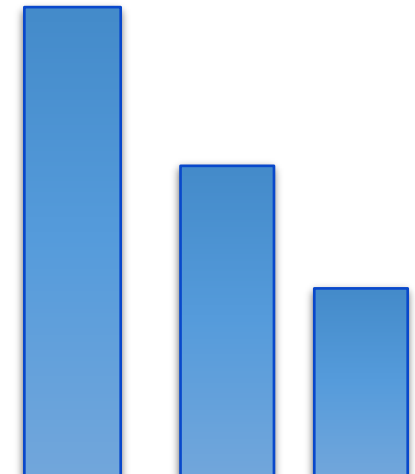
IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Swarnabha Mitra
18/10/2023

Outline

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



Executive Summary

- The world is rapidly developing and so is the technology around us, the recent changes in the space industry has been phenomenal and the harbinger of these company is SpaceX.
- The recent changes in the space industry is quiet spectacular and is always in need of some quality overview.
- SpaceX is a cutting edge company making us think of the possibilities of space travel and rocket based transportation system.
- The company is leaps ahead in cost cutting their cost of flights because of the ability of the SpaceX rockets to take a landing after being launched and it reduces costs dramatically with a rocket costing around $55 million only.
- We here are trying to discover the secrets of this landings performing so well by checking the metrics and strategies used by SpaceX and try to quantify their success in the field.

Introduction

The recent development in the Space industry has been remarkable and the leader of these private industries is SpaceX and have been the pioneers of space rocket manufacturing.

We are going to Analyze the data from SpaceX to predict the success of a flight along with the key factors or features which are responsible for the flight success and try to quantify a relation between different variables, like Booster Version, frequency of their launch, the Orbit of the rocket, etc.

We perform visualizations and explanatory analysis to find the dependencies of the features, normalize the parameters and perform different machine learning algorithms like support vector machine, logistic regression, K-Nearest Neighbour etc.

We found the sites at which the SpaceX launches took place and populated the data in a map, found out the factors affecting the success of the landing and trained multiple models to find out the one which works the best.

Section 1

Methodology

Methodology

Executive Summary

- Finding the reasons and secrets of SpaceX success and quantifying it
- Finding out metrics which helps in the landing.

Data collection methodology:

- Firstly data was extracted from the SpaceX API
- Retrieved more data from Wikipedia page tables to complete the data required.

Perform data wrangling

- A target column was setup depending whether the landing was successful or not.

Performed exploratory data analysis (EDA) using visualization and SQL

- Using Visualization tried to find the relations between various metrics, e.g. Payload Mass and Orbit
- Using SQL find the various types of boosters and the different types of landings.
- Also grouping the success and crashed into outcomes.

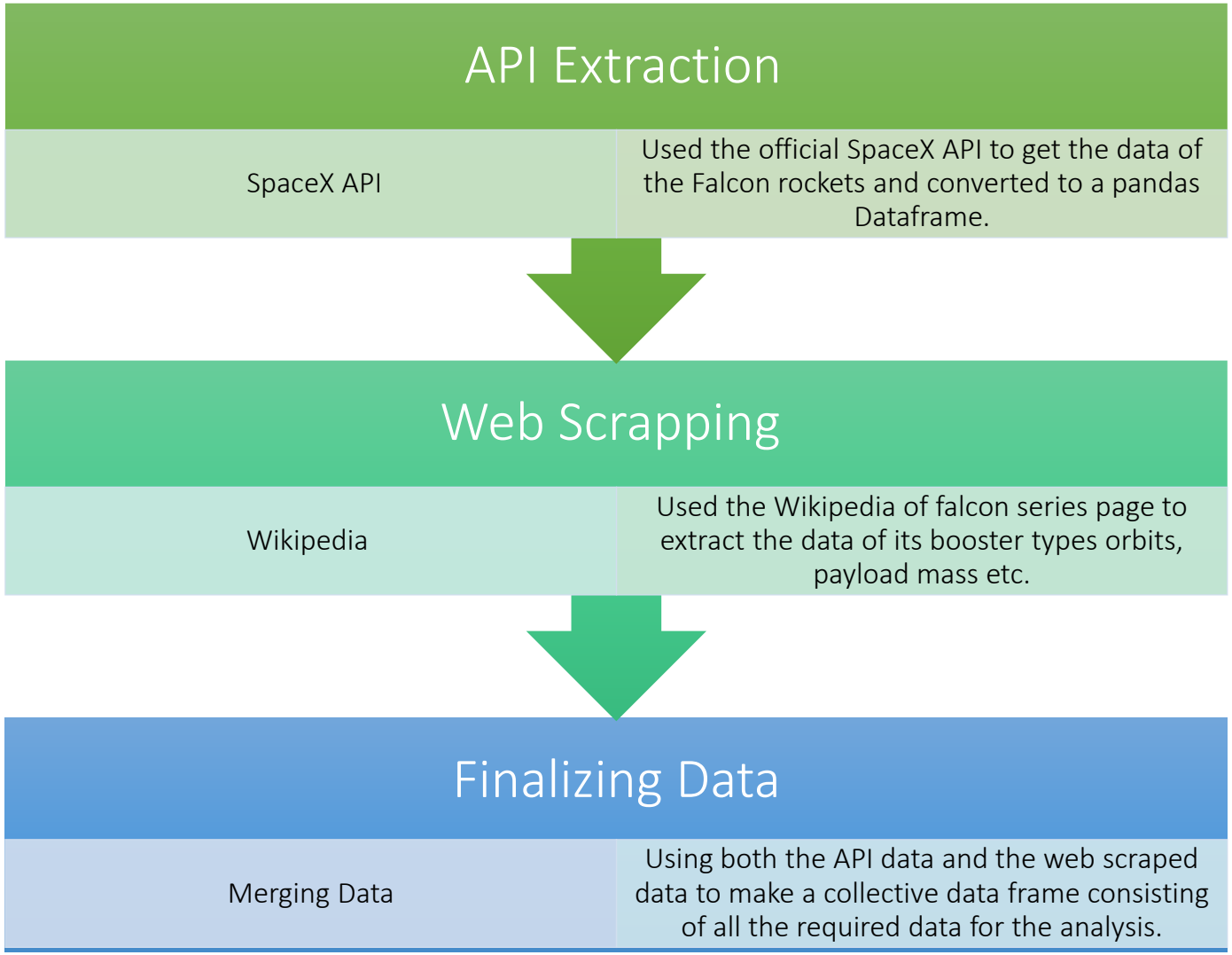
Performed interactive visual analytics using Folium and Plotly Dash

- Used Folium to check the data in an interactive map and check the successful launches.
- Used plotly to plot a dashboard to further check how the flights change with the change of payload.

Performed predictive analysis using classification models

- Created different models like KNN, SVM, logistic regression etc.
- Evaluated the success rates and plotted confusion matrix of different models.
- Decided the model based on the success rate.

Data Collection



Data Collection – SpaceX API

Using the SpaceX API by using the requests module to get the data from these URL :

https://api.spacexdata.com/v4/rockets/

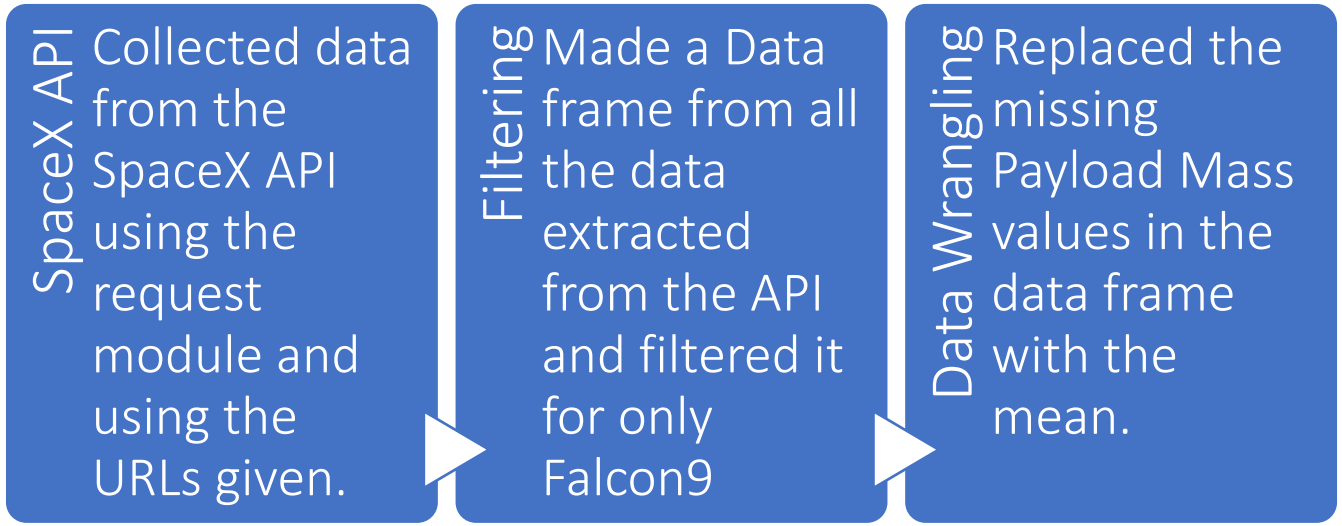
https://api.spacexdata.com/v4/launchpads/

https://api.spacexdata.com/v4/payloads/

https://api.spacexdata.com/v4/cores/

Then created different columns with the different data we extracted into a data frame.

https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

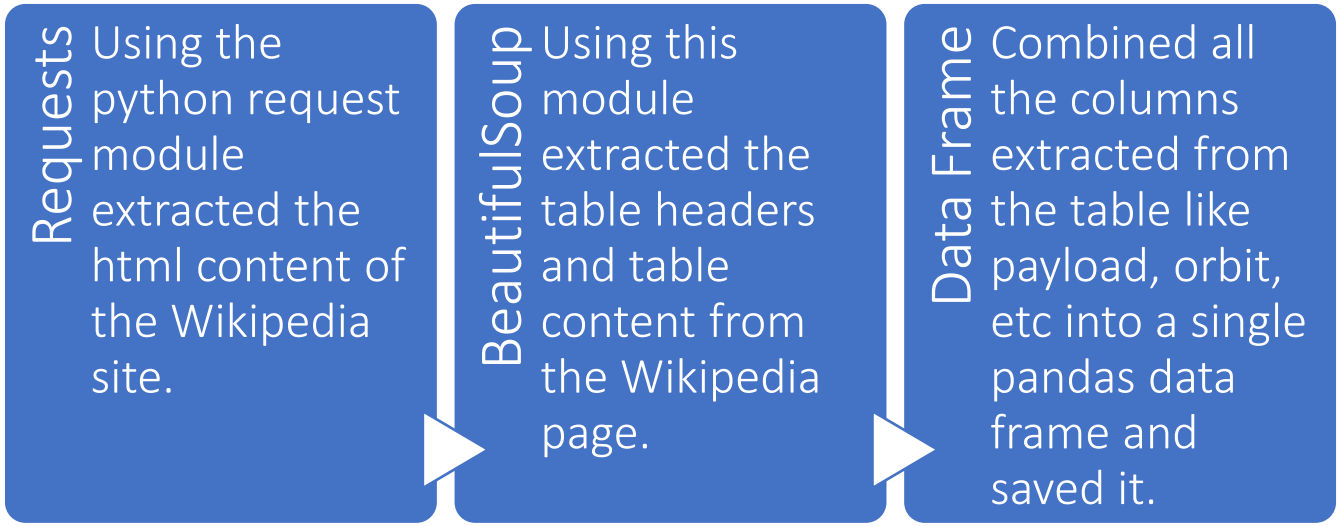


Data Collection - Scraping

- Using request module we parsed the content of the following site

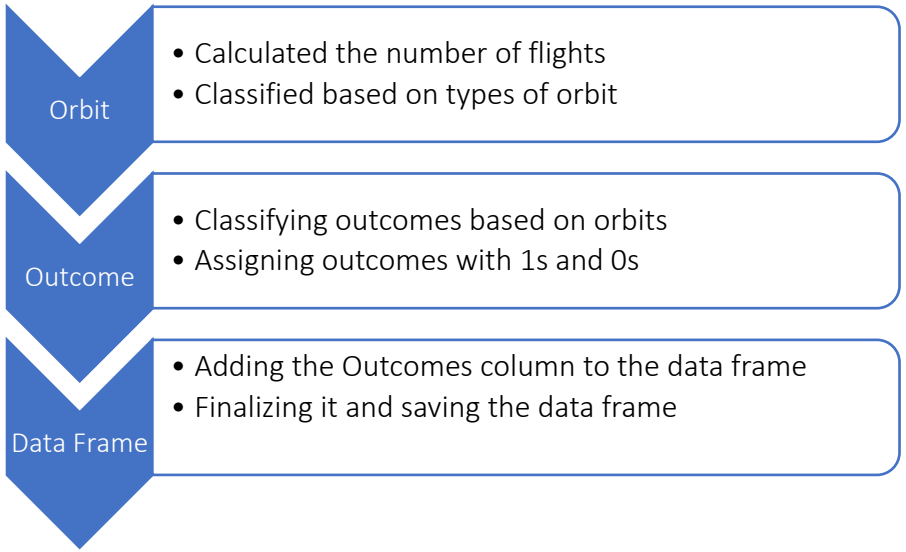
"https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

- Using Beautiful Soup Extracted the table data in the site and extracted data like flight number, payload, payload mass, etc.
- Converted the data into a pandas data frame and saved it to a file.
- GitHub Link : https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

- We used the data frame saved in during the data collection to check out the number of flights taking place in each orbit.
- We created a label called Outcome in the data frame and filled in the data with 1s and 0s, where 1 is marked for successful landings and 0 for unsuccessful landings.
- Then saved the pandas data frame into the same file.
- GitHub Link : https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- Firstly we plotted scatter plot graphs against Launch Site to check the factors affecting the Launch Site parameter.
  - Launch Site vs Flight Number
  - Launch Site vs Payload
  - Launch Site vs Payload Mass
- Secondly we plotted scatter plot graphs against Orbit to check the factors affecting the Orbit parameter,
  - Orbit vs Flight Number
  - Orbit vs Payload
- Trends
  - Plotted bar graph for the success of different Orbits using Class
  - Plotted Line graph to see the trend of yearly success by using Class and Date.

GitHub Link
: https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

- EDA Queries using SQL
    1. Listing Unique Launch Sites
    2. Figuring out the total payload carried by the NASA rockets
    3. The average payload mass of Falcon9 v1.1 for understanding the factor compared to the total and how many flights have been taken place.
    4. Listing the first successful landing for projecting the success in coming years
    5. Listing the suitable range of successful payload used for a range of 4000 to 6000 payload mass in kg.
    6. The maximum payload carried by a booster.
    7. Counting the number of failures and number of successful landings to track the rate of success.
    8. Ordering the number of success and failures in the time frame of 2010-06-04 and 2017-03-20
- GitHub Link : https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

Markers	Added markers on the test sites based on success and faliure and clustered them to see the number of flights per site.
Reasons	Marked green for successful landings Marked red for unsuccessful landings Clustered them into groups using a circle to show the number of flights per location Added lines to nearest roads and highways.
GitHub	https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

Pie chart

- Added a dropdown for the user to interact and see different pie charts comparing the success and failures.
- Also consisted of a pie chart showing all the different types of boosters used and their proportions.

Scatter Plot

- Added a slider for interacting with the payload mass so that we can see how the success rate of the payload mass varies with it being increased or decreased.
- This was achieved through a scatter plot showing the success as 1 and failures as 0 and used color/hue to show the different Booster Version Category found in the legend.

GitHub Link : https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

Features

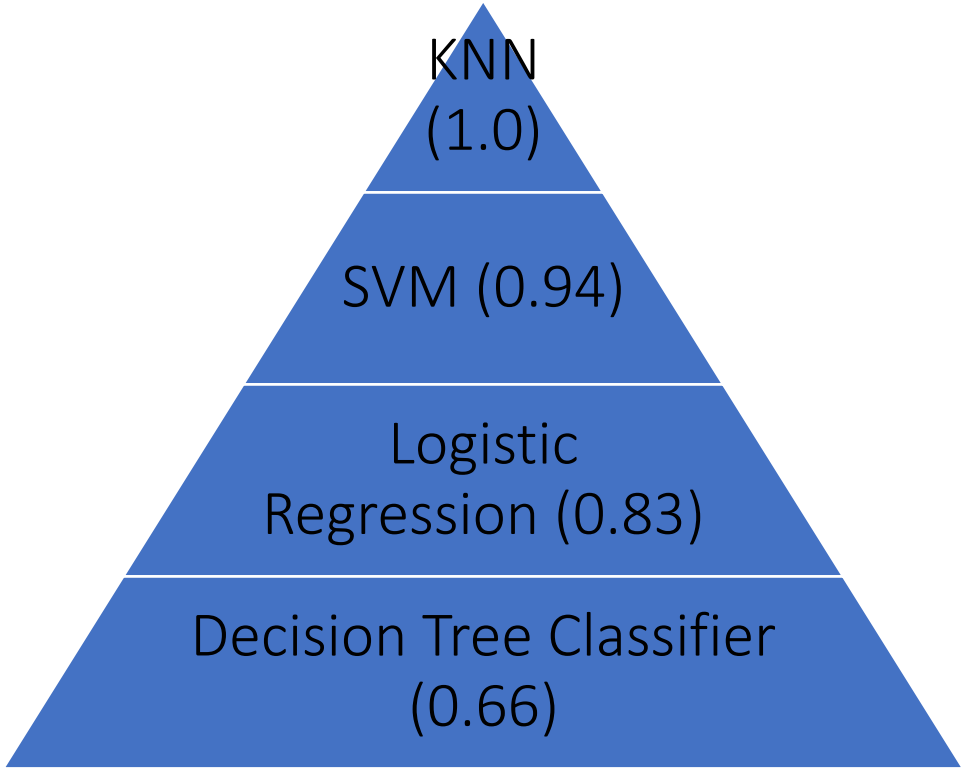
- Extracted the outcomes column and assigned it to Y
- Standardised features using Sklearn Standard Scaler

Data Split

- Used Sklearn train_test_split to split data
- Used GridSearchCV to tune hyperparameter

We fitted GridSearchCV with different machine learning models and evaluated the score of them based on their accuracy on the test data. The order is given as follows:

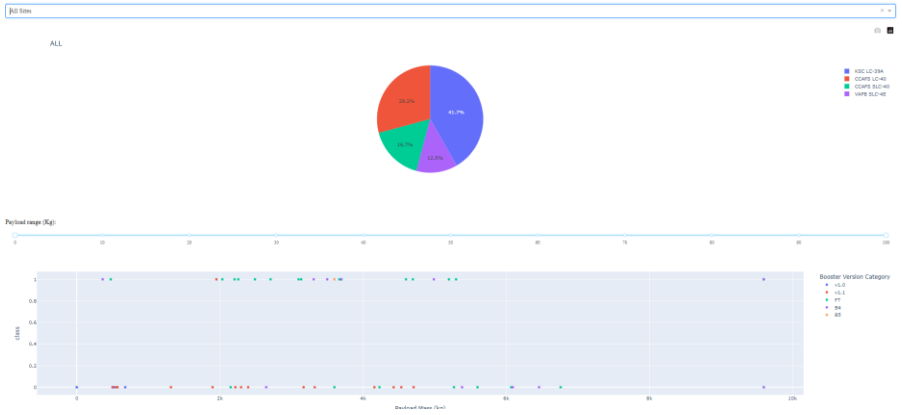
GitHub Link : https://github.com/DevSwarnabha/IBM-Data-Science-Capstone/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb



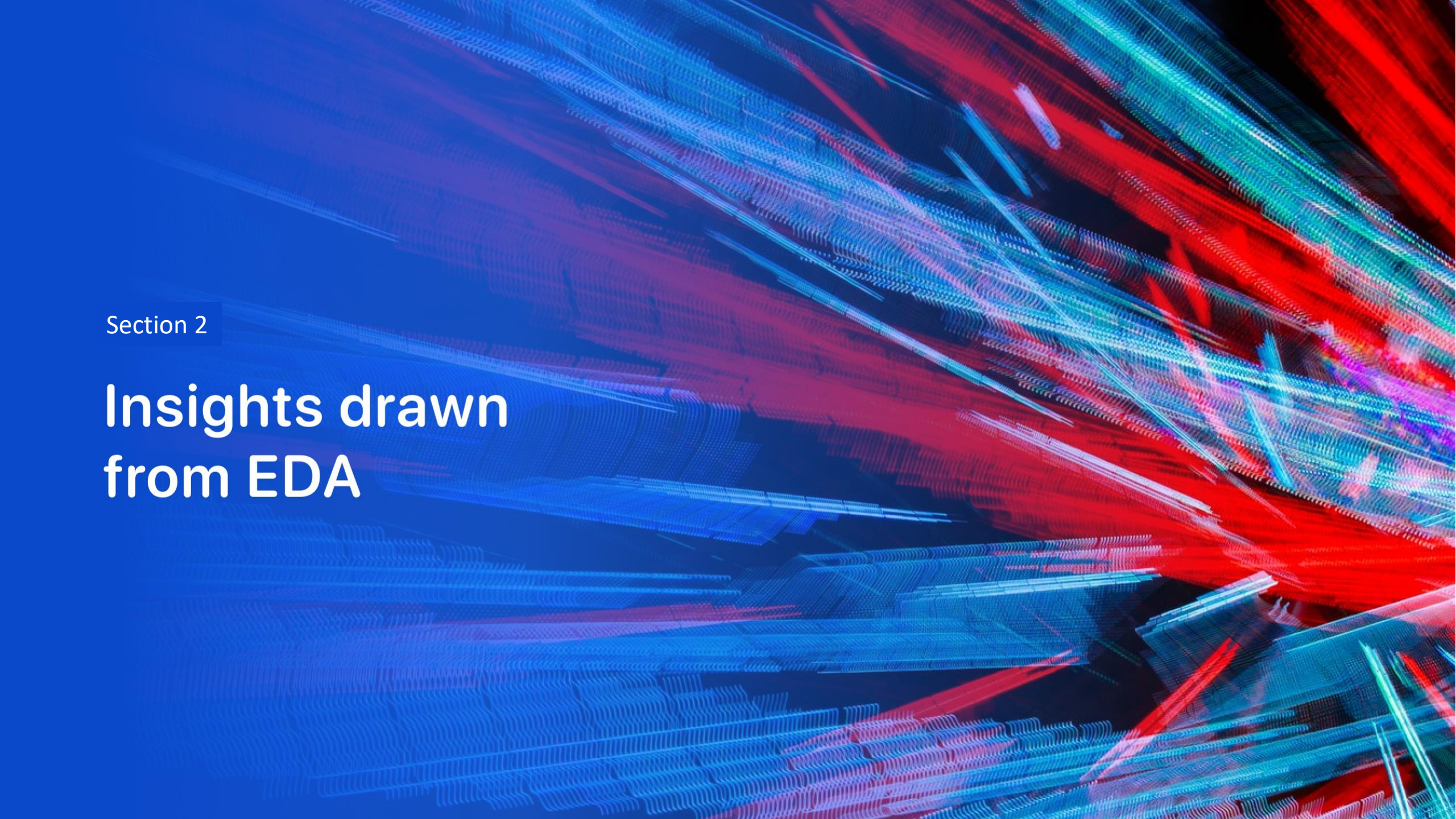
Results

- EDA :
  - ISS and VLEO orbit have high success rate after 50 flight number and shows a positive correlation with payload mass.
  - The success of flights have seen a positive trend of success rate over the years and a big jump of success at the 2015 but a dip at 2018.
  - 2015-12-22 was the date of the first successful landing
  - The total payload mass used by NASA is 45596 whereas the average payload mass is 2928.4 kgs.

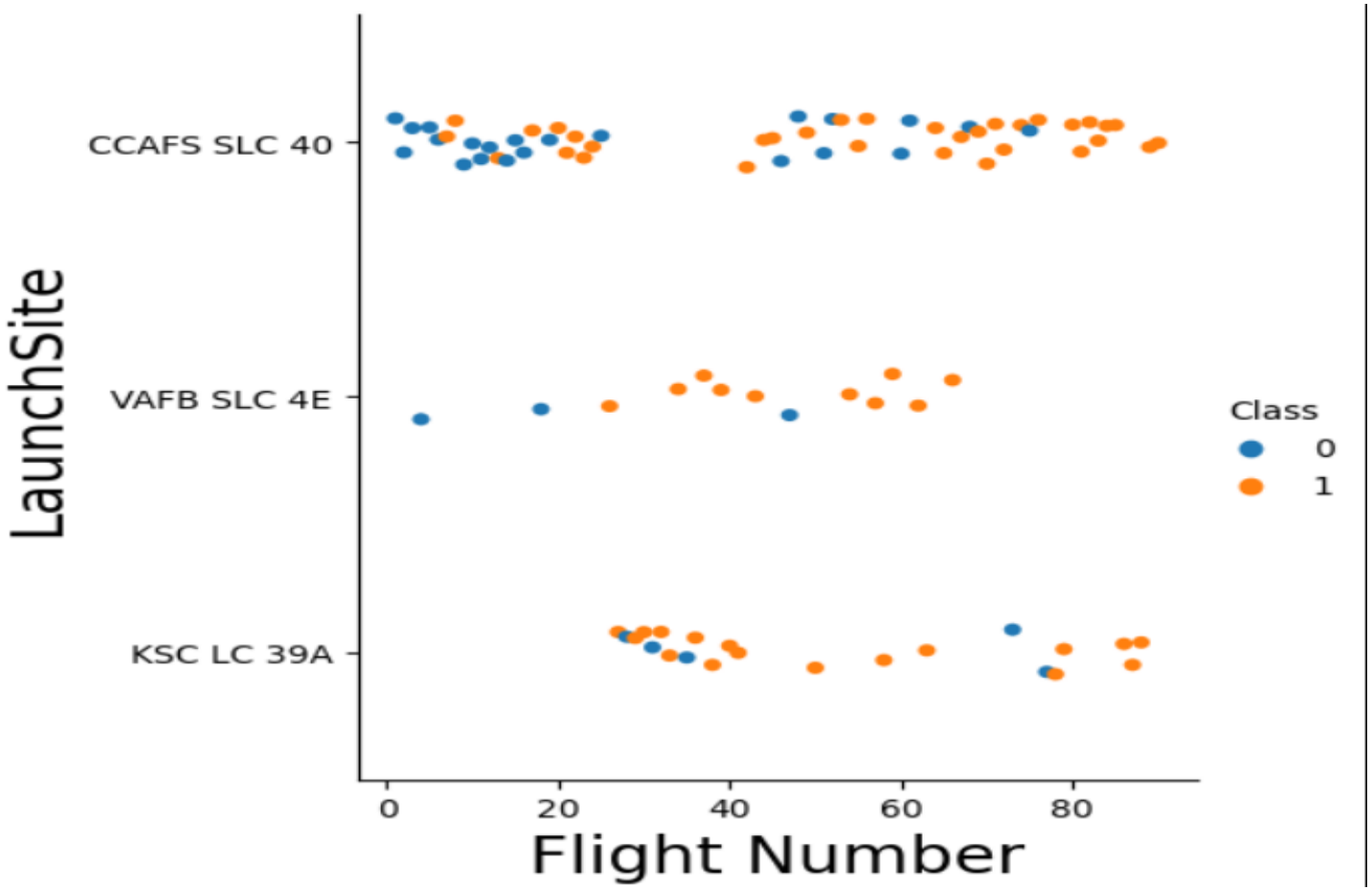
DASHBOARD :



- Predictive analysis :
  - The KNN model was found to work the best with a R^2 score of 1.0.
  - Followed by SVM (0.94), Logistic Regression (0.83) and finally Decision Tree Classifier with a score of 0.66

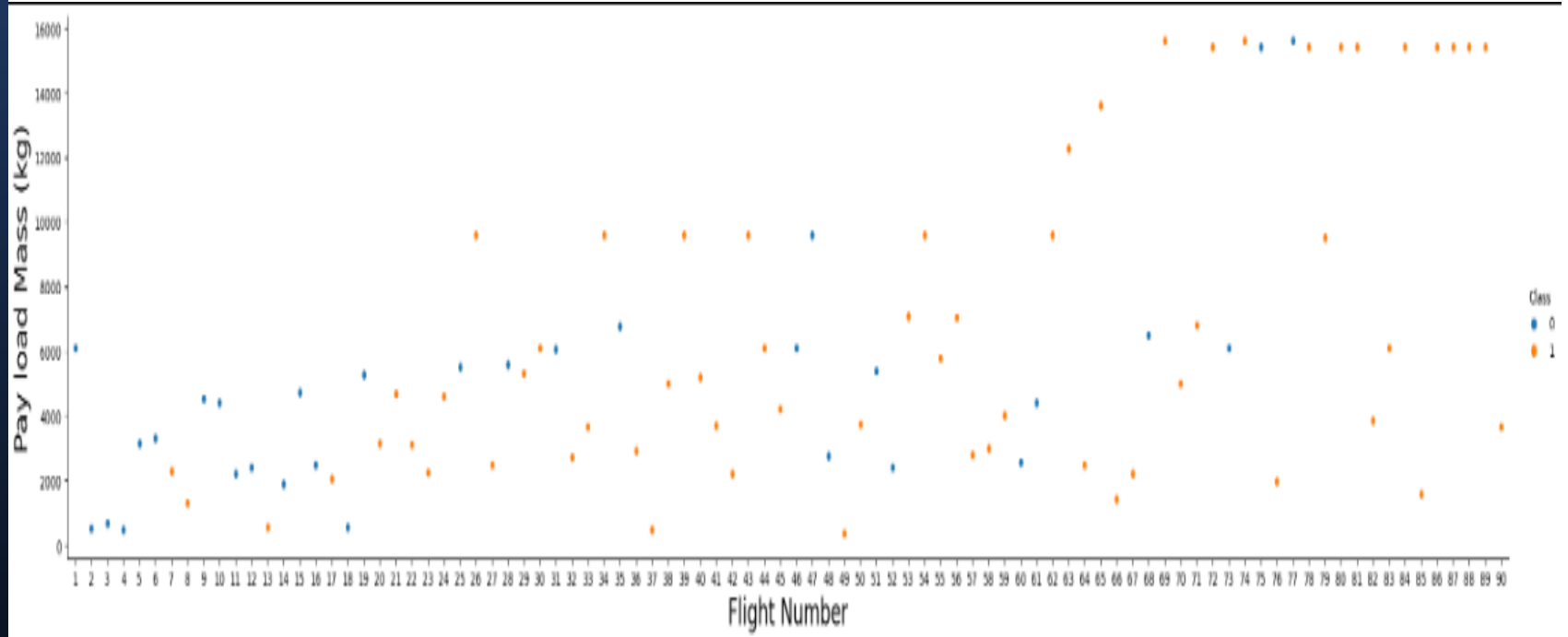

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



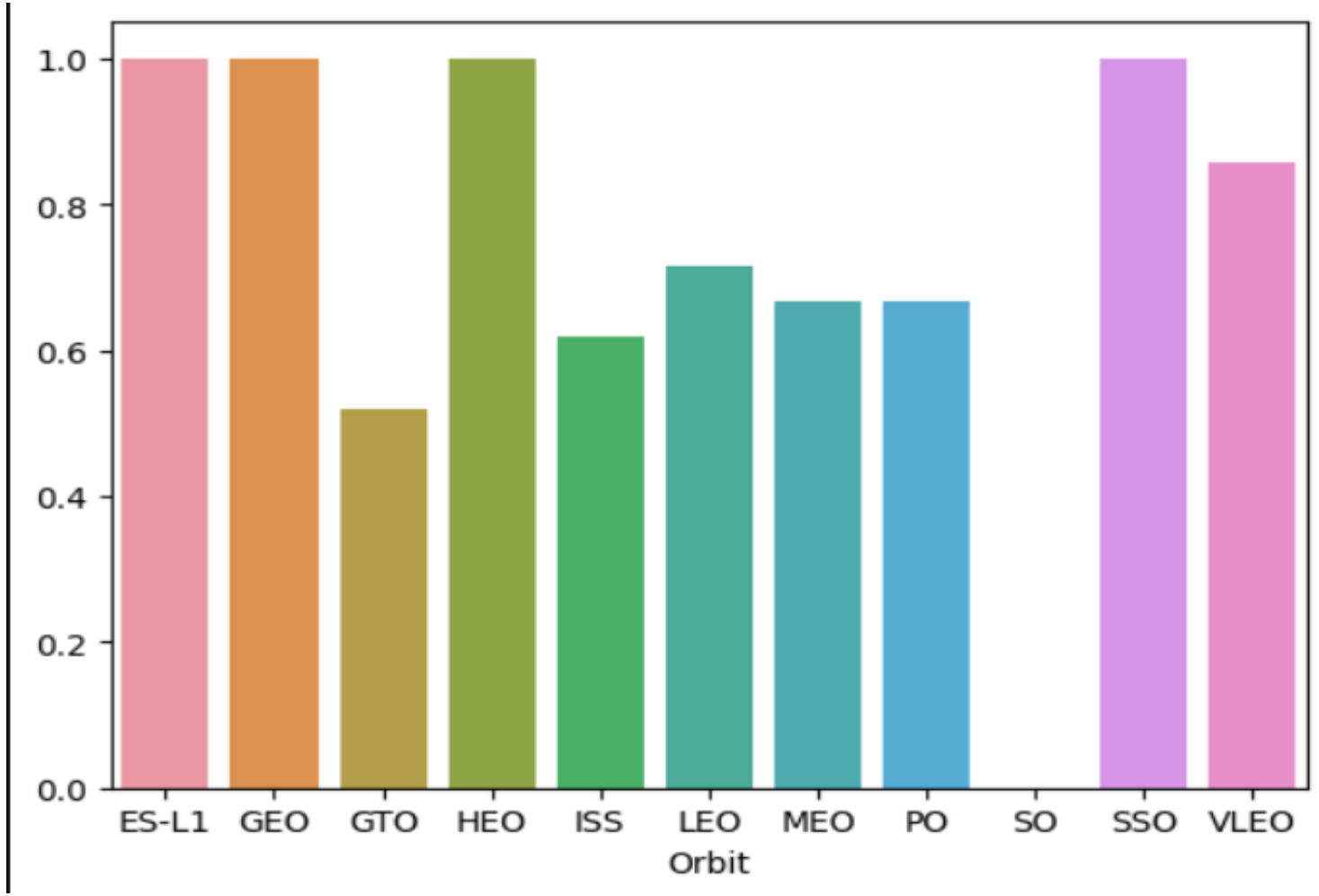
Plotted a scatter plot of the Launch Site vs Flight Number and used Class as Colour for the scatters to represent success or failure of the flights

Payload vs. Launch Site



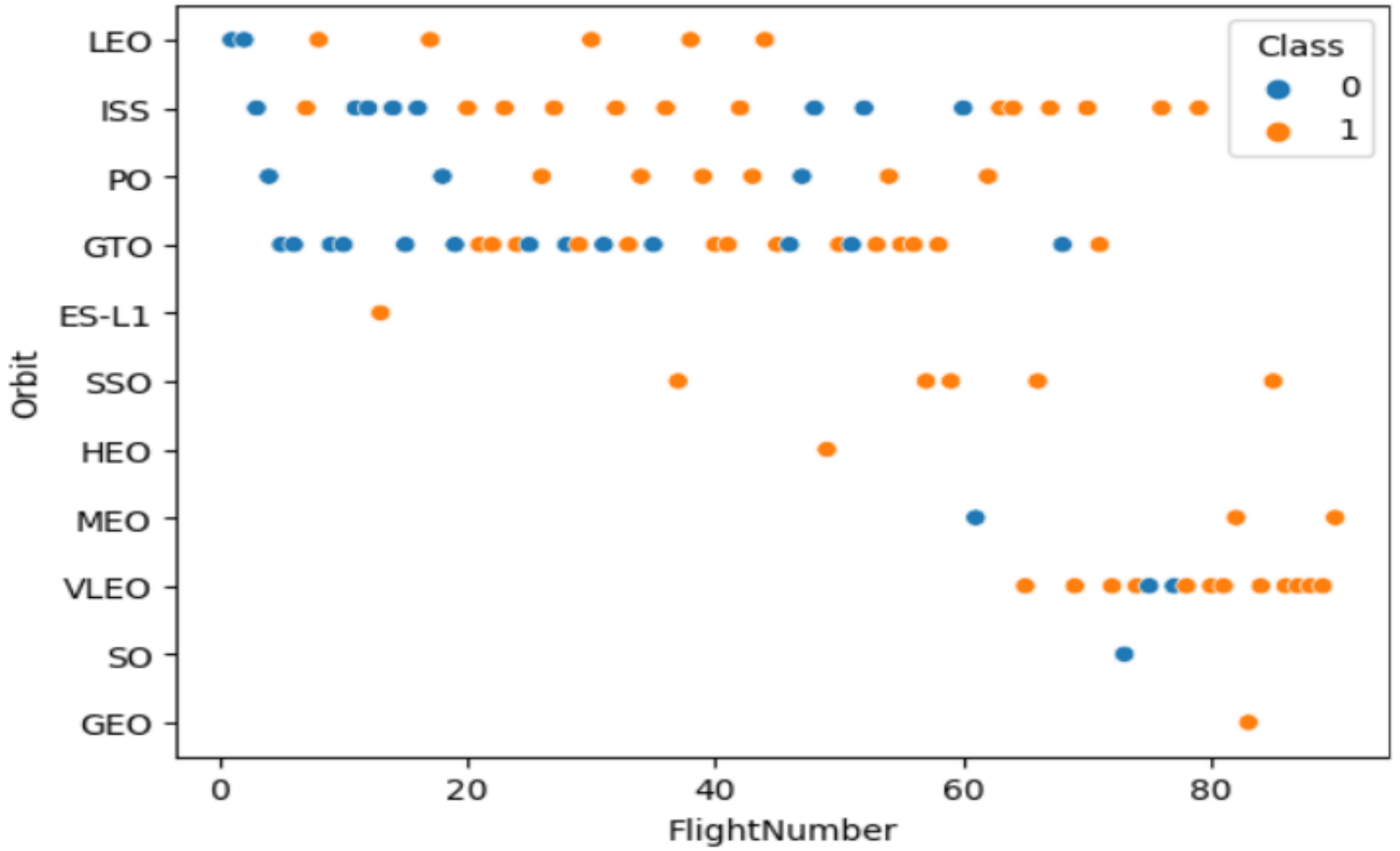
Plotted a scatter plot of the Payload Mass (kg) vs Flight Number and used Class as Colour for the scatters to represent success or failure of the flights.

Success Rate vs. Orbit Type



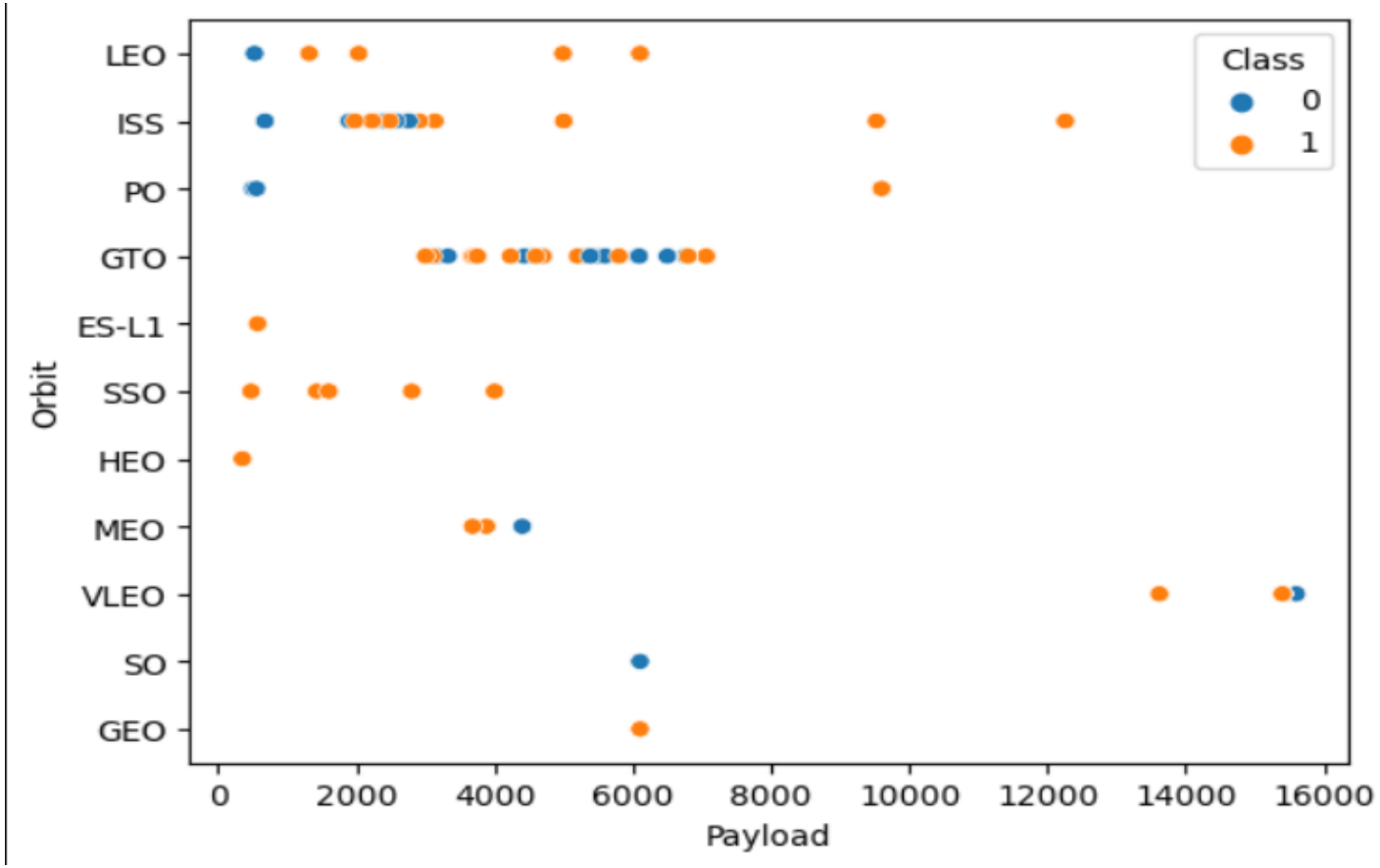
Plotted a bar plot of the Success Rate vs Orbit Type and used Class as Colour for the bars to represent success or failure of the flights of different Orbits.

Flight Number vs. Orbit Type



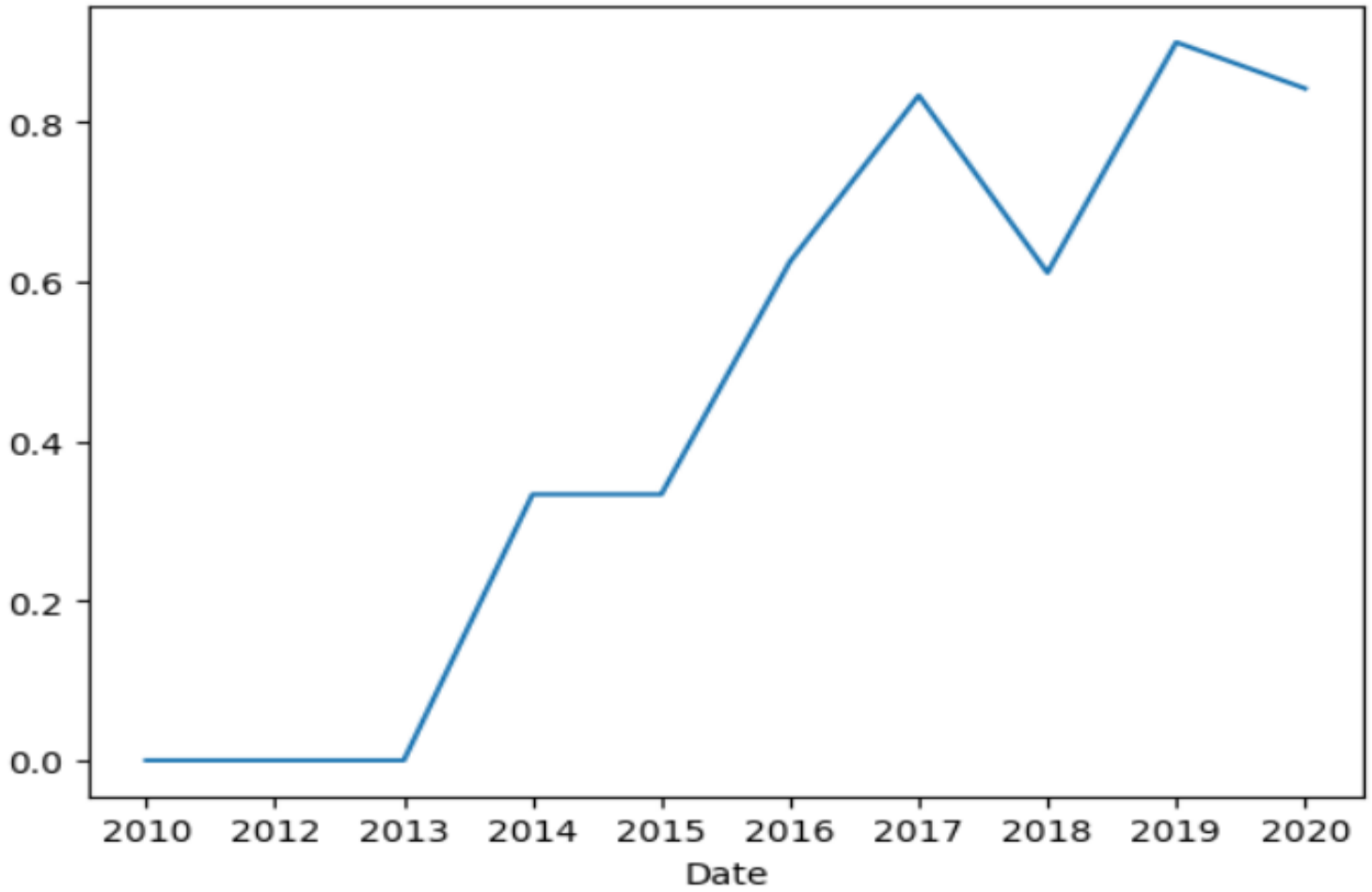
Plotted a scatter plot of the Flight Number vs Orbit Type and used Class as Colour for the scatters to represent success or failure of the flights.

Payload vs. Orbit Type



Plotted a scatter plot of the Orbit Type vs Payload Mass and used Class as Colour for the scatters to represent success or failure of the flights.

Launch Success Yearly Trend



Plotted a line chart plot of the Success Rate vs Date (Yearly) to show the positive growth over the years.

All Launch Site Names

- List of Unique Launch Site :

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

- Query :
  - %sql Select DISTINCT("Launch_Site") from SPACEXTABLE
- Explanation
  - We selected the "Launch Site" from the SPACEXTABLE
  - Used the DISTINCT function for unique values from the list

Launch Site Names Begin with 'CCA'

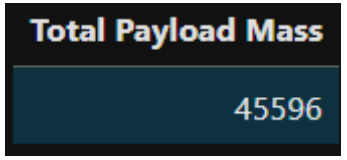
- Five Launch Sites that begin with CCA :

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- Query :
  - %sql Select * from SPACEXTABLE where "Launch_Site" like "CCA%" LIMIT 5;
- Explanation
  - Selected Launch Sites where the name is like CCA% where % is used for unknown following characters.

Total Payload Mass

- The total Payload Mass launched by NASA (CRS) :



Total Payload Mass
45596

- Query :
  - %sql Select SUM(PAYLOAD_MASS__KG_) "Total Payload Mass" FROM SPACEXTABLE WHERE CUSTOMER LIKE "NASA (CRS)"
- Explanation :
  - Selected Payload Mass from the SPACEXTABLE where the customers are NASA and used the SUM function to calculate the total payload.

Average Payload Mass by F9 v1.1

- Average Payload Mass of F9 v1.1

Average Payload Mass
2928.4

- Query :
  - %sql Select AVG(PAYLOAD_MASS__KG_) "Average Payload Mass" FROM SPACEXTABLE WHERE Booster_Version LIKE "F9 v1.1"

- Explanation :
  - Used the Payload Mass Kg column From the SPACEXTABLE where the booster was F9 v1.1
  - Used the AVG function on the column to find the Average Payload of the F9 v1.1

First Successful Ground Landing Date

- First Successful Ground Landing Date :

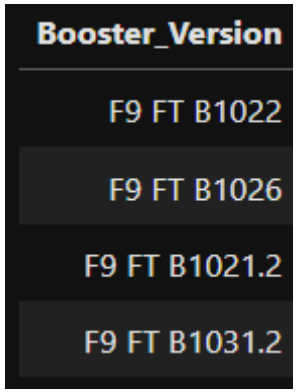
MIN(Date)
2015-12-22

- Query :
  - %sql Select MIN(Date) from SPACEXTABLE WHERE Landing_Outcome LIKE "Success (ground pad)"

- Explanation :
  - Selected the Date column from the SPACEXTABLE where the values are like Success (ground pad)
  - Used the MIN function to find the minimum value.

Successful Drone Ship Landing with Payload between 4000 and 6000

- Successful Drone Ship Landing with Payload between 4000 and 6000 :



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

- Query :
  - %sql SELECT Booster_Version from SPACEXTABLE WHERE Landing_Outcome LIKE "Success (drone ship)" AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
- Explanation :
  - Selected the Booster Version column from the SPACEXTABLE where the Landing Outcome value is Success (drone ship)
  - Also filtered the payload mass kg column to find the values for the range of 4000 and 6000

Total Number of Successful and Failure Mission Outcomes

- Total Number of Successful and Failure Mission Outcomes :

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

- Query :
    - %sql SELECT Distinct(Mission_Outcome),COUNT(Mission_Outcome) FROM SPACEXTABLE Group by Mission_Outcome
- Explanation :
    - Selected Mission Outcome and the Count of Mission Outcome From SPACEXTABLE
    - Grouped the Mission Outcome and used COUNT function to count how many flights took place.

Boosters Carried Maximum Payload

- Boosters Carried Maximum Payload :

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

- Query :
  - %sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
- Explanation :
  - Selected Booster Version Column from SPACEXTABLE
  - Where the Payload Mass is selected using a sub query that returns the maximum payload.

2015 Launch Records

- List of records of the failed drone ship launches taking place in 2015 :

substr(Date, 6,2)	Booster_Version	Launch_Site
10	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

- Query :
  - %sql SELECT substr(Date, 6,2), Booster_Version, Launch_Site FROM SPACEXTABLE WHERE Landing_Outcome like "Failure (drone ship)" and substr(Date,0, 5)='2015'
- Explanation :
  - Selected sub string of Date, Booster Version and Launch Site From SPACEXTABLE
  - Selected where the Landing Outcome is Failure (drone ship) and the substring(year) from the Date is 2015

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

- Ranking of Landing Outcomes Between 2010-06-04 and 2017-03-20 :

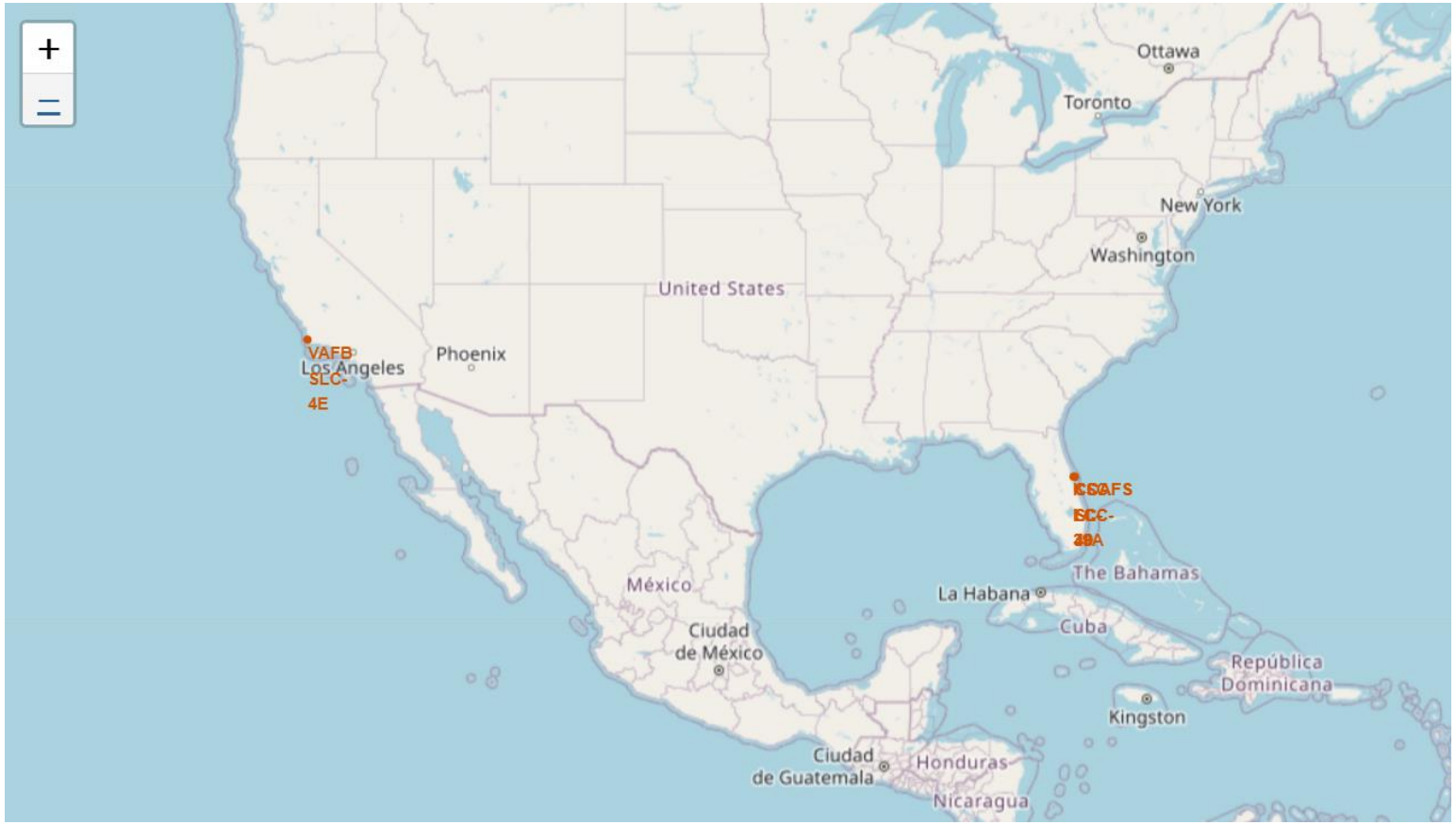
Landing_Outcome	COUNT(Landing_Outcome)
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

- Query :
  - %sql SELECT Landing_Outcome,COUNT(Landing_Outcome) FROM SPACEXTABLE WHERE "Date" BETWEEN "2010-06-04" and "2017-03-20" GROUP BY Landing_Outcome ORDER BY COUNT(Landing_Outcome) DESC
- Explanation :
  - Selected Landing Outcome, number of Landing Outcome From SPACEXTABLE
  - Selected between 2010-06-04 to 2017-03-20 Dates
  - Grouped by Landing Outcomes and ordered it by the count of landing outcomes in descending order.

Section 3

Launch Sites Proximities Analysis

Launch Sites

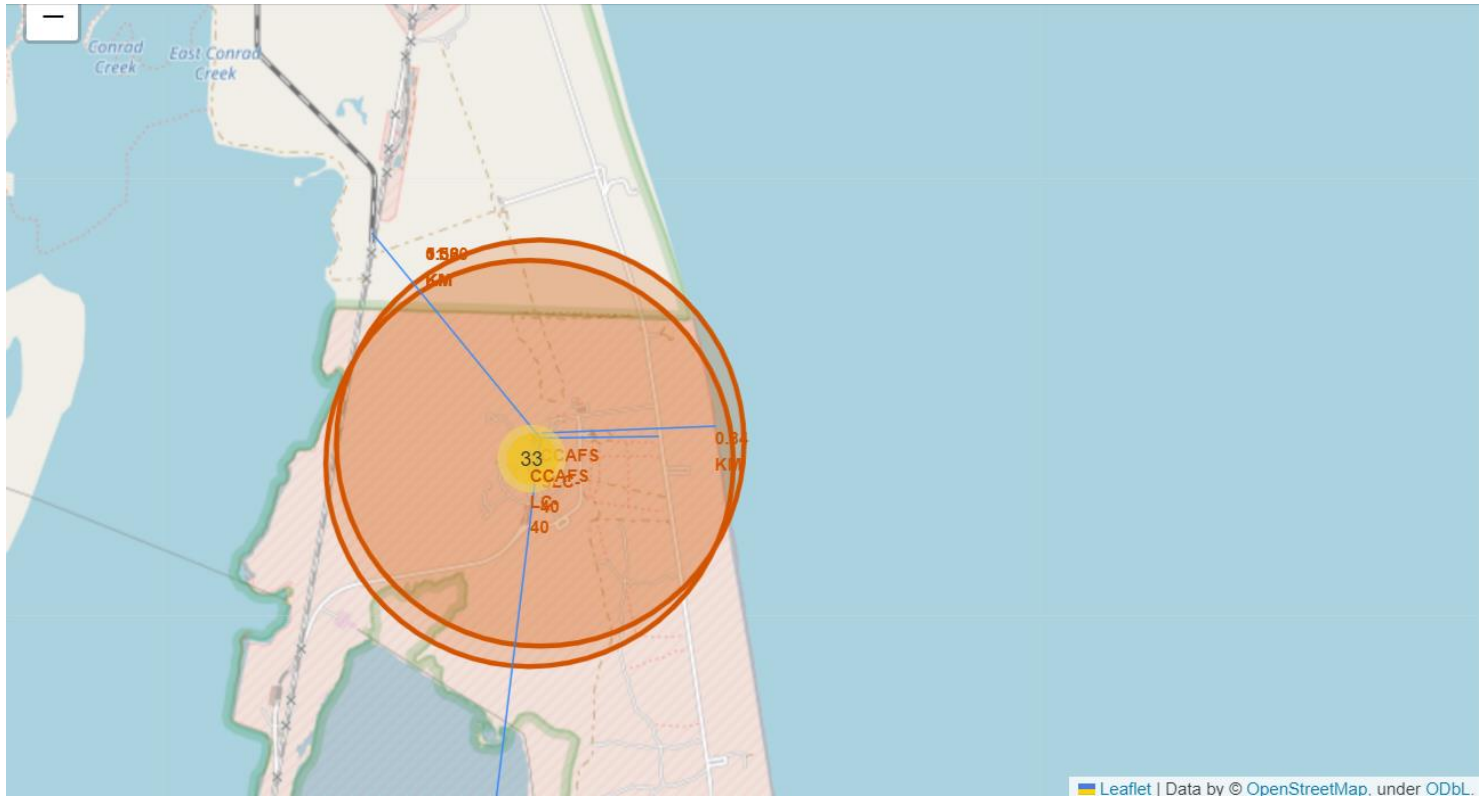


We found the different launch sites for the rockets that we have marked on the map, we can see that the launches have taken place in Los Angeles and Florida.

Launch Clustering and Classification

Clustered all the launches based on their launch locations.
Marked green markers for successful launches and red for unsuccessful ones.

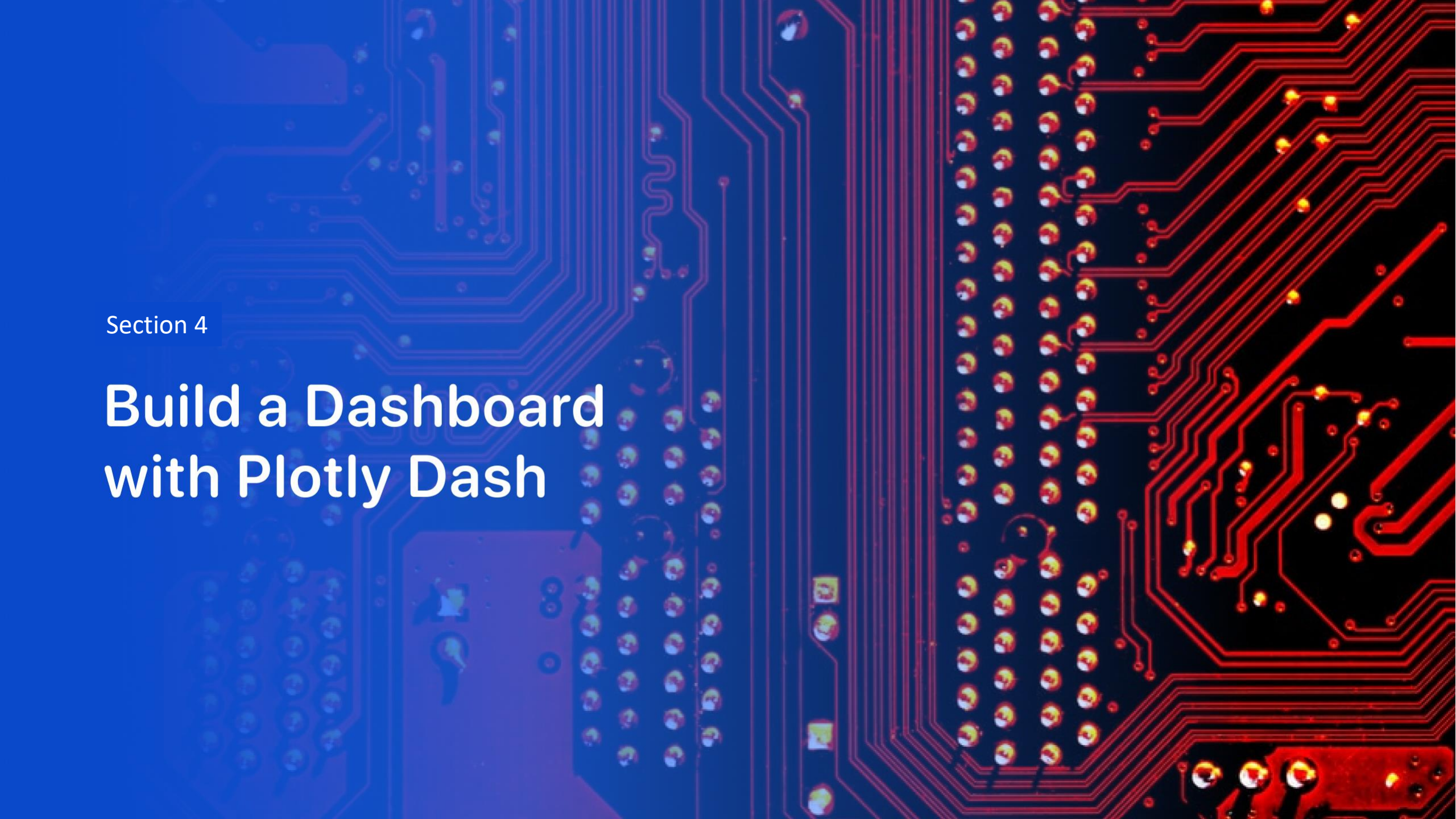
Significant landmarks



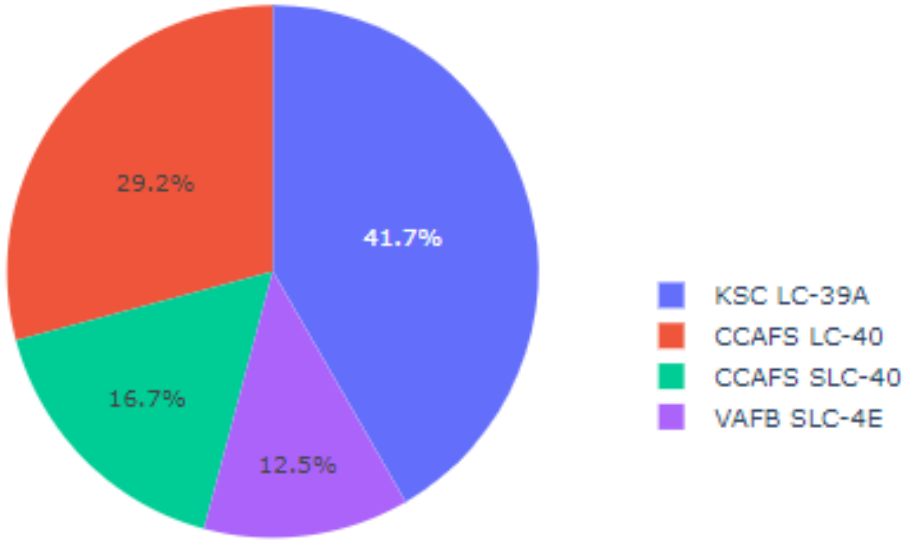
Marked the following landmarks :

- Railway
- Highway
- Coastline
- City

Found the proximity of these landmarks and how suitable the site is based on that.


Section 4
Build a Dashboard with Plotly Dash

Usage of Launch Sites

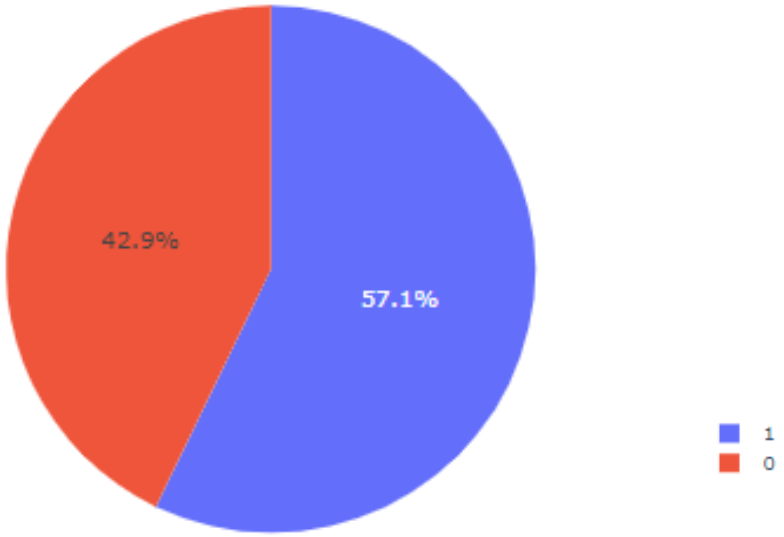


ALL

The KSC LC-39A has been the most used launch site with a strong 41.7% usage, whereas the VAFB SLC-4E has been used the least with only 12.5%.

The VAFB SLC-4E and CCAFS SLC-40 requires more launches for better overview over the types of the location and their advantages

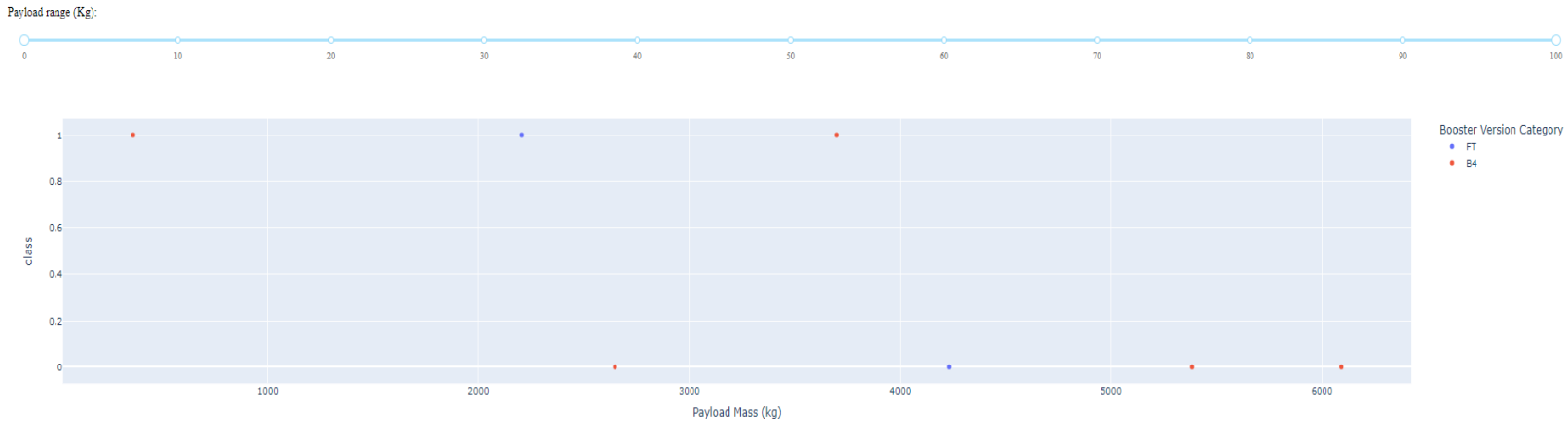
CCAFS SLC-40 success rate



CCAFS SLC-40

The CCAFS SLC-40 has a low percentage of usage over the other launch sites, only 16.7% but it is highly effective in terms of success rate, we can clearly see that it succeeds 57.1% time which is 17.1% better than its closest competitor the VAFB SLC-4E at 40%, we can see a clear trend of SLC locations working better than their other counterpart.

Ideal range of Payload Mass for CCAFS SLC-40

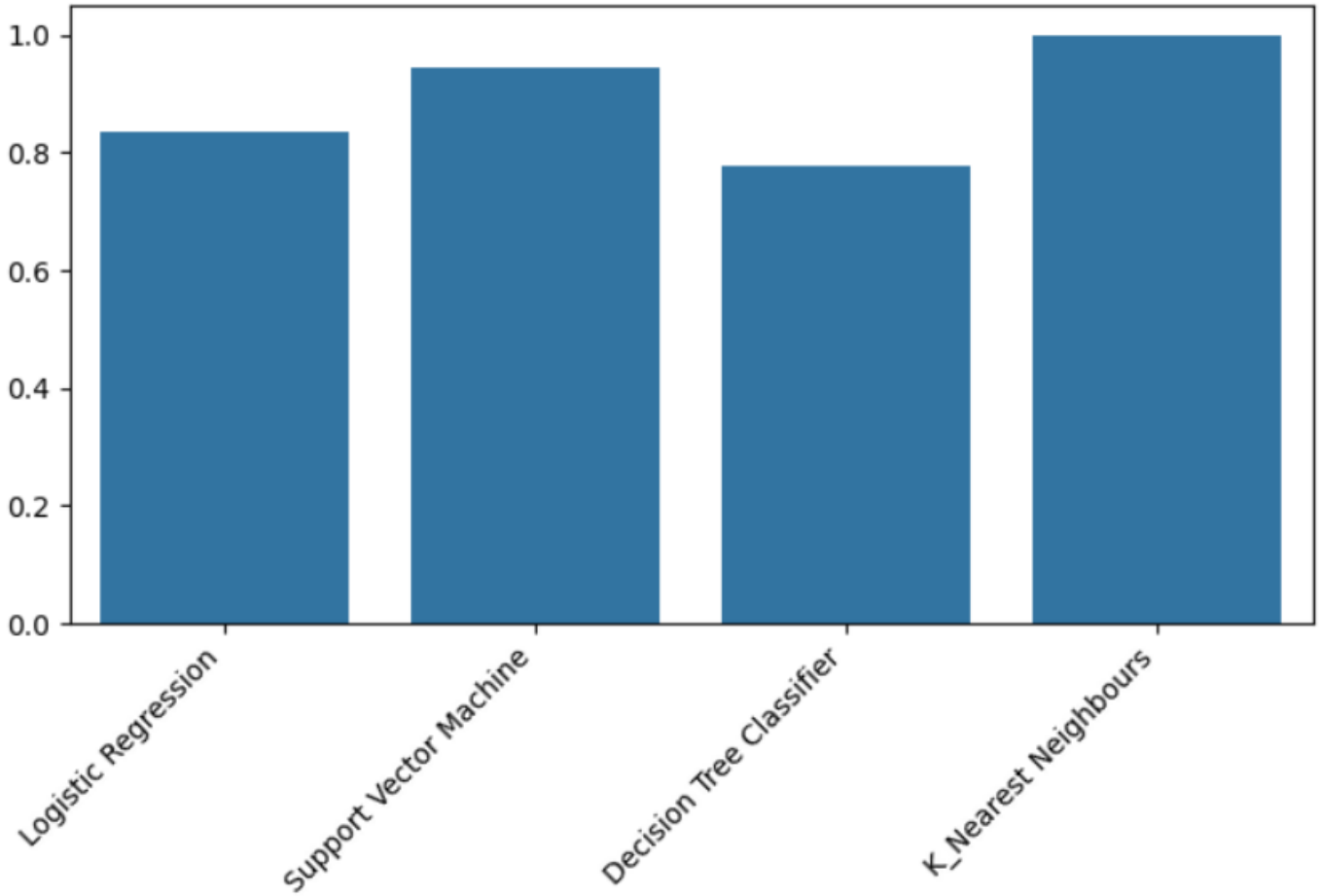


The CCAFS SLC-40 has the highest success rate out of all the other location, the CCAFS SLC-40 is significantly successful in low to medium range of payload mass whereas it starts failing for heavier payloads after 4,000 kg of payload mass. This location is ideal for low and quick secured launches with a significant chance of success. For heavier launches KSC LC-39A which has a good record for payloads under 5,500 kg and anything heavier can be used by VAFB SLC – 4E which has shown promise for very heavy payloads, upwards of almost 10,000 kg of payload mass.

Section 5

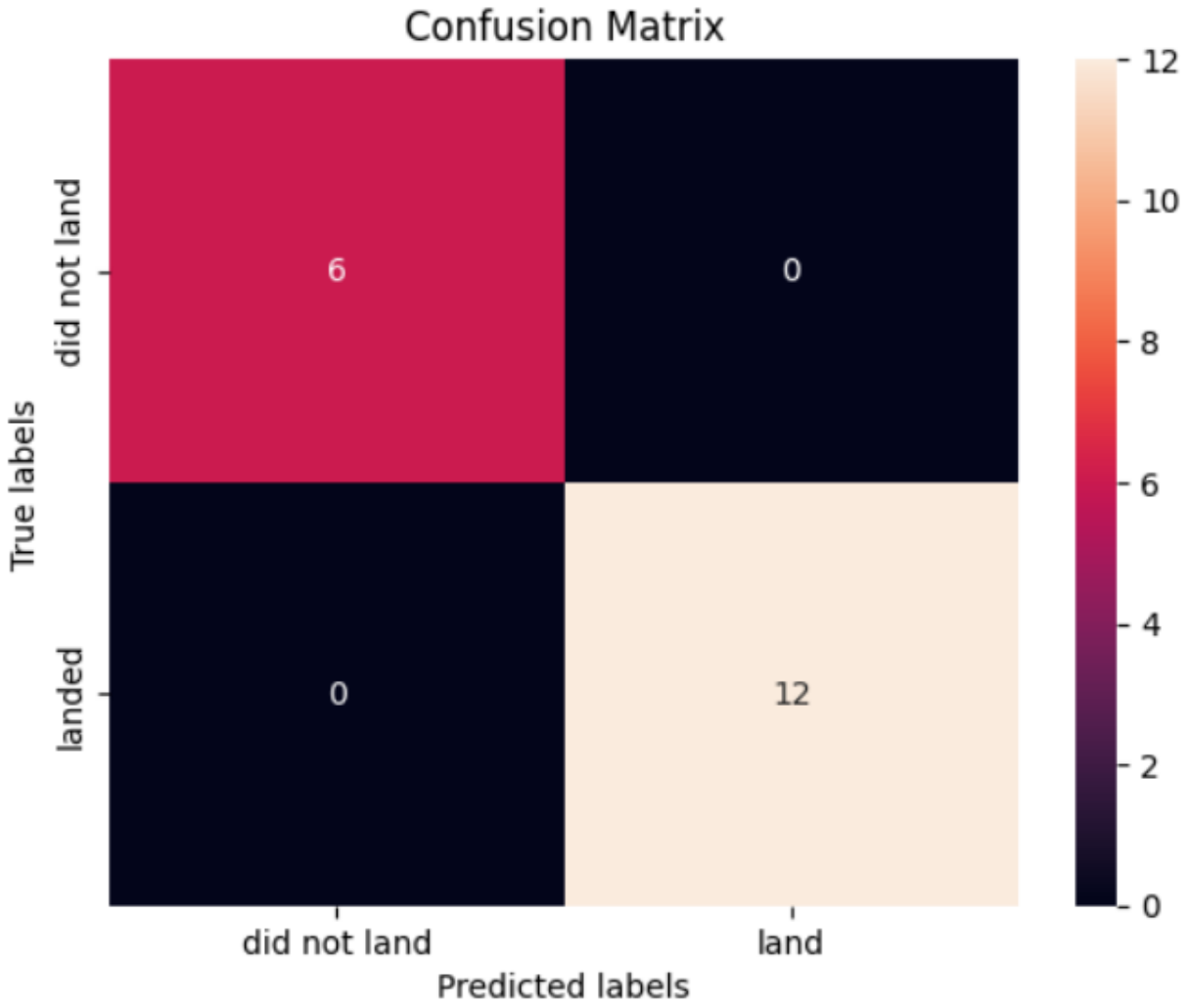
Predictive Analysis (Classification)

Classification Accuracy



From the bar chart we can easily see, the K_Nearest Neighbour algorithm does the best in this classification with a perfect score of 1.0.

Confusion Matrix



We can clearly see that the KNN algorithm had 6 true positives, 12 true negatives and 0 false positive and false negative, meaning it classified all the data perfectly.

Conclusions

- The success of payload mass has significantly improved with the increase in number of flights.
- The success rate of launches has increased after 2013 and a significant increase after 2015.
- Average of around payload mass 3,000 kg whereas around 10,000 kg is used for a flight.
- Suited the locations by checking its connectivity with road, water and city.
- The launch site CCAFS SLC-40 is the best performing with a 57.1% success rate.
- The K-Nearest Neighbour algorithm is the best performing with 6 true positives, 12 true negatives and 0 false positive and false negative.

Appendix

```
# 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-4E0', '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
                                # If a specific launch site was selected, show the Success vs. Failed counts for the site
                                html.Div(dcc.Graph(id='success-pie-chart')),
                                html.Br(),

                                html.P("Payload range (Kg):"),
                                # TASK 3: Add a slider to select payload range
                                #dcc.RangeSlider(id='payload-slider',...)
```

Appendix

```
html.P("Payload range (Kg):"),
# TASK 3: Add a slider to select payload range
#dcc.RangeSlider(id='payload-slider',...)

dcc.RangeSlider(id='payload-slider',
    min=0, max=10000, step=1000,
    marks={0: '0',
        1000 : '10',
        2000 : '20',
        3000 : '30',
        4000 : '40',
        5000 : '50',
        6000 : '60',
        7000 : '70',
        8000 : '80',
        9000 : '90',
        10000: '100'},
    value=[min_payload, max_payload]),

# TASK 4: Add a scatter chart to show the correlation between payload and launch success
html.Div(dcc.Graph(id='success-payload-scatter-chart')),
])
```

Appendix

```python
# TASK 2:
# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output
# Function decorator to specify function input and output
@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))
def get_pie_chart(entered_site):
    filtered_df = spacex_df
    if entered_site == 'ALL':
        fig = px.pie(filtered_df.groupby("Launch Site").sum(), values='class',
        names=filtered_df["Launch Site"].unique(),
        title="ALL")
    else:
        filtered_df = spacex_df.loc[spacex_df["Launch Site"] == entered_site]
        fig = px.pie(filtered_df.groupby("class").count(), values='Flight Number',
        names= filtered_df["class"].unique(),
        title=entered_site)

    return fig
        # return the outcomes piechart for a selected site
# TASK 4:
# Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output

@app.callback(Output(component_id='success-payload-scatter-chart', component_property='figure'),
              [Input(component_id='site-dropdown', component_property='value'),
              Input(component_id="payload-slider", component_property="value")])
def get_scatter(entered_site, payload):
    if entered_site == "ALL":
        filtered_df = spacex_df
        filtered_df = filtered_df.loc[((filtered_df["Payload Mass (kg)"] >= payload[0]) & (filtered_df["Payload Mass (kg)"] <= payload[1]))]
        fig = px.scatter(filtered_df, x = "Payload Mass (kg)",y = "class", color = "Booster Version Category")

    else:
        filtered_df = spacex_df.loc[spacex_df["Launch Site"] == entered_site]
        filtered_df = filtered_df.loc[ ( (filtered_df["Payload Mass (kg)"] >= payload[0]) & (filtered_df["Payload Mass (kg)"] <= payload[1]))]
        fig = px.scatter(filtered_df, x = "Payload Mass (kg)",y = "class", color = "Booster Version Category")

    return fig



# Run the app
if __name__ == '__main__':
    app.run_server()
```

Thank you!