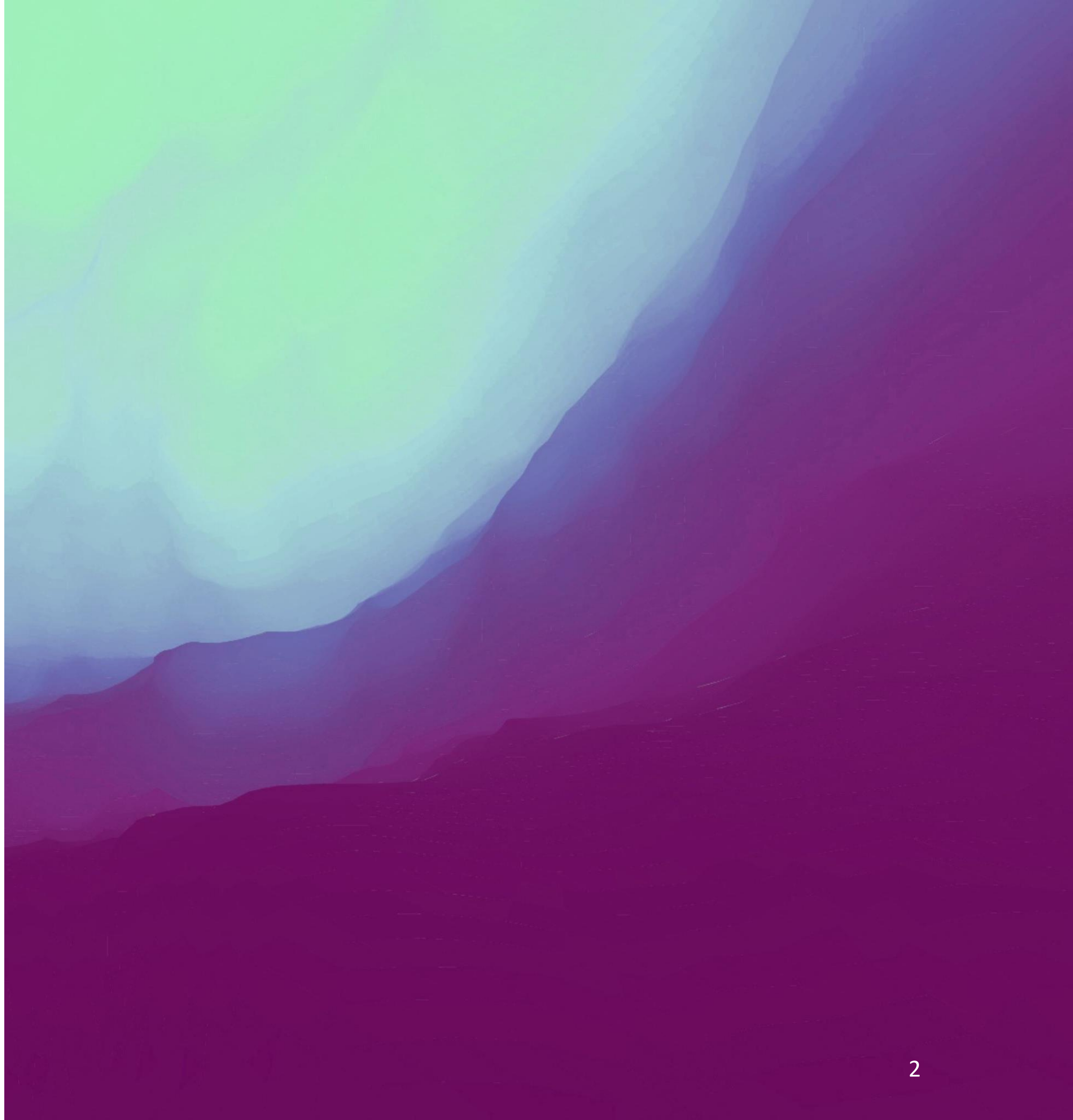
IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Bharadwaj Y V N S
22-09-2023

Outline



Executive Summary

- 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 from Machine Learning Lab

Introduction

- While other companies charge upwards of 165 million dollars per launch, SpaceX, a revolutionary firm, offers rocket launches, notably Falcon 9 launches, for as little as 62 million dollars. The majority of these savings can be attributed to SpaceX's brilliant idea to reuse the launch's first stage by re-landing the rocket for the following flight. The price will decrease considerably more if this method is repeated. The objective of my project, which I am working on as a data scientist for a firm that competes with SpaceX, is to build a machine learning pipeline to forecast how the first stage will land in the future. The success of this initiative will determine how much to offer SpaceX for a rocket launch.

- The problems included: Identifying all factors that influence the landing outcome. The relationship between each variables and how it is affecting the outcome. The best condition needed to increase the probability of successful landing

Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

- Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
- Data was processed using one-hot encoding for 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

To complement the data obtained through the SpaceX API, we recognized the need to gather additional information that could influence the landing outcome. We turned to web scraping techniques to extract data from external sources.

Our web scraping efforts focused on obtaining data related to meteorological conditions, satellite configurations, and historical space missions. This enriched dataset provides a more comprehensive view of the variables that might impact the landing success of Falcon 9 rockets.

Data Collection – SpaceX API

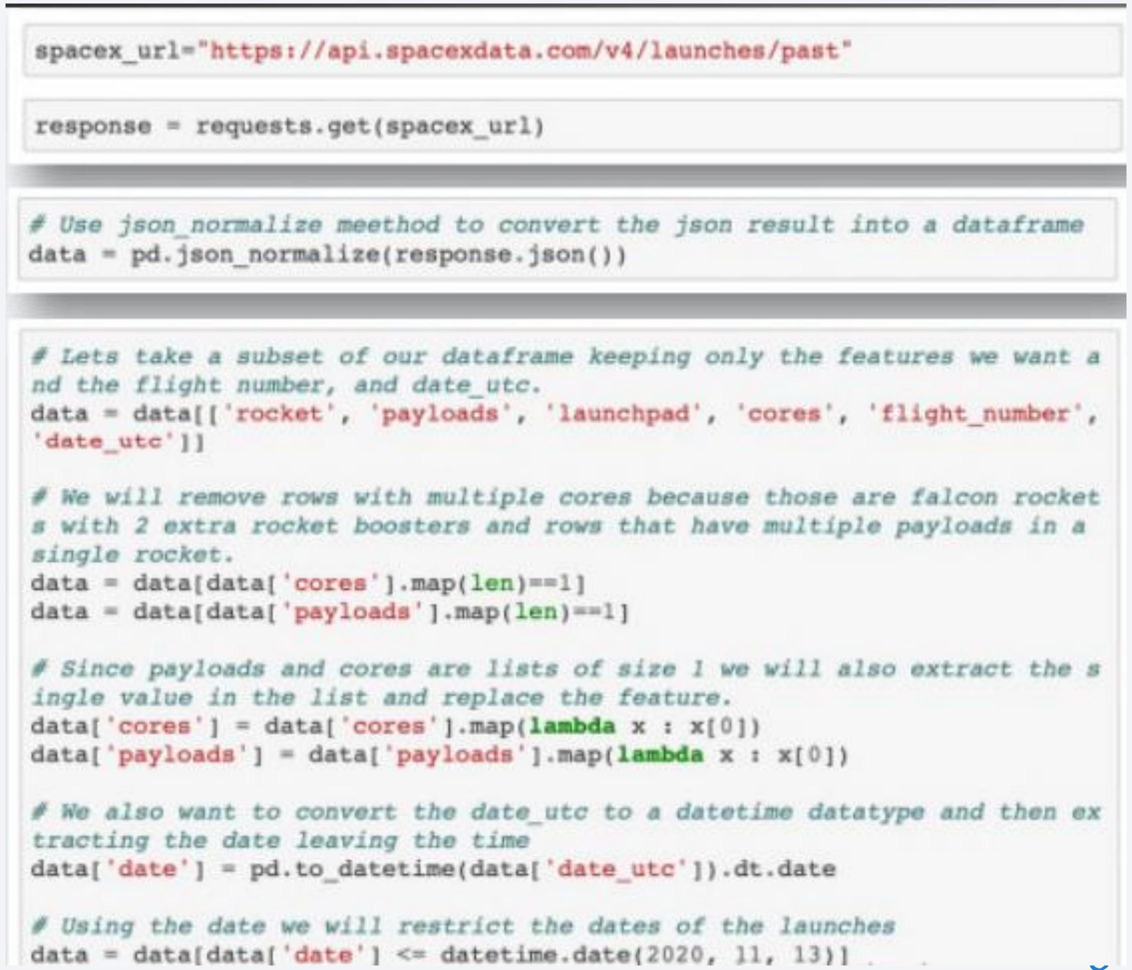
In this section, we'll outline how we collected crucial data using SpaceX's REST API. Our data collection process was essential in building a robust dataset for our machine learning pipeline.

Key Phrases:

SpaceX REST API: We accessed SpaceX's API to gather real-time data on Falcon 9 launches, including launch details, outcomes, and relevant information.

Data Retrieval: We utilized RESTful HTTP requests to retrieve data from the SpaceX API, ensuring we had access to the most up-to-date information.

Data Validation: To ensure data integrity, we validated and cleaned the retrieved data, addressing any missing or erroneous values.



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

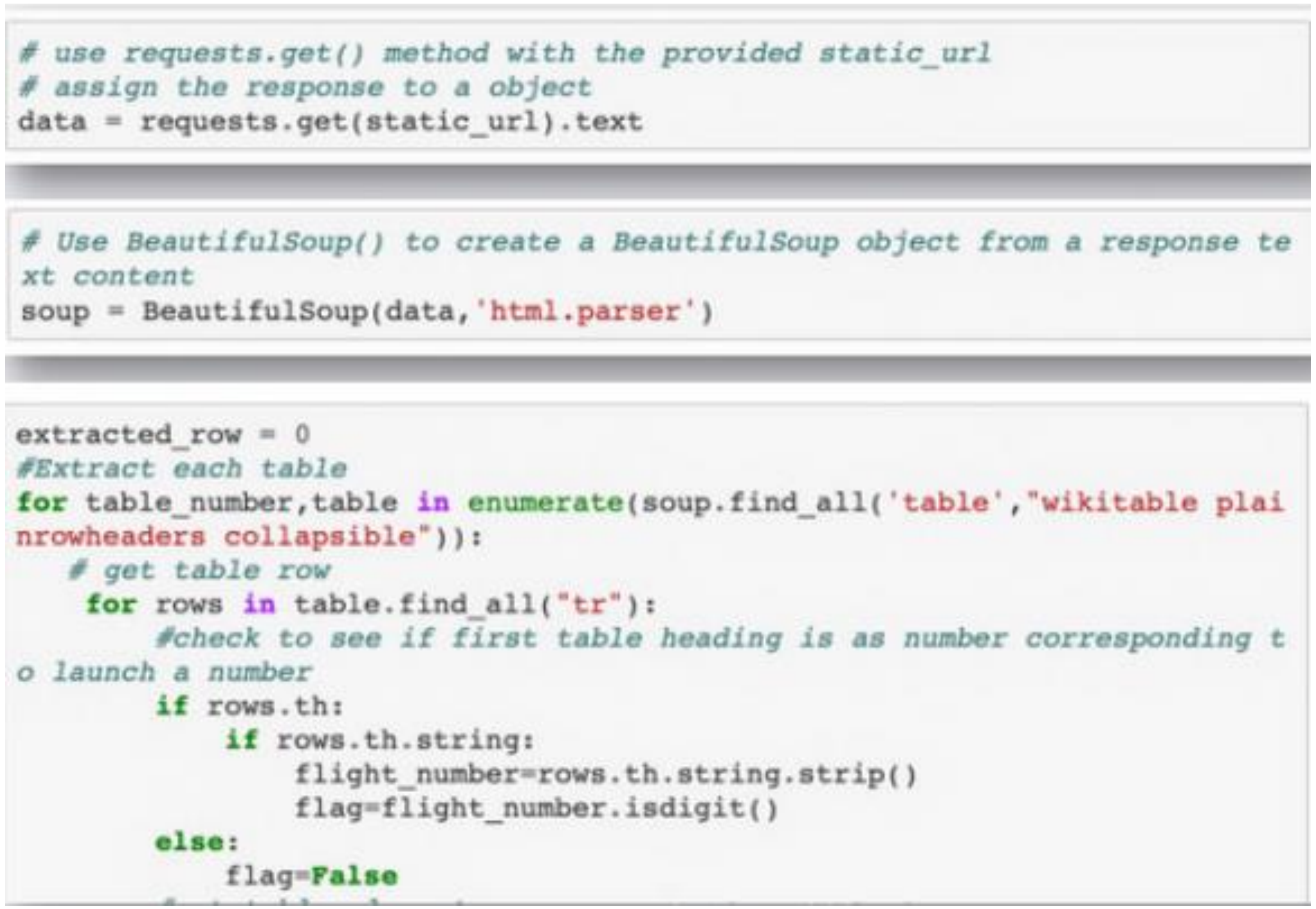
```
response = requests.get(spacex_url)
```

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

```
# Lets take a subset of our dataframe keeping only the features we want a
nd the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number',
'date_utc']]

# We will remove rows with multiple cores because those are falcon rocket
s with 2 extra rocket boosters and rows that have multiple payloads in a
single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the s
ingle value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then ex
tracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection - Scraping

- Get request for rocket launch data using API .
- Use json_normalize method to convert json result to dataframe .
- Performed data cleaning and filling the missing values.
- Request the Falcon9 Launch Wiki page from url.
- Create a BeautifulSoup from the HTML response.
- Extract all column/variable names
- from the HTML header.



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

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

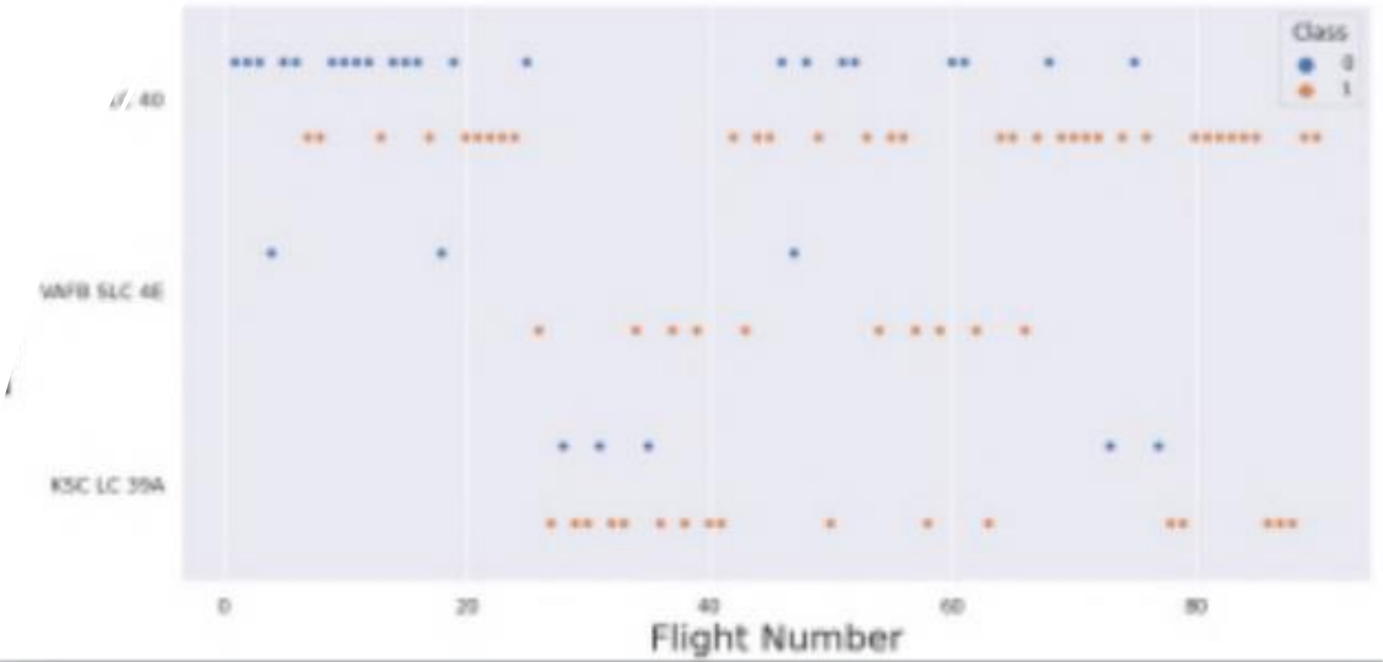
```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plai
nrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding t
o launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

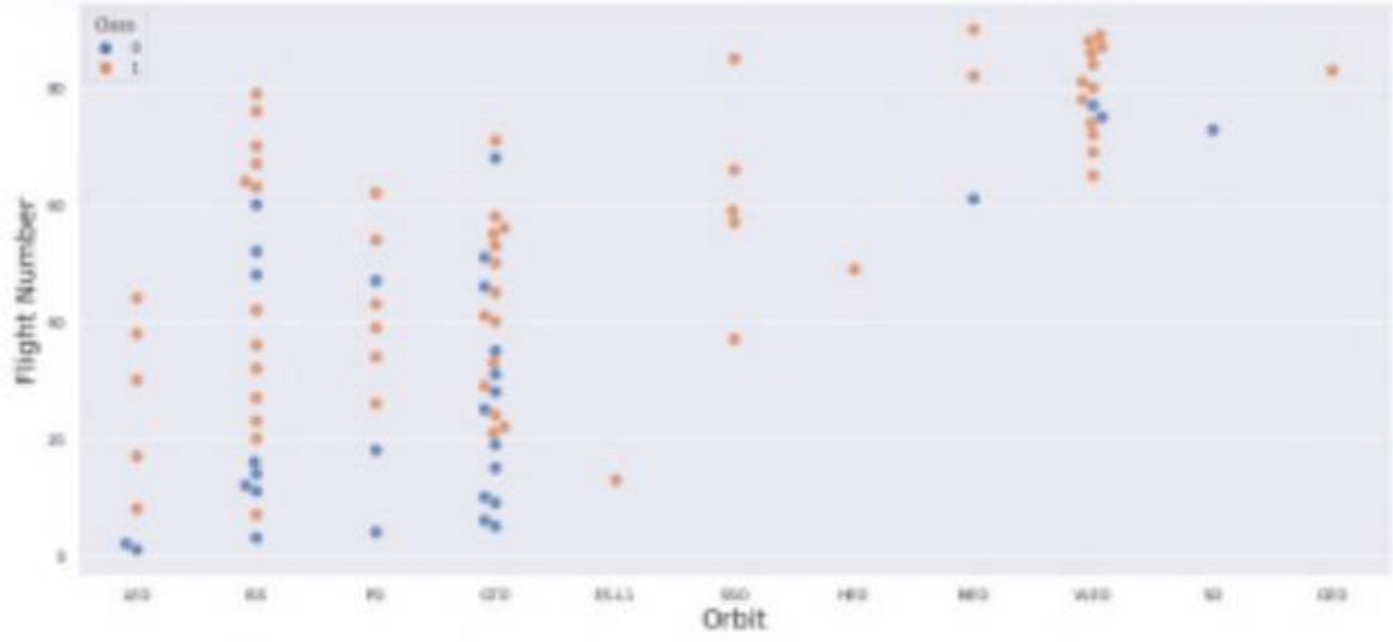
Data Wrangling

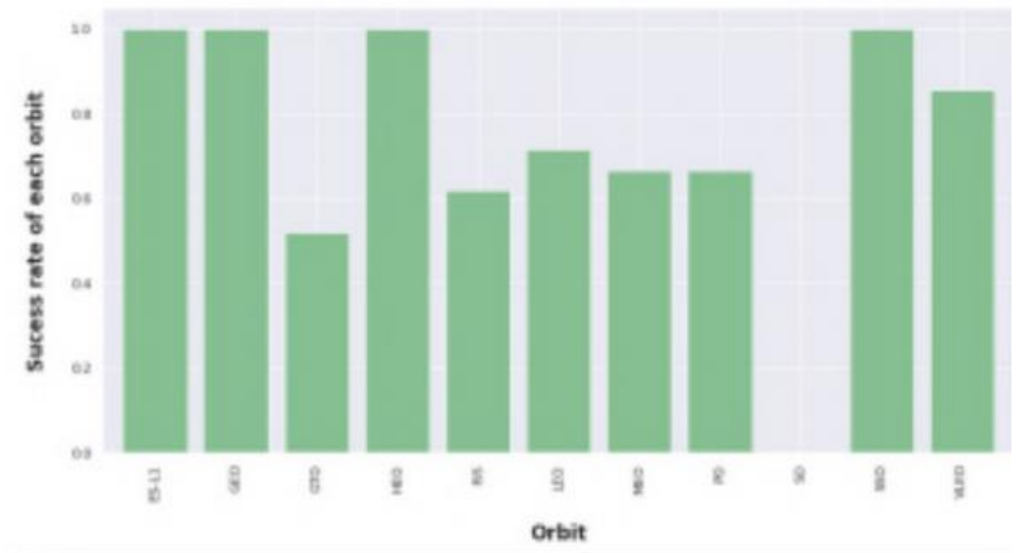
- For processing the data, we will first calculate the number of launches on each site, then calculate the number and occurrence of mission outcome per orbit type.
- We then create a landing outcome label from the outcome column.
- This will make it easier for further analysis, visualization, and ML. Lastly, we will export the result to a CSV.

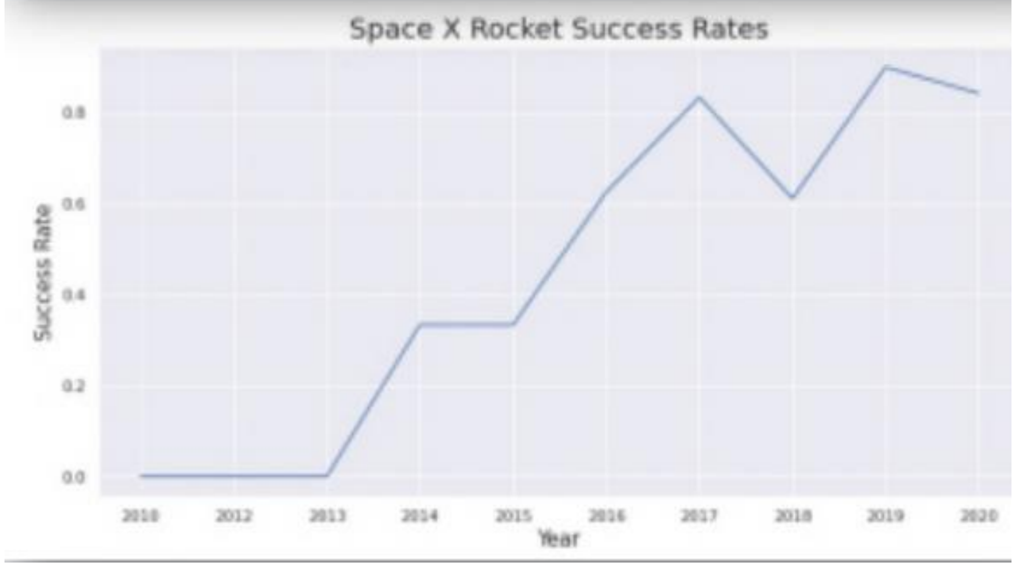
EDA with Data Visualization

- The association between the variables, such as between the Payload and Flight Number, was first determined using a scatter graph.
- Launch Site and Flight Number.
- The launch pad and payload.
- Orbital Type and Flight Number.
- Payload and kind of orbit.
- Scatter plots demonstrate the interdependence of qualities. Following the identification of a pattern in the graphs. It is relatively simple to identify the variables that have the most influence on the outcome of the landing.







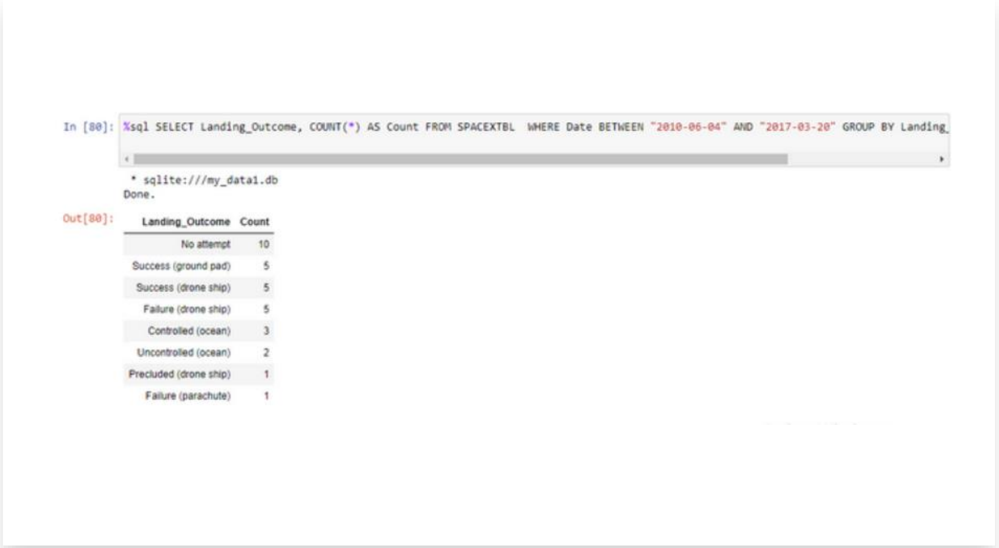


EDA with Data Visualization

- Once we get a hint of the relationships using scatter plot. We will then use further visualization tools such as bar graph and line plots graph for further analysis. Bar graphs is one of the easiest way to interpret the relationship between the attributes. In this case, we will use the bar graph to determine which orbits have the highest probability of success. We then use the line graph to show a trends or pattern of the attribute over time which in this case, is used for see the launch success yearly trend. We then use Feature Engineering to be used in success prediction in the future module by created the dummy variables to categorical columns.

EDA with SQL

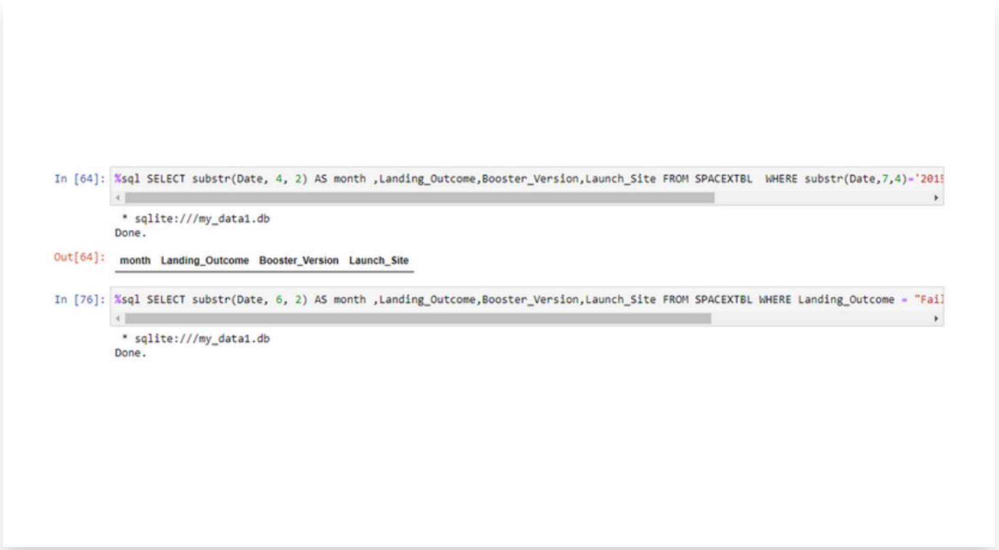
- Using SQL, we had performed many queries to get better understanding of the dataset, Ex: - Displaying the names of the launch sites.
- Displaying 5 records where launch sites begin with the string ‘CCA’. –
- Displaying the total payload mass carried by booster launched by NASA (CRS).
- Displaying the average payload mass carried by booster version F9 v1.1.
- Listing the date when the first successful landing outcome in ground pad was achieved.
- Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000. - Listing the total number of successful and failure mission outcomes.
- Listing the names of the booster_versions which have carried the maximum payload mass.
- Listing the failed landing_outcomes in drone ship, their booster versions, and launch sites names for in year 2015. - Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order.



```
In [80]: %sql SELECT Landing_Outcome, COUNT(*) AS Count FROM SPACEXTBL  WHERE Date BETWEEN "2010-06-04" AND "2017-03-20" GROUP BY Landing_

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

Out[80]:

Landing_Outcome	Count
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



```
In [64]: %sql SELECT substr(Date, 4, 2) AS month ,Landing_Outcome,Booster_Version,Launch_Site FROM SPACEXTBL  WHERE substr(Date,7,4)='2015

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

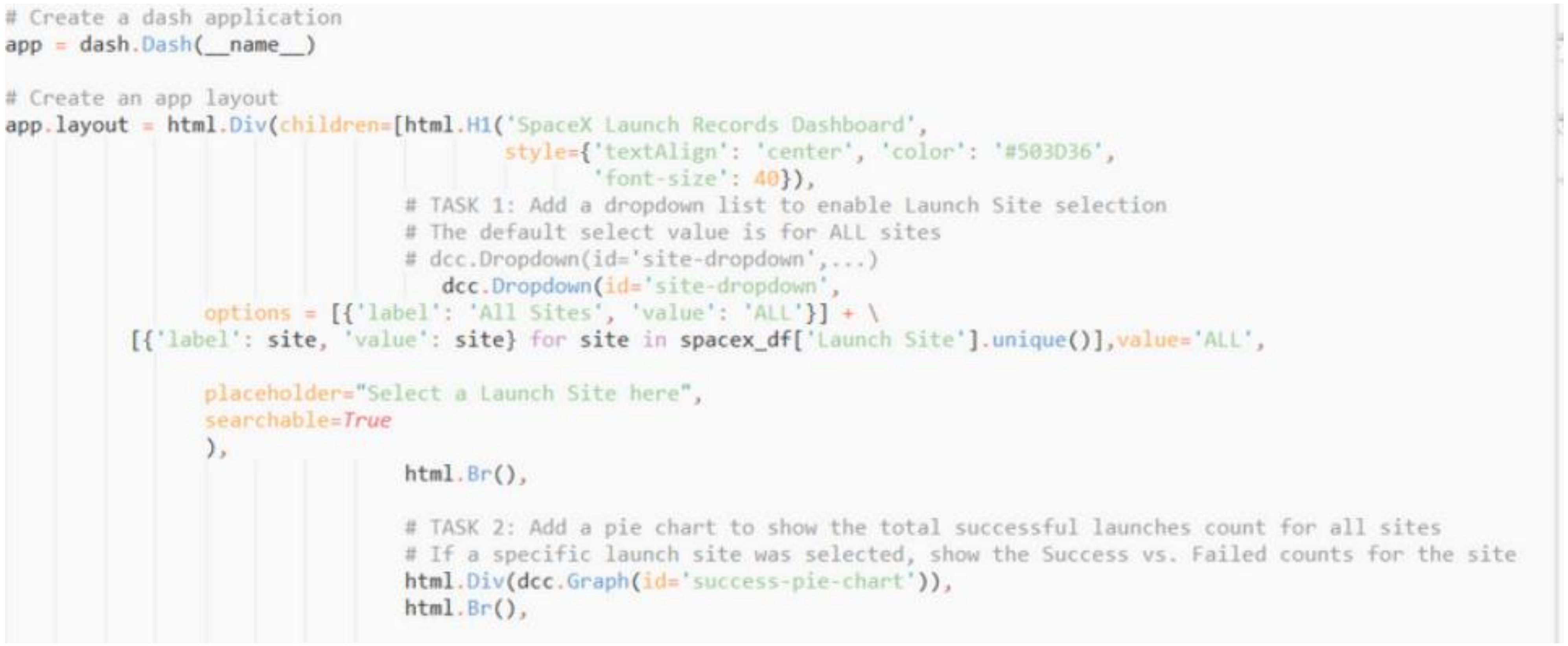
Out[64]:

month	Landing_Outcome	Booster_Version	Launch_Site

```
In [76]: %sql SELECT substr(Date, 6, 2) AS month ,Landing_Outcome,Booster_Version,Launch_Site FROM SPACEXTBL WHERE Landing_Outcome = "Fail

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

Build an Interactive Map with Folium

- To visualize the launch data into an interactive map. We took the latitude and longitude coordinates at each launch site and added a circle marker around each launch site with a label of the name of the launch site. We then assigned the dataframe launch_outcomes(failure,success) to classes 0 and 1 with Red and Green markers on the map in MarkerCluster(). We then used the Haversine's formula to calculated the distance of the launch sites to various landmark to find answer to the questions of:
- How close the launch sites with railways, highways and coastlines?
- How close the launch sites with nearby cities?



```
# 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': site, 'value': site} for site in spacex_df['Launch Site'].unique()],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(),
```

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash which allowing the user to play around with the data as they need.
- We plotted pie charts showing the total launches by a certain sites.
- We then plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

Predictive Analysis (Classification)

- Building the Model
  - Load the dataset into NumPy and Pandas
  - Transform the data and then split into training and test datasets
  - Decide which type of ML to use set the parameters and algorithms to GridSearchCV and fit it to dataset.
- Evaluating the Model
  - Check the accuracy for each model
  - Get tuned hyperparameters for each type of algorithms.
  - Plot the confusion matrix.

Predictive Analysis (Classification)

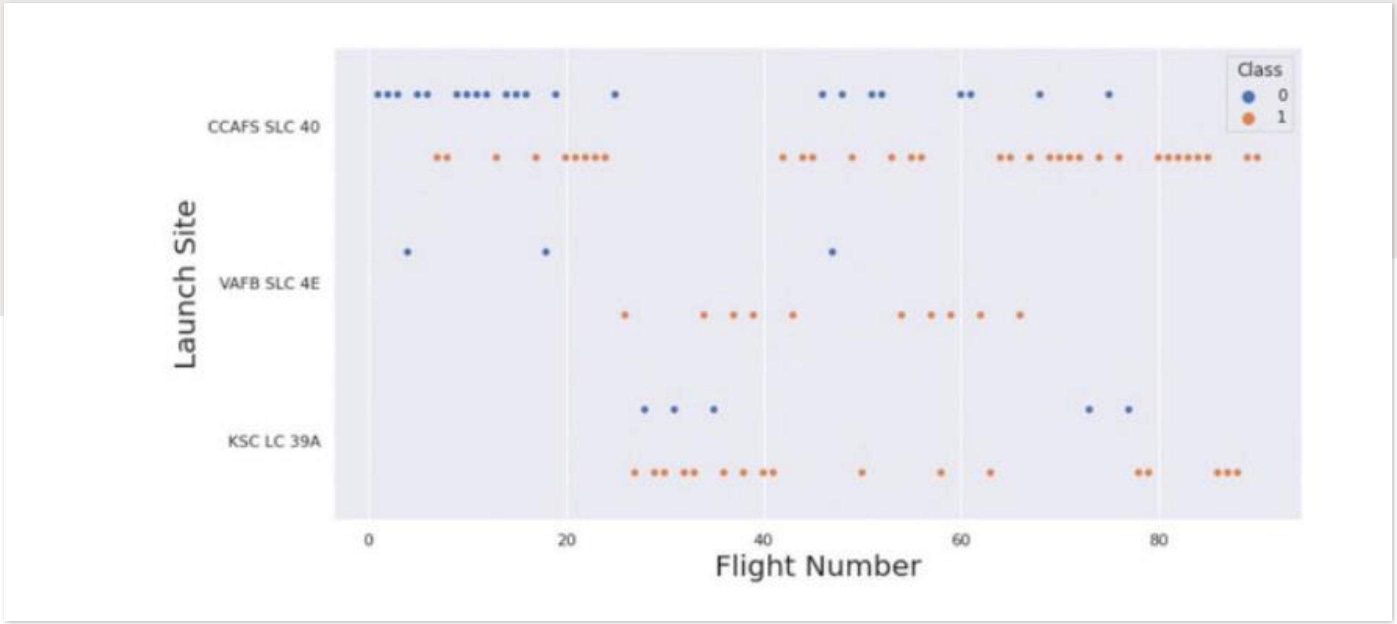
- Improving the Model
  - Use Feature Engineering and Algorithm Tuning

- Find the Best Model
  - The model with the best accuracy score will be the best performing model.

Section 2

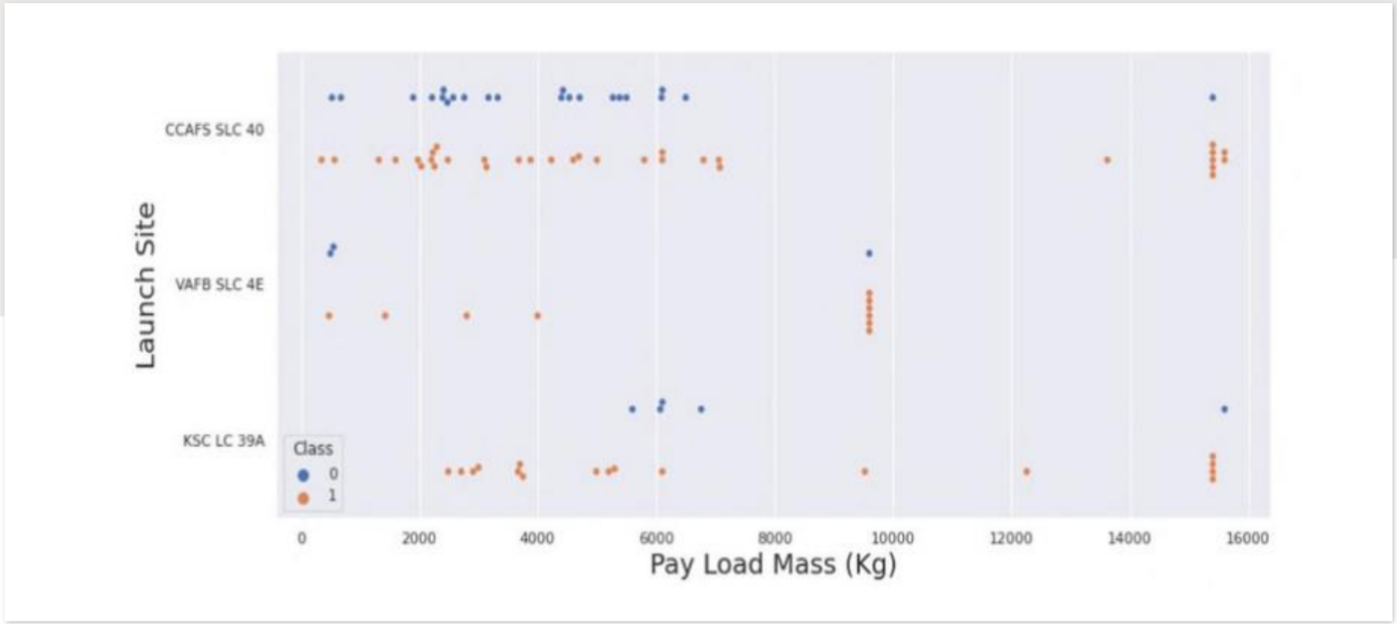
Insights drawn from EDA

Flight Number vs. Launch Site



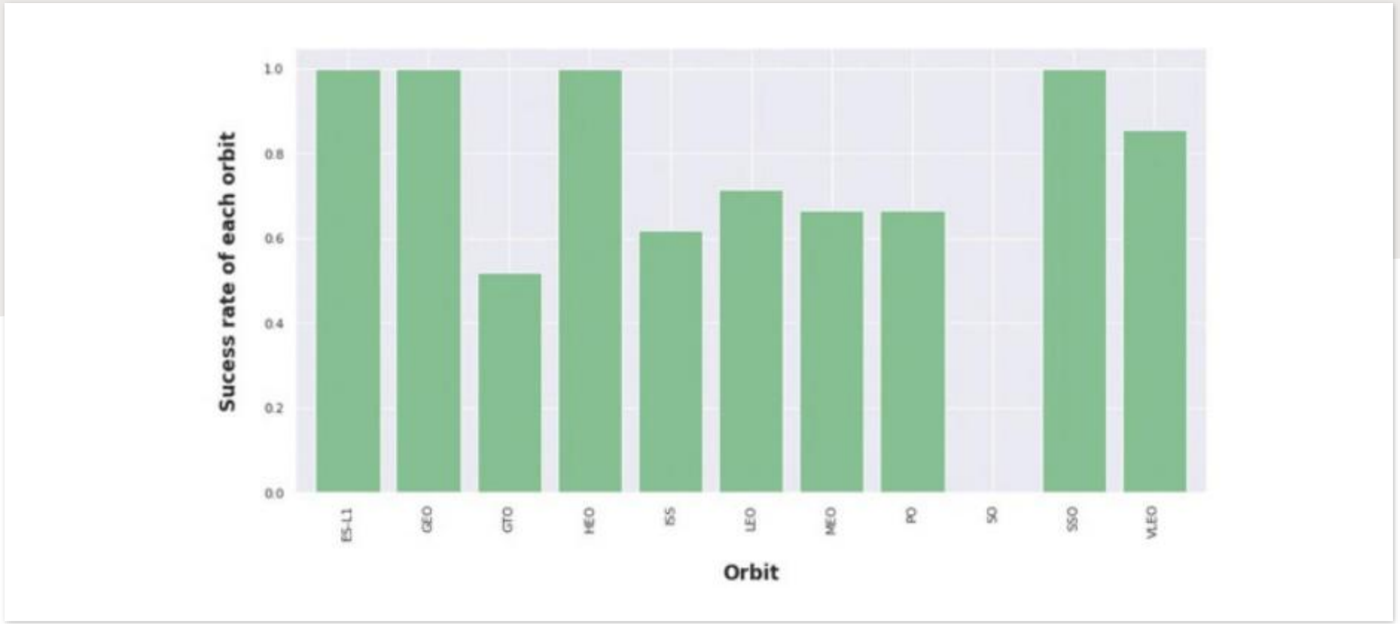
- This scatter plot shows that the larger the flights amount of the launch site, the greater the the success rate will be. However, site CCAFS SLC40 shows the least pattern of this.

Payload vs. Launch Site



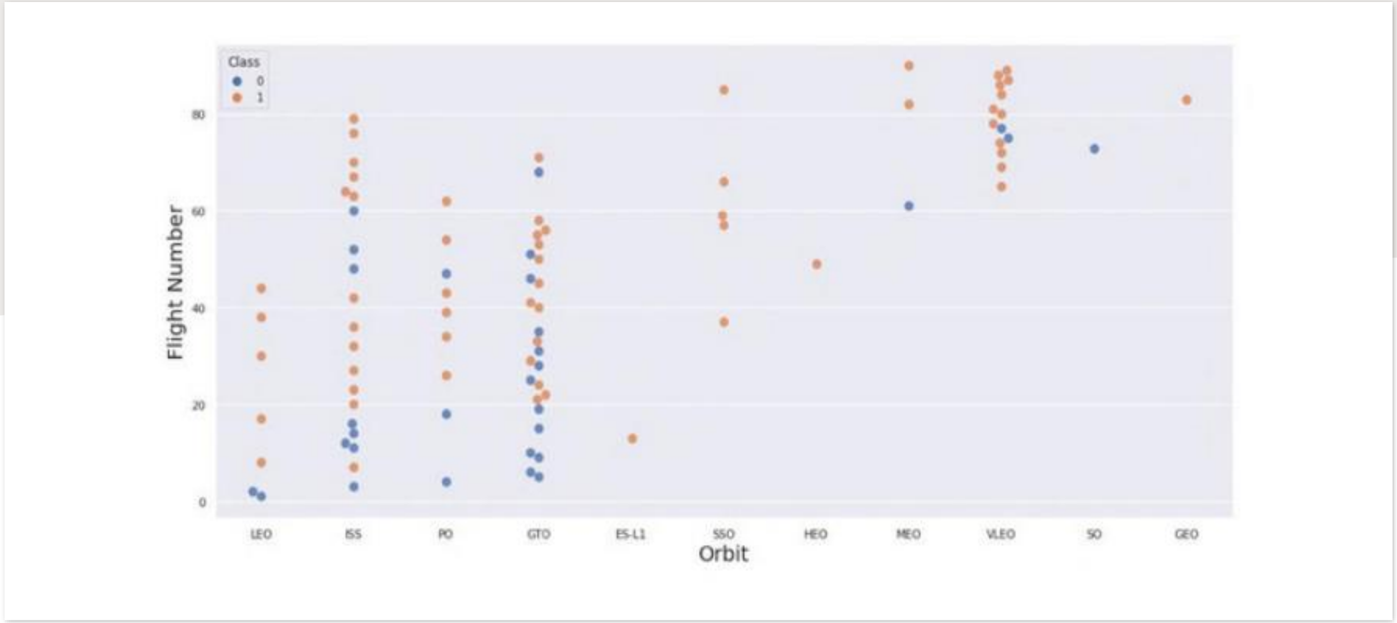
- This scatter plot shows once the pay load mass is greater than 7000kg, the probability of the success rate will be highly increased. However, there is no clear pattern to say the launch site is dependent to the pay load mass for the success rate.

Success Rate vs. Orbit Type



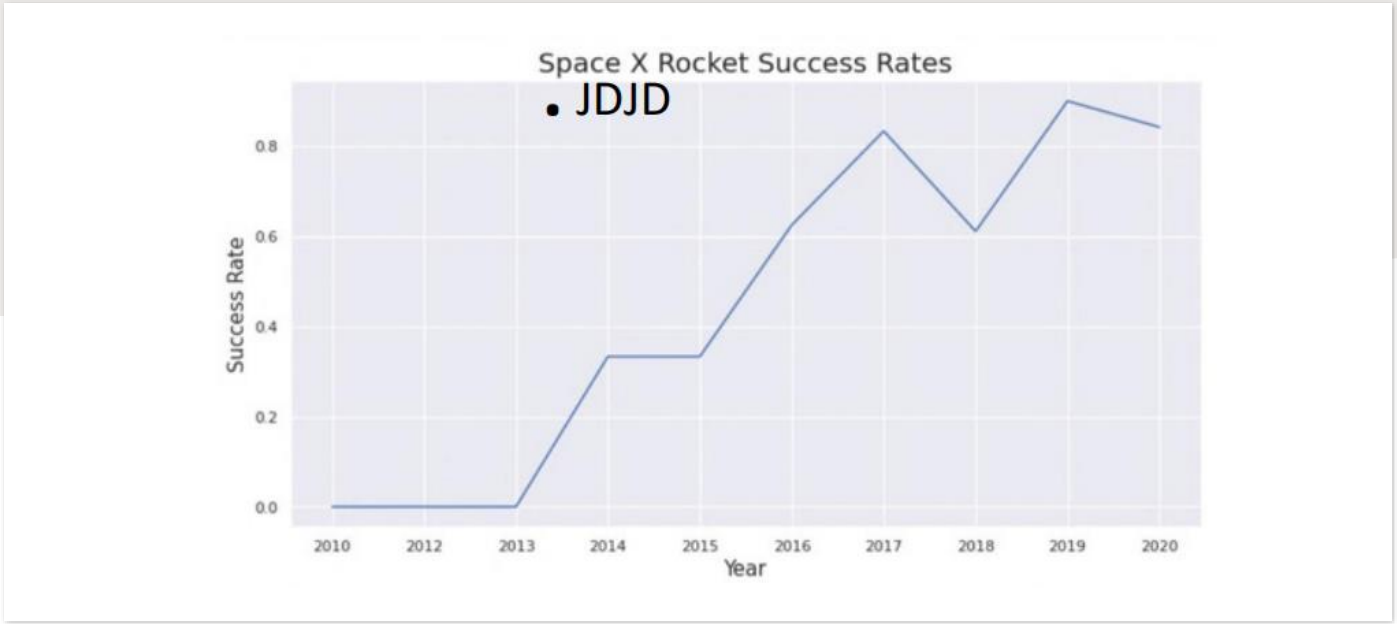
- This figure depicted the possibility of the orbits to influences the landing outcomes as some orbits has 100% success rate such as SSO, HEO, GEO AND ES-L1 while SO orbit produced 0% rate of success. However, deeper analysis show that some of this orbits has only 1 occurrence such as GEO, SO, HEO and ES-L1 which mean this data need more dataset to see pattern or trend before we draw any conclusion.

Flight Number vs. Orbit Type

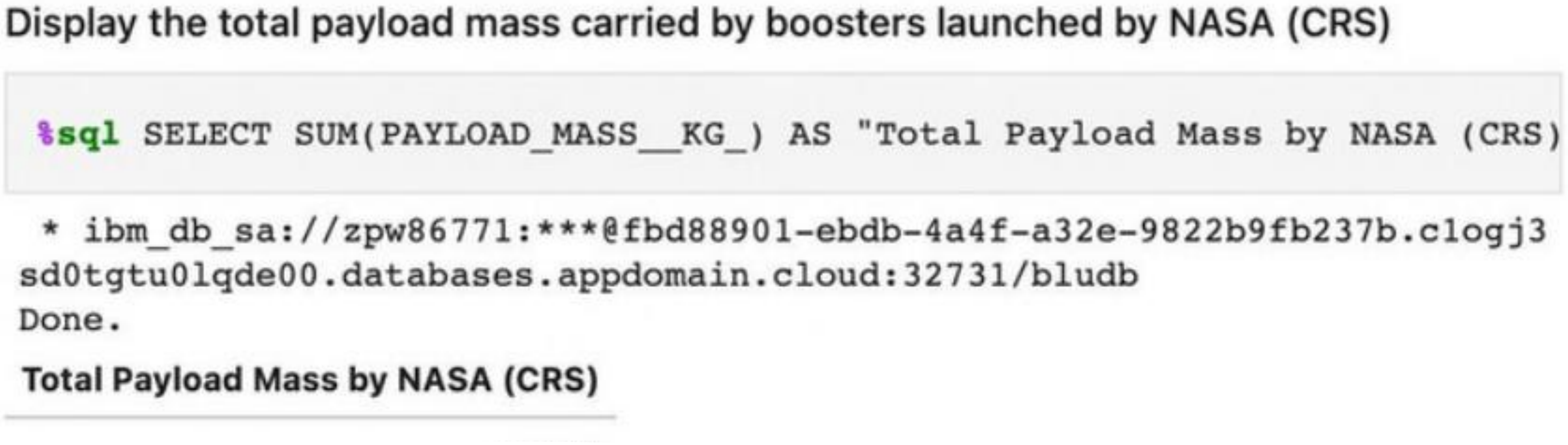


- This scatter plot shows that generally, the larger the flight number on each orbits, the greater the success rate (especially LEO orbit) except for GTO orbit which depicts no relationship between both attributes. Orbit that only has 1 occurrence should also be excluded from above statement as it's needed more dataset.

Launch Success Yearly Trend



- This figures clearly depicted and increasing trend from the year 2013 until 2020. If this trend continue for the next year onward. The success rate will steadily increase until reaching 1/100% success rate.



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

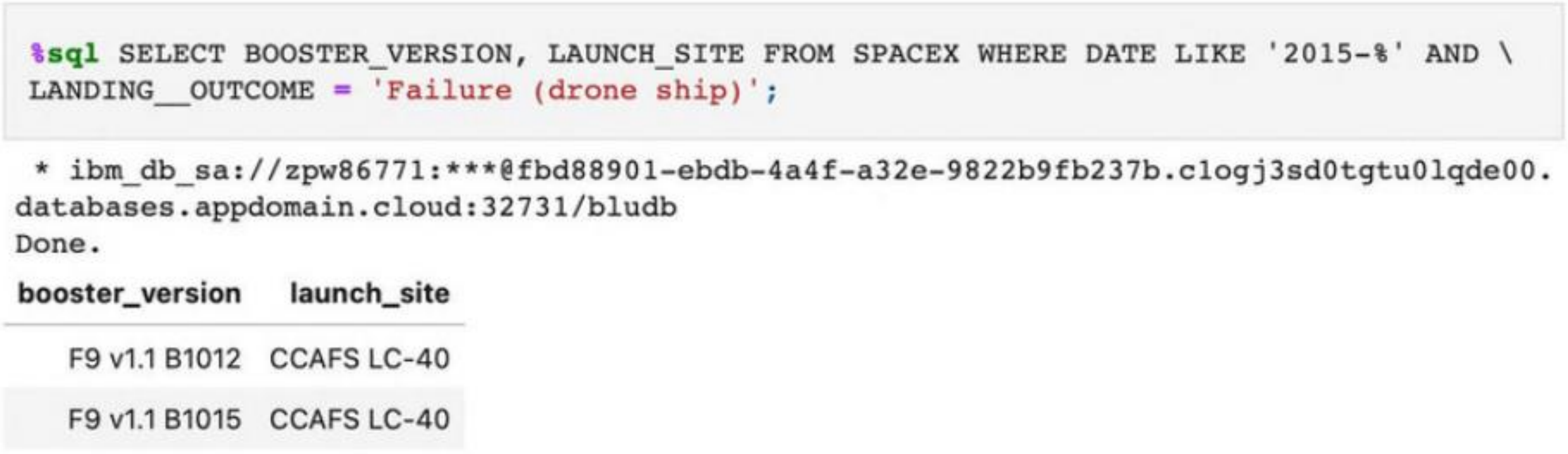
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu01qde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Payload Mass by NASA (CRS)

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 45596 using the query ABOVE



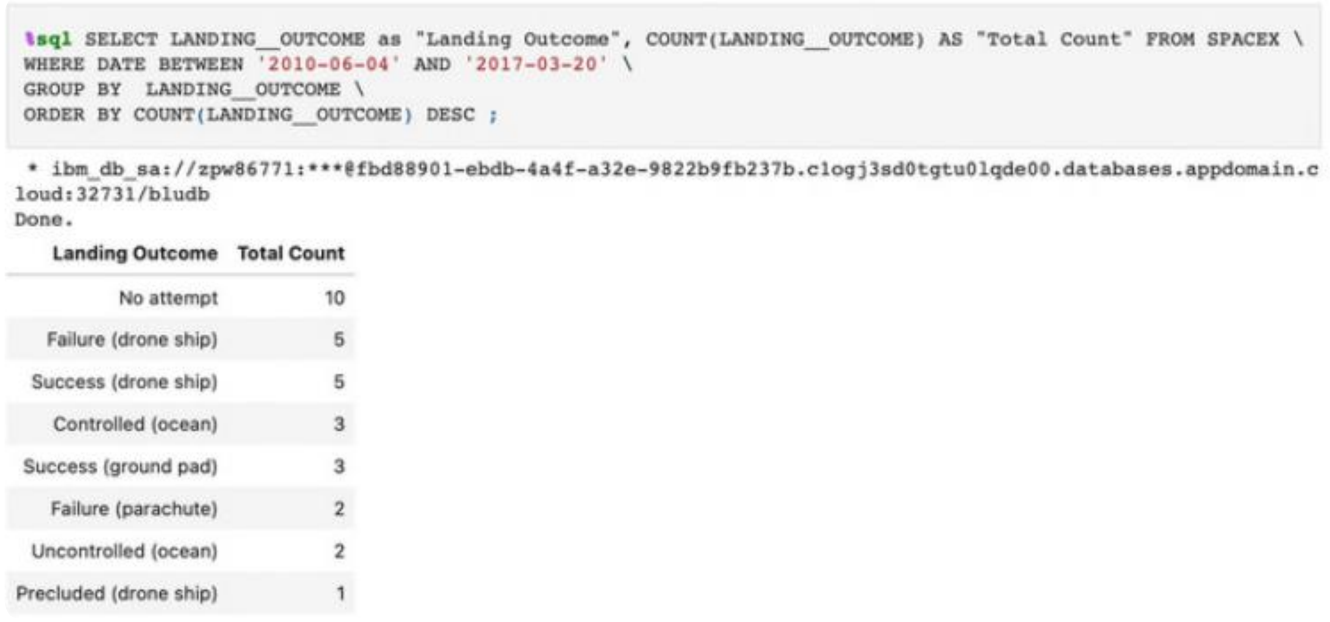
```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.
databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

2015 Launch Records

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



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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.c
loud:32731/bludb
Done.
```

Landing Outcome	Total Count
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

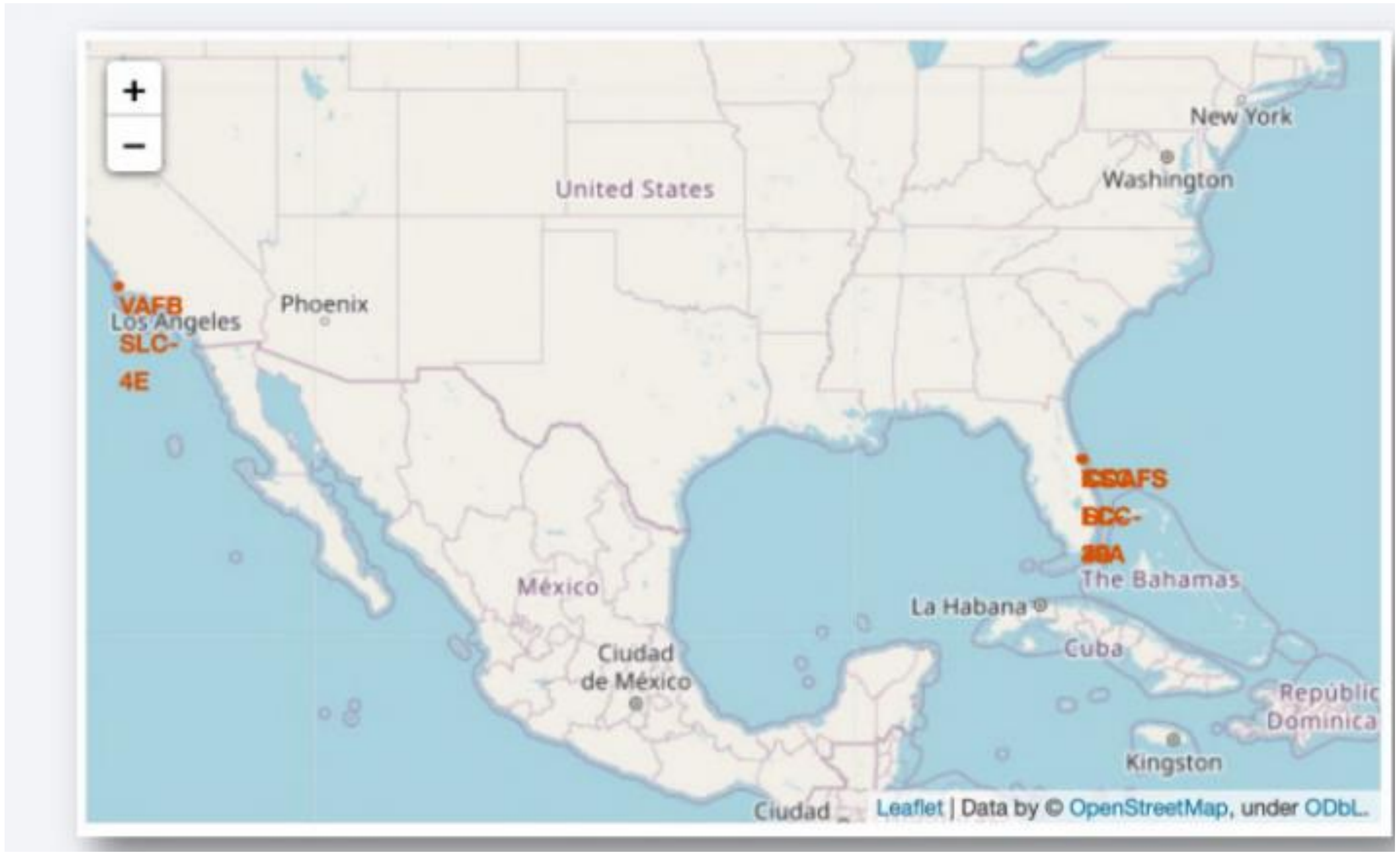
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN2010-06-04 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.

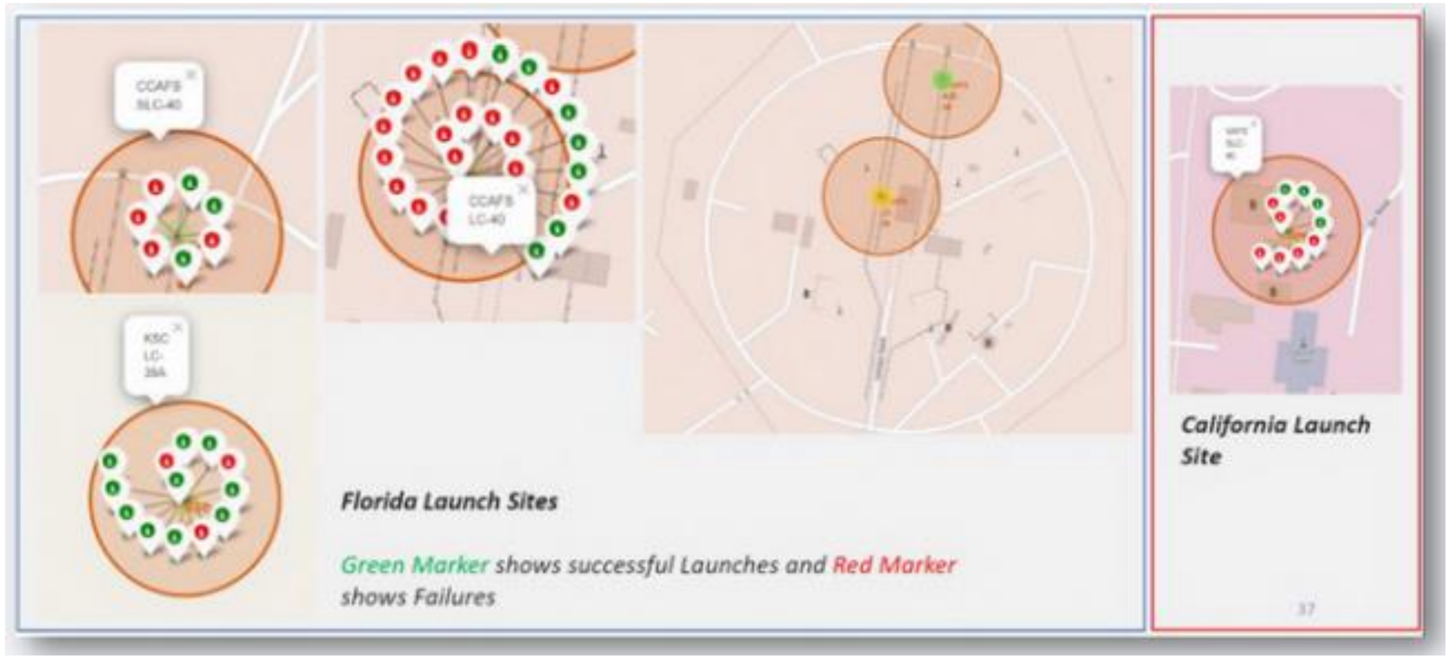
Section 3

Launch Sites Proximities Analysis

Launch Site Location



Markers showing launch sites with color labels

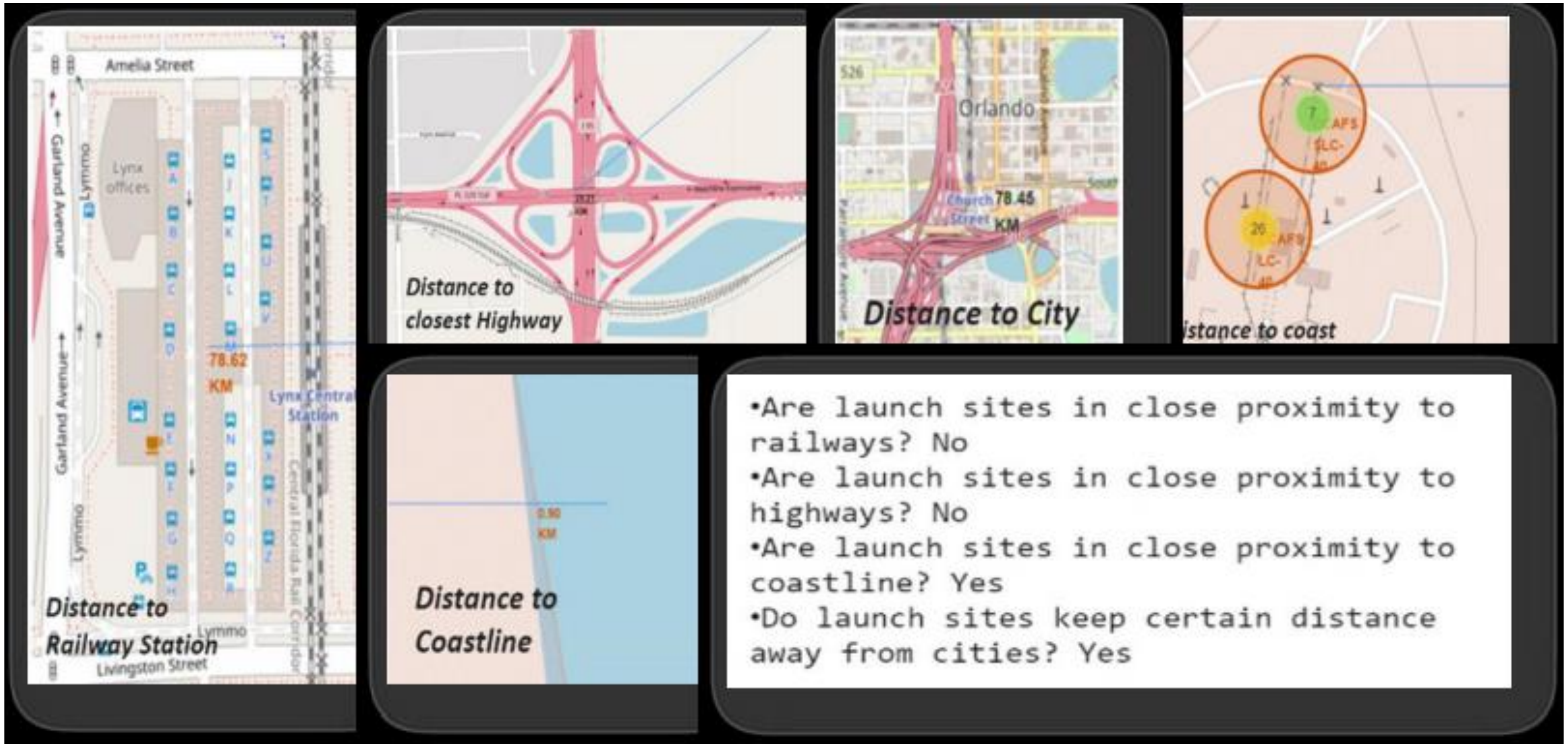


Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

Launch Sites Distance to Landmarks

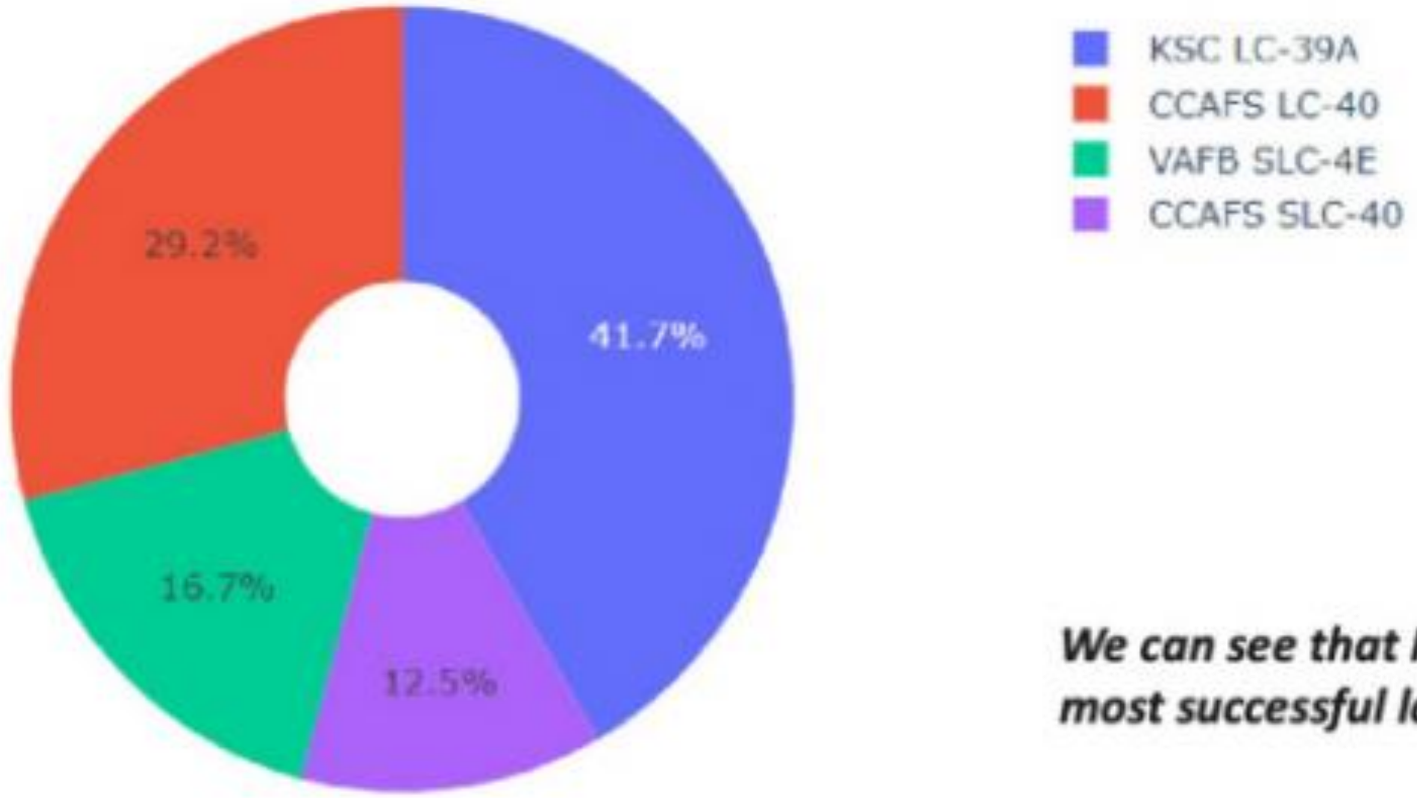


•Are launch sites in close proximity to railways? No
•Are launch sites in close proximity to highways? No
•Are launch sites in close proximity to coastline? Yes
•Do launch sites keep certain distance away from cities? Yes

Section 4

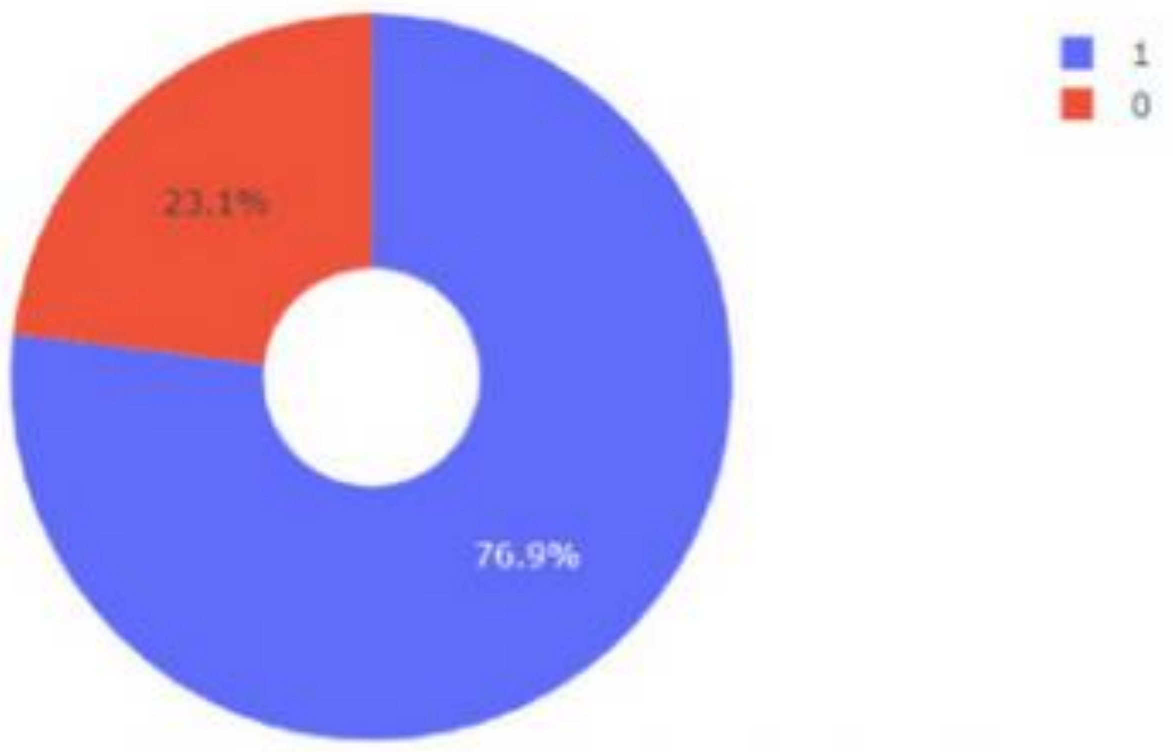
Build a Dashboard with Plotly Dash

The success percentages by each site:



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

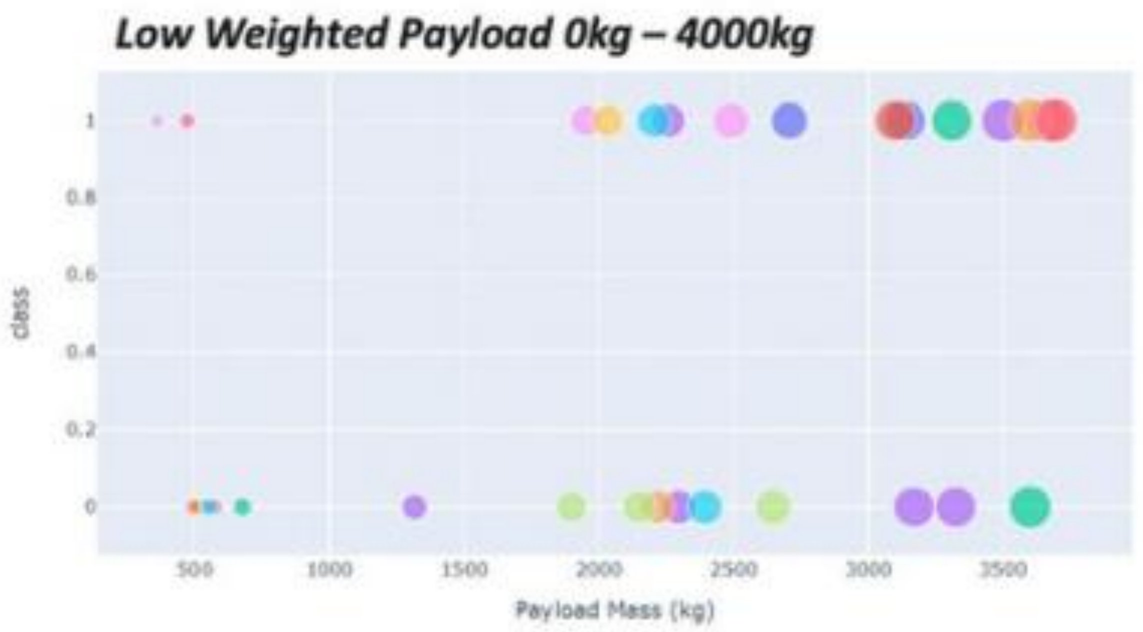
Highest success rate

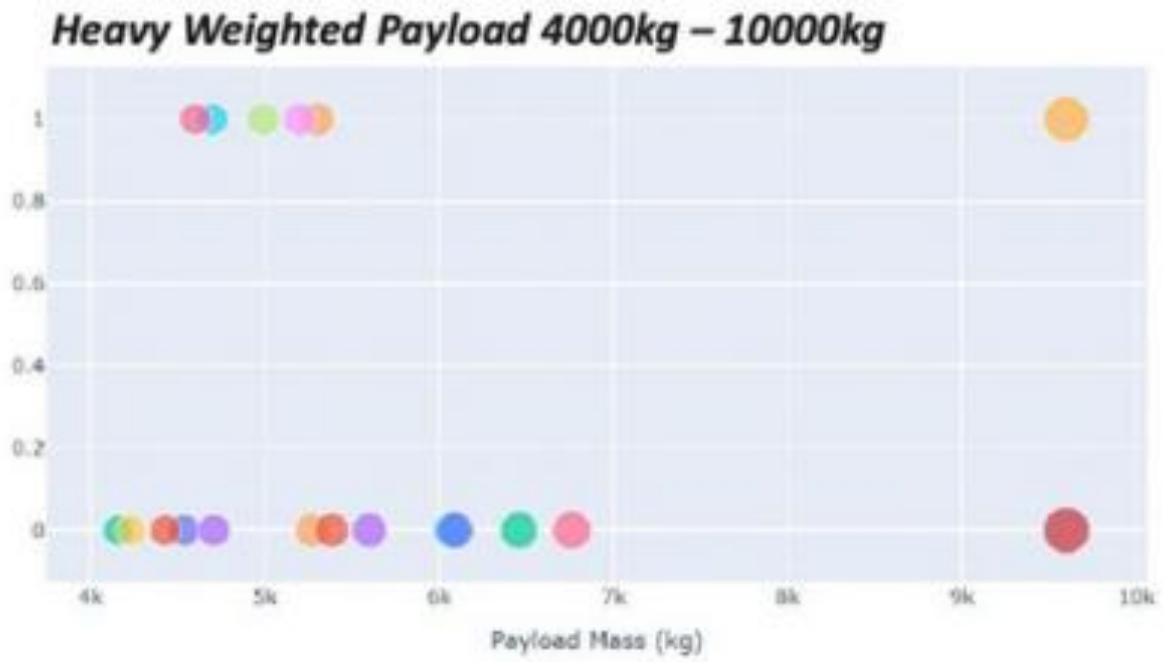


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch Outcome Scatter Plot

Success rate for low weighted payload is higher than heavy weighted payload

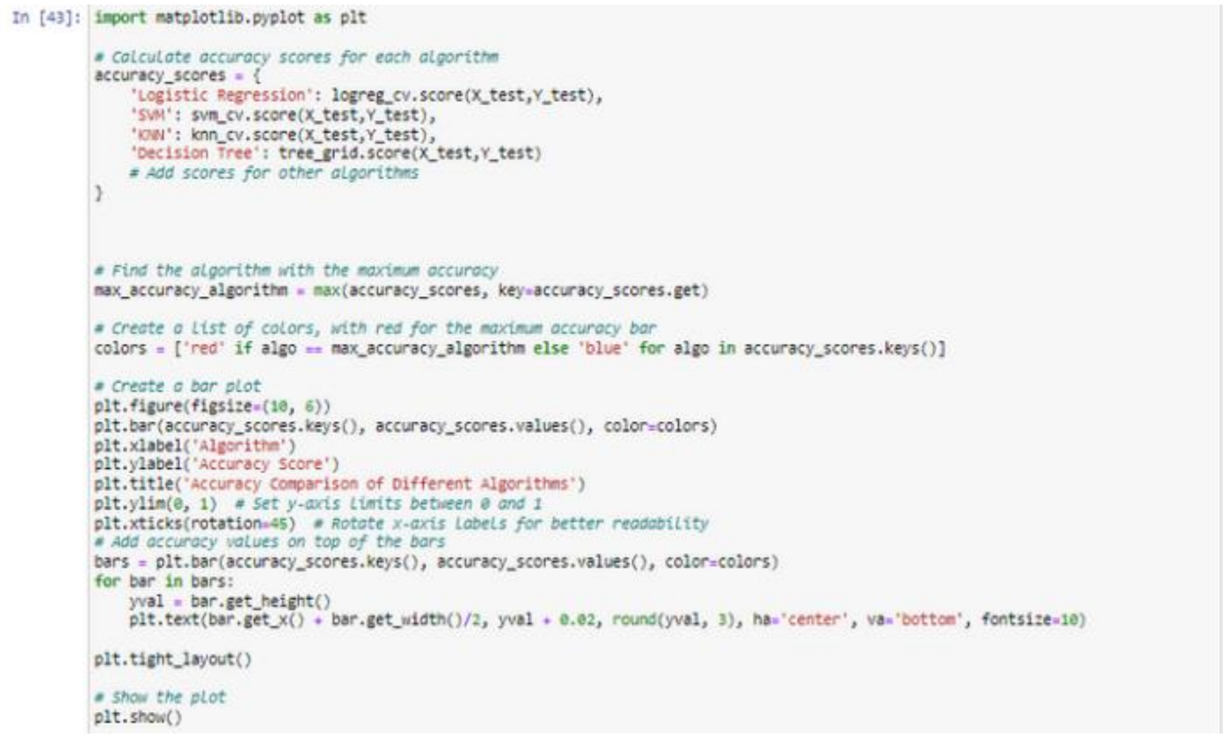




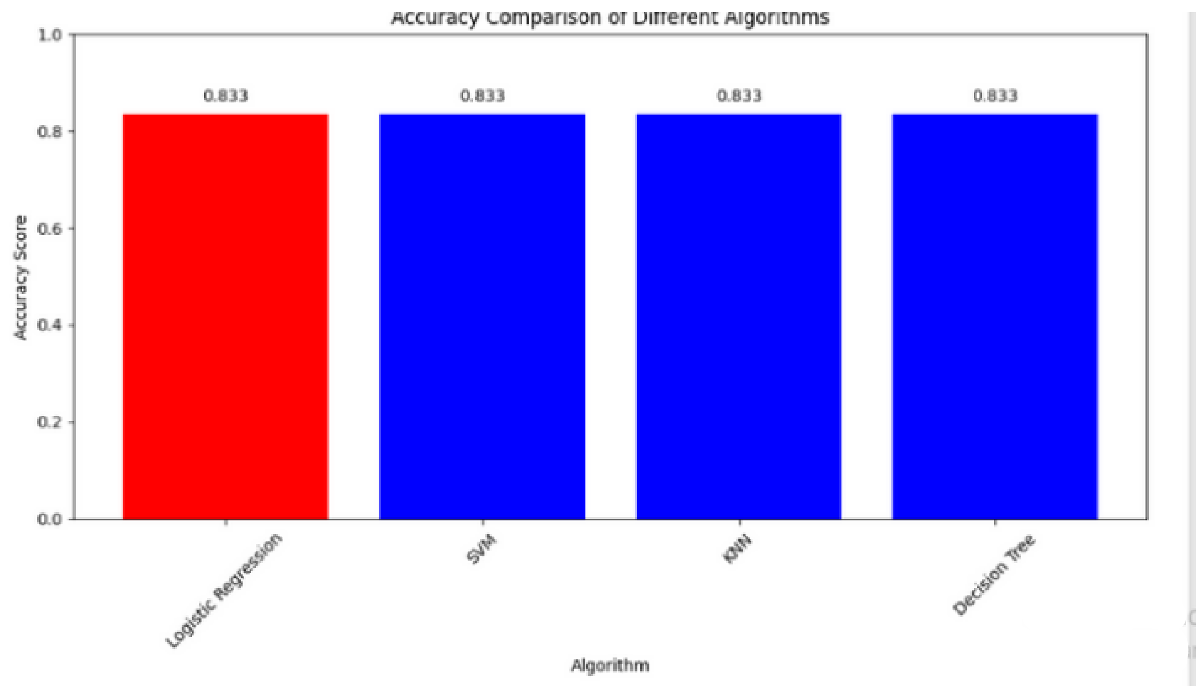
Section 5

Predictive Analysis (Classification)

Classification Accuracy

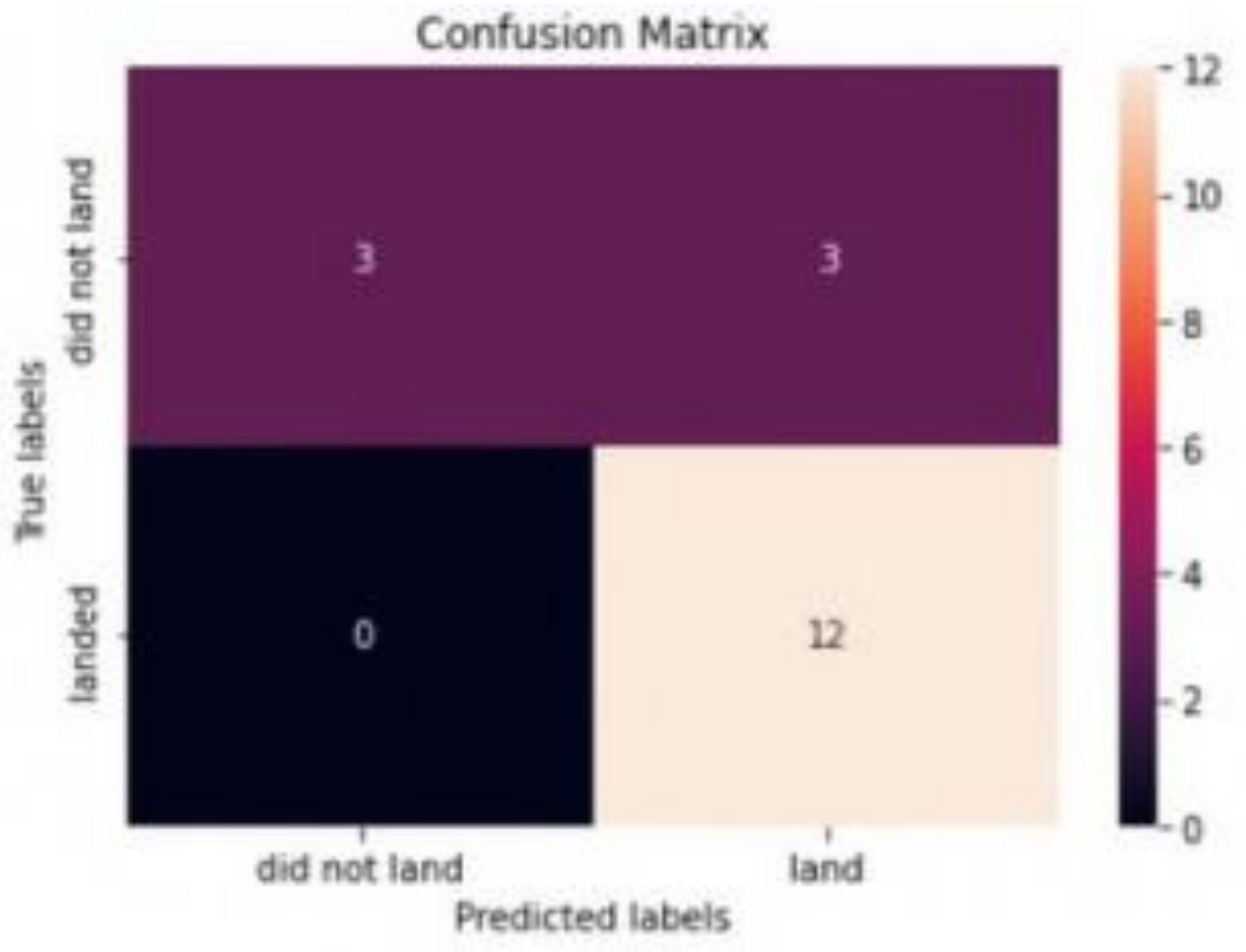


```
In [43]: import matplotlib.pyplot as plt

         # Calculate accuracy scores for each algorithm
         accuracy_scores = {
             'Logistic Regression': logreg_cv.score(X_test,Y_test),
             'SVM': svm_cv.score(X_test,Y_test),
             'KNN': knn_cv.score(X_test,Y_test),
             'Decision Tree': tree_grid.score(X_test,Y_test)
             # Add scores for other algorithms
         }



         # Find the algorithm with the maximum accuracy
         max_accuracy_algorithm = max(accuracy_scores, key=accuracy_scores.get)

         # Create a list of colors, with red for the maximum accuracy bar
         colors = ['red' if algo == max_accuracy_algorithm else 'blue' for algo in accuracy_scores.keys()]

         # Create a bar plot
         plt.figure(figsize=(10, 6))
         plt.bar(accuracy_scores.keys(), accuracy_scores.values(), color=colors)
         plt.xlabel('Algorithm')
         plt.ylabel('Accuracy Score')
         plt.title('Accuracy Comparison of Different Algorithms')
         plt.ylim(0, 1)  # Set y-axis limits between 0 and 1
         plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
         # Add accuracy values on top of the bars
         bars = plt.bar(accuracy_scores.keys(), accuracy_scores.values(), color=colors)
         for bar in bars:
             yval = bar.get_height()
             plt.text(bar.get_x() + bar.get_width()/2, yval + 0.02, round(yval, 3), ha='center', va='bottom', fontsize=10)

         plt.tight_layout()

         # Show the plot
         plt.show()
```



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

- The Tree Classifier Algorithm is the best Machine Learning approach for this dataset.
- The low weighted payloads (which define as 4000kg and below) performed better than the heavy weighted payloads.
- Starting from the year 2013, the success rate for SpaceX launches is increased, directly proportional time in years to 2020, which it will eventually perfect the launches in the future.
- KSC LC-39A have the most successful launches of any sites; 76.9% SSO orbit have the most success rate; 100% and more than 1 occurrence.

Thank you!