IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

ANDERSON CHUKWUDI OKI
15/08/24

Outline

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

Executive Summary

The quest of Space exploration is getting more thrilling than ever, in this project I predicted the if the first stage of Falcon 9 will land successfully. To achieve this I collected data from the SpaceX website using webscaping utilizing the BeautifulSoup python package, then I converted the data to CSV using pandas which is the library I used for data wranggling. I cleaned the data and got the features and target variable where I used the Numpy library to convert the target variable to Numpy array. I used the train_test_split which is a module from Sklearn library to split the data into 80% for training and 20% for testing.

I used the logistic Regression, Support Vector Machine, Decision Tree and K nearest neighbor classifiers to determine which is best fit for the problem. SVM is among the models that performed greatly with an accuracy score of 83% and correctly predicted the the Falcon that landed successfully.

Introduction

Space exploration is important on the goal to make humans multiplanetary, the project aims to provide good data analysis on the Falcon 9 SpaceX data to predict the chances of successful landing so as to help save the cost of space exploration.

This project aims to answer the following questions;

- To find the names of the launch sites and analyse how it affects the successful landing rate.
- To find the best parameters using GridsearchCv for a particular Machine learning Algorithm
- To find the best Machine Learning Algorithm that works best for this particular problem

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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

- Describe how data sets were collected.

The data sets were collected from the Wikipedia page using the request API to get the data after that I used the BeatifulSoup library to create and object from the response I got from the Wikipedia page when this was successful I then applied a custom function to extract all the column names from the the websites. Furthermore I proceeded by creating a data frame by parsing the launch HTML tables which made brings me to the end of data collection because now the data is stored as a variable called df ready for manipulation. To make sure that the data is in the right way I called the isnull() method on it to get the number of missing variables per column. Finally I exported the data as a CSV file (Commma Separated Values)

Data Collection – SpaceX API

Here, I made a call to the SpaceX API using request and the pares it through the pandas json method and normalized the data thereby converting it into a dataframe the filled in missing values.

Github link :

https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/jupyter-labs-spacex-data-collection-api.ipynb

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

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

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

Check the content of the response

```
[12]: #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:

```
[13]: 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

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

```
[14]: 200
```

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

```
[16]: # 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

```
[17]: # Get the head of the dataframe
data.head()
```

Data Collection - Scraping

- A request was made to SpaceX API for data collection the I used BeautifulSoup Library to webscrape

- The GitHub link to the SpaceX API call: https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/jupyter-labs-webscraping.ipynb

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686
```

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.

In [11]:
```
# use requests.get() method with the provided static_url
# assign the response to a object

page = requests.get(static_url)
```

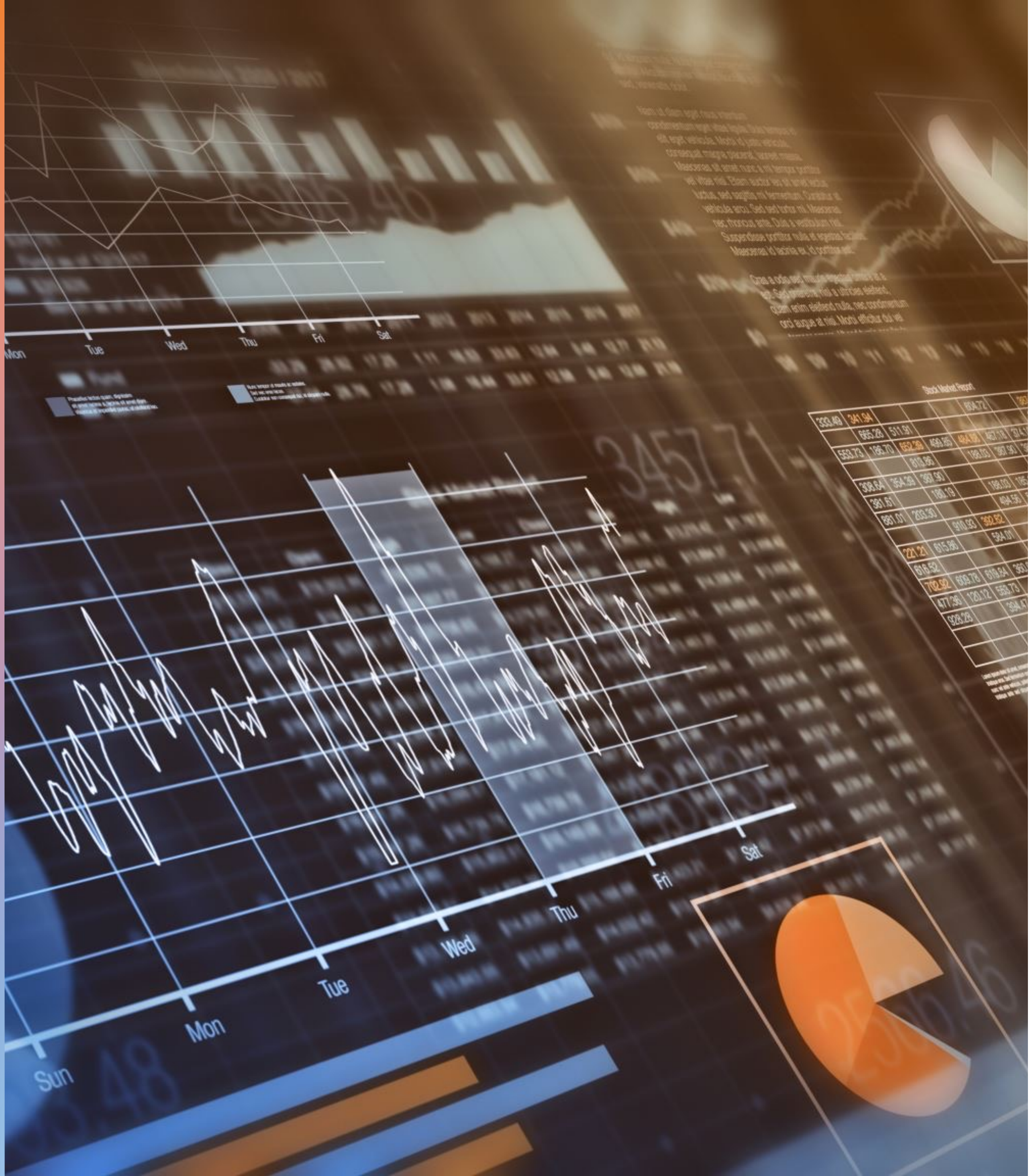
Create a `BeautifulSoup` object from the HTML `response`

In [12]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content

soup = BeautifulSoup(page.text, "html")
```

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

In [13]:
```
# Use soup.title attribute
soup.title
```



Data Wrangling

- Data Wrangling is a process of manipulating the raw data, data cleaning, feature engineering so as to make the data suitable for analytics and predictive purposes. This process started with finding the missing values, which only the launchpad feature has, the I proceeded to know the data types that the features has which was basically float, integer and object (string). Thereafter I used the value_method() to know the number of launch from each site. To finish this I created the outcome feature which will serve as the target variable by knowing if the land was successful or a failure, since is a classification being True or False, I converted it to an integer 1 for success and the 0 for failure.
- The github link to the data wrangling notebook: https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- I plotted a scatter plot which is used to find the relationship between 2 variables in other words correlation, also I plotted a bar plot know know the obit which has the highest success rate, followed by a line plot which is used to visualize trends.
- The GitHub link: https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/edadataviz.ipynb

EDA with SQL

- Displayed the names of the unique launch sites in the space mission
- Displayed 5 records where launch site begins with CCA
- Displayed the total pay load mass carried by Boosters launched by NASA (CRS)
- Displayed the average pay load mass carried by Booster F9 v1.1
- List the total number of successful and failure mission outcomes
- List the names of the booster versions which have carried the maximum payload mass
- GitHub URL: https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

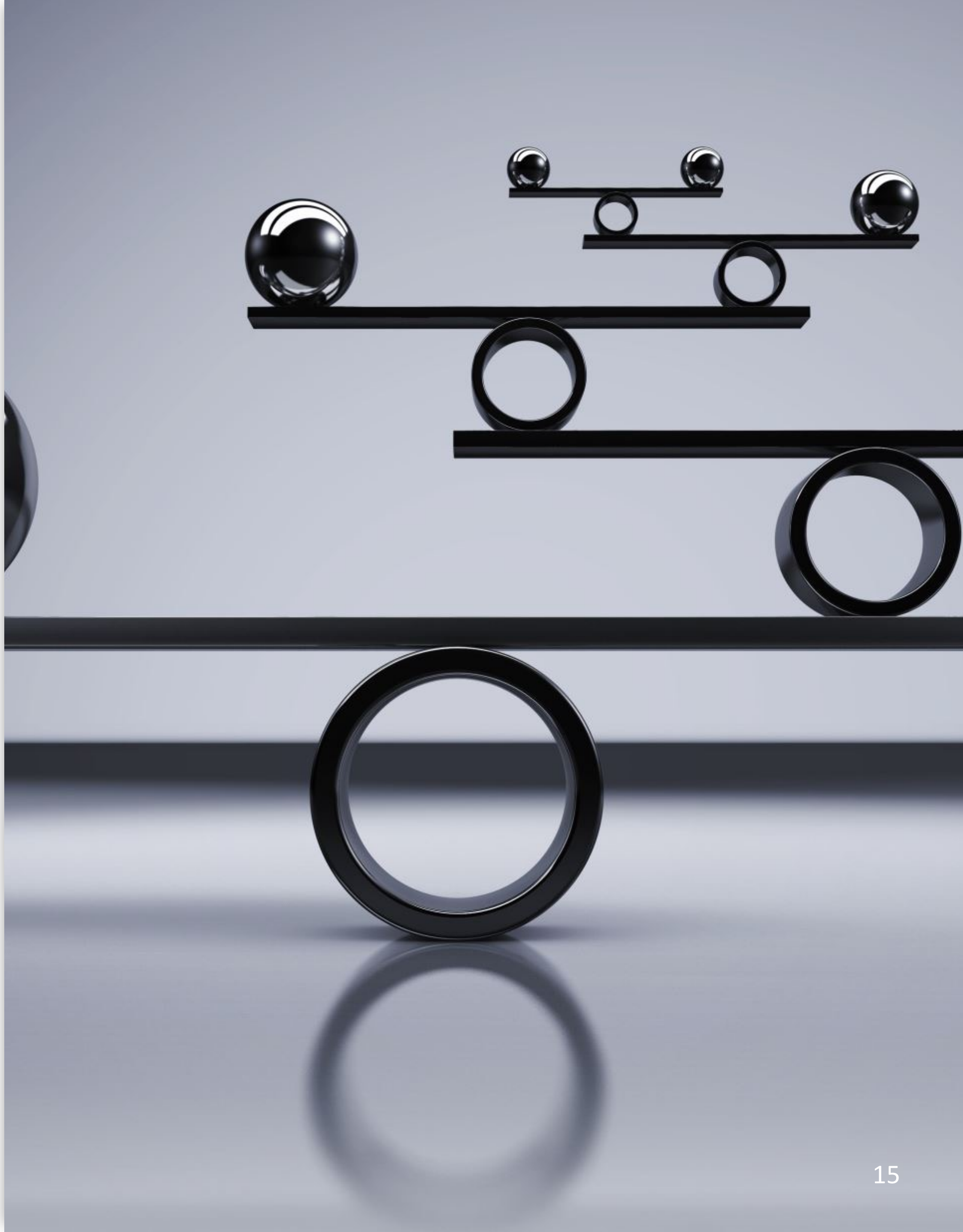
- I marked all launch sites and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- The marker was used to mark the success/failed rate for each sites on the map
- Later I calculated the distance between a launch site to it's proximities
- GitHub URL : https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- I added a pie chart to give a proportion of all the successful launch sites and added a scatter plot for success count on the payload mass for all sites.
- Visualization is key to understanding data, making the data interactive makes it more alive and then proper analysis can be done in real time
- GitHub URL: https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/spacex_dash_app.py

Predictive Analysis (Classification)

- Predictive analysis is the stage where you use all the knowledge that you have gathered through the data wrangling and exploration phase to make future prediction.
- This started by converting the target variable to a numpy array followed by standardizing the features to make them of equal weights then splitting the data sets into training and testing sets by assigning 20% to testing, the I trained and predicted with 4 Machine learning models, logistic Regression, SVM, Decision Trees and KNN. Using the gridSearchCV to find the best parameters and train with the best parameters.
- GitHub URL : https://github.com/CK151/Machine_Learning_Projects/blob/main/Classiffication/Data%20Science%20Capstone%20Project%20(IBM)/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

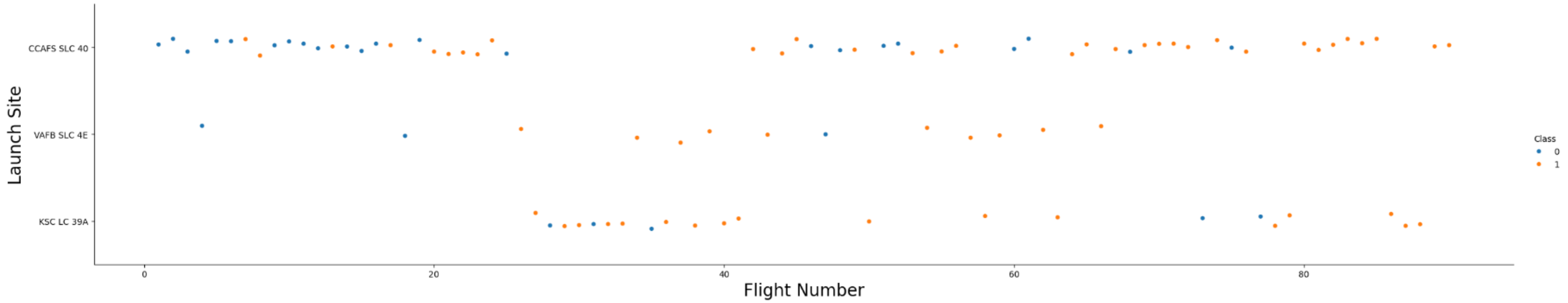


Results

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

Section 2

Insights drawn from EDA

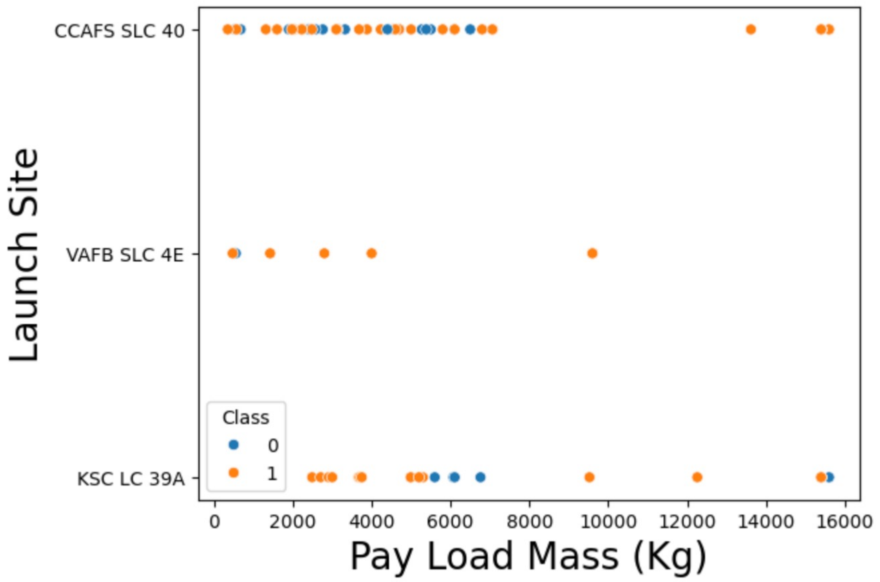


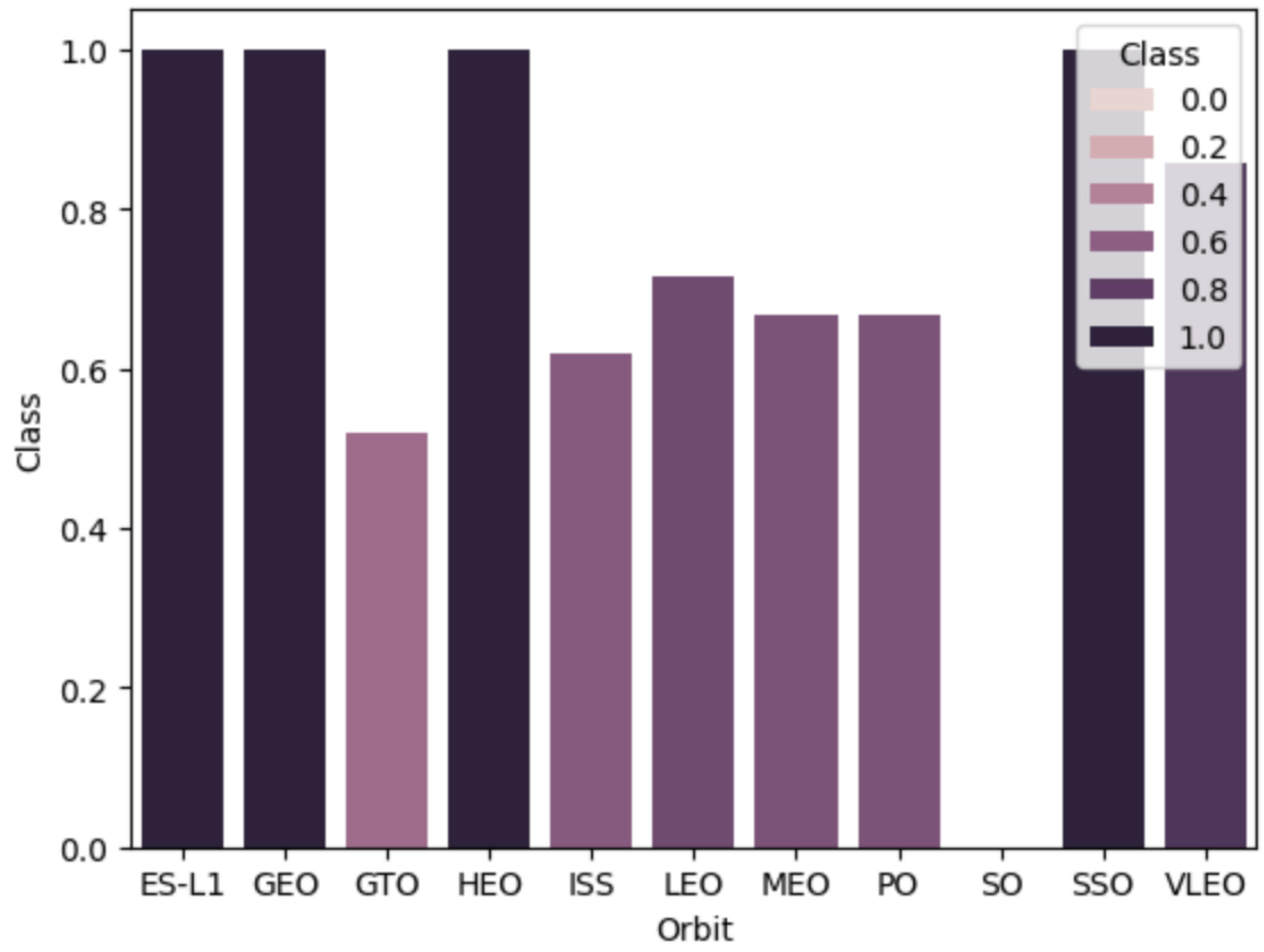
Flight Number vs. Launch Site

- We can see that as the flight number increases the success rate also increases, the CCAFS SLC launch site has the highest first flight number test

Payload vs. Launch Site

- Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000)



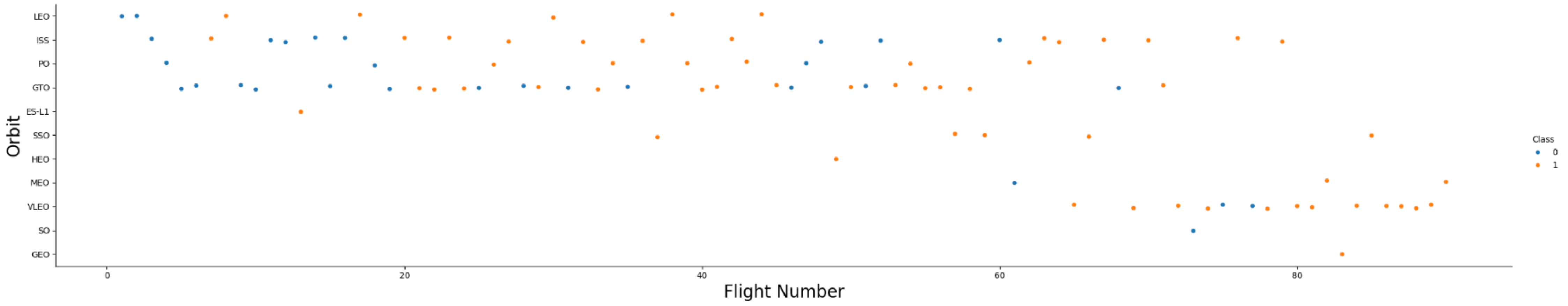


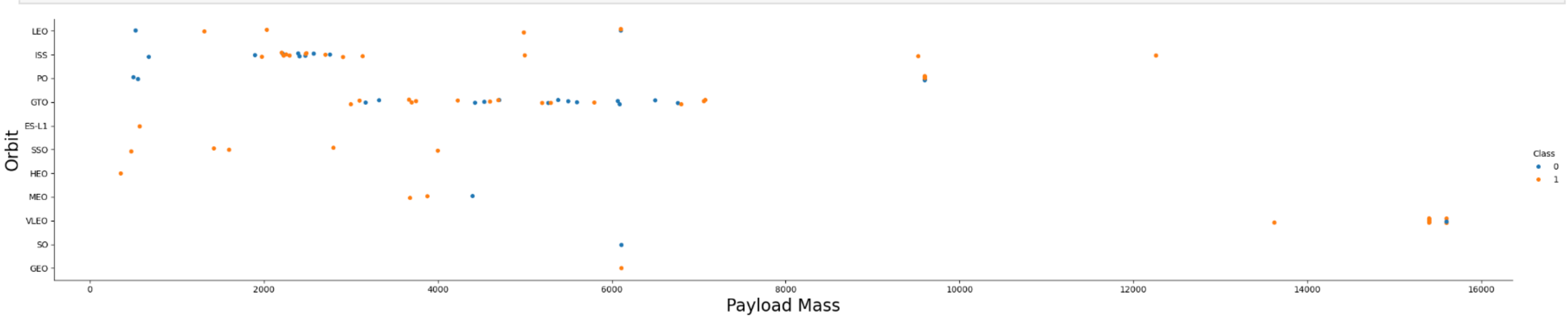
Success Rate vs. Orbit Type

The orbits that has the highest success rate are; ES-L1, GEO, HEO, SSO

Flight Number vs. Orbit Type

- You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.



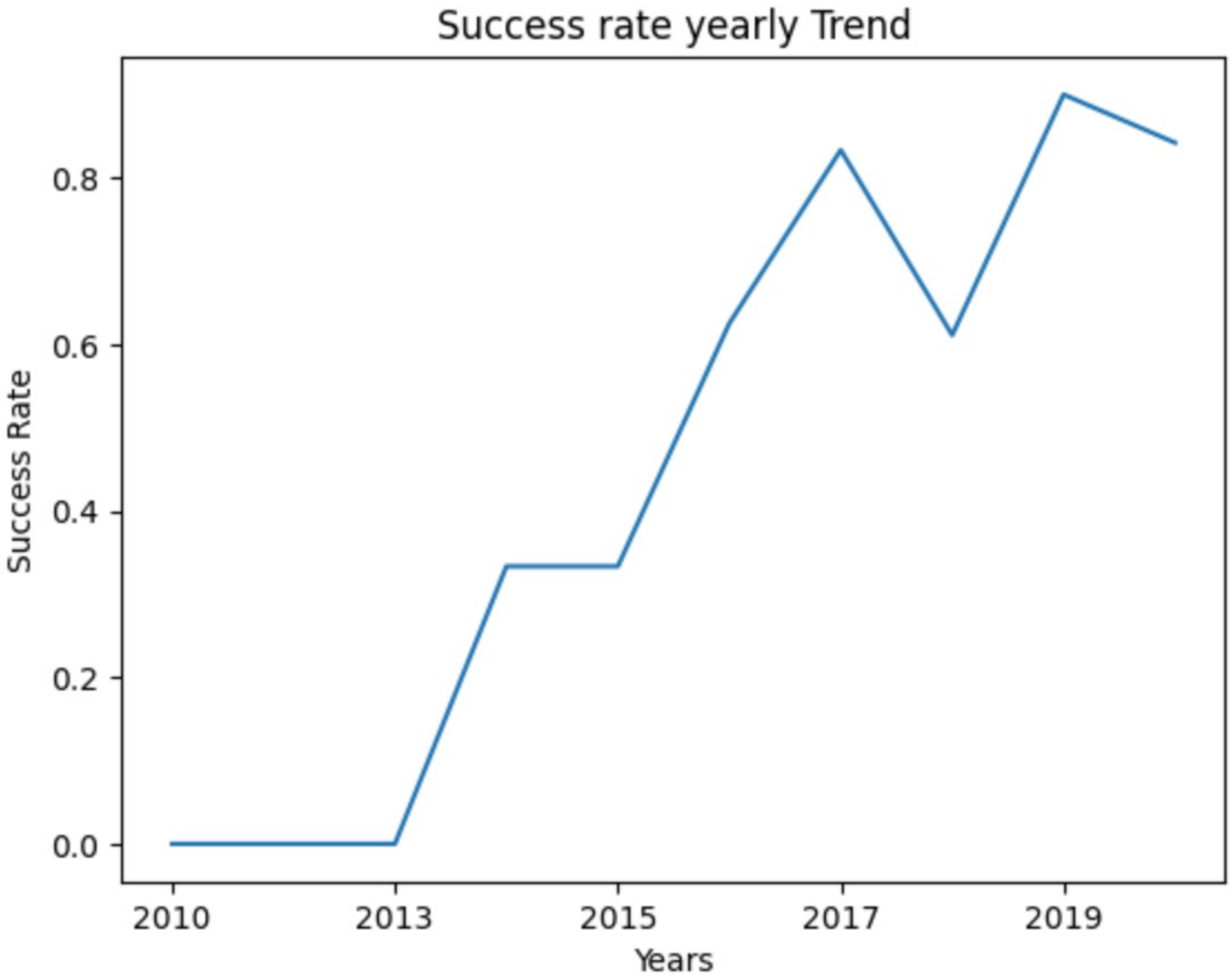


Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

Launch Success Yearly Trend

You can observe that the sucess rate since 2013 kept increasing till 2019



All Launch Site Names

Here using the DISTINCT clause to get the name of unique launch sites

```
%sql select distinct Launch_site from SPACEXTBL
```

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

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

Launch Site Names Begin with 'CCA'

- For this I used the WHERE and LIKE clause to get only the launch sites that begins with “CCA” and also. LIMIT clause to limit it to the first 5

In [16]:
```
%sql SELECT Launch_site from SPACEXTBL where Launch_site like "CCA%" limit 5;
```

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

Out[16]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

The total payload carried by boosters from NASA

```
%sql select sum(PAYLOAD_MASS__KG_) as sum_payload_mass from SPACEXTBL where Customer = "NASA (CRS)";
```

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

sum_payload_mass
45596

Average Payload Mass by F9 v1.1

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

```
In [23]: %sql select AVG(PAYLOAD_MASS__KG_) as AVG_payload_mass from SPACEXTBL where Booster_Version = "F9 v1.1";

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

Out[23]:

AVG_payload_mass
2928.4

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad

Hint:Use min function

```
%sql select min(Date) FROM SPACEXTBL where Landing_Outcome = "Success (ground pad)";
```

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

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- The names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

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

[40]:

```
%sql select Booster_Version from SPACEXTBL where Landing_Outcome = "Success (drone ship)" and PAYLOAD_MASS__KG_ E
```

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

[40]:

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

Total Number of Successful and Failure Mission Outcomes

The total number of successful and failure mission outcomes

]:

```
%sql SELECT Mission_Outcome, COUNT(*) FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

]:

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

Boosters Carried Maximum Payload

The names of the booster which have carried the maximum payload mass

Booster_Version	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 Records

- The failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

Done.

Out[49]:

Booster_Version	Launch_Site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

Done.

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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

[51]:

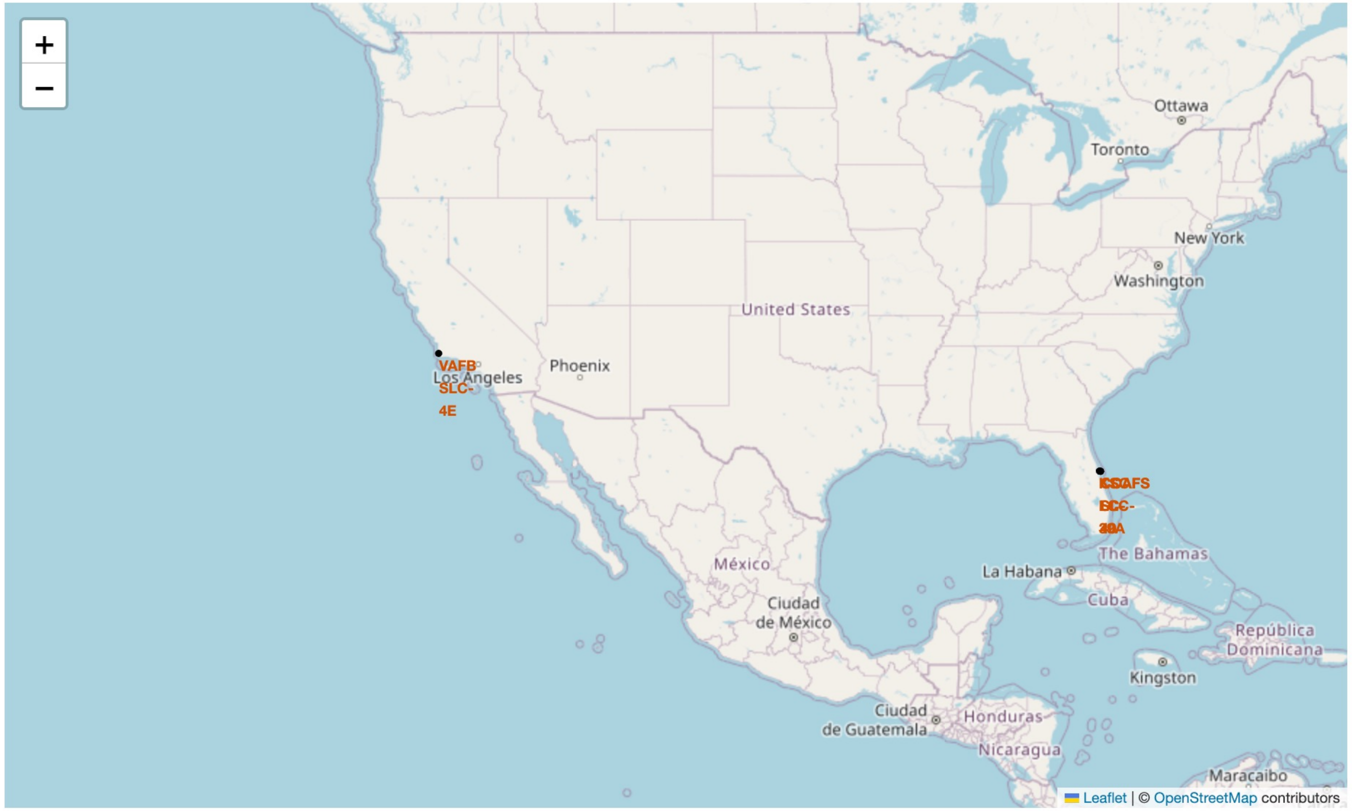
Landing_Outcome	qty
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

All Launch Sites Global Markers

- It is visible that all launch sites are located in the United States of America

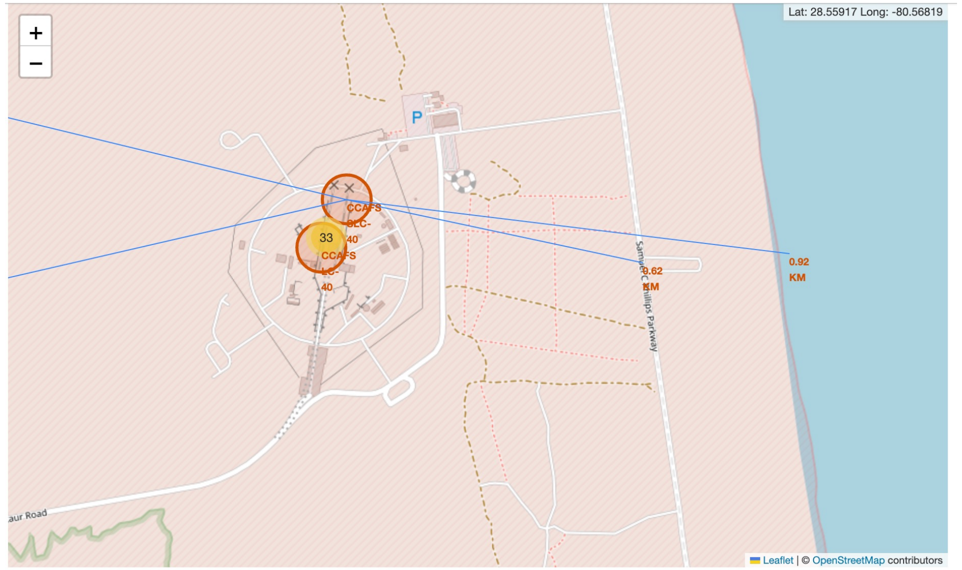


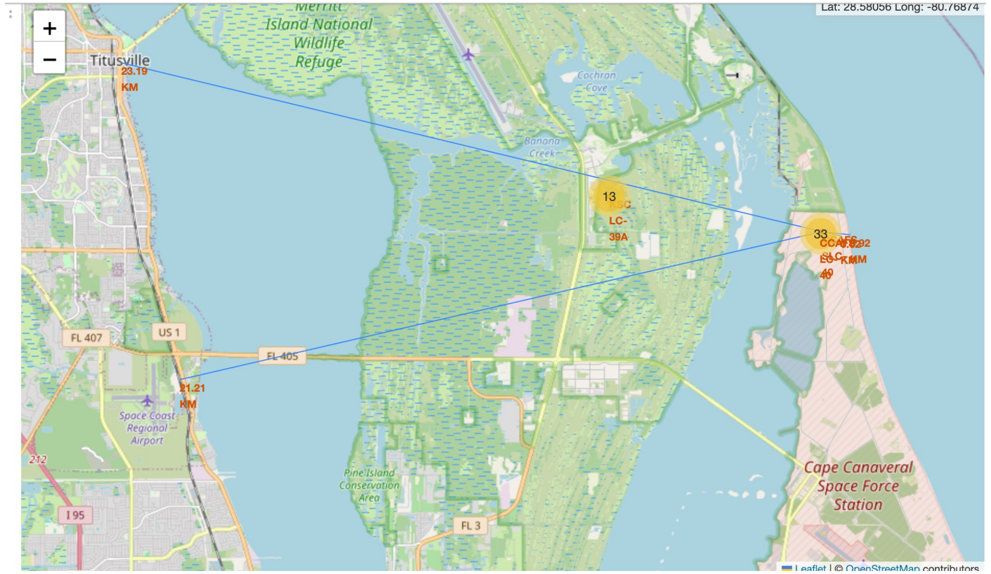
Color labels showing the outcome of a launch

The red marker indicates an unsuccessful launch while the green marker shows a successful launch

Launch site distance to proximities

- The images show the distance to proximities of a launch site



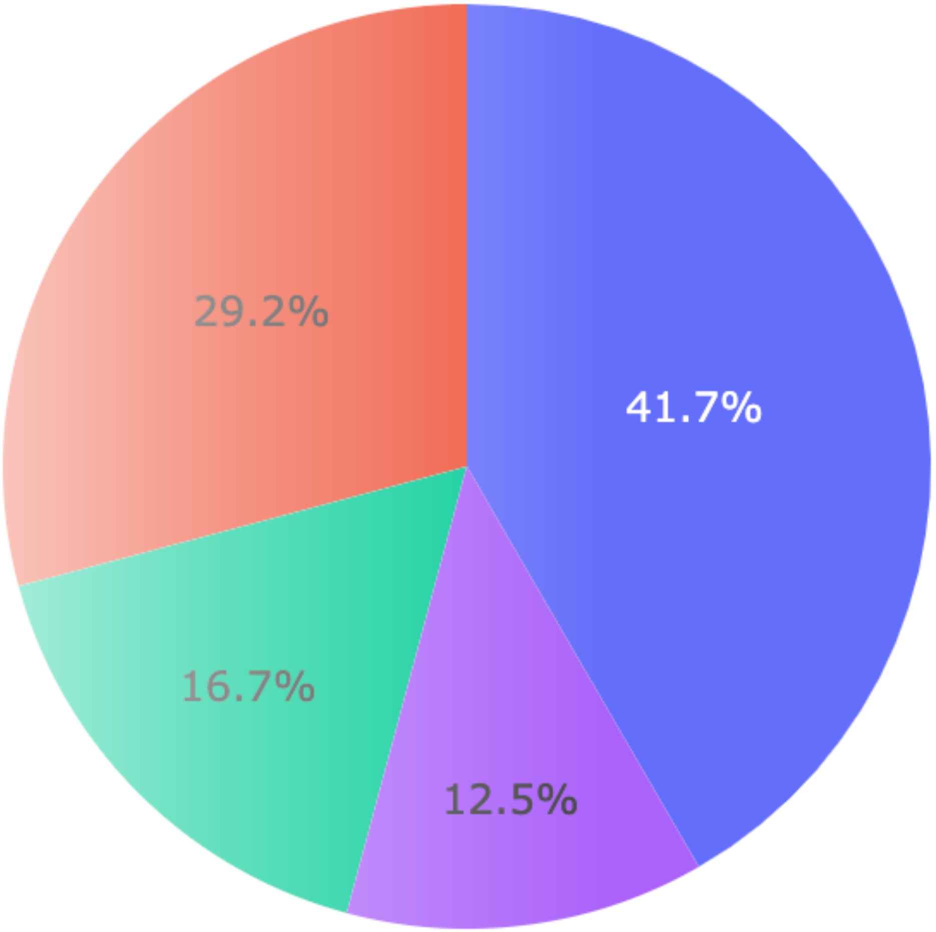


Section 4

Build a Dashboard with Plotly Dash

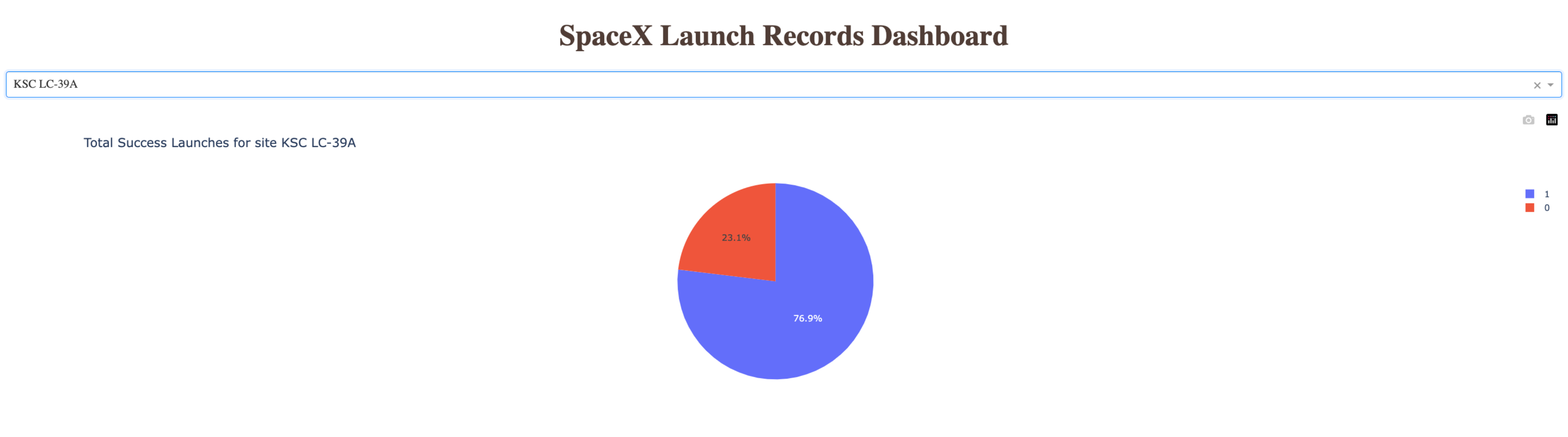
A pie Chart showing all the launch success for all sites

The success rate for all launch sites success with their keys



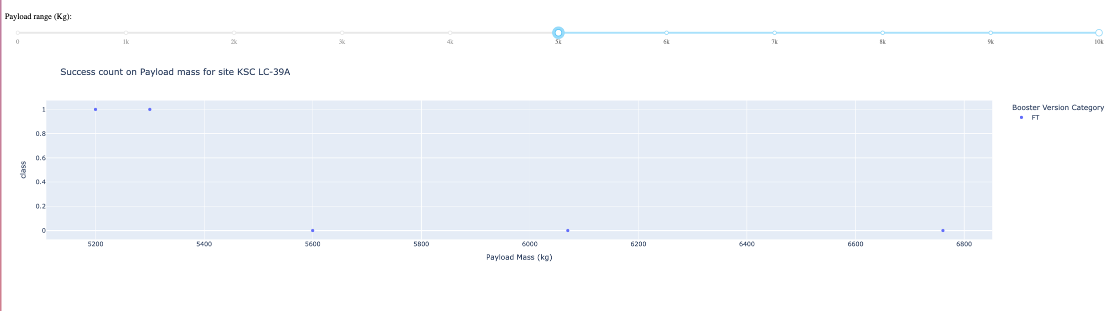
Pie Chart of the KSC LC-39 Launch site

This launch site has the highest success rate with 76.9% launch success



Payload Vs Launch Outcome scatter plot

The scatter plot shows that the greater the payload the less chances of success and vice visa

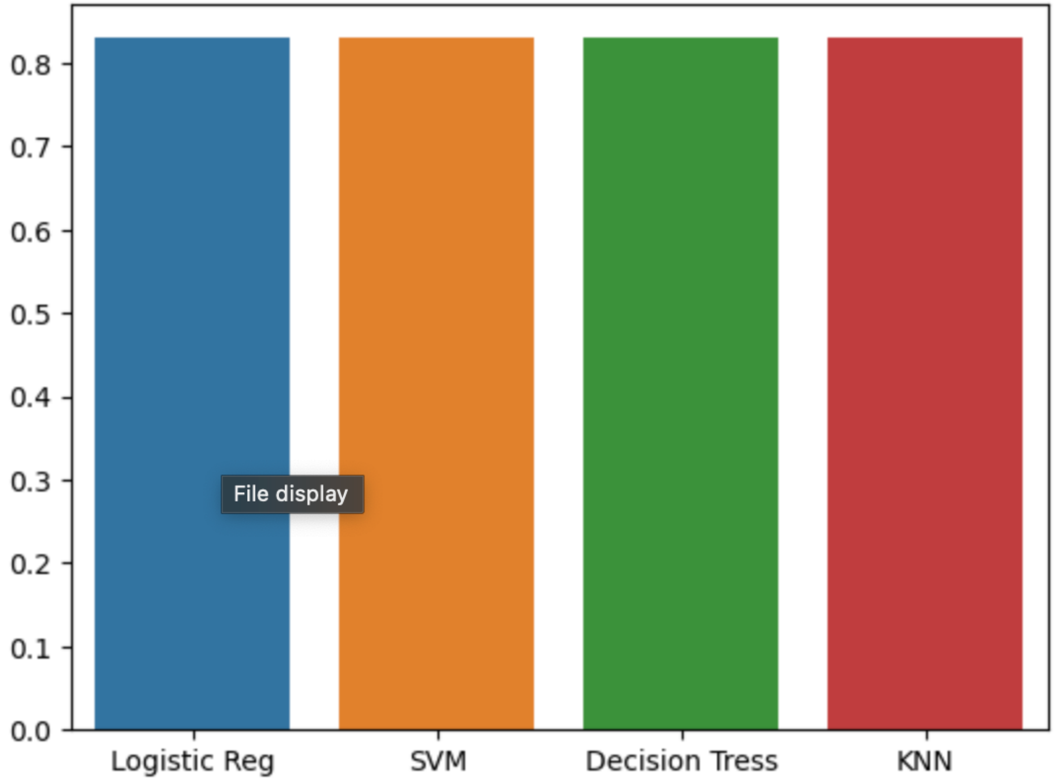


Section 5

Predictive Analysis (Classification)

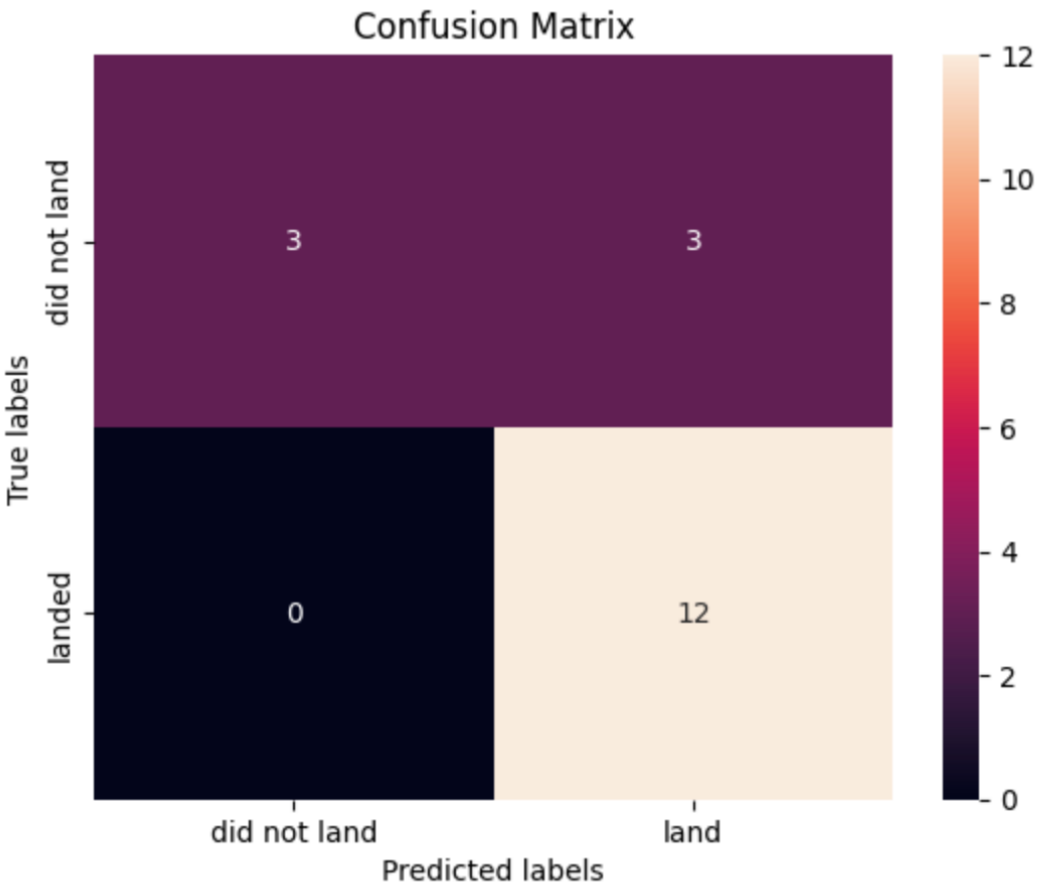
Classification Accuracy

- The bar chart above shows that all the classification Algorithm used in this project has same accuracy score.



Confusion Matrix

- Since the models has similar performance I arbitrary chose the KNN model which correctly classifies 15 out of 18 correctly.



Conclusions

- From the data we are able to understand that launch sites are far away from cities and closer to coastlines
- We can also understand that as the number of flights increases so does the success rate which means an improvement has been made
- The orbits that has the highest success rate are; ES-L1, GEO, HEO, SSO
- We can also conclude that the lesser the payload the more chances of success
- From the predictive analysis we can say that the models that we used are very good and has similar performance however we can get more data and/or try different preprocessing methods

Appendix

```python
# Import required libraries
import pandas as pd
import dash
import dash_html_components as html
import dash_core_components as dcc
from dash.dependencies import Input, Output
import plotly.express as px

# Read the airline data into pandas dataframe
spacex_df = pd.read_csv("spacex_launch_dash.csv")
max_payload = spacex_df['Payload Mass (kg)'].max()
min_payload = spacex_df['Payload Mass (kg)'].min()

# Create a dash application
app = dash.Dash(__name__)

# Create an app layout
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color': '#503D36',
                                               'font-size': 40}),
                                # TASK 1: Add a dropdown list to enable Launch Site selection
                                # The default select value is for ALL sites
                                #dcc.Dropdown(id='site-dropdown',...)
                                dcc.Dropdown(id="site-dropdown",
                                             options=[
                                                     {'label': 'All Sites', 'value': 'ALL'},
                                                     {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},
                                                     {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},
                                                     {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},
                                                     {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}
                                             ],
                                             value="ALL",
                                             placeholder="Select a Launch Site here",
                                             searchable=True
                                            ),
                                html.Br(),

                                # TASK 2: Add a pie chart to show the total successful launches count for all sites
```

Thank you!