****

***Programming For Artificial Intelligence***

***Assignment - 2***

**Name:**

Ali Maqsood.

**Roll no:**

SU92-BSAIM-F23-050.

**Department:**

Software Engineering Department.

**Program:**

Artificial Intelligence.

**Section:**

BSAI-4A

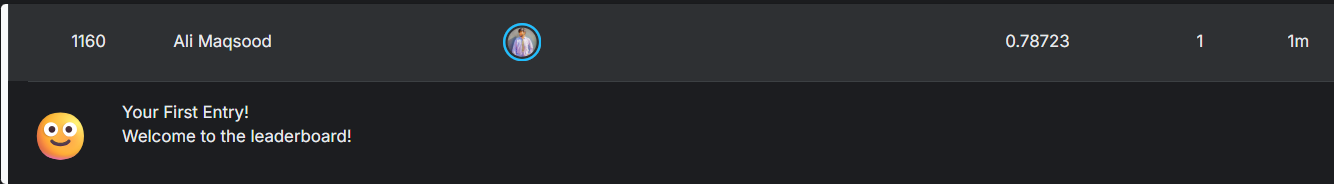
**Question # 1:**

Kaggle Competition: Spaceship Titanic Passenger Problem.

**Documentation:**

This code is solving the **Spaceship Titanic prediction problem** a Kaggle challenge using a Random Forest model. It first loads *train.csv* and *test.csv*, drops the columns that are not needed i.e., *Passenger Id* and *Name*, and handles missing values by filling *Home Planet, Cabin,* and *Destination* with "Unknown". The target column “*Transported*” is separated from the training features. The *Cabin* column is simplified to just its first letter deck as done in the last problem. Then, categorical features (*Home Planet, Cabin, Destination*) are encoded into numbers for further process using Label Encoder. A Random Forest Classifier is trained on the processed training data and used to predict whether passengers in the test set were *Transported*. In the end, the predictions are written to *submission.csv* alongside passenger IDs for identification. This file than is submitted to Kaggle for ranking.

**Ranking:**



**Code:**

import csv

import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

train=pd.read\_csv("train.csv")

test=pd.read\_csv("test.csv")

train.drop(columns=["PassengerId","Name"],inplace=True)

test.drop(columns=["PassengerId","Name"],inplace=True)

# Filling missing values

train["HomePlanet"].fillna("Unknown", inplace=True)

test["HomePlanet"].fillna("Unknown", inplace=True)

train["Cabin"].fillna("Unknown", inplace=True)

test["Cabin"].fillna("Unknown", inplace=True)

train["Destination"].fillna("Unknown", inplace=True)

test["Destination"].fillna("Unknown", inplace=True)

# Dropping output columns

train\_x=train.drop(columns=["Transported"])

train\_y=train["Transported"]

# converting the cabin into deck only

train\_x["Cabin"]=train\_x["Cabin"].str[0]

test["Cabin"]=test["Cabin"].str[0]

# Encoding categorical features

le\_home=LabelEncoder()

le\_cabin=LabelEncoder()

le\_dest=LabelEncoder()

train\_x["HomePlanet"]=le\_home.fit\_transform(train\_x["HomePlanet"])

test["HomePlanet"]=le\_home.transform(test["HomePlanet"])

train\_x["Cabin"]=le\_cabin.fit\_transform(train\_x["Cabin"])

test["Cabin"]=le\_cabin.transform(test["Cabin"])

train\_x["Destination"]=le\_dest.fit\_transform(train\_x["Destination"])

test["Destination"]=le\_dest.transform(test["Destination"])

model=RandomForestClassifier(random\_state=42)

model.fit(train\_x,train\_y)

predictions=model.predict(test)

psid=pd.read\_csv("test.csv")["PassengerId"]

with open("submission.csv","w",newline="") as f:

    writer=csv.writer(f)

    writer.writerow(["PassengerId","Transported"])

    for i in range(len(predictions)):

        writer.writerow([psid[i],predictions[i]])