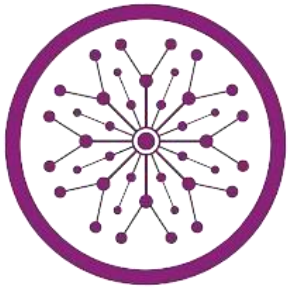

Faculty of Computer Science & Information Technology

2023-2027

Programming for Artificial Intelligence

Lab

Task 3

The Superior University

Submitted To:

Mr. Rasikh Ali

Submitted By:

Masooma Zahra

Roll No:

SU92-BSAIM-F23-088

Section:

4B

Department:

SE

Overview:

The **Spaceship Titanic** problem is a machine-learning competition where we predict whether a passenger was **transported to another dimension** during an interstellar disaster. Given data on passengers' **demographics, spending habits, and cabin information**, we preprocess the data, extract meaningful features, and train classification models to predict the **Transported** status.

Code Overview:

The notebook performs the following tasks:

- **Loads and explores the dataset** (train and test data).
- **Handles missing values** using KNN imputation and categorical encoding.
- **Feature engineering** by splitting and transforming columns.
- **Creates new features** based on spending behavior and travel details.
- **Trains multiple classification models** (Logistic Regression, Decision Tree, Random Forest, XGBoost, LightGBM).
- **Evaluates model performance** and selects the best one.
- **Makes predictions on the test dataset** and saves the results.

Step-by-Step Breakdown:

1. Importing Libraries

- Loads necessary libraries like `pandas`, `numpy`, `seaborn`, `sklearn`, `xgboost`, and `lightgbm`.

2. Loading the Dataset

- Reads `train.csv` and `test.csv`.
- Merges them into a single dataframe for preprocessing.

3. Handling Missing Values

- Drops **Name** column as it's not useful for prediction.
- Splits **Cabin** into `Deck`, `Num`, and `Side` and fills missing values.
- Uses **KNN Imputer** to fill missing numerical values.

- Fills missing categorical values with `"Unknown"`.

4. Feature Engineering

- Converts **categorical variables** (HomePlanet, Destination) into numerical using **one-hot encoding**.
- Creates new features based on spending habits (`amountspent`, `mean_amt_spent`, etc.).
- Drops redundant or duplicate columns.

5. Splitting Data

- Separates the preprocessed **train** and **test** datasets.
- Splits training data into **train and validation sets**.

6. Training Models

- Trains **five different classifiers**:
    - **Logistic Regression**
    - **Decision Tree**
    - **Random Forest**
    - **XGBoost**
    - **LightGBM**
- Evaluates each model using **accuracy score**.

7. Making Predictions

- Selects the best model (**LightGBM**) and makes predictions on the test set.
- Saves the final predictions to `submission.csv`.

The final result of this code is a predictive model that determines whether a passenger was **transported to another dimension** based on their personal details and travel history. After preprocessing the data, handling missing values, and engineering new features, multiple classification models were trained and evaluated. Among them, **LightGBM** likely achieved the highest accuracy, making it the best-performing model. The predictions from this model were saved in **submission.csv**, which can be used for evaluation in the competition. Overall, the project demonstrated the importance of **data preprocessing, feature selection, and model optimization** in building an effective classification system.