Winning Space Race with Data Science

Emilie Yu
Feb 2025

Github Repo: https://github.com/emilieyyu/capstone/tree/main

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection through API
  - Data collection with Web Scraping
  - Data Wrangling
  - Exploratory Analysis with SQL
  - Exploratory Analysis with Data Visualization
  - Exploratory Analysis with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Analysis
  - Interactive Analytics
  - Predictive Analytics

Introduction

SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX 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 SpaceX for a rocket launch.

Questions:

1. What factors determine if the rocket will land successfully?
2. The interaction amongst features that determine the success rate of a landing?
3. What operating conditions needs to be in place to ensure successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Space X API via web scraping from Wikipedia
- Perform data wrangling
  - One hot encoding applied to categorial features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Data using GET request to SpaceX API
- Decode response as json, turn it into pandas dataframe .json_normalize()
- Clean data
- Webscrape using beautifulsoup

Data Collection – SpaceX API

GET request using SpaceX API to retrieve data.

View more on Github: https://github.com/emilieyyu/capstone/blob/main/jupyter-labs-spacex-data-collection-api-v2.ipynb

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

```python
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```python
response = requests.get(spacex_url)
```

```
# Use json_normalize method to convert the json result into a da

# decode response content as json
static_json_df = res.json()# Get the head of the dataframe

# apply json_normalize
data = pd.json_normalize(static_json_df)

data.head(5)
```

Data Collection - Scraping

- Webscrape Falcon 9 launch records using BeautifulSoup

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_
```

```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

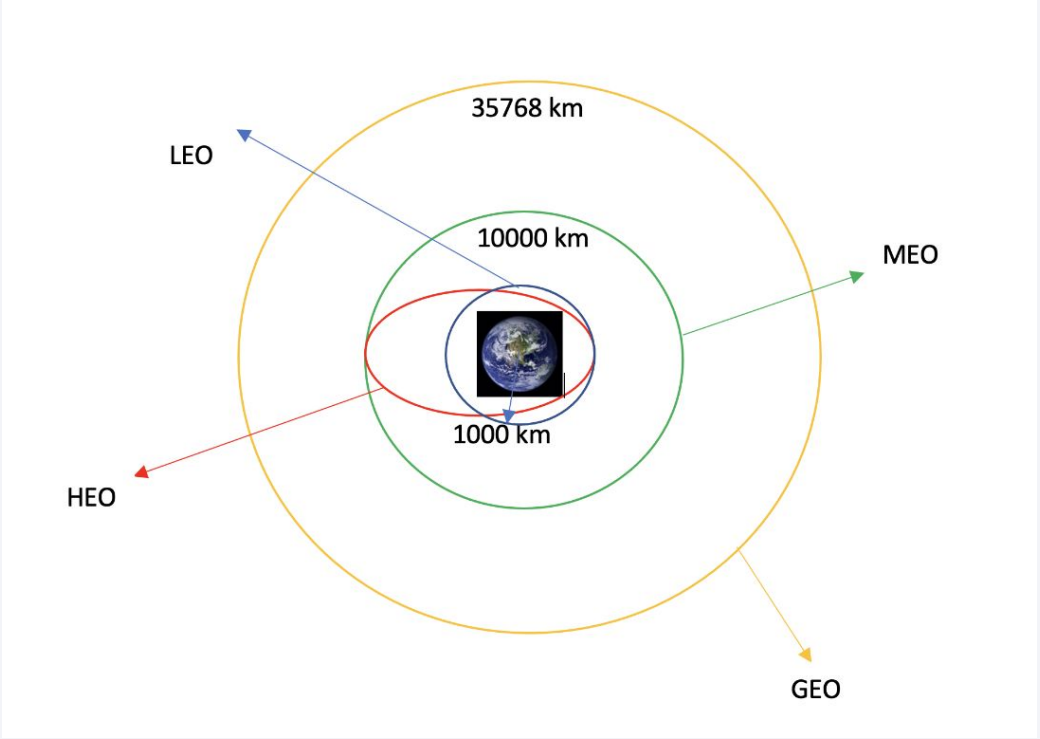
- View more on Github: https://github.com/emilieyyu/capstone/blob/main/jupyter-labs-webscraping-v2.ipynb

```
# Use BeautifulSoup() to create a BeautifulSoup object from a
soup = BeautifulSoup(html_data.text, 'html.parser')
```

```
column_names = []

# Apply find_all() function with `th` element on first_launch_tab
# Iterate each th element and apply the provided extract_column_
# Append the Non-empty column name (`if name is not None and len(
element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

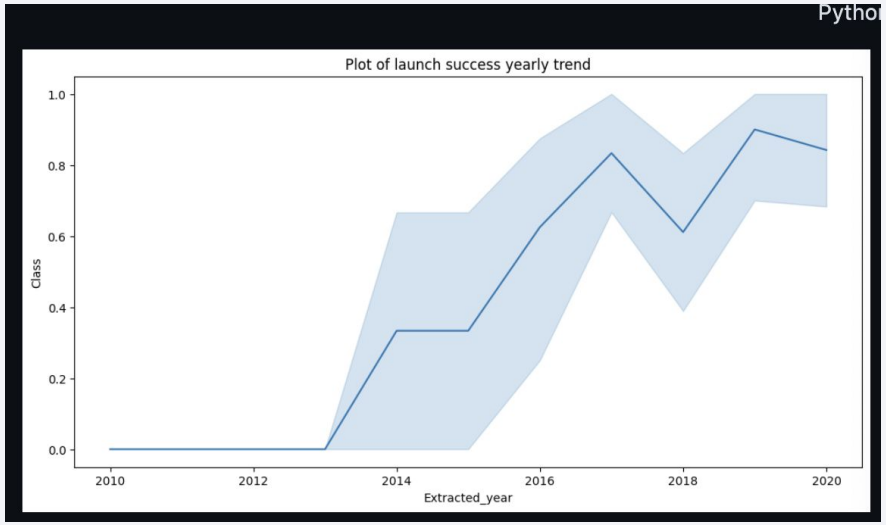
Data Wrangling

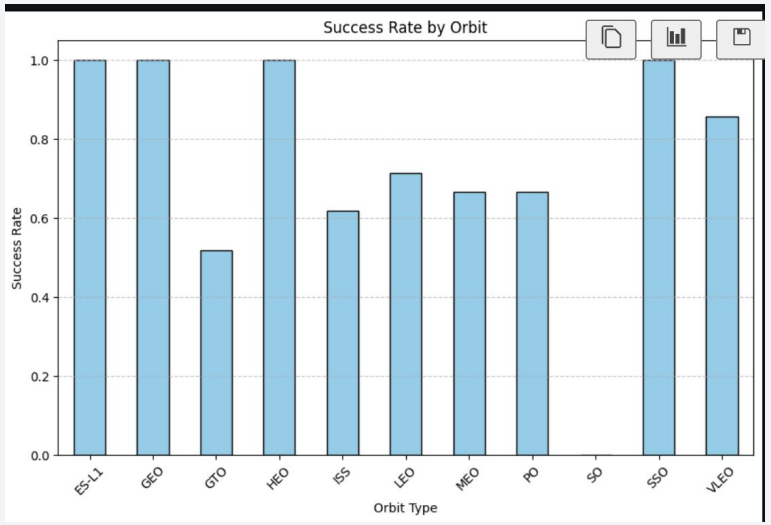
- Exploratory data analysis and training labels
- calculated # launches at each site, # occurrences of each orbit
- created landing outcome label
- View more on Github: https://github.com/emilieyyu/capstone/blob/main/labs-jupyter-spacex-Data%20wrangling-v2.ipynb



EDA with Data Visualization

- Plot a line chart to visualize yearly trend
- Bar chart to show success rate of each orbit
- View more on Github: https://github.com/emilieyyu/capstone/blob/main/jupyter-labs-eda-dataviz-v2.ipynb





EDA with SQL

- Queries performed:
  - names of unique launch sites
  - total payload mass carried by boosters
  - avg payload mass carried by booster
  - total # successful and failure mission
  - failed landing outcomes in drone ship, booster version and launch site names
- View more on Github:

https://github.com/emilieyyu/capstone/blob/main/jupyter-labs-eda-sql-edx-sqlite-v2.ipynb

Build an Interactive Map with Folium

- We used markers, circles, lines to mark the success or failure of launches for each site
- we assigned either 0 or 1 - 0 for failure, 1 for success
- color labeled marker clusters for relatively high success rate
- calculated distance between launch site to proximities
    - if they are near railways, highways, coastlines
    - distance away from cities
- View more on Github:

https://github.com/emilieyyu/capstone/blob/main/lab-jupyter-launch-site-location-v2.ipynb

Build a Dashboard with Plotly Dash

- Pie charts showing total launches by certain sites
- Plotted scatter graphs to show relationship between outcome and payload for different booster versions

Predictive Analysis (Classification)

- Utilized numpy and pandas
- transformed the data
- trained and test data
- used accuracy as metric, gave a score on test accuracy
- created confusion matrix
- used gridserach CV for logistic regression
- finally compared and found wich method performs best

- View more on Github: https://github.com/emilieyyu/capstone/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb

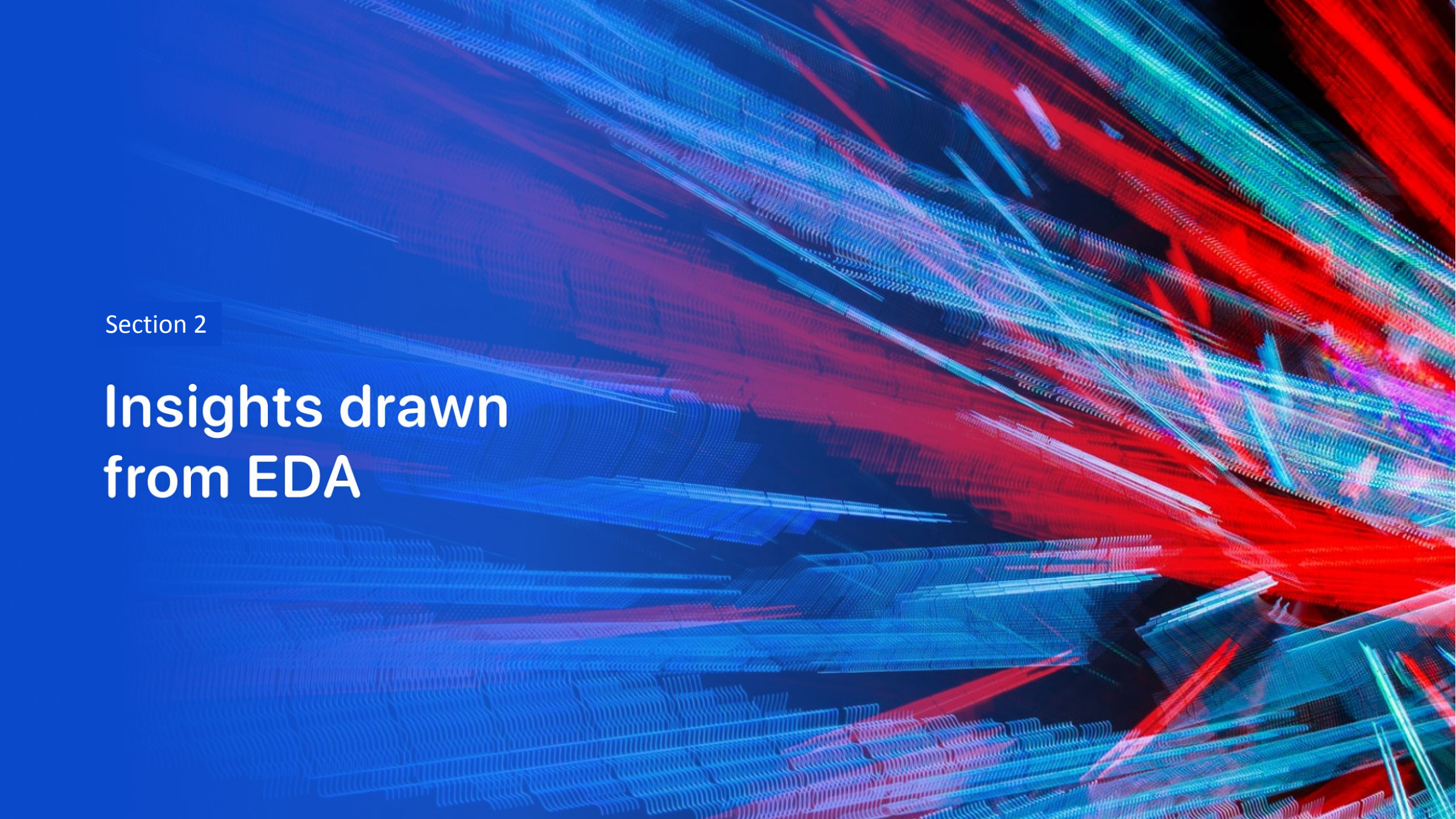
```
Logistic Regression Test Accuracy: 0.8333333333333334
Support Vector Machine Test Accuracy: 0.8333333333333334
Decision Tree Test Accuracy: 0.7222222222222222
K-Nearest Neighbors Test Accuracy: 0.8333333333333334

Best performing model: Logistic Regression with accuracy: 0.83333333
```

Results

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

Section 2

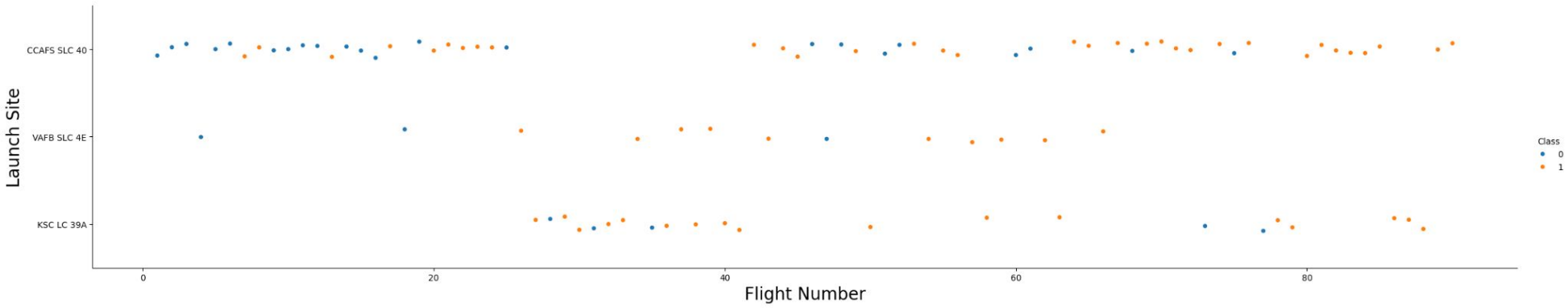
Insights drawn from EDA



Flight Number vs. Launch Site

- scatter plot of Flight Number vs. Launch Site

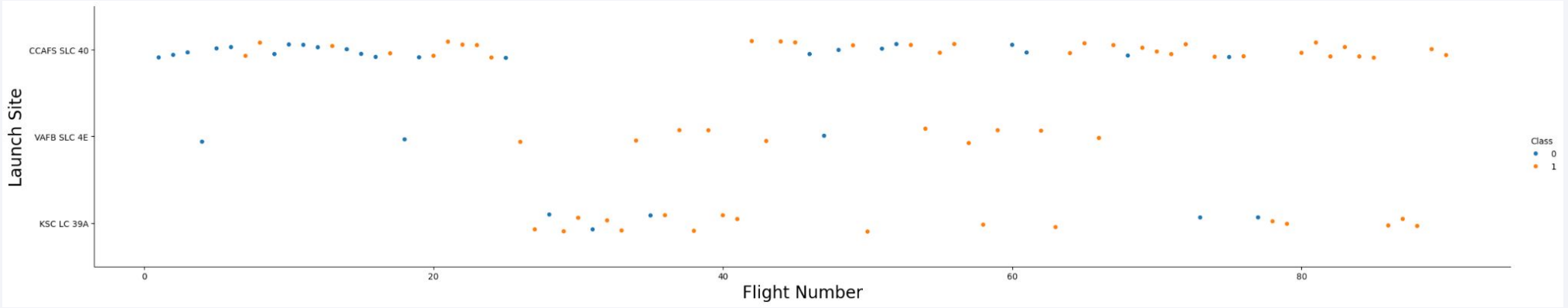
- The larger the flight amount at launch site, the greater the success rate



Payload vs. Launch Site

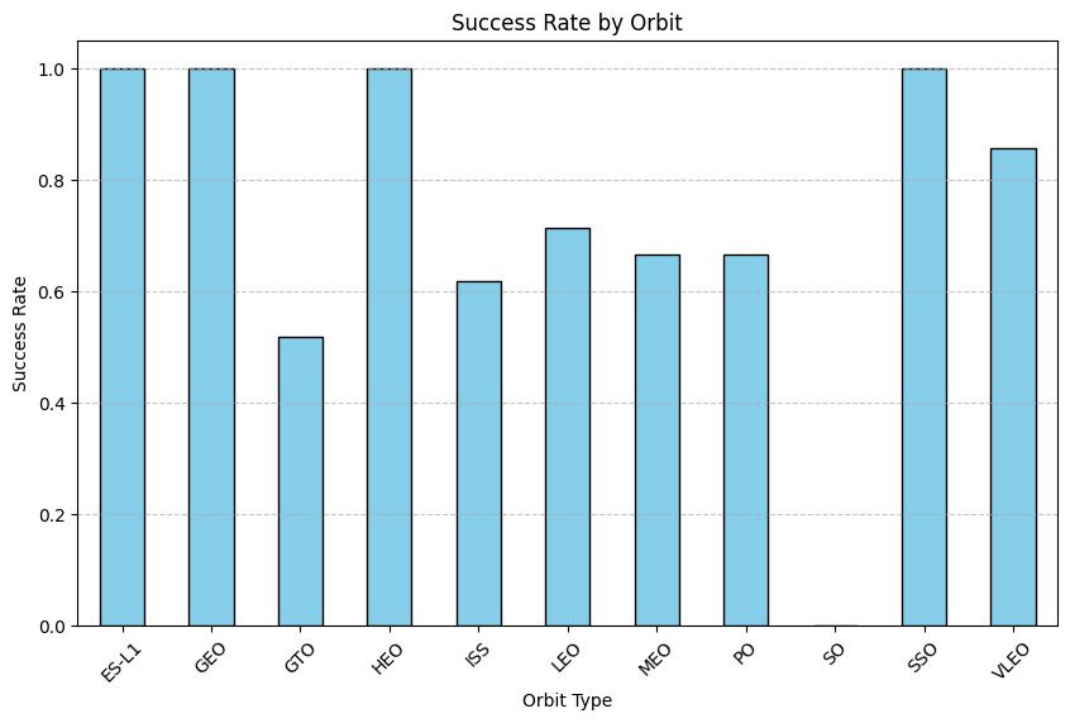
- Scatter plot of Payload vs. Launch Site

- the greater the payload mass for launch site, the higher the success rate.



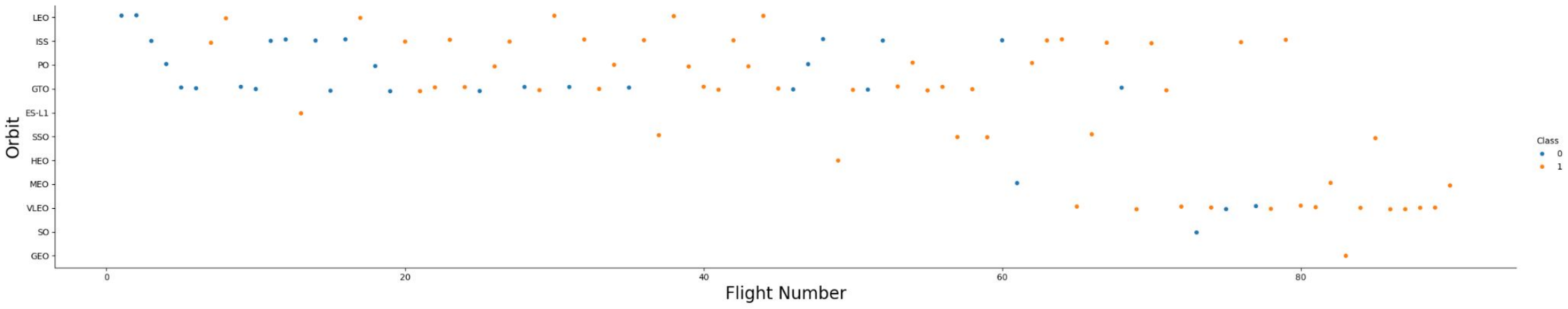
Success Rate vs. Orbit Type

- Bar chart for the success rate of each orbit type
- ES-L1, GEO, HEO, SSO, VLEO had highest success rates.



Flight Number vs. Orbit Type

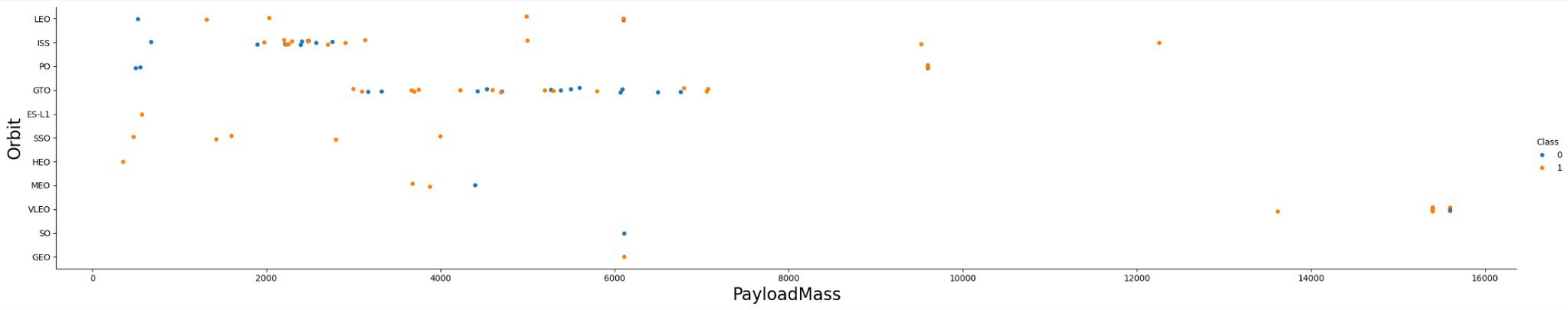
- Scatter point of Flight number vs. Orbit type
- LEO orbit: success is relate to # of flights
- GTO orbit: no relationship between flight # and orbit



Payload vs. Orbit Type

- Scatter point of payload vs. orbit type

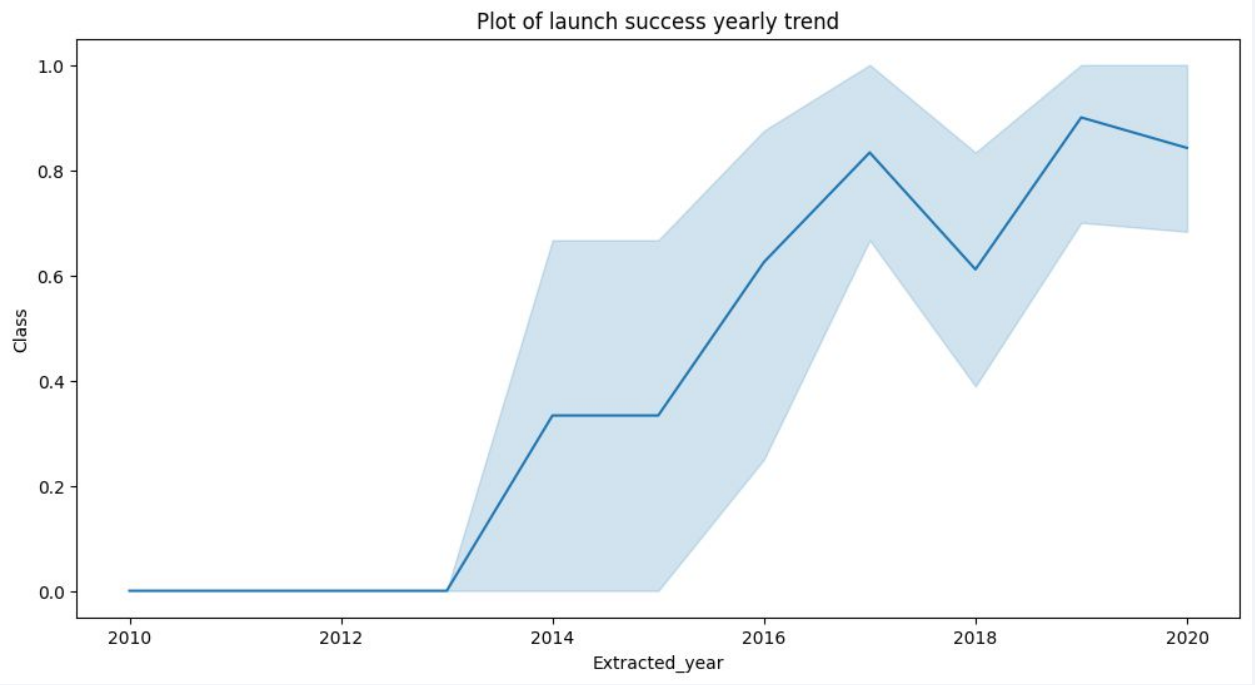
- More successful landing for PO, LEO and ISS orbits with heavier payloads.



Launch Success Yearly Trend

- Line chart of yearly average success rate

- success rate went up ever since 2013



All Launch Site Names

- keyword ‘DISTINCT’ to show unique launch sites

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Launch Site Names Begin with 'KSC'

- Find 5 records where launch sites' names start with `KSC`

```
task_2 = '''
         SELECT *
         FROM SpaceX
         WHERE LaunchSite LIKE 'KSC%'
         LIMIT 5
         '''
create_pandas_df(task_2, database=conn)
```

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Average Payload Mass by F9 v1.1

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

```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on drone ship.

```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

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

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

Boosters Carried Maximum Payload

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

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                              SELECT MAX(PayloadMassKG)
                              FROM SpaceX
                              )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

2017 Launch Records

- 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

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2017-01-01' AND '2017-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

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

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

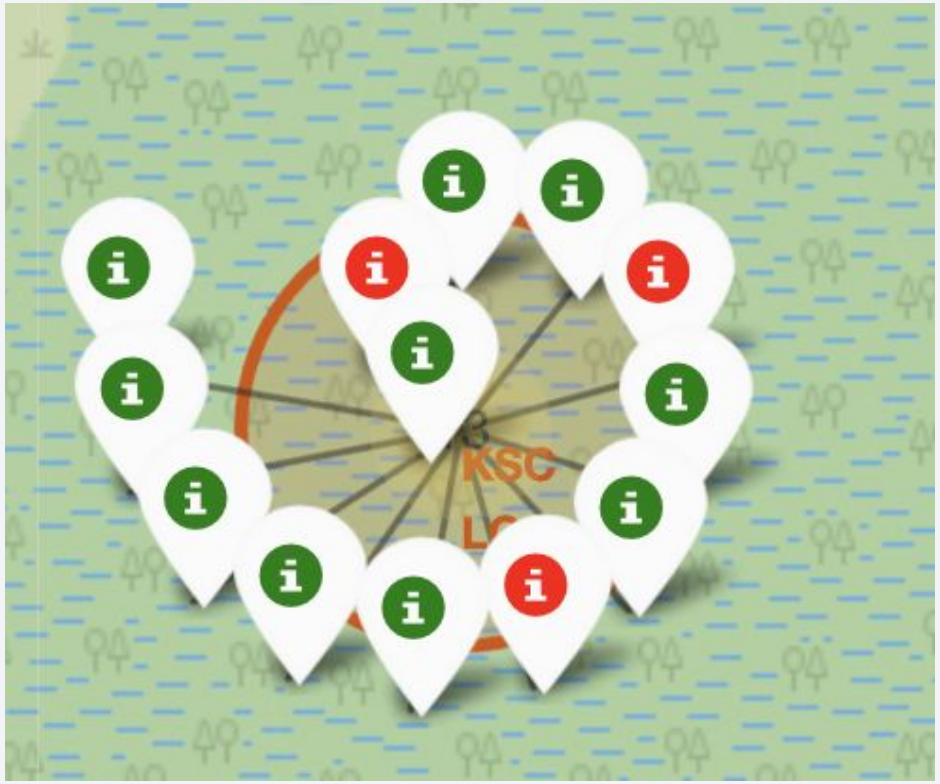

Section 3
Launch Sites
Proximities Analysis

Global SpaceX Launch Sites

- All are located in North America, specifically in United States.

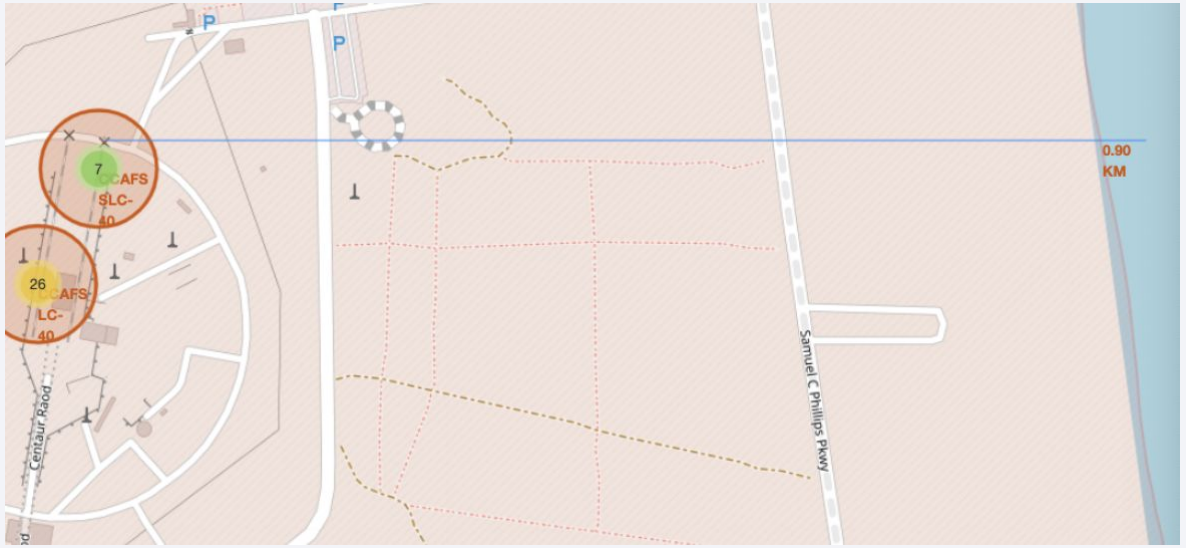
Folium Map at Launch Sites

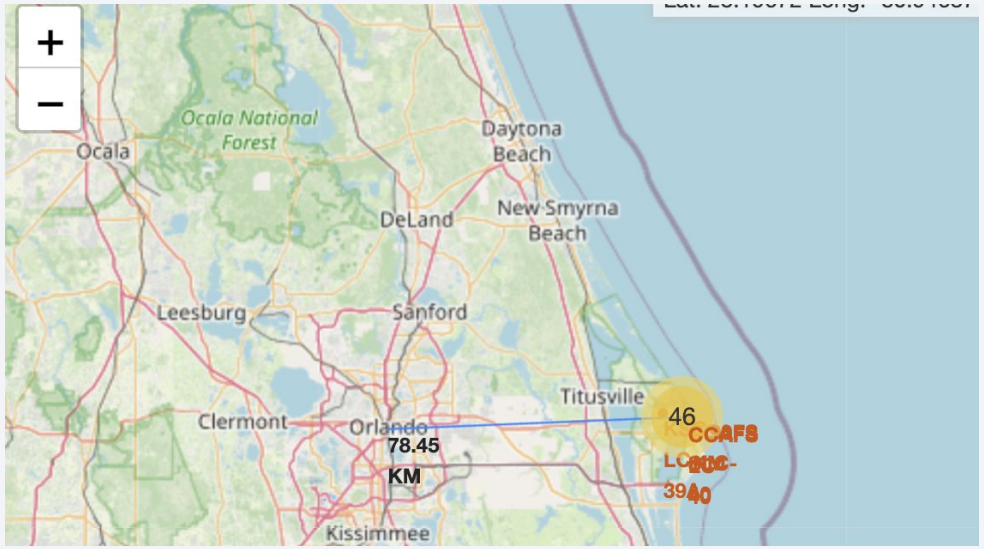
- Green = successful launches
- Red = Failed launches

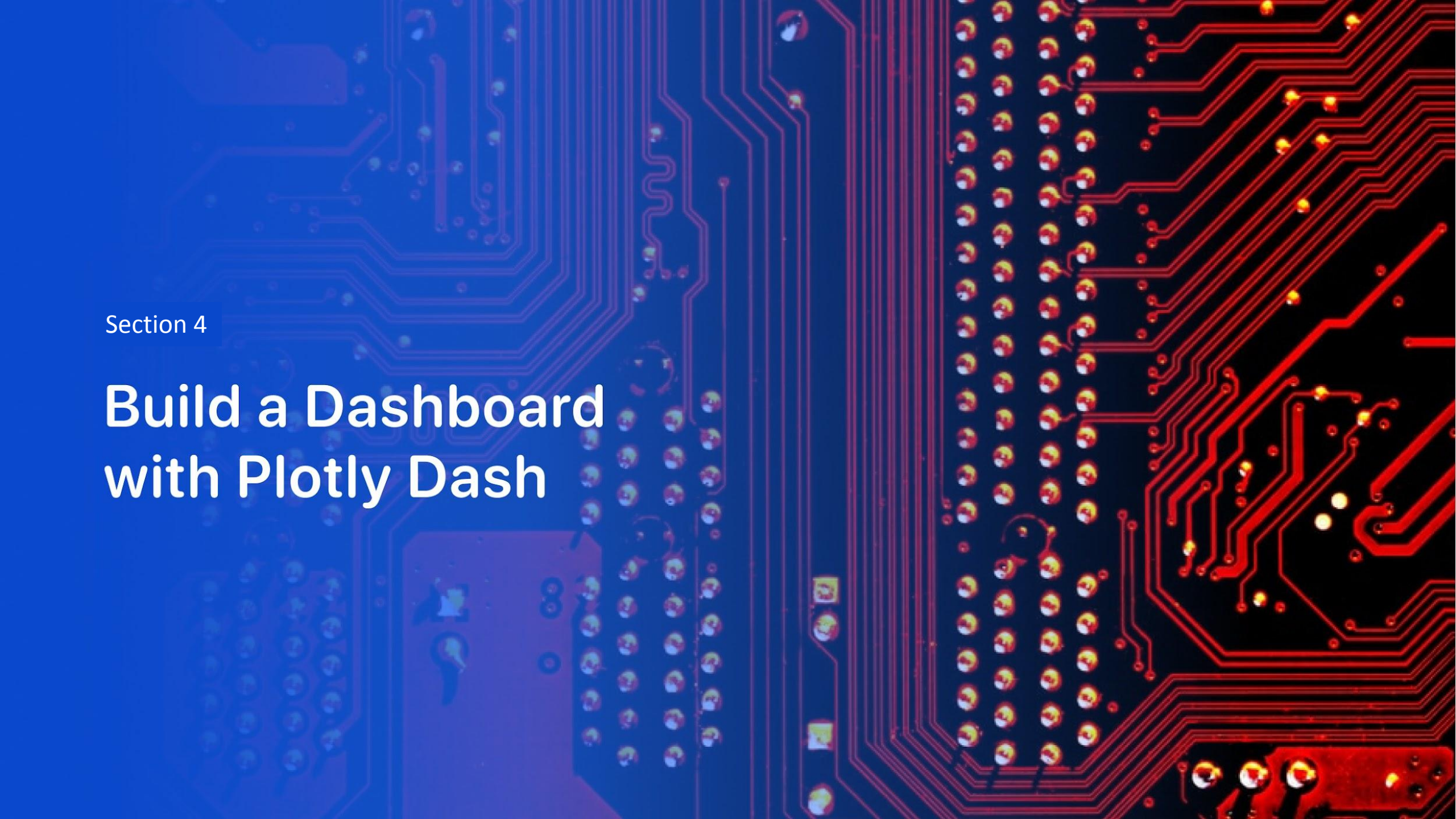


Launch Site Distance to Proximities

- proximities to coastline
- proximities to cities

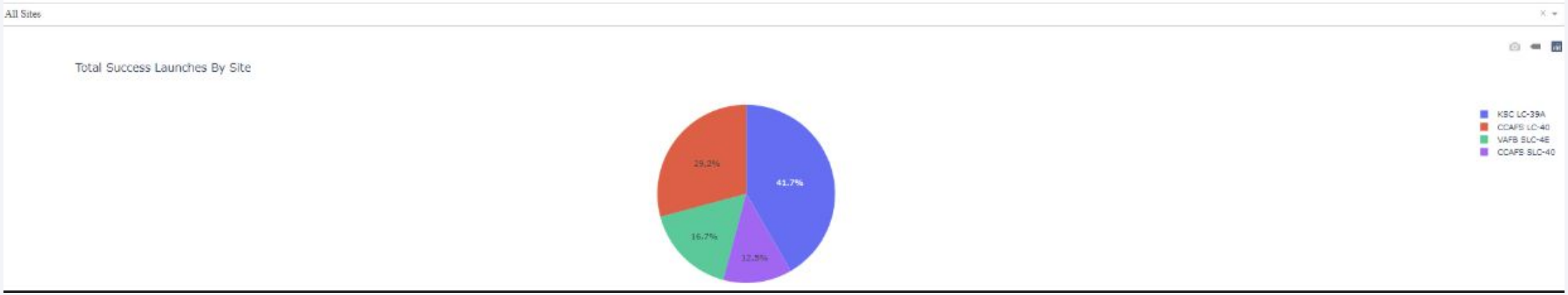





Section 4
Build a Dashboard with Plotly Dash

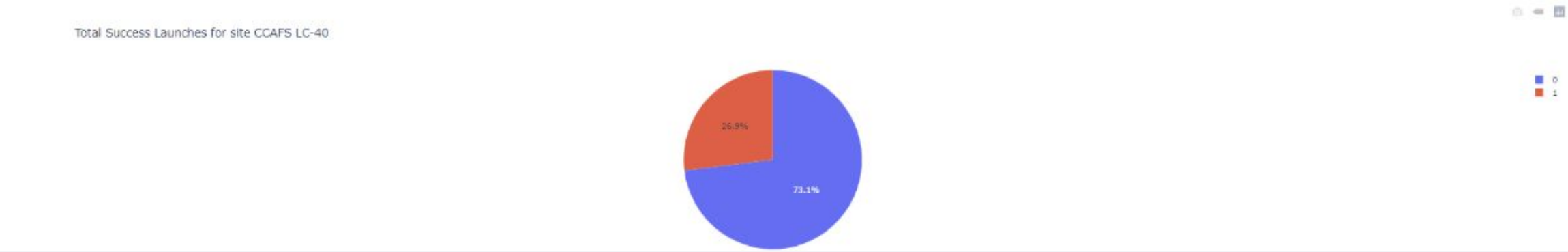
Pie Chart for All Sites

- Total Success Launches by Site



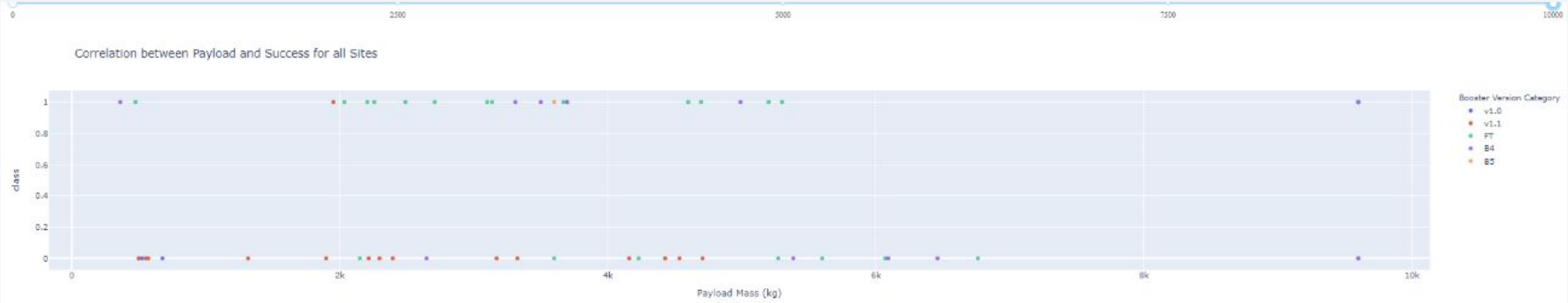
Pie Chart Launch Site Highest Success

- 23.1% fail rate, 76.9% success rate



<Dashboard Screenshot 3>

- Correlation between Payload and Success for all Sites



Section 5

Predictive Analysis (Classification)

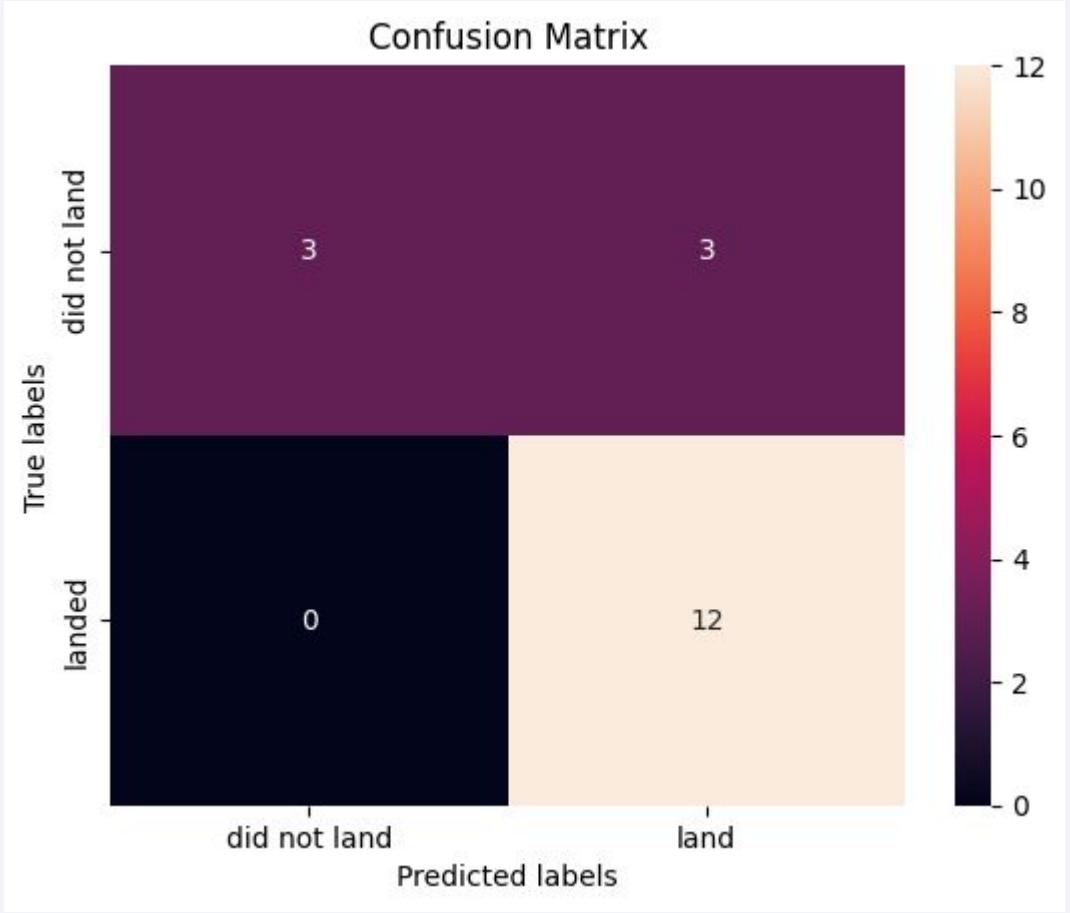
Classification Accuracy

```
# Compute test accuracies
logreg_accuracy = logreg_cv.score(X_test, Y_test)
svm_accuracy = svm_cv.score(X_test, Y_test)
tree_accuracy = tree_cv.score(X_test, Y_test)
knn_accuracy = knn_cv.score(X_test, Y_test)

# Print test accuracies
print("Logistic Regression Test Accuracy:", logreg_accuracy)
print("Support Vector Machine Test Accuracy:", svm_accuracy)
print("Decision Tree Test Accuracy:", tree_accuracy)
print("K-Nearest Neighbors Test Accuracy:", knn_accuracy)

# Find the best model
best_model = max(
    [("Logistic Regression", logreg_accuracy),
     ("SVM", svm_accuracy),
     ("Decision Tree", tree_accuracy),
     ("KNN", knn_accuracy)],
    key=lambda x: x[1]
)

print("\nBest performing model:", best_model[0], "with accuracy:", best_mode
```

```
Logistic Regression Test Accuracy: 0.8333333333333334
Support Vector Machine Test Accuracy: 0.8333333333333334
Decision Tree Test Accuracy: 0.7222222222222222
K-Nearest Neighbors Test Accuracy: 0.8333333333333334
```

Confusion Matrix

- The main issue is false positives (unsuccessful landing marked as successful by the classifier)



Conclusions

- The larger the flight amount at launch, the greater the success rate
- Success rates increase since 2013 and goes up afterwards
- ES-L1, GEO, HEO, SSO, VLEO has most success

Thank you!