

A photograph of a SpaceX Falcon 9 rocket standing vertically on its launch pad. The rocket is white with black markings and is supported by its three landing legs. The background is a clear blue sky with some light clouds. The text "SPACE X FALCON 9 LAUNCH - DATA ANALYSIS" is overlaid in large, bold, red letters with a white outline.

SPACE X FALCON 9 LAUNCH - DATA ANALYSIS

Sneha MV
2-Sep-2024

OUTLINE

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

EXECUTIVE SUMMARY

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

INTRODUCTION

- Project background and context

Space X 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 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

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

METHODOLOGY

Executive Summary

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

DATA COLLECTION

- The data was collected using various methods
 - Data collection was done using get request to the SpaceX API.
 - Next, decoded the response content as a Json using `.json()` function call and turn it into a pandas dataframe using `.json_normalize()`.
 - We then cleaned the data, checked for missing values and fill in missing values where necessary.
 - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
 - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

DATA COLLECTION – SPACE X API

We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

The link to the notebook is [Spacex-falcon-9-launching-project/jupyter-labs-spacex-data-collection-api.ipynb](https://github.com/sneham0312/Spacex-falcon-9-launching-project/blob/main/jupyter-labs-spacex-data-collection-api.ipynb) at main · sneham0312/Spacex-falcon-9-launching-project (github.com)

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

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

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

```
Check the content of the response
```

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

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

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

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
[ ]: static_json_url="https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json"
```

We should see that the request was successful with the 200 status response code

```
[ ]: response.status_code
```

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

```
[ ]: # Use json_normalize method to convert the json result into a dataframe
```

Using the dataframe `data` print the first 5 rows

```
[ ]: # Get the head of the dataframe
```

DATA COLLECTION - SCRAPING

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is [Spacex-falcon-9-launching-project/jupyter-labs-webscraping.ipynb](https://github.com/sneham0312/Spacex-falcon-9-launching-project) at main · sneham0312/Spacex-falcon-9-launching-project (github.com)

To keep the lab tasks consistent, you will be asked to scrape the data from a snapshot of the [List of Falcon 9 and Falcon Heavy launches](#) updated on [9th June 2021](#)

```
[4]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

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

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
[8]: # use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url).text
```

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

```
[9]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, 'html.parser')
```

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

```
[10]: # Use soup.title attribute
print(soup.title)
```

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

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

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external referenc

DATA WRANGLING

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.

93 90 11-05 Falcon 9 3681.0 MeO 40 ASD5 1 True False True 5e9e3032383ecbbb234e/ca 5.0 8 810b2 -80

90 rows × 17 columns

Data Wrangling

We can see below that some of the rows are missing values in our dataset.

```
[33]: data_falcon9.isnull().sum()
```

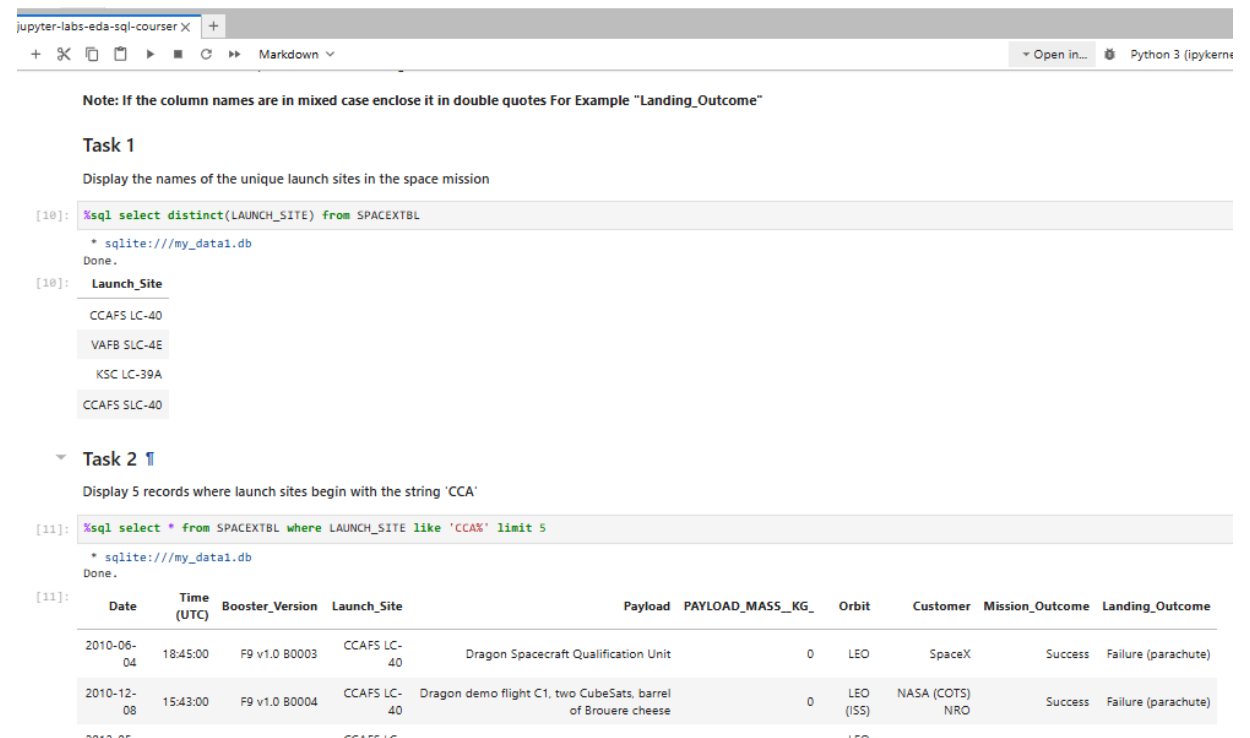
```
[33]: FlightNumber    0
      Date          0
      BoosterVersion 0
      PayloadMass    5
      Orbit         0
      LaunchSite    0
      Outcome       0
      Flights       0
      GridFins      0
      Reused        0
      Legs          0
      LandingPad    26
      Block        0
      ReusedCount   0
      Serial       0
      Longitude     0
      Latitude      0
      dtype: int64
```

Before we can continue we must deal with these missing values. The `LandingPad` column will retain None values to represent when landing pads were not used.

Task 3: Dealing with Missing Values

EDA WITH SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
 - The names of unique launch sites in the space mission.
 - The total payload mass carried by boosters launched by NASA (CRS)
 - The average payload mass carried by booster version F9 v1.1
 - The total number of successful and failure mission outcomes
 - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook [Spacex-falcon-9-launching-project/jupyter-labs-eda-sql-coursera_sqlite.ipynb](https://github.com/sneham0312/Spacex-falcon-9-launching-project) at main · sneham0312/Spacex-falcon-9-launching-project (github.com)



The screenshot shows a Jupyter Notebook titled 'jupyter-labs-eda-sql-coursera X'. It contains two tasks:

Task 1
Display the names of the unique launch sites in the space mission

```
[10]: %sql select distinct(LAUNCH_SITE) from SPACEXTBL
* sqlite:///my_data1.db
Done.
```

The output for Task 1 is a table with one column, 'Launch_Site', containing four unique launch sites: CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, and CCAFS SLC-40.

Task 2
Display 5 records where launch sites begin with the string 'CCA'

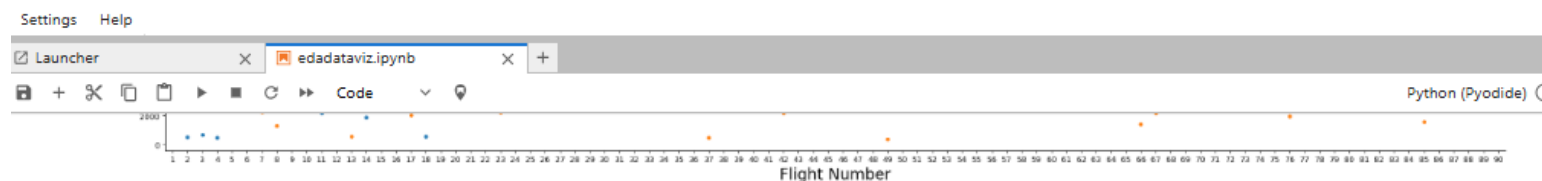
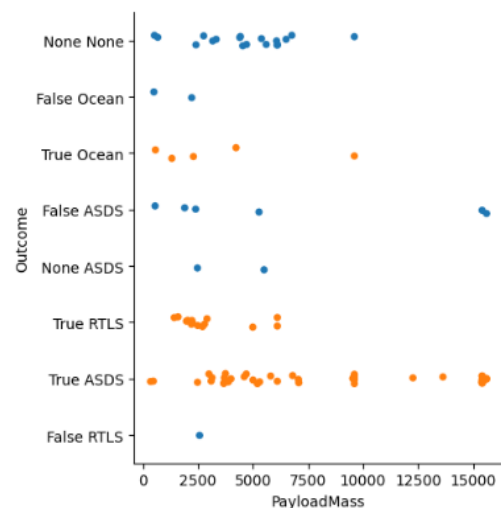
```
[11]: %sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
* sqlite:///my_data1.db
Done.
```

The output for Task 2 is a table with 11 columns: Date, Time (UTC), Booster_Version, Launch_Site, Payload, PAYLOAD_MASS_KG, Orbit, Customer, Mission_Outcome, and Landing_Outcome. It displays 5 records where the launch site starts with 'CCA'.

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)
2012-05-04	00:37:00	F9 v1.0 B0005	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2012-05-22	00:00:00	F9 v1.0 B0006	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)

EDA LAB WITH VISUALIZATION

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.

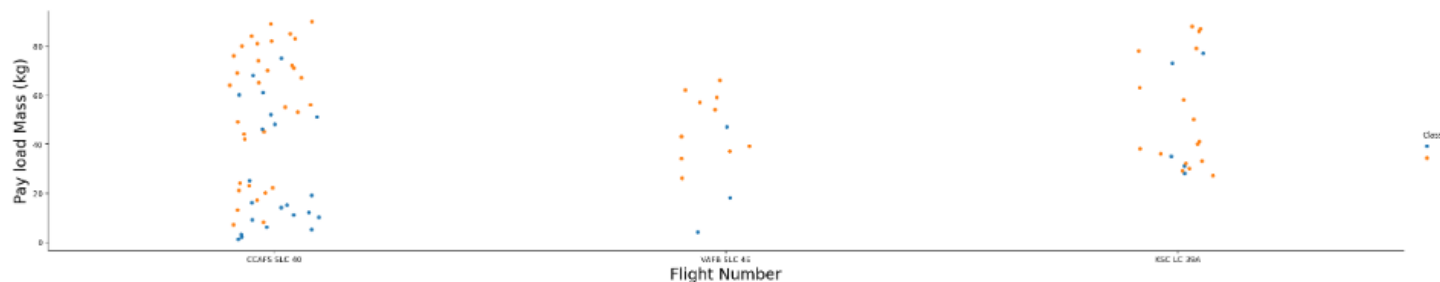


Next, let's drill down to each site visualize its detailed launch records.

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'`

```
[6]: # 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="FlightNumber",x="LaunchSite",hue='Class',data=df, aspect=5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



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

TASK 2: Visualize the relationship between Payload Mass and Launch Site

INTERACTIVE VISUAL ANALYTICS WITH FOLIUM

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
 - Are launch sites near railways, highways and coastlines.
 - Do launch sites keep certain distance away from cities.

BUILD A DASHBOARD WITH PLOTLY DASH

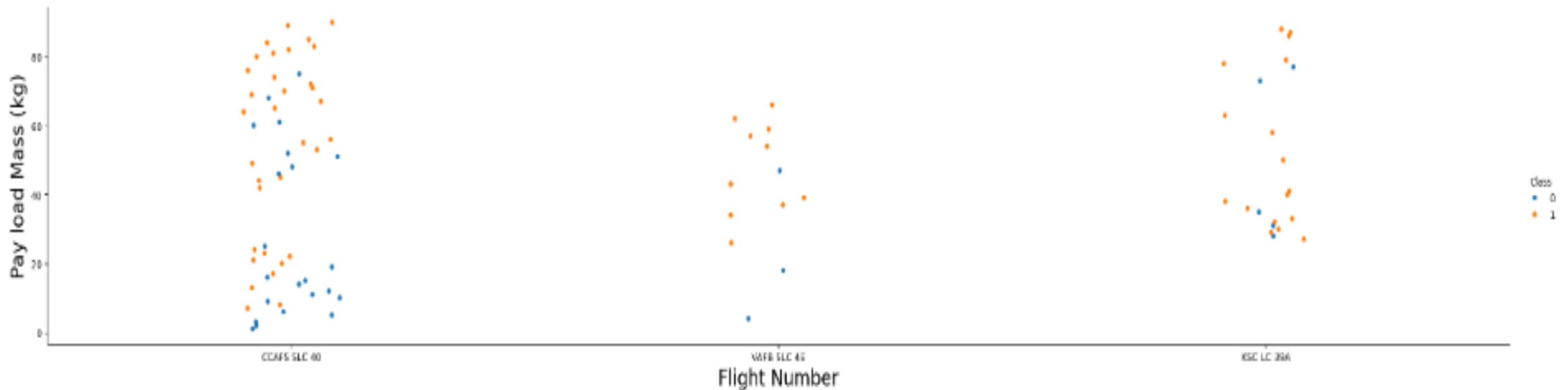
- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

PREDICTIVE ANALYSIS

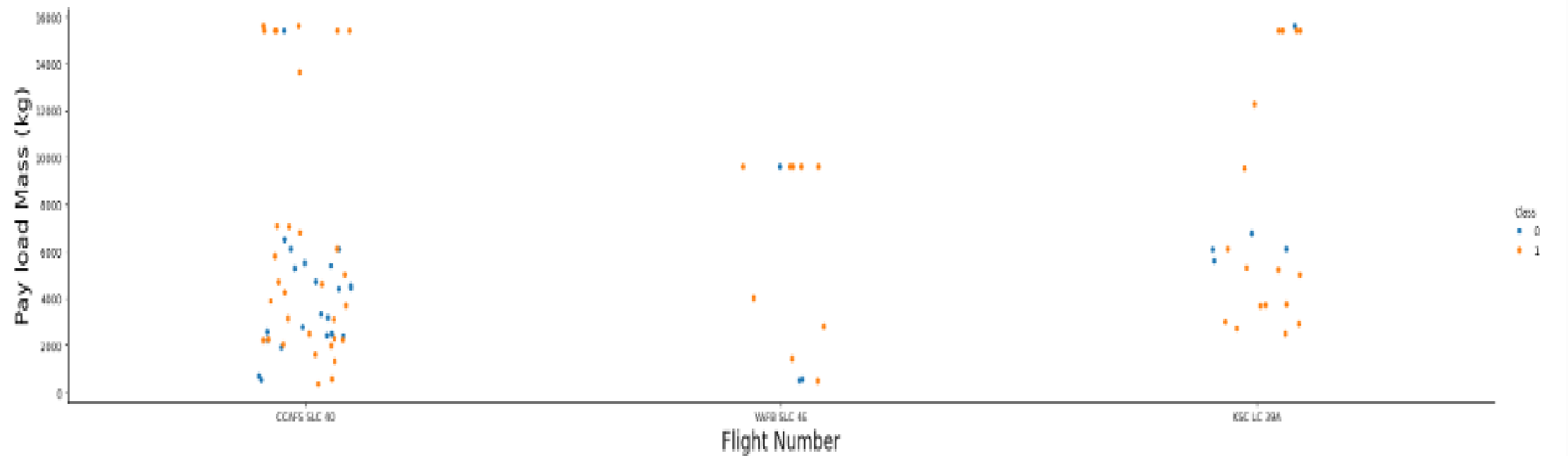
- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is [Spacex-falcon-9-launching-project/SpaceX_Machine Learning Prediction_Part_5.ipynb](https://github.com/sneham0312/Spacex-falcon-9-launching-project/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.ipynb) at main · sneham0312/Spacex-falcon-9-launching-project (github.com)

FLIGHT NO.VS LAUNCH SITE

- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.

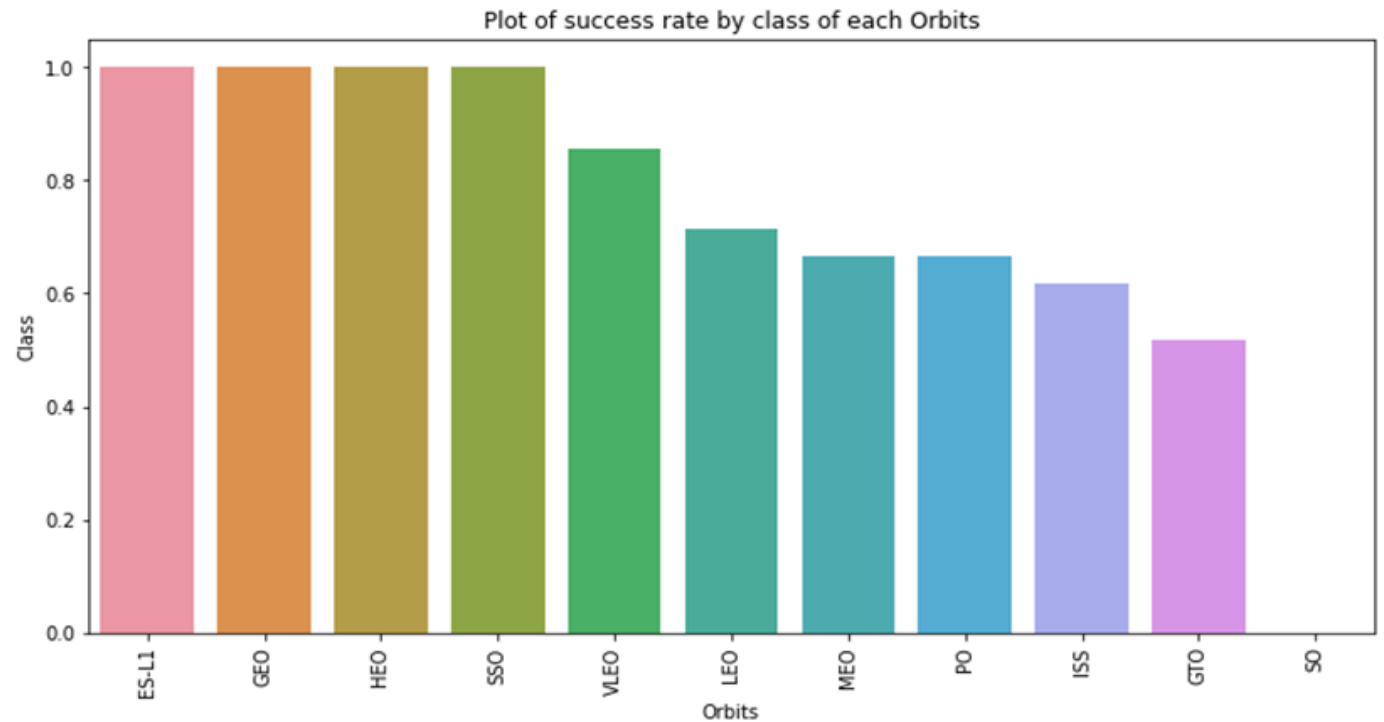


- The greater the payload mass for launch site CCAFS SLC 40 the higher the success rate for the rocket



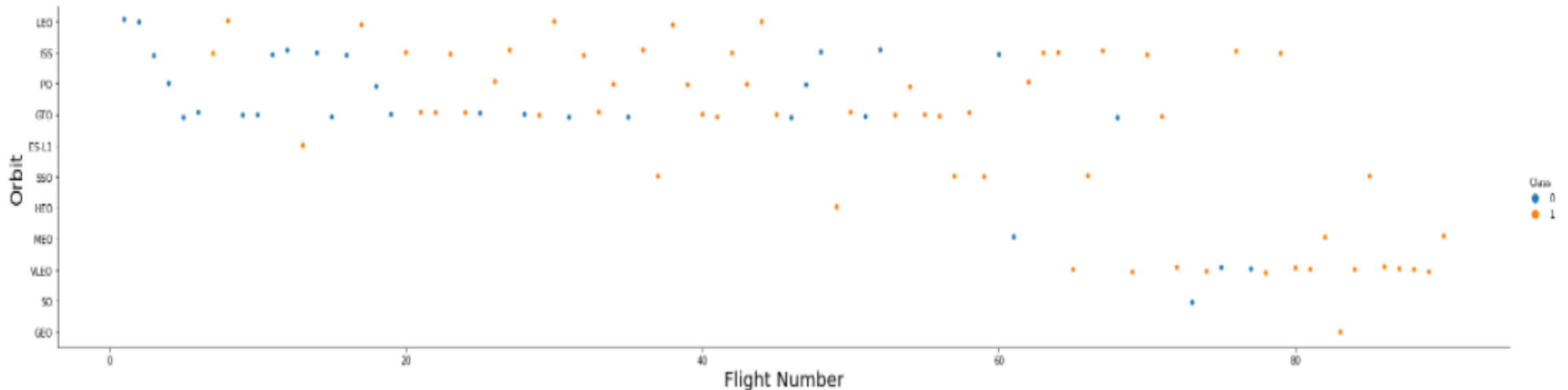
SUCCESS RATE VS ORBIT TYPE

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



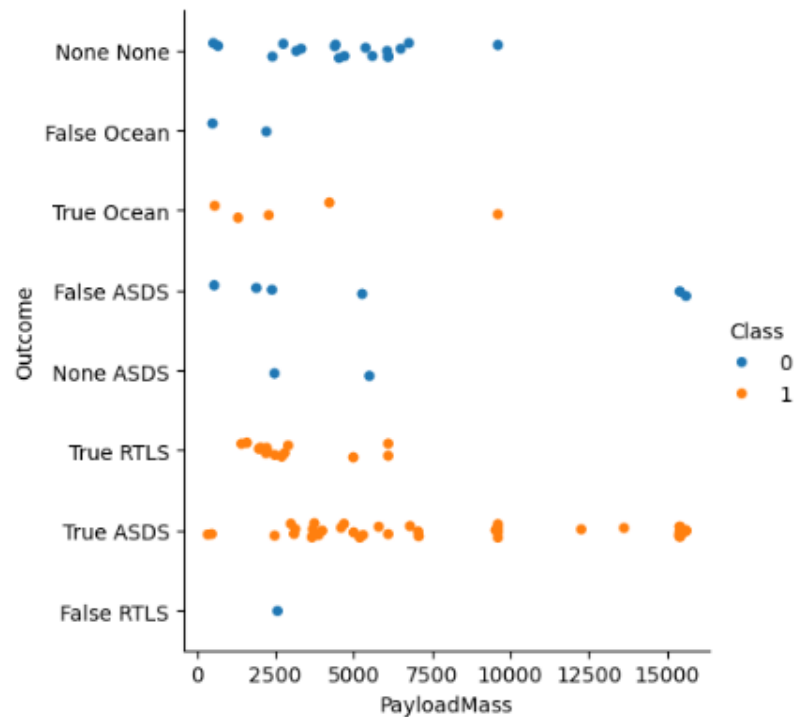
FLIGHT NO. VS ORBIT TYPE

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



PAYLOAD VS ORBIT TYPE

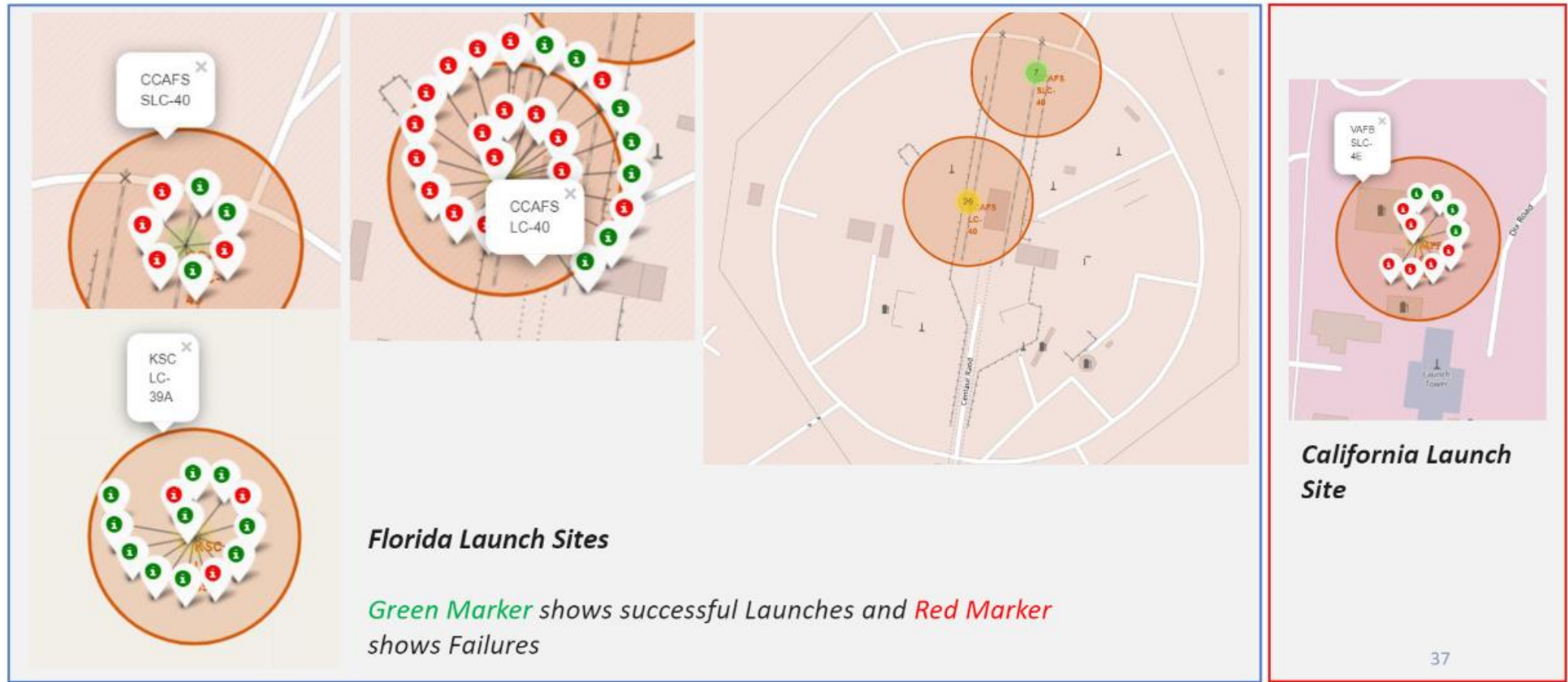
- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS..



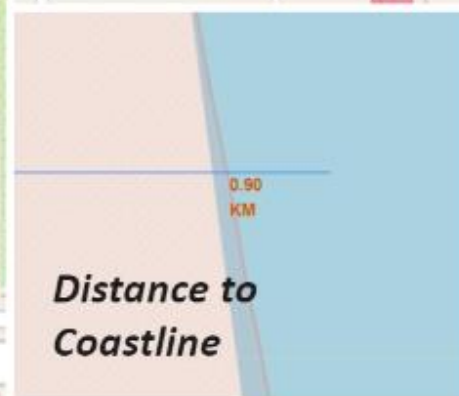
ALL LAUNCH SITES GLOBAL MAP MARKERS



MARKERS SHOWING LAUNCH SITES WITH COLOR LABELS



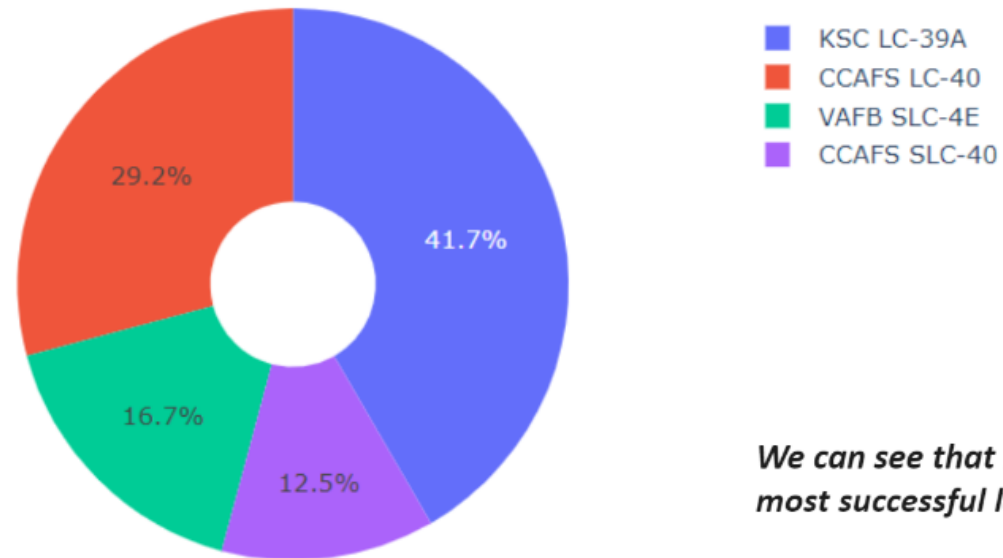
LAUNCH SITE DISTANCE TO LANDMARKS



- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

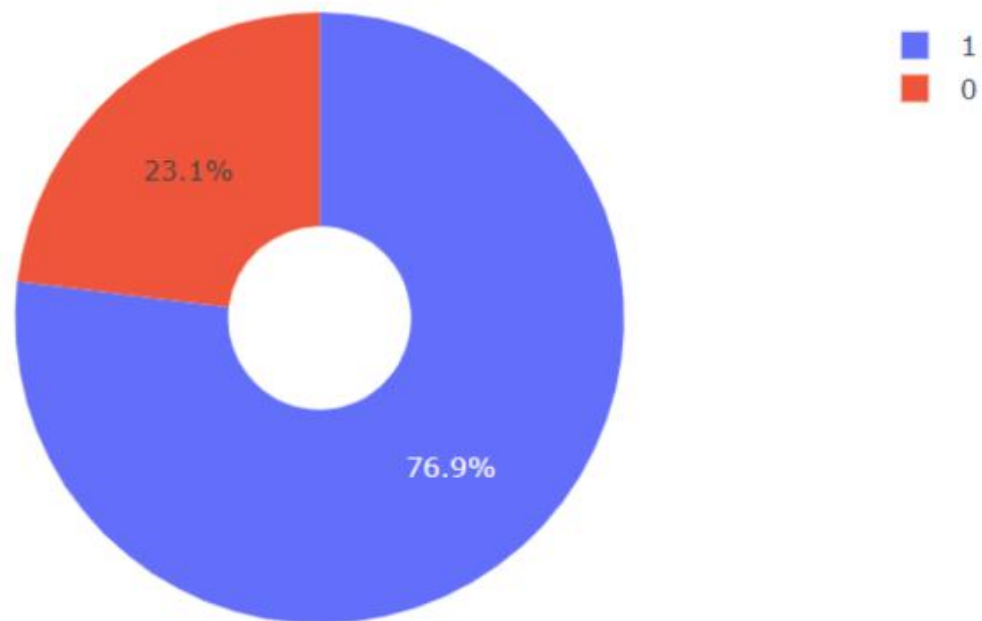
PIE CHART SHOWING THE SUCCESS PERCENTAGE ACHIEVED BY EACH LAUNCH SITE

Total Success Launches By all sites



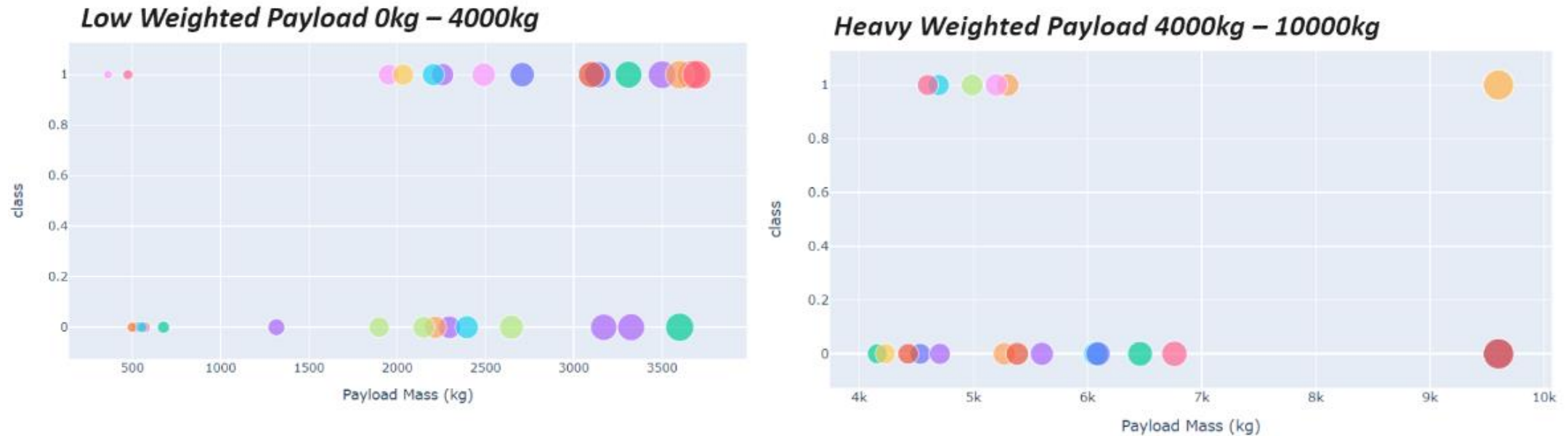
We can see that KSC LC-39A had the most successful launches from all the sites

PIE CHART SHOWING THE LAUNCH SITE WITH THE HIGHEST LAUNCH SUCCESS RATIO



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

SCATTER PLOT OF PAYLOAD VS LAUNCH OUTCOME FOR ALL SITES, WITH DIFFERENT PAYLOAD SELECTED IN THE RANGE SLIDER



We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

CLASSIFICATION ACCURACY

- The decision tree classifier is the model with the highest classification accuracy

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

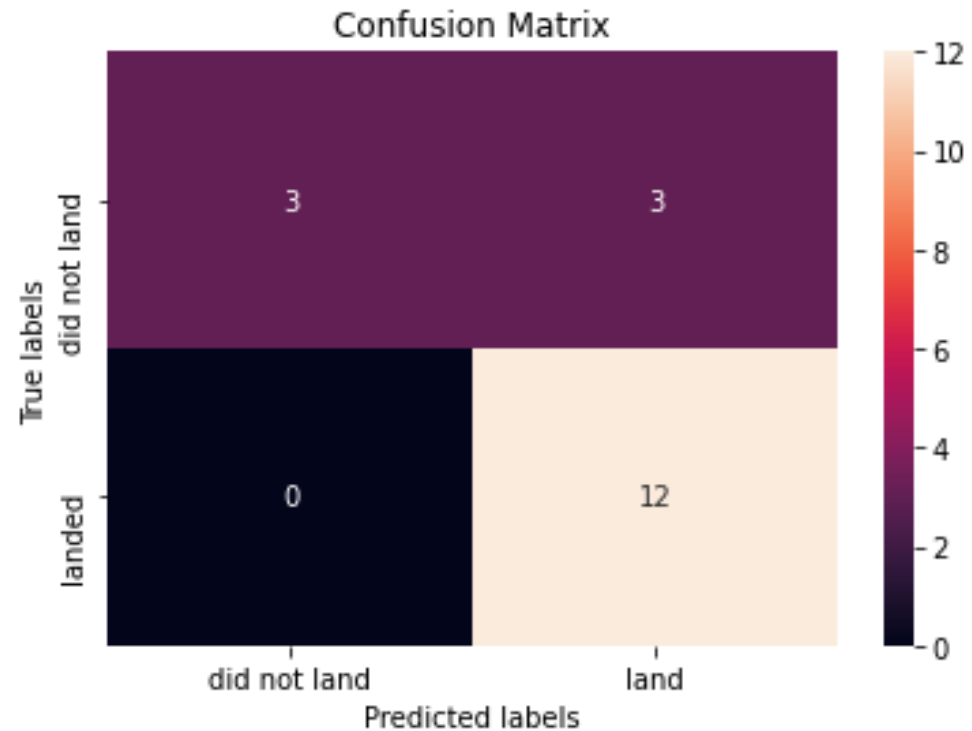
bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

Best model is DecisionTree with a score of 0.8732142857142856

Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}

CONFUSION MATRIX

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives . unsuccessful landing marked as successful landing by the classifier.



CONCLUSION

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.



THANK YOU