Added lines to the map representing these distances

Build an Interactive Map with Folium

5. Key Code Snippets

- Data Loading and Preparation.
- Map Initialization and Markers.
- Distance Calculation

6. Result

- An interactive map showcasing SpaceX launch sites, with added layers for distances and clusters, enhancing data visualization and spatial analysis.

Github Link: https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%203%20(Interactive%20Visual%20Analytics%20and%20Dashboards)/lab_jupyter_launch_site_location.ipynb

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
- Explain why you added those objects
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build an Interactive Map with Folium

Objective

Develop an interactive web dashboard using Dash to visualize SpaceX launch records, focusing on launch site statistics and payload analysis.

1. Data Preparation
   - Imported SpaceX launch data into a pandas DataFrame
   - Identified minimum and maximum payload values for scaling in visualizations.
2. Dashboard Components
   - Dropdown List
     - Allows selection of a specific launch site or view data for all sites.
     - Dynamically updates the pie chart and scatter plot based on the selected site
   - Pie Chart
     - For All Sites: Displays the distribution of total successful launches across all sites..
     - For Specific Site: Shows a breakdown of successful vs. failed launches
   - Payload Range Slider
     - Users can filter data by payload mass to examine its impact on launch success.
   - Scatter Chart
     - Visualizes the relationship between payload mass and launch success, with points colored by launch site..

Build an Interactive Map with Folium

3. Key Features
    - Interactive Dropdown: Updates pie chart to reflect launch site selection
    - Dynamic Range Slider: Filters scatter plot based on the payload mass range
    - Real-time Visualization: Refreshes charts based on user input for immediate insights
4. Code Overview
    - Dropdown and Pie Chart Callback

```
@app.callback(Output('success-pie-chart', 'figure'),  Input('site-dropdown', 'value’))
def get_pie_chart(entered_site):
```

    - Payload Slider and Scatter Chart Callback

```
@app.callback(Output('success-payload-scatter-chart', 'figure'), Input('payload-slider', 'value’))
def get_scatter_chart(payload):
```

Github Link : https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%203%20(Interactive%20Visual%20Analytics%20and%20Dashboards)/spacex_dash_app.py

Predictive Analysis (Classification)

Objective

Identify the best-performing classification model for predicting Falcon 9 first stage landing success.

Models Tested:

1. Logistic Regression
2. Support Vector Machine (SVM)
3. Decision Tree
4. K-Nearest Neighbors (KNN)

Process:

- Data Preprocessing: Standardized feature scaling and split into training (80%) and test (20%) sets.
- Hyperparameter Tuning: Used GridSearchCV for model optimization with 10-fold cross-validation.

Predictive Analysis (Classification)

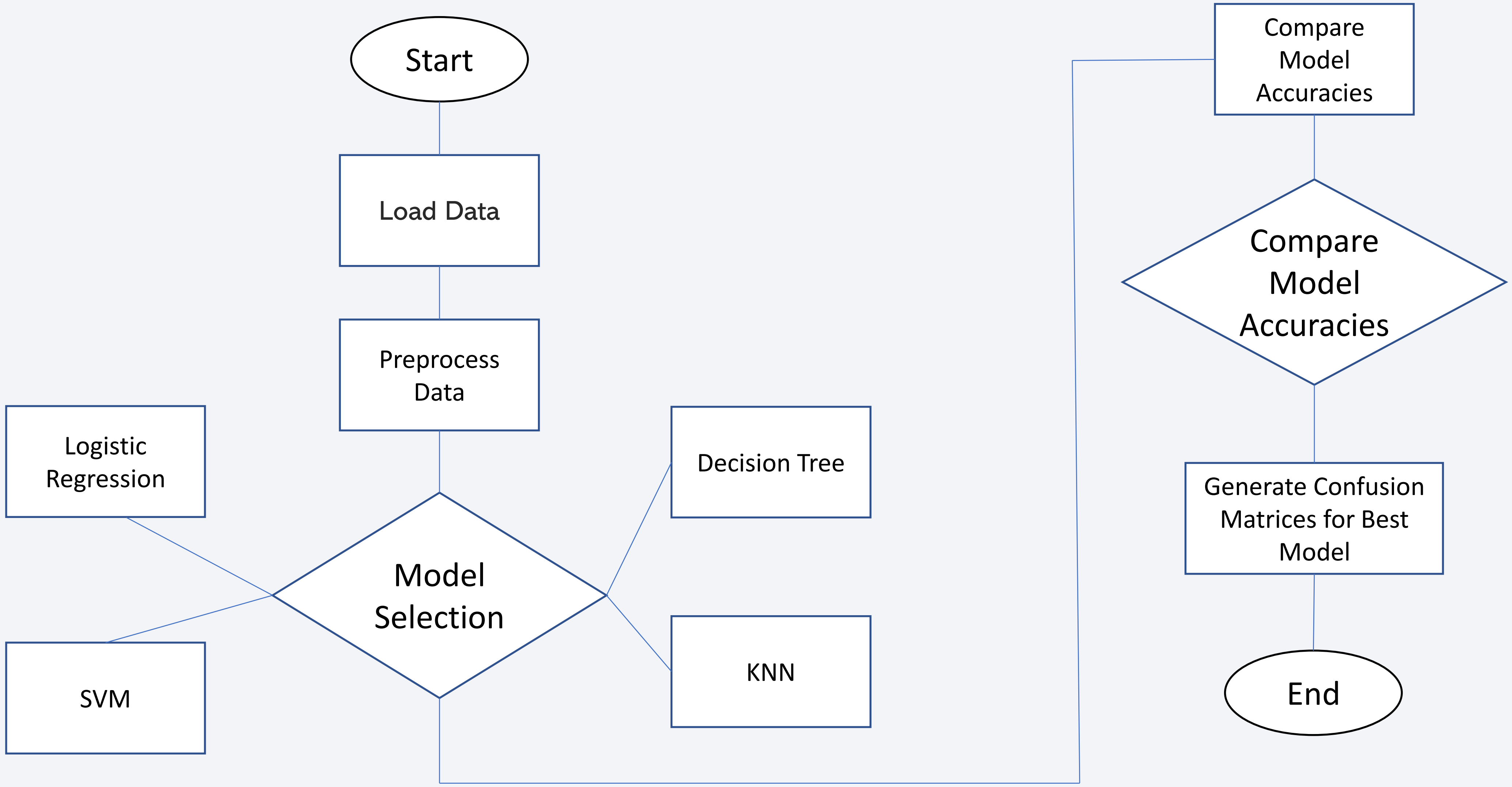
Model Performance

1. Logistic Regression
   - Best Parameters: C=1, penalty='l2', solver='lbfgs’
   - Accuracy: 0.8333333333333334
   - Confusion matrix: we see that logistic regression can distinguish between the different classes. We see that the major problem is false positives
2. SVM
   - Best Parameters:'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid’
   - Accuracy: 0.8482142857142856
3. Decision Tree
   - Best Parameters: 'criterion': 'entropy', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'random’
   - Accuracy:0.8642857142857142
4. KNN
   - Best Parameters {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
   - Accuracy: 0.8333333333333334

Best Performing Model

- Method: Logistic Regression.
- Accuracy: 0.8333333333333334

Predictive Analysis (Classification)



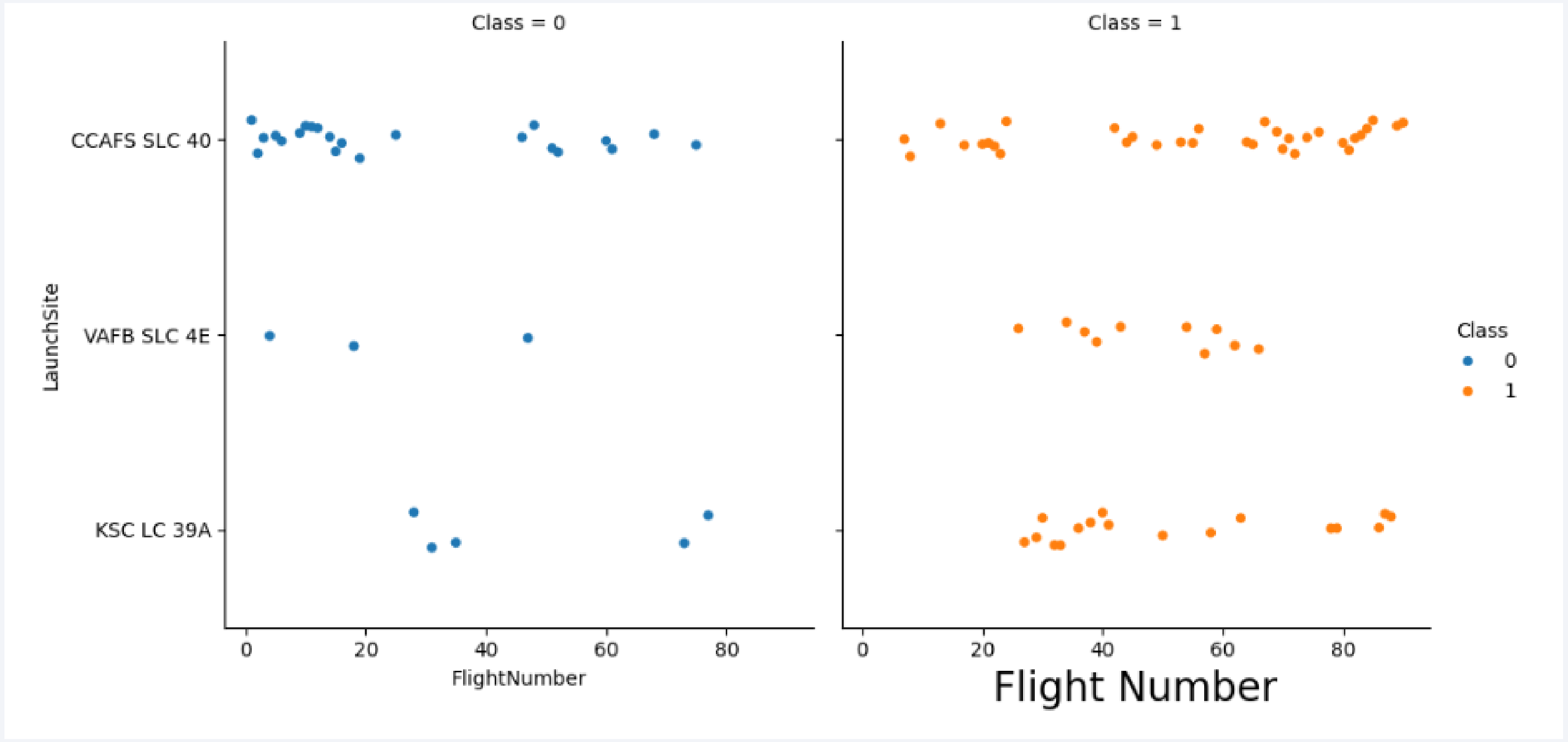
Predictive Analysis (Classification)

Github Link : https://github.com/ikigamisama/IBM-Data-Science-Capstone/blob/master/Module%204%20(Predictive%20Analysis)/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

Exploratory data analysis results

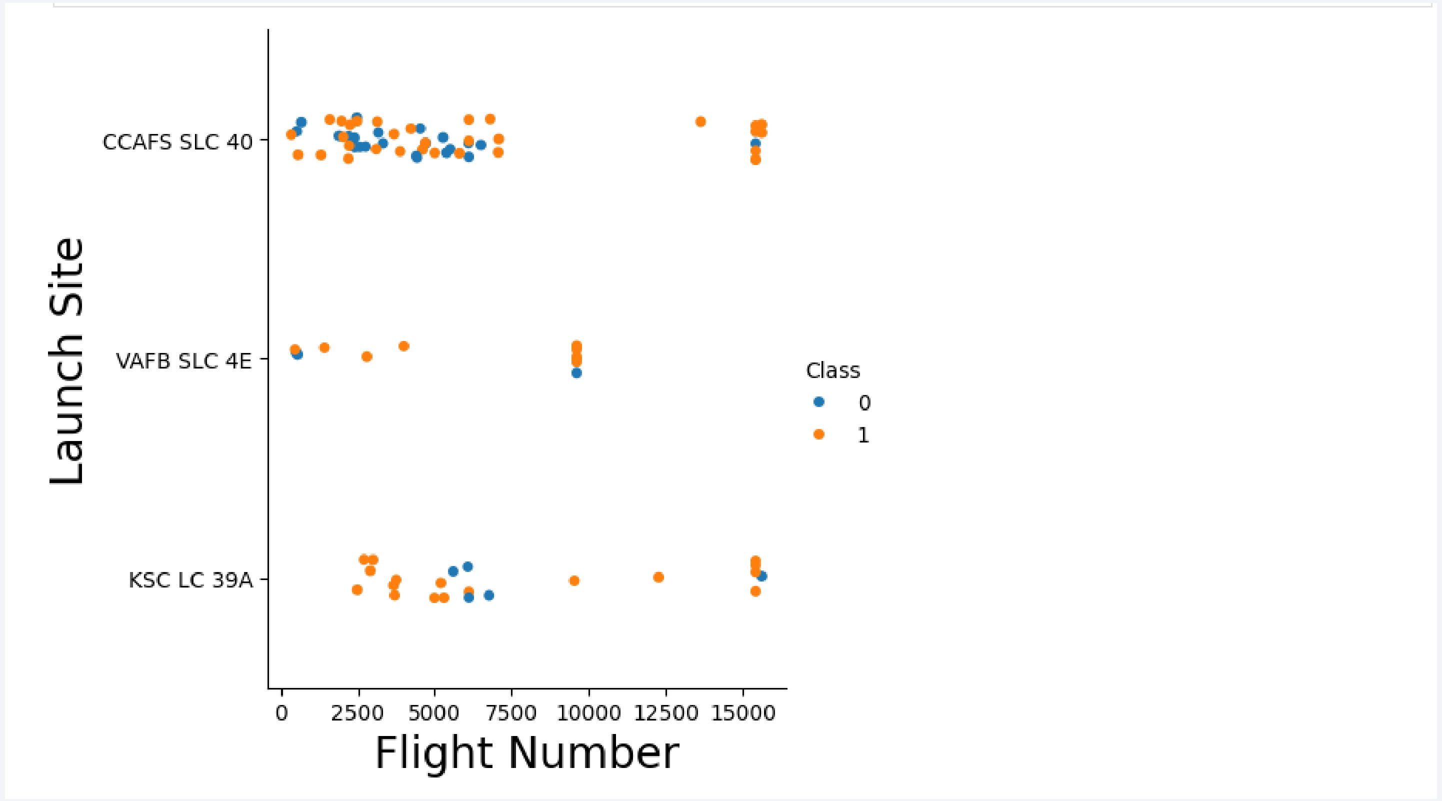
TASK 1: Visualize the relationship between Flight Number and Launch Site



Results

Exploratory data analysis results

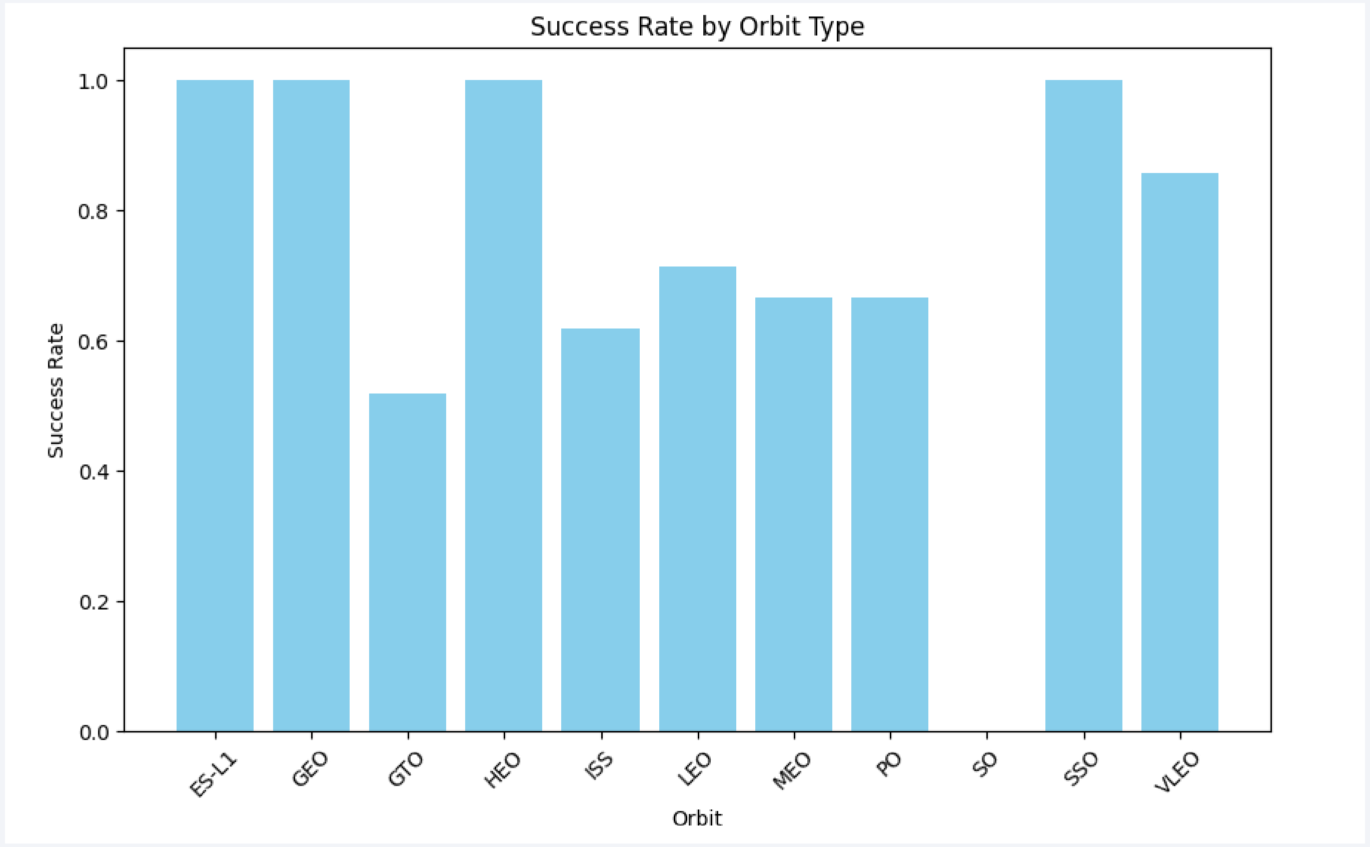
TASK 2: Visualize the relationship between Payload and Launch Sit



Results

Exploratory data analysis results

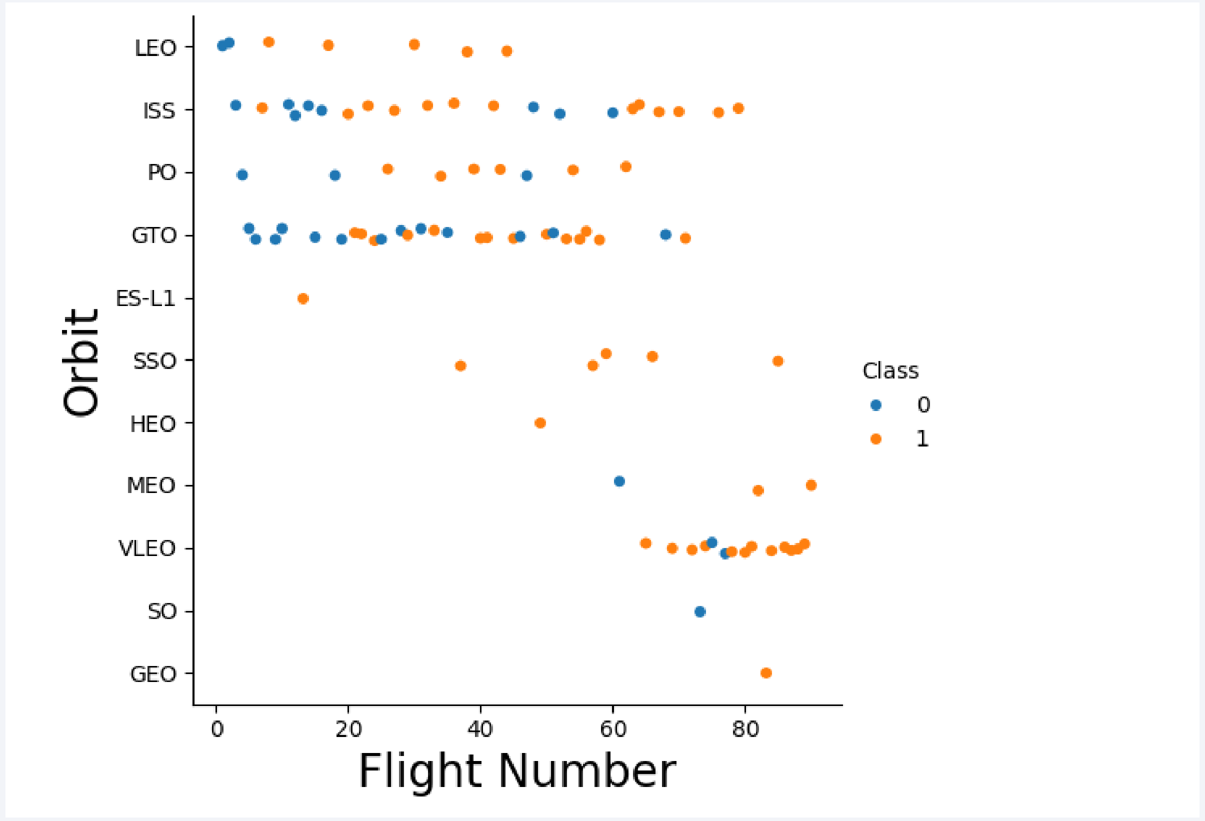
TASK 3: Visualize the relationship between success rate of each orbit type



Results

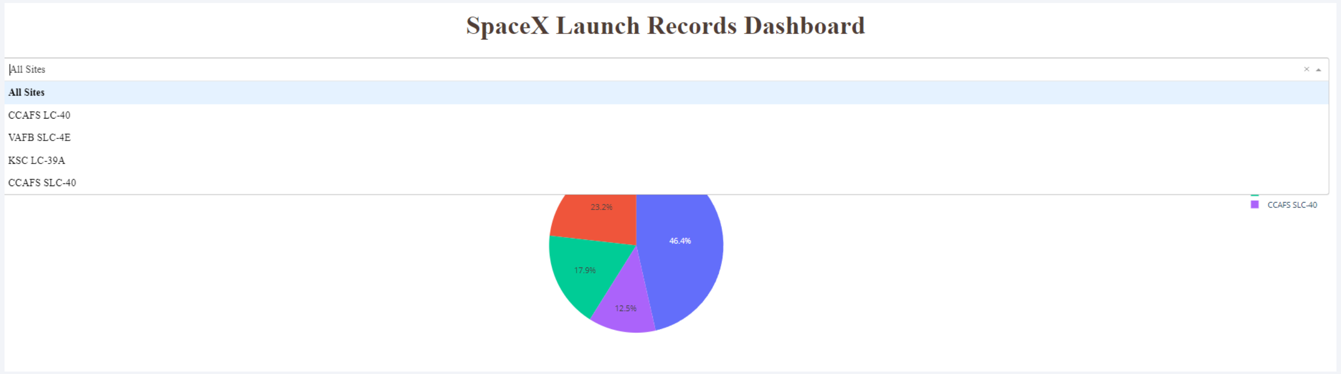
Exploratory data analysis results

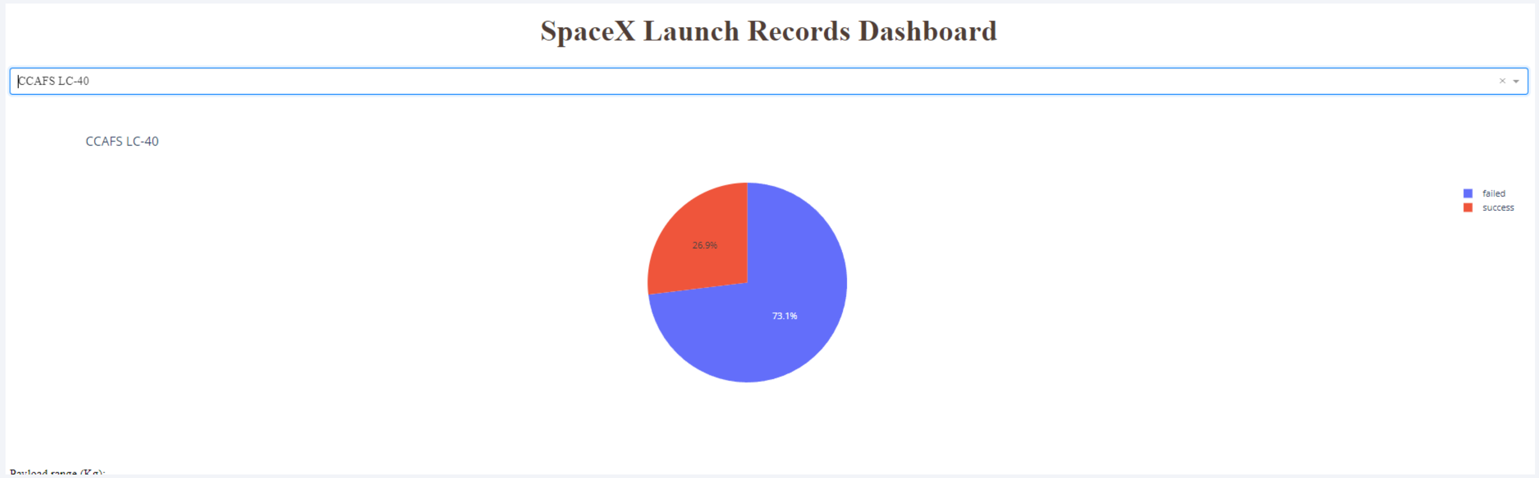
TASK 4: Visualize the relationship between FlightNumber and Orbit type



Results

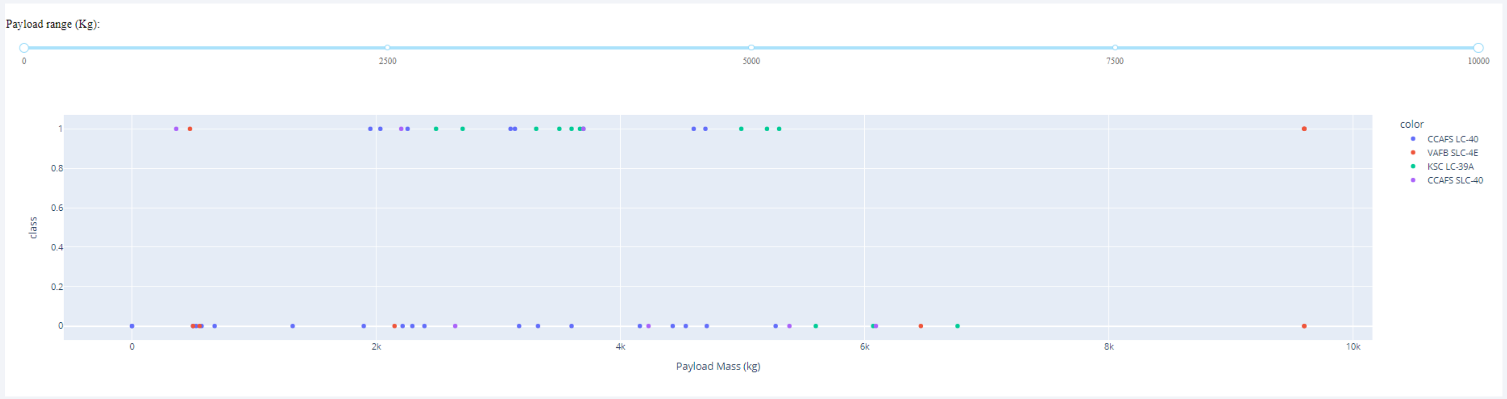
Interactive analytics demo in screenshots

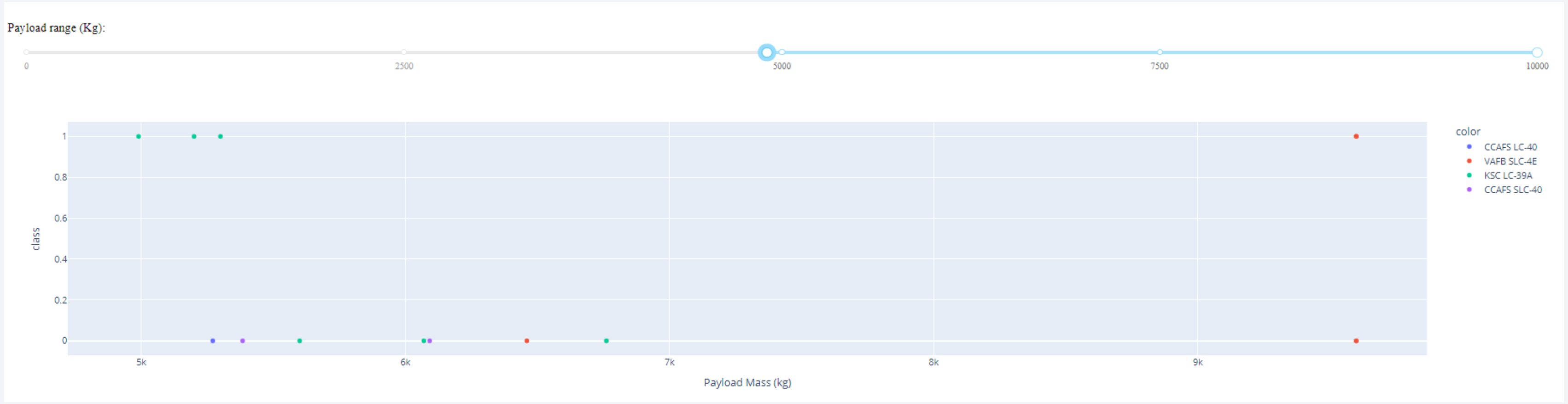




Results

Interactive analytics demo in screenshots

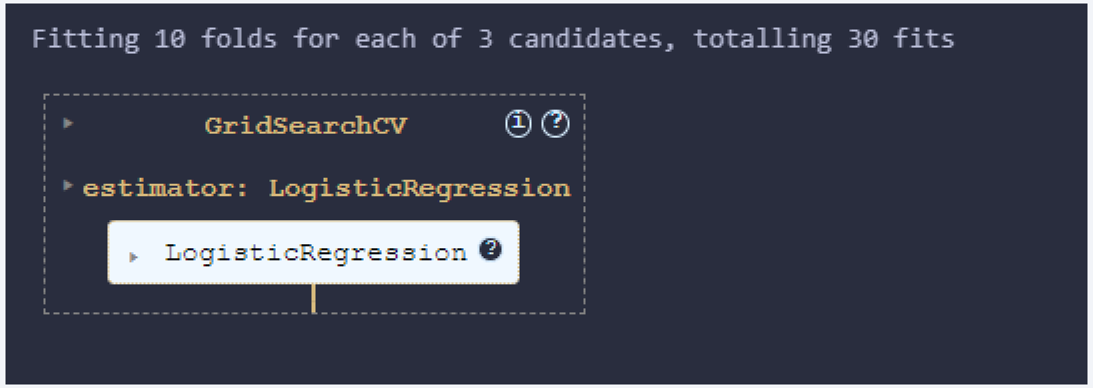




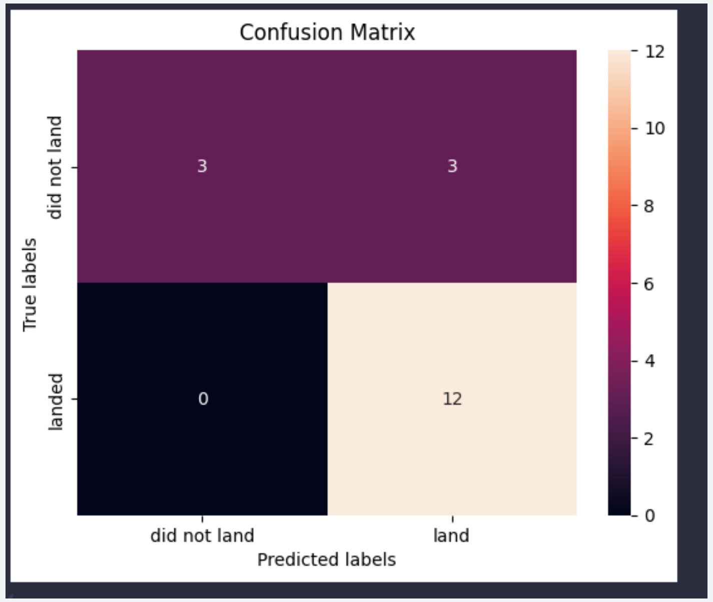
Results

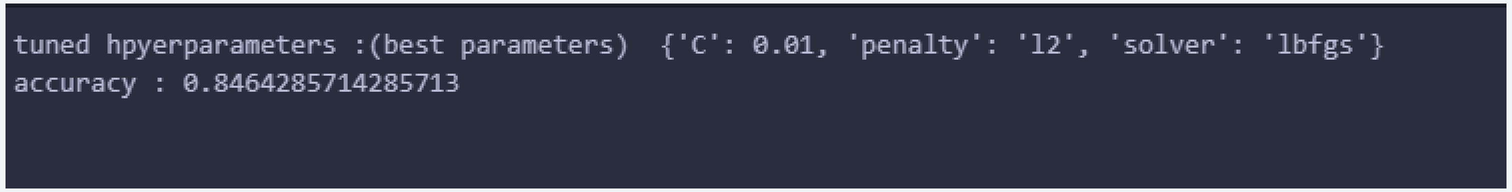
Predictive analysis results

Logistic Regression



```
Fitting 10 folds for each of 3 candidates, totalling 30 fits
```



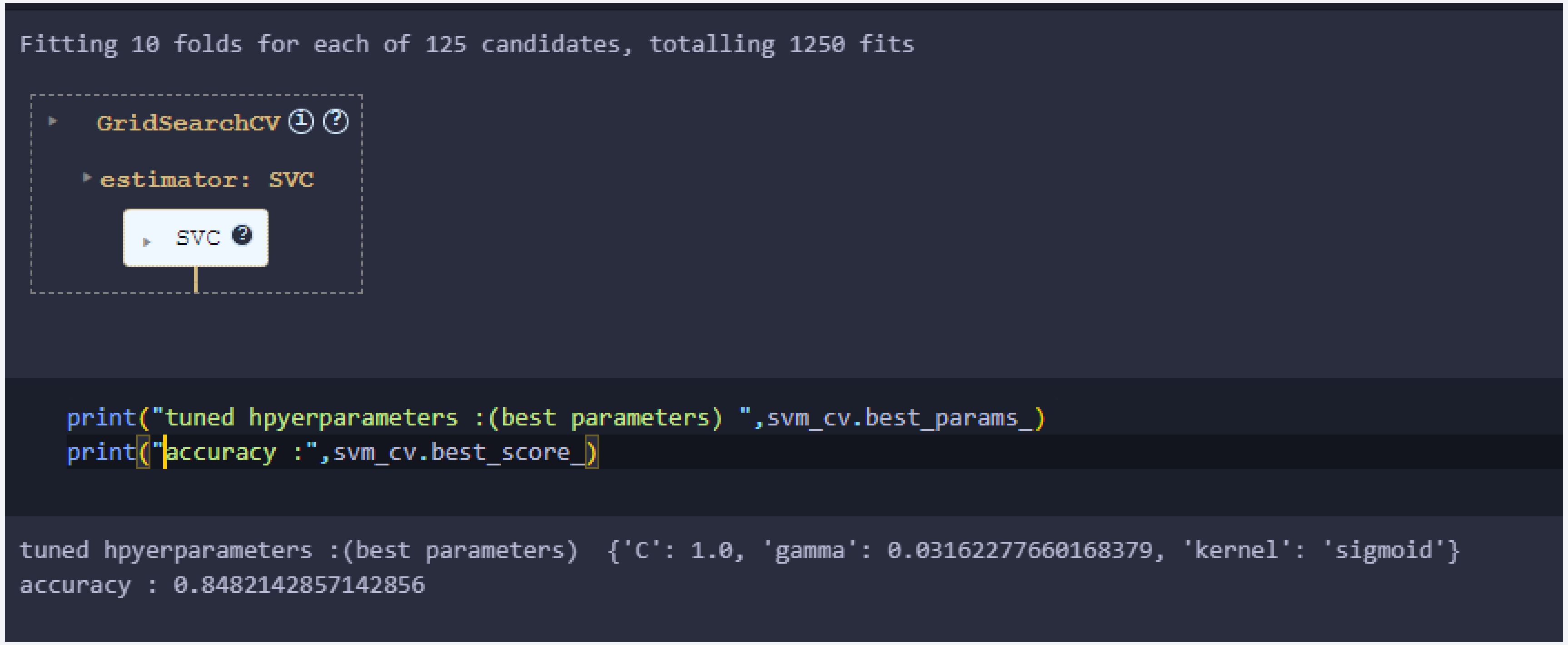


```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

Results

Predictive analysis results

SVC



```
Fitting 10 folds for each of 125 candidates, totalling 1250 fits
```

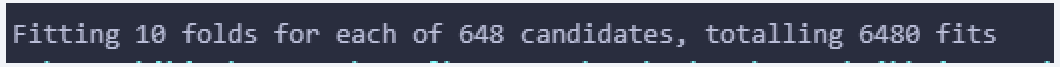
```
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
print("accuracy :",svm_cv.best_score_)
```

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

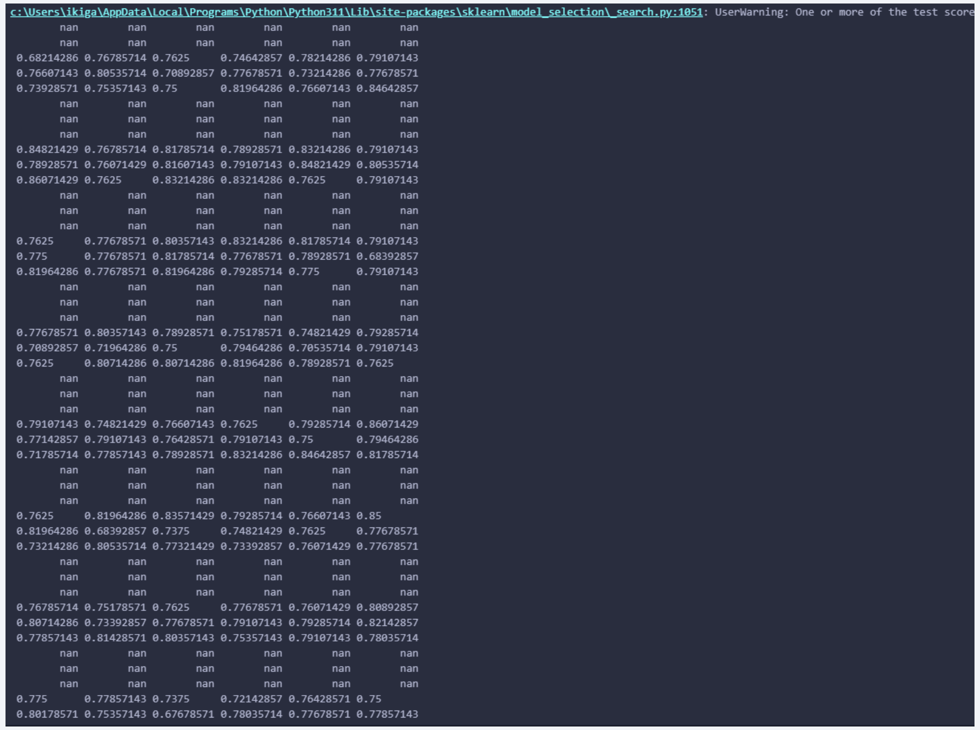
Results

Predictive analysis results

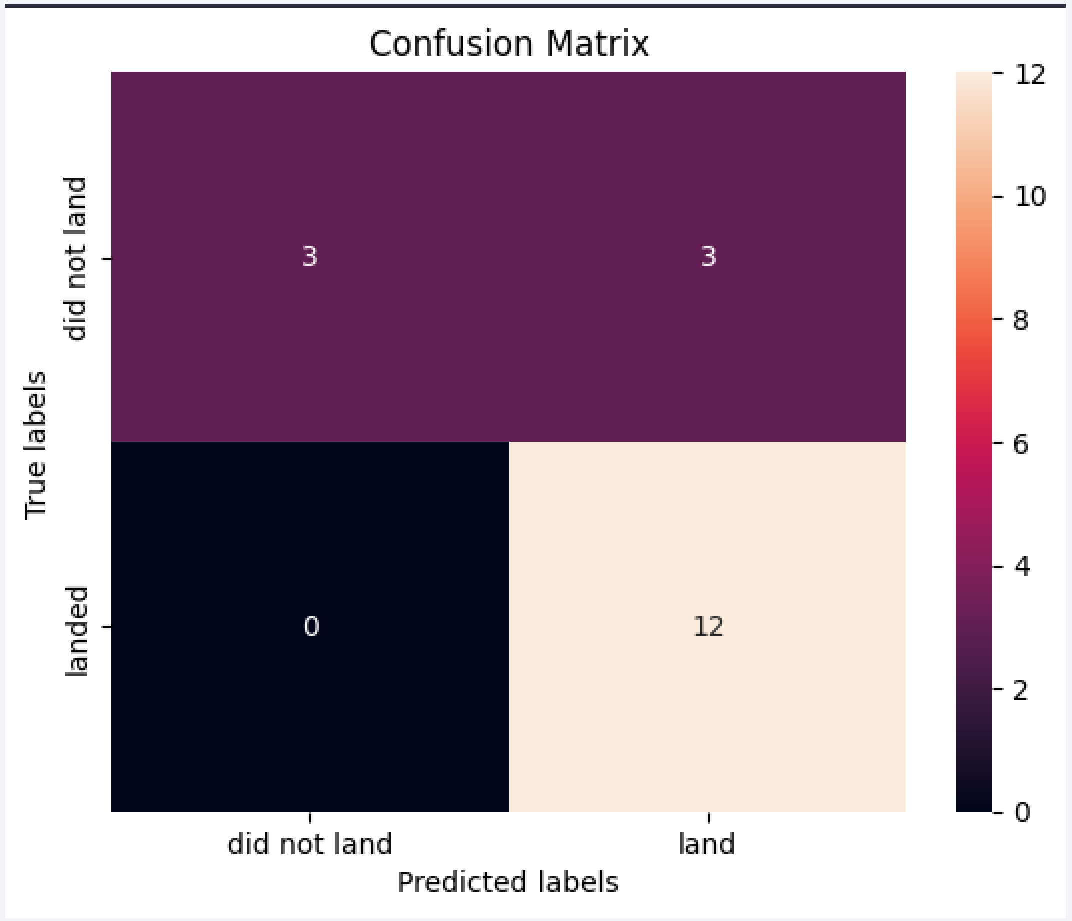
Decision Tree



```
Fitting 10 folds for each of 648 candidates, totalling 6480 fits
```



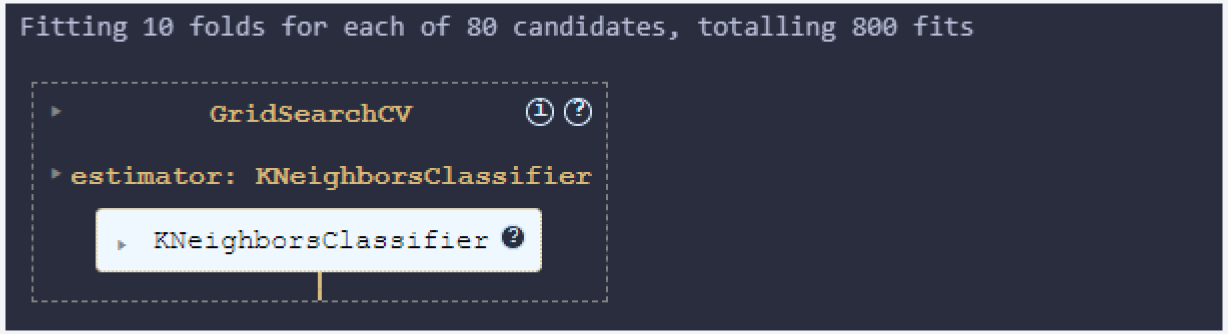
```
c:\Users\ikiga\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\model_selection\_search.py:1051: UserWarning: One or more of the test score
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.68214286 0.76785714 0.7625     0.74642857 0.78214286 0.79107143
 0.76607143 0.80535714 0.70892857 0.77678571 0.73214286 0.77678571
 0.73928571 0.75357143 0.75       0.81964286 0.76607143 0.84642857
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.84821429 0.76785714 0.81785714 0.78928571 0.83214286 0.79107143
 0.78928571 0.76071429 0.81607143 0.79107143 0.84821429 0.80535714
 0.86071429 0.7625     0.83214286 0.83214286 0.7625     0.79107143
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.7625     0.77678571 0.80357143 0.83214286 0.81785714 0.79107143
 0.775      0.77678571 0.81785714 0.77678571 0.78928571 0.68392857
 0.81964286 0.77678571 0.81964286 0.79285714 0.775      0.79107143
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.77678571 0.80357143 0.78928571 0.75178571 0.74821429 0.79285714
 0.70892857 0.71964286 0.75       0.79464286 0.70535714 0.79107143
 0.7625     0.80714286 0.80714286 0.81964286 0.78928571 0.7625
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.79107143 0.74821429 0.76607143 0.7625     0.79285714 0.86071429
 0.77142857 0.79107143 0.76428571 0.79107143 0.75       0.79464286
 0.71785714 0.77857143 0.78928571 0.83214286 0.84642857 0.81785714
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.7625     0.81964286 0.83571429 0.79285714 0.76607143 0.85
 0.81964286 0.68392857 0.7375     0.74821429 0.7625     0.77678571
 0.73214286 0.80535714 0.77321429 0.73392857 0.76071429 0.77678571
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.76785714 0.75178571 0.7625     0.77678571 0.76071429 0.80892857
 0.80714286 0.73392857 0.77678571 0.79107143 0.79285714 0.82142857
 0.77857143 0.81428571 0.80357143 0.75357143 0.79107143 0.78035714
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.775      0.77857143 0.7375     0.72142857 0.76428571 0.75
 0.80178571 0.75357143 0.67678571 0.78035714 0.77678571 0.77857143
```



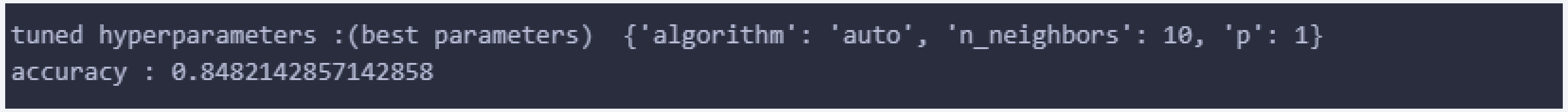
Results

Predictive analysis results

K-Neighbors



```
Fitting 10 folds for each of 80 candidates, totalling 800 fits
```

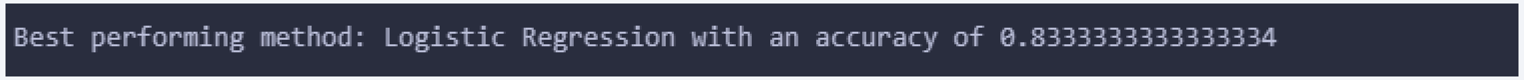


```
tuned hyperparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

Results

Predictive analysis results

Finding the method performs best:



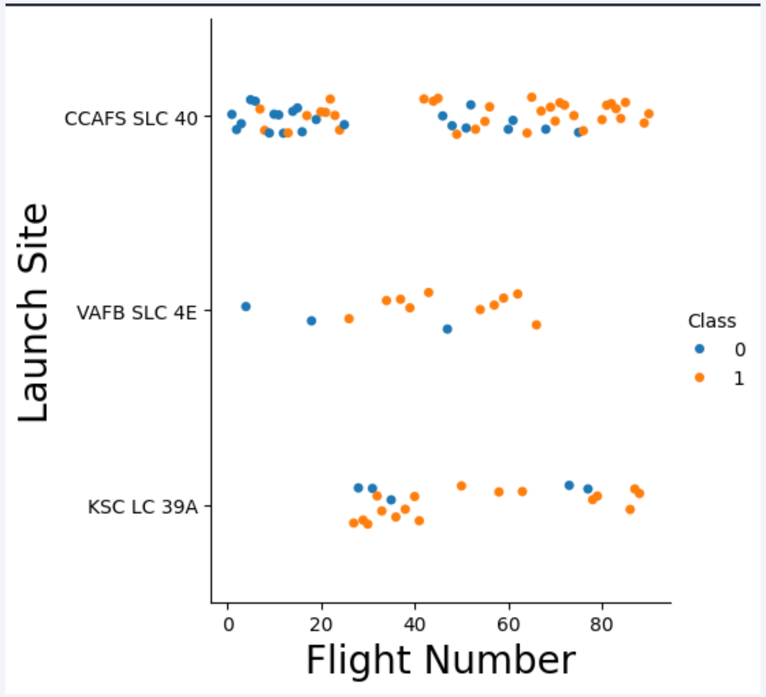
```
Best performing method: Logistic Regression with an accuracy of 0.8333333333333334
```

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

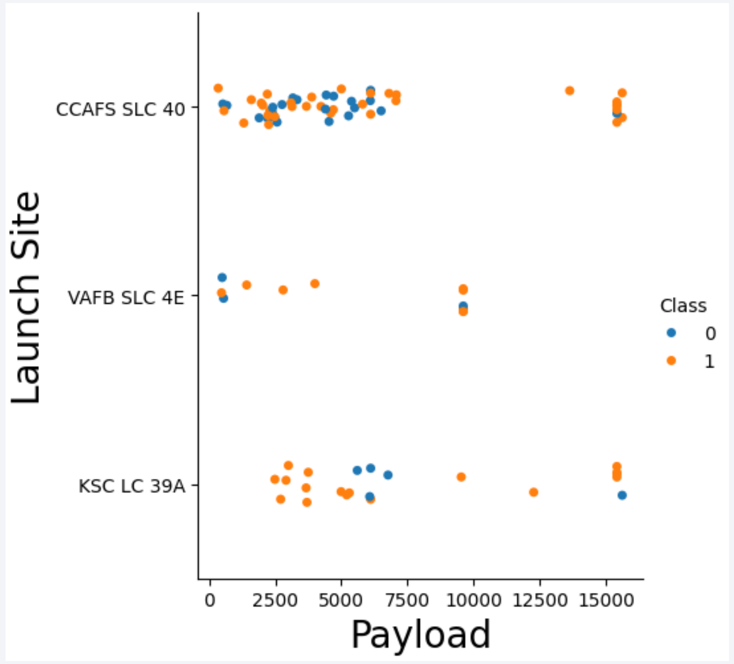
Show a scatter plot of Flight Number vs. Launch Site



We can see that Launch Site in Y axis and Flight Number in x-axis which in comparison the class = 1 are many compare to class = 0 in Launch site that CCAPS SLC 40 has many class on it

Flight Number vs. Launch Site

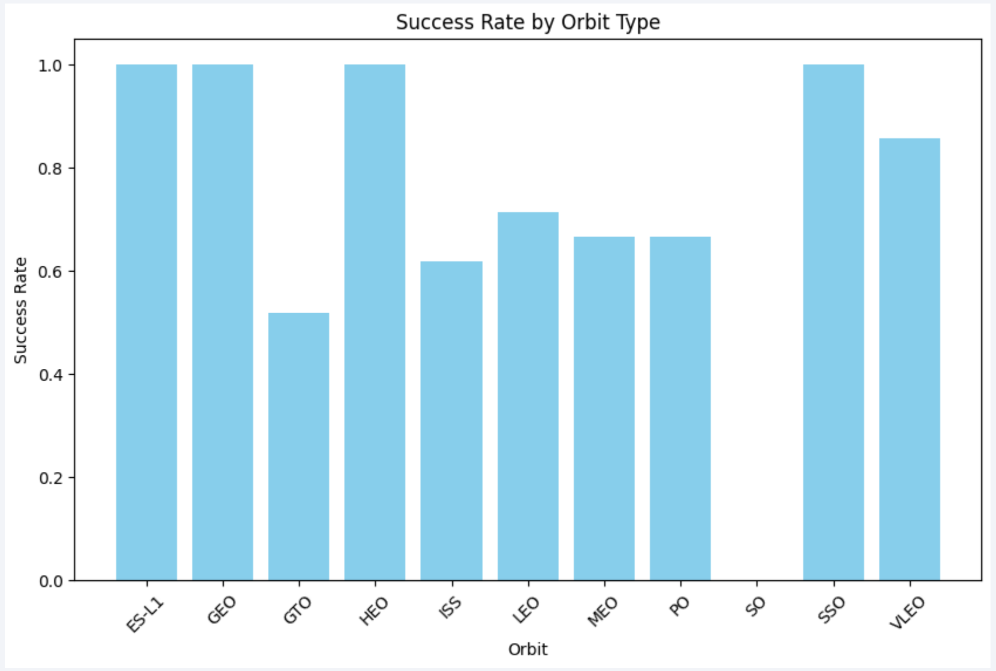
Show a scatter plot of Payload vs. Launch Site



We can see that Launch Site in Y axis and Payload in x-axis which in comparison the class = 1 are many compare to class = 0 and in Launch site that CCAPS SLC 40 has many class on it

Flight Number vs. Launch Site

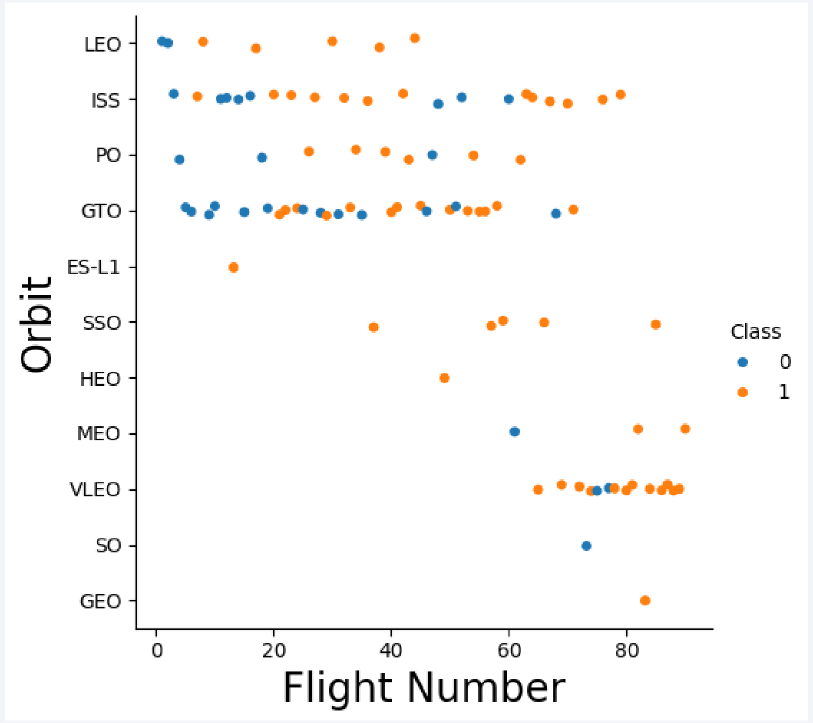
Show a bar chart for the success rate of each orbit type



In this chart, the success rate of ES-L1, GEO, HEO, and SSO are within the same count of Success Rate

Flight Number vs. Launch Site

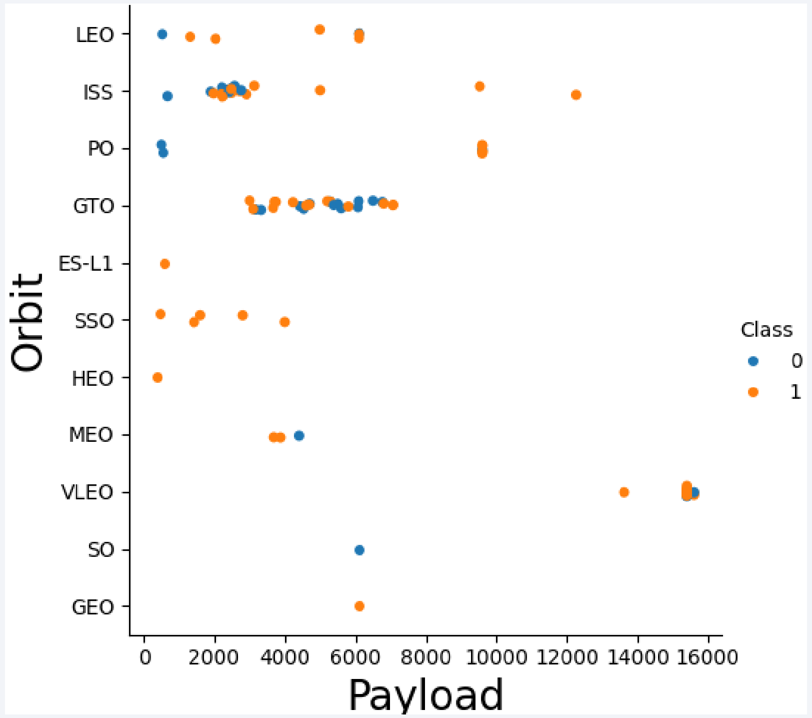
Show a scatter point of Flight number vs. Orbit type



As we can see the scatter point on which the VLEO has many on the estimate 60 – 80 while lISS is leading on the count following by GTO.

Flight Number vs. Launch Site

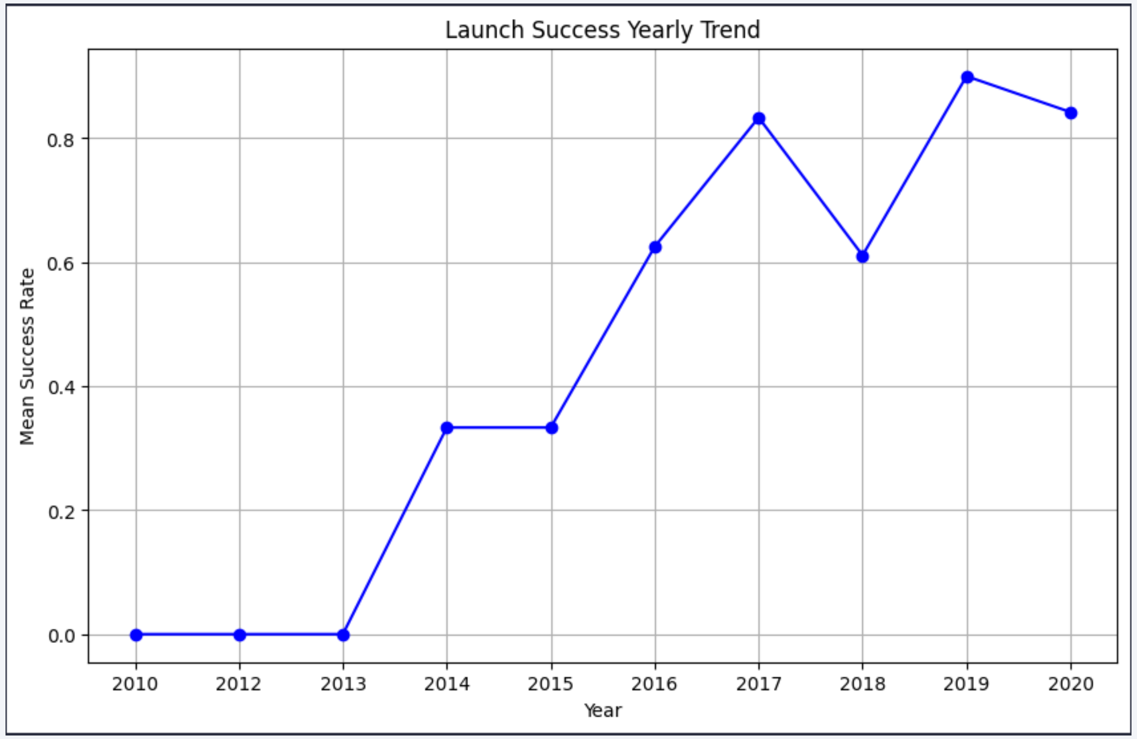
Show a scatter point of payload vs. orbit type



On the scatter point the GTO are intact between 2000 to not more than 800 which they compress in each other, VLEO has the highest Payload

Flight Number vs. Launch Site

Show a line chart of yearly average success rate



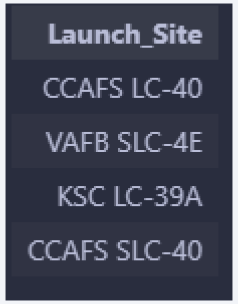
As we can see in line chart as the yearly progress each year the launch success is increase until in 2017 which decrease a bit and in 2019 the highest of all count success rate

All Launch Site Names

Find the names of the unique launch sites

Query: SELECT DISTINCT Launch_Site FROM SPACEXTABLE

Output:



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

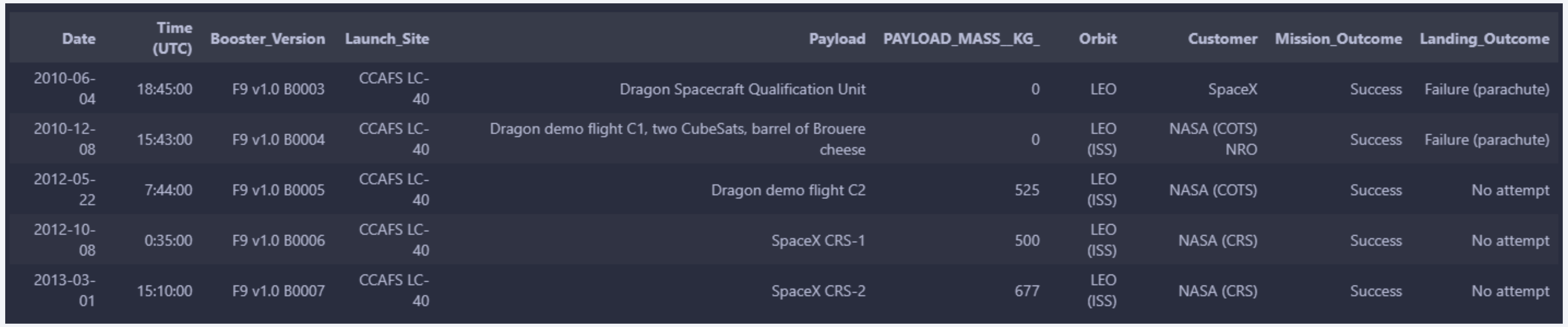
Explanation: Retrieves a unique list of all launch sites from the SPACEXTABLE. The DISTINCT keyword ensures that only distinct (unique) launch sites are returned, removing any duplicates.

Launch Site Names Begin with 'CCA'

Find 5 records where launch sites begin with `CCA`

Query: SELECT * FROM SPACEXTABLE WHERE Launch_Site Like 'CCA%' LIMIT 5

Output:



Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

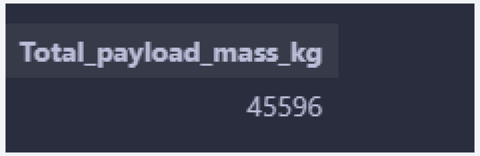
Explanation: Retrieve all columns from SPACEXTABLE where the Launch_Site starts with 'CCA'. The LIKE 'CCA%' clause filters the results, and LIMIT 5 restricts the output to the first 5 matching records.

Total Payload Mass

Calculate the total payload carried by boosters from NASA

Query: SELECT SUM(PAYLOAD_MASS__KG_) as Total_payload_mass_kg FROM SPACEXTABLE WHERE customer = 'NASA (CRS)’

Output:



```
Total_payload_mass_kg
                45596
```

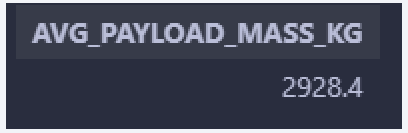
Explanation: Calculates the total payload mass in kilograms for all launches where the customer is 'NASA (CRS)'. The SUM(PAYLOAD_MASS__KG_) function adds up the payload masses, and the result is labeled as Total_payload_mass_kg.

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1

Query: SELECT AVG(PAYLOAD_MASS__KG_) as AVG_PAYLOAD_MASS_KG FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1’

Output:



AVG_PAYLOAD_MASS_KG
2928.4

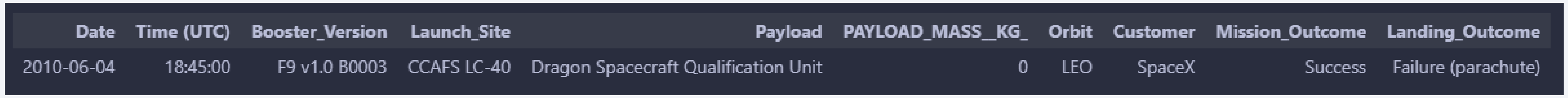
Explanation: Calculates the average payload mass in kilograms for all launches that used the 'F9 v1.1' booster version. The AVG(PAYLOAD_MASS__KG_) function computes the average, and the result is labeled as AVG_PAYLOAD_MASS_KG..

First Successful Ground Landing Date

Find the dates of the first successful landing outcome on ground pad

Query: SELECT * FROM SPACEXTABLE ORDER BY Date asc limit 1

Output:



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

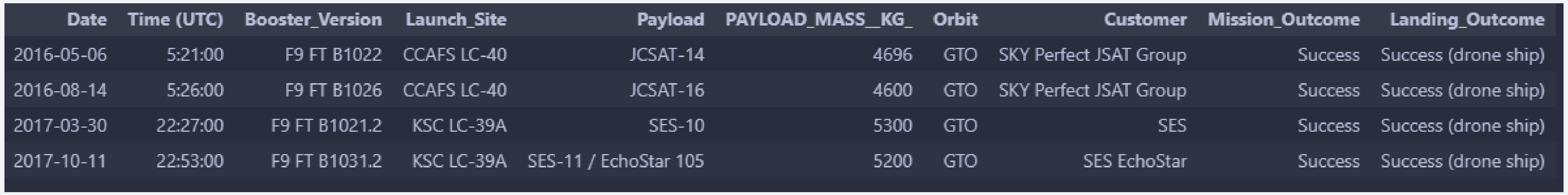
Explanation:Retrieves the earliest launch record from SPACEXTABLE. The ORDER BY Date ASC clause sorts the records by date in ascending order, and LIMIT 1 ensures only the first (earliest) record is returned.

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

Query: SELECT * FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ >= 4000 AND PAYLOAD_MASS__KG_ < 6000

Output:



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

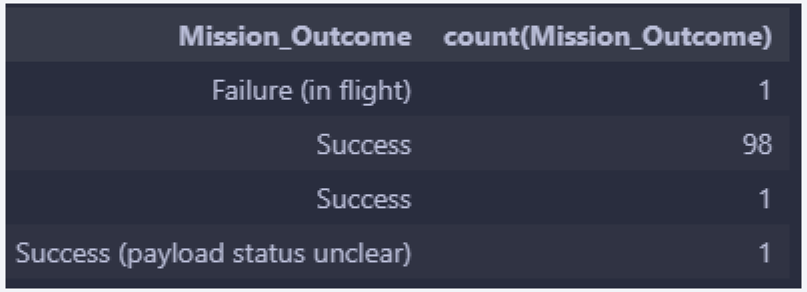
Explanation: Retrieve all columns from SPACEXTABLE where the landing outcome was a successful drone ship landing and the payload mass is between 4000 and 5999 kilograms (inclusive of 4000 but exclusive of 6000).

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

Query: SELECT Mission_Outcome, count(Mission_Outcome) FROM SPACEXTABLE Group By Mission_Outcome

Output:



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

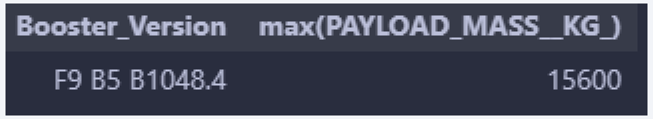
Explanation: Counts the number of occurrences of each mission outcome in SPACEXTABLE. The GROUP BY Mission_Outcome clause groups the records by the Mission_Outcome field, and the COUNT(Mission_Outcome) function provides the count for each group..

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

Query: SELECT * FROM (SELECT Booster_Version, max(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)

Output:



Booster_Version	max(PAYLOAD_MASS__KG_)
F9 B5 B1048.4	15600

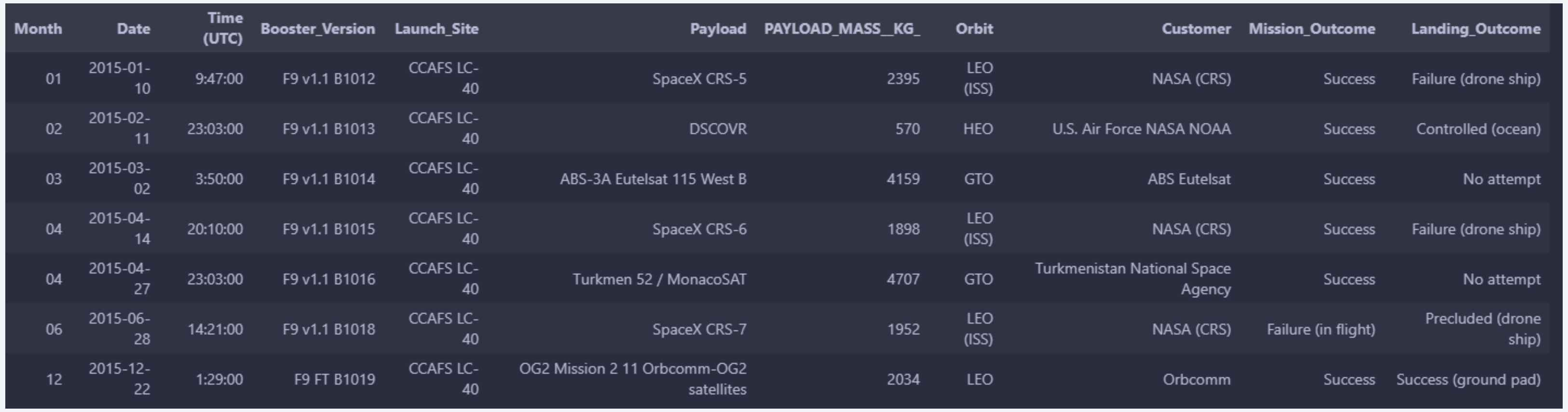
Explanation: Retrieve all columns from the result of a subquery that finds the maximum payload mass for each booster version in SPACEXTABLE. The MAX(PAYLOAD_MASS__KG_) function identifies the highest payload mass

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

Query: SELECT substr(Date, 6,2) as Month, * FROM SPACEXTABLE WHERE substr(Date, 0,5) = '2015'

Output:



Month	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
01	2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
02	2015-02-11	23:03:00	F9 v1.1 B1013	CCAFS LC-40	DSCOVR	570	HEO	U.S. Air Force NASA NOAA	Success	Controlled (ocean)
03	2015-03-02	3:50:00	F9 v1.1 B1014	CCAFS LC-40	ABS-3A Eutelsat 115 West B	4159	GTO	ABS Eutelsat	Success	No attempt
04	2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
04	2015-04-27	23:03:00	F9 v1.1 B1016	CCAFS LC-40	Turkmen 52 / MonacoSAT	4707	GTO	Turkmenistan National Space Agency	Success	No attempt
06	2015-06-28	14:21:00	F9 v1.1 B1018	CCAFS LC-40	SpaceX CRS-7	1952	LEO (ISS)	NASA (CRS)	Failure (in flight)	Precluded (drone ship)
12	2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

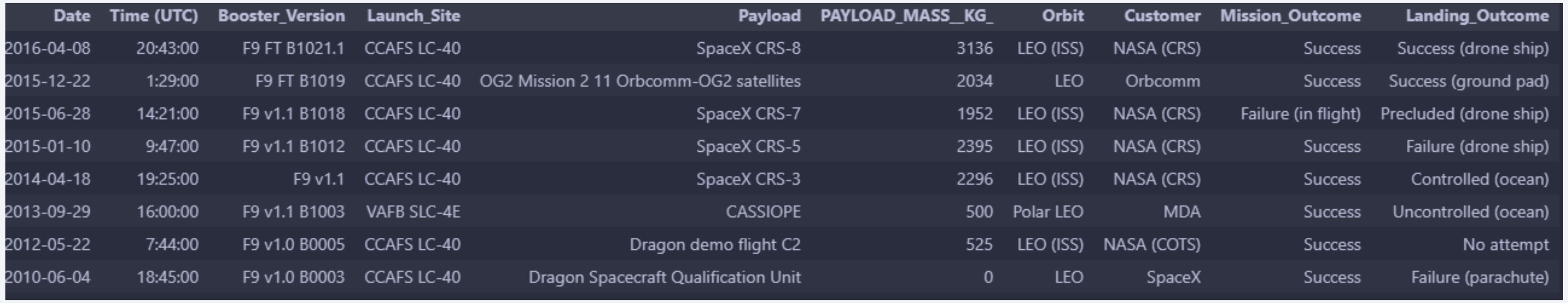
Explanation: Retrieve all columns from SPACEXTABLE along with the month extracted from the Date column for records from the year 2015. The SUBSTR(Date, 6, 2) function extracts the month part of the date, and SUBSTR(Date, 1, 4) = '2015' filters the results to include only those from the year 2015

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

Query: SELECT * FROM (SELECT * FROM SPACEXTABLE WHERE DATE(date) BETWEEN '2010-06-04' AND '2017-03-20') GROUP BY Landing_Outcome ORDER BY date DESC

Output:



Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)
2015-06-28	14:21:00	F9 v1.1 B1018	CCAFS LC-40	SpaceX CRS-7	1952	LEO (ISS)	NASA (CRS)	Failure (in flight)	Precluded (drone ship)
2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2014-04-18	19:25:00	F9 v1.1	CCAFS LC-40	SpaceX CRS-3	2296	LEO (ISS)	NASA (CRS)	Success	Controlled (ocean)
2013-09-29	16:00:00	F9 v1.1 B1003	VAFB SLC-4E	CASSIOPE	500	Polar LEO	MDA	Success	Uncontrolled (ocean)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)

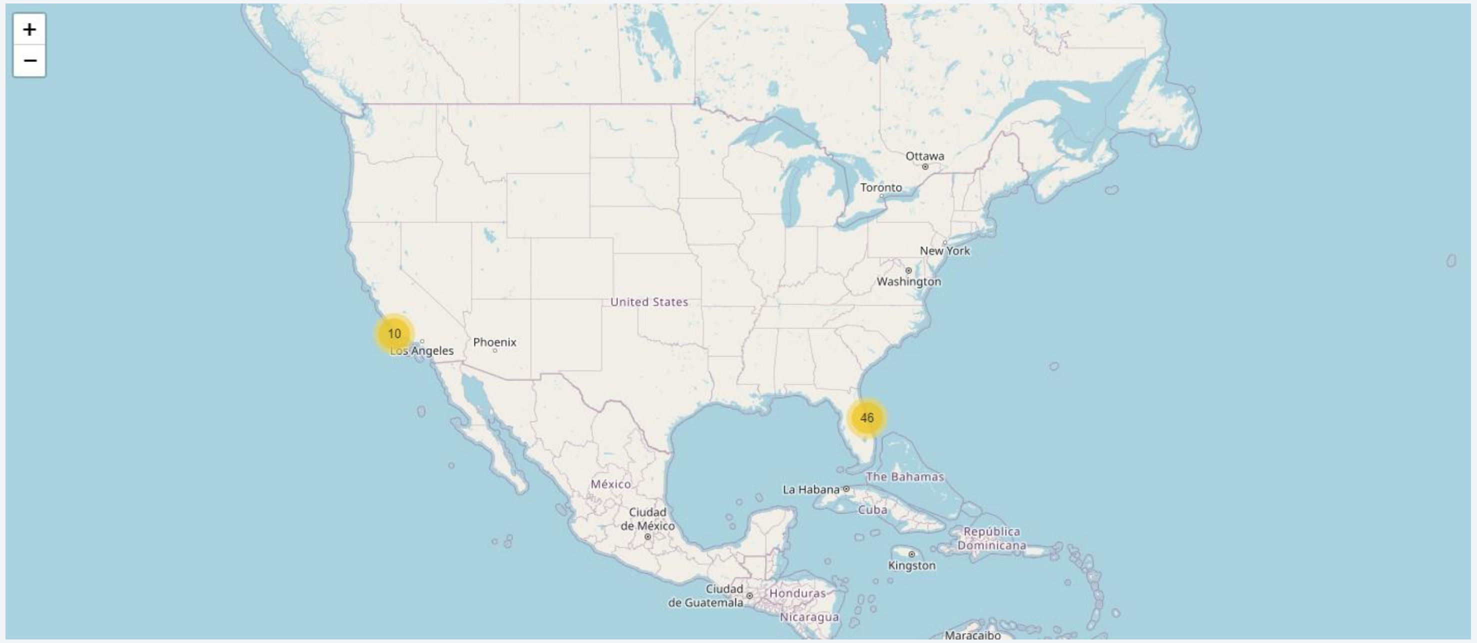
Explanation: Retrieve all records from SPACEXTABLE where the date falls between June 4, 2010, and March 20, 2017. It then groups the results by Landing_Outcome and sorts the grouped results by date in descending order.

Section 3

Launch Sites Proximities Analysis

Locations of Launching Area for SpaceX

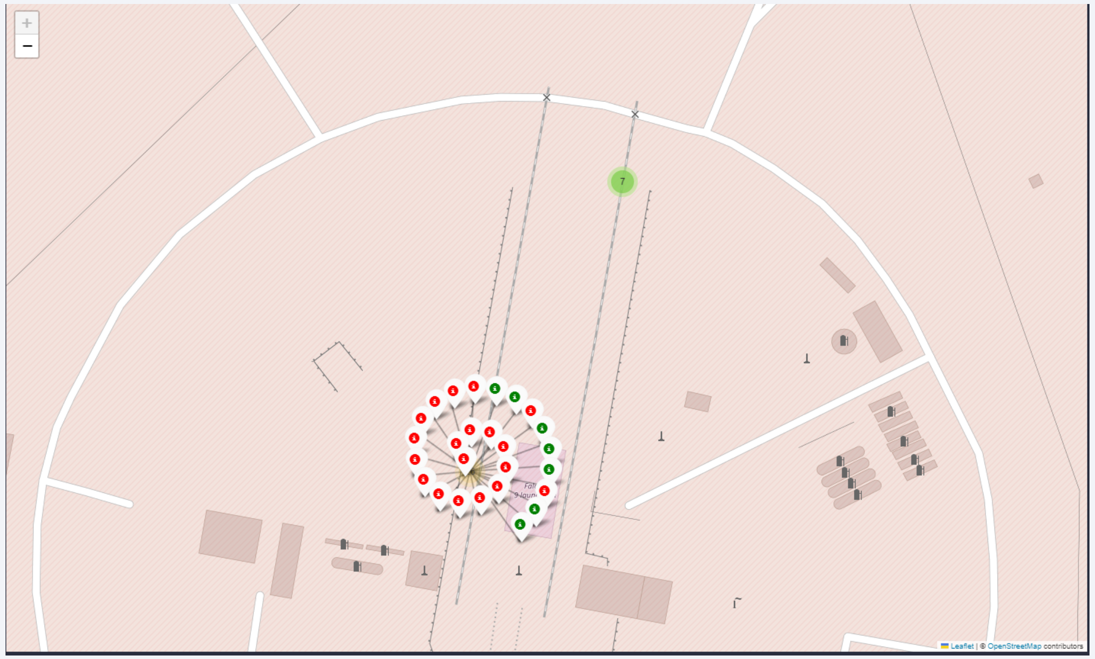
Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map



The only location for SpaceX Launch exist in West Coast located in State of California and for the East Code is onMiami State

Color Differentiation in Assessing Launch Outcomes

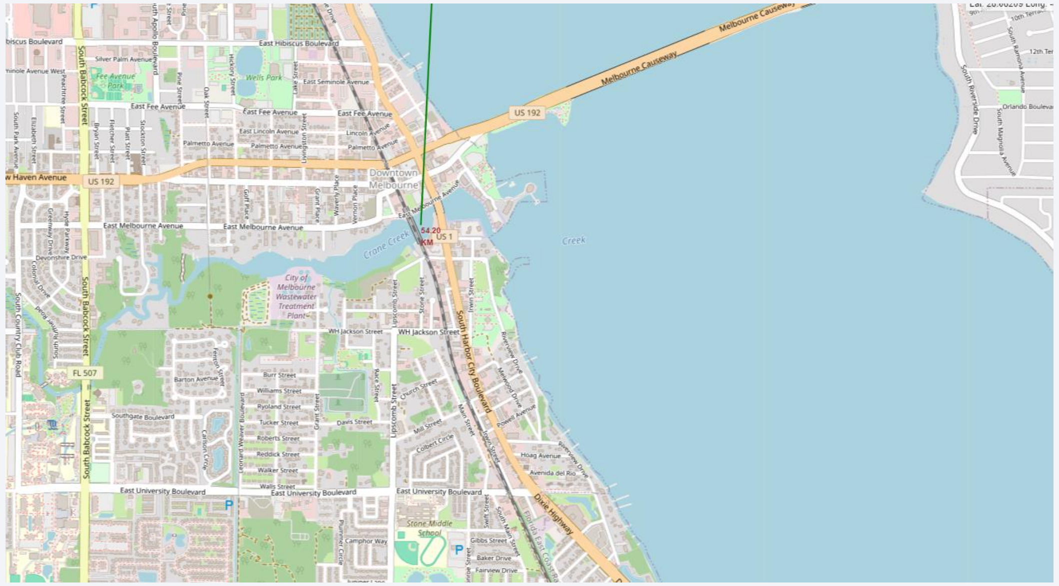
Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map

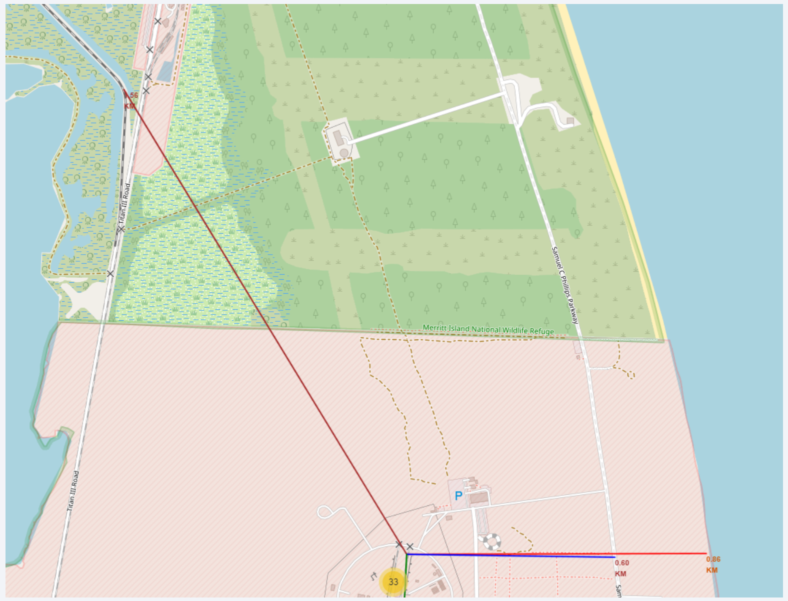


As per example in which located in Falcon 9 launchpad only 7 launch is success and other color is failed

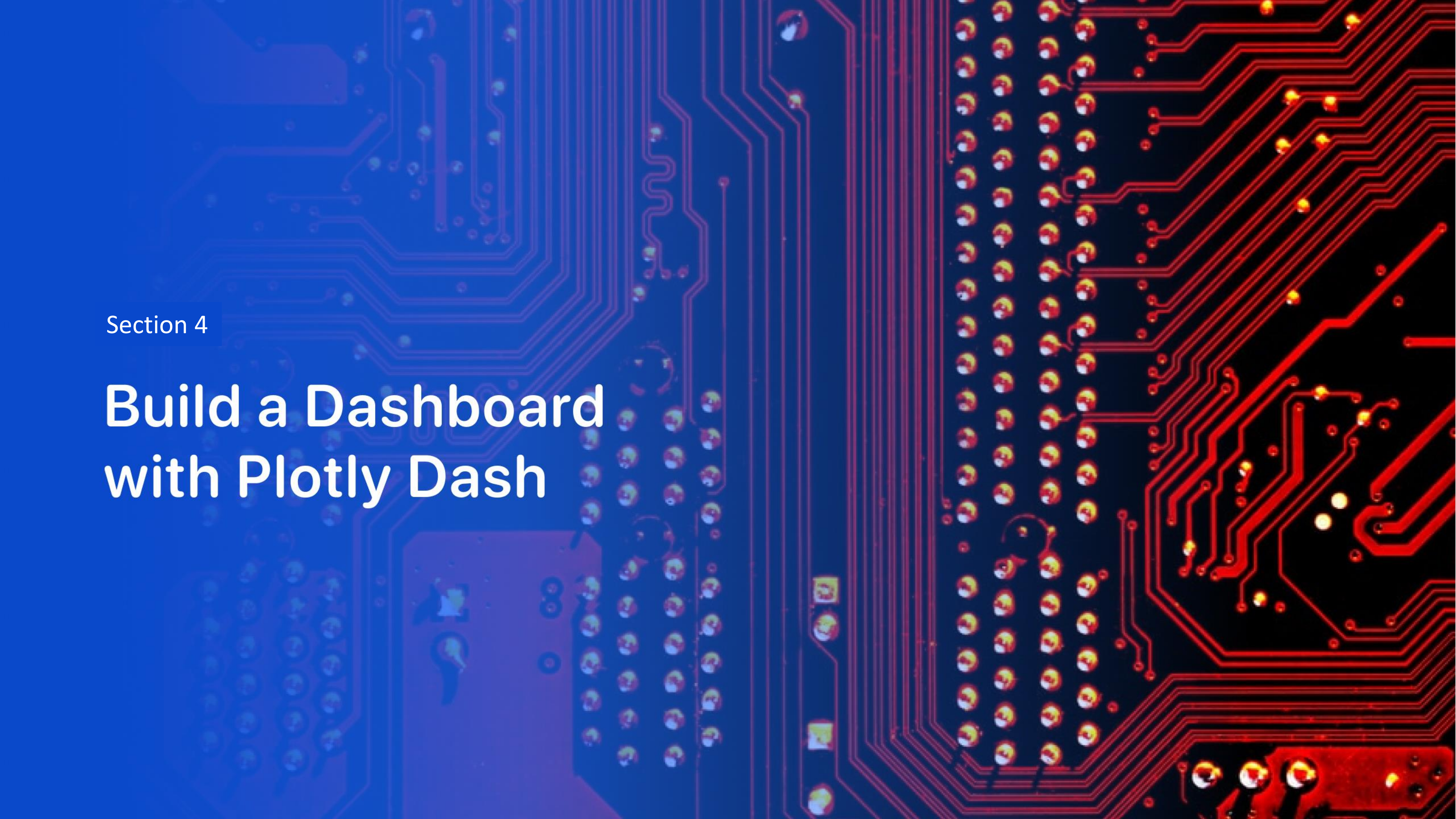
Locating and Calculate distance between a launch site and its proximities

Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed





The Distance calculated from a launch site to city of Melbourne is 54.20 KM while the distance from site to road is 0.60 KM, to coast line 0.86KM, and to the railroad is 1.56 KM

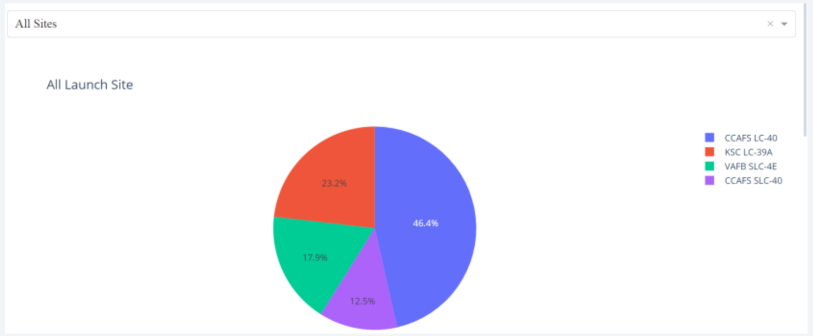


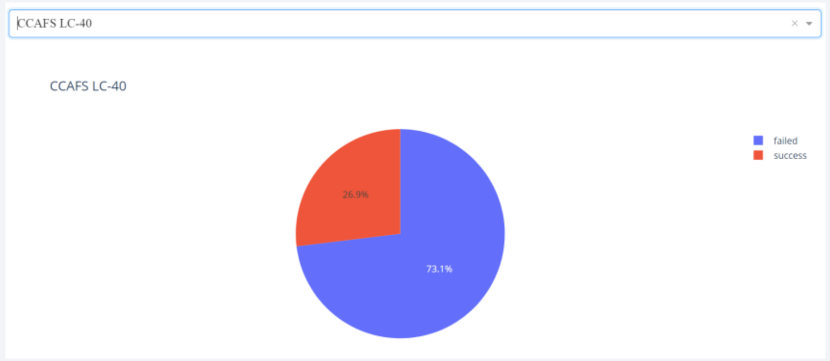
Section 4

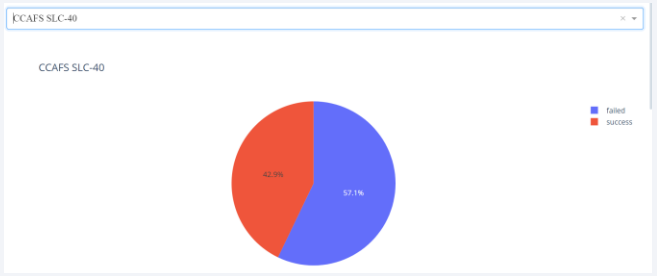
Build a Dashboard with Plotly Dash

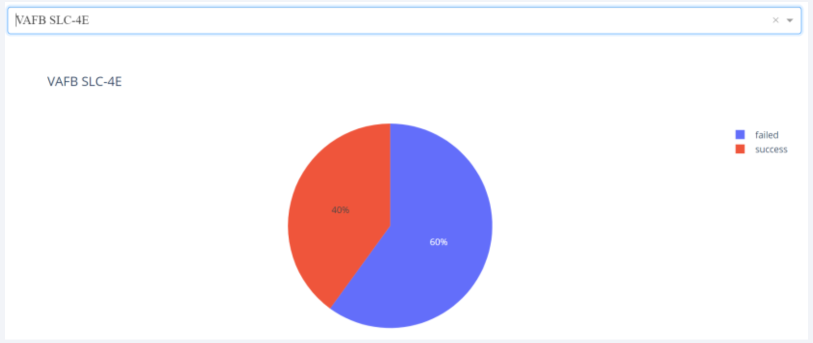
Pie chart of Launch Site Success Rate

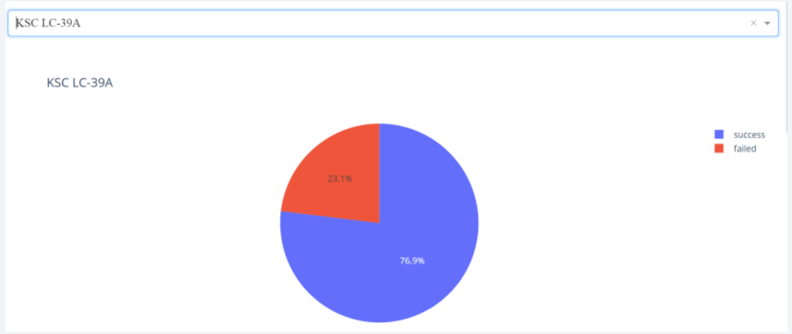
Show the screenshot of launch success count for all sites, in a piechart









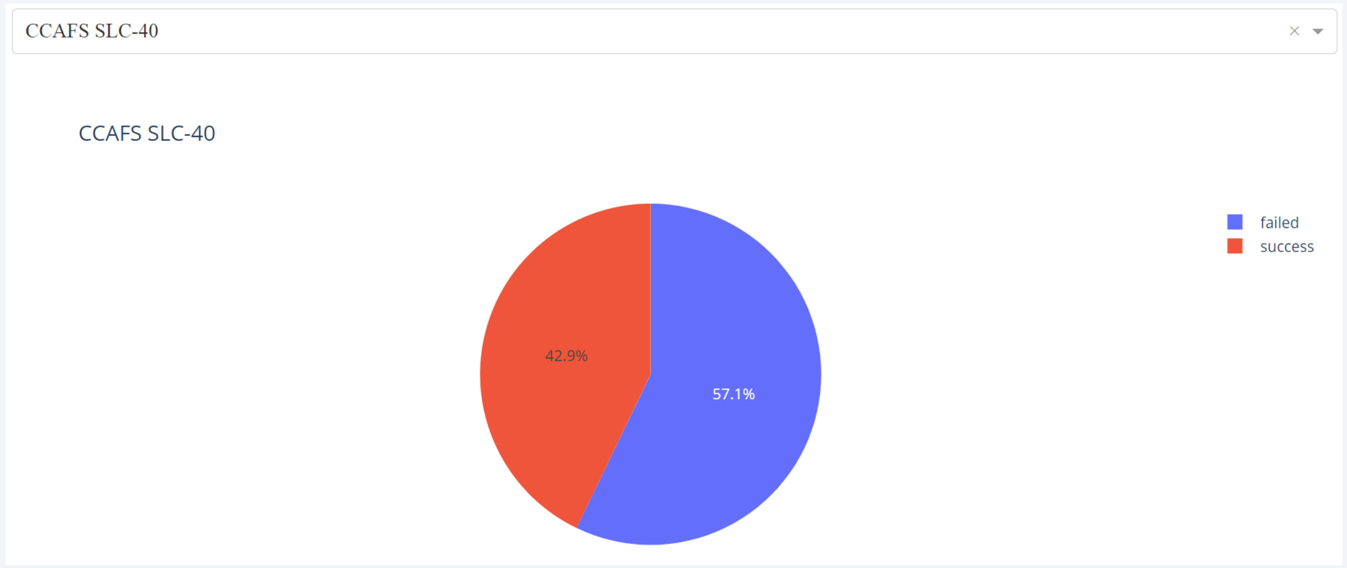


The Highest ratio of success in these Pie chart is CCAFS SLC where the lowest is

KSC LC -39A

Highest Success Rate Launch in Pie Chart

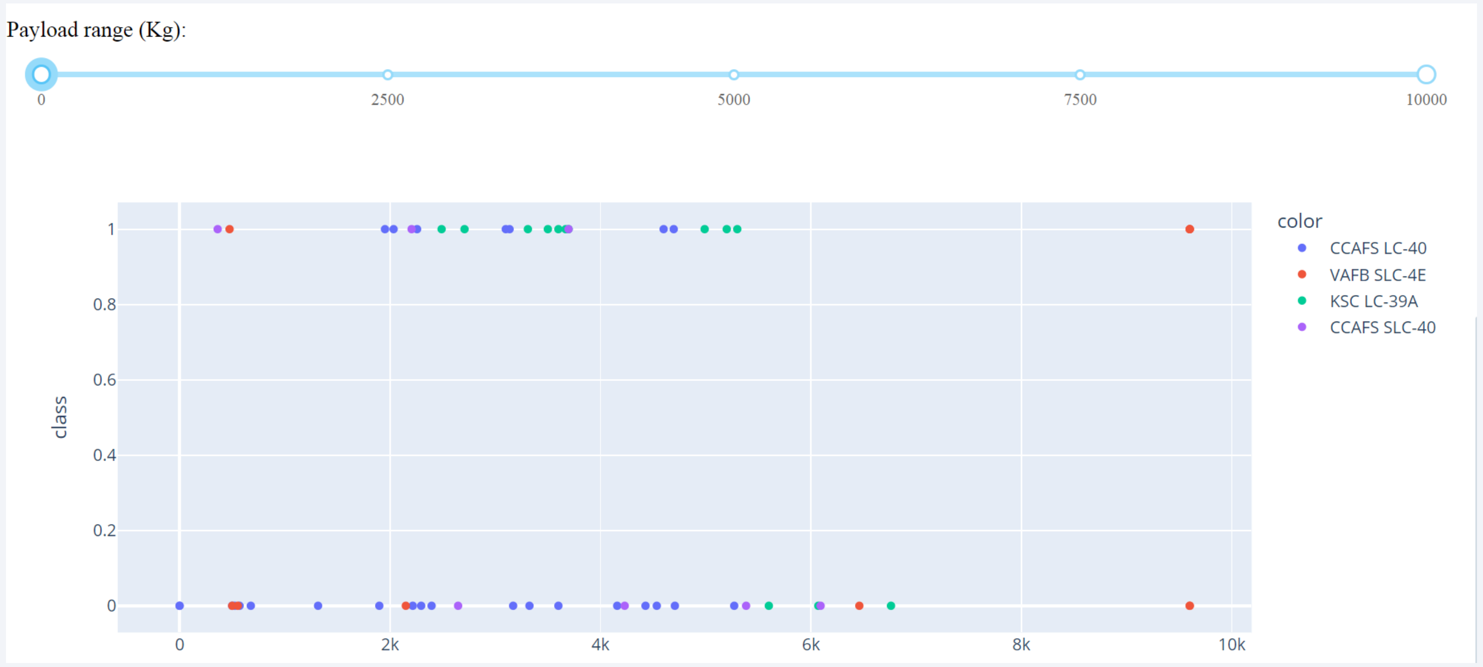
Show the screenshot of the piechart for the launch site with highest launch success ratio



The Success Rate of this site is 42.9% or has 3 success launch and then has the failed count of 4

Scatter Plot of Payload Range

Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider



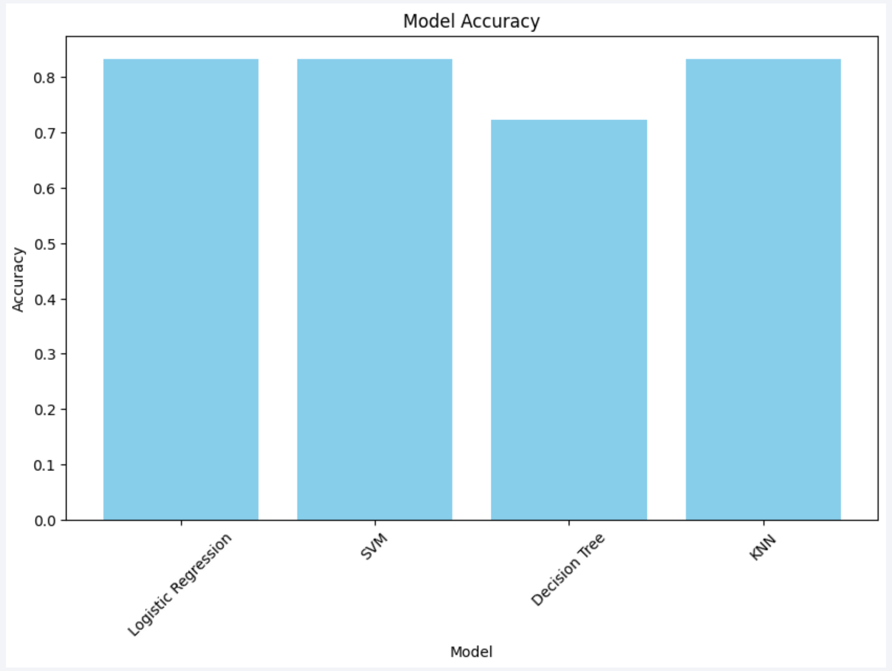
As we can see in the scatter plot CCAFS LC-40 (Blue): Consistent presence across the entire payload range, **VAFB SLC-4E** (Red): Appears mostly in lower payload ranges (up to about 3,500 Kg), with a couple of points near the highest payload range (around 10,000 Kg). **KSC LC-39A** (Green): Appears across a broad range of payloads, mostly in Class 1. **CCAFS SLC-40** (Purple): Similar to CCAFS LC-40, present across the payload range.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Visualize the built model accuracy for all built classification models, in a bar chart

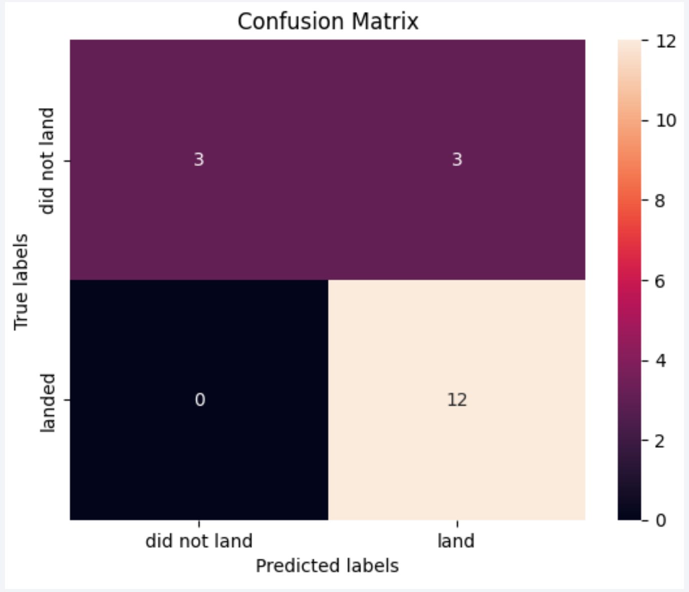


The model Logistic Regression, Decision Tree and KNN are the same value with

0.8333333333333334

Confusion Matrix

Show the confusion matrix of the best performing model with an explanation



Examining the confusion matrix, we see that logistic regression can distinguish between the different classes. We see that the major problem is false positives.

Conclusions

Overview of Models Tested:

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

Performance Summary:

- Logistic Regression: Accuracy = 0.8333
- SVM: Accuracy = 0.8333
- Decision Tree: Accuracy = 0.7222
- KNN: Accuracy = 0.8333

Best Performing Model::

- Model: Logistic Regression
- Accuracy: 0.8333

Key Findings:

- Logistic Regression, SVM, and KNN all achieved the same highest accuracy of 0.8333, indicating strong performance.
- The Decision Tree model performed slightly lower with an accuracy of 0.7222

Implications:

- Logistic Regression, with its high accuracy, is recommended for predicting Falcon 9 first stage landing success
- Additional considerations may include model complexity, interpretability, and computational efficiency when selecting the final model.

Appendix

Python Code Libraries:

- import pandas as pd
- import numpy as np
- import matplotlib.pyplot as plt
- import seaborn as sns
- from sklearn import preprocessing
- from sklearn.model_selection import train_test_split
- from sklearn.model_selection import GridSearchCV
- from sklearn.linear_model import LogisticRegression
- from sklearn.svm import SVC
- from sklearn.tree import DecisionTreeClassifier
- from sklearn.neighbors import KNeighborsClassifieri
- import requestsSQL

Appendix

SQL Queries:

1. SQLite

Data Gathering:

1. SpaceXAPI
2. Wikipedia Page for SpaceX

Relevant Assets:

1. Python code snippets
2. SQL queries
3. Charts
4. Notebook outputs
5. Data sets created during the project

Thank you!