DMML:

Team:

- Aritra Banerjee (MDS-2018-28)
- Aishwarya R (MDS-2018-08)

Packages Used:

- mlxtend
- sklearn
- BeautifulSoup
- numpy
- Time
- Warnings
- Pandas

Documentation:

- sklearn :

➔ Naive bayes - MultinomialNB :: Used classifier to classify the data

➔ OneVsRestClassifier :: To handle multiple labeled data.

➔ TfidfVectorizer:: Vectorize the string using " Term Frequency" and " Inverse-Document Frequency ".

➔ f1_score, precision_score, recall_score, accuracy_score :: Metrics to evaluate the model.

- Time : time.clock() function in the package "time" gives us two timestamps for start and end times which when subtracted gives us the time spent on running the code.

- Warnings : Warning messages are used to alert the user of some condition in a program, where that condition (normally) doesn't warrant raising an exception and terminating the program.

- Pandas : Pandas is an open source library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

- Mlxtend: Used TransactionEncoder under mlxtend.preprocessing to create sparse matrix.

- BeautifulSoup :: Used to parse .sgm file and taking inputs.

Code Description:

This code is written typically for the reuters-21578 in the UCI machine learning library. It will not run for files having some format.

The code is written in certain parts:

- The first part of the code is loading the packages and opening file f to write output.

- Next data in imported using “BeautifulSoup” as Modified Apte ("ModApte") Split mentioned as documentation. A clock is set to measure the time taken.

- After the above step we have a train data train category test data and test category. This step contains removing duplicates from train data and moving some file from test to train which have such category which never occurred in train.

- Using some regex function to remove numerical values and unnecessary string from both train and test.

- Starting the clock and using “transactionencoder” to create a sparse matrix of the the train label and test label. Vectorizing the train body using TF-IDF with max_features = 5000.

- Building the model MultinomialNB using alpha = 0.01 and the model is getting fitted with the training data.

- Predicting labels based on the test data and stopping the time.

- Evaluating the model using precision, F1 score and accuracy.
- Mapping predicted values to the categories and zipping original label and predicted label.
- Printing all required details and closing the file.