IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Raja Bharat Rangaswamy
24-Dec-2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection via API, Web Scraping
  - Exploratory Data Analysis(EDA) with Data Visualization
  - EDS with SQL
  - Interactive Map with Folium
  - Dashboards with Plotly Dash
  - Predictive Analysis
- Summary of all results
  - Exploratory Data Analysis results
  - Interactive maps and dashboard
  - Predictive results

Introduction

- Project background and context
  - SpaceX is a revolutionary company who has disrupt the space industry by offering a rocket launches specifically Falcon 9 as low as 62 million dollars; while other providers cost upward of 165 million dollar each. Most of this saving thanks to SpaceX astounding idea to reuse the first stage of the launch by re-land the rocket to be used on the next mission. Repeating this process will make the price reduce even further. As a data scientist of a startup rivaling SpaceX, the goal of this project is to create the machine learning pipeline to predict the landing outcome of the first stage in the future. This project is crucial in identifying the right price to bid against SpaceX for a rocket launch.
- Problems that needs to be answered:
  - 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 collected using SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Datasets are collected from REST SpaceX API and Webscrapping Wikipedia

- The information obtained by the API are rocket launches, payload information.
  - The Space X REST API URL is api.spacexdata.com/v4/

SpaceX Rest API call

API returns JSON file

Make Dataframe from JSON

Clean Data and export it

- The information obtain by the webscrapping of Wikipedia are launches, landing, payload information.
  - URL is https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Get HTML response from Wikipedia

Extract data with BeautifulSoup

Make Dataframe

Export Data

Data Collection – SpaceX API

1. Getting Response from API

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

2. Convert Response to JSON File

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

3. Transform Data

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

4. Create Dictionary with Data

```
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}
```

5. Create Dataframe

```
data = pd.DataFrame.from_dict(launch_dict)
```

6. Filter Dataframe

```
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

7. Export to File

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```


Link to Codes

Data Collection - Scraping

1. Getting Response from HTML

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

2. Create BeautifulSoup Object

```
soup = BeautifulSoup(response.text, "html5lib")
```

3. Find all Tables

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

4. Get Column Names

```
for th in first_launch_table.find_all('th'):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0 :
        column_names.append(name)
```

5. Create Dictionary

```
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each value to be an empty list
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']=[]
```

6. Add Date to Keys

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all
    # get table row
     for rows in table.find_all("tr"):
            #check to see if first table heading is a
            if rows.th:
                if rows.th.string:
                    flight_number=rows.th.string.stri
                    flag=flight_number.isdigit()
```

7. Create Dataframe from Dictionary

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

8. Export to File

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


Link to Codes

Data Wrangling

- In the dataset, there are several cases where the booster did not land successfully.
  - True Ocean, True RTLS, True ASDS means the mission has been successful.
  - False Ocean, False RTLS, False ASDS means the mission was a failure.
- We need to transform string variables into categorical variables where 1 means the mission has been successful and 0 means the mission was a failure.

1. Calculate launches number for each site

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

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

2. Calculate the number and occurrence of each orbit

```
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
```

3. Calculate number and occurrence of mission outcome per orbit type

```
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
```

4. Create landing outcome label from Outcome Column

```
landing_class = []
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
df['Class']=landing_class
```

5. Export to File

```
df.to_csv("dataset_part_2.csv", index=False)
```

Link to Codes

EDA with Data Visualization

- Scatter Graphs
  - 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 relationship between variables. This relationships called correlation.

- Bar Graph
  - Success rate vs. Orbit

Bar graphs show the relationship between numeric and categoric variables.

- Line Graph
  - Success rate vs. Year

Line graphs show data variables and their trends. Line graphs can help to show global behavior and make prediction for unseen data.

[Link to Codes]

EDA with SQL

- We performed SQL queries to gather and understand data from dataset:
  - Displaying the names of the unique launch sites in the space mission.
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS).
  - Display average payload mass carried by booster version F9 v1.1.
  - List the date when the first successful landing outcome in ground pad was achieved.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - List the total number of successful and failure mission outcomes.
  - List the names of the booster stations which have carried the maximum payload mass.
  - List the records which will display the month names, failure landing outcomes in drone ship, booster versions, launch site for the months in year 2015.
  - Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

Link to Codes

Build an Interactive Map with Folium

- Folium map object is a map centered on NASA Johnson Space Center at Houson, Texas
    - Red circle at NASA Johnson Space Center's coordinate with label showing its name (folium.Circle, folium.map.Marker).
    - Red circles at each launch site coordinates with label showing launch site name (folium.Circle, folium.map.Marker, folium.features.DivIcon).
    - The grouping of points in a cluster to display multiple and different information for the same coordinates (folium.plugins.MarkerCluster).
    - Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing. (folium.map.Marker, folium.Icon).
    - Markers to show distance between launch site to key locations (railway, highway, coastway, city) and plot a line between them. (folium.map.Marker, folium.PolyLine, folium.features.DivIcon)
- These objects are created in order to understand better the problem and the data. We can show easily all launch sites, their surroundings and the number of successful and unsuccessful landings.

Link to Codes

Build a Dashboard with Plotly Dash

- Dashboard has dropdown, pie chart, range slider and scatter plot components
  - Dropdown allows a user to choose the launch site or all launch sites (dash_core_components.Dropdown).
  - Pie chart shows the total success and the total failure for the launch site chosen with the dropdown component (plotly.express.pie).
  - Rangeslider allows a user to select a payload mass in a fixed range (dash_core_components.RangeSlider).
  - Scatter chart shows the relationship between two variables, in particular Success vs Payload Mass (plotly.express.scatter)

Link to Codes

Predictive Analysis (Classification)

- Data preparation
  - Load dataset
  - Normalize data
  - Split data into training and test sets.
- Model preparation
  - Selection of machine learning algorithms
  - Set parameters for each algorithm to GridSearchCV
  - Training GridSearchModel models with training dataset
- Model evaluation
  - Get best hyperparameters for each type of model
  - Compute accuracy for each model with test dataset
  - Plot Confusion Matrix
- Model comparison
  - Comparison of models according to their accuracy
  - The model with the best accuracy will be chosen (see Notebook for result)

[Link to Codes]

Results

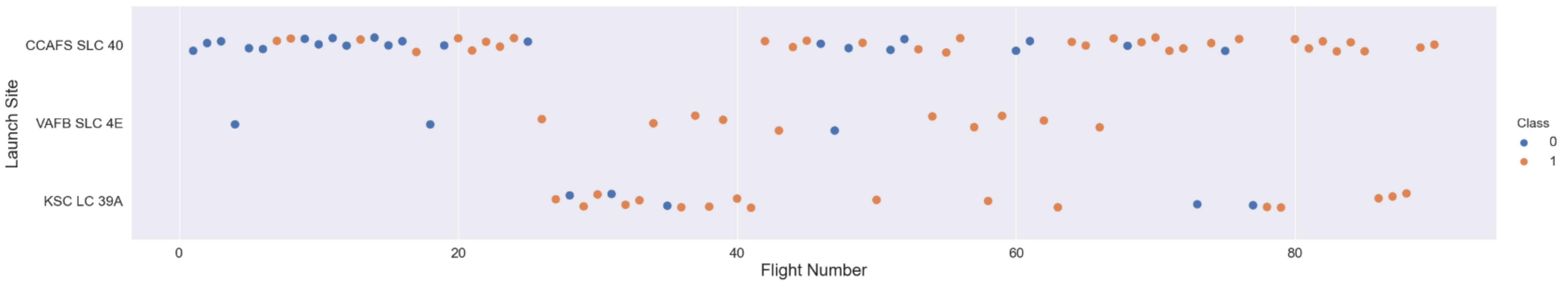
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

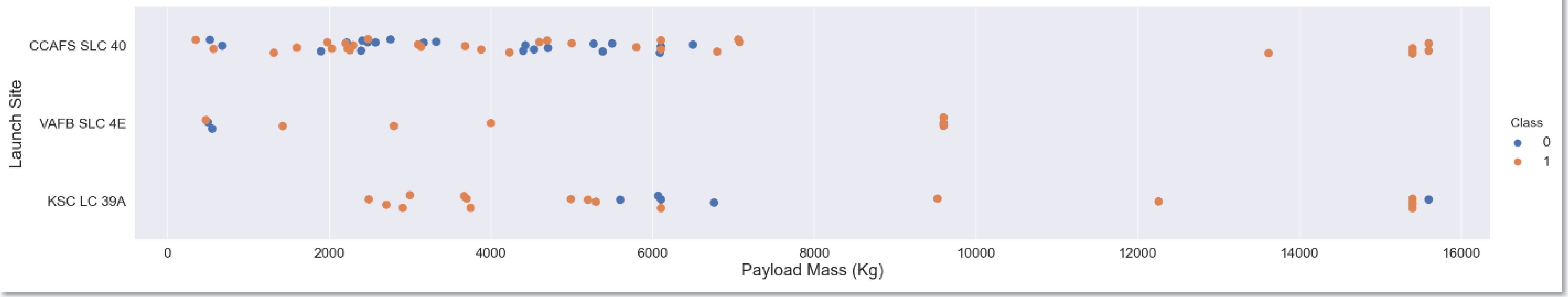
```
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5, s = 10)
sns.set(font_scale=1.5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Raise in success rate, for each launch site can be seen from the plot.

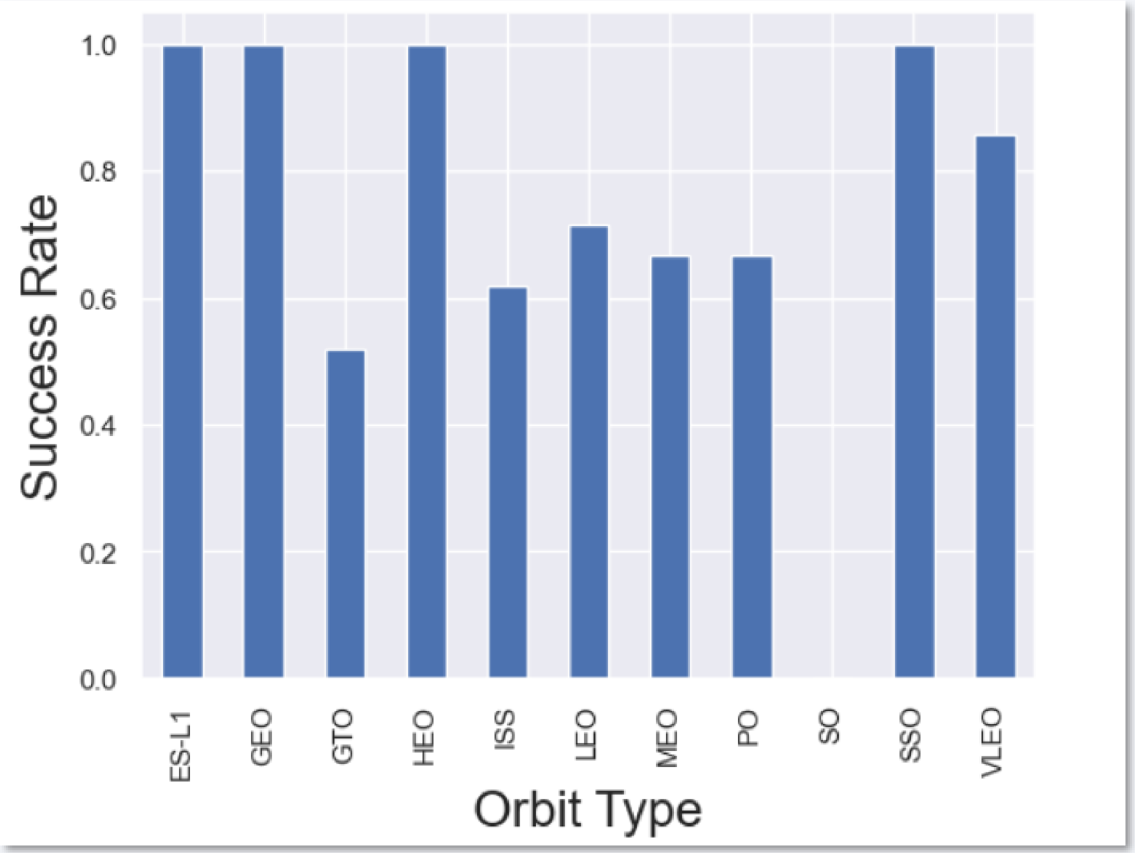
Payload vs. Launch Site

```
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5, s = 10)
sns.set(font_scale=1.5)
plt.xlabel("Payload Mass (Kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Depending on the launch site, heavier payload may be considered for successful landing. On the other hand, too heavy payload can make the landing fail.

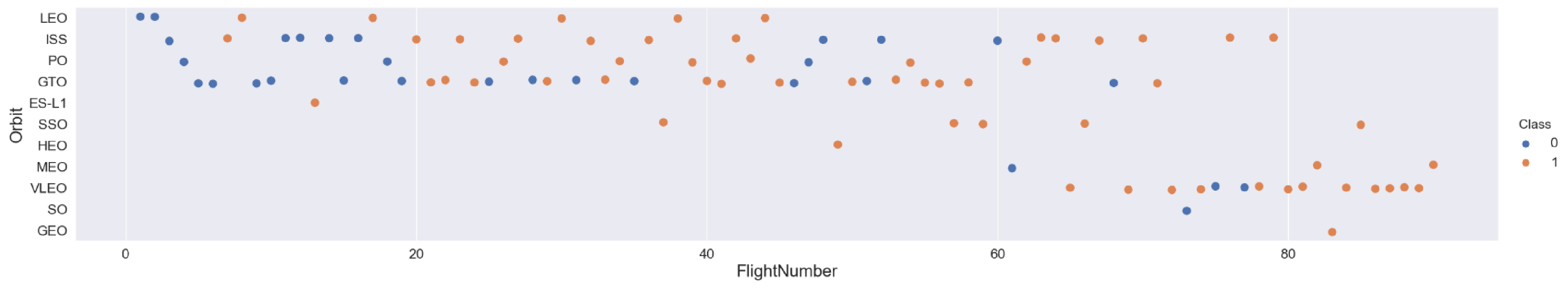
Success Rate vs. Orbit Type



With this plot, we can see success rate for different orbit types. We note that ES-L1, GEO, HEO, SSO have the best success rate.

Flight Number vs. Orbit Type

```
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5, s = 10)
sns.set(font_scale=1.5)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



We notice that the success rate increases with the number of flights for the LEO orbit. For some orbits like GTO, there is no relation between the success rate and the number of flights. But we can suppose that the high success rate of some orbits like SSO or HEO is due to the knowledge learned during former launches for other orbits.

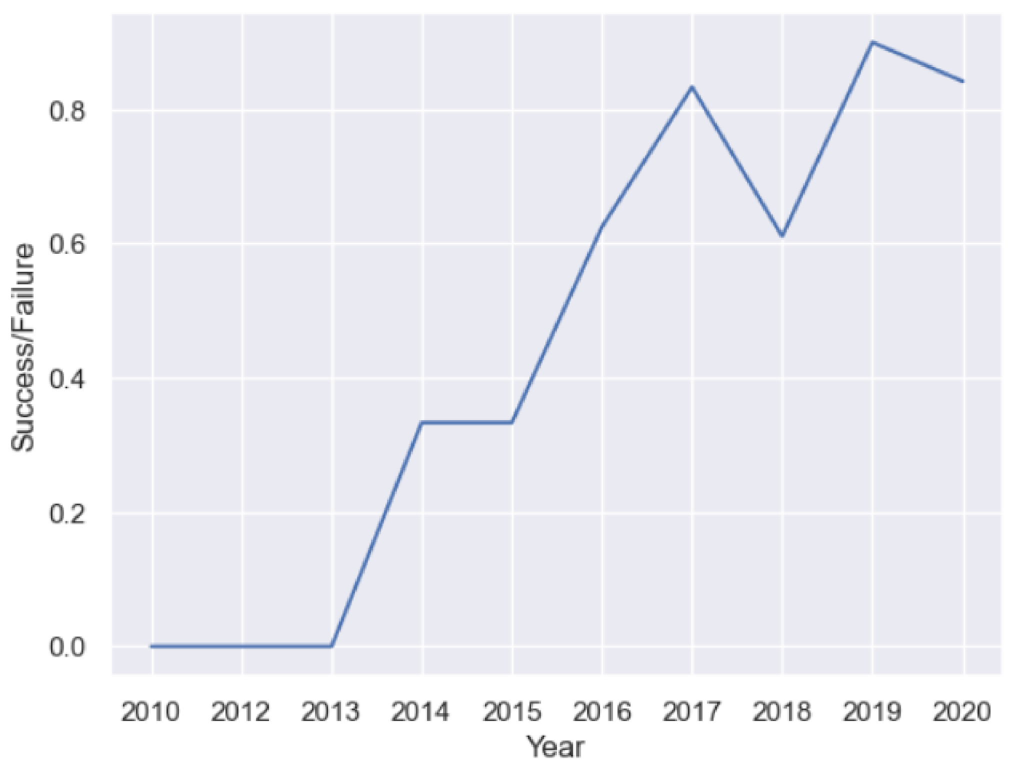
Payload vs. Orbit Type

```
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5, s = 10)
sns.set(font_scale=1.5)
plt.xlabel("Payload Mass (Kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

The weight of the payloads can have a great influence on the success rate of the launches in certain orbits. For example, heavier payloads improve the success rate for the LEO orbit. Another finding is that decreasing the payload weight for a GTO orbit improves the success of a launch.

Launch Success Yearly Trend

```
plt.plot(average_by_year["Year"],average_by_year["Class"])
sns.set(font_scale=1)
plt.xlabel("Year")
plt.ylabel("Success/Failure")
plt.show()
```



Since 2013, we can see an increase in the Space X Rocket success rate.

All Launch Site Names

SQL Query

Task 1

Display the names of the unique launch sites in the space mission

```
%%sql
SELECT DISTINCT "LAUNCH_SITE", COUNT(*) AS 'Number_of_Launches'
FROM SPACEXTBL
GROUP BY "LAUNCH_SITE";
```

Results

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

Launch_Site	Number_of_Launches
CCAFS LC-40	26
CCAFS SLC-40	34
KSC LC-39A	25
VAFB SLC-4E	16

Explanation

The use of DISTINCT in the query allows to remove duplicate LAUNCH_SITE

Launch Site Names Begin with 'CCA'

SQL Query

Task 2

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

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

Results

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

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

Explanation

The WHERE clause followed by LIKE clause filters launch sites that contain the substring CCA. LIMIT 5 shows 5 records from filtering.

Total Payload Mass

SQL Query

Task 3

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

```
%%sql
SELECT SUM("PAYLOAD_MASS__KG_")
FROM SPACEXTBL
WHERE "Customer" = 'NASA (CRS)';
```

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

Results

SUM("PAYLOAD_MASS__KG_")
45596

Explanation

The query returns the sum of all payload masses where the customer is NASA(CRS).

Average Payload Mass by F9 v1.1

SQL Query

Task 4

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

```
%%sql
SELECT AVG("PAYLOAD_MASS__KG_")
FROM SPACEXTBL
WHERE "Booster_Version" LIKE 'F9 v1.1%';
```

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

Results

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

Explanation

The query returns the average of all payload masses where the booster version contains the substring F9 v1.1.

First Successful Ground Landing Date

SQL Query

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%%sql
SELECT MIN("DATE") AS "Succesful Landing"
FROM SPACEXTBL
WHERE "LANDING _OUTCOME" = 'Success (ground pad)';
```

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

Results

Succesful Landing
22-12-2015

Explanation

With this query, we select the oldest successful landing.

The WHERE clause filters dataset in order to keep only records where landing was successful. With the MIN function, we select the record with the oldest date.

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query

Task 6

List the names of the boosters which have success in drone ship and have 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;
```

Results

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

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

Explanation

This query returns the booster version where landing was successful and payload mass is between 4000 and 6000 kg. The WHERE and AND clauses filter the dataset.

Total Number of Successful and Failure Mission Outcomes

SQL Query

Task 7

List the total number of successful and failure mission outcomes

```
%%sql
SELECT (SELECT COUNT("MISSION_OUTCOME")
        FROM SPACEXTBL
        WHERE "MISSION_OUTCOME" LIKE '%Success%') AS SUCCESS,
        (SELECT COUNT("MISSION_OUTCOME")
         FROM SPACEXTBL
         WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS FAILURE;
```

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

Results

SUCCESS	FAILURE
100	1

Explanation

With the first SELECT, we show the subqueries that return results. The first subquery counts the successful mission. The second subquery counts the unsuccessful mission. The WHERE clause followed by LIKE clause filters mission outcome. The COUNT function counts records filtered.

Boosters Carried Maximum Payload

SQL Query

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

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

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

Results

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

Explanation

We used a subquery to filter data by returning only the heaviest payload mass with MAX function. The main query uses subquery results and returns unique booster version (SELECT DISTINCT) with the heaviest payload mass.

2015 Launch Records

SQL Query

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

```
%%sql
SELECT SUBSTR("DATE",4,2) AS Month, "LANDING _OUTCOME", "BOOSTER_VERSION", "LAUNCH_SITE"
FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Failure (drone ship)'
    AND SUBSTR(Date,7,4)='2015';
```

Results

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

Month	Landing _Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Explanation

This query returns month, booster version, launch site where landing was unsuccessful and landing date took place in 2015. SUBSTR function process date in order to take month or year. SUBSTR(DATE, 4, 2) shows month. SUBSTR(DATE,7, 4) shows year.

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

SQL Query

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%%sql
SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") AS "TOTAL_NUMBER"
FROM SPACEXTBL
WHERE DATE BETWEEN '04-06-2010' AND '20-03-2017'
GROUP BY "LANDING _OUTCOME"
ORDER BY "TOTAL_NUMBER" DESC;
```

Results

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

Landing _Outcome	TOTAL_NUMBER
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

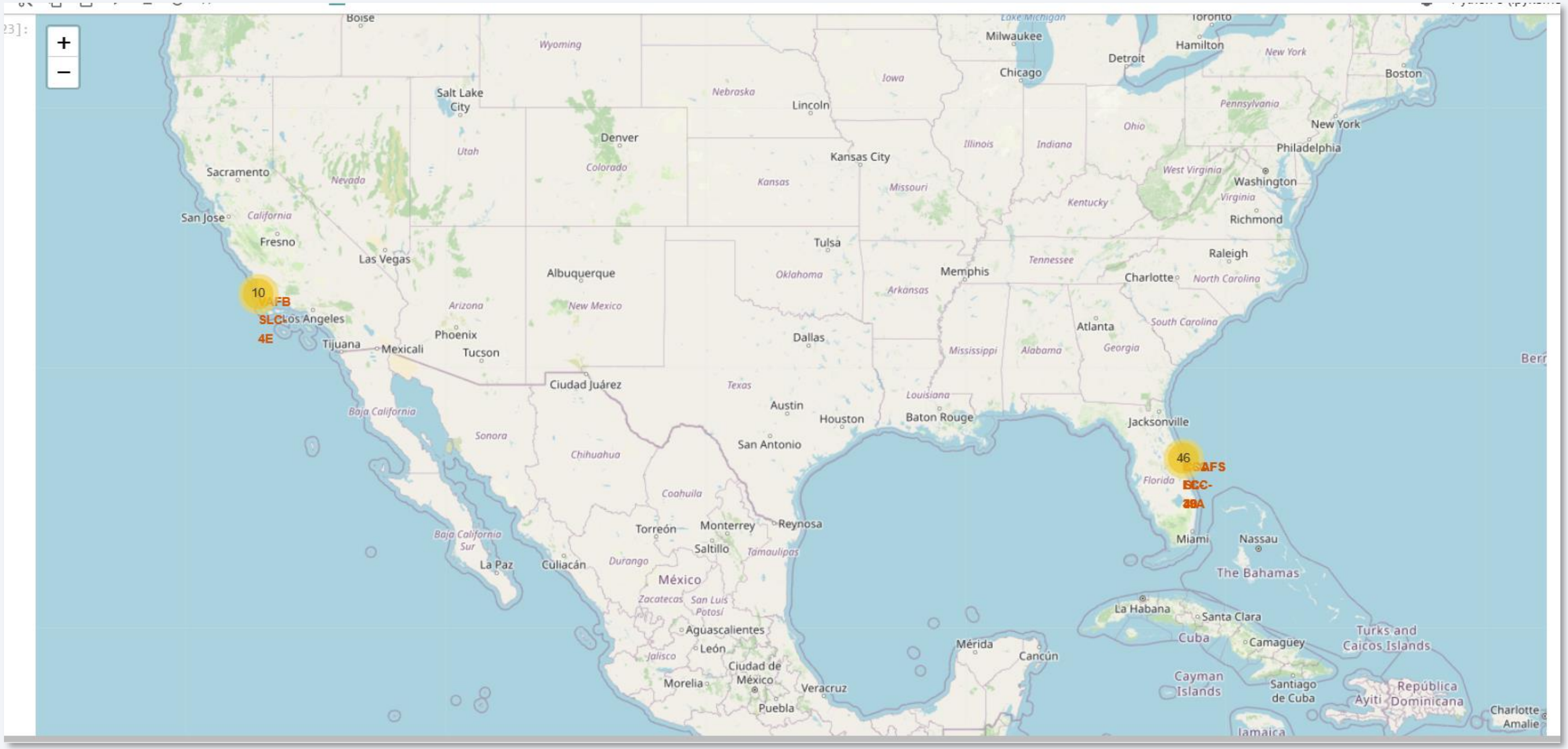
Explanation

This query returns landing outcomes and their count where the date is between 04/06/2010 and 20/03/2017. The GROUP BY clause groups results by landing outcome and ORDER BY COUNT DESC shows results in decreasing order.

Section 3

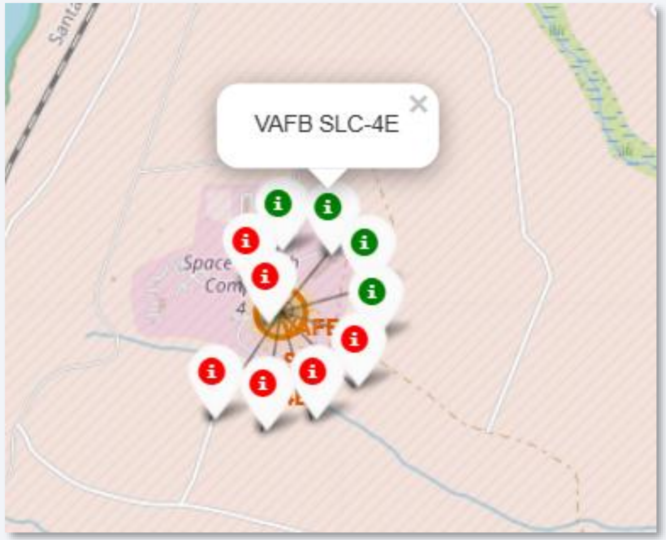
Launch Sites Proximities Analysis

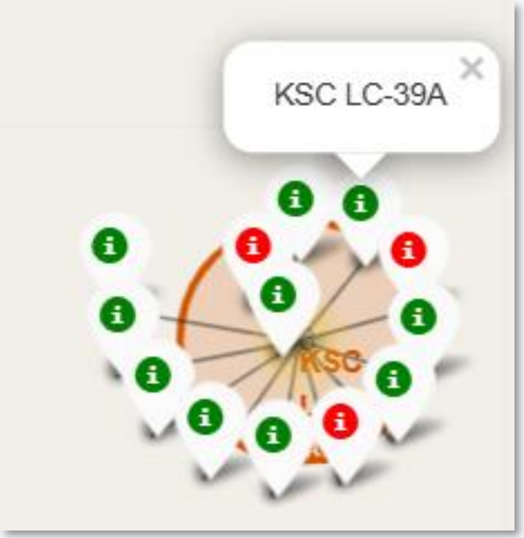
Folium Map – Ground stations

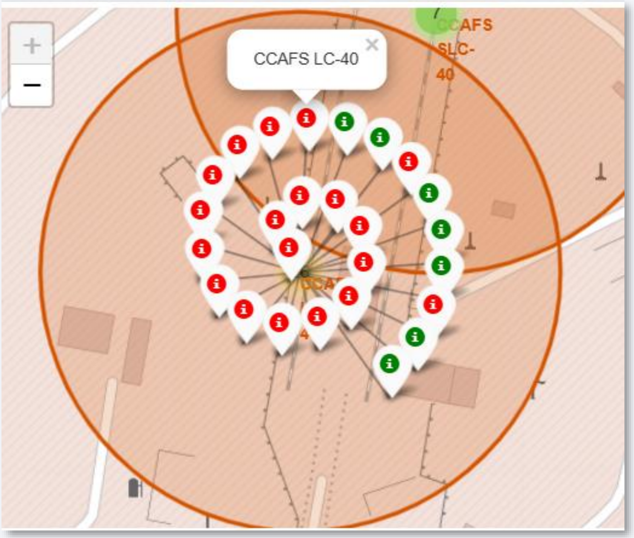


All Space X launch sites are located on the coastal line of the United States.

Folium Map Color Labeled Markers

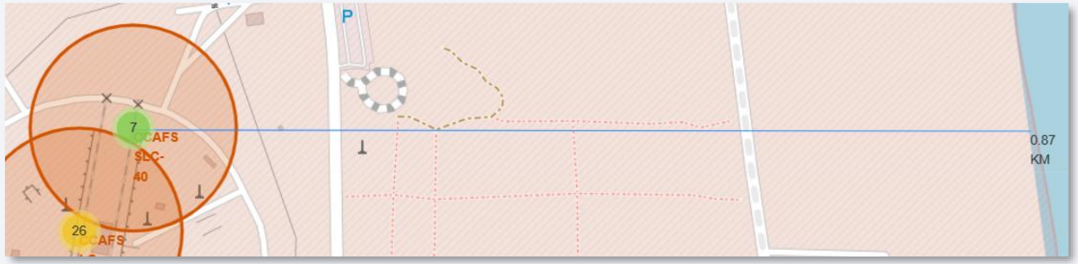


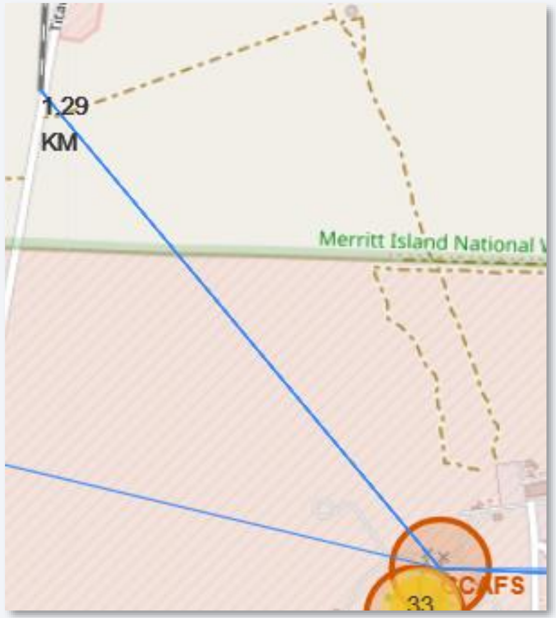




Green marker represents successful launches, **Red** marker represents unsuccessful launches. We note that KSC LC-39A has a higher launch success rate.

Folium Map – Distance between CCAFS SLC-40 and its proximities



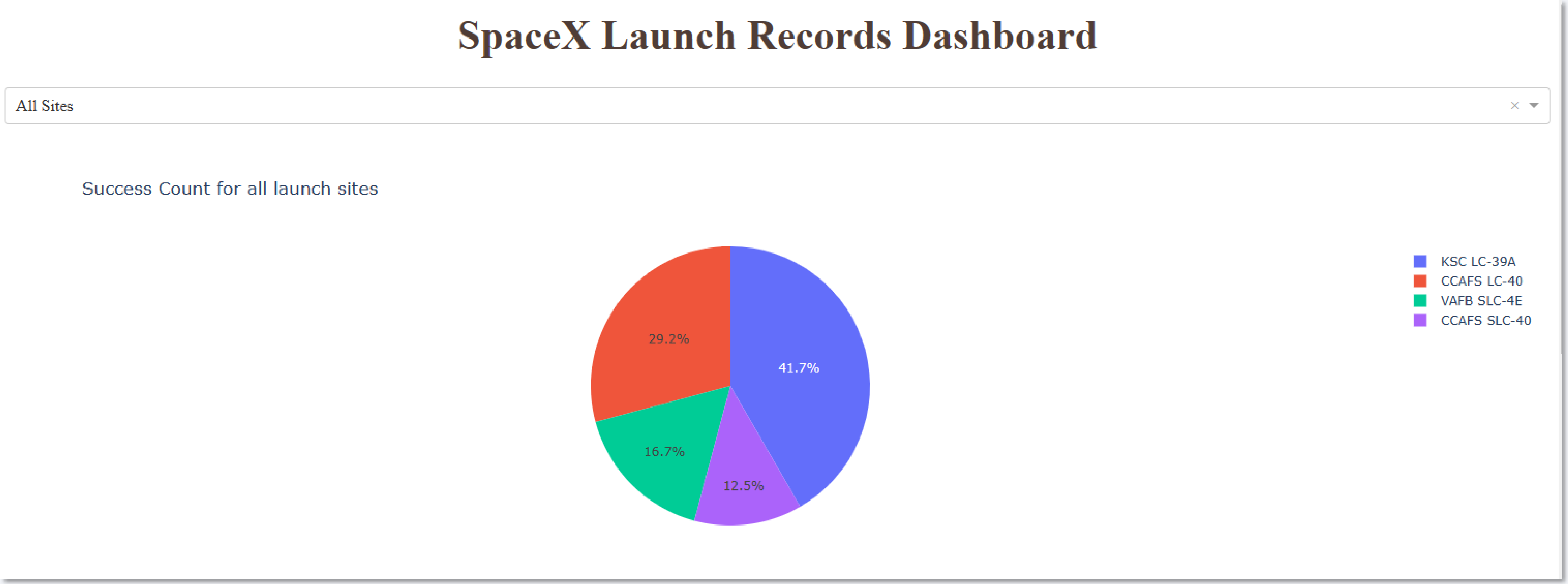


- Is CCAFS SLC-40 in close proximity to railways ? YES
- Is CCAFS SLC-40 in close proximity to highways ? YES
- Is CCAFS SLC-40 in close proximity to coastline ? YES
- Is CCAFS SLC-40 in close proximity to cities ? NO

Section 4

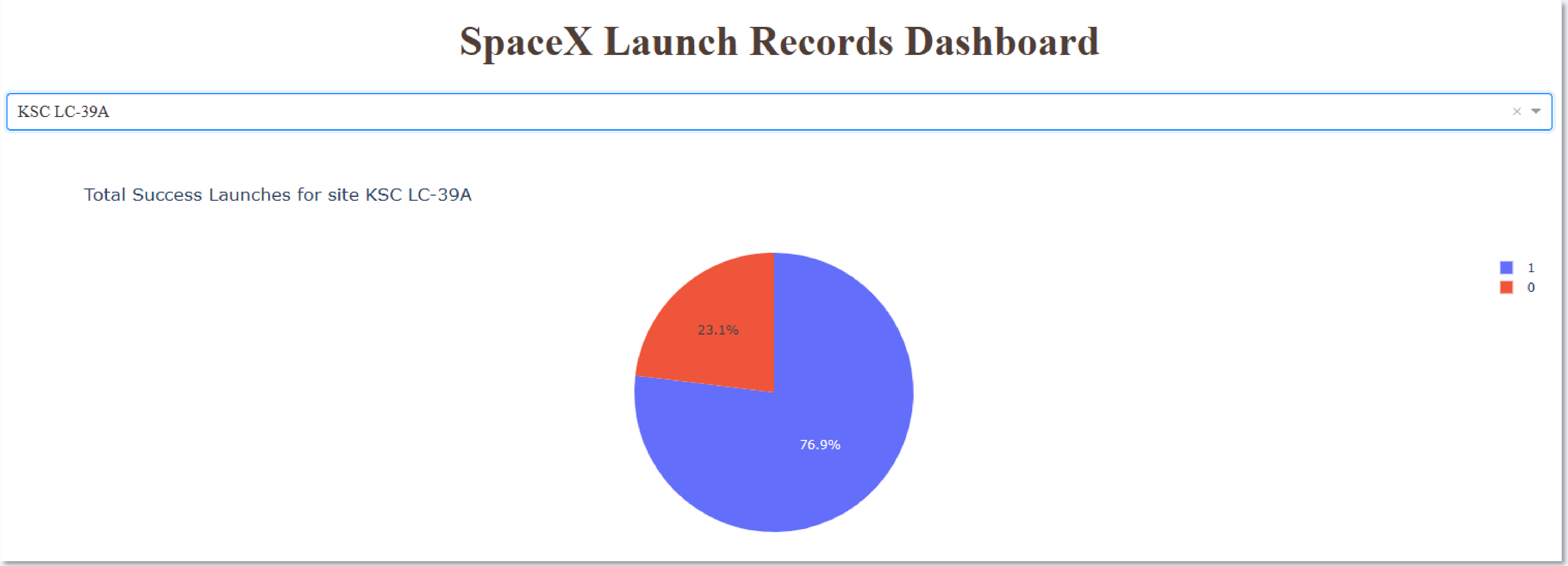
Build a Dashboard with Plotly Dash

Dashboard – Total Success by Site



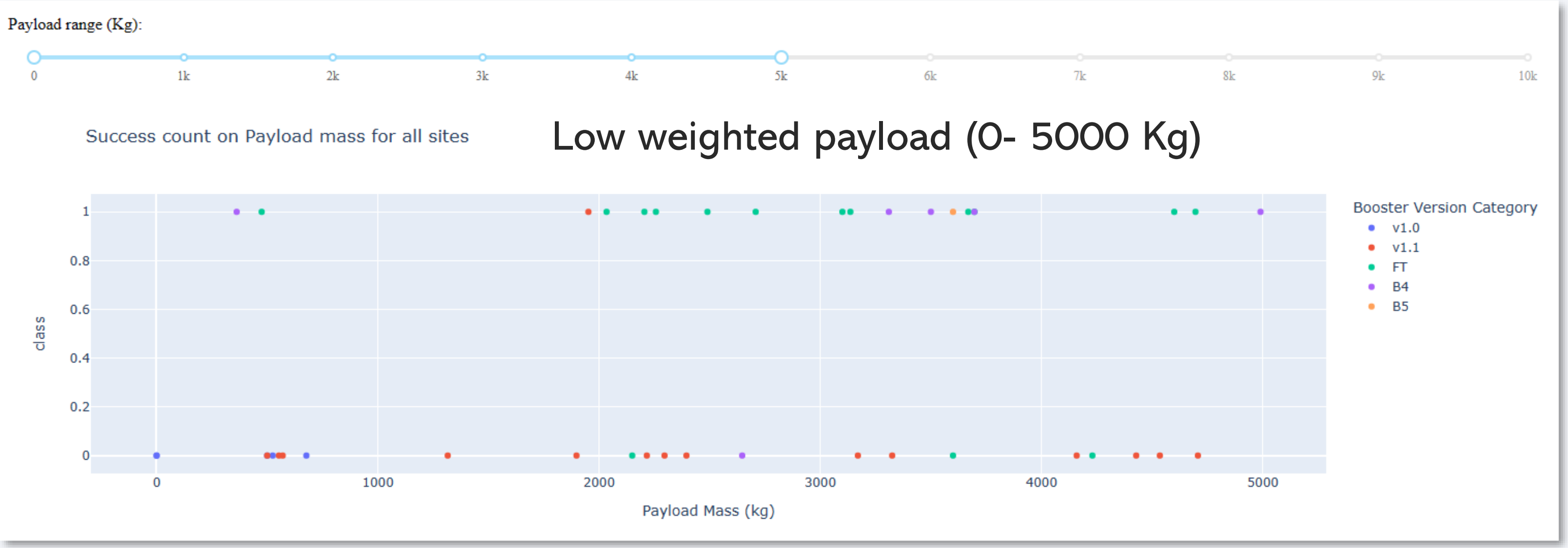
We can see KSC LC-39A ha the best success rate of launches

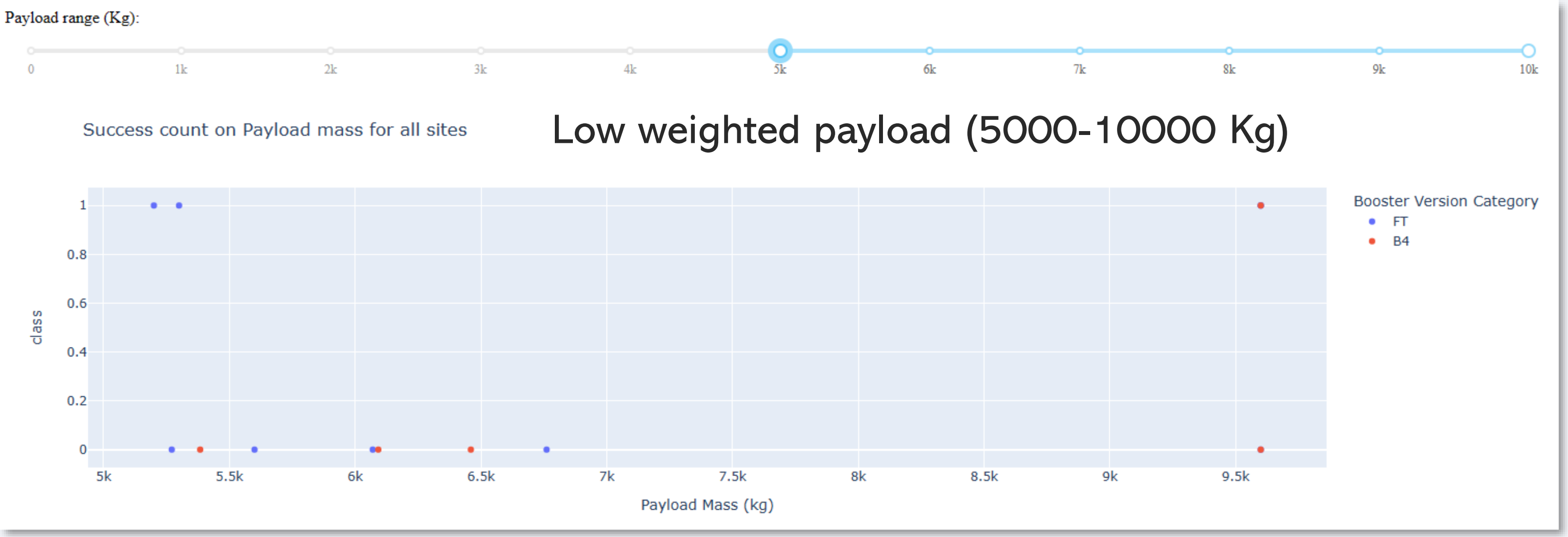
Dashboard – Total Success Launches for Site KSC LC-39A



We can see that KSC LC-39A has achieved a **76.9%** success rate while getting a **23.1%** failure rate.

Dashboard – Payload Mass vs Outcome for all sites with different payload mass.





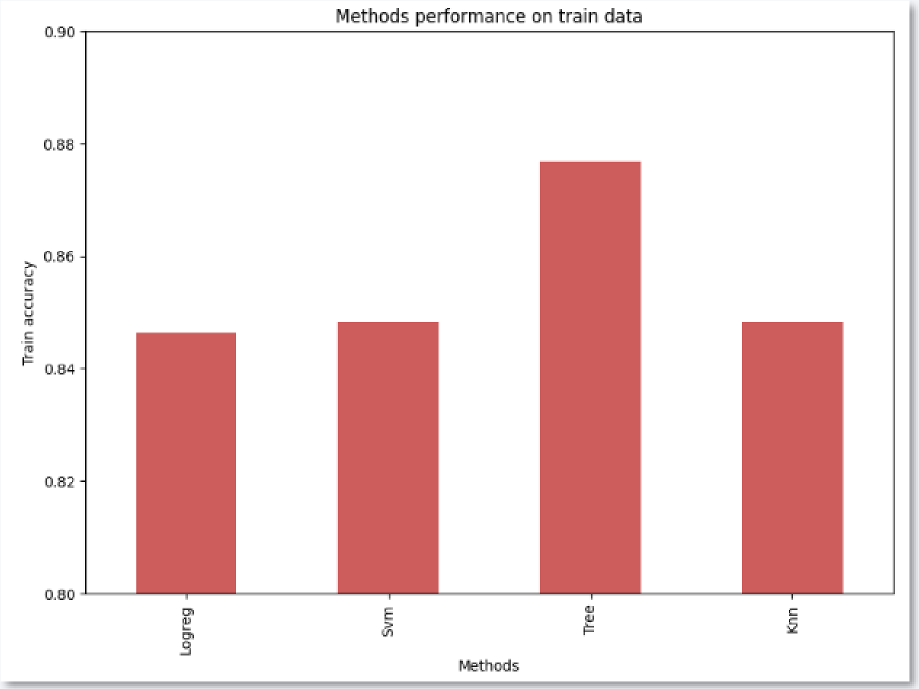
Low weighted payloads below 5000 Kg have a better success rate than the heavy weighted payloads.

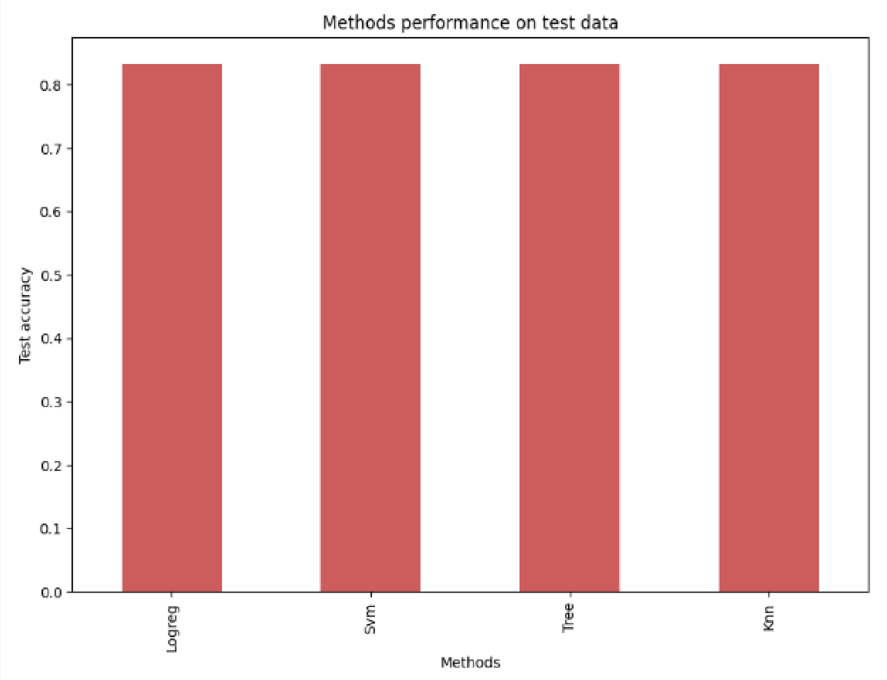
Section 5

Predictive Analysis (Classification)

Classification Accuracy

	Accuracy Train	Accuracy Test
Tree	0.876786	0.833333
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333

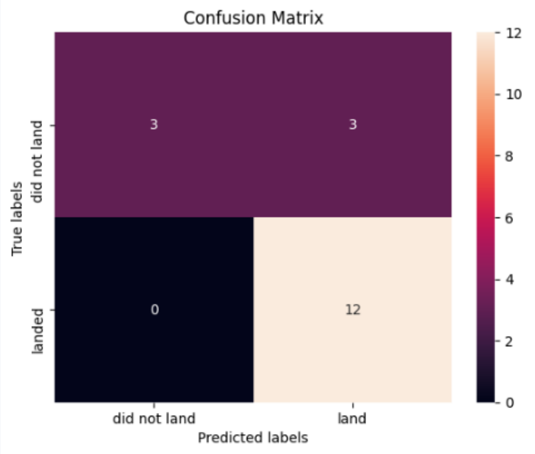




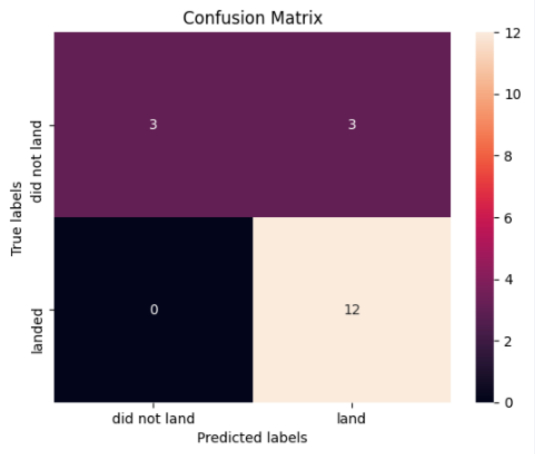

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 2, 'splitter': 'random'}
accuracy : 0.8767857142857143
```

For accuracy test, all methods were performed similar. We could get more test data to decide between them. But for all the tests done till now indicates, we should go with the decision tree.

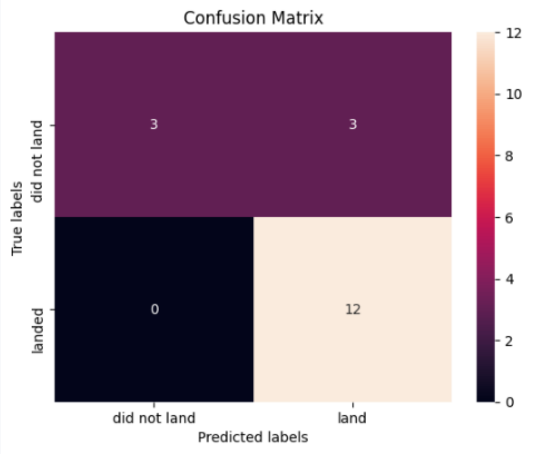
Confusion Matrix



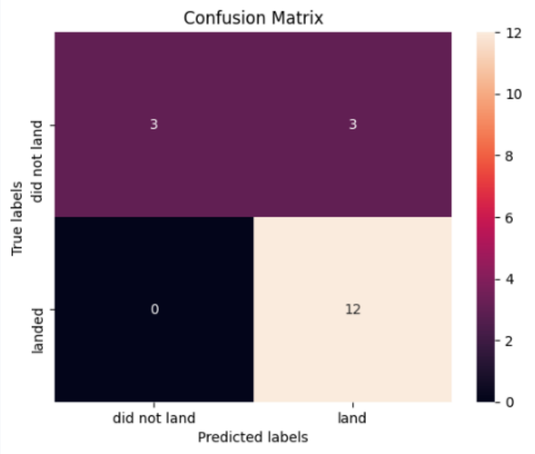
Logistic Regression



Decision Tree

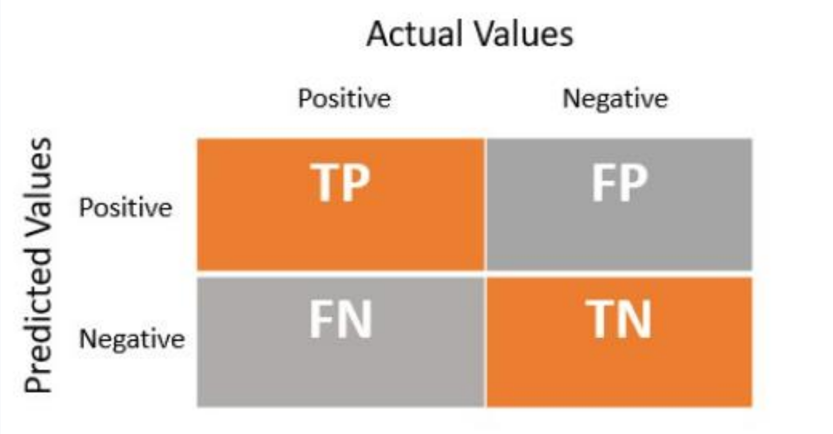


KNN



SVM

As the test accuracy are all equal, the confusion matrices are also identical. The main challenge in these models are false positives.



Conclusions

- The success of a mission can be explained by several factors such as the launch site, the orbit and especially the number of previous launches. Indeed, we can assume that there has been a gain in knowledge between launches that allowed to go from a launch failure to a success.
- The orbits with the best success rates are GEO, HEO, SSO, ES-L1.
- Depending on the orbits, the payload mass can be a decided to take into account for the success of a mission. Some orbits require a light or heavy payload mass. But generally low weighted payloads perform better than the heavy weighted payloads.
- With the current data, we cannot explain why some launch sites are better than others (KSC LC-39A is the best launch site). To get an answer to this problem, we could obtain atmospheric or other relevant data.
- For this dataset, we choose the Decision Tree Algorithm as the best model even if the test accuracy between all the models used is identical. We choose Decision Tree Algorithm because it has a better train accuracy.

Thank you!