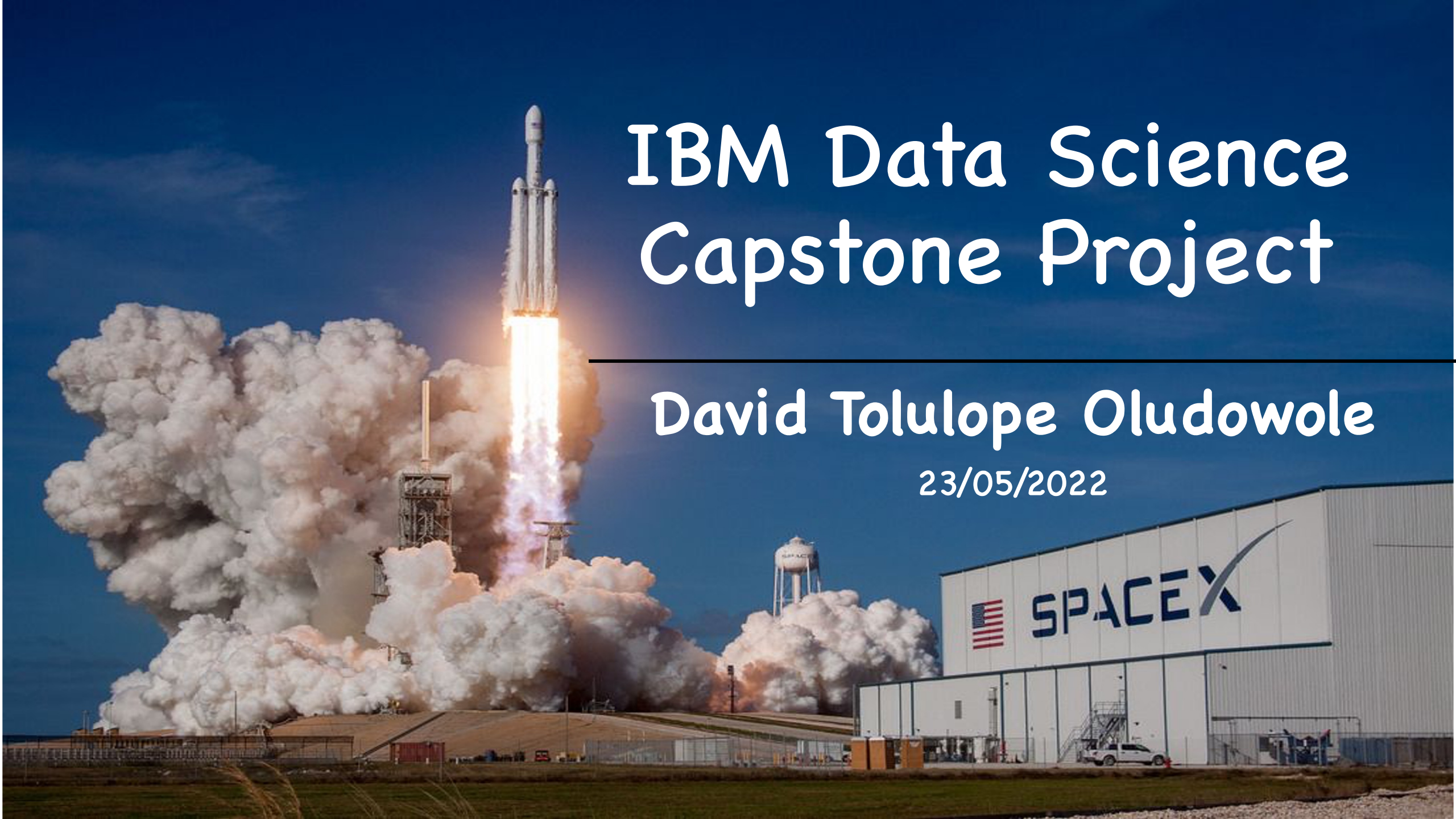


IBM Data Science Capstone Project

David Tolulope Oludowole

23/05/2022

Executive Summary

- In this project, we present data collected from SpaceX and Wikipedia.
- We explored the data using Exploratory Data Analysis EDA using Python and SQL.
- Visualisation maps (Folium) and Dashboards were also generated to show relevant information as regards successful landings.
- Machine Learning models (Logistic Regression, Support Vector Machine, Decision Tree Classifier and K Nearest Neighbours) were also deployed to model the dataset.

Table of Contents

Introduction

- The launching of a SpaceX Falcon 9 rockets cost approx. $62m
- This is way cheaper compared to other providers (Cost approx. $165m)
- The difference is price is because SpaceX rockets can land, and be re-used again.
- If we can determine if the first stage will land, we can determine the cost of the launch.
- This information will guide us if our new company Space Y should compete in the Space travel sector

SpaceX Falcon 9 Rocket – The Verge

Methodology

Methodology

- **Data Collection**
  - Using GET requests from SpaceX REST API
  - Web scraping from Wikipedia's page
- **Data Wrangling and Manipulation**
  - Removing NaN values with the .fillna() method
  - Calculating number of launches and missions using the .value_counts() method
- **Exploratory Data Analysis (EDA)**
  - Using SQL, Python Libraries (Pandas, Matplotlib) to visualise data and explore realtionships
- **Interactive Data Analysis**
  - Generating geospatial analysis using Folium and interactive dashboards using Plotly Dash
- **Data Modelling and Evaluation**

Data Collection (SPACEX REST API)

Step 1

Make GET requests from SpaceX REST API

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

response = requests.get(spacex_url)
```

Convert the response to a .json file and use pandas to generate the data frame.

```
# Use json_normalize meethod to convert the json result into a dataframe

data = pd.json_normalize(response.json())
```

Step 2

Clean Data

```
# We also want to convert the date_utc to a datetime datatype and then ext
racting 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)]
```

Create Lists
Call Functions
Convert the response to a .json file

```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

Step 3

Create Pandas Dataframe

```
# Create a data from launch_dict

data2 = pd.DataFrame(launch_dict)
```

Step 4

Filter Data in the dataframe, Replace missing values

```
# Replace the np.nan values with its mean value

temp = data_falcon9['PayloadMass'].replace(np.nan, pm_mean)
data_falcon9['PayloadMass'] = temp
data_falcon9
```

Data Collection (WEB SCRAPING)

Step 1

Request HTML page

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

Assign response to an object

```
# assign the response to a object

page = requests.get(static_url)
```

Step 2

Create BeautifulSoup object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

Fill all the table in the HTML Page

```
html_tables=soup.find_all('table')
```

Step 3

Extract Column Names from tables in HTML Page

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

for i in first_launch_table.find_all('th'):
    if extract_column_from_header(i)!=None:
        if len(extract_column_from_header(i))>0:
            column_names.append(extract_column_from_header(i))
```

Step 4

Use column names as keys in dictionaries

Convert to Pandas

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

```
df=pd.DataFrame(launch_dict)
```

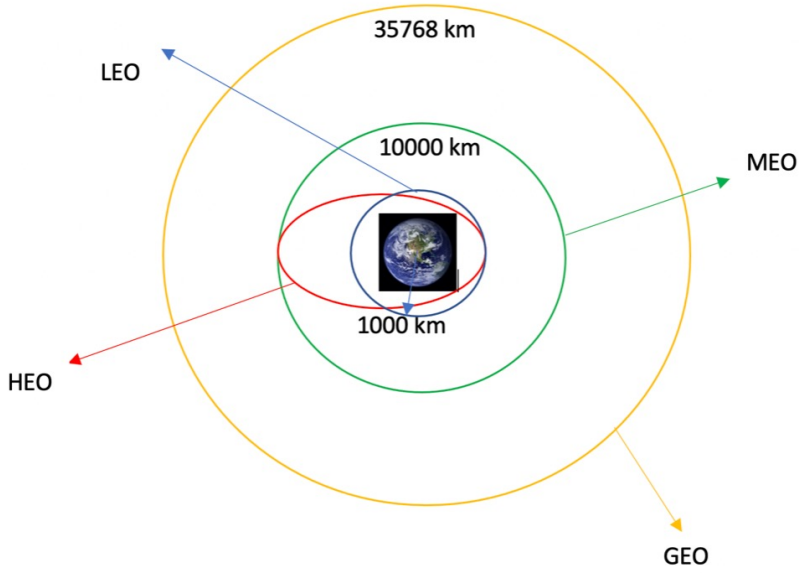
Data Wrangling and Manipulation

- The dataset contains several SpaceX launch facilities and each location is in the LaunchSite column.

```
# Apply value_counts() on column LaunchSite

df["LaunchSite"].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```



- Initial Data Exploration [No of Launches, Occurrence of each Orbit, Landing outcome per orbit]

```
# Apply value_counts on Orbit column

df["Orbit"].value_counts()
```

```
GTO     27
ISS     21
VLEO    14
PO       9
LEO      7
SSO      5
MEO      3
ES-L1    1
HEO      1
SO       1
GEO      1
Name: Orbit, dtype: int64
```

```
# landing_outcomes = values on Outcome column

landing_outcomes = df["Outcome"].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

Data Wrangling and Manipulation

- Displaying the Landing Outcome Columns

```
# landing_outcomes = values on Outcome column

landing_outcomes = df["Outcome"].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

- Creating a list [landing_class]to check if the booster would land successfully or not.

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

def onehot(item):
    if item in bad_outcomes:
        return 0
    else:
        return 1
landing_class = df["Outcome"].apply(onehot)
landing_class
```

- Exporting the Data frame to a .csvfile

```
df['Class']=landing_class
df[['Class']].head(8)
```

```
df.to_csv("dataset_part\_2.csv", index=False)
```

Exploratory Data Analysis [EDA] with Data Visualisation

- Exploratory Data Analysis was performed on certain variables and displayed using various tools

SCATTER PLOTS

- Flight Number vs Launch Site
- Payload vs Launch Site
- Orbit Type vs Flight Number
- Payload vs Orbit Type

BAR CHARTS

- Success Rate vs Orbit Type

LINE CHARTS

- Success Rate vs Year

**This analysis were used to compare relationships between different variables in the dataset

Exploratory Data Analysis [EDA] with SQL

- Loading the Dataset using the IBM DB2 Database
- Query the Data using Python
- Performed different queries (10) to understand the dataset better
- Queries included [Displaying: names of unique launch sites, average payload mass carried by booster version etc……]

Interactive Visual Analysis [IVA]

FOLIUM

- Visualising the Data on Folium was done in the following steps
  - Marking all the launch sites on a map
  - Marking successful and unsuccessful landing on the map
  - Calculating distance from launch sites to key locations (E.g. Railway, Highway and City)

Interactive Dashboard

PLOTLY DASH

- Creating an interactive dashboard with Pie charts and Scatter Plots/Graphs
- Pie chart
  - Used to show distribution of successful launches across all launch sites
  - Shows success/failure ratio for each individual site
- Scatter plot
  - Shows us how success varies across different launch sites, payload mass and booster version

Predictive Analysis [Classification]

Model Development → **Model Evaluation** → **Best Fit Classification**

Steps for model development:

- Loading dataset
- Performing necessary data transformations (standardise and pre-process)
- Split data into training and test data sets, using train_test_split()
- Decide which type of machine learning algorithms are most appropriate
- Creating a GridSearchCV (Logreg, SVM, Decision Tree and KNN Model)
- Fitting the object to the parameters
- Using the training data set to train the model

- Plotting and examining the Confusion matrix
- Checking accuracy
- Checking tuned hyperparameters

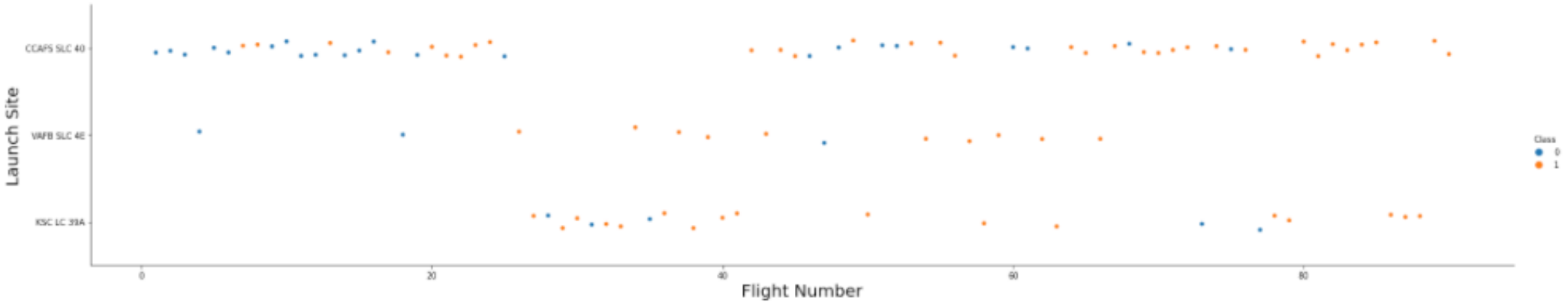
- Review Accuracy Score
- Check which accuracy score is the highest to determine the best performing model

Results and Discussion

- Exploratory Data Analysis
- Interactive Analysis
- Predictive Analysis

Exploratory Data Analysis with Visualization

Launch Site vs Flight Number

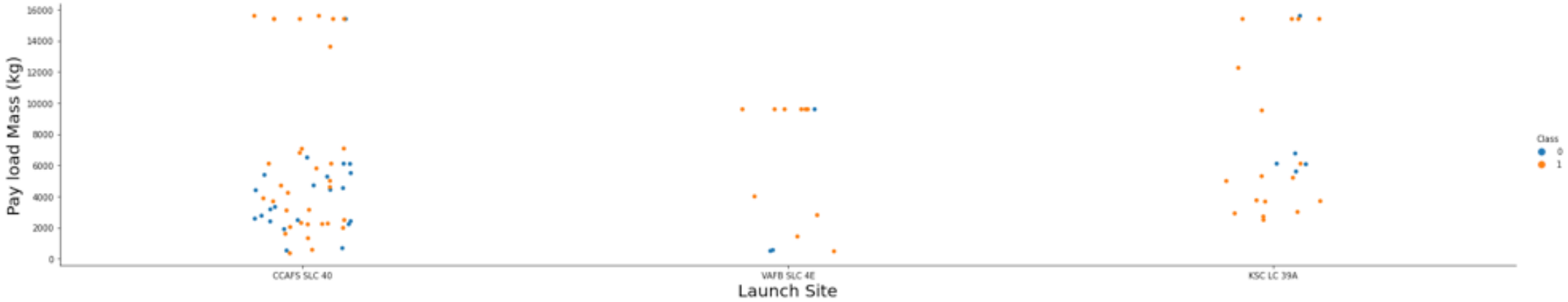


The scatter plot of Launch Site vs. Flight Number shows that:

- Increase in success rate at launch site.
- Most of the early flights that were launched from CCAFS SLC 40 were generally unsuccessful.
- Flights launched from KSC LC 39A were successful.

Exploratory Data Analysis with Visualization

Payload Mass vs Launch Site



The scatter plot of Payload mass vs Launch Site shows that:

- Payload mass above 7000 kg have some successful landing, but little data for this launches
- There is no correlation between payload mass and success rate for launch sites

Exploratory Data Analysis with Visualization

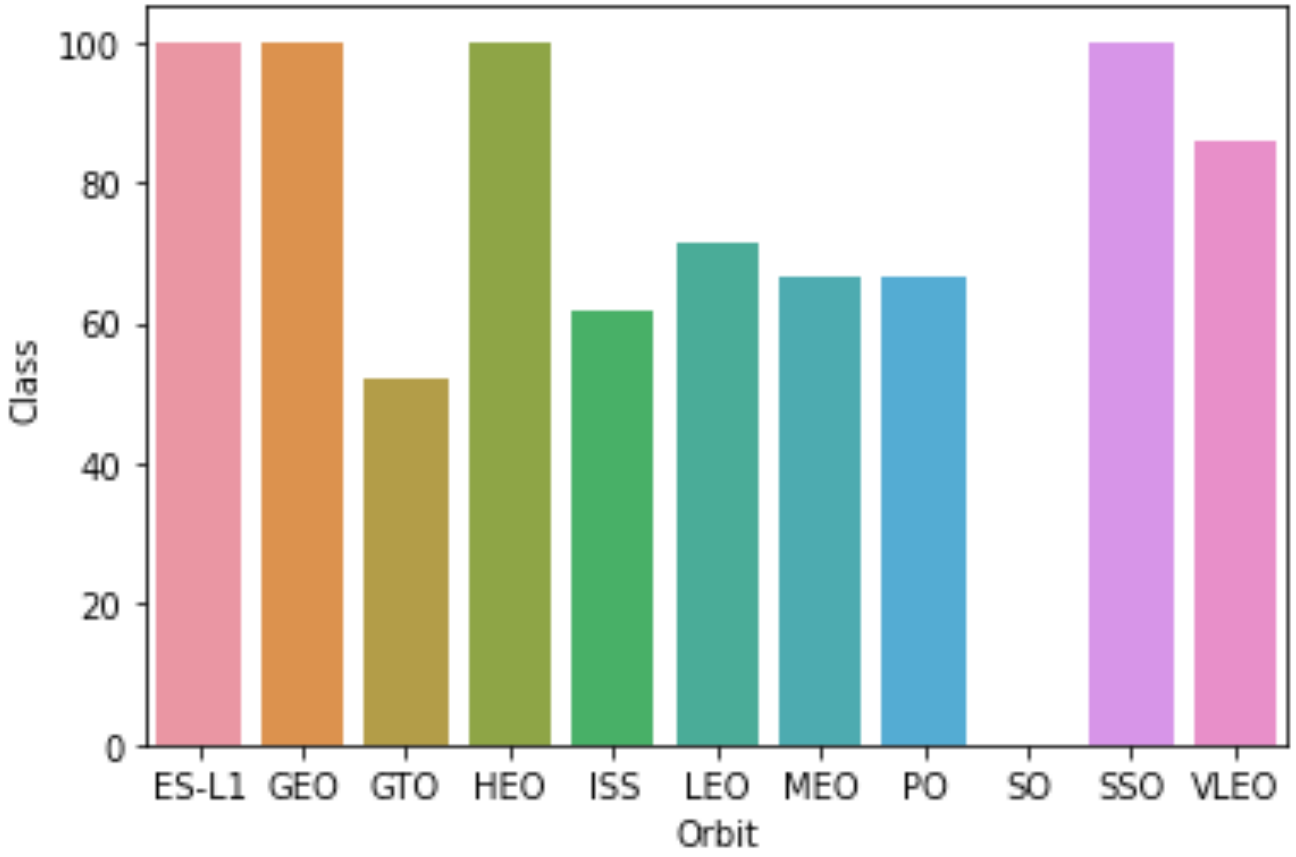
Success rate of each orbit type

- Orbits with 100% success rate

ES-L1 (Earth-Sun First Lagrangian Point),
GEO (Geostationary Orbit),
HEO (High Earth Orbit),
SSO (Sun-synchronous Orbit)

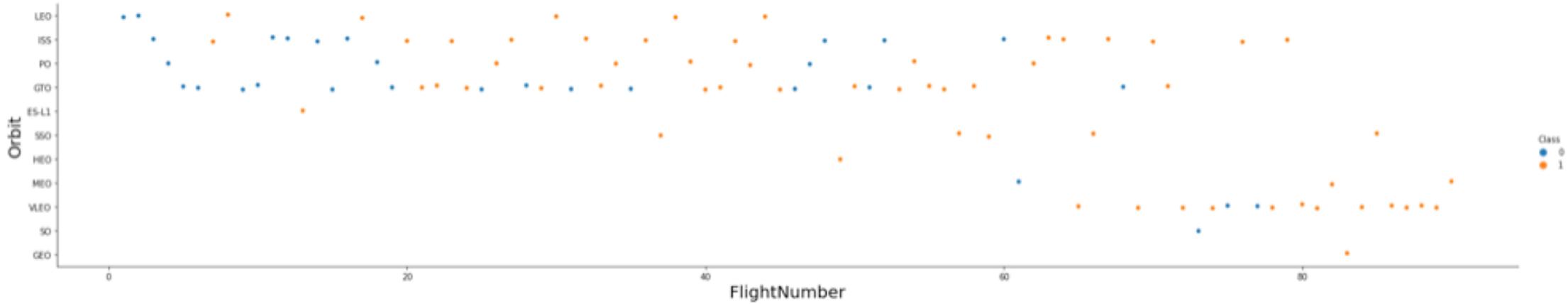
- Orbits with 0% success rate

SO (Heliocentric Orbit)



Exploratory Data Analysis with Visualization

Orbit Type vs Flight Number

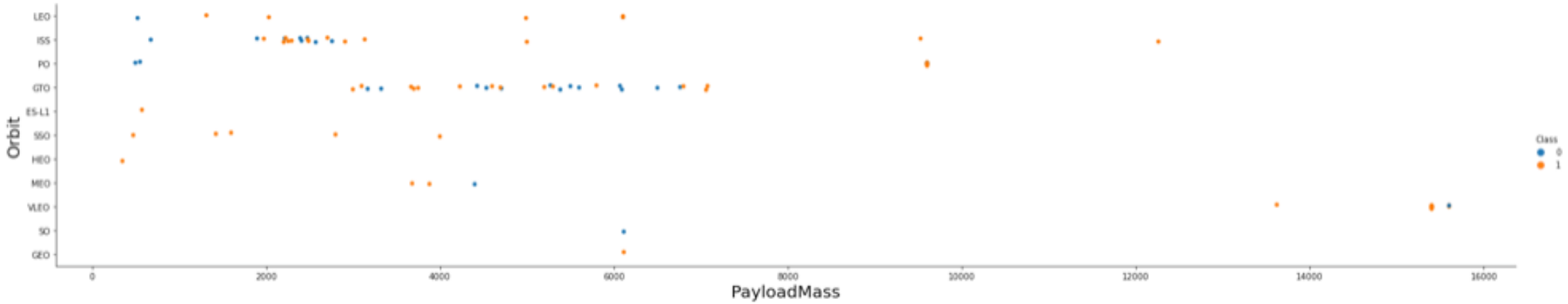


The scatter plot of Orbit Type vs Flight Number shows that:

- The 100% success rate of GEO, HEO, and ES-L1 orbits can be explained by only having 1 flight into the respective orbits.
- Success rate in SSO is more impressive, with 5 successful flights.
- Relationship between Flight Number and Success Rate for GTO is little.

Exploratory Data Analysis with Visualization

Orbit Type vs Payload Mass



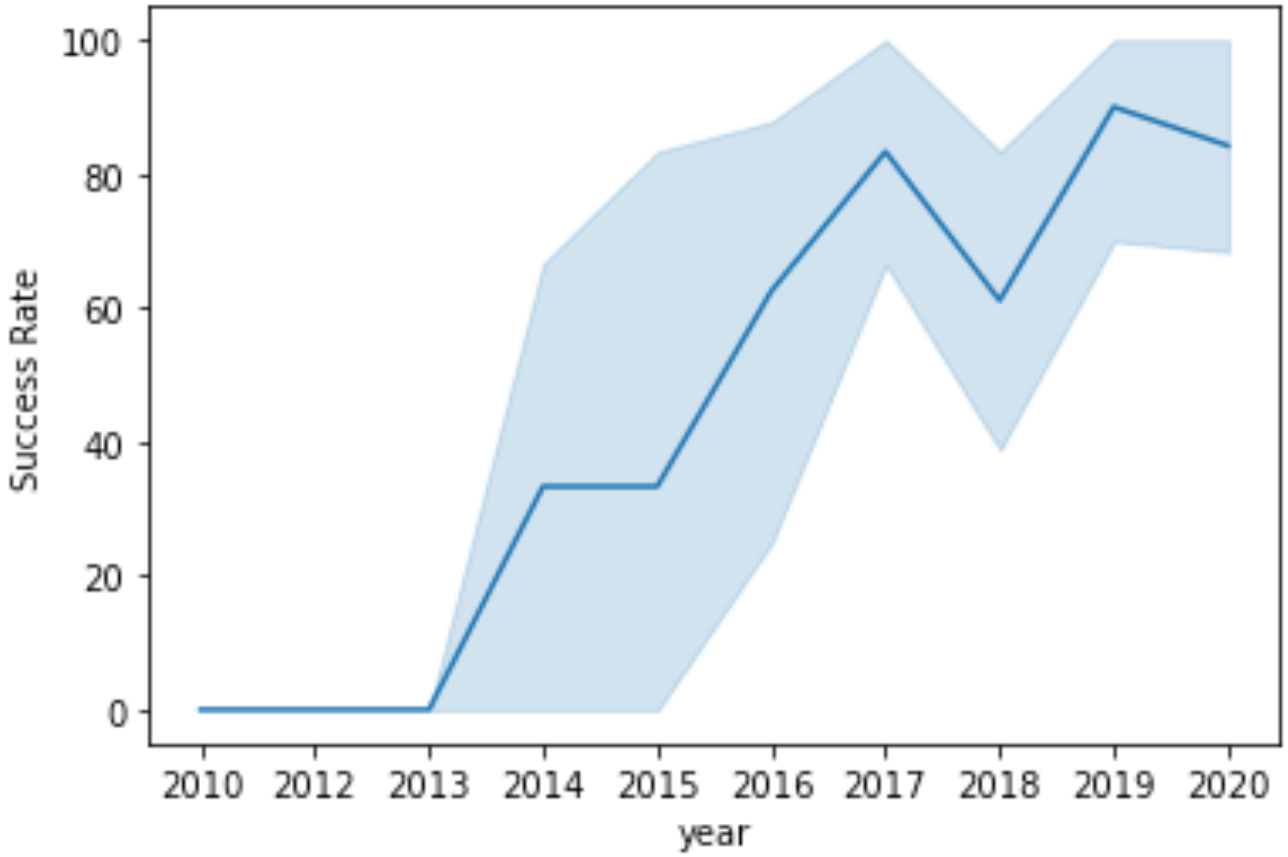
The scatter plot of Orbit Type vs Payload Mass shows that:

- Orbits types (PO, ISS and LEO) have more success with heavy payloads
- Relationship between payload mass and success rate in GTO is unclear.
- VLEO (Very Low Earth Orbit) launches are associated with heavier payloads.

Exploratory Data Analysis with Visualization

Launch Success Yearly Trend

- Between 2010 – 2013, all landings were unsuccessful
- After 2013, success rate for launches increased (minor dips in 2018 and 2020)



Exploratory Data Analysis with SQL

All Launch Name Sites

Find the names of the unique sites.

```
%sql SELECT UNIQUE(LAUNCH_SITE) FROM SPACEXTBL;
```

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

**The word UNIQUE returns only unique values from the LAUNCH_SITE column of the SPACEXTBL table.

Exploratory Data Analysis with SQL

Launch Sites with names beginning with 'CCA'

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

```
%sql SELECT LAUNCH_SITE FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

**LIMIT 5 fetches only 5 records, and the LIKE keyword is used with the wild card 'CCA%' to retrieve string values beginning with 'CCA'.

Exploratory Data Analysis with SQL

Total payload mass carried by Boosters from NASA

```
 %sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEXTBL \ WHERE CUSTOMER = 'N
ASA (CRS)';
```

sum_payload_mass_kg
45596

**The SUM keyword is used to calculate the total of the LAUNCH column, and the SUM keyword (and the associated condition) filters the results to only boosters from NASA (CRS).

Exploratory Data Analysis with SQL

Average Payload Mass Carried by Booster Version F9 v1.1

```
%sql select avg(payload_mass__kg_) as Average from SPACEXDATASET where booster_version like 'F
9 v1.1%'
```

avg_payload_mass_kg
2928

**The AVG keyword is used to calculate the average of the PAYLOAD_MASS__KG_ column, and the WHERE keyword (and the associated condition) filters the results to only the F9 v1.1 booster version.

Exploratory Data Analysis with SQL

First Successful Landing Date

```
%sql select min(date) as Date from SPACEXDATASET where mission_outcome like 'Success'
```

first_success
2015-12-22

**The MIN keyword is used to calculate the minimum of the DATE column, i.e. the first date, and the WHERE keyword (and the associated condition) filters the results to only the successful ground pad landings.

Exploratory Data Analysis with SQL

Names of Boosters with Success in Drone Ship and Payload Mass > 4000 and < 6000.

```
%sql select booster_version from SPACEXDATASET where (mission_outcome like 'Success')
AND (payload_mass__kg_ BETWEEN 4000 AND 6000) AND (landing__outcome like 'Success (drone ship)
')
```

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

**The WHERE keyword is used to filter the results to include only those that satisfy both conditions in the brackets (as the AND keyword is also used). The BETWEEN keyword allows for 4000 < x < 6000 values to be selected.

Exploratory Data Analysis with SQL

Total Number of Successful and Failure Missions

```
%sql SELECT mission_outcome, count(*) as Count FROM SPACEXDATASET GROUP by mission_outcome ORD
ER BY mission_outcome
```

mission_outcome	no_outcome
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

**The COUNT keyword is used to calculate the total number of mission outcomes, and the GROUPBY keyword is also used to group these results by the type of mission outcome.

Exploratory Data Analysis with SQL

Names of Booster_Versions that have carried Maximum Payload Mass

```
maxm = %sql select max(payload_mass__kg_) from SPACEXDATASET
maxv = maxm[0][0]
%sql select booster_version from SPACEXDATASET where payload_mass__kg_=(select max(payload_mass__kg_) from SPACEXDATASET)
```

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

**The SELECT statement within the brackets finds the maximum payload, and this value is used in the WHERE condition. The DISTINCT keyword is then used to retrieve only distinct /unique booster versions.

Exploratory Data Analysis with SQL

Failed Landing Outcomes in Drone Ship for 2015

```
%sql select MONTHNAME(DATE) as Month, landing__outcome, booster_version, launch_site
from SPACEXDATASET where DATE like '2015%' AND landing__outcome like 'Failure (drone ship)'
```

MONTH	landing__outcome	booster_version	payload_mass__kg_	launch_site
January	Failure (drone ship)	F9 v1.1 B1012	2395	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	1898	CCAFS LC-40

**The WHERE keyword is used to filter the results for only failed landing outcomes, AND only for the year of 2015.

Exploratory Data Analysis with SQL

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

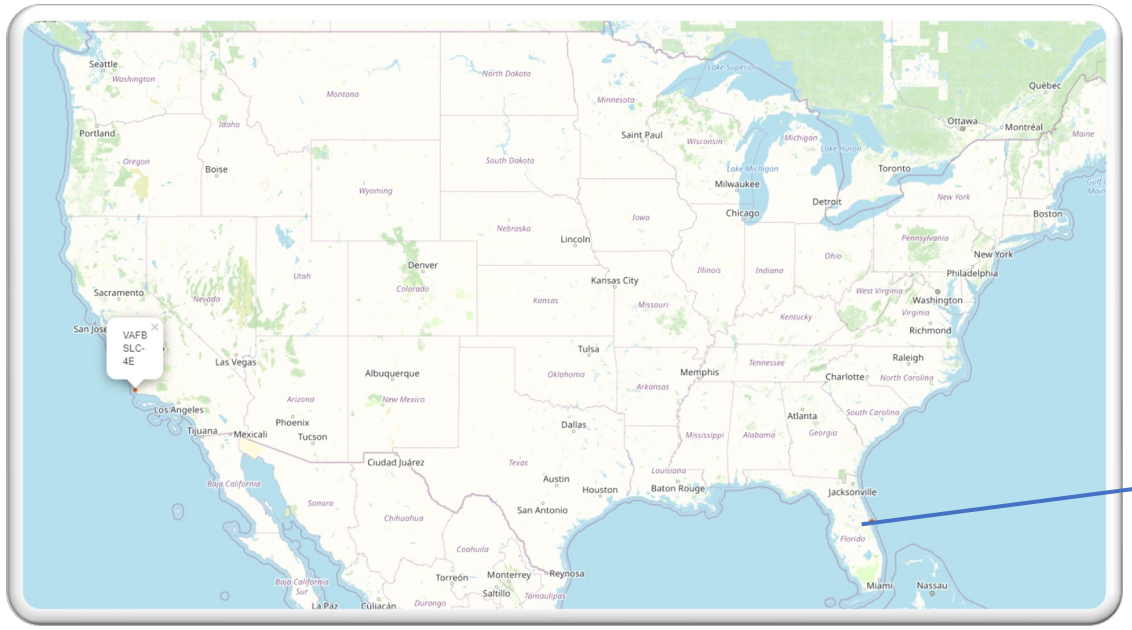
```
%sql select landing__outcome, count(*) as count from SPACEXDATASET
where Date >= '2010-06-04' AND Date <= '2017-03-20'
GROUP by landing__outcome ORDER BY count Desc
```

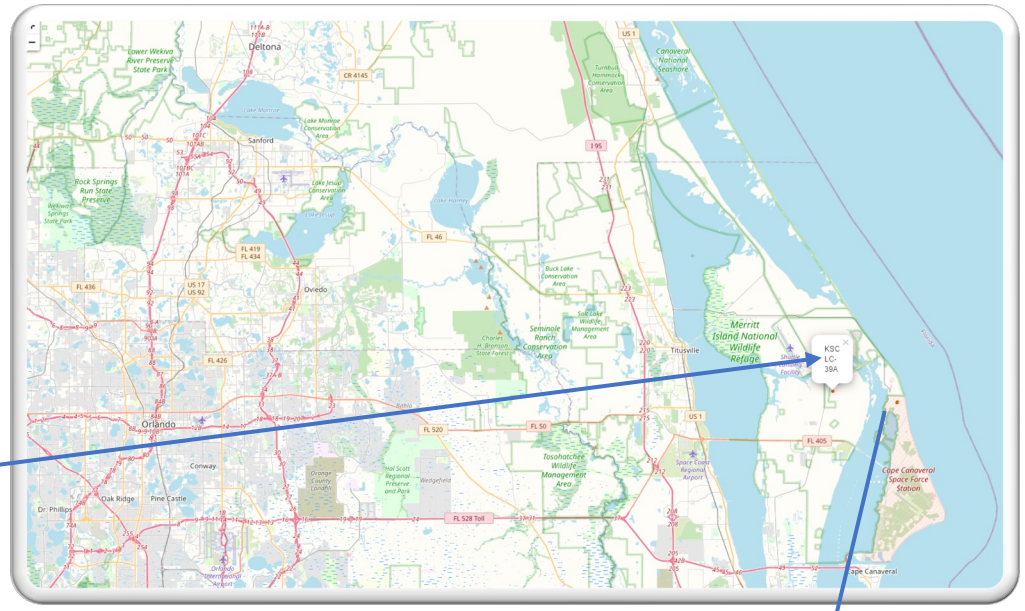
landing__outcome	no_outcome
Success (drone ship)	5
Success (ground pad)	3

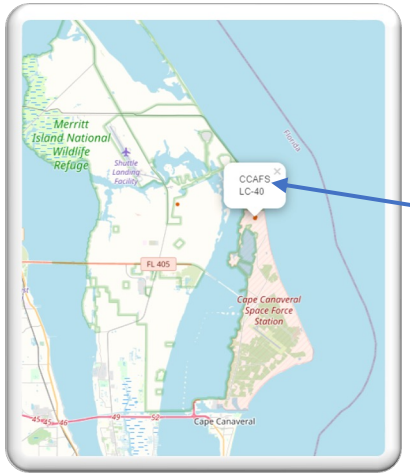
**The WHERE keyword is used with the BETWEEN keyword to filter the results to dates only within those specified. The results are then grouped and ordered, using the keywords GROUP BY and ORDER BY, respectively, where DESC is used to specify the descending order.

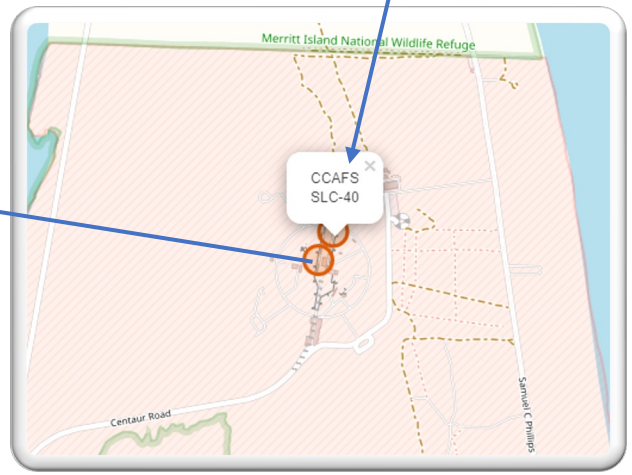
Interactive Analysis with FOLIUM

Launch Site Locations





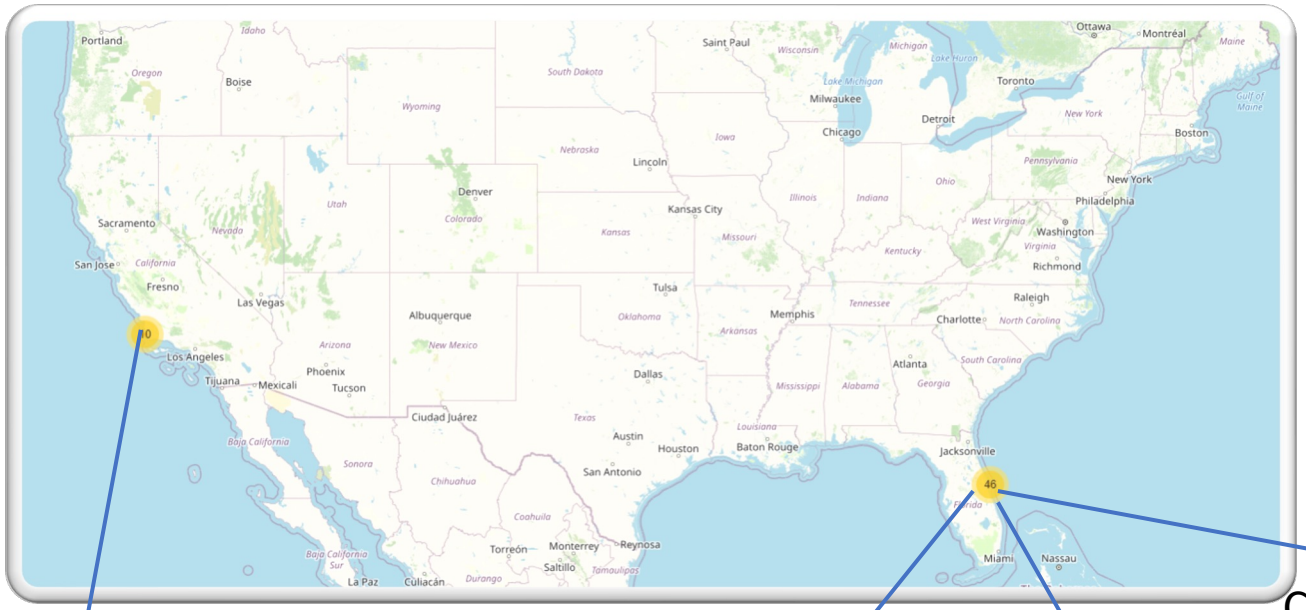




SpaceX launch sites are on coasts of the United States of America, specifically Florida and California.

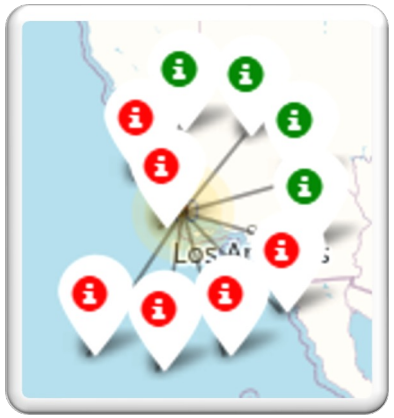
Interactive Analysis with FOLIUM

Success and Failed Launches For Each Site

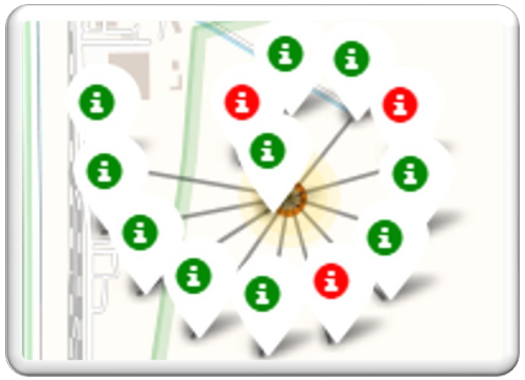


Launches have been grouped into clusters, green icons for successful launches, and red icons for failed launches.

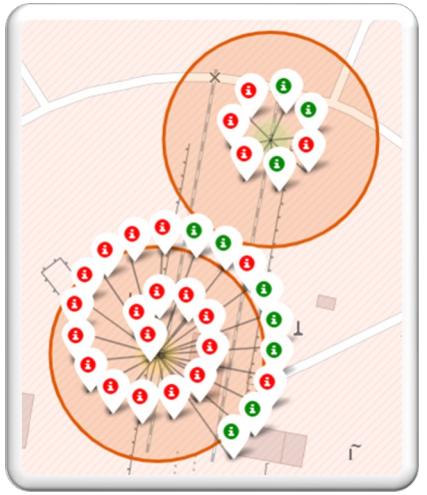
VAFB SLC-4E



KSC LC-39A

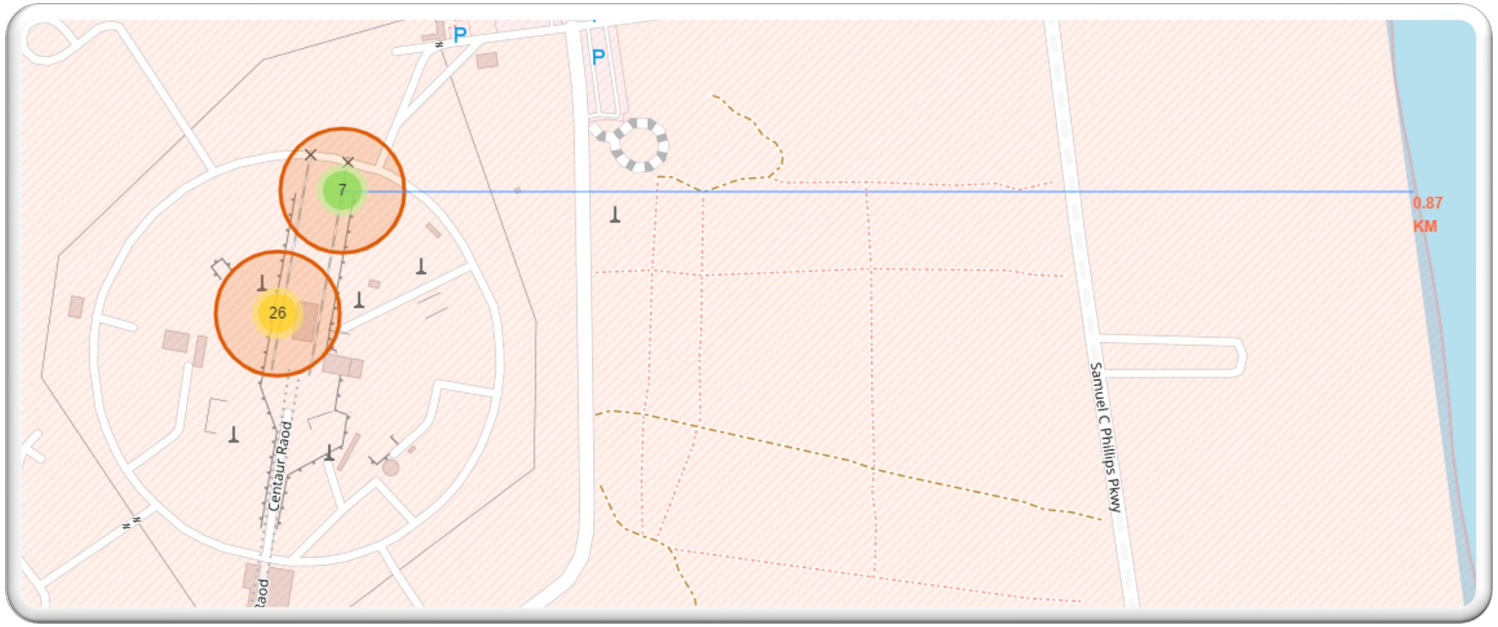


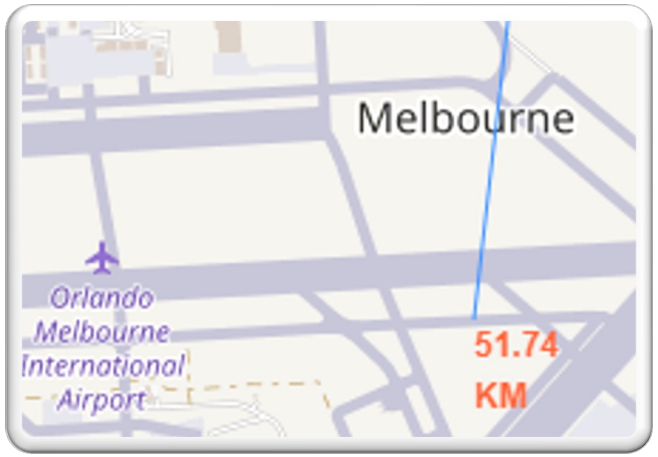
CCAFS SLC-40 and CCAFS LC-40

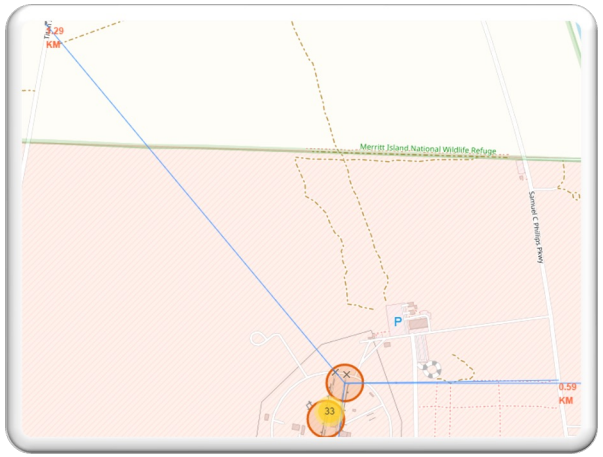


Interactive Analysis with FOLIUM

Location Proximities of Launch Sites to Key Locations







- Launch sites in close proximity to railways? YES.
- Launch sites in close proximity to highways? YES.

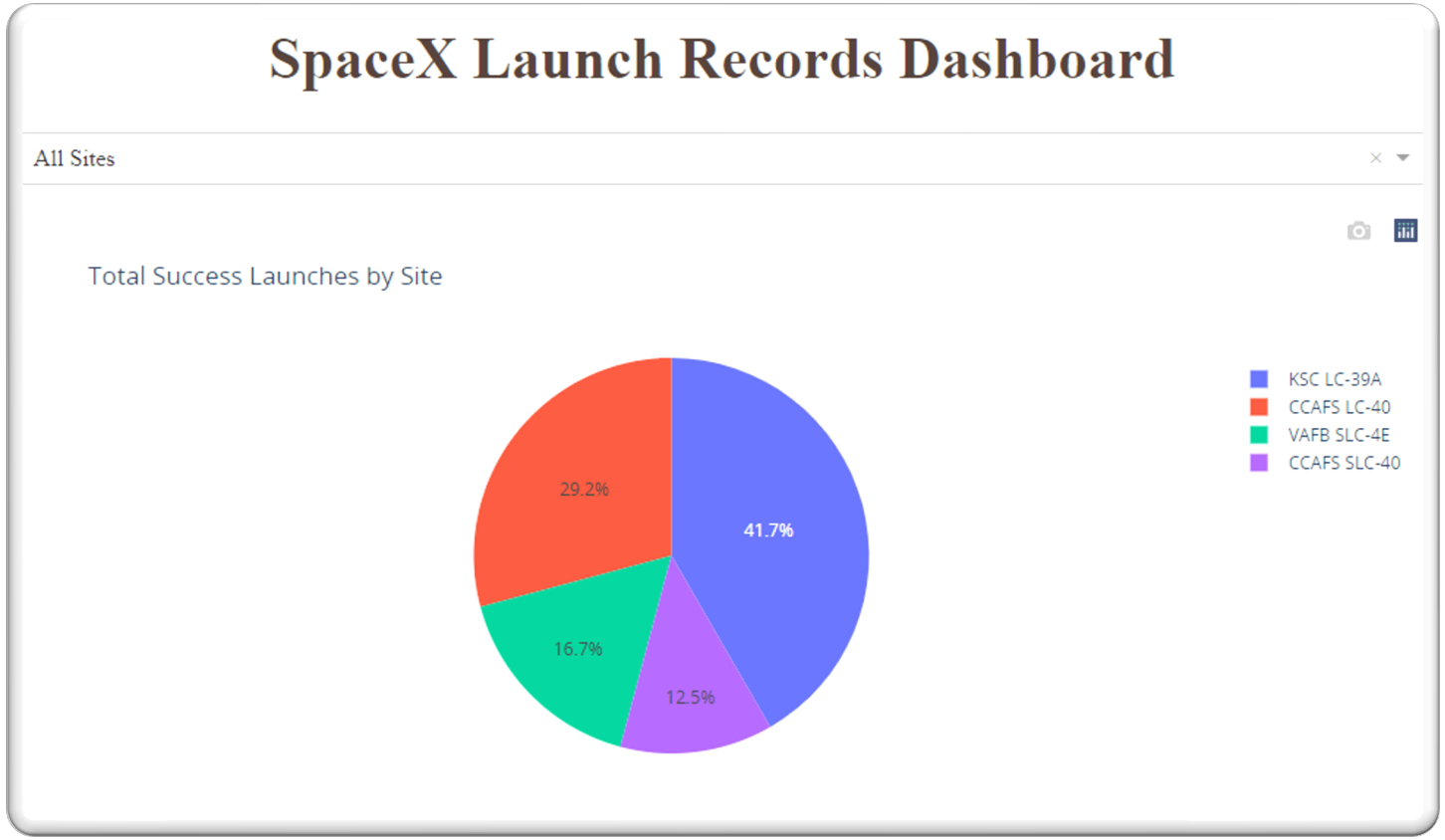
Nearest highway = 0.59km away.

- Launch sites in close proximity to railways? YES.

Nearest railway = 1.29 km away.

Interactive Dashboard with PLOTLY DASH

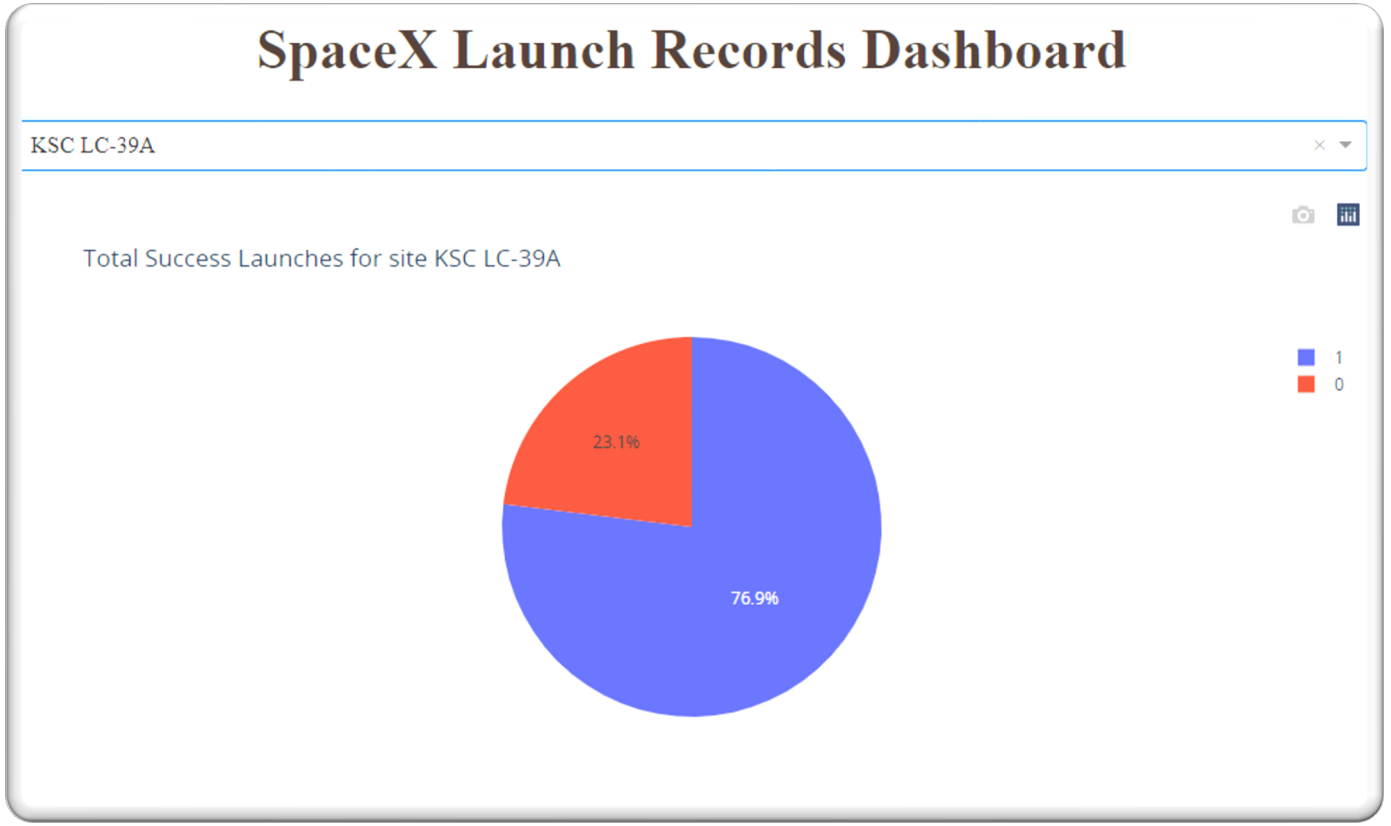
Launch Success Count for All Sites



- The launch site KSC LC-39 A had the most successful launches, with 41.7% of the total successful launches.

Interactive Dashboard with PLOTLY DASH

Highest Launching Success Ratio



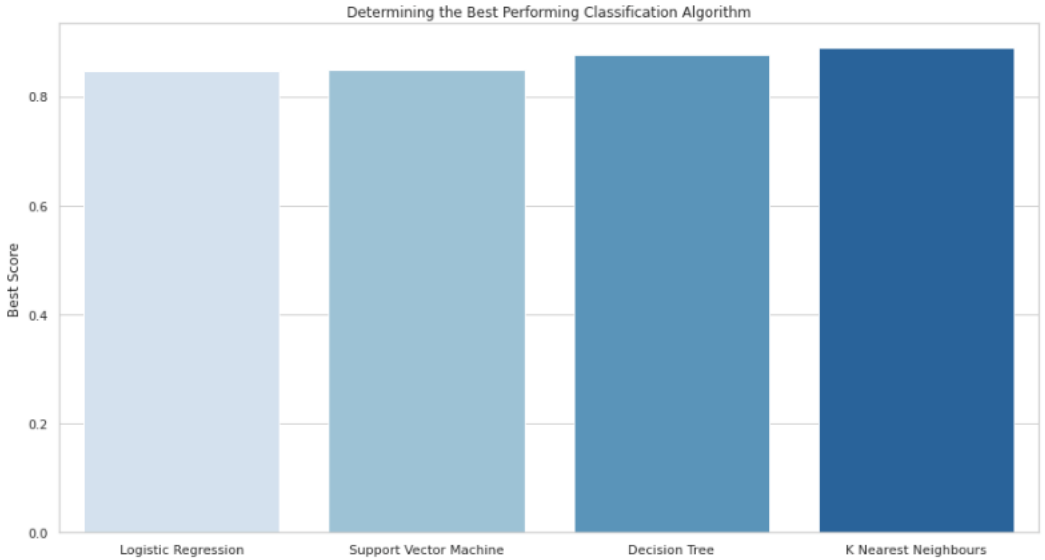
- Launch site KSC LC-39 also has the highest ratio success ratio with a ratio of 76.9%.

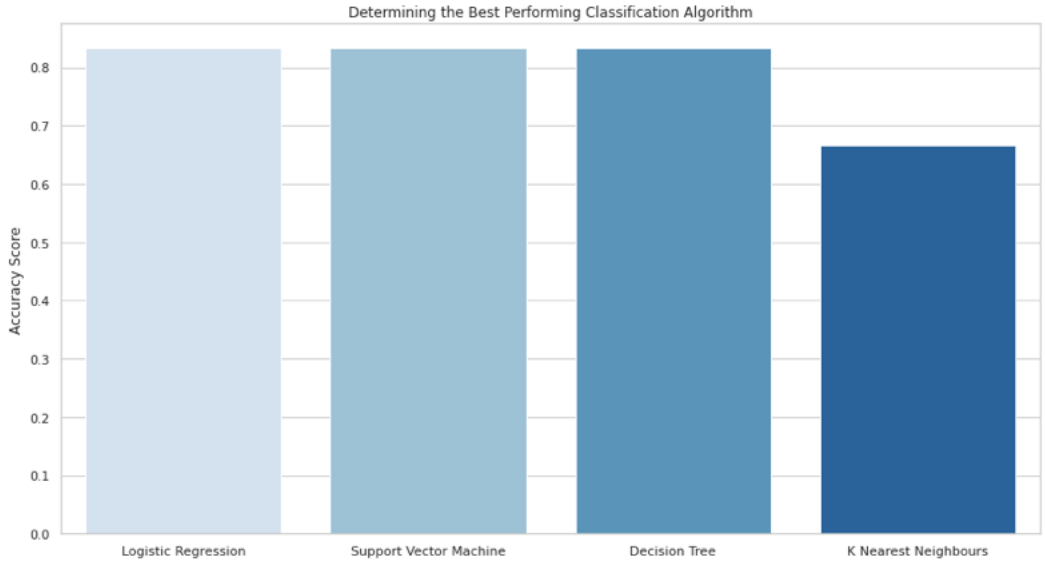
Interactive Dashboard with PLOTLY DASH

Payload Mass vs Success vs Booster version Cateegory

Predictive Analysis - Classification

Classification Accuracy





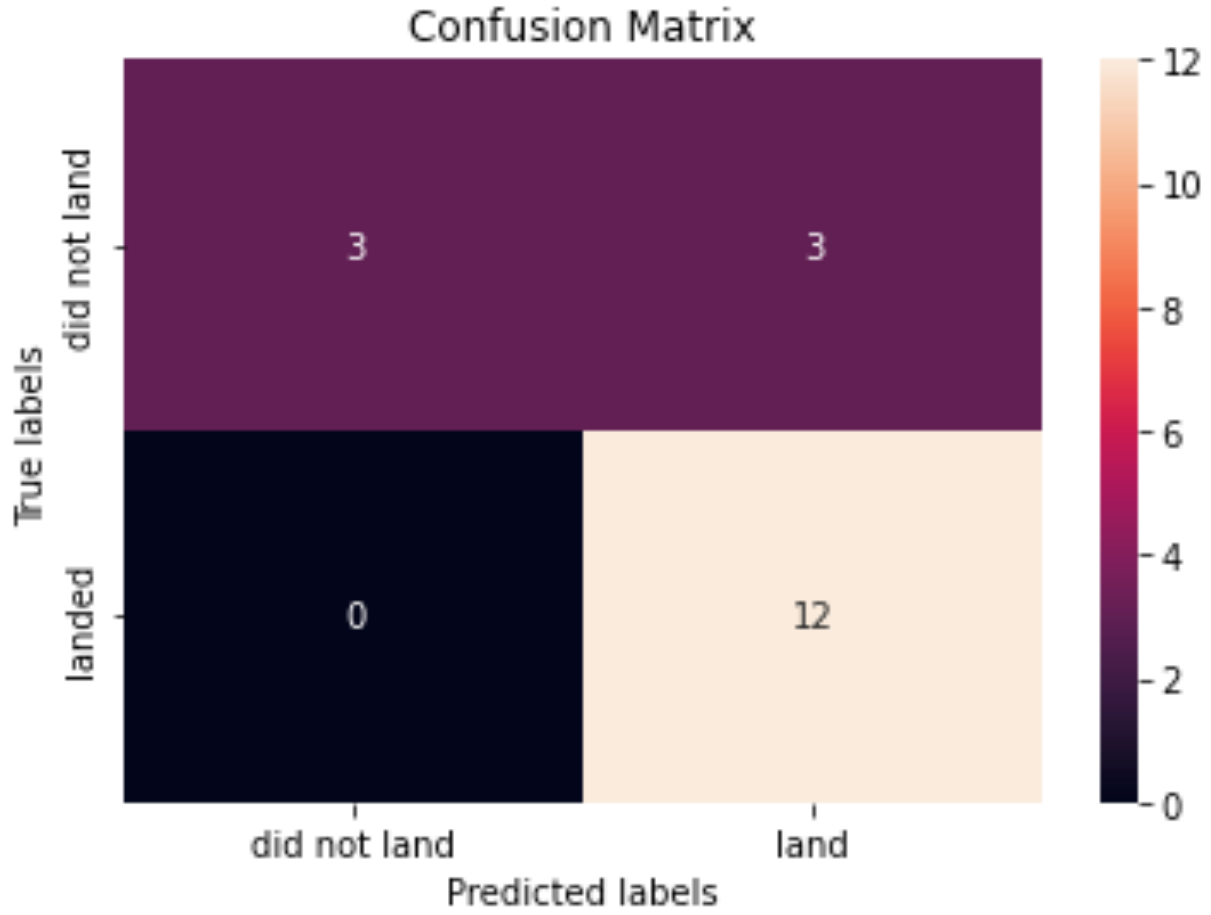
	Algorithm	Accuracy Score	Best Score
0	Logistic Regression	0.833333	0.846429
1	Support Vector Machine	0.833333	0.848214
2	Decision Tree	0.833333	0.876786
3	K Nearest Neighbours	0.666667	0.889286

- The Decision Tree model has the highest classification accuracy

Predictive Analysis - Classification

Confusion Matrix

- The models predicted 12 successful landings when the true label was successful landing.
- The models predicted 3 unsuccessful landings when the true label was unsuccessful landing.
- The models predicted 3 successful landings when the true label was unsuccessful landings (false positives).



Conclusion

Conclusion

- As the number of flights increased, the rate of success at a launch site increased.
- Most of the early flights were unsuccessful.
- Between 2010 and 2013, all landings were unsuccessful
- After 2013, the success rate generally increased, despite small dips in 2018 and 2020.
- Orbit types ES-L1, GEO, HEO, and SSO, have the highest success rate of 100%.
- Launch site KSC LC-39 A had the most successful launches, with 41.7% of the total successful launches, and also the highest rate of successful launches, with a 76.9% success rate.

Appendix

- https://github.com/dheyveid/IBM-Data-Science-Capstone-Project