Sentiment Analysis & Neural Networks

An insight into hotel reviews and predictions

Nawaz A.Rahman

Thesis Report 30p

Simon Celinder

Data Scientist

Göteborg | 2 May 2023

# Table of Contents

[An insight into hotel reviews and predictions 1](#_Toc134441459)

[Table of Contents 2](#_Toc134441460)

[List of Abbreviations 3](#_Toc134441461)

[Background 4](#_Toc134441462)

[Boundaries 4](#_Toc134441463)

[Methods 5](#_Toc134441464)

[Implementation 7](#_Toc134441465)

[Dataset Pre-processing 7](#_Toc134441466)

[Text Processing 9](#_Toc134441467)

[Text Analysis 10](#_Toc134441468)

# List of Abbreviations

NLTK - Natural Language Toolkit

EDA - Exploratory Data Analysis

N\_grams -

LSTM - Long Short-Term Memory

VADER - Valence Aware Dictionary and Sentiment Reasoner

# Abstract

Sentiment analysis is a technique that has found greater use in a variety of industries, including hotel management. This project's goal is to conduct a sentiment analysis of hotel reviews using NLP tools like LSTM and NLTK. The project will investigate how effectively these tools function in order to categorize hotel reviews as either positively or negatively based on the information provided.

One of the major benefits of using sentiment analysis is that it allows for a quick and efficient analysis of large amounts of data, which may be quite helpful for hoteliers. This project will contribute to a better understanding of how the hotel experience may be improved by examining what factors influence the classification of reviews as positive or negative.

The outcome of the sentiment analysis can also be used to enhance the guest experience at hotels. The most common word combinations and phrases used in negative reviews can be analyzed to help hoteliers pinpoint the most pressing and pressing problem areas. On this basis, hoteliers can enhance their services and, as a result, improve customer satisfaction and community.

In conjunction with this project, LSTM and NLTK will be improved to increase their effectiveness for sentiment analysis of hotel reviews and, as a result, make it possible for hoteliers to identify and fix issues that impair guests’ experiences quickly and easily.

# Boundaries

The project's primary goal is to analyze the sentiments of hotel reviews that have been collected from online platforms. Only text data will be used to train and test models with the aid of NLP tools like LSTM and NLTK. The project won't include implementing the results; rather, it will just provide guidance on how to use them timprove the guest experience. The project's timeframe is restricted to the class term, and work will be done on regular workdays.

# Methods

1. Dataset Pre-processing

* Perform exploratory data analysis (EDA) to gain insights about the dataset
* Clean the dataset by removing irrelevant columns, duplicates, and missing values
* Create a new column for the sentiment labels (positive or negative) based on the star ratings and/or text reviews
* Extract the positive column and word count column for further analysis

1. Text Processing

* Use the Natural Language Toolkit (NLTK) library for text processing and cleaning

1. Text Analysis

* Check the review weights by counting reviews by rating
* Use word count to see how the review rating is affected
* Use word frequency analysis to identify frequently occurring words and n-grams (unigrams, bigrams and trigrams)

1. Model Preparation

* Split the pre-processed dataset into training and testing sets
* Determine the number of words in the vocabulary
  + Perform visual inspection using boxplots and displot
  + Implement the Interquartile Range (IQR) method to detect and remove outliers
* Tokenize the text data and create a word-to-index dictionary
* Perform padding to ensure that each review has the same length
* Perform one-hot encoding on the sentiment labels
* Reshape the data from 2D to 3D to prepare for LSTM modeling

1. Model Building

* Take steps to balance the dataset weights
* Save the model weights using the ModelCheckpoint function to prevent overfitting
* Build an LSTM model using Keras libraries
* Train my own embeddings
* Train the LSTM model on the training set
* Evaluate the LSTM model's performance using accuracy, precision, recall, and F1-score metrics
* Generate a confusion matrix and summarize the results

1. Model Comparison

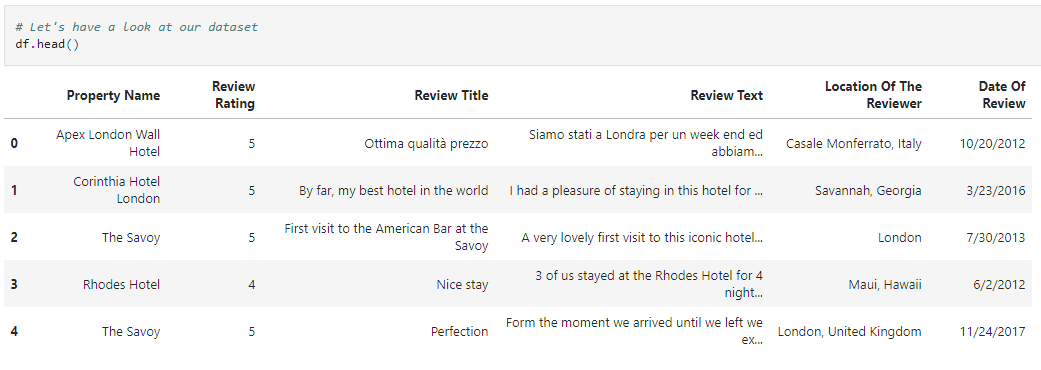
* Use the VADER model as a baseline for comparison
* Use the VADER model to make predictions on the test set
* Compare the true positive values against the LSTM and VADER predictions
* Evaluate the LSTM and VADER models based on their scores and prediction accuracy
* Compare the LSTM and VADER models and identify their strengths and weaknesses

Overall, this method involves preprocessing the dataset, cleaning the text data, preparing the data for modeling, building the LSTM model, evaluating the model's performance, and comparing it to the VADER model. This approach can provide a comprehensive analysis of sentiment analysis using LSTM and VADER models.

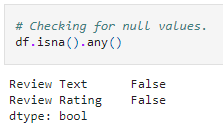
## Implementation

### Dataset Pre-processing

This is what the dataset looks like loading it in to jupyter notebook. As stated in the boundaries this project will only be using “Review Text” and “Review Rating”.

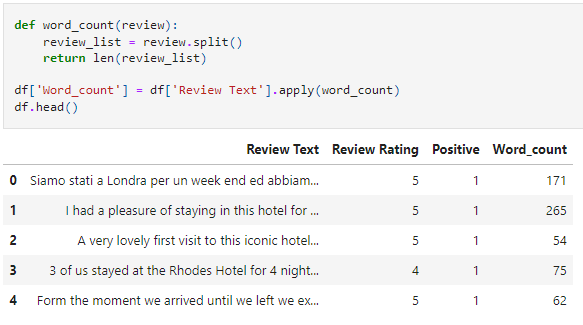


Next, the step will be to remove everything except the two features mentioned above.  
  
As seen in the picture to the right, the dataset contains 27,330 rows of data with two features.

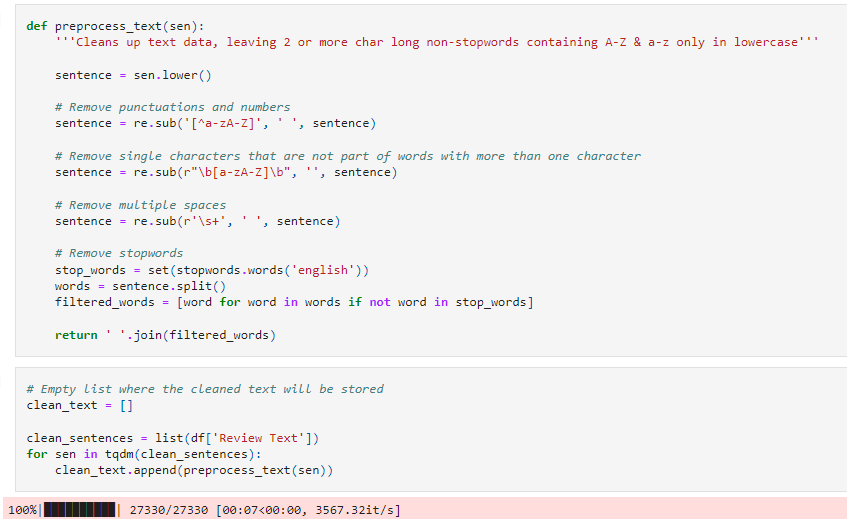
Checking for null values in the dataset because training model with null values can affect the performance. If the dataset contains null values, the model may learn to associate these null values with specific outcomes, leading to biased predictions.

A new column is created here for the sentiment labels (positive or negative) based on the star ratings and/or text reviews.

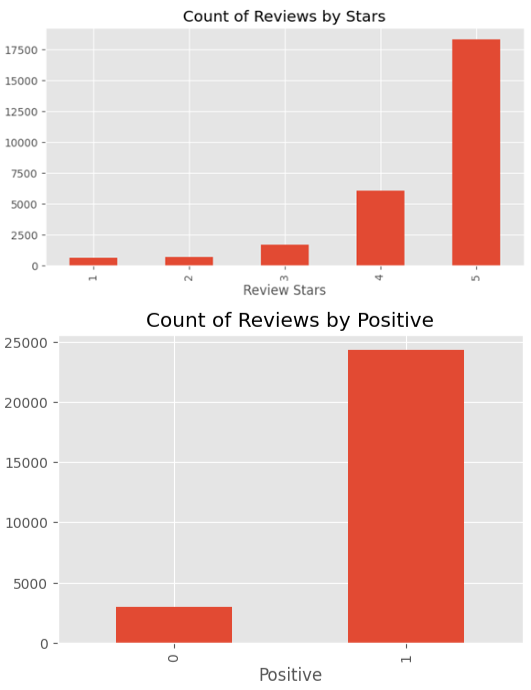
Rating 3 or below is considered negative and 4-5 is positive.

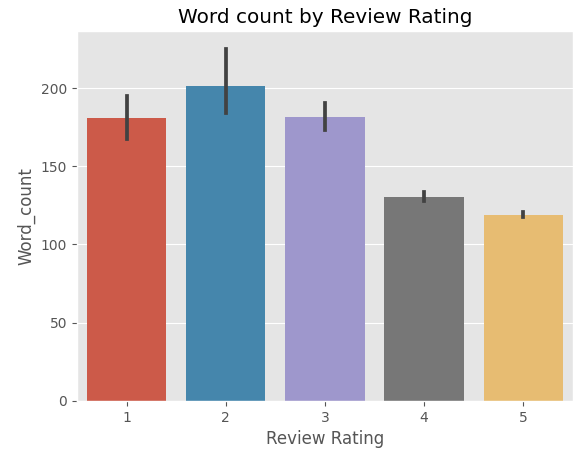
The following function splits the reviews by word, and then creates a new column called "Word\_count".

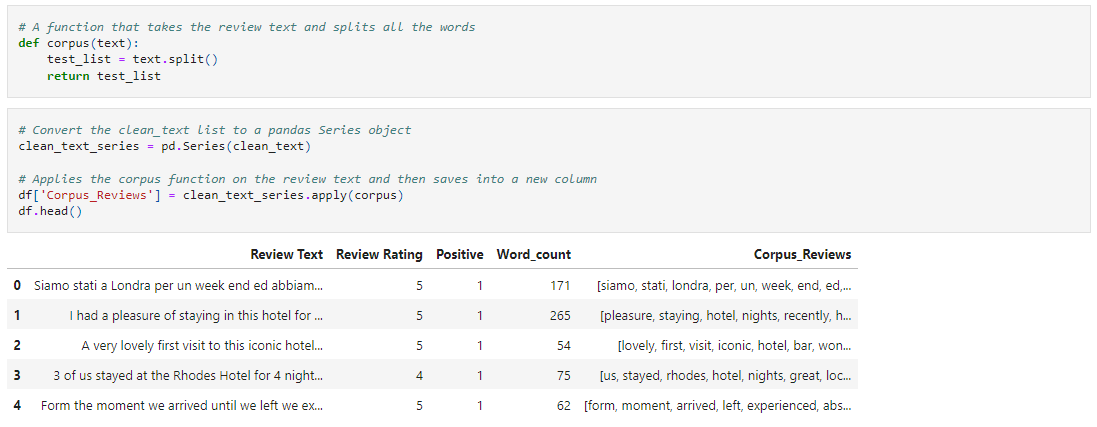
### Text Processing

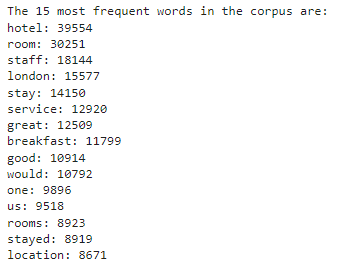
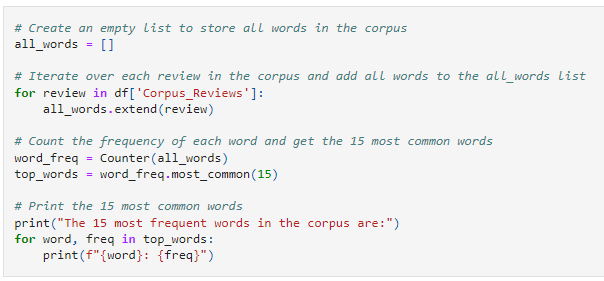
The text processing and cleaning step involves utilizing the Natural Language Toolkit (NLTK) library. The function is applied to the text column, which performs several actions, including removing punctuations and numbers, eliminating single characters that are not part of any words with more than one character, removing any instances of multiple spaces, and finally, removing all stop words (such as "a", "an", "and", "as", "at", "but", etc.) from the sentences.

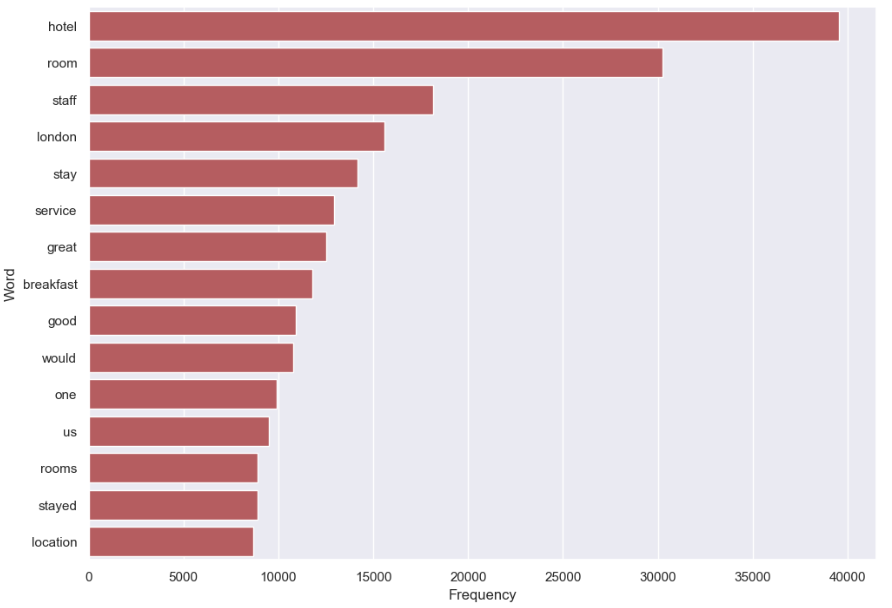
### Text Analysis

  
The two plots on the right show the review ratings based on the count of reviews made. As observed in the plots, the data is heavily skewed towards positive reviews. Therefore, when training the LSTM model, we should keep in mind that the weights are uneven and this may lead to biased accuracy towards the majority class, which in this case is positive reviews.

It's important to take steps to balance the dataset and adjust the training parameters to mitigate this issue which will be done in a later stage.  
  
This plot shows average word count of the review rating. Based on the graph, it can be inferred that the number of words decreases as the star rating increases. This is not surprising, as people tend to have more to say when they are dissatisfied which often leads to more emotion being involved.

The next step in the analysis, which involves analyzing the most frequent n-grams in the reviews. A function called corpus() is created for this purpose, which first cleans the review text using the preprocess\_text() function created earlier using the NLTK library. The cleaning process includes removing punctuations, numbers, single characters, multiple spaces, and stop words. The reason for this is to remove unnecessary elements that don't provide significant information about the sentiment and make it harder to find the words that matter. By removing these elements, it becomes easier to identify the most frequent n-grams in the reviews.

After creating the corpus, an empty list called all\_words is initialized. A for loop then iterates over each review in the corpus and adds all the words to the list. Next, the Counter() class is used to count the occurrence of each word in the list, resulting in a unigram frequency count.

Because it is a unigram, it is hard to get valuable sentiment from the review. In this case it is hard to know what sentiment hotel, room, staff has. In cases like this, it may be necessary to use n-grams with a higher value of n (e.g. bigrams or trigrams) to get a more accurate representation of the sentiment.