Lab 2

Task: Kaggle Competition: Spaceship Titanic

Here's a step-by-step explanation of the code:

- **Import necessary libraries**:
  - Libraries such as `pandas`, `numpy`, `matplotlib`, `seaborn`, and various scikit-learn components are imported to handle data manipulation, visualization, model building, and evaluation.
- **Load the datasets**:
  - `train_df` and `test_df` are loaded from CSV files, representing the training and test datasets, respectively.
- **Data Exploration**:
  - The `head()`, `info()`, and `describe()` functions are used to get a quick look at the data, check for any missing data, and understand basic statistics like mean, median, etc.
- **Check for missing values**:
  - The `isnull().sum()` method checks if there are any missing (null) values in the columns of the training dataset.
- **Drop unnecessary columns**:
  - Unnecessary columns such as `PassengerId`, `Name`, and `Ticket` are dropped from both the training and test datasets, as they won't contribute to model prediction.
- **Handle missing values**:
  - Missing `Age` values are filled with the median of the column using `fillna()`.
  - Missing values for `Embarked` are filled with the most frequent value (mode) using `fillna()` as well.
- **One-Hot Encoding for categorical variables**:
  - The categorical variables `Pclass`, `Sex`, and `Embarked` are transformed into binary columns using one-hot encoding (via `pd.get_dummies()`), with `drop_first=True` to avoid multicollinearity.
- **Split the data into features and target**:
  - Features (X) are all the columns except `Transported`, while the target variable (y) is the `Transported` column.
- **Split the dataset into training and validation sets**:
  - The dataset is split into training (80%) and validation (20%) sets using `train_test_split()`.
- **Feature Scaling (Optional)**:
  - Although RandomForest doesn't require feature scaling, `StandardScaler` is used to scale the features to a similar range (for consistency and future use with other models).
- **Initialize and train the model**:
  - A `RandomForestClassifier` model is created with 100 trees and trained on the training set (`X_train` and `y_train`).
- **Make predictions on the validation set**:

    - The trained model is used to predict the target variable (`Transported`) on the validation set (`X_valid`).
- **Evaluate the model**:
    - The accuracy of the model is calculated using `accuracy_score()`.
    - A confusion matrix is generated to evaluate how well the model predicted the correct classes.
    - A classification report is printed to provide additional metrics like precision, recall, and F1-score.
- **Preprocess the test dataset**:
    - The test dataset is preprocessed in the same way as the training dataset, including handling missing values and performing one-hot encoding for categorical columns.
- **Ensure consistency between train and test data**:
    - The test dataset is transformed to ensure that it has the same columns as the training data (important when predicting).
- **Make predictions on the test dataset**:
    - The trained model is used to make predictions on the test dataset (`X_test`).
- **Prepare the submission file**:
    - A DataFrame is created with `PassengerId` and the predicted `Transported` values from the test predictions.
- **Save predictions to a CSV file**:
    - The predictions are saved to a CSV file (`submission.csv`) for submission.

OUTPUT