SPACEX ROCKET LANDING PREDICTIVE ANALYSIS

(DATA SCIENCE AND MACHINE LEARNING CAPSTONE PROJECT)

EXECUTIVE SUMMARY

In this project, we embarked on a journey through the realm of data science, exploring various methodologies to analyze and derive insights from datasets. Our objective was to delve into the SpaceX Falcon 9 first stage landing prediction, a critical aspect of space exploration and cost optimization in the aerospace industry.

Key Findings:

- Leveraging machine learning algorithms, we successfully predicted the landing outcome of Falcon 9 first stages.
- Our analysis revealed significant cost savings potential by accurately predicting first stage landings.

Methodologies Used:

- Data collection and wrangling
- Exploratory data analysis (EDA) with interactive visual analytics
- Predictive analysis using machine learning algorithms

INTRODUCTION

Our journey began with a clear problem statement: to predict the landing outcome of Falcon 9 first stages. With the rise of reusable rocket technology, accurate landing predictions are crucial for cost-effective space missions. We aimed to explore datasets related to SpaceX Falcon 9 launches and landings to address this challenge.

DATA COLLECTION AND DATA WRANGLING METHODOLOGY

Data Collection:

- We obtained datasets from reliable sources, including SpaceX and NASA, containing information on Falcon 9 launches and landing outcomes.

Data Wrangling:

- We cleaned and preprocessed the data, handling missing values and ensuring data consistency.

Challenges such as inconsistent formatting and data discrepancies were addressed during this phase.

EDA AND INTERACTIVE VISUAL ANALYTICS METHODOLOGY

Exploratory Data Analysis (EDA):

- Utilizing descriptive statistics and data visualization techniques, we gained insights into the characteristics and patterns of Falcon 9 launch and landing data.

Interactive Visual Analytics:

- Leveraging tools like Matplotlib and Seaborn, we created interactive visualizations to explore the data dynamically.
- These visualizations enabled deeper exploration and understanding of the dataset.

PREDICTIVE ANALYSIS METHODOLOGY

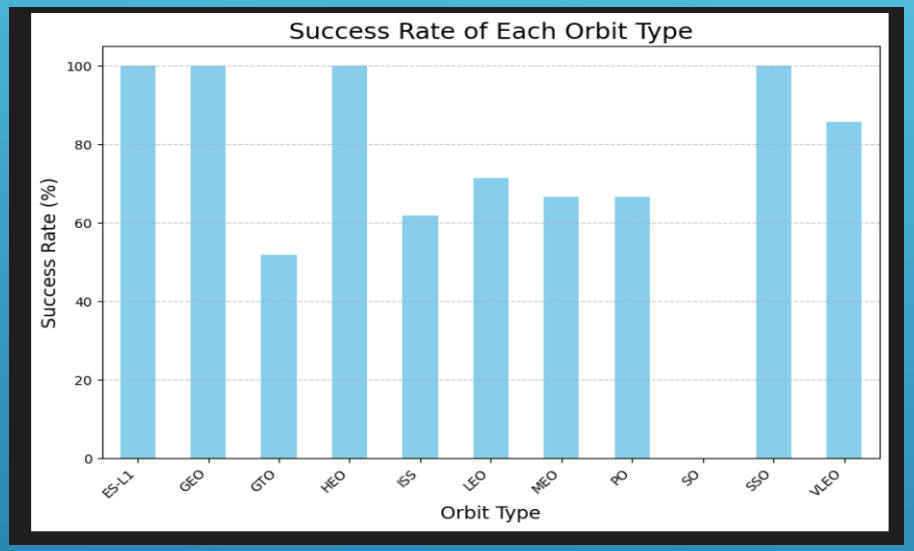
Predictive Analysis:

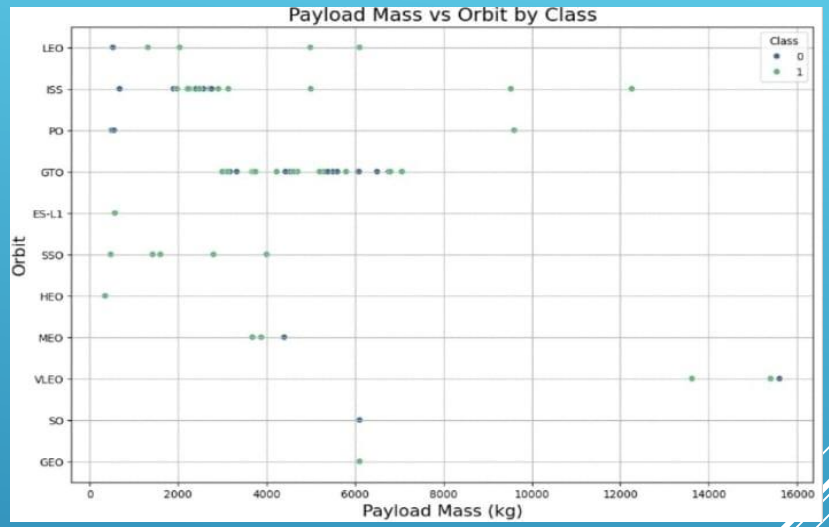
- We applied machine learning algorithms, including logistic regression, support vector machines (SVM), decision trees, and k-nearest neighbors (KNN), to predict Falcon 9 landing outcomes.
- The dataset was split into training and testing sets for model training and evaluation.

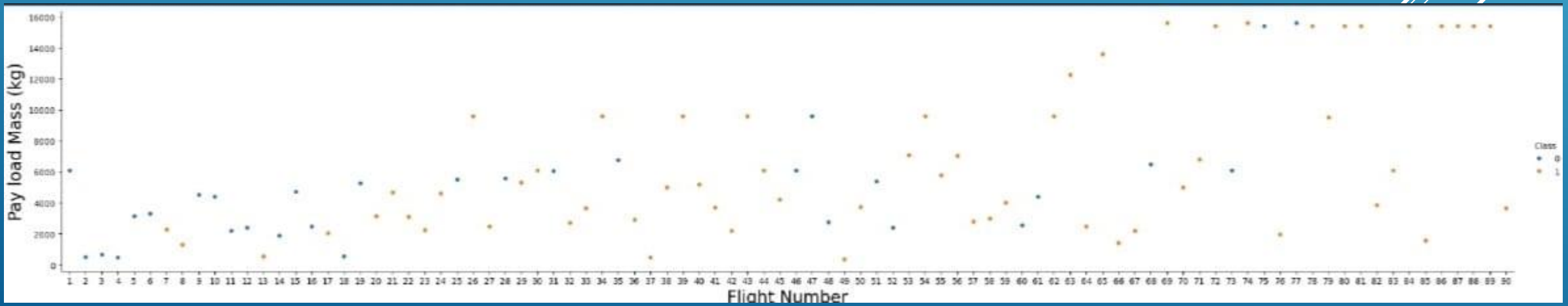
Model Evaluation:

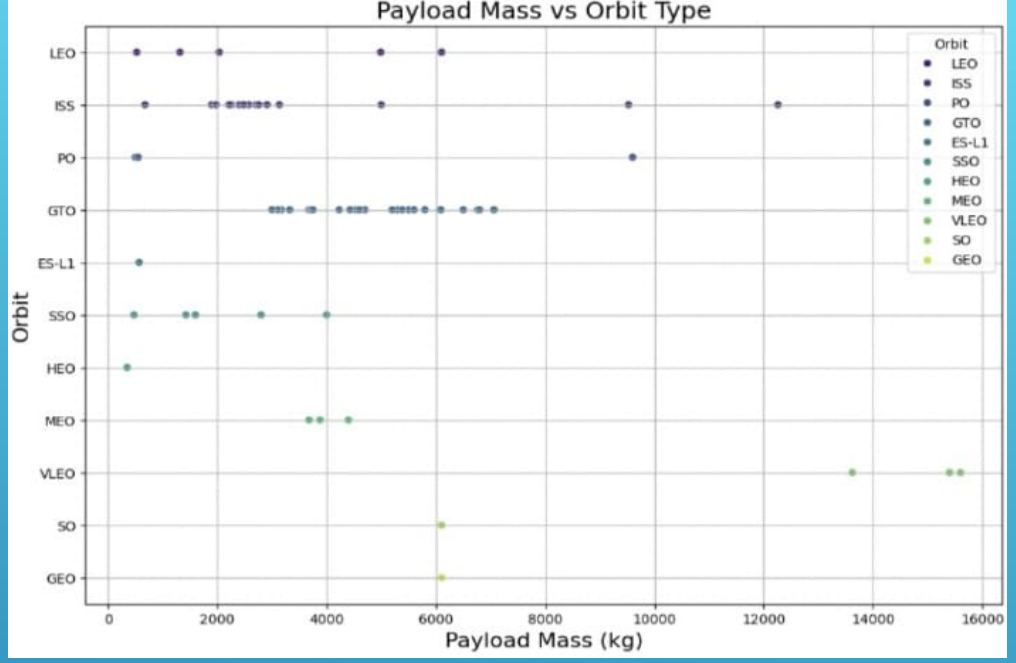
- Performance metrics such as accuracy, precision, recall, and F1 score were used to evaluate the predictive models.

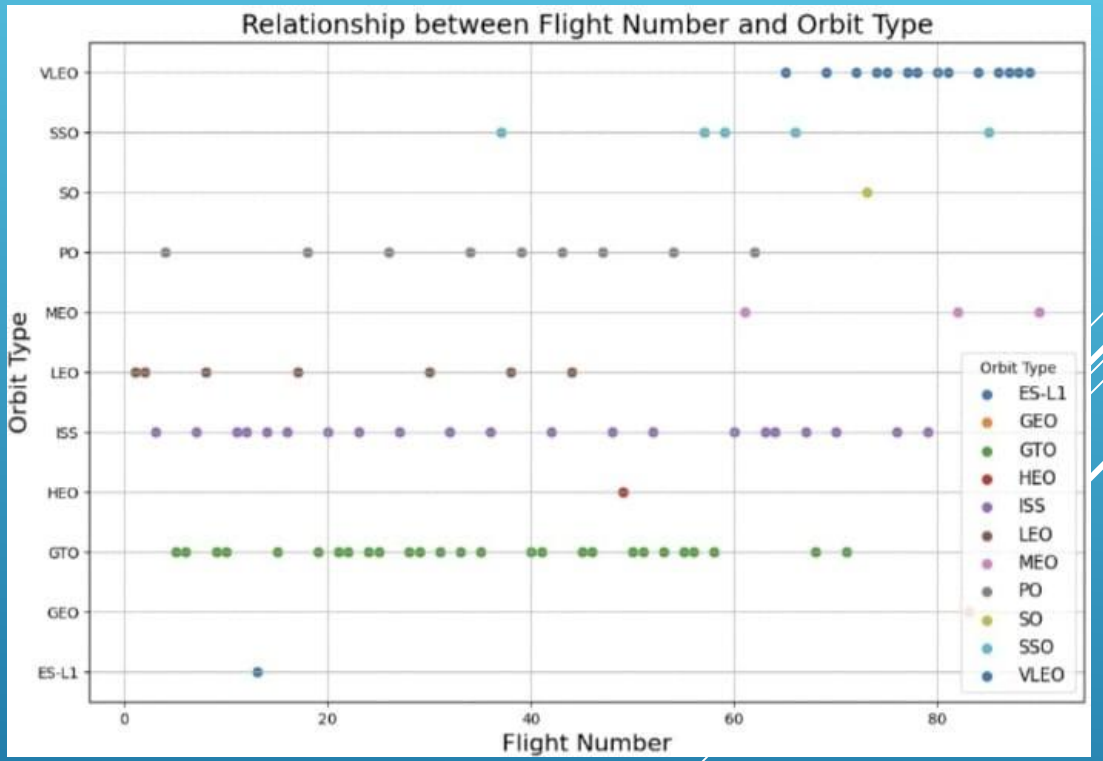
EDA WITH VISUALIZATION RESULTS:

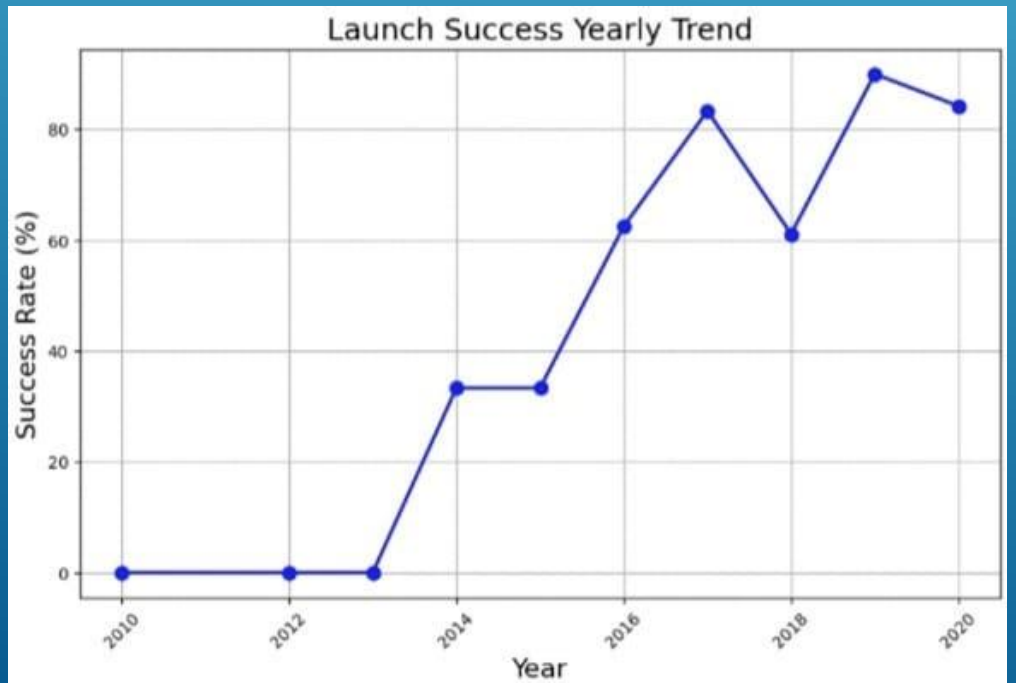


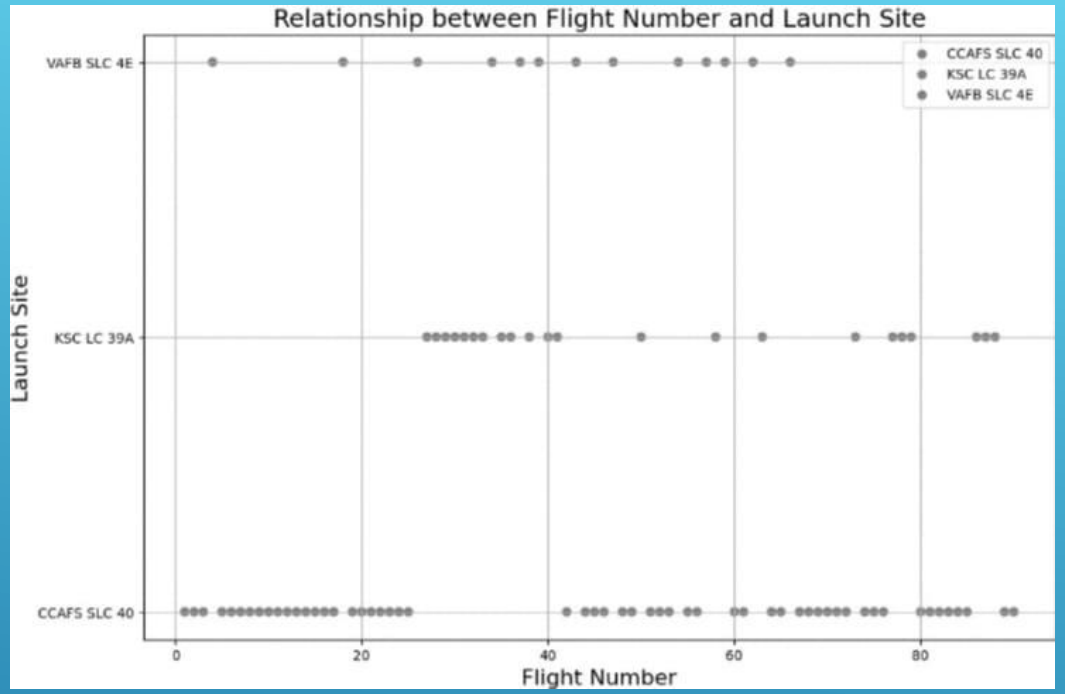


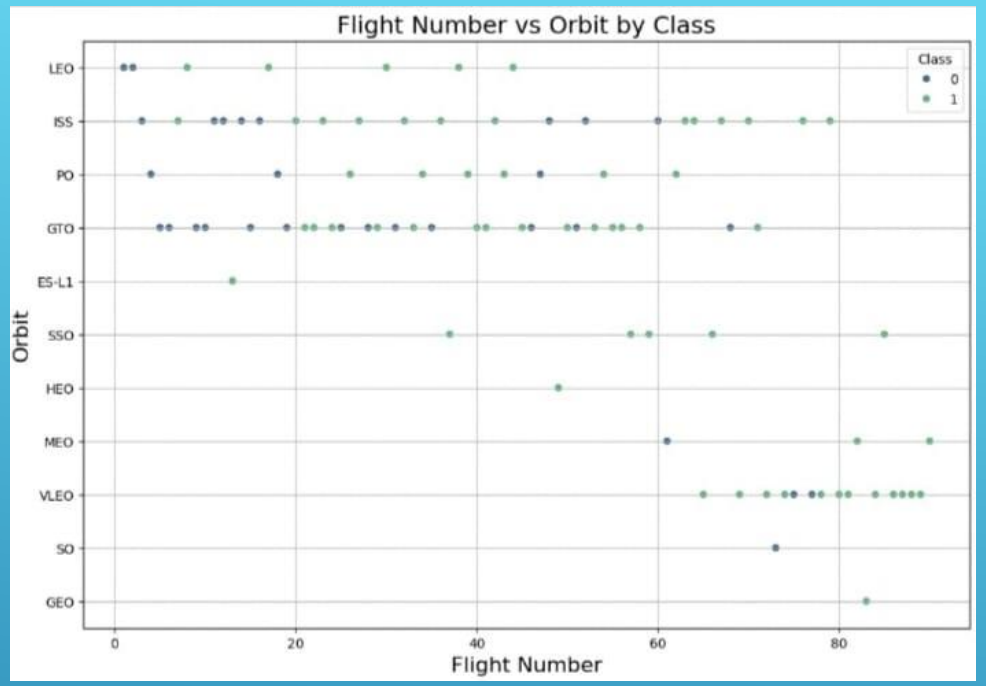


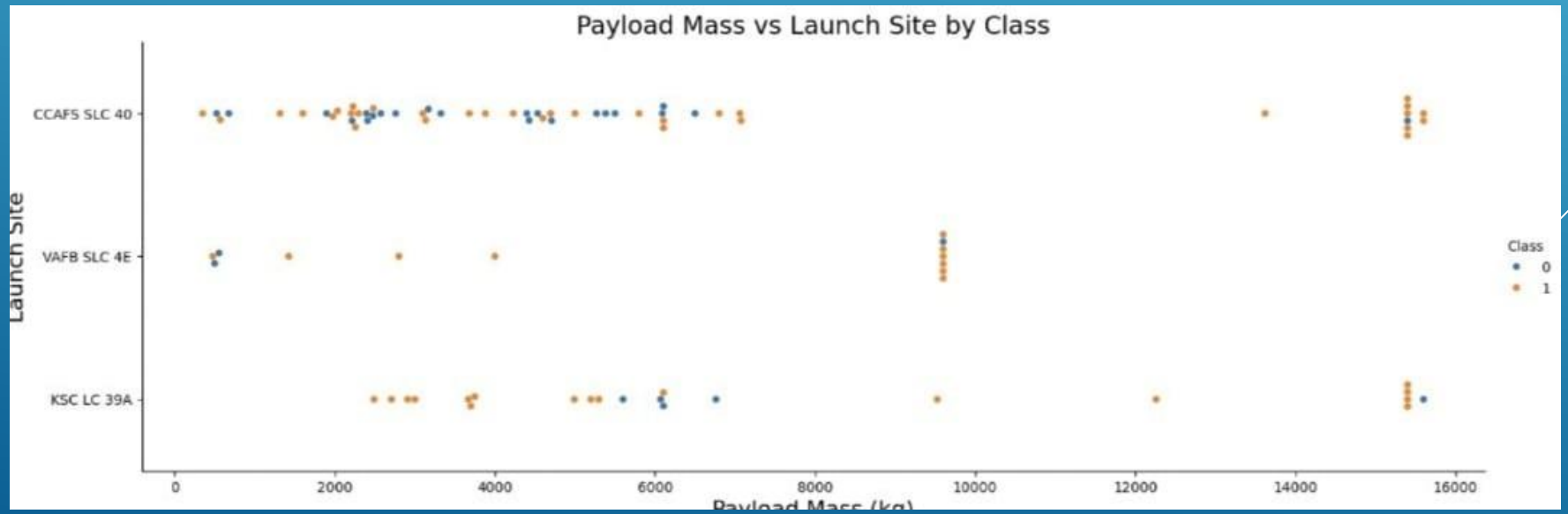

Payload Mass vs Orbit Type
Orbit
LEO
ISS
PO
GTO
ES-L1
SSO
HEO
MEO
VLEO
SO
GEO
0
2000
4000
6000
8000
10000
12000
14000
16000
Payload Mass (kg)

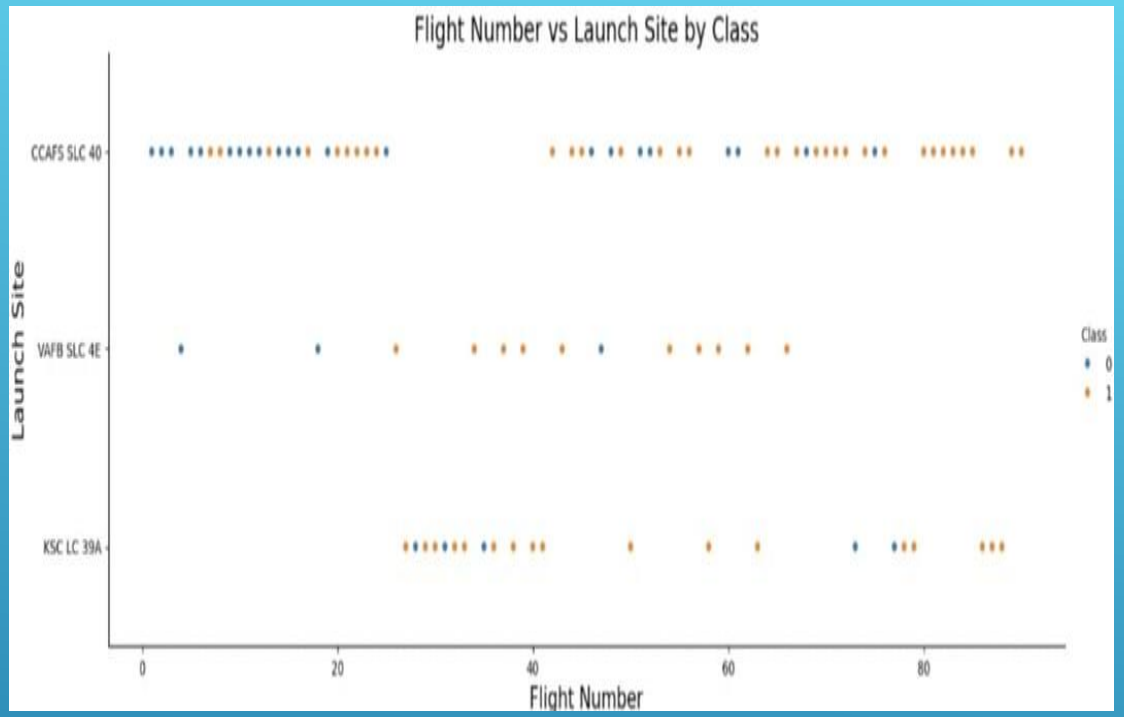

Relationship between Flight Number and Orbit Type
Orbit Type
VLEO
SSO
SO
PO
MEO
LEO
ISS
HEO
GTO
GEO
ES-L1
0
20
40
60
80
Flight Number

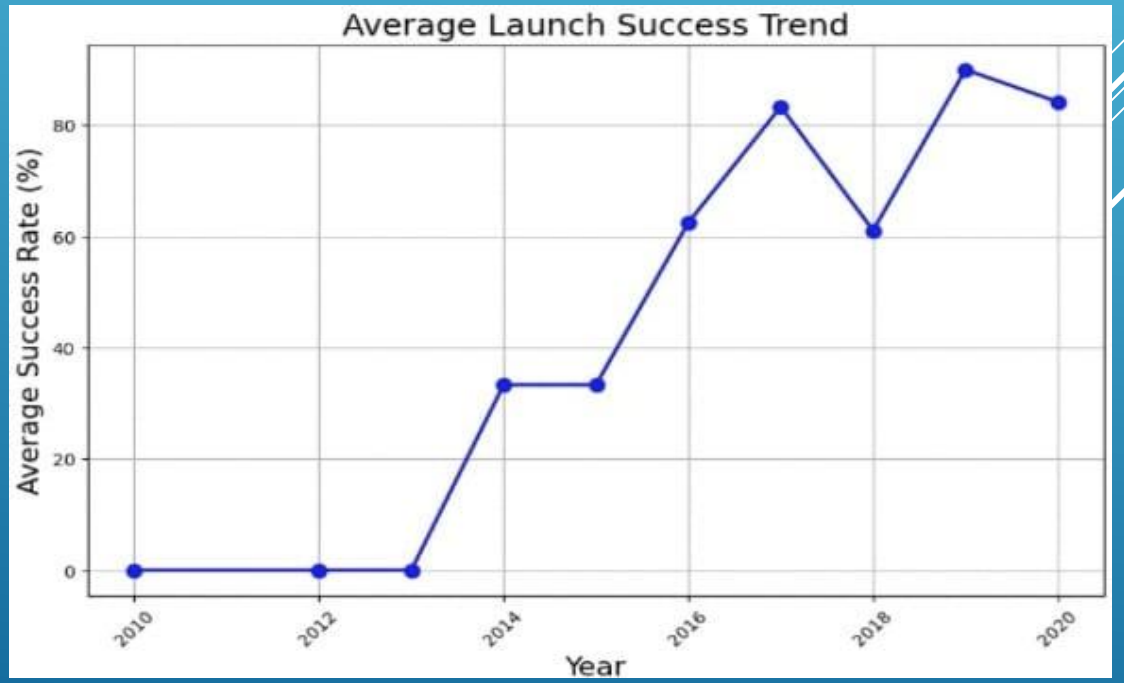

Launch Success Yearly Trend
Success Rate (%)
0
20
40
60
80
2010
2012
2014
2016
2018
2020
Year


Relationship between Flight Number and Launch Site
CCAFS SLC 40
KSC LC 39A
VAFB SLC 4E
Launch Site
Flight Number

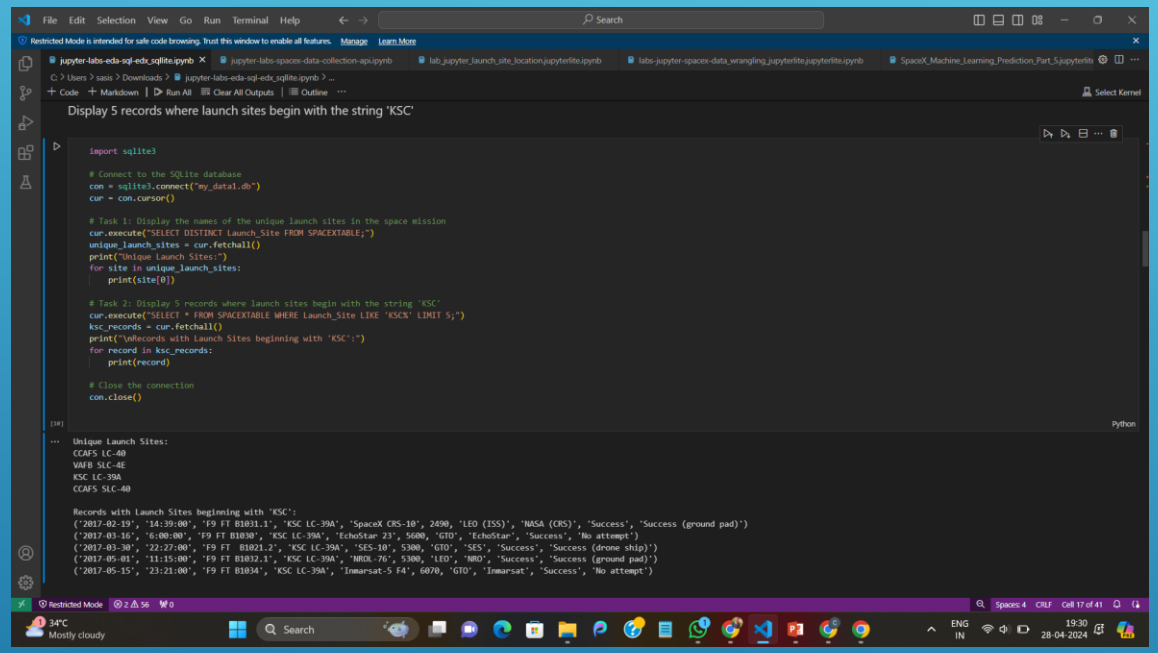

Flight Number vs Orbit by Class
Class
0
1
LEO
ISS
PO
GTO
ES-L1
SSO
HEO
MEO
VLEO
SO
GEO
Orbit
Flight Number

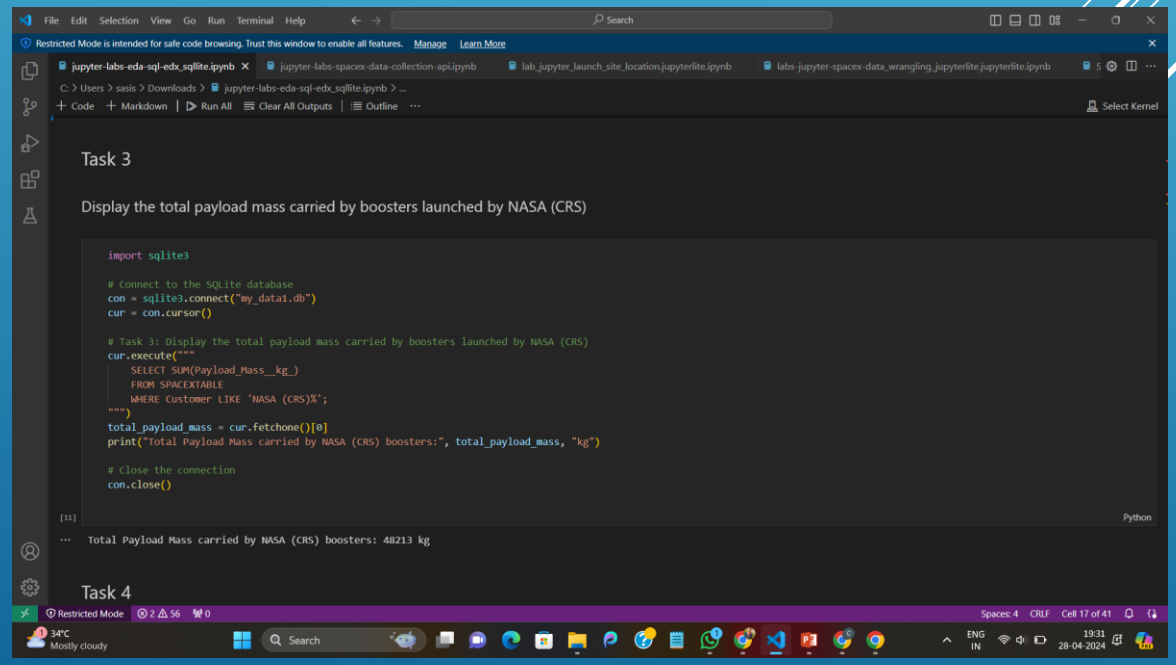

Payload Mass vs Launch Site by Class
CCAFS SLC 40
VAFB SLC 4E
KSC LC 39A
Launch Site
Class
0
1
Payload Mass (kg)


Flight Number vs Launch Site by Class
CCAFS SLC 40
VAFB SLC 4E
KSC LC 39A
Launch Site
0
20
40
60
80
Flight Number
Class
0
1


Average Launch Success Trend
Average Success Rate (%)
0
20
40
60
80
2010
2012
2014
2016
2018
2020
Year

EDA WITH SQL RESULTS:





Total Payload Mass carried by NASA (CRS) boosters: 48213 kg

Task 4

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

```
import sqlite3

# Connect to the SQLite database
con = sqlite3.connect("my_data1.db")
cur = con.cursor()

# Task: Display the average payload mass carried by booster version F9 v1.1
cur.execute("""
    SELECT AVG(Payload_Mass__kg_)
    FROM SPACEXTABLE
    WHERE Booster_Version = 'F9 v1.1';
""")
average_payload_mass = cur.fetchone()[0]
print("Average Payload Mass carried by booster version F9 v1.1:", average_payload_mass, "kg")

# Close the connection
con.close()
```

[12] Python

```
Average Payload Mass carried by booster version F9 v1.1: 2928.4 kg
```

Task 5

List the date where the succesful landing outcome in drone ship was acheived.

Hint:Use min function

```
import sqlite3

# Connect to the SQLite database
con = sqlite3.connect("my_data1.db")
cur = con.cursor()

# Task: List the dates where the successful landing outcome in drone ship was achieved
cur.execute("""
    SELECT Date
    FROM SPACEXTABLE
    WHERE Landing_Outcome = 'Success (drone ship)';
""")
landing_dates = cur.fetchall()
print("Dates of successful landing outcome on drone ship:")
for date in landing_dates:
    print(date[0])

# Close the connection
con.close()
```

[13] Python

```
Dates of successful landing outcome on drone ship:
2016-04-08
2016-05-06
2016-05-27
2016-08-14
2017-01-14
2017-03-30
2017-06-23
2017-06-25
2017-08-24
2017-10-09
2017-10-11
2017-10-30
2018-04-18
2018-05-11
```

Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

jupyter-labs-eda-sql-edx_sqlite.ipynb | jupyter-labs-spacex-data-collection-api.ipynb | lab_jupyter_launch_site_location.jupyterlite.ipynb | labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb | SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite

C: > Users > sasis > Downloads > jupyter-labs-eda-sql-edx_sqlite.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ... Select Kernel

Python

Task 6

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

```
import sqlite3

# Connect to the SQLite database
con = sqlite3.connect("my_data1.db")
cur = con.cursor()

# Task: List the names of the boosters with success in ground pad and payload mass between 4000 and 6000 kg
cur.execute("""
    SELECT Booster_Version
    FROM SPACEXTABLE
    WHERE Landing_Outcome = 'Success (ground pad)'
    AND Payload_Mass__kg_ > 4000
    AND Payload_Mass__kg_ < 6000;
""")
boosters = cur.fetchall()
print("Boosters with success in ground pad and payload mass between 4000 and 6000 kg:")
for booster in boosters:
    print(booster[0])

# Close the connection
con.close()
```

[14] Python

```
Boosters with success in ground pad and payload mass between 4000 and 6000 kg:
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1
```

Task 7

Restricted Mode | 2 ⚠ 56 | 0 Spaces: 4 CRLF Cell 17 of 41

34°C Mostly cloudy Search ENG IN 19:31 28-04-2024

Search Help

Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

jupyter-labs-spacex-data-collection-api.ipynb | lab_jupyter_launch_site_location.jupyterlite.ipynb | labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb | SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite

...l-edx_sqlite.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ... Select Kernel

Task 7

List the total number of successful and failure mission outcomes

```
import sqlite3

# Connect to the SQLite database
con = sqlite3.connect("my_data1.db")
cur = con.cursor()

# Task: List the total number of successful and failed mission outcomes
cur.execute("""
    SELECT Mission_Outcome, COUNT(*)
    FROM SPACEXTABLE
    GROUP BY Mission_Outcome;
""")
mission_outcomes = cur.fetchall()
print("Total number of successful and failed mission outcomes:")
for outcome, count in mission_outcomes:
    print(outcome, ":", count)

# Close the connection
con.close()
```

[15] Python

```
Total number of successful and failed mission outcomes:
Failure (in flight) : 1
Success : 98
Success  : 1
Success (payload status unclear) : 1
```

Python

Restricted Mode | 2 ⚠ 56 | 0 Spaces: 4 CRLF Cell 17 of 41

34°C Mostly cloudy Search ENG IN 19:31 28-04-2024

Task 8

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

```
import sqlite3

# Connect to the SQLite database
con = sqlite3.connect("my_data1.db")
cur = con.cursor()

# Task 8: List the names of the booster_versions which have carried the maximum payload mass using a subquery
cur.execute("""
    SELECT Booster_Version
    FROM SPACEXTABLE
    WHERE Payload_Mass__kg_ = (
        SELECT MAX(Payload_Mass__kg_)
        FROM SPACEXTABLE
    );
""")
max_payload_boosters = cur.fetchall()
print("Booster Versions with maximum payload mass:")
for booster in max_payload_boosters:
    print(booster[0])

# Close the connection
con.close()
```

```
Booster Versions with maximum payload mass:
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
```

Task 9

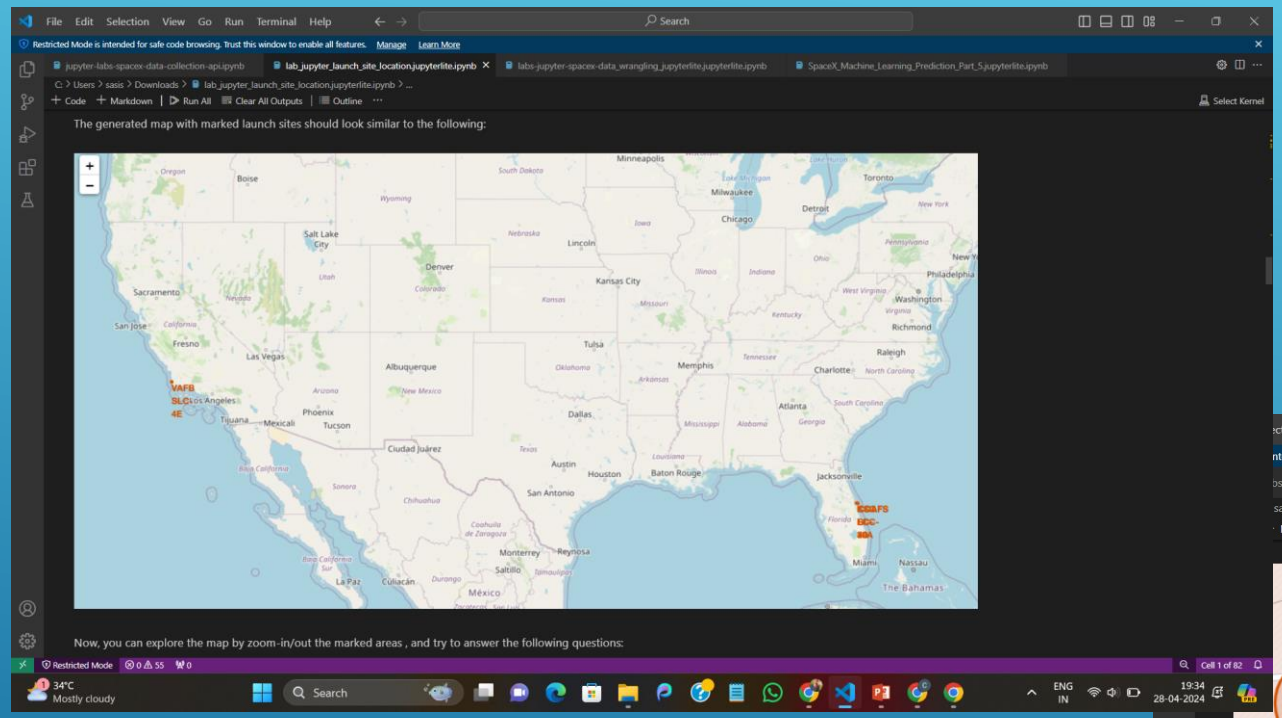
List the records which will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017

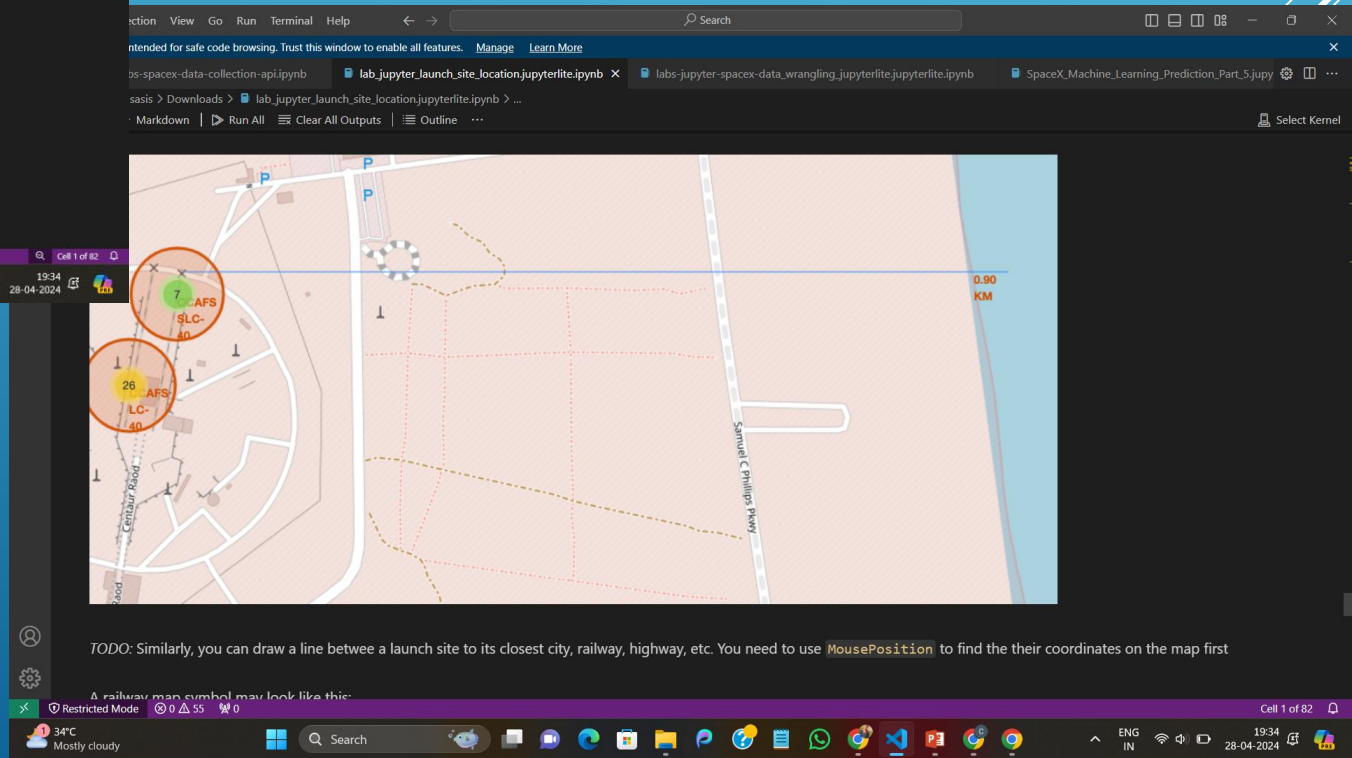
Note: SQLLite does not support monthnames. So you need to use substr(Date,6,2) for month, substr(Date,9,2) for date, substr(Date,0,5), ='2017' for year.

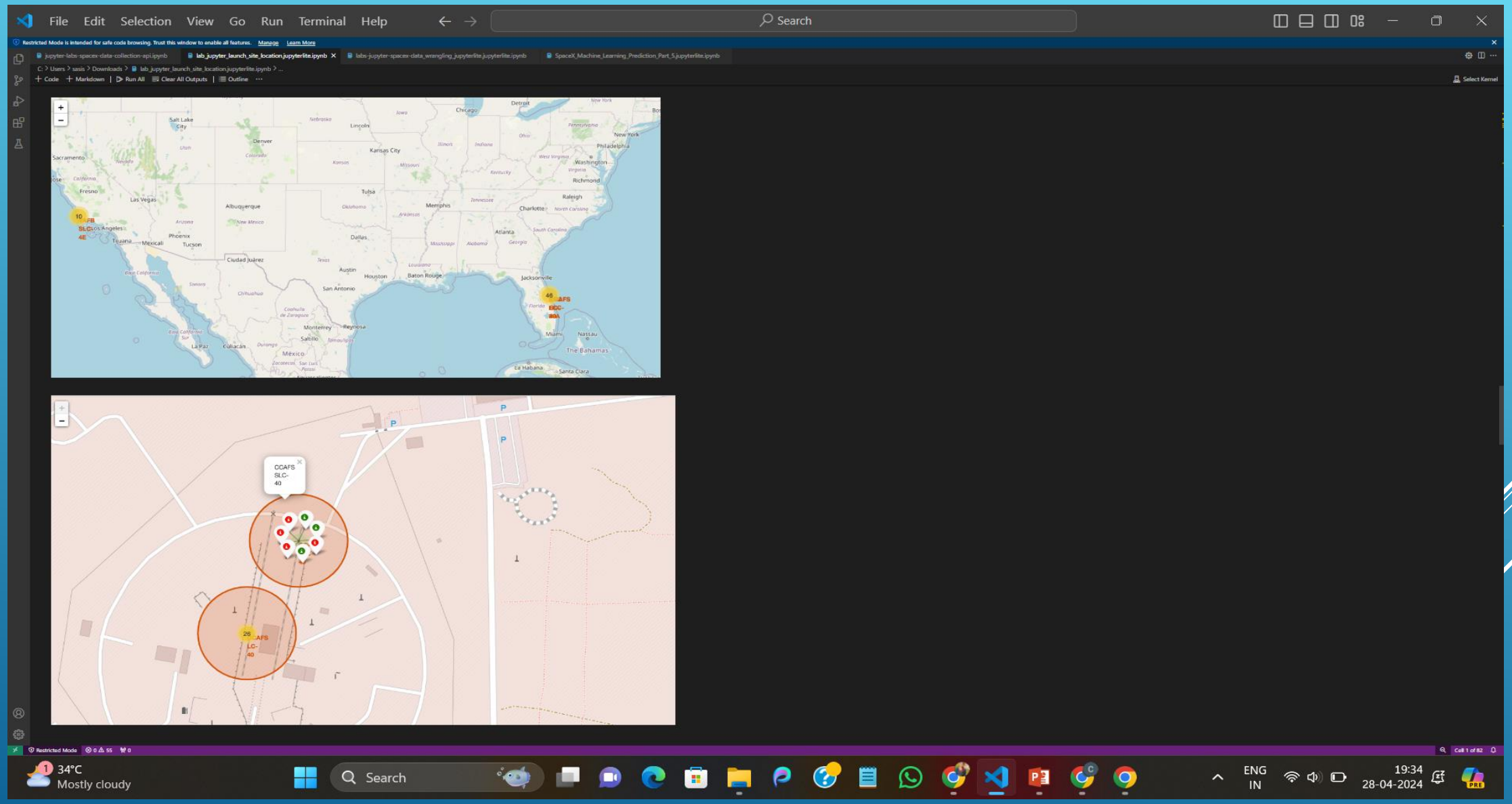
```
import sqlite3

# Connect to the SQLite database
con = sqlite3.connect("my_data1.db")
cur = con.cursor()

# Task: List the records for successful landing outcomes in ground pad, booster versions, and launch sites for the months in year 2017
cur.execute("""
    SELECT
        substr(Date, 6, 2) AS Month,
        Landing_Outcome,
        Booster_Version,
        Launch_Site
    FROM
        SPACEXTABLE
    WHERE
        substr(Date, 0, 5) = '2017' AND
        Landing_Outcome = 'Success (ground pad)'
""")
records_2017 = cur.fetchall()
print("Records for successful landing outcomes in ground pad, booster versions, and launch sites for the months in year 2017:")
for record in records_2017:
    print(record)

# Close the connection
con.close()
```

```
Records for successful landing outcomes in ground pad, booster versions, and launch sites for the months in year 2017:
('02', 'Success (ground pad)', 'F9 FT B1031.1', 'KSC LC-39A')
('05', 'Success (ground pad)', 'F9 FT B1032.1', 'KSC LC-39A')
('06', 'Success (ground pad)', 'F9 FT B1035.1', 'KSC LC-39A')
('08', 'Success (ground pad)', 'F9 B4 B1039.1', 'KSC LC-39A')
('09', 'Success (ground pad)', 'F9 B4 B1040.1', 'KSC LC-39A')
('12', 'Success (ground pad)', 'F9 FT  B1035.2', 'CCAFS SLC-40')
```

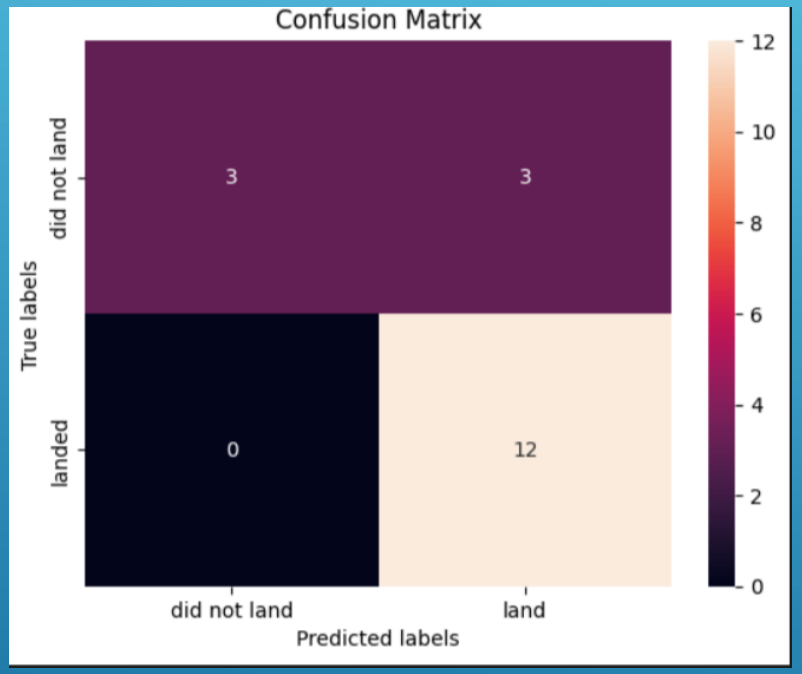
FOLIUM:


The generated map with marked launch sites should look similar to the following:
Now, you can explore the map by zoom-in/out the marked areas , and try to answer the following questions:

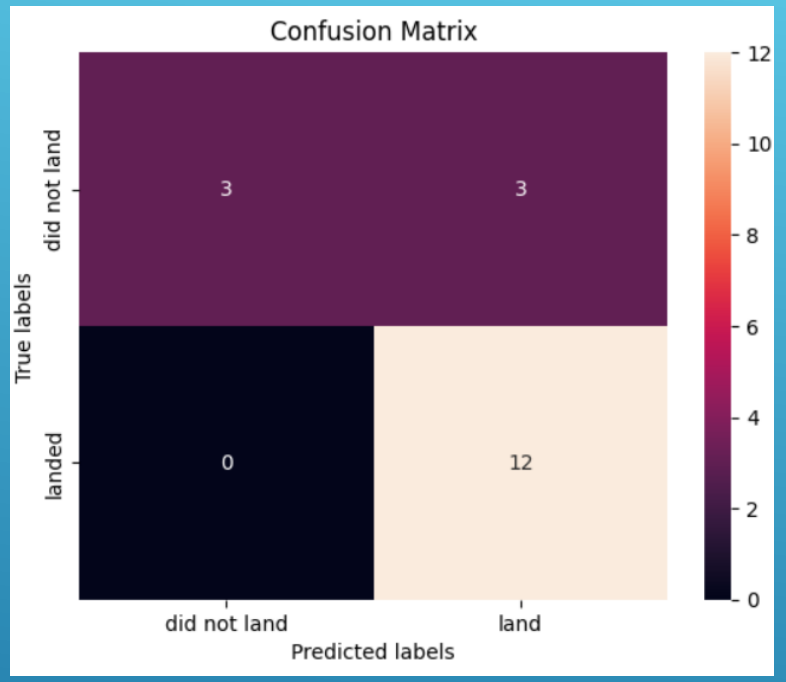

0.90 KM
Samuel C Phillips Pkwy
Centaur Road
TODO: Similarly, you can draw a line betwee a launch site to its closest city, railway, highway, etc. You need to use `MousePosition` to find the their coordinates on the map first


File Edit Selection View Go Run Terminal Help
Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
jupyter-labs-spacex-data-collection-api.ipynb
lab_jupyter_launch_site_location.jupyterlite.ipynb
labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb
SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb
Code Markdown Run All Clear All Outputs Outline
Select Kernel
Salt Lake City
Denver
Sacramento
Kansas City
Chicago
Detroit
New York
Philadelphia
Washington
Richmond
Raleigh
Charlotte
Atlanta
Memphis
Tulsa
Dallas
Houston
Austin
San Antonio
Baton Rouge
Jacksonville
Albuquerque
Phoenix
Tucson
Las Vegas
Fresno
Los Angeles
Tijuana
Mexicali
Ciudad Juárez
Monterrey
Reynosa
Saltillo
Culiacán
La Paz
México
Miami
Nassau
The Bahamas
La Habana
Santa Clara
CCAFS SLC-40
CCAFS LC-40
Restricted Mode
34°C
Mostly cloudy
Search
ENG IN
19:34
28-04-2024

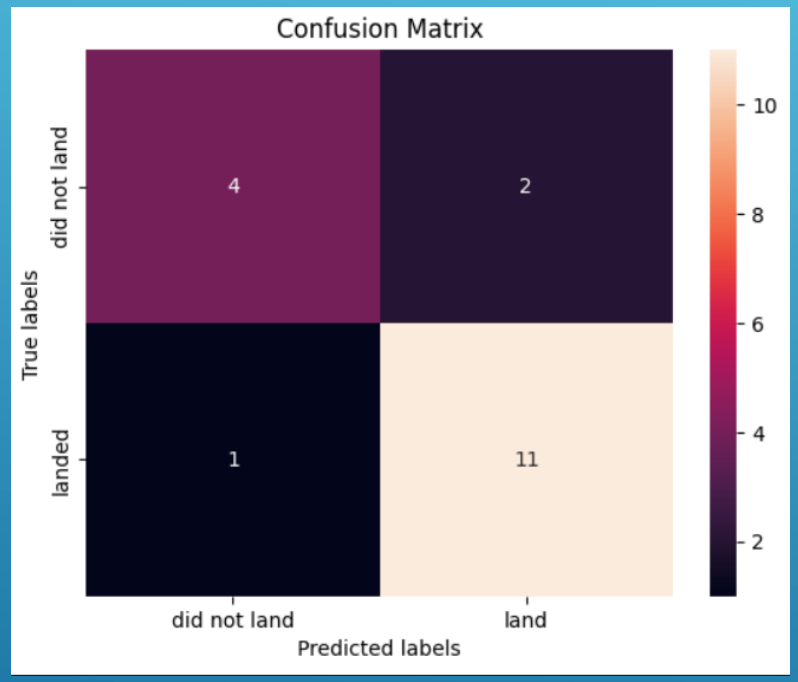
PREDICTIVE ANALYSIS RESULTS:



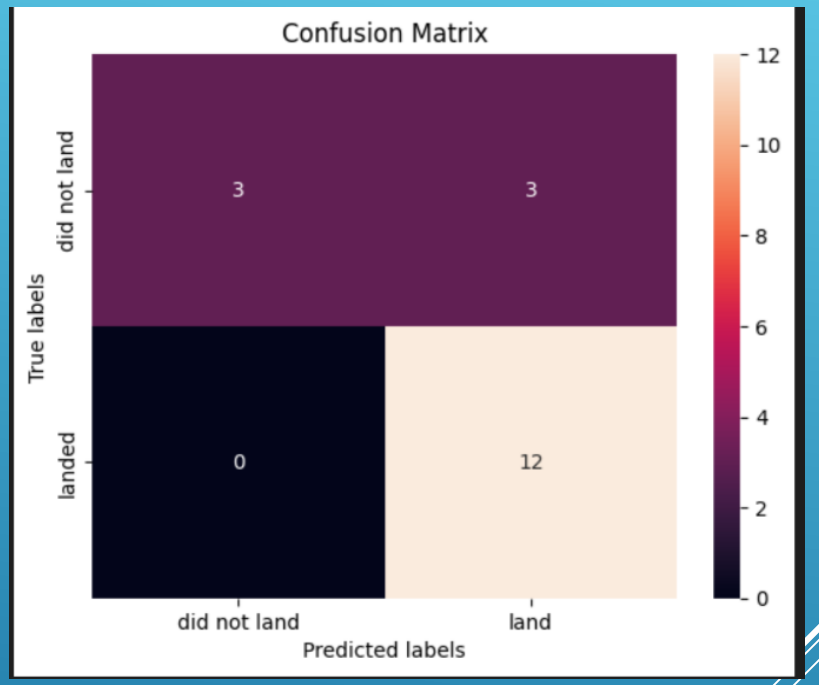
Logistic Regression



Support Vector Machine



Decision Tree Classifier



K-Neighbors Classifier

CONCLUSION:

In conclusion, the project demonstrated the power of geospatial visualization techniques in analyzing and understanding complex spatial data related to space launch operations. The insights gained from this analysis contribute to ongoing efforts in advancing space exploration and technology development. Our analysis revealed significant cost savings potential by accurately predicting first stage landings.