IBM DATA SCIENCE CAPSTONE PROJECT

Space X Falcon 9 Landing Analysis

Kuan-Hua Wang

09/2023

Outline

1 Executive Summary

Summary of Methodologies

- Data Collection
- Data Wrangling
- Exploratory Data Analysis
- Interactive Visual Analysis
- Predictive Analysis (Classification)

Summary of Results:

- Exploratory Data Analysis (EDA) results
- Map analytics
- Interactive dashboard
- Predictive analysis result

2 Introduction

- In this capstone, we will predict if the Falcon 9 first stage will land successfully.

- 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.

Methodology

3

Methodology Summary

1 Data Collection

- Requesting rocket launch data from SpaceX API
- Web Scraping
- Dealing with Missing Values
  - ✓ Using the .mean()and the .replace() to replace np.nan

2 Data Wrangling

- Using the .value_counts() method
  - ✓ Number of launches on each site
  - ✓ Number and occurrence of each orbit
  - ✓ Number and occurrence of mission outcome per orbit type
- Creating the booster landing outcome label
  - ✓ 0 - failure / 1- success

3 Exploratory Data Analysis

- Using Matplotlib to visualize and find out patterns
- Using SQL queries to manipulate and evaluate the dataset

4 Interactive Visual Analysis

- Using Folium to Geospatial analysis
- Creating an interactive dashboard using Plotly Dash

5 Predictive Analysis

- Using Scikit-Learn to:
  - ✓ Pre-process (standardize) the data
  - ✓ Split the data into training and testing data using train_test_split
  - ✓ Train different classification models
  - ✓ Find hyperparameters using GridSearchCV
- For each classification model
  - ✓ Assessing the accuracy
  - ✓ Plotting confusion matrices

1 Data Collection with API

1

Request to the SpaceX API
start requesting rocket launch data from SpaceX API with the URL

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

response = requests.get(spacex_url)
```

2

Turn it into a Pandas dataframe using .json_normalize()

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

3

Clean the requested data

- Use and call custom logic function to clean the data to retrieve data and fill the lists
- The data from these requests will be stored in lists and will be used to create a new dataframe
- Create a Pandas data frame from the dictionary launch_dict.

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

```
# Call getBoosterVersion
getBoosterVersion(data)
```

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

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

```
# Call getCoreData
getCoreData(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}
```

```
# Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

1 Data Collection with API

4 Filter the dataframe to only include Falcon 9 launches

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

5 removed some values we should reset the FlgihtNumber column

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

6 Dealing with Missing Values

```
# Calculate the mean value of PayloadMass column
data_falcon9_mean=data_falcon9['PayloadMass'].mean()

# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan,data_falcon9_mean)
```

1 Data Collection with Web Scraping

1

Request the Falcon9 Launch Wiki page from its URL

- let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response
- Create a BeautifulSoup object from the HTML response

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_ar
# use requests.get() method with the provided static_url
response = requests.get(static_url)
# assign the response to a object
data = response.text
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data, 'html.parser')
```

2

Extract all column/variable names from the HTML table header

```
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 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)
```

3

Create a data frame by parsing the launch HTML tables

```
df = pd.DataFrame.from_dict(launch_dict,orient='index').T
```

```
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']=[]
```

2 Data Wrangling

1

Using the .value_counts() method

1. Number of launches on each site
2. Number and occurrence of each orbit
3. Number and occurrence of mission outcome per orbit type

2

Creating the booster landing outcome label

- 1 means the booster successfully landed
- 0 means it was unsuccessful

2 Data Wrangling

1-1 Calculate the number of launches on each site

The data contains several **Space X launch facilities:**
Cape Canaveral Space Launch Complex 40 **VAFB SLC 4E** , Vandenberg Air Force Base Space Launch Complex 4E (**SLC-4E**), Kennedy Space Center Launch Complex 39A **KSC LC 39A .**
The location of each Launch Is placed in the column **LaunchSite**

```
# 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
```

2 Data Wrangling

1-2 Each launch aims to an dedicated orbit

Calculate the number and occurrence of each orbit

Use the method .value_counts() to determine the number and occurrence of each orbit in the column 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
```

2 Data Wrangling

1-3 Calculate the number and occurence of mission outcome per orbit type

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it to a variable landing_outcomes.

```
# 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
```

`True Ocean` means the mission outcome was successfully landed to a specific region of the ocean while `False Ocean` means the mission outcome was unsuccessfully landed to a specific region of the ocean. `True RTLS` means the mission outcome was successfully landed to a ground pad `False RTLS` means the mission outcome was unsuccessfully landed to a ground pad. `True ASDS` means the mission outcome was successfully landed to a drone ship `False ASDS` means the mission outcome was unsuccessfully landed to a drone ship. `None ASDS` and `None None` these represent a failure to land.

2 Data Wrangling

2. Create a landing outcome label from Outcome column

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

We create a set of outcomes where the second stage did not land successfully:

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

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

landing_class = []

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

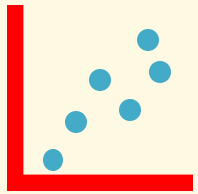
```
print(landing_class)
```

```
[0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1]
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

3 EDA with Matplotlib

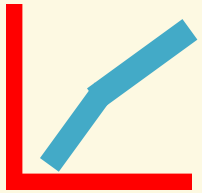
SCATTER CHART



- **Flight Number and Launch Site**
- **Payload and Launch Site**
- **FlightNumber and Orbit type**
- **Payload and Orbit type**

Observe relationships, or correlations, between two numeric variables.

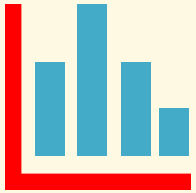
LINE CHART



- **Launch success yearly trend**

Contain numerical values on both axes, and are generally used to show the change of a variable over time.

BAR CHART



- **Success rate of each orbit type**

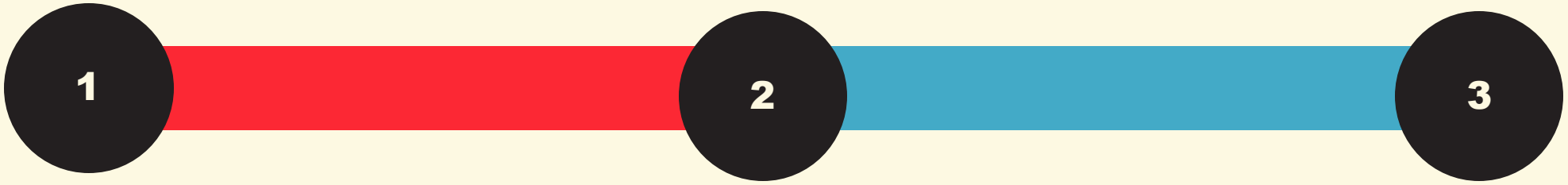
Compare a numerical value to a categorical variable. Horizontal or vertical bar charts can be used, depending on the size of the data.

3 EDA with SQL

To gather some information from the dataset, use SQL queries

1. Display the names of the unique launch sites in the space mission
2. Display 5 records where launch sites begin with the string ‘CCA’
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display the average payload mass carried by booster version F9 v1.1
5. List the date when the first successful landing outcome on a ground pad was achieved
6. List the names of the boosters which had success on a drone ship and a payload mass between 4000 and 6000 kg
7. List the total number of successful and failed mission outcomes
8. List the names of the booster versions which have carried the maximum payload mass
9. List the failed landing outcomes on drone ships, their booster versions, and launch site names for 2015
10. 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

4 Interactive Visual Analysis with Folium



Mark all launch sites on a map

Create and add folium.Circle and folium.Marker for each launch site on the site map

Mark the success/failed launches for each site on the map

If a launch was successful (class=1), then we use a green marker and if a launch was failed, we use a red marker (class=0)

Calculate the distances between a launch site to its proximities

4 Interactive Visual Analysis

1

Pie chart

Success Count for all launch sites

1. Which site has the largest successful launches?
2. Which site has the highest launch success rate?

- Add a dropdown list to enable Launch Site selection
- Add a pie chart to show the total successful launches count for all sites

2

Scatter chart

Correlation between payload and success or all sites

1. Which payload range(s) has the highest launch success rate?
2. Which payload range(s) has the lowest launch success rate?
3. Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate?

5 Predictive Analysis

1 Model Development

- Perform exploratory Data Analysis and determine Training Labels
- create a column for the class
- Standardize the data
- Split into training data and test data

2 Model Evaluation

- For SVM, Classification Trees and Logistic Regression
- Calculate the accuracy on the test data using the method score
- Fit the object to find the best parameters from the dictionary parameters

3

Find the best performing classification model

Result

Exploratory Data Analysis

Interactive Visual Analysis

Predictive Analysis

Exploratory Data Analysis

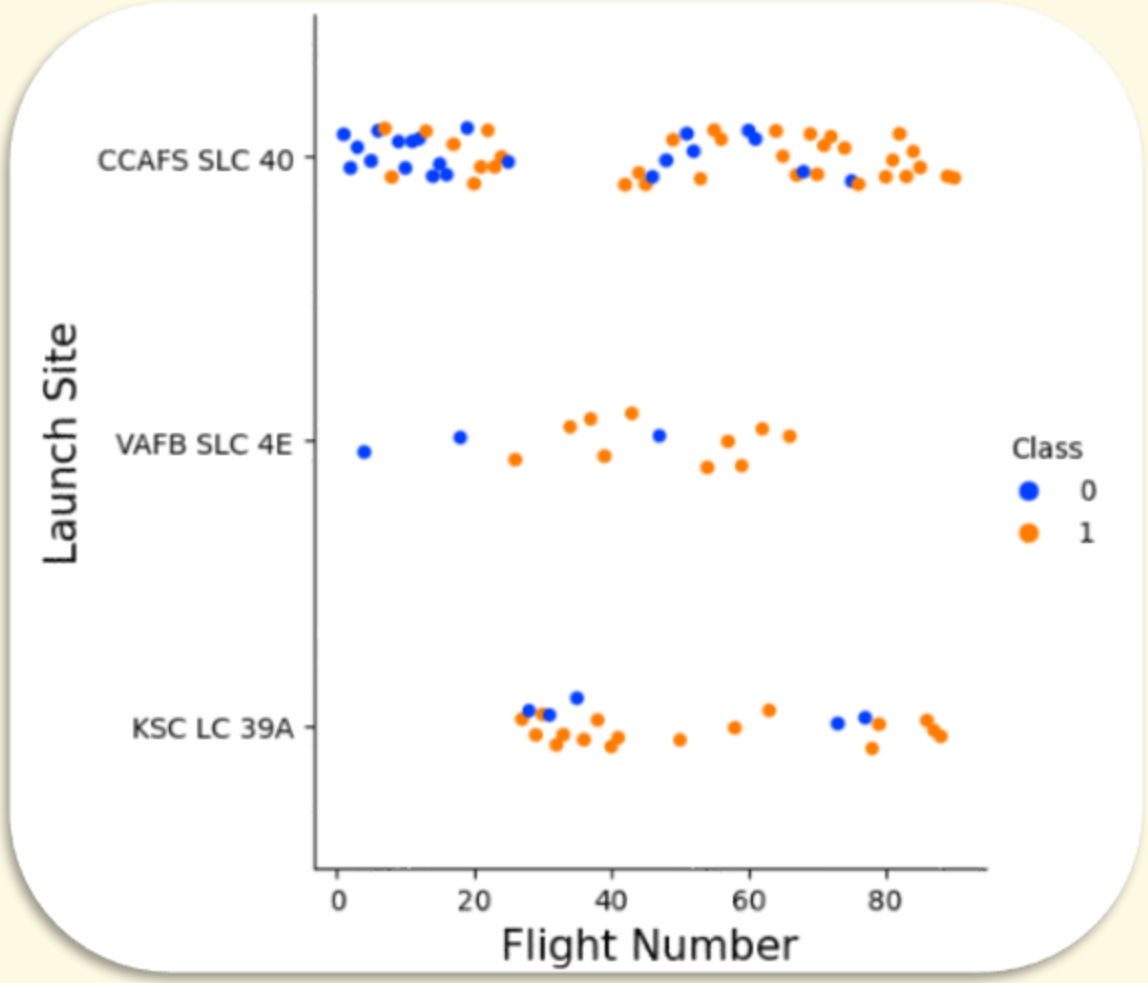
1 EDA with Matplotlib

Flight Number VS. Launch Site

- **Generally speaking, as the number of flights increases, the success rate also increases.**
- **CCAFS SLC 40 has a lot of early flights and a high unsuccessful rate.**
- **VAFB SLC 4E and KSC LC 39A have a high success rate.**
- **KSC LC 39A doesn't have early flights and has a high success rate.**

Label

- 1 means the booster successfully landed
- 0 means it was unsuccessful



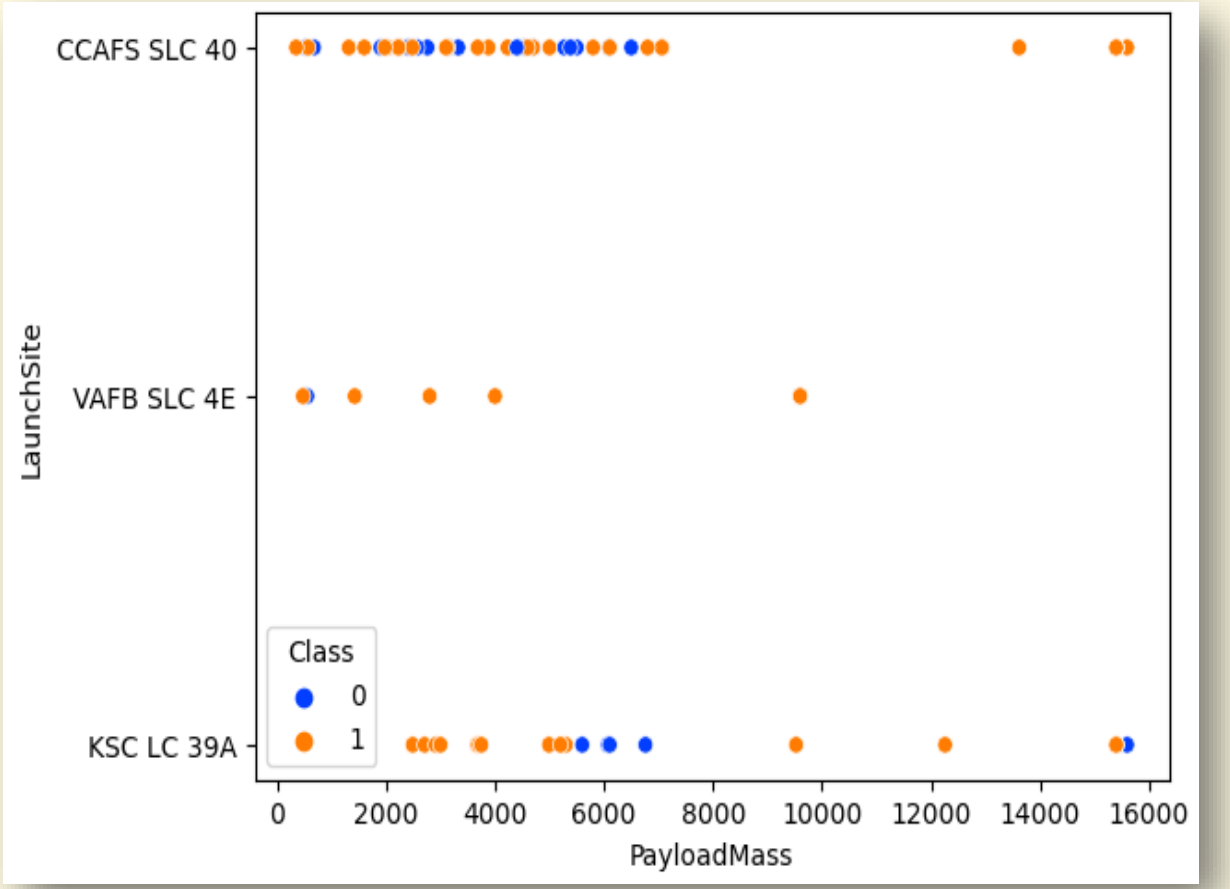
1 EDA with Matplotlib

Payload VS. Launch Site

- **Generally, Launchsite with over 7000 PayloadMass has a high success rate.**
- **For VAFB SLC 4E and KSC LC 39A, if PayloadMass is lower than 5500, it has a high success rate.**
- **VAFB SLC 4E are no rockets launched for heavypayload mass(greater than 10000).**

Label

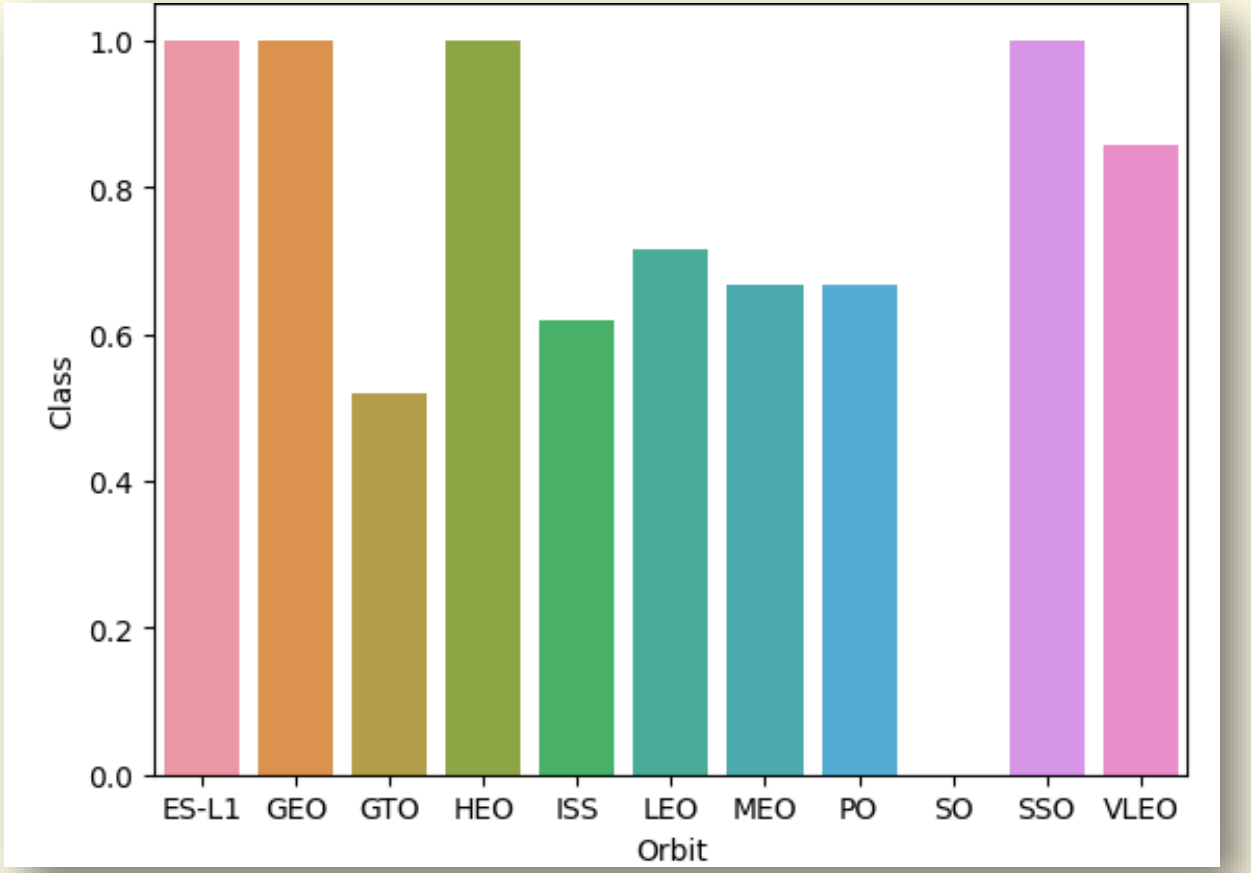
- 1 means the booster successfully landed
- 0 means it was unsuccessful



1 EDA with Matplotlib

Success rate VS. Orbit type

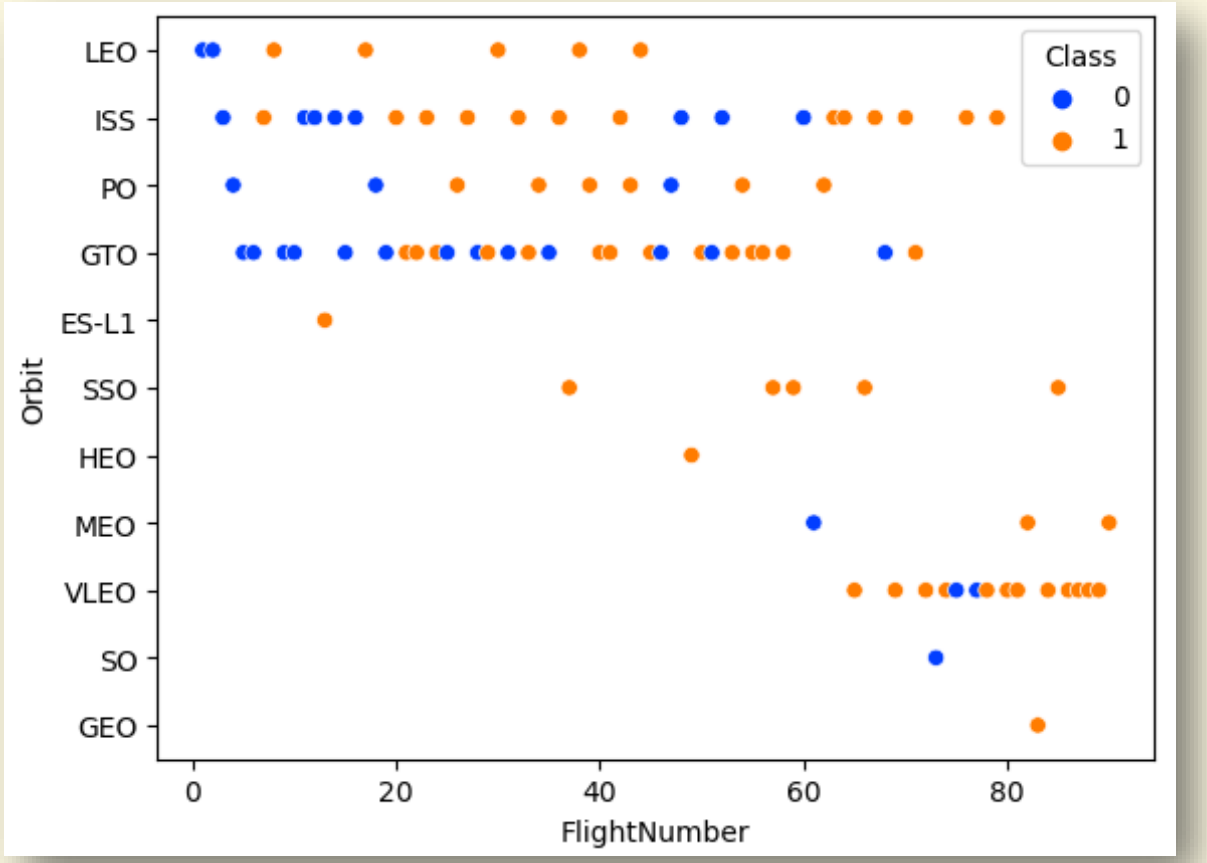
- Orbit with 100% success rate.
  - ES-L1, GEO, HEO, SSO
- Orbit with 0 % success rate
  - SO
- Orbit with success rate between 50%~75%
  - GTO, ISS, MEO, PO, LEO, VLEO



1 EDA with Matplotlib

FlightNumber VS. Orbit type

- **Generally, Flight Number increases, the success rate increases.**
- **For LEO, only the early launches are unsuccessful and other launches are successful.**
- **ES-L1, SSO, HEO, and GEO have 100% success rate, but they have low Flight numbers.**



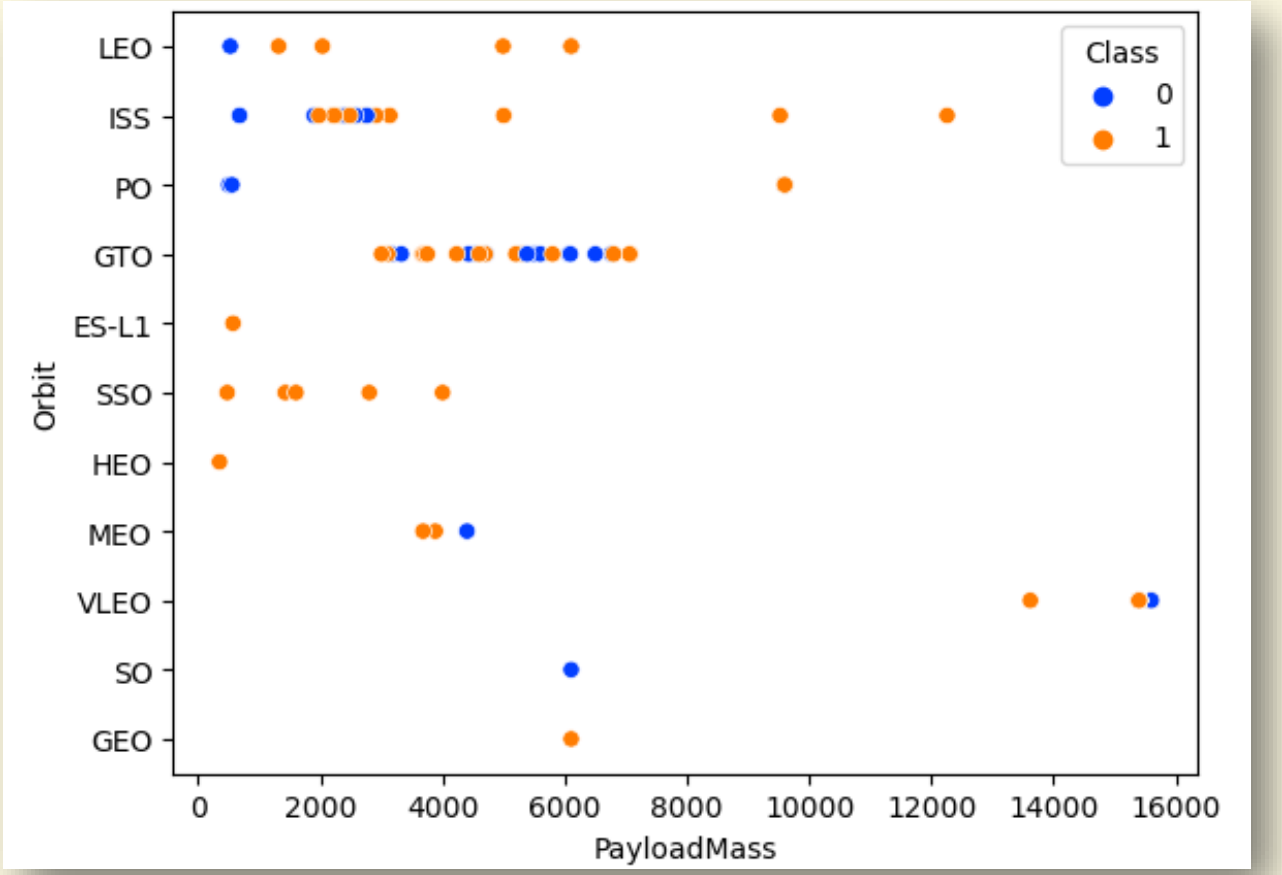
Label

- 1 means the booster successfully landed
- 0 means it was unsuccessful

1 EDA with Matplotlib

PayloadMass VS. Orbit type

- **For LEO, ES-L1, SSO, and HEO, their PayloadMass are lower than 6000 and have a high success rate.**
- **LEO, ISS, and PO have higher success rates for heavy PayloadMass.**



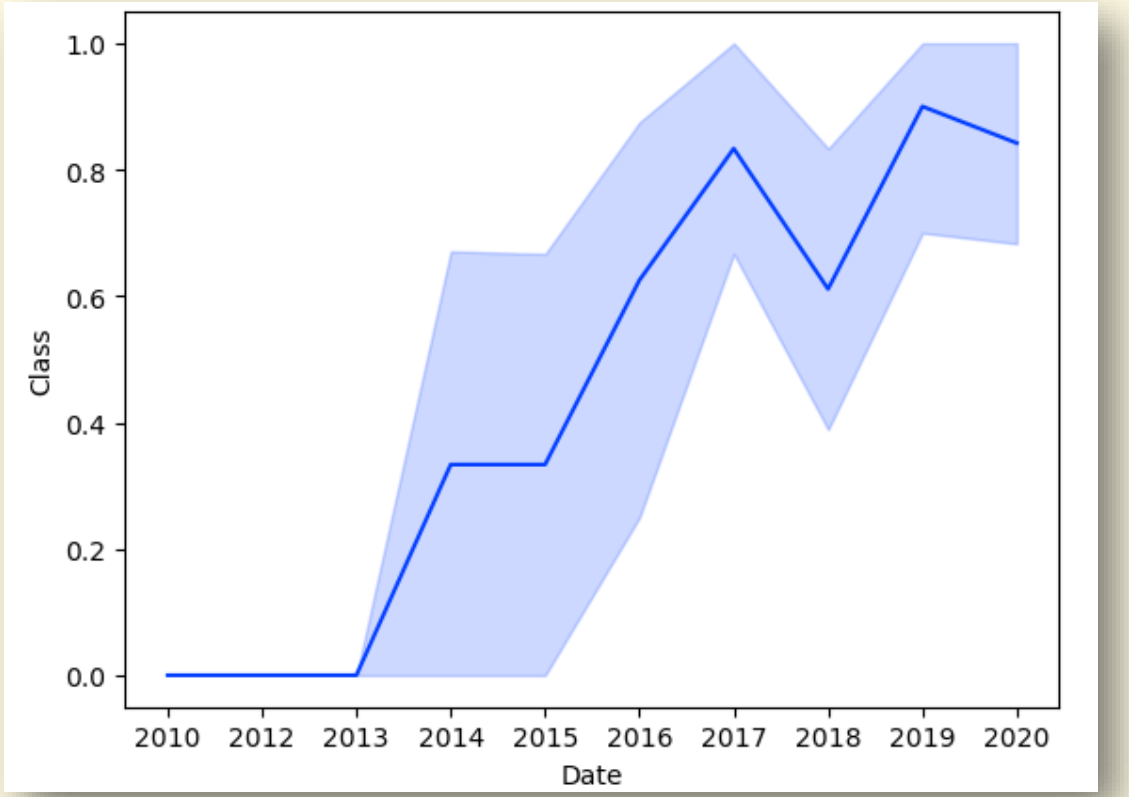
Label

- 1 means the booster successfully landed
- 0 means it was unsuccessful

1 EDA with Matplotlib

Timeline

- They have a unsuccess rate between 2010 AND 2013.
- Generally, their success rate rapidly increases after 2013.
- After 2016 success rate rapidly decreased until 2018, then after 2018 the success rate increased.
- Generally, the success rate is over 50% after 2016.



2 EDA with SQL

Task 1

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

```
%sql SELECT Launch_Site  FROM SPACEXTBL group by Launch_Site;
```

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

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

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

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

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

2 EDA with SQL

Task 2

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

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

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Task 3

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

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

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

total_payload_mass
45596

2 EDA with SQL

Task 4

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

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

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

average_payload_mass
2534.6666666666665

2 EDA with SQL

Task 5

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

Hint:Use min function

```
%sql SELECT Landing_Outcome FROM SPACEXTBL GROUP BY Landing_Outcome;
```

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

Landing_Outcome
Controlled (ocean)
Failure
Failure (drone ship)
Failure (parachute)
No attempt
No attempt
Precluded (drone ship)
Success
Success (drone ship)
Success (ground pad)
Uncontrolled (ocean)

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

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

first_succesful_landing
2015-12-22

2 EDA with SQL

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_ >4000 AND  PAYI
```

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

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

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT Mission_Outcome,COUNT(Mission_Outcome) AS total_mission_outcomes FROM SPACEXTBL GROUP BY Mission_Outcome ;
```

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

Mission_Outcome	total_mission_outcomes
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

2 EDA with SQL

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.
```

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

2 EDA with SQL

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, 6, 2) AS month ,Booster_Version, Launch_Site  FROM SPACEXTBL WHERE Landing_Outcome='Failure (drone
```

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

month	Booster_Version	Launch_Site
10	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

2 EDA with SQL

Task 10

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.

```
%sql SELECT Landing_Outcome,count(Landing_Outcome) as count_of_landing_outcomes  FROM SPACEXTBL \
WHERE  Date between '2010-06-04' and '2017-03-20' \
group by Landing_Outcome\
order by count_of_landing_outcomes  desc ;
```

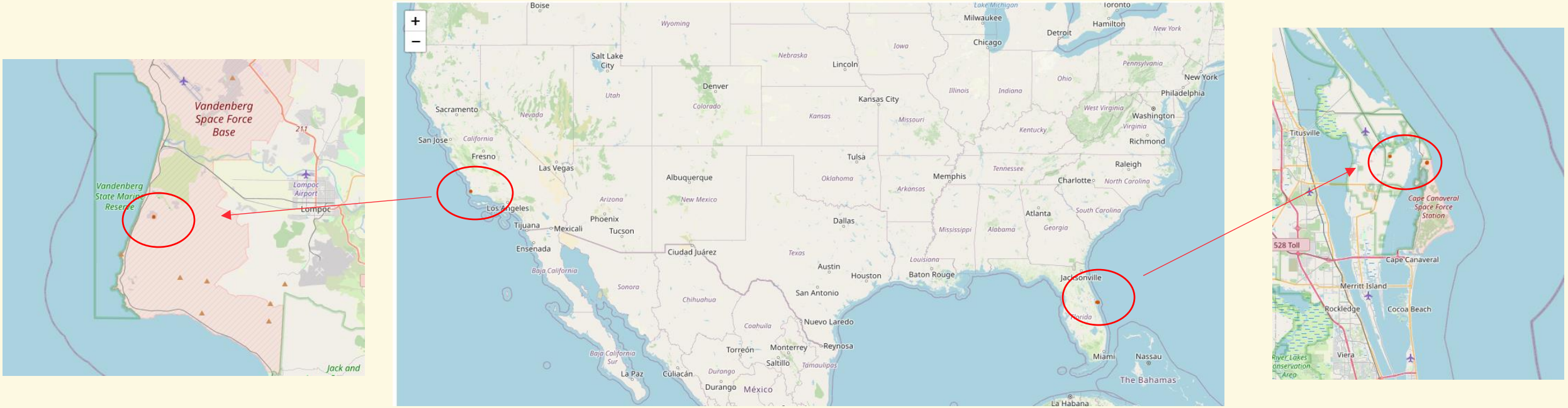
```
* sqlite:///my_data1.db
Done.
```

Landing_Outcome	count_of_landing_outcomes
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Interactive Visual Analysis

1 Interactive Visual Analysis with Folium

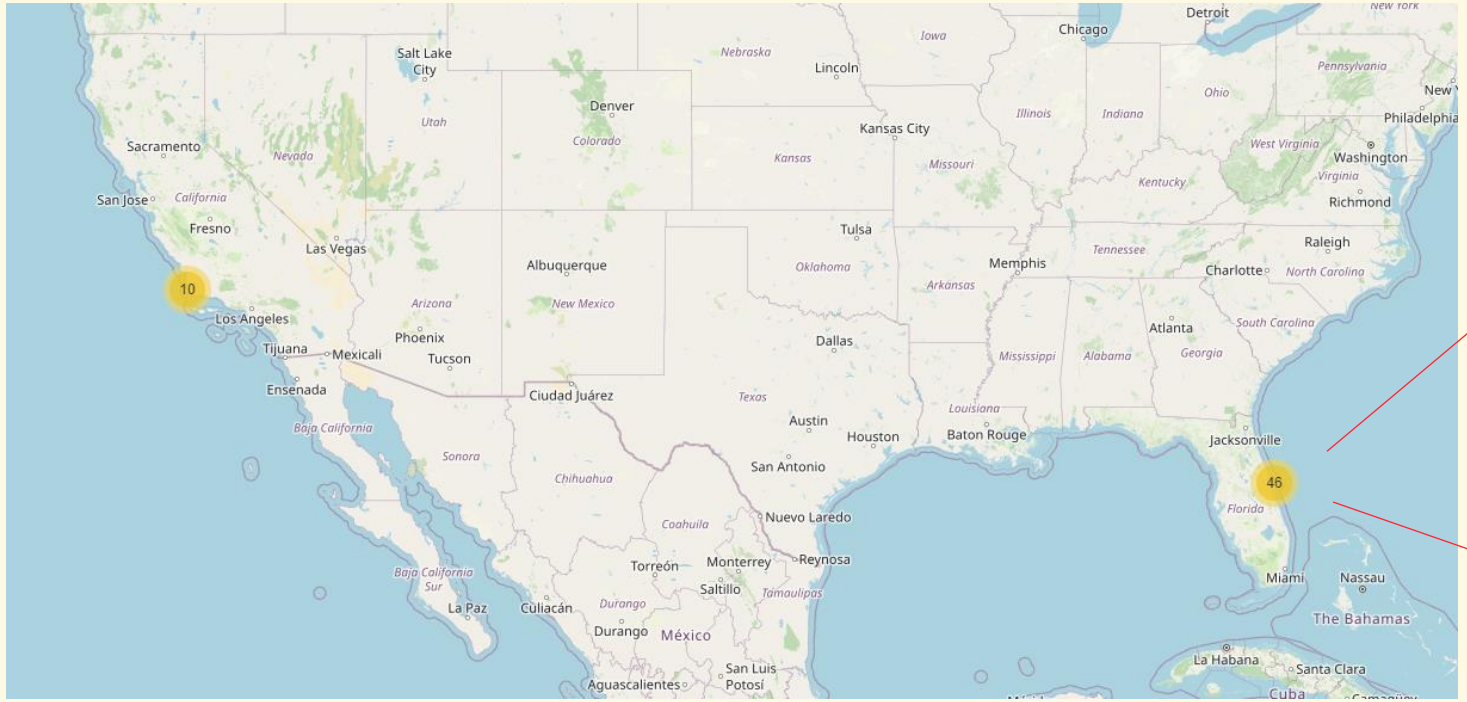
- **TASK 1:** Mark all launch sites on a map

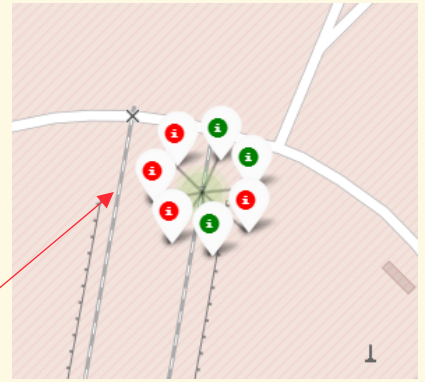


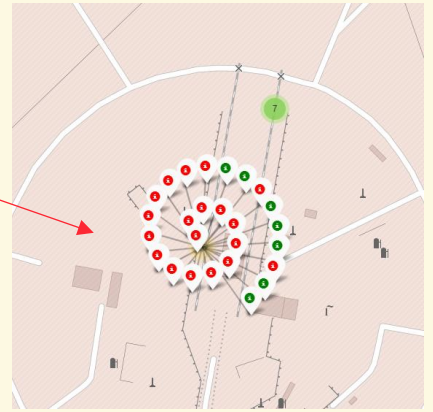
1 Interactive Visual Analysis with Folium

TASK 2: Mark the success/failed launches for each site on the map

- Green icons for successful launches
- Red icons for failed launches

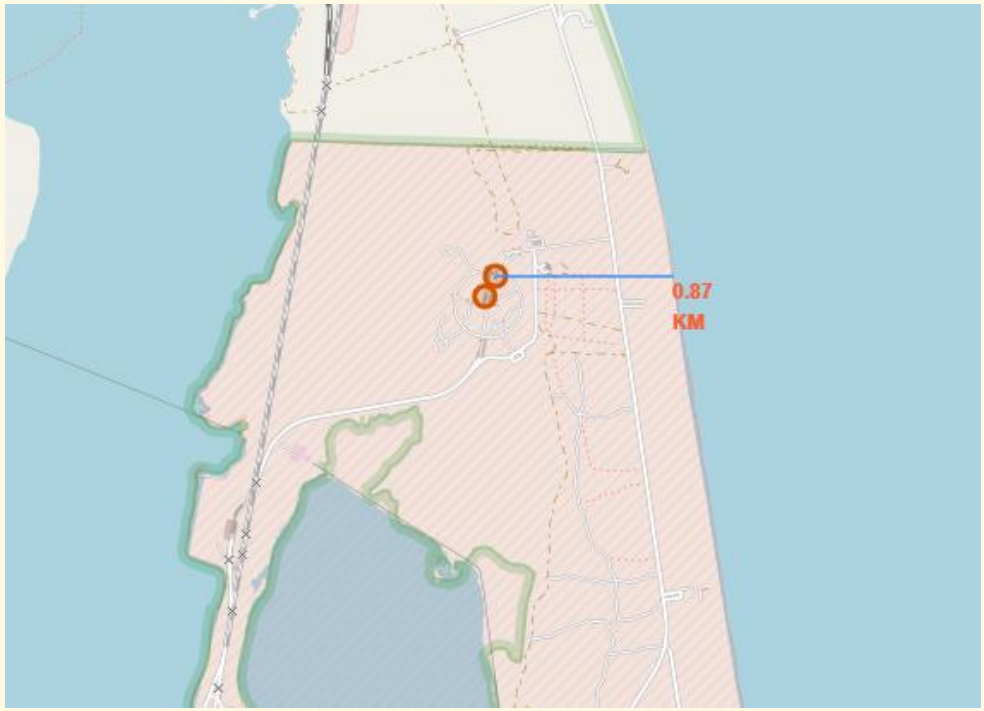




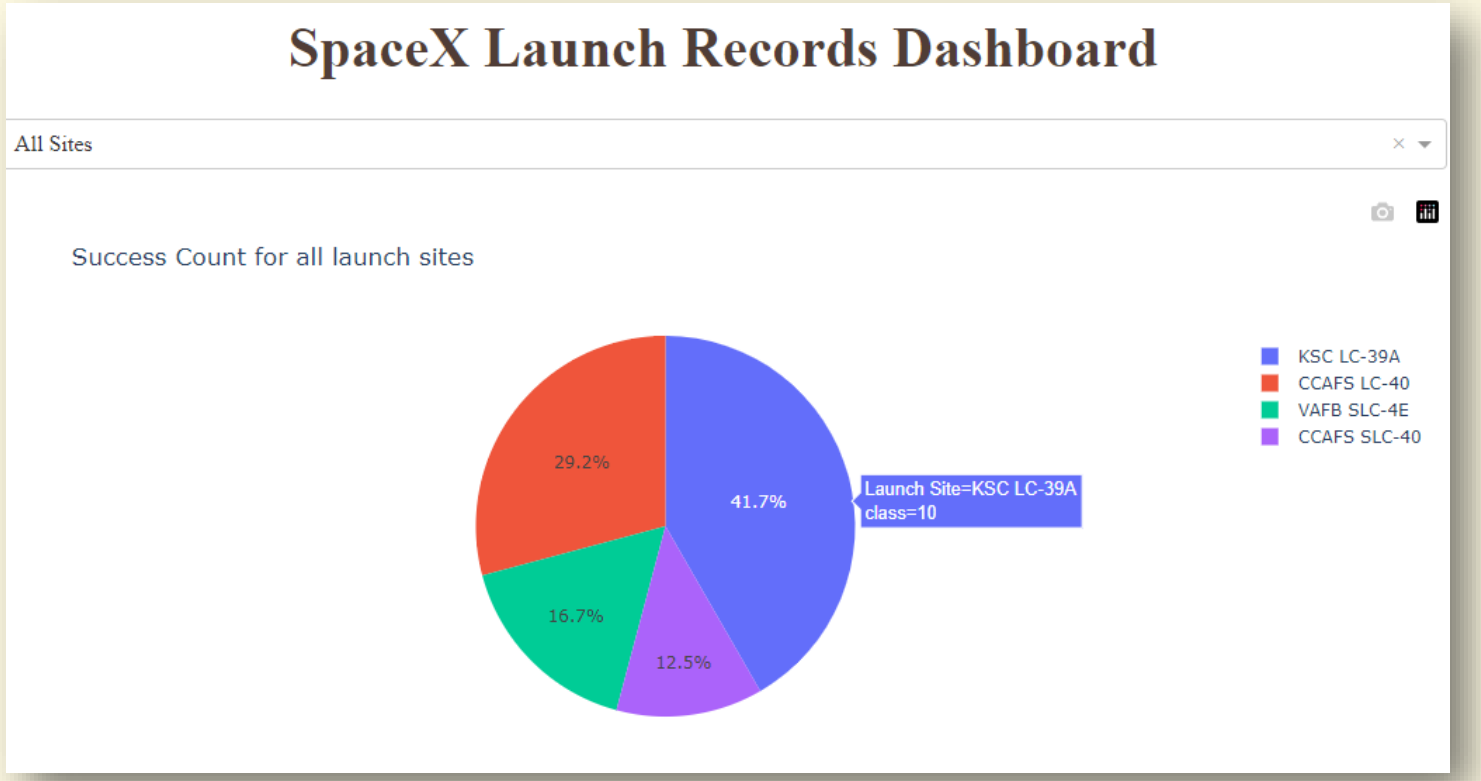


1 Interactive Visual Analysis with Folium

TASK 3: Calculate the distances between a launch site to its proximities

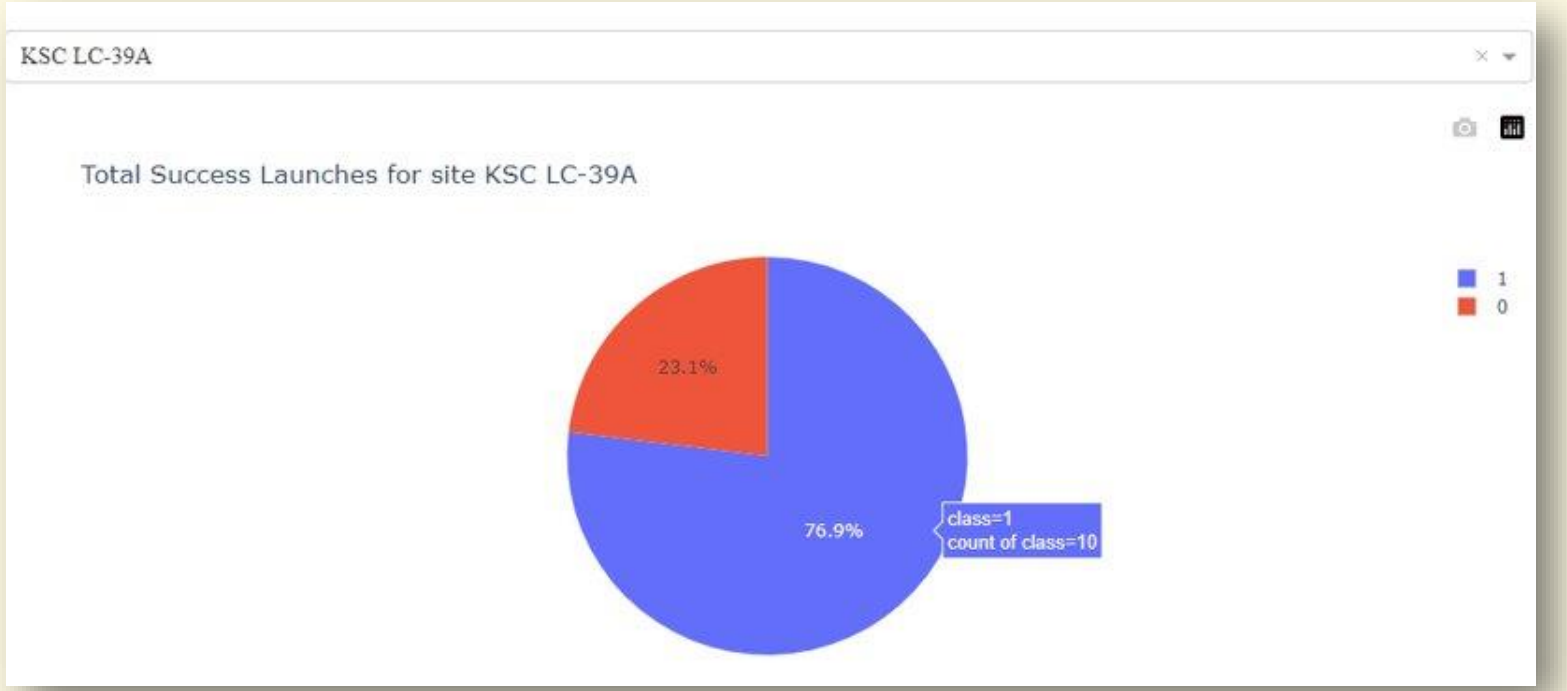


2 Interactive Visual Analysis with Ploty Dash



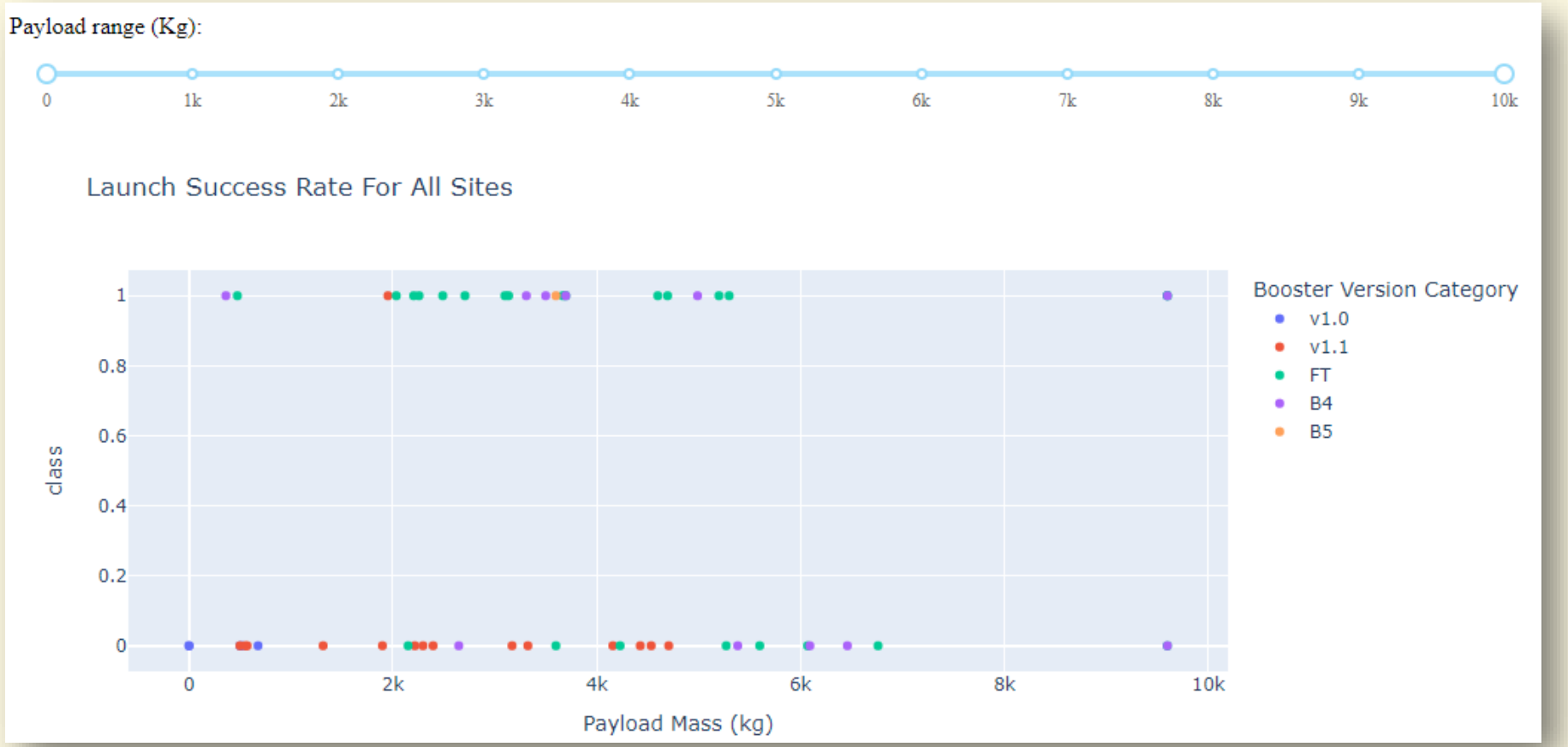
The launch site KSC LC-39A with 41.7% success rate of all launch sites.

2 Interactive Visual Analysis with Ploty Dash



The launch site KSC LC-39A with a 76.9% success rate.

2 Interactive Visual Analysis with Ploty Dash



- **V1.1 has low success rate with low Payload Mass**
- **FT has high success rate with lower than 6k Payload Mass**
- **Generally, low Payload Mass has high success rate**

Predictive Analysis

1 Predictive Analysis

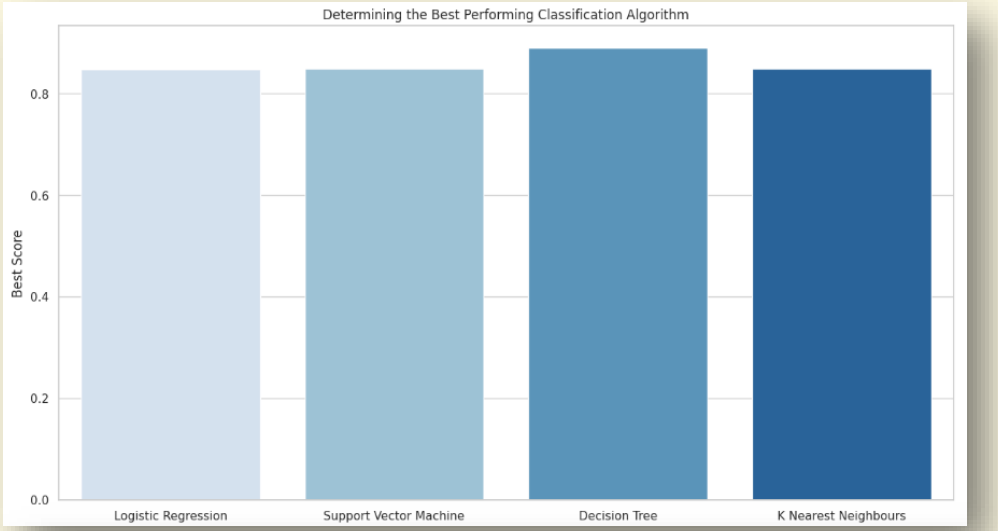
I calculate the Accuracy Score and Best Score for SVM, Classification Trees, and Logistic Regression.

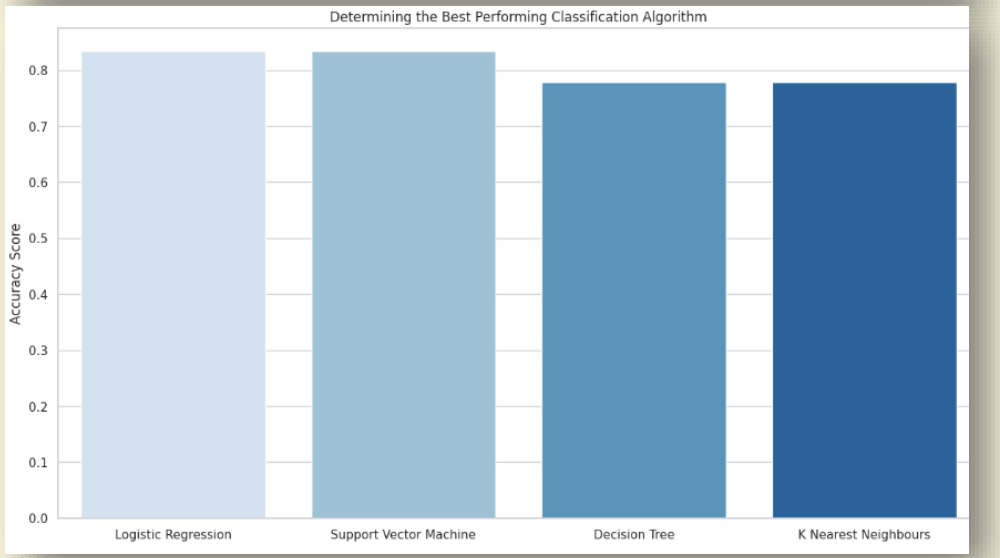
They have similar Accuracy Scores for two group, But the Decision Tree is the higher Best Score.

The Best Score: 88%

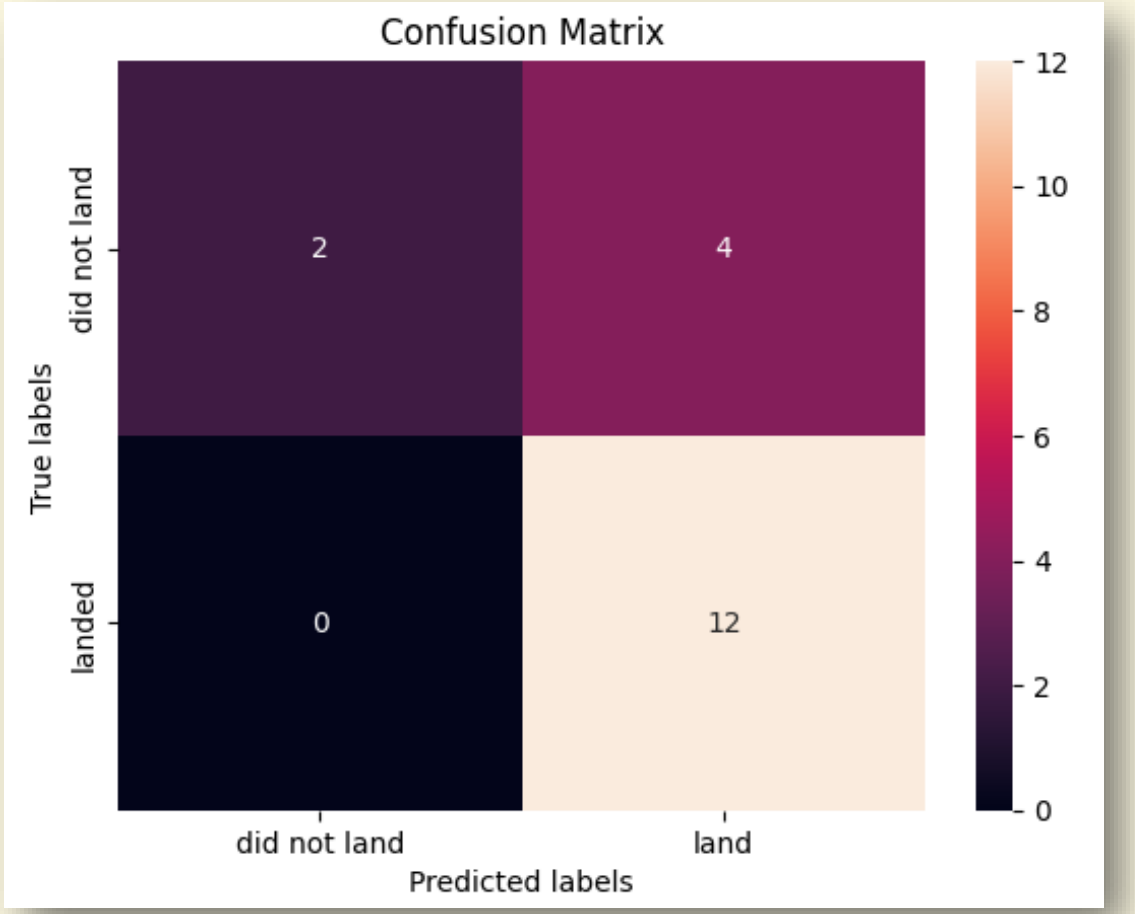
The Accuracy Score: 77%

	Algorithm	Accuracy Score	Best Score
0	Logistic Regression	0.833333	0.846429
1	Support Vector Machine	0.833333	0.848214
2	Decision Tree	0.777778	0.889286
3	K Nearest Neighbours	0.777778	0.848214





1 Predictive Analysis



This is the confusion matrix of the Decision Tree model.

This confusion matrix shows only 4 results are incorrect. 14 results are correct.

Conclusions

1 Conclusions

- **Flight Number VS. Launch Site**
  - Generally speaking, as the number of flights increases, the success rate also increases.
  - CCAFS SLC 40 has a lot of early flights and a high unsuccessful rate.
  - VAFB SLC 4E and KSC LC 39A have a high success rate.

- **Payload VS. Launch Site**
  - Generally, Launch site with over 7000 Payload Mass has a high success rate.
  - For VAFB SLC 4E and KSC LC 39A, if Payload Mass is lower than 5500, it has a high success rate.

- **For Orbit type**
  - LEO, ES-L1, SSO, and HEO, their Payload Mass are lower than 6000 and have a high success rate.
  - ES-L1, SSO, HEO, and GEO have 100% success rate, but they have low Flight numbers.

- The best classification model is Decision Tree with the Best Score: 88%

Thank you

Kuan-Hua Wang

s1003536@gmail.com