IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Manuel G.M.
18/04/2024

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context

SpaceX is a company with very ambitious objectives. one of them is to offer rocket launches below the cost of 62 million, thus revolutionizing the space industry. Well below what other companies offer.

The cost difference is due to the idea of reusing the first stage of the launch by re-land the rocket to be used on the next mission.

This project helps identify the correct price to compete against SpaceX in rocket launches.

- Problems we want to find answers

Factors determine the success of rocket launches.

Success rate of successful landings with the different variables.

Best conditions needed to increase the success rate of landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web-scrapping 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.

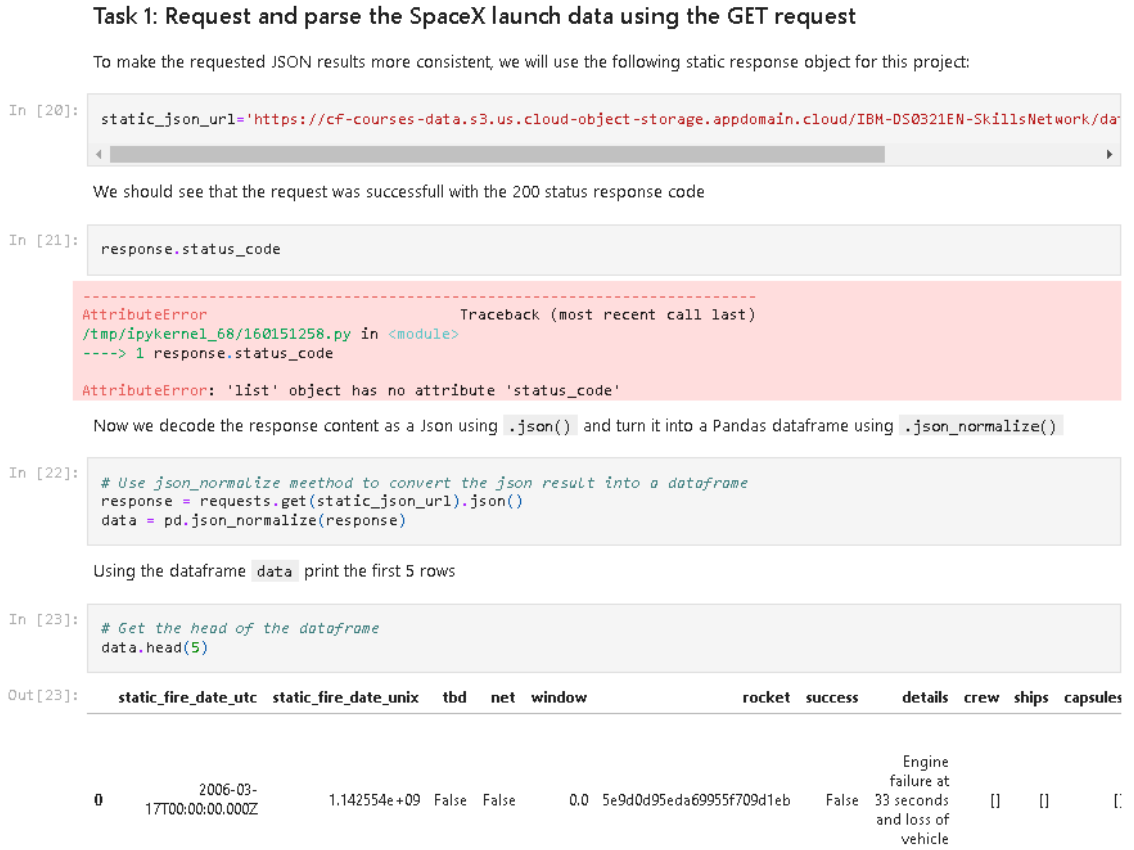
Data collection was done using get request to the API of SpaceX. The we using .json() and turn it into a dataframe with pandas with .json_normalize().

Then we cleaned data, like missing values replacing them.

Also, we perfomed web-scrapping from Wikipedia with BeautifulSoup. Extracting launch records of Falcon 9, transforming a pandas dataframe for analysis.

Data Collection – SpaceX API

- We used the get request to collect data, convert to json and then convert into a dataframe.
- The link to the notebook is https://github.com/manuelgm92/SpaceY/blob/main/1.%20jupyter-labs-spacex-data-collection-api.ipynb



Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

In [20]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/da
```

We should see that the request was successfull with the 200 status response code

In [21]:
```
response.status_code
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
/tmp/ipykernel_68/160151258.py in <module>
----> 1 response.status_code

AttributeError: 'list' object has no attribute 'status_code'
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

In [22]:
```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

Using the dataframe `data` print the first 5 rows

In [23]:
```
# Get the head of the dataframe
data.head(5)
```

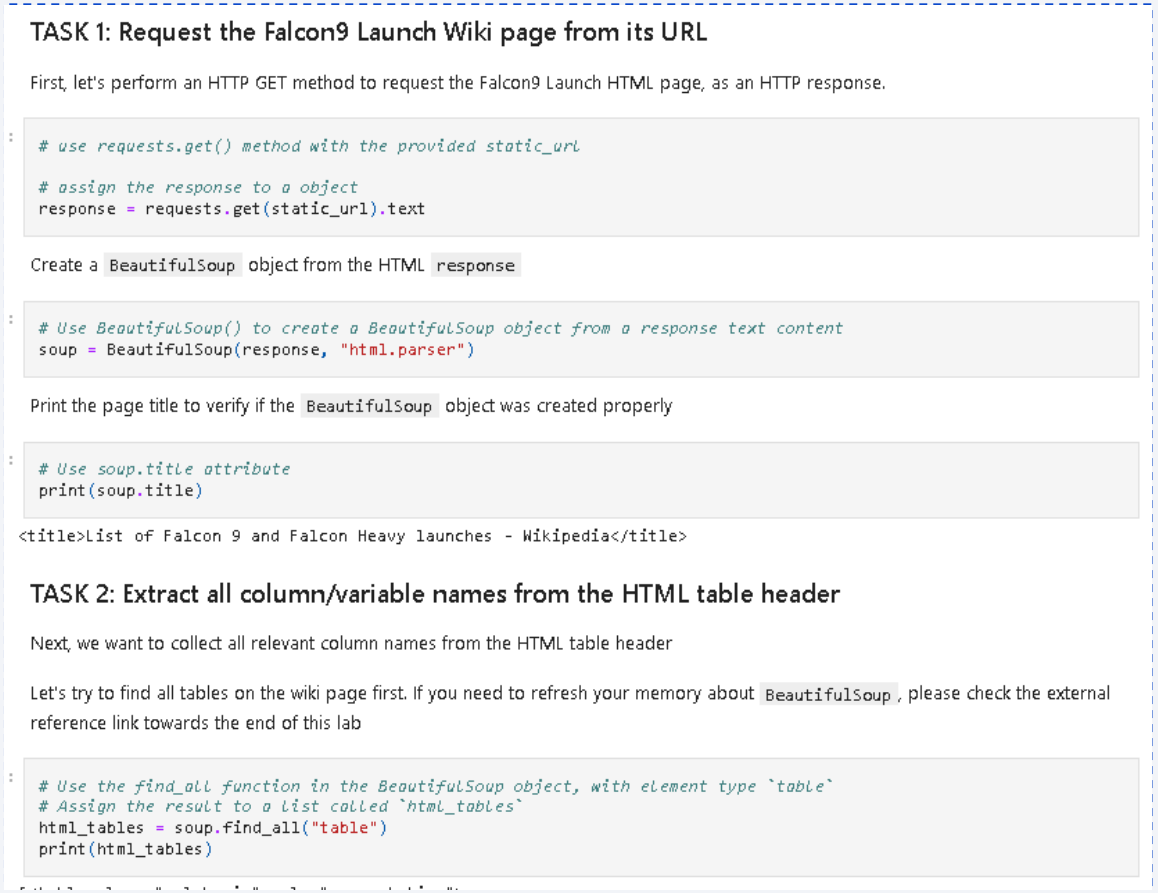
Out[23]:

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	crew	ships	capsules
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Engine failure at 33 seconds and loss of vehicle	[]	[]	[

Data Collection – Scrapping

- We applied web-scraping converting a pandas dataframe.

- The GitHub URL of the completed SpaceX API calls notebook https://github.com/manuelgm92/SpaceY/blob/main/2.%20jupyter-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.

```
# use requests.get() method with the provided static_url

# assign the response to a object
response = requests.get(static_url).text
```

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

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

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

```
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

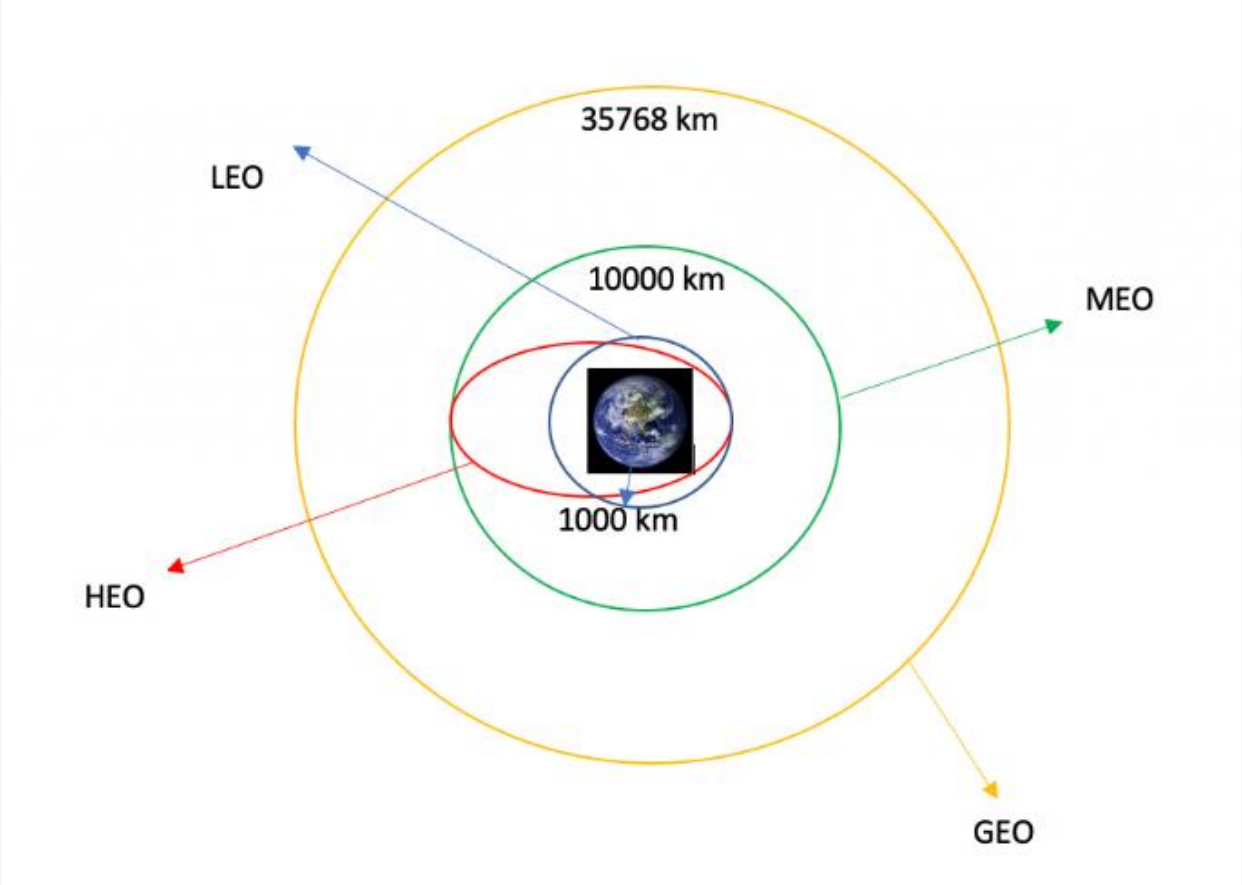
Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```
# 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")
print(html_tables)
```

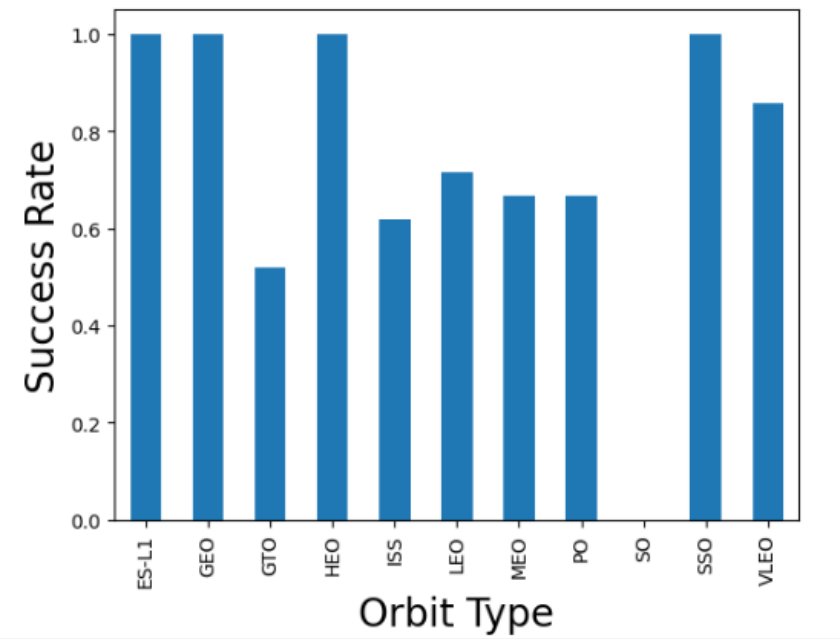
Data Collection - Scraping

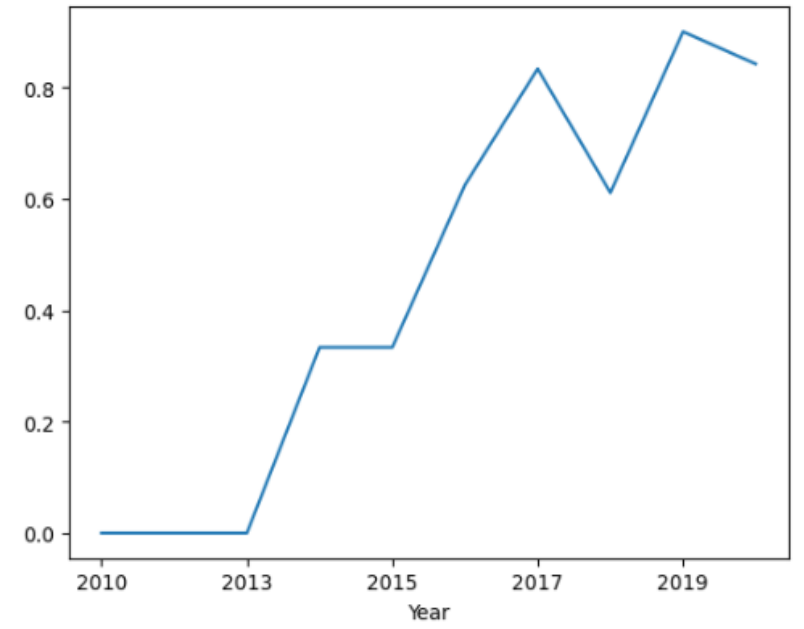
- Calculate the number of launches on each site, the number of each orbit and mission outcome of the orbits.
- The GitHub URL of the notebook, https://github.com/manuelgm92/SpaceY/blob/main/3.%20labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- We explored the data by visualizing the success rate of each orbit and the launch success yearly trend.
- The link to the notebook: https://github.com/manuelgm92/SpaceY/blob/main/5.%20edadataviz.ipynb



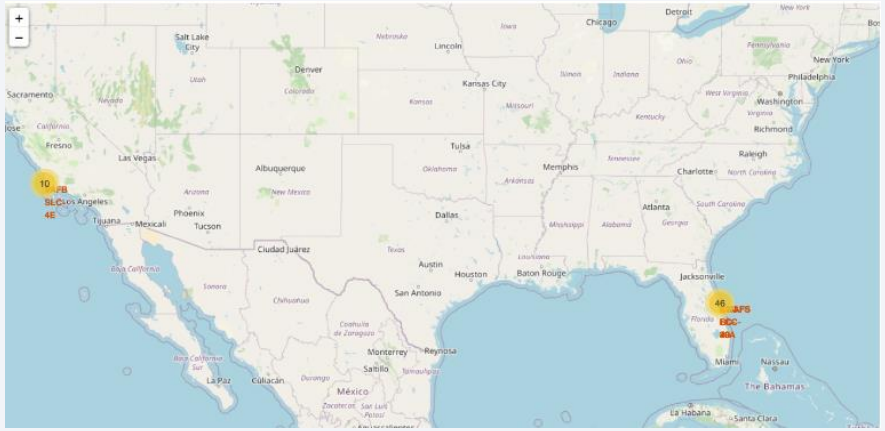


EDA with SQL

- We loaded dataset into a PostgreSQL database. We applied EDA using SQL queries to answer questions about launches rocket, landigns, …
- The link to the notebook is https://github.com/manuelgm92/SpaceY/blob/main/4.%20jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium and a Dashboard with Plotly Dash

- We personalize some things about launch sites on a map with Folium
- https://github.com/manuelgm92/SpaceY/blob/main/6.%20lab_jupyter_launch_site_location.ipynb



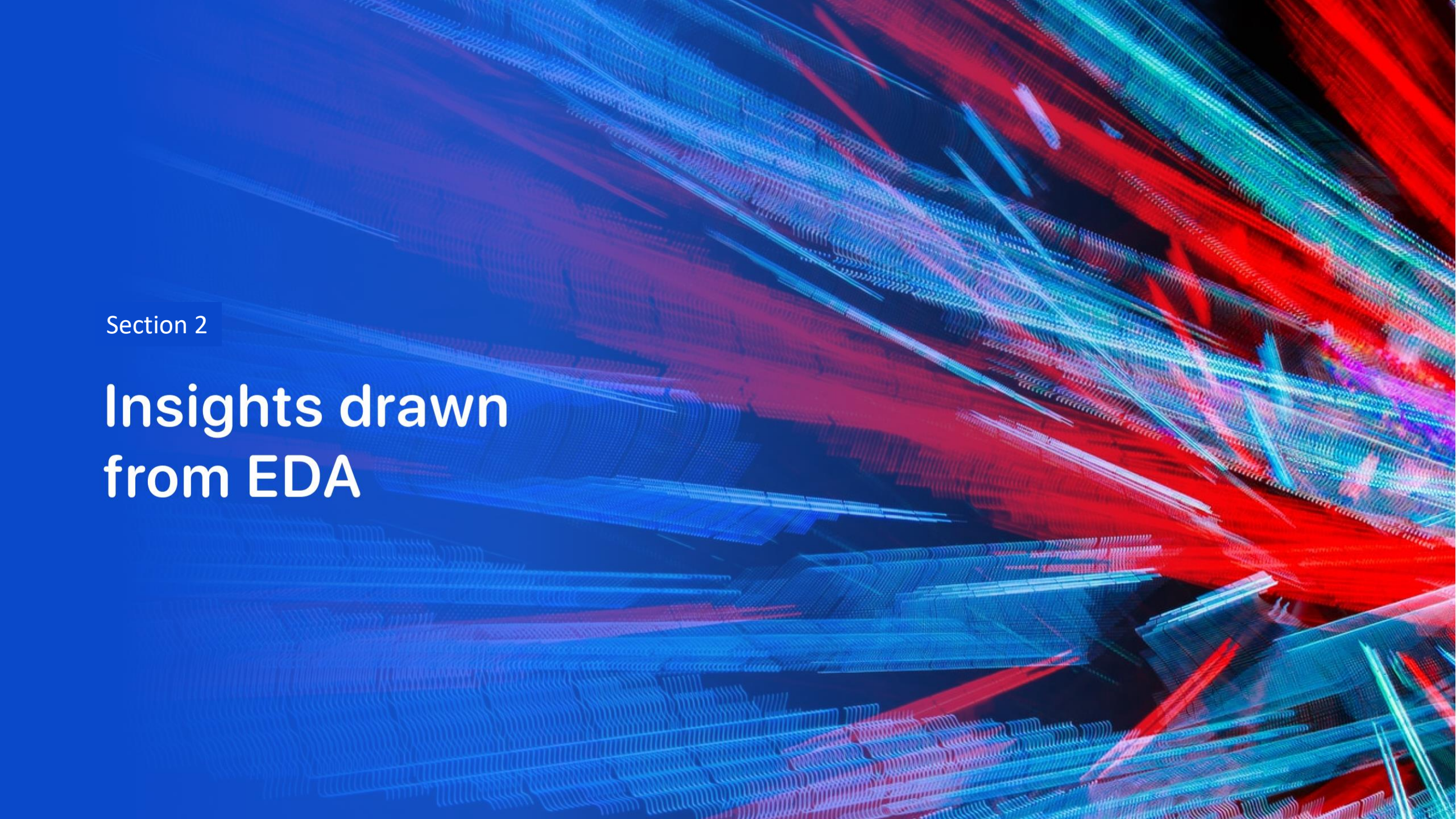
- We built an interactive Dashboard with Ploty Dash.
- https://github.com/manuelgm92/SpaceY/blob/main/7.%20spacex_dash_app.py

Predictive Analysis (Classification)

- We have used libraries of python like numpy and pandas to loaded, transform, split the data to training and test it.
- Bulding of different machine learning models.
- We improve the model using techniques of machine learning, founded the best classification model.
- The link to the notebook is https://github.com/manuelgm92/SpaceY/blob/main/8.%20SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

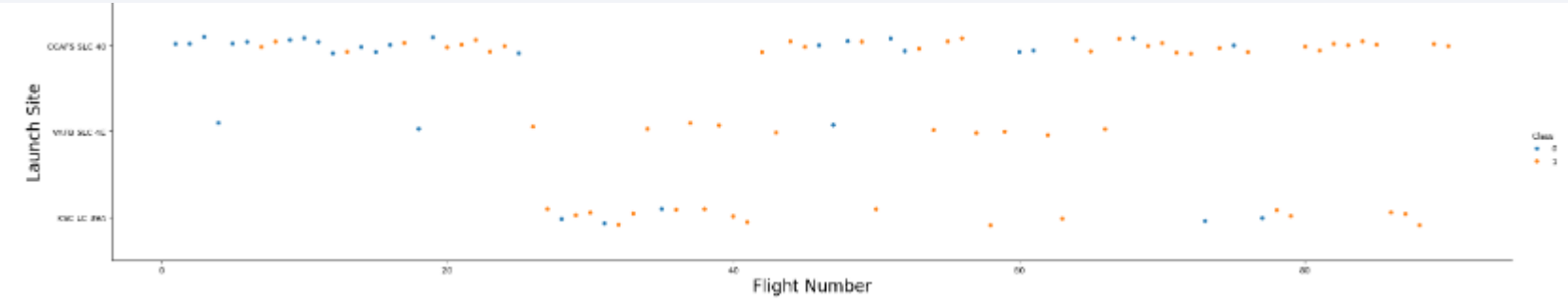
Results

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


Section 2
Insights drawn from EDA

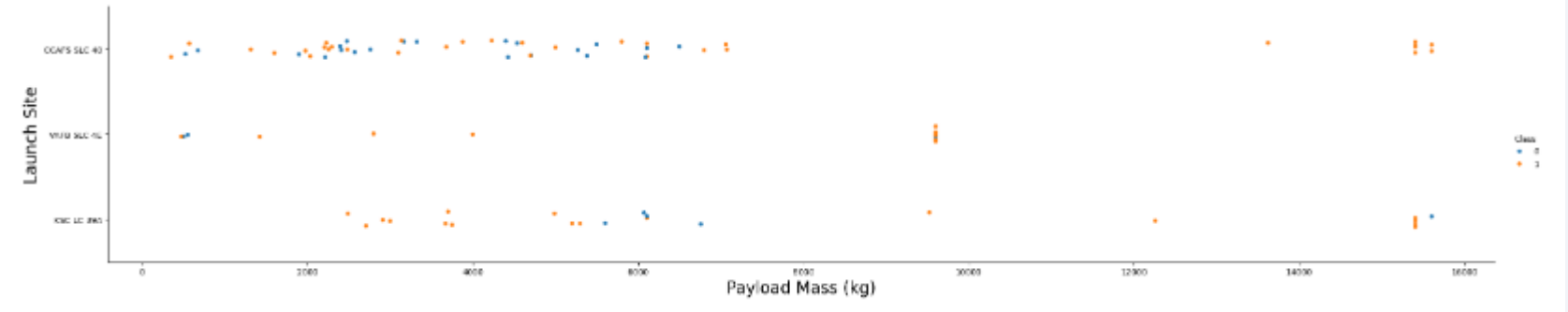
Flight Number vs. Launch Site

We can observe that the greater the number of flights at a launch site, the greater the success rate.



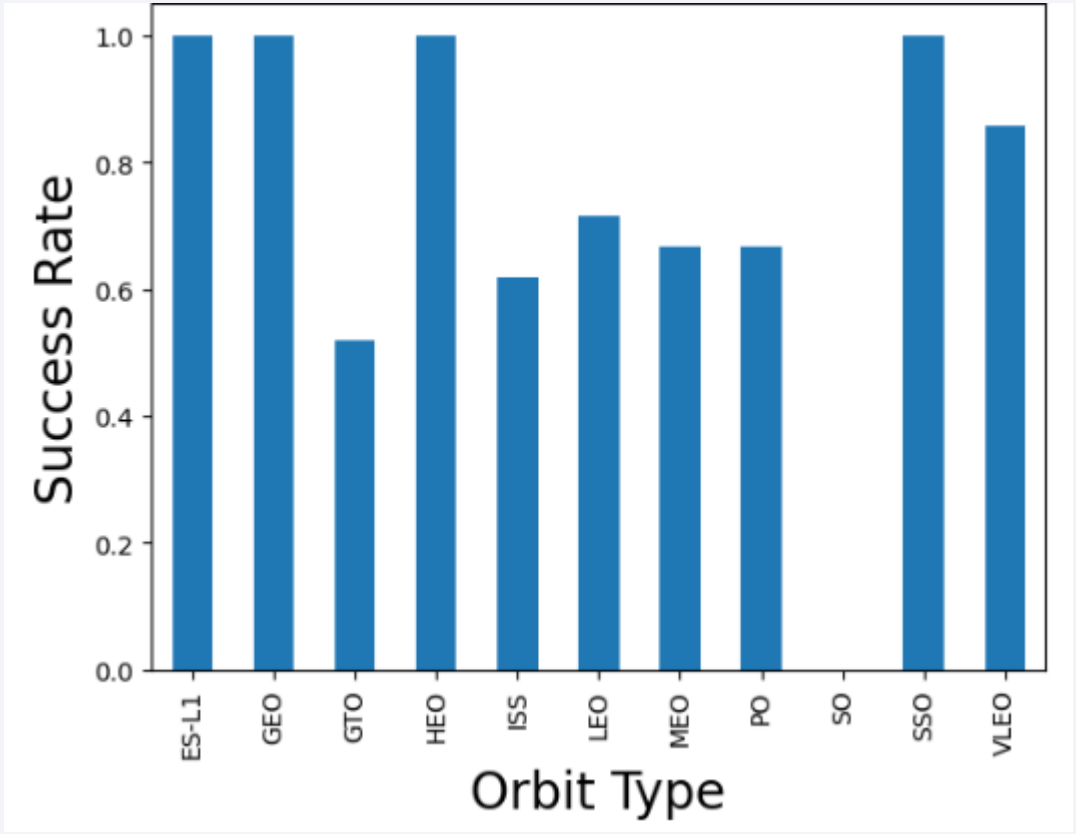
Payload vs. Launch Site

- Now if we observe Payload Vs. Launch Site scatter point chart we will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).



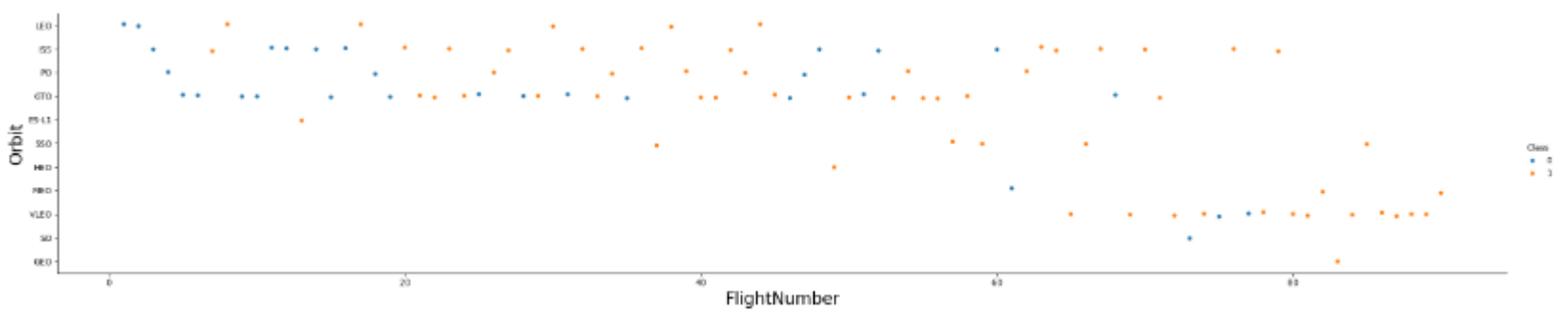
Success Rate vs. Orbit Type

- We can observe that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



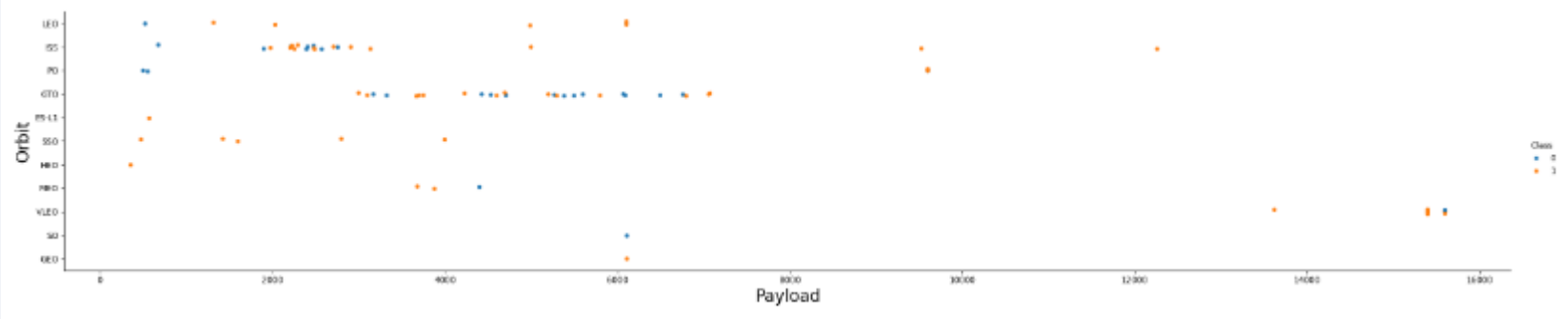
Flight Number vs. Orbit Type

- We can see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



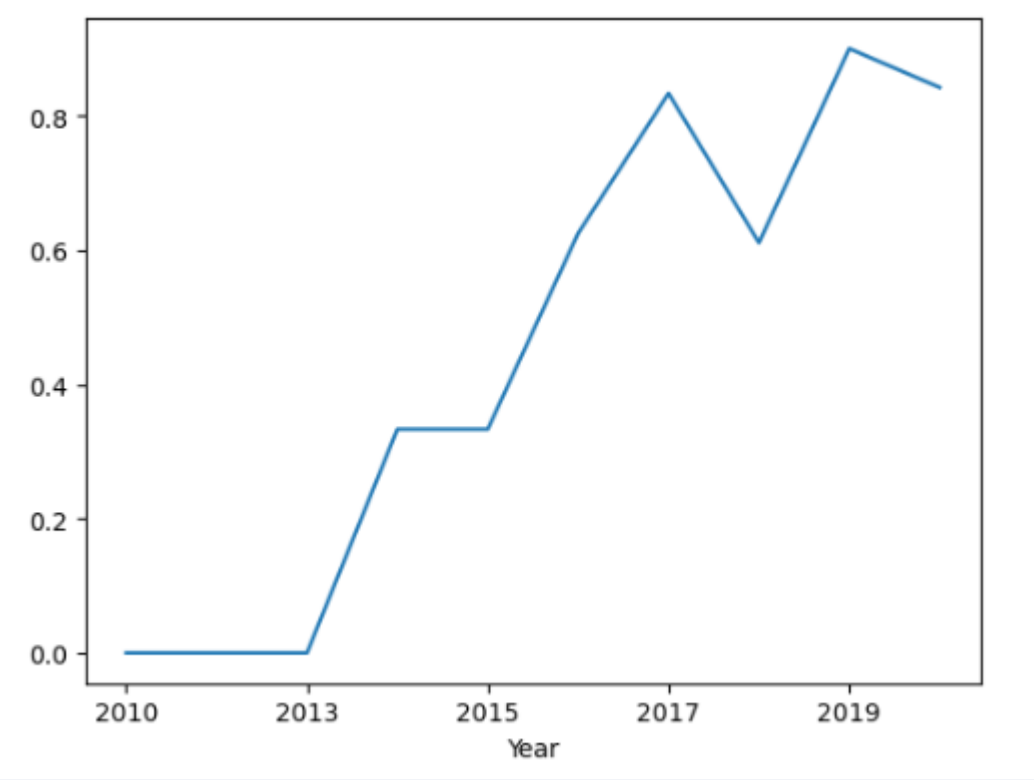
Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.



Launch Success Yearly Trend

- We can observe that the sucess rate since 2013 kept increasing till 2020



All Launch Site Names

- We find the names of the unique launch sites in the space mission with SELECT DISTINCT.

Task 1

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

In [18]:
```
%sql select distinct(LAUNCH_SITE) from SPACEXTBL
```

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

Out[18]:

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

Launch Site Names Begin with 'CCA'

- We used that query to display 5 records where launch sites begin with the string `CCA`

Task 2

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

In [19]:
```
%sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

Out[19]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcom
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attemp
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attemp
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attemp

Total Payload Mass

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

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

```
In [20]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
 * sqlite:///my_data1.db
Done.
```

Out[20]:

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- We calculate the average payload mass carried by booster version F9 v1.1 as 2928.4 with the SQL query below.

Task 4

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

```
In [21]: %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
 * sqlite:///my_data1.db
Done.
Out[21]: avg(PAYLOAD_MASS__KG_)
         2928.4
```

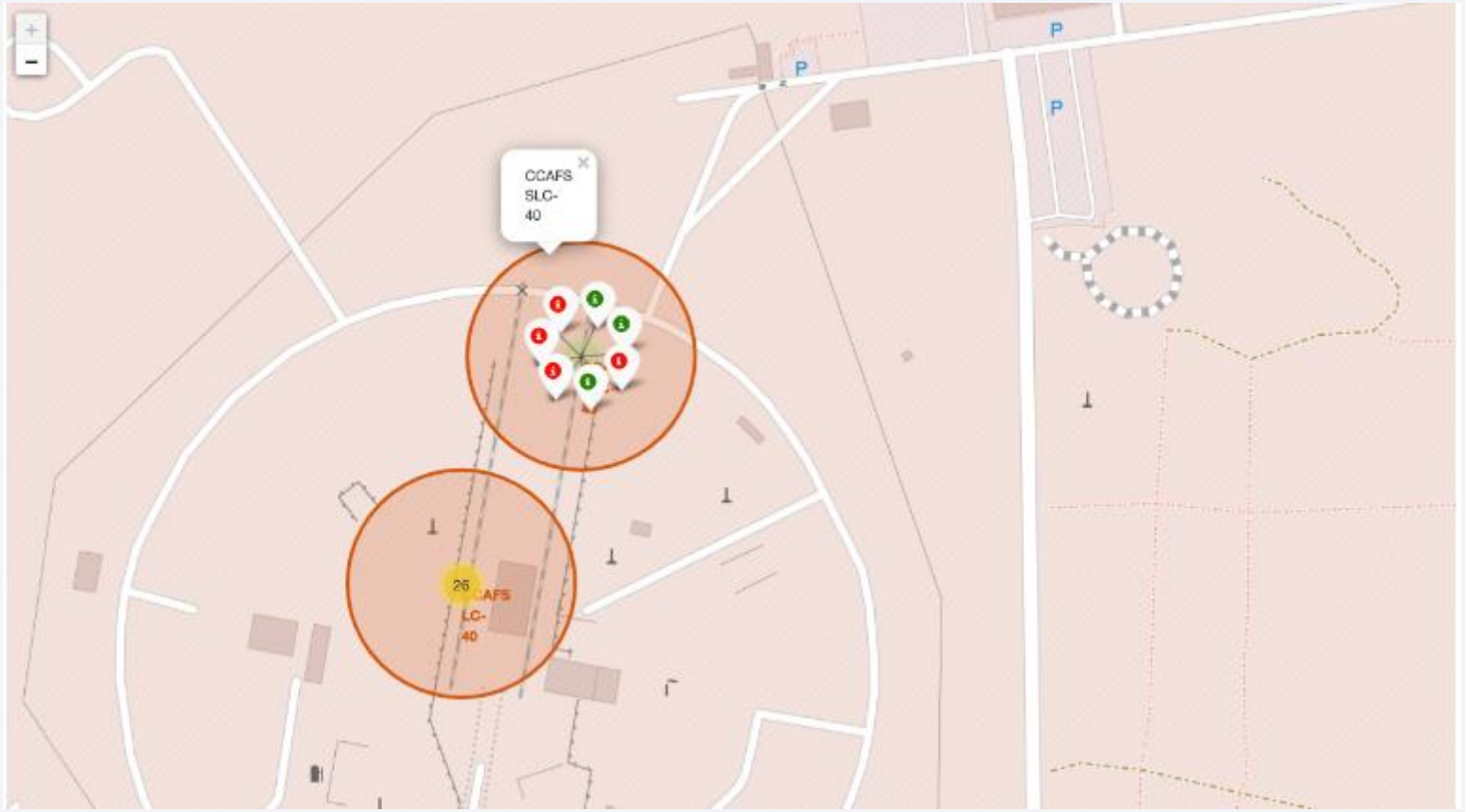
Section 3

Launch Sites Proximities Analysis

Folium Map


Minneapolis
Boise
Toronto
Milwaukee
Detroit
Chicago
Salt Lake City
Lincoln
Denver
Kansas City
Philadelphia
Sacramento
Washington
San Jose
Richmond
Fresno
Tulsa
Raleigh
Las Vegas
Albuquerque
Memphis
Charlotte
VAFB SLC
Los Angeles
4E
Phoenix
Atlanta
Tijuana
Mexicali
Tucson
Dallas
Ciudad Juárez
Austin
Houston
Baton Rouge
Jacksonville
San Antonio
CCAFS SLC-40
Monterrey
Reynosa
Miami
Nassau
Saltillo
La Paz
Culiacán
México
The Bahamas

Folium Map



Folium Map


P
P
P
7
CAFS
SLC-
40
26
AFS-
LC-
40
0.90
KM
Samuel C Phillips Pkwy
Centaur Road

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Best model was the decision tree classifier with the highest classification accuracy,

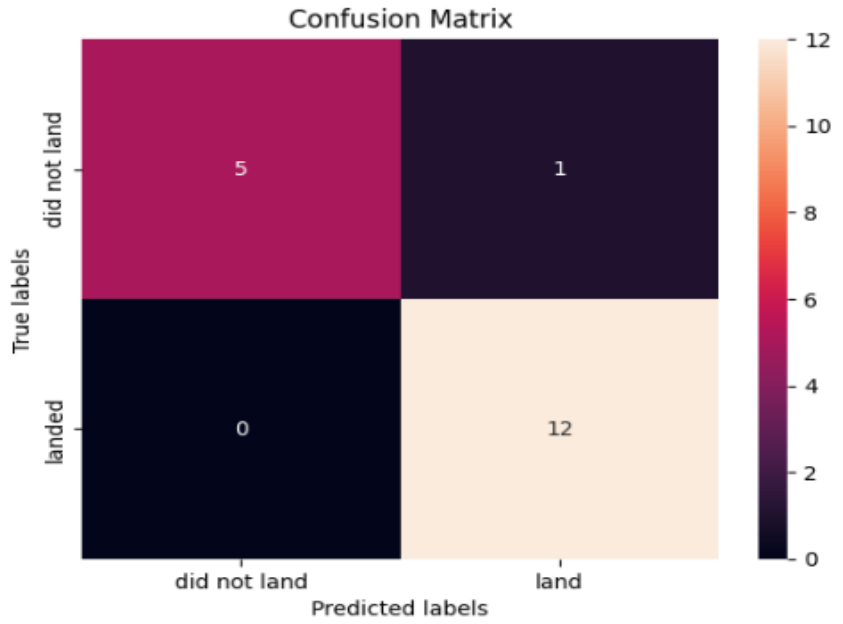
Calculate the accuracy of tree_cv on the test data using the method `score`:

In [29]:
```
print("test set accuracy :",tree_cv.score(X_test, Y_test))
```

```
test set accuracy : 0.9444444444444444
```

We can plot the confusion matrix

In [30]:
```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate was increasing from 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO are the most success rate.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!