CSCS 460 – Machine Learning

Assignment 1 (Task 2) Deadline: To be submitted along with Task 3

- The aim of this assignment is to get familiar with machine learning toolkits.
- Submit source code (when Task 3 will be given). Don't submit snapshots of the code as the code files will be used for plagiarism check.
- Install Python Environment for your OS. Use Anaconda as it's the most convenient to setup and scikit-learn is preinstalled.
- Alternatively, you can use Google colab that has preinstalled scikit-learn.

1) **Text Preprocessing:** You are given two files. "train.csv" and "test.csv". Both files have two columns "review" and "sentiment". The first column is the input text while the second column is the label. You need to perform the following preprocessing steps on both files.
    a. Read both csv files. You can use "pandas" or "polars" libraries to read it as dataframe in Python.
    b. For each review, remove all **
** tokens. You can use "re" library in Python that uses regular expressions or you can use any other library of your choice that allow string removl/replacement.
    c. Change labels from text to integers (0 for negative and 1 for positive). You can do it directly in pandas dataframe or use scikit-learn label encoder.
    d. Separate out labels and texts. You should have train texts, train labels, test texts, and test labels in separate variables.
    e. For the preprocessed texts above, convert them to vectors based on frequency (counts) using scikit-learn. Do this for both train and test splits. Remember, you can only use train texts to determine the vocabulary of the dataset. **Hint:** You will have one vectorizer that has seen training data and will be used to transform the testing data.
2) **Feature Importance:**
    a. For the preprocessed texts above, convert them to vectors based on binary occurrences of words using scikit-learn. Do this for both train and test splits. Save these vector representations in two variables.
    b. Use information gain to calculate feature importance (words importance to be specific) for train split only. Sort the words based on the information gain values in descending order and output top 10 (most important) features (words). Output last 10 (least important) features (words). You can use scikit-learn mutual_info_classif method to calculate feature importance or write your own function to calculate information gain.