Max Eduardo Lazarini Wienandts

Project: Sentiment Analysis and Topic Modeling.

YouTube video: https://youtu.be/a0wW95m3q5w

Abstract:

This project will analyze and implement several models of sentiment analysis and topic definition. The applied models were Long Short-Term Memory (LSTM), Bidirectional Encoder Representations from Transformers (BERT), DistilBERT (a distilled version of BERT), and Latent Dirichlet Allocation (LDA). The specific problem statement that this project will answer are "Is BERT and DistilBERT appropriate to sentiment analysis? Is Latent Dirichlet Allocation (LDA) appropriate for topic modeling?". To answer these questions, it was used a dataset with 1,676,974 observations from Kaggle with positive and negative enterprise reviews from Glassdoor. Moreover, it has made available 4 Jupyter notebooks "1 ETL and EDA.ipynb", "2 LSTM BERT DistilBERT.ipynb", "3 Topic modeling.ipynb", and "4 Production.ipynb". These notebooks will guide users on how to train and apply these models to a new dataset or in a specific new review. The results of this project will show that not always it is worth to use state of the art models such as BERT and DistilBERT, and that despite the simplicity of the LDA model, this algorithm is appropriate for topic modeling.

# Index

## Problem statement

Is BERT and DistilBERT appropriate to sentiment analysis? Is Latent Dirichlet Allocation (LDA) appropriate for topic modeling?

## Introduction

With a huge number of reviews that an enterprise can receive in a day, rapidly identifying its sentiment and its topic is an advantage.

BERT (Bidirectional Encoder Representations from Transformers) and DistilBERT (a distilled version of BERT) are state of the art language models with its architecture based in transformers. However, are they appropriate to solve a simple problem as classifying a review as positive or negative?

To answer this question, we are not just interested in the accuracy of the model. We also need to take into account the difficult of applying it to solve our specific problem, the resources needed, and time consumption. Therefore, we will analyze these two models plus a baseline model with just one Long Short-Term Memory (LSTM) hidden layer.

On the other hand, Latent Dirichlet Allocation is a Bayesian topic model that is easy to apply, does not demand too much computational power, and runs relatively fast. However, it is an unsupervised model and we can only choose the quantity of topics that the model returns, we cannot choose the topics themselves. Hence, we need to analyze if the topics returned will be significant and easy to interpret.

This project will use positive and negative enterprise reviews from Glassdoor (www.glassdoor.com). One advantage of using the reviews from Glassdoor is that they are already labeled by the user who wrote them. Moreover, the website asks its user to write a positive and negative review at the same time, this guarantees that the dataset will be balanced.

This report is followed by 4 Jupyter notebooks “1 ETL and EDA.ipynb”, “2 LSTM BERT DistilBERT.ipynb”, “3 Topic modeling.ipynb”, and “4 Production.ipynb”, and an youtube video. After reading and watching all the available material, you will be able to not only make a sound decision about applying or not BERT, DistilBERT and LDA models, but you will also learn how to train and apply them to a real word problem.

First, I will describe the data. I will write about the data extraction, its size, some needed preprocess, and some exploratory data analysis such as word count and most used words.

Second, I will describe the models used. I will write about some advantages and disadvantages of each model, and demonstrate how to apply them using Jupyter Notebook, including the preprocess that each model require. Here, I will also specify the hardware and software used to solve this problem.

After, I will display and compare the results of each model.

Finally, I will end this report with some conclusions.

## The dataset

The information contained in this section can be found in the Jupyter notebook "1 ETL and EDA.ipynb". The libraries needed to reproduce this study are numpy, pandas, matplotlib, seaborn, re, ntlk(it is necessary to download 'stopwords'), scikit-learn (sklearn), and wordcloud.

The data used in this project is 838,566 positive and negative reviews from companies in the United Kingdom. The reviews were posted in the website Glassdoor (www.glassdoor.com [1]), but the compiled data was collected from Kaggle (https://www.kaggle.com/datasets/davidgauthier/glassdoor-job-reviews?sort=most-comments [2]).

The original table from Kaggle has several columns, but this project will only use three columns:

- firm: The name of the company;
- pros: The positive review about the company; and
- cons: The negative review about the company.

A sample with 5% of this table with the renamed columns is available as the name df_sample.csv.

The columns "pros" and "cons" were renamed to "1" and "0" respectively. After renaming the columns, the function melt from Pandas was applied to put all the positive and negative reviews in the same column, and to create a new column "target" expressing if the sentiment of the review. The new table still have three columns, but now they are:

- firm: The name of the company;
- review: The positive or negative review about the company; and
- target: "1" if the review is positive, "1" otherwise.

The melt function defines an order in the dataset, i.e., put all the positive reviews first. To avoid any bias in this study, after the melt, the dataset must be shuffled.

This new dataset has only 15 missing values in the review column. Considering that this table has 1,677,132 observations, these 15 rows were dropped from the study. The reviews containing only punctuations were also removed. The dataset ended up with 1,676,974 observations. In which 838,553 (50%) are positive reviews, and 838,421 (50%) are negative.

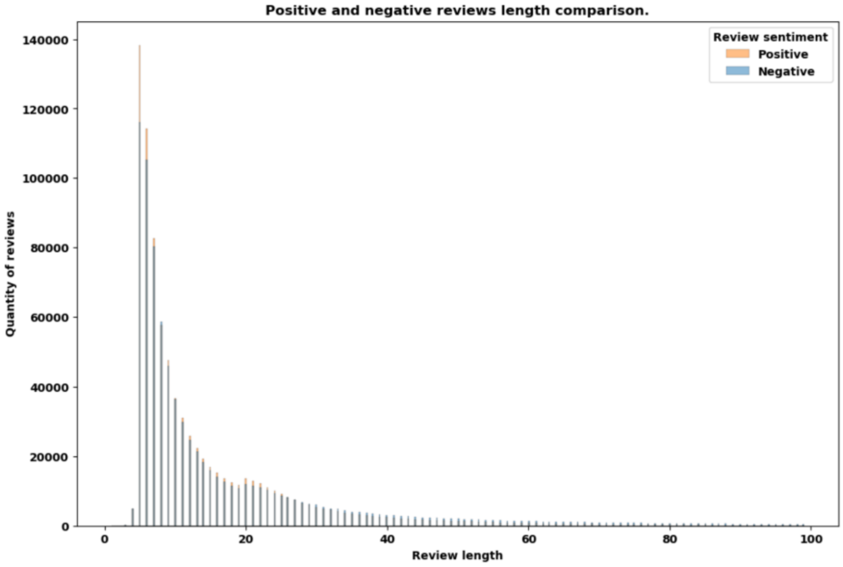
Before starting any exploratory data analysis, the names of the companies were encoded with a simple label encoder algorithm. This column will not be used in any model, but I decided to encode the name of the companies to maintain their privacy.

Some of the reviews had the special characters "\r" and "\n". Probably, this was an error when extracting and saving the dataset in Kaggle. To solve this, these characters were substituted by one blank space.

The biggest review has 3172 words, and the smallest one has only one word. Below is a table with some basic statistics about the length of the reviews:

| | | | |
|---|---|---|---|
| Mean | 18.58 | Standard Deviation | 32.82 |
| Minimum | 1 | Maximum | 31720 |
| 25th Percentile | 6 | 50th Percentile | 9 |
| 75th Percentile | 20 | 95th Percentile | 57 |

Below is the plot of the length distribution:

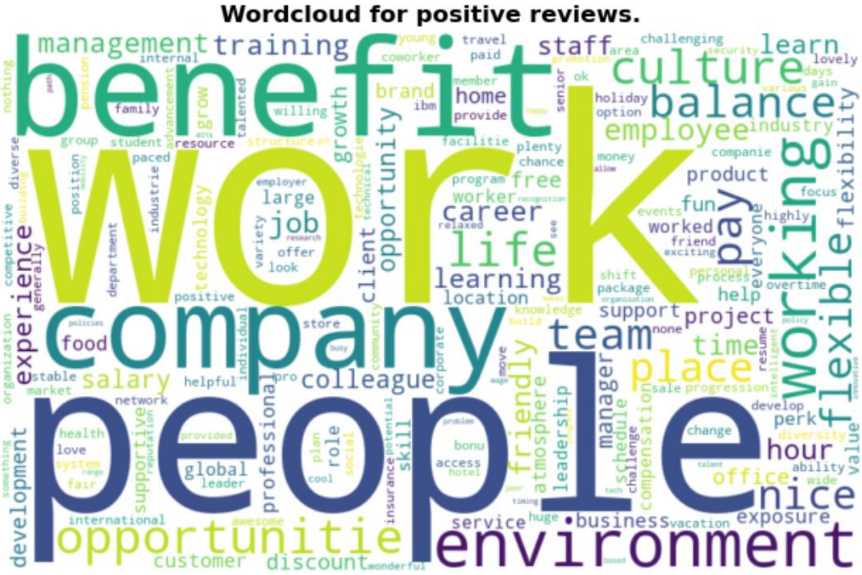


The table and plot show that choosing a maximum length of 60 for each review is already enough to contain the full review for more than 95% of the observations.

One last interesting study to do with this dataset before modeling is verifying the most used words with a wordcloud plot. Before plotting, it is necessary some preprocessing:

- Transform all words in lower case;
- Remove punctuations
- Remove stop words such as “a”, “it”, etc. These stop words were based in the English vocabulary of the library nltk; and
- Remove some words that are not relevant to our study in specific. A list with all the extra words removed can be found in the Jupyter notebook “1 ETL and EDA.ipynb”

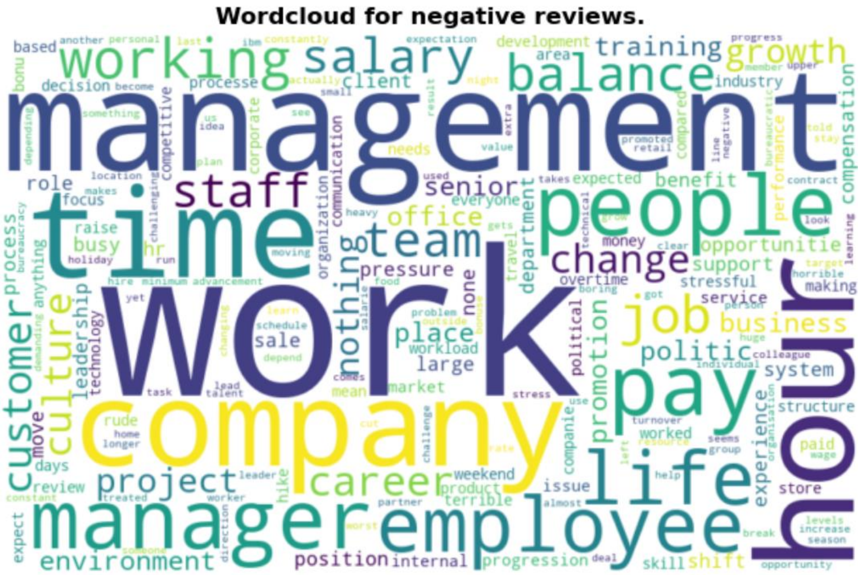
Below is the wordcloud for all the positive reviews:



The most used words were work, benefit, and people. This mean that the compensation and benefits, and the co-workers are the most important topics when writing a positive review.

It is possible to notice that the word “work” also appears as “working”. This is not a problem for the models LSTM, BERT, and DistilBERT, but it is a problem for the model LDA. To solve this, it was used the function PorterStemmer from nltk.

Below is the wordcloud for all the negative reviews:



For the negative reviews, the word management gains more relevance. This shows that bad management is a common problem for most of the companies.

After the modifications in the dataset done in this section, it was created the dataset “df_sentiment_analysis_topic_modeling.csv”.

## Models

The information contained in this section can be found in the Jupyter notebook “2 LSTM BERT DistilBERT.ipynb” and “3 Topic modeling.ipynb”. The libraries needed to reproduce this study are time, json, pprint, numpy, pandas, matplotlib, seaborn, scikit-learn (sklearn), nltk (it is necessary to download ‘stopwords’), gensim, pyLDAvis, tensorflow, and transformers.

The machine used to run the models LSTM, BERT, and DistilBERT was a personal computer running Ubuntu 22.04.2 LTS with:

- Processor: Intel® Core™ i7-11700F @ 2.50GHZ;
- RAM: 16.0 GB; and
- Video: NVIDIA GeForce RTX 2060.

The machine used to run the LDA model was a notebook running Windows 11 with:

- Processor: Intel® Core™ i7-9750H CPU @ 2.60GHZ 2.59GHz;
- RAM: 16.0 GB; and
- Video: NVIDIA GeForce RTX 2060.

### Long Short-Term Memory (LSTM)

The LSTM model have an important property that avoids vanishing and exploding gradients. Andrew Glassner, in the book “Deep Learning a Visual Approach” [3], describes this model as having the characteristics of changing its internal state frequently (short-term memory), at the same time that allows to retain some information in the state for a long time. Glassner call this as a “selectively persistent short-term memory”.

The LSTM hidden layer will be used to build a baseline model to compare with the BERT and DistilBERT models.

This baseline model has an embedding layer with the vocabulary size of 101,615, and an output size of 64. Following, a LSTM layer with 32 units. The output layer is a dense layer with one unit and sigmoid as activation function. The loss function is the binary cross entropy, and the optimizer is Adam. In total, this model has 6,515,809 trainable parameters.

However, before training this model, it is necessary to do some preprocessing with the data. The following steps were applied:

- Remove of punctuation;
- Create a vocabulary, adding values for padding, and unknown words;
- Encode the reviews transforming each word in a number defined in the previous step;
- Pad and truncate the reviews so all the observations have the same length.

Considering that more than 95% of the reviews have less than 60 words, it was chosen 60 as the maximum length of each review. This mean that any reviews with less than 60 words were completed with 0 before the text (padding), and any review with more than 60 words were truncated at 60 words.

The advantage of this model is that it is simple to implement and fast to train.

The disadvantage is that it may be too simple and it does not capture all the relationships among the words in a message. This model only considers that the current word is influenced solely by preceding words. Hence, this model may have poor performance.

## Bidirectional Encoder Representations from Transformers (BERT)

The BERT model uses transformers blocks and is a general-purpose language model. This model is composed by a word and a positional embedder, followed by multiple transformer encoder blocks. Differently from the LSTM model that is unidirectional and consider that one word is only impacted by the previously words, BERT consider the impact of every word on every other word.

This model is extremely complex with more than 109,482,240 parameters. Luckly, HuggingFace has some pretrained versions of this model that can be used with transfer learning. The BERT version used in this study was “bert-base-uncased” (https://huggingface.co/bert-base-uncased [4]).

The only preprocess needed is to use the right tokenizer available in the transformer library. This tokenizer already transforms all text in lower case, separates punctuations, encode the message with its own vocabulary, adds the special tokens CLS and SEP, pads and truncate the reviews (the maximum length used was 60), and returns a tensor vector.

The BERT model used was the “bert-base-uncased” from HuggingFace with the parameters fixed. Following, a dense layer with 32 units and ReLU as activation function, and a dropout layer with a drop rate of 0.2. The output layer is a dense layer with one unit and sigmoid as activation function. The loss function is the binary cross entropy, and the optimizer is Adam. In total, this model has 109,506,881 untrainable parameters and 24,641 trainable ones.

The advantage of this model, besides the one expressed in the first paragraph, is that it was trained in a large dataset about several topics and it is ready to use. Normally, the model output is changed so it can better adapt for a specific problem. This way, the number of trainable parameters is greatly reduced. What is more, it is possible to fine tune it by unfreezing all its layers, using a small learning rate, and training it for some few epochs.

The disadvantage of this model is that, even with the reduced number of trainable parameters, it takes a long time to train. Moreover, the model requires a considerable processing power, preferably a GPU with large VRAM.

## DistilBERT (a distilled version of BERT)

The website HuggingFace (https://huggingface.co/docs/transformers/model_doc/distilbert [5]) describes this model as a "small, fast, cheap, and light Transformer model trained by distilling BERT base." HuggingFace affirms that this model runs 60% faster and maintain 95% of BERT's performance.

The DistilBERT model has 66,387,521 parameters. Hugging face also has a pretrained version of this model. The DistilBERT version used in the study was "distilbert-base-uncased" (https://huggingface.co/distilbert-base-uncased [6])

As in the BERT model, the only necessary preprocess is to use the right tokenizer available in the transformer library.

The DistilBERT model used was the "distilbert-base-uncased" from HuggingFace with the parameters fixed. Following, a dense layer with 32 units and ReLU as activation function, and a dropout layer with a drop rate of 0.2. The output layer is a dense layer with one unit and sigmoid as activation function. The loss function is the binary cross entropy, and the optimizer is Adam. In total, this model has 66,378,521 untrainable parameters and 24,641 trainable ones.

The advantages of this model are the same from the BERT model, including the characteristics expressed by HuggingFace in the first paragraph.

The disadvantages also are the same from the BERT model, but in a smaller scale. Despite the reduced size, it is still preferable to train this model in a GPU.

## Latent Dirichlet Allocation (LDA)

David M. Blei, Andrew Y. Ng and Michael I. Jordan, in your article Latent Dirichlet Allocation [7], describe this model as "a generative probabilistic model for collections of discrete data such as text corpora". Differently from LSTM, BERT, and DistilBERT models, LDA works with bag-of-words. This means that it is not considered the influence of one word in another. The LDA model only consider the number of times that a word appears in a document.

Before training this model, it is necessary to do some preprocessing with the data. The following steps were applied:

- Lower case all words;
- Remove punctuations;
- Remove stop words;
- Remove frequent words that are not appropriate to be used as topic;
- Tokenize the reviews in unigrams;
- Stem or lemmatize the words (in this study it was used stemming);

- Count the number of times that each word appears in one document, or vectorize the words using TF-IDF.

LDA is an unsupervised model. This means that it is the algorithm that decides what the topics will be. Nevertheless, it is still necessary to choose the number of topics. To make this decision, it was plotted the Intertopic Distance Map and verified if the topics resulted from the model were overlapping. The final number of topics chosen in this study was four.

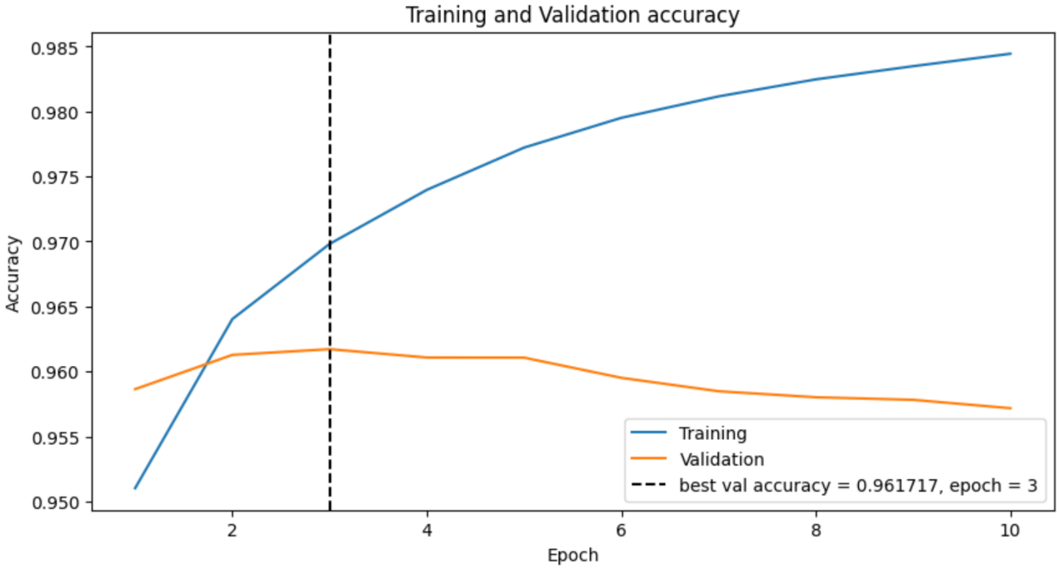
The advantages of this model are its simplicity to apply in any set documents, it is relatively fast to train, and it does not demand so much computational power.

Its disadvantages are that it does not consider the relationship among words, it demands some preprocessing before training, not all words inside a topic are relevant for the problem in specific, and may return topics that overlaps with one another.

# Results

In total, it was considered 1,676,974 positive and negative reviews. 20% of this data (335,395 observations) were randomly selected as the test set. 20% of the remain data (268,316 observations) were randomly selected as the validation set. The rest (1,073,263 observations) is the training set.

## Long Short-Term Memory (LSTM)



The baseline model has an impressive accuracy of 0.96 in the validation set. Moreover, the training process took only three epochs to calculate the optimal weights.

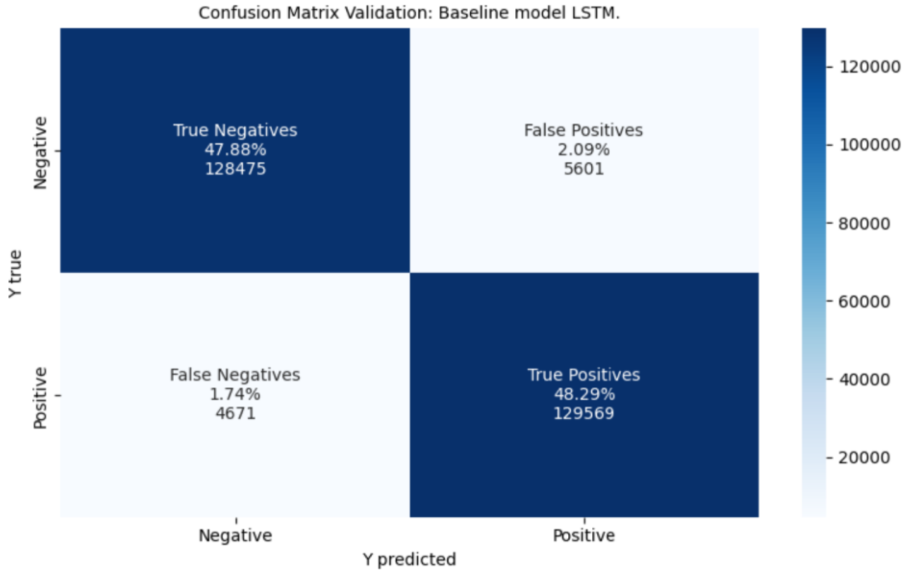
The preprocessing needed to model took less than one minute. However, the training time was 21 minutes. The predict time for the validation data set was less than one minute.

Below are the performance metrics for the training and validation dataset:

| Dataset | Accuracy | Precision | Recall | Specifity | F1 | ROC AUC |
|---|---|---|---|---|---|---|
| Training | 0.97 | 0.97 | 0.98 | 0.97 | 0.97 | 0.996 |
| Validation | 0.96 | 0.96 | 0.96 | 0.96 | 0.96 | 0.992 |

It is possible to increase the performance of this model resolving the overfitting problem with, i.e., dropout layers. Nonetheless, this model with only three epochs does not have overfitting.

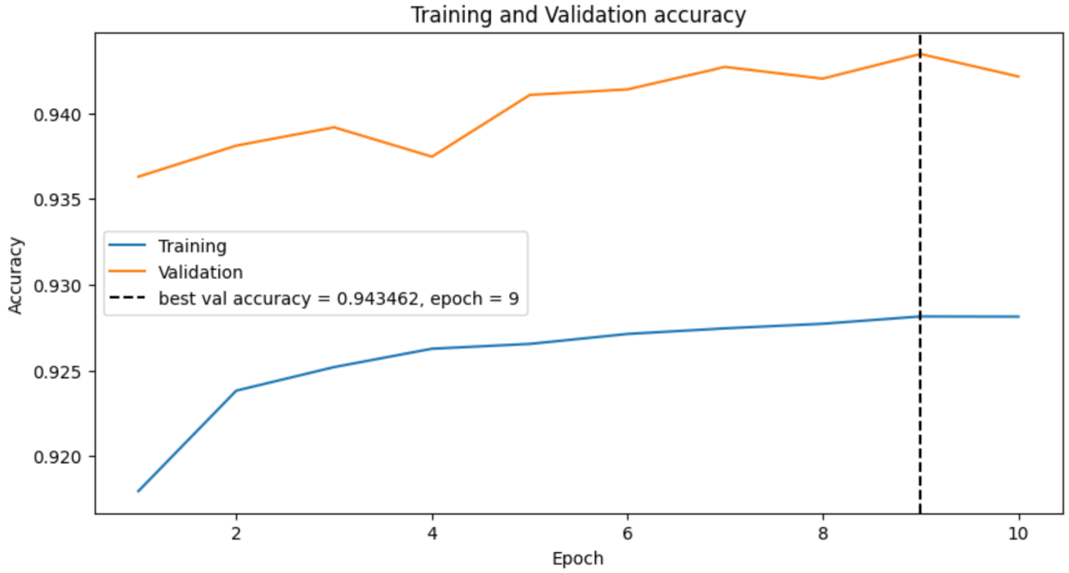
The confusion matrix of the validation set is:



It is interesting to notice that this model does not have any particular bias. The false positives and the false negatives are each close to 2%.

In general, the baseline model already has a good performance, and runs relatively fast. Moreover, it is possible to ameliorate it by adding a dropout layer.

## Bidirectional Encoder Representations from Transformers (BERT)



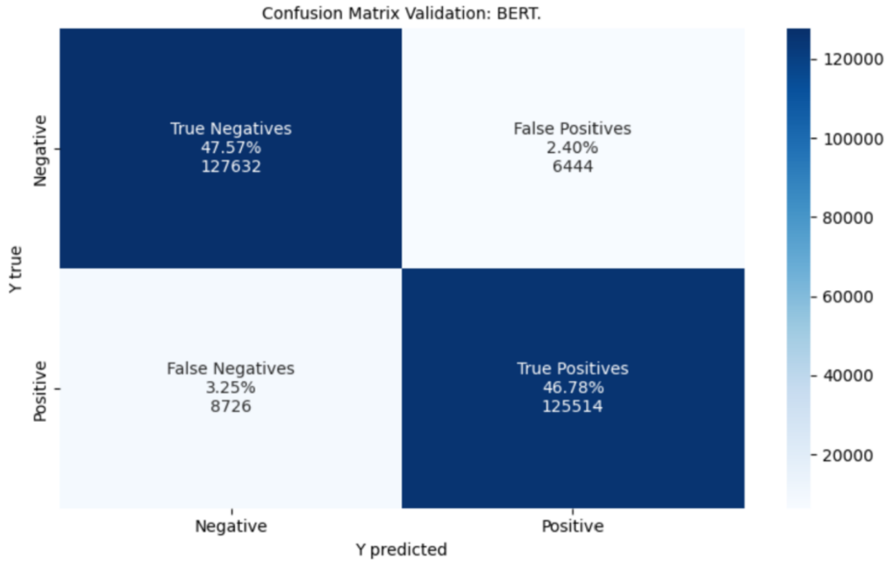
This model does not have a problem of overfitting. Nevertheless, its accuracy is lower than the baseline model, and it takes more epochs to calculate the optimal weights.

The preprocessing needed to model took almost six minutes. The training time took more than 12 hours. The predict time for the validation data set took 14 minutes.

Below are the performance metrics for the training and validation dataset:

| Dataset | Accuracy | Precision | Recall | Specifity | F1 | ROC AUC |
|---|---|---|---|---|---|---|
| Training | 0.94 | 0.95 | 0.94 | 0.95 | 0.94 | 0.986 |
| Validation | 0.94 | 0.95 | 0.93 | 0.95 | 0.94 | 0.985 |

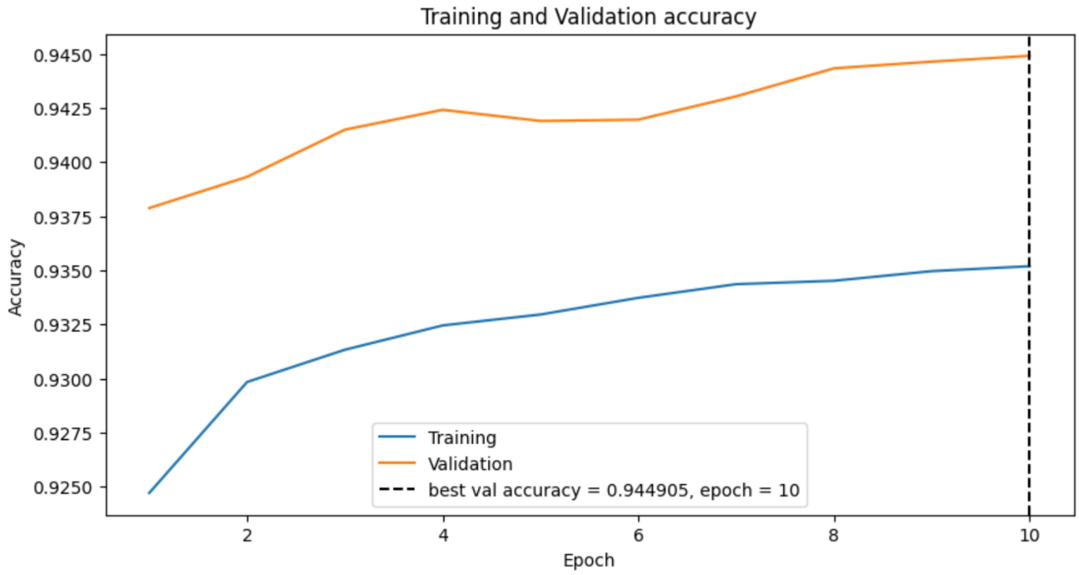
The confusion matrix of the validation set is:



This model also does not have any kind of bias. The difference between false positives and false negatives is less than 1%.

This model was worst than the baseline one and took more than 30 times to train. It is possible to increase its performance by increasing the number of epochs. Moreover, its is possible to perform fine-tuning in this model by unfreezing the parameters in the BERT layers and training it with a lower learning rate. However, this model is expensive in relation to time, and investing more in fixing the overfitting problem of the baseline model is more productive.

## DistilBERT (a distilled version of BERT)



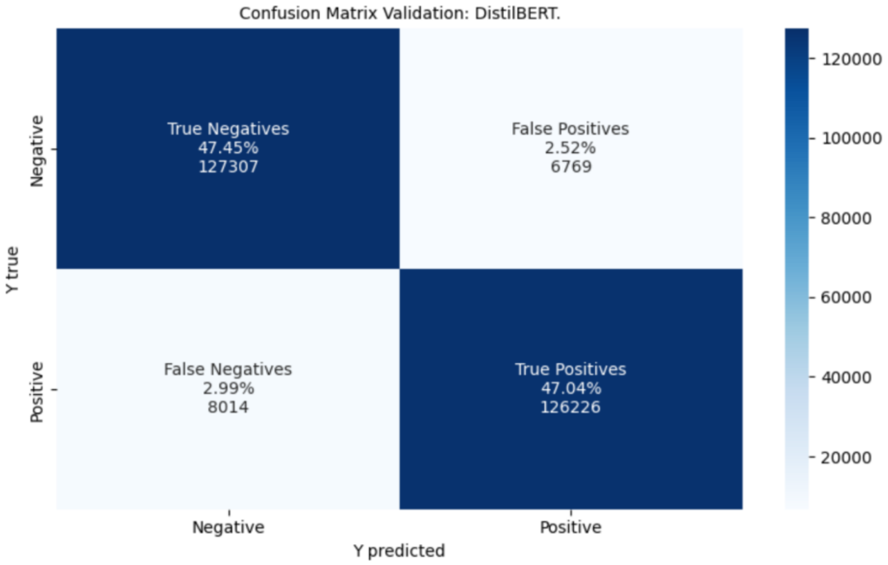
This model also does not have a problem of overfitting. Its accuracy is almost the same as the accuracy of BERT. Also, it is clear that its performance can increase with more epochs.

The preprocessing needed to model took almost six minutes, the same as BERT. On the other hand, the training time took 6 hours, half the time needed to train BERT. The predict time for the validation data set was 7 minutes, also half the prediction time for BERT.

Below are the performance metrics for the training and validation dataset:

| Dataset | Accuracy | Precision | Recall | Specifity | F1 | ROC AUC |
|---|---|---|---|---|---|---|
| Training | 0.95 | 0.95 | 0.94 | 0.95 | 0.94 | 0.986 |
| Validation | 0.94 | 0.95 | 0.94 | 0.95 | 0.94 | 0.986 |

The confusion matrix of the validation set is:



This model also does not have any kind of bias. The difference between false positives and false negatives is less than 1%.

This model has the same performance of BERT and took less time to train and predict. What is more, it is possible to increase its performance by increasing the epochs and performing fine-tuning unfreezing the parameters of DistilBERT. Unfortunately, its performance is lower than the baseline model and it takes almost 20 times more to train. This model is still expensive in relation to time, and again investing more in fixing the overfitting problem of the baseline model is more productive.

## Comparison among the three models

Below are the performance metrics for the validation dataset for all models:

| Model | Accuracy | Precision | Recall | Specifity | F1 | ROC AUC |
|---|---|---|---|---|---|---|
| LSTM | 0.96 | 0.96 | 0.96 | 0.96 | 0.96 | 0.992 |
| BERT | 0.94 | 0.95 | 0.93 | 0.95 | 0.94 | 0.985 |
| DistilBERT | 0.94 | 0.95 | 0.94 | 0.95 | 0.94 | 0.986 |

Below are the performance metrics for the test dataset for all models:

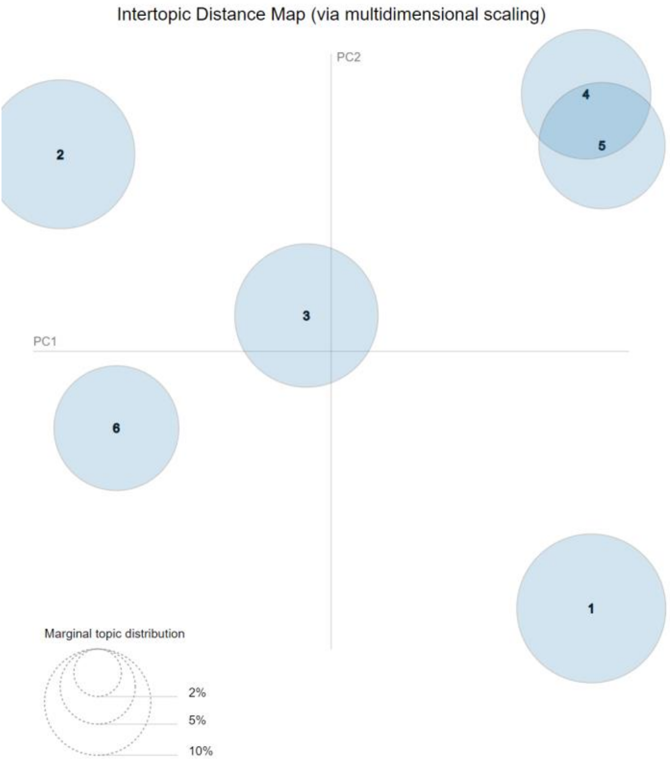
| Model | Accuracy | Precision | Recall | Specifity | F1 | ROC AUC |
|---|---|---|---|---|---|---|
| LSTM | 0.96 | 0.96 | 0.96 | 0.96 | 0.96 | 0.991 |

| | | | | | |
|---|---|---|---|---|---|
| BERT | 0.94 | 0.95 | 0.94 | 0.95 | 0.94 | 0.985 |
| DistilBERT | 0.94 | 0.95 | 0.94 | 0.95 | 0.94 | 0.986 |

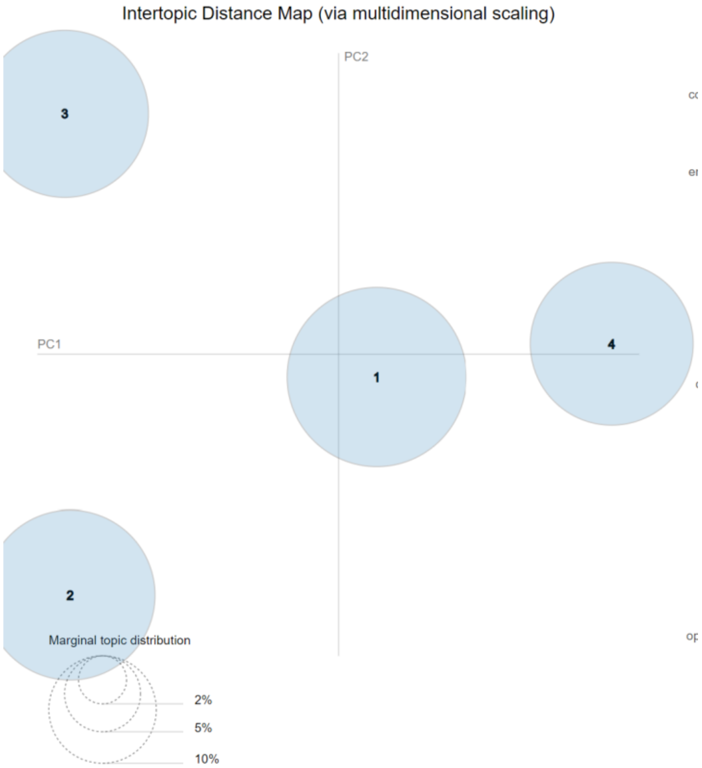
There is not a significant difference in performance among these modes. Hence, the decision to what model should be used must be based in time and resources. For this specific problem, the LSTM model is the right choice.

## Latent Dirichlet Allocation (LDA)

The Glassdoor website rate the companies in 6 topics: Culture and Values, Diversity and Inclusion, Work/Life Balance, Senior Management, Compensation and Benefits, and Career Opportunities. Therefore, it was tried to run the LDA model with 6 topics. Below is the Intertopic Distance Map.



We can see that topics 4 and 5 are overlapping. When reducing the number of topics to 5, these two topics were still overlapping. Hence, the final model has only 4 topics. Below is the Intertopic Distance Map.



The words that characterize each of these 4 topics are:

- Topic 0: hour, pay, salary, work, benefits
- Topic 1: nothing, business, custom, staff, get
- Topic 2: management, company, change, work, employee
- Topic 3: work, life, balance, people, environment

We can see that some words don't help to discriminate a topic, i.e., work, get, and custom. The ideal would be to add these words in our vector new_stopwords, in "1 ETL and EDA.ipynb", remove them, and run the algorithm again. However, we will not do this here because of time. Given these topics, we can discriminate them as:

- 0: compensation and Benefits;
- 1: Staff;
- 2: Senior Management; and
- 3: Work/Life Balance

This model was not able to discriminate all the topics considered by Glassdoor, but it was able to identify four different and relevant topics for the specific problem.

The Jypyter notebook "4 Production.ipynb" has two additional studies. In the first one, it is selected the reviews of a company, analyzed the proportion of positive and negative reviews predicted by the LSTM model, and the proportion of each topic predicted by the LDA model. The second one, consider a custom new review and predicts its sentiment and topic.

## Conclusion

At least when a large training set is available, it does not worth the time and resources needed to run BERT or DistilBERT. A recursive neural network using LSTM is faster, uses less resources, and may perform better (in this specific case, it did perform better).

The idea of using any model in the BERT family is to apply transfer learning in cases when a large training set is not available. Therefore, to decide what model to use in a specific case, it is appropriate to first do a baseline model. If the performance of this baseline model is already or close to satisfactory, the ideal is to not use transfer learning and improve the baseline model. However, if the performance of the baseline model is not satisfactory, the use of a foundation model as base to transfer learning may be the solution.

The LDA model was capable in finding four different and relevant topics for the specific problem. However, it was not capable of discriminating six different topics as suggested by Glassdoor. The LDA model is suggested to use when we have limited time and resources. Nevertheless, if a more complex result is needed for a specific problem, it is advisable to use a more complex algorithm using deep neural networks.

## References

[1] www.glassdoor.com (Accessed: 31 July 2023)

[2] https://www.kaggle.com/datasets/davidgauthier/glassdoor-job-reviews?sort=most-comments (Accessed: 31 July 2023)

[3] Glassner, Andrew S. Deep learning: a visual approach. San Francisco, CA: No Starch Press, Inc., 2021. ISBN: 9781718500723.

[4] https://huggingface.co/bert-base-uncased (Accessed: 31 July 2023)

[5] https://huggingface.co/docs/transformers/model_doc/distilbert (Accessed: 31 July 2023)

[6] https://huggingface.co/distilbert-base-uncased

[7] Blei, David M.; Ng, Andrew Y.; Jordan, Micharl I. Latent Dirichlet Allocation. Journal of Machine Learning Research 3 (2003) 993-1022.