IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Simonpaolo Lopez
5/10/2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection through API
  - Data collection with web scraping
  - Data wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis Result
  - Interactive Analytics in screenshots
  - Predictive analytics result

Introduction

- Project background and context
    - SpaceX advertises Falcon 9 rocket launches at a cost of $62 million, significantly lower than the $165 million typically charged by other providers. This reduced cost is largely due to SpaceX’s ability to reuse the first stage of the rocket. Therefore, if we can predict whether the first stage will successfully land, we can estimate the cost of a launch. This insight can be valuable for companies bidding against SpaceX for rocket launch contracts. The aim of this project is to develop a machine learning pipeline that predicts whether the first stage of a rocket will land successfully.
- Problems you want to find answers
    - What factors influence the success of a rocket landing?
    - What operating conditions are required to ensure a successful landing program?
    - How do various features interact to affect the success rate of a landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scaped from Wikipedia
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- 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

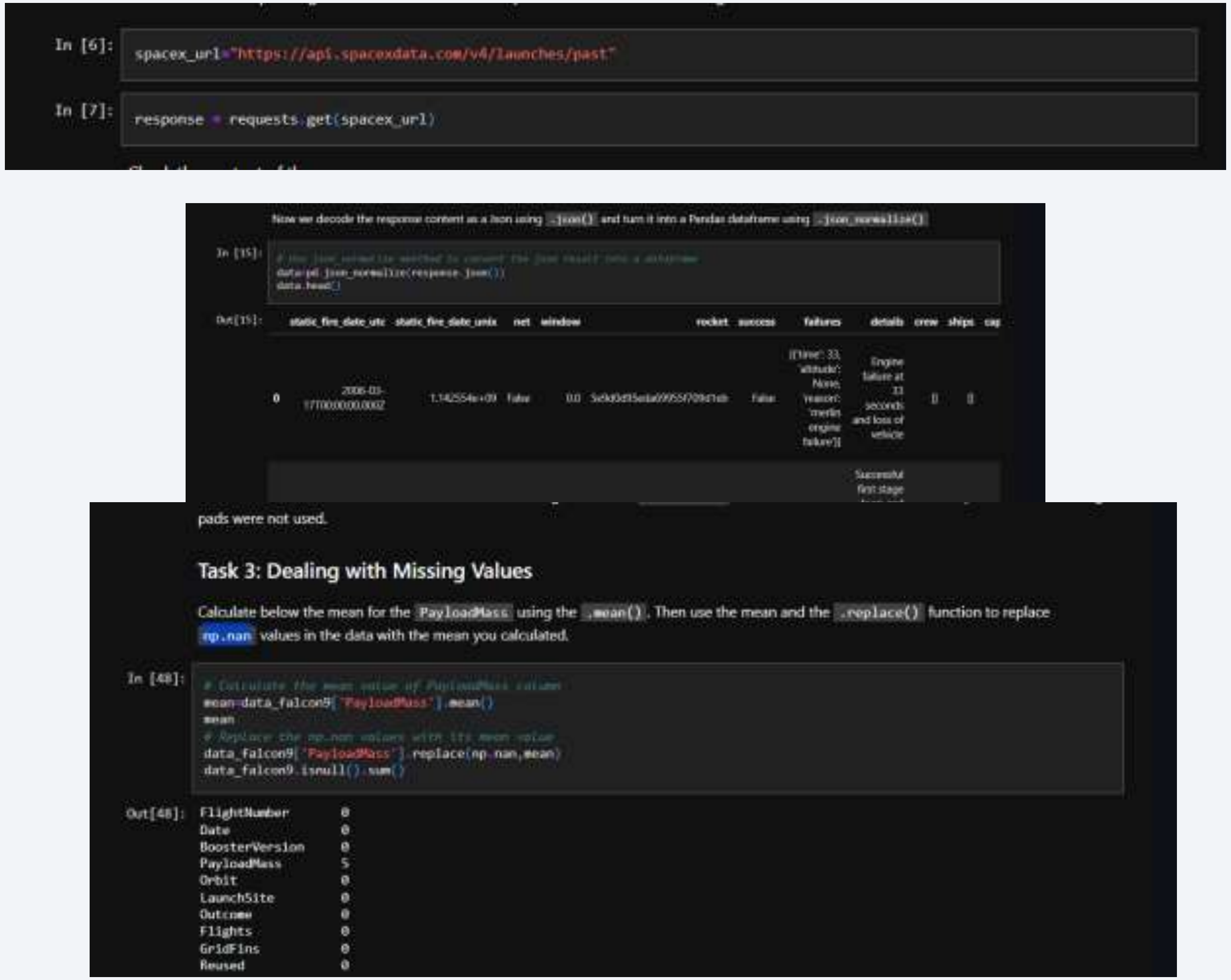
- Describe how data sets were collected.
  - Data was collected using a GET request from the SpaceX API.
  - The API response was decoded into JSON, then converted into a pandas DataFrame using .json_normalize().
  - Web scraping was performed using BeautifulSoup to extract Falcon 9 launch records from Wikipedia.
  - The goal was to parse the HTML table of launch records and convert it into a pandas DataFrame for analysis.
  - Data cleaning involved checking for and filling in missing values.

Data Collection – SpaceX API

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

- The link to the notebook is

https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- I applied web scraping to Falcon 9 launch records with beautifulSoup
- Parsed the table and converted it into pandas dataframe
- The link to the notebook is

https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/jupyter-labs-webscraping.ipynb

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
[11]:
# use requests.get() method with the provided static_url
# assign the response to a object
response=requests.get(static_url)
```

```
[11]: <Response [200]>
```

```
[ ]:
```

Create a `BeautifulSoup` object from the HTML `response`

```
[16]:
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup=BeautifulSoup(response.content)
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
[17]:
# Use soup.title attribute
soup.title
```

```
[17]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

reference link towards the end of this lab

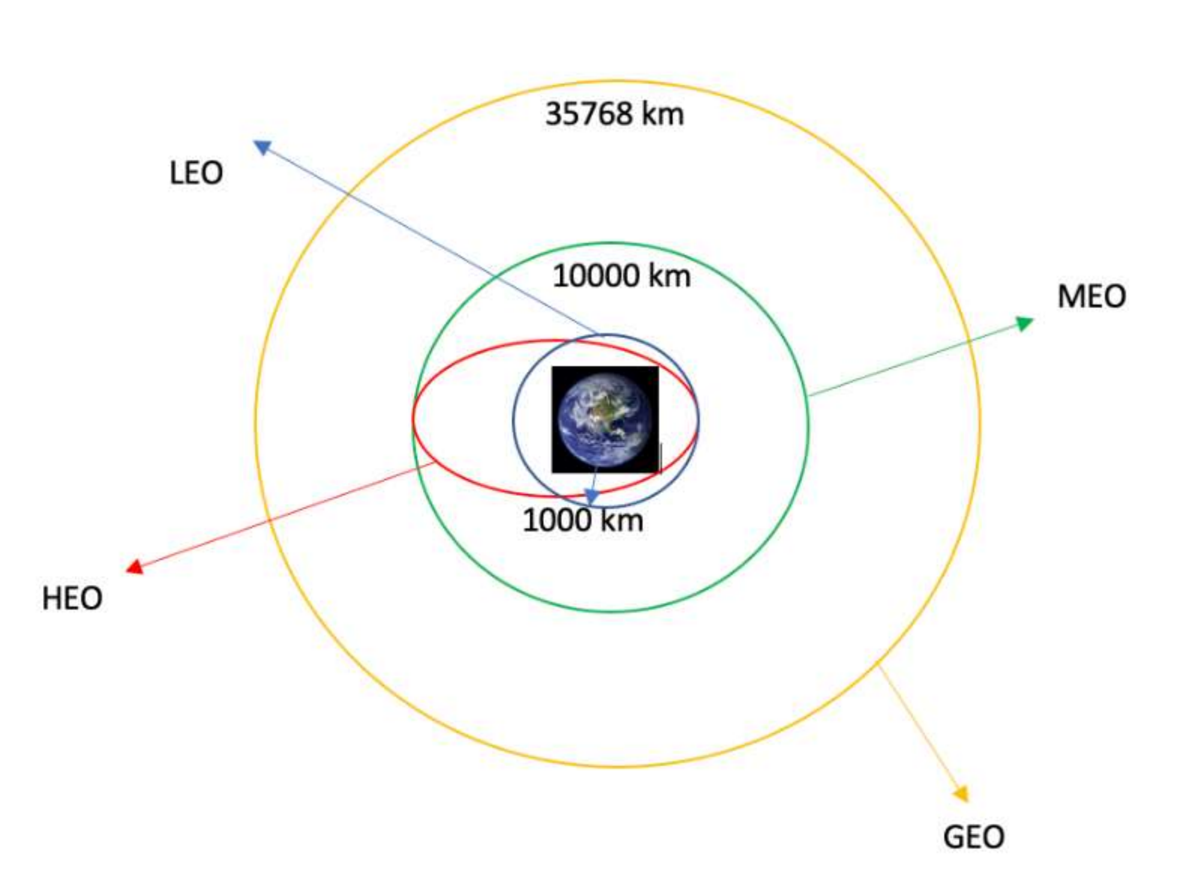
```
In [19]:
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables=soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

Data Wrangling

- Performed exploratory data analysis and determined the training labels.
- Calculated the number of launches at each site, and the number and occurrence of each orbits
- Created landing outcome label from outcome column and exported the results to csv
- The link to the notebook is

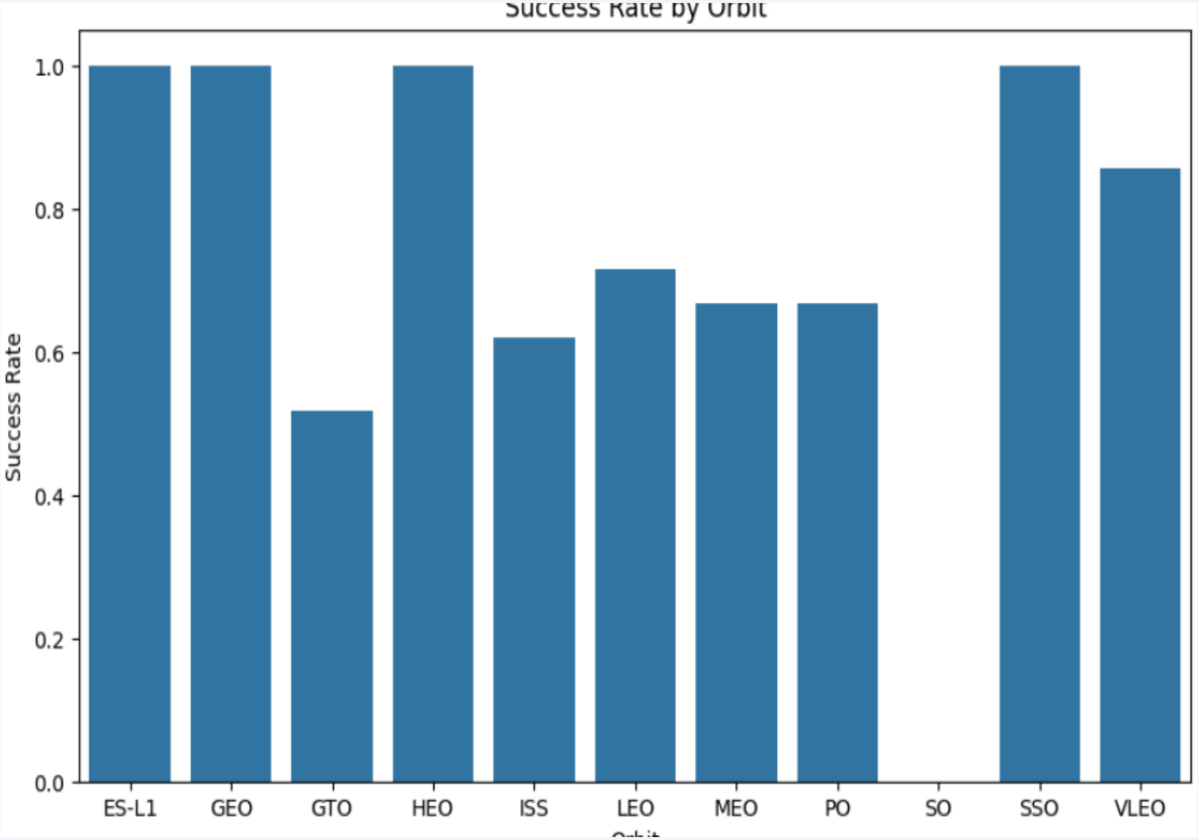
https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/labs-jupyter-spacex-Data%20wrangling-v2.ipynb



EDA with Data Visualization

- Explored the data by visualizing the relationship between flight number and launch site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.
- The link to the notebook is

https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/edadataviz.ipynb



EDA with SQL

- Loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- Applied EDA with SQL to get insight from the data. Wrote queries to find out:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by booster launched by NASA
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names
- The link to the notebook is

https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- **Mapping Launch Sites**: All launch sites were marked on a folium map, and map objects like markers, circles, and lines were added to represent the success or failure of launches at each site.
- **Classifying Launch Outcomes**: Launch outcomes (success or failure) were assigned numeric classes: 0 for failure and 1 for success.
- **Success Rate Analysis**: Color-labeled marker clusters were used to identify launch sites with relatively high success rates.
- **Proximity Calculation**: The distances between launch sites and nearby infrastructures such as railways, highways, and coastlines were calculated. Questions were answered, such as whether launch sites are close to these features or maintain certain distances from cities.

The link to the notebook is

https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/lab_jupyter_launch_site_location_with_folium.ipynb

Build a Dashboard with Plotly Dash

The image outlines additional steps in the project:

1. **Interactive Dashboard**: An interactive dashboard was created using Plotly Dash.
2. **Pie Charts**: Pie charts were plotted to display the total number of launches by specific sites.
3. **Scatter Graph**: A scatter plot was created to show the relationship between launch outcomes and payload mass (in kilograms) for different booster versions.
4. **Notebook Link**: A link to the project's notebook is provided: https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/app.py

Predictive Analysis (Classification)

- **Data Loading and Preparation**: Data was loaded using NumPy and Pandas, then transformed and split into training and testing sets.
- **Model Building and Hyperparameter Tuning**: Different machine learning models were built and hyperparameters were optimized using GridSearchCV.
- **Evaluation Metric**: Accuracy was used as the evaluation metric. Feature engineering and algorithm tuning were applied to improve the model's performance.
- **Best Model Selection**: The best-performing classification model was identified.
- **Notebook Link**: The link to the notebook with the machine learning work is https://github.com/simoneloop/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

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

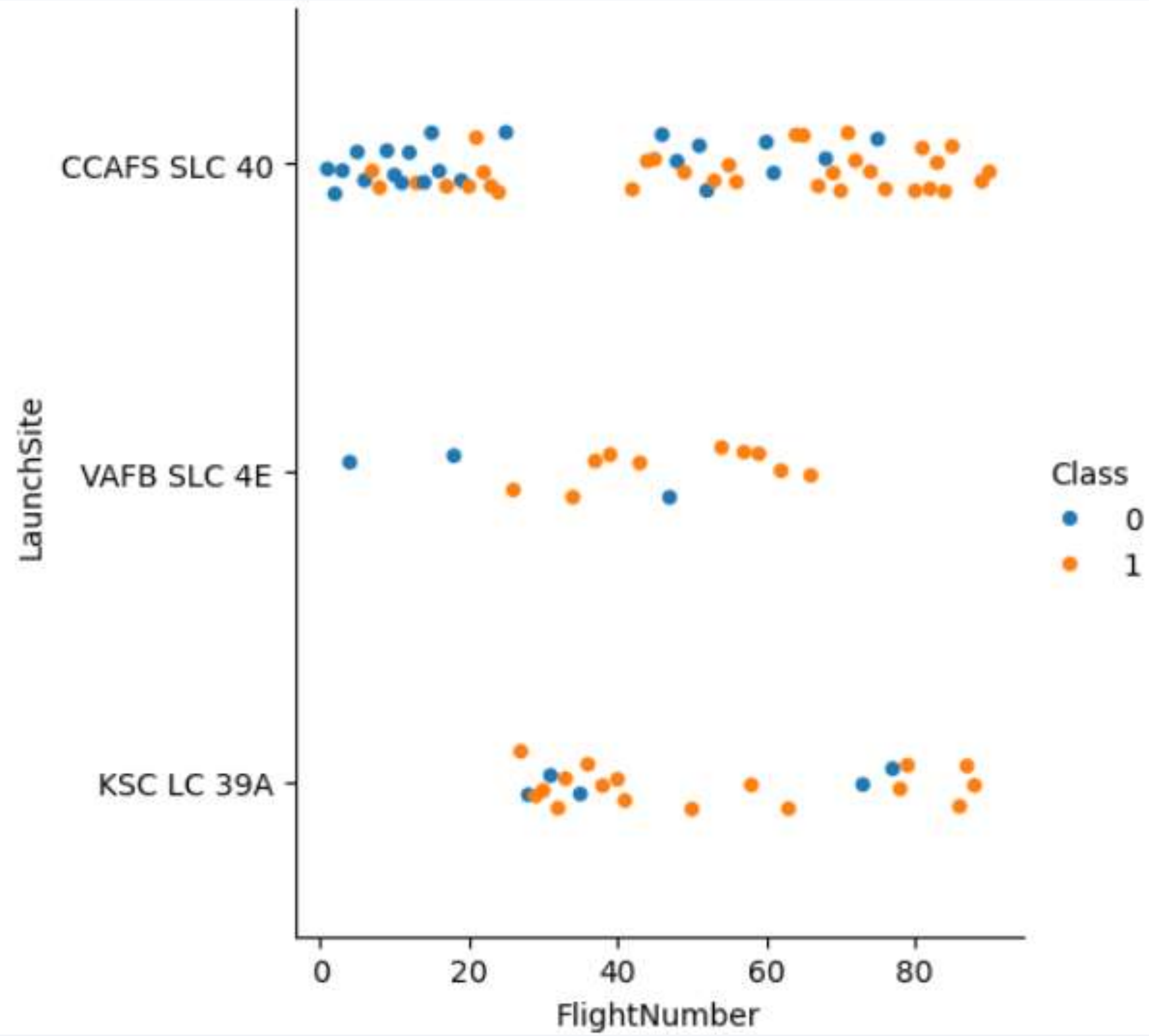
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site

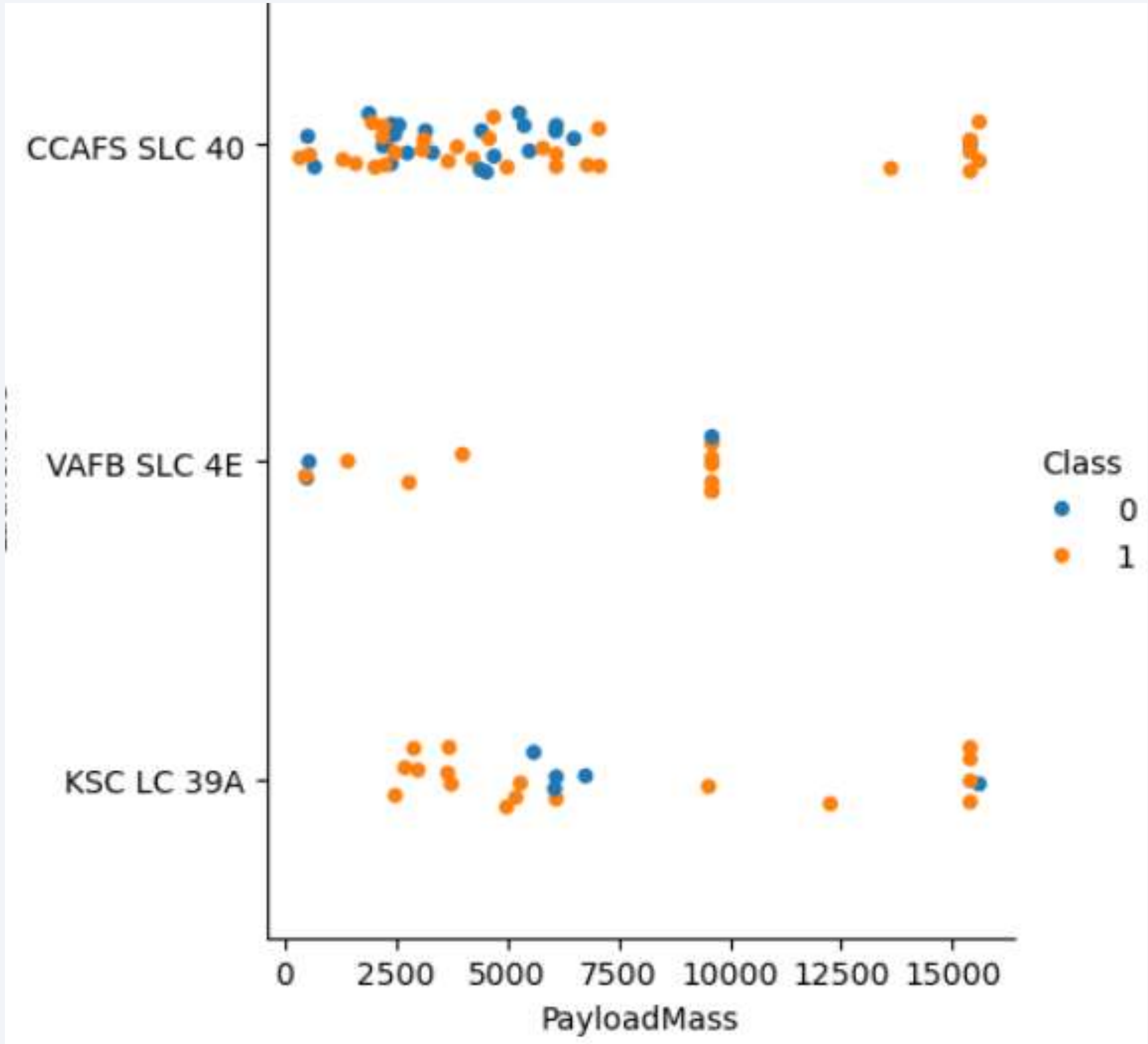
- The larger the flight amount at a launch site, the greater the success rate at a launch site.



Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site

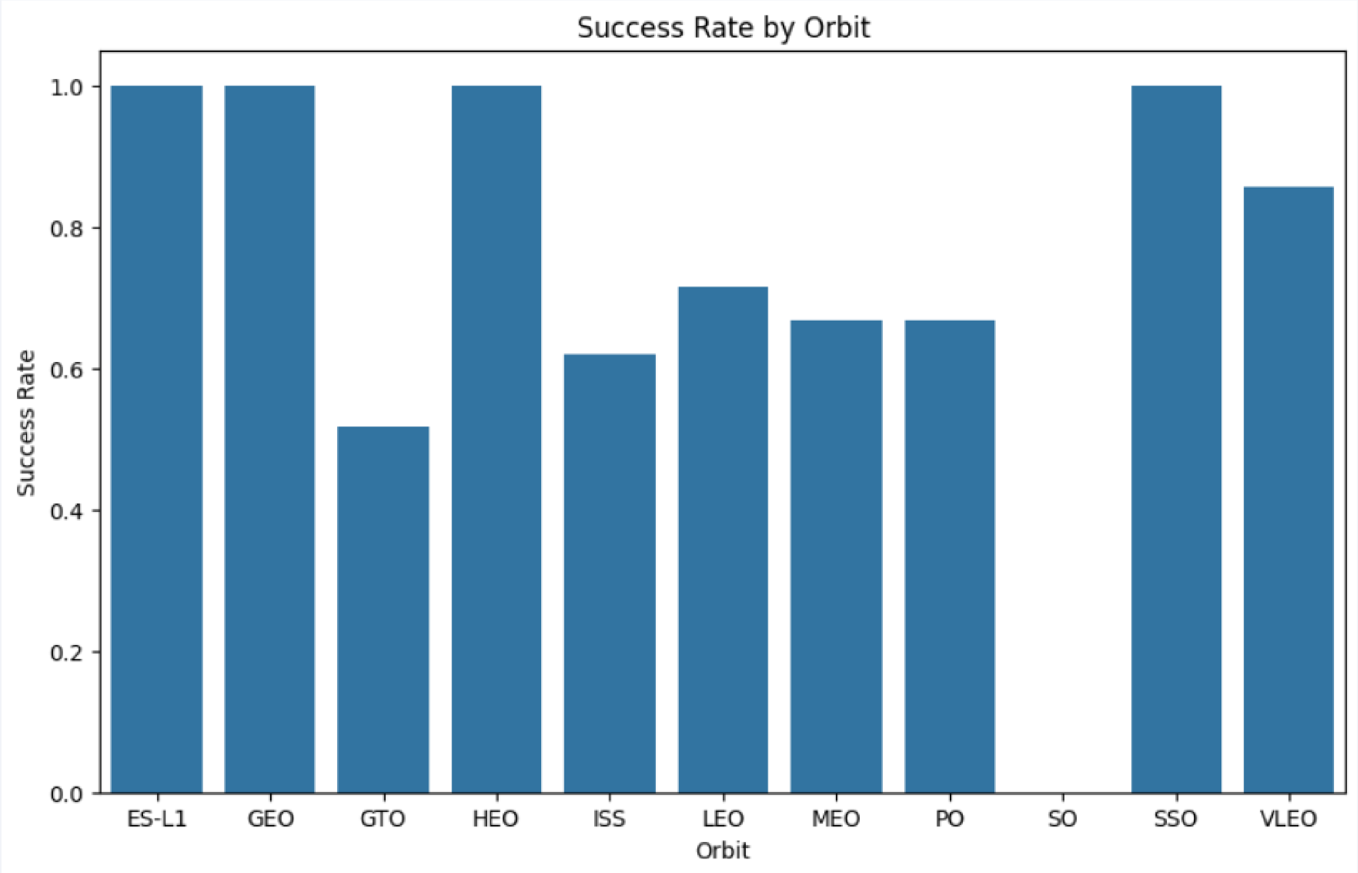
- The greater the payload mass for launchsite CCAFS SLC 40 the higher the success rate for the rocket



Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type

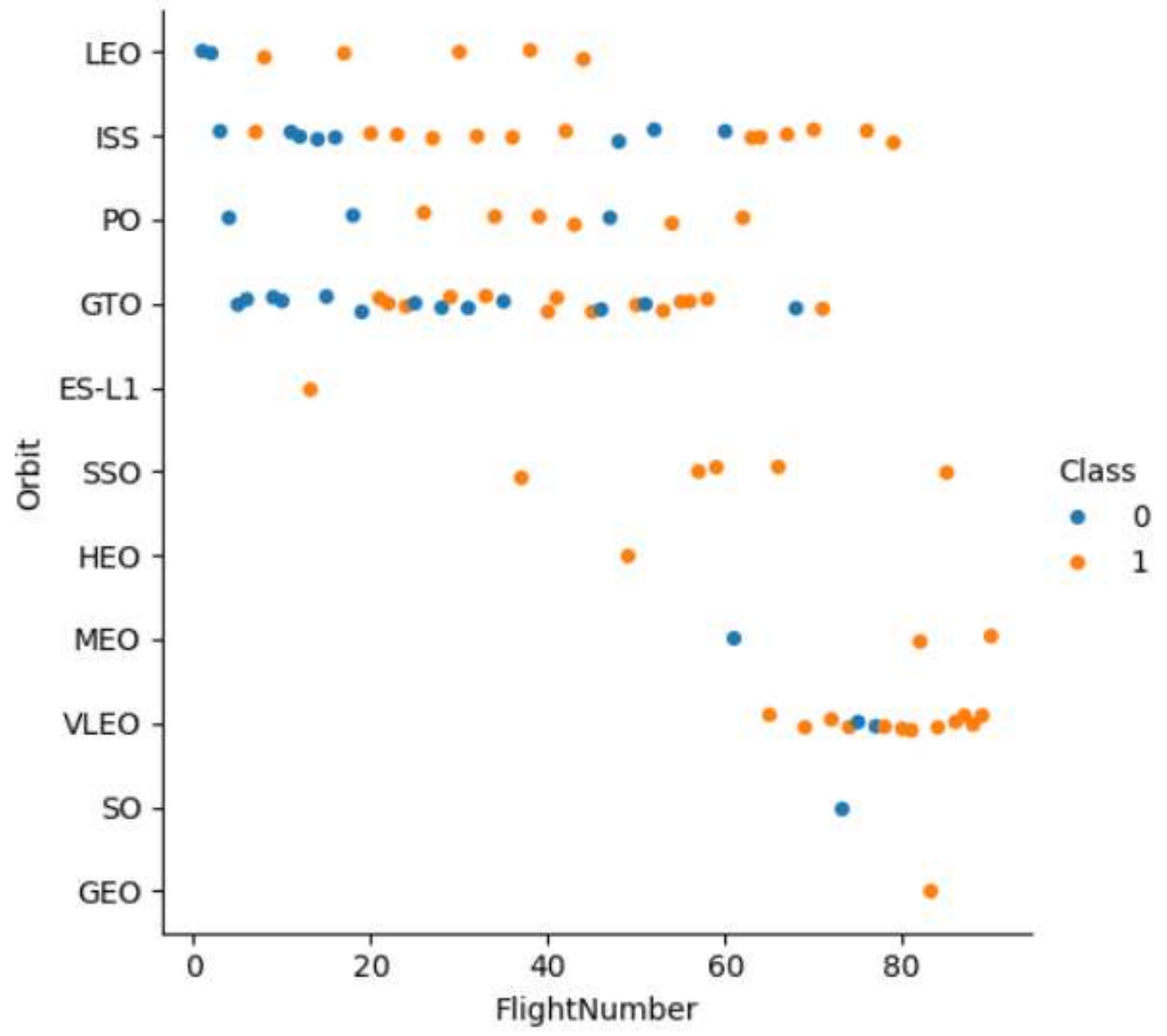
- From the plot we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate



Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type

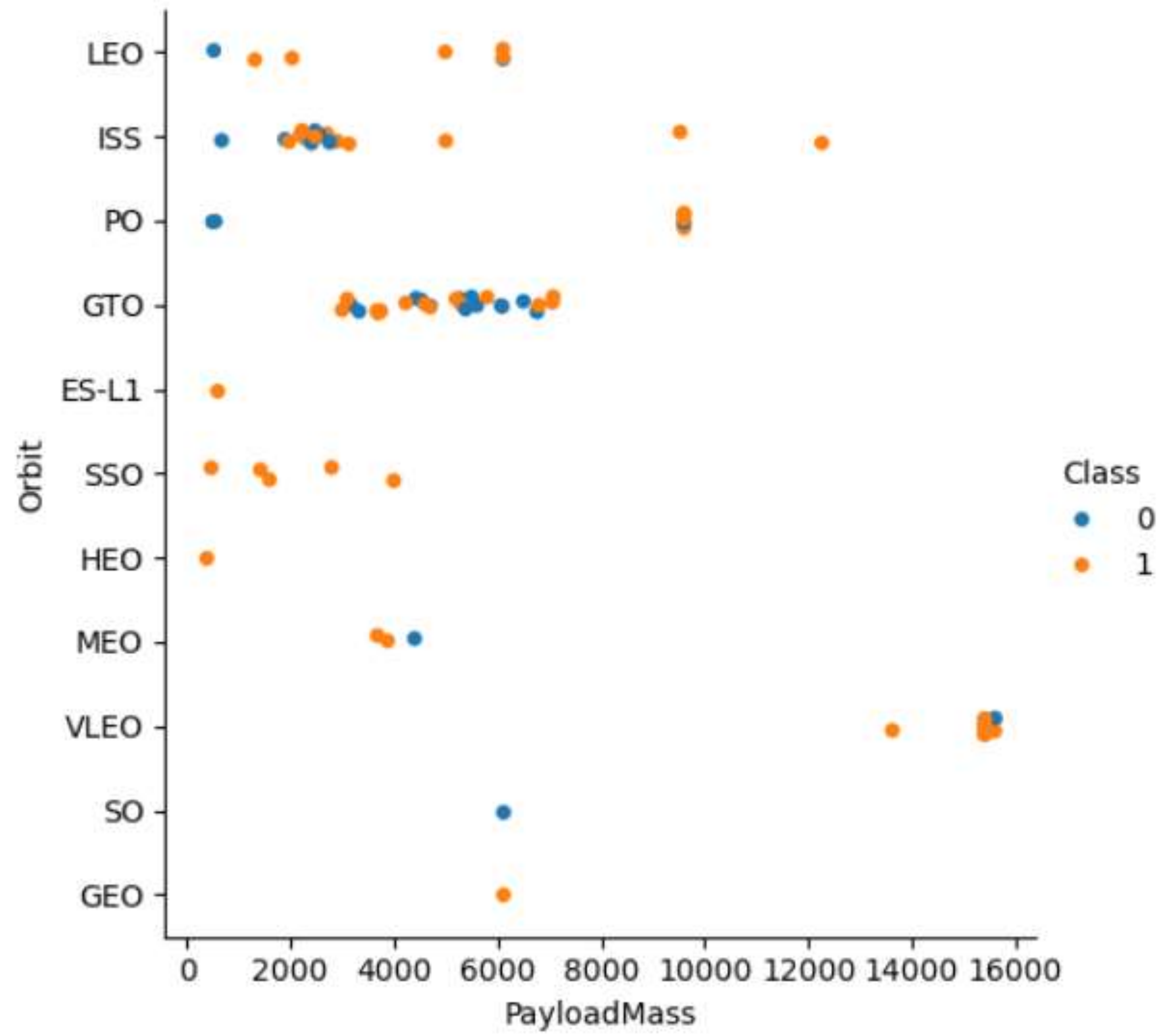
- The plot shows the flight number vs orbit type. We can observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit there is no relationship



Payload vs. Orbit Type

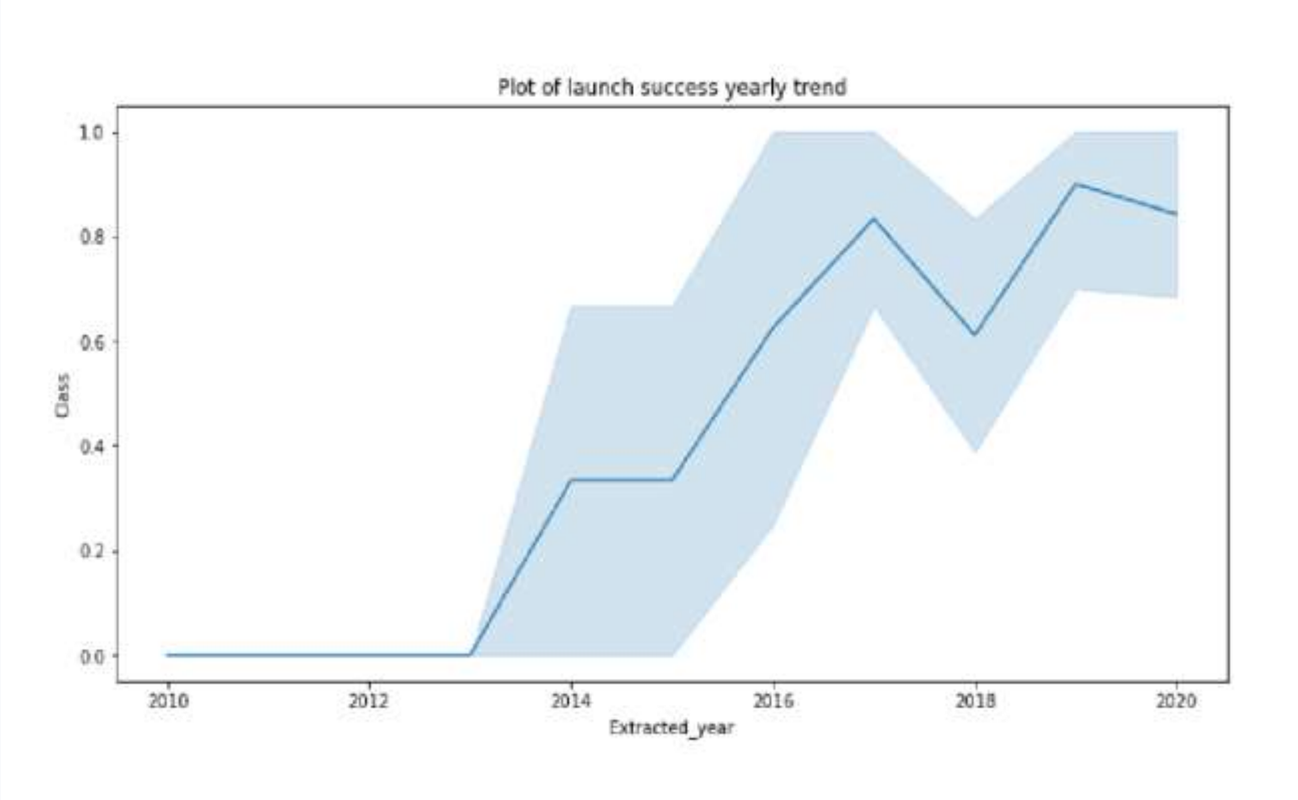
- Show a scatter point of payload vs. orbit type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits



Launch Success Yearly Trend

- We can observe that success rate since 2013 kept on increasing till 2020



All Launch Site Names

- Find the names of the unique launch sites
- I used Distinct to display unique launch site

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

In [10]:

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Used “Like” and “%” to find pattern that start with CCA and limited results to 5

Task 2

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

In [11]:

```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	c
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- I used “SUM” keyword to aggregate and sum the column “PAYLOA

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

In [12]:

```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Like the task before but with average and only where BoosterVersion was “F9 v1.1”

Task 4

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

In [13]:

```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- To Find the first I used min function like the minimum of dates

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

Hint:Use min function

In [14]:

```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- I could use the keywords “between” too

In [15]:

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

Task 7

List the total number of successful and failure mission outcomes

In [16]:

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

The total number of successful mission outcome is:

	successoutcome
0	100

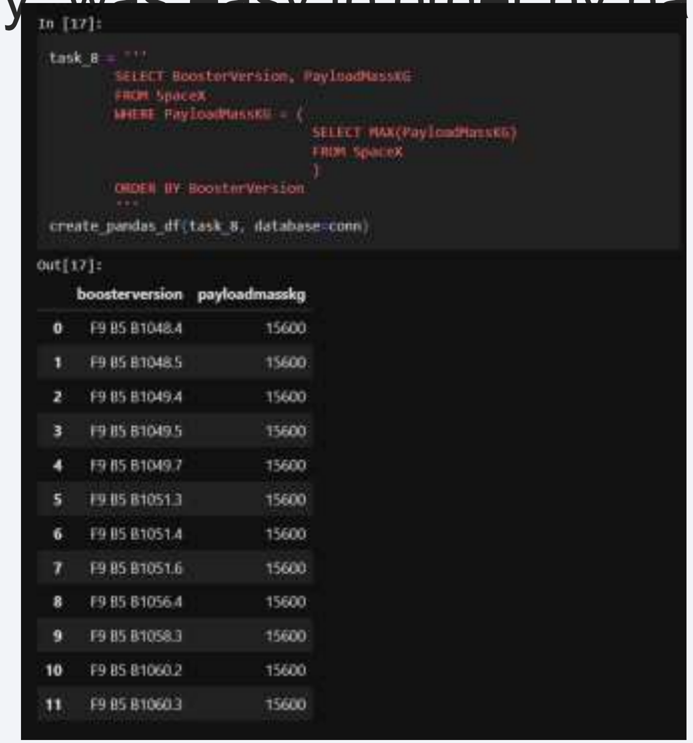
The total number of failed mission outcome is:

Out[16]:

	failureoutcome
0	1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- With the keyword “Order by” was easy to order by payload mass



In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [18]:

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

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

In [19]:

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN2010-06-04 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

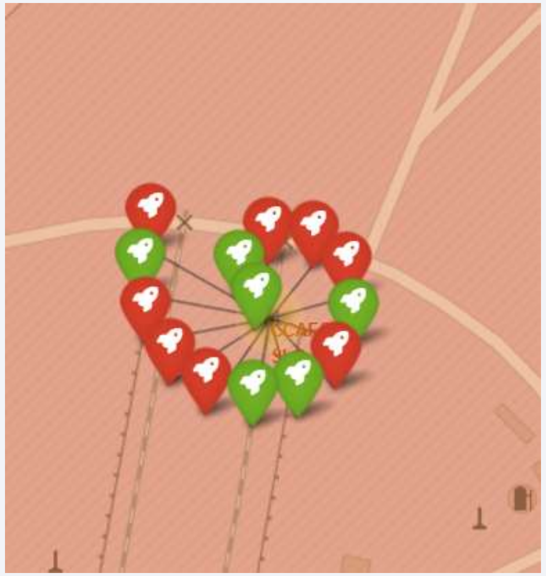
Section 3

Launch Sites Proximities Analysis

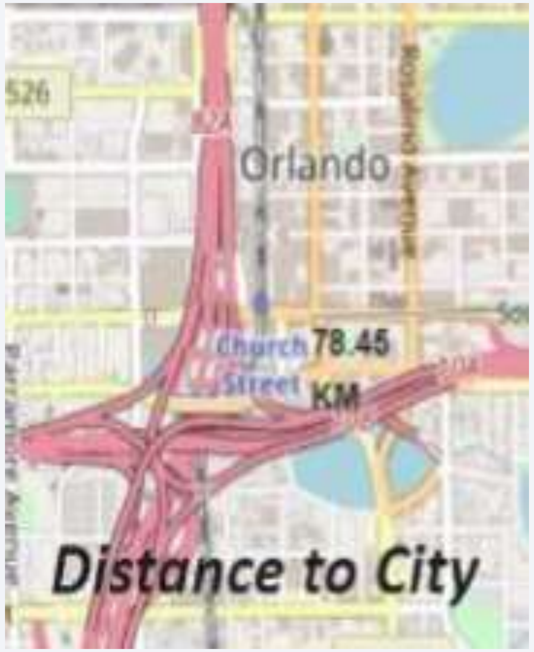
All launch sites global map markers

SpaceX launch site are in the united states of america coasts. Florida and California

Markers showing launch sites with color labels



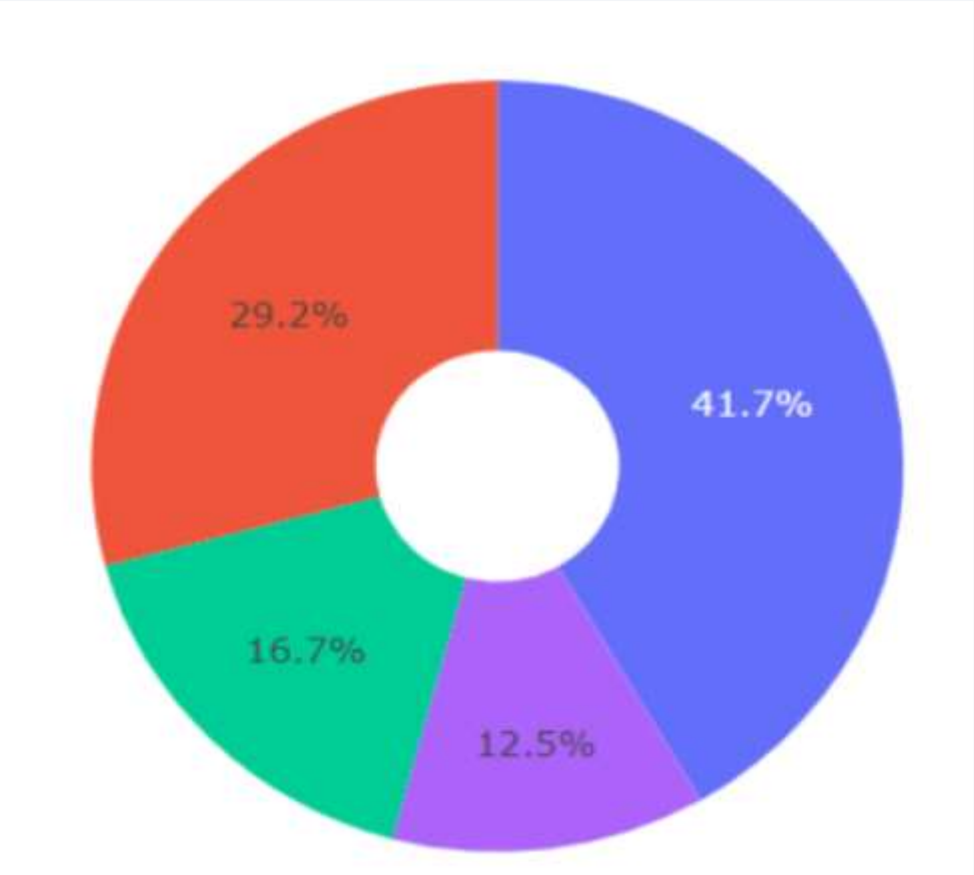
Launch Site distance to landmarks

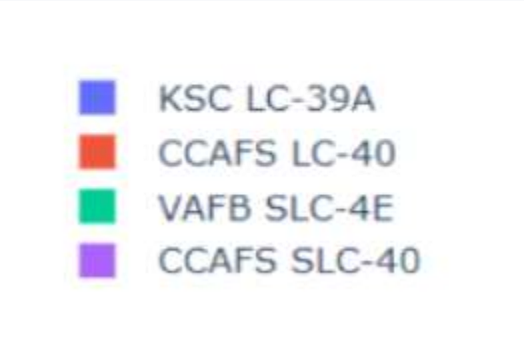


Section 4

Build a Dashboard with Plotly Dash

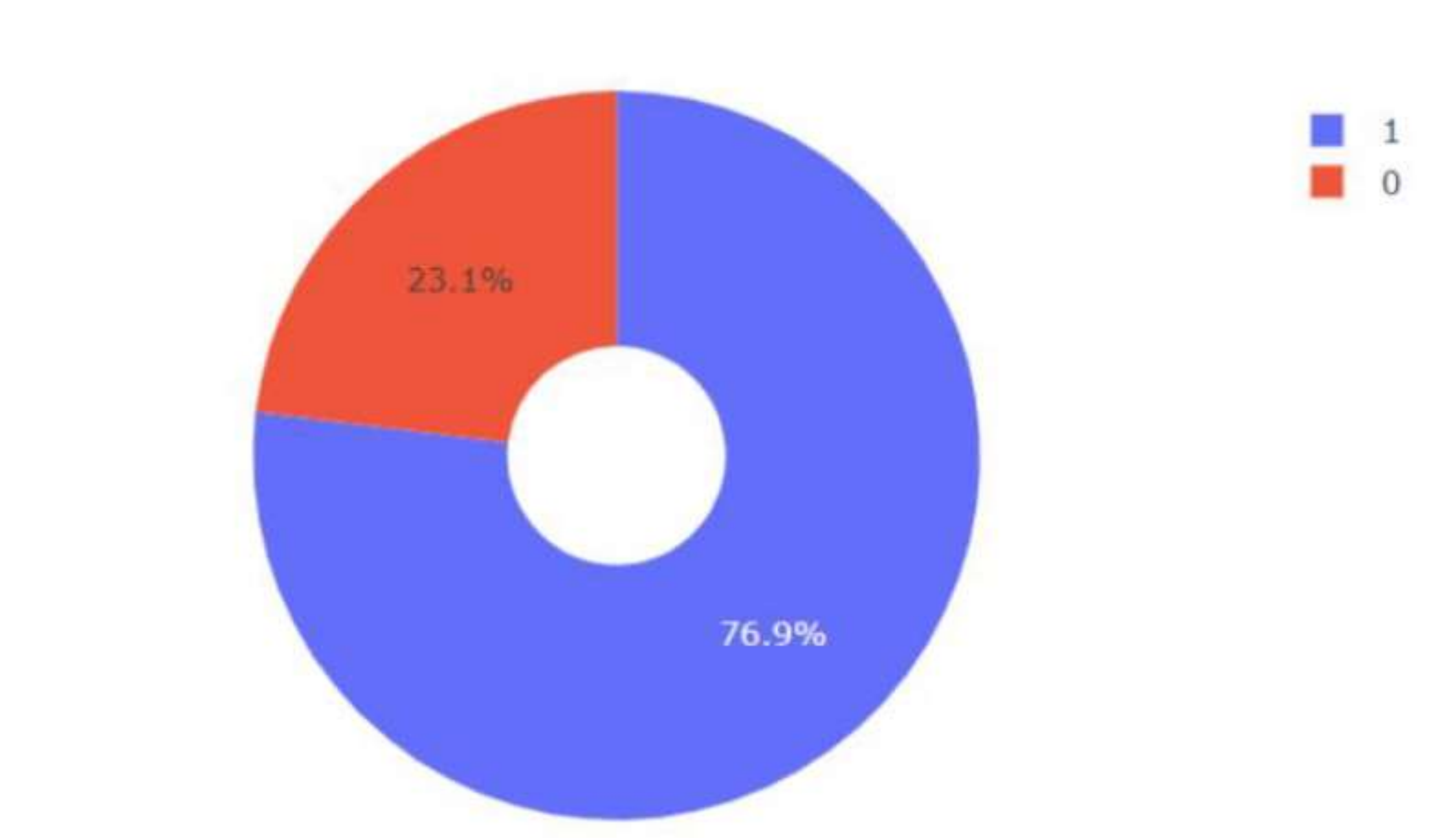
Pie chart showing the success percentage achieved by each launch site





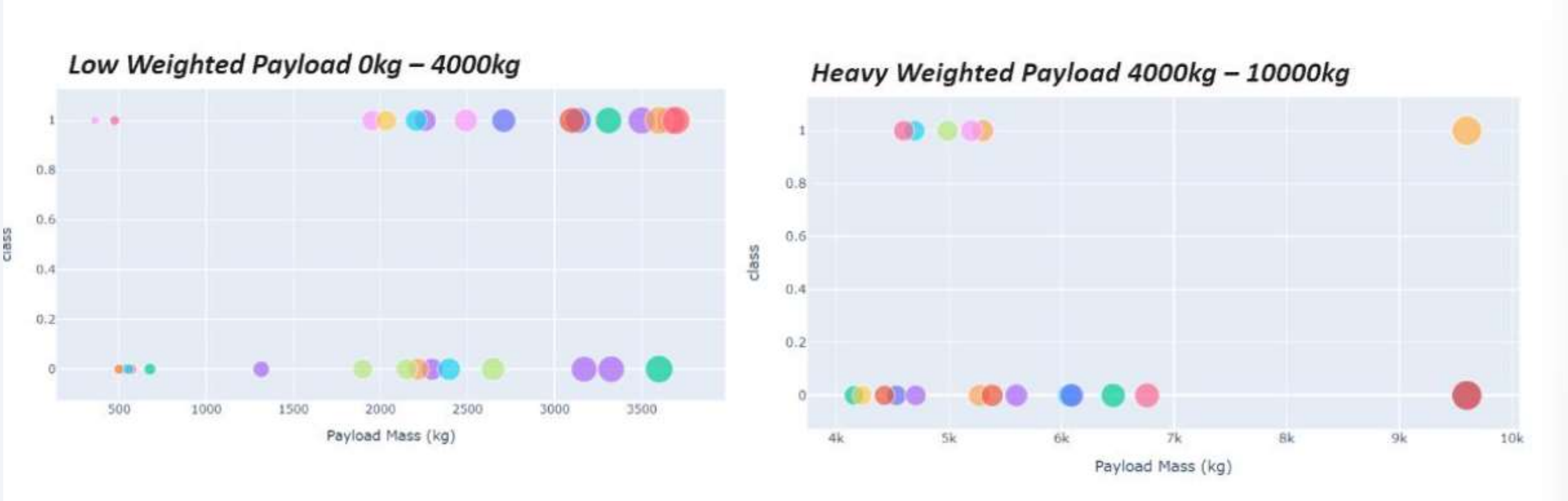
KSC LC-39° had the most successful launches from all the sites

Pie chart showing the Launch site with the highest launch success ratio



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Scatter plot of Payload vs Launch Outcome for all sites, with differentpayload selected in the range slider41



The success rate for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Find the method performs best:

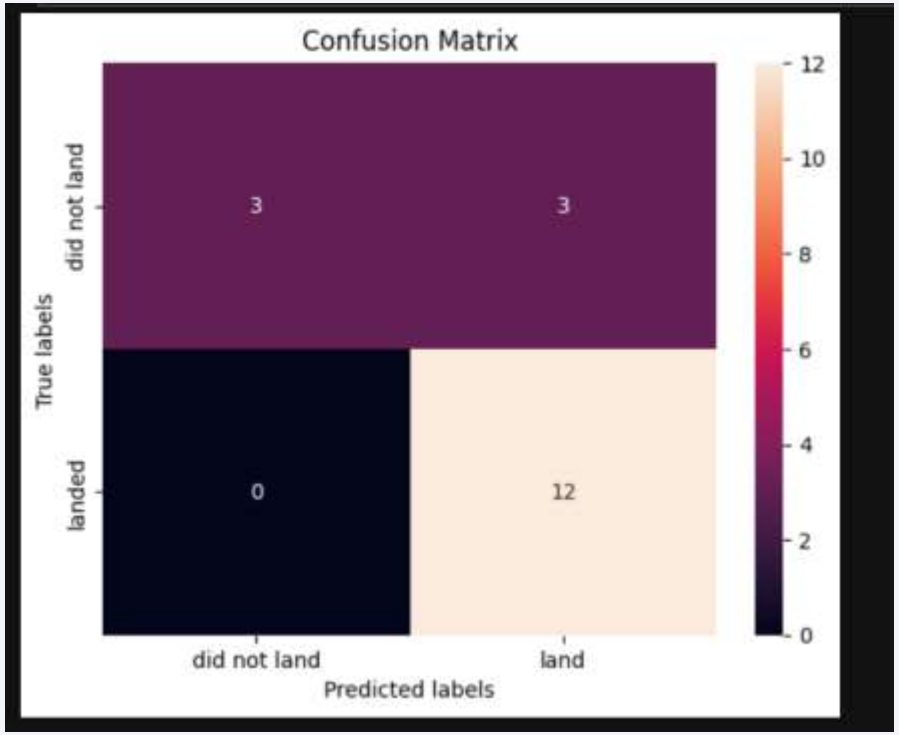
```
models = {'KNeighbors': knn_cv.best_score_,
          'DecisionTree': tree_cv.best_score_,
          'LogisticRegression': logreg_cv.best_score_,
          'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm, 'with a score of', models[bestalgorithm])

if bestalgorithm == 'DecisionTree':
    print('Best params is:', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is:', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is:', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is:', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix



The confusion matrix for the decision treeclassifier shows that the classifier candistinguish between the different classes.The major problem is the false positives .i.e.,unsuccessful landing marked as successfullanding by the classifier.

Conclusions

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launchsite.
- Launch success rate started to increase in 2013 till 2020.
- 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.Conclusions

Thank you!