Sentiment Analysis on IMDB Dataset

1. Task: Sentiment Analysis on IMDB Dataset

2. Dataset used: IMDB dataset of 50k movie reviews

The dataset consists of 50k movie reviews which can be used for binary sentiment classification. Each review is accompanied by the sentiment associated with it, i.e. *"positive or negative"*.

3. Data Preprocessing:

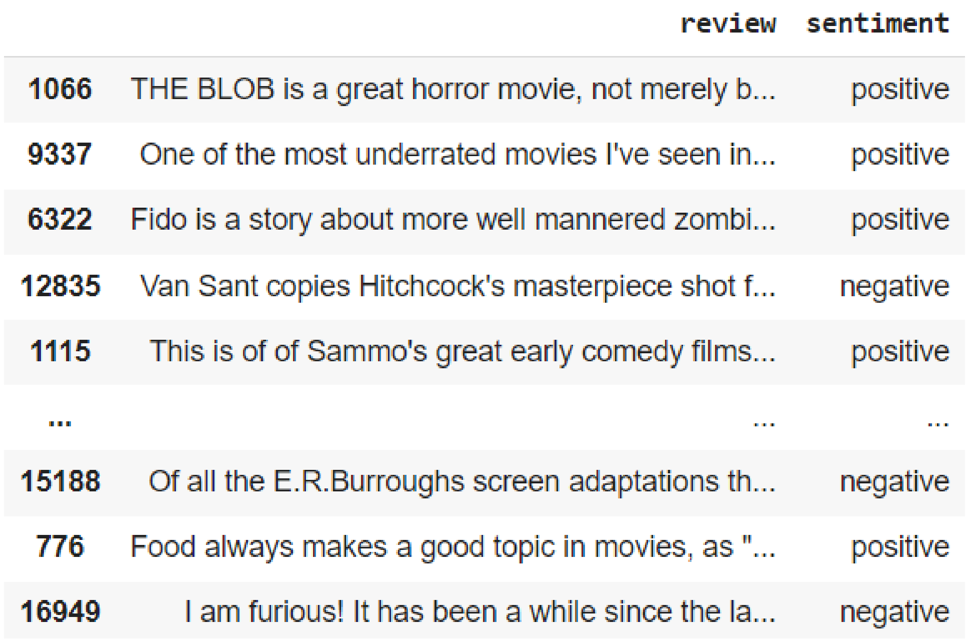
Steps Involved:

3.1.Converted the CSV into a dataframe with column "***review"*** containing the reviews given by the customers and "***Sentiment"*** assigned by them.

3.2. Removed HTML tags containing html and css markup from the text column.

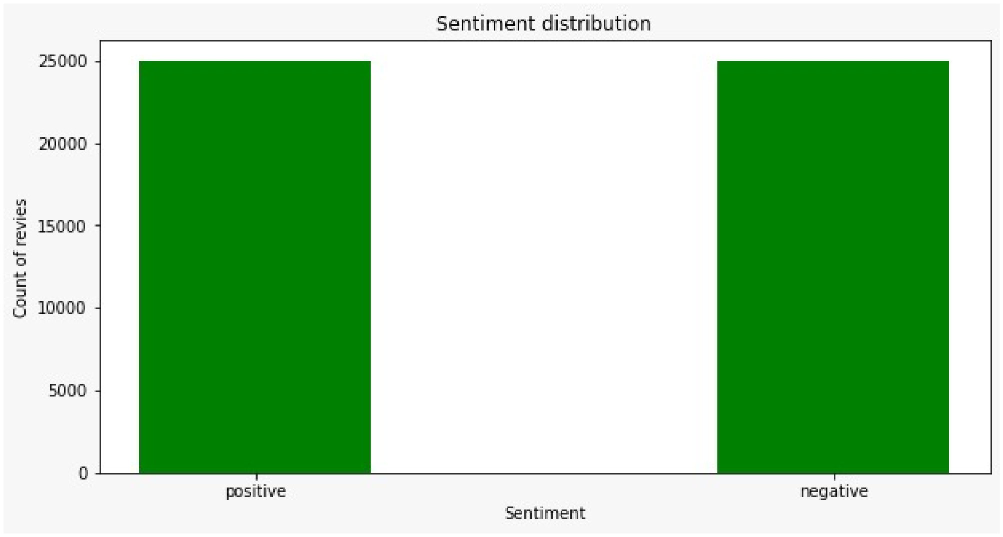
3.3. Removed the punctuations from the text column and also lemmatized its contents.

Our dataset looks like the below figure which is further used to predict the sentiment for any movie review.



	review	sentiment
1066	THE BLOB is a great horror movie, not merely b...	positive
9337	One of the most underrated movies I've seen in...	positive
6322	Fido is a story about more well mannered zombi...	positive
12835	Van Sant copies Hitchcock's masterpiece shot f...	negative
1115	This is of of Sammo's great early comedy films...	positive
...	...	...
15188	Of all the E.R.Burroughs screen adaptations th...	negative
776	Food always makes a good topic in movies, as "...	positive
16949	I am furious! It has been a while since the la...	negative

3.4. Checked the sentiment distribution within each sentiment class. We found that there is no sentiment imbalance within the reviews, as shown in the below distribution.



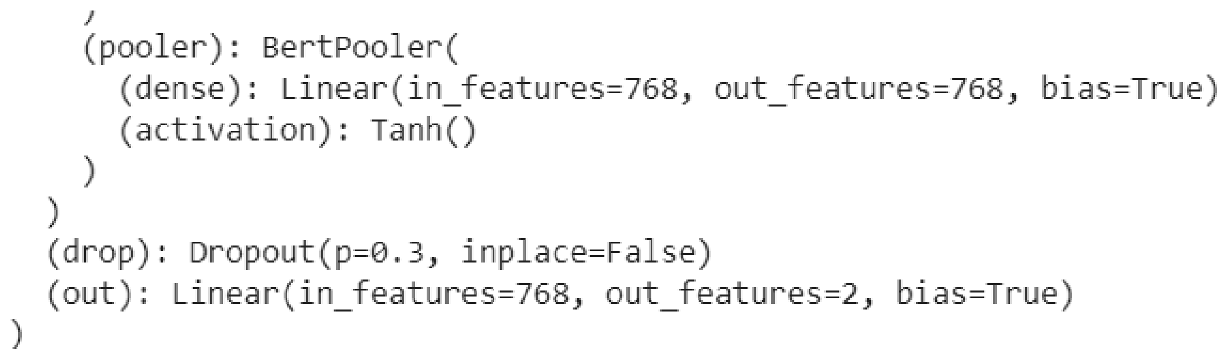
3.5. Divided our dataset into train, validation and test split in the ratio 60:20:20.
3.6. The train data was further used to train the model.

4. Training the Model:

4.1. Imported the pre-trained BERT model, which is a transformer based model and added

1.1. Dropout layer with probability 0.3

1.2. One extra linear layer with input features as 768 and output features as 2 (since we have two sentiment classes in our data, i.e. positive & negative). We have used the ***'bert-base-uncased'*** architecture of BERT for our analysis.



```
      )
      (pooler): BertPooler(
        (dense): Linear(in_features=768, out_features=768, bias=True)
        (activation): Tanh()
      )
    )
    (drop): Dropout(p=0.3, inplace=False)
    (out): Linear(in_features=768, out_features=2, bias=True)
  )
```

4.2. We have used BertTokeniser to tokenize our input train & validation data to the model.
4.3. Trained the Bert Model on train data with following hyperparameters:

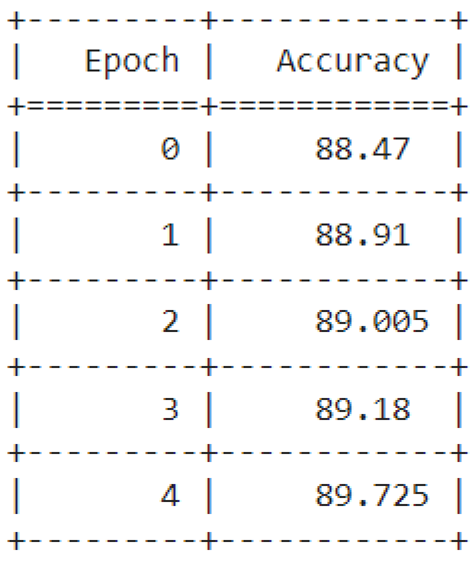
TRAIN_MAX_LEN	140
VALID_MAX_LEN	140

TRAIN_BATCH_SIZE	16
VALID_BATCH_SIZE	16
EPOCHS	5
LEARNING_RATE	3e-5
BERT_MODEL	'bert-base-uncased'
Loss Function used	CrossEntropy Loss
Optimizer used	Adam Optimizer

If review text length is too big then we will clip it to 140 characters. The Batch size used to train the model was taken as 16 and learning rate of 3e-5.

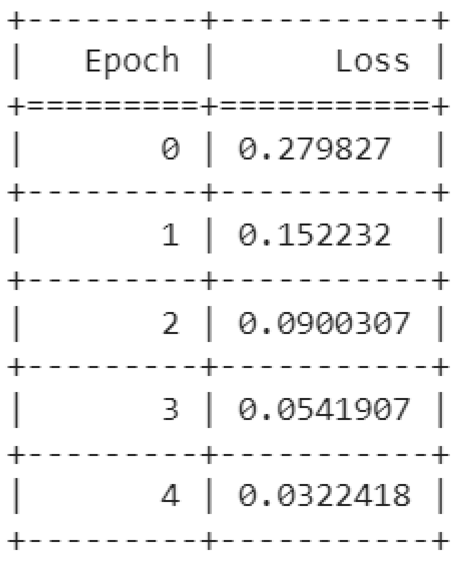
4.4. We have trained our model for 5 epochs and stored the one with best accuracy as the bin file which is used for predicting rating of the test data.

5. Observation:



```
+---------+------------+
|   Epoch |   Accuracy |
+=========+============+
|       0 |     88.47  |
+---------+------------+
|       1 |     88.91  |
+---------+------------+
|       2 |     89.005 |
+---------+------------+
|       3 |     89.18  |
+---------+------------+
|       4 |     89.725 |
+---------+------------+
```

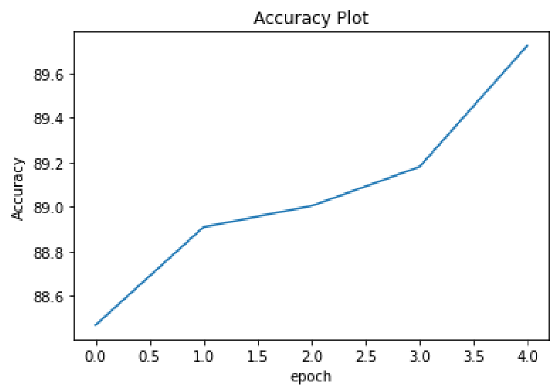
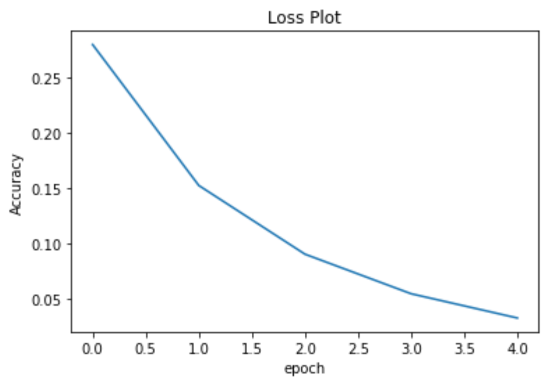
Table1: Validation Accuracy



```
+---------+-----------+
|   Epoch |      Loss |
+=========+===========+
|       0 | 0.279827  |
+---------+-----------+
|       1 | 0.152232  |
+---------+-----------+
|       2 | 0.0900307 |
+---------+-----------+
|       3 | 0.0541907 |
+---------+-----------+
|       4 | 0.0322418 |
+---------+-----------+
```

Table2: Training Loss

Plot1: Validation Accuracy and Train Loss plot for 5 epochs

Validation Accuracy Plot for 5 epochs	Train Loss plot for 5 epochs
	

Here we can see that there is a decrease in training loss per epoch and the validation accuracy was increasing with the number of epochs. We have saved the best performing model in our external memory for the inference part.