SPACEX

PREDICTING FALCON 9 LAUNCHES

A DATA-DRIVEN ANALYSIS OF SPACEX'S LAUNCH

MORGAN RASSATT

8/15/24

IBM DATA SCIENCE PROFESSIONAL CERTIFICATE

INTRODUCTION

WHAT WILL BE DISCOVERED AND WHY IS IT IMPORTANT?

- UTILIZE DATA-DRIVEN METHODS TO PREDICT THE LIKELIHOOD OF A SUCCESSFUL FALCON 9 ROCKET LANDING.
- APPLY MACHINE LEARNING MODELS TO ESTIMATE LAUNCH COSTS, AIDING IN BUDGETING AND COST-SAVING STRATEGIES.
- ANALYZE HOW FACTORS SUCH AS PAYLOAD MASS, ORBIT TYPE, AND LAUNCH LOCATION INFLUENCE THE SUCCESS RATE OF FALCON 9 MISSIONS.

TABLE OF CONTENTS

OUR CONTENT TODAY IS
DIVIDED INTO SIX PARTS.
EACH PART WILL BE DESCRIBED
WITH EXAMPLES.

EXECUTIVE SUMMARY

OBJECTIVE

- OUR PRIMARY OBJECTIVE IS TO ACHIEVE THE HIGHEST POSSIBLE ACCURACY IN PREDICTING THE SUCCESS RATE OF FALCON 9 ROCKET LANDINGS

Findings

- KEY CONTRIBUTORS IN THE SUCCESS OF LANDING INCLUDE: LAUNCH SITE, AND PAYLOAD MASS

Implications

- CHANGES IN THESE VARIABLES WILL SIGNIFICANTLY CHANGE THE LIKELYHOOD OF THE ROCKET LANDING

METHODOLOGY

DATA SOURCES

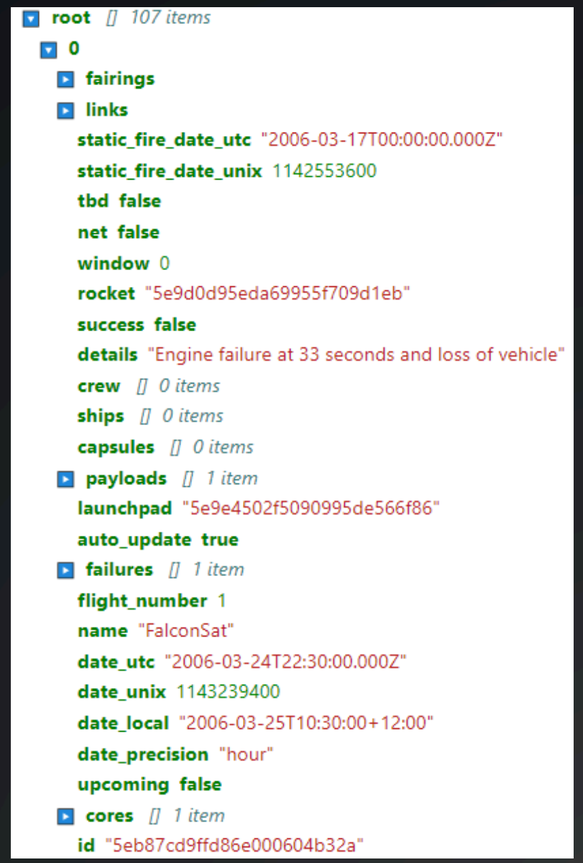
- REQUESTED ROCKET LAUNCH DATA FROM SPACEX API
- USED A RESPONSE OBJECT TO MAKE THE JSON DATA MORE CONSISTENT
- DECODED THE RESPONSE CONTENT AND TURNED IT INTO A PANDAS DATA FRAME
- USED THE IDS FROM THE API TO GET INFO ABOUT THE LAUNCHES, SPECIFICALLY COLUMNS: ROCKET, PAYLOADS, LAUNCHPAD, AND CORES

DATA CLEANING

- FILTERED DATA TO ONLY INCLUDE FALCON 9 LAUNCHES
- DEALT WITH MISSING VALUES BY REPLACING THEM WITH THE MEAN OF THE RESPECTIVE COLUMN
- CREATED A LANDING OUTCOME COLUMN (CLASSIFICATION) BASED ON THE OUTCOME COLUMN

A LOOK AT THE DATA

ORIGINAL DATA IN JSON FORM:



DECODED AND TURNED INTO A PANDAS DATAFRAME:

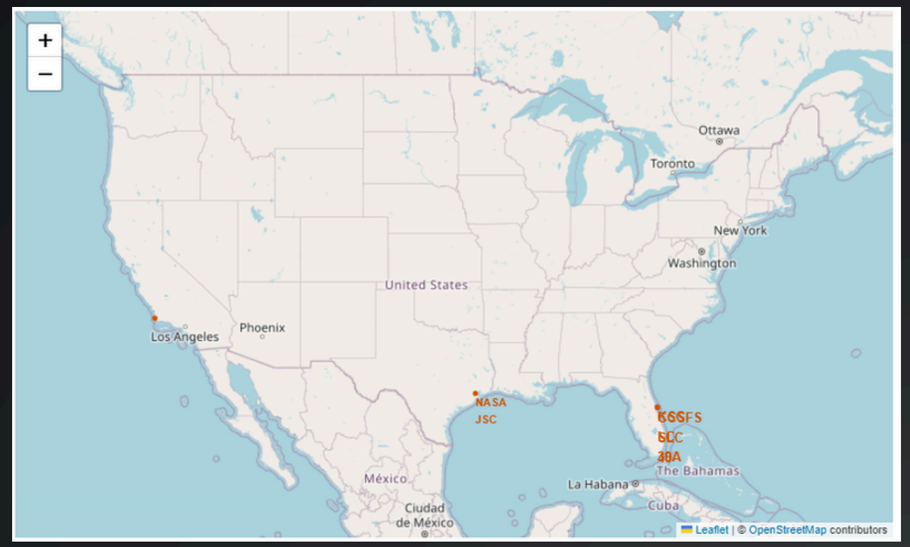
```
Index(['fairings', 'links', 'static_fire_date_utc', 'static_fire_date_unix',
       'tbd', 'net', 'window', 'rocket', 'success', 'details', 'crew', 'ships',
       'capsules', 'payloads', 'launchpad', 'auto_update', 'failures',
       'flight_number', 'name', 'date_utc', 'date_unix', 'date_local',
       'date_precision', 'upcoming', 'cores', 'id'],
      dtype='object')
```

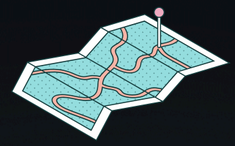
DATA CLEANED AND FILTERED:

```
Index(['FlightNumber', 'Date', 'BoosterVersion', 'PayloadMass', 'Orbit',
       'LaunchSite', 'Outcome', 'Flights', 'GridFins', 'Reused', 'Legs',
       'LandingPad', 'Block', 'ReusedCount', 'Serial', 'Longitude',
       'Latitude'],
      dtype='object')
```

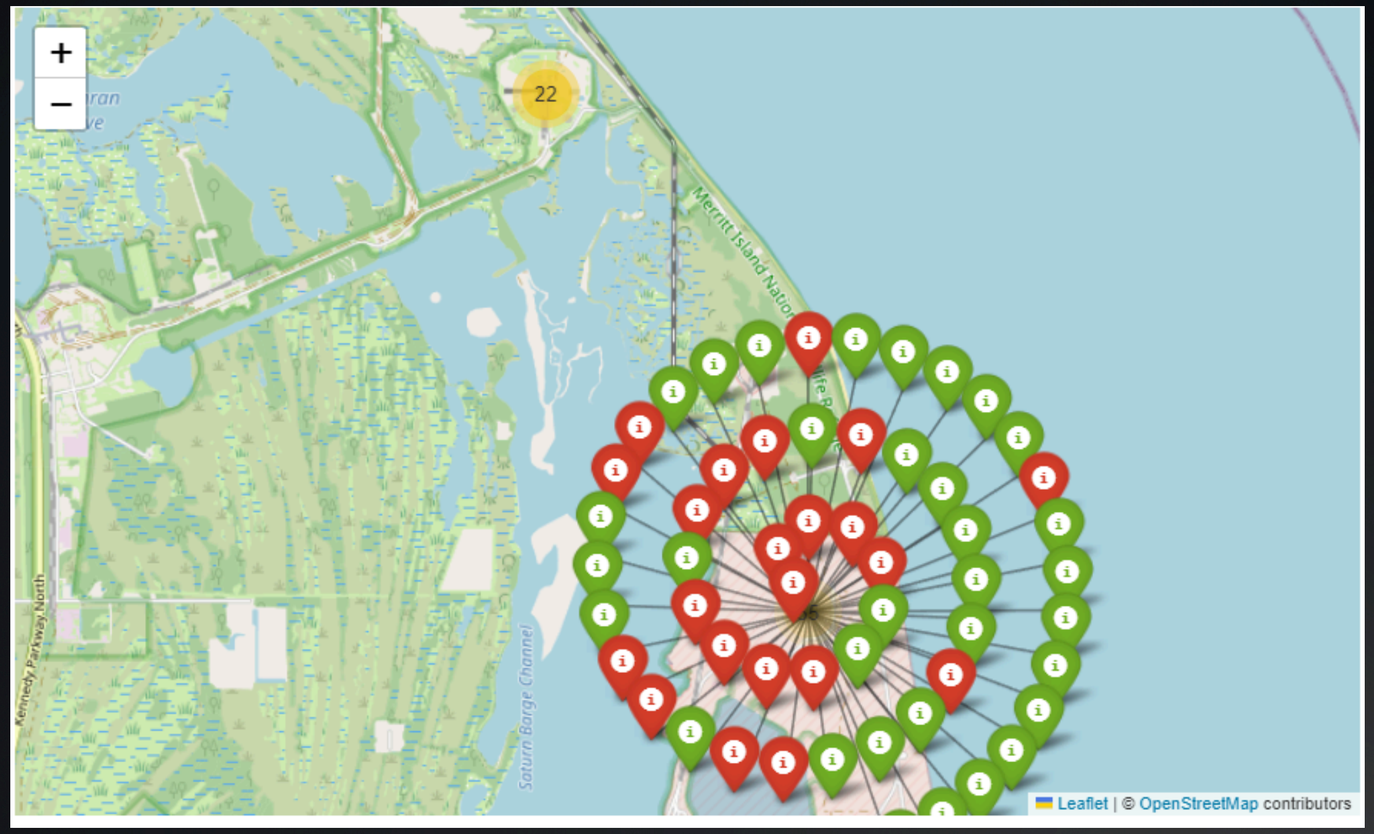
INTERACTIVE MAP

AN INTERACTIVE MAP WAS CREATED WITH FOLIUM TO VISUALIZE THE LAUNCH SITES (*SEE JUPYTER NOTEBOOK TO USE*)





POINTS WERE ADDED AND COLOR CODED TO SHOW THE LOCATION OF SUCCESSFUL AND UNSUCCESSFUL LAUNCHES (*GREEN=SUCCESSFUL, RED=UNSUCCESSFUL*)

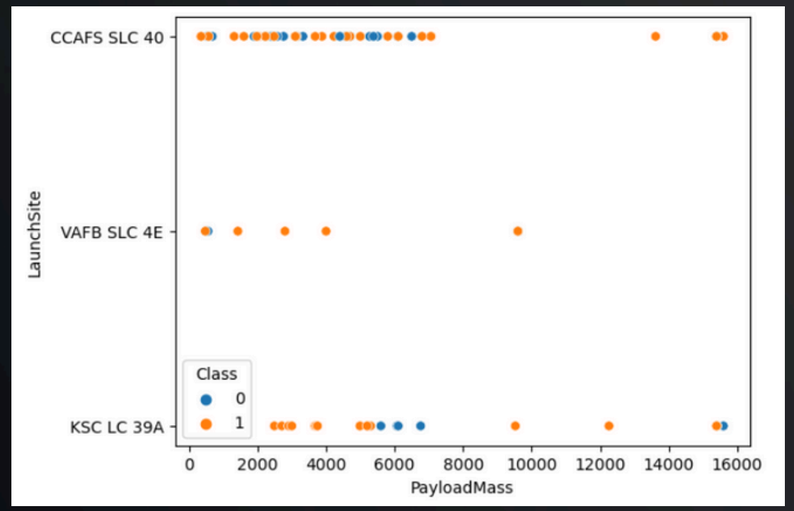


ANALYTICAL METHODS

WE CAN ANALYZE KEY RELATIONSHIPS BETWEEN VARIABLES AND IF THE LANDING WAS SUCCESSFUL BY USING SCATTERPLOTS AND A BAR CHART (1=YES, 0=NO)

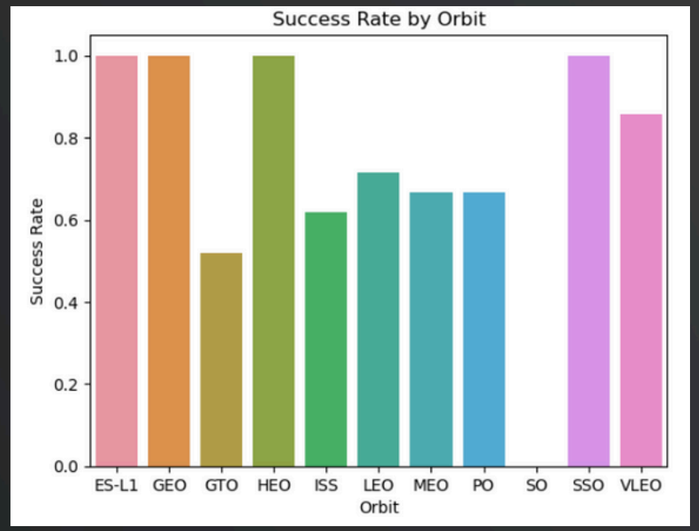
PAYLOAD & LAUNCH SITE

THE LAUNCH SITE DOESN'T HAVE MUCH OF AN IMPACT ON LANDING SUCCESS



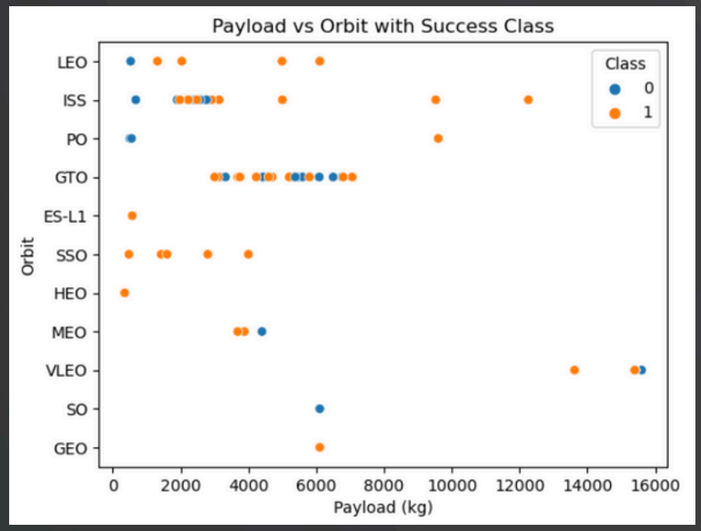
SUCCESS RATE OF EACH ORBIT

ORBITS LIKE ES-L1, GEO, HEO, SSO, AND VLEO HAVE A 100% SUCCESS RATE, WHILE GTO HAS THE LOWEST SUCCESS RATE.



PAYLOAD & ORBIT TYPE

CERTAIN ORBITS ARE MORE CHALLENGING THAN OTHERS, ESPECIALLY WHEN HANDLING DIFFERENT PAYLOAD MASSES

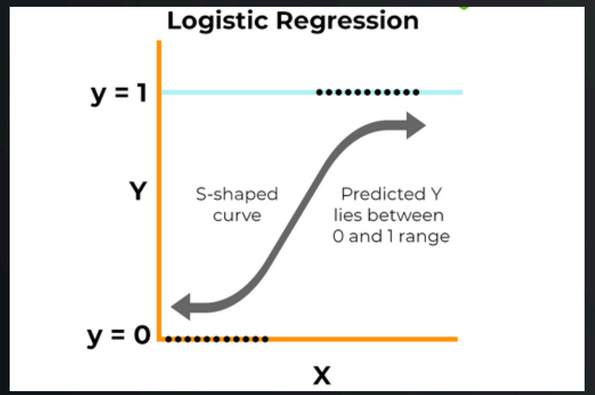


MACHINE LEARNING MODELS

WE CAN USE CLASSIFICATION MACHINE LEARNING MODELS SUCH AS LOG REGRESSION, DECISION TREE, AND KNN TO MAKE PREDICTIONS ON LANDING SUCCESS. WE CAN ASSESS THESE MODELS ACCURACY BY R2 SCORE AND A CONFUSION MATRIX

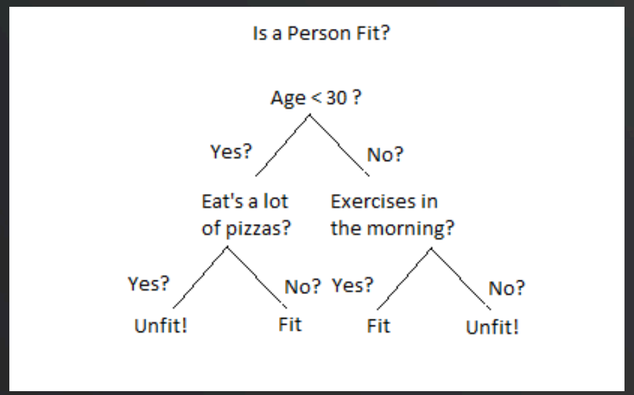
LOGISTIC REGRESSION

WE CAN PREDICT THE PROBABILITY OF A BINARY OUTCOME (YES/NO)



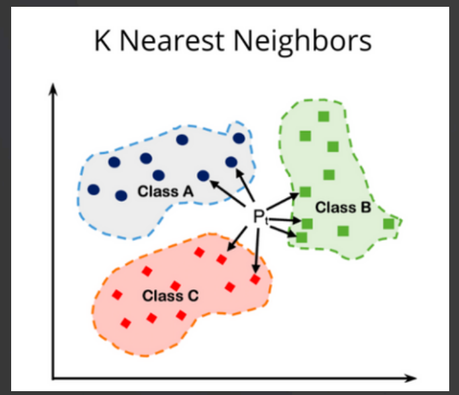
DECISION TREE

USES A TREE-LIKE MODEL OF DECISIONS AND THEIR POSSIBLE OUTCOMES

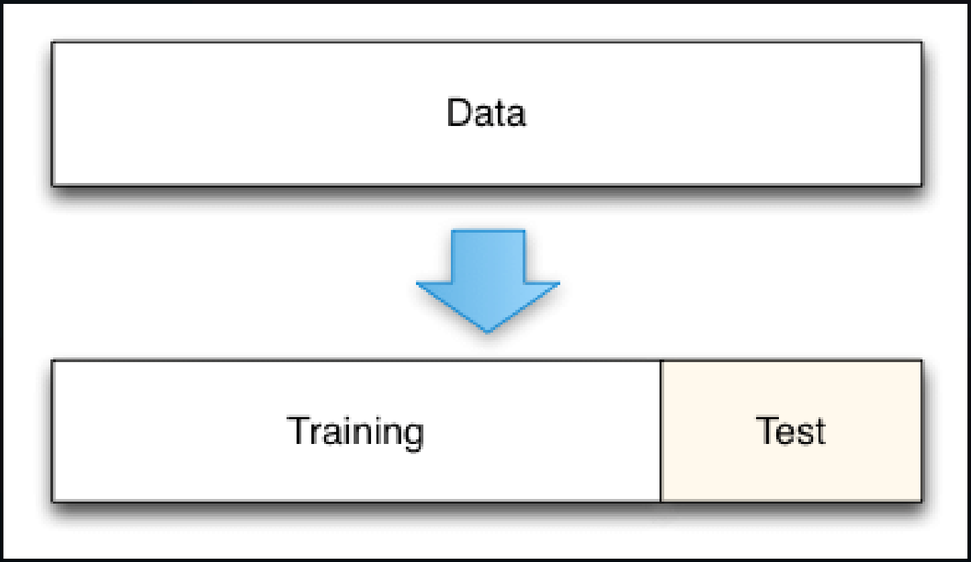


K-NEAREST NEIGHBOR

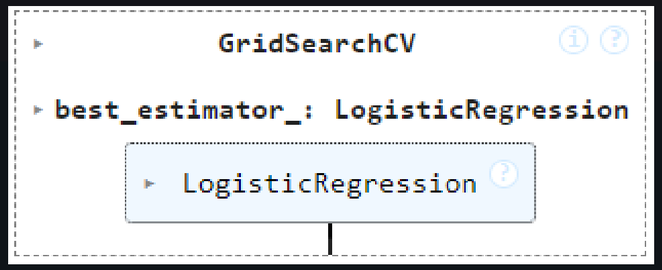
USES THE PROXIMITY OF DATA POINTS ON A GRAPH TO MAKE PREDICTS BASED ON THE GROUPING

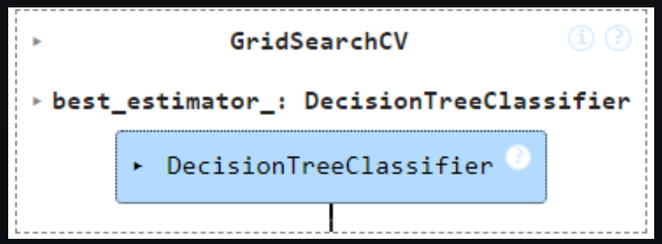


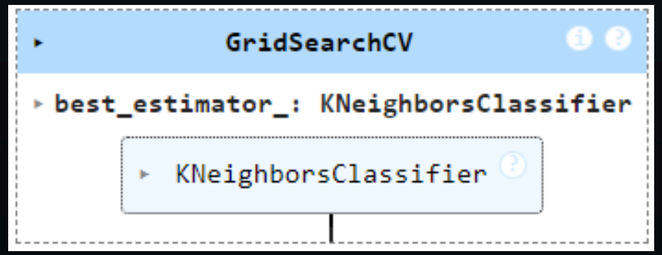
TO TEST OUR MODELS, DATA WILL BE SPLIT INTO TRAINING (80%) AND TESTING (20%) DATASETS



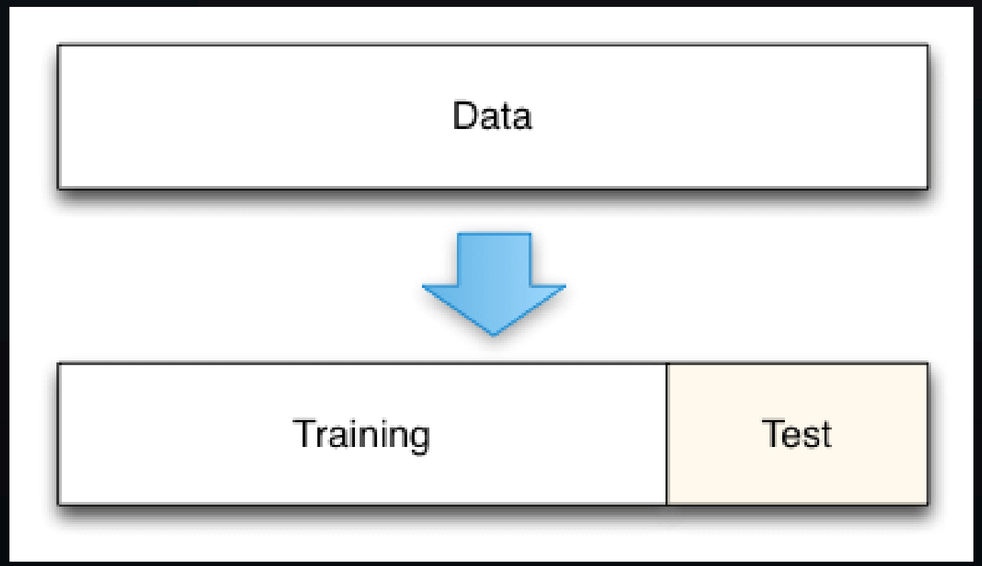
ALL CLASSIFIER OBJECTS WILL BE MADE AND GRID SEARCH WILL BE DONE TO FIND THE BEST PARAMETERS FOR EACH MODEL



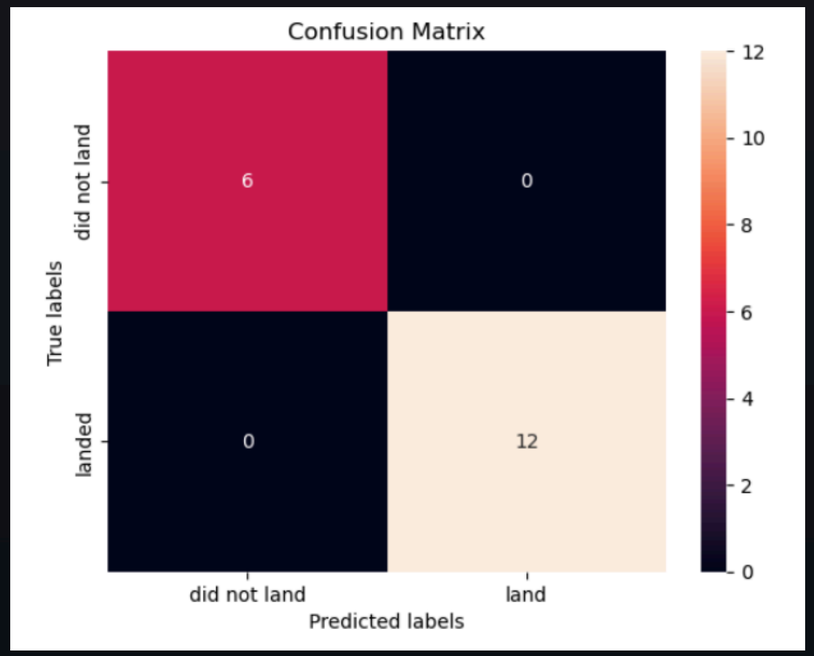




THE MODEL WILL BE EVALUATED ON HOW WELL IT PERFORMS ON THE TESTING DATA



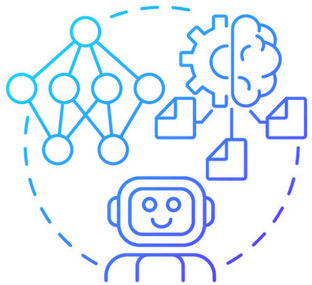
WE CAN USE R2 SCORE AND A CONFUSION MATRIC TO ASSESS HOW WELL THE MODEL MAKES THE CORRECT PREDICITON.



ALL THE MODELS PERFORM WITH 100% ACCURACY ON THE TESTING DATA

APPLY THE MODELS TO ALL THE DATA

DEFINING MODEL



MACHINE LEARNING

WE WILL TEST THE ALL THE MODELS GIVEN THE ENTIRE DATASET AND ASSESS THE ACCURACY WITH THE R2 SCORE

OUTCOME:

LOGISTIC REGRESSION AND K-NEAREST NEIGHBOR BOTH HAD THE HIGHEST R2 SCORE OF 0.6667 WHICH MAKES THEM THE BEST MODEL FOR PREDICTING LANDING SUCCESS

USE CASES:

THESE MODELS CAN HELP IN RISK MANAGEMENT BY HIGHLIGHTING KEY FACTORS THAT INCREASE THE RISK OF LANDING FAILURE. THESE MODELS CAN ALSO BE REFINED TO IMPROVE THE PREDICTIVE POWER

CONCLUSION

- THE BEST MODELS TO USE FOR PREDICTING LANDING SUCCESS ARE LOGISTIC AND KNN MODELS

- THE BIGGEST FACTOR THAT CONTRIBUTE TO LANDING SUCCESS ARE ORBIT, LAUNCH SITE, AND PAYLOAD MASS

- BOTH KNN AND LOGISTIC MODELS TESTED WITH 100% ON THE TESTING DATA. BOTH ALSO HAD AN R2 SCORE OF 0.6667 ON THE ENTIRE DATASET

- THIS MEANS EITHER MODEL WILL BE EFFECTIVE GIVEN THE CURRENT DATASET. IT WILL BE IMPORTANT TO KEEP TESTING THE MODELS AS THE DATASET GETS BIGGER TO FIND THE MODEL THAT BEST SUITS THE DATA

- THE ‘GTO’ ORBIT HAS THE LOWEST SUCCESS RATE, AND ‘ES-L1’ AND ‘SSO’ HAVE PERFECT SUCCESS RATES

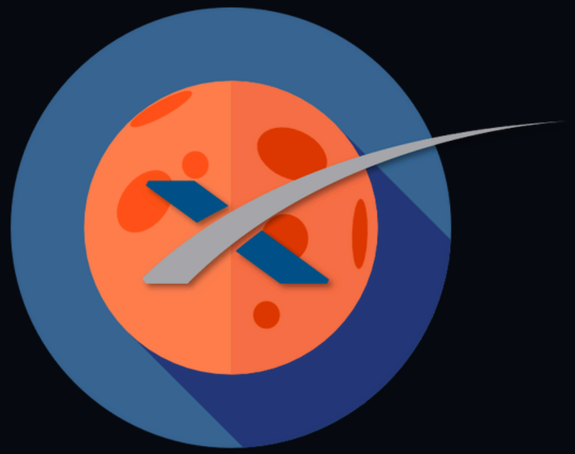
- HIGHER PAYLOAD MASSES TEND TO BE ASSOCAITED WITH SUCCESFUL LANDINGS AT ALL SITES

- SUCCESSFUL PAYLOAD MASSES AND ORBITS ARE THE MAIN TWO FACTORS AND SHOULD BE TAKEN INTO ACCOUNT BEFORE LAUNCH SITE

APPENDIX

ALL DATA WAS PULLED THROUGH THE SPACEX API:

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json



CODE SNIPPITS:

```
# Show a visual of the relationship between success rate of each orbit type
success_rate = df.groupby('Orbit')['landing_class'].mean().reset_index()

# Plot bar chart
sns.barplot(x='Orbit', y='landing_class', data=success_rate)

# Set chart labels
plt.xlabel('Orbit')
plt.ylabel('Success Rate')
plt.title('Success Rate by Orbit')

# Display the plot
plt.show()
```

```
for index, row in launch_sites_df.iterrows():
    coordinate = [row['Latitude'], row['Longitude']]  # Extract Lat and Lon

    # Add a Circle
    circle = folium.Circle(
        location=coordinate,
        radius=1000,  # Radius in meters
        color='#d35400',
        fill=True
    ).add_child(folium.Popup(row['LaunchSite']))

    # Add a Marker with a text label
    marker = folium.map.Marker(
        location=coordinate,
        icon=DivIcon(
            icon_size=(20, 20),
            icon_anchor=(0, 0),
            html='<div style="font-size: 12px; color:#d35400;"><b>%s</b></div>' % row['LaunchSite'],
        )
    )
```

```
# Now we can create a logicstic regression object and a grid search object
# We can fit the object with best parameters using the param dictionary
param ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}
# Create the logistic regression object
logreg = LogisticRegression()

# Set up the GridSearchCV object
logreg_cv = GridSearchCV(logreg, param, cv=10)

# Fit the GridSearchCV object to the data (assuming X_train and Y_train are defined)
logreg_cv.fit(X_train, Y_train)
```