

**The Superior University, Lahore**



**TASK (Fall 2023)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course Title: | Programming for AI | | | | Course Code: | CAI601410 | Credit Hours: | 1 |
| Instructor: | Sir Rasikh | | | | Program Name: | BSDS | | |
| Semester: | 4th | Batch: | F23 | Section: | BSDSM-4A | Date: | Febrary 24 , 2024 | |
| Time Allowed: |  | | | | Maximum Marks: | |  | |
| Student’s Name: | Rafay Noor | | | | Reg. No. | 024 | | |
| **Task 2: Spaceship Titanic** | | | | | | | | |

**Introduction and Methodology**

The code is to make prediction whether a passenger was transported to another dimension in the Spaceship Titanic dataset. The dataset contains information like home planet, cabin, cryosleep status, age, VIP status, and spending amounts. The goal is to train a machine learning model to predict the **Transported** column.

**Importing Libraries**

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

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

* **pandas** and **numpy** handle data manipulation.
* **matplotlib** and **seaborn** are for data visualization.
* **sklearn** provides tools for machine learning, including data splitting, preprocessing, and classification.

**2. Loading the Data**

The dataset is loaded using pandas.

train\_df = pd.read\_csv('train.csv')

test\_df = pd.read\_csv('test.csv')

* train\_df is used to train the model.
* test\_df contains unseen data to evaluate model performance.

**3. Data Preprocessing**

Handling missing values and encoding categorical features.

train\_df.fillna(train\_df.median(), inplace=True)

test\_df.fillna(test\_df.median(), inplace=True)

Missing numerical values are replaced with the median to avoid bias.

label\_enc = LabelEncoder()

categorical\_columns = ['HomePlanet', 'CryoSleep', 'Cabin', 'Destination', 'VIP']

for col in categorical\_columns:

    train\_df[col] = label\_enc.fit\_transform(train\_df[col].astype(str))

    test\_df[col] = label\_enc.transform(test\_df[col].astype(str))

* Categorical columns (like HomePlanet and Destination) are converted into numerical values using LabelEncoder.

**4. Splitting Data into Training and Testing Sets**

X = train\_df.drop(columns=['PassengerId', 'Name', 'Transported'])

y = train\_df['Transported']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

* Drops non-relevant columns (PassengerId, Name).
* Splits 80% of the data for training and 20% for testing.

**5. Model Training (Random Forest Classifier)**

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

* Uses **RandomForestClassifier**, an ensemble learning method that improves accuracy by combining multiple decision trees.
* The model is trained on X\_train and y\_train.

**7. Making Predictions on Test Data**

test\_predictions = model.predict(test\_df.drop(columns=['PassengerId', 'Name']))

submission = pd.DataFrame({'PassengerId': test\_df['PassengerId'], 'Transported': test\_predictions})

submission.to\_csv('submission.csv', index=False)

* Predicts **Transported** values for test data.
* Saves predictions in **submission.csv** for submission.

**Conclusion**

* The model is built using **Random Forest**, a powerful ensemble learning method.
* **Label encoding** is used for categorical data.
* The final predictions are saved in a CSV file for competition submission.