# Training Methodology

# Data Processing:

First, the required libraries were imported.

The train dataset is loaded and the number of unique words in the text and aspect columns are counted.

This is used as the num\_words argument in the Tokenizer for each.

A label encoder was also created for deep learning model.

The text and aspects were tokenized to dataframes and concatenated.

The class was used as is for the ML models while it was converted to a matrix using the to\_categorical.

The train and validation splits for the ML models were obtained using train test split function.

While, the deep learning model had it inbuilt argument for validation split.

# Model Training:

A basic model with categorical crossentropy loss and adam optimizer is created and run for several epochs.

Alongside deep learning, linear regression, knn, decision tree, svc models were used to create several models.

In the train.py file in the src folder, the model selection has been facilitated with command line arguments -m with parameters dl for deep learning, lr for logistic regression, knn for k nearest neighbours, dt for decision tree classifier and svc for support vector classifier.

For each model, the precision, f1-score, recall are calculated for each class of the validation data created from the train test split function.

**Highest Score Model:**

It is observed that SVC had the highest score among the models with accuracy 0.725.

# Training Script Arguments:

For training script, the arguments are -m MODEL\_TYPE, -o OUTPUT\_NAME -e EPOCHS (for deep learning model)

The -o argument uniquely identifies a model and hence should be different for different runs of script for saving old models and encoders.