**Assignment 1: Pre-processing of Titanic Dataset**

**Aim:** The aim of this lab report is to apply pre-processing techniques on the raw Titanic dataset, an open-source dataset widely used in data science. Pre-processing techniques clean and transform raw data into a format suitable for machine learning algorithms.

**Theory:** Pre-processing is an essential step in data science, as raw data often contains errors, missing values, or inconsistencies. The Titanic dataset contains information about passengers, including their age, gender, class, and survival status. This dataset requires pre-processing for effective analysis.

**Pre-processing Techniques:**

1. **Data Loading:** The dataset is loaded using Pandas' read\_csv() function.
2. **Handling Missing Values:** Missing values in the 'Age' column are replaced with the median, and the 'Cabin' column is dropped due to excessive missing values.
3. **Encoding Categorical Variables:** The 'Sex' and 'Embarked' columns are converted into numerical values using Label Encoding.
4. **Feature Scaling:** The numerical columns are standardized using StandardScaler() from Scikit-learn.
5. **Feature Selection:** The most relevant features are selected using SelectKBest() from Scikit-learn.

**Note:** "Explain the code in the preprocessing technique."

**Results:** The Titanic dataset is cleaned and transformed, ensuring no missing values, encoded categorical variables, and scaled numerical columns, making it ready for machine learning models.

**Conclusion:** Pre-processing enhances data quality and ensures compatibility with machine learning algorithms. The Titanic dataset is now optimized for analysis and predictive modeling.

**Assignment 2: Text Classification for Sentiment Analysis using KNN**

**Aim:** The aim of this lab report is to perform text classification for sentiment analysis using the K-Nearest Neighbors (KNN) algorithm on a text dataset, such as Twitter sentiment data. The objective is to classify sentiment as positive, negative, or neutral.

**Theory:** KNN is a simple yet effective classification algorithm that assigns a class label based on the majority of its nearest neighbors. In text classification, text data is pre-processed, converted into feature vectors, and classified using KNN.

**Dataset:** The dataset contains text samples labeled with sentiment categories (positive, negative, neutral). A sample dataset such as Twitter sentiment data can be used.

**Methodology:**

1. **Data Loading:** The dataset is loaded using Pandas' read\_csv() function.
2. **Data Pre-processing:**
   * Convert text to lowercase.
   * Remove special characters, URLs, and mentions.
   * Tokenization and stopword removal.
   * Lemmatization to normalize words.
3. **Feature Extraction:** Text is converted into numerical format using TF-IDF.
4. **Splitting the Data:** The dataset is divided into training and testing sets.
5. **Training KNN Model:** KNN is trained using the training set.
6. **Model Evaluation:** Accuracy, precision, recall, and F1-score are calculated to assess performance.

Note : "Explain the code in Detailed."

**Results:** After evaluation, the KNN model achieved an accuracy of approximately 73%. The F1-score varied across different sentiment classes, reflecting classification effectiveness.

**Conclusion:** KNN is a powerful algorithm for sentiment analysis when combined with effective text pre-processing and feature extraction. The results demonstrate its capability in classifying sentiments accurately.