IBM DATA SCIENCE CAPSTONE PROJECT

Kathy Yuen

15 Sep 2021

OUTLINE

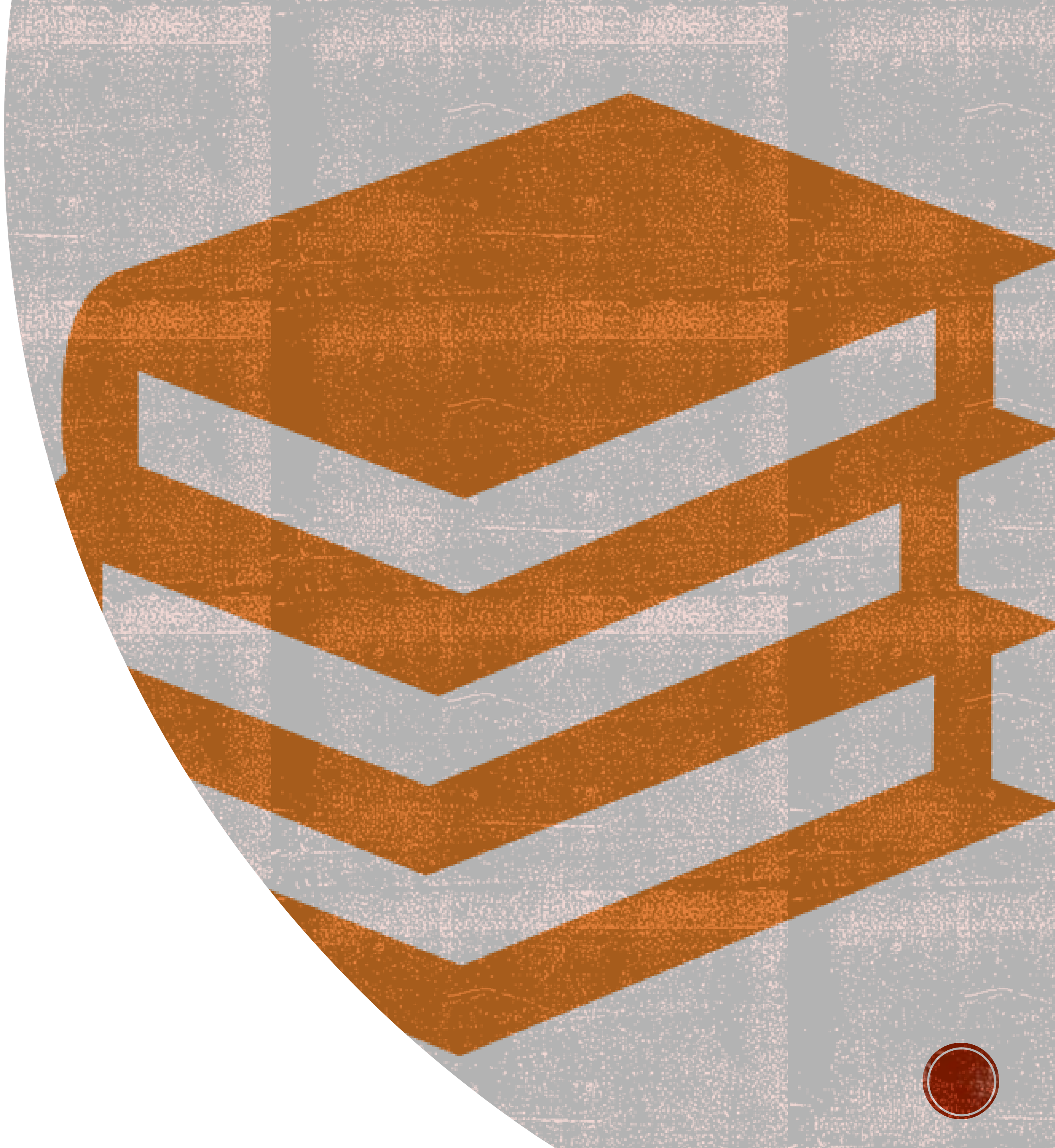
- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Discussion
  - Findings & Implications
- Conclusion
- Appendix

EXECUTIVE SUMMARY

- Summary of methodologies
  - Data Collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a dashboard with Plotly Dash
  - Predictive analysis
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

INTRODUCTION

- Project Background
  - We predicted if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 millions; other providers cost upward of 165 millions 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.
- Common problems that needed solving.
  - What influences if the rocket will land successfully?
  - The effect each relationship with certain rocket variables will impact in determining the success rate of a successful landing.
  - What conditions does SpaceX have to achieve to get the best results and ensure the best rocket success landing rate.



METHODOLOGY

- Data collection methodology:
  - SpaceX Rest API
  - Web Scrapping from Wilkipedia
- Performed data wrangling
- Performed exploratory data analysis (EDA) with visualization and SQL
- Performed interactive visual analytics using Folium and Plotly Dash
- Performed predictive analysis with classification models

DATA COLLECTION SPACEX API

simplified flow chart

1 .Getting Response from API

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

2. Converting Response to a .json file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Apply custom functions to clean data

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

```
getBoosterVersion(data)
```

4. Assign list to dictionary then dataframe

```
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}
```

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

5. Filter dataframe and export to flat file (.csv)

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

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

WEB SCRAPPING

simplified flow chart

1 .Getting Response from HTML

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

2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

3. Finding tables

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

4. Getting column names

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']


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'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Appending data to keys (refer) to notebook block 12

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
            # get table row
             for rows in table.find_all("tr")
                 #check to see if first table
```

7. Converting dictionary to dataframe

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

8. Dataframe to .CSV

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

DATA WRANGLING

Introduction

In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, 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 on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.
We mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

Process

Perform Exploratory Data Analysis EDA on dataset

Calculate the number of launches at each site

Calculate the number and occurrence of each orbit

Calculate the number and occurrence of mission outcome per orbit type

Export dataset as .CSV

Create a landing outcome label from Outcome column

Work out success rate for every landing in dataset

[GitHub URL to Notebook]()

Each launch aims to an dedicated orbit, and here are some common orbit types:

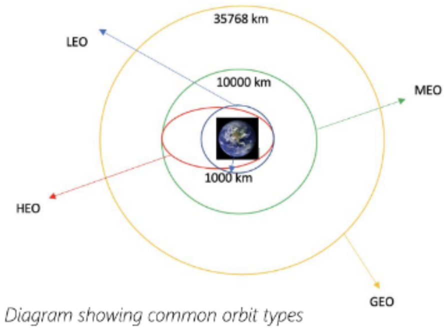


Diagram showing common orbit types SpaceX uses

EDA WITH DATA VISULIZATION

- Different Graphs were drawn to show the relationship among the data:
  - Scatter Graphs
  - Bar Graph
  - Line Graph

EDA WITH SQL

- Performed SQL queries to gather information about the dataset:

- Displaying the names of the unique launch sites in the space mission
- Displaying 5 records where launch sites begin with the string 'KSC'
- Displaying the total payload mass carried by boosters launched by NASA (CRS)
- Displaying average payload mass carried by booster version F9 v1.1
- Listing the date where the successful landing outcome in drone ship was achieved.
- Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
- Listing the total number of successful and failure mission outcomes
- Listing the names of the booster_versions which have carried the maximum payload mass.
- Listing the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
- Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

BUILD INTERACTIVE MAP WITH FOLIUM

- To visualize the Launch Data into an interative map.
- Using Haversine's formula to calculate the distance from the Launch SItes to different places to find out the surrunroudings. Lines are drawn on the map to measure the distances.

BUILD INTERACTIVE DASHBOARD WITH POLTLY

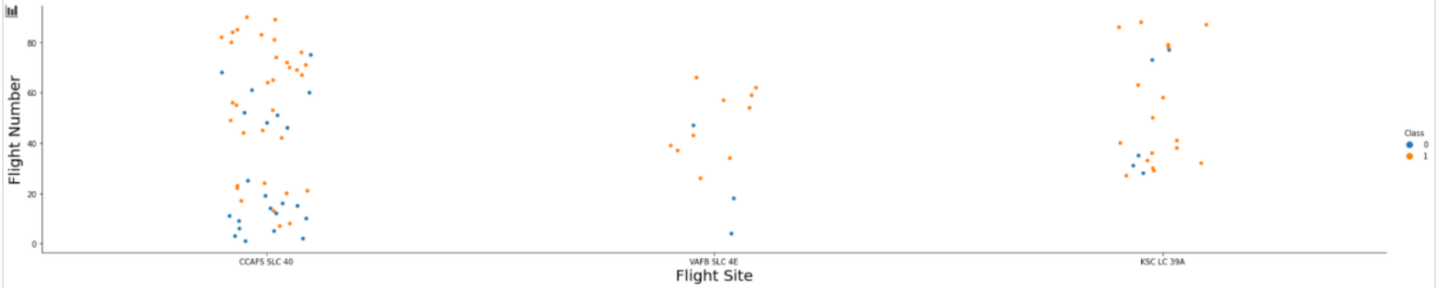
- To visualize the Launch Data into an interative dashboard with pie chart and scatter graph.
- Pie Chart
  - Showing th total launches at a certain site or all sites with a drop down menu.
- Scatter Graph
  - Showing the relationship with the outcome and payload mass for the different booster versions.

PREDICTIVE ANALYSIS

- Building model
- Evaluating model
- Improving model
- Finding the best performing model with the most accurate score

RESULTS- EDA WITH VISUALIZATION

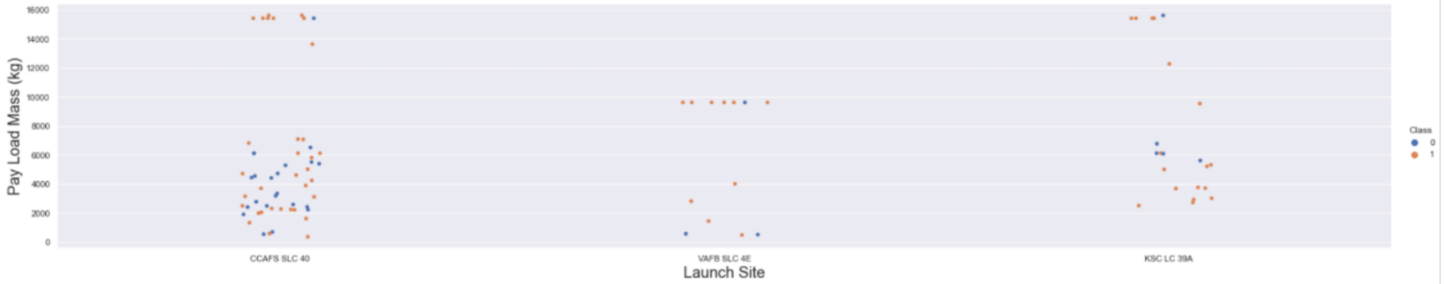
Flight Number vs. Flight Site



The more amount of flights at a launch site the greater the success rate at a launch site.

RESULTS- EDA WITH VISUALIZATION

Payload Mass vs. Launch Site

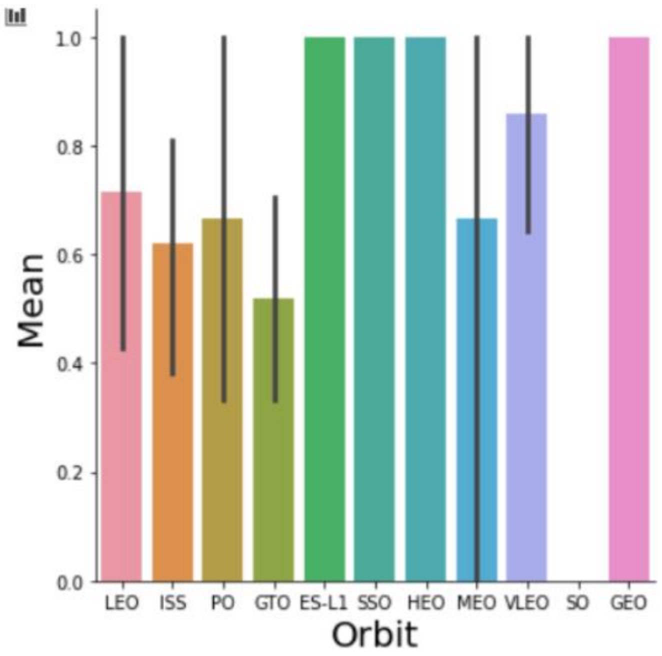


The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket. There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependant on Pay Load Mass for a success launch.

RESULTS- EDA WITH VISUALIZATION

Success rate vs. Orbit type

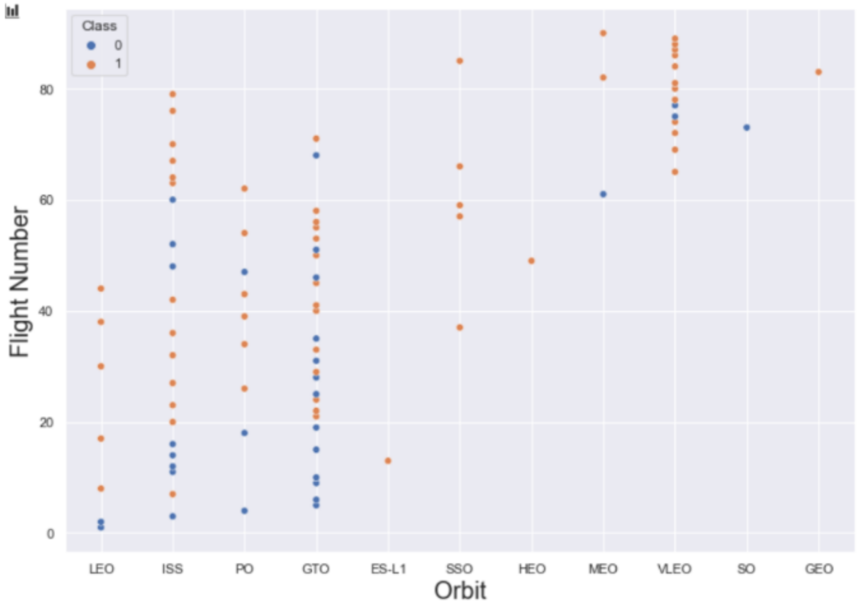
Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate



RESULTS- EDA WITH VISUALIZATION

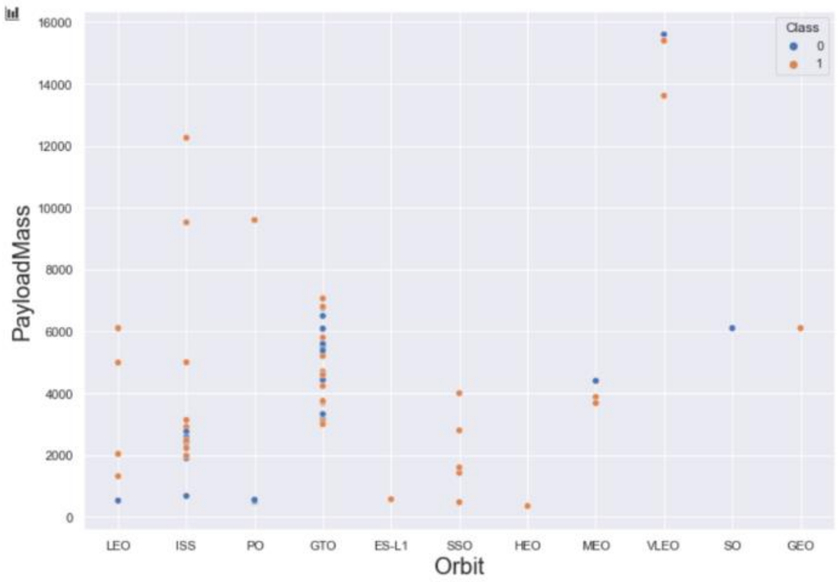
Flight Number vs. Orbit type

You should 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.



RESULTS- EDA WITH VISUALIZATION

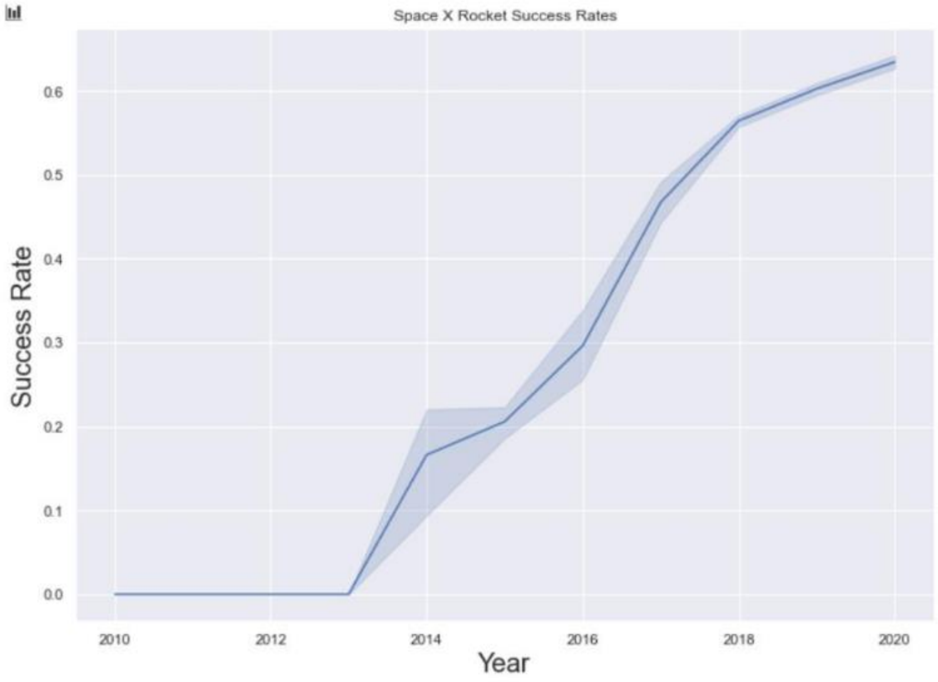
Payload vs. Orbit type



You should observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

RESULTS- EDA WITH VISUALIZATION

Launch success yearly trend



you can observe that the success rate since 2013 kept increasing till 2020

RESULTS- EDA WITH SQL

SQL QUERY

select DISTINCT Launch_Site
from tblSpaceX

Unique Launch Sites
CCAFS LC-40
CCAFS SLC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

QUERY EXPLAINATION

Using the word ***DISTINCT*** in the query means that it will only show Unique values in the ***Launch_Site*** column from ***tblSpaceX***

RESULTS- EDA WITH SQL

SQL QUERY

select TOP 5 * from tblSpaceX
WHERE Launch_Site LIKE 'KSC%'

QUERY EXPLAINATION

Using the word ***TOP 5*** in the query means that it will only show 5 records from ***tblSpaceX*** and ***LIKE*** keyword has a wild card with the words ***'KSC%'*** the percentage in the end suggests that the Launch_Site name must start with KSC.

	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

RESULTS- EDA WITH SQL

SQL QUERY

select SUM(PAYLOAD_MASS_KG_) TotalPayloadMass from tblSpaceX
where Customer = 'NASA (CRS)'",'TotalPayloadMass

	Total Payload Mass
0	45596

QUERY EXPLAINATION

Using the function ***SUM*** summates the total in the column ***PAYLOAD_MASS_KG_***

The ***WHERE*** clause filters the dataset to only perform calculations on ***Customer NASA (CRS)***

RESULTS- EDA WITH SQL

SQL QUERY

select AVG(PAYLOAD_MASS_KG_) AveragePayloadMass from tblSpaceX
where Booster_Version = 'F9 v1.1'

	Average Payload Mass
0	2928

QUERY EXPLAINATION

Using the function ***AVG*** works out the average in the column ***PAYLOAD_MASS_KG_***

The ***WHERE*** clause filters the dataset to only perform calculations on ***Booster_version F9 v1.1***

RESULTS- EDA WITH SQL

SQL QUERY

select MIN(Date) SLO from tblSpaceX where Landing_Outcome = “Success (drone ship)”

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

QUERY EXPLAINATION

Using the function ***MIN*** works out the minimum date in the column ***Date***

The ***WHERE*** clause filters the dataset to only perform calculations on ***Landing_Outcome Success (drone ship)***

RESULTS- EDA WITH SQL

SQL QUERY

select Booster_Version from tblSpaceX where Landing_Outcome = 'Success (ground pad)'
AND Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000

	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

QUERY EXPLAINATION

Selecting only ***Booster_Version***

The ***WHERE*** clause filters the dataset to ***Landing_Outcome = Success (drone ship)***

The ***AND*** clause specifies additional filter conditions ***Payload_MASS_KG_*** > 4000 AND ***Payload_MASS_KG_ < 6000***

RESULTS- EDA WITH SQL

SQL QUERY

```
SELECT(SELECT Count(Mission_Outcome) from tblSpaceX where Mission_Outcome
LIKE '%Success%’) as Successful_Mission_Outcomes,
(SELECT Count(Mission_Outcome) from tblSpaceX where Mission_Outcome
LIKE '%Failure%’) as Failure_Mission_Coutcomes
```

	Successful_Mission_Outcomes	Failure_Mission_Outcomes
0	100	1

QUERY EXPLAINATION

a much harder query I must say, we used subqueries here to produce the results. The ***LIKE ‘%foo%’*** wildcard shows that in the record the ***foo*** phrase is in any part of the string in the records for example.

PHRASE “(Drone Ship was a Success)”
LIKE ‘%Success%’
Word ‘Success’ is in the phrase the filter will include it in the dataset

RESULTS- EDA WITH SQL

SQL QUERY

```
SELECT DISTINCT Booster_Version, MAX(PAYLOAD_MASS
_KG_) AS [Maximum Payload Mass]
FROM tblSpaceX GROUP BY Booster_Version
ORDER BY [Maximum Payload Mass] DESC
```

	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

QUERY EXPLAINATION

Using the word ***DISTINCT*** in the query means that it will only show Unique values in the ***Booster_Version*** column from ***tblSpaceX***
GROUP BY puts the list in order set to a certain condition.
DESC means its arranging the dataset into descending order

RESULTS- EDA WITH SQL

SQL QUERY

SELECT DATENAME(month, DATEADD(month, MONTH(CONVERT(date, Date, 105)), 0) - 1) AS Month, Booster_Version, Launch_Site, Landing_Outcome
FROM tblSpaceX
WHERE (Landing_Outcome LIKE N'%Success%') AND (YEAR(CONVERT(date, Date, 105)) = '2017')

Month	Booster_Version	Launch_Site	Landing_Outcome
January	F9 FT B1029.1	VAFB SLC-4E	Success (drone ship)
February	F9 FT B1031.1	KSC LC-39A	Success (ground pad)
March	F9 FT B1021.2	KSC LC-39A	Success (drone ship)
May	F9 FT B1032.1	KSC LC-39A	Success (ground pad)
June	F9 FT B1035.1	KSC LC-39A	Success (ground pad)
June	F9 FT B1029.2	KSC LC-39A	Success (drone ship)
June	F9 FT B1036.1	VAFB SLC-4E	Success (drone ship)
August	F9 B4 B1039.1	KSC LC-39A	Success (ground pad)
August	F9 FT B1038.1	VAFB SLC-4E	Success (drone ship)
September	F9 B4 B1040.1	KSC LC-39A	Success (ground pad)
October	F9 B4 B1041.1	VAFB SLC-4E	Success (drone ship)
October	F9 FT B1031.2	KSC LC-39A	Success (drone ship)
October	F9 B4 B1042.1	KSC LC-39A	Success (drone ship)
December	F9 FT B1035.2	CCAFS SLC-40	Success (ground pad)

QUERY EXPLAINATION

a much more complex query as I had my ***Date*** fields in SQL Server stored as ***NVARCHAR*** the ***MONTH*** function returns name month. The function ***CONVERT*** converts ***NVARCHAR*** to ***Date***.

WHERE clause filters ***Year*** to be 2017

RESULTS- EDA WITH SQL

SQL QUERY

```
SELECT COUNT(Landing_Outcome)
FROM     tblSpaceX
WHERE  (Landing_Outcome LIKE '%Success%')
AND (Date > '04-06-2010')
AND (Date < '20-03-2017')
```

QUERY EXPLAINATION

Function ***COUNT*** counts records in column
WHERE filters data

LIKE (wildcard)
AND (conditions)
AND (conditions)

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

RESULTS- INTERACTIVE MAP WITH FOLIUM

RESULTS- INTERACTIVE MAP WITH FOLIUM

Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

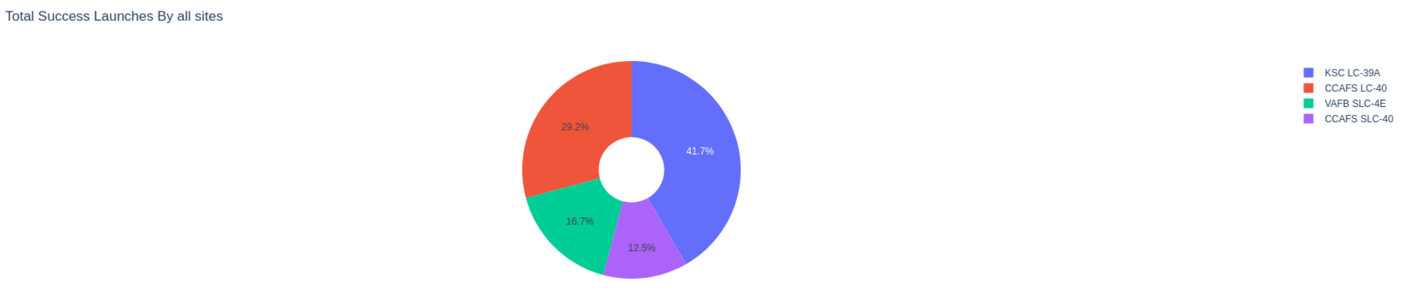
RESULTS- INTERACTIVE MAP WITH FOLIUM

- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

RESULTS- INTERACTIVE MAP WITH FOLIUM

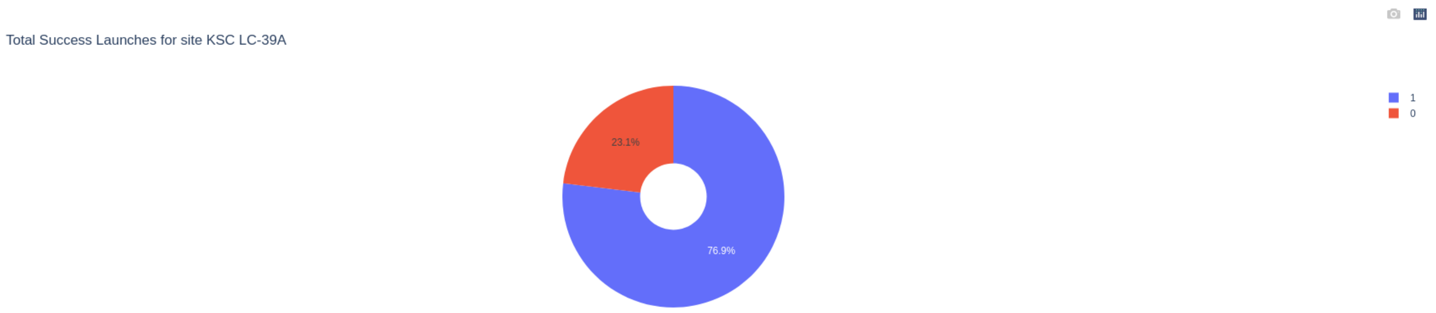
DASHBOARD

https://kathy2027-8050.theiadocker-1-labs-prod-theiak8s-3-tor01.proxy.cognitiveclass.ai/



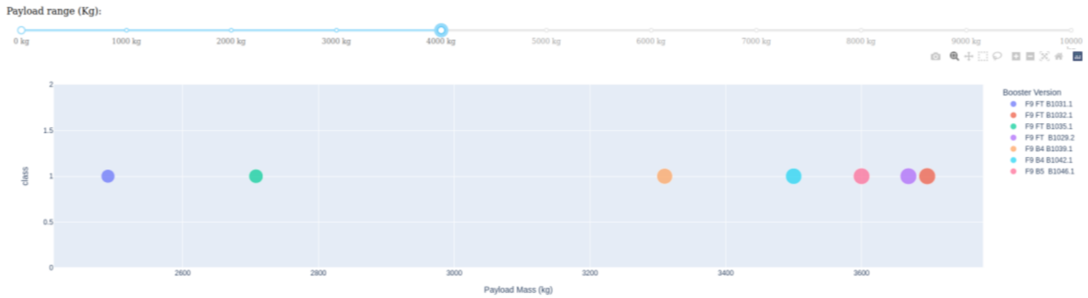
DASHBOARD TAB 1

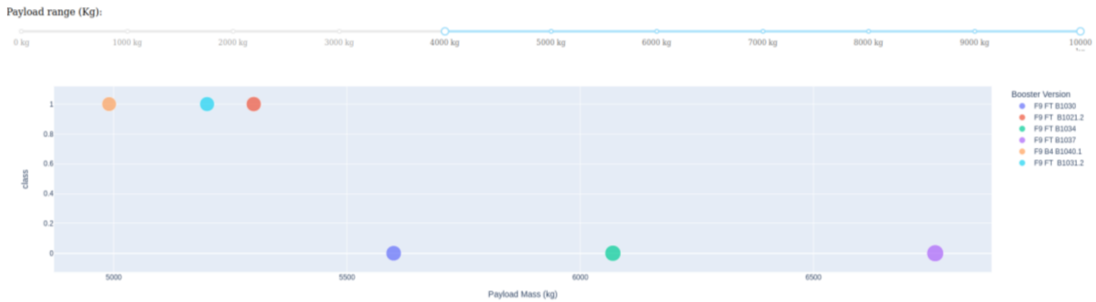
KSC LC-39A had the most successful launches from all the sites.



DASHBOARD TAB 2

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





DASHBOARD TAB 3

The success rates for low weigted payloads is higher.

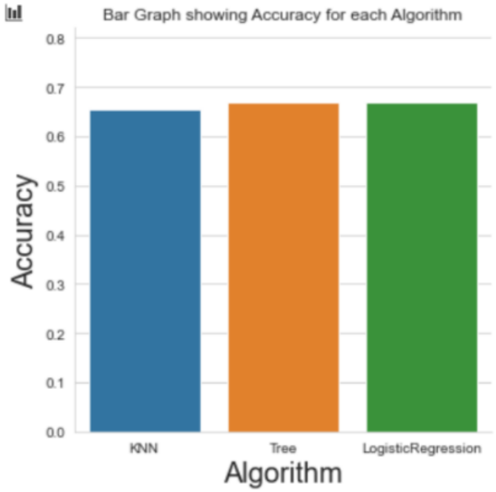
RESULTS- PREDICTIVE ANALYSIS

Classification Accuracy using training data

As you can see our accuracy is extremely close but we do have a winner its down to decimal places! using this function

```
bestalgorithm = max(algorithms, key=algorithms.get)
```

	Accuracy	Algorithm
0	0.653571	KNN
1	0.667857	Tree
2	0.667857	LogisticRegression



The tree algorithm wins!!

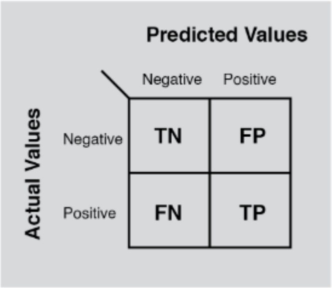
```
Best Algorithm is Tree with a score of 0.6678571428571429
Best Params is : {'criterion': 'gini', 'max_depth': 2, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
```

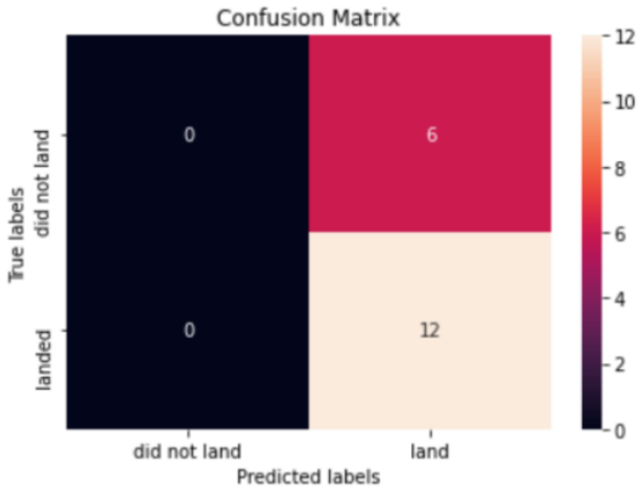
After selecting the best hyperparameters for the decision tree classifier using the validation data, we achieved 83.33% accuracy on the test data.

RESULTS- PREDICTIVE ANALYSIS

Confusion Matrix for the Tree

Examining the confusion matrix, we see that Tree can distinguish between the different classes. We see that the major problem is false positives.





CONCLUSION

- KSC LC-39A had the most successful launches from all the sites.
- Orbit GEO, HEO, SSO, ES-L1 have the best succss rate.
- Low weighed payloads perform better than the heavier one.
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches.
- The Tree Classifier Algorithm is the best for Machine Learning for this dataset.

APPENDIX-HAVERSINE FORMULA

Introduction

The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. Important in navigation, it is a special case of a more general formula in spherical trigonometry, the law of haversines, that relates the sides and angles of spherical triangles.

Usage

Why did I use this formula? First of all, I believe the Earth is round/elliptical. I am not a Flat Earth Believer! Jokes aside when doing Google research for integrating my ADGGoogleMaps API with a Python function to calculate the distance using two distinct sets of {longitudinal, latitudinal} list sets. Haversine was the trigonometric solution to solve my requirements above.

Formula

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi 1 \cdot \cos\varphi 2 \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right)$$

$$c = 2 \cdot \mathrm{atan2}\left(\sqrt{a}, \sqrt{(1-a)}\right)$$

$$d = R \cdot c$$