IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Giuseppe Palermino

24 March 2022

Outline

IBM Data Science Capstone Project – SpaceX

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- **Summary of Methodologies**
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)

- **Summary of All Results**
  - Exploratory data analysis results
  - Interactive analytics in screenshots
  - Predictive analytics results

Introduction

- **Project background and context**

  SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. The goal of this project is to create a machine learning pipeline to predict if the first stage will land successfully.

- **Problems needed to be answered**

  - With what factors the rocket will land successfully?
  - The interaction among various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

- **Data collection methodology:**
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- **Perform data wrangling:**
  - One-hot encoding was applied to categorical features, for Machine Learning and dropping irrelevant columns.
- **Perform exploratory data analysis (EDA) using visualization and SQL:**
  - Plotting : Scatter Graphs, Bar Graphs to show relationships between variables to show patterns of data.
- **Perform interactive visual analytics using Folium and Plotly Dash**
- **Perform predictive analysis using classification models:**
  - Build, tune, and evaluate classification models

Data Collection

The data was collected using two methods:

SpaceX API

- Data collection was done using get request to the SpaceX API. We worked with SpaceX launch data that is gathered from the SpaceX REST API. We requested and parsed the SpaceX launch data using the GET request
- This API provided launches data including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.

Web Scrapping

- We performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
- The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection SpaceX API

We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

Use SpaceX REST API → API returns SpaceX data in .JSON → Filter DF for Falcon 9 only → Data Cleaning → Normalize data into flat data file such as .csv

GitHub URL

1 .Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url).json()
```

2. Converting Response to a .json file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Apply custom functions to clean data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
getBoosterVersion(data)
```

4. Assign list to dictionary then dataframe

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
data= pd.DataFrame(launch_dict)
```

5. Filter dataframe and export to flat file (.csv)

```
data_falcon9 = data[data.BoosterVersion == 'Falcon 9']
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection Web Scrapping

We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup. Then, we parsed the table and converted it into a pandas dataframe.

Get HTML Response from Wikipedia → Extract data using beautiful soup → Parse HTML table into a List -> Dictionary → Data Cleaning → Normalize data into flat data file such as .csv

[GitHub URL]

1 .Getting Response from HTML

```
data  = requests.get(static_url).text
```

2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(data, 'html5lib')
```

3. Finding tables

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

4. Getting column names

```
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
# Append the Non-empty column name (`if name is not None and len(name) > 0`
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

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


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'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Appending data to keys (refer) to notebook block 12

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
             # get table row
             for rows in table.find_all("tr")
                 #check to see if first table
```

7. Converting dictionary to dataframe

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

8. Dataframe to .CSV

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.

- In the data set, there are several different cases where the booster did not land successfully:
  - ✓ True Ocean means the mission outcome was successfully landed to a specific region of the ocean
  - ✓ True RTLS means the mission outcome was successfully landed to a ground pad
  - ✓ True ASDS means the mission outcome was successfully landed on a drone

We mainly convert the outcomes into Training Labels with 1 means the booster successfully Landed and 0 means it was unsuccessful.

GitHub URL

Perform Exploratory Data Analysis (EDA)

~~Calculate the number of~~ launches at each site

~~Calculate the number and~~ occurrence of each orbit

Calculate the number and occurrence of mission outcome per orbit type

Export dataset as .CSV

Create a landing outcome label from Outcome column

Work out success rate for every landing in dataset

Data Wrangling

Each launch aims to dedicated orbit, and here are some common orbit types:

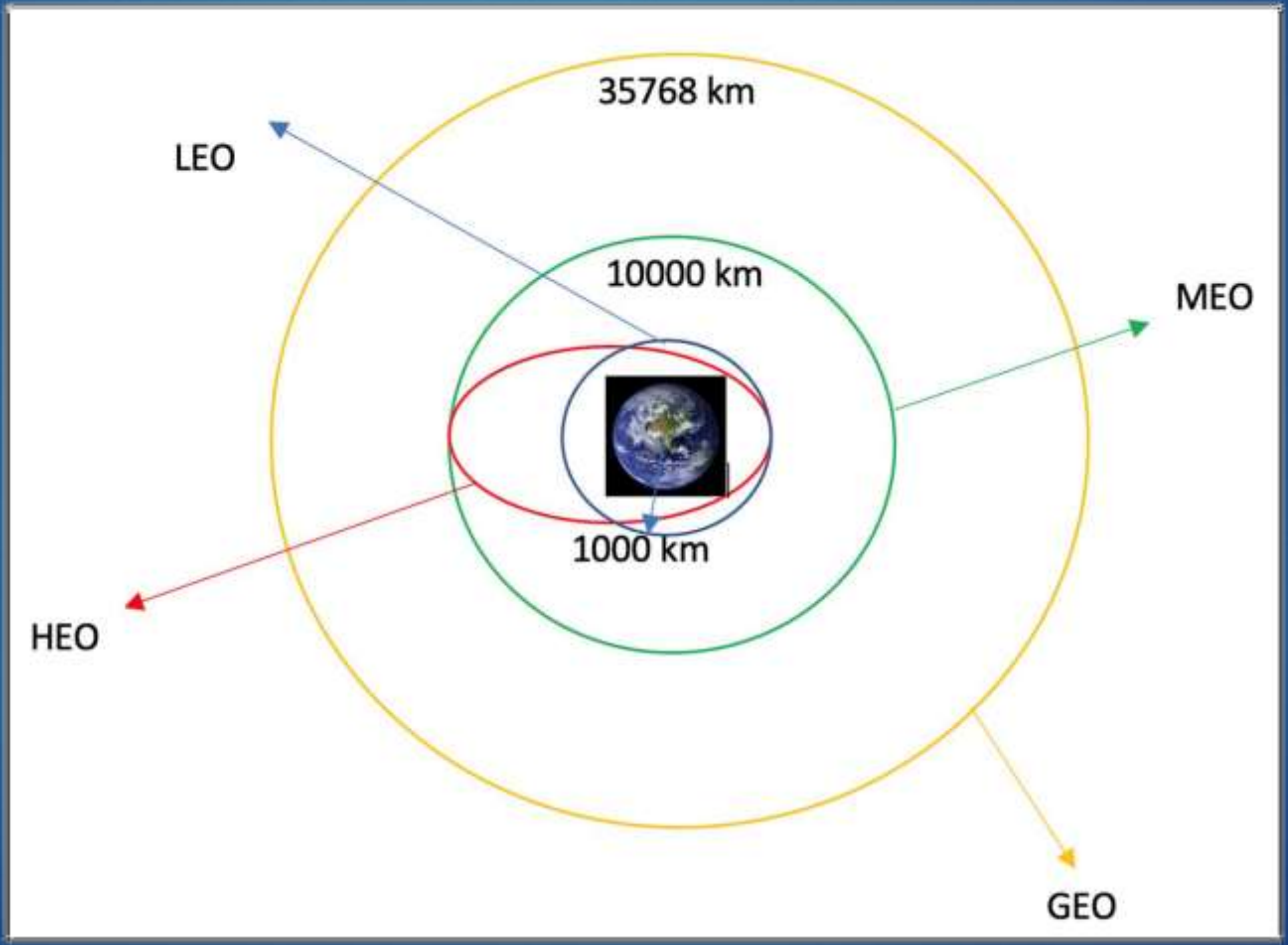


Diagram showing common orbit types SpaceX uses

EDA with Data Visualization

Scatter Graphs being drawn:

Flight Number VS. Payload Mass

Flight Number VS. Launch Site

Payload VS. Launch Site

Orbit VS. Flight Number

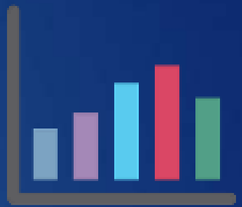
Payload VS. Orbit Type

Orbit VS. Payload Mass

Scatter plots show how much one variable is affected by another. The relationship between two variables is called correlation. Scatter plots usually consist of a large body of data.

GitHub URL

Bar Graph being drawn:



Mean VS. Orbit

A bar diagram makes it easy to compare sets of data between different groups at a glance. The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time.

Line Graph being drawn:

Success Rate VS. Year

Line graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded

EDA with SQL

Performed SQL queries to gather information about the dataset.

We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:

1. Displaying the names of the unique launch sites in the space mission
2. Displaying 5 records where launch sites begin with the string 'KSC’
3. Displaying the total payload mass carried by boosters launched by NASA (CRS)
4. Displaying average payload mass carried by booster version F9 v1.1
5. Listing the date where the successful landing outcome in drone ship was achieved.
6. Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
7. Listing the total number of successful and failure mission outcomes
8. Listing the names of the booster_versions which have carried the maximum payload mass.
9. Listing the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
10. Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

GitHub URL

Build an Interactive Map with Folium

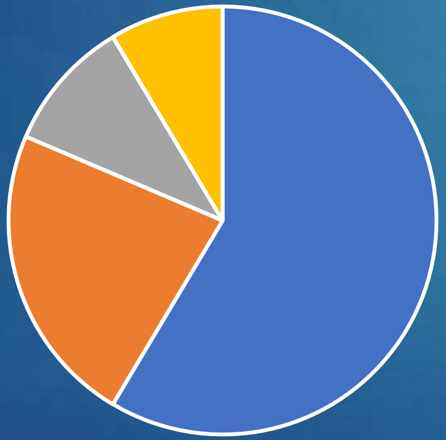
- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1 (.i.e., 0 for failure, and 1 for success).
- Using the color-labeled marker clusters (Green and Red), we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities using Haversine’s formula. We answered some questions, for instance:
  - 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

GitHub URL

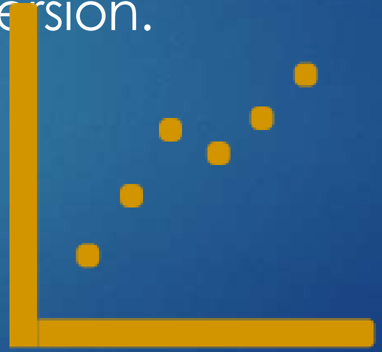
Build a Dashboard with Plotly Dash

We built an interactive dashboard with Plotly dash.

We plotted **Pie Chart** showing the total launches by a certain site / all sites.



- We plotted **Scatter Graph** showing the relationship with
  - Outcome and Payload Mass (Kg) for the different booster version.



Predictive Analysis (Classification)

BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

EVALUATING MODEL

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

IMPROVING MODEL

- Feature Engineering
- Algorithm Tuning

BEST PERFORMING CLASSIFICATION MODEL

- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

GitHub URL

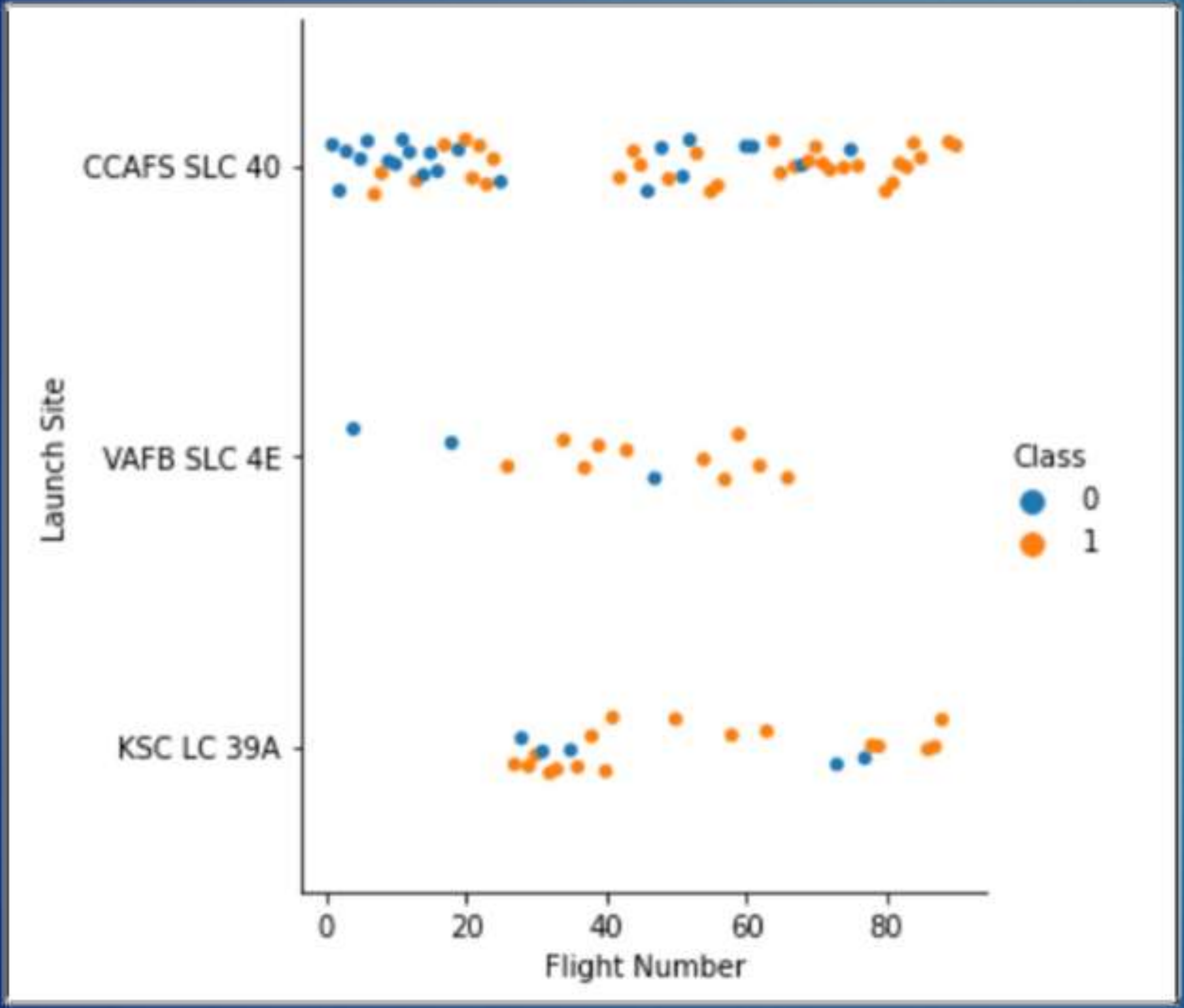
Results

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

Section 2

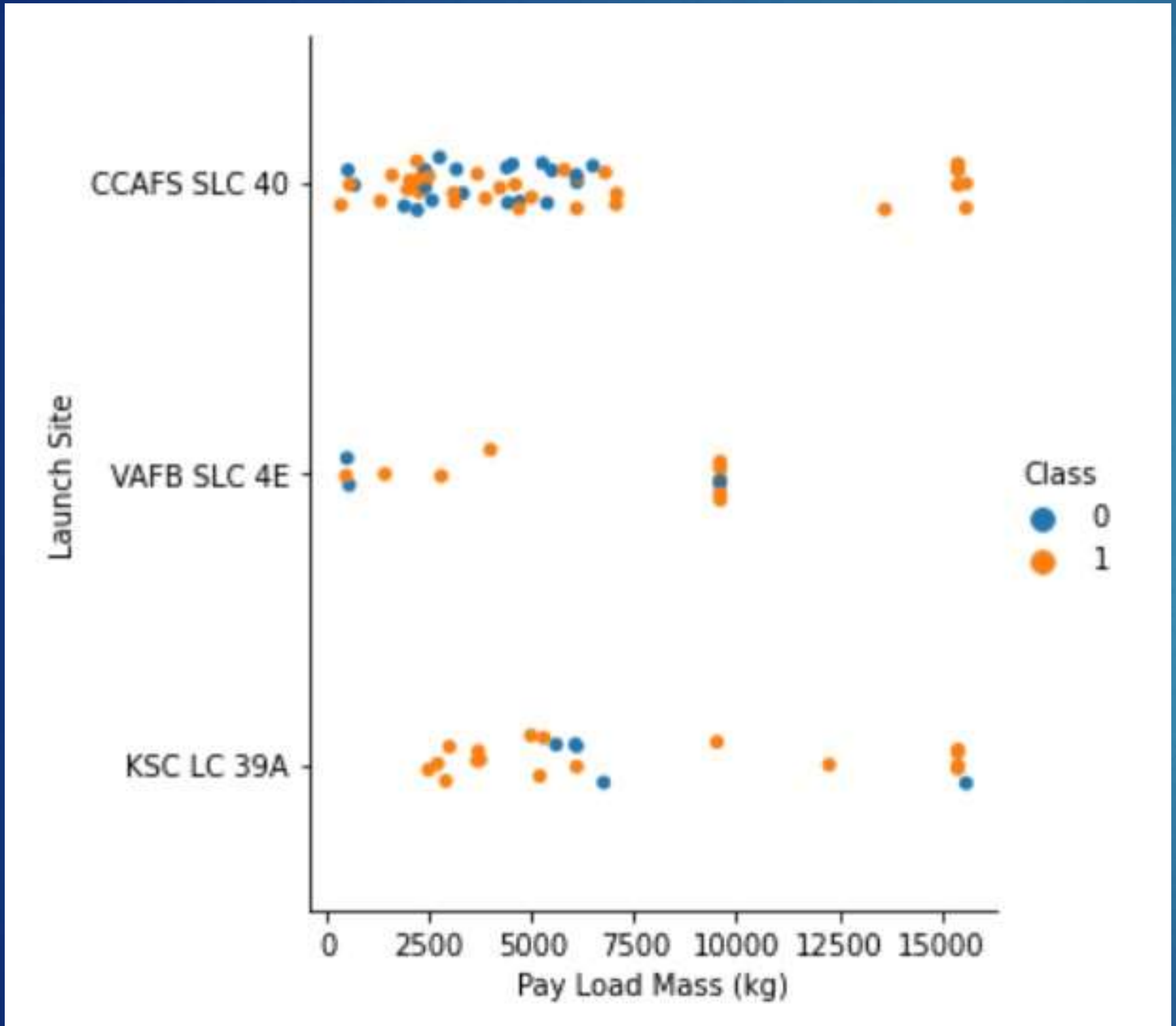
Insights drawn from EDA

Flight Number vs. Flight Site



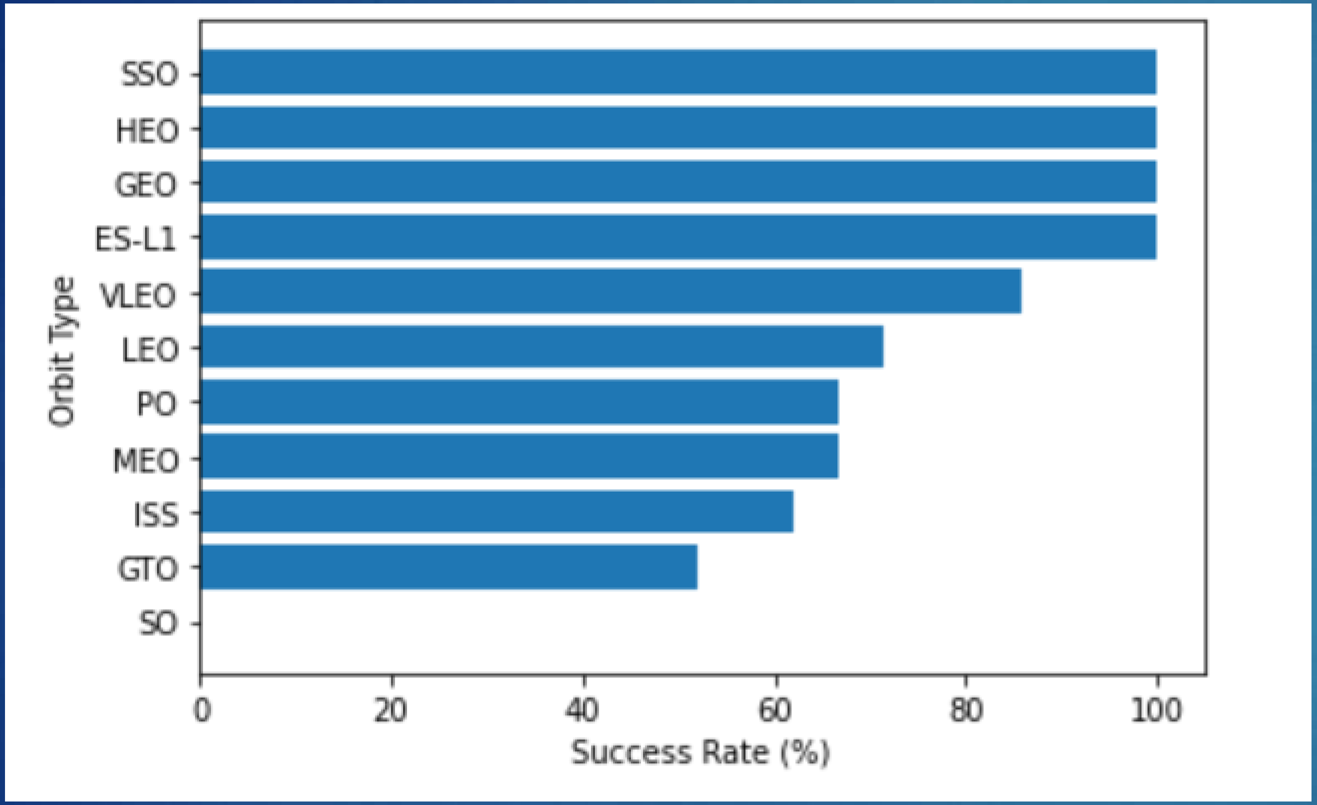
From the plot, we can found that the larger the flight amount at a launch site, the greater the success rate at a launch site.

Payload vs. Launch Site



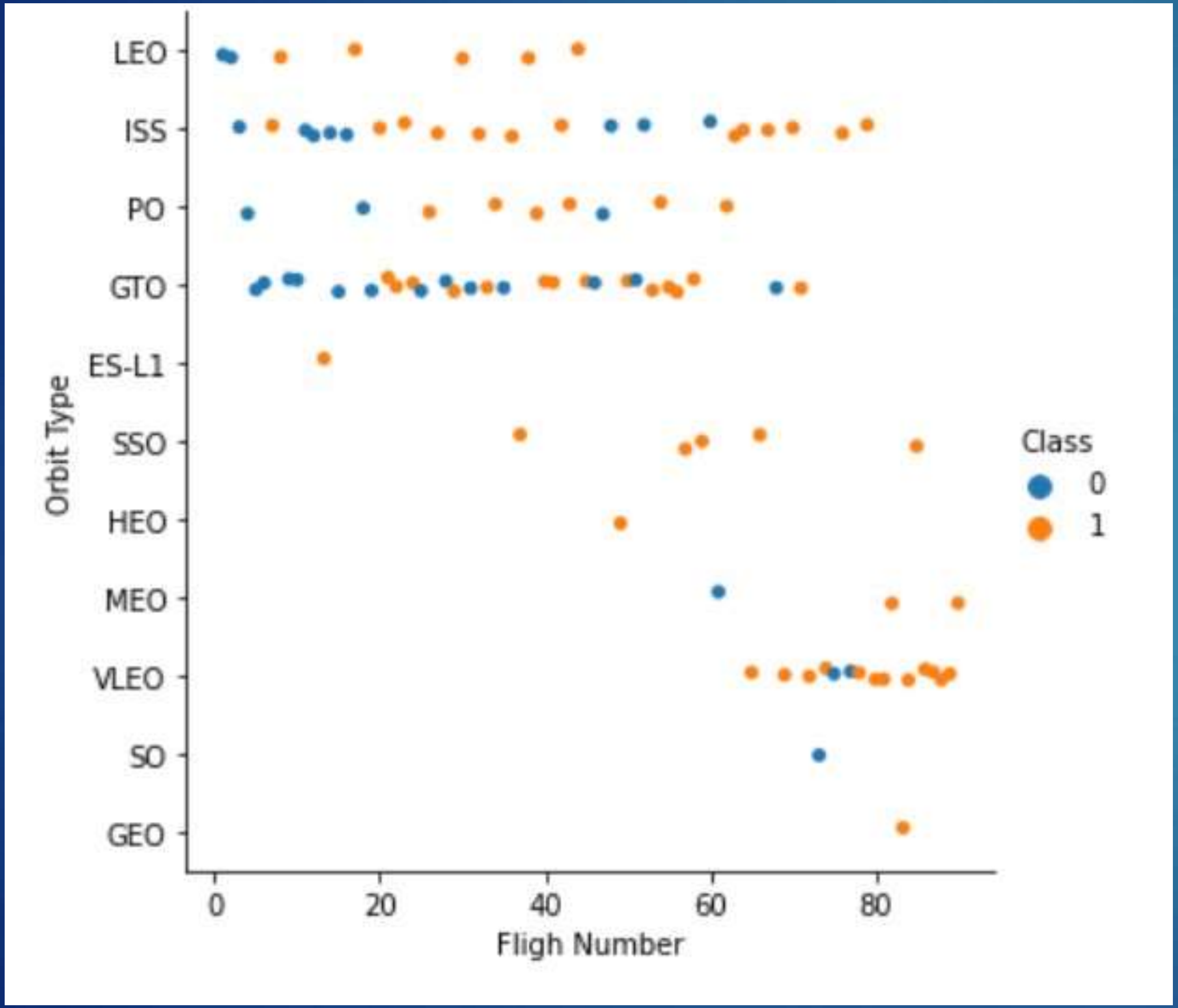
From the plot, we can found that The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket.

Success Rate vs. Orbit Type



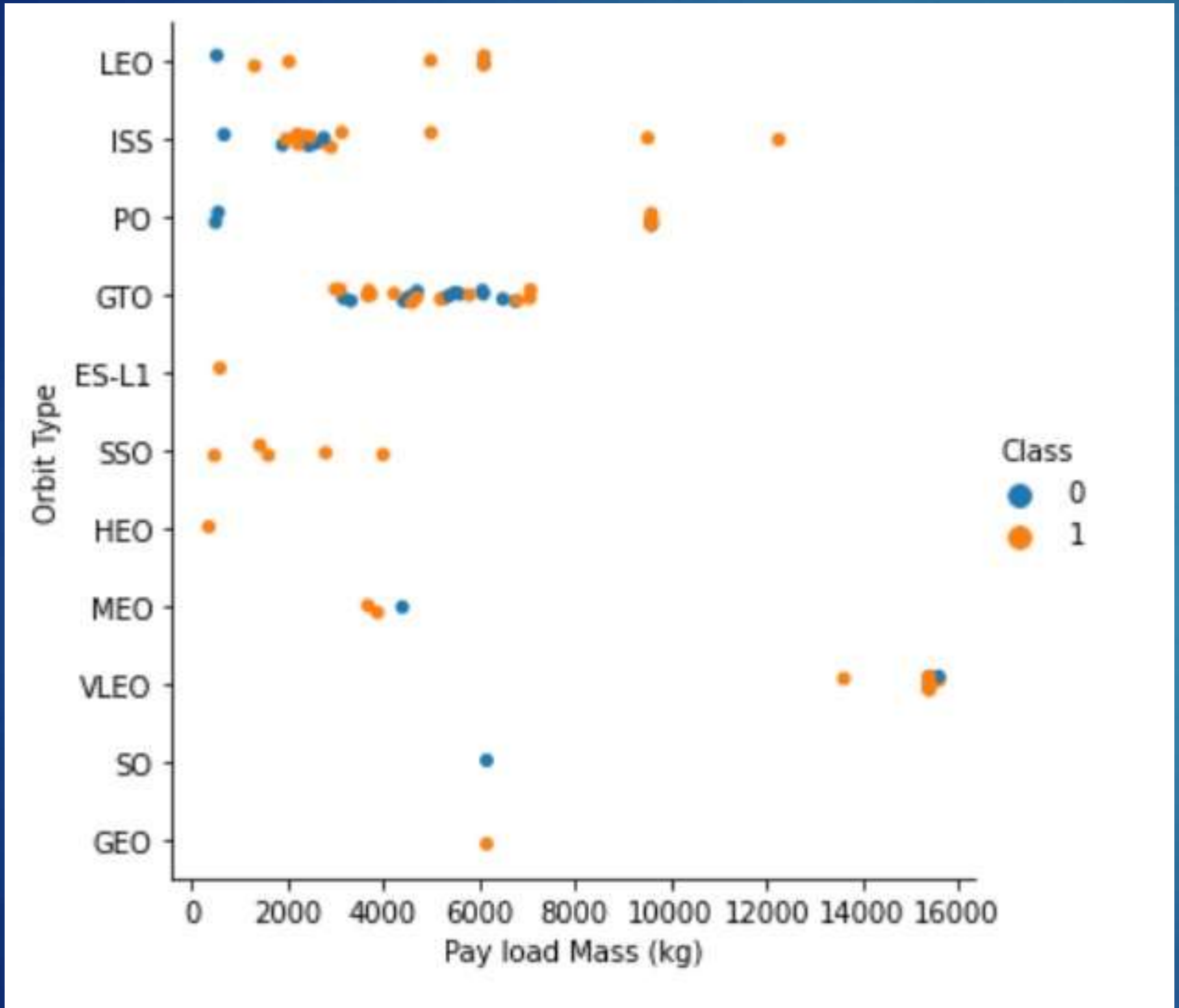
From the plot, we can found that Orbit GEO, HEO, SSO, and ES-L1 has the best Success Rate

Flight Number vs. Orbit Type



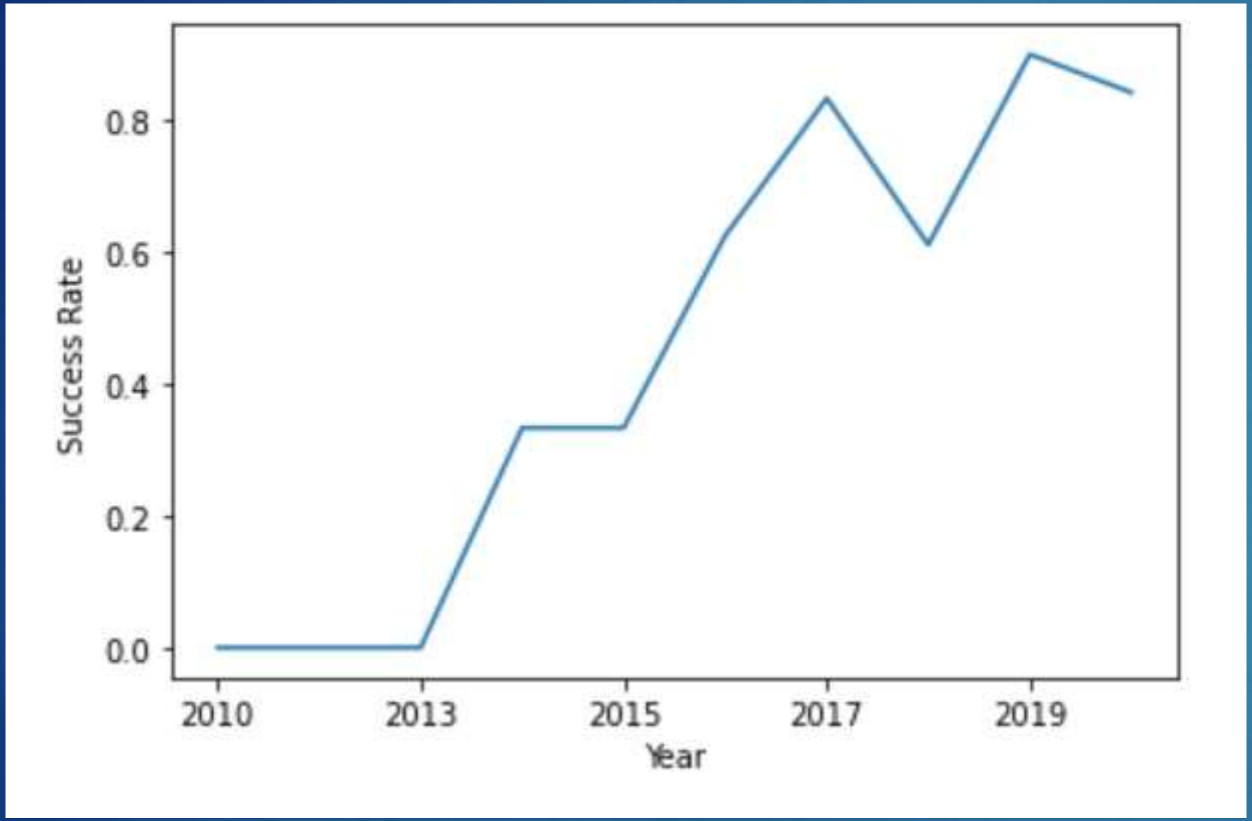
From the plot, we can found that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

Payload vs. Orbit Type



From the plot, we can found that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.

Launch Success Yearly Trend



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

All Unique Launch Site Names

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

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL;
```

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

Launch Site Names Begin with 'CCA'

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

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	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
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

We used the query above to display 5 records where launch sites begin with `CCA`.

Total Payload Mass

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

```sql
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) AS total_payload_mass_kg
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)';
```

total_payload_mass_kg
45596

Average Payload Mass by F9 v1.1

We calculated the average payload mass carried by booster version F9 v1.1 as 2928.

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) AS avg_payload_mass_kg
FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

avg_payload_mass_kg
2928

First Successful Ground Landing Date

We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

```
%%sql
SELECT MIN(DATE) AS first_successful_landing_date
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

first_successful_landing_date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (drone ship)'
    AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000);
```

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

Total Number of Successful and Failure Mission Outcomes

```
%%sql
SELECT MISSION_OUTCOME, COUNT(*) AS total_number
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME;
```

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

Boosters Carried Maximum Payload

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

```
%%sql
SELECT DISTINCT BOOSTER_VERSION, PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL);
```

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

2015 Launch Records

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

```
%%sql
SELECT LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND YEAR(DATE) = '2015';
```

landing_outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

```
%%sql
SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS total_number
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING__OUTCOME
ORDER BY total_number DESC
```

landing__outcome	total_number
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

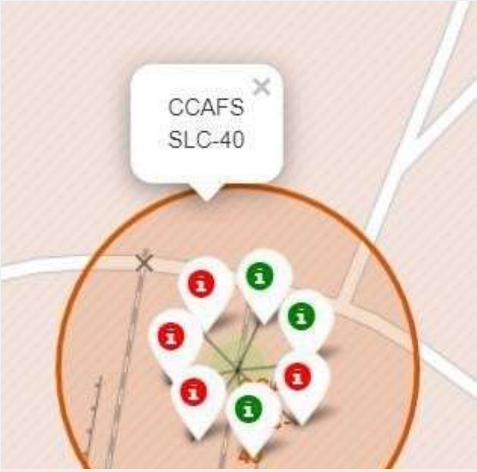
- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

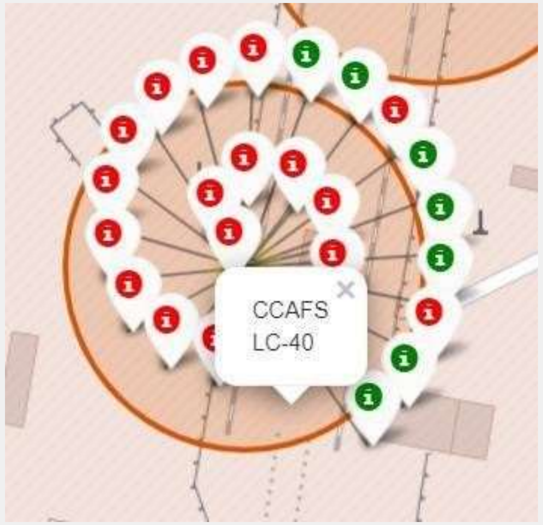
Section 3

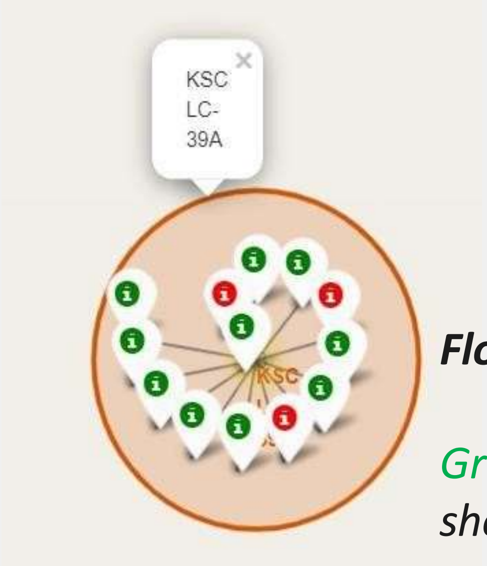
Launch Sites Proximities Analysis

All Launch Sites Global Map Markers

Markers Showing Launch Sites With Color Labels





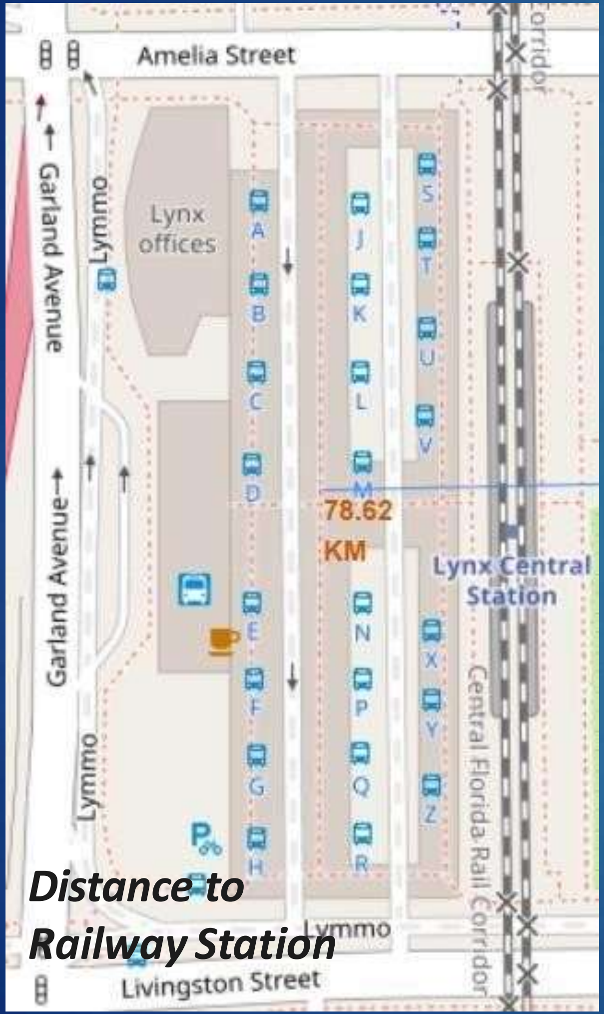


Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

Launch Site Distance to Landmarks

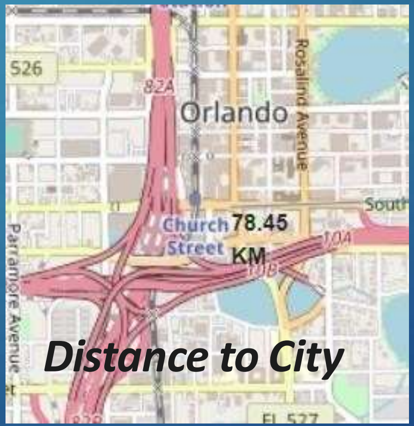


Distance to Railway Station

Distance to closest Highway

Distance to coast

Distance to Coastline



Distance to City

- 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

Section 5

Predictive Analysis (Classification)

Classification Accuracy

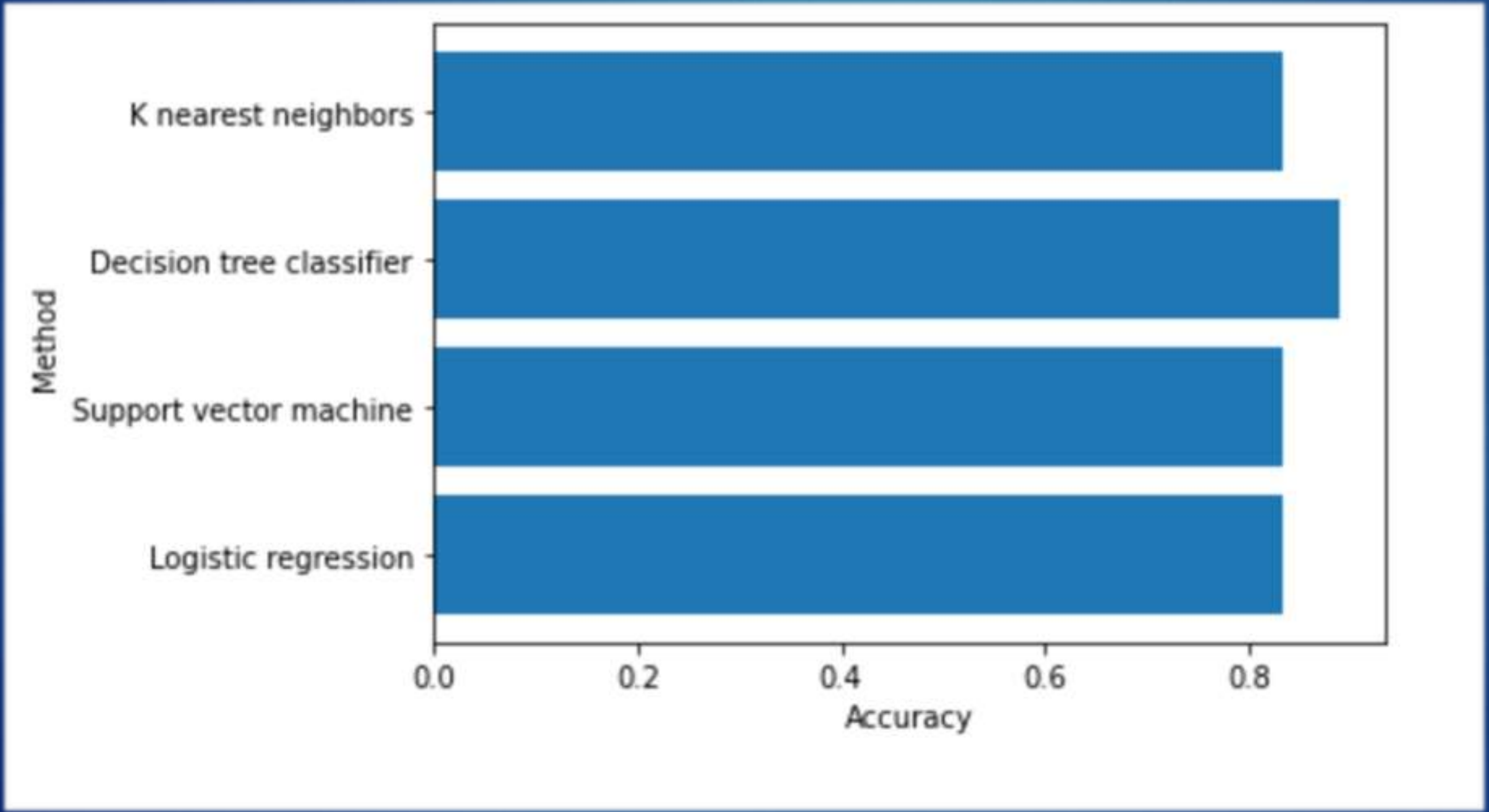
The method which performs best is " Decision Tree " with a score of (0.8875)

```
algorithms = {'KNN':knn_cv.best_score_,
              'Decision Tree':tree_cv.best_score_,
              'Logistic Regression':logreg_cv.best_score_,
              'SVM':svm_cv.best_score_}

best_algorithm = max(algorithms, key= lambda x: algorithms[x])

print('The method which performs best is \"',best_algorithm,'\" with a score of',algorithms[best_algorithm])
```

```
The method which performs best is " Decision Tree " with a score of 0.8875
```

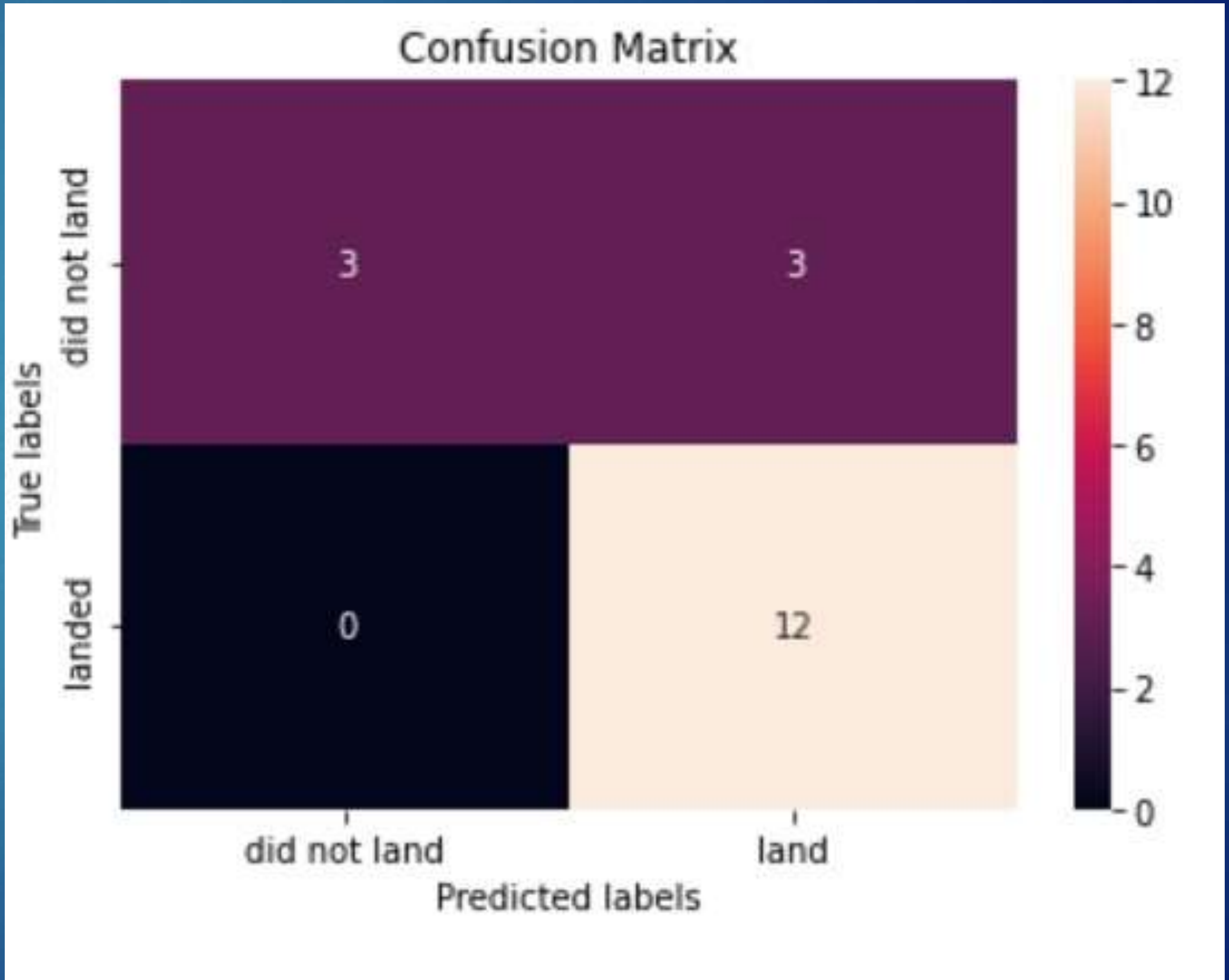
Classification Accuracy

The method which performs best is " Decision Tree " with a score of (0.8875)



Confusion Matrix

The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Low weighted payloads perform better than the heavier payloads
- Launch success rate started to increase in 2013 till 2020. The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!