IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Amrutha
16/9/2024

Outline

Executive Summary

Summary of Methodologies

In this analysis, we applied various methodologies to evaluate and improve the performance of a K-Nearest Neighbors (KNN) classifier. The process involved the following key steps:

Data Preprocessing:
•Split data into training and test sets.
•Applied feature scaling for normalization

Model Training:
•Used K-Nearest Neighbors (KNN) with cross-validation for hyperparameter tuning.
•Selected optimal parameters for the KNN model.

Model Evaluation:
•Calculated accuracy on test data.
•Plotted confusion matrix to visualize prediction accuracy.
•Evaluated precision, recall, and F1-score for performance insights.

Executive Summary

Results

In this analysis, we applied various methodologies to evaluate and improve the performance of a K-Nearest Neighbors (KNN) classifier. The process involved the following key steps:

Data Preprocessing:

•Split data into training and test sets.

•Applied feature scaling for normalization

Model Training:

•Used K-Nearest Neighbors (KNN) with cross-validation for hyperparameter tuning.

•Selected optimal parameters for the KNN model.

Model Evaluation:

•Calculated accuracy on test data.

•Plotted confusion matrix to visualize prediction accuracy.

•Evaluated precision, recall, and F1-score for performance insights.

Introduction

Project background and context

- Objective: Analyze SpaceX launch data to uncover key insights and trends.
- Data Source: SpaceX launch records including launch sites, payload mass, outcomes, and booster versions.
- Tools Used: Data Analysis: Python, Pandas
- Visualization: Plotly Dash

Introduction

Key Questions Addressed

- Launch Site Performance:
- Which site has the most successful launches?
- Which site has the highest success rate?
- Payload Mass Analysis:
- How does payload mass influence launch success?
- Booster Version Performance:
- Which Falcon 9 booster version has the highest success rate?

Correlation Analysis:

- Are there patterns between payload mass and booster performance?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

  Data Source: SpaceX launch data obtained from publicly available records.
  Data Collection:

  Downloaded dataset using Python libraries (`wget`, `pandas`).

  Initial exploration and validation of data quality.

- Perform data wrangling
  - Cleaned missing and inconsistent values.
  - Transformed data types and formats for analysis.
  - Filtered and categorized data based on launch sites, payload mass, and outcomes.

Methodology

Perform exploratory data analysis (EDA) using visualization and SQL

- Techniques Used:
  Visualization: Used matplotlib, seaborn, and plotly for graphical representations.
  SQL Queries: Executed to extract key statistics and patterns.
- Key Findings:
  Trends and anomalies in launch success rates.
  Correlations between payload mass and launch outcomes.

- Perform interactive visual analytics using Folium and Plotly Dash
- Tools:
  Folium: Mapped geographic distribution of launch sites.
  Plotly Dash: Built interactive dashboards for real-time data exploration.

- User Interaction: Enabled site selection and payload range adjustment for dynamic insights.

Methodology

- Perform predictive analysis using classification models
- **Modeling Approach**:
  - Employed classification models like KNN, Decision Trees, and Logistic Regression.
- **Model Building**:
  - Feature selection and engineering for optimal model performance.
  - Hyperparameter tuning using cross-validation techniques.
- **Model Evaluation**:
  - Evaluated models based on accuracy, precision, recall, and F1-score.
  - Utilized confusion matrix and ROC curves for comprehensive performance analysis.

Data Collection

- Describe how data sets were collected.

1. Data Source Identification:

- Public Databases: NASA, SpaceX, and other open data platforms.
- APIs and Web Scraping: Utilized APIs for real-time data fetching.

2. Data Acquisition:

- Downloading Datasets:
- Command: wget "dataset_url"
- Data File: spacex_launch_dash.csv
- APIs: Fetched additional data such as booster versions and payload specifications.

Data Collection

- Describe how data sets were collected.

3. Data Storage:

- Local Storage: CSV files stored locally for analysis.
- Cloud Storage: Backed up datasets on cloud for seamless access.

4. Data Preprocessing:

- Cleaning: Handled missing values and inconsistent formats.
- Transformation: Converted data types, normalized payload metrics.

Data Collection

- **Flowchart: Data Collection Process**

Start

⬇️

Data Source Identification(Identify APIs, public databases)

⬇️

Data Acquisition(Download CSV, Use APIs)

⬇️

Data Storage(Save locally, cloud backup)

⬇️

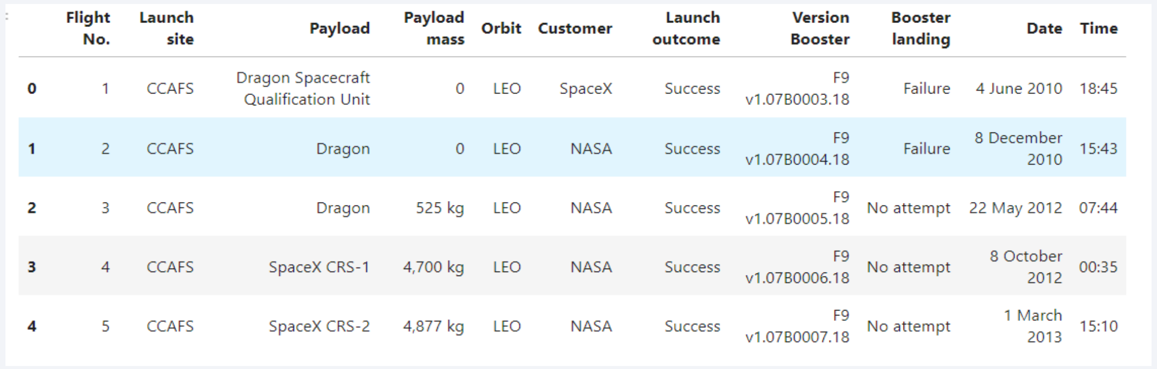
Data Preprocessing(Cleaning, Transformation)

⬇️

End

Data Collection - Scraping

Git Link



	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	F9 v1.07B0003.18	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.07B0004.18	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.07B0005.18	No attempt	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success	F9 v1.07B0006.18	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success	F9 v1.07B0007.18	No attempt	1 March 2013	15:10

Start

↓

API Endpoint Identification (Identify relevant SpaceX API endpoints)

↓

API Request (Perform GET requests to retrieve data)

↓

Data Parsing (Extract JSON data fields)

↓

Data Storage(Save data locally or to cloud storage)

↓

Data Preprocessing (Cleaning, Transformation)

↓

End

Data Wrangling

Data Processing Steps:

- **Loading Data:**
    - Import datasets (CSV, API data).
    - Use libraries like Pandas for data handling.
- **Cleaning Data:**
    - Handle missing values (imputation or removal).
    - Remove duplicates.
- **Transforming Data:**
    - Convert data types (e.g., strings to dates).
    - Normalize or scale features as needed.

Data Wrangling

- **Feature Engineering:**
  - Create new features from existing ones (e.g., extracting year from date).
  - Encode categorical variables.
- **Data Validation:**
  - Check for inconsistencies or errors.
  - Validate data integrity.
- **Final Dataset Preparation:**
  - Save the cleaned and processed dataset for analysis.

Data Wrangling

Start

↓

Loading Data(Import datasets using Pandas)

↓

Cleaning Data(Handle missing values, remove duplicates)

↓

Transforming Data(Convert data types, normalize features)

↓

Feature Engineering(Create new features, encode categorical variables)

↓

Data Validation(Check for inconsistencies, validate integrity)

↓

Final Dataset Preparation(Save cleaned dataset for analysis)

↓

End

Git Link

EDA with Data Visualization

1. Objectives of EDA:

Understand data distribution and relationships between variables.

Identify trends, patterns, and anomalies within the dataset.

2. Key Steps in EDA:

Data Cleaning: Handled missing values and outliers.

Summary Statistics: Generated descriptive statistics for key features.

3. Data Visualizations:

1. Distribution Plots: Histogram of Launch Success Rates.
2. Correlation Matrix: Heatmap of Feature Correlations.
3. Time Series Analysis: Line Plot of Launches Over Time.
4. Categorical Analysis: Bar Chart of Rocket Types vs. Success Rate.

4. Key Insights:

Trends Identified: Increase in successful launches over time.

Anomalies Detected: Identify any unusual drops in success rates during specific periods.

Git Link

EDA with SQL

Task 1: Retrieve unique launch sites.

```
%sql SELECT DISTINCT LAUNCH_SITE AS "Launch_Sites" FROM SPACEX;
```

Task 2: Display records where launch sites start with 'CCA'

```
%sql SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

Task 3: Calculate total payload mass for missions conducted by NASA (CRS)

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)"
  FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)';
```

Git Link

EDA with SQL

Task 4: Compute average payload mass for booster version F9 v1.1.

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass F9 v1.1"
FROM SPACEX
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

Task 5: Find the date of the first successful landing on the ground pad.

```
%sql SELECT MIN(DATE) AS "First Successful Landing Outcome in Ground Pad"
 FROM SPACEX
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

EDA with SQL

Task 6: List booster versions with successful drone ship landings and payload mass between 4000-6000 kg.

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

Task 7: Count the number of successful and failed missions

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX
   WHERE MISSION_OUTCOME LIKE 'Success%';
```

EDA with SQL

%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';

Task 8: Identify booster versions that carried the maximum payload mass.

%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);

Task 9: Retrieve failed landing outcomes in drone ship in 2015, along with booster versions and launch sites.

%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE year(DATE) = '2015' AND LANDING__OUTCOME = 'Failure (drone ship)';

EDA with SQL

Task 10: Rank landing outcomes by count between specific dates.

```
%sql SELECT LANDING__OUTCOME AS "Landing Outcome",
   COUNT(LANDING__OUTCOME) AS "Total Count"

FROM SPACEX

WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY
   LANDING__OUTCOME ORDER BY COUNT(LANDING__OUTCOME) DESC;
```

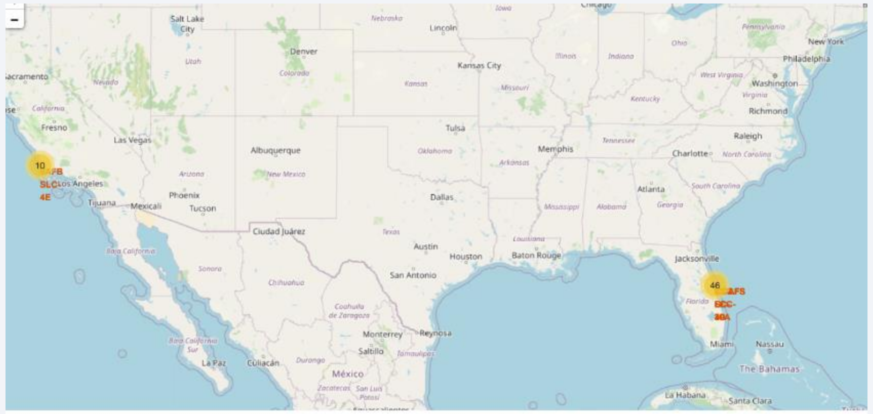
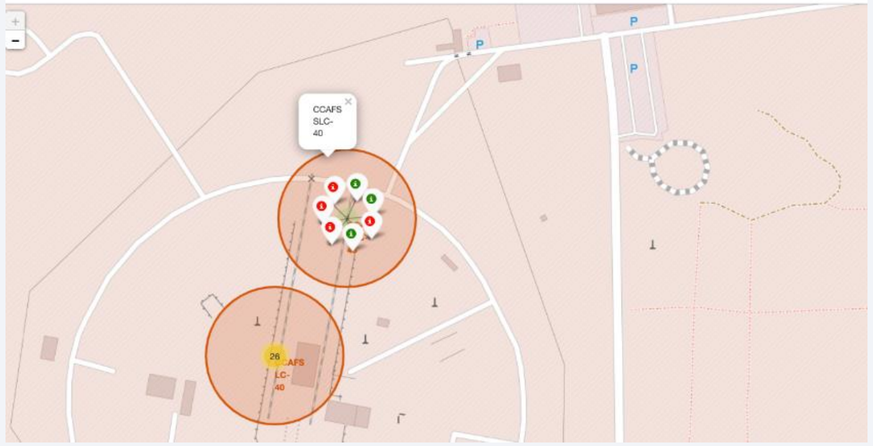
Build an Interactive Map with Folium

- **Task 1: Mark all launch sites on a map**
- **Data Preparation:**
  - Download the `spacex_launch_geo.csv` file, which contains the launch sites' names, latitudes, longitudes, and the success/failure status of each launch.
  - Create a new DataFrame (`launch_sites_df`) with unique launch sites and their coordinates.
- **Map Initialization:**
  - Initialize a Folium map centered around NASA Johnson Space Center in Houston, Texas.
  - Add a blue circle and label marker to represent NASA JSC.
- **Add Launch Site Markers:**
  - Define a dictionary `launch_sites_dict` for storing the launch site names and coordinates.
  - Create custom functions to add markers and labels on the map.
  - Loop through the `launch_sites_dict` to add each launch site to the map using `add_marker_on_map` function.
  - The generated map displays all SpaceX launch sites with circles and labels for visual clarity.

Git link

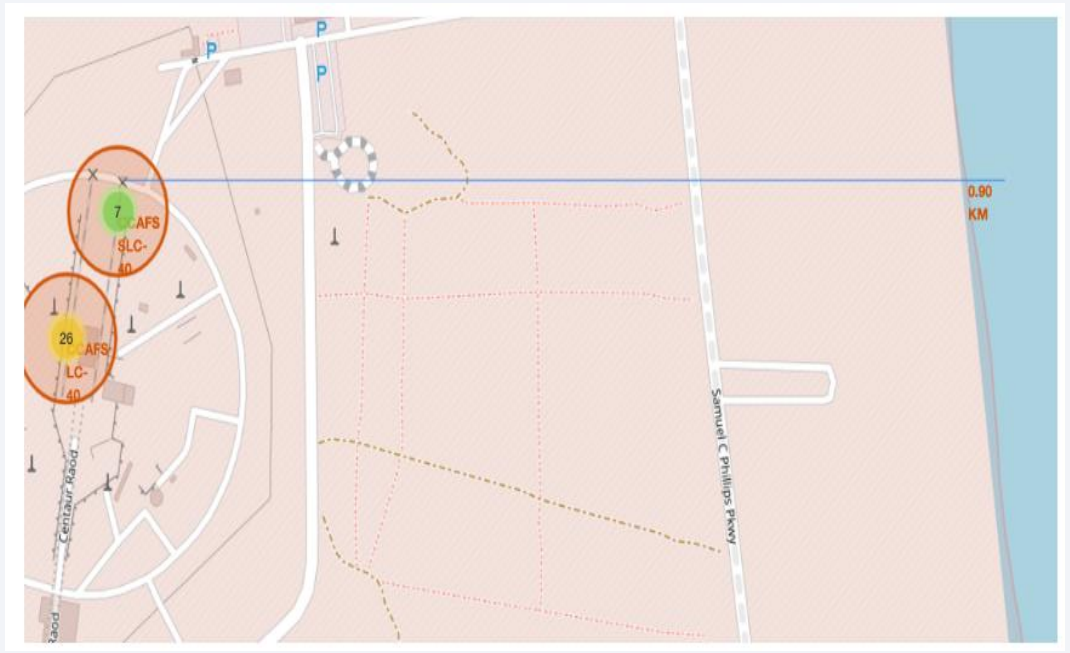
Build an Interactive Map with Folium

- **Task 2: Mark the success/failed launches for each site on the map**
- **Success/Failure Marker Setup:**
  - Use the `class` column in `spacex_df` to determine the success (class = 1) or failure (class = 0) of each launch.
  - Create a `marker_color` column in `spacex_df` to store the color codes: 'green' for success and 'red' for failure.
- **Add Launch Outcome Markers:**
  - Initialize a `MarkerCluster` object to manage multiple markers in the same location.
  - For each launch, add a marker to the map using `add_marker_cluster_on_map` function, which color-codes the marker based on the outcome.
  - The resulting map allows you to visually assess the success rate of each launch site.

Build an Interactive Map with Folium

- **Calculate the distances between a launch site to its proximities**
- **Distance Calculation Function:**
  - Define a `calculate_distance` function using the Haversine formula to compute distances between two geographical points.
- **Mouse Position Plugin:**
  - Add a `MousePosition` plugin to the map, which allows you to retrieve the coordinates of points of interest (POIs) such as railways, highways, and coastlines near each launch site.
- **Proximity Analysis:**
  - Zoom in on each launch site and identify nearby geographical features like roads, coastlines, and infrastructure.
  - Record the coordinates of these features and use the `calculate_distance` function to determine their distance from the launch sites.



Build a Dashboard with Plotly Dash

- **Interactive Visualizations:**
  - **Line Charts:** Show launch success over time.
  - **Bar Charts:** Compare the number of launches by rocket type.
  - **Scatter Plots:** Analyze the relationship between payload and launch success.
- **User Interactions:**
  - **Dropdown Menus:** Select rocket types or specific launch years.
  - **Hover Effects:** Display detailed data points on mouse hover.
  - Insights Gained:
- Identified trends in launch success rates over the years.
- Analyzed the impact of payload size on launch outcomes

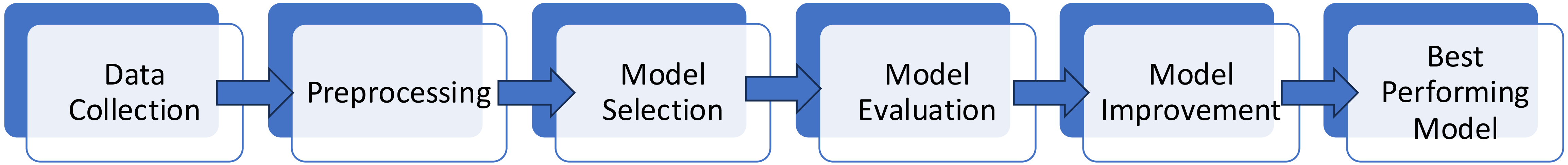
Predictive Analysis (Classification)

- Model Development Process:
- **Data Collection:**
  - Utilized SpaceX launch data via API.
- **Data Preprocessing:**
  - Handled missing values and performed feature engineering.
  - Encoded categorical variables and scaled numerical features.
- **Model Selection:**
  - Chose several classification algorithms:
    - Logistic Regression
    - Decision Trees
    - Random Forest
    - Support Vector Machine (SVM)

Predictive Analysis (Classification)

- **Model Evaluation:**
  - Split data into training and testing sets (80/20).
  - Used metrics:
    - Accuracy
    - Precision
    - Recall
    - F1 Score
  - Performed Cross-Validation for robust evaluation.
- **Model Improvement:**
  - Tuned hyperparameters using Grid Search.
  - Employed techniques like:
    - Feature selection
    - Ensemble methods
    - Data augmentation
- **Best Performing Model:**
  - Selected the model with the highest F1 Score and lowest overfitting.

Predictive Analysis (Classification)



Results

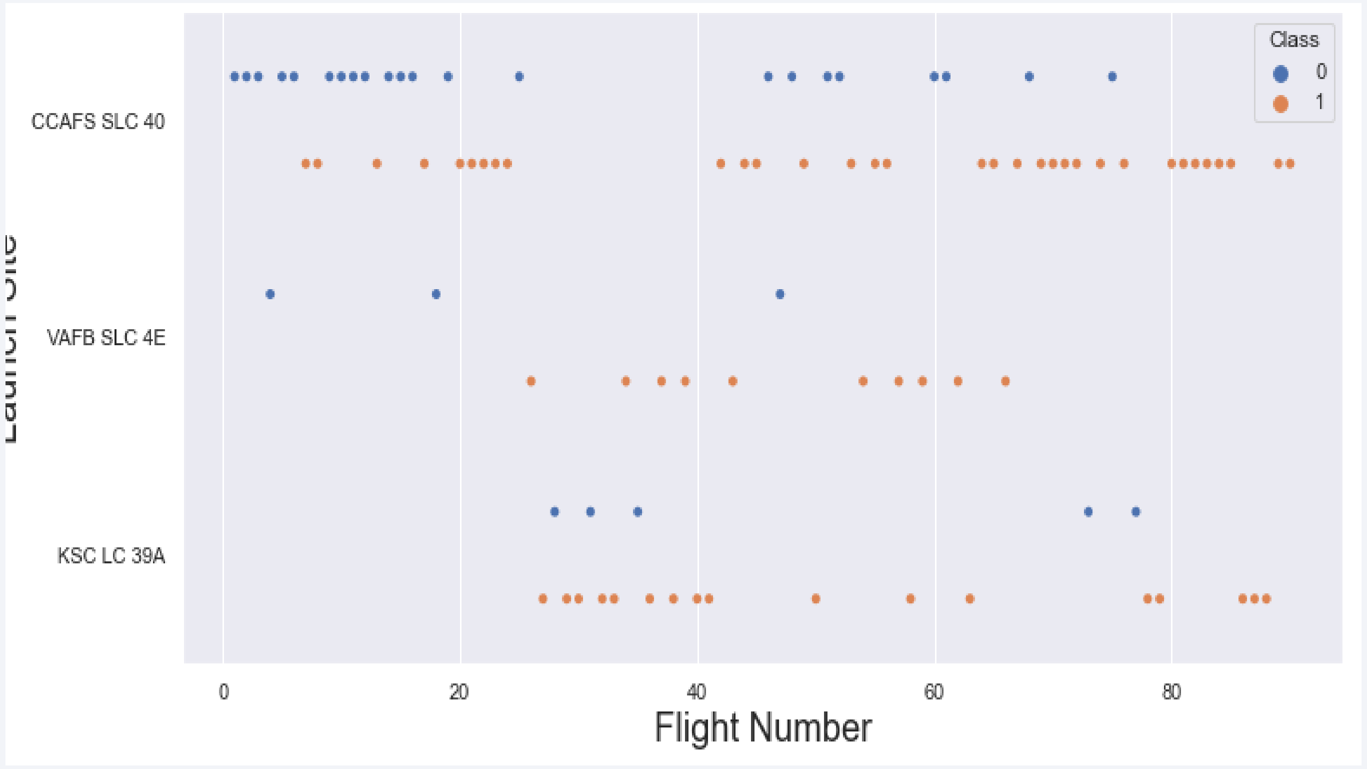
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

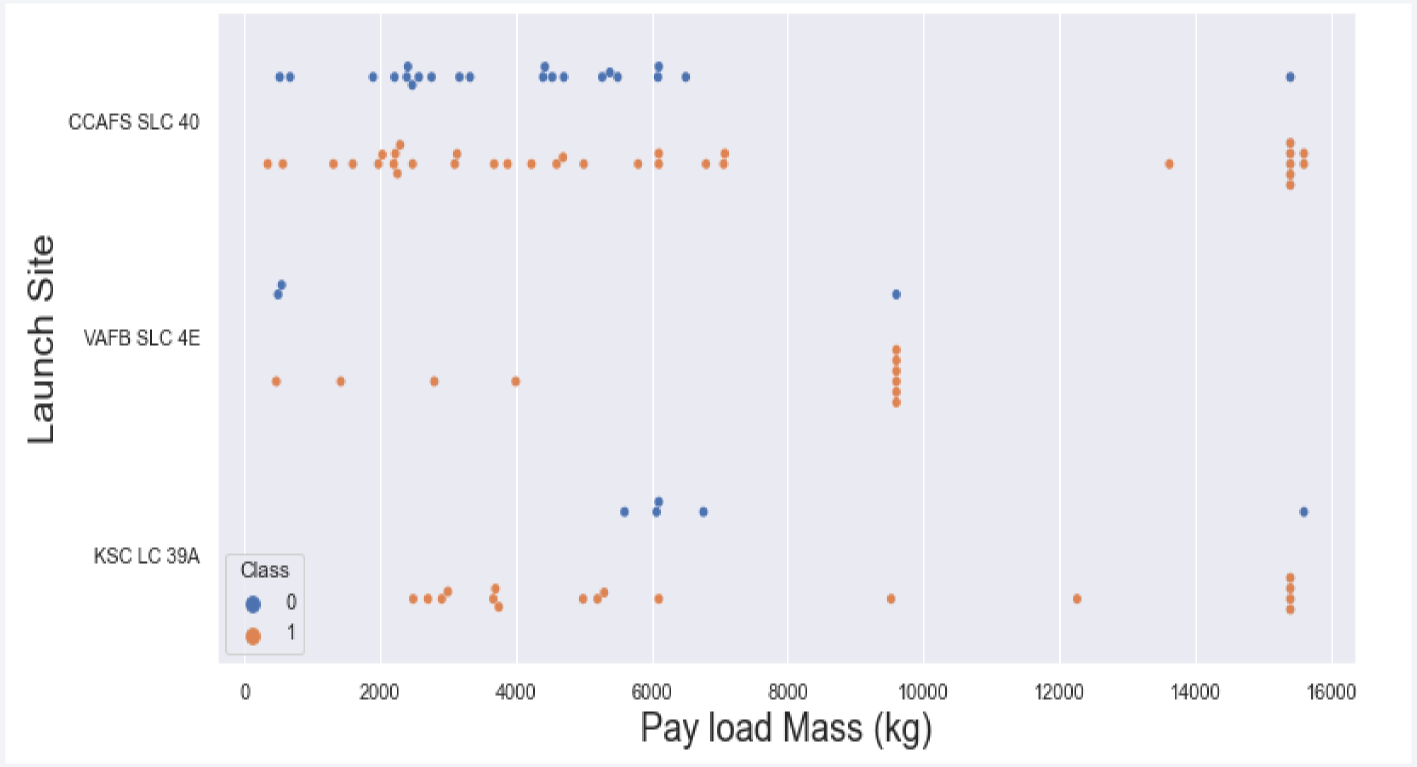
Flight Number vs. Launch Site

With Higher flite Number (> 30) success rate for rocket is increasing



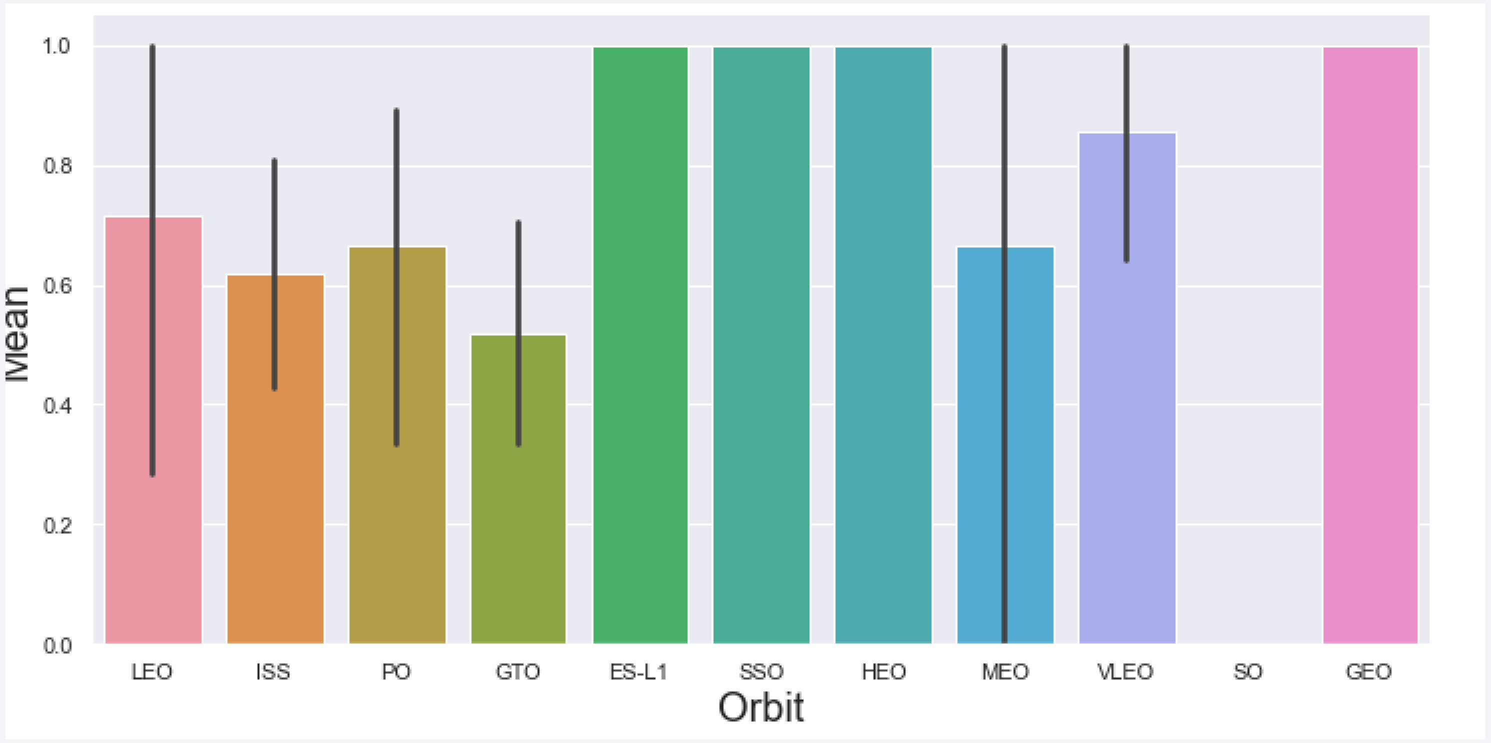
Payload vs. Launch Site

Greater he payload mass higher the success rate. But no clear pattern to take decision.



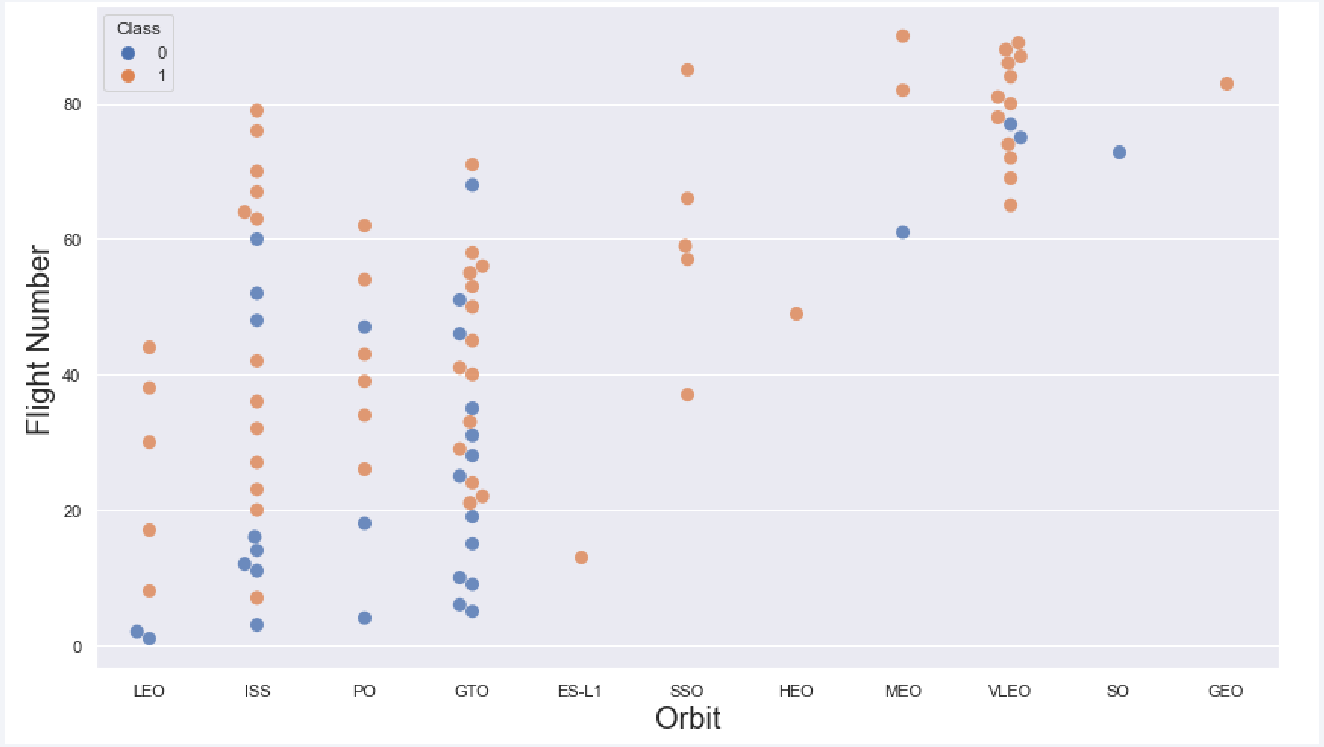
Success Rate vs. Orbit Type

- ES-L1, SSO, HEO, GEO have highest success rate.



Flight Number vs. Orbit Type

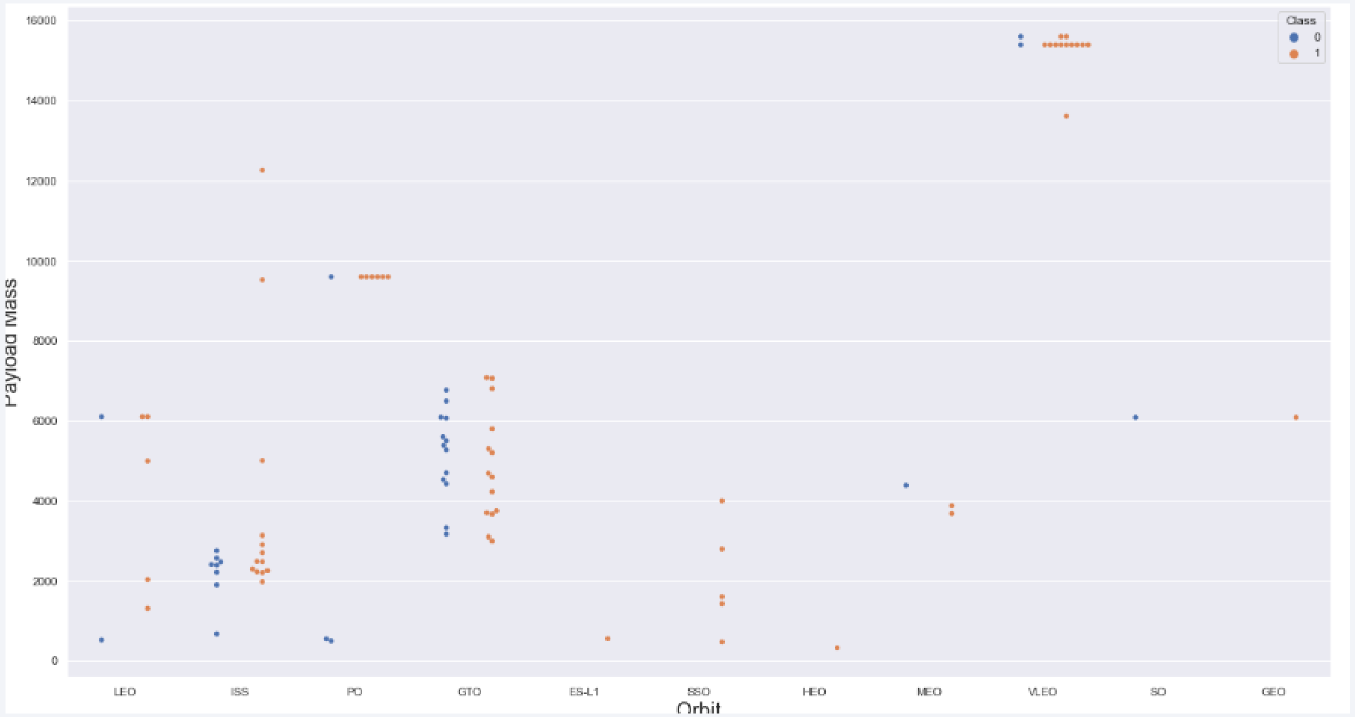
- Success rate increases for LEO orbit with number of flights



Payload vs. Orbit Type

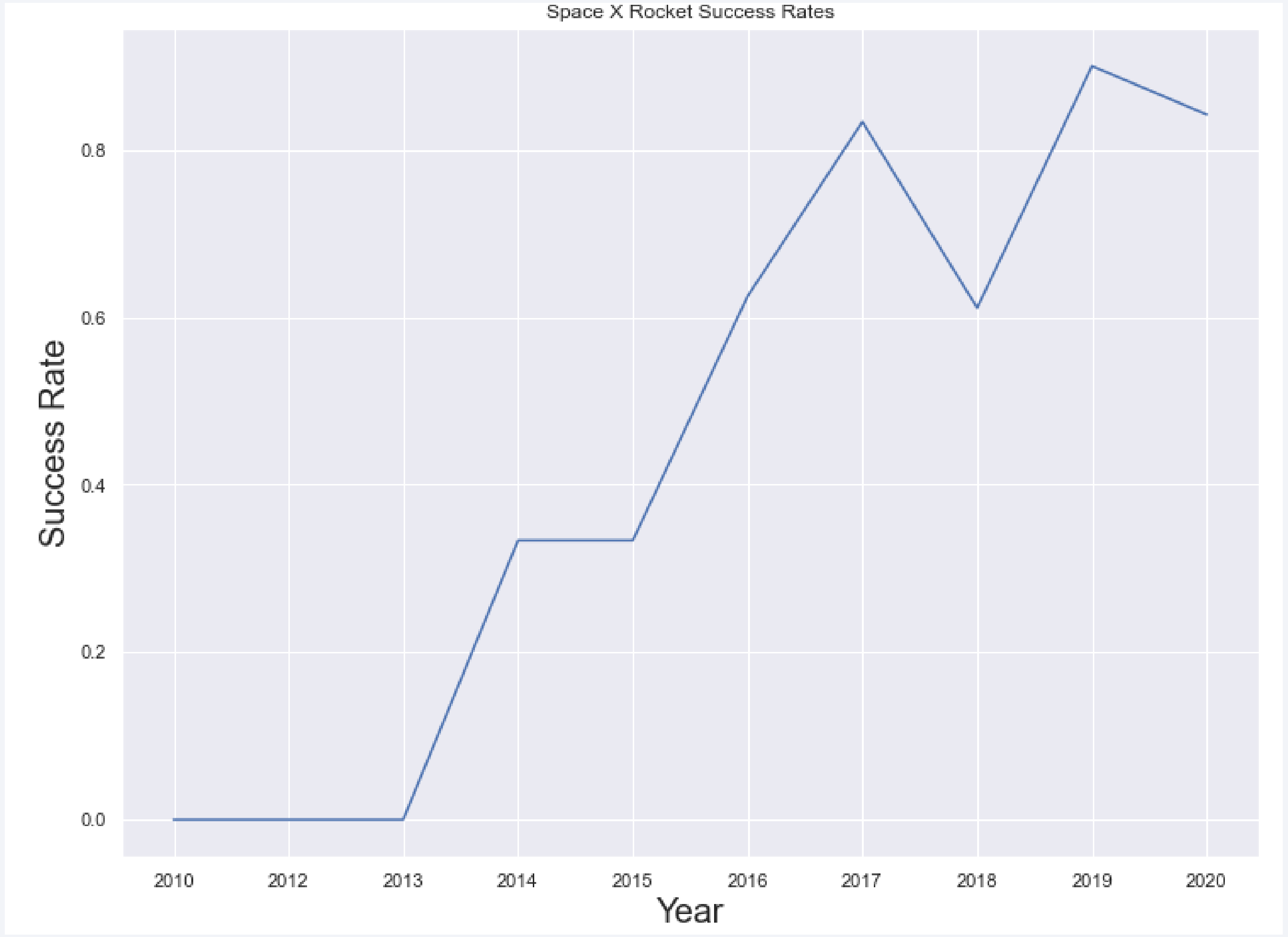
- Heavy payloads have a negative influence on MEO, GTO, VLEO

- Heavy payloads have a positive impact on LEO and ISS



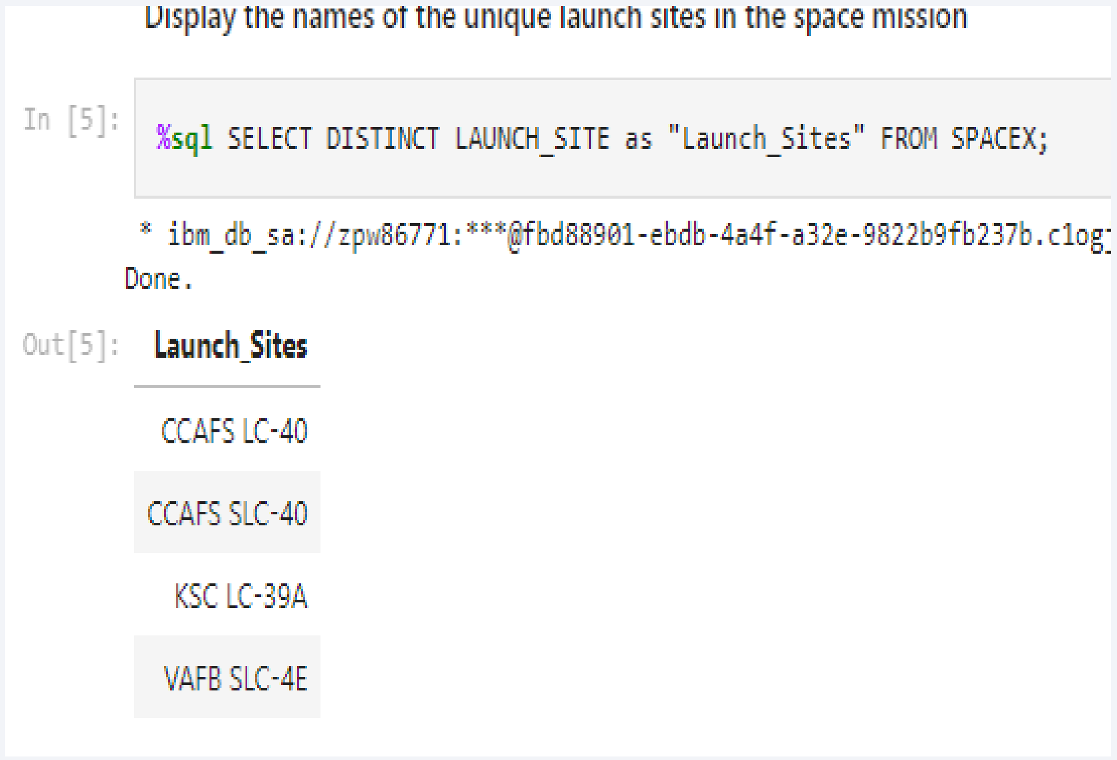
Launch Success Yearly Trend

- Success rate has been increasing since 2013 with a small dip in 2018.



All Launch Site Names

This query retrieves a list of unique launch sites from the SpaceX dataset. It helps in identifying the different locations SpaceX uses for launching rockets.



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

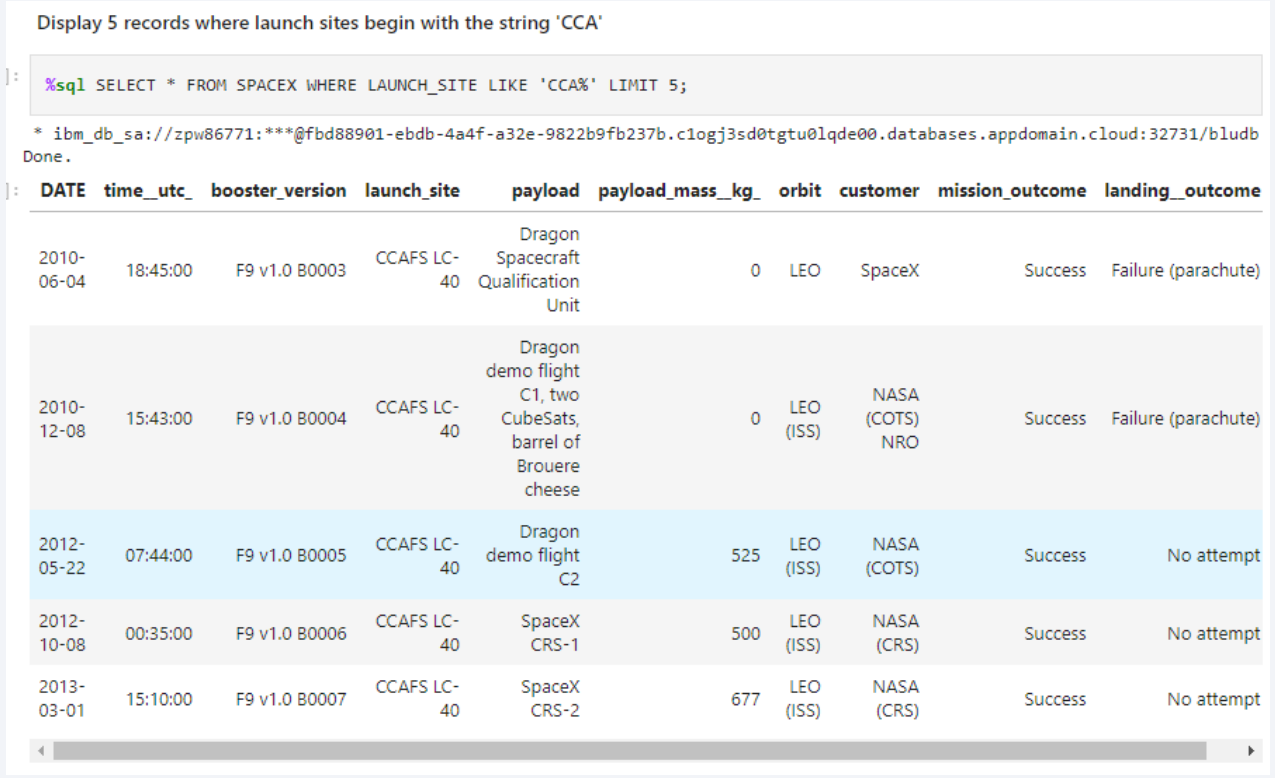
```
In [5]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;

 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1og
Done.
```

Out[5]:

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

Launch Site Names Begin with 'CCA'

- This query filters the records to show only those launch sites that start with 'CCA' (likely referring to Cape Canaveral sites) and limits the output to 5 records.



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

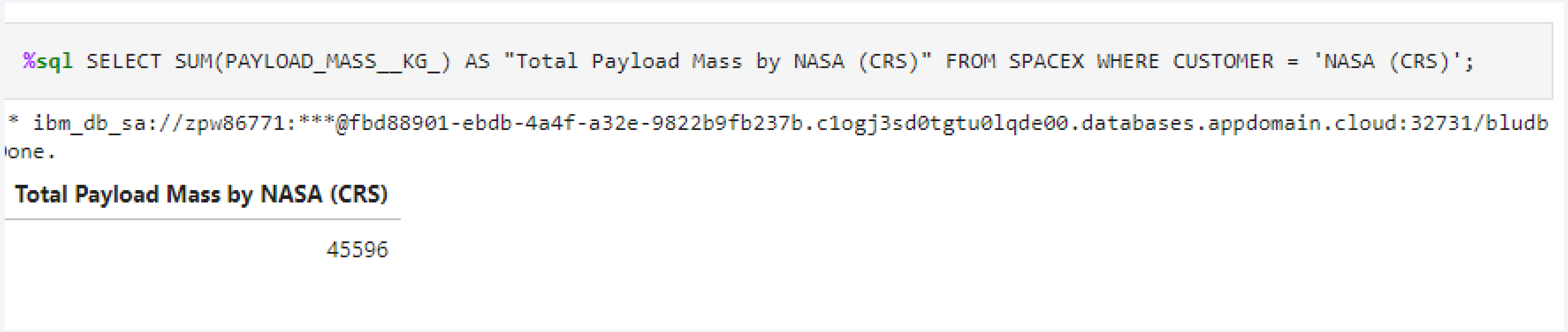
```
%sql SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

Total Payload Mass

- This query calculates the total payload mass (in kg) for missions where the customer is NASA (CRS). It helps understand the total cargo delivered by SpaceX for NASA's CRS missions.



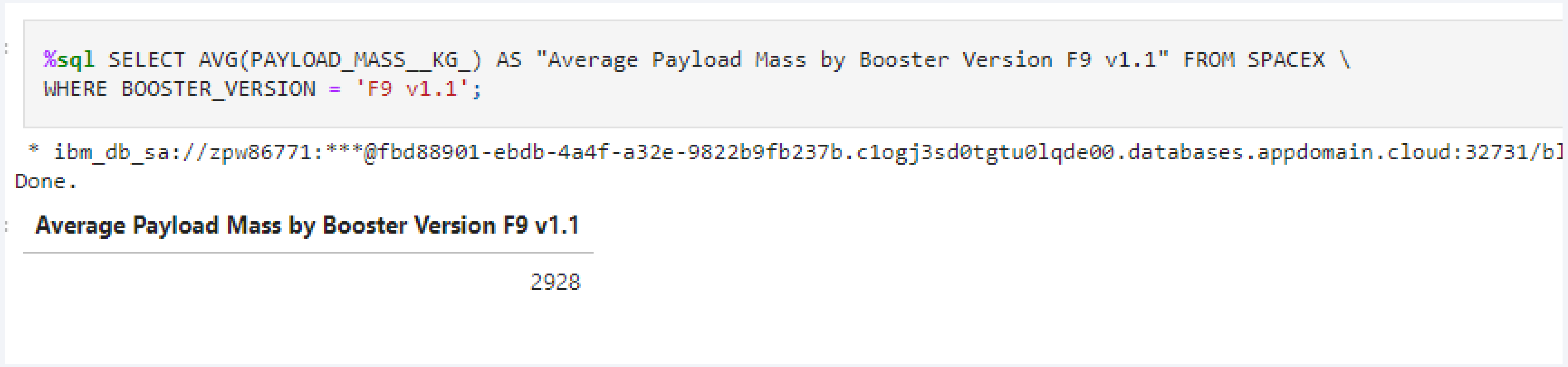
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)';
```

```
* ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
one.
```

Total Payload Mass by NASA (CRS)
45596

Average Payload Mass by F9 v1.1

- This query computes the average payload mass for missions using the F9 v1.1 booster version. It helps evaluate the payload capacity of this specific booster model.



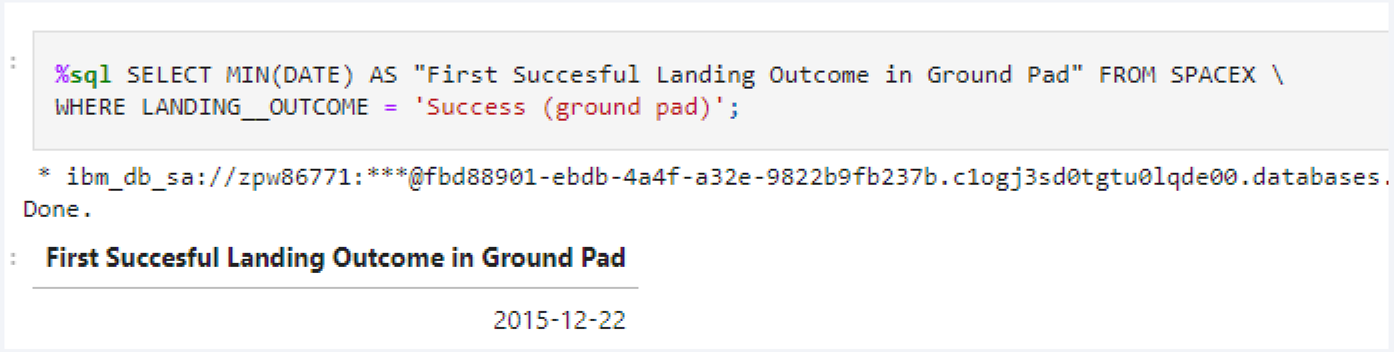
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster Version F9 v1.1" FROM SPACEX \
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bl
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

First Successful Ground Landing Date

- This query identifies the date of the first successful landing on a ground pad by SpaceX. It helps track SpaceX's progress in achieving successful landings on solid ground.



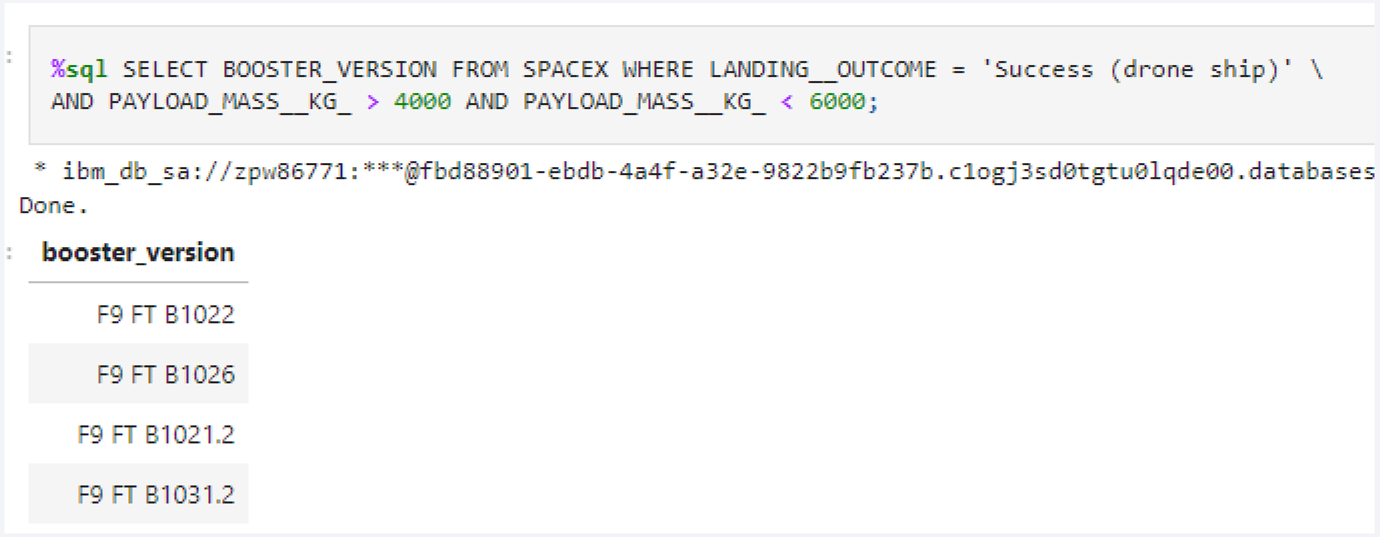
```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad" FROM SPACEX \
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- This query lists the booster versions that successfully landed on a drone ship and carried a payload between 4000 and 6000 kg, helping evaluate performance for specific mission profiles.



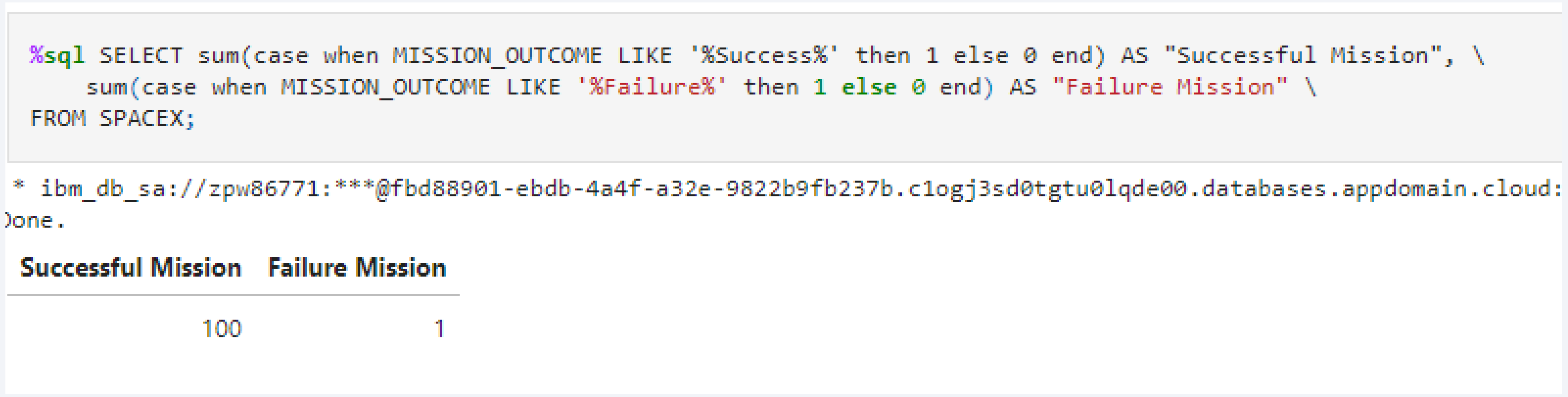
```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

- This query counts the number of successful missions in the SpaceX dataset, allowing a quick overview of the company's mission success rate.



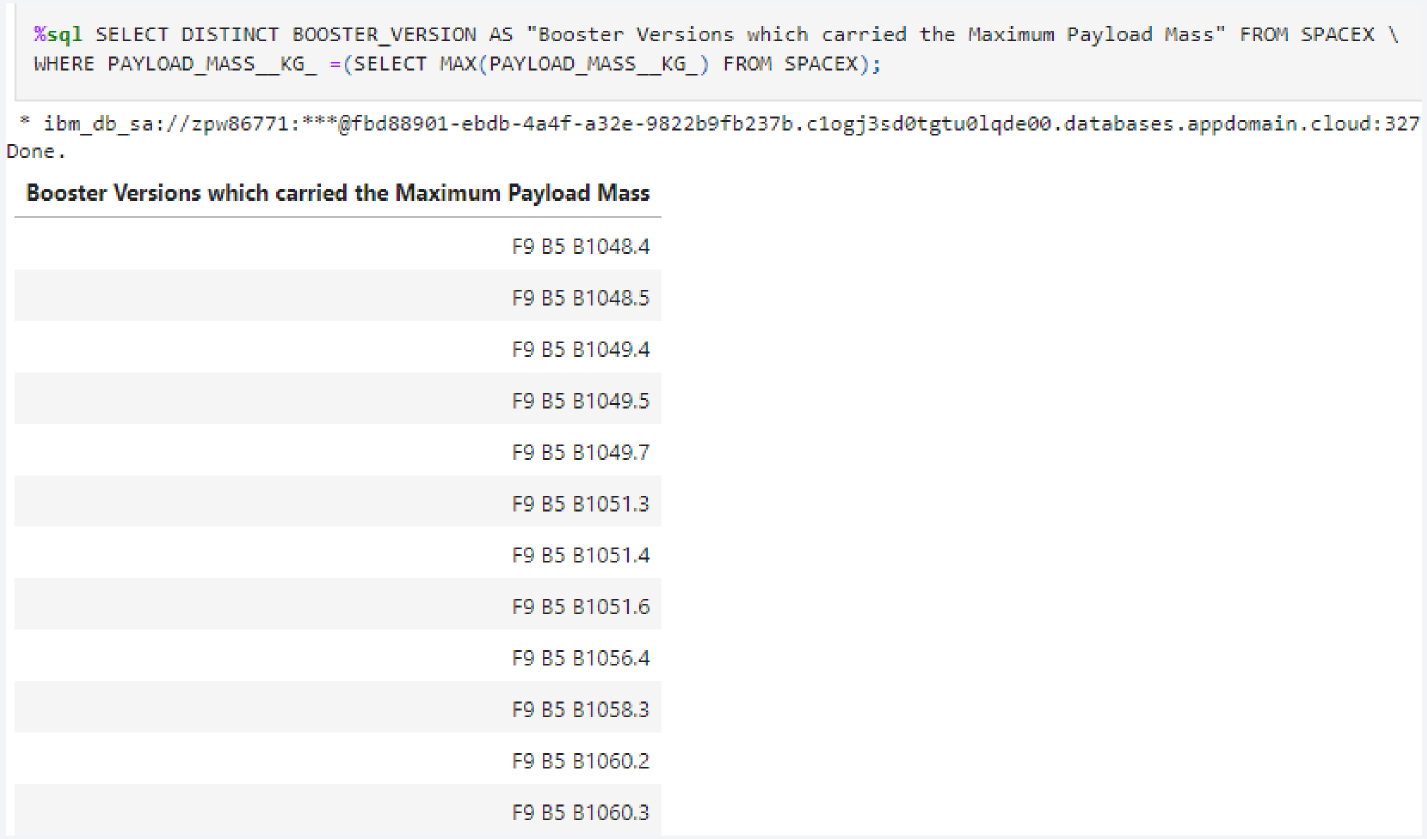
```
%sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Mission", \
    sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission" \
FROM SPACEX;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:
Done.
```

Successful Mission	Failure Mission
100	1

Boosters Carried Maximum Payload

- This query identifies the booster versions that carried the maximum payload mass. It helps in pinpointing which booster model had the highest load capacity during SpaceX missions.



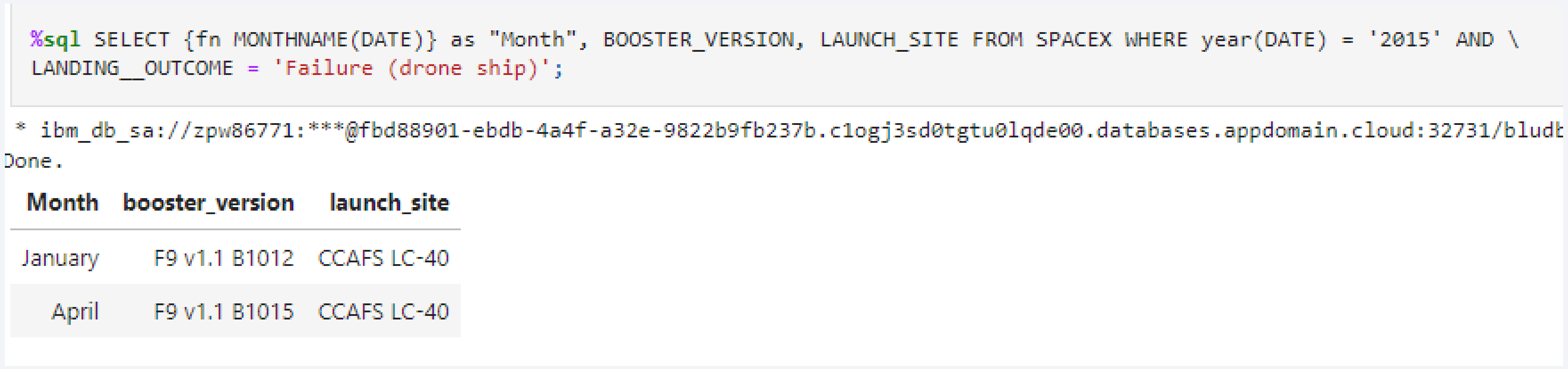
```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:327
Done.
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

- This query retrieves the booster versions and launch sites for missions in 2015 where the landing outcome on a drone ship was a failure. It provides insights into which boosters and sites were involved in unsuccessful drone ship landings during that year.



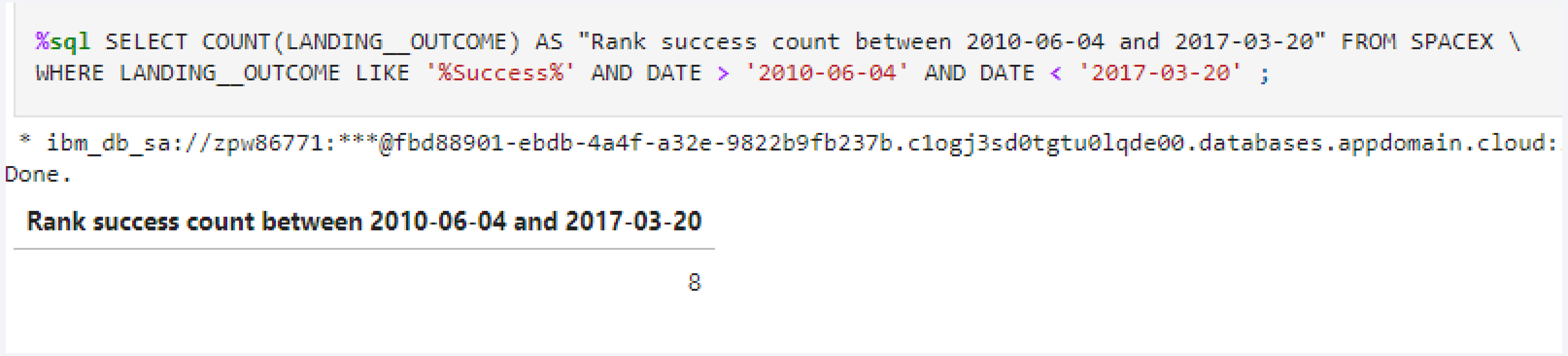
```
%sql SELECT {fn MONTHNAME(DATE)} as "Month", BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE year(DATE) = '2015' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Month	booster_version	launch_site
January	F9 v1.1 B1012	CCAFS LC-40
April	F9 v1.1 B1015	CCAFS LC-40

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

- This query ranks the landing outcomes by the number of occurrences between June 4, 2010, and March 20, 2017. It provides insight into the frequency of different landing outcomes (such as success or failure) over this time period and helps in analyzing SpaceX's landing success rate over the years.



```
%sql SELECT COUNT(LANDING__OUTCOME) AS "Rank success count between 2010-06-04 and 2017-03-20" FROM SPACEX \
WHERE LANDING__OUTCOME LIKE '%Success%' AND DATE > '2010-06-04' AND DATE < '2017-03-20' ;
```

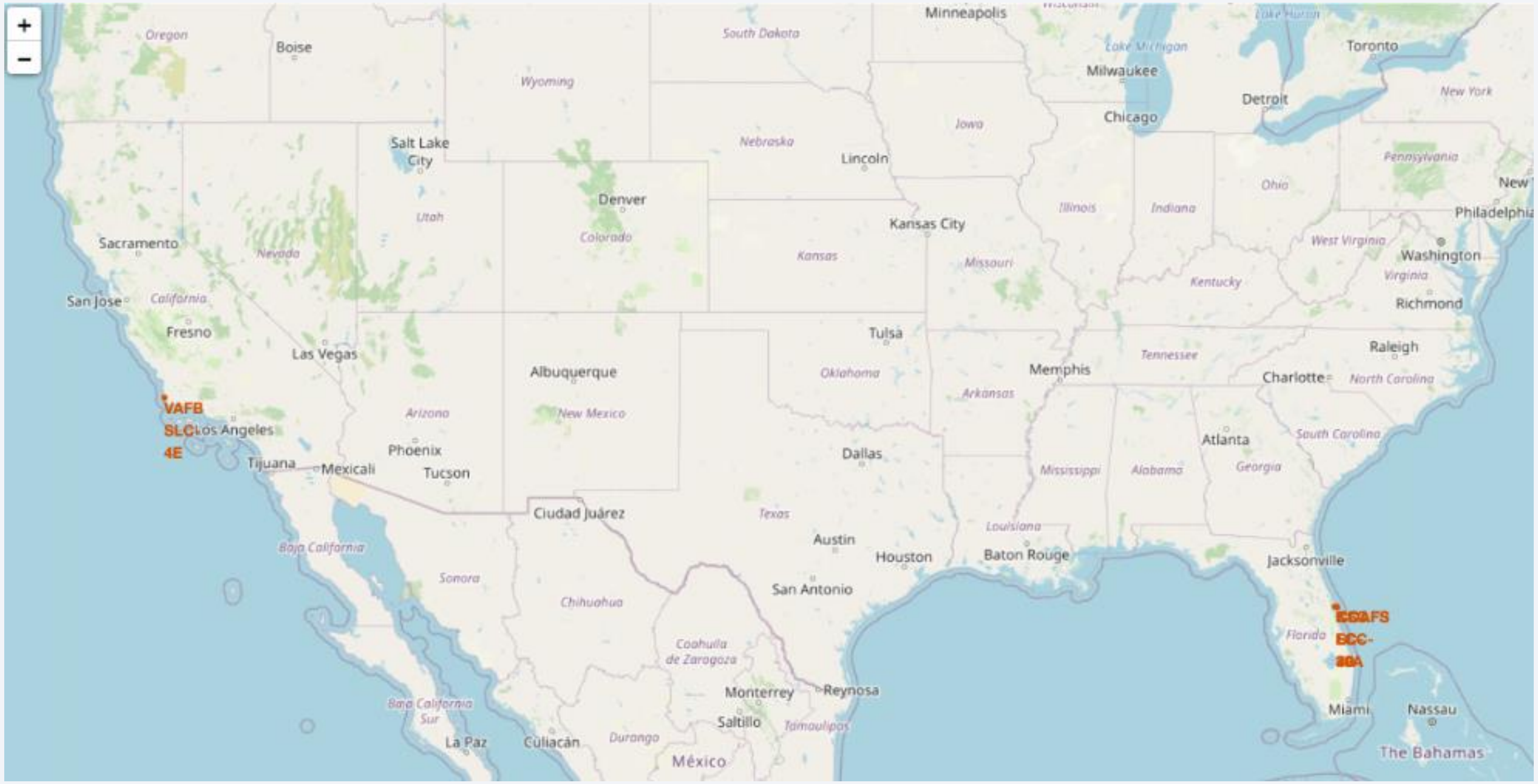
```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:
Done.
```

Rank success count between 2010-06-04 and 2017-03-20
8

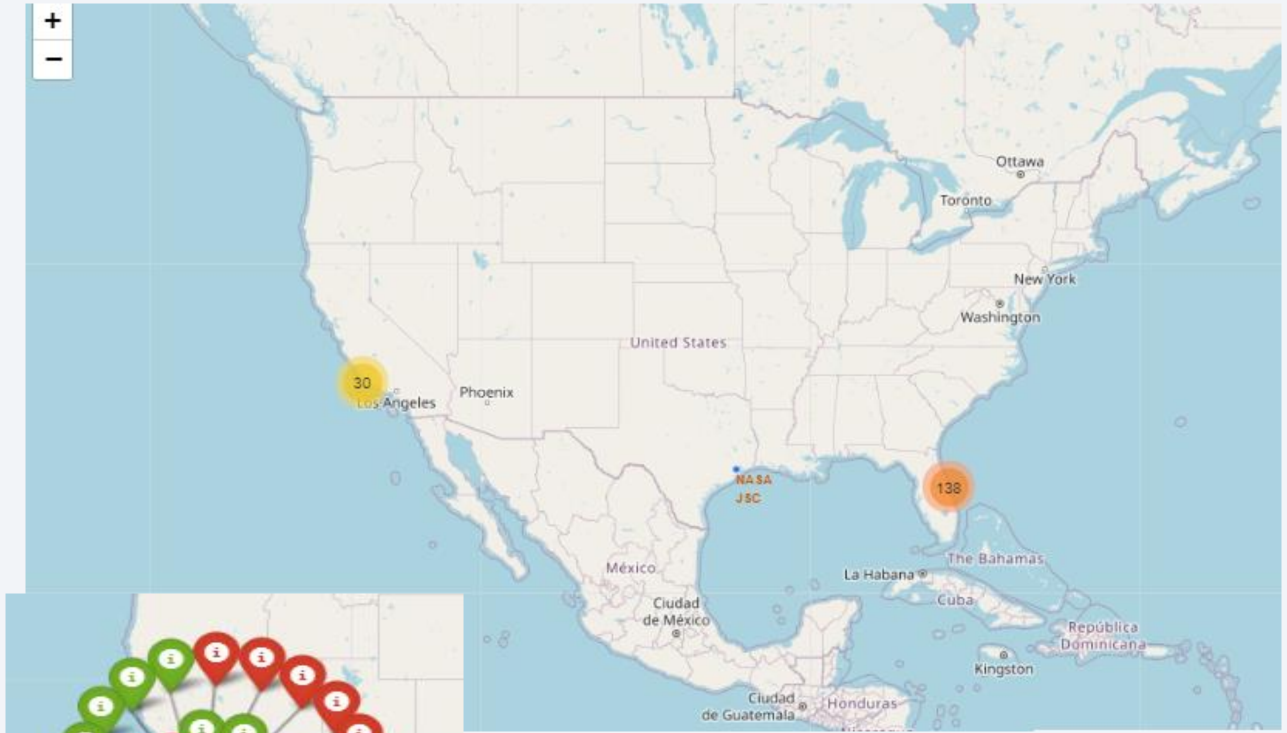
Section 3

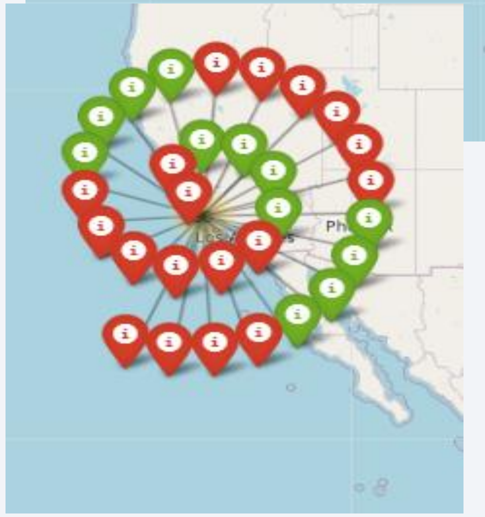
Launch Sites Proximities Analysis

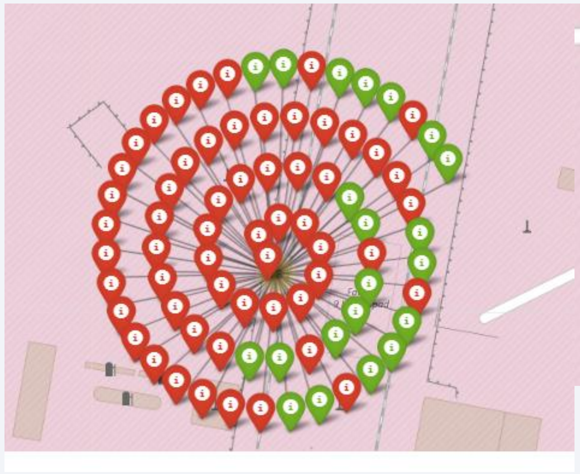
All launch sites on a map

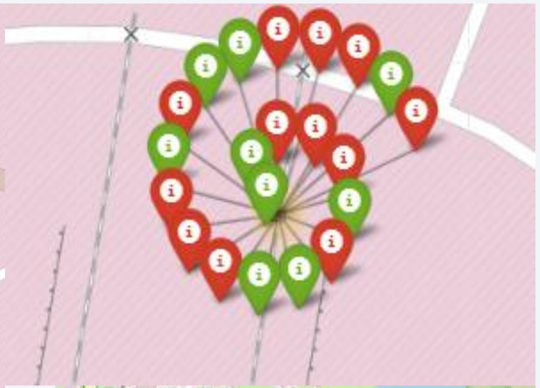


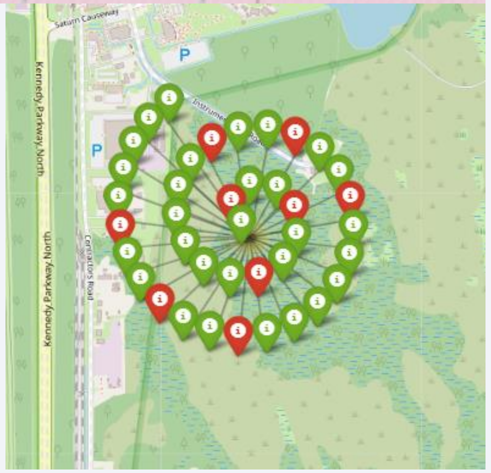
The success/failed launches for each site on the map



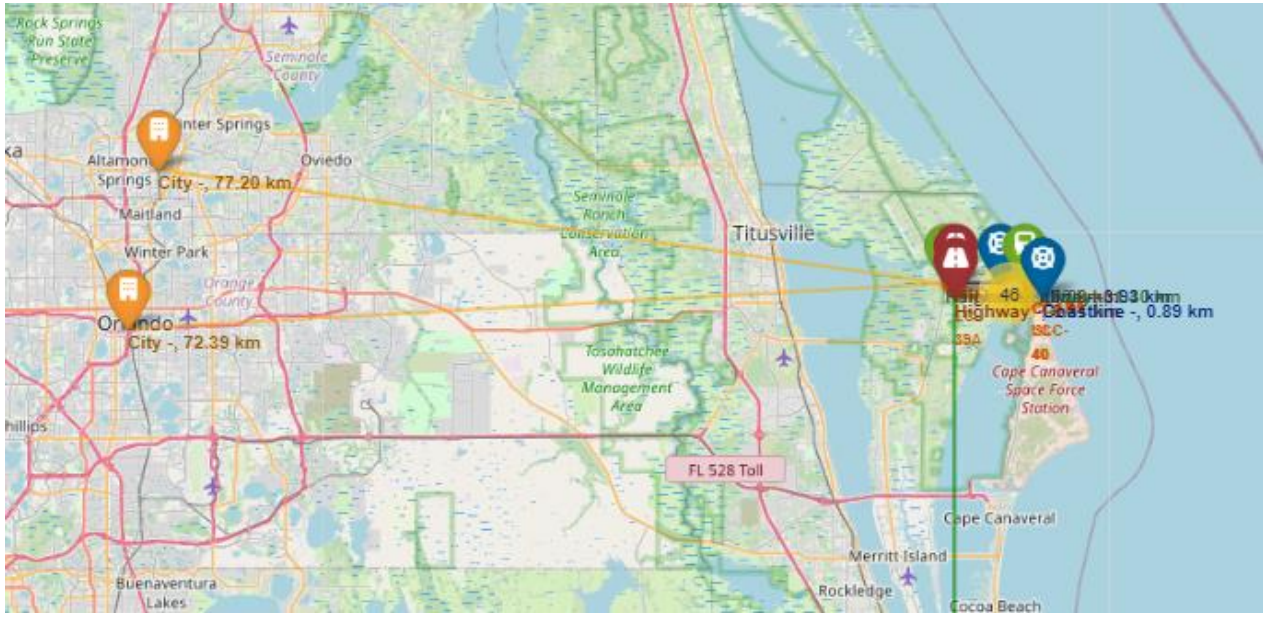


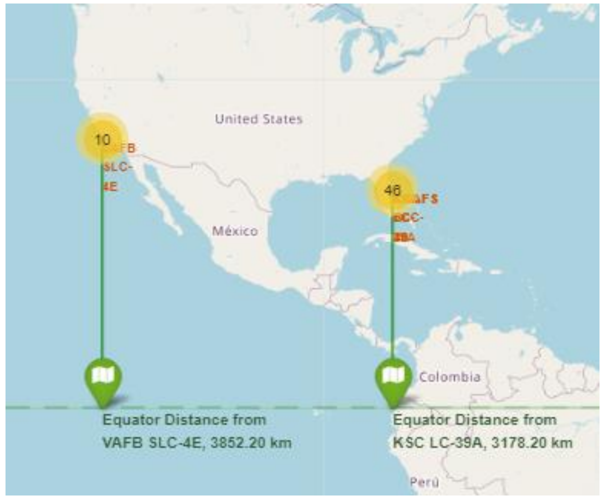




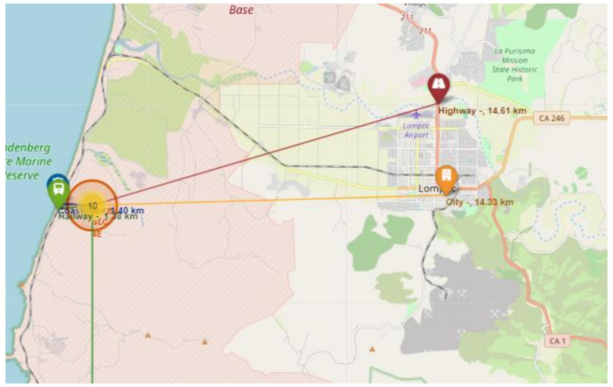


Launch Site Distances



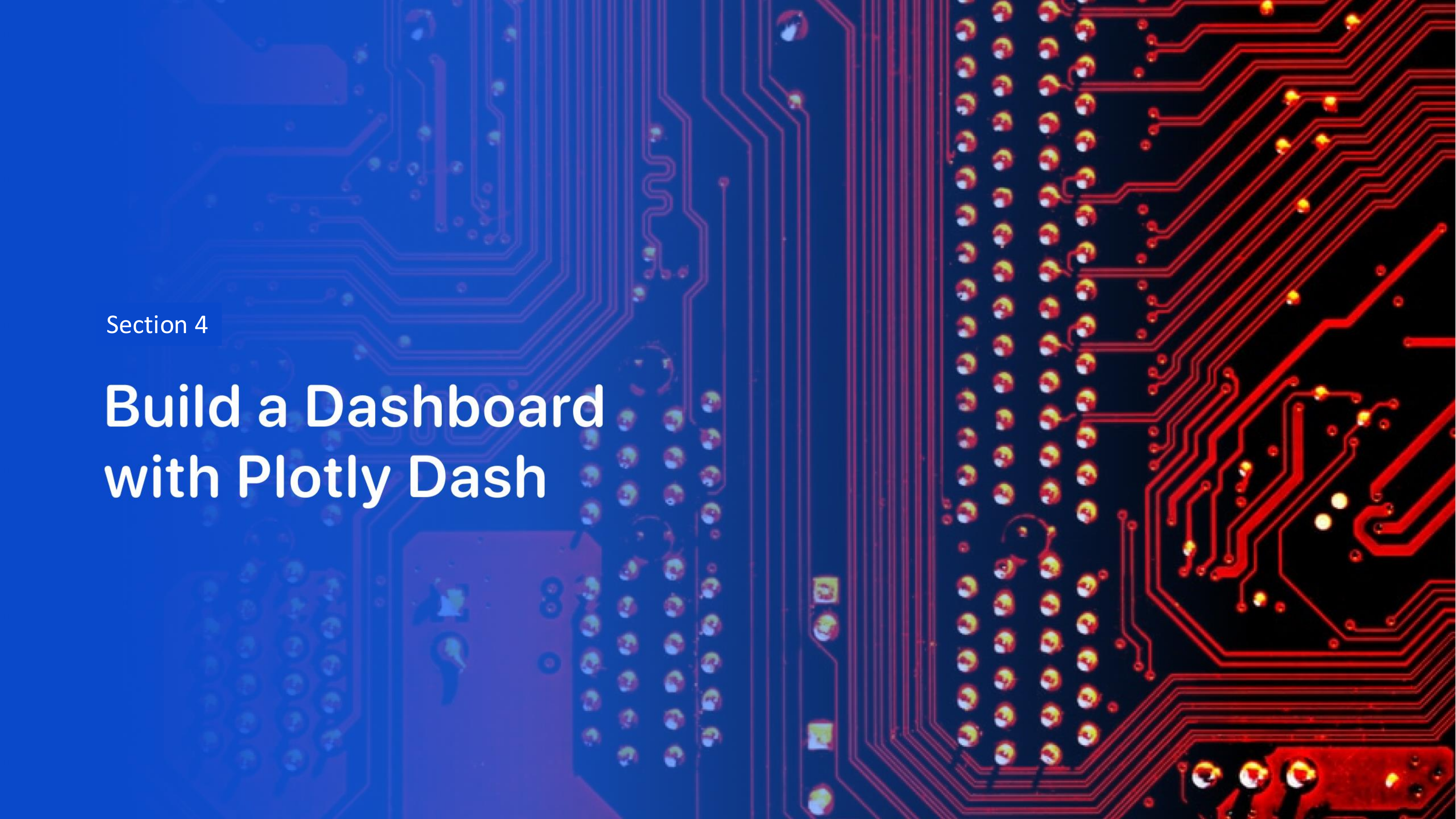


United States
México
Colombia
Equator Distance from VAFB SLC-4E, 3852.20 km
Equator Distance from KSC LC-39A, 3178.20 km

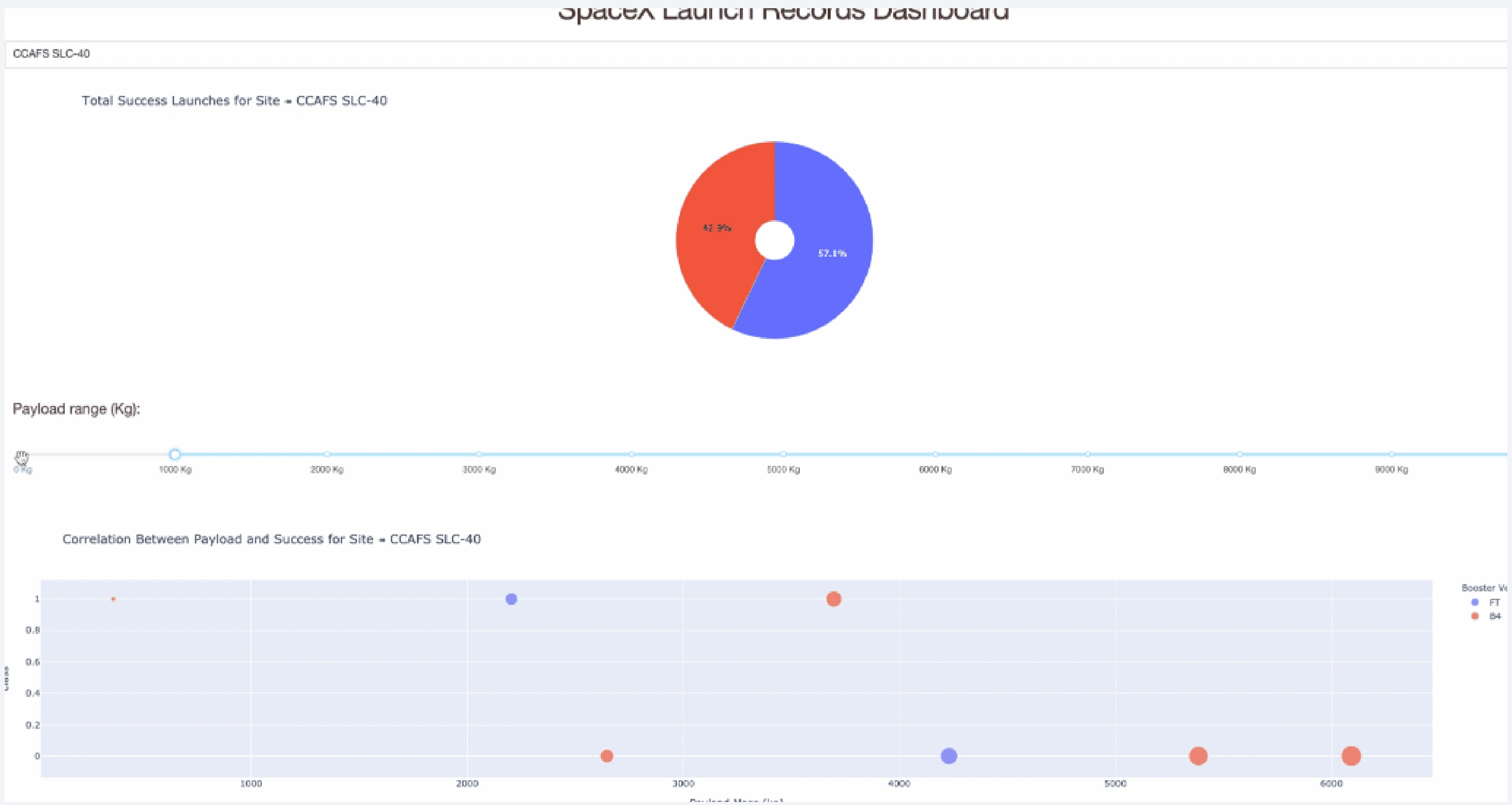


Section 4

Build a Dashboard with Plotly Dash



Dashboard Screenshot



SpaceX Launch Records Dashboard

CCAFS SLC-40

Total Success Launches for Site = CCAFS SLC-40

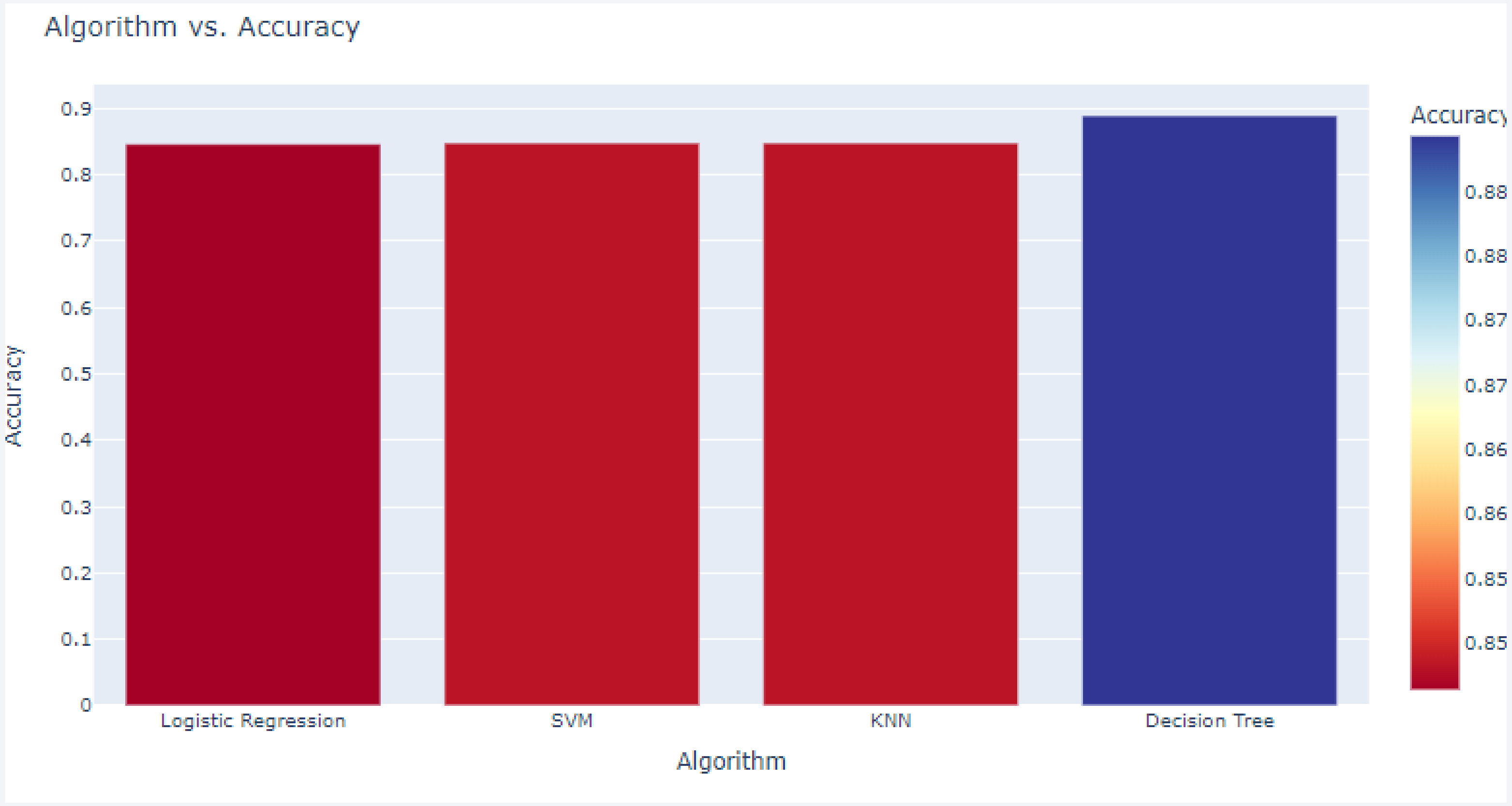
Payload range (Kg):

Correlation Between Payload and Success for Site = CCAFS SLC-40

Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix

Key Metrics

From the confusion matrix, several important metrics can be derived:

- **Accuracy**:
  - Measures the overall correctness of the model.

  Accuracy=TP+TNTP+TN+FP+FN\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}Accuracy=TP+TN+FP+FNTP+TN
- **Precision**:
  - Indicates the accuracy of positive predictions.

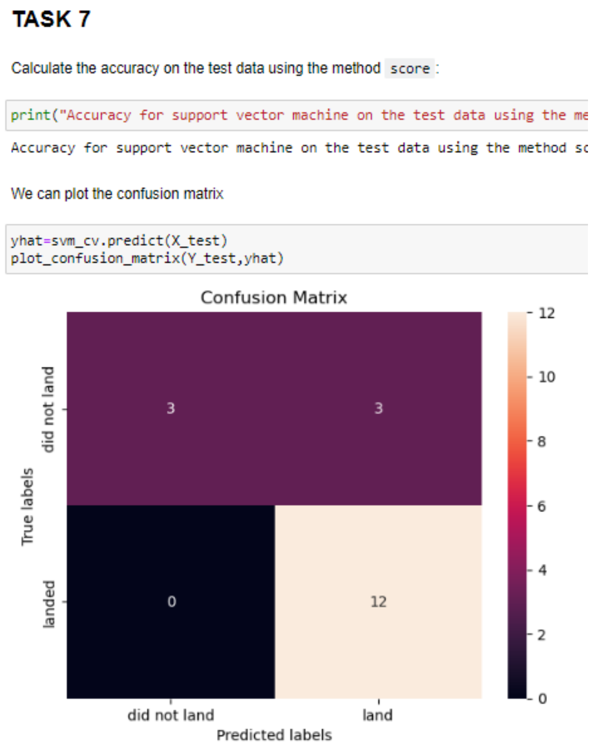
  Precision=TPTP+FP\text{Precision} = \frac{TP}{TP + FP}Precision=TP+FPTP
- **Recall (Sensitivity)**:
  - Measures the ability to find all positive instances.

  Recall=TPTP+FN\text{Recall} = \frac{TP}{TP + FN}Recall=TP+FNTP
- **F1 Score**:
  - Combines precision and recall into a single metric.

  F1 Score=2×Precision×RecallPrecision+RecallF1 \text{ Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}F1 Score=2×Precision+RecallPrecision×Recall
- **Specificity**:
  - Measures the ability to find all negative instances.

  Specificity=TNTN+FP\text{Specificity} = \frac{TN}{TN + FP}Specificity=TN+FPTN

Confusion Matrix



TASK 7

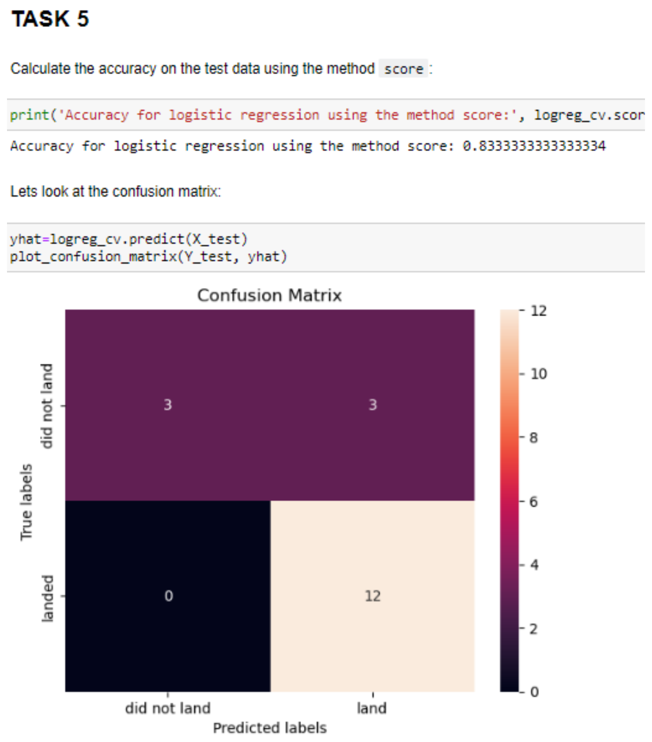
Calculate the accuracy on the test data using the method `score` :

```
print("Accuracy for support vector machine on the test data using the me
```

```
Accuracy for support vector machine on the test data using the method sc
```

We can plot the confusion matrix

```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



TASK 5

Calculate the accuracy on the test data using the method `score` :

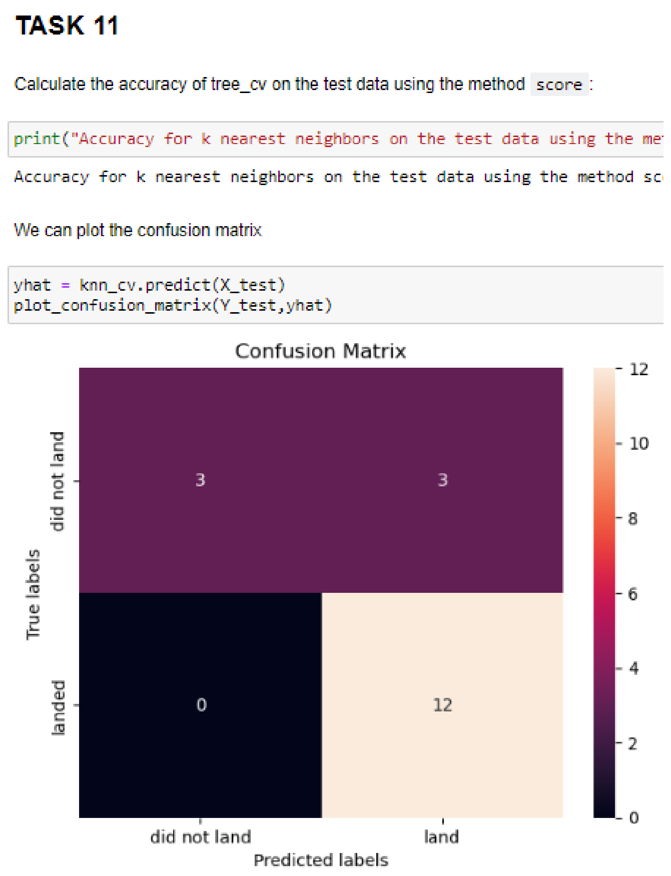
```
print('Accuracy for logistic regression using the method score:', logreg_cv.scor
```

```
Accuracy for logistic regression using the method score: 0.8333333333333334
```

Lets look at the confusion matrix:

```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test, yhat)
```

Confusion Matrix



TASK 11

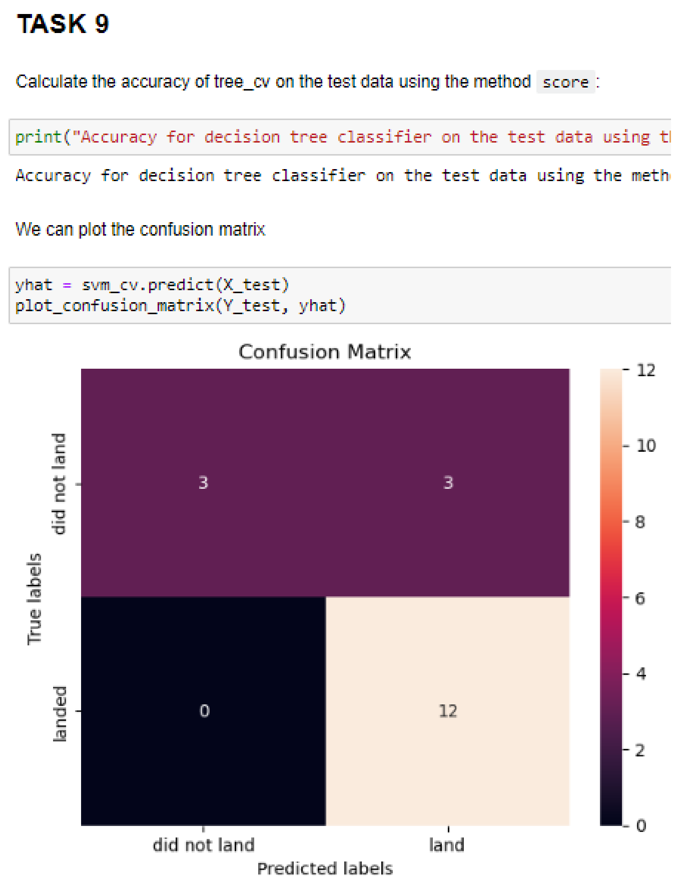
Calculate the accuracy of tree_cv on the test data using the method `score` :

```
print("Accuracy for k nearest neighbors on the test data using the me
```

```
Accuracy for k nearest neighbors on the test data using the method sc
```

We can plot the confusion matrix

```
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



TASK 9

Calculate the accuracy of tree_cv on the test data using the method `score` :

```
print("Accuracy for decision tree classifier on the test data using t
```

```
Accuracy for decision tree classifier on the test data using the meth
```

We can plot the confusion matrix

```
yhat = svm_cv.predict(X_test)
plot_confusion_matrix(Y_test, yhat)
```

Conclusions

1. Orbits ES-L1, GEO, HEO, SSO has highest Sucess rates
2. Success rates for SpaceX launches has been increasing relatively with time and it looks like soon they will reach the required target
3. KSC LC-39A had the most successful launches but increasing payload mass seems to have negative impact on success
4. Decision Tree Classifier Algorithm is the best for Machine Learning Model for provided dataset

Thank you!