IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Masahiro Nakata
August 30th, 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection methodology:
  - Perform data wrangling
  - Perform exploratory data analysis (EDA) using visualization and SQL
  - Perform interactive visual analytics using Folium and Matplotlib
  - Perform predictive analysis using classification models
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

- Project background and context
  - 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.

- Problems should be found answers
  - Predicting if the Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - By utilizing SpaceX API and Web Scraping
- Perform data wrangling
  - By utilizing Pandas
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Matplotlib
- Perform predictive analysis using classification models
  - By utilizing Scikit-learn

Data Collection

- How data sets were collected.
  - SpaceX API
  - Web Scraping
- Data collection process with key phrases and flowcharts
  - See subsequent slides

Data Collection – SpaceX API

- Collecting the data through SpaceX API and making sure the data is in the correct format.
- https://github.com/nakatama225/IBM_DS_Final_Project/

Flowchart in this process

Task 1: Request and parse the SpaceX launch data using the GET request

↓

Task 2: Filter the dataframe to only include Falcon 9 launches

↓

Task 3: Dealing with Missing Values

Data Collection - Scraping

- Web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled, List of Falcon 9 and Falcon Heavy launches.

- https://github.com/nakatama225/IBM_DS_Final_Project/

Flowchart in this process

TASK 1: Request the Falcon9 Launch Wiki page from its URL

↓

TASK 2: Extract all column/variable names from the HTML table header

↓

TASK 3: Create a data frame by parsing the launch HTML tables

Data Wrangling

- Performing some Exploratory Data Analysis (EDA) through Pandas to find some patterns in the data and determine what would be the label for training supervised models.

- https://github.com/nakatama225/IBM_DS_Final_Project/

Flowchart in this process

TASK 1: Calculate the number of launches on each site

↓

TASK 2: Calculate the number and occurrence of each orbit

↓

TASK 3: Calculate the number and occurrence of mission outcome per orbit type

↓

TASK 4: Create a landing outcome label from Outcome column

EDA with Data Visualization

- Employing Scatter plots, bar charts, and line graphs to visualize data
- https://github.com/nakatama225/IBM_DS_Final_Project/

EDA with SQL

- SQL queries you performed
  - SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
  - SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
  - SELECT SUM("Payload_Mass__kg_") FROM SPACEXTABLE WHERE "Customer"='NASA (CRS)';
  - SELECT AVG("Payload_Mass__kg_") FROM SPACEXTABLE WHERE "Booster_Version"='F9 v1.1';
  - SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome"='Success (ground pad)';
  - SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome"='Success (drone ship)' AND "Payload_Mass__kg_" BETWEEN 4000 AND 6000;
  - SELECT "Landing_Outcome", COUNT(*) FROM SPACEXTABLE GROUP BY "Landing_Outcome";
- https://github.com/nakatama225/IBM_DS_Final_Project/

Build an Interactive Map with Folium

We first need to create a folium `Map` object, with an initial center location to be NASA Johnson Space Center at Houston, Texas.

```
In [7]: # Start location is NASA Johnson Space Center
nasa_coordinate = [29.559684888503615, -95.0830971930759]
site_map = folium.Map(location=nasa_coordinate, zoom_start=10)
```

We could use `folium.Circle` to add a highlighted circle area with a text label on a specific coordinate. For example,

```
In [8]: # Create a blue circle at NASA Johnson Space Center's coordinate with a popup label showing its name
circle = folium.Circle(nasa_coordinate, radius=1000, color='#d35400', fill=True).add_child(folium.Popup('NASA Johnson Space Center'))
# Create a blue circle at NASA Johnson Space Center's coordinate with a icon showing its name
marker = folium.map.Marker(
    nasa_coordinate,
    # Create an icon as a text label
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % 'NASA JSC',
        )
    )
site_map.add_child(circle)
site_map.add_child(marker)
```

- Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.
- https://github.com/nakatama225/IBM_DS_Final_Project/

Build a Dashboard with Matplotlib

- Pie Chart and Scatter Plot
- To dive deeply more the visual analytics on SpaceX launch data

  * Originally, we planned to use Plotly and Dash, but since they could not be successfully executed in both cloud and local environments, we have adopted Matplotlib visualization.

- https://github.com/nakatama225/IBM_DS_Final_Project/

Predictive Analysis (Classification)

- Determined the best model by utilizing various modules of scikit-learn and comparing the prediction accuracy of the models.
- https://github.com/nakatama225/IBM_DS_Final_Project/

Flowchart in this process

TASK 1: Create a NumPy array from the column Class in data, by applying the method to_numpy() then assign it to the variable Y,make sure the output is a Pandas series (only one bracket df['name of column']).

↓

TASK 2: Standardize the data in X then reassign it to the variable X using the transform provided below.

↓

TASK 3: Use 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. The training data and test data should be assigned to the following labels.

↓

TASK 4: Create ML model object then create a GridSearchCV object "the model"_cv with cv = 10. Fit the object to find the best parameters from the dictionary parameters.

↓

TASK 5: Calculate the accuracy on the test data using the method score of "the model" created.

Results

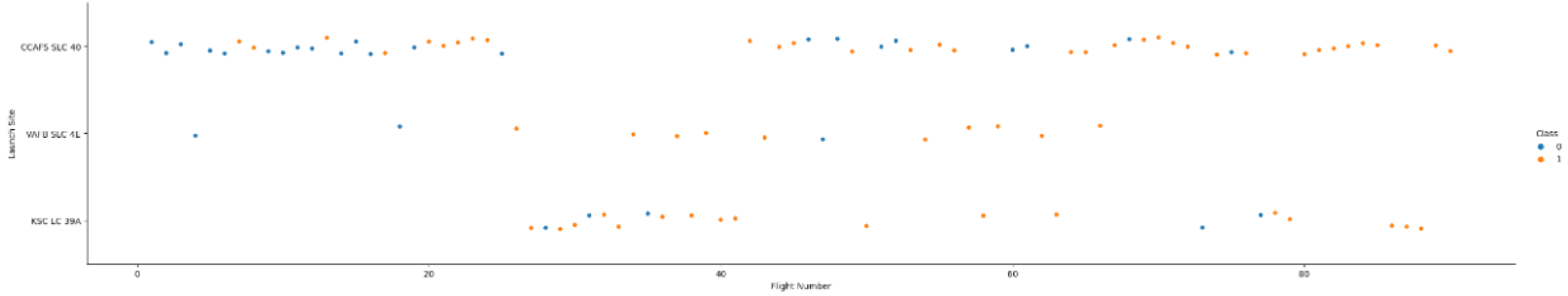
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

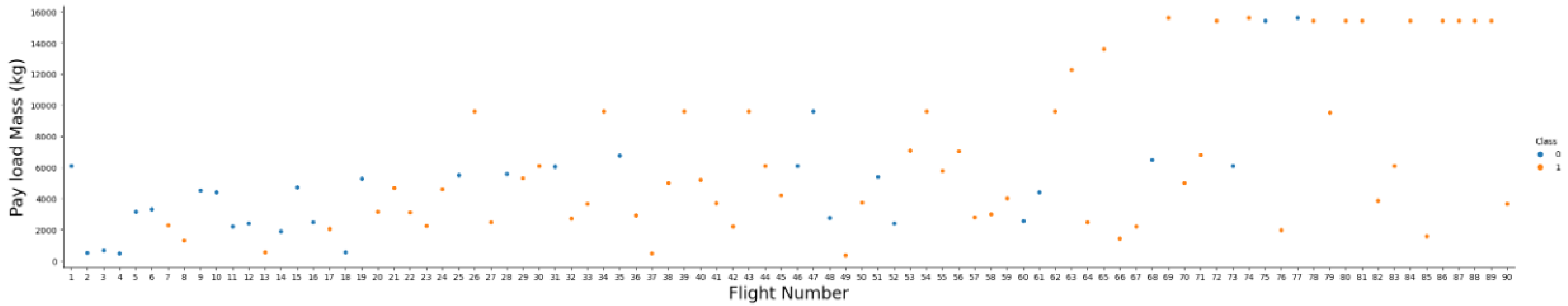
```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect=5)
plt.xlabel("Flight Number")
plt.ylabel("Launch Site")
plt.show()
```



- Regardless of site, it appears that the probability of success is higher with a greater number of flights.

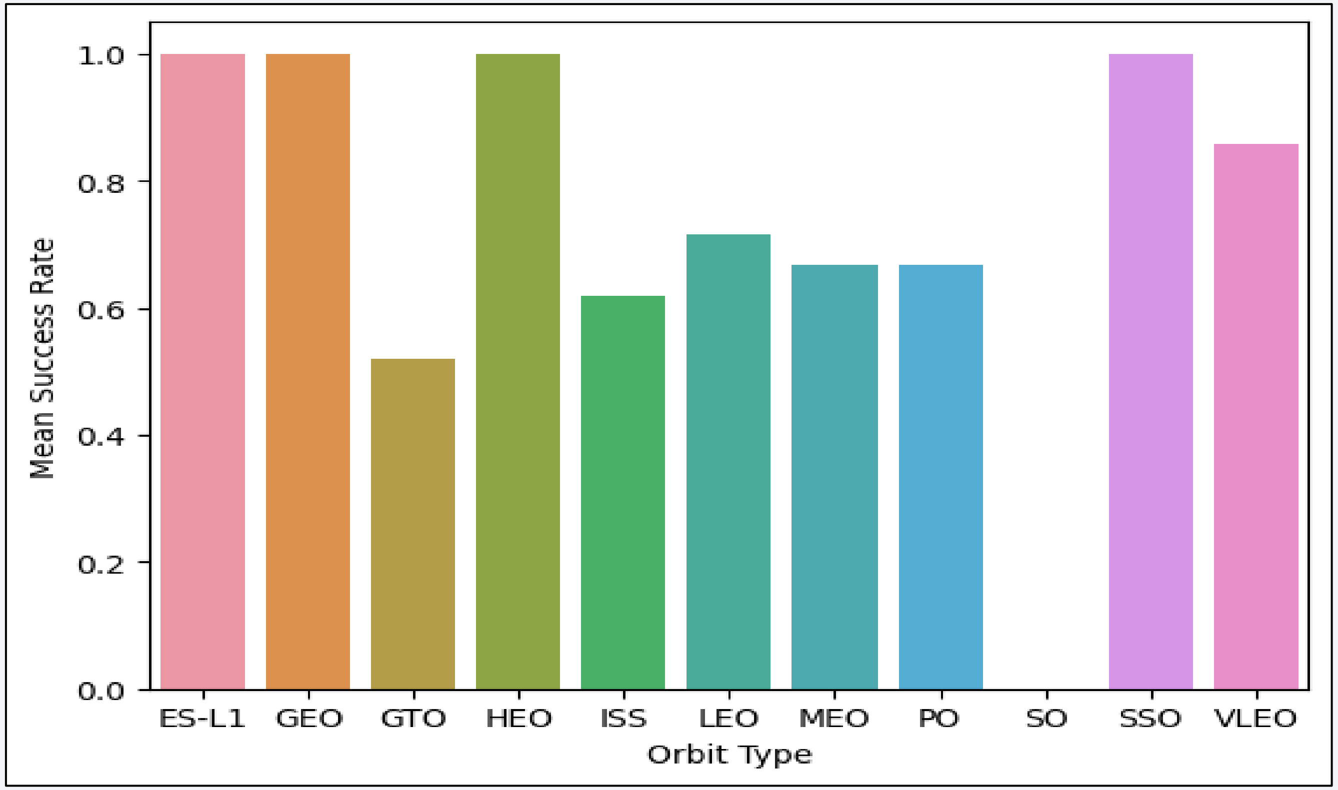
Payload vs. Launch Site

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect=5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



- There appears to be a positive correlation.

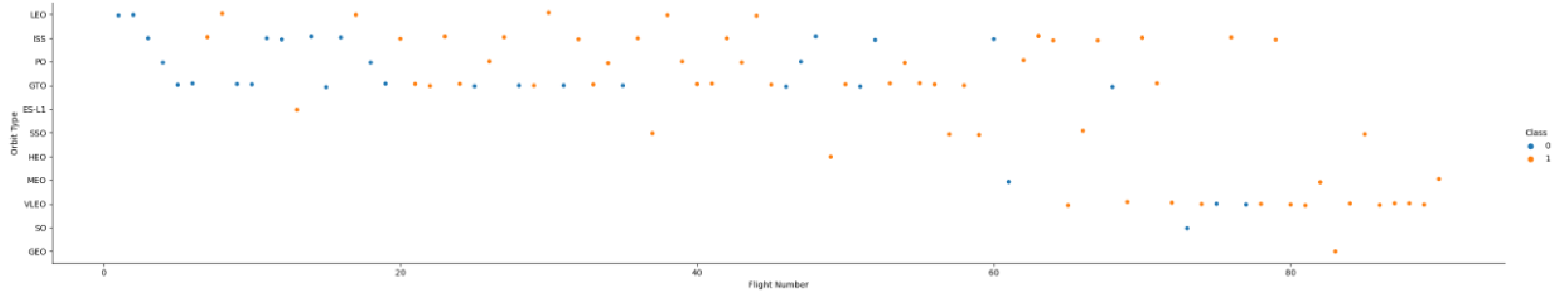
Success Rate vs. Orbit Type



- There appears to be a difference in the success rate depending on the orbit type.

Flight Number vs. Orbit Type

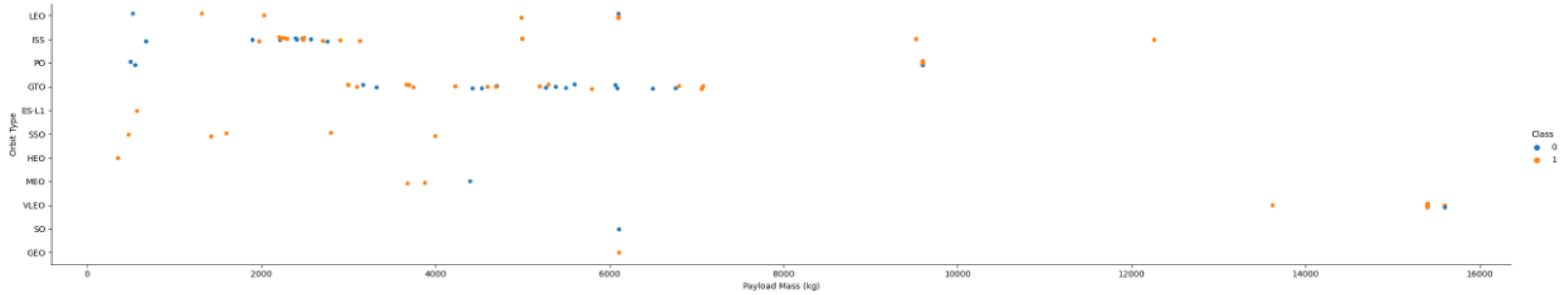
```python
### TASK  4: Visualize the relationship between FlightNumber and Orbit type
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect=5)
plt.xlabel("Flight Number")
plt.ylabel("Orbit Type")
plt.show()
```



- Regardless of orbit type, it appears that the probability of success is higher with a greater number of flights.

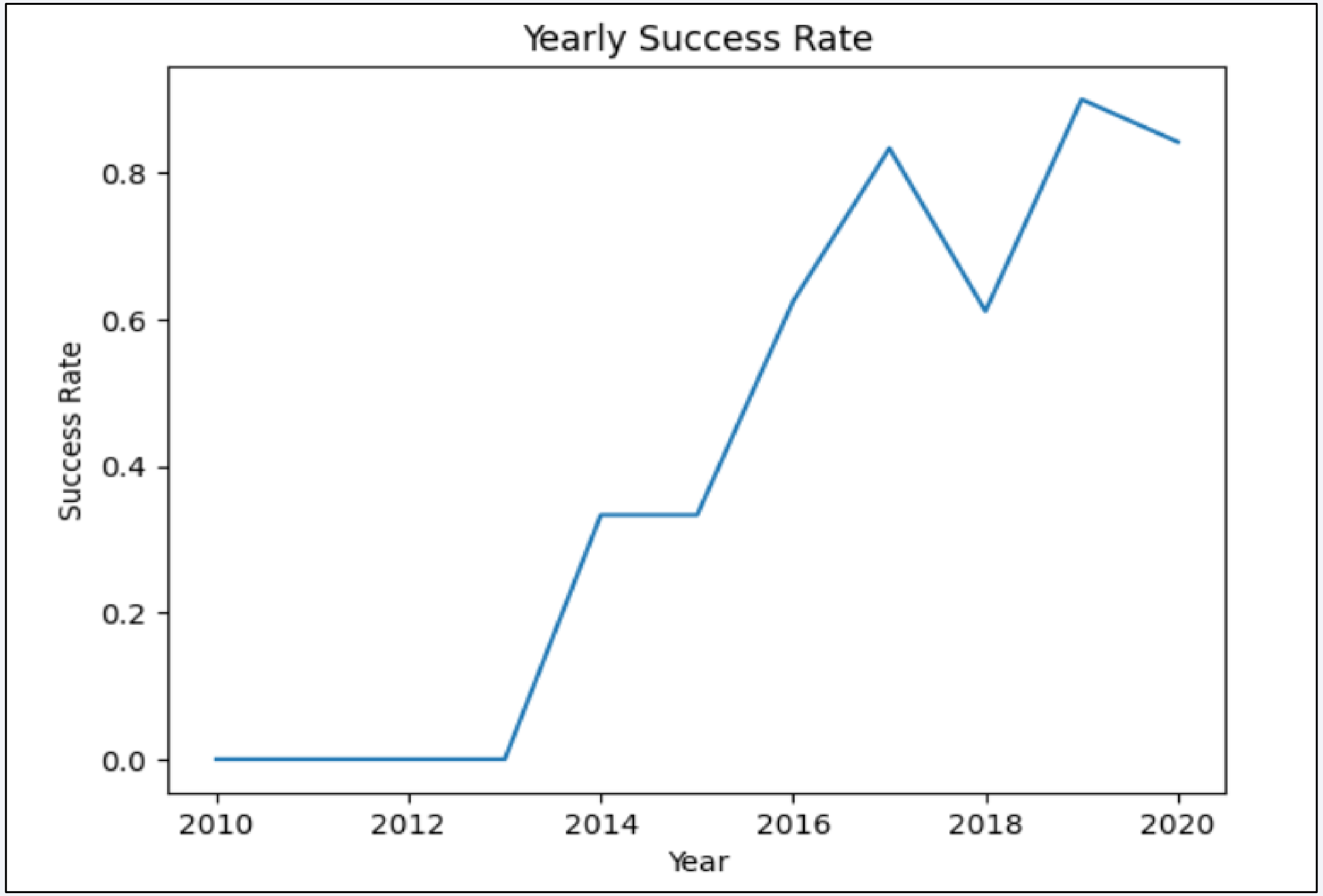
Payload vs. Orbit Type

```
### TASK  5: Visualize the relationship between Payload and Orbit type
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect=5)
plt.xlabel("Payload Mass (kg)")
plt.ylabel("Orbit Type")
plt.show()
```



- No clear overall tendency can be seen, but there may be a tendency for each orbit type.

Launch Success Yearly Trend



- There appears to be seen that the success rate is high as the newer years.

All Launch Site Names

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

```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

```
 * sqlite:///my_data1.db
Done.
```

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

- There appears to be seen four main launch sites.

Launch Site Names Begin with 'CCA'

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

```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- As the figure shows.

Total Payload Mass

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

```
%sql SELECT SUM("Payload_Mass__kg_") FROM SPACEXTABLE WHERE "Customer"='NASA (CRS)';
```

```
 * sqlite:///my_data1.db
Done.
```

SUM("Payload_Mass__kg_")
45596

- As the figure shows.

Average Payload Mass by F9 v1.1

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

```
%sql SELECT AVG("Payload_Mass__kg_") FROM SPACEXTABLE WHERE "Booster_Version"='F9 v1.1';
```

```
 * sqlite:///my_data1.db
Done.
```

AVG("Payload_Mass__kg_")
2928.4

- As the figure shows.

First Successful Ground Landing Date

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

Hint:Use min function

```
%sql SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome"='Success (ground pad)';
```

```
 * sqlite:///my_data1.db
Done.
```

MIN("Date")
2015-12-22

- As the figure shows.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome"='Success (drone ship)' AND "Payload_Mass__kg_" BETWEEN 4000 AND 6000;
```

```
 * sqlite:///my_data1.db
Done.
```

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

- Regarding Drone Ship Landing with Payload between 4000 and 6000, there appears to be seen that the Booster version starting with "F9 FT B10" is extracted.

Total Number of Successful and Failure Mission Outcomes

- When success and failure are compared, it appears that success is more common.
- Also, there appears to be that No attempt is seen not quite a few.

List the total number of successful and failure mission outcomes

```
%sql SELECT "Landing_Outcome", COUNT(*) FROM SPACEXTABLE GROUP BY "Landing_Outcome";
```

```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	COUNT(*)
Controlled (ocean)	5
Failure	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	21
No attempt	1
Precluded (drone ship)	1
Success	38
Success (drone ship)	14
Success (ground pad)	9
Uncontrolled (ocean)	2

Boosters Carried Maximum Payload

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

```
%sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Payload_Mass__kg_" = (SELECT MAX("Payload_Mass__kg_") FROM SPACEXTABLE);
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version
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

- As a result, there appears to be seen that the Booster version starting with "F9 B5 B10" is extracted.

2015 Launch Records

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.

```
%sql SELECT *, SUBSTR(Date, 4, 2) as month FROM SPACEXTABLE WHERE "Landing_Outcome" LIKE 'Failure%' AND SUBSTR(Date, 7, 4)='2015';
```

```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome	month

- As the figure shows, the result number of data is zero.

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.

```
%sql SELECT "Landing_Outcome", COUNT(*) as count
FROM SPACEXTABLE WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome" ORDER BY count DESC;
```

```
 * sqlite:///my_data1.db
Done.
```

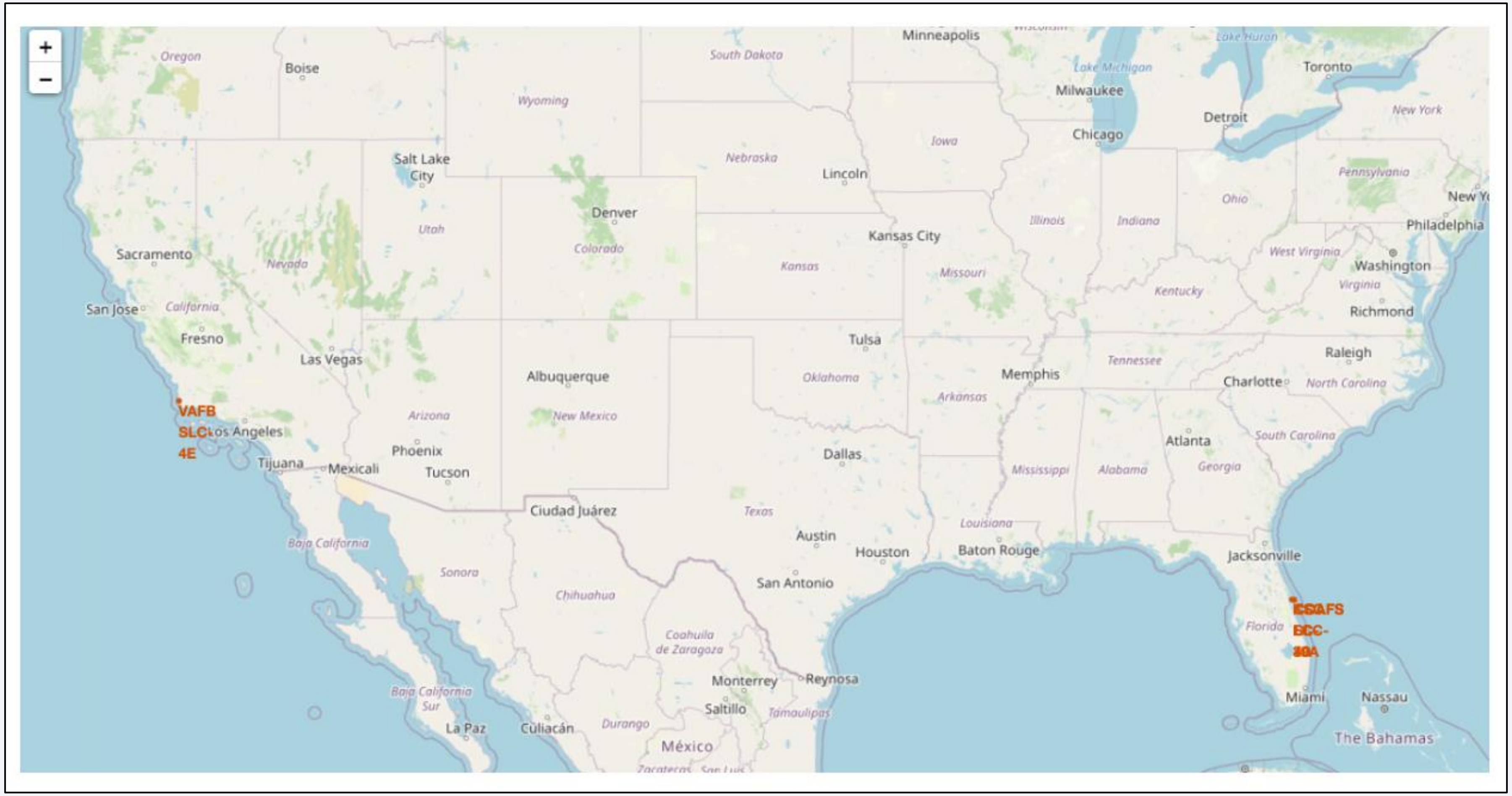
Landing_Outcome	count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

- There appears to be seen that No attempt is the highest number in this query conditions.

Section 3

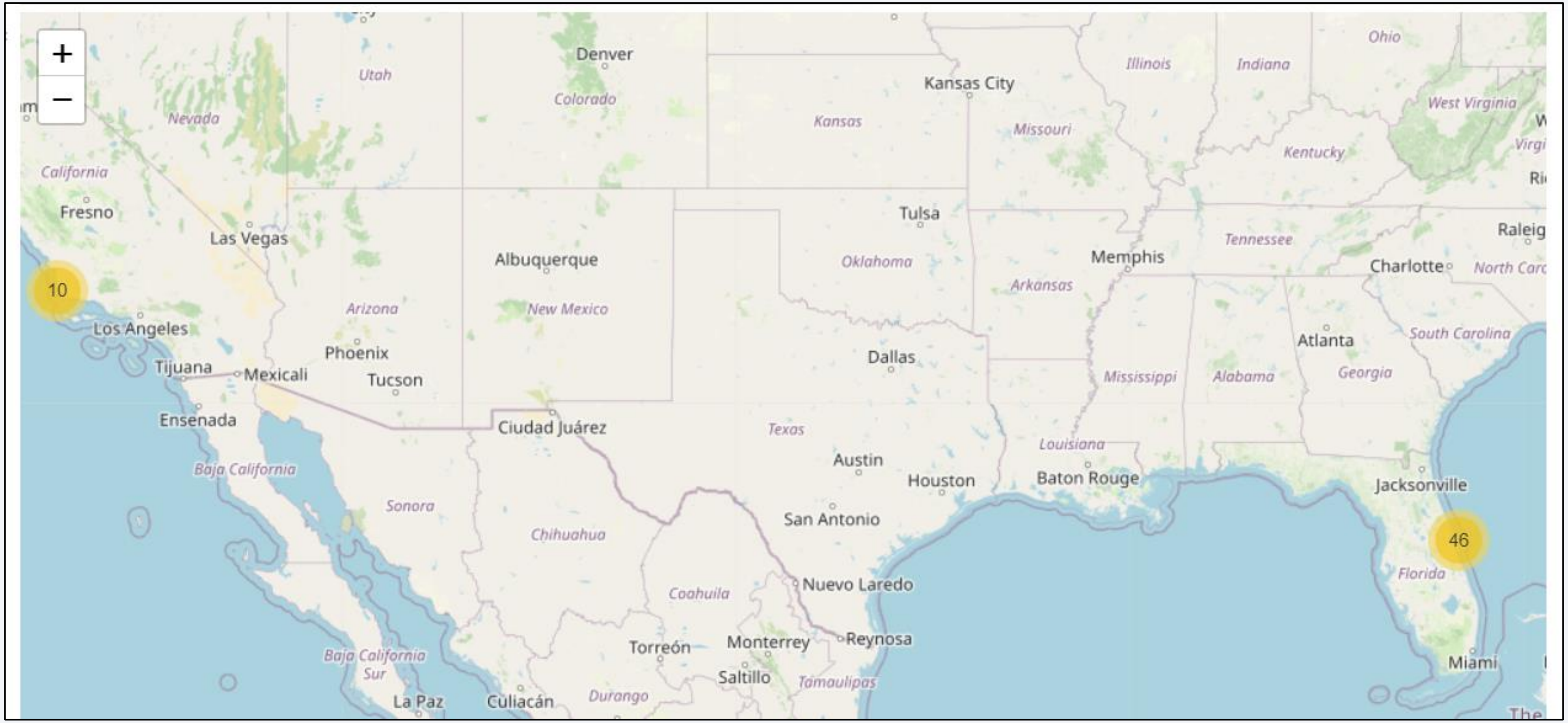
Launch Sites Proximities Analysis

All launch sites on a global map



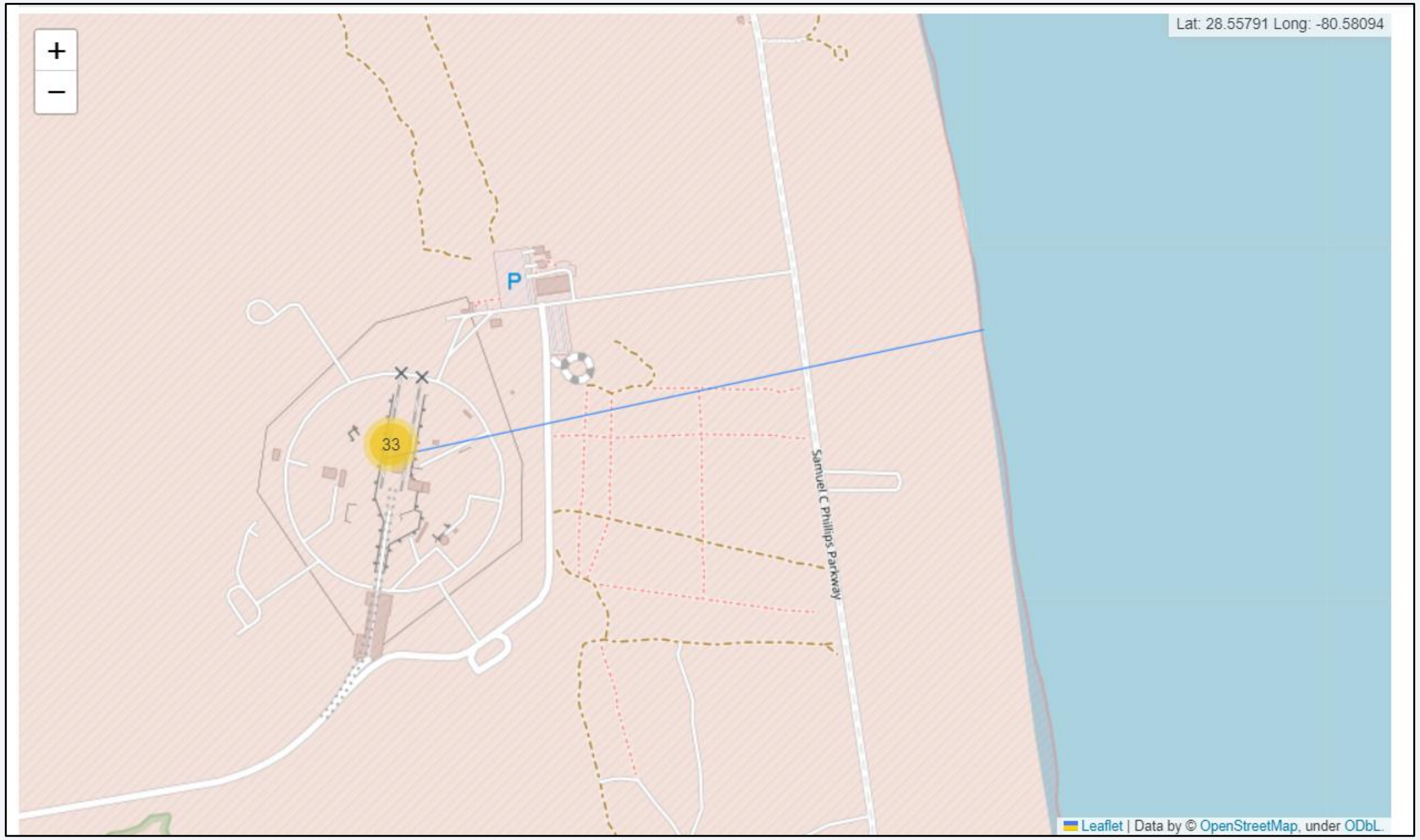
- It can be seen that the launch sites are located on the west and east coasts respectively.

All launch sites count on a global map



- It can be seen that there are 46 data points on the East Coast and 10 on the West Coast.

Straight line from 33 data points on the East Coast to the nearest coastline

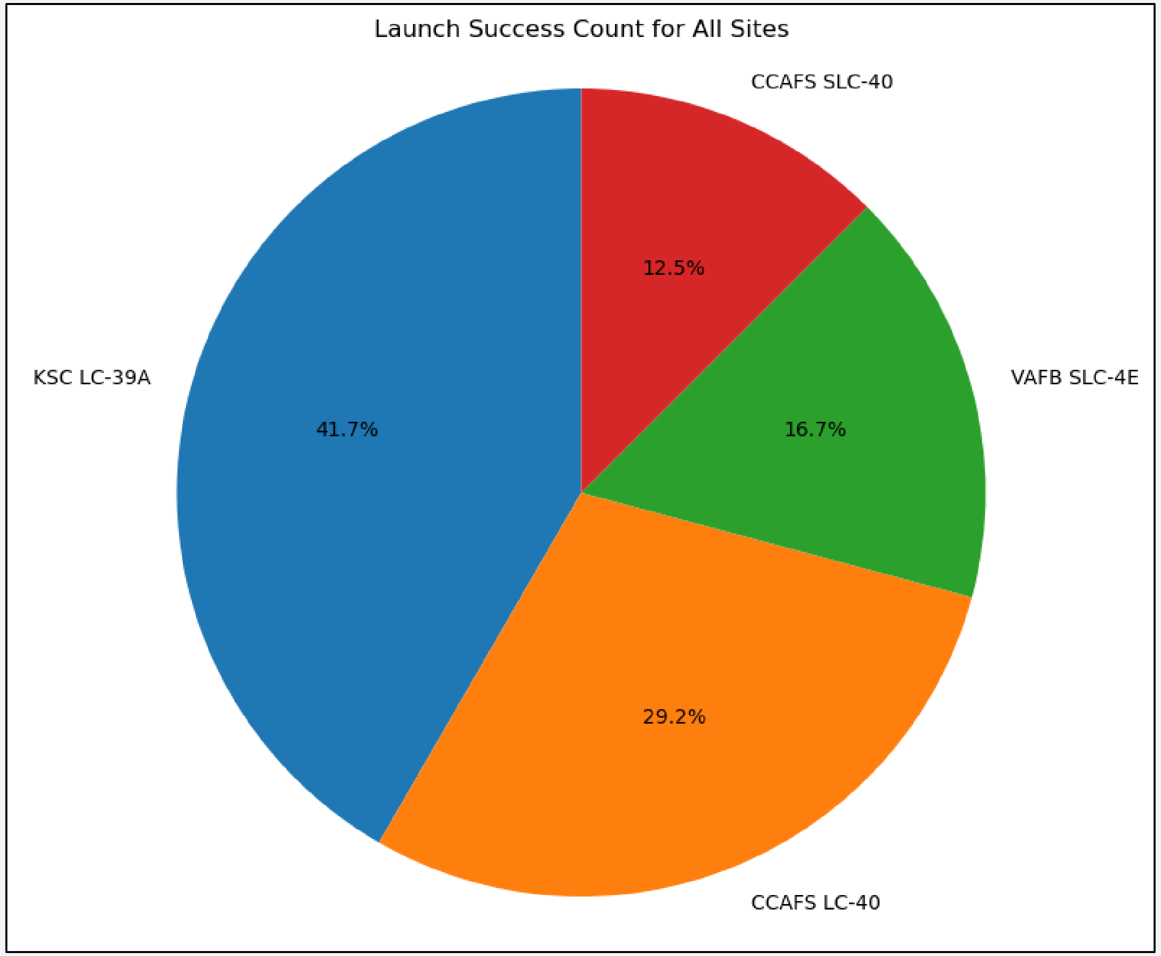


- It can be seen that a straight line from 33 data points on the East Coast to the nearest coastline.

Section 4

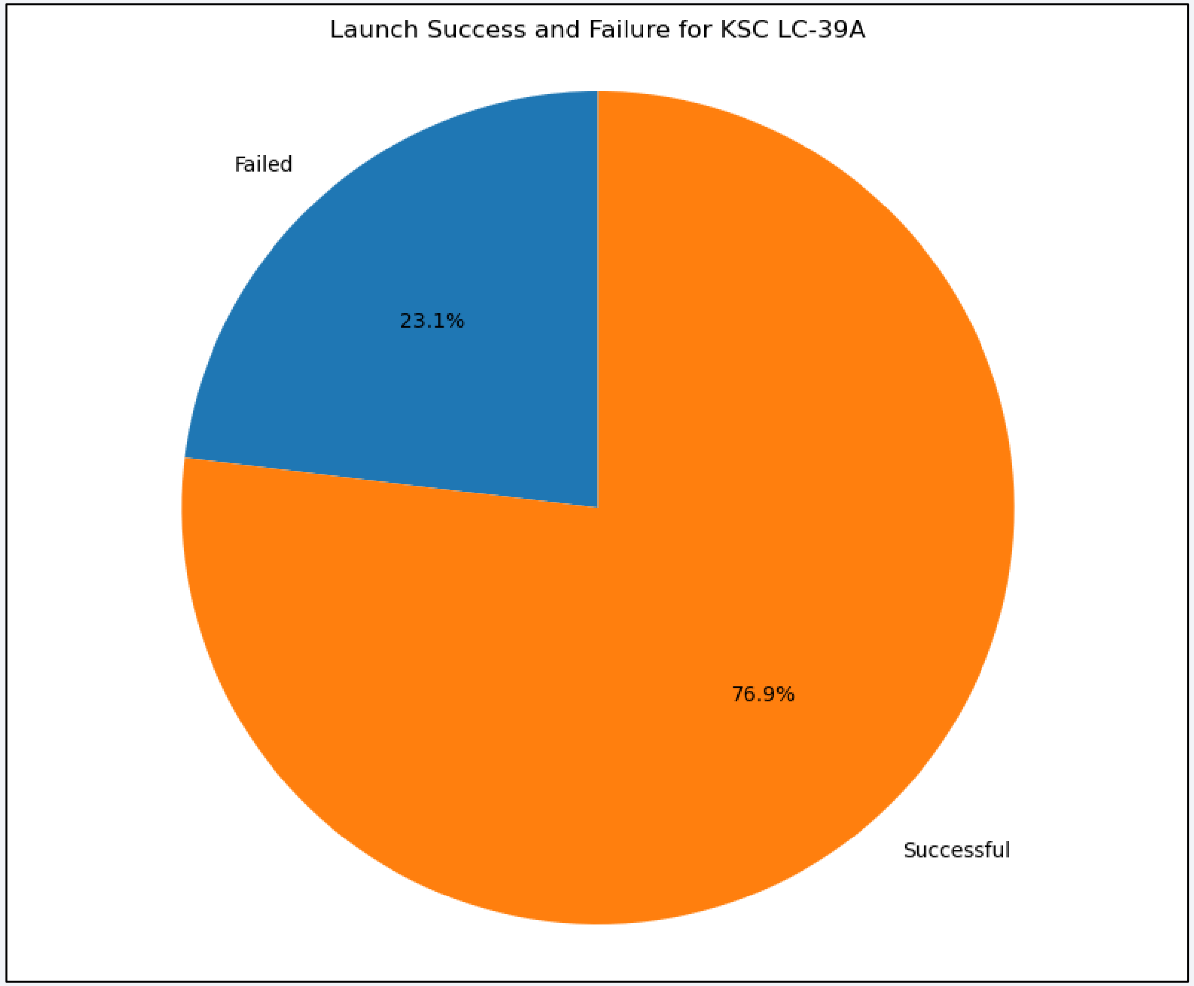
Build a Dashboard with Plotly Dash

Launch success count for all sites



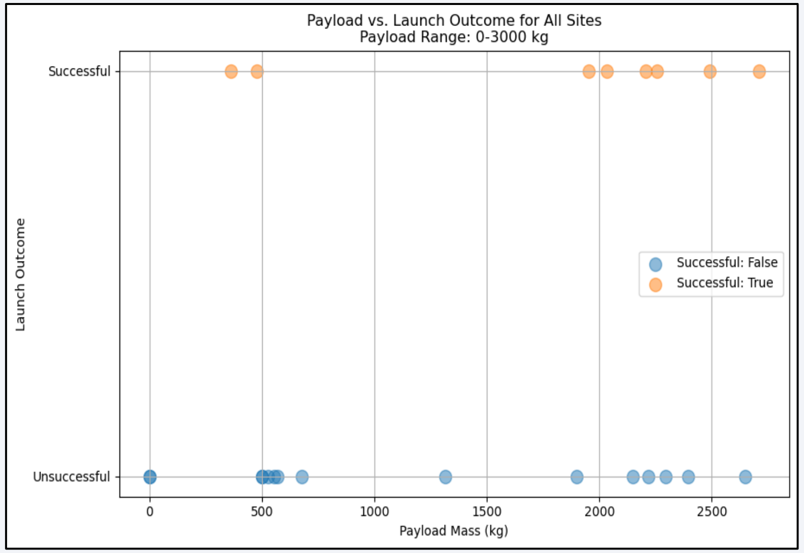
- It can be seen that "KSC LC-39A" accounts for 40% of the total.

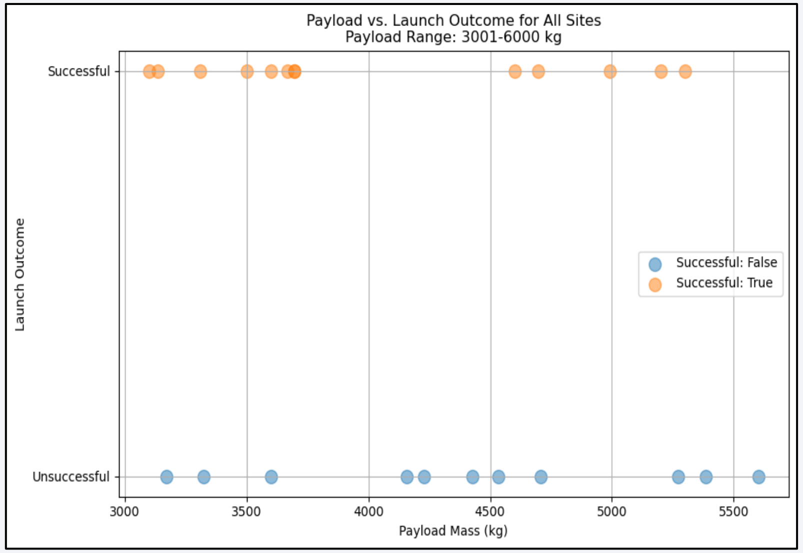
Piechart for the launch site with highest launch success ratio

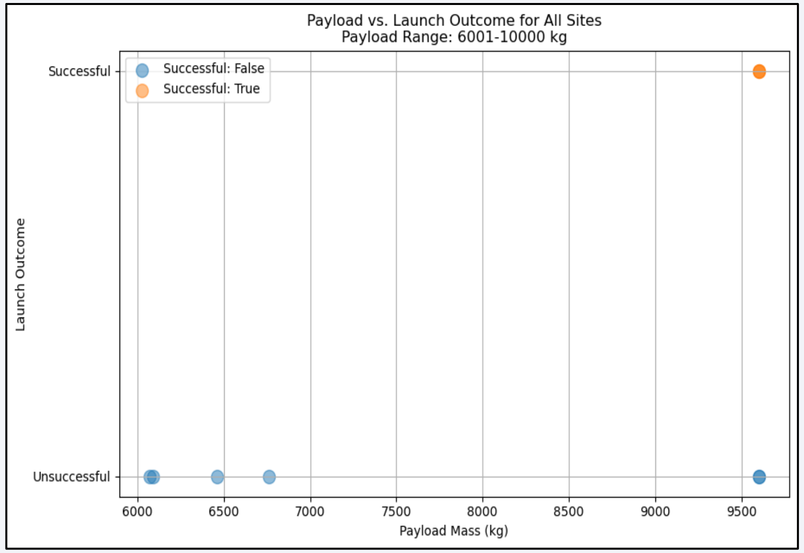


- In "KSC LC-39A“, it can be seen that more than 70% succeeded.

Payload vs. Launch Outcome for different payload ranges







- It can be seen that the higher the Payload Mass, the more a little likely it is to be successful.
- It can be seen that the lower the Payload Mass, the more a little likely it is to be successful.
- It can be seen that it is difficult to succeed regardless of Payload Mass.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

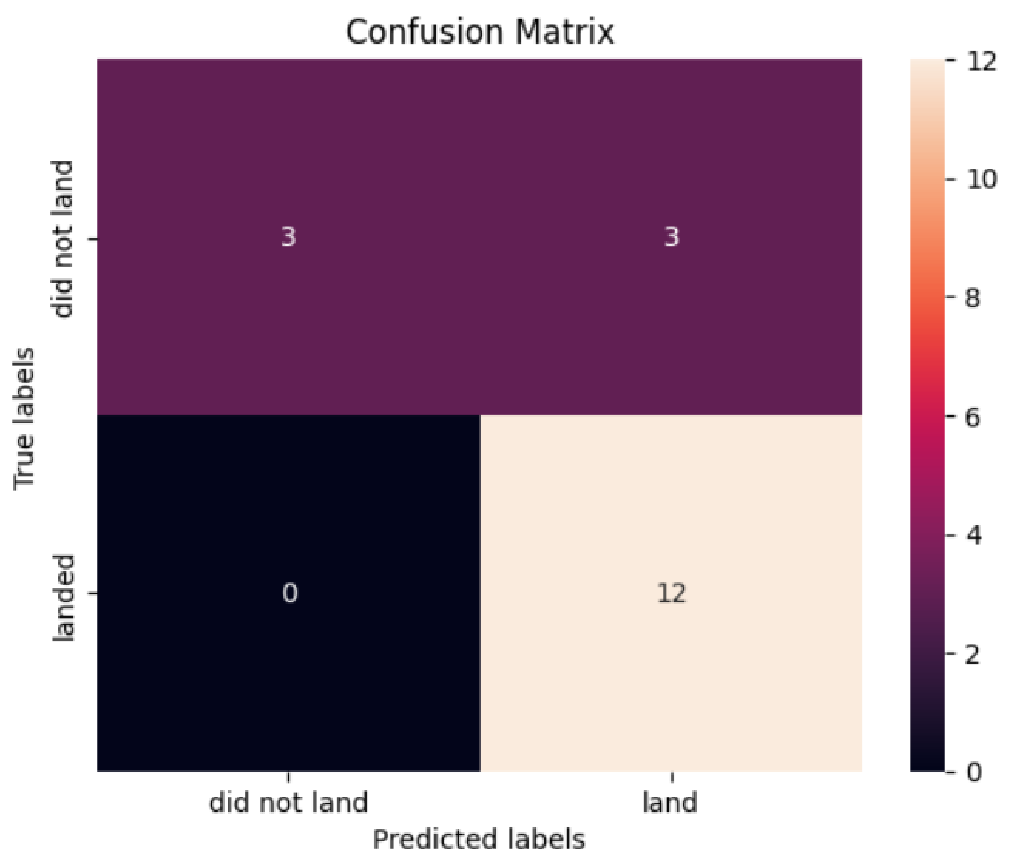
	Model	Best Score (Validation)	Test Accuracy
0	Logistic Regression	0.846429	0.833333
1	Support Vector Machine	0.848214	0.833333
2	Decision Tree	0.889286	0.833333
3	K-Nearest Neighbors	0.848214	0.833333

- It can be seen that the decision tree model had the highest score among the models executed in this project.

Confusion Matrix

- It can be seen that the overall prediction accuracy is high, and in this test data, the actual successful data is accurately predicted.

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



Conclusions

- Within the scope of the data set in this project, the prediction by the decision tree model was found to be effective.
- Applying these model predictions to future launch projects would allow for the calculation of expected costs associated with success or failure.

- On the other hand, for future issues, it should be noted that the data sample size for this project is not necessarily large and that a statistically detailed analysis has not been conducted.

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
- https://github.com/nakatama225/IBM_DS_Final_Project/

Thank you!