IBM Deep Learning and Reinforcement Learning Final Assignment

By

Ahmed Shahriar Sakib

Introduction

The objective of this project is to perform sentiment analysis on Hotel Review Dataset using Deep Neural Network.

Data Summary

Source: https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe

This dataset contains 515,000 customer reviews and scoring of 1493 luxury hotels across Europe. Meanwhile, the geographical location of hotels is also provided for further analysis.

Data Content

The csv file contains **17 fields**. The description of each field is as below:

Hotel_Address: Address of hotel.

Review_Date: Date when reviewer posted the corresponding review.

Average_Score: Average Score of the hotel, calculated based on the latest comment in the last year.

Hotel_Name: Name of Hotel

Reviewer_Nationality: Nationality of Reviewer

Negative_Review: Negative Review the reviewer gave to the hotel. If the reviewer does not give the negative review, then it should be: 'No Negative'

ReviewTotalNegativeWordCounts: Total number of words in the negative review.

Positive_Review: Positive Review the reviewer gave to the hotel. If the reviewer does not give the negative review, then it should be: 'No Positive'

ReviewTotalPositiveWordCounts: Total number of words in the positive review.

Reviewer_Score: Score the reviewer has given to the hotel, based on his/her experience

TotalNumberofReviewsReviewerHasGiven: Number of Reviews the reviewers has given in the past.

TotalNumberof_Reviews: Total number of valid reviews the hotel has.

Tags: Tags reviewer gave the hotel.

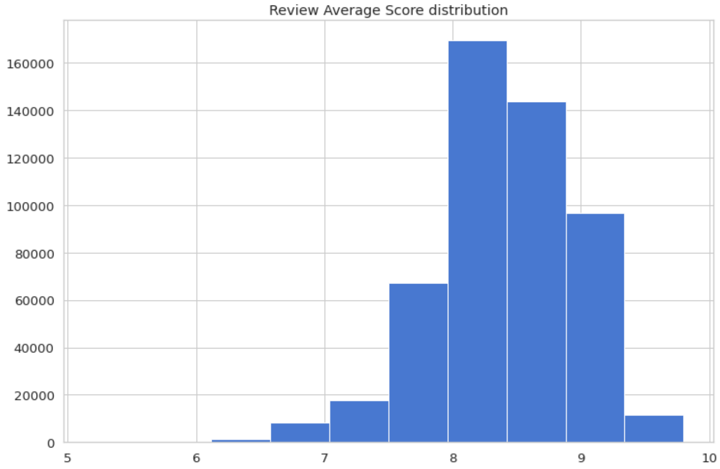
dayssincereview: Duration between the review date and scrape date.

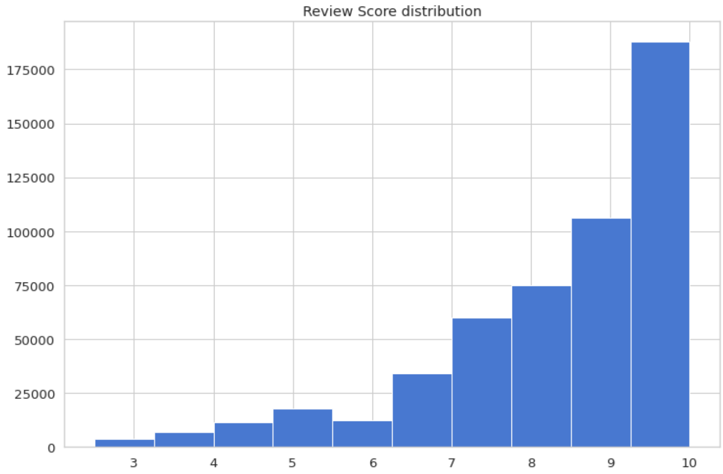
AdditionalNumberof_Scoring: There are also some guests who just made a scoring on the service rather than a review. This number indicates how many valid scores without review in there.

lat: Latitude of the hotel

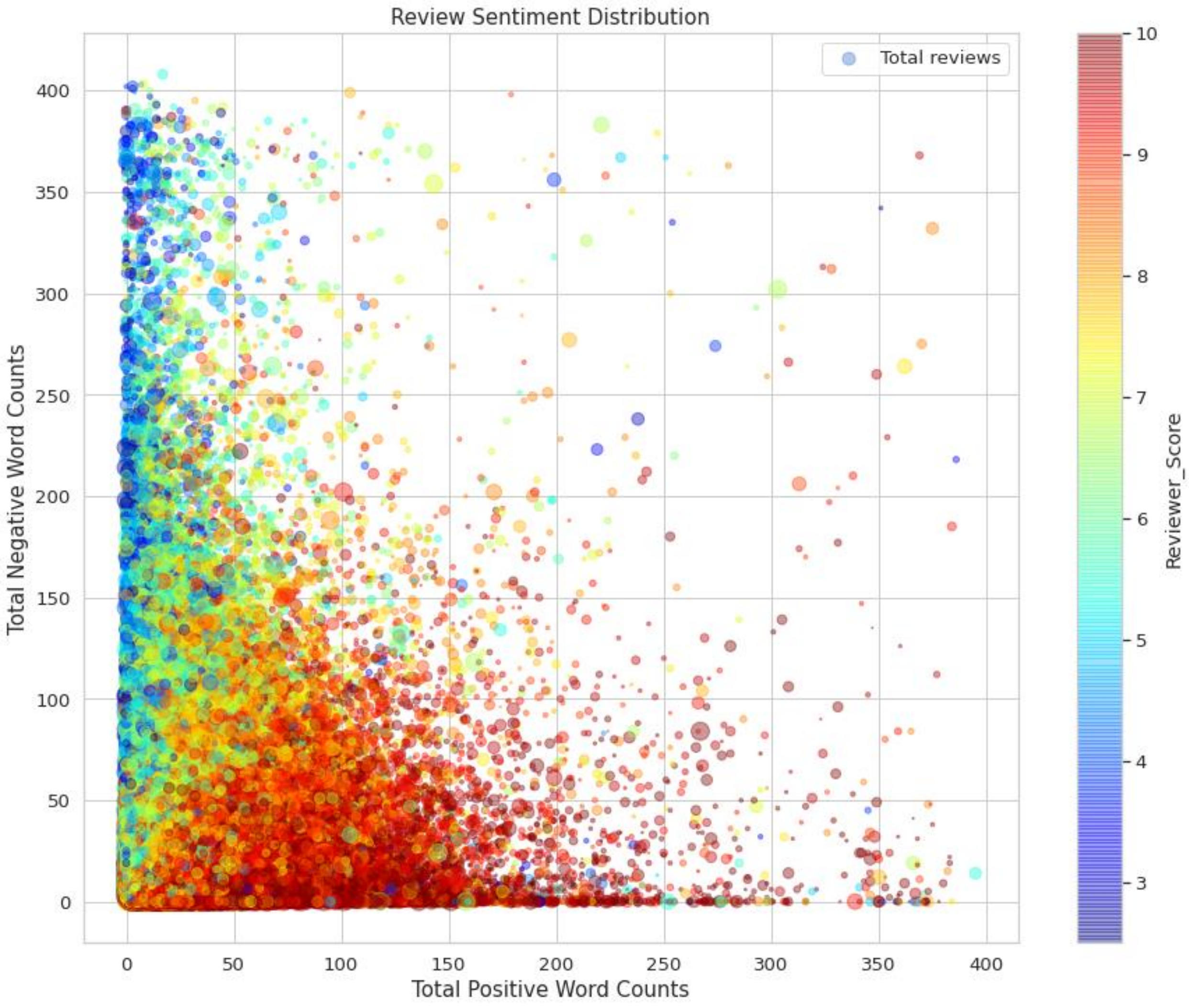
lng: longtitude of the hotel

Exploratory Data Analysis

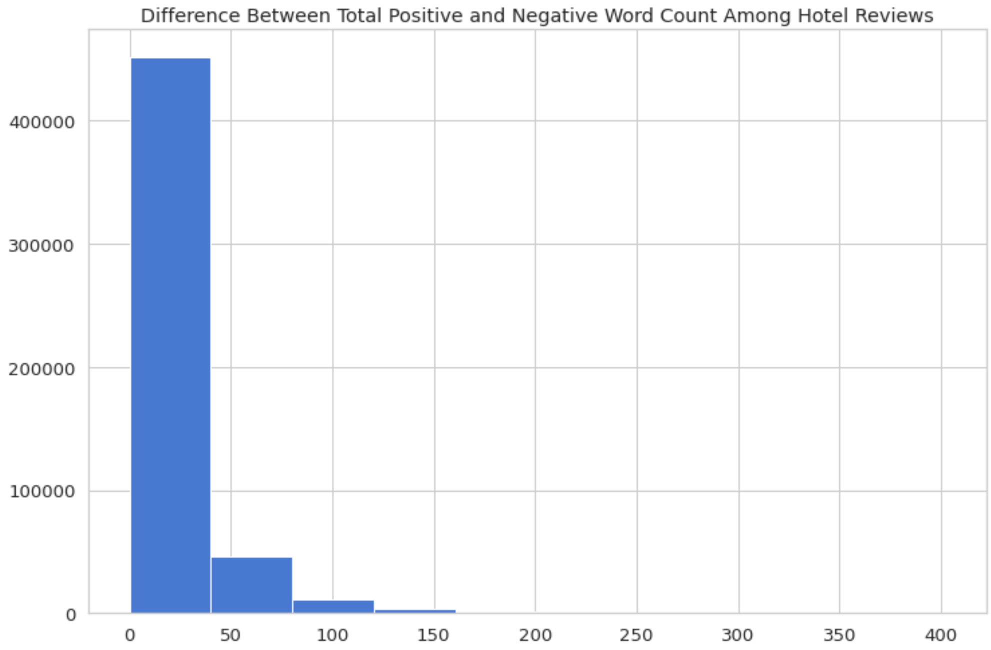




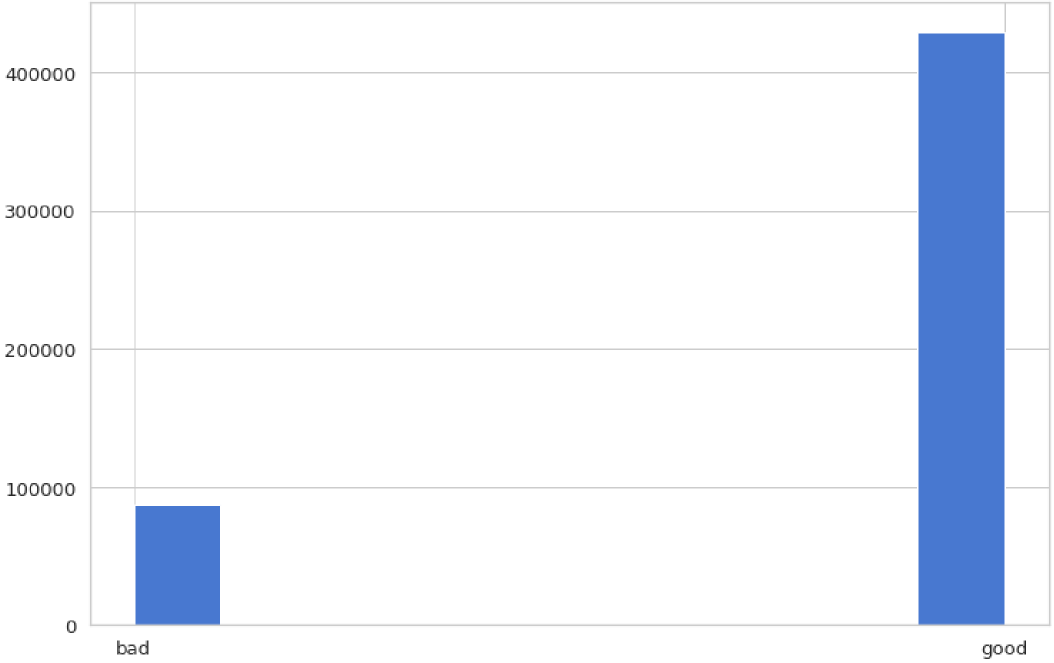
- Most of the scores fall into 9 and 10, hence the mean is around ~8 and from the distribution it is clearly visible that highest score count ranged from 9 to 10



- Number of total positive word counts in single review is higher than number of negative words counts in single review observation, but there are a greater number of people who gave positive reviews, so the density of positive word counts is overall higher in contrast with negative word count. The circle of the above graph denotes Total reviews/ 100, a metric for review density. The bigger the circle are, the higher the density of total reviews are. The reviewer score ranged from 2.5 to 10.

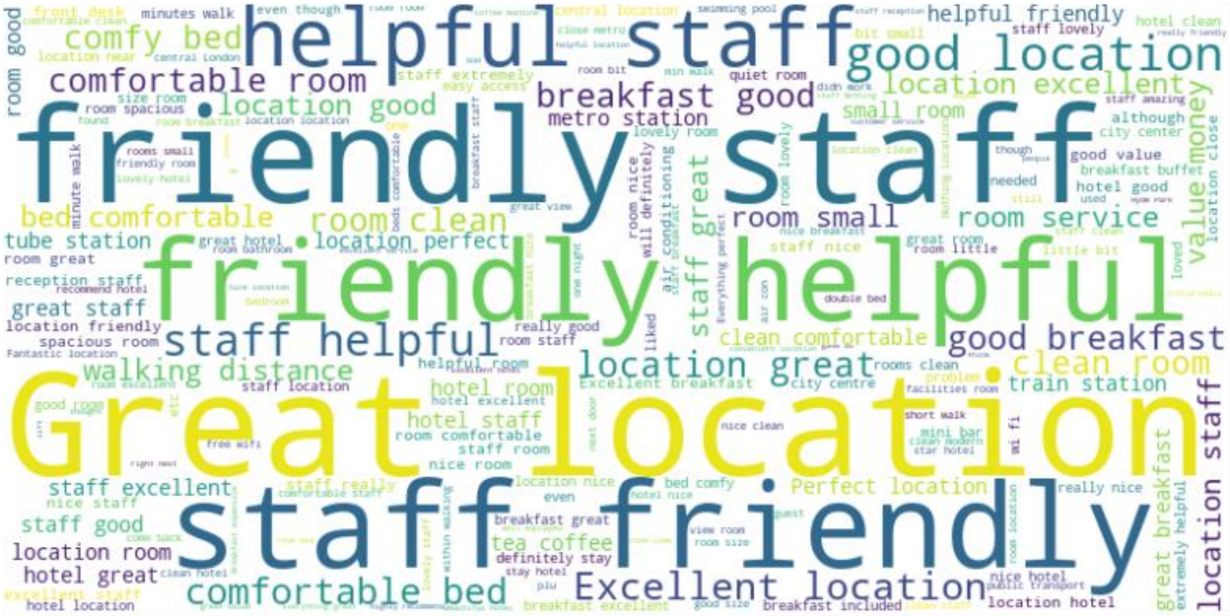


- The above graph denotes the difference between positive word count and negative word count for every reviews. The difference is less than 50 for most cases

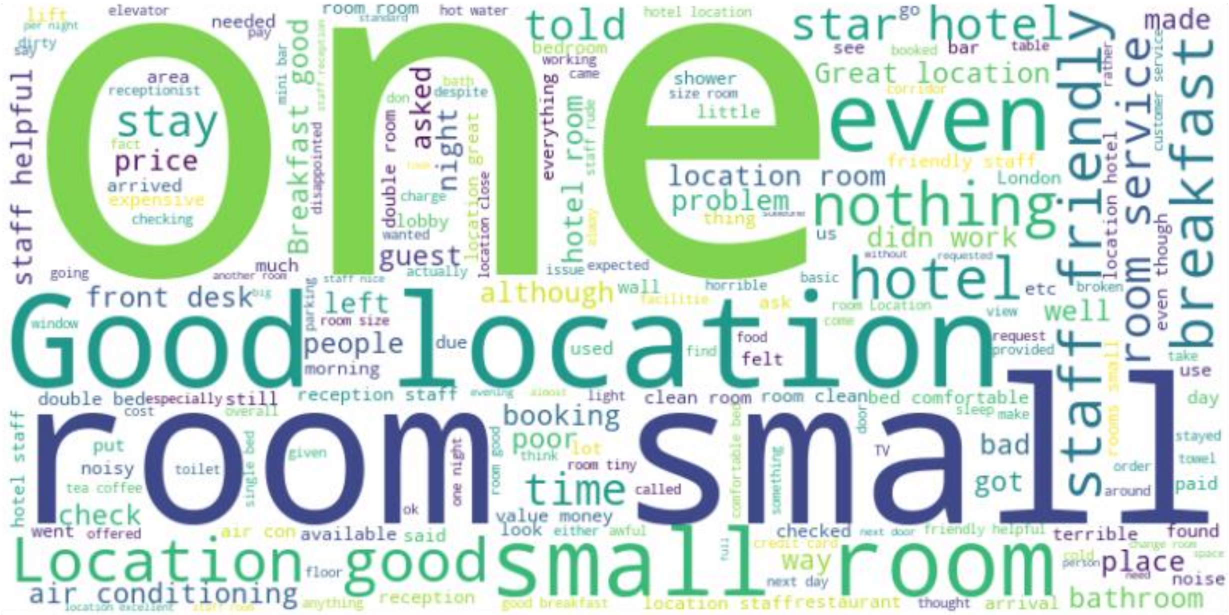


- The review data is imbalanced

- Word cloud for Positive review, phrase highlights – friendly staff, great location, comfortable bed, good breakfast etc.

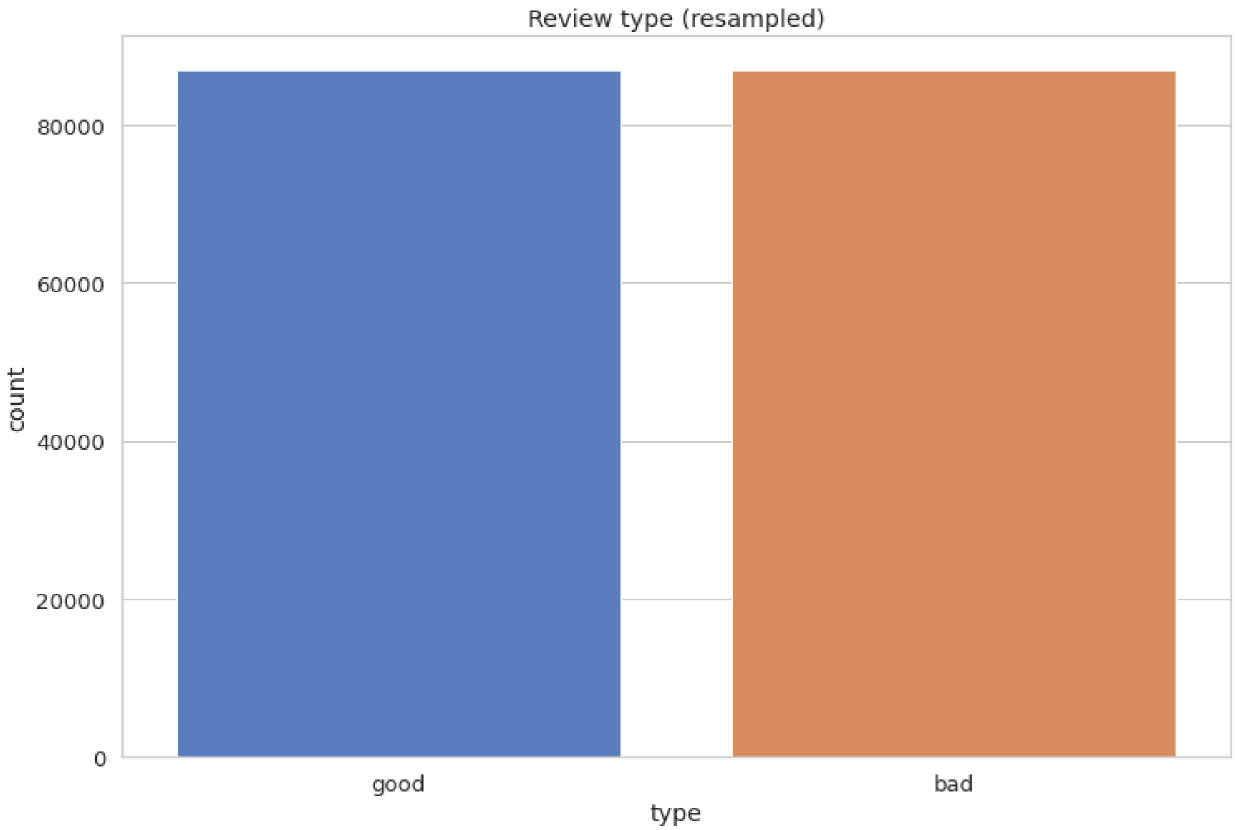


- Word cloud for negative review phrase highlights – room small, terrible, room service, small room, air conditioning, poor etc.



Data Preprocessing

1. Under sampled the positive review to achieve a balanced distribution between reviews
2. Replaced 'No Positive' and 'No Negative' with whitespace for reviews with no positive or no negative reviews respectively.
3. Merged 'positive review' and 'negative review' column to create a merged single column 'review'
4. Added new feature 'Reviewer_type' by annotating 'good' if the 'Reviewer_Score' if greater than 7 else marked as 'bad' and
5. One hot encoded the 'reviewe_type' target column
6. Resampled the final dataset containing two column, feature - "review" and "reviewe_type"" as target
7. Split the dataset in 75:25 ratio
8. Generated word embedding on "review" feature using Universal Sentence Encoder Tensorflow Hub Model



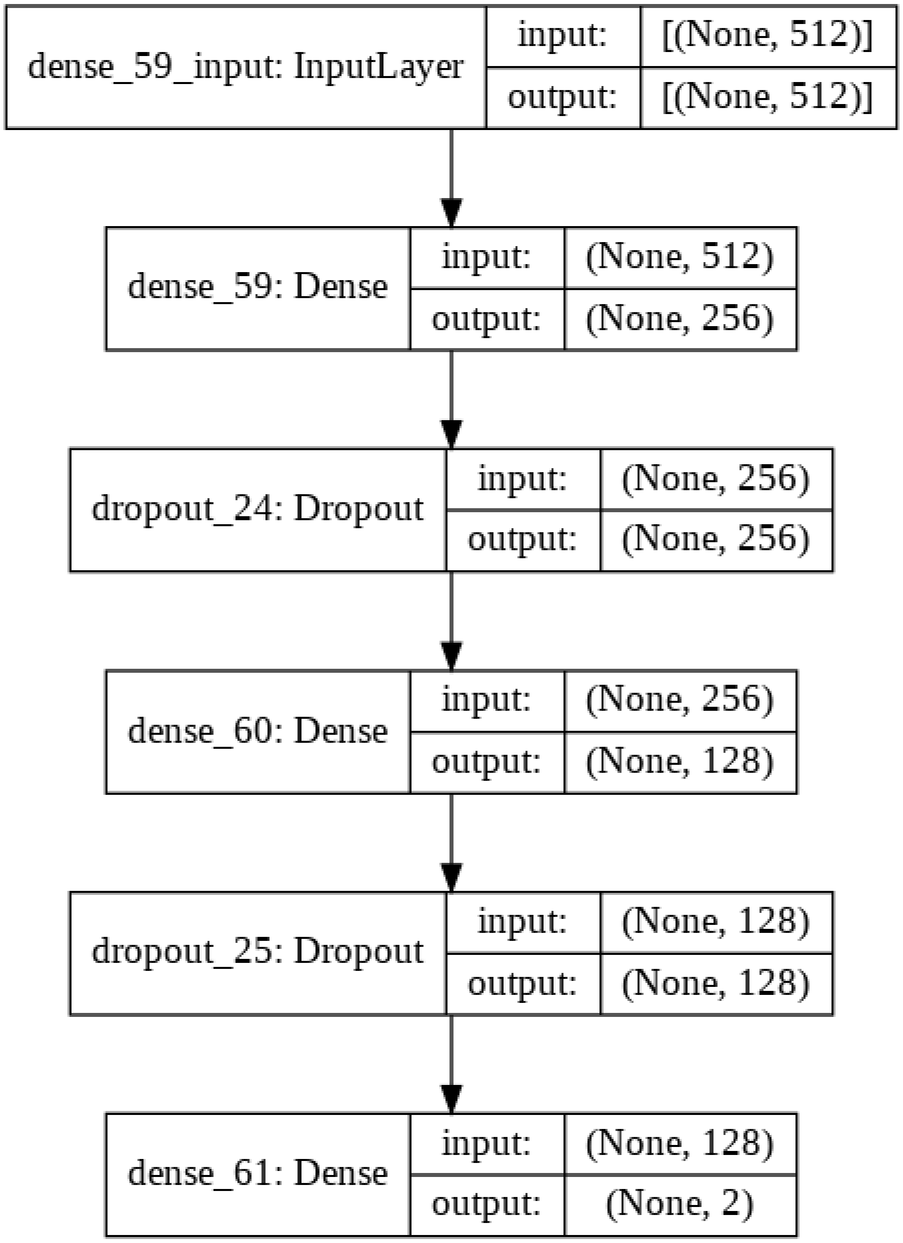
- Resampled the imbalanced dataset

Modeling

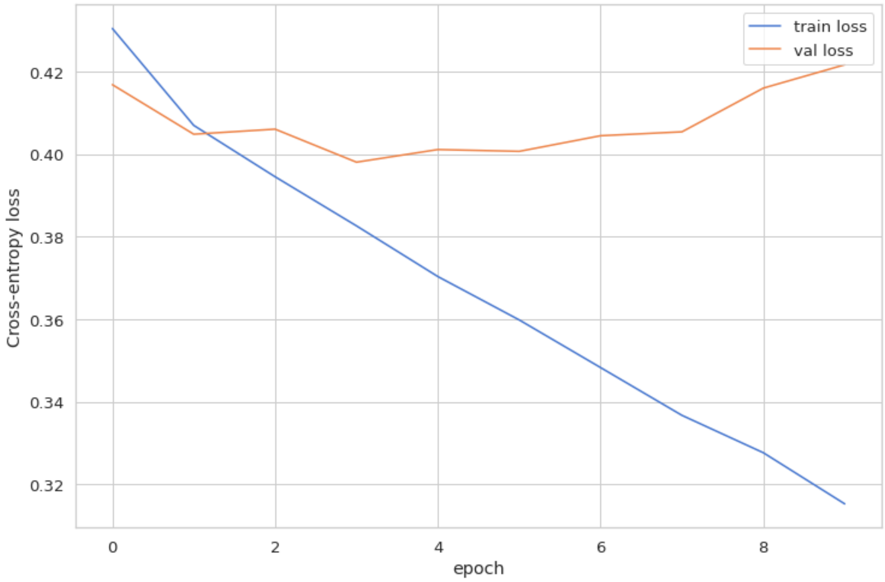
- All models ran with epochs 10, batch size 16, validation data 10%, “Adam” optimizer (learning_rate=0.001~0.0005), “categorical crossentropy” loss and shuffle data

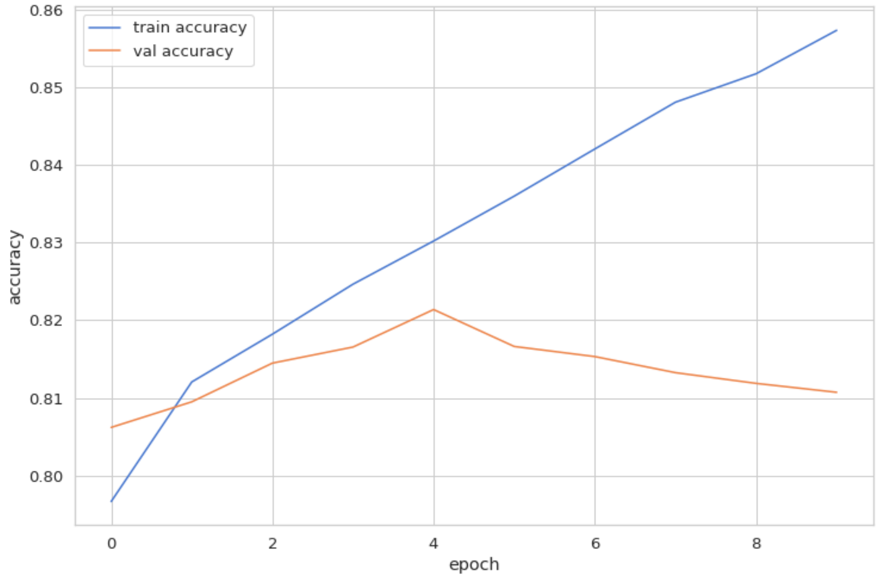
Baseline Model

Architecture



Evaluation

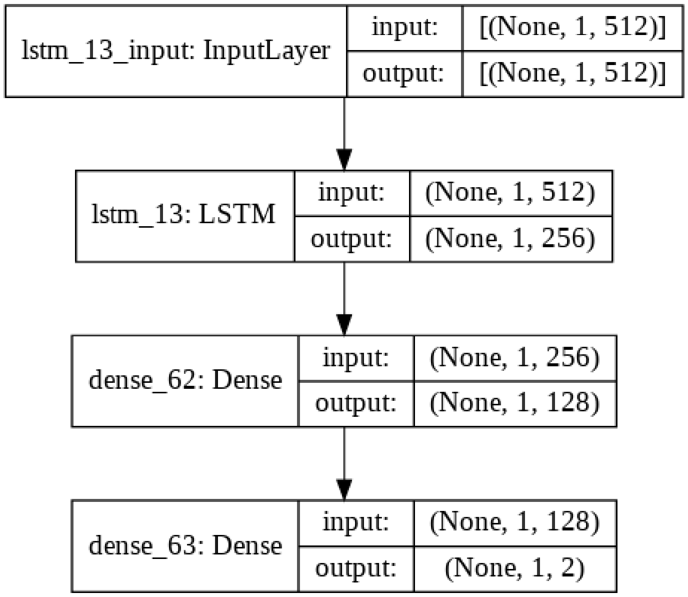




- Test Dataset Performance: loss: **0.4358** - accuracy: **0.8080**

LSTM Model-1

Architecture



lstm_13_input: InputLayer | input: [(None, 1, 512)] | output: [(None, 1, 512)]

lstm_13: LSTM | input: (None, 1, 512) | output: (None, 1, 256)

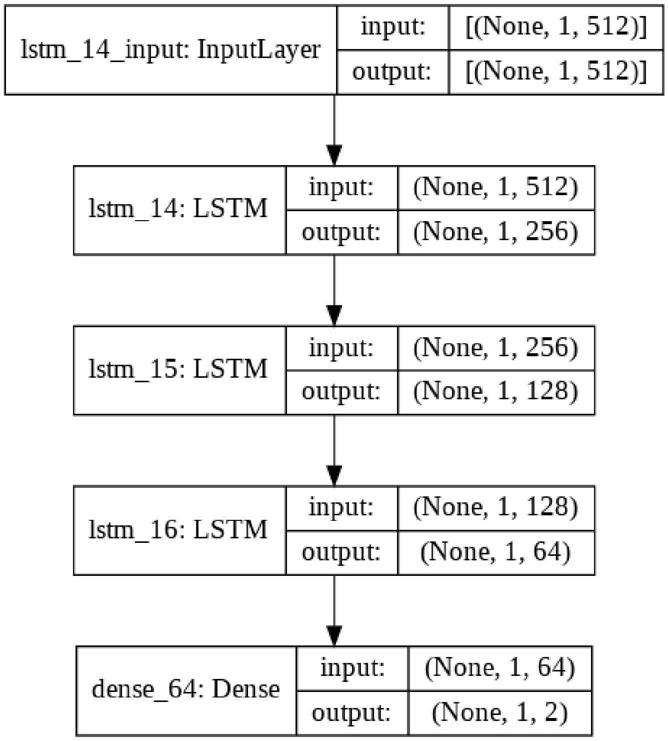
dense_62: Dense | input: (None, 1, 256) | output: (None, 1, 128)

dense_63: Dense | input: (None, 1, 128) | output: (None, 1, 2)

- Test Dataset Performance: loss: **0.4151** - accuracy: **0.8023**

LSTM Model-2

Architecture



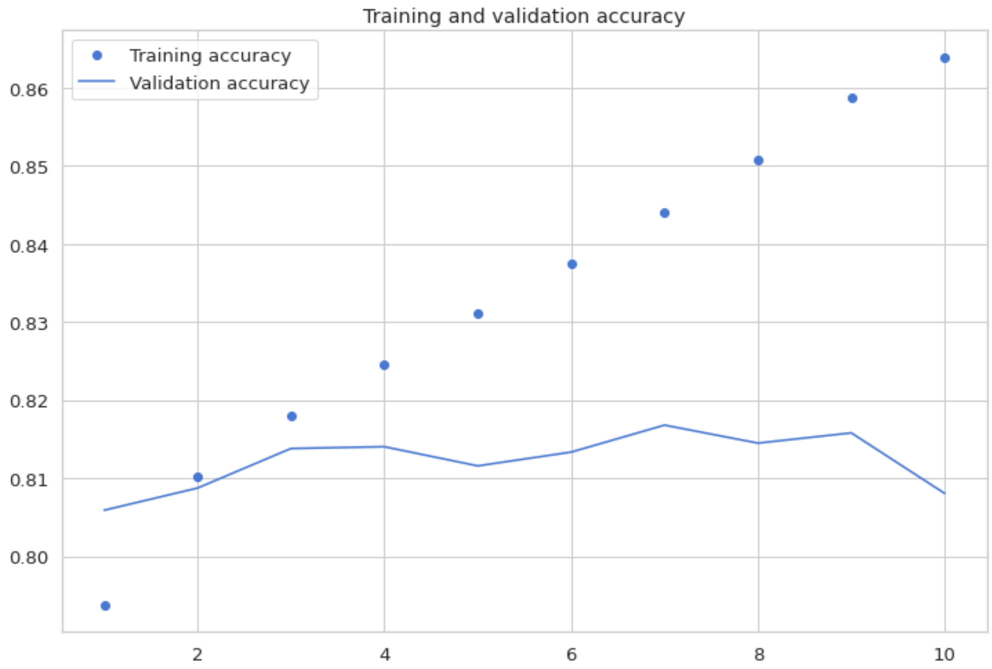
lstm_14_input: InputLayer | input: [(None, 1, 512)] | output: [(None, 1, 512)]

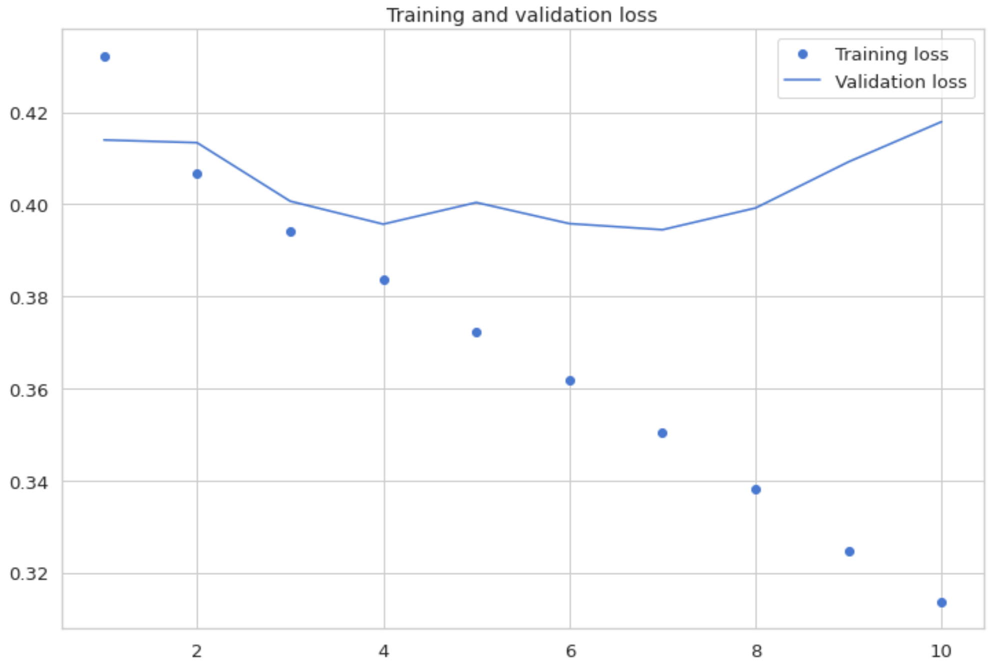
lstm_14: LSTM | input: (None, 1, 512) | output: (None, 1, 256)

lstm_15: LSTM | input: (None, 1, 256) | output: (None, 1, 128)

lstm_16: LSTM | input: (None, 1, 128) | output: (None, 1, 64)

dense_64: Dense | input: (None, 1, 64) | output: (None, 1, 2)





- Performance on Test Dataset: loss: **0.4256** - accuracy: **0.8091**

Conclusion & Future Work

- It’s clear that LSTM performs comparatively better than baseline fully connected dense model.

- In order to achieve a balanced distribution between positive and negative reviews, the positive reviews are down sampled, hence discarding lots of valuable information.
- More resampling technique such as stratifying, Synthetic oversampling such as SMOTE etc., options should be explored
- Need to try with different number of epochs, number of neurons and layers , optimizer and learning rate as well to improve performance
- Need to explore different regularization methods such as early stopping to prevent overfitting the model
- Need to apply cross-validation to make the best out of the model

Full Project Can be found here – GitHub-SA-Hotel-Reviews