Applied Data Science for Rocket launching

Grégoire Hébert - 25/01/2022

Outline

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

Executive Summary

The goal of this project is to estimate if the first stage of Falcon 9 would land successfully or not, as it plays a major role in predicting the price of its relaunch. The data for this project was sourced from SpaceX REST API, and Wikipedia. After performing some data wrangling In order to determine the best predictors for our outcome, Exploratory Data Analysis (EDA) and feature scaling were done with the help of visualization using scatter and line plots. Later some Machine Learning (ML) models were created to predict future outcomes.

The results showed that the outcome was dependent on the orbit, mass of payload, launch site, and various other technical factors such as gridfins, cores, etc.

Introduction

The evolution of technologies has changed the lives of people a lot, and with the current technologies, we are on the verge of building commercial space flights; which could make humans multi-planetary species. There are major companies in this space race, namely Blue Origin, Virgin Galactic, and SpaceX. The current leader in this race seems to be SpaceX, and the reason behind that is the reusability of their stage 1. This difference reduces the launch price from 165M$ (average price of their competitors) to 62M$ for SpaceX.

The problem that we are trying to answer is the following : how can we predict the launch price of Falcon 9, so that we can use this data for companies that want to compete with SpaceX? Predicting whether stage 1 will land successfully or not plays a crucial role in predicting the launch price. There are many variables involved, and we need to predict which one are crucial for reusing this stage 1.

Section 1 : Methodology

Methodology step by step :

- **Data collection methodology :**

  The data was collected from SpaceX REST API and from Wikipedia, using BeautifulSoup to perform web scraping and retrieve these data

- **Data wrangling steps :**
  - The null values were handled by replacing them with the mean value
  - One-hot encoding was done on categorical variables such as orbit, launch site, landing pad and serial.
- **Exploratory Data Analysis (EDA),** using visualization and SQL
- Creation of **interactive visual analytics**, using Folium and Plotly Dash
- Performing **predictive analysis** using classification models
- Using various **Machine Learning models** like SVM, logistic regression, tree classifier and k nearest neighbours

Step 1 : Data Collection

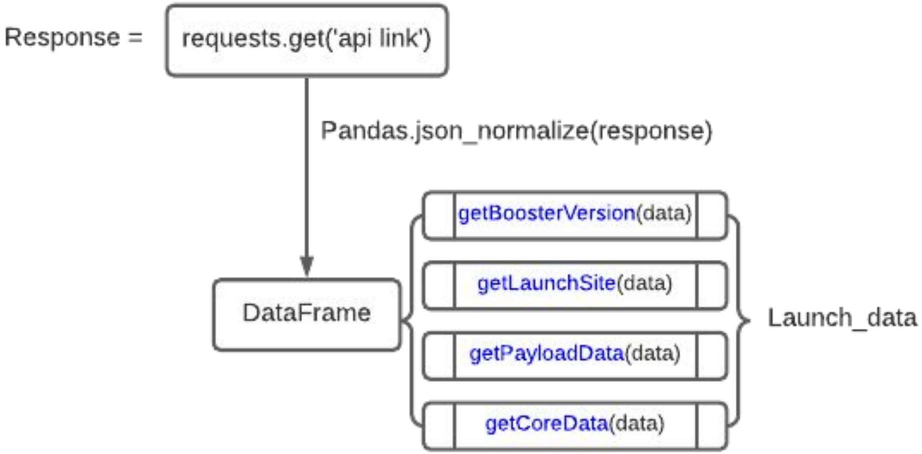
- The first datasets were collected directly from the **SpaceX REST API** : https://api.spacexdata.com/v4/launches/past

→ This dataset provides data like type of rocket used, payload, launch / landing specifications, landing outcome, for each launch.

- Data was also collected via **web scraping, using BeautifulSoup**, as the list of Falcon 9 and Falcon Heavy launches is available online on Wikipedia : List of Falcon/ 9/ and Falcon Heavy launches - Wikipedia

Step 1 : Data Collection using SpaceX REST API

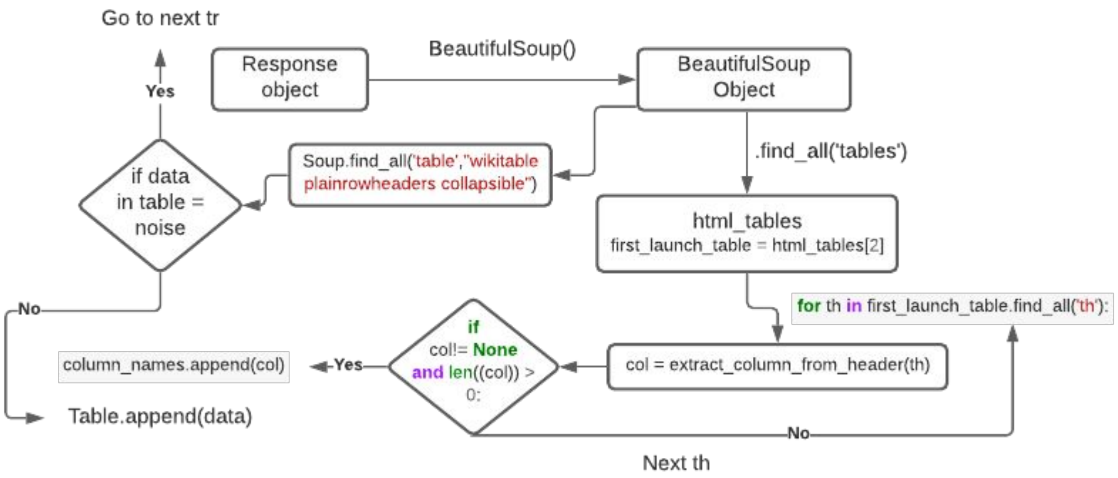
- First a **request object** was created using the API.
- The **response object was converted into a dataframe** using pandas.json_normalize(response.json())



For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/spacex-data-collection-api.ipynb

Step 2 : Web scraping through Wikipedia

- Thanks to the **BeautifulSoup** framework, we were able to **collect data directly from a Wikipedia page**. Here is the coding process :



For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/spacex-webscraping.ipynb

Data wrangling

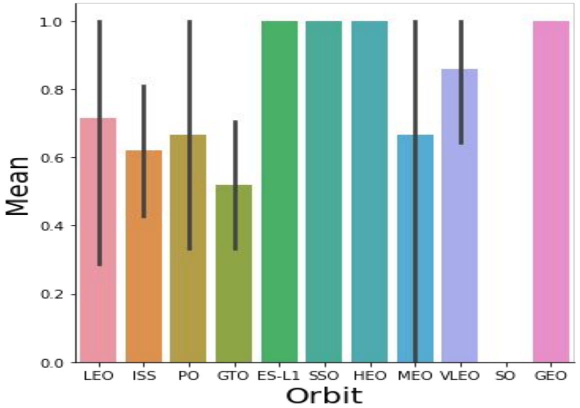
- The original dataset contained Falcon 1 and Falcon 9 launches ; we first filtered it to keep only the **Falcon 9 data**.
- Then we adressed the **missing values** on two columns:
    - When “PayloadMass” was missing, we replaced it by the mean value
    - When “Landingpad” was missing we left a null value, because it meant that no landing pad were used.
- We used **one-hot encoding** to represent some categorical data into binary vectors (failure = 0, success = 1) and be able to “feed” them into ML algorithms. Thanks to this, we found out that the success rate was 66%.

For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/Data%20Wrangling.ipynb

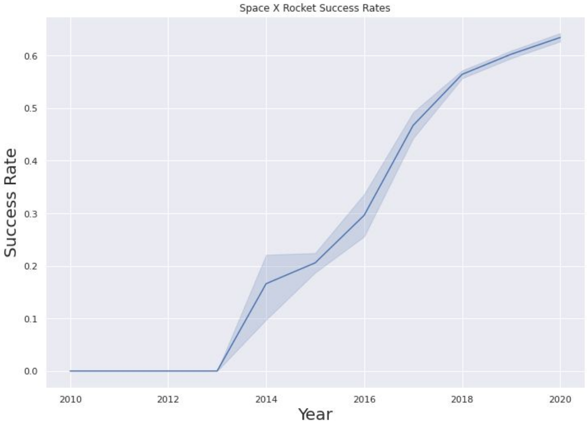
Exploratory Data Analysis (EDA) with Data Visualisation

We created some bar and line graphs to show how variable relates one to another.

- Bar Graph : **Mean vs Orbit**



- Line Graph : **Success Rate vs Year**

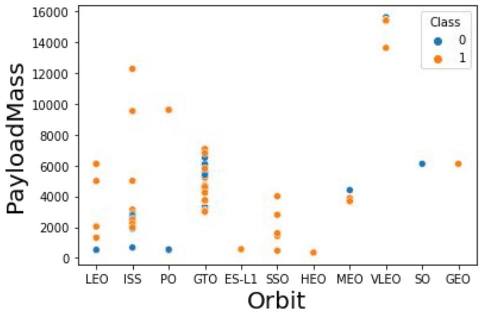


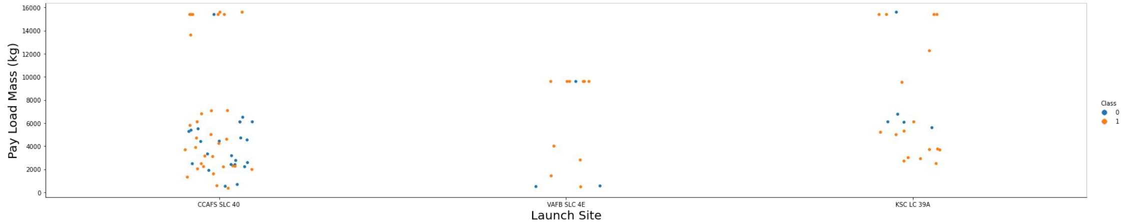
For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/eda-dataviz.ipynb

EDA with Data Visualisation

We also created some **scatter plots** to understand better our data.

- Flight number vs PayloadMass
- Flight number vs Launch Site
- PayloadMass vs Launch Site
- Orbit vs Flight number
- Payload vs Orbit Type





For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/eda-dataviz.ipynb

EDA with SQL

We performed **SQL queries** to gather information about our dataset. Here are the information we extracted :

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

In [8]:
```
db.GetRecordsOfColumn('select DISTINCT Launch_Site from tblSpaceX','Launch_Site')
```

Out[8]:
```
['CCAFS LC-40', 'CCAFS SLC-40', 'CCAFSSLC-40', 'KSC LC-39A', 'VAFB SLC-4E']
```

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

In [41]:
```
import pyodbc
import pandas as pd
import numpy as np
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=localhost;'
                      'Database=SpaceX;'
                      'User ID=admin;Password=admin;')
cursor = conn.cursor()

cursor.execute("select TOP 5 * from tblSpaceX WHERE Launch_Site LIKE 'KSC%'")
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

Out[41]:

	Date	Time_UTC	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS_KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	19-02-2017	2021-07-02 14:39:00.0000000	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
1	16-03-2017	2021-07-02 06:00:00.0000000	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
2	30-03-2017	2021-07-02 22:27:00.0000000	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
3	01-05-2017	2021-07-02 11:15:00.0000000	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
4	15-05-2017	2021-07-02 23:21:00.0000000	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

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

In [57]:
```
TPM = db.GetRecordsOfColumn("select SUM(PAYLOAD_MASS_KG_) TotalPayloadMass from tblSpaceX where Customer = 'NASA (CRS)'",'TotalPay
ndf= pd.DataFrame(TPM)
ndf.columns = ['Total Payload Mass']
ndf
```

Out[57]:

	Total Payload Mass
0	45596

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

In [62]:
```
APM = db.GetRecordsOfColumn("select AVG(PAYLOAD_MASS_KG_) AveragePayloadMass from tblSpaceX where Booster_Version = 'F9 v1.1'",'Av
ndf= pd.DataFrame(APM)
ndf.columns = ['Average Payload Mass']
ndf
```

Out[62]:

	Average Payload Mass
0	2928

For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/EDA%20SQL.ipynb

EDA with SQL

We performed **SQL queries** to gather information about our dataset. Here are the information we extracted :

List the date where the succesful landing outcome in drone ship was acheived.

In [64]:
```
SLO = db.GetRecordsOfColumn("select MIN(Date) SLO from tblSpaceX where Landing_Outcome = 'Success (drone ship)'","SLO')
ndf= pd.DataFrame(SLO)
ndf.columns = ['Date which first Successful landing outcome in drone ship was acheived.']
ndf
```

Out[64]:

	Date which first Successful landing outcome in drone ship was acheived.
0	06-05-2016

List the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 600(

In [69]:
```
SLO = db.GetRecordsOfColumn("select Booster_Version from tblSpaceX where Landing_Outcome = 'Success (ground pad)' AND Payload_MASS_K
ndf= pd.DataFrame(SLO)
ndf.columns = ['Date which first Successful landing outcome in drone ship was acheived.']
ndf
```

Out[69]:

	Date which first Successful landing outcome in drone ship was acheived.
0	F9 FT B1032.1
1	F9 B4 B1040.1
2	F9 B4 B1043.1

List the total number of successful and failure mission outcomes

In [84]:
```
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=localhost;'
                      'Database=SpaceX;'
                      'User ID=admin;Password=admin;')

cursor = conn.cursor()

cursor.execute("SELECT(SELECT Count(Mission_Outcome) from tblSpaceX where Mission_Outcome LIKE '%Success%') as Successful_Mission_Outc
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

Out[84]:

	Successful_Mission_Outcomes	Failure_Mission_Outcomes
0	100	1

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

In [94]:
```
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=localhost;'
                      'Database=SpaceX;'
                      'User ID=admin;Password=admin;')

cursor = conn.cursor()

cursor.execute("SELECT DISTINCT Booster_Version, MAX(PAYLOAD_MASS_KG_) AS [Maximum Payload Mass] FROM tblSpaceX GROUP BY Booster
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

Out[94]:

	Booster_Version	Maximum Payload Mass
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
...	...	...
92	F9 v1.1 B1003	500
93	F9 FT B1038.1	475
94	F9 B4 B1045.1	362
95	F9 v1.0 B0003	0
96	F9 v1.0 B0004	0

97 rows × 2 columns

For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/EDA%20SQL.ipynb

EDA with SQL

We performed **SQL queries** to gather information about our dataset. Here are the information we extracted :

List the records which will display the month names, succesful landing_outcomes in ground pac
in year 2017

In [96]:
```
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=localhost;'
                      'Database=SpaceX;'
                      'User ID=admin;Password=admin;')

cursor = conn.cursor()

cursor.execute("SELECT   DateName( month , DateAdd( month , MONTH(CONVERT(date,Date, 105)) , 0 ) - 1
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

Out[96]:

	Month	Booster_Version	Launch_Site	Landing_Outcome
0	January	F9 FT B1029.1	VAFB SLC-4E	Success (drone ship)
1	February	F9 FT B1031.1	KSC LC-39A	Success (ground pad)
2	March	F9 FT B1021.2	KSC LC-39A	Success (drone ship)
3	May	F9 FT B1032.1	KSC LC-39A	Success (ground pad)
4	June	F9 FT B1035.1	KSC LC-39A	Success (ground pad)
5	June	F9 FT B1029.2	KSC LC-39A	Success (drone ship)
6	June	F9 FT B1036.1	VAFB SLC-4E	Success (drone ship)
7	August	F9 B4 B1039.1	KSC LC-39A	Success (ground pad)
8	August	F9 FT B1038.1	VAFB SLC-4E	Success (drone ship)
9	September	F9 B4 B1040.1	KSC LC-39A	Success (ground pad)
10	October	F9 B4 B1041.1	VAFB SLC-4E	Success (drone ship)
11	October	F9 FT B1031.2	KSC LC-39A	Success (drone ship)
12	October	F9 B4 B1042.1	KSC LC-39A	Success (drone ship)

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

n [90]:
```
sl = db.GetRecordsOfColumn("SELECT COUNT(Landing_Outcome) AS sl FROM dbo.tblSpaceX WHERE (Landing_Outcome LIKE '%Success%') AND (Date >'04-06-2010') AN

ndf= pd.DataFrame(sl)
ndf.columns = ['Successful Landing Outcomes Between 2010-06-04 and 2017-03-20']
ndf
```

ut[90]:

	Successful Landing Outcomes Between 2010-06-04 and 2017-03-20
0	34

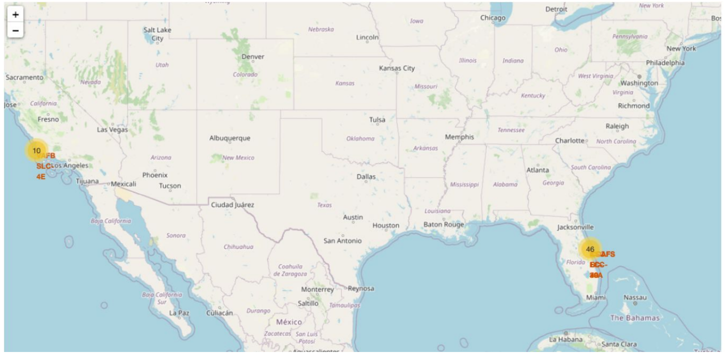
For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/EDA%20SQL.ipynb

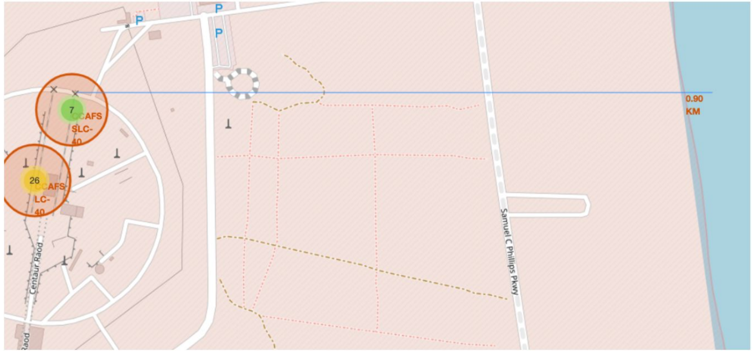
Interactive Map with Folium

In order to visualize the data on an interactive map, we used different types of markers, highlighted circles, etc. for each different launch sites. Cluster objects were created to visualize the launch outcomes :

- 0 = red = unsuccessful
- 1 = green = successful

We also added a polyline object to show distance (calculated using Haversine’s formula) between the launch site and various landmarks such as railways, highways, etc.





For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/Interactive%20folium%20map

Flask / Plotly Dash dashboard

An **interactive web application** was created using **Dash**, with a drop-down menu to **select the launch site, and range-slider to choose the range of payload**.

Interactive pie chart showing success rate of all launch sites by default, and a **scatter plot** showing launch outcomes of all sites according to their payloads in the default range(0-10000) were added.

Dropdown menu would allow the user to choose the launch site that would alter the figure of pie chart and scatter plot to show outcomes of that launch site, and through the range-slider user can select the range of payload on the x-axis of scatter plot. These interactions would allow the user to visualise the data more in depth according to his needs.

For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/Dashboard%20SpaceX%20Dataset.ipynb

Predictive Analysis (Classification)

We used several types of Machine Learning (ML) models to try and predict the outcome of new launches, based on existing data. We used :

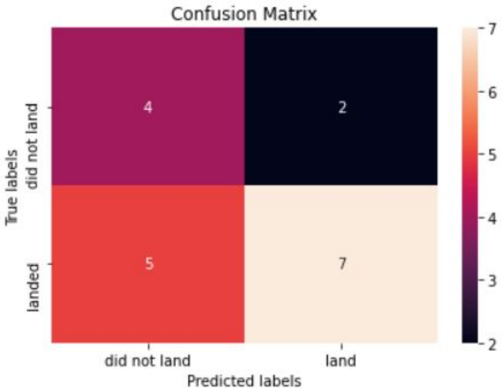
- logistic regression
- SVM
- decision tree
- KNN

After building, training, evaluating the performance and improving our different models, we evaluated the predictive performance of each models to find the best one. In the end, the best was KNN with a R^2 of 0.83.

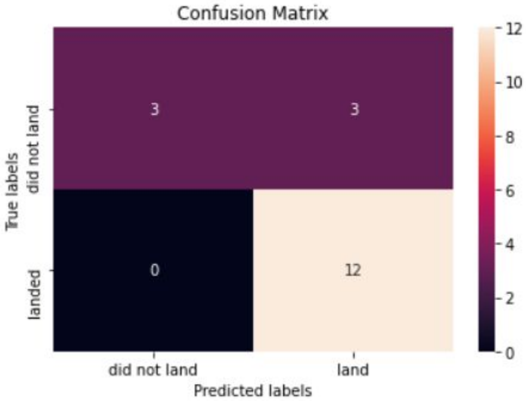
For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/machine-learning-prediction-spacex.ipynb

Predictive Analysis (Classification) : Results

Confusion matrix of the tree algorithm on the test data :



Confusion matrix of the KNN algorithm on the test data :



We can see that KNN classified with a better accuracy : for example there are no false negatives (when the model predicted it would not land, it really did not land in reality).

For the complete code and output, please follow the Github link below:
https://github.com/Greg156/IBM-Data-Science-Final-Project-SpaceX-/blob/main/machine-learning-prediction-spacex.ipynb