****

**Programming for AI**

**Name : Muhammad Maayer Hassan**

**Subject: PAI (Lab)**

**Submiting to: Sir Rasikh**

**Roll no: su-92-bsaim-f23-099**

# Spaceship Titanic Kaggle Competition - Code Explanation

## 1. Introduction

This document explains the implementation of the Spaceship Titanic machine learning model developed for the Kaggle competition. The goal is to predict whether a passenger was transported to an alternate dimension using personal records from the ship's database.

## 2. Data Loading and Preprocessing

The dataset is read into Pandas DataFrames for easy manipulation. Seaborn and Matplotlib are imported for data visualization. Scikit-learn preprocessing tools are used for handling missing values and encoding categorical features.

Example code:

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
from sklearn.model\_selection import train\_test\_split, GridSearchCV  
from sklearn.preprocessing import StandardScaler, OneHotEncoder  
from sklearn.compose import ColumnTransformer  
from sklearn.pipeline import Pipeline  
from sklearn.impute import SimpleImputer  
from sklearn.ensemble import RandomForestClassifier  
from xgboost import XGBClassifier  
from lightgbm import LGBMClassifier  
from sklearn.metrics import accuracy\_score, classification\_report  
  
# Load dataset  
train\_df = pd.read\_csv("train.csv")  
test\_df = pd.read\_csv("test.csv")

## 3. Exploratory Data Analysis (EDA)

EDA helps understand the data distribution and identify missing values.

Example code:

print(train\_df.info())  
print(train\_df.describe())  
sns.heatmap(train\_df.isnull(), cbar=False)  
plt.show()

## 4. Feature Engineering

Feature engineering involves transforming raw data into meaningful features for the model.

Example code:

train\_df[['Deck', 'Num', 'Side']] = train\_df['Cabin'].str.split('/', expand=True)  
test\_df[['Deck', 'Num', 'Side']] = test\_df['Cabin'].str.split('/', expand=True)  
train\_df[['Group', 'Number']] = train\_df['PassengerId'].str.split('\_', expand=True)  
test\_df[['Group', 'Number']] = test\_df['PassengerId'].str.split('\_', expand=True)

## 5. Data Preprocessing

Handling missing values and encoding categorical data.

Example code:

numeric\_features = ['Age', 'RoomService', 'FoodCourt', 'ShoppingMall', 'Spa', 'VRDeck']  
categorical\_features = ['HomePlanet', 'CryoSleep', 'Destination', 'VIP', 'Deck', 'Side']  
  
numeric\_transformer = Pipeline(steps=[  
 ('imputer', SimpleImputer(strategy='median')),  
 ('scaler', StandardScaler())])  
  
categorical\_transformer = Pipeline(steps=[  
 ('imputer', SimpleImputer(strategy='most\_frequent')),  
 ('onehot', OneHotEncoder(handle\_unknown='ignore'))])  
  
preprocessor = ColumnTransformer(  
 transformers=[  
 ('num', numeric\_transformer, numeric\_features),  
 ('cat', categorical\_transformer, categorical\_features)])

## 6. Model Training and Evaluation

Multiple models are trained and compared.

Example code:

models = {  
 'RandomForest': RandomForestClassifier(random\_state=42),  
 'XGBoost': XGBClassifier(use\_label\_encoder=False, eval\_metric='logloss'),  
 'LightGBM': LGBMClassifier()  
}  
  
X = train\_df.drop(columns=['Transported'])  
y = train\_df['Transported']  
X\_train, X\_val, y\_train, y\_val = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
for name, model in models.items():  
 pipeline = Pipeline(steps=[('preprocessor', preprocessor),  
 ('classifier', model)])  
 pipeline.fit(X\_train, y\_train)  
 y\_pred = pipeline.predict(X\_val)  
 print(f'Model: {name}')  
 print(f'Accuracy: {accuracy\_score(y\_val, y\_pred):.4f}')  
 print(classification\_report(y\_val, y\_pred))

## 7. Final Model Selection and Prediction

The best model is selected and used for predictions.

Example code:

best\_model = XGBClassifier(use\_label\_encoder=False, eval\_metric='logloss')  
best\_pipeline = Pipeline(steps=[('preprocessor', preprocessor),  
 ('classifier', best\_model)])  
best\_pipeline.fit(X, y)  
  
X\_test = test\_df.drop(columns=['PassengerId'])  
test\_df['Transported'] = best\_pipeline.predict(X\_test)  
submission = test\_df[['PassengerId', 'Transported']]  
submission.to\_csv('submission.csv', index=False)

## 8. Conclusion

This notebook provides a step-by-step approach to solving the Spaceship Titanic competition, covering:

- Data preprocessing & feature engineering

- Model selection & evaluation

- Final submission generation