IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

TOLA ABRAHAMS
03-22-2023
Lutollar@gmail.com

Outline

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

Executive Summary

- **Summary of methodologies**
- Data Collection: using GET request, data wrangling and formatting
- Data wrangling: perform EDA and determining training labels
- EDA & Visualization: perform EDA & feature engineering with Panda & Matplotlib
- EDA with SQL: query the data for insights on datasets from the database
- Analysis with SQL: perform analysis & visualization with maps on Folium
- Visualization using Plotly: Build interactive real-time dashboards for visualization using plotly dash
- Classification using Machine Learning : Build various classification models and test for best performance
- **Summary of all results**
- Data was collected, cleaned, formatted and exported to csv
- Data was analyzed and labelled with dependant and target variables and further split into training and testing set
- maps, charts and plots showed insights into launch site, landing success rate, payload mass and booster versions

Introduction

- SpaceX has gained worldwide attention for a series of historic milestones. It is the only private company ever to return a spacecraft from low-earth orbit, which it first accomplished in December 2010. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars whereas 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 SpaceX for a rocket launch. In this project, we will collect and make sure the data is in the correct format from an API clean the data and analyse it for insights, visualize the trends and build several classification models to predict the success of future launch based on the data provided.

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**

  Get request sent to SpaceX API, Data wrangling, cleaning and formatting

  Scrapping Falcon9 Launch table from its Wiki URL page and parsing it to a dataframe

- **Perform data wrangling**

  Perform Exploratory Analysis & Determine training tables

- **Perform exploratory data analysis (EDA) using visualization and SQL**

  Perform Exploratory data analysis & feature engineering using Pandas & Matplotlib

  Understanding the SpaceX dataset, loading the data into the tables in DB2 database and executing SQL queries to understand the SpaceX dataset

- **Perform interactive visual analytics using Folium and Plotly Dash**

  Build a Folium map object with launch site coordinates and with markers showing proximities to coastlines, railroads, highways and cities.

  Build an interactive dashboard visual on SpaceX data in real time using plotly dash

- **Predictive analysis using classification models**

  Perform EDA and determine the training labels, create column for “Class” our target variable, standardize the data, split into training and test data for classification and test the models for accuracy to determine best performing model.

Data Collection

Get request sent to SpaceX API, Data wrangling, cleaning and formatting.

- Import Libraries and define functions
- Request rocket launch data from SpaceX API with URL
- Request & Parse SpaceX launch data using GET Request
- Decode the data and turn it into a Pandas dataframe
- Filter the dataframe to only include our target variable
- Deal with missing values and replace them
- Export the cleaned data into CSV

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/DATA%20COLLECTION%20LAB%20SPACEX%20CAPESTONE.ipynb

Data Collection – Web Scraping

Scrapping Falcon9 Launch table from its Wiki URL page and parsing it to a dataframe

- Import libraries
- Define functions to scrape HTML table
- Request the falcon 9 Launch wiki page from its URL
- Extract all Columns / Variable names from HTML table header
- Create a dataframe by parsing the launch HTML table
- Export the table to CSV

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/WEBSCRAPPING%20LAB%20SPACEX%20CAPESTONE.ipynb

Data Wrangling

Perform Exploratory Analysis & Determine training tables

- Import libraries & Define auxiliary functions
- Load the dataset and clean the data
- Calculate the number of launches on each site
- Calculate the number of occurrences of each orbit
- Calculate the number of occurrences of mission outcome per orbit type
- Create a landing outcome label from outcome column as target categorical variable
- Export the analysed data into CSV

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/DATA%20WRANGLING%20LAB%20SPACEX%20CAPESTONE_2.ipynb

EDA with Data Visualization

Perform Exploratory data analysis & feature engineering using Pandas & Matplotlib

- Import libraries & define auxiliary functions
- Import dataset and perform EDA to visualize the trends
- Visualize relationship between flight number and launch site
- Visualize relationship between payload and launch site
- Visualize the relationship between success rate of each orbit type
- Visualize the relationship between flight number and orbit type
- Visualize the relationship between payload and orbit type
- Visualize the launch success yearly trend
- Create dummy variable for feature engineering using one hot encoding
- Call all numeric columns to float64
- Export the final data to CSV

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/EDA%20Visualization%20Lab%20SPACEX%20Capstone.ipynb

EDA with SQL

Understanding the SpaceX dataset, loading the data into the tables in DB2 database and executing SQL queries to understand the SpaceX dataset

- Download the data and connect to the DB2 database
- Explore data by displaying names of unique launch sites
- Records of launch sites with CCA
- Display total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 V1.1
- List the date when the first successful landing outcome in the past was achieved
- Names of boosters with success in drone ships with payload between 4000-6000
- List the total number of successful and failure mission outcomes
- Names of booster versions which have carried maximum payload mass
- Query drone failure outcome, booster version and launch site for 2015
- Ranking successful landing outcomes from June 2010 to March 2017

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/EDA%20SQL%20LITE%20LAB%20SPACEX%20CAPSTONE.ipynb

Build an Interactive Map with Folium

Build a Folium map object with launch site coordinates and with markers showing proximities to coastlines, railroads, highways and cities.

- Launch site location analysis using maps with Folium
- Create a map object with Folium using NASA launch site coordinates as centre
- Mark all launch sites on the map using folium marker object
- Mark all successful and failed mission sites on the map using marker
- Calculate the distance between the launch site locations and their closest proximities to highways, railroads, coastlines and cities
- Use the insights obtained from the map to draw conclusions on the launch site locations

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/ANALYSIS%20AND%20VIZ%20WITH%20FOLIUM%20SPACEX%20CAPSTONE_3.ipynb

Build a Dashboard with Plotly Dash

Build an interactive dashboard visual on SpaceX data in real time using Plotly dash

- Create a dash application component which contains input components such as dropdown list and range sliders to display pie chart and scatter point chart
- Add a launch site dropdown input component
- Add a callback function that renders “Success pie chart” based on selected site dropdown
- Adda arrange slider to select payload
- Add a call back function to render the “success payload scatter plot”
- Launch the interactive web dashboard on a private IP/Port: 127.0.0.1 / 8050

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/Dash%20Interactive%20Dashboard%20SPACEX%20CAPSTONE.ipynb

Predictive Analysis (Classification)

Perform EDA and determine the training labels, create column for "Class" our target variable, standardize the data, split into training and test data for classification and test the models for accuracy to determine best performing model.

- Import libraries and load the dataset and Define the plot_confusion_matrix
- Create NumPy array with column "Class" and assign it to variable Y then we Standardize the data in X and assign it to Variable X
- Split the data into training and testing data
- Create Logistic regression object and GridSearchCV object and find best parameters : {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'} and accuracy : 0.8464285714285713 determine test accuracy:0.8333333333333334 and plot confusion matrix
- Create SVM object and GridSearchCV object and find best parameters: {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'} and accuracy : 0.8482142857142856 determine test accuracy:0.8482142857142856 and plot confusion matrix
- Create Decision tree classifier object with GridSearchCV object and find best parameters: {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 2, 'splitter': 'random'} and accuracy : 0.9017857142857144 determine the test accuracy: 0.8333333333333334 and plot confusion matrix
- Create a KNN object with GridSearchCV object and find best parameters: {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1} and accuracy : 0.8482142857142858 determine the test accuracy : 0.8333333333333334 and plot a confusion matrix
- We compare all the models to determine the best performing which is the decision tree classifier with training accuracy: 0.9017857142857144

https://github.com/MoAbbazi/IBM-Data-science-capstone/blob/main/MACHINE%20LEARNING%20LAB%20SPACEX%20CAPSTONE_5.ipynb

Results

Exploratory data analysis results

- From our scatter plot for Flight number Vs Payload mass see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.)
- Also from our Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000)
- From scatter plot between orbit and Flight number we see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.
- Also from the scatter plot between Orbit and Payload mass With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Results

Predictive analysis results

- After using the function train_test_split to split the data X and Y into training and test data. Set the parameter test_size to 0.2 and random_state to 2 we have a train shape of 72 and test shape of 18.

- We obtained best parameters for the Logistic regression as tuned hpyerparameters :(best parameters) {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}, accuracy : 0.8464285714285713

- For our SVM we obtained tuned hpyerparameters :(best parameters) {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}, accuracy : 0.8482142857142856

- For our Decision tree we obtained tuned hpyerparameters :(best parameters) {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 2, 'splitter': 'random'}, accuracy : 0.9017857142857144

- And for the KNN we got tuned hpyerparameters :(best parameters) {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}accuracy : 0.8482142857142858

- We plot a confusion matrix for each of the classification methods used.

- Each model had the same Accuracy on test data: 0.8333333333333334

Section 2

Insights drawn from EDA

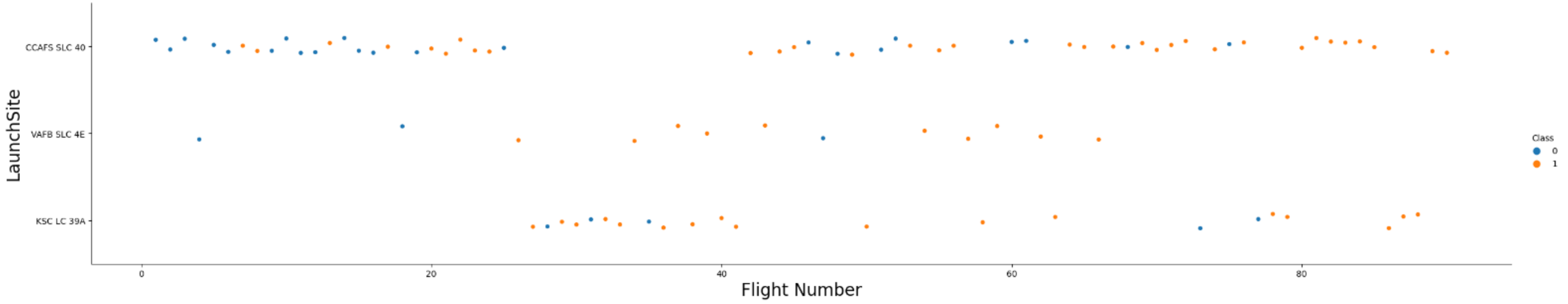
Flight Number vs. Launch Site

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

Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

In [4]:

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



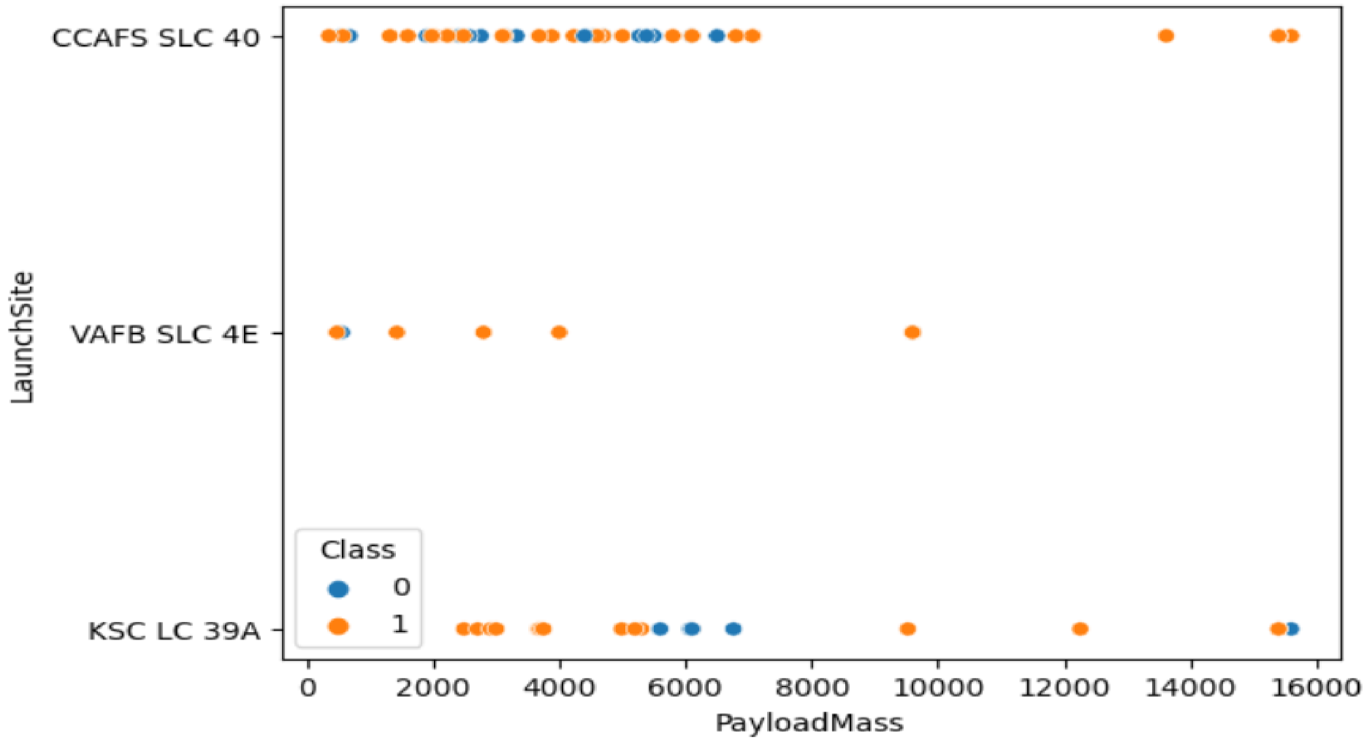
Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

The VAFB SLC Launch site showed more flight success from flight number 18 upwards.

Payload vs. Launch Site

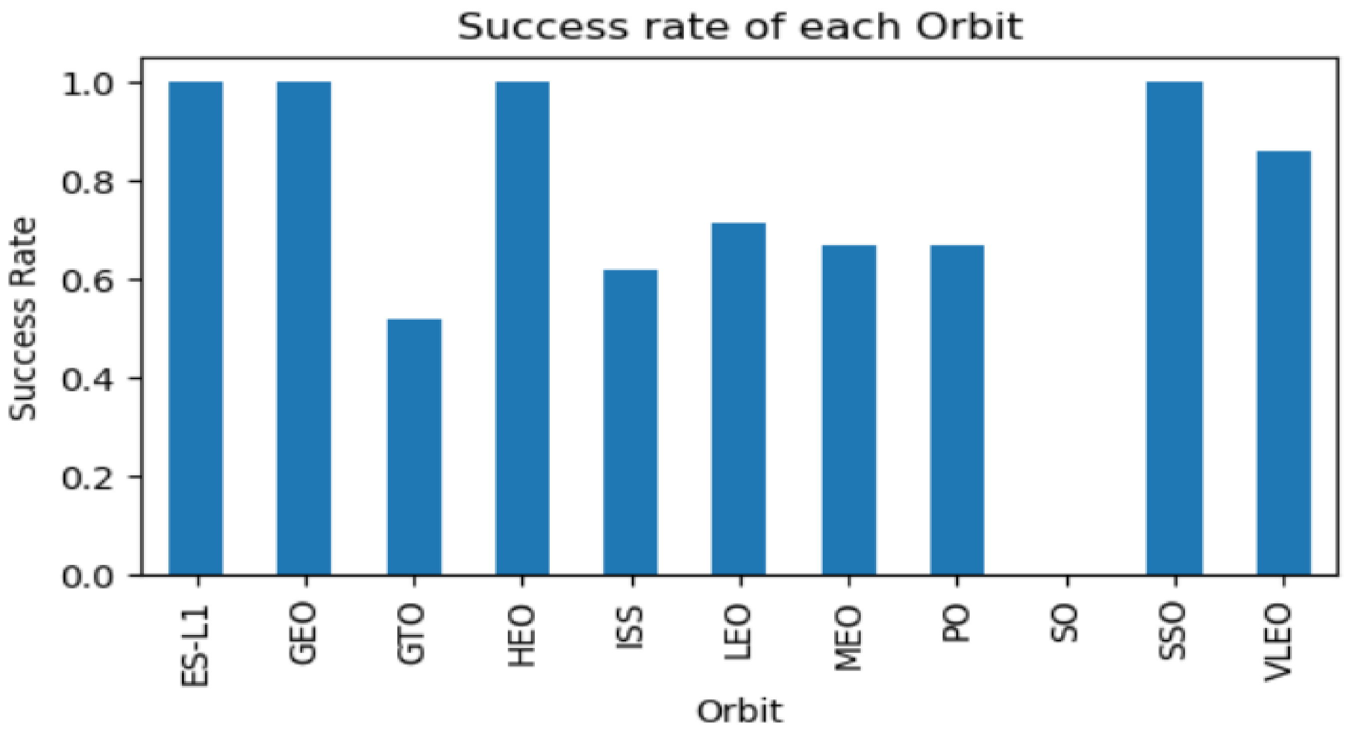
We also want to observe if there is any relationship between launch sites and their payload mass.

[9]:
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and 
sns.scatterplot(data=df, x="PayloadMass", y="LaunchSite", hue="Class")
plt.show()
```



The launch site VAFB SLC showed success with payload mass from 1000 while KSC showed no success at +/-6000

Success Rate vs. Orbit Type



Analyze the ploted bar chart try to find which orbits have high sucess rate.

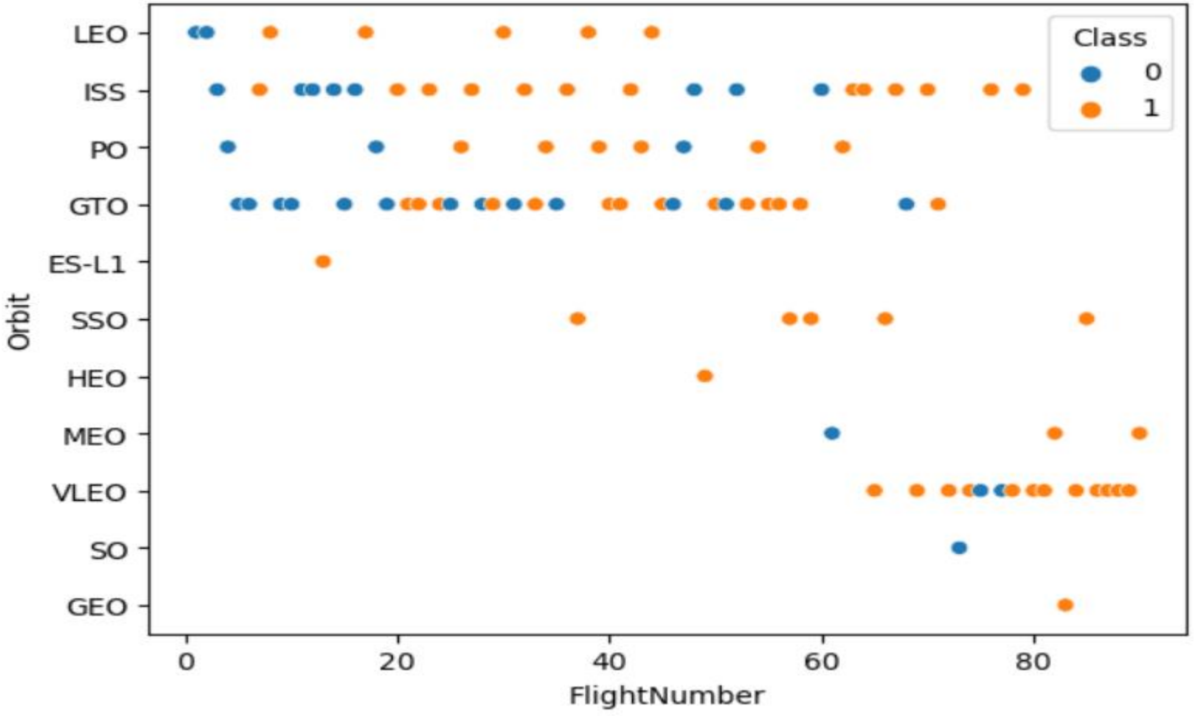
The Orbits ES-L1, GEO, HEO, SSO and VLEO all have very high success rates

Flight Number vs. Orbit Type

For each orbit, we want to see if there is any relationship between FlightNumber and Orbit type.

In [34]:

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to
sns.scatterplot(data=df, x="FlightNumber", y="Orbit", hue="Class")
plt.show()
```

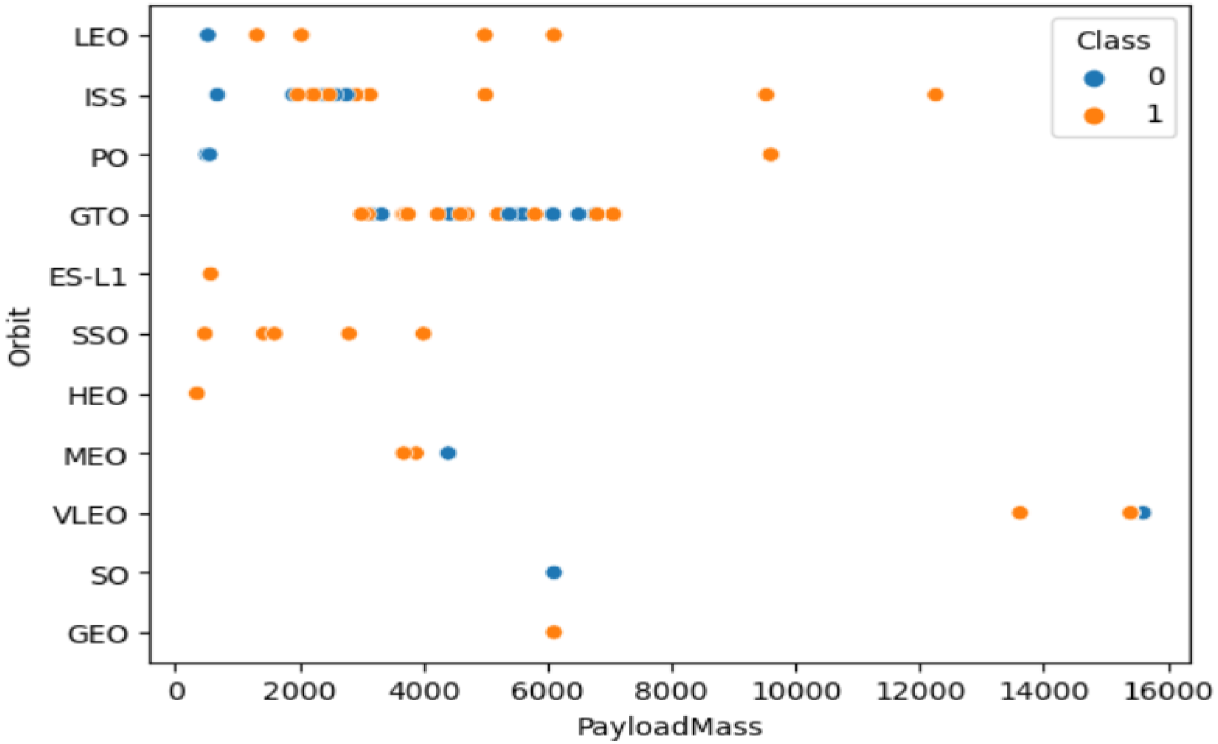


You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type

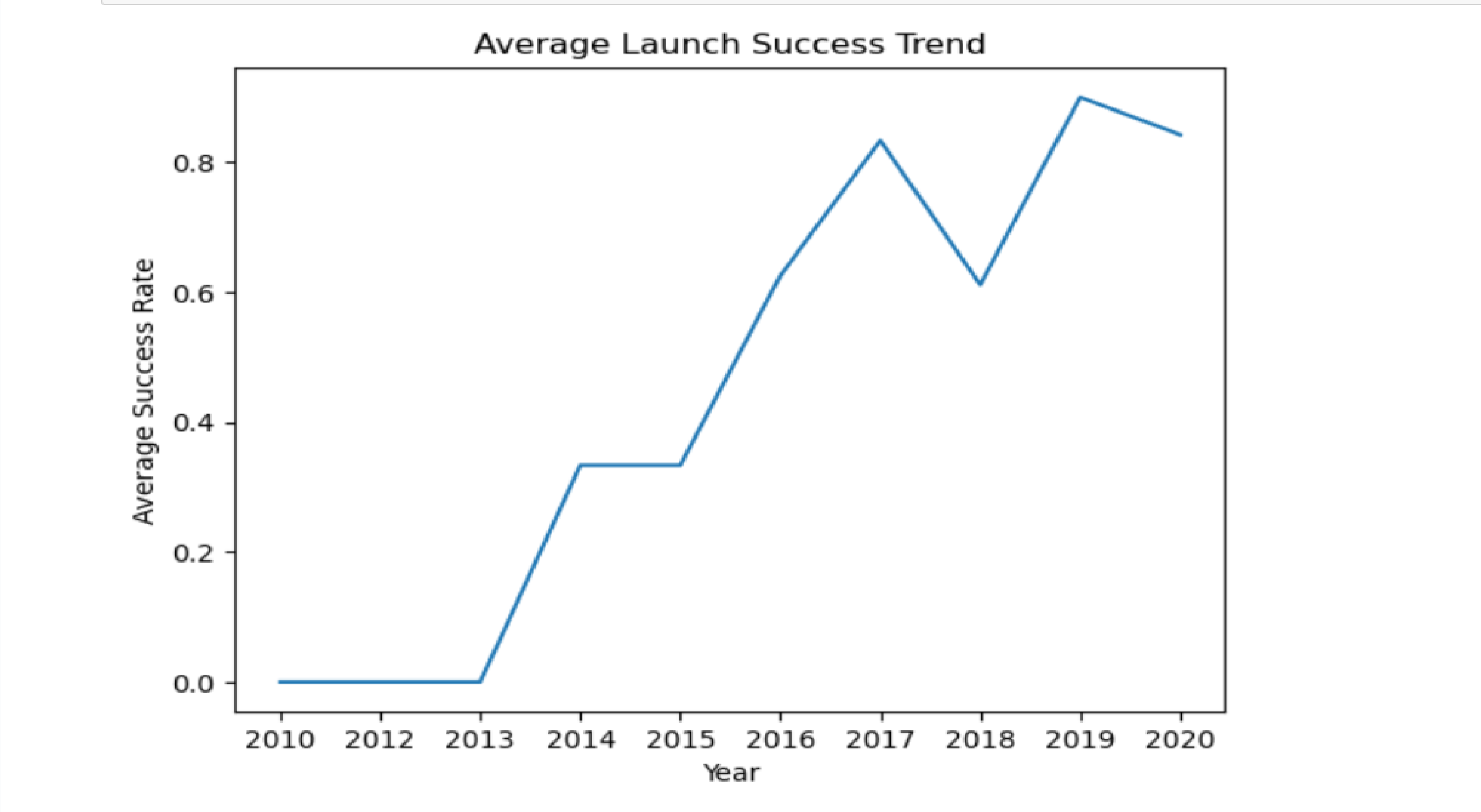
Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

In [60]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the
sns.scatterplot(data=df, x="PayloadMass", y="Orbit", hue="Class")
plt.show()
```



With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
However for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here.

Launch Success Yearly Trend



You can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

Task 1

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

In [28]:
```
unique_launch_sites = pd.read_sql('SELECT DISTINCT Launch_Site FROM SPACEXTBL', con)
print(unique_launch_sites)
```

```
     Launch_Site
0   CCAFS LC-40
1   VAFB SLC-4E
2    KSC LC-39A
3  CCAFS SLC-40
```

There are four unique launch site in the space mission CCAFS LC, VAFB SLC, KSC LC, CCAFS SLC.

Launch Site Names Begin with 'CCA'

Task 2

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

In [31]:
```
launch_sites = pd.read_sql("SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5", con)
launch_sites.head()
```

Out[31]:

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
0	04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	08-12-2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

ALL 5 RECORDS WITH CCA WHERE LAUNCHED TO LEO ORBIT BY NASA AND SPACEX AND ALL WERE SUCCESSFUL

Total Payload Mass

Task 3

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

In [34]:
```
PayloadMass_total = pd.read_sql('SELECT SUM(PAYLOAD_MASS__KG_) as total_payload_mass FROM "SPACEXTBL" WHERE CUSTOMER = "NASA (CRS)"', con)
PayloadMass_total
```

Out[34]:

	total_payload_mass
0	45596

The total payload mass carried by boosters launched by NASA (CRS) is 45596.

Average Payload Mass by F9 v1.1

Task 4

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

In [35]:
```
PayloadMass_avg = pd.read_sql('SELECT AVG(PAYLOAD_MASS__KG_) as average_payload_mass FROM "SPACEXTBL" WHERE Booster_version = "F9 v1.1"', con)
PayloadMass_avg
```

Out[35]:

	average_payload_mass
0	2928.4

The average payload mass carried by booster version F9 v1.1 is 2928.4

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

In [36]:

```
# Execute the query and retrieve the results
query = "SELECT MIN(`Landing _Outcome`) FROM `SPACEXTBL` WHERE `Landing _Outcome` = 'Success (ground pad)'"
result = con.execute(query).fetchall()

# Print the result
print(result)
```

```
[('Success (ground pad)',)]
```

No record was found for successful ground landing date.

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

In [38]:
```
results = pd.read_sql_query("SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE `Landing _Outcome` = 'Success (drone ship)' AND PAYL
print(results)
```

```
  Booster_Version
0     F9 FT B1022
1     F9 FT B1026
2  F9 FT  B1021.2
3  F9 FT  B1031.2
```

There were 4 records found for the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

In [44]:

```
results_7 = pd.read_sql_query("SELECT MISSION_OUTCOME, COUNT(*) as COUNT FROM SPACEXTBL GROUP BY MISSION_OUTCOME", con)
print(results_7)
```

```
                    Mission_Outcome  COUNT
0               Failure (in flight)      1
1                           Success     98
2                          Success       1
3  Success (payload status unclear)      1
```

The total number of successful missions were 100 while the failed mission is 1.

Boosters Carried Maximum Payload

Task 8

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

In [45]:
```
results_7 = pd.read_sql_query("SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)", con)
print(results_7)
```

```
   Booster_Version
0    F9 B5 B1048.4
1    F9 B5 B1049.4
2    F9 B5 B1051.3
3    F9 B5 B1056.4
4    F9 B5 B1048.5
5    F9 B5 B1051.4
6    F9 B5 B1049.5
7   F9 B5 B1060.2
8   F9 B5 B1058.3
9    F9 B5 B1051.6
10   F9 B5 B1060.3
11  F9 B5 B1049.7
```

There were a total of 12 F9 B5 Booster versions that have carried the maximum payload mass

2015 Launch Records

Task 9

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [41]:

```
query = "SELECT strftime('%m', Date) AS month, `Landing _Outcome`, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE substr(D
results = pd.read_sql_query(query, con)
print(results)
```

```
  month      Landing _Outcome Booster_Version  Launch_Site
0  None  Failure (drone ship)    F9 v1.1 B1012  CCAFS LC-40
1  None  Failure (drone ship)    F9 v1.1 B1015  CCAFS LC-40
```

There were 2 records found for month names, failure landing_outcomes in drone ship, booster versions and launch_site for the months in the year 2015

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

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

In [44]:
```
query = "SELECT COUNT(*) as success_count FROM SPACEXTBL WHERE `Landing _Outcome` LIKE 'Success%' AND Date BETWEEN '2010-06-0
results = pd.read_sql_query(query, con)
print(results)
```

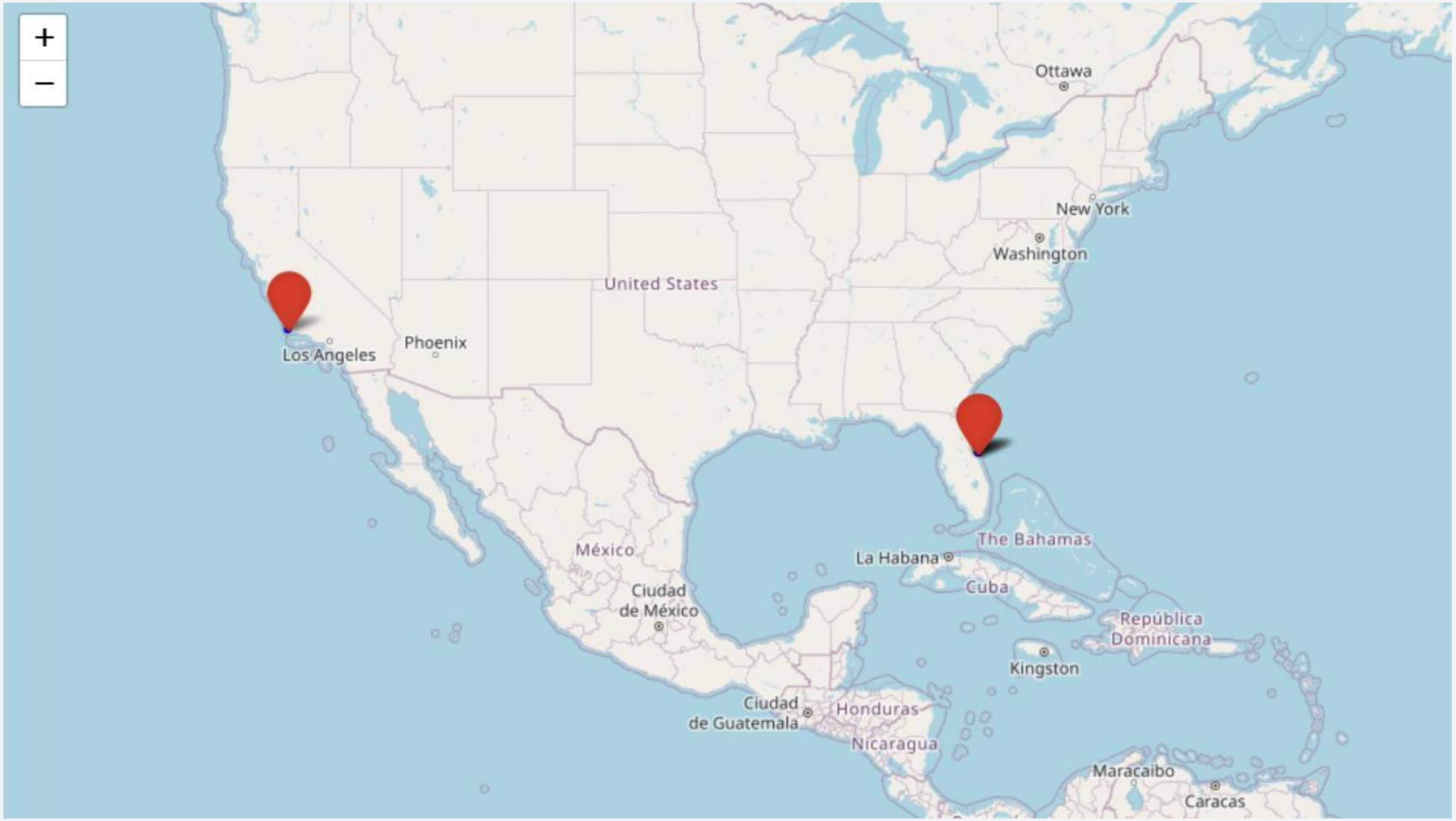
```
   success_count
0              0
```

There was no record found for landing outcomes between June 2010 and March 2017

Section 3

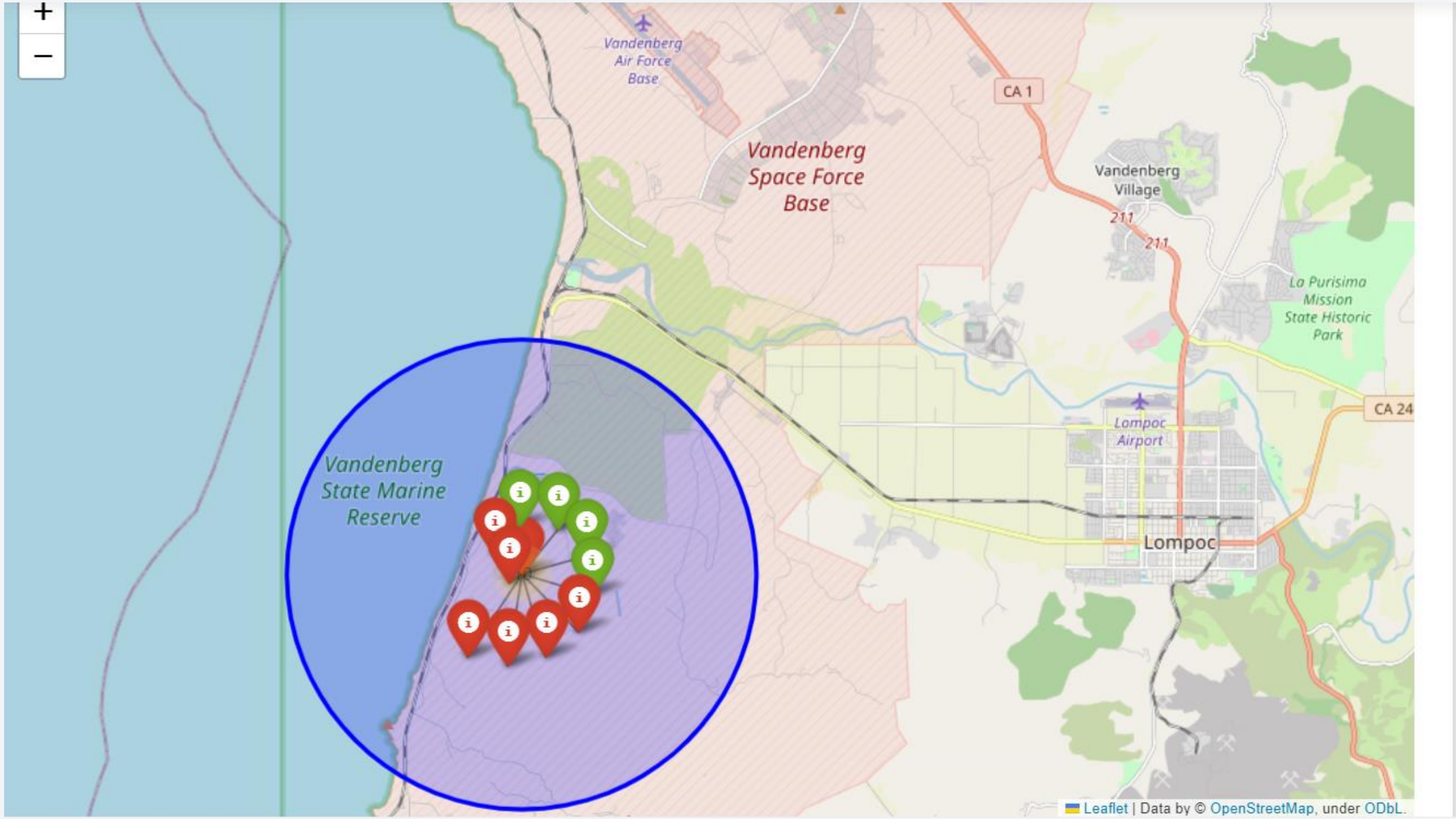
Launch Sites Proximities Analysis

Folium Map showing Launch sites



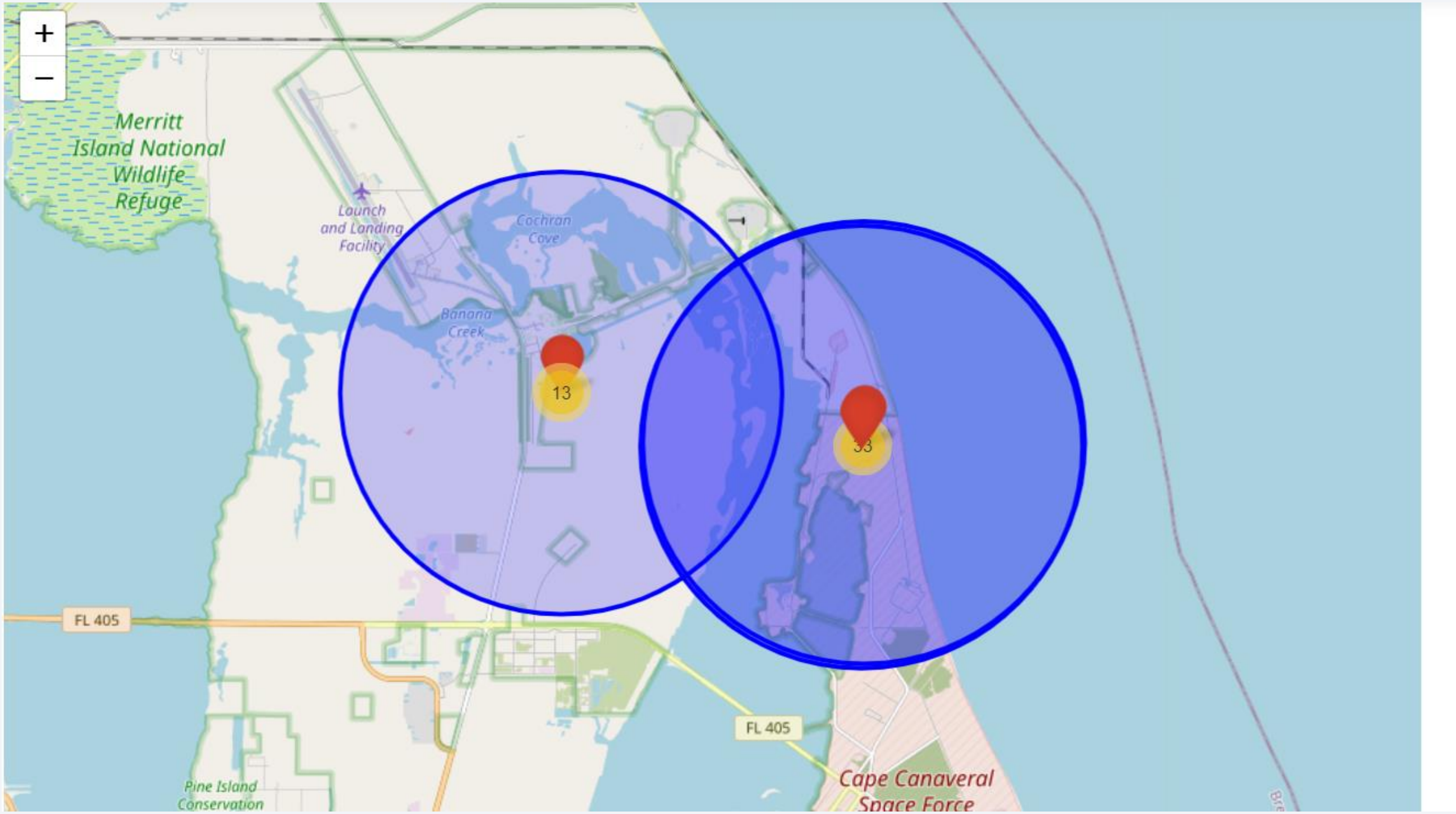
The Folium map showing Launch sites for SpaceX Falcon 9 missions

Folium Map showing mission outcome Launch sites



Folium map with clusters and markers indicating successful and failed mission launch sites

Folium Map showing Proximities

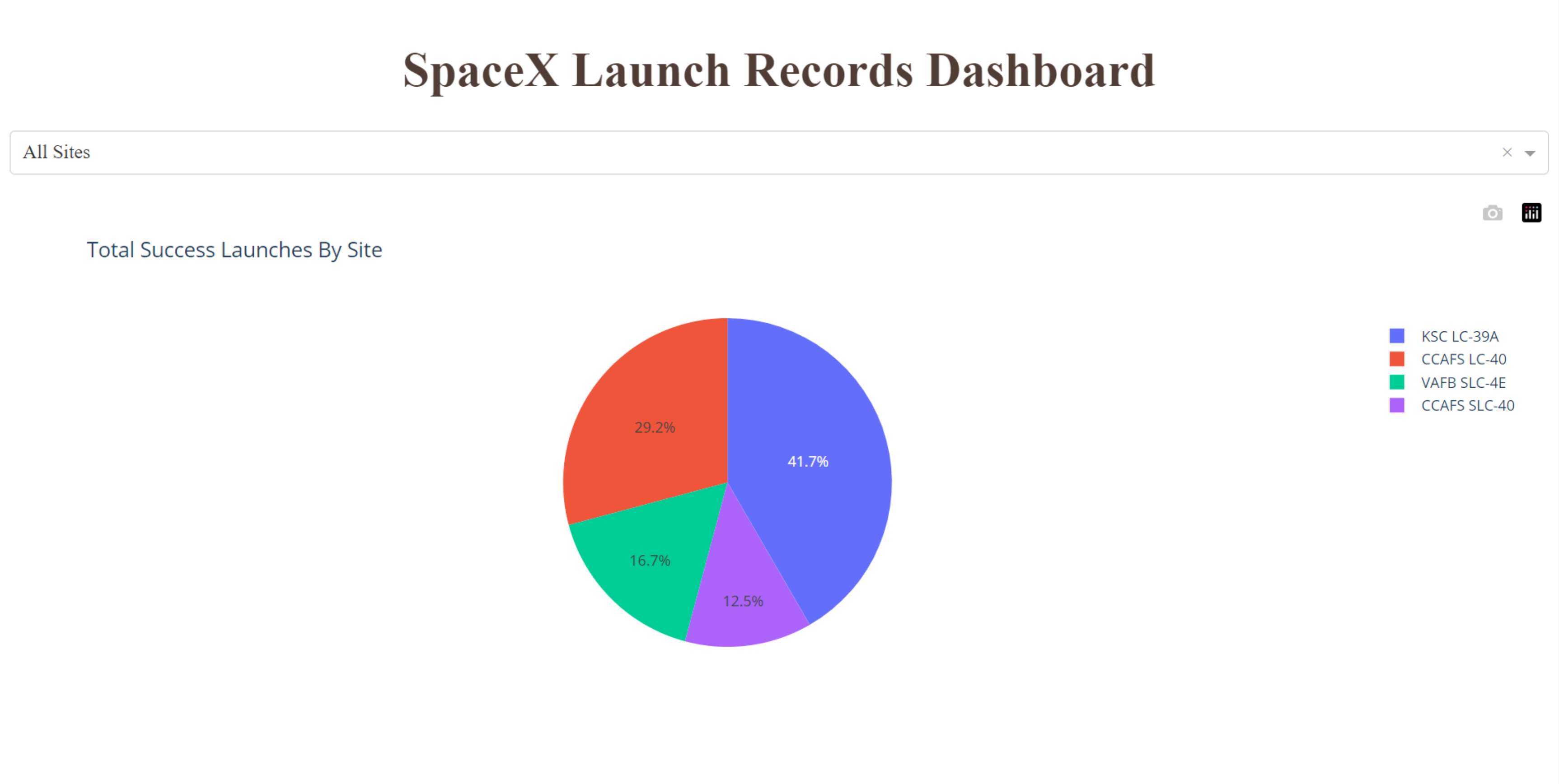


Folium map showing launch site proximities to highways, railroads, coastlines and cities

Section 4

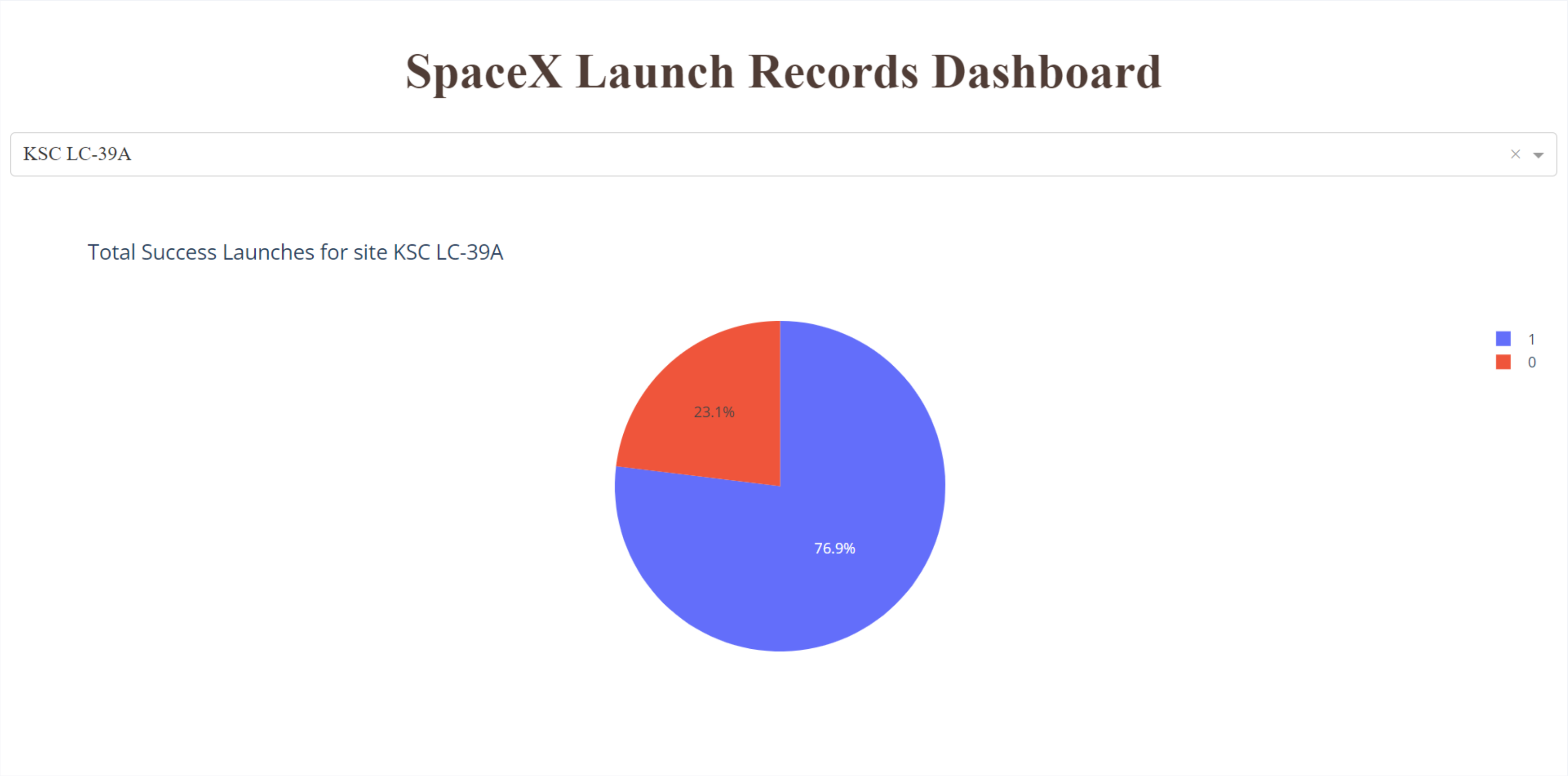
Build a Dashboard with Plotly Dash

SpaceX Dashboard showing Total success Launch by site



The pie chart shows total success launches for all launch sites with KSC LC 39A & CCAFS LC 40 performing best

Dashboard showing Pie chart of best Launch site



The pie chart shows the best performing launch site KSC LC-39A with 76.9% success & 23.1% failed missions

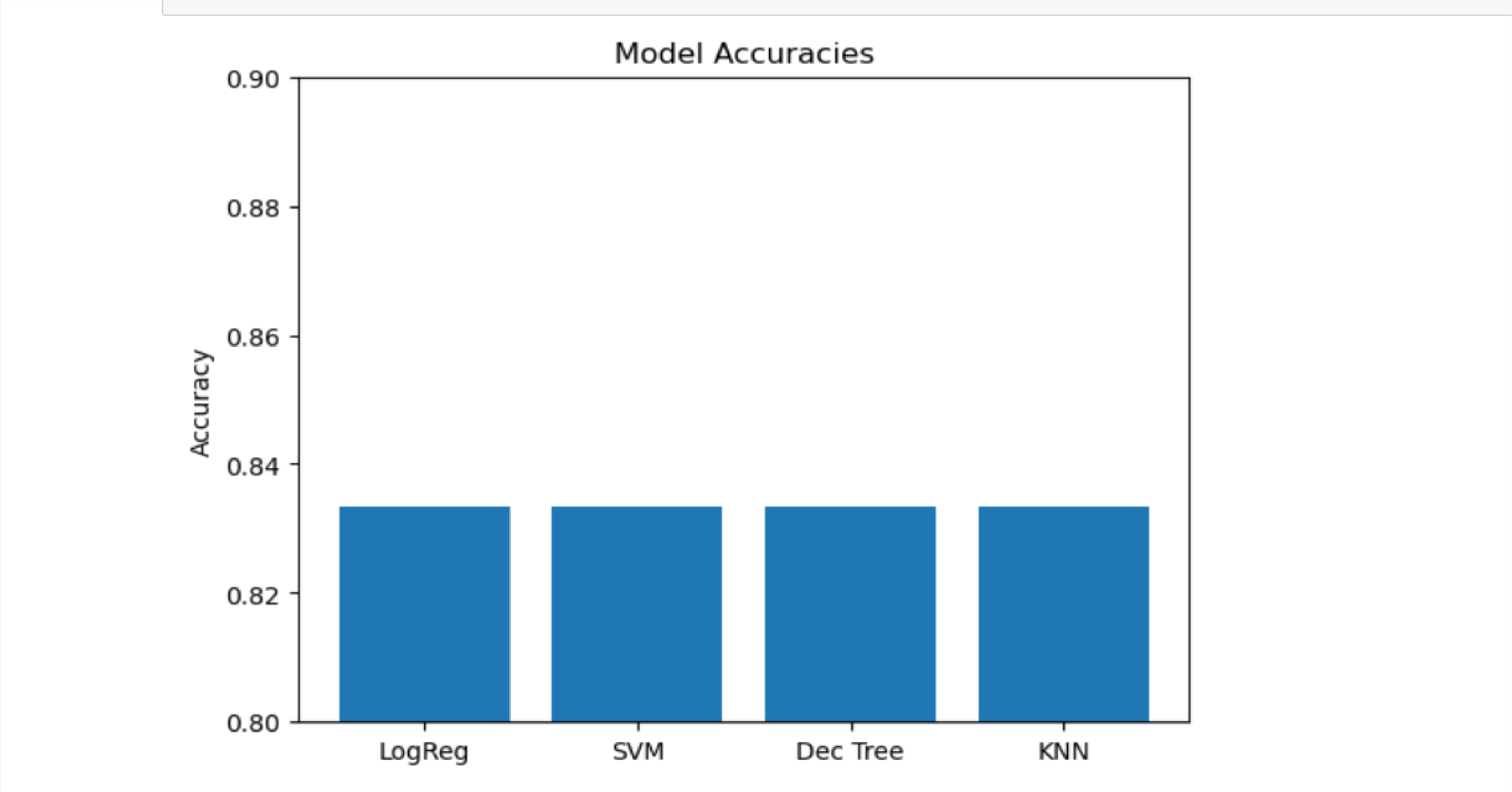
Dashboard showing Payload mass & Success rate

The scatter plot shows booster v1.0 and FT performing best with success from 0 to 10k and 6k maximum Payload mass respectively while others had success below 5k.

Section 5

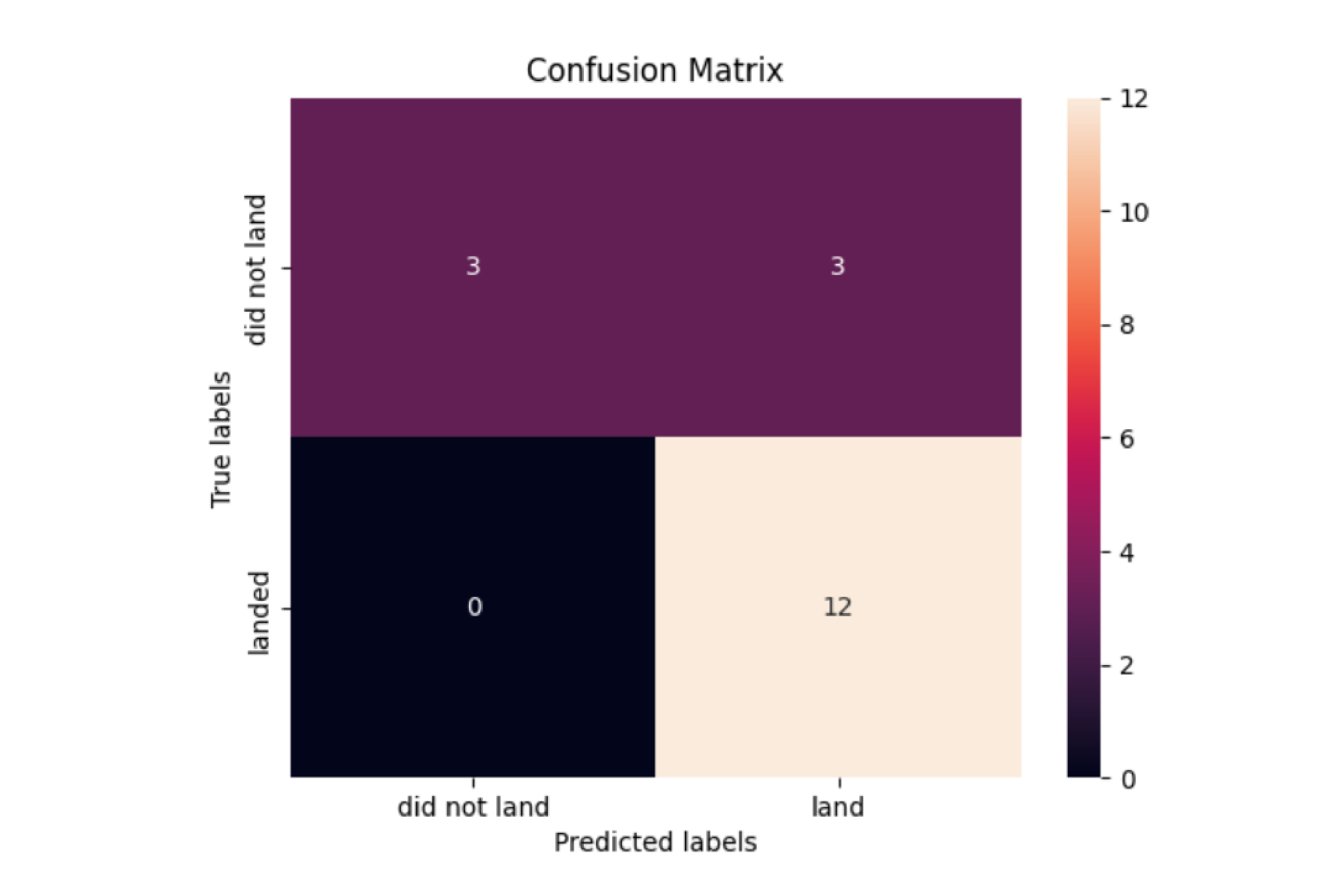
Predictive Analysis (Classification)

Classification Accuracy



All models had the same accuracy in the test set of 0.8333333333333334 while the Decision tree classifier had a higher accuracy in the training set with 0.9017857142857144

Confusion Matrix



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

Conclusions

- From the bar chart shown in the visualization plot the best Orbits with high success mission rates are ES LI, GEO, HEO and SSO
- Our scatter plot from the Plotly interactive dashboard shows most Booster version success had a Payload mass of 2,000 to 6,000 with FT being the best booster version and B4 being next with capacity of payload mass of 10,000
- From the Plotly pie chart we discovered KSC LC-39A to be the best launch site with 41.7% success rate, while CCAFS LC-40 had 29.2%, VAFB SLC-4E had 16.7 and CCAFS SLC-40 had the least with 12.5% success rate.
- Our Folium map shows successful launches were within proximities of highways, railroads and coastlines but not within proximities of any city.
- Our Classification model shows the Decision tree classifier was the best prediction model with the same test accuracy of 0.8333333333333334 as other models tested but had a higher training accuracy of 0.9017857142857144

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project
- In the SQL Lab an extra code to force install pandas to run the code was required

  “pip install pandas --force-reinstall”

- In the Folium map lab the “geopy” package library was unable to be installed after several attempts, hence geodesic distance was not calculated.

  “NameError: name 'geodesic' is not defined”

- All notebooks, codes and assets are available in the GitHub URL Link below.
- https://github.com/MoAbbazi/IBM-Data-science-capstone

Thank you!