WINNING THE SPACEX FLIGHT WITH DATA SCIENCE

ONYIUKE AFAM RAPHAEL

16/07/2023

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - Data collection
    - SpaceX API
    - Scrapping
    - Wrangling
    - EDA With Data Visualization
    - EDA With SQL
  - Dashboard
    - Build an interacting map with Folium
    - Build a dashboard with plotly dash
    - Predictive analysis (classification)
    - Results
- Discussion
  - Insight drawn from EDA
- Conclusion
- Appendix
  - Launch success yearly trend
  - Classification Accuracy
  - Launch site names begin with SSA

EXECUTIVE SUMMARY

- Summary of Methodology
  - Data Collection via API, Web Scraping
  - Exploratory Data Analysis (EDA) with Data Visualization
  - EDA with SQL
  - Interactive Map with Folium
  - Dashboards with Plotly Dash
  - Predictive Analysis
- Summary of all results
  - Exploratory Data Analysis results
  - Interactive maps and dashboard
  - Predictive results

INTRODUCTION

- Project background and context
  - The aim of this project is to predict if the Falcon 9 first stage will successfully land. SpaceX says on its website that the Falcon 9 rocket launch cost 62 million dollars. Other providers cost upward of 165 million dollars each. The price difference is explained by the fact that SpaceX can reuse the first stage. By determining if the stage will land, we can determined the cost of a launch. This information is interesting for another company if it wants to compete with SpaceX for a rocket launch.
- Problems you want to find answers
  - What are the main characteristics of a successful or failed landing?
  - What are the effects of each relationship of the rocket variables on the success or failure of a landing?
  - What are the conditions which allow SpaceX to achieve the best landing success rate?

METHODOLOGY

- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Host Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analysis using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Datasets are from Rest SpaceX API and webscrapping Wikipedia
  - The information obtained by the API are rocket, launches, payload information
    - The Space X REST API URL is api.spacexdata.com/v4/

- The information obtained by the webscrapping of Wikipedia are launches, landing, payload information
  - URL is https://en.wikipedia.org/w/index.php?title=List of Falcon 9 and FalconHeavy launches&oldid=1027686922

Data Collection - SpaceX API

1. Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

2. Convert Response to JSON File

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

3. Transform data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
getBoosterVersion(data)
```

4. Create dictionary with 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}
```

5. Create dataframe

```
data = pd.DataFrame.from_dict(launch_dict)
```

6. Filter dataframe

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

7. Export to file

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Link to code

Data Collection - Scraping

1. Getting Response from HTML

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

2. Create BeautifulSoup Object

```
soup = BeautifulSoup(response.text, "html5lib")
```

3. Find all tables

```
html_tables = soup.findAll('table')
```

4. Get column names

```
for th in first_launch_table.find_all('th'):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0 :
        column_names.append(name)
```

5. Create dictionary

```
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. Add data to keys

```
#Extract each table
for table_number,table in enumerate(soup.find_all
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is a
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.stri
                flag=flight_number.isdigit()
```

7. Create dataframe from dictionary

```
df=pd.DataFrame(launch_dict)
```

8. Export to file

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

- In the dataset, there are several cases where the booster did not land successully.
  - True Ocean, True RTLS, True ASDS means the mission has been successful.
  - False Ocean, False RTLS, False ASDS means the mission was a failure.
- We need to transform string variables into categorical variables where 1 means the mission has been successful and 0 means the mission was a failure.

1. Calculate launches number for each site

```
df['LaunchSite'].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

2. Calculate the number and occurence of each orbit

```
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
SO        1
ES-L1     1
HEO       1
GEO       1
Name: Orbit, dtype: int64
```

3.Calculate number and occurrence of outcome per orbit type

```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
None ASDS       2
False Ocean     2
False RTLS      1
Name: Outcome, dtype: int64
```

4.Create landing outcome mission label from Outcome column

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

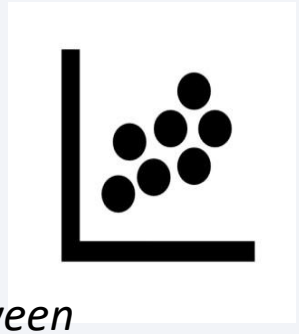
5. Export to file

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

EDA with Data Visualization

Scatter Graphs

Flight Number vs. Payload Mass
Flight Number vs. Launch Site
Payload vs. Launch Site
Orbit vs. Flight Number
Payload vs. Orbit Type
Orbit vs. Payload Mass



Scatter plots show relationship between variables. This relationship is called the correlation.

Bar Graph

Success rate vs. Orbit

Bar graphs show the relationship between numeric and categoric variables.

Line Graph

Success rate vs. Year

Line graphs show data variables and their trends. Line graphs can help to show global behavior and make prediction for unseen data.

EDA with SQL

- We performed SQL queries to gather and understand data from dataset:
  - Displaying the names of the unique launch sites in the space mission.
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS).
  - Display average payload mass carried by booster version F9 v1.1.
  - List the date when the first successful landing outcome in ground pad was achieved.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - List the total number of successful and failure mission outcomes.
  - List the names of the booster versions which have carried the maximum payload mass.
  - 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.
  - Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

Build an Interactive Map with Folium

- Folium map object is a map centered on NASA Johnson Space Center at Houson, Texas
  - Red circle at NASA Johnson Space Center's coordinate with label showing its name *(folium.Circle, folium.map.Marker).*
  - Red circles at each launch site coordinates with label showing launch site name *(folium.Circle, folium.map.Marker, folium.features.DivIcon).*
  - The grouping of points in a cluster to display multiple and different information for the same coordinates

(folium.plugins.MarkerCluster).

  - Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing.

(folium.map.Marker, folium.Icon).

  - Markers to show distance between launch site to key locations (railway, highway, coastway, city) and plot a line between them.

(folium.map.Marker, folium.PolyLine, folium.features.DivIcon)

These objects are created in order to understand better the problem and the data. We can show easily all launch sites, their surroundings and the number of successful and unsuccessful landings.

Build a Dashboard with Plotly Dash

- Dashboard has dropdown, pie chart, rangeslider and scatter plot components
  - Dropdown allows a user to choose the launch site or all launch sites *(dash_core_components.Dropdown)*.
  - Pie chart shows the total success and the total failure for the launch site chosen with the dropdown component *(plotly.express.pie)*.
  - Rangeslider allows a user to select a payload mass in a fixed range *(dash_core_components.RangeSlider)*.
  - Scatter chart shows the relationship between two variables, in particular Success vs Payload Mass *(plotly.express.scatter)*.

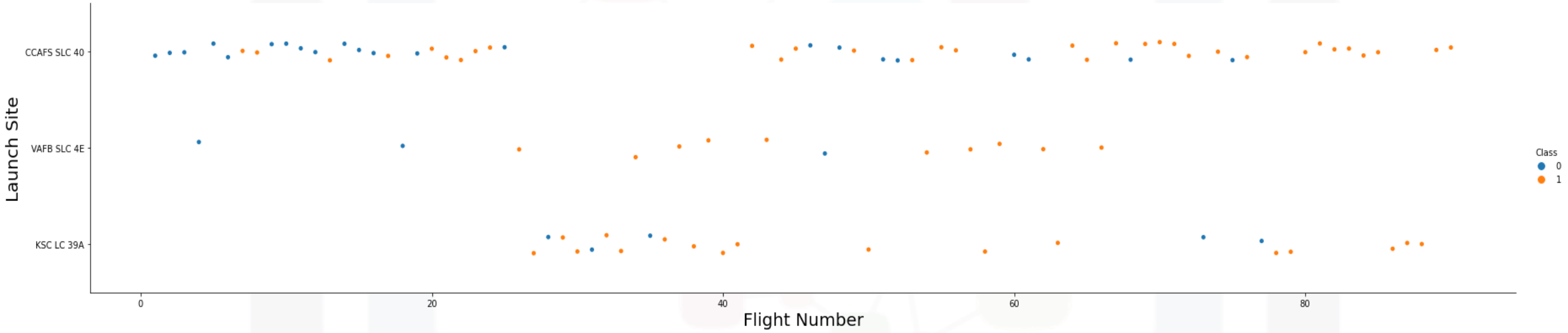
Predictive Analysis (Classification)

- Data preparation
  - Load dataset
  - Normalize data
  - Split data into training and test sets.
- Model preparation
  - Selection of machine learning algorithms
  - Set parameters for each algorithm to GridSearchCV
  - Training GridSearchModel models with training dataset
- Model evaluation
  - Get best hyperparameters for each type of model
  - Compute accuracy for each model with test dataset
  - Plot Confusion Matrix
- Model comparison
  - Comparison of models according to their accuracy
  - The model with the best accuracy will be chosen (see Notebook for result)

Results

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

Insight Drawn From EDA



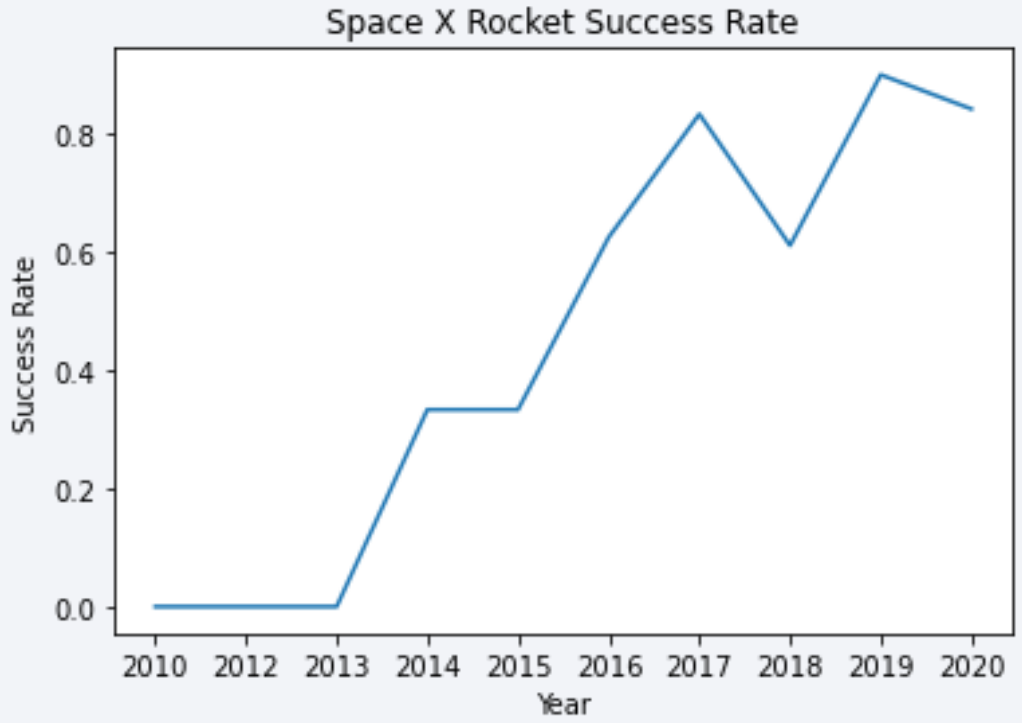
We observe that, for each site, the success rate is increasing.

Flight Number vs Launch Site

Conclusions

- The success of a mission can be explained by several factors such as the launch site, the orbit and especially the number of previous launches. Indeed, we can assume that there has been a gain in knowledge between launches that allowed to go from a launch failure to a success.
- The orbits with the best success rates are GEO, HEO, SSO, ES-L1.
- Depending on the orbits, the payload mass can be a criterion to take into account for the success of a mission. Some orbits require a light or heavy payload mass. But generally low weighted payloads perform better than the heavy weighted payloads.
- With the current data, we cannot explain why some launch sites are better than others (KSC LC-39A is the best launch site). To get an answer to this problem, we could obtain atmospheric or other relevant data.
- For this dataset, we choose the Decision Tree Algorithm as the best model even if the test accuracy between all the models used is identical. We choose Decision Tree Algorithm because it has a better train accuracy.

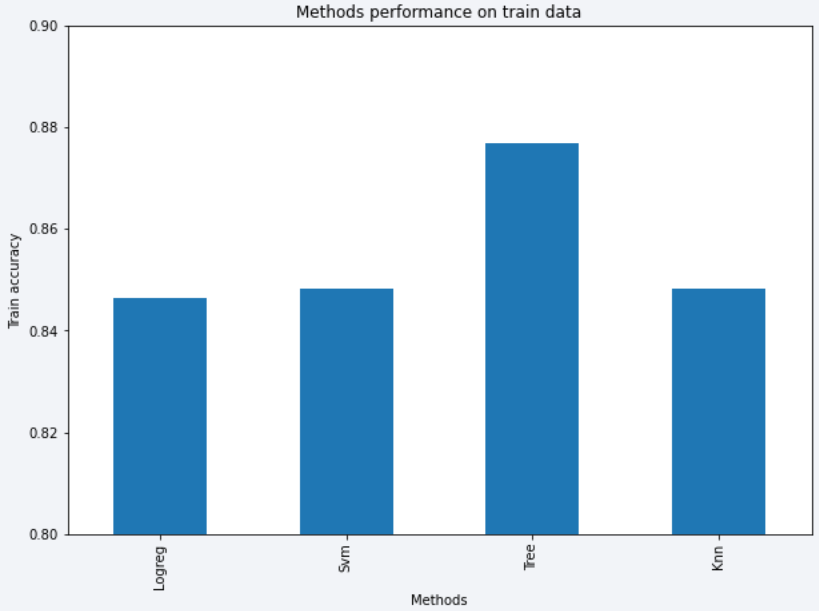
Launch Success Yearly Trend

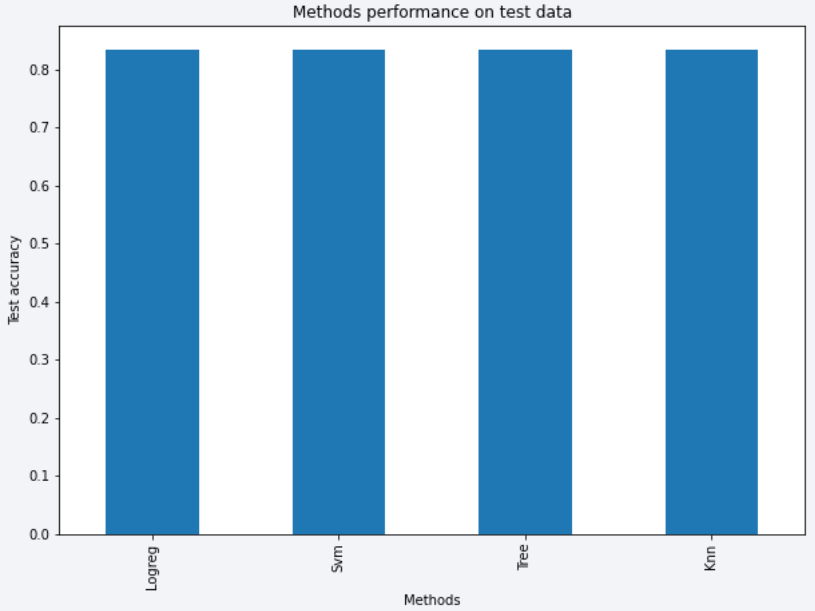


Since 2013, we can see an increase in the Space X Rocket success rate.

Classification Accuracy

	Accuracy Train	Accuracy Test
Tree	0.876786	0.833333
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333





For accuracy test, all methods performed similar. We could get more test data to decide between them. But if we really need to choose one right now, we could take the decision tree.

Decision tree best parameters

```
tuned hyperparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'sqrt', 'min_samples_leaf':
4, 'min_samples_split': 2, 'splitter': 'random'}
```

Launch Site Names Begin with 'CCA'

SQL Query

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

Results

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
08-12-2010	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
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)

Explanation

The WHERE clause followed by LIKE clause filters launch sites that contain the substring CCA. LIMIT 5 shows 5 records from filtering.