IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Zanabria Adriazola Daniel Antonio>
<9/02/2024>

Outline

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

Executive Summary

- Summary of methodologies
  - ✓ Data Collection through SPACEX API
  - ✓ Data Collection with Web Scraping
  - ✓ Data Wrangling
  - ✓ Exploratory Data Analysis using SQL
  - ✓ Exploratory Data Analysis with Data Visualization
  - ✓ Interactive Visual Analytics with Folium
  - ✓ Machine Learning Prediction
- Summary of all results
  - ✓ Exploratory Data Analysis result
  - ✓ Interactive Dashboards analytics in screenshots
  - ✓ Predictive ML algorithms result

Introduction

- Project background and context

SpaceX promotes Falcon 9 rocket launches on its website, quoting a price of 62 million dollars, significantly lower than other providers whose costs exceed 165 million dollars each. The substantial savings stem from SpaceX's capability to recycle the first stage. Consequently, the ability to predict whether the first stage will successfully land plays a crucial role in determining the overall launch cost. This predictive information becomes valuable when a competing company seeks to bid against SpaceX for a rocket launch. The project's objective is to establish a machine learning pipeline dedicated to forecasting the successful landing of the first stage.

- Problems you want to find answers

✓ What are the most influential or predictive features contributing to the prediction of a successful landing?

✓ How accurately can the model predict whether the rocket will successfully land?

✓ How the features vary in different launch scenarios?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was gathered by utilizing both the SpaceX API and conducting web scraping from Wikipedia
- Perform data wrangling
  - The missing values were identified and replaced them with the mean, The label for ML models was identified, which is the “Class” label. It is set to 1 when the landing was successful and 0 otherwise
- 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

- The information was gathered through diverse approaches. Initially, we utilized a get request to access the SpaceX API for data retrieval. Following this, we decoded the response content into a Json format using the .json() function and transformed it into a Pandas dataframe using .json_normalize(). Subsequently, we underwent a data cleaning process, addressing missing values by filling them in as needed.
- Also, I used web scraping from Wikipedia specifically for Falcon 9 launch records, employing BeautifulSoup. The aim was to extract launch records presented in HTML tables, parse them, and convert the information into a Pandas dataframe for subsequent analysis.

Data Collection – SpaceX API

- We employed a get request to the SpaceX API to gather data, performed data cleaning on the retrieved information, and conducted basic data wrangling.

- The link to see this is: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/Collecting%20the%20data.ipynb

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

Check the content of the response

In [8]:
```
print(response.content)
```

b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":"2006-03-17T00:00:00.000Z","static_fire_date_unix":1142553600,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":3

In [10]:
```
response.status_code
```

Out[10]: 200

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

In [11]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data_json = response.json()
data = pd.json_normalize(data_json)
```

Data Collection - Scraping

- We use BeautifulSoup to webscrap Falcon 9 launch records from Wikipedia.
- We converted the table into a pandas dataframe helping us to some functions to get the data.
- The link to the see this is: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/Data%20Collection%20with%20Web%20Scrapping.ipynb

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

Create a `BeautifulSoup` object from the HTML `response`

```python
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, "html.parser")
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```python
# Use soup.title attribute
title = soup.title
print(title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

In [8]:

```python
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

In [9]:

```python
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Versio
n,<br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#cite_note-booster-11">[b]</a></sup>
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-12">[c]</a></sup>
</th>
```

Data Wrangling

- We determined the training label for the ML models, which is the “Class”. It is set to 1 when the landing was successful and 0 otherwise.
- We calculated the number of each launch site, and the occurence of each orbits.
- We calculated the number and occurrence of mission outcome per orbit type using .value_counts() method on the column “Outcome”
- More detail here: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/Data%20wrangling.ipynb

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df["Outcome"]]
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

```
df['Class']=landing_class
df[['Class']].head(8)
```

```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

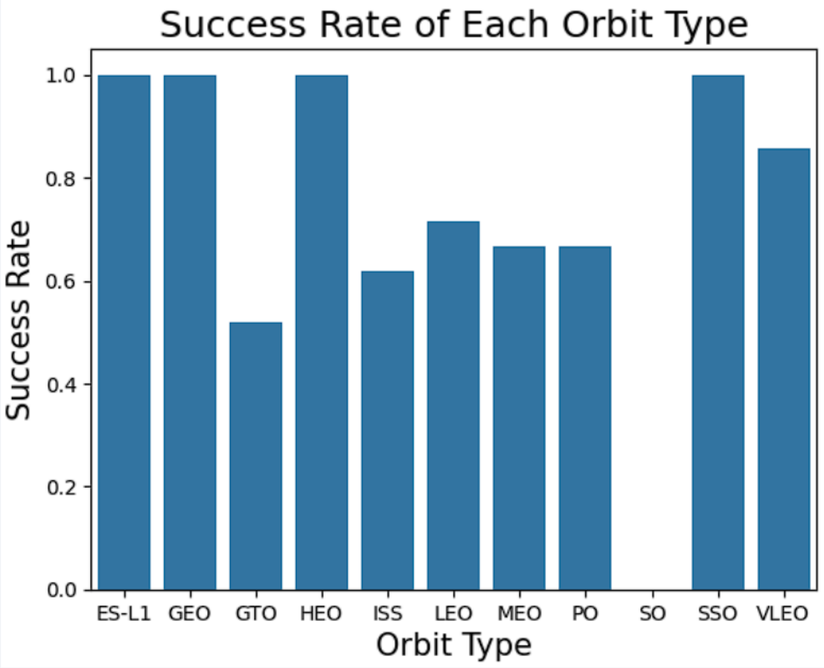
```
# landing_outcomes = values on Outcome column
landing_outcomes = df["Outcome"].value_counts()
landing_outcomes
```

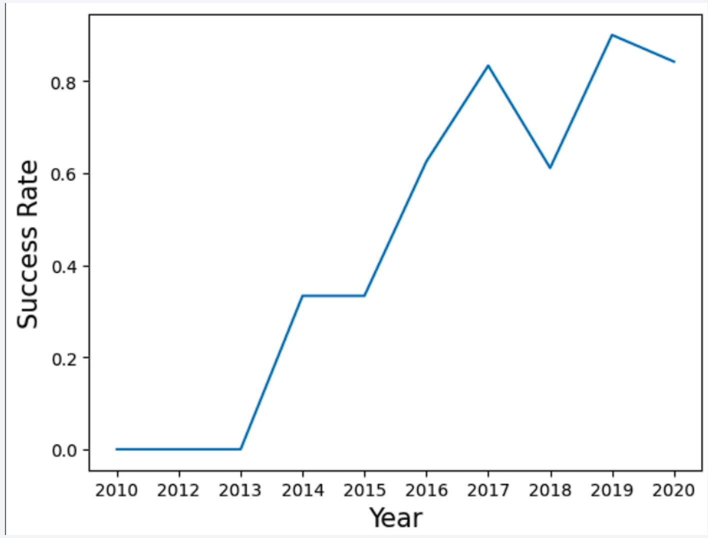
```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

EDA with Data Visualization

- We explored the data using cat plots to see the relationship between some features like flight number and launch site, payload mass and launch site, flight number and orbit type, success rate of each orbit type using a bar plot, and finally the launch success yearly trend, in which we used a line plot.

- You can see the plots here: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/EDA%20with%20Visualization.ipynb





EDA with SQL

First, we connected the database, used pandas to read a CSV file and created a table in the SQLite database.

We did some queries to find out:

- The names of unique launch sites in the space mission.
- Displayed the average payload mass carried by booster version F9 v1.1.
- Listed the names of the boosters with success in the ground pad and payload mass between 4000 and 6000.
- The total number of successful and failure mission outcomes.

All queries can be found here: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- We marked all launch sites on a map using folium.Circle and folium.Marker.
- We marked the success and failed launches for each site on the map, we created a MarkerCluster object first
- Using the color green and red we identified the success and failed launches respectively.
- Finally, we calculated the distances between a launch site to its proximities.
- More details here: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/Launch%20Sites%20Locations%20Analysis%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- We build an interactive dashboard using Plotly dash.
- We created pie charts to illustrate the cumulative launches from specific sites
- We generated a scatter plot to depict the correlation between Outcome and Payload Mass (Kg) across various booster versions.
- You can see this here: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- We utilized numpy and pandas to input the data, modified the data, and divided it into training and testing sets
- We build different machine learning models such as logistic regression, support vector machine, decision tree, and K nearest neighbors classification algorithm.
- We used GridSearchCV which allows us to test parameters of classification algorithms and find the best one.
- We used accuracy as the metric for our model and we found the best performing classification model.
- More details here: https://github.com/DanielZanabria06/Capstone-Project--IBM/blob/main/Machine_Learning_Prediction.ipynb

Results

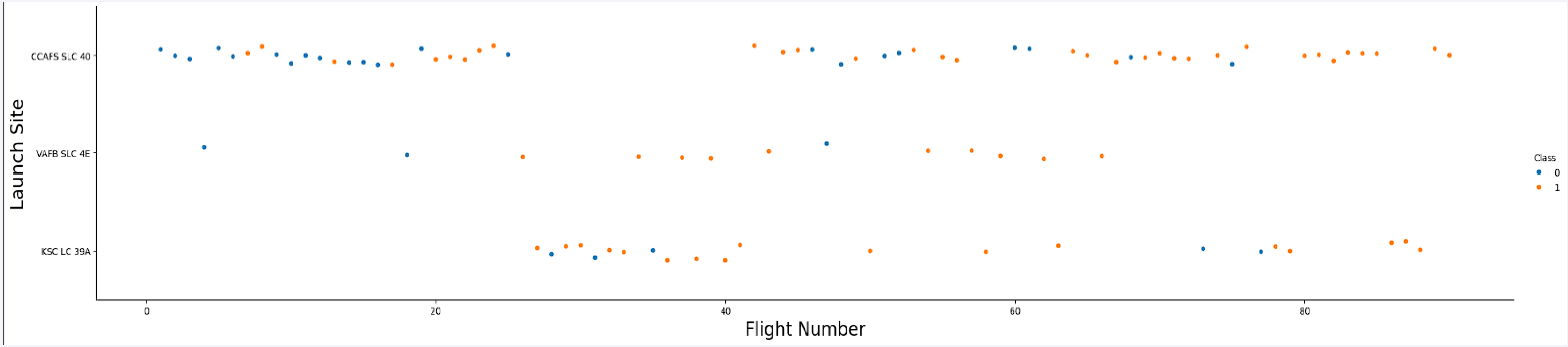
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

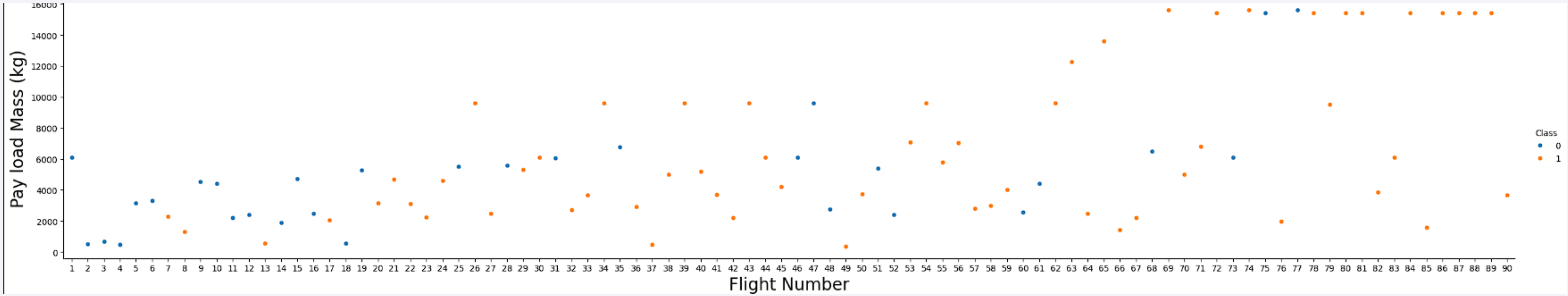
Flight Number vs. Launch Site

From the plot, it was observed that as the number of flights increases at a launch site, there is a corresponding rise in the success rate at that particular launch site. This suggests a positive correlation between the flight amount and the success rate at the launch site. In simpler terms, more flights at a site tend to be associated with a higher likelihood of success for each launch from that site.



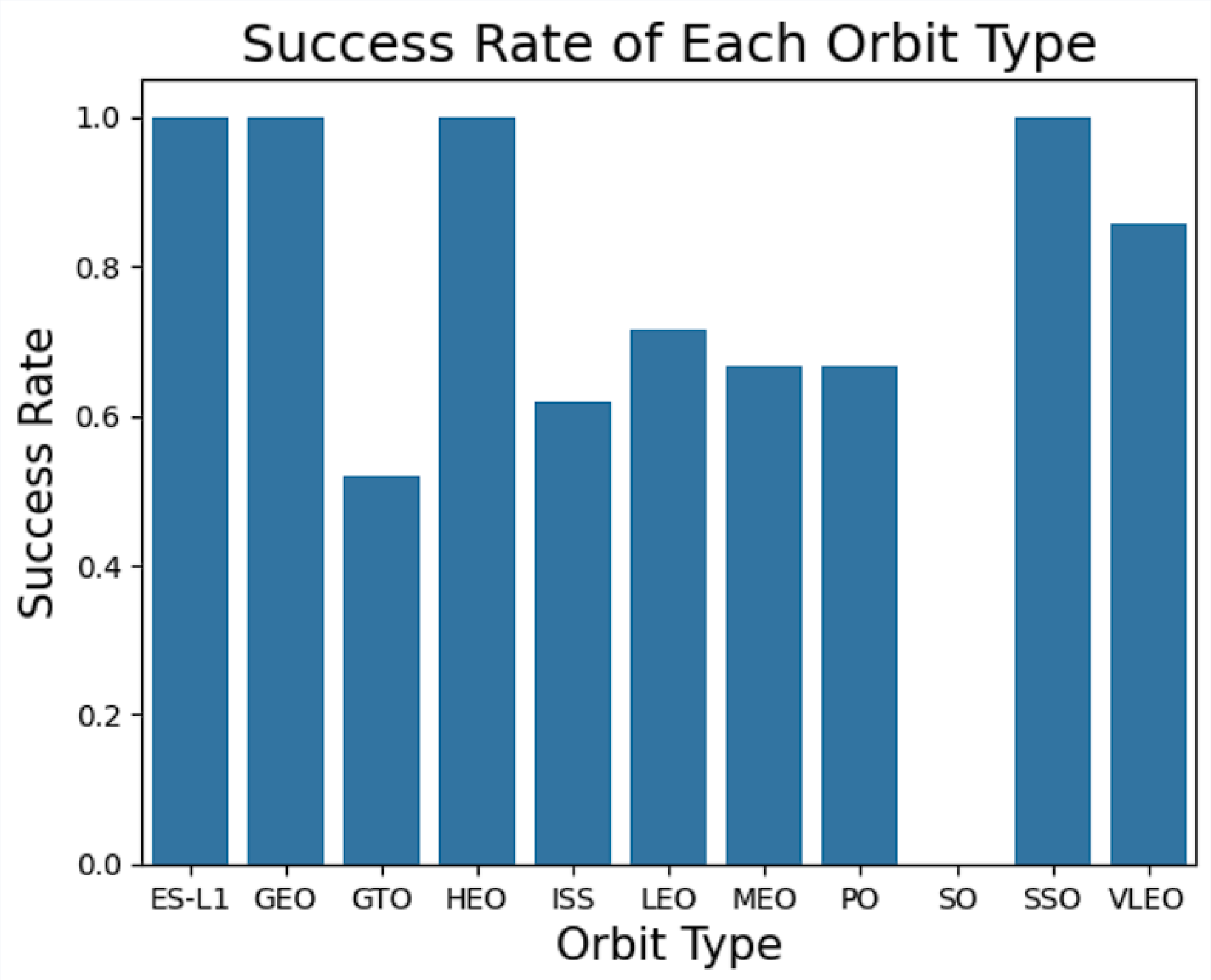
Payload vs. Launch Site

The payload mass is important; it seems the more massive the payload, the less likely the first stage will return. You can see that for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass (greater than 10000).



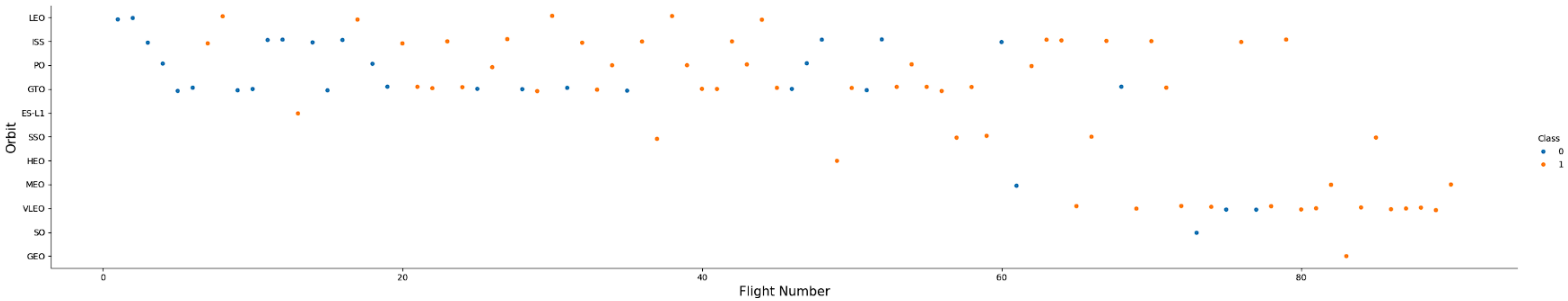
Success Rate vs. Orbit Type

From the plot, we can see that ES-L1, GEO, HEO, SSO had the most success rate.



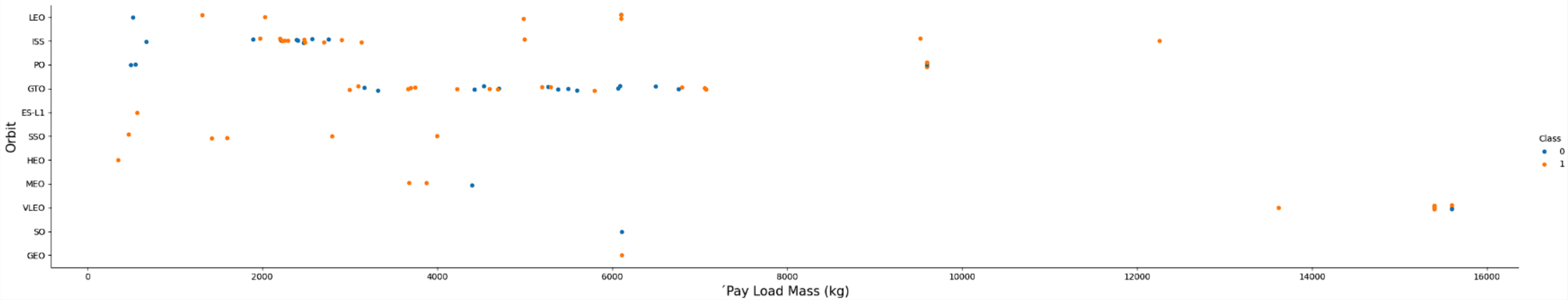
Flight Number vs. Orbit Type

In the Low Earth Orbit (LEO), there seems to be a connection between the success rate and the number of flights, indicating that as the number of flights increases, the success rate tends to follow suit. However, in the Geostationary Transfer Orbit (GTO), there appears to be no discernible correlation between the number of flights and the success rate.



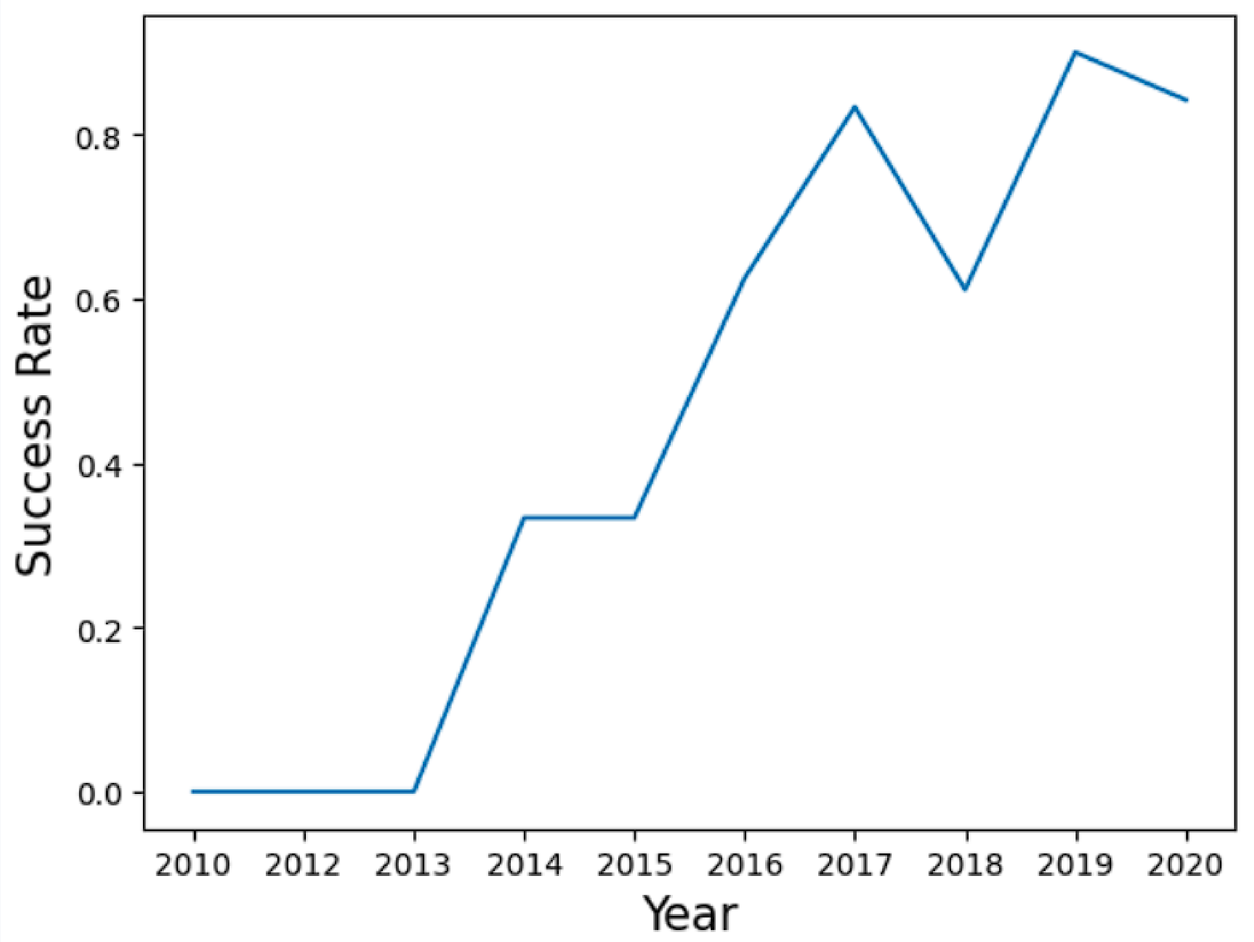
Payload vs. Orbit Type

For substantial payloads, there is a higher rate of successful or positive landings observed in Polar, LEO, and ISS missions. However, in the case of GTO, distinguishing between positive landing rates and negative landings (unsuccessful missions) is challenging since both outcomes are present in this category.



Launch Success Yearly Trend

We can notice from this plot that the success rate has been consistently rising from 2013 to 2020.



All Launch Site Names

This SQL query retrieves and displays the unique launch site names from the "Launch_Site" column in the "SPACEXTBL" table.

We used the key word DISTINCT to show only unique launch sites.

In [8]:
```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTBL;
```

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

Out[8]:

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

Launch Site Names Begin with 'KSC'

We used this query lo display 5 records where launch sites begin with ‘KSC’ for this we used LIMIT 5 and LIKE “KSC%”

In [9]:
```
%sql SELECT * FROM SPACEXTBL WHERE "Launch_Site" LIKE "KSC%" LIMIT 5;
```

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

Out[9]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-03-16	6:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
2017-05-01	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
2017-05-15	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

Total Payload Mass

We calculated the total payload mass using SUM("PAYLOAD_MASS__KG_") and the result was 45596.

```
In [10]:  %sql SELECT SUM("PAYLOAD_MASS__KG_") AS Total_Payload_Mass FROM SPACEXTBL WHERE "Customer" = "NASA (CRS)";

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

Out[10]:

Total_Payload_Mass
45596

Average Payload Mass by F9 v1.1

We calculated the average payload mass carried by booster version F9 v1.1.

In [14]:
```
%sql SELECT AVG("PAYLOAD_MASS__KG_") AS Average_Payload_Mass FROM SPACEXTBL WHERE "Booster_Version" = "F9 v1.1";
```

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

Out[14]:

Average_Payload_Mass
2928.4

First Successful Ground Landing Date

We use MIN("Date") to display the first date where the successful landing outcome in drone ship was achieved.

```
In [18]: %sql SELECT MIN("Date") FROM SPACEXTBL WHERE "Landing_Outcome" = "Success (drone ship)";

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

Out[18]:

MIN("Date")
2016-04-08

Successful Drone Ship Landing with Payload between 4000 and 6000

We used WHERE clause to filter boosters which have land successfully on drone ship and we used AND to filter the booster's payload mass between 4000 and 6000.

```
[12]: 3ooster_Version" FROM SPACEXTBL WHERE "Landing_Outcome" = "Success (drone ship)" AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000;
```

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

[12]:

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

Total Number of Successful and Failure Mission Outcomes

This SQL query retrieves and summarizes the count of each unique "Mission_Outcome" from the "SPACEXTBL" table. The result includes different outcomes such as "Failure (in flight)," "Success," and "Success (payload status unclear).

```
In [24]: %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") AS TOTAL_NUMBER FROM SPACEXTBL GROUP BY MISSION_OUTCOME;

 * sqlite:///my_data1.db
Done.
Out[24]:
```

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

Boosters Carried Maximum Payload

In [25]:
```
%sql SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);
```

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

Out[25]:

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

This SQL query selects the "Booster_Version" from the "SPACEXTBL" table where the "PAYLOAD_MASS__KG_" is equal to the maximum payload mass in the entire dataset. The query aims to identify the booster version associated with the mission having the highest payload mass.

2017 Launch Records

This query extracts specific information from the "SPACEXTBL" table. It selects the month (using the "Date" column), "Landing_Outcome," "Booster_Version," and "Launch_Site" for entries where the year is 2017 and the landing outcome is 'Success (ground pad).' The result provides details about successful landings on ground pads in 2017, including the specific month, landing outcome, booster version, and launch site.

The query is:

%sql SELECT substr("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTBL WHERE substr("Date", 0, 5) = '2017' AND "Landing_Outcome" = 'Success (ground pad)';

In [26]:

```
%sql SELECT substr("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTBL WHERE substr('
```

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

Out[26]:

Month	Landing_Outcome	Booster_Version	Launch_Site
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

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

We selected Landing outcomes and the COUNT of landing outcomes, we used WHERE to filter the dates for landing outcomes.

We used GROUP BY to group by Landing Outcome and ORDER BY to order the clause in DESC order.

[16]:
```
%%sql
SELECT "Landing_Outcome", COUNT(*) AS Count_of_Outcomes
FROM SPACEXTBL
WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome"
ORDER BY Count_of_Outcomes DESC;
```

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

[16]:

Landing_Outcome	Count_of_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

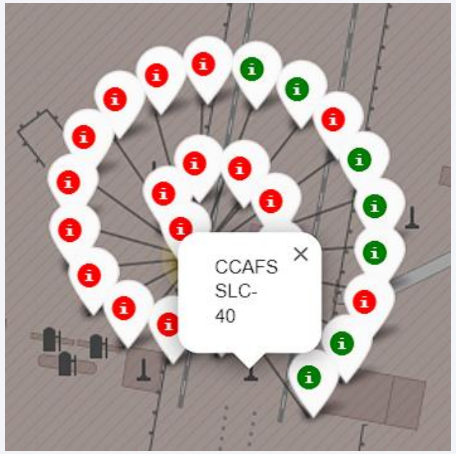
Section 3

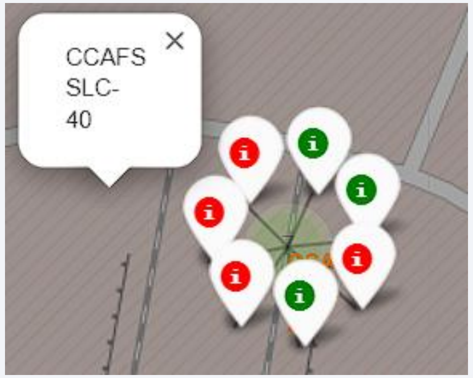
Launch Sites Proximities Analysis

Launch Sites global map markers

We can see that launch sites are in proximity to the equator line and to the coast.

Markers showing successful and failure launches

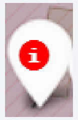




Florida Launch Sites

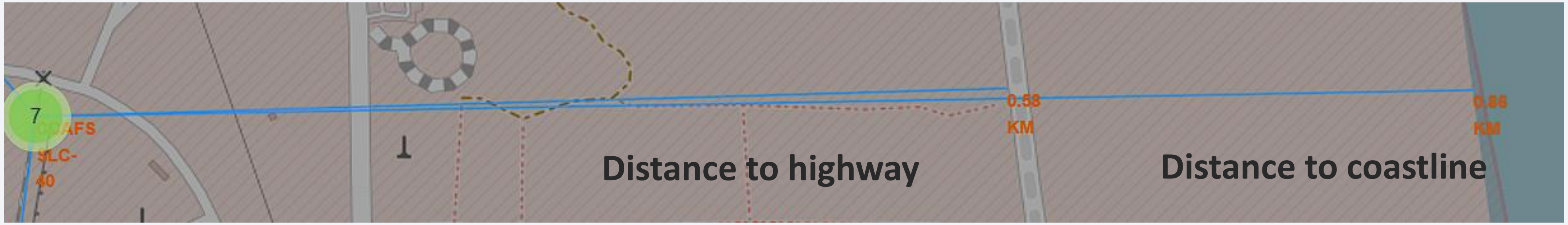
California
Launch Site

Successful launches

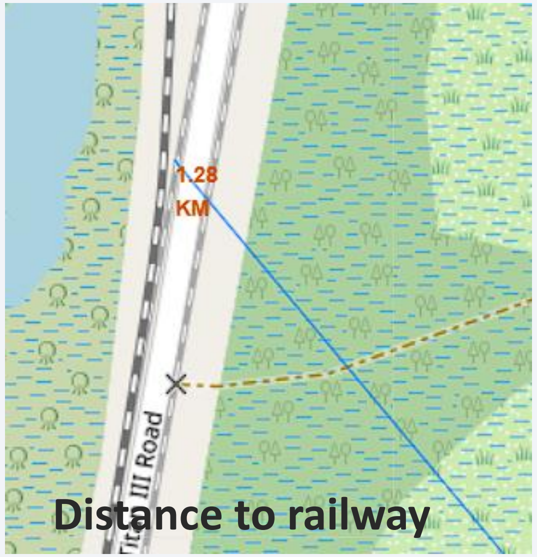


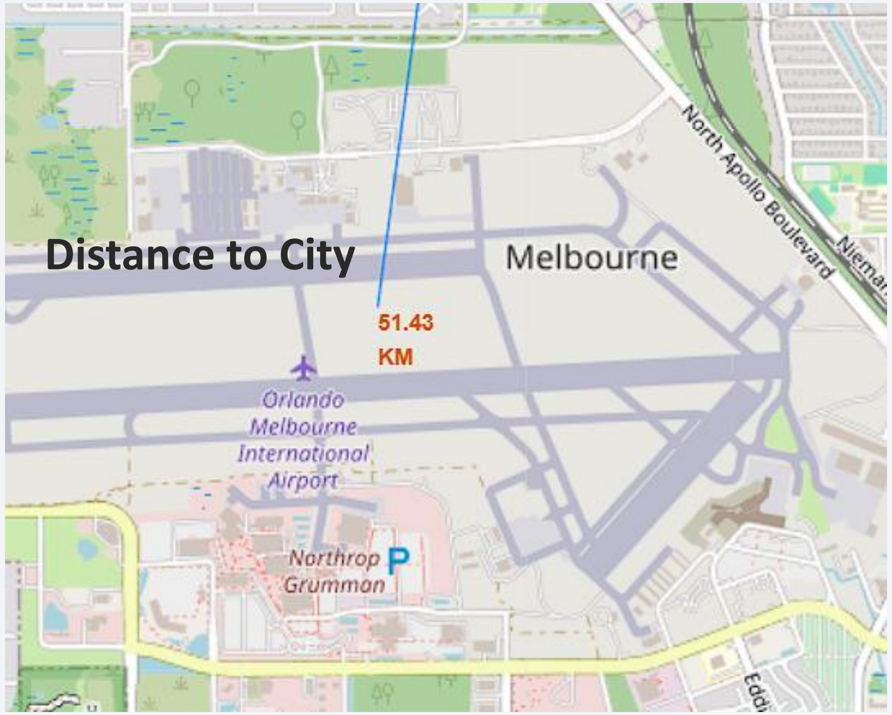
Failures launches

Distance of launch sites proximities



- Launch sites in close proximity to railways? Yes
- Launch sites in close proximity to highways? Yes
- Launch sites in close proximity to coastline? Yes
- Launch sites in close proximity to cities? No

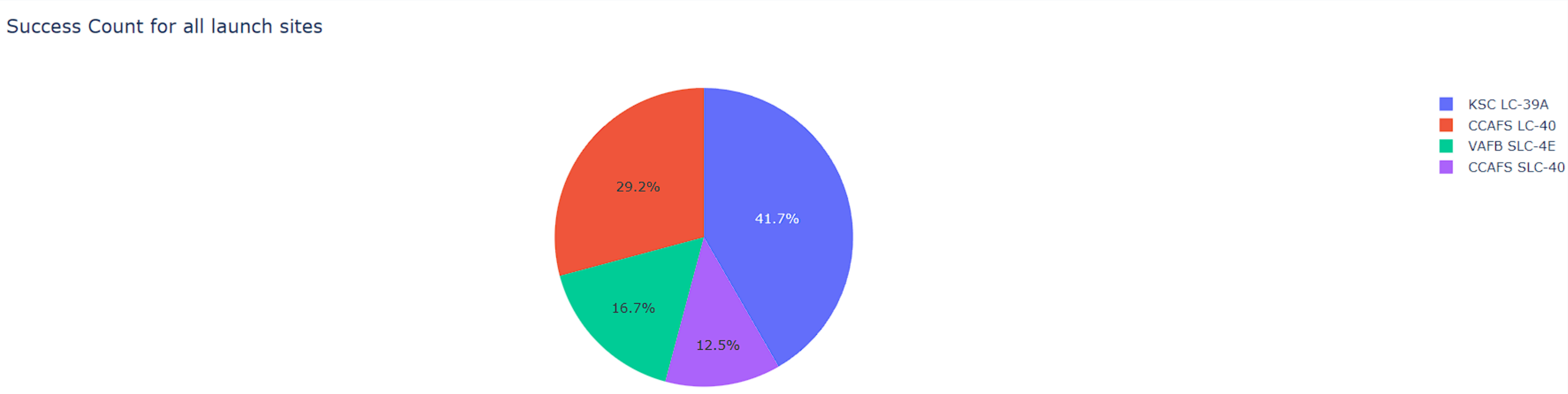




Section 4

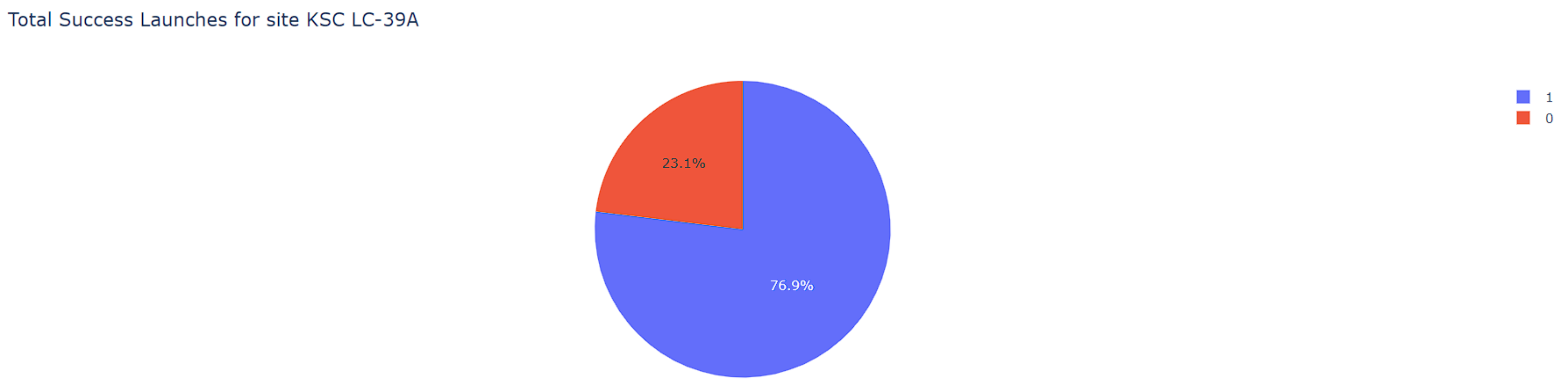
Build a Dashboard with Plotly Dash

Pie chart about percentage success by each launch site



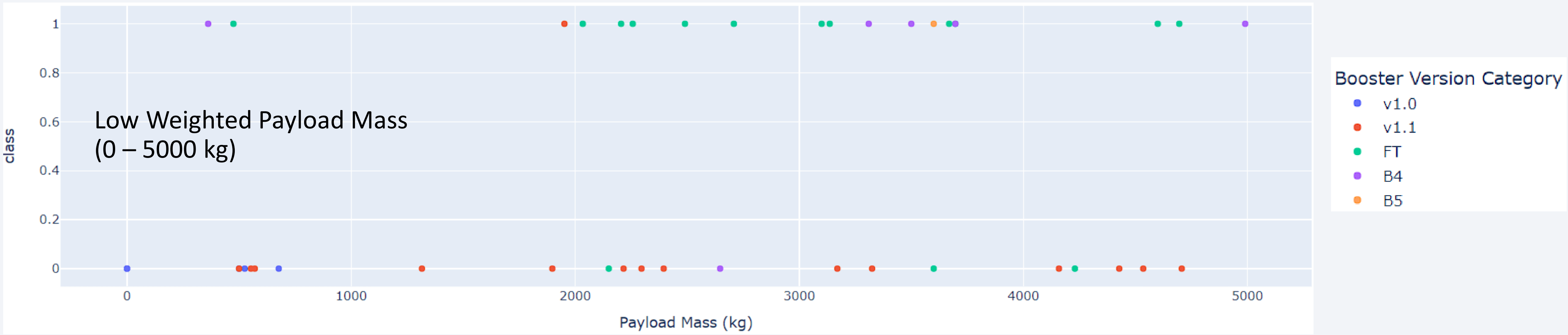
We can see that KSC LC-39A had the most successful launches from all the sites

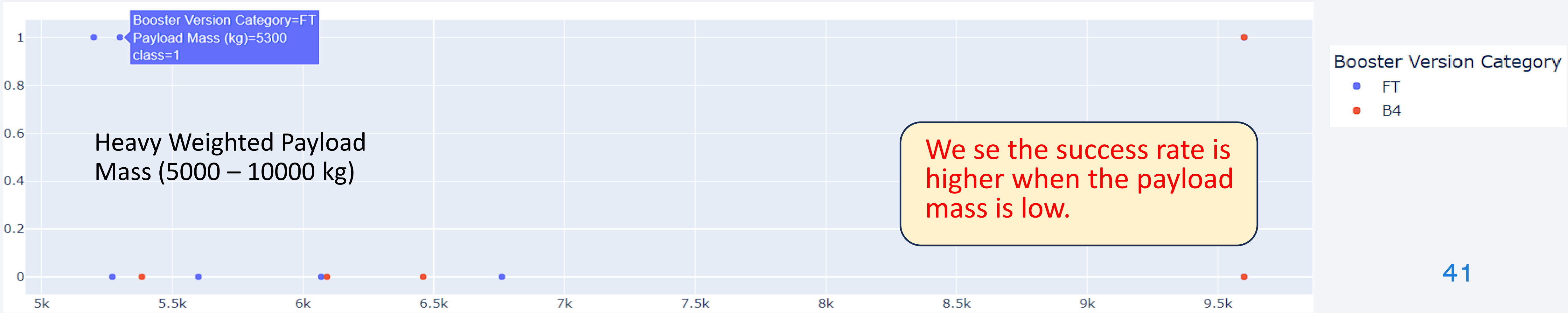
Pie chart for the launch site with highest launch success ratio



The launch site KSC LC-39A attained a success rate of 76.9%, accompanied by a failure rate of 23.1%.

Payload vs Launch Outcome for all sites



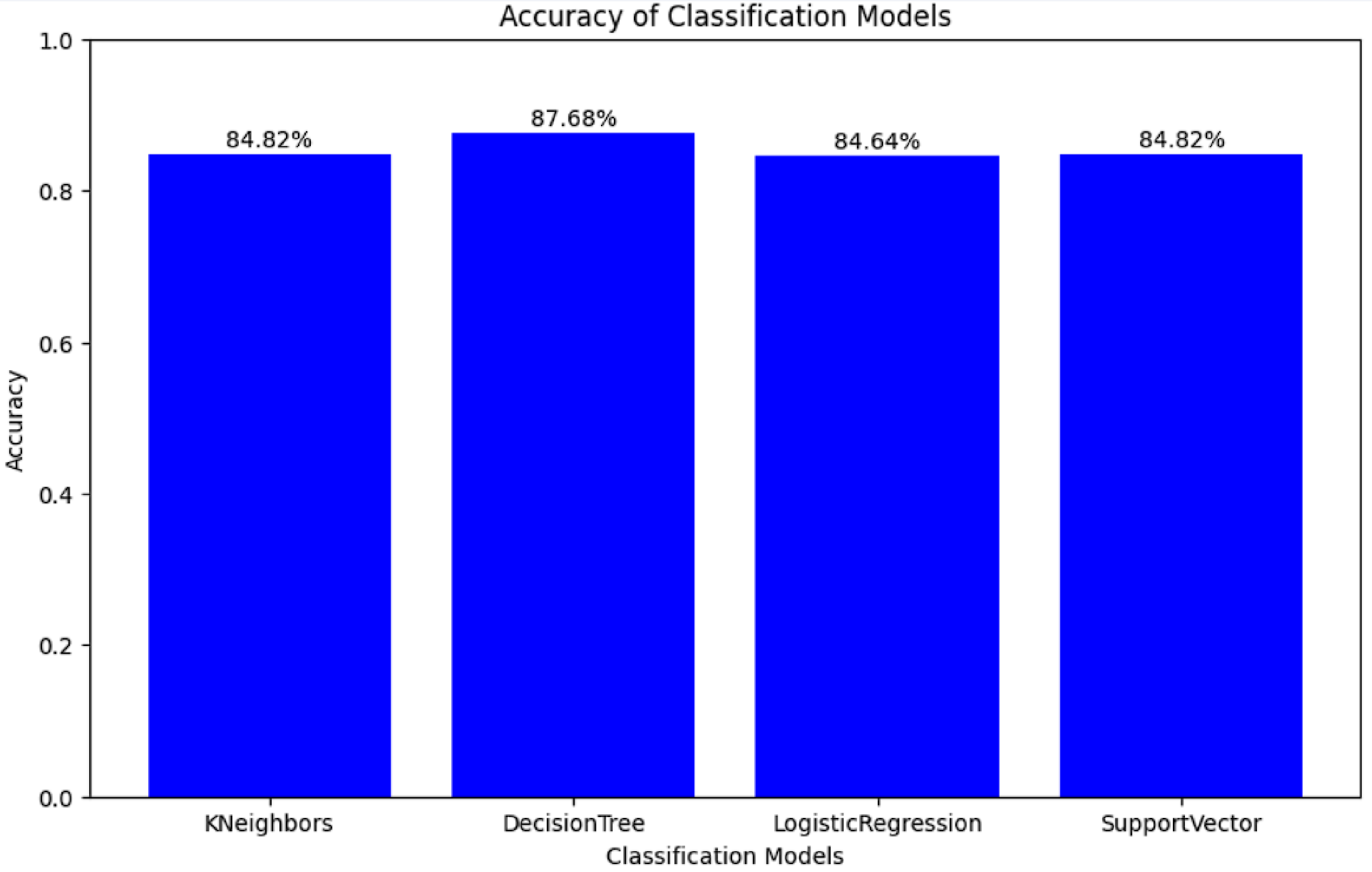


We se the success rate is higher when the payload mass is low.

Section 5

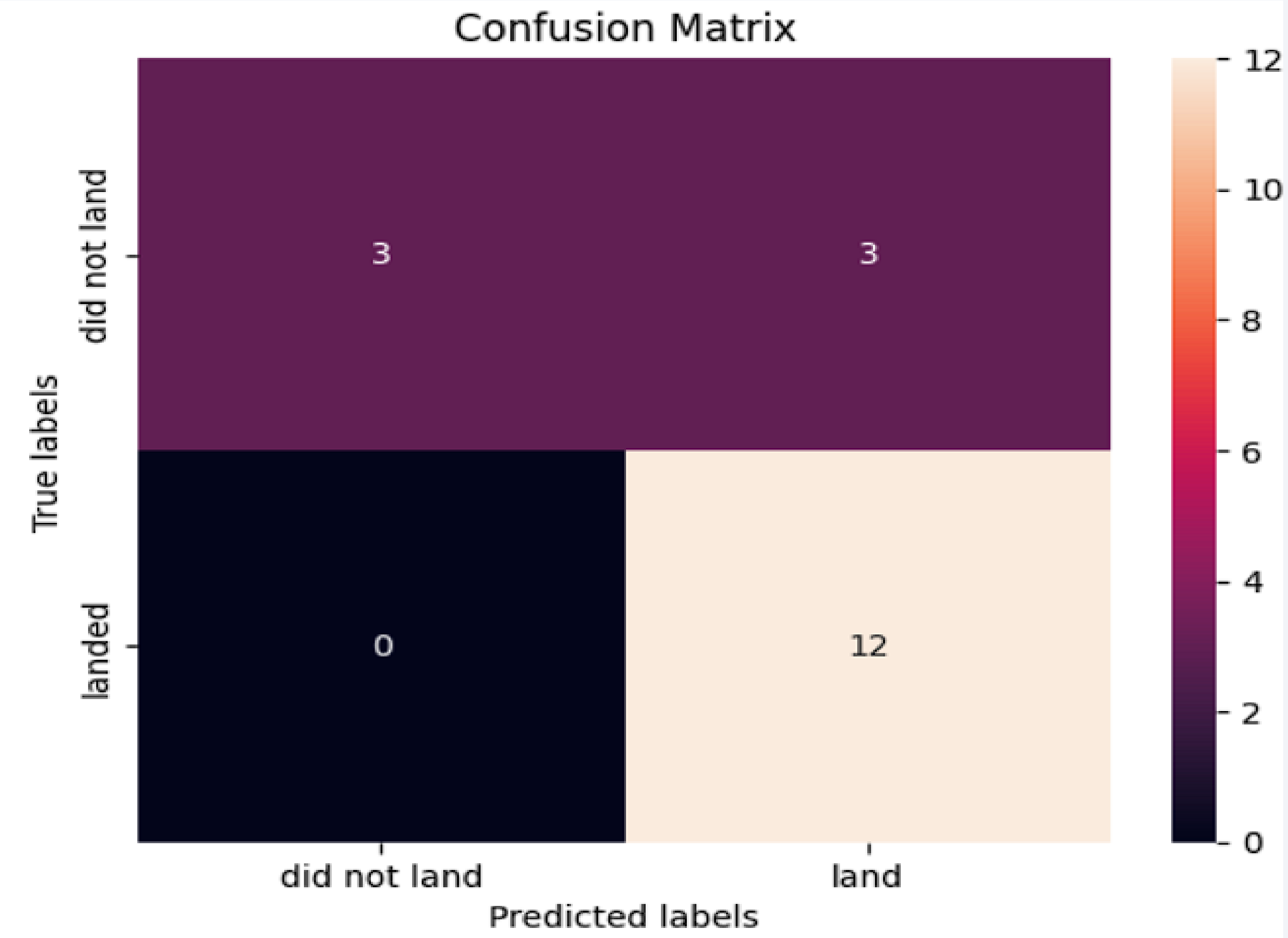
Predictive Analysis (Classification)

Classification Accuracy



The decision tree classifier is the model with the highest classification accuracy (87.68%)

Confusion Matrix



The confusion matrix of the decision tree classifier indicates its ability to differentiate between various classes. However, a notable issue lies in the occurrence of false positives, where the classifier wrongly identifies unsuccessful landings as successful ones.

Conclusions

We can conclude:

- A higher number of flights at a launch site correlates with an increased success rate at that specific site.
- Among the machine learning algorithms assessed, the Decision Tree classifier proved to be the most effective for this particular task.
- The analysis highlights the influence of payload mass, launch site location, and the target orbit on achieving a higher success rate. This underscores the importance of carefully considering these factors in mission planning to enhance the likelihood of mission success.
- The meticulous data wrangling process, encompassing web scraping from SpaceX API and handling missing data through mean imputation, played a pivotal role. This approach ensured the acquisition of a clean and comprehensive dataset, laying a strong foundation for the successful development of machine learning models.

Appendix

Code to obtain the algorithm with the highest accuracy and code to generate the bar chart displaying the accuracy of each developed ML model.

```
model_scores = {'KNeighbors': knn_cv.best_score_,
                'DecisionTree': tree_cv.best_score_,
                'LogisticRegression': logreg_cv.best_score_,
                'SupportVector': svm_cv.best_score_}

best_model = max(model_scores, key=model_scores.get)

print(f'The best-performing model is {best_model} with a score of {model_scores[best_model]}.')
```

```
model_names = list(model_scores.keys())
model_accuracies = list(model_scores.values())

plt.figure(figsize=(10, 6))
plt.bar(model_names, model_accuracies, color='blue')

plt.title('Accuracy of Classification Models')
plt.xlabel('Classification Models')
plt.ylabel('Accuracy')
plt.ylim(0, 1)

for i, accuracy in enumerate(model_accuracies):
    plt.text(i, accuracy + 0.01, f'{accuracy:.2%}', ha='center')

plt.show()
```

Thank you!