Applied Data Science Capstone

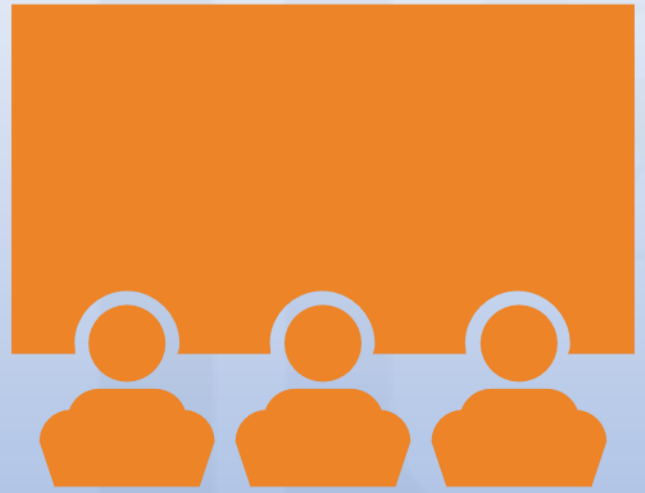
SAQIB MAHMOOD

14.11.2022

IBM Developer

SKILLS NETWORK

OUTLINE



- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Discussion
  - Findings & Implications
- 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

- 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 COLLECITON

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we 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 SCPACE-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.
- https://github.com/msaqibibm/Applied-Data-Science-Capstone/blob/9312b7ec1bc1d9fda5a43997d39925a316d225b8/jupyter-labs-spacex-data-collection-api.ipynb

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 successfull with the 200 status response code

```
response.status_code
```

```
200
```

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

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

Using the dataframe `data` print the first 5 rows

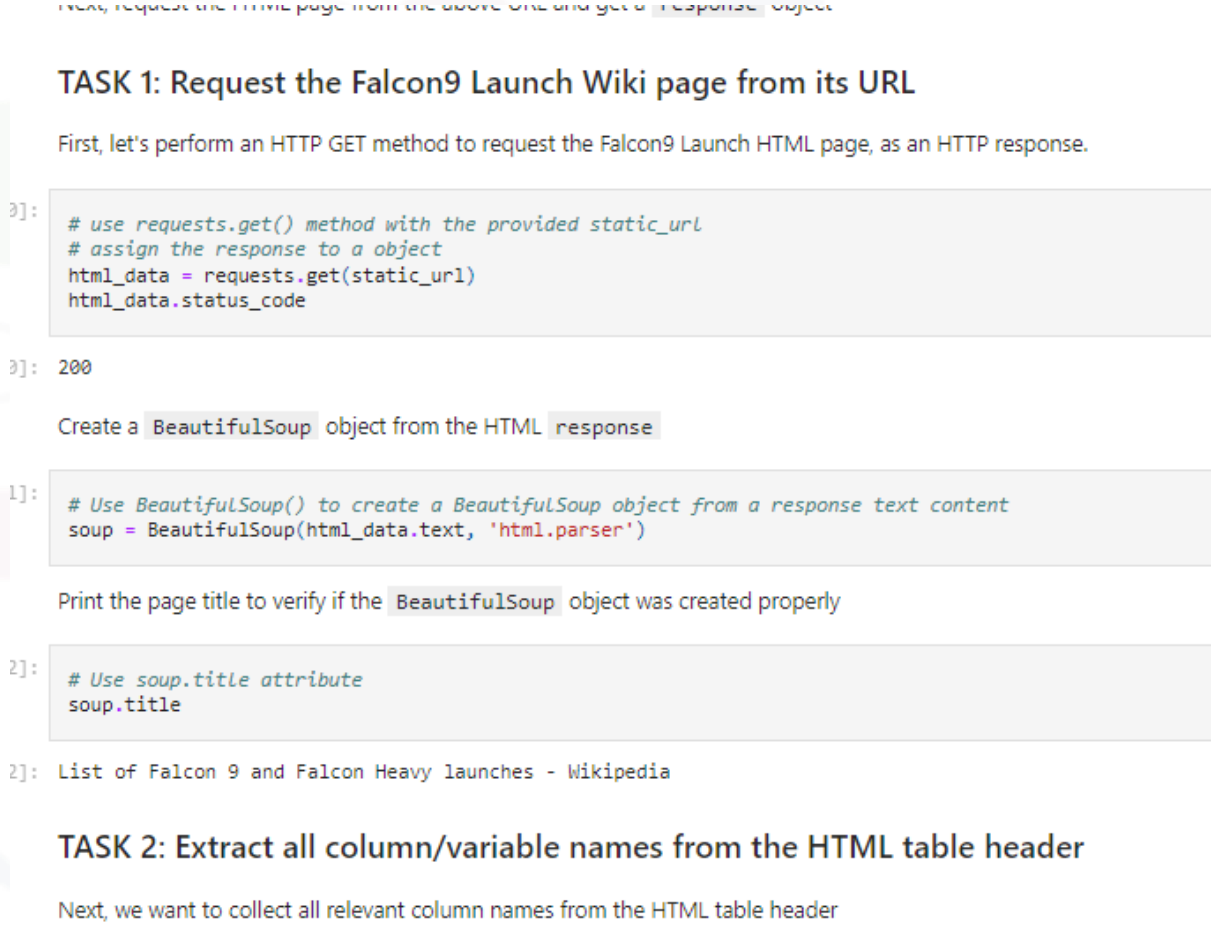
```
# Get the head of the dataframe
data.head(5)
```

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	payloads
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0e4b5b6c3bb0006eeb1e1]	5e9e45

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.

- https://github.com/msaqibibm/Applied-Data-Science-Capstone/blob/9312b7ec1bc1d9fda5a43997d39925a316d225b8/jupyter-labs-webscraping.ipynb



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.

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

```
200
```

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

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

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

```
# Use soup.title attribute
soup.title
```

```
List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

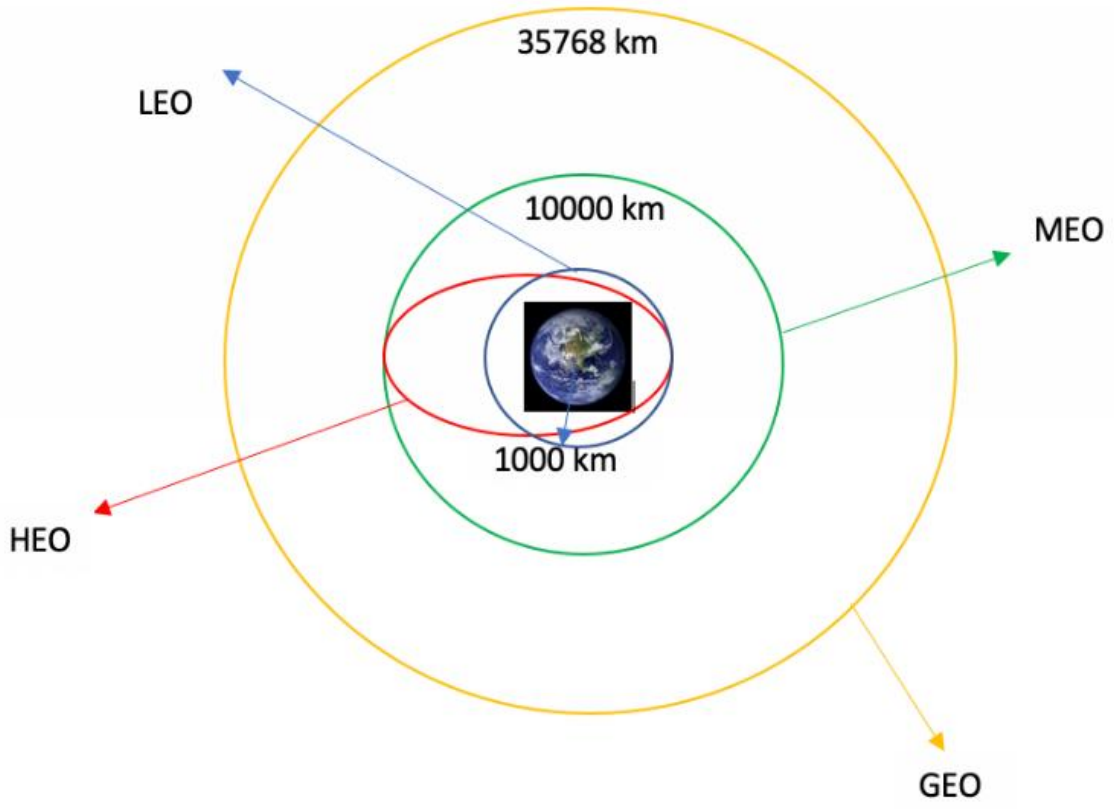
TASK 2: Extract all column/variable names from the HTML table header

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

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.

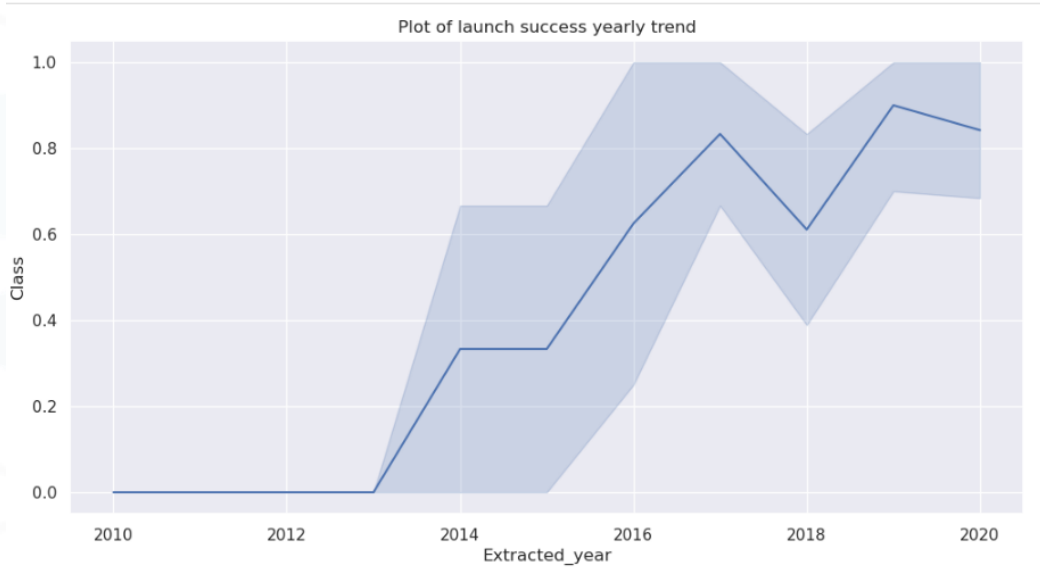
- https://github.com/msaqibibm/Applied-Data-Science-Capstone/blob/9312b7ec1bc1d9fda5a43997d39925a316d225b8/labs-jupyter-spacex-Data%20wrangling.ipynb

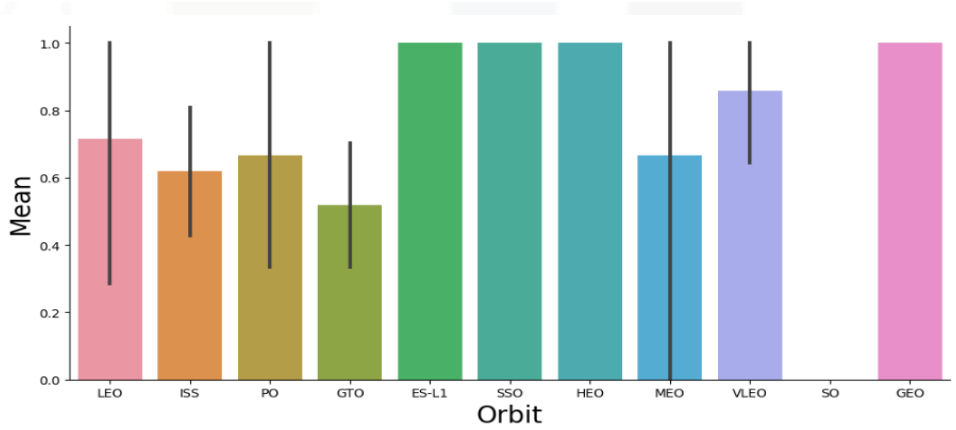


EDA WITH DATA VISUALIZATION

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.

https://github.com/msaqibibm/Applied-Data-Science-Capstone/blob/9312b7ec1bc1d9fda5a43997d39925a316d225b8/jupyter-labs-eda-dataviz%20(1).ipynb





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.

https://github.com/msaqibibm/Applied-Data-Science-Capstone/blob/9312b7ec1bc1d9fda5a43997d39925a316d225b8/jupyter-labs-eda-sql-coursera2.ipynb

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	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	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Task 3

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

BUILD AN INTERACTIVE MAP 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 POLTY 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.

https://github.com/msaqibibm/Applied-Data-Science-Capstone/blob/9312b7ec1bc1d9fda5a43997d39925a316d225b8/spacex_dash_app.py

PREDECTIVE ANALYSIS (CLASSIFICATION)

- 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.

- https://github.com/msaqibibm/Applied-Data-Science-Capstone/blob/9312b7ec1bc1d9fda5a43997d39925a316d225b8/spacex_dash_app.py

RESULTS

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

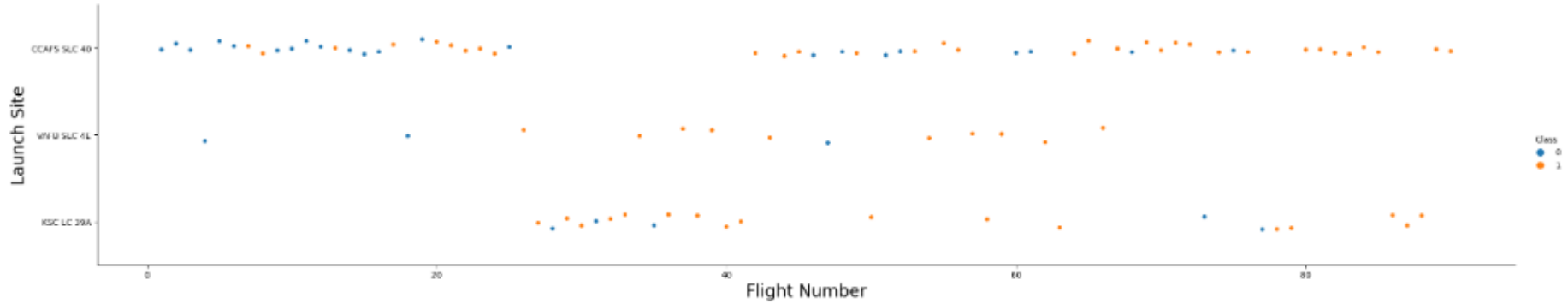
FLIGHT NUMBER VS LAUNCH SITE

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

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'`

[10]:
```
# 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("Launch Site", fontsize=20)
plt.show()
```

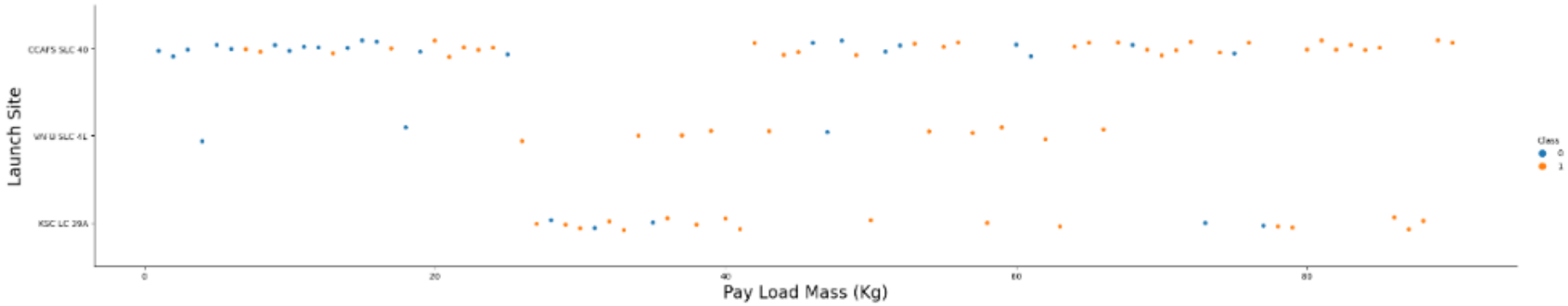


PAYLOAD VS LAUNCH SITE

TASK 2: Visualize the relationship between Payload and Launch Site

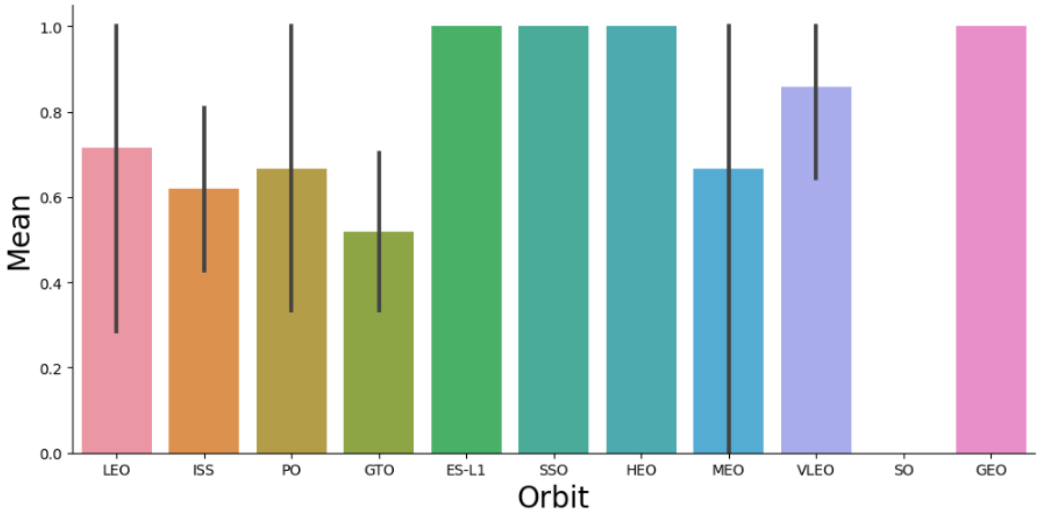
We also want to observe if there is any relationship between launch sites and their payload mass.

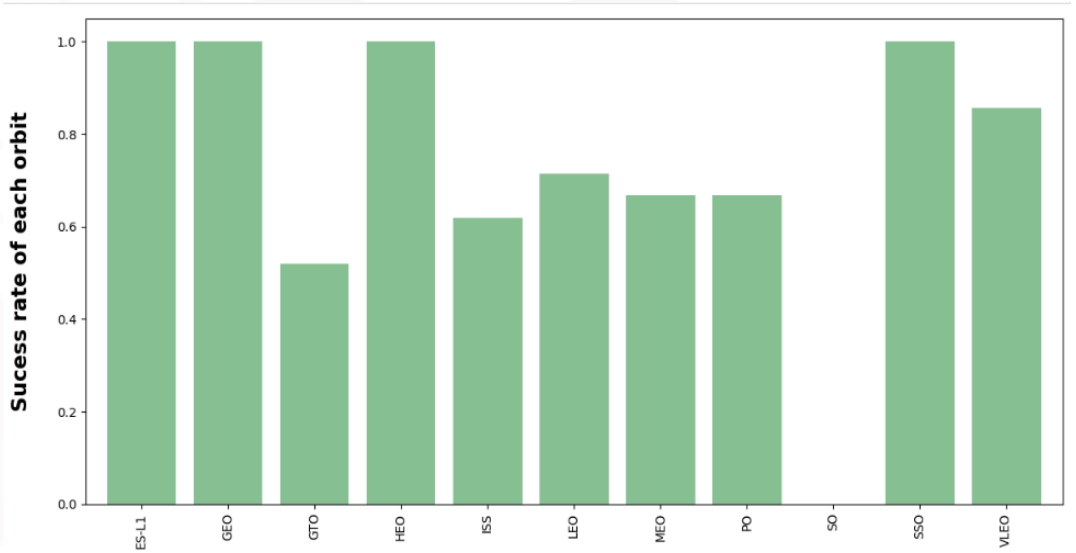
[15]:
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
plt.figure(figsize=(14,8))
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay Load Mass (Kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



SUCCESS RATE VS ORBIT TYPE

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





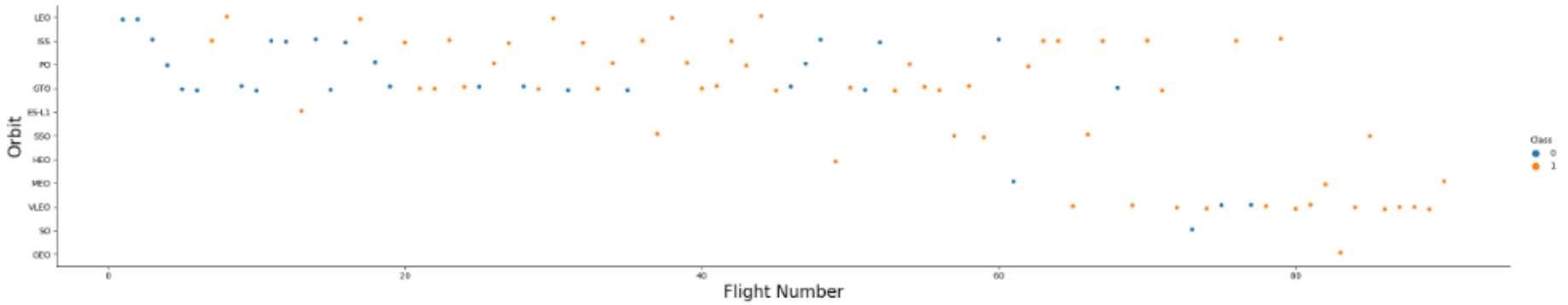
FLIGHT NUMBER VS ORBIT TYPE

- 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.

TASK 4: Visualize the relationship between FlightNumber and Orbit type

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

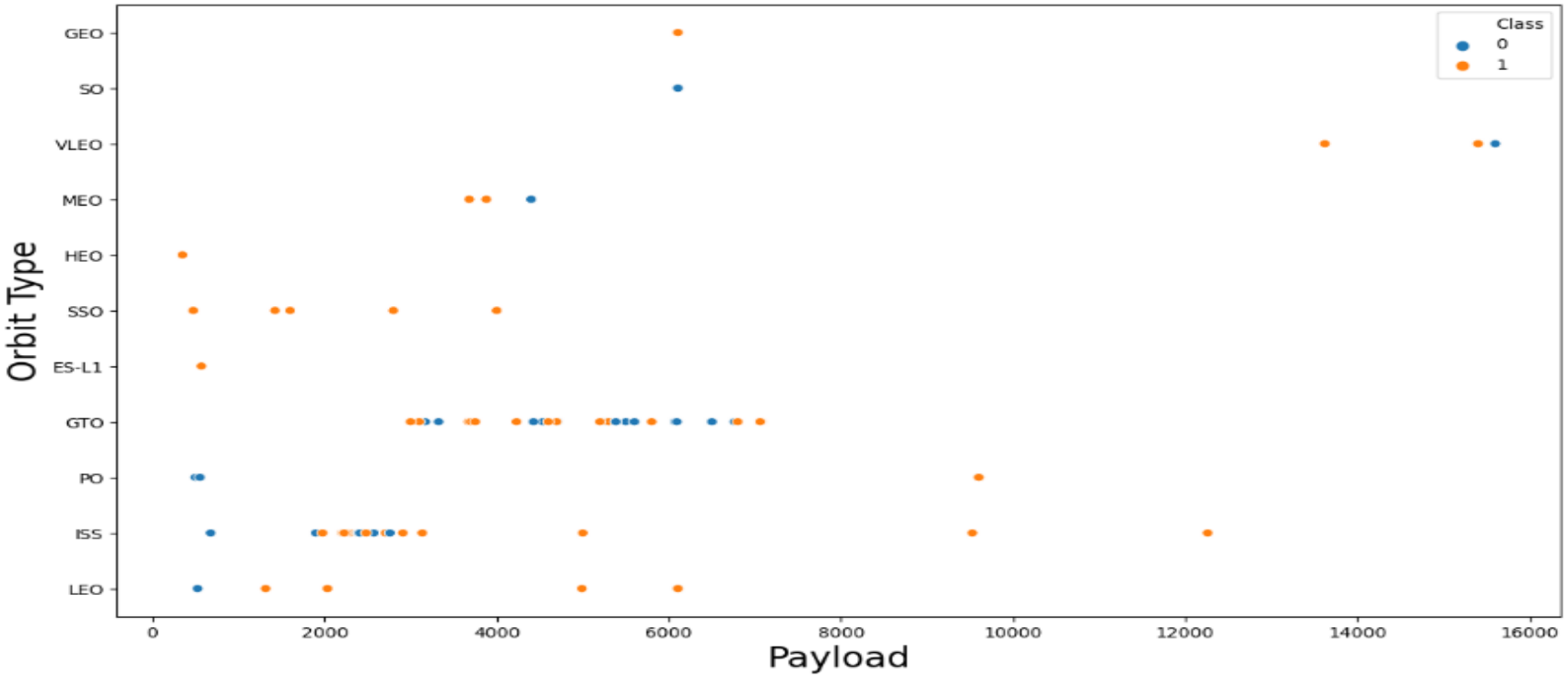
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



PAY LOAD VS ORBIT TYPE

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.

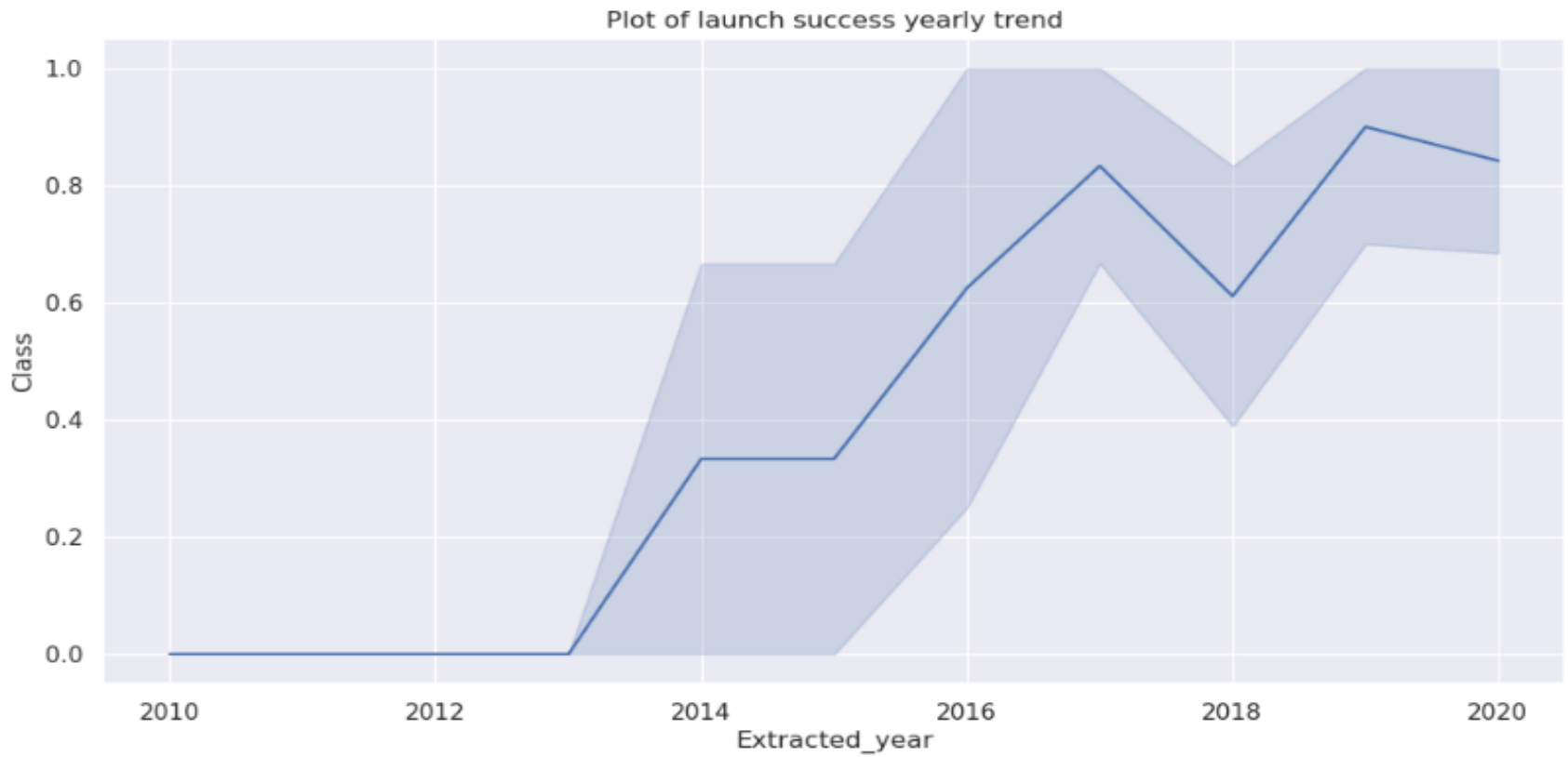
```
plt.figure(figsize=(14,8))
sns.scatterplot(x="PayloadMass", y="Orbit", hue="Class", data = df)
plt.xlabel("Payload",fontsize=20)
plt.ylabel("Orbit Type",fontsize=20)
plt.show()
```



LAUNCH SUCCESS YEARLY TREND

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.

```
# plot line chart
fig, ax=plt.subplots(figsize=(12,6))
sns.lineplot(data=df_copy, x='Extracted_year', y='Class')
plt.title('Plot of launch success yearly trend');
plt.show()
```



ALL LAUNCH SITE NAME

- We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

```
1 SELECT DISTINCT LAUNCH_SITE AS "Launch_Sites" FROM SPACEX;
```

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

LAUNCH SITE NAMES BEGIN WITH “CCA”

❖Launch Site Names Begin with 'CCA'

```
1 SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

Date	Time...	Booster_...	Launch_Site	Payload	PAYL...	Orbit	Customer	Mission_Outcome	Landing__Outcome
04-06-2010	18:45:00	F9 v1.0 B0...	CCAFS LC-40	Dragon Spacec...	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	15:43:00	F9 v1.0 B0...	CCAFS LC-40	Dragon demo fli...	0	LEO (ISS)	NASA (COT...	Success	Failure (parachute)
22-05-2012	07:44:00	F9 v1.0 B0...	CCAFS LC-40	Dragon demo fli...	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0...	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0...	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

TOTAL PAYLOAD MASS

- We calculated the total payload carried by boosters from NASA using the query below

```
1 SELECT SUM (PAYLOAD_MASS__kg_) FROM SPACEX WHERE CUSTOMER LIKE 'NASA(CRS)';
```

SUM (PAYLOAD_MASS__kg_)
NULL

AVERAGE PAYLOAD MASS

```
1 SELECT AVG (PAYLOAD_MASS__KG_) FROM SPACEX WHERE BOOSTER_VERSION = 'F9 v1.1';
```

AVG(PAYLOAD_MASS__KG_)
2928.4

FIRST SUCCESSFUL GROUND LANDING DATE

```
1 SELECT MIN (DATE) AS "First Successful Landing" FROM SPACEX
2 WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

First Successful Landing
01-05-2017

SUCCESSFUL DRON SHIP LANDING WITH PAYLOAD BETWEEN 4000 AND 6000

```
1 SELECT BOOSTER_VERSION FROM SPACEX
2 WHERE LANDING_OUTCOME = 'Success (drone ship)'
3 AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000
```

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

TOTAL NUMBER OF SUCCESSFUL AND FAILURE MISSION OUTCOMES

```
1 SELECT sum(CASE WHEN (MISSION_OUTCOME) LIKE '%Success%' THEN 1 ELSE 0 END) AS "Successful Mission",
2     sum(CASE WHEN (MISSION_OUTCOME) LIKE '%Failure%' THEN 1 ELSE 0 END) AS "Failure Mission"
3 FROM SPACEX;
```

Successful Mission	Failure Mission
100	1

```
1 SELECT COUNT (MISSION_OUTCOME) AS "failure mission "
2 FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Fail%'
```

failure mission
1

```
1 SELECT COUNT (MISSION_OUTCOME) AS "succesful mission" FROM SPACEX
2 WHERE MISSION_OUTCOME LIKE 'Success%'
3
```

succesful mission
100

BOOSTERS CARRIED MAXIMUM PAYLOAD

- We determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

```
1 SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions", payload_mass__kg_ FROM SPACEX
2 WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

Booster Versions	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 LAUNCH RECORD

We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

```
1 SELECT booster_version, launch_site, landing__outcome FROM SPACEX
2 WHERE landing__outcome LIKE 'Failure (drone ship)'
3 AND DATE BETWEEN "01-01-2015" AND "31-12-2015"
```

Booster_Version	Launch_Site	Landing__Outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1017	VAFB SLC-4E	Failure (drone ship)
F9 FT B1020	CCAFS LC-40	Failure (drone ship)
F9 FT B1024	CCAFS LC-40	Failure (drone ship)

RANK LANDING OUTCOMES BETWEEN 04-06-2010 AND 20-03-2017

We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.

We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

```
SELECT LANDING__OUTCOME AS "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACE
WHERE DATE BETWEEN '04-06-2010' AND '20-03-2017'
GROUP BY  LANDING__OUTCOME
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

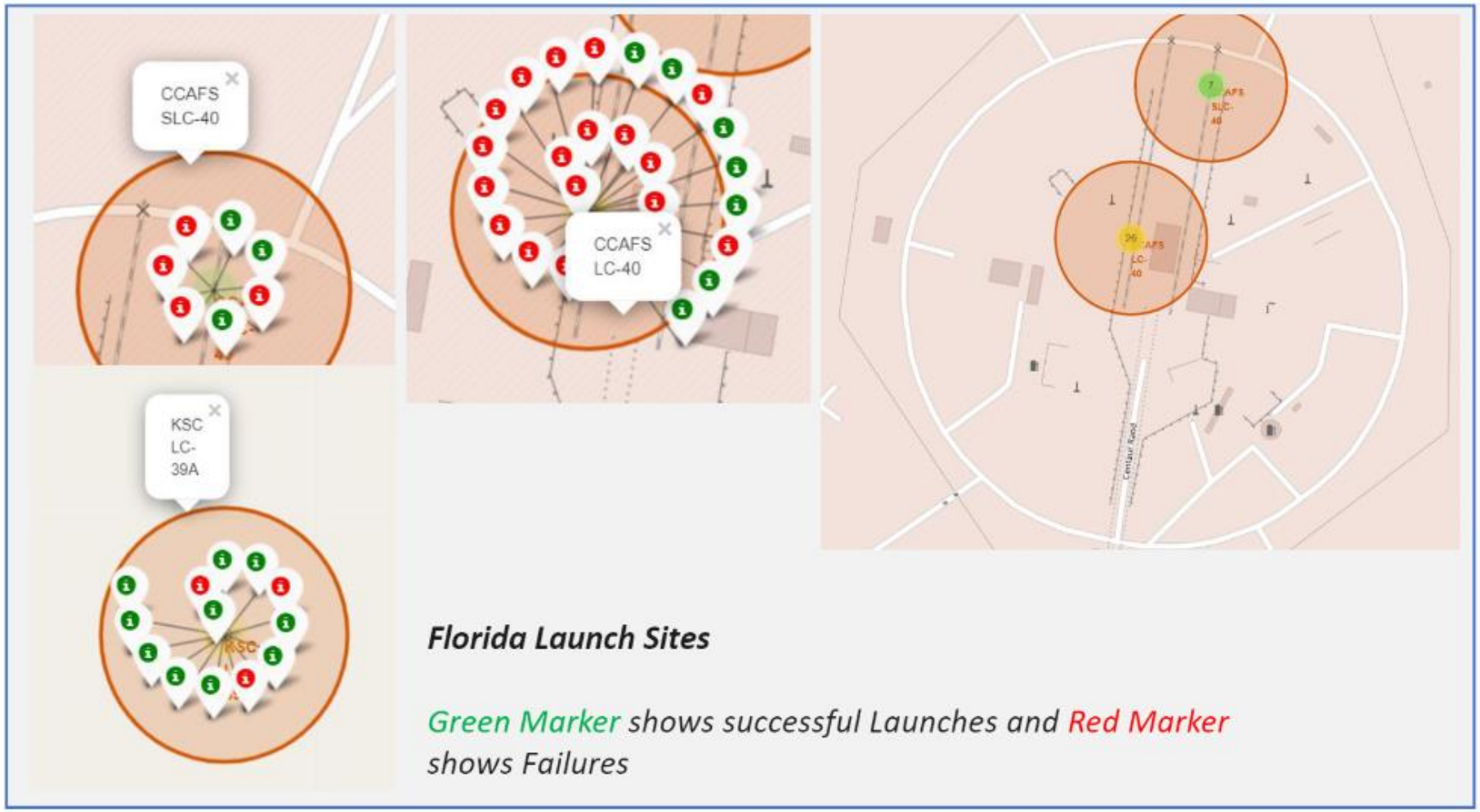
Landing Outcome	Total Count
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

LAUNCH SITE PROXIMITIES ANALYSIS

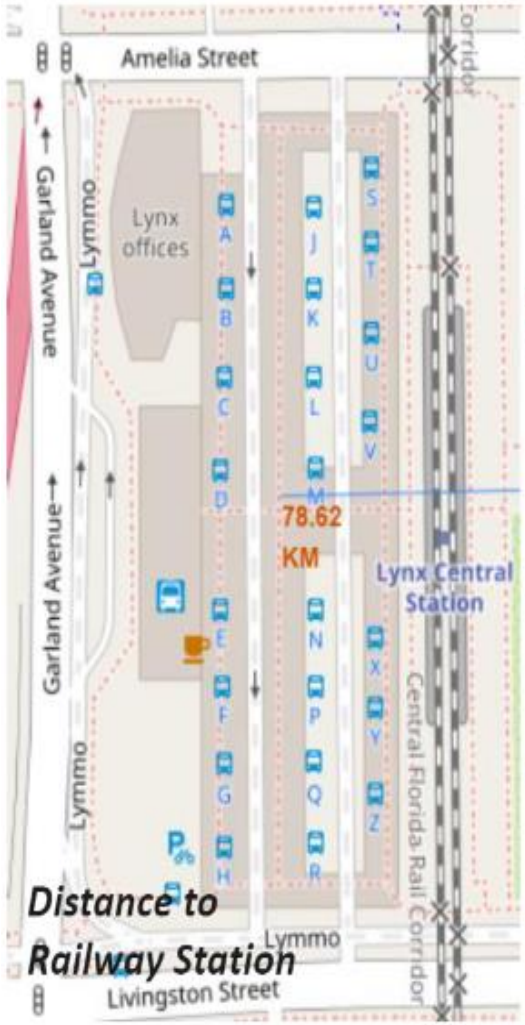
ALL LAUNCH SITES GLOBAL MAP MARKERS

We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

MARKERS SHOWING LAUNCH SITES WITH COLOR LABLES



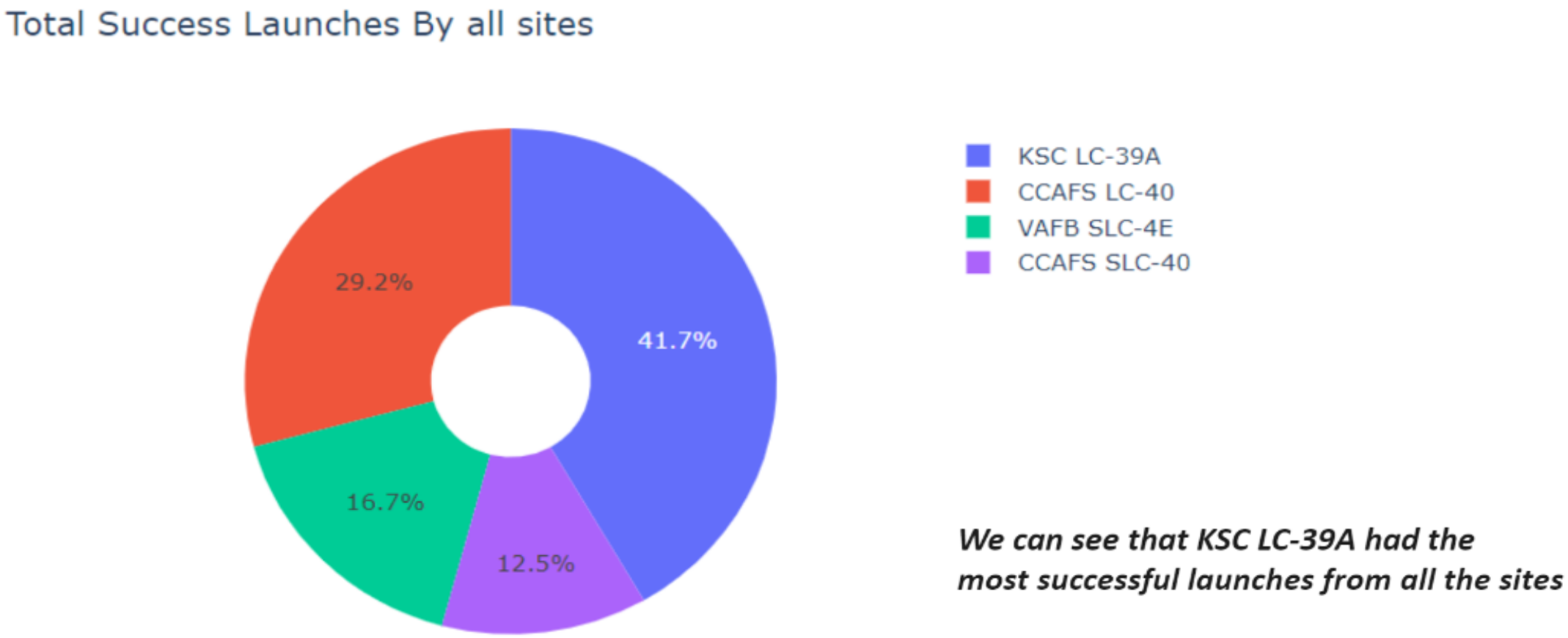
LAUNCH SITE DISTANCE TO LANDMARKS



- 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

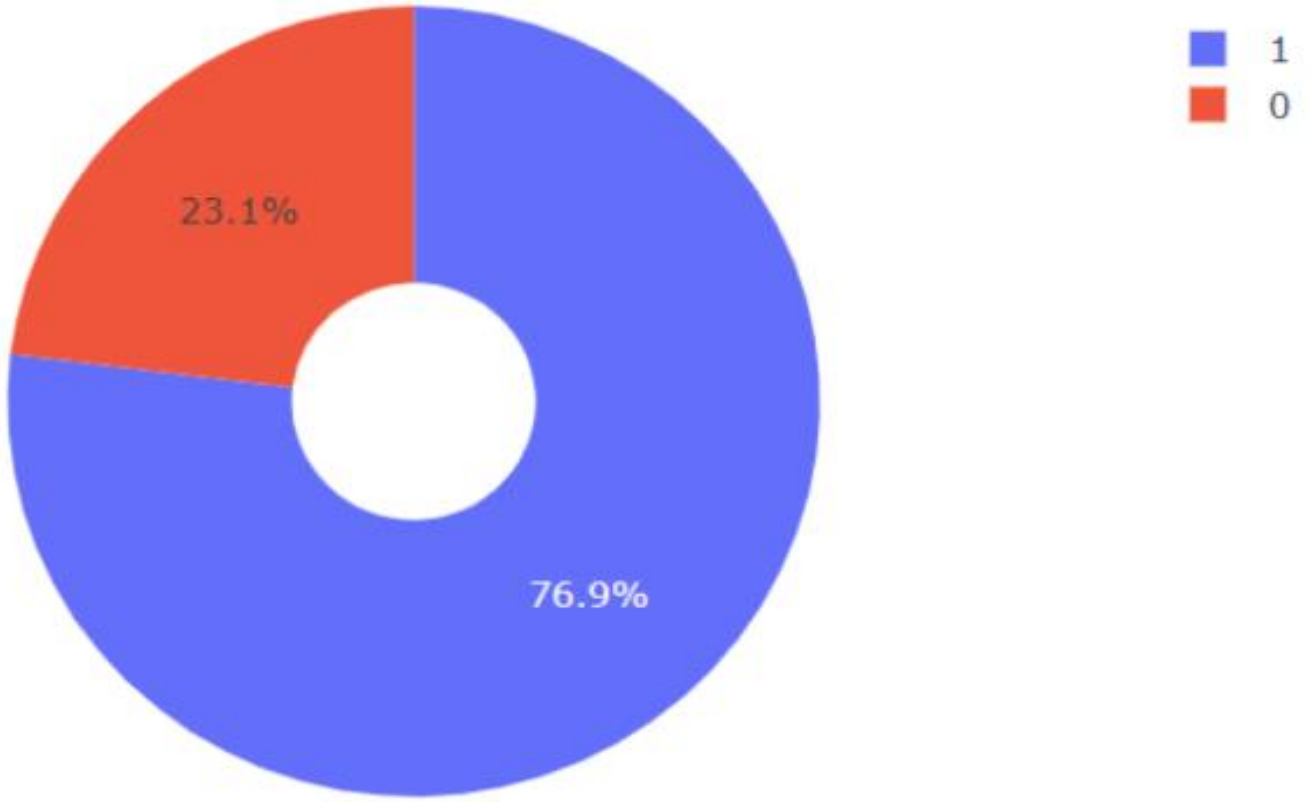
BUILD DASHBOAD WITH POTLY DASH

PIE CHART SHOWING THE SUCCESS PERCENTAGE ACHIVED BY EACH LAUNCH SITE



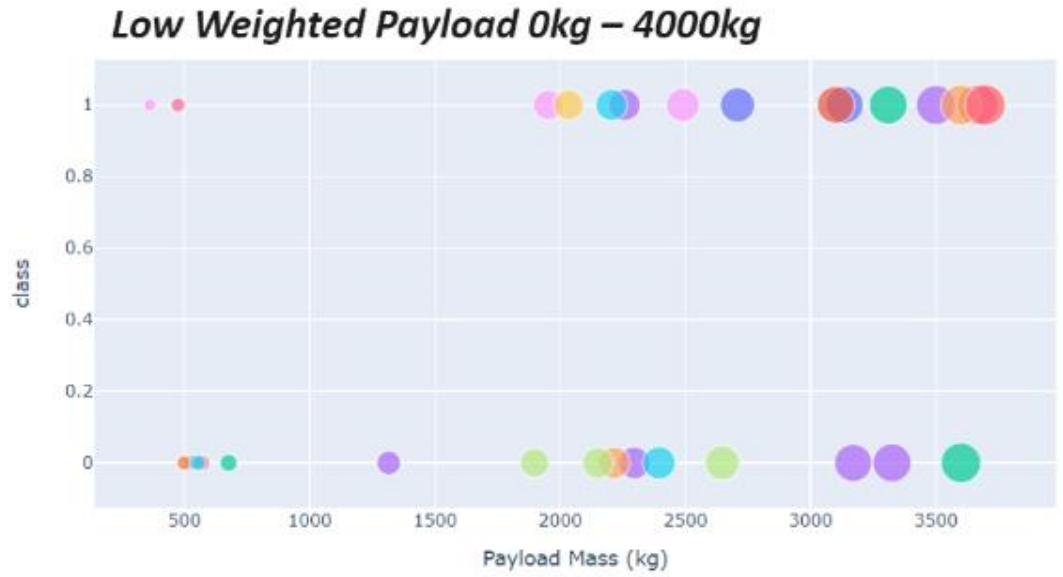
We can see that KSC LC-39A had the most successful launches from all the sites

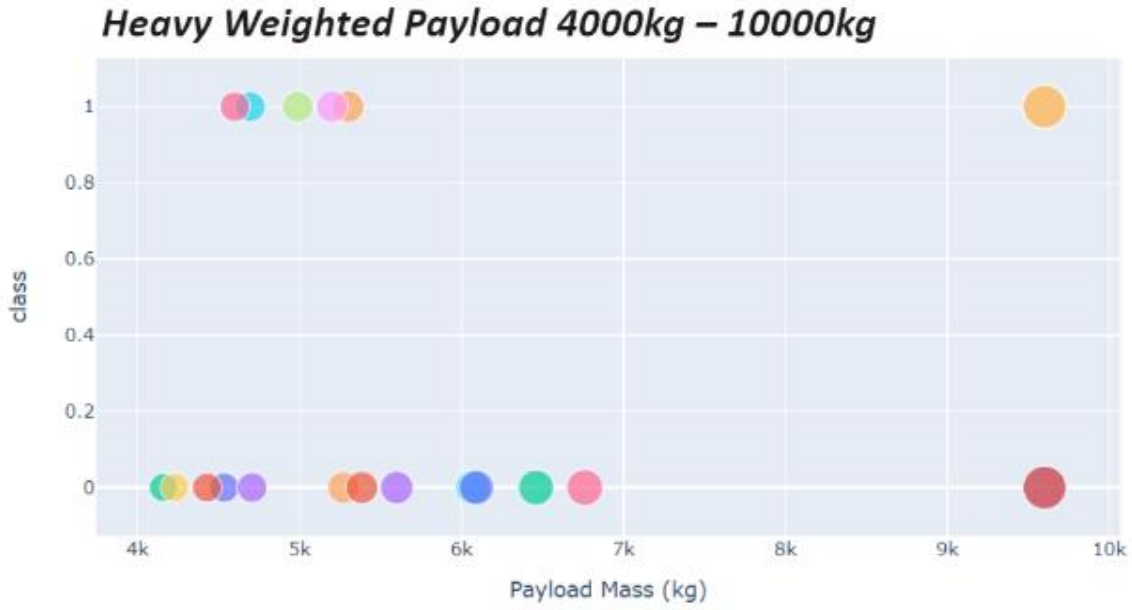
PIE CHART SHOWING THE LAUNCH SITE WITH THE HIGHEST LAUNCHSUCCESS RATIO



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 RANGE SLIDER





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

PREDECTIVE ANALYSIS
CLASSIFICATION

CLASSIFICATION ACCURACY

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

TASK 12

Find the method performs best:

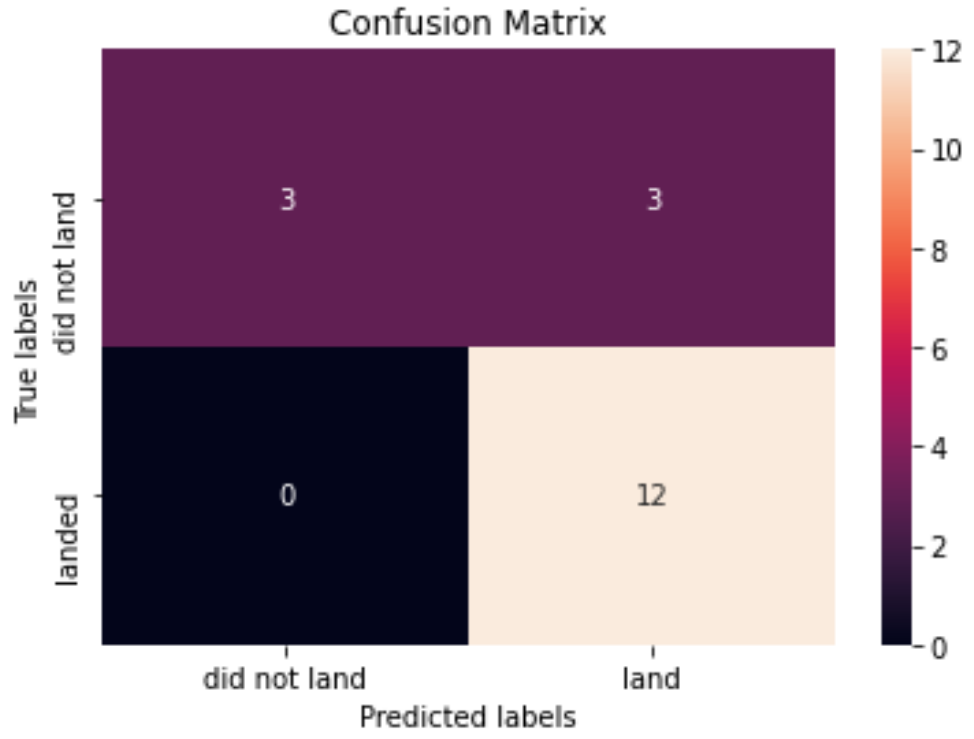
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

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 .i.e., unsuccessful landing marked as successful landing by the classifier.



CONCLUSIONS

- 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