Rocket first stage landing analysis for SpaceY

Bhuvaneshwari S Baranwal

28-10-2023

Outline

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

Executive Summary

Summary of methodologies

In an effort to lower the launch cost overall, this work makes an attempt to forecast the reuse of a rocket's first stage. The methods used in the process can be summarized as follows,

- *Collection* of SpaceX launch data using webcrapping and API
- *Preprocessing* the data and organising the collected data to identify the required variables
- *Data visualisation* through examination of the correlations between characteristics, like launch seats, launch results, orbit type, and success rate, etc.
- To comprehend information like the maximum payload, average payload, and total number of successful and failed launches, analyse the data using *SQL* queries.
- Explore the success rates of launch sites and their closeness to geographic landmarks using **Folium mapping**. Showcase the launch locations with the highest success rates and effective payload ranges using ***dash and plotly.***
- Create ***classification models*** utilising decision trees, logistic regression, support vector machines (SVM), and K-nearest neighbour (KNN) to forecast landing results.

Summary of all results

Explaratory data analysis :

The success of launches has increased with time.Among landing locations, KSC LC-39A has the best success rate.There is a 100% success rate for orbits ES-L1, GEO, HEO, and SSO.

Visualisation and Analytics:

• The majority of launch locations are located close to the coast and close to the equator.

Predictive analytics:

• On the test set, every model performed similarly. The decision tree model outperformed marginally.

Introduction

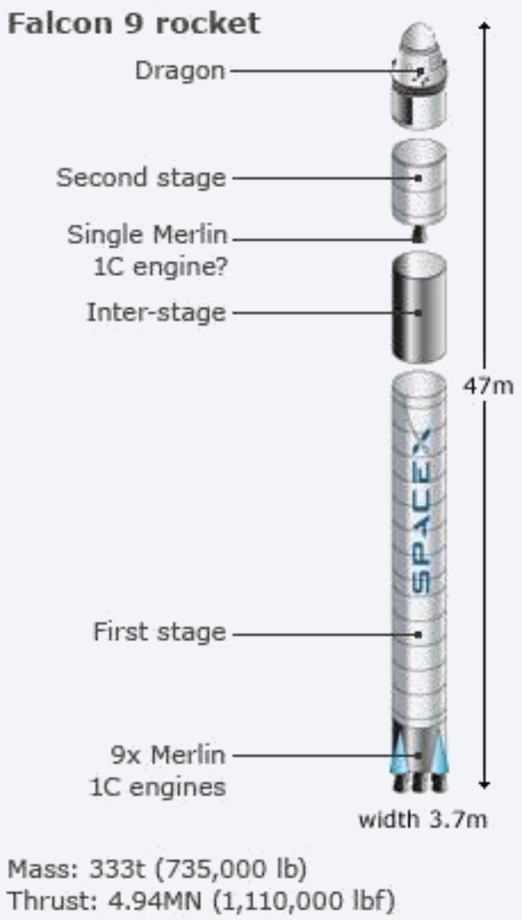
- **Project background and context**

The price of of rocket launches play a huge role in space missions sending aircrafts to international space station (ISS). One such successful agency is SpaceX. SpaceX has been very successful and including manned missions. The main reason of these successful rocket launches is because they are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

- **About Falcon 9:**

Falcon 9 is a reusable, two-stage rocket designed and manufactured by SpaceX for the reliable and safe transport of people and payloads into Earth orbit and beyond. Falcon 9 is the world's first orbital class reusable rocket. Reusability allows SpaceX to refly the most expensive parts of the rocket, which in turn drives down the cost of space access.

Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. Spaces X's Falcon 9 launch like regular rockets.

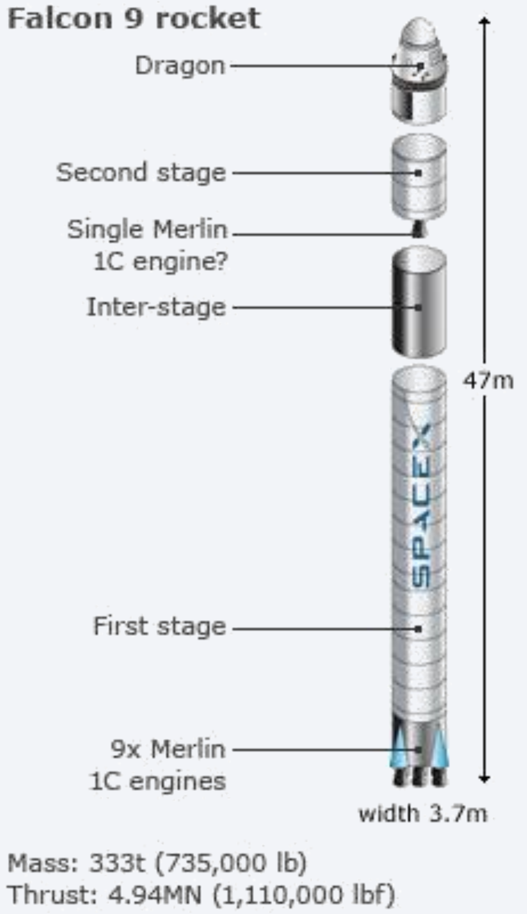


Problem statement and objectives

This investigation intends to ascertain if the first stage can land successfully so that it may be reused in order to minimise costs, hence helping the company compete with SpaceX in launching successful trips to the International Space Station.

The project is carried out in this manner:

- By compiling comprehensive data on Space X launches, such as the quantity of flights, launch locations, launch vehicle types, payloads transported, orbits reached, and so forth.
- To have a deeper understanding of the variables, visualise the data using various data visualisation approaches and build dashboards.
- Determine whether or not SpaceX will be able to reuse the first stage after a launch by employing a machine learning model and publicly available data.



Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

- SpaceX launch data for this project is collected using ***SpaceX REST API .***
- Web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled `List of Falcon 9

Perform data wrangling

- Checking data types of the attributes making changes if needed
- Checking for missing values, replacing them with a defined value.
- Creating/removing columns as per the requirement

Perform exploratory data analysis (EDA) using visualization and SQL

- Creation of plots to understand the effect of different attributes on the landing outcome and discover patterns
- SQL queries to estimate total and average of variables to get insight on the outcome

Perform interactive visual analytics using Folium and Plotly Dash

- Using Folim and Dash plotly to get pie – Success rate based launch sites and scatter plot – launch outcome based on payload range

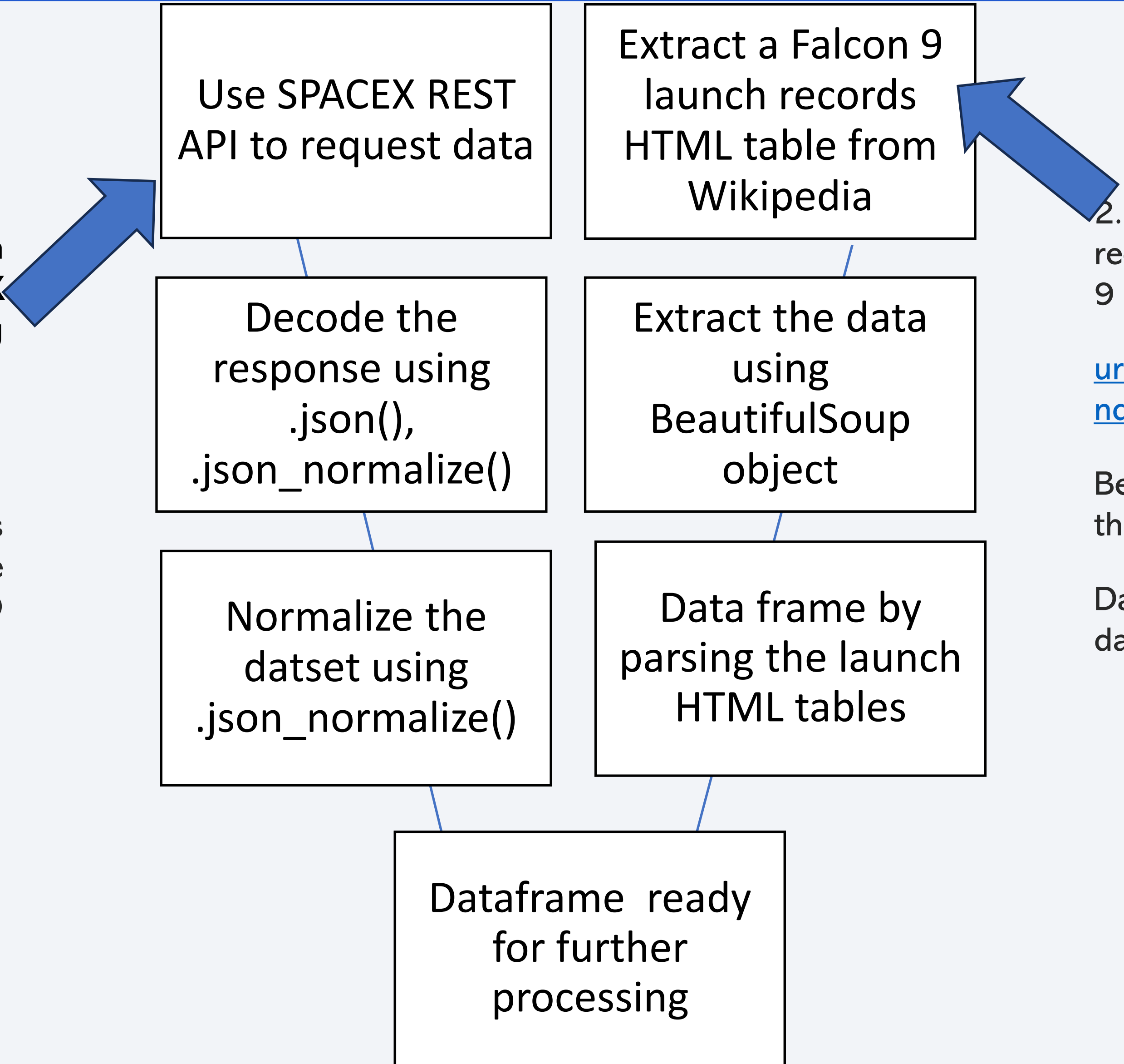
Perform predictive analysis using classification models

- Builing, testing and checking accuracy of various classification models and predicting the best model

Data Collection

1. SpaceX's rocket launch data obtained from **SpaceX REST API** with the following URL: https://api.spacexdata.com/v4/launches/past

Created data frame is filtered to contain only the required Falcon 9 information.



2. **Web scraping** to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches

[url:https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches](https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches)

BeautifulSoup object was used to extract data from the HTML table.

Dataframe is created by parsing the HTML table data

Data Collection – SpaceX API

The flowchart presents the API calls used to get the SpaceX launching data and the subsequent processes used to derive the Falcon 9 dataframe.

Github link:

https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%201_lab_spacex-data-collection-api.ipynb

1. Sending get request to the API

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

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

2. Decoding the response using .json () & .json_normalize ()

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

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data = pd.json_normalize(static_response_df)
data.head(5)
```

3. Data cleaning using custom functions

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

4. Obtained list conversion into dictionary

```
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. Filtering the dataframe with Falcon 9 data

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9= launch_df[launch_df['BoosterVersion']!='Falcon 1']
```

Now that we have removed some values we should reset the FlgihtNumber column

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

Data Collection - Scraping

Web scraping Falcon 9 and Falcon Heavy Launches Records from Wikipedia.

Githublink:

https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%201_lab_spacex-data-collection-webscraping.ipynb

1. HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

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

2. Create a BeautifulSoup object from the HTML response

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

3. Finding HTML tables using find_all()

```
# Use the find_all function in the BeautifulSoup object,
html_tables=soup.find_all('table')
# Assign the result to a list called `html_tables`
html_tables
```

4. Extracting column names

```
column_names = []
column=soup.find_all('th')
for x in range(len(column)):
    try:
        name = extract_column_from_header(column[x])
        if (name is not None and len(name)>0):
            column_names.append(name)
    except:
        pass
```

5. create an empty dictionary with keys for the extracted columns

```
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. Fill in the parsed launch record values into launch dictionary and creation of dataframe

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

Data Wrangling

Checking % of missing values in all attributes

Calculate the number of launches on each site

Calculate the number and occurrence of each orbit

Calculate the number and occurence of mission outcome of the orbits

Create a landing outcome label from Outcome column

Github link: https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%201_labs-jupyter-spacex-Data%20wrangling.ipynb

Data Wrangling

1. Identify and calculate the percentage of the missing values in each attribute

```
df.isnull().sum()/len(df)*100
```

2. Determination of the number of launches on each site

```
# Apply value_counts() on column LaunchSite
print(df['LaunchSite'].value_counts())
```

3. Determining the number and occurrence of each orbit in the column Orbit

```
# Apply value_counts on Orbit column
print(df['Orbit'].value_counts())
```

4. Determining the number of landing outcomes

```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

5. Create a list where the element is zero if the corresponding row in Outcome is in the set bad outcome

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class= []
for key,value in df['Outcome'].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

6. Export the dataframe into a CSV file

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

EDA with Data Visualization

The Falcon 9 dataset was visualised using various combinations of variables to investigate the impact of different parameters on the landing outcome.

- ✓ **FlightNumber vs LaunchSite scatter plot:** To visualize each site's launch records as different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.
- ✓ **Flight number vs. payloadmass scatter plot:** To analyze the impact of the flight number and payload factors on the launch result is displayed in a scatter plot having the launch result (a successful or unsuccessful launch) as Hue.
- ✓ **Successrate vs. orbit type bar chart:** To check if the rate of successful launches depends on the type of the orbit it is launched to.
- ✓ **FlightNumber and Orbit type scatter plot:** For each orbit, to see if there is any relationship between FlightNumber and the type of the orbit.
- ✓ **Payload mass vs. Orbit scatter plot:** to reveal the relationship between Payload and Orbit type
- ✓ **Year vs. Successrate line plot:** To study the yearly successrate trend of the launches.

https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%202_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

After loading the SQL extension and establish a connection with the database the following queries were performed

- To display the names of the **unique launch sites** in the space mission
- To display 5 records where **launch sites begin with the string 'CCA'**
- To estimate the **total payload mass** carried by **boosters launched by NASA (CRS)**
- To calculate the **average payload mass** carried **by booster version F9 v1.1**
- To identify the date when the **first successful landing outcome in ground pad** was acheived.
- To list the names of the boosters which have success **in drone ship** and **have payload mass greater than 4000 but less than 6000**
- To get **the total number of successful and failure mission outcomes**
- To list the names of the **booster_versions** which have carried **the maximum payload mass.**
- To 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.**
- To **rank the count of landing outcomes** (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%202-labs-eda-sql-coursera_sqllite.ipynb

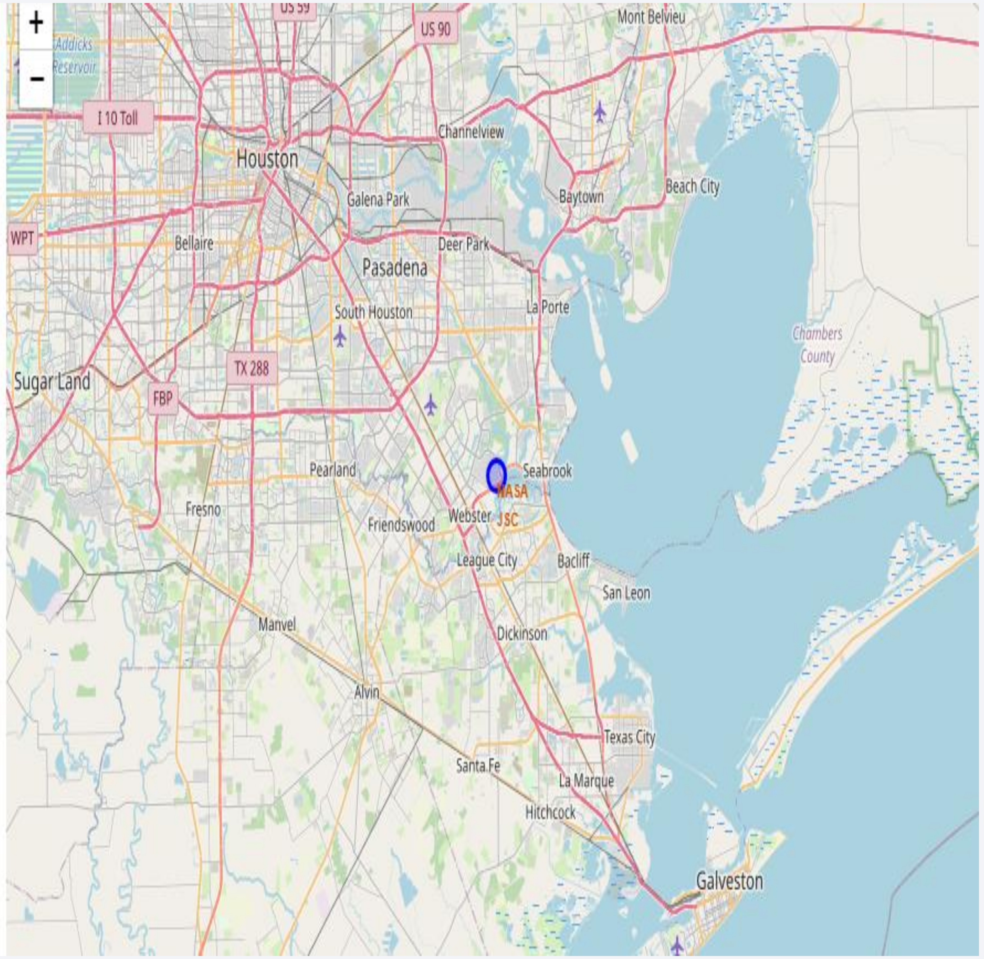
Build an Interactive Map with Folium

Folium map object: Initially a folium Map object, with an initial center location to *be NASA Johnson Space Center (NASA JSC) at Houston, Texas* was created.

Circle objects:
- folium.Circle () object was used to add a highlighted blue circle area with a text label on a specific coordinate correspond to *NASA JSC.*
- To indicate each launch site a red circle was added for the data frame launch_sites in data frame.

Colored Markers of Launch Outcomes: Added colored markers of successful (green) and unsuccessful (red) launches at each launch site to show which launch sites have high success rates.

Polyline Between a Launch Site to Proximities: Added colored lines to show distance between launch site CCAFS SLC40 and its proximity to the nearest coastline, railway, highway, and city.



https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%203_lab_launch_site_location.jupyterlite.ipynb

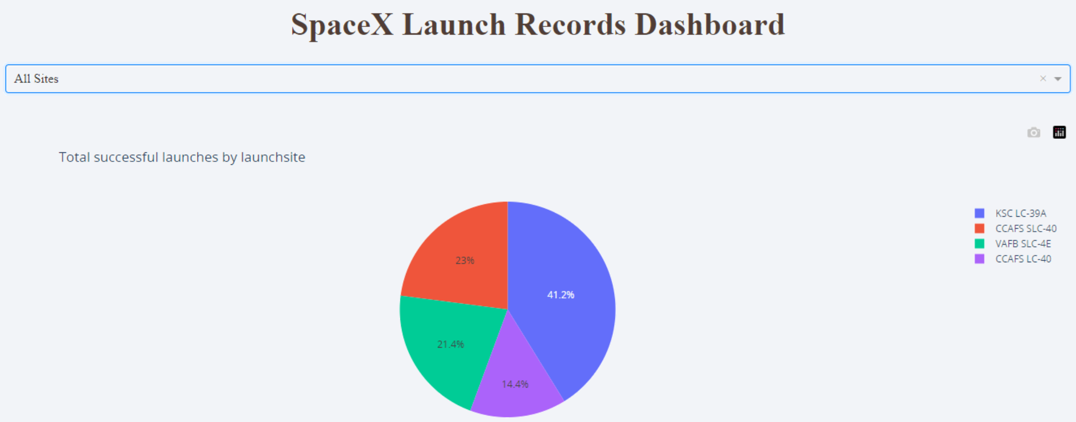
Build a Dashboard with Plotly Dash

Dropdown list: Dropdown list containing all and individual launch Sites was used with a placeholder *'Select a Launch Site here*

Payload mass slider: Slider to select payload rang in kg was added to select the range and visualize the relationship between payload and successrate.

Pie chart: To show percentage of successful and failed launches for all and the selected launch sites

Success-payload-scatter-chart Rate by Booster Version : To see the correlation between Payload mass and Launch Success having booster version as the hue.



https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%203%20lab_spacex_dash_app.py

Predictive Analysis (Classification)

Create a NumPy array using **np.array ()** from the column Class in dataset

Standardize the data in X using **standard scalar, fit and transform** the data

Split the data into train and test sets using **train_test_split ()** object

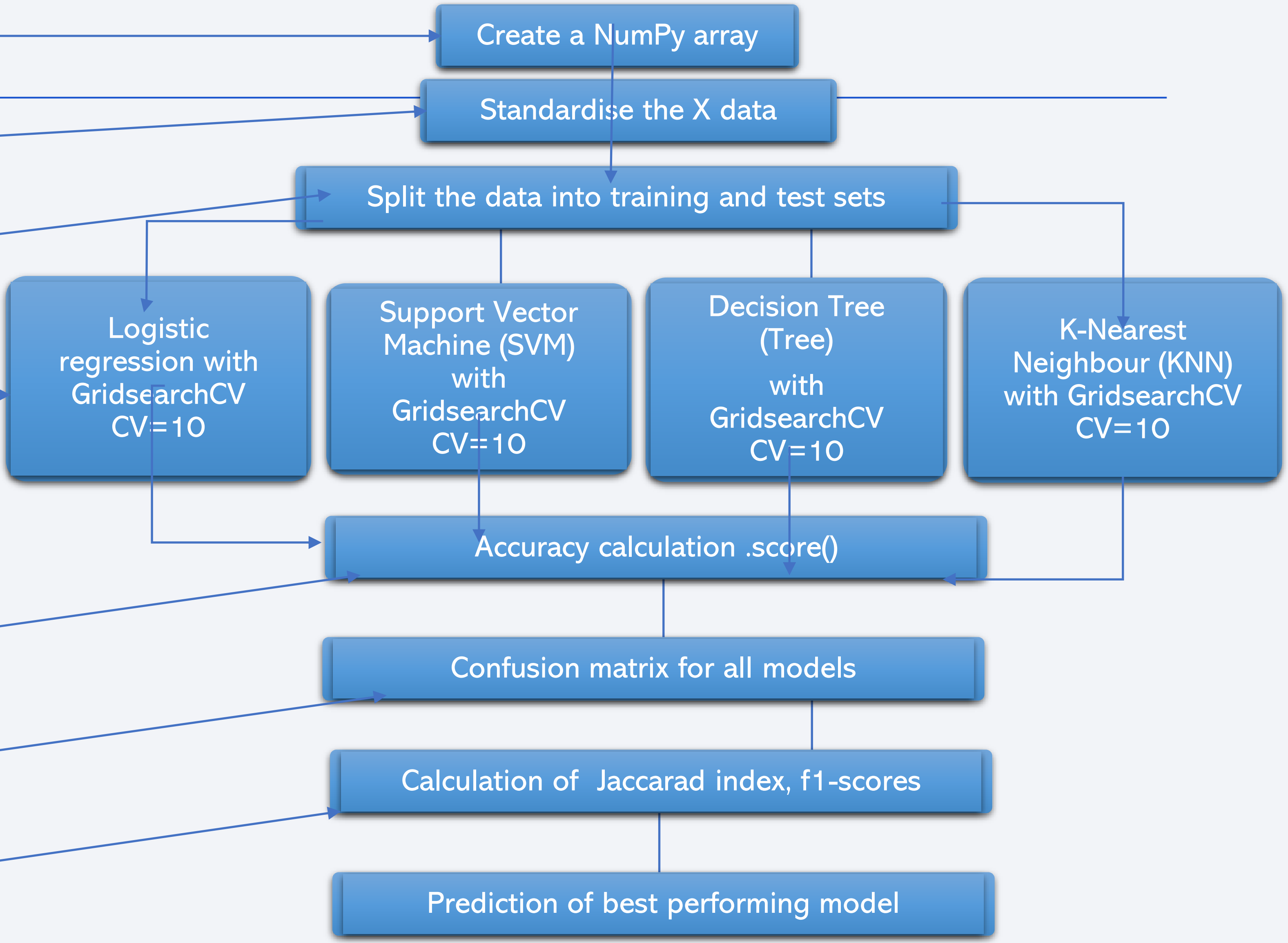
Creation of the following **predictive algorithms using GridSearchCV with CV=10 to train the data**

1. Logistic regression (logreg)
2. Support Vector Machine (SVM)
3. Decision Tree (Tree)
4. K-Nearest Neighbour (KNN)

Calculating accuracy of the models on the test data using **.score ()**

Creation of **confusion matrix** for all the models

Identify the best model using Jaccard score, f1 score and the calculated accuracy score



https://github.com/BhuvanaSakthivel/Capstone-Spacex-Final-project/blob/main/Week%204_SpaceX_Machine_Learning_Prediction.jupyterlite.ipynb

Results

The comprehensive observations from the explanatory data analysis carried out in the preceding sections will be included in the section that follows.

This includes,

- **Exploratory data analysis results**

  Results obtained from data visualization and SQL queries

- **Interactive analytics demo in screenshots**

  Observations from folium maps and screen shots created by plotly Dash app

- **Predictive analysis results**

  Trained and tested predictive models and their respective accuracy scores

Section 2

Insights obtained from EDA

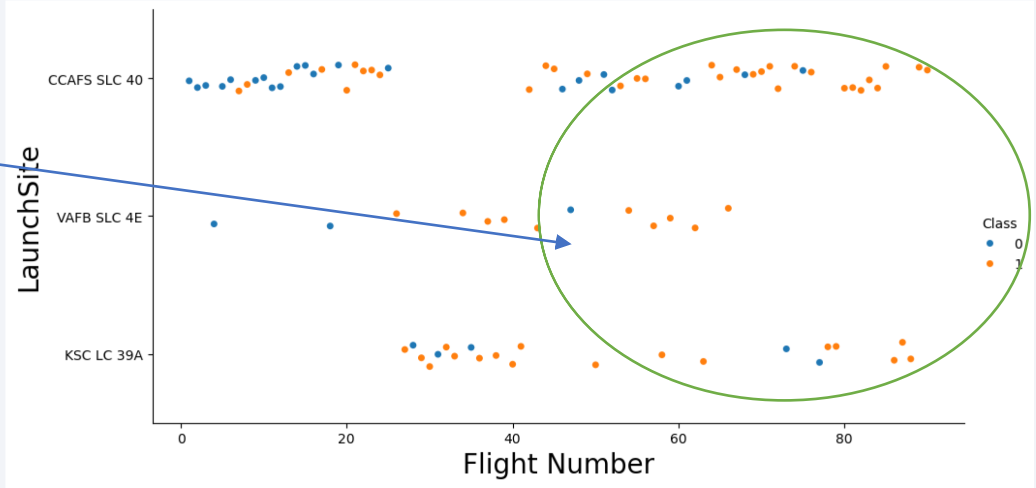
Flight Number vs. Launch Site

Increased success trend upon increasing flight number

Earlier flights has more failure rates (blue color)

Latest flights has more success rate (orange color)

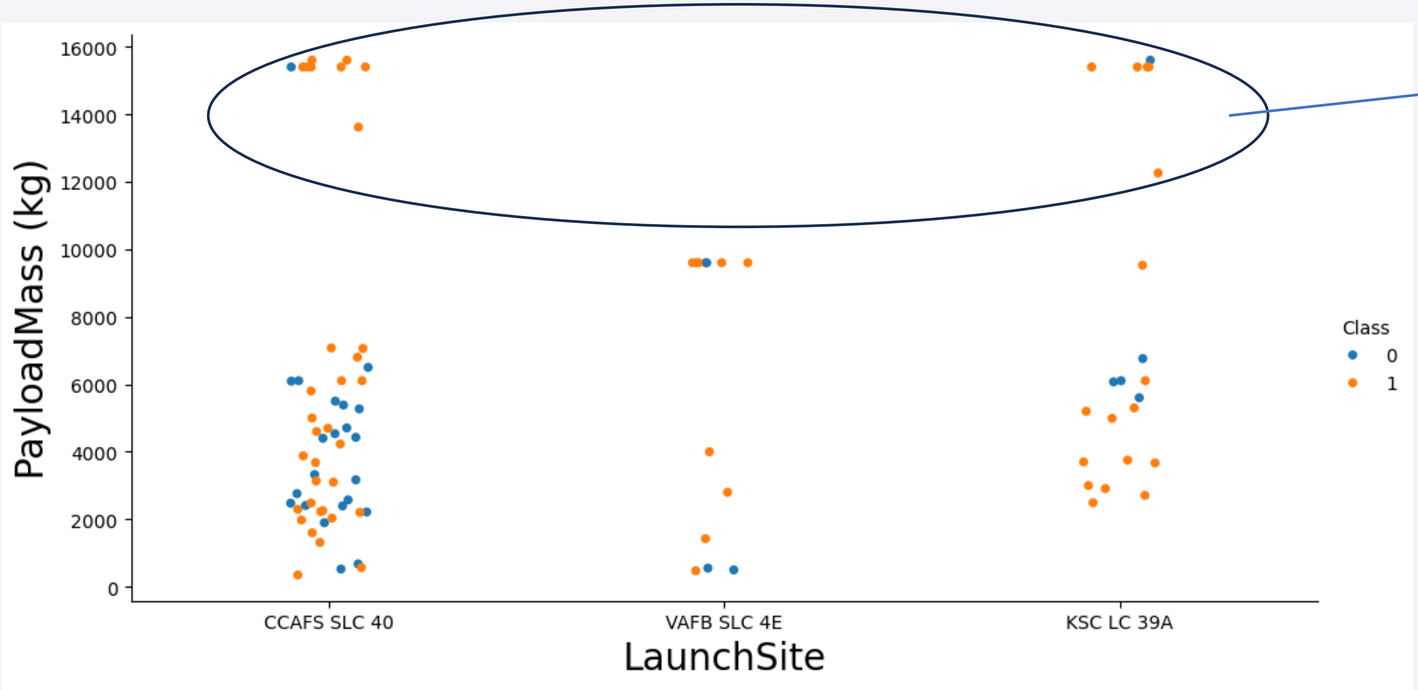
CCAFS SLC 40 launch site has majority of successful launches



Flight number vs. Launch site
Hue: class – success/failure

We see that as the flight number increases, the first stage is more likely to land successfully.

Payload vs. Launch Site



Payload mass (kg) vs. Launch site
Hue: class – success/failure

The more massive the payload, more likely the launch is successful.

Payload <= 8000 - KSC LC 39 A has good success rate

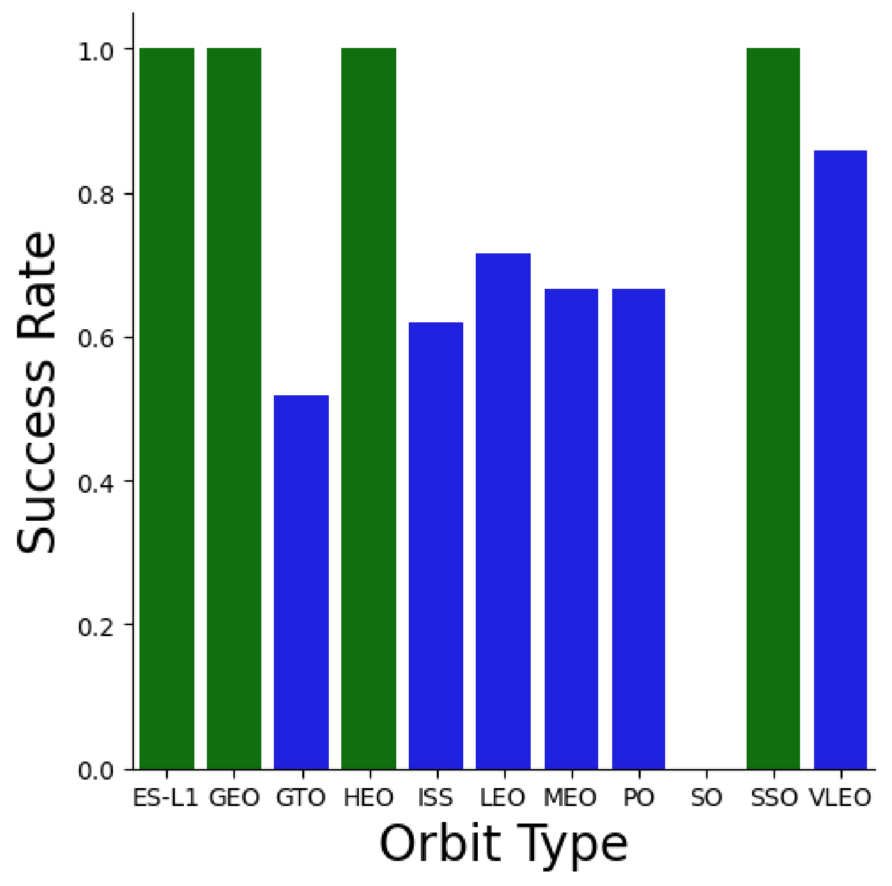
Payload < 8000 and > 10000 – good success rate of VAFB SLC 4E

Payload > 10000 – good success rate for both CCAFS SLC 40, KSC LC 39 A

Success Rate vs. Orbit Type

Analysis revealed the following results.

Orbit type	Success-Rate
ES-L1, GEO, HEO, SSO	**100 %**
GTO,ISS,LEO,MEO ,PO,VLEO	50-80 %
SO	0%



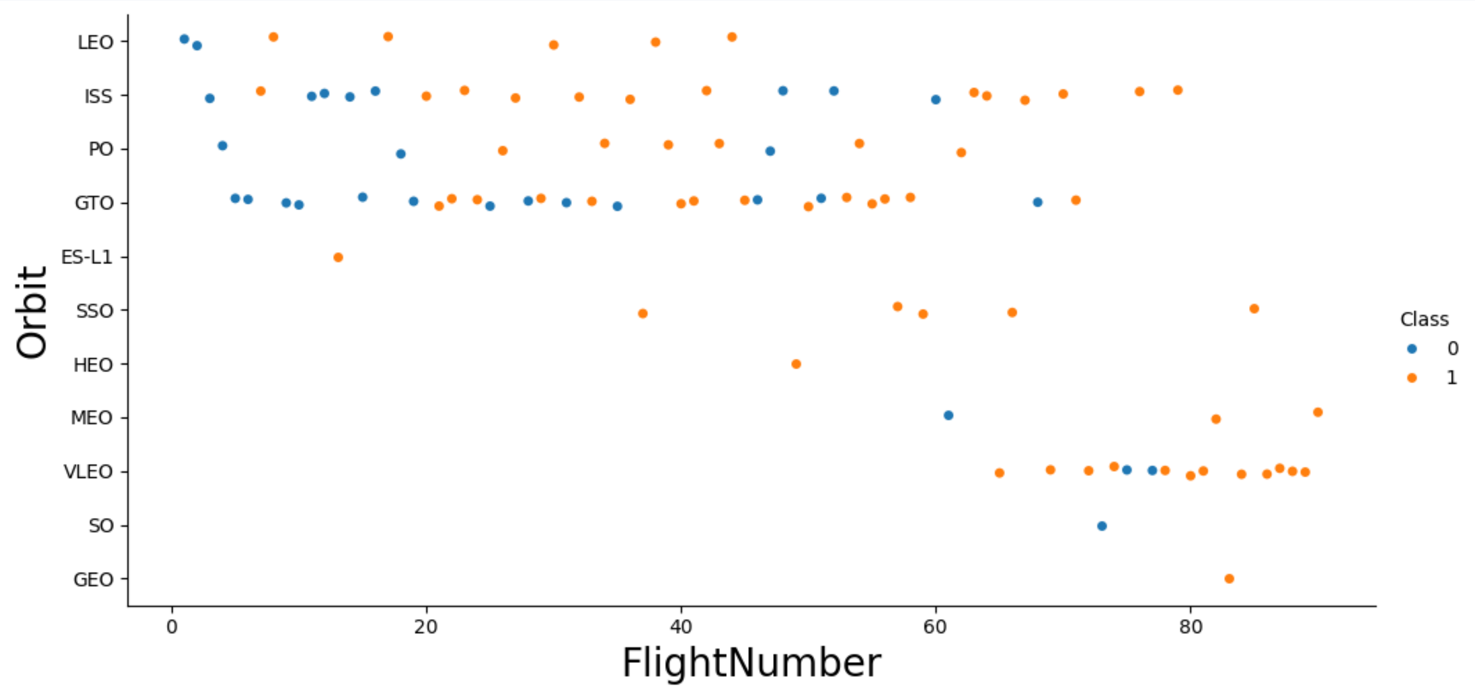
Orbit Type vs. Success Rate
Green – 100 % success rate
Blue- below 100 % success rate

Flight Number vs. Orbit Type

Most orbits - Increasing success rate with increasing flight number

LEO – Best follows the trend

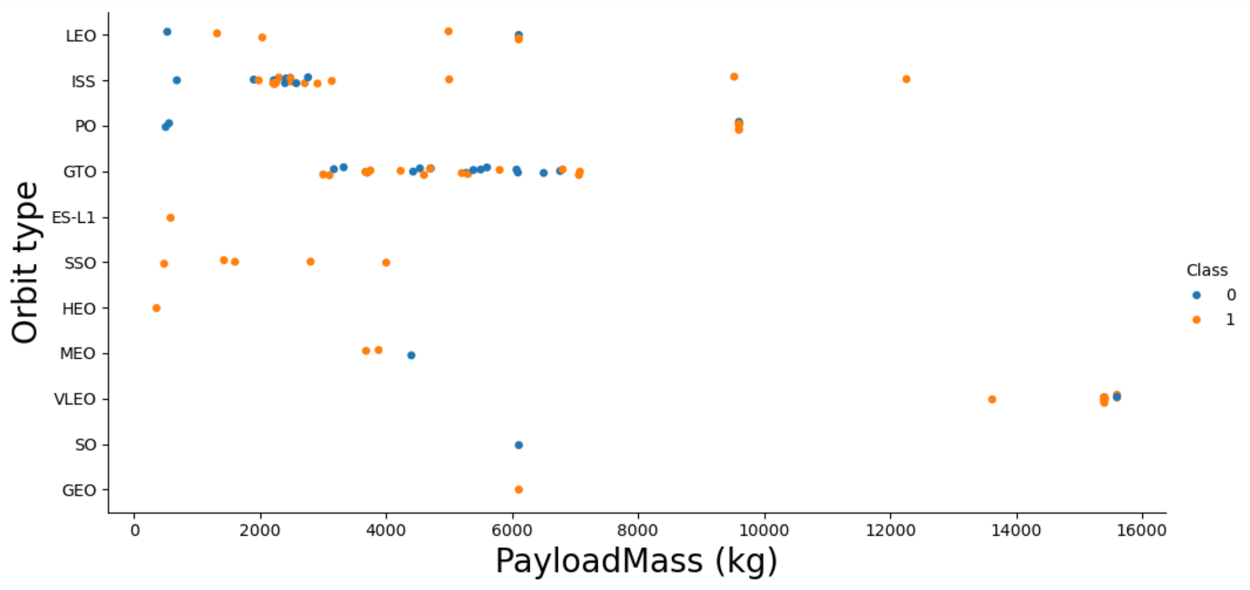
GTO – Mixed rates of success and failure with increased flight number



Flight number vs. Orbit
Hue: class – success/failure

Payload vs. Orbit Type

- Payloads > 10000 – Good success rate ISS, PO, VLEO
- 5000 < Payload > 10000 – good success rate for LEO and ISS
- Payload < 5000 – good success rate for LEO, ISS, Es-L1, SSO, HEO, MEO
- SSO has 100 % success rate for lighter payload mass
- GTO has a mixed rate trend for lighter payloads.

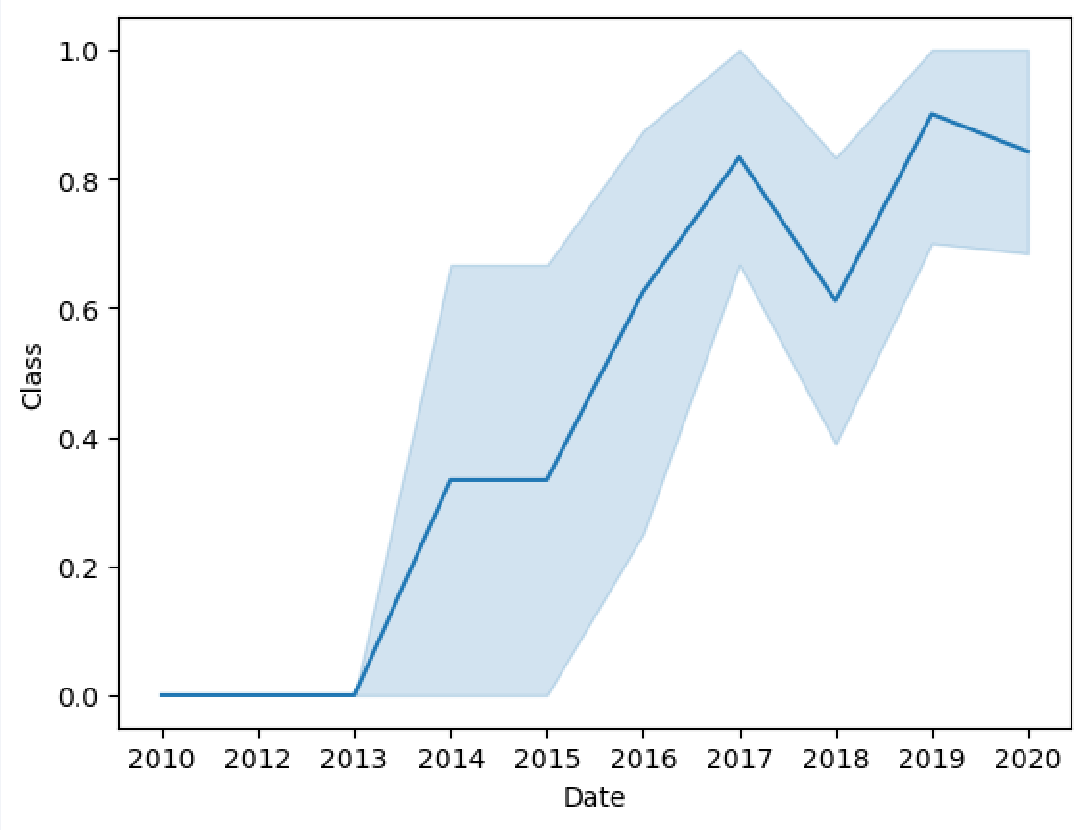


Payload mass vs. Orbit
Hue: class – success/failure

Launch Success Yearly Trend

Continuous increasing success rate after 2013 – 2017 and 2019 - 2020

Slight decrease at 2018



Year vs. success rate
Hue: class – success/failure

Launch Site infromation

1. List of all the launch sites

```
%sql SELECT DISTINCT Launch_Site as launch_sites FROM SPACEXTBL
```

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

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

SQL query using **DISTINCT** was used to obtain the names of all the launch sites

2 . First 5 launch site rows begin with `CCA`

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

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

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

SQL query using **WHERE clause's LIKE '%' and LIMIT** value; functions were used to obtain the first 5 rows of the dataset with launch site that starts with CCA.

Payload Mass

Total payload mass

To Calculate the total payload mass carried by boosters from NASA (CRS) the **SUM function** was used along with the WHERE clause to identify customer as NASA CRS

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

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

SUM (PAYLOAD_MASS__KG_)
45596

Average payload mass by F9 v1.1

To Calculate the average payload mass carried by booster version F9 v1.1 the **AVG function** was used along with the WHERE clause to identify the mentioned booster version

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

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

AVG (PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

To find the dates of the first successful landing outcome on ground pad SQL query was coded by selecting DATE and LAUNCH_SITE with a **WHERE** clause to filter landing outcome as Success(ground pad). As only the first date is required, the date is limited by **LIMIT 1**.

```
%sql SELECT  DATE, LAUNCH_SITE FROM SPACEXTBL WHERE LANDING_OUTCOME='Success (ground pad)' LIMIT 1;
```

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

Date	Launch_Site
2015-12-22	CCAFS LC-40

Successful Drone Ship Landing with Payload between 4000 and 6000

- To list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000, the query is performed using the following cluases.
- **WHERE** clause to filter landing outcome to Success(drone ship)
- Payload mass is limited using **Between 4000 AND 6000**

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

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

Booster_Version	PAYLOAD_MASS__KG_	Landing_Outcome
F9 FT B1022	4696	Success (drone ship)
F9 FT B1026	4600	Success (drone ship)
F9 FT B1021.2	5300	Success (drone ship)
F9 FT B1031.2	5200	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failure mission outcomes,

✓ **Sum** (*CASE WHEN Boolean_expression THEN result_expression [...n] [ELSE else_result_expression] END)* is used.

✓ The calculated total number of successful and failure missions are assigned to different columns

✓ **Group By** function is used to group the results according to the Mission_outcome.

```
%sql SELECT MISSION_OUTCOME, sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) \
AS "Number of successful missions", \
sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Number of failure missions" \
FROM SPACEXTBL\
GROUP BY MISSION_OUTCOME;
```

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

Mission_Outcome	Number of successful missions	Number of failure missions
Failure (in flight)	0	1
Success	98	0
Success	1	0
Success (payload status unclear)	1	0

Boosters Carried Maximum Payload

- The Booster version was selected using **DISTINCT** function.
- A **subquery** with a **WHERE** clause having **max ()** function was used to calculate the maximum payload mass inside the initial select query.

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass", \
PAYLOAD_MASS__KG_ FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Booster Versions which carried the Maximum Payload Mass	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

- The failed landing outcomes in drone ship displayed with their booster versions, and launch site names for the year 2015.
- SUBSTR(*string* FROM *start* FOR *length*) was used to extract the month and the year 2015 from the Date column.

```
%sql SELECT substr(DATE,6,2) as Month, LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL\
WHERE LANDING_OUTCOME = "Failure (drone ship)" AND substr(Date,0,5)='2015';
```

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

Month	Landing_Outcome	Booster_Version	Launch_Site
10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

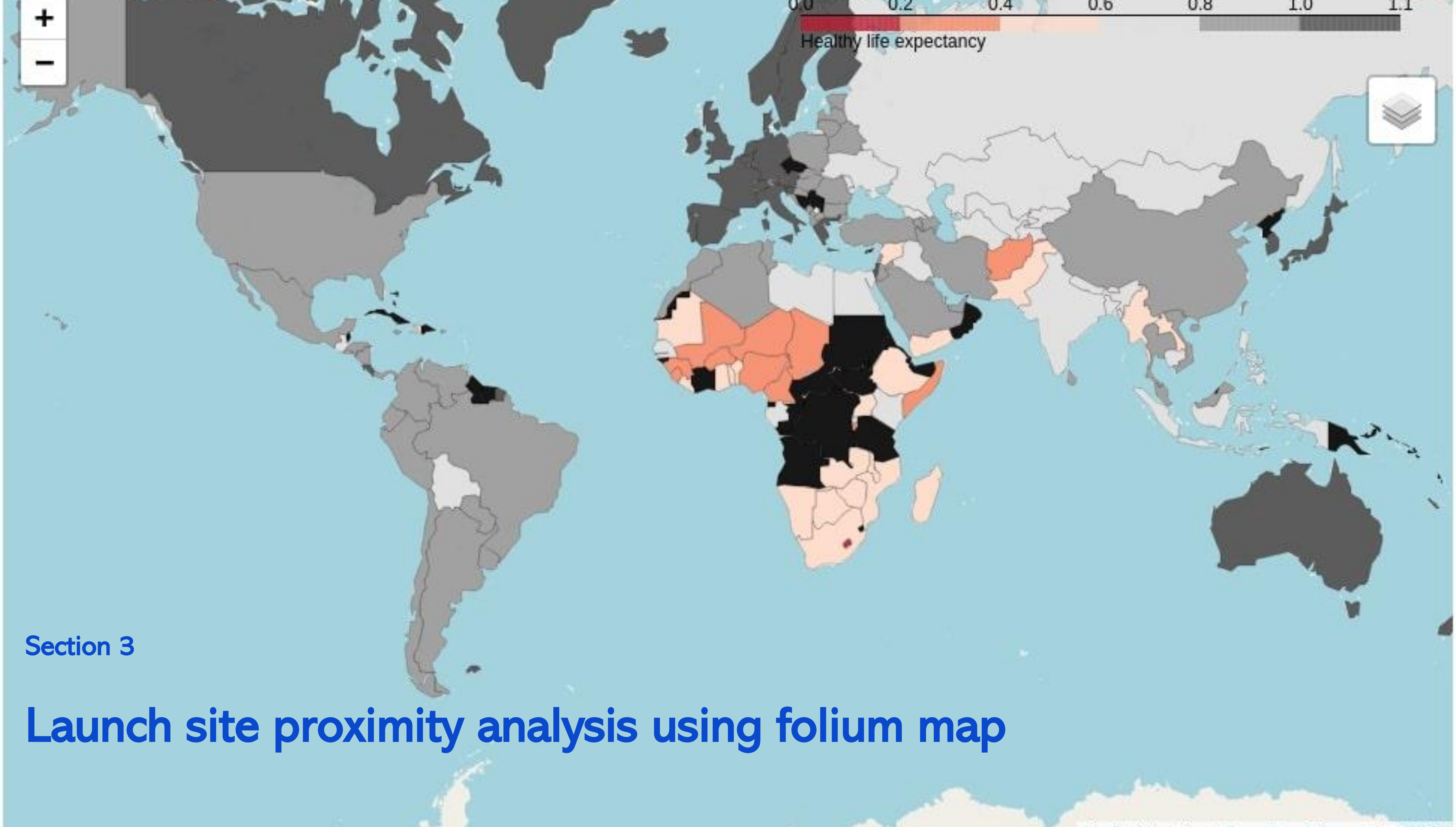
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- Count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) was calculated using the **COUNT ()** function.
- **WHERE** clause was used to restrict the count between particular period
- The landing outcomes are grouped by **GROUP BY** function
- The counted landing outcome value in a separate column is ranked by sorting the column using **SORT BY DESC** function.

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

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

Date	Landing_Outcome	Total count
2012-05-22	No attempt	10
2015-12-22	Success (ground pad)	5
2016-08-04	Success (drone ship)	5
2015-10-01	Failure (drone ship)	5
2014-04-18	Controlled (ocean)	3
2013-09-29	Uncontrolled (ocean)	2
2015-06-28	Precluded (drone ship)	1
2010-08-12	Failure (parachute)	1



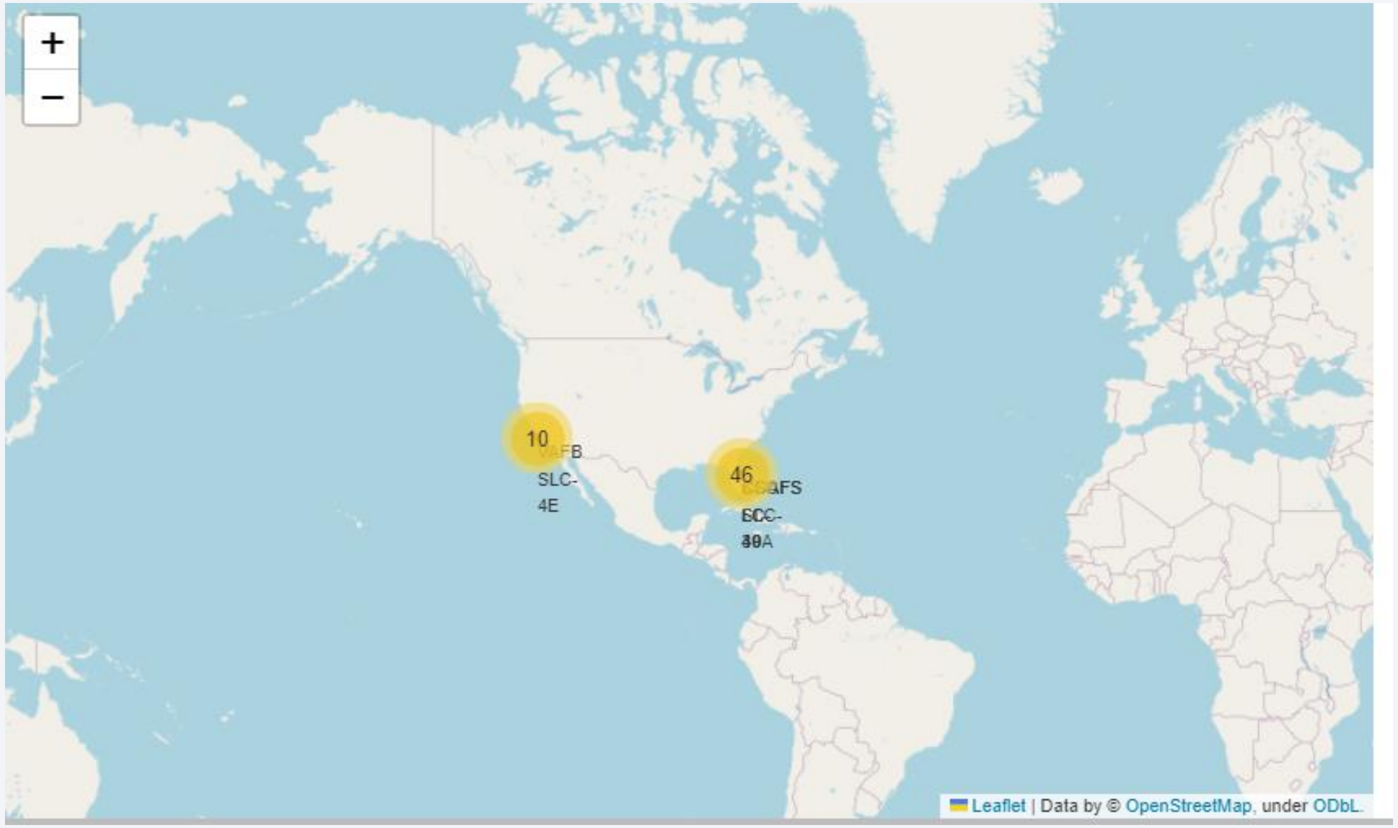
Section 3

Launch site proximity analysis using folium map

Folium map with launch sites and launches.

A folium circle and marker are used on the map to pinpoint the launch site and the number of launches associated with it.

Launch locations are on coast and closer to the equator than farther north. This is to benefit from the rotational velocity and favourable orbit paths. Launches are typically eastward to take advantage of the velocity boost to rocket velocity caused by the earth's rotation. As a result, launch sites are typically chosen

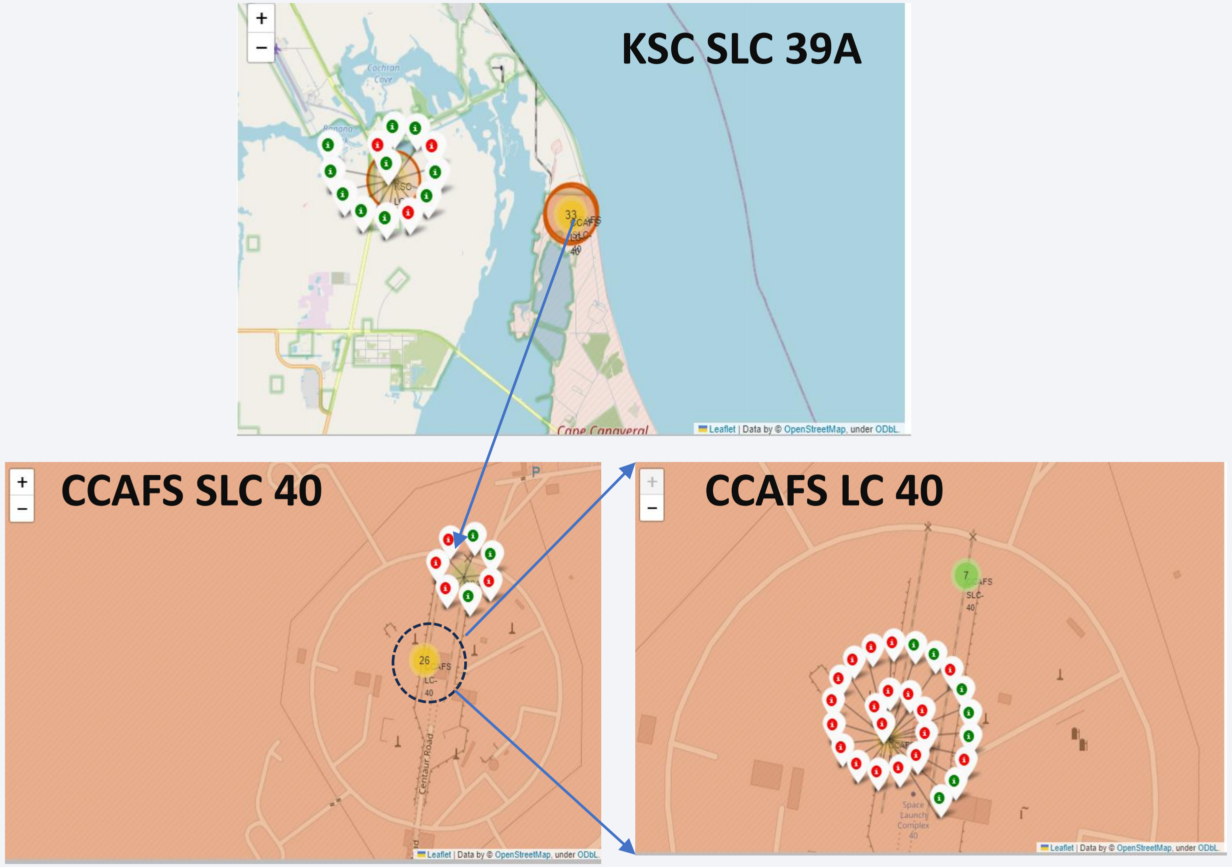


Launch outcomes

The images show the launch outcomes labelled with different popup colours.

The green popups shows successful outcome and the red ones are failure outcomes.

Site	Green	Red	Success %
KSCSLC 39A	**10**	**3**	**76 %**
CCAFS SLC 40	3	4	42 %
CCAFS LC 40	7	19	26 %



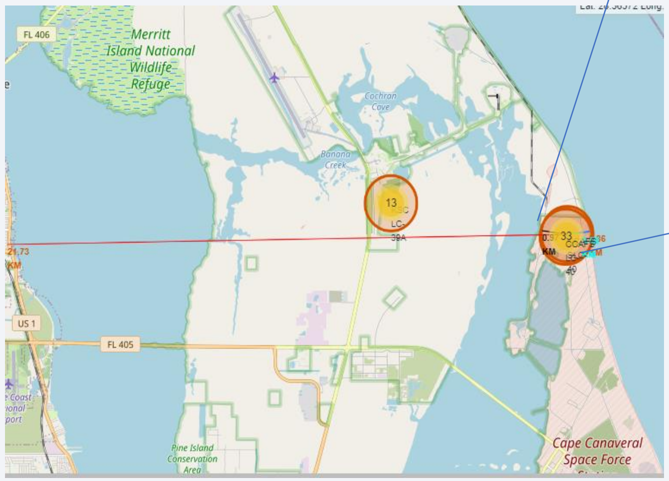
Launchsite CCAFS LC 40 and its close proximities

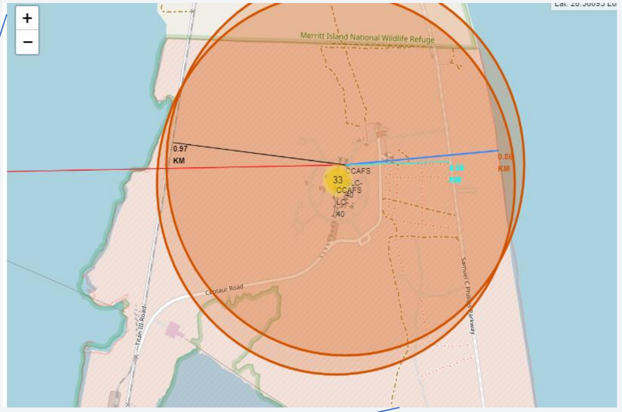
The folium map in image shows the lauch site CCAFS LC 40 with its proximities such as railway, highway, coastline, city with distance calculated.

Distance from launchsite to city: 21.72 km (Red line)
Distance from launchsite to railwayline: 0.97 km (Black line)
Distance from launchsite to highwayline: 0.58km (Aqua line)
Distance from launchsite to coastalline: 0.86 km (Blue line)

Observations:

- Far away from city for safety precautions.
- Close to railway line and highway to avail ample transpiration and infrastructure facilities
- Close enough to coast to gain advantage of using the water body for landing stages and also save public from mishaps



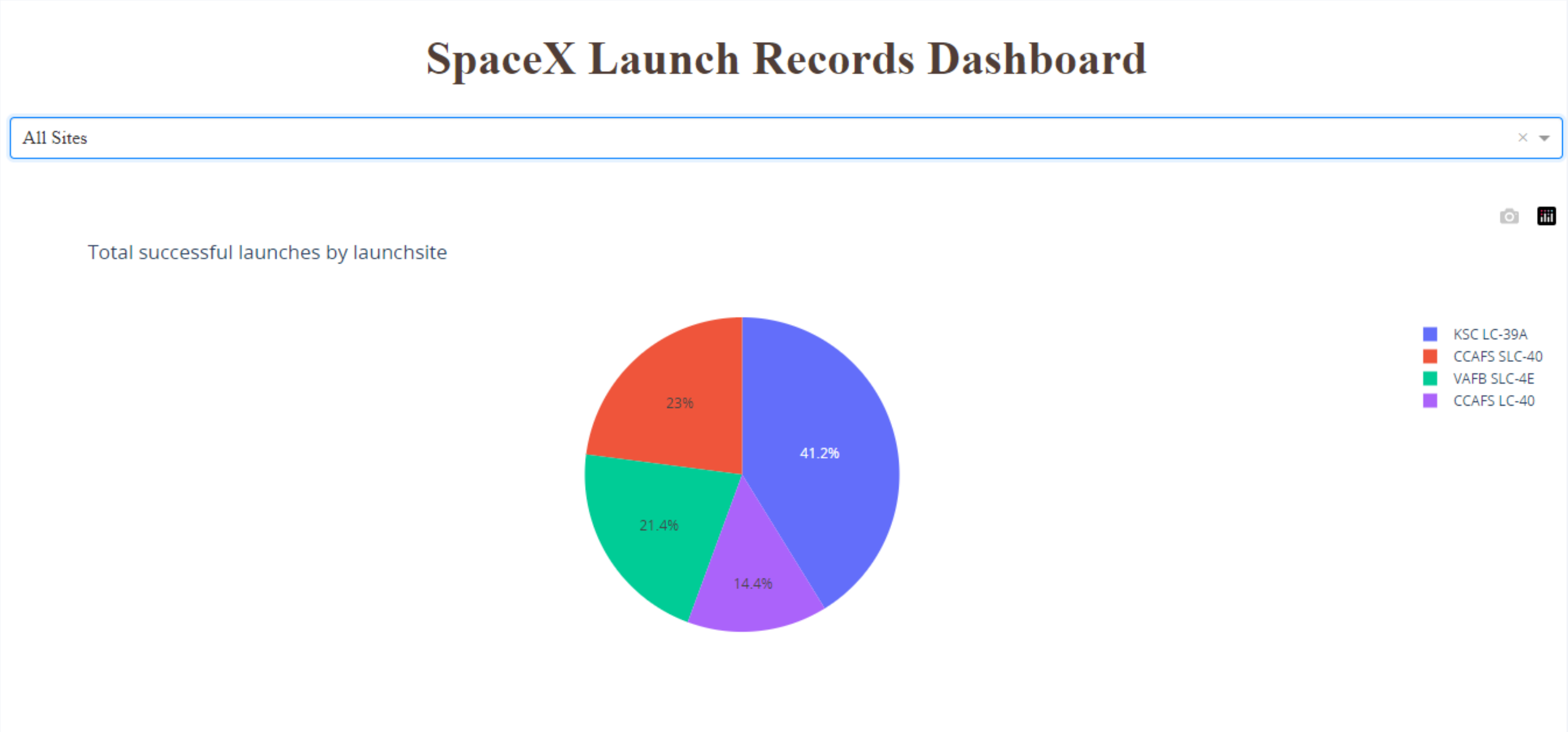


Section 4

Building dashboard with plotly Dash

SpaceX Launch records dashboard-

Success count of all sites



The dashboard app is created using dropdown and a pie chart.

The place holder with All sites shows the amount of contribution of each site on the overall success rate as follows,

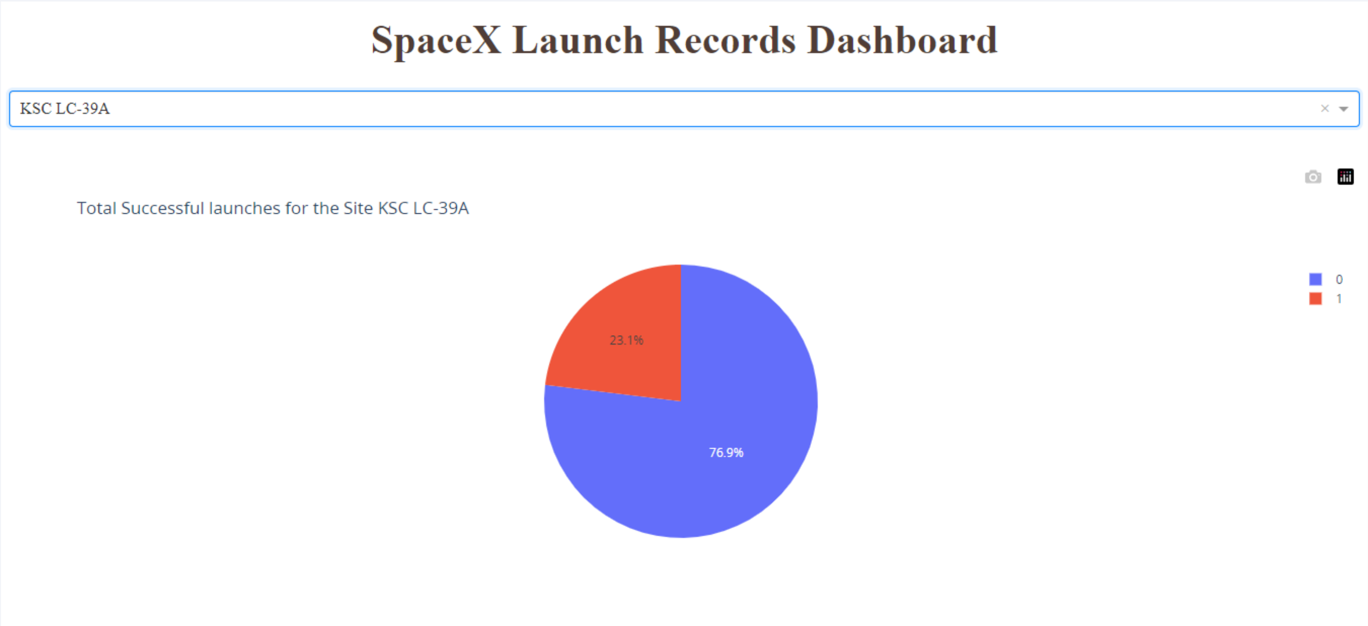
Site name	contribution
KSC LC 39A	41.2 %
CCAFS SLC 40	23 %
VAFBSLC 4E	21.4 %
CCAFS LC 40	14.4 %

SpaceX Launch records dashboard-

Success count of site with highest success ratio

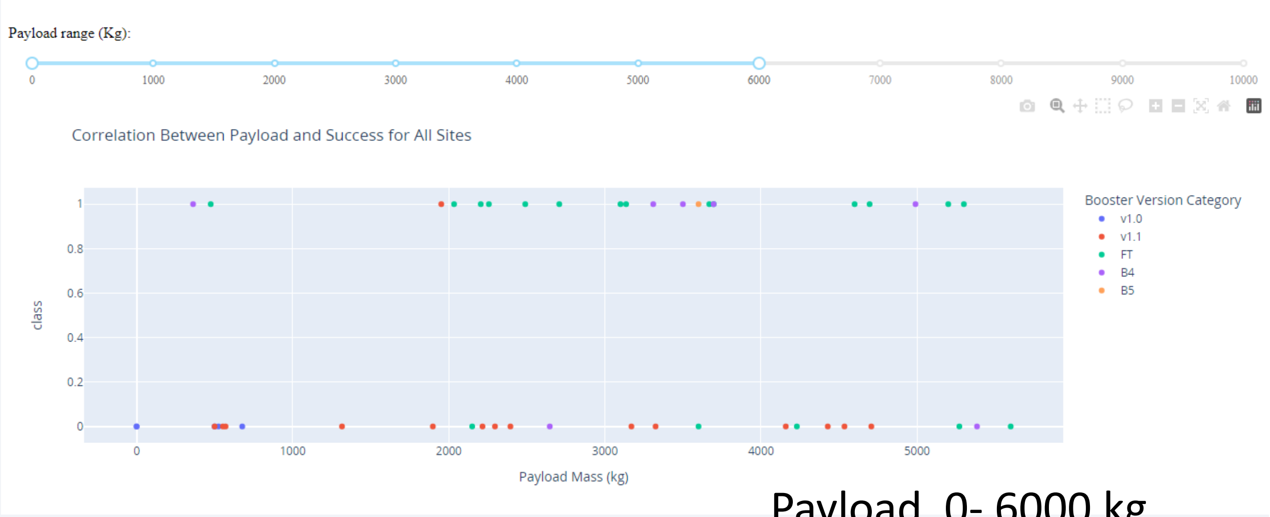
The Launch site KSC LC 39 A contributed nearly half of the success rate (ie) 41.7 %.

Having **10 successful** launch record among the total of 13 launches.



Payload mass vs. success rate for different booster versions

Payload 0 - 4000 kg



Payload 0- 6000 kg

Higher success rate when the payload range is between 2000 kg – 5000 kg

FT booster version (green dots) contributed most in this region

Payload 0- 10000 kg

Section 5

Predictive Analysis (Classification)

Classification Accuracy & Best model prediction

Accuracy scores:

Accuracy scores calculated is Identical for all the models.

The created accuracy score data frame is,

	Model	Jaccard Score	F1-Score	Accuracy Score
0	Logreg	0.5	0.888889	0.833333
1	SVM	0.5	0.888889	0.833333
2	D-Tree	0.5	0.888889	0.833333
3	KNN	0.5	0.888889	0.833333

Prediction of best model:

```
models = {'Logistic Regression':logreg_cv.best_score_,
          'Support Vector Machine': svm_cv.best_score_,
          'Decision Tree':tree_cv.best_score_,
           'K Nearest Neighbors':knn_cv.best_score_}

best_algorithm = max(models, key=models.get)
print('The best model is', best_algorithm,'with a score of', models[best_algorithm])
if best_algorithm == 'Logistic Regression':
    print('Best params is :', logreg_cv.best_params_)
if best_algorithm == 'Support Vector Machine':
    print('Best params is :', svm_cv.best_params_)
if best_algorithm == 'Decision Tree':
    print('Best params is :', tree_cv.best_params_)
if best_algorithm == 'K Nearest Neighbors':
    print('Best params is :', knn_cv.best_params_)
```

```
The best model is Decision Tree with a score of 0.8892857142857145
Best params is : {'criterion': 'gini', 'max_depth': 2, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sam
ples_split': 2, 'splitter': 'best'}
```

The best model is Decision Tree with a score of 0.8892857142857145

Confusion Matrix of Decision tree model

Summary

- ✓ Confusion matrix summarizes the performance of a classification algorithm.
- ✓ All models exhibited identical confusion matrix
- ✓ 3 observation is predicted positive and is actually negative.

Output:

True positive (TP) - 12

True negative (TN) - 3

False positive (FP)- 3

False Negative (FN) -0

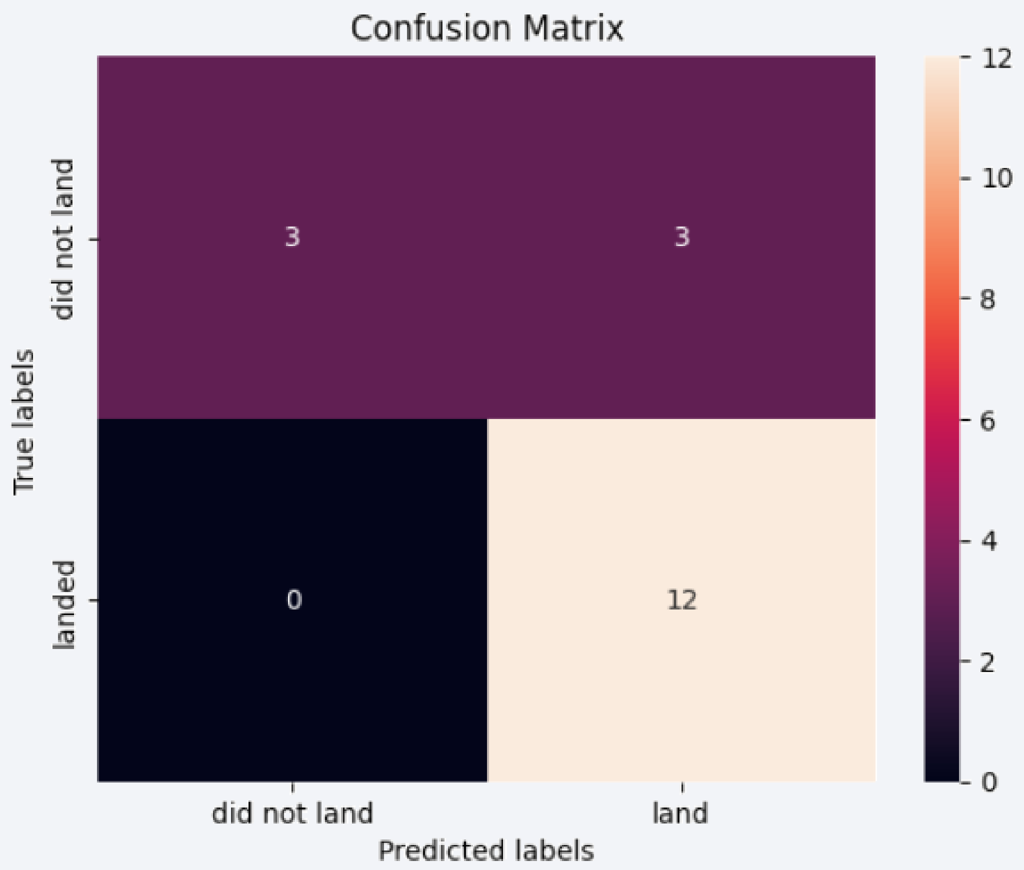
Precision = TP / (TP + FP) =12 / 15 = .80

Recall = TP / (TP + FN) • 12 / 12 = 1

F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

= 2 * (.8 * 1) / (.8 + 1) = 0.8888

Accuracy = (TP + TN) / (TP + TN + FP + FN) = 0.8333



Summary

SQL queries and data visualization tools are used to analyze the main aspects that impact the launch outcome.

The following are the observations:

- The equator and coast are near launch site sites that were determined via folium map analysis to be advantageous because of the earth's spinning speed.
- CCAFS SLC 40 launch pad and LEO orbit have a high success rate as the number of flights increases.
- KSC LC 39A and CCAFS SLC 40 achieved excellent success rates for larger payload launches.
- Orbital payload dependence showed that SSO is optimal for lower masses whereas ISS, PO, and VLEO are optimal for bigger masses.
- The launchsite KSC LC 39A demonstrated the highest overall success rate of 76%, accounting for 42% of the total launches' success rate.
- Models of classification Logistic regression, Supporvector machine, Decision Tree, and K closest neighbours all produced equal accuracy and confusion matrices.
- The decision tree model with optimal parameters was predicted to be the best.

Conclusion

- Given that models rely heavily on data, which may or may not have varying degrees of ease of classification, a larger data collection may aid in the construction of a more appropriate model with a longer lifespan.
- The accuracy levels of the predicted models were equal. Precision and recall numbers can at times provide some statistics when we observe equal accuracy values. The confusion matrices are also the same in this case.
- Therefore, in addition to the trained model, a classifier such as the Random Forest classifier can be incorporated. It is an ensemble learning method composed of decision trees that combines many classifiers to improve a model's performance.


Thank you !
SPACEX