DEPI

# SPACE_Y PROJECT

# Our Team

Shokri Mohamed

Mohamed Elsayed

Mahmoud Ashraf

Mohammed Waseem

Hossam Elsherbiny

Ibrahim Ali

# Scenario

Fairing
Spacecraft
Second Stage
Payload Attach Fitting
First Stage
Solid Rocket
Motors
Interstage
Fairing
Centerbody Section

# My Role in Data Wrangling

- My primary responsibility in the Space Y project was focused on Data Wrangling.
- Data Wrangling involves cleaning, organizing, and structuring raw data to make it usable for analysis and insights.
- I combined and prepared datasets from various SpaceX sources to ensure all the necessary data was available in a clear and structured format.

# Why Data Wrangling is Critical

- Data Wrangling is a fundamental process because raw data often comes in complex and unstructured formats.
- By cleaning and organizing the data, we ensure that the project has a solid foundation for accurate analysis.
- I ensured that relevant data like rocket types, payload mass, and launch outcomes were easily accessible in one dataset.

# Main and Supporting Datasets

- The core of my work involved processing a main dataset with past SpaceX launches.
  - Supporting datasets included:
    - Rockets dataset for booster details
    - Payload dataset for mass and orbit details
    - Launchpads dataset for site locations

# Merging Data for Better Insights

- I used merge operations to integrate supporting datasets into the main launches dataset.
- New columns like BoosterVersion, LaunchSite, and PayloadMass were added to enrich the data.
- This process made it easier to analyze key factors affecting each launch.

# Working with Nested Data

- Some datasets, like payloads and cores, had nested data (lists inside each launch).
- I used data explosion techniques to convert these lists into rows, making each payload and core analyzable individually.
- This allowed us to compute metrics like total payload mass for each launch.

# Summary of My Contributions

Key Contributions:

- Cleaned and organized data from multiple SpaceX datasets
- Merged essential data points like booster versions, payload mass, and launch sites
- Handled complex data structures to make the data ready for analysis

# Key Processing Steps Overview

### Merging launch site information

This involves combining coordinates and names of launch sites for accurate data representation.

### Processing payload data

In this step, mass and orbit information of the payload is processed to ensure compatibility with the mission.

### Adding core information

Core details including block and serial numbers are integrated into the processing steps.

### Creating landing outcome information

This step focuses on detailing the results of landing operations for future reference and analysis.

### Converting landing pad codes

Landing pad codes are translated into actual names to facilitate better understanding and communication.

# Focus on Falcon 9 Launches

### Filtering only Falcon 9 launches

This step ensures the analysis is concentrated solely on Falcon 9 data, eliminating irrelevant information.

### Fixing date formats

Correcting date formats improves consistency and readability of the launch data.

### Reordering and renaming columns for clarity

This process enhances the organization of the data, making it easier to interpret and analyze.

### Handling missing payload mass data

Addressing gaps in payload mass data is crucial for accurate analysis of launch metrics.

# Final Dataset Overview

Comprehensive Information about Falcon 9 Launches
The end result is a clean dataset with comprehensive information about each Falcon 9 launch.

# what is the percentage of the successful landings

# What is the success rate of each launching site ?

| **LaunchSite** | sum | count | SuccessRate |
|---|---|---|---|
| CCSFS SLC 40 | 34 | 60 | 0.566667 |
| KSC LC 39A | 18 | 23 | 0.782609 |
| VAFB SLC 4E | 10 | 15 | 0.666667 |

# What is the success rate across the years ?

| Date | Class |
|---|---|
| **2010** | **0.000000** |
| **2012** | **0.000000** |
| **2013** | **0.000000** |
| 2014 | 0.333333 |
| 2015 | 0.285714 |
| 2016 | 0.555556 |
| 2017 | 0.833333 |
| 2018 | 0.550000 |
| 2019 | 0.909091 |
| 2020 | 0.850000 |

# What Is the Correlation Between Orbit Type and Mission Success?

# What are the top 10 types of cores that led to successful landings

| Serial | Class |
|---|---|
| B1049 | 6 |
| B1051 | 6 |
| B1048 | 4 |
| B1059 | 4 |
| B1046 | 3 |
| B1056 | 3 |
| B1058 | 3 |
| B1060 | 3 |
| B1021 | 2 |
| B1029 | 2 |

# What are the blocks of these cores ?

this means if we want to launch a core it has to be of block 5

# is there any reusable cores?

# Interactive Visualization

interactive Maps using folium

Dashboard using Dash

# Map using folium

## Objectives

- determine the location of launchsites
- Launch site Success Count
- Calculate Proximities

# Map using folium

## Elements Used

- Map object
- Plugins
  - Marker Cluster
  - Mouse Position
- Features
  - DivIcon

# Map using folium

## Operation

- Creating the Map object using initial location which is taken to be the mean of all sites location
- Creating 3 Circles for each LaunchSite with marker on It in form of text which is the name of the LS
- add a marker Cluster
- add markers for success with green color and ones for failures with red color.

# Map using folium

## Operation

- Create a mouse postion object and add it to the map.
- Create a distance function
- Use mouse locations to determine proximity locations
- Use Distance function to calculate proximity distance

# Does it Work ? !

# DashBoard Using Dash

Objectives

- Display the Success rate of Each site using pie chart.
- Payloadmass Distrbution with Success / Failure acording to Booster version.

# DashBoard using Dash

## Elements

- Dash Library ,HTML ,dcc
- Input / OutPut for Callback
- plotly express for plotting

# DashBoard using Dash

## Operation

- Create a Dash Instant
- make layout
  - title
  - Dropdown box
  - slider
  - Graphs for PieChart and Scatter plot

# DashBoard using Dash

# DashBoard using Dash

# Does it Work ? !

# ML Algorithms

***Logistic Regression***

***Support Vector Machine (SVM)***

***K-Nearest Neighbors (KNN)***

***Decision Tree***

***Random Forest***

# Preprocessing Phase

01 Split the data into features (X) and target (y)

02 Convert categorical features into numerical and use standardization

03 Split the data into train and test

04 Use Grid Search to find the best hyperparameters for each model

| | Flight_Number | BoosterVersion | PayloadMass | Orbit | LaunchSite | Outcome | Flights | GridFins | Reused | Legs | LandingPad | Block | ReusedCount | Serial | Longitude | Latitud |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | Falcon 9 | 5694.158673 | LEO | CCSFS SLC 40 | None - None | 1 | False | False | False | NaN | 1.0 | 0 | B0003 | -80.577366 | 28.56185 |
| 1 | 2 | Falcon 9 | 5694.158673 | LEO | CCSFS SLC 40 | None - None | 1 | False | False | False | NaN | 1.0 | 0 | B0004 | -80.577366 | 28.56185 |
| 2 | 3 | Falcon 9 | 525.000000 | LEO | CCSFS SLC 40 | None - None | 1 | False | False | False | NaN | 1.0 | 0 | B0005 | -80.577366 | 28.56185 |
| 3 | 4 | Falcon 9 | 800.000000 | ISS | CCSFS SLC 40 | None - None | 1 | False | False | False | NaN | 1.0 | 0 | B0006 | -80.577366 | 28.56185 |
| 4 | 5 | Falcon 9 | 677.000000 | ISS | CCSFS SLC 40 | None - None | 1 | False | False | False | NaN | 1.0 | 0 | B0007 | -80.577366 | 28.56185 |

| | Flight_Number | PayloadMass | Flights | GridFins | Reused | Legs | Block | ReusedCount | Longitude | Latitude | ... | B1060 | B1062 | SLC 40 | SLC 4E | LC 39A | JRTI-1 | OCISLY | LZ-1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 14 | 2395.0 | 1 | 1 | 0 | 1 | 1.0 | 0 | -80.577366 | 28.561857 | ... | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 17 | 1898.0 | 1 | 1 | 0 | 1 | 1.0 | 0 | -80.577366 | 28.561857 | ... | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 2 | 19 | 2477.0 | 1 | 1 | 0 | 1 | 1.0 | 0 | -80.577366 | 28.561857 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 3 | 20 | 2034.0 | 1 | 1 | 0 | 1 | 1.0 | 0 | -80.577366 | 28.561857 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 4 | 21 | 553.0 | 1 | 1 | 0 | 1 | 1.0 | 0 | -120.610829 | 34.632093 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

# Logistic Regression

## Best Hyperparameters

C: 0.1
Penalty: L2
Solver: lbfgs

## Metrics

Training accuracy: 0.85
Testing accuracy: 0.86
Precision: 0.86
Recall: 1.00
F1 Score: 0.92

# Support Vector Machine (SVM)

## Best Hyperparameters

C: 0.001
gamma: 0.001
kernel: linear

## Metrics

Training accuracy: 0.85
Testing accuracy: 0.86
Precision: 0.86
Recall: 1.00
F1 Score: 0.92

# K-Nearest Neighbors (KNN)

## Best Hyperparameters

algorithm: auto
n_neighbors: 3
p: 2

## Metrics

Training accuracy: 0.85
Testing accuracy: 0.86
Precision: 0.86
Recall: 1.00
F1 Score: 0.92

# Decision Tree

## Best Hyperparameters

criterion: entropy
max_depth: 6
max_features: sqrt
min_samples_leaf: 2
min_samples_split: 10
splitter: best

## Metrics

Training accuracy: 0.9267
Testing accuracy: 0.93
Precision: 1.00
Recall: 0.92
F1 Score: 0.96

# Random Forest

## Best Hyperparameters

criterion: gini
max_depth: 10
max_features: sqrt
min_samples_leaf: 1
min_samples_split: 2
n_estimators: 100

## Metrics

Training accuracy: 0.81
Testing accuracy: 1.00
Precision: 1.00
Recall: 1.00
F1 Score: 1.00

# ML FLOW & UI

*Connect to the ml flow server*

log model to ml flow

UI using Gradio

- !pip install pyngrok
- !pip install mlflow
- !pip install gradio

## Upload launches_data_ML

- Connect to the mlflow server

```
mlflow.set_experiment("Space_Y_models")
```

```
Space_Y_models
```

# Enter your authtoken

```
print("Enter your authtoken, which can be copied from https://dashboard.ngrok.com/signup")
conf.get_default().auth_token = getpass.getpass()
port=5000
public_url = ngrok.connect(port).public_url
print(f' * ngrok tunnel \"{public_url}\" -> \"http://127.0.0.1:{port}\"')
input("Resume. ?  : ")
```

```
Enter your authtoken, which can be copied from https://dashboard.ngrok.com/signup
```

# connect agent to the server

# log model to ml flow

- Log hyperparameters
- Log metrics
- Log the model

# UI using Gradio

***since our inputs are a lot we can categorize them in 5 categories(tabs)***

- **Basic flight info** : payloadmass , number of flughts
- **Rocket Features** : griffins , legs,reused , reused count
- **Launch and Lading info** : longitude , latitude, launchsize,LandingPad
- **Orbital info** : orbit
- **Booster serial numbe**r : serial

Basic Flight Info | Rocket Features | Launch and Landing Info | Orbital Info | Booster Serial Numbers

Payload Mass (kg)

Number of Flights

Prediction

Submit

Basic Flight Info | Rocket Features | Launch and Landing Info | Orbital Info | Booster Serial Numbers

- [ ] Grid Fins
- [ ] Reused
- [ ] Legs

Block (1-5)

1

Reused Count

0

Prediction

Submit

Basic Flight Info | Rocket Features | Launch and Landing Info | Orbital Info | Booster Serial Numbers

Longitude

.

Latitude

.

Launch Site

CCSFS SLC 40

Landing Pad

JRTI-1

Prediction

Basic Flight Info | Rocket Features | Launch and Landing Info | Orbital Info | Booster Serial Numbers

Orbit

LEO

Prediction

Submit

Basic Flight Info | Rocket Features | Launch and Landing Info | Orbital Info | Booster Serial Numbers

Booster Serial

B0003

Prediction

Submit