**Deliverable 2**

260860011 Takuya Ishii

1. **Project idea** In this project, I will attempt to create a text sentiment predictor that takes a phrase as input and returns its sentiment on a scale of 0 to 4 based on its positivity.
2. **Data Preprocessing**

Dataset (also stated in Deliverable 1)

Kaggle (<https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews>)

The dataset consists of train and test table. The following 3 columns are present in both tables: PhraseId, SentenceId, and Phrase. Only the train table contains Sentiment label column. For each phrase, sentiment has been measured on a scale of 0 to 4 with 0 being most negative and 4 being most positive.

Since the test table doesn’t have Sentiment label column, I’ll be only using the train table.

Graphical user interface, application, Teams

Description automatically generated

1. Reducing the size of the table  
   The train table contains 156060 rows. Each review is divided into different phrases and they are stored in respective rows. By extracting the top row of the series of rows that shares the same SentenceId, I only preserved rows with the full review. After this operation, the table now contains 8529 rows. Table

   Description automatically generated
2. Tokenization (including cleaning and lemmatization)  
   Next step is to tokenize a review in each row. After each review becomes a list of words, I removed non-alphabetic characters and lemmatized each word in the list. Insert this list as “vocabulary” column in the table.  
    Table

   Description automatically generated
3. Create binary-bag-of-word expression  
   By checking the presence of 10000 most common words in the entire dataset (8529 rows) in each tokenized review, create “BoW” column. Note that the list of common words is also is also cleaned and lemmatized.  
   A picture containing table

   Description automatically generated
4. **Machine Learning Model**

Since this project deals with classification problem, I chose Naïve Bayes and Logistic Regression as my candidates. The training was performed using scikit learn. Training, validation, test splits were done in the ratio of roughly 7 : 1.5 : 1.5, which seem to be common in many ML training.

1. Multinomial Naïve Bayes

For this model, I could tune the smoothing parameter alpha (see codes provided) but after multiple testing using validation set, the default setting alpha=1.0 seem to perform well.

1. Logistic Regression  
   For optimization, I chose ‘sag’(stochastic average gradient) as it’s compatible with multiclass problems. It seems that regularization is applied automatically by default.
2. **Preliminary Result**

For each model (Multinomial Naïve Bayes and Logistic Regression), accuracy score, mean squared error and confusion matrix are examined. Both models are showing very similar results but Naïve Bayes is performing slightly better. While accuracy scores aren’t very high for both models, we observe that the coloured cells are distributed diagonally, which means our models are somewhat working.

Calendar

Description automatically generatedA picture containing calendar

Description automatically generated

**Logistic Regression**

**Multinomial Naïve Bayes**

1. **Next Step**

My preliminary attempt didn’t yield very competitive accuracy score. This could be due to the relatively small size of the dataset. A possible improvement can be using the dataset of original size (156060 rows) to train the model. Reducing the size of the table in preprocessing step might not have been necessary…