IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Adam Lim Han Jung>
<24/2/2025>

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection API with web scraping
  - Data wrangling
  - Exploratory data analysis using SQL
  - Exploratory data analysis with data visualization
  - Interactive visual analytics using Folium and Plotly Dash
  - Machine learning prediction
- Summary of all results
  - Exploratory data analysis result
  - Interactive analytics in screenshots
  - Predictive analytics result

Introduction

- Project background and context
  - The commercial space industry is evolving, with companies like SpaceX leading due to reusable rocket technology, significantly reducing launch costs. Falcon 9's first stage is crucial and expensive, but not always recovered. Therefore, the goal is to predict if the first stage will land successfully to determine the cost of the launch, using Machine Learning
- Problems you want to find answers
  - Feature Interactions: Relationships between factors (e.g., payload, orbit, weather, launch site) significantly impact landing success.
  - Orbit Influence: Certain orbits have higher landing success rate.
  - Optimal Machine Learning Model: Identifying the best algorithm for predicting first-stage landing success.

Section 1

Methodology

Methodology

- Data collection methodology:
  - Using SpaceX Rest API
  - Using Web Scrapping from Wikipedia
- Perform data wrangling
  - Filtering the data
  - Dealing with missing values
  - Using One Hot Encoding to prepare the data to binary classification
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Building, tuning and evaluation of the classification models

Data Collection

- API Data Retrieval
  - Used GET request to fetch the data from the SpaceX API
  - Response is received in JSON format and converted using json_normalize() into a pandas DataFrame
- Data Cleaning
  - Identified and handled missing values where necessary
- Web Scraping
  - Extracted Falcon 9 launch records from Wikipedia using Beautiful Soup.
  - Parsed the HTML table and converted it to a pandas DataFrame for future analysis.

Data Collection – SpaceX API

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

In [7]:

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

In [8]:

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

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

In [13]:

```
# Use json_normalize meethod to convert the json result into a dataframe
json_response = response.json()
data = pd.json_normalize(json_response)
data
```

In [29]:

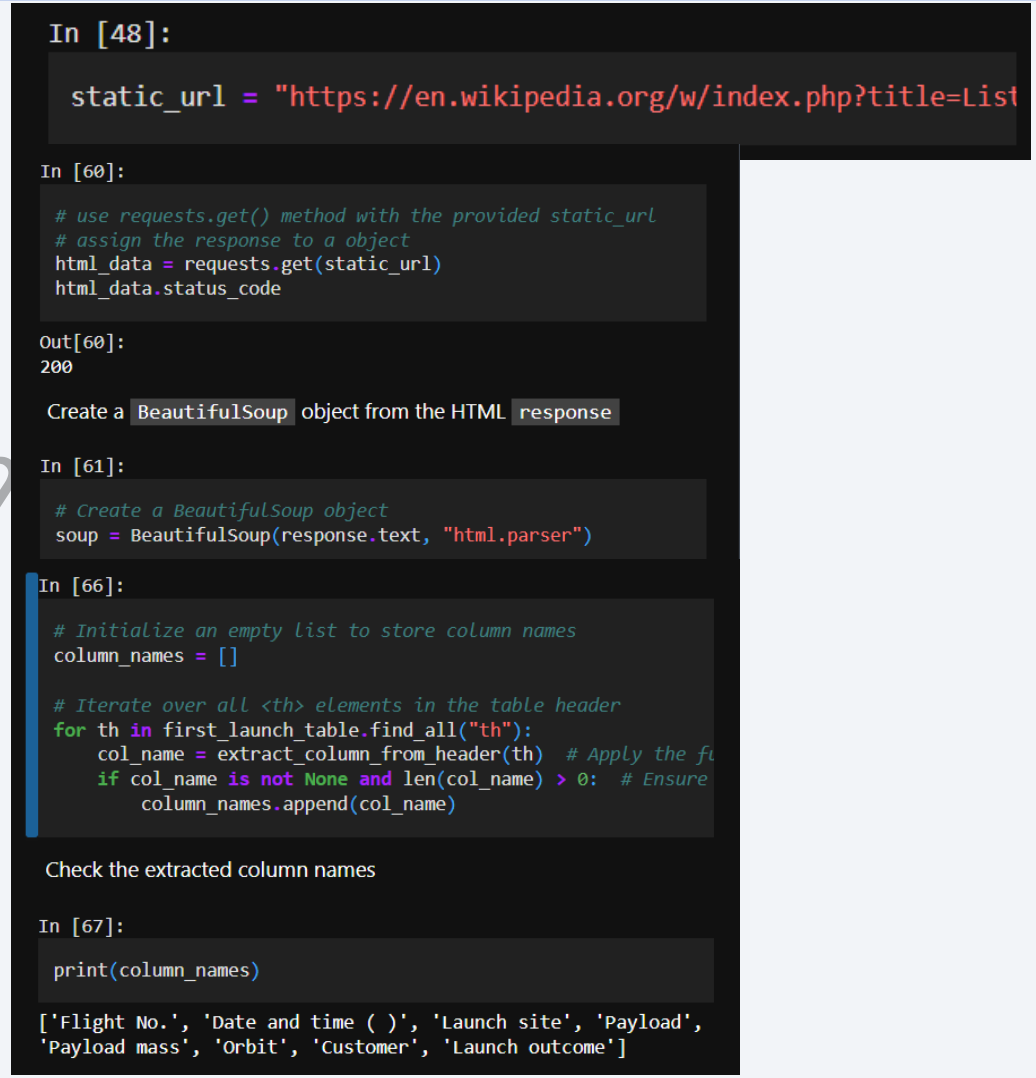
```
# Calculate the mean value of PayloadMass column
x = data["PayloadMass"].mean()
data["PayloadMass"].replace(np.nan,x)
# Replace the np.nan values with its mean value
```

- The GET request function is used to retrieve the data, then it is turned into pandas data frame and then used for calculation

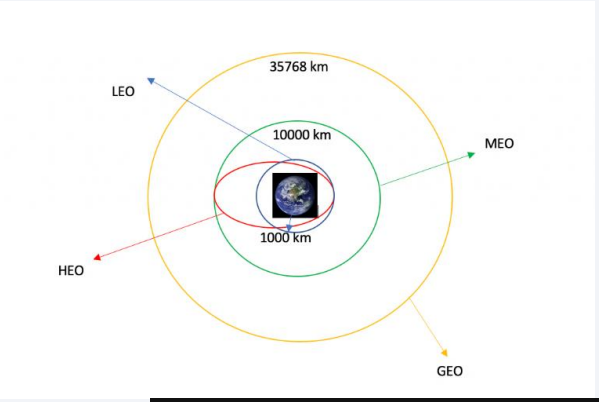
- [IBM_Data_Science_CS/jupyter-labs-spacex-data-collection-api-v2 (1).ipynb at main · Jung028/IBM_Data_Science_CS]

Data Collection - Scraping

- Then web scaping was done on the Falcon 9 launch records using Beautiful Soup Library
- Then parsed the table and converted to pandas data frame for extraction of the column names
- [IBM_Data_Science_CS/jupyter-labs-webscraping (1).ipynb at main · Jung028/IBM_Data_Science_CS]



Data Wrangling



TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

In [13]:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
# Define bad outcomes
bad_outcome = {"Failure", "Fail", "Lost", "Destroyed"}

# Create landing_class: 0 if outcome is in bad_outcome, otherwise 1
landing_class = df["Outcome"].apply(lambda x: 0 if any(bad in x for bad in bad_out

# Display the first few values
landing_class.head()
```

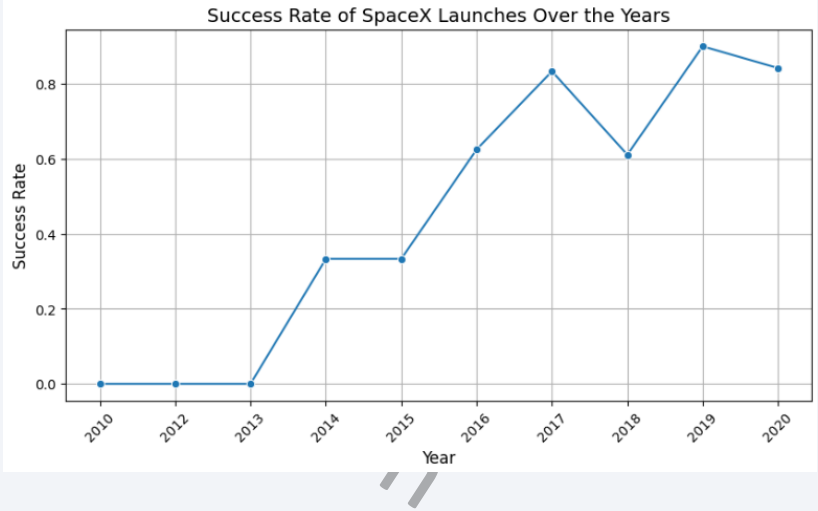
Out[13]:

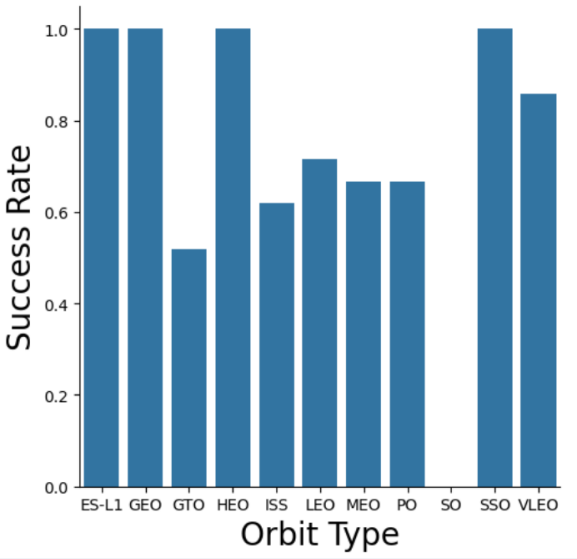
```
0    1
1    1
2    1
3    1
4    1
Name: Outcome, dtype: int64
```

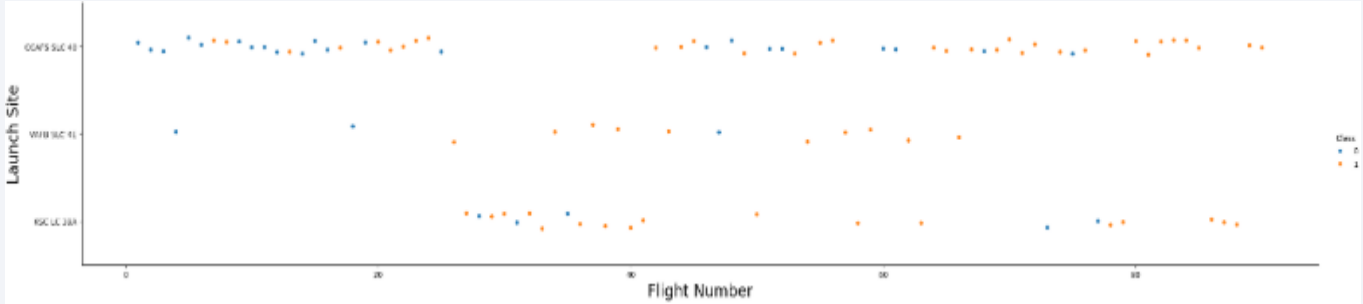

- To determine the training labels, data analysis exploration is done
- Calculate the number of occurrences of each orbit, number of launches on each site.
- Create a landing outcome label from 'Outcome' column
- Export data to csv
- [IBM_Data_Science_CS/jupyter-labs-eda-sql-coursera_sqllite.ipynb at main · Jung028/IBM_Data_Science_CS]

EDA with Data Visualization

- Then visualize the data, with
- Payload and launch site diagram,
- Flight number and launch site,
- Success rate of each orbit type
- Launch success per year
- [IBM_Data_Science_CS/edadataviz (1).ipynb at main · Jung028/IBM_Data_Science_CS]







EDA with SQL

- Call the SQL queries such as :
  - Names of unique launch sites in the space mission
  - Total payload mass caried by boosters launched by “NASA (CRS)”
  - Names of boosters successfully landed on drone ship with payload of 4000-6000kg
  - Number of “successful” and “failure” mission results
  - Data of first successful landing on the ground pad
  - 5 records where launch sites start with “CCA”
  - List the booster versions that have carried the maximum payload mass.
  - Display records showing the month names, failed landing outcomes on drone ships, booster versions, and launch sites for each month in 2015.
  - Rank the number of successful landing outcomes between June 4, 2010, and March 20, 2017, in descending order.

- IBM_Data_Science_CS/jupyter-labs-eda-sql-coursera_sqllite.ipynb at main · Jung028/IBM_Data_Science_CS

Build an Interactive Map with Folium

- Obtained the latitude and longitude coordinates for each launch site and added circle markers with name labels to visualize all launch sites on an interactive map.
- Marked the success or failure of launches at each site on the map, assigning 0 for failed launches and 1 for successful launches.
- Used `MarkerCluster()` to add marker colors (red for 0, green for 1) for these classes.
- Calculated the distances between each launch site and its proximities, answering the following questions
  - How close are the launch sites to railways, highways, and coastlines?
  - How close are the launch sites to nearby cities?
- IBM_Data_Science_CS/lab_jupyter_launch_site_location_.ipynb at main · Jung028/IBM_Data_Science_CS

Build a Dashboard with Plotly Dash

- Developed an interactive dashboard using Plotly Dash.
- Created pie charts to display the total number of launches at each site.
- Plotted scatter graphs to show the relationship between “Outcome” and “Payload Mass (kg)” for different booster versions.
- IBM_Data_Science_CS/spacex_dash_app.py at main · Jung028/IBM_Data_Science_CS

Predictive Analysis (Classification)

- Loaded data into a dataframe and created a numpy array using the 'Class' column.
- Normalized the data and split it into training and testing sets.
- Created classifier objects for Logistic Regression, Support Vector Machine, Decision Tree, and K-Nearest Neighbors.
- Used GridSearchCV to find the best parameters for each machine learning algorithm.
- Calculated the accuracy of each algorithm to determine the best one for the project.
- IBM_Data_Science_CS/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · Jung028/IBM_Data_Science_CS

Results

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

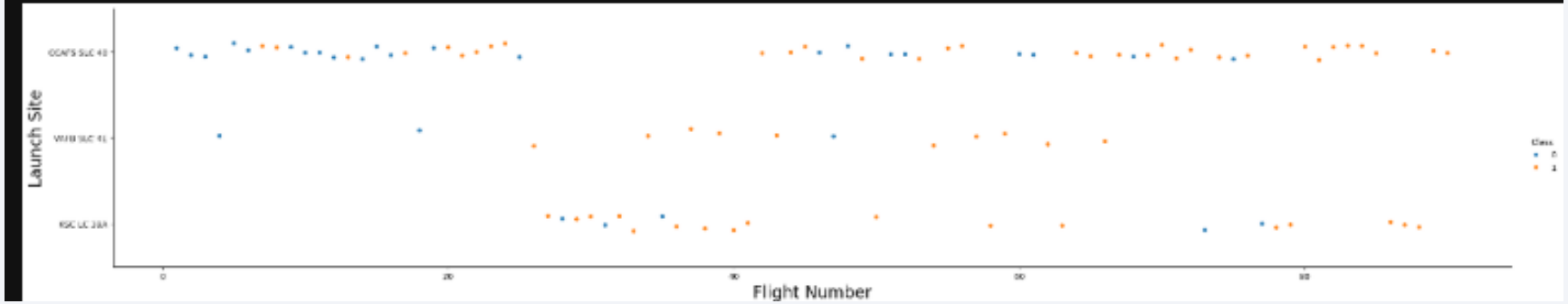
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- This scatter plot indicates that a higher flight number at a launch site correlates with a higher success rate.

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be th
sns.catplot(x='FlightNumber', y='LaunchSite', hue='Class', data=df, aspect=5)
plt.xlabel('Flight Number', fontsize=20)
plt.ylabel('Launch Site', fontsize=20)
plt.show()
```

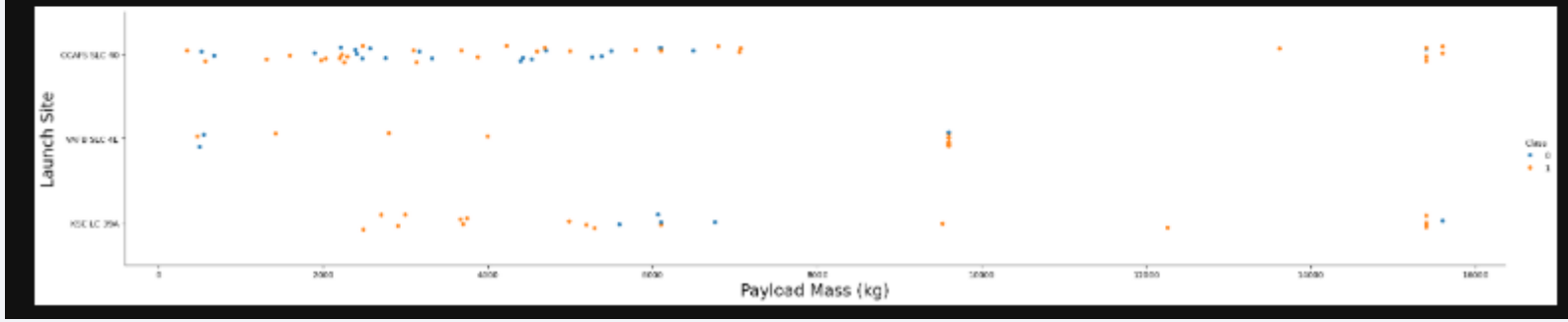


Payload vs. Launch Site

- Payload masses greater than 9000 kg generally result in higher success rates. However, the graph does not show a specific pattern.

In [7]:

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis t
sns.catplot(x='PayloadMass', y='LaunchSite', hue='Class', data=df, aspect = 5)
plt.xlabel('Payload Mass (kg)',fontsize=20)
plt.ylabel('Launch Site',fontsize=20)
plt.show()
```

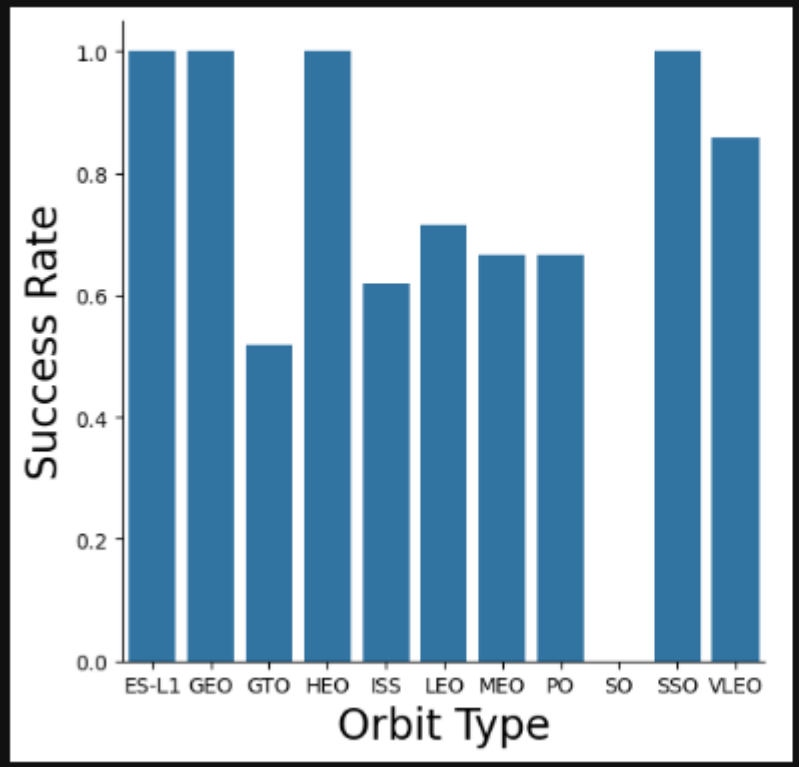


Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO, VLEO orbits has higher success rate than 80% while other orbits are around 60%.
- SO orbit has a success rate of 0

In [8]:

```
# HINT use groupby method on Orbit column and get the mean of Class column
sns.catplot(x= 'Orbit', y = 'Class', data = df.groupby('Orbit')['Class'].mean().reset_index(), kind = 'bar')
plt.xlabel('Orbit Type',fontsize=20)
plt.ylabel('Success Rate',fontsize=20)
plt.show()
```



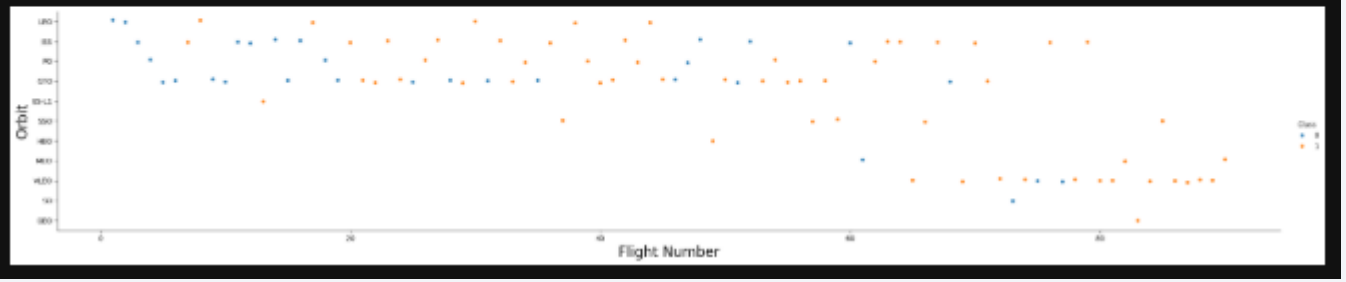
Analyze the plotted bar chart to identify which orbits have the highest success rates.

Flight Number vs. Orbit Type

- The larger the flight number on each orbit, the greater the success rate.
- Especially for the orbit LEO

In [9]:

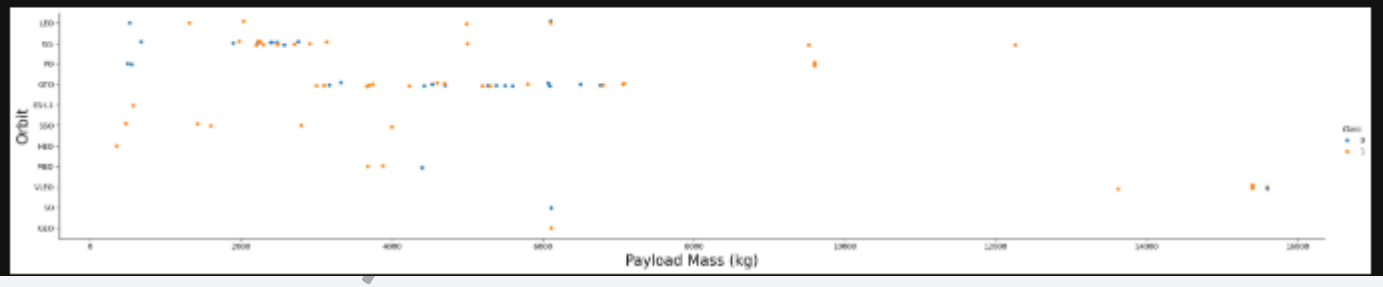
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(x = 'FlightNumber', y = 'Orbit', hue = 'Class', data = df, aspect = 5)
plt.xlabel('Flight Number', fontsize = 20)
plt.ylabel('Orbit', fontsize = 20)
plt.show()
```



Payload vs. Orbit Type

- Higher payload mass positively impacts LEO, ISS, and PO orbits. However, it negatively affects MEO and VLEO orbits, while GTO orbits show no significant relationship.

In [10]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(x = 'PayloadMass', y = 'Orbit', hue = 'Class', data = df, aspect = 5)
plt.xlabel('Payload Mass (kg)', fontsize = 20)
plt.ylabel('Orbit', fontsize = 20)
plt.show()
```



Launch Success Yearly Trend

- Success rate from 2013 is increasing till 2020, but had a drop at 2017.
- Bounced back from 2018 to above 0.8

In [17]:

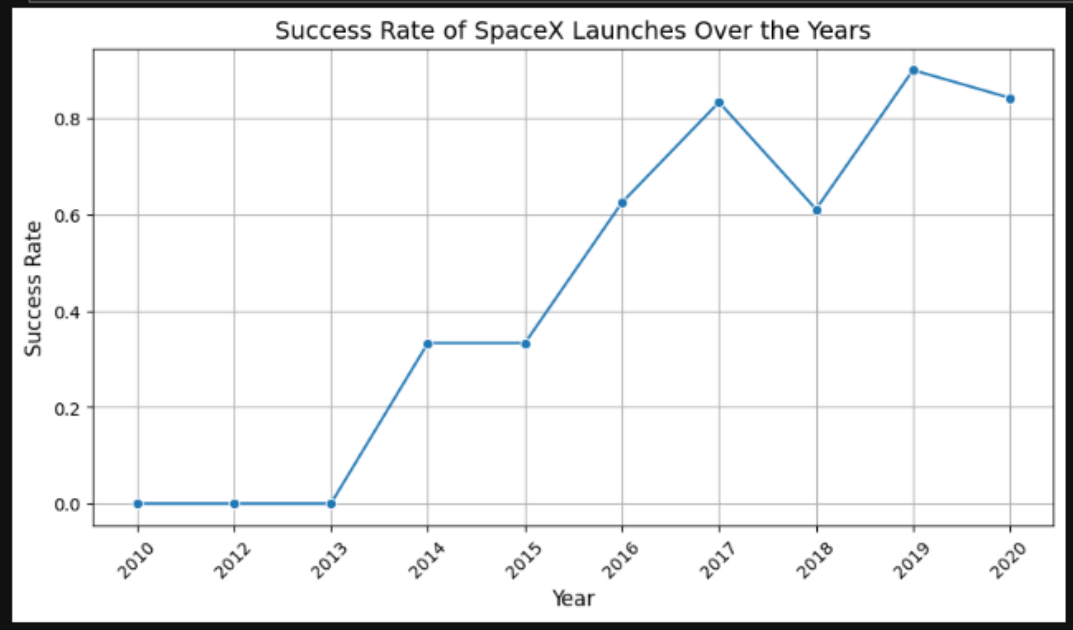
```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
# Extract the year from the Date column
df["Year"] = df["Date"].astype(str).str[:4]

# Calculate success rate per year
success_rate = df.groupby("Year")["Class"].mean()

# Plot the line chart
plt.figure(figsize=(10,5))
sns.lineplot(x=success_rate.index, y=success_rate.values, marker="o", linestyle="-")

# Labels and Title
plt.xlabel("Year", fontsize=12)
plt.ylabel("Success Rate", fontsize=12)
plt.title("Success Rate of SpaceX Launches Over the Years", fontsize=14)
plt.xticks(rotation=45)
plt.grid(True)

# Show the plot
plt.show()
```



you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

- SQL Query syntax DISTINCT query removes all duplicate values of LAUNCH_SITE

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

```
In [67]: %sql select distinct launch_site from SPACEXTBL;
 * sqlite:///my_data1.db
Done.
```

Out[67]:

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

Launch Site Names Begin with 'CCA'

- The WHERE launch_site LIKE 'CCA%' clause filters the records to include only those where the launch_site column values start with the string ‘CCA’.
- The LIMIT 5 clause restricts the output to the first 5 records that match this filter.
- So, this query will display 5 records where the launch sites begin with ‘CCA’.

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

```
In [68]: %sql select * from SPACEXTBL where launch_site like 'CCA%' limit 5;
 * sqlite:///my_data1.db
Done.
```

Out[68]:

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

- Executed an SQL query to calculate the total payload mass for NASA (CRS) missions
- .Used the SUM function to aggregate payload mass values from the SPACEXTBL table.
- Applied a filter to include only records where the customer is 'NASA (CRS)'.
- Query ran successfully on the my_data1.db database.
- The total payload mass transported for NASA (CRS) missions is 45,596 kg


```
In [69]: %sql select sum(payload_mass__kg_) as total_payload_mass from SPACEXTBL where customer = 'NASA (CRS)';
 * sqlite:///my_data1.db
Done.
Out[69]: total_payload_mass
         45596
```

Average Payload Mass by F9 v1.1

- The SQL query calculated the average payload mass for missions using the F9 v1.1 booster.
- Used the AVG function to determine the mean payload mass from the SPACEXTBL table.
- Applied a filter to select only records where the booster version includes 'F9 v1.1'.
- The average payload mass transported per mission with F9 v1.1 is 2,534.67 kg.


```
In [70]: %sql select avg(payload_mass__kg_) as average_payload_mass from SPACEXTBL where booster_version like '%F9 v1.1%';
 * sqlite:///my_data1.db
Done.
Out[70]: average_payload_mass
         2534.6666666666665
```

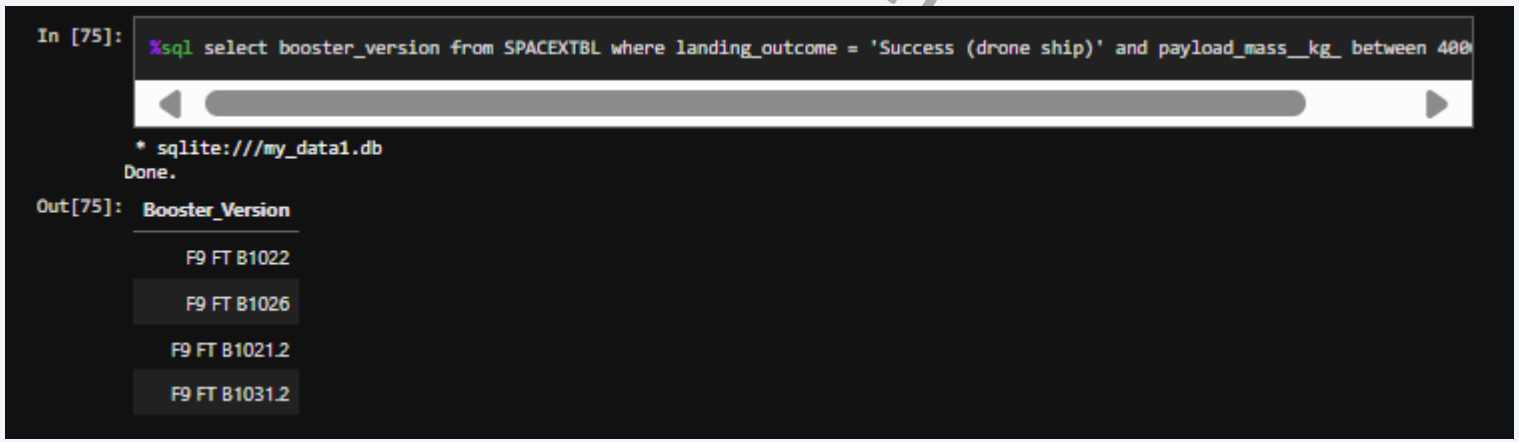
First Successful Ground Landing Date

- Used the MIN function to retrieve the earliest successful landing on a ground pad.
- Filtered the SPACEXTBL table for records where landing_outcome = 'Success (ground pad)'.
- The first successful ground pad landing occurred on December 22, 2015.
- This marked a significant milestone in reusable rocket technology.


```
In [74]: %sql select min(date) as first_successful_landing from SPACEXTBL where landing_outcome = 'Success (ground pad)';
 * sqlite:///my_data1.db
Done.
Out[74]: first_successful_landing
---------------------------------
                      2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- The query filters boosters that landed successfully on a drone ship.
- Includes only missions with a payload mass between 4000 kg and 6000 kg.
- Four Falcon 9 Full Thrust (F9 FT) boosters met the criteria.
- Highlights the reusability and performance of these boosters in mid-range payload missions.



Total Number of Successful and Failure Mission Outcomes

- The COUNT(*) function calculates the total number of missions for each unique mission outcome.
- The GROUP BY mission_outcome clause groups records based on their mission outcome category.
- A total of 100 missions were evaluated.98 missions were labeled as "Success," but there's a duplicate entry with a count of 1, suggesting possible data inconsistency.


```
In [76]: %sql select mission_outcome, count(*) as total_number from SPACEXTBL group by mission_outcome;
 * sqlite:///my_data1.db
Done.
```

Out[76]:

Mission_Outcome	total_number
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- The MAX() function returns the largest value of the selected column.
- This query shows that the names of the booster versions which have carried the maximum payload mass

```
In [79]: %sql select booster_version from SPACEXTBL where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL);
 * sqlite:///my_data1.db
Done.
```

Out[79]:

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

2015 Launch Records

- substr(Date, 6, 2) AS month: Extracts the month from the Date column.substr
- (Date, 1, 4) = '2015': Filters records for the year 2015.
- WHERE landing_outcome = 'Failure (drone ship)': Selects only failed drone ship landings.
- In 2015, there were two failed landings on a drone ship.
- Both failures occurred at CCAFS LC-40 (Cape Canaveral Air Force Station).

In [83]:

```
%%sql
SELECT substr(Date, 6, 2) AS month,
       Date,
       booster_version,
       launch_site,
       landing_outcome
FROM SPACEXTBL
WHERE landing_outcome = 'Failure (drone ship)'
AND substr(Date, 1, 4) = '2015';
```

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

Out[83]:

month	Date	Booster_Version	Launch_Site	Landing_Outcome
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- Most common outcome: "No attempt" (10 instances) – indicating missions without landing attempts.
- Successful landings:
  - 5 on drone ships
  - 3 on ground pads
- Failed landings:
  - 5 failures on drone ships
  - 2 failures due to parachutes
  - 2 uncontrolled ocean landings

```
In [84]: %%sql select landing_outcome, count(*) as count_outcomes from SPACEXTBL
             where date between '2010-06-04' and '2017-03-20'
             group by landing_outcome
             order by count_outcomes desc;
 * sqlite:///my_data1.db
Done.
```

Out[84]:

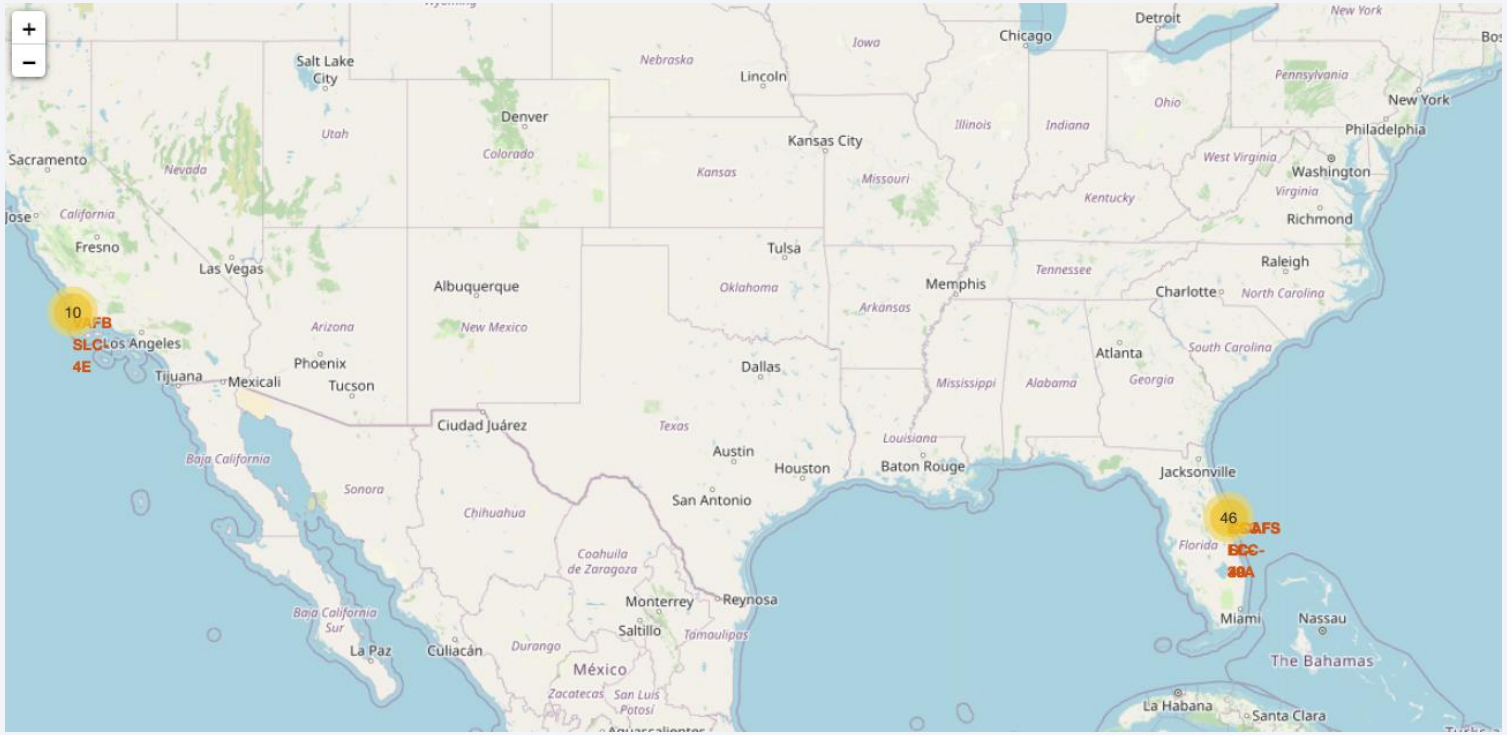
Landing_Outcome	count_outcomes
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

Location of the Launch Sites

- All SpaceX launch sites are based in the United States, specifically in California and Florida.

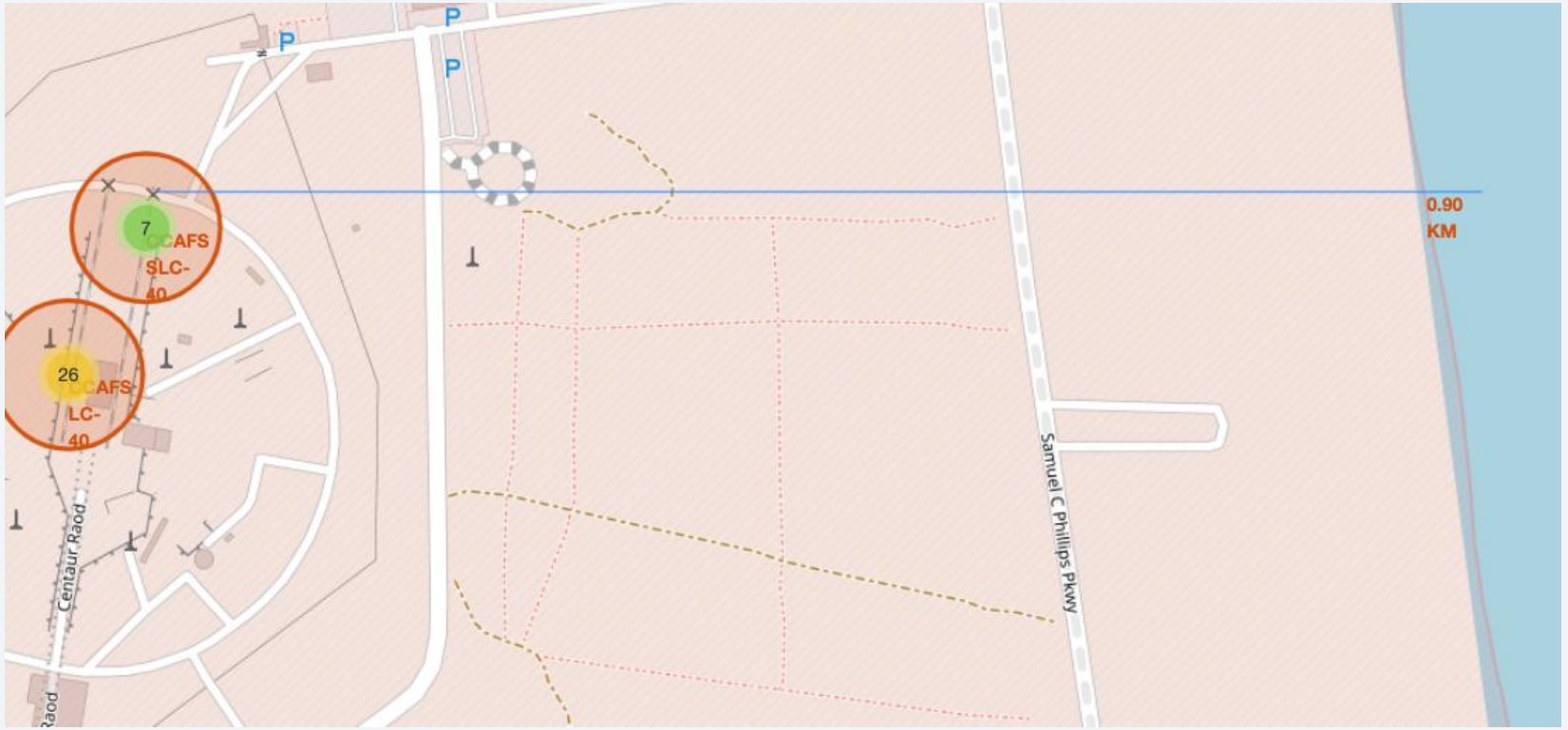


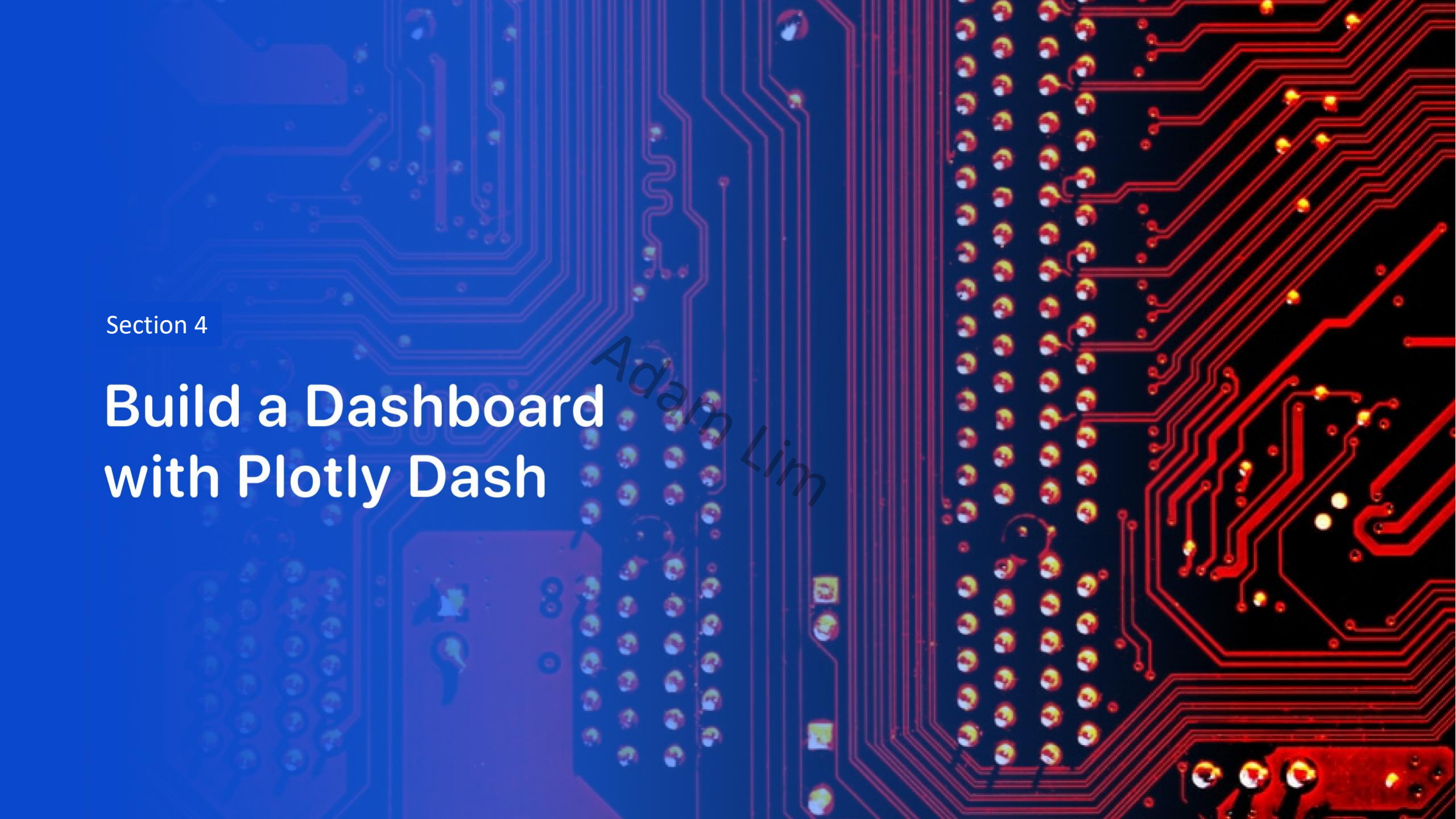
Markers Showing Launch Sites

- Green marker represents successful launch, red marker represents failed launch.
- CCAFS SLC-40 has the highest success rate

Distance of Launch Site to Coastline

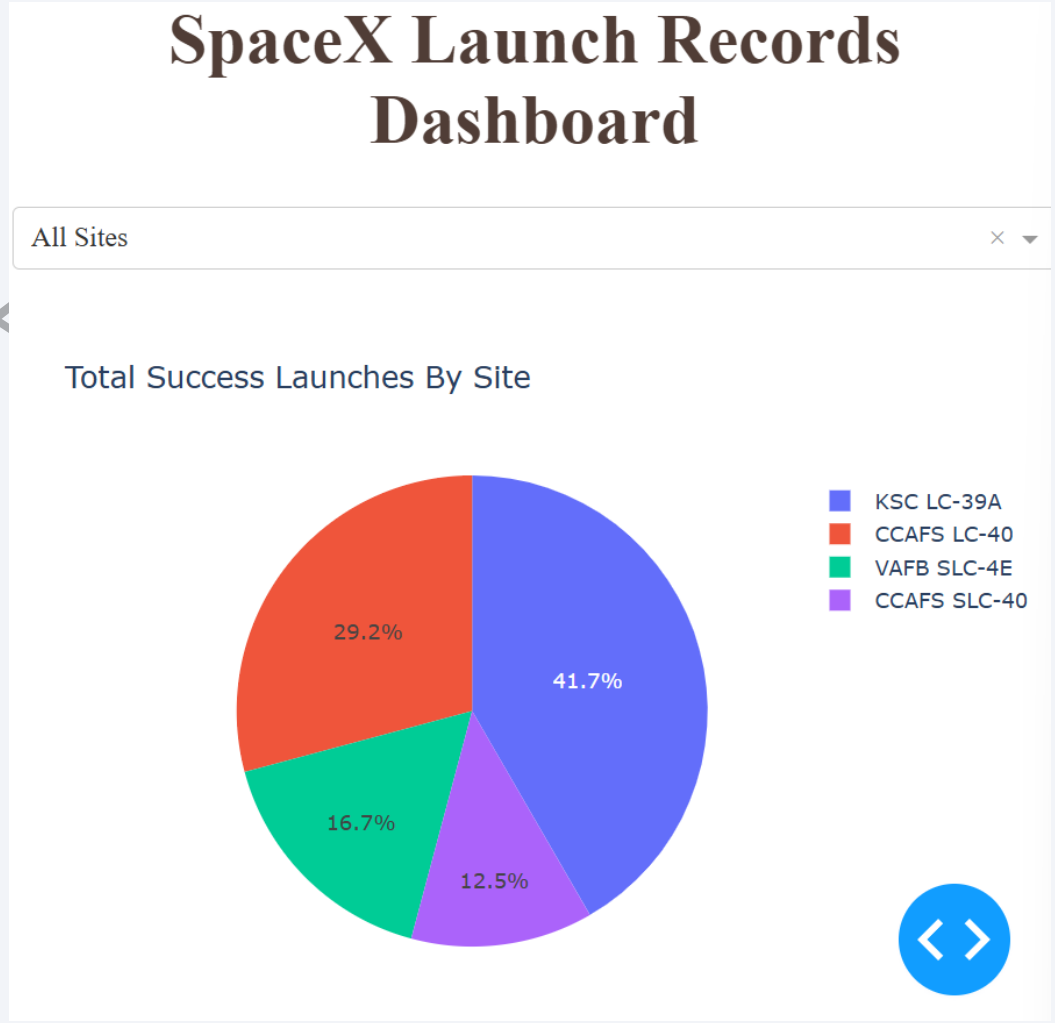
- The nearest is 0.90km from AFS SLC-40 to the Coastline




Section 4
Build a Dashboard with Plotly Dash
Adam Lim

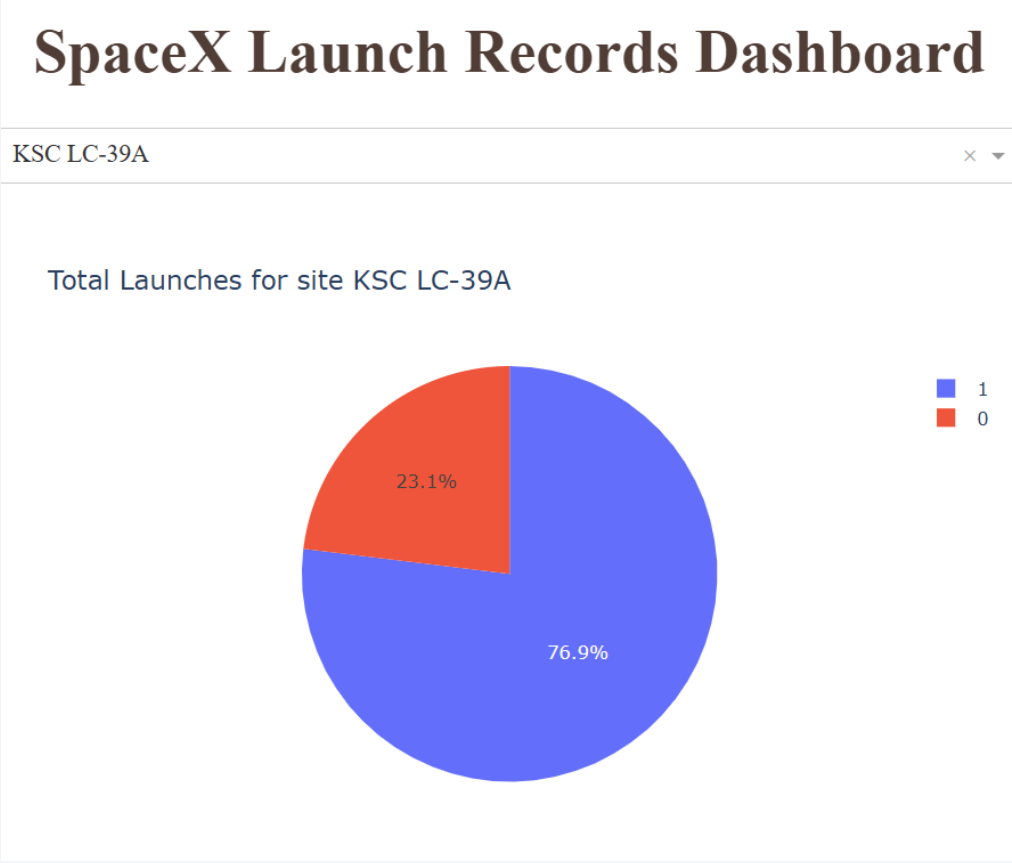
Total Success Launches By Site

- KSC LC-39A has most successful launches
- CCAFS SLC-40 has the least number of successful launches



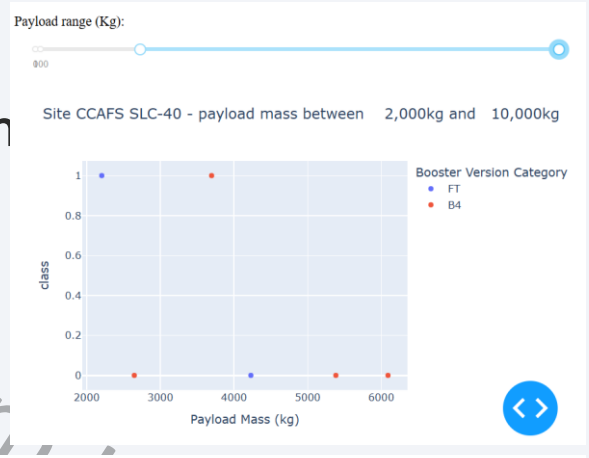
Success Percentage of KSC LC-39A

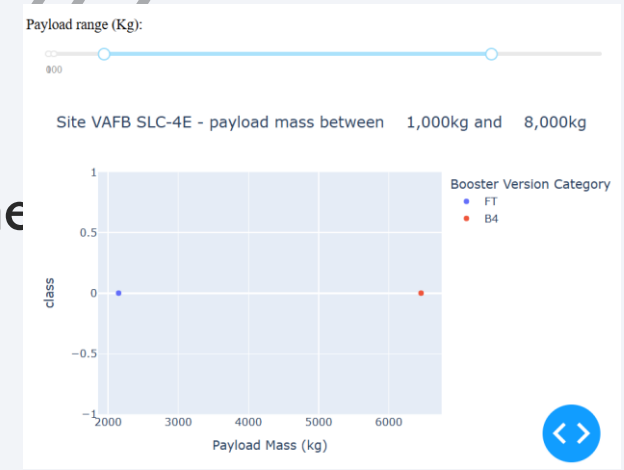
- 76.9% of the total launches were successful, while 23.1% were failed ones

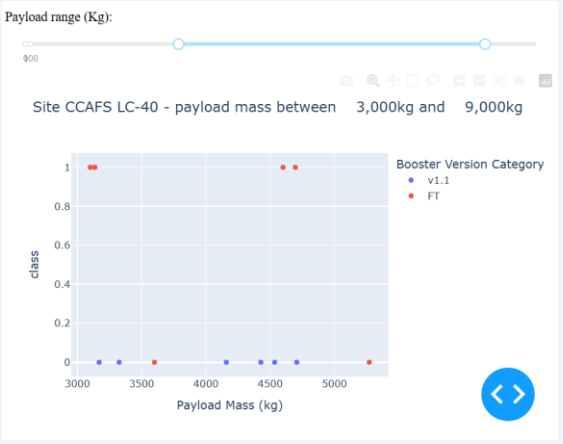


Payload vs. Launch Outcome

- CCAF SCL-40 has more failures for B4 in range 2-10,000kg payload range
- KLSC LC-39A has all success launches for B4 in payload range 3-7KG
- VAFB SLC-4E has success for both Boosters within 1-8,000kg
- CCAFS LC-40 has more success launche for B4 compared to FT in payload 3-9,000kg.





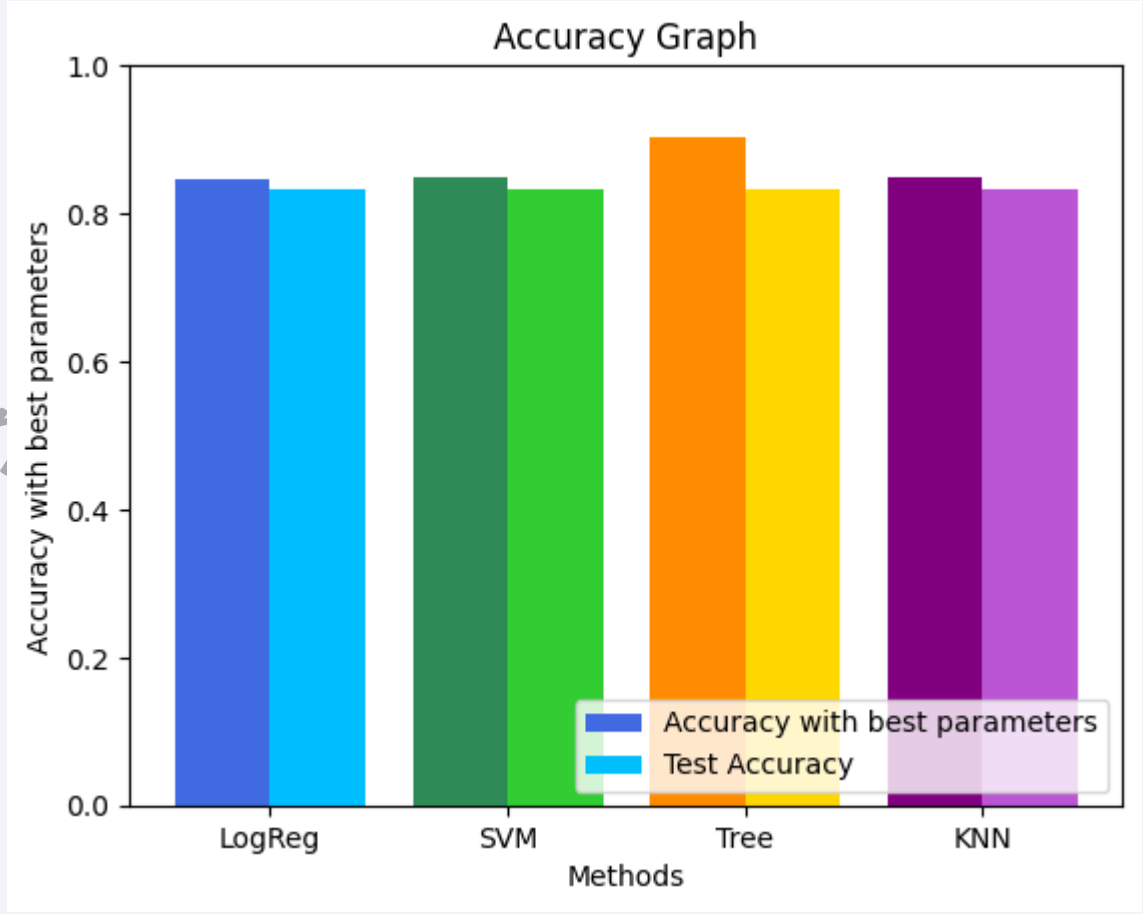


Section 5

Predictive Analysis (Classification)

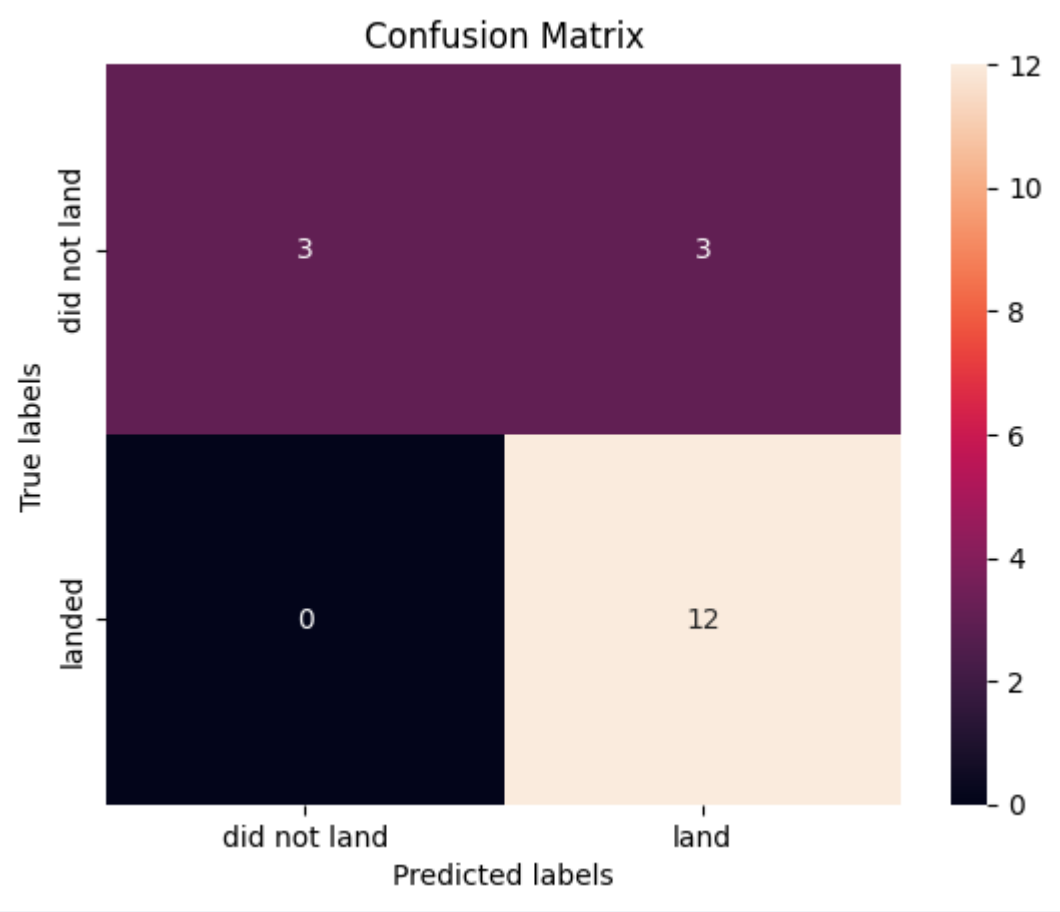
Classification Accuracy

- All methods had the same test accuracy, but the Decision Tree classifier achieved the highest accuracy of 0.9017 with optimal parameters.



Confusion Matrix

- The confusion matrix for the Decision Tree classifier is shown on the right.
- The algorithm predicted ‘land’ when the actual outcome was ‘did not land,’ indicating that false positives were the main issue.



Conclusions

- The Decision Tree classifier demonstrated the highest accuracy for this task.
- KSC LC-39A recorded the highest number of successful launches.
- Launch success rates remained steady from 2010 to 2013, then saw an upward trend from 2013 to 2020.
- Flights from frequently used launch sites tend to achieve higher success rates.
- Lighter payloads generally outperform heavier ones in terms of launch success.

Appendix

- https://api.spacexdata.com/v4/launches/past
- https://api.spacexdata.com/v4/payloads/
- https://api.spacexdata.com/v4/launchpads/

Thank you!